China 2015
WRF Tutorial
Slide Presentations

WRF Overview

Wei Wang

WRF Modeling System Overview

Wei Wang & Jimy Dudhia

Nansha, Guangdong, China

December 2015

What is WRF?

- WRF: Weather Research and Forecasting Model
  - Used for both research and operational forecasting
- It is a supported "community model", i.e. a free and shared resource with distributed development and centralized support
- Its development is led by NCAR, NOAA/ESRL and NOAA/NCEP/EMC with partnerships at AFWA, FAA, DOE/PNNL and collaborations with universities and other government agencies in the US and overseas

What is ARW?

- WRF has two dynamical cores: The Advanced Research WRF (ARW) and Nonhydrostatic Mesoscale Model (NMM)
  - Dynamical core includes mostly advection, pressure-gradients, Coriolis, buoyancy, filters, diffusion, and time-stepping
- Both are Eulerian mass dynamical cores with terrain-following vertical coordinates
- ARW support and development are centered at NCAR/MMM
- NMM development is centered at NCEP/EMC and support is provided by NCAR/DTC; now HWRF only
- This tutorial is for only the ARW core
- Both are downloadable in the same WRF tar file
- Physics, the software framework, and parts of data pre- and post-processing are shared between the dynamical cores

WRF as a Community Model

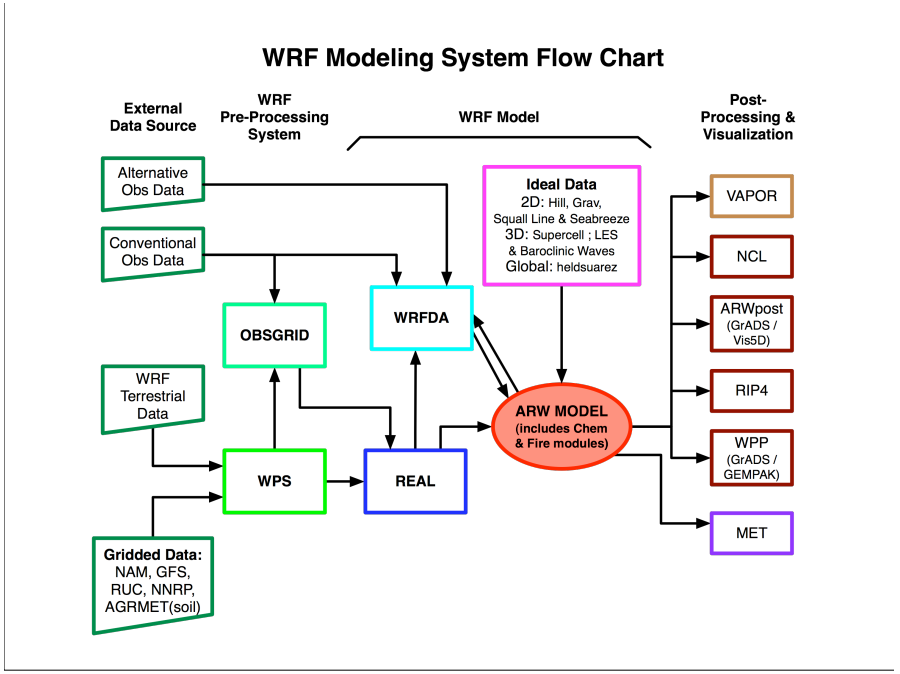
- Version 1.0 WRF was released December 2000
- Version 2.0: May 2004 (NMM added, EM nesting)
- Version 2.1: August 2005 (EM becomes ARW)
- Version 2.2: December 2006 (WPS released)
- Version 3.0: April 2008 (includes global ARW version)
- Version 3.1: April 2009
- Version 3.2: April 2010
- Version 3.3: April 2011
- Version 3.4: April 2012
- Version 3.5: April 2013
  - Version 3.5.1 September 2013 (bug-fix/minor release)
- Version 3.6: April 2014
  - Version 3.6.1 August 2014 (bug-fix/minor release, current version)
- Version 3.7: April 2015 (current version)
  - Version 3.7.1: August 2015 (bug fixes)

What can WRF be used for?

- Atmospheric physics/parameterization research
- Case-study research
- Real-time NWP and forecast system research
- Data assimilation research
- Teaching dynamics and NWP
- Regional climate and seasonal time-scale research
- Coupled-chemistry applications
- Global simulations
- Idealized simulations at many scales (e.g. convection, baroclinic waves, large eddy simulations)

Who uses WRF?

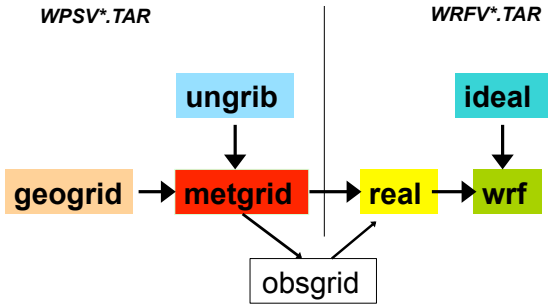
- Academic atmospheric scientists (dynamics, physics, weather, climate research)
- Forecast teams at operational centers
- Applications scientists (e.g. air quality, hydrology, utilities, wind/solar energy)



Modeling System Components

- WRF Pre-processing System
  - Real-data interpolation for NWP runs (WPS)
  - Program for adding more observations to analysis (obsgrid)
- WRF Model (ARW dynamical core)
  - Initialization programs for real and idealized data (*real.exe/ ideal.exe*)
  - Numerical integration program (*wrf.exe*)
- Graphics and verification tools including MET
- WRF Data Assimilation
- WRF-Chemistry
- WRF-Fire – wildland model for forest fires

WPS and WRF Program Flow



Real-Data Applications

- Numerical weather prediction
- Meteorological case studies
- Regional climate
- Applications: air quality, wind energy, hydrology, etc.

Real-Data Applications

- Need time-*independent* information for chosen *domain* (simulation grid area)
- Program *geogrid*:
  - Map projection information
    - 2d gridded latitude, longitude, Coriolis parameter, map-scale factors, etc.
  - Topographic information
    - 2d gridded elevation, vegetation and soil categories, etc.

Real-Data Applications

- Need time-*dependent* information
- Initial conditions (initial model time)
- Lateral boundary conditions (later times)
  - except if running WRF globally
- *Programs ungrib* and *metgrid*:
  - 3d fields of horizontal wind, temperature, geopotential height, relative humidity
  - 2d fields of surface and/or sea-level pressure, surface temperature, relative humidity, horizontal winds
  - Time-sensitive land-surface fields: snow-cover, soil temperature, soil moisture

Real-Data Applications

- Regional domains need *specified* lateral boundary conditions at later times (e.g. every 3-6 hours) through forecast period
  - 3d fields of horizontal wind, temperature, geopotential height, water vapor
  - 2d field of surface pressure
- Long simulations (> 1 week) also need lower boundary condition at later times
  - 2d fields of sea-surface temperature, sea-ice, vegetation fraction

Real-Data Applications

- Lateral Boundary Conditions (linear in time)
  - The *wrfbdy* file contains later gridded information at model points in a zone (e.g.) 5 points wide around the domain
  - The boundary fields are linearly time-interpolated from boundary times to the current model time
  - This specifies the outer values, and is used to nudge the next (e.g) 4 interior points
- Lower Boundary Condition (step-wise)
  - New SSTs and a few others are read in and overwritten at each analysis time from *wrflowinp* file

WPS Functions

- Program *geogrid*:
  - Define simulation domain area (and nests)
  - Produce terrain, landuse, soil type etc. on the simulation domain ("static" fields)
- Program *ungrib*:
  - De-grib GRIB files for meteorological data (u, v, T, q, surface pressure, soil data, snow data, sea-surface temperature, etc.)
- Program *metgrid*:
  - Interpolate meteorological data to WRF model grid (horizontally)
  - Optionally add more observations to analysis (separate *obsgrid* program)

WPS Data

- *geogrid*:
  - We provide elevation, land-use, soil type data (static fields)
  - Or user can input own static data in same easy-to-write format
- *metgrid*: Supports input of time-dependent data (dynamic fields)
  - *ungrib* can provide these from GriB files
  - Or user can input own data in same "intermediate format" (simple binary files)

WRF real and ideal functions

- Program *real*:
  - Creates initial and boundary condition files for real-data cases
  - Does vertical interpolation to model levels (when using WPS)
  - Does vertical dynamic (hydrostatic) balance
  - Does soil vertical interpolations and land-use mask checks
- Program *ideal*:
  - Programs for setting up idealized case
  - Simple physics and usually single sounding
  - Initial conditions and dynamic balance

WRF Model

- Program *wrf*:
  - Dynamical core is compile-time selectable
  - Uses initial conditions from *real* or *ideal*
  - Real-data cases use boundary conditions from REAL
  - Runs the model simulation with run-time selected namelist switches (such as physics choices, timestep, length of simulation, etc.)
  - Outputs history and restart files

ARW Dynamics

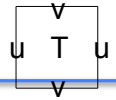
Key features:

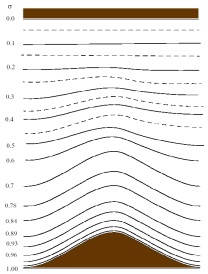
- Fully compressible, non-hydrostatic (with hydrostatic option)
- Mass-based terrain following coordinate, η

$$\eta = \frac{(\pi - \pi_t)}{\mu}, \qquad \mu = \pi_s - \pi_t$$

where π is hydrostatic pressure,
μ is column mass

- Arakawa C-grid staggering





ARW Model

Key features:

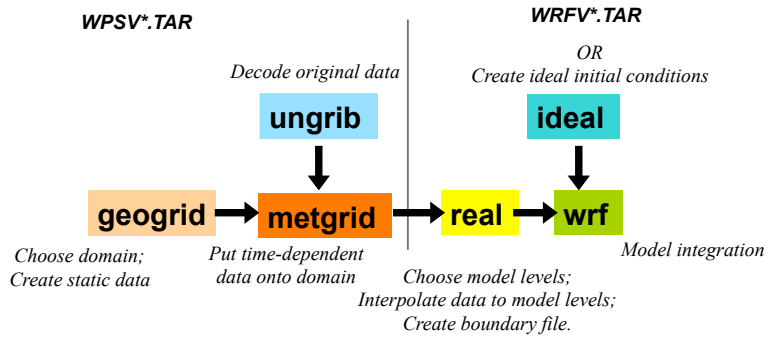
- 3rd-order Runge-Kutta time integration scheme
- High-order advection scheme
- Scalar-conserving (positive definite option)
- Complete Coriolis, curvature and mapping terms
- Two-way and one-way nesting

ARW Model

Key features:

- Choices of lateral boundary conditions suitable for real-data and idealized simulations
  - Specified, Periodic, Open, Symmetric, Nested
- Full physics options to represent atmospheric radiation, surface and boundary layer, and cloud and precipitation processes
- Grid-nudging and obs-nudging (FDDA)
- Digital Filter Initialization option

Summary of WPS and WRF Programs



Graphics and Verification Tools

- RIP4 (Read, Interpolate and Plot)
- Unified Post-Processor (UPP)
  - Conversion to GriB (for GrADS and GEMPAK)
- MET (Model Evaluation Toolkit)
- NCAR Graphics Command Language (NCL)
- ARWpost
  - Conversion program for GrADS
- VAPOR (3D visualization tool)
- IDV (3D visualization tool)

Basic Software Requirement

- Fortran 90/95 compiler
  - Code uses standard f90 (very portable)
- C compiler
  - "Registry"-based automatic Fortran code generation (for argument lists, declarations, nesting functions, I/O routines)
- Perl
  - configure/compile scripts
- netcdf library
  - for I/O (other I/O formats semi-supported)
- Public domain mpich for MPI
  - if using distributed memory option

WRFDA (Data Assimilation)

- Variational data assimilation (3D-Var and 4D-Var)
- Ensemble DA
- Hybrid variational/ensemble DA

Function

- Ingest observations to improve WRF input analysis from WPS
- May be used in cycling mode for updating WRF initial conditions after WRF run
- Also used for observation impact data studies

WRF-Chem

- Supported by NOAA/ESRL
- Includes chemistry species and processes, many chemistry options
- Also needs emissions data
- Included in WRF tar file, but requires separate compilation option

Portability

- Runs on Unix single, OpenMP and MPI platforms:
  - IBM SP AIX (xlf)
  - Linux (PGI, Intel, g95, gfortran, Pathscale compilers)
  - SGI Altix (Intel)
  - Cray XT (PGI, Pathscale)
  - Mac Darwin (xlf, PGI, Intel, g95 compilers)
  - Others (HP, Sun, SGI Origin, Compaq)

User Support

- email: wrfhelp@ucar.edu
- User Web pages:
  http://www.mmm.ucar.edu/wrf/users/
  - Latest update for the modeling system
  - WRF software download
  - Various documentation
    - Users' Guides (both cores)
    - Technical Note (ARW Description)
    - Tutorial presentations

Tutorial Schedule

- Lectures for WRF: Mon – Wed
- Practice and Q&A for WRF: Mon – Wed
  ~ 5 – 6 pm
- Lectures for WRF-Chem: Wed – Thur

Scope of Tutorial

- What's in the modeling system
  – Pre-processing programs and model
- How to run the modeling system
- Hints on choosing options
- Advanced usage
  – Adding your own input or output data
  – Making changes to the code
  – Tips on making best configuration

Scope of Tutorial

- Note: we cover a wide range of topics in much detail due to having a wide range of new users
  – New users can focus on areas of their own interest
  – Don't feel you have to learn everything in the lectures now
  – Minimum you need to learn is how to run the programs and what they do
    - Daily practice sessions (~2 hr) are a basic part

WPS: Description of General Functions

Michael Duda

The WRF Preprocessing System: Description of General Functions

Michael Duda

*NCAR is sponsored by the National Science Foundation

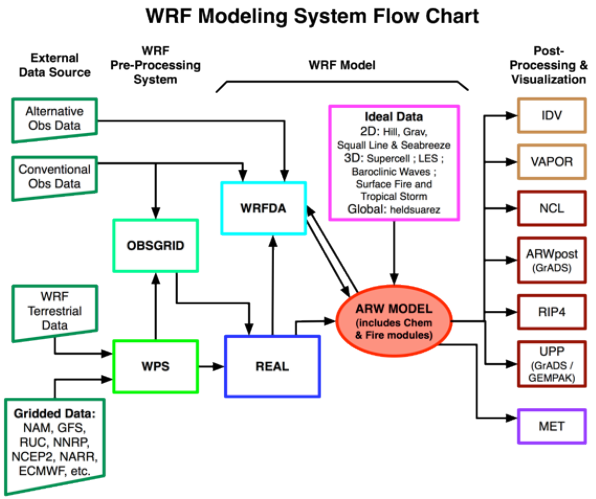
Purpose of this Lecture

In this lecture, our goals are to:

1) Understand the purpose of the WPS
2) Learn what each component of the WPS does
3) Understand why the components work as they do

- The details of *actually running* the WPS are covered this afternoon
- *Advanced features* of the WPS are described on Wednesday

WRF Modeling System Flowchart

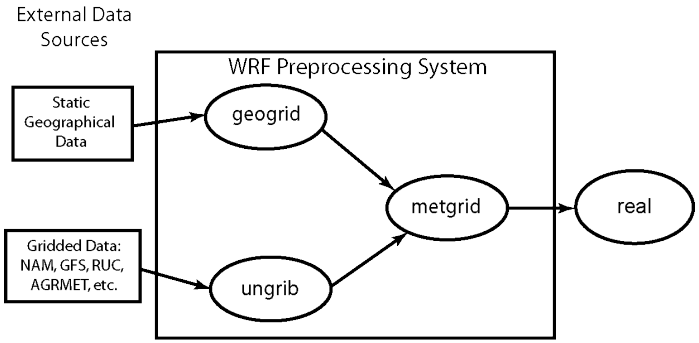


Purpose of the WPS

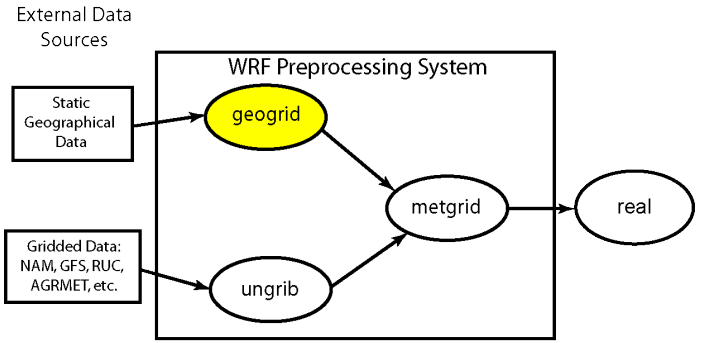
The purpose of the WPS is to prepare input to WRF for real-data simulations:

1. Defines simulation coarse domain and ARW nested domains
2. Computes latitude, longitude, map scale factors, and Coriolis parameters at every grid point
3. Interpolates time-invariant terrestrial data to simulation grids (e.g., terrain height and soil type)
4. Interpolates time-varying meteorological fields from another model onto simulation domains

WPS Program Flowchart



The *geogrid* program



geogrid: think geographical

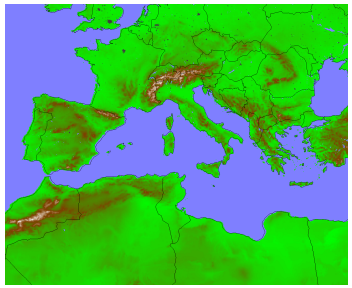
The *geogrid* program

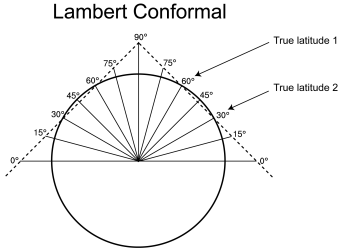
- For WRF model domains, geogrid defines:
  - Map projection (all domains must use the same projection)
  - Geographic location of domains
  - Dimensions of domains
- Geogrid provides values for static (time-invariant) fields at each model grid point
  - Compute latitude, longitude, map scale factor, 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)

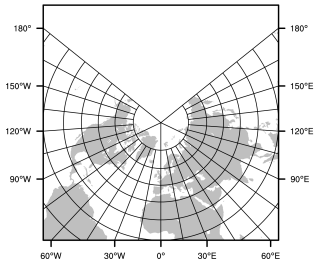
Geogrid: Defining model domains

- First, we choose a map projection to use for the domains; why?
  - The real earth is (roughly) an ellipsoid
  - But WRF computational domains are defined by rectangles in the plane
- ARW can use any of the following projections:
  1. Lambert conformal
  2. Mercator
  3. Polar stereographic
  4. Latitude-longitude (for global domain, you *must* choose this projection!)

ARW Projections: Lambert Conformal

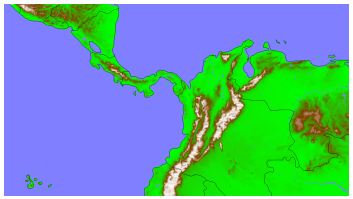


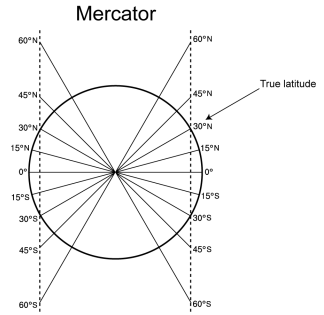


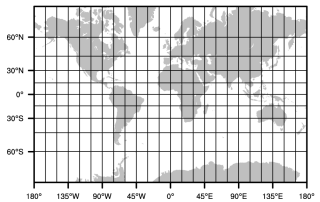


- Well-suited for mid-latitudes
- Domain cannot contain either pole
- Domain cannot be periodic in west-east direction
- Either one or two *true latitudes* may be specified
  - If two are given, the order doesn't matter

ARW Projections: Mercator

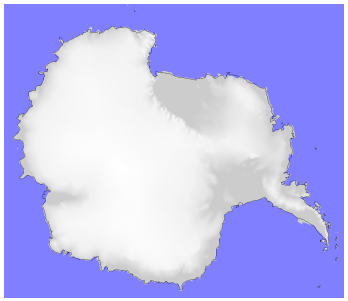


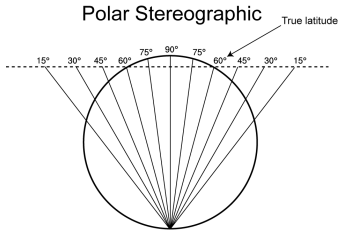


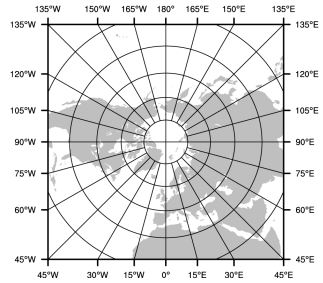


- Well-suited for low-latitudes
- May be used for "channel" domain (periodic domain in west-east direction)
- A single true latitude is specified
  - Cylinder intersects the earth's surface at +/- truelat

ARW Projections: Polar Stereographic

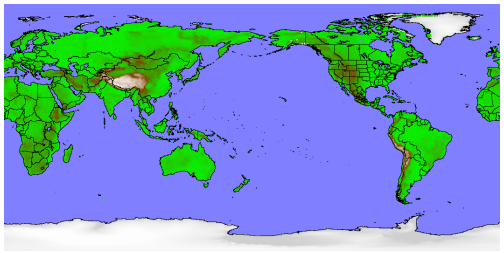


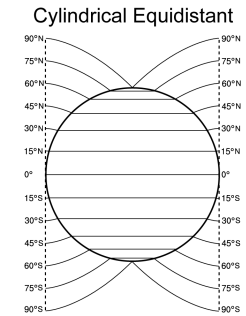


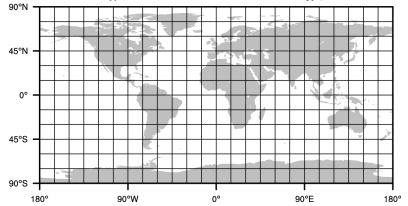


- Good for high-latitude domains, especially if domain must contain a pole
- A single true latitude is specified

ARW Projections: Cylindrical Equidistant







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

Why do map projections matter?

Each choice of map projection and associated parameters distorts distances at a given point on the globe differently

Geographic grid distance in WRF at a point is given by

$$\Delta x_{geographical} = \Delta x_{nominal}/m$$

where *m* is a *map scale factor.*

Maximum stable timestep in WRF is determined by geographic grid distance, not nominal (i.e., namelist) grid distance!

Map scale factor is a 2-d field available in the geogrid output files

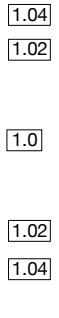
- Can easily check min/max map scale factor using, e.g., ncview!

Why do map projections matter?

Example:

- Nominally 27 km grid
- Lambert conformal projection
- True latitude 1 = 23.14
- True latitude 2 = 23.14

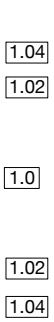
Choosing both true latitudes in the center of the WRF domain leads to maximum map scale factors of 1.0975, corresponding to a *minimum physical grid distance of 27/1.0975 = 24.6 km.*



Above: Contours of map scale factor (white; interval 0.01) with true latitudes (red).

Why do map projections matter?

We can reduce the maximum map scale factor at the expense of grid resolution...



Above: Reference projection as on previous slide; maximum grid distance is 27 km (at true latitude).

Above: The maximum map scale factor is 1.03, but the minimum is 0.924, corresponding to a physical distance of 29.2 km.

Why do map projections matter?

... but if we insist that the maximum grid distance is at most 27 km, we must reduce the *nominal* grid distance to accommodate the map scale factors!

Above: With a nominal grid distance of 27 km, the map scale factors of 0.924 in the center of the map correspond to a physical distance of 29.2 km.

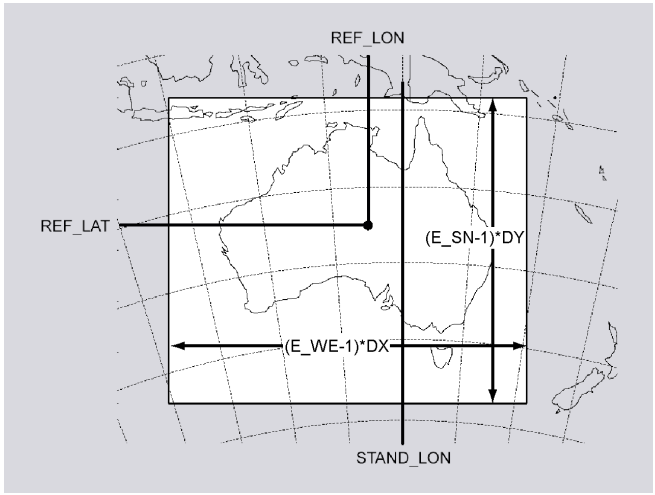
Above: Reducing the nominal grid distance to 25 km, the map scale factors of 0.924 in the center of the map correspond to a physical distance of 27.06 km.

Geogrid: Defining Model Domains

- Define projection of domains using a subset of the following parameters
  - **MAP_PROJ**: 'lambert', 'mercator', 'polar', or 'lat-lon'
  - **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*

See p. 3-9 and 3-43

Geogrid: Defining ARW Domains

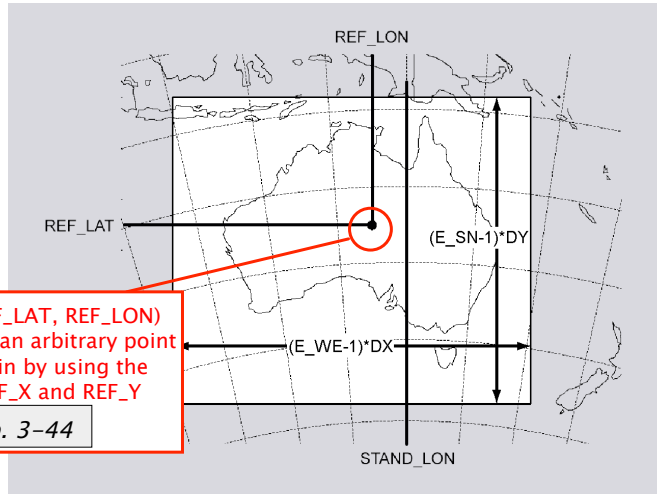


Geogrid: Defining Model Domains

- Define the area covered (dimensions and location) by 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
  - **E_SN**: Number of velocity points in south-north direction

See p. 3-13 and 3-42

Geogrid: Defining ARW Domains



In ARW, (REF_LAT, REF_LON) can refer to an arbitrary point in the domain by using the variables REF_X and REF_Y

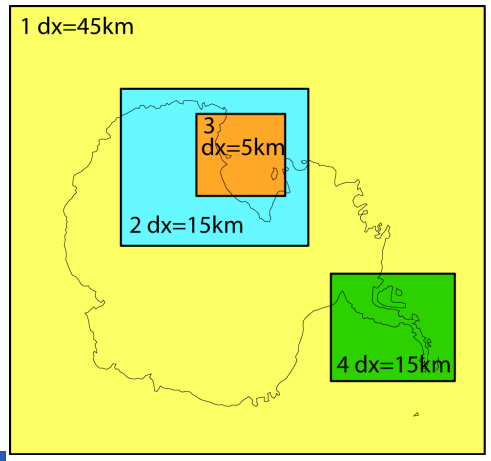
See p. 3-44

Geogrid: Nesting Basics

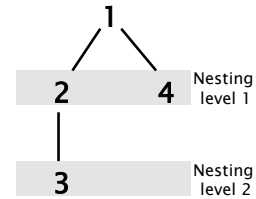
- A *nested domain* is a domain that is wholly contained within its *parent domain* and that receives information from its parent, and that may also feed information back to its parent
  - A nested domain has exactly one *parent*
  - A domain may have one or more *children*
- 2-*way nests* on the same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

Geogrid: Nesting Example

Example configuration - 4 domains



Each domain is assigned a *domain ID #*



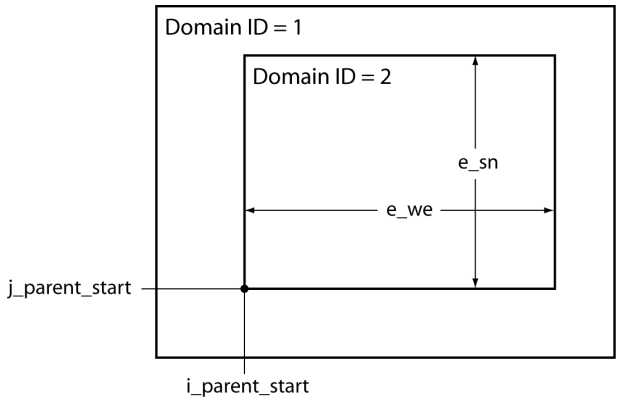
Nesting structure shown as a tree for the domains at left

Geogrid: Defining Nested Domains

- 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

See p. 3-20 and 3-42

Geogrid: Defining Nested Domains



The grid spacing (*dx*) of domain 2 is determined by grid spacing of domain 1 and the *parent_grid_ratio*

Geogrid: Defining Nested Domains

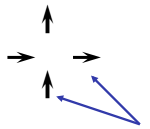
A nested domain must cover an integer number of parent-domain grid cells, and *e_we* and *e_sn* represent the number of *velocity-staggered points.*

Therefore, in ARW, nest dimensions must satisfy

$$\frac{e\_we - 1}{parent\_grid\_ratio} = n_i$$

$$\frac{e\_sn - 1}{parent\_grid\_ratio} = n_j$$

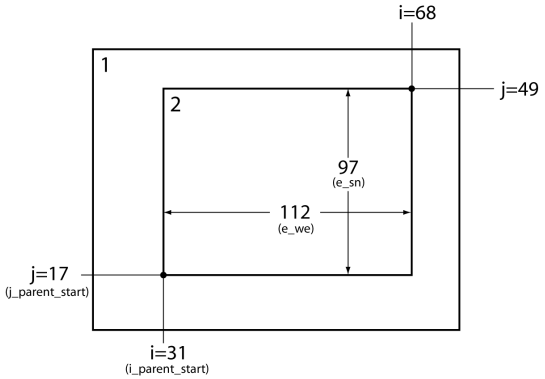
where n_i and n_j are any integers.



Velocity-staggered points, located on cell faces.

Geogrid: Nesting example

Assuming *parent_grid_ratio* = 3



In ARW, nest dimensions must satisfy

$$\frac{e\_we - 1}{parent\_grid\_ratio} = n_i$$

$$\frac{e\_sn - 1}{parent\_grid\_ratio} = n_j$$

where n_i and n_j are any integers.

Geogrid: Interpolating 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

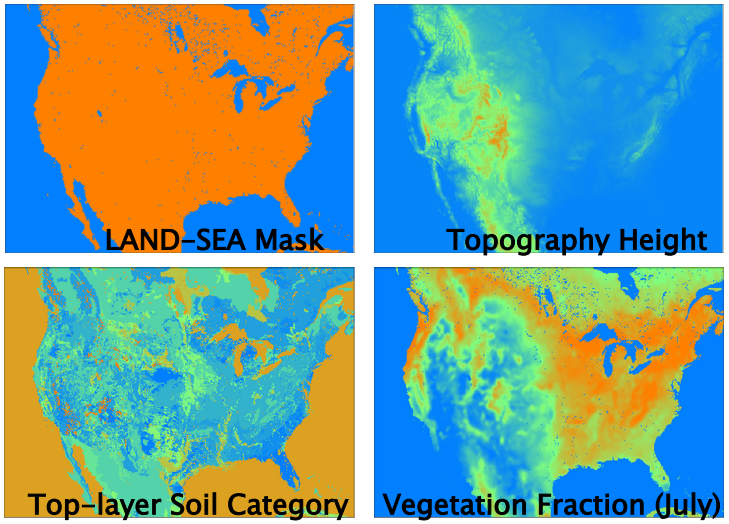
Geogrid: Program Output

- 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

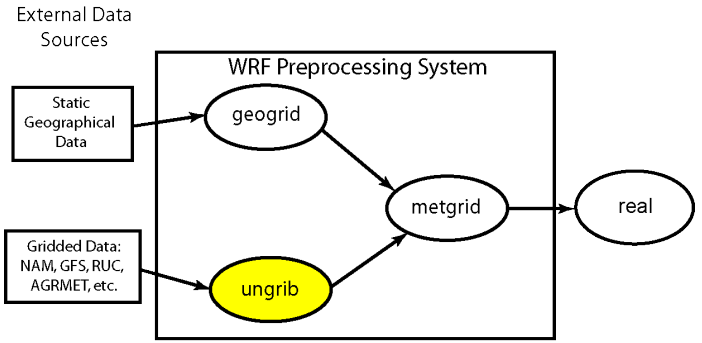
  (where *n* is the domain ID number)
- Example:

  geo_em.d01.nc
  geo_em.d02.nc (nest)
  geo_em.d03.nc (nest)

Geogrid: Example Output Fields



The *ungrib* program



ungrib: think un+grib

What is a GRIB file, anyway?

- 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 lossy 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
- 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

Ungrib: Vtables

How does ungrib know which fields to extract?

Using Vtables (think: 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: NAM 104, NAM 212, GFS, AGRMET, and others

Ungrib: Example 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
11	105	2		T	K	Temperature at 2 m
52	105	2		RH	%	Relative Humidity at 2 m
33	105	10		U	m s-1	U at 10 m
34	105	10		V	m s-1	V at 10 m
1	1	0		PSFC	Pa	Surface Pressure
130	102	0		PMSL	Pa	Sea-level Pressure
144	112	0	10	SM000010	kg m-3	Soil Moist 0-10 cm below grn layer (Up)
144	112	10	40	SM010040	kg m-3	Soil Moist 10-40 cm below grn layer
144	112	40	100	SM040100	kg m-3	Soil Moist 40-100 cm below grn layer
144	112	100	200	SM100200	kg m-3	Soil Moist 100-200 cm below gr layer
85	112	0	10	ST000010	K	T 0-10 cm below ground layer (Upper)
85	112	10	40	ST010040	K	T 10-40 cm below ground layer (Upper)
85	112	40	100	ST040100	K	T 40-100 cm below ground layer (Upper)
85	112	100	200	ST100200	K	T 100-200 cm below ground layer (Bottom)
91	1	0		SEAICE	proprtn	Ice flag
81	1	0		LANDSEA	proprtn	Land/Sea flag (1=land,2=sea in GRIB2)
7	1	0		HGT	m	Terrain field of source analysis
11	1	0		SKINTEMP	K	Skin temperature (can use for SST also)
65	1	0		SNOW	kg m-2	Water equivalent snow depth
223	1	0		CANWAT	kg m-2	Plant Canopy Surface Water
224	1	0		SOILCAT	Tab4.213	Dominant soil type category
225	1	0		VEGCAT	Tab4.212	Dominant land use category

Ungrib: GRIB2 Vtable Entries

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

Ungrib: Vtables

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, g2print*
- Use existing Vtable as a template
- Check documentation in Chapter 3 of the Users' Guide for more information about Vtables

See p. 3-35

Ungrib: 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

See p. 3-33

  - *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)

Ungrib: Program Output

- 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:

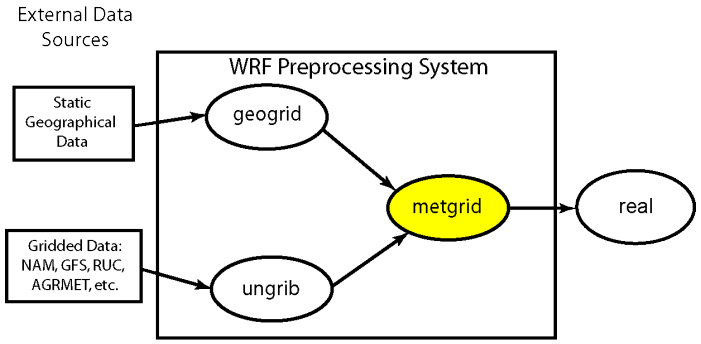
  FILE:2007-07-24_00

  FILE:2007-07-24_06

  FILE:2007-07-24_12

ungrib can also write intermediate files in the MM5 or WRF SI format! *(To allow for use of GRIB2 data with MM5, for example)*

The *metgrid* program



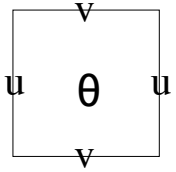
metgrid: think meteorological

The *metgrid* program

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

Metgrid: 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!*)

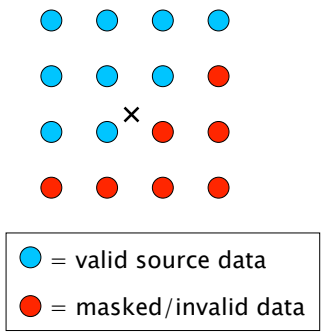


A single ARW grid cell, with "u", "v", and "θ" points labeled.

Metgrid: 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., 1×10^{30}) in field itself to identify masked grid points

Metgrid: Masked Interpolation



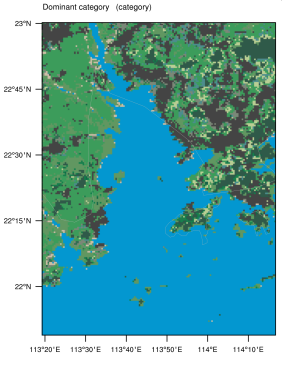
Suppose we need to interpolate to point X

- Using red points as valid data can give a bad interpolated value!
- Masked interpolation only uses valid blue points to interpolate to X

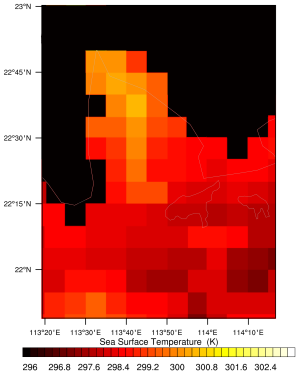
Not every interpolation option can handle masked points; we'll address this issue in the advanced WPS lecture

Metgrid: Masked Interpolation

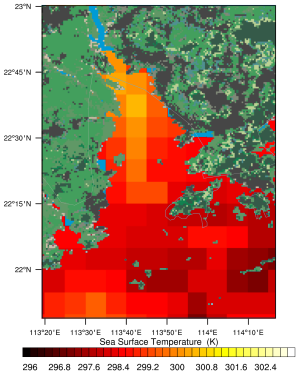
Common fields that require masked interpolation include SST, soil moisture, and soil temperature.



A high-resolution WRF domain centered on Zhujiang River Estuary.



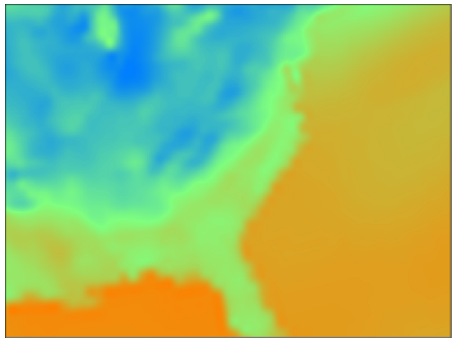
SST data on a 0.083-degree grid, with missing data (black) over land.



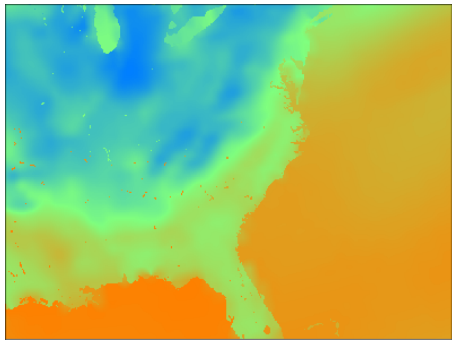
SST data overlaid with land use; blue areas represent WRF water cells that must receive SST values via masked interpolation.

Metgrid: Masked Interpolation

Masked interpolation can also be used for any field, e.g., to improve the resolution of coastlines in the field.



Skin temperature field interpolated from GFS 0.5-deg field with no mask using a sixteen-point interpolator.

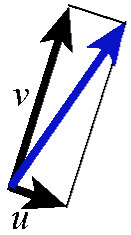


Skin temperature field interpolated using masks: GFS water points interpolated to model water points, GFS land points interpolated to model land points.

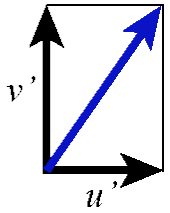
Metgrid: Wind Rotation

- Input wind fields (U-component + V-component) are either:
  - **Earth-relative**: U-component = westerly component; V-component = 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

Metgrid: Wind Rotation 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* rotations: one from source grid to earth grid and a second from earth grid to WRF grid

Metgrid: Constant Fields

- 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:2007-07-24_00'`

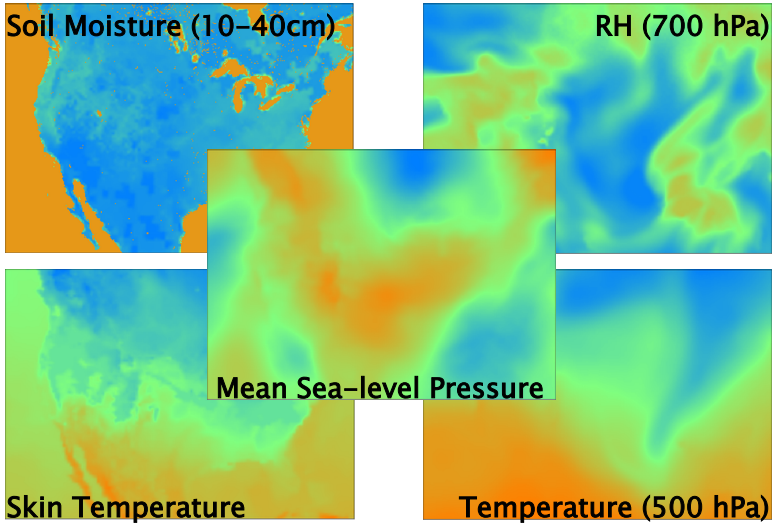
Metgrid: Program Output

- For coarse domain, one file per time period
  - In ARW, we also get the first time period for all nested grids
- Files contain static fields from geogrid plus interpolated meteorological fields
- Filenames:

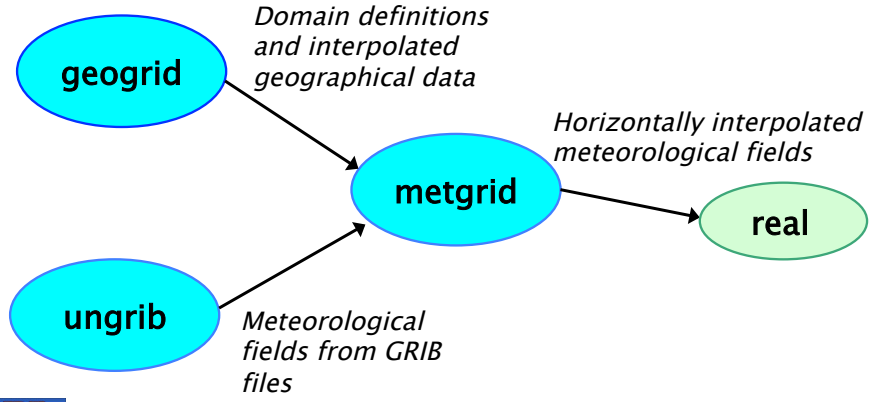
  ARW: `met_em.d0n.YYYY-MM-DD_HH:mm:ss.nc`

  (where *n* is the domain ID number)

Metgrid: Example Output

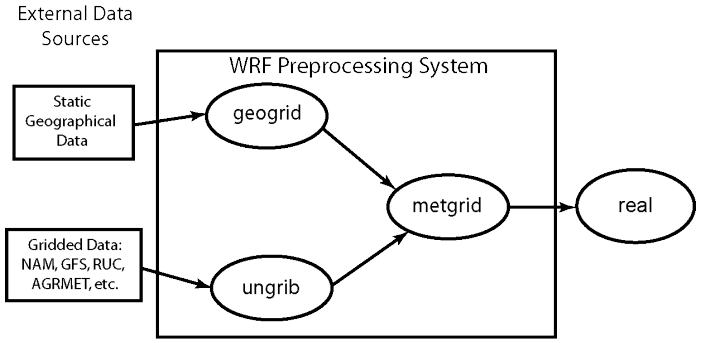


WPS Summary



And finally…

Vertical interpolation to WRF eta levels is performed in the *real* program



Questions?

Program Real: Description of General Functions

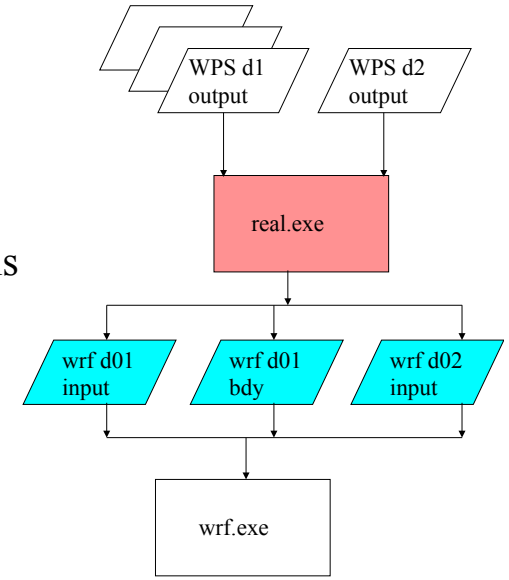
Dave Gill

Real

Description of General Functions

Dave Gill

gill@ucar.edu



Real program in a nutshell

- Function
- Standard input variables
- Base State
- Standard generated output
- Vertical interpolation
- Soil level interpolation
- Nested processing

Function

- The WRF model pre-processor is ***real.exe***
- The real.exe program is available ***serial*** or ***DM parallel*** (primarily for aggregate memory purposes, as opposed to timing performance)
- This program is automatically generated when the model is built and the requested use is for a real data case
- The real.exe program takes data ***from WPS*** and transform the data ***for WRF***
- Similar to the ARW idealized data pre-processor, real.exe is tightly coupled to the WRF model through the ***Registry***

Function

- ***3D forecast*** or simulation
- ***Meteorological input*** data that primarily originated from a previous forecast or analysis, probably via the WPS package
- Anticipated ***utilization of physics*** packages for microphysics, surface conditions, radiation, convection, and boundary layer (maybe usage of nudging capabilities)

Function

- A non-Cartesian ***projected domain***
  - Lambert conformal, Mercator, polar stereographic, rotated latitude/longitude (global or regional)
- Selection of ***realistic static fields*** of topography, land use, vegetation, and soil category data
- Requirement of ***time dependent*** lateral boundary conditions for a regional forecast

Function

- Generation of ***diagnostics*** necessary for assumed WRF model input
- Input field ***adjustment*** for consistency of static and time dependent fields (land mask with soil temperature, etc.)
- ARW: computation of ***reference*** and ***perturbation*** fields
- Generation of ***initial*** state for each of the requested domains
- Creation of a ***lateral boundary file*** for the most coarse domain
- ***Vertical interpolation*** for 3d meteorological fields and for sub-surface soil data

Standard Input Variables

- The metgrid program typically provides meteorological data to the real program.
- Coordinate:
  - The real program is able to input and correctly process any ***strictly monotonically oriented*** vertical coordinate
    - Isobaric: OK
    - Sigma: OK
    - Hybrid: OK

Standard Input Variables

- The metgrid program typically provides meteorological data to the real program.
- Mandatory:
  - 3d and surface: horizontal winds, temperature, relative humidity, geopotential height
  - 3d soil: soil temperature
  - 2d fields: surface pressure, sea-level pressure, land mask
- Optional (but desirable):
  - 3d soil: soil moisture
  - 2d fields: topography elevation of input data, SST, sea-ice, skin temperature

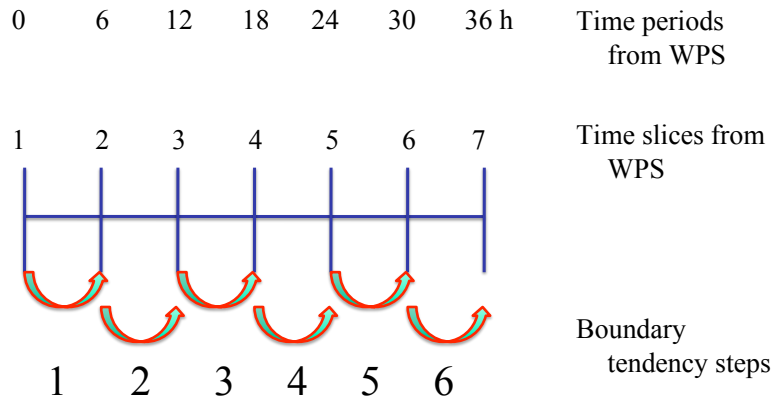
Base State

- Several of the mass-point fields are ***separated*** into a time-independent ***base state*** (also called a reference state) and a ***perturbation*** from the base state
- The base state fields are only functions of the ***topography*** and a few user-selectable constants
- If the ***topography changes***, such as with a moving nest, the base state fields are modified
- ***Feedback*** for 2-way nesting also impacts base state fields through topographic averaging – ***inside of the WRF model***
- No base state computations are required ***prior to the real program***

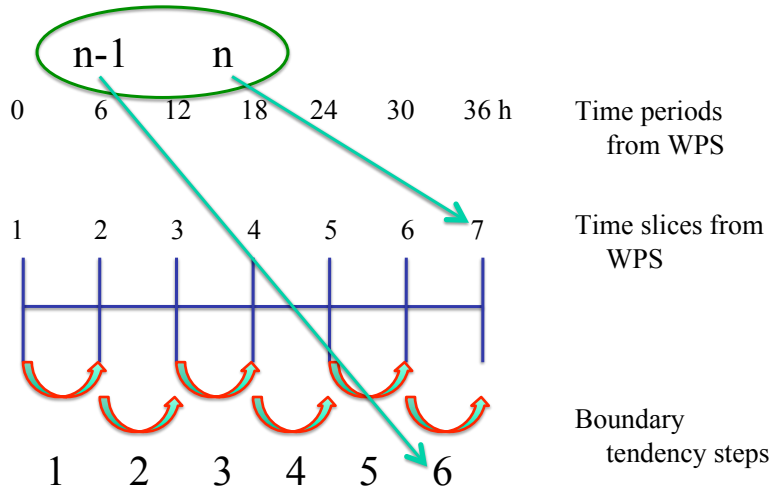
Standard Generated Output

- For regional forecasts, the real program generates both an both an initial (***wrfinput_d01***) and a lateral boundary (***wrfbdy_d01***)
- The boundary file is not required for ***global forecasts*** with ARW
- The ***initial condition*** file contains a ***single time period*** of data
- These files contain data used directly by the WRF model
- The initial condition file may be ingested by the ***WRFDA*** code (referred to as a *cold-start*)
- If ***n*** times were processed with WPS and real, the lateral boundary file contains ***n-1*** time slices

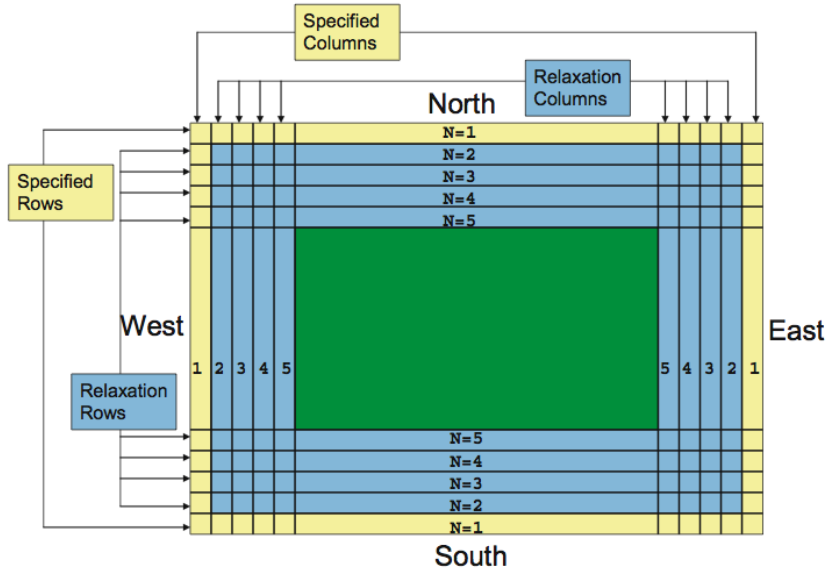
Lateral Boundary Condition Times



Lateral Boundary Condition Times



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



Vertical Interpolation

- A number of vertical interpolation options are available to users
- The options can have a significant impact on the initial conditions passed to the model
- More information is contained in the info file ***README.namelist*** in the ***run*** directory
- Options are located in the ***&domains*** namelist record of ***namelist.input***

Vertical Interpolation

Make sure input data is vertically ***ordered*** as expected

Input 3-D pressure and T, topo, Z, moisture used to compute total ***surface pressure***

Compute target ***vertical coordinate*** using normalized dry column pressure pressure

The ***h surfaces*** may be computed or selected

Vertically interpolate input fields in pressure to the h surfaces in dry pressure: default all variables linear in log(pressure)

Vertical Interpolation

- Select reasonable h levels, or let the real program do it for you
- Verify that the ***"thicknesses" are acceptable***, generally about the same value in the free-atmosphere and less than 1000 m
- It is ***SAFEST to NOT initially choose h values***
  - Initially, ***select the number*** of h levels
  - ***Plot profiles*** of the resultant heights
  - ***Adjust the hlevels*** accordingly
- A few namelist options, the terrain elevation, and eta levels completely define the model coordinate for the WRF code

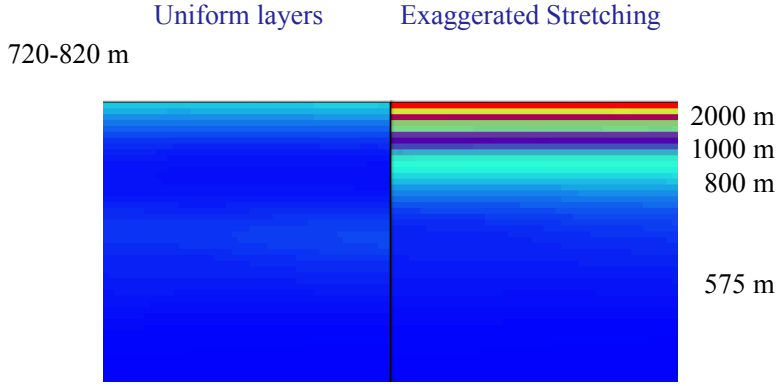
Vertical Interpolation

- The ***h surfaces*** are computed with a few NML parameters:

```
&domains
e_vert              = 50,     50,    50
p_top_requested     = 1000,

&dynamics
base_temp           = 290.
iso_temp            = 200
```

Vertical Interpolation

Vertical cross sections of model height field, with 50 vertical levels and ptop = 10 hPa, above the PBL.

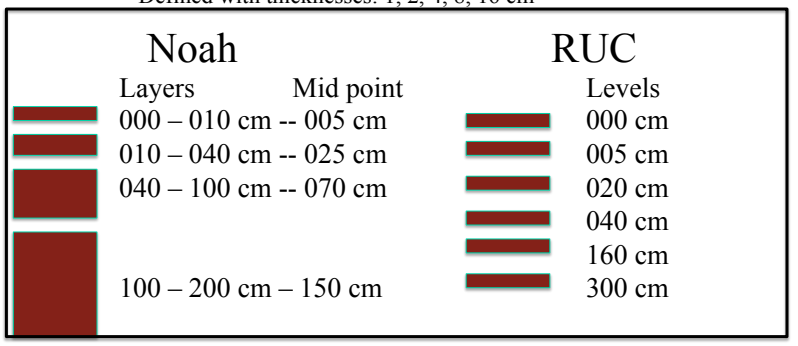


Physical Parameterization Settings

- The real program and the WRF model are tightly coupled
- Most physical parameterization settings in the namlist.input are IGNORED by real
- EXCEPT
  - ***sf_surface_physics***
  - Land surface model (processes soil temperature and soil moisture)
  - Different schemes in WRF use ***differing numbers of layers***
  - The layers are defined in real from the metgrid output

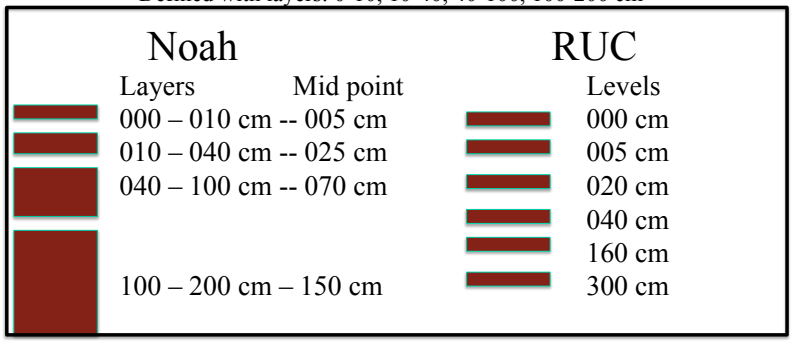
Soil Level Interpolation

- The WRF model supports several Land Surface schemes:
  - sf_surface_physics = 1, Slab scheme
  - 5 layers
  - Defined with thicknesses: 1, 2, 4, 8, 16 cm



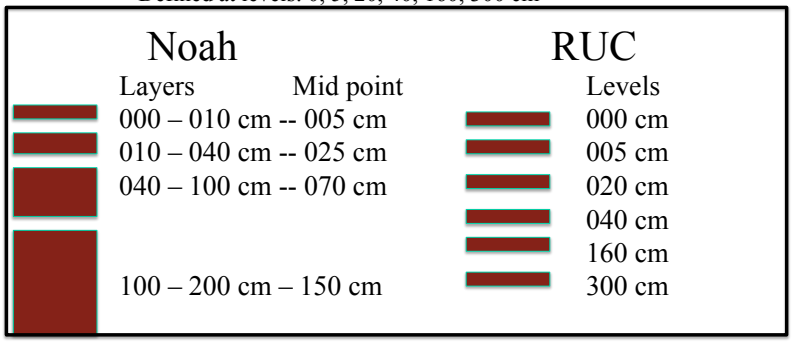
Soil Level Interpolation

- The WRF model supports several Land Surface schemes:
  - sf_surface_physics = 2, Unified Noah scheme
  - 4 layers
  - Defined with layers: 0-10, 10-40, 40-100, 100-200 cm



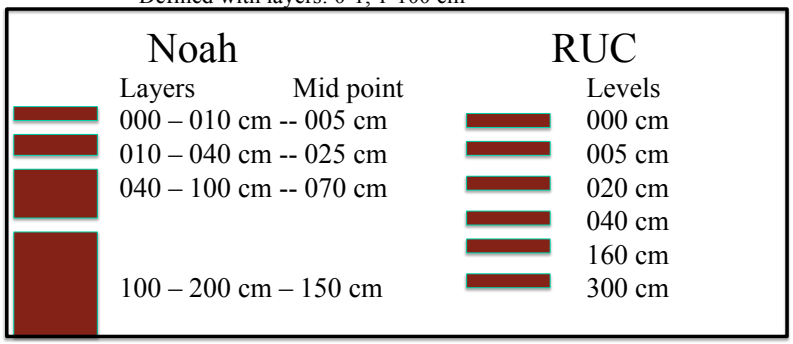
Soil Level Interpolation

- The WRF model supports several Land Surface schemes:
  - sf_surface_physics = 3, RUC scheme
  - 6 levels
  - Defined at levels: 0, 5, 20, 40, 160, 300 cm



Soil Level Interpolation

- The WRF model supports several Land Surface schemes:
  - sf_surface_physics = 7, PX scheme
  - 2 layers
  - Defined with layers: 0-1, 1-100 cm



Nested Processing

- May read ***multiple domain input files*** from metgrid
- Requires only the ***initial time for the fine domains***, unless doing nudging or SST update

- ***No horizontal interpolation from parent to child***

- ***No consistency checks*** between domains (handled in the feedback step for the WRF model)
- A ***wrfinput_d0x*** file is created for each processed input domain
- A ***lateral boundary file*** is created only for the ***most coarse*** domain

Real program in a nutshell

- Function
- Standard input variables
- Base State
- Standard generated output
- Vertical interpolation
- Soil level interpolation
- Nested processing

Real program in a nutshell: PART 2

- Access to everything
- Eta levels
- Metgrid flags
- Adding a variable for vertical interpolation
- Vertical interpolation
- Tracers
- Trajectories
- Options

Access to Everything

- The primary location to modify the real program is the dyn_em/module_initialize_real.F file
- Contains:
  - Registry information
  - All of the namelist settings selected
  - Variables from the Metgrid program
  - Variables to be sent to the WRF model
- Called for every time period, for every domain

Access to Everything

- The value of every variable input into the WRF model is controlled through module_initialize_real.F
- All variables are accessed through the derived data type "grid"

```
DO j=jts,MIN(jde-1,jte)
   DO i=its,MIN(ide-1,ite)
      grid%sst(i,j) = grid%sst(i,j) + 1
   END DO
END DO
```

Access to Everything

- The dynamics variables have two time levels, indicated by the _1 and _2 suffixes. Only the _2 variables are sent to WRF.
- Some variables sent to WRF are diagnostic only

```
DO j = jts, min(jde-1,jte)
   DO i = its, min(ide,ite)
      grid%u10(i,j)=grid%u_gc(i,1,j)
   END DO
END DO
```

Eta Levels

- The vertical coordinate, eta, used in the WRF model is defined inside of the real program.
- The user may allow the real program to choose the levels (select only the number of levels in the namelist.input file)

```
 &domains
 e_vert    = 30,    30,    30,
 /

&domains
 e_vert    = 30,    40,    50,
 /
```

Eta Levels

- Often the user needs to specify the eta levels (coordinate this with your model top)
- Use the automatic generation to your advantage
- Specify how many levels ABOVE the PBL that you require. Add 8 to this value. For example, you require 50 vertical levels above the PBL.

```
 &domains
 e_vert    = 58,    58,    58,
 /
```

Eta Levels

- Run the real program (single or small domain, one time level), make sure the level thicknesses are OK (< 1000 m)

```
 Converged znw(kte) should be about 0.0 =  -5.2081142E-04
Full level index =    1     Height =     0.0 m
Full level index =    2     Height =    56.6 m      Thickness =   56.6 m
Full level index =    3     Height =   137.9 m      Thickness =   81.4 m
Full level index =    4     Height =   244.7 m      Thickness =  106.8 m
Full level index =    5     Height =   377.6 m      Thickness =  132.9 m
Full level index =    6     Height =   546.3 m      Thickness =  168.7 m
Full level index =    7     Height =   761.1 m      Thickness =  214.8 m
Full level index =    8     Height =  1016.2 m      Thickness =  255.0 m
Full level index =    9     Height =  1207.1 m      Thickness =  190.9 m
Full level index =   10     Height =  1401.8 m      Thickness =  194.6 m
Full level index =   11     Height =  1600.3 m      Thickness =  198.5 m
Full level index =   12     Height =  1802.8 m      Thickness =  202.5 m
Full level index =   13     Height =  2196.1 m      Thickness =  393.3 m
```

Eta Levels

- Get the computed levels from ncdump, after running the real program

```
> ncdump -v ZNW wrfinput_d01
data:

 ZNW =
  1, 0.993, 0.983, 0.97, 0.954, 0.934, 0.909, 0.88, 0.8587637, 0.8375274,
    0.8162911, 0.7950548, 0.7550299, 0.7165666, 0.6796144, 0.6441237,
    0.6100466, 0.5773363, 0.5459476, 0.5158363, 0.4869595, 0.4592754,
    0.4327437, 0.407325, 0.382981, 0.3596745, 0.3373697, 0.3160312,
    0.2956253, 0.2761188, 0.2574798, 0.2396769, 0.2226802, 0.2064602,
    0.1909885, 0.1762376, 0.1621807, 0.1487919, 0.1360459, 0.1239184,
    0.1124378, 0.1017038, 0.09166772, 0.08228429, 0.07351105, 0.06530831,
    0.05763897, 0.05046835, 0.04376402, 0.03749565, 0.0316349, 0.02615526,
    0.02103195, 0.01624179, 0.01176313, 0.007575703, 0.003660574, 0 ;
```

Eta Levels

- Re-run the real program (all domains, all time periods) with the new levels in the nml variable **`eta_levels`**
- Replace the **PBL values** with those of your choosing.
- Augment the number of vertical levels (e_vert)
- Note that both e_vert and eta_levels are full levels

Eta Levels

```
&domains
eta_levels =
    1, 0.993, 0.983, 0.97, 0.954, 0.934, 0.909, 0.88,
    0.8587637, 0.8375274,
    0.8162911, 0.7950548, 0.7550299, 0.7165666, 0.6796144, 0.6441237,
    0.6100466, 0.5773363, 0.5459476, 0.5158363, 0.4869595, 0.4592754,
    0.4327437, 0.407325, 0.382981, 0.3596745, 0.3373697, 0.3160312,
    0.2956253, 0.2761188, 0.2574798, 0.2396769, 0.2226802, 0.2064602,
    0.1909885, 0.1762376, 0.1621807, 0.1487919, 0.1360459, 0.1239184,
    0.1124378, 0.1017038, 0.09166772, 0.08228429, 0.07351105, 0.06530831,
    0.05763897, 0.05046835, 0.04376402, 0.03749565, 0.0316349, 0.02615526,
    0.02103195, 0.01624179, 0.01176313, 0.007575703, 0.003660574, 0
/
```

- Maybe replace with

```
    1, 0.999, 0.998, 0.996, 0.993, 0.990, 0.980. 0.970, 0.960, 0.950,
       0.940, 0.930, 0.920, 0.910, 0.900, 0.890, 0.880, 0.870,
```

Eta Levels

- For vertical nesting, follow the similar procedure for each domain.
- Each domain will need a specification of eta levels
- The assignment of the single eta_levels array is split into pieces for easier understanding

Eta Levels

```
 &domains
 max_dom            = 2,
 e_vert             = 35,    45,
 eta_levels(1:35)   = 1., 0.993, 0.983, 0.97, 0.954, 0.934,
                      0.909, 0.88, 0.840, 0.801, 0.761, 0.722,
                      0.652, 0.587, 0.527, 0.472, 0.421, 0.374,
                      0.331, 0.291, 0.255, 0.222, 0.191, 0.163,
                      0.138, 0.115, 0.095, 0.077, 0.061, 0.047,
                      0.035, 0.024, 0.015, 0.007, 0.
eta_levels(36:81)  =  1.0000, 0.9946, 0.9875, 0.9789, 0.9685,
                      0.9562, 0.9413, 0.9238, 0.9037, 0.8813,
                      0.8514, 0.8210, 0.7906, 0.7602, 0.7298,
                      0.6812, 0.6290, 0.5796, 0.5333, 0.4901,
                      0.4493, 0.4109, 0.3746, 0.3412, 0.3098,
                      0.2802, 0.2524, 0.2267, 0.2028, 0.1803,
                      0.1593, 0.1398, 0.1219, 0.1054, 0.0904,
                      0.0766, 0.0645, 0.0534, 0.0433, 0.0341,
                      0.0259, 0.0185, 0.0118, 0.0056, 0.
 vert_refine_method = 0,     2,
```

Metgrid Flags

- The real program and the WRF model are able to communicate directly through the Registry file
- The real program is only able to talk with the Metgrid program through the input data stream
- Specific information about the incoming data is contained in special flags that the user may set in the Metgrid table file – usually, related to THIS VARIABLE EXISTS

```
========================================
name=PMSL
        interp_option=sixteen_pt+four_pt+average_4pt
        flag_in_output=FLAG_SLP
========================================
```

Metgrid Flags

```
> ncdump -h met_em.d01.2000-01-24_12:00:00.nc | grep FLAG
                :FLAG_METGRID = 1 ;
                :FLAG_EXCLUDED_MIDDLE = 0 ;
                :FLAG_SOIL_LAYERS = 1 ;
                :FLAG_SNOW = 1 ;
                :FLAG_PSFC = 1 ;
                :FLAG_SM000010 = 1 ;
                :FLAG_SM010040 = 1 ;
                :FLAG_SM040100 = 1 ;
                :FLAG_SM100200 = 1 ;
                :FLAG_ST000010 = 1 ;
                :FLAG_ST010040 = 1 ;
                :FLAG_ST040100 = 1 ;
                :FLAG_ST100200 = 1 ;
                :FLAG_SLP = 1 ;
                :FLAG_TAVGSFC = 1 ;
                :FLAG_QNWFA = 1 ;
                :FLAG_QNIFA = 1 ;
                :FLAG_SOILHGT = 1 ;
                :FLAG_MF_XY = 1 ;
```

Metgrid Flags

- The real program uses this information when deciding how to do many operations:
  - Is the input from Metgrid?
  - Method to compute surface pressure
  - Use RH vs mixing ratio vs specific humidity computations
  - Excluded middle processing
  - Average surface air temperature for lake temperatures
  - Water/Ice friendly vertical interpolation
  - Which levels of soil data are present
- All flags for the Metgrid to real data transfer are contained in share/module_optional_input.F

Metgrid Flags

```
flag_slp      = 0

flag_name(1:8) = 'SLP     '
CALL wrf_get_dom_ti_integer ( fid, 'FLAG_' // &
     flag_name, itmp, 1, icnt, ierr )
IF ( ierr .EQ. 0 ) THEN
   flag_slp      = itmp
END IF
```

Adding a Variable for Vertical Interpolation

- This process is manual
- Every new input 3d variable that needs to be interpolated needs to have an explicit block of code added
- Mass-point variables (such as would be used in all physics schemes) are straight forward, as they may be largely copied using the existing templates already in place
- Most vertical interpolation options are supplied from the namelist.input file
- All interpolation is handled in dry pressure

Adding a Variable for Vertical Interpolation

```
CALL vert_interp ( grid%t_gc , grid%pd_gc , &
     grid%t_2 , grid%pb , &
     grid%tmaxw , grid%ttrop , grid%pmaxw , grid%ptrop , &
     grid%pmaxwnn , grid%ptropnn , &
     flag_tmaxw , flag_ttrop , &
     config_flags%maxw_horiz_pres_diff , &
     config_flags%trop_horiz_pres_diff , &
     config_flags%maxw_above_this_level , &
     num_metgrid_levels , 'T' , &
     interp_type , lagrange_order , t_extrap_type , &
     lowest_lev_from_sfc , use_levels_below_ground , &
     use_surface , zap_close_levels , force_sfc_in_vinterp , &
     ids , ide , jds , jde , kds , kde , &
     ims , ime , jms , jme , kms , kme , &
     its , ite , jts , jte , kts , kte )
```

Tracers

- The WRF model is able to advect arrays of passive scalars (tracer 4d array)
- As with all other variables going into the WRF model, this data is available to be set in the real program
- These variables must be coordinated with the Registry names, as the tracer index is an automatically manufactured name

```
# Tracer Scalars
#
state real tr17_1 ikjftb  tracer 1 -  irhusdf=(bdy_interp:dt) \
      "tr17_1" "tr17_1" "Dimensionless"
```

Tracers

- As with all 4d arrays, no space is allocated unless the packaged variables are requested for processing at run-time

```
package   tracer_test1  tracer_opt==2 -  tracer:tr17_1
```

Tracers

```
!  Template for initializing tracer arrays.
!  A small plane in the middle of the domain at
!  lowest model level is defined.

IF (config_flags%tracer_opt .eq. 2) THEN
  DO j = (jde + jds)/2 - 4, (jde + jds)/2 + 4, 1
    DO i = (ide + ids)/2 - 4, (ide + ids)/2 + 4, 1
      IF ( ( its .LE. i .and. ite .GE. i ) .and. &
           ( jts .LE. j .and. jte .GE. j ) ) THEN
        tracer(i, 1, j, P_tr17_1) = 1.
      END IF
    END DO
  END DO
END IF
```

Trajectories

- The user may specify specific (i,j,k) locations in the model domain to follow parcels: traj_i, traj_j, traj_k (hard coded in the module_initialize_real.F file)
- The current number of trajectory locations is small, 25, and is a run-time option that the user sets in the nml file

```
&domain
 num_traj            = 25,

&physics
 traj_opt            = 1,
```

Trajectories

- The trajectory code uses the lat,lon locations, so the initial (i,j) value of the lat,lon is assigned

```
IF (config_flags%num_traj .gt. 0 .and.
    config_flags%traj_opt .gt. 0) THEN
  DO j = (jde + jds)/2 - 2, (jde + jds)/2 + 2, 1
    DO i = (ide + ids)/2 - 2, (ide + ids)/2 + 2, 1
      IF ( its .LE. i    .and. ite .GE. i   .and.  &
           jts .LE. j    .and. jte .GE. j ) THEN
        grid%traj_i   (icount) = i
        grid%traj_j   (icount) = j
        grid%traj_k   (icount) = 10
        grid%traj_lat (icount) = grid%xlat(i,j)
        grid%traj_long(icount) = grid%xlong(i,j)
      END IF
```

Options

- When there are strong normal topo gradients along the outer rows and columns of the most-coarse domain, smoothing the topography to match the incoming first guess data is a good idea.
- This is the same sort processing that is done to make the child and parent domains more consistent in the area of the LBC forcing

```
&domains
 smooth_cg_topo = .true.
/
```

Real program in a nutshell: PART 2

- Access to everything
- Eta levels
- Metgrid flags
- Adding a variable for vertical interpolation
- Vertical interpolation
- Tracers
- Trajectories
- Options

WRF and WPS: Compile

Kelly Werner

WRF & WPS: Compile

Kelly Werner
NCAR/MMM
December 2015

1

Installing Steps

- *Check system requirements*
- Installing libraries
- Download source data
- Compile WRFV3
- Compile WPS
- Download initial/BC datasets

2

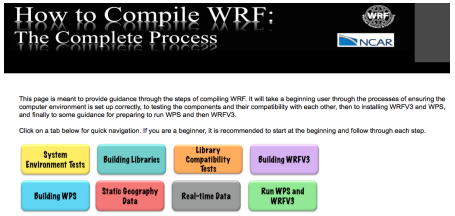
System Requirements

- On what kinds of systems will WRF run?
  - Generally any 32- or 64-bit hardware, running a UNIX-like operating system
  - You may also use dual-booting into a UNIX-like OS (e.g., Windows with Linux built parallel)
- Examples of acceptable systems:
  - Laptops, desktops, and clusters running Linux
  - Laptops and desktops running MacOS X
  - Clusters running Unix-like: Linux, AIX

3

Check System Requirements

- Webpage: http://www2.mmm.ucar.edu/wrf/OnLineTutorial/compilation_tutorial.php



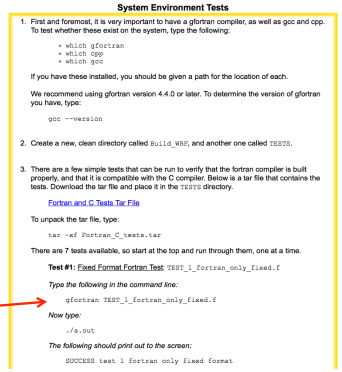
4

Check System Requirements

- It is mandatory to have a Fortran (e.g., gfortran) compiler, a C compiler, and cpp on your system. To test whether these exist on your system, type:
  - which gfortran
  - which cpp
  - which gcc
  - If installed, you will be given a path for each
- Fortran compiler should be version 4.4.0, or later

  Check this by typing (for csh):

  ```
  gcc -version
  ```
- Tests available for checking that your fortran compiler is built properly, and that it is compatible with the C compiler.



System Environment Tests

1. First and foremost, it is very important to have a gfortran compiler, as well as gcc and cpp. To test whether these exist on the system, type the following:

   ```
   % which gfortran
   % which cpp
   % which gcc
   ```

   If you have these installed, you should be given a path for the location of each.

   We recommend using gfortran version 4.4.0 or later. To determine the version of gfortran you have, type:

   ```
   gcc --version
   ```

2. Create a new, clean directory called `Build_WRF`, and another one called `TESTS`.

3. There are a few simple tests that can be run to verify that the fortran compiler is built properly, and that it is compatible with the C compiler. Below is a tar file that contains the tests. Download the tar file and place it in the `TESTS` directory.

   Fortran and C Tests Tar File

   To unpack the tar file, type:

   ```
   tar -xf Fortran_C_tests.tar
   ```

   There are 7 tests available, so start at the top and run through them, one at a time.

   Test #1: Fixed Format Fortran Test: `TEST_1_fortran_only_fixed.f`

   Type the following in the command line:

   ```
   gfortran TEST_1_fortran_only_fixed.f
   ```

   Now type:

   ```
   ./a.out
   ```

   The following should print out to the screen:

   ```
   SUCCESS test 1 fortran only fixed format
   ```


Additional Necessary Requirements

- Scripting languages (testing available in test package):
  - csh
  - perl
  - sh
- UNIX commands:

ar	head	sed	awk
hostname	sleep	cat	ln
sort	cd	ls	tar
cp	make	touch	cut
mkdir	tr	expr	mv
uname	file	nm	wc
grep	printf	which	gzip
rm			

Installing Steps

- Check system requirements
- *Installing libraries*
- Download source data
- Compile WRFV3
- Compile WPS
- Download initial/BC datasets

Installing Libraries

- NetCDF (needed by WRF and WPS)
- Optional libraries for GRIB2 meteteorological data support
  - JasPer (JPEG 2000 “lossy” compression library)
  - PNG (“lossless” compression library)
  - Zlib (compression library used by PNG)
- Optional MPI library (for building in parallel):
  - MPICH2

Installing Libraries

- Installation of these libraries (MPICH2, NetCDF, JasPer, zlib, and libpng) is NOT part of the WPS and WRF installation scripts
- VERY IMPORTANT!
  - Make sure these libraries are installed using the same compilers as will be used to install WRF and WPS
- Downloads for the libraries, with installation instructions, and library compatibility tests are also included on the compilation website

9

Installing Libraries: NetCDF

```
setenv DIR directory-where-your-tar-files-are
setenv CC gcc
setenv CXX g++
setenv FC gfortran
setenv FCFLAGS -m64      # FCFLAGS may be needed on some systems
setenv F77 gfortran
setenv FFLAGS -m64       # FFLAGS may be needed on some systems

tar xzvf netcdf-4.1.3.tar.gz       # no '.gz' if downloaded to most Macs
cd netcdf-4.1.3
./configure --prefix=$DIR/netcdf --disable-dap  --disable-netcdf-4 --
disable-shared
make
make install
setenv PATH $DIR/netcdf/bin:$PATH
setenv NETCDF $DIR/netcdf
cd ..
```

10

Installing Libraries: MPICH2

- In principle, any implementation of the MPI-2 standard should work with WRF; however, we have the most experience with MPICH
- *Assuming environment variables for netCDF install are already set:*

```
tar xzvf mpich-3.0.4.tar.gz   # no '.gz' if downloaded to most Macs
cd mpich-3.0.4
./configure --prefix=$DIR/mpich
make
make install
setenv PATH $DIR/mpich/bin:$PATH
cd ..
```

11

Installing Libraries: zlib

- *Assuming environment variables from netCDF install are already set:*

```
tar xzvf zlib-1.2.7.tar.gz         # no '.gz' if downloaded to most Macs
cd zlib-1.2.7
./configure --prefix=$DIR/zlib
make
make install
cd ..
```

12

Installing Libraries: libpng

- *Assuming environment variables from netCDF install are already set*

```
tar xzvf libpng-1.2.50.tar.gz      # no '.gz' if downloaded to most Macs
cd libpng-1.2.50
./configure --prefix=$DIR/libpng
make
make install
cd ..
```

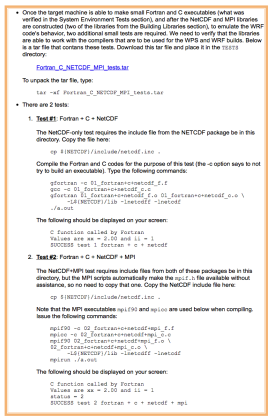

Installing Libraries: JasPer

- *Assuming environment variables from netCDF install are already set*

```
tar xzvf jasper-1.900.1.tar.gz     # no '.gz' if downloaded to most Macs
cd jasper-1.900.1
./configure --prefix=$DIR/jasper
make
make install
cd ..
```


Installing Libraries: Compatibility

- Make sure libraries are compatible with compilers
- Test 1
  - Fortran + C + netCDF
- Test 2
  - Fortran + C + netCDF + MPI



Library Compatibility Tests

- Once the target machine is able to make small Fortran and C executables (what was verified in the System Environment Tests section), and after the NetCDF and MPI libraries are constructed (two of the libraries from the Building Libraries section), to emulate the WRF code's behavior, two additional small tests are required. We need to verify that the libraries are able to work with the compilers that are to be used for the WPS and WRF builds. Below is a tar file that contains these tests. Download this tar file and place it in the `TESTS` directory:

  Fortran_C_NETCDF_MPI_tests.tar

  To unpack the tar file, type:

  ```
  tar -xf Fortran_C_NETCDF_MPI_tests.tar
  ```

- There are 2 tests:

1. **Test #1**: Fortran + C + NetCDF

   The NetCDF-only test requires the include file from the NETCDF package be in this directory. Copy the file here:

   ```
   cp ${NETCDF}/include/netcdf.inc .
   ```

   Compile the Fortran and C codes for the purpose of this test (the -c option says to not try to build an executable). Type the following commands:

   ```
   gfortran -c 01_fortran+c+netcdf_f.f
   gcc -c 01_fortran+c+netcdf_c.c
   gfortran 01_fortran+c+netcdf_f.o 01_fortran+c+netcdf_c.o \
       -L${NETCDF}/lib -lnetcdff -lnetcdf
   ./a.out
   ```

   The following should be displayed on your screen:

   ```
   C function called by Fortran
   Values are xx = 2.00 and ii = 1
   SUCCESS test 1 fortran + c + netcdf
   ```

2. **Test #2**: Fortran + C + NetCDF + MPI

   The NetCDF+MPI test requires include files from both of these packages be in this directory, but the MPI scripts automatically make the `mpif.h` file available without assistance, so no need to copy that one. Copy the NetCDF include file here:

   ```
   cp ${NETCDF}/include/netcdf.inc .
   ```

   Note that the MPI executables `mpif90` and `mpicc` are used below when compiling. Issue the following commands:

   ```
   mpif90 -c 02_fortran+c+netcdf+mpi_f.f
   mpicc -c 02_fortran+c+netcdf+mpi_c.c
   mpif90 02_fortran+c+netcdf+mpi_f.o \
   02_fortran+c+netcdf+mpi_c.o \
       -L${NETCDF}/lib -lnetcdff -lnetcdf
   mpirun -/a.out
   ```

   The following should be displayed on your screen:

   ```
   C function called by Fortran
   Values are xx = 2.00 and ii = 1
   status = 2
   SUCCESS test 2 fortran + c + netcdf + mpi
   ```


Installing Steps

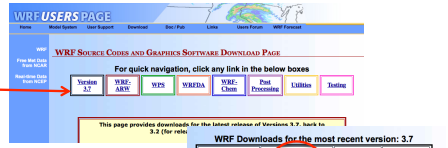
- Check system requirements
- Installing libraries
- *Download source data*
- Compile WRFV3
- Compile WPS
- Download initial/BC datasets

Download WRF & WPS Code

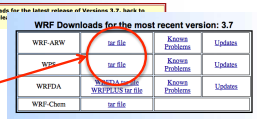
- Download WRF & WPS source code from:

  http://www2.mmm.ucar.edu/wrf/users/download/get_source.html
  - Click 'New User,' register and download, or
  - Click 'Returning User,' enter your email, and download

Step 1: Click here for the latest released code (recommended)

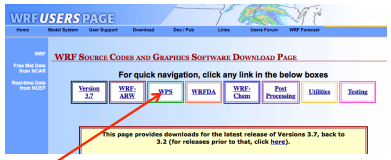


Step 2: Click on tar files to download



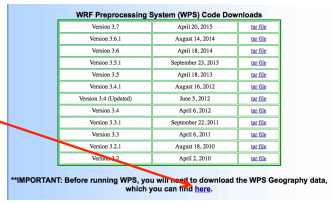
Download Static Geographical Data

- From the WRF Download page:
  http://www2.mmm.ucar.edu/wrf/users/download/get_sources.html



Step 1: Click 'WPS' box

Step 2: Click 'here' to get geography data

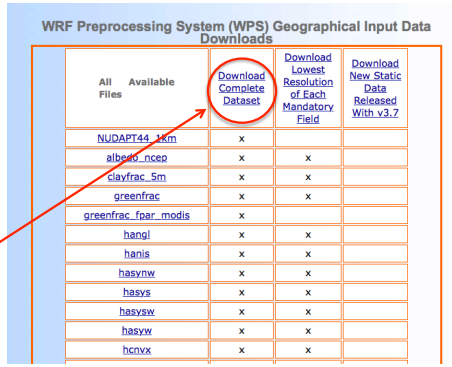


Download Static Geographical Data

- Geographical Input and Data Download Page:
  http://www2.mmm.ucar.edu/wrf/users/download/get_sources_wps_geog.html

geog.tar.gz
~ 15 GB when uncompressed

This is the one you want



Installing Steps

- Check system requirements
- Installing libraries
- Download source data
- *Compile WRFV3*
- Compile WPS
- Download initial/BC datasets

Choosing a Compiler

Compiler	Compile Time	Run Time
GNU 4.8.2 *FREE*	12.63 Mins	4.18 Mins
Intel 12.1.5	27.75 Mins	3.88 Mins
PGI 13.3-0	24.86 Mins	4.25 Mins

*Compile: dmpar/nesting, no large-file support

*Run: single domain, small domain (74x61), 6 hours, 16 processors

Step 1: Configure for WRFV3

- Inside the WRFV3/ directory, type: **`./configure`**

```
checking for perl5... no
checking for perl... found /usr/bin/perl (perl)
Will use NETCDF in dir: /glade/apps/opt/netcdf/4.3.0/intel/12.1.5
PHDF5 not set in environment. Will configure WRF for use without.
Will use 'time' to report timing information
$JASPERLIB or $JASPERINC not found in environment, configuring to build without grib2 I/O...
------------------------------------------------------------------------
Please select from among the following Linux x86_64 options:

  1. (serial)   2. (smpar)   3. (dmpar)   4. (dm+sm)   PGI (pgf90/gcc)
  5. (serial)   6. (smpar)   7. (dmpar)   8. (dm+sm)   PGI (pgf90/pgcc): SGI MPT
  9. (serial)  10. (smpar)  11. (dmpar)  12. (dm+sm)   PGI (pgf90/gcc): PGI accelerator
 13. (serial)  14. (smpar)  15. (dmpar)  16. (dm+sm)   INTEL (ifort/icc)
                                         17. (dm+sm)   INTEL (ifort/icc): Xeon Phi (MIC architecture)
 18. (serial)  19. (smpar)  20. (dmpar)  21. (dm+sm)   INTEL (ifort/icc): Xeon (SNB with AVX mods)
 22. (serial)  23. (smpar)  24. (dmpar)  25. (dm+sm)   INTEL (ifort/icc): SGI MPT
 26. (serial)  27. (smpar)  28. (dmpar)  29. (dm+sm)   INTEL (ifort/icc): IBM POE
 30. (serial)               31. (dmpar)                PATHSCALE (pathf90/pathcc)
 32. (serial)  33. (smpar)  34. (dmpar)  35. (dm+sm)   GNU (gfortran/gcc)
 36. (serial)  37. (smpar)  38. (dmpar)  39. (dm+sm)   IBM (xlf90_r/cc_r)
 40. (serial)  41. (smpar)  42. (dmpar)  43. (dm+sm)   PGI (ftn/gcc): Cray XC CLE
 44. (serial)  45. (smpar)  46. (dmpar)  47. (dm+sm)   CRAY CCE (ftn/gcc): Cray XE and XC
 48. (serial)  49. (smpar)  50. (dmpar)  51. (dm+sm)   INTEL (ftn/icc): Cray XC
 52. (serial)  53. (smpar)  54. (dmpar)  55. (dm+sm)   PGI (pgf90/pgcc)
 56. (serial)  57. (smpar)  58. (dmpar)  59. (dm+sm)   PGI (pgf90/gcc): -f90=pgf90
 60. (serial)  61. (smpar)  62. (dmpar)  63. (dm+sm)   PGI (pgf90/pgcc): -f90=pgf90

Enter selection [1-63] :


Compile for nesting? (0=no nesting, 1=basic, 2=preset moves, 3=vortex following) [default 0]:
```

- Output from configuration: a file called 'configure.wrf'

Configure Debugging Options for WRFV3

- ./configure –d
  - No optimization
  - Extra debugging
- ./configure –D
  - No optimization
  - Checks uninitialized variables, floating point traps, etc.
  - Useful for adding/updating new code
- ./configure –r8
  - Double precision for Intel and PGI
  - Does not work with GNU

Parallel Compile Option for WRFV3

- To build WRF in parallel
  - *setenv J "-j 2"*
  - On average, only 60% of original time

# of Processors	Time to Compiler
1	22.8 Mins
2	14.92 Mins
3	9.33 Mins
4	8.02 Mins
5	7.23 Mins
6	6.68 Mins

*Around 4 processors, it reaches state of equilibrium

configure.wrf File: Useful Tips

- NETCDFPATH : internally set by build system based on $NETCDF
- PNETCDF = For users who have access to parallel netcdf, use the env variable PNETCDF identically to how NETCDF is set (point to the PNETCDF top-level directory)

25

Step 2: Compile WRFV3

- In the WRFV3/ directory, type:

  `./compile em_case >& log.compile` → Important in case there are compile problems

  Where **`em_case`** is one of the following
  (type **`./compile`** to see all options)

`em_real` (3d real case)
`em_quarter_ss`
`em_b_wave`
`em_les`
`em_heldsuarez`
`em_tropical_cyclone`
`em_convrad`
→ 3d Ideal

`em_hill2d_x`
`em_squall2d_x`
`em_squall2d_y`
`em_grav2d_x`
`em_seabreeze2d_x`
→ 2d Ideal

`em_scm_xy` (1d ideal)

Compilation should take at most ~30 mins

26

Successful Compilation

- If the compilation is successful, you should find these executables in **`WRFV3/main`** (non-zero size):

  Real data case:

  `wrf.exe` – model executable
  `real.exe` – real data initialization
  `ndown.exe` – one-way nesting
  `tc.exe` – for tc bogusing (serial only)

  Ideal case:

  `wrf.exe` – model executable
  `ideal.exe` – ideal case initialization

  ***Note: Each ideal case compile creates a different executable, but with the same name**

- These executables are linked to 2 different directories (**`WRFV3/run`** and **`WRFV3/test/em_real`**). You can go to either place to run WRF.

27

Unsuccessful Compilation

- Use your 'log.compile' file to search for errors!
  - Search for 'Error' with a capital 'E'
- Use our Frequently Asked Questions web page for help
  - www2.mmm.ucar.edu/wrf/users/FAQ_files/FAQ_wrf_intallation.html
- Before recompiling:
  - issue a 'clean –a'
  - If you need to make changes to the configure.wrf file, do this after the 'clean –a', and then save the edited file.
  - Recompile
- Contact wrfhelp@ucar.edu

28

Installing Steps

- Check system requirements
- Installing libraries
- Download source data
- Compile WRFV3
- *Compile WPS*
- Download initial/BC datasets

Step 1: Configure for WPS

- Inside the WPS/ directory, type: **`./configure`**

```
Will use NETCDF in dir: /glade/apps/opt/netcdf/4.3.0/intel/12.1.5
$JASPERLIB or $JASPERINC not found in environment. Using default values for library paths...
-----------------------------------------------------------------------
Please select from among the following supported platforms.

   1.  Linux x86_64, gfortran     (serial)
   2.  Linux x86_64, gfortran     (serial_NO_GRIB2)
   3.  Linux x86_64, gfortran     (dmpar)
   4.  Linux x86_64, gfortran     (dmpar_NO_GRIB2)
   5.  Linux x86_64, PGI compiler   (serial)
   6.  Linux x86_64, PGI compiler   (serial_NO_GRIB2)
   7.  Linux x86_64, PGI compiler   (dmpar)
   8.  Linux x86_64, PGI compiler   (dmpar_NO_GRIB2)
```

- Choose to compile WPS **serially**, even if you compile WRFV3 in parallel (unless you have a very large domain)
  **NOTE: if you do compile WPS in parallel, ungrib.exe must run serially
- Output from configuration: a file called 'configure.wps'

Step 2: Compile WPS

- In the WPS/ directory, type:
  `./compile >& log.compile`
- Compilation should only take a few minutes
- If successful, these executables should be in your WPS/ directory (and they are linked, respectively, from their source code directories):

```
geogrid.exe -> geogrid/src/geogrid.exe
ungrib.exe -> ungrib/src/ungrib.exe
metgrid.exe -> metgrid/src/metgrid.exe
```

Unsuccessful WPS Compilation

No geogrid.exe or metgrid.exe

- WPS makes use of the external I/O libraries in the *WRFV3/external* directory - The libraries are built when WRF is installed
- Check that you used the exact same compiler (and version) as you used to compile WRFV3
- Check that you are using the same netCDF that you used to build WRFV3
- Have you changed the name or path of the WRFV3/ directory?
  - If so, you need to change the following line in the configure.wps file:
    WRF_DIR = ../WRFV3
  - Save the file and recompile

Unsuccessful WPS Compilation

No ungrib.exe

- Make sure you have installed your jasper, zlib, and libpng libraries correctly.
- Make sure that you are using the correct path and format for the following lines in the configure.wps file

```
COMPRESSION_LIBS = -L/${DIR}/UNGRIB_LIBRARIES/lib –ljasper –lpng –lz
COMPRESSION_INC = -I/${DIR}/UNGRIB_LIBRARIES/include
```

Save configure.wps and recompile

./clean -a

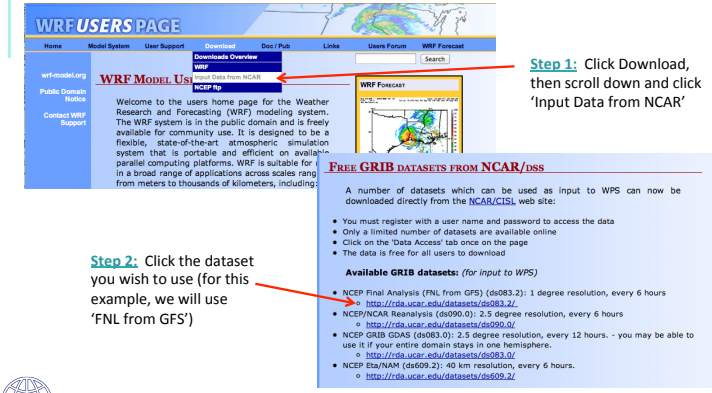
- The './clean –a' command is something that should be used when you have made corrections to your configure.wrf file, configure.wps file, or any changes to the registry. If you have made any of these changes, or if you plan to recompile your code from scratch, you must issue a 'clean –a' before recompiling.
- If you made any changes to any subroutines within the code, you will need to recompile your code, but you do NOT need to issue the 'clean –a' command, nor do you need to reconfigure. You will simply just recompile. This compilation should take a lot less time than a clean compile.

Installing Steps

- Check system requirements
- Installing libraries
- Download source data
- Compile WRFV3
- Compile WPS
- *Download initial/BC datasets*

Download Datasets

- From the WRF Users' page: http://www2.mmm.ucar.edu/wrf/users/



Step 1: Click Download, then scroll down and click 'Input Data from NCAR'

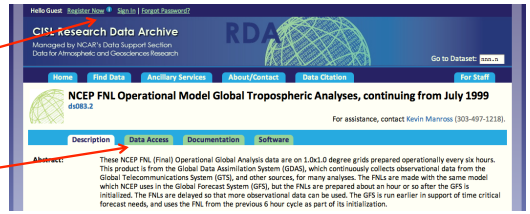
Step 2: Click the dataset you wish to use (for this example, we will use 'FNL from GFS')

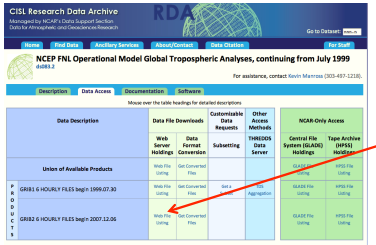
*Note: NOMADS site has several types of useful data: http://nomads.ncdc.noaa.gov

Download Datasets (continued)

Step 3: Register, or sign in, if you already have an account

Step 4: Click 'Data Access'

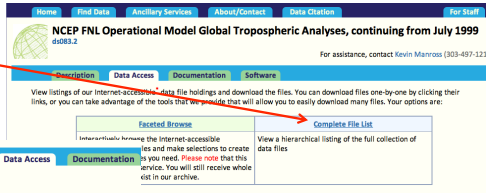


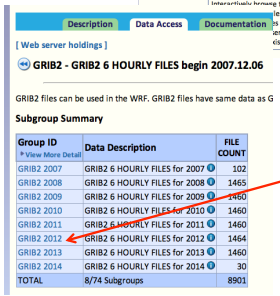


Step 5: Click 'Web File Listing' for the span of years you need

Download Datasets (continued)

Step 6: Click 'Complete File List'



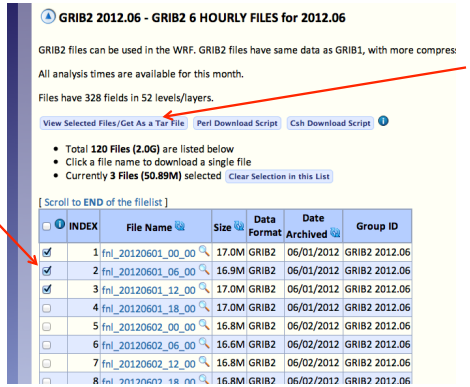


Group ID	Data Description	FILE COUNT
GRIB2 2007	GRIB2 6 HOURLY FILES for 2007	102
GRIB2 2008	GRIB2 6 HOURLY FILES for 2008	1465
GRIB2 2009	GRIB2 6 HOURLY FILES for 2009	1460
GRIB2 2010	GRIB2 6 HOURLY FILES for 2010	1460
GRIB2 2011	GRIB2 6 HOURLY FILES for 2011	1460
GRIB2 2012	GRIB2 6 HOURLY FILES for 2012	1464
GRIB2 2013	GRIB2 6 HOURLY FILES for 2013	1460
GRIB2 2014	GRIB2 6 HOURLY FILES for 2014	30
TOTAL	8/74 Subgroups	8901

Step 7: Click the year you need. After this, You will click the month you need (not shown)

Download Datasets (continued)

Step 8: Click a box for each time span that you need



INDEX	File Name	Size	Data Format	Date Archived	Group ID
1	fnl_20120601_00_00	17.0M	GRIB2	06/01/2012	GRIB2 2012.06
2	fnl_20120601_06_00	16.9M	GRIB2	06/01/2012	GRIB2 2012.06
3	fnl_20120601_12_00	17.0M	GRIB2	06/01/2012	GRIB2 2012.06
4	fnl_20120601_18_00	17.0M	GRIB2	06/01/2012	GRIB2 2012.06
5	fnl_20120602_00_00	16.8M	GRIB2	06/02/2012	GRIB2 2012.06
6	fnl_20120602_06_00	16.6M	GRIB2	06/02/2012	GRIB2 2012.06
7	fnl_20120602_12_00	16.8M	GRIB2	06/02/2012	GRIB2 2012.06
8	fnl_20120602_18_00	16.8M	GRIB2	06/02/2012	GRIB2 2012.06

Step 9: Once you have chosen All your times, click on the 'View Selected Files/Get As a Tar File' button
To download one tar file with all your Dates/times

Questions?

WPS:
Set-up and Run

Michael Duda

Running the WRF Preprocessing System

Michael Duda

Overview

- How to run through the WPS for basic cases
  - Basic steps for running the WPS
    - Geogrid
    - Ungrib
    - Metgrid
- WPS utility programs
- Common WPS mistakes

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,
/
```

```
&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/'
/
```


Running geogrid

STEP 1: Edit `namelist.wps`

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

Which WRF core?
For ARW, set to 'ARW'
For NMM, set to 'NMM'

Total number of model domains, including nests, for ARW.

Format for geogrid output files; 2=netCDF is recommended.

See p. 3-8 and 3-37

Running geogrid

STEP 1: Edit `namelist.wps`

```
&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,
   dx                    = 15000,
   dy                    = 15000,

   geog_data_res         = '5m',  '2m',

    . . .
/
```

Nesting: Who is the parent? What is the grid ratio for each nest? Where is it located in its parent?

Domain sizes: How many grid points does the domain have? What is the grid spacing?

Static data: What resolution of source data to interpolate from for each domain?
'30s', '2m', '5m', or '10m'?

See p. 3-9, 3-19, and 3-38

Running geogrid

STEP 1: Edit `namelist.wps`

```
&geogrid
   ...

   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/'
/
```

Map projection: What projection to use? What are the parameters of the projection?

See p. 3-9 and 3-40

Static data: Where are the data directories (e.g., topo_30s) located?

See p. 3-41

Running geogrid

STEP 2: 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

  …

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

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

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

Running geogrid

STEP 3: Check that geogrid ran successfully

If geogrid ran sucessfully, this message should be printed:

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

If there was an error, check for an ERROR or WARNING message in the `geogrid.log` file, or for a system error, like "Segmentation fault".

Choosing Static Datasets

WPS v3.7 supports several land cover datasets, and the next release (3.8?) will support two different topography datasets

Land use:

- USGS 24-class, 30-arc-second resolution
- USGS 24-class + inland water, 30-arc-second resolution
- MODIS 20-class, 30- and 15-arc-second resolution
- MODIS 20-class + inland water, 30-arc-second resolution
- NLCD 2011 40-class, 9-arc-second resolution

Terrain:

- GTOPO30
- GMTED2010 (will be available in WPS v3.8)

Choosing Static Datasets

Selection of alternate static datasets is performed using the `geog_data_res` namelist option in the &geogrid record

Prefix the usual `geog_data_res` selection with the name for the land use or topography dataset to be used.

E.g.,

```
geog_data_res = 'nlcd2011_9s+30s'
```

to use NLCD 2011 9-arc-second land cover, and 30-arc-second resolution for other static fields.

Global Land Cover Datasets

Consider an example 1-km domain centered over Beijing:

USGS 30-arc-second resolution, from ~1993 data; the USGS data are used by default

MODIS 30-arc-second resolution, from 2001(?) data; select using `'modis_30s'`

MODIS 15-arc-second resolution, most prevalent category between 2001 and 2010; select using `'modis_15s'`

Identifying Inland Water Bodies

Two land cover datasets also provide a special category to identify "inland water bodies", which can sometimes require special treatment, e.g., when initializing SST field or running the lake model in WRF.

MODIS 30-arc-second:

- Selected using `'modis_lakes'`

USGS 30-arc-second:

- Selected using `'usgs_lakes'`

We'll discuss the use of lake categories for initializing the SST field in the "WPS Advanced Features" talk on Wednesday.

A domain over Scandinavia using MODIS 21-class land cover; lake category shown in dark blue.

NLCD Land Use (Continental U.S. Only)

For the WRF domains over the Continental U.S., one can use high-resolution land cover from the National Land Cover Database (NLCD).

NLCD 2011 9-arc-second:

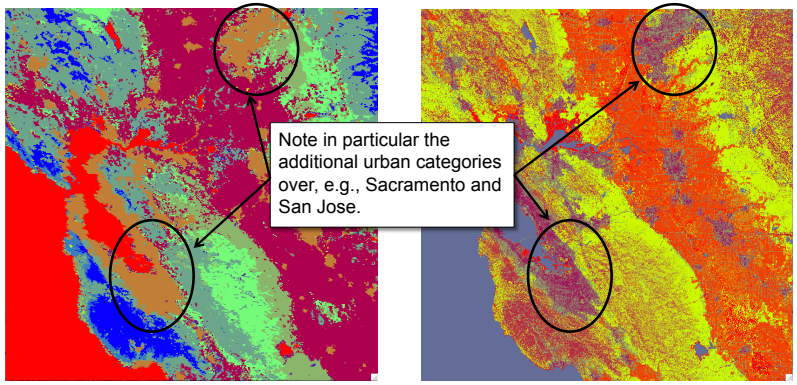
- Selected using `'nlcd2011_9s'`

Besides high spatial resolution, the NLCD data provides four new urban categories:

1. Developed Open Space
2. Developed Low Intensity
3. Developed Medium Intensity
4. Developed High Intensity

NLCD Land Use (Continental U.S. Only)

For the WRF domains over the Continental U.S., one can use high-resolution land cover from the National Land Cover Database (NLCD).



Above: (left) A 250-m WRF domain covering San Francisco Bay using MODIS 15-arc-second land cover data; (right) the same domain using NLCD 2011 9-arc-second data.

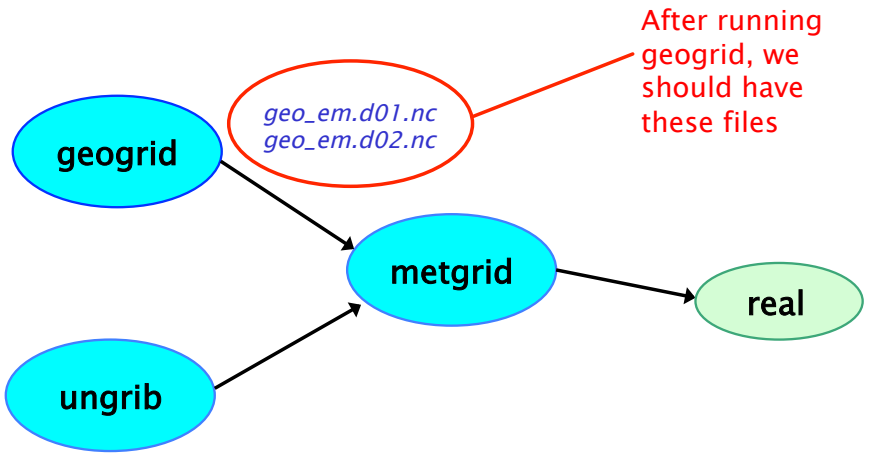
GMTED2010 Terrain (coming in v3.8)

In the next release of WRF, we intend to supply a newer, more accurate terrain dataset from the USGS: GMTED2010*.

Left: Terrain elevation difference in meters (GMTED2010 minus GTOPO30). Note that the scale does not cover the full range of the differences.

*https://lta.cr.usgs.gov/GMTED2010

Running geogrid



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,
/
```

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


Running ungrib

STEP 1: Edit `namelist.wps`

```
&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 time range: Between which times should ungrib process GRIB data?

Data frequency: How many seconds between output files for ungrib? E.g., 10800 s = 3 hrs

See p. 3-14, and 3-38

Running ungrib

STEP 1: Edit `namelist.wps`

```
&ungrib
   out_format = 'WPS',

   prefix = 'GFS',
/
```

Intermediate file format: Which format to use for intermediate files? 'WPS', 'SI', or 'MM5' are possible; 'WPS' is recommended.

Intermediate file names: Gives prefix for intermediate files. Prefix can include a path. E.g., 'XZY' would give intermediate files named XYZ:*yyyy-mm-dd_hh.*

See p. 3-14, 3-23, and 3-41

Running ungrib

STEP 2: Link the correct Vtable to the file name "Vtable" in the run directory

- Some Vtables are provided with WPS in the **`WPS/ungrib/Variable_Tables`** directory
  - E.g., Vtable.GFS, Vtable.SST, Vtable.ECMWF

See p. 3-15

- Ungrib always expects to find a file named **`Vtable`** in the run directory

```
> ln -s ungrib/Variable_Tables/Vtable.GFS Vtable
> ls Vtable
Vtable -> ungrib/Variable_Tables/Vtable.GFS
```

Running ungrib

STEP 3: Link GRIB files to the correct file names in the run directory

- Ungrib always expects GRIB files to be named `GRIBFILE.AAA`, `GRIBFILE.AAB`, `GRIBFILE.AAC`, etc., in the run directory
- The `link_grib.csh` script can be used to link GRIB files to these file names:

```
> link_grib.csh /data/GRIB/GFS/gfs*
> ls GRIBFILE.*
GRIBFILE.AAA -> /data/GRIB/GFS/gfs_060401_00_00
```

See p. 3-15

Running ungrib

STEP 4: Run 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
```


Running ungrib

STEP 5: Check that ungrib ran successfully

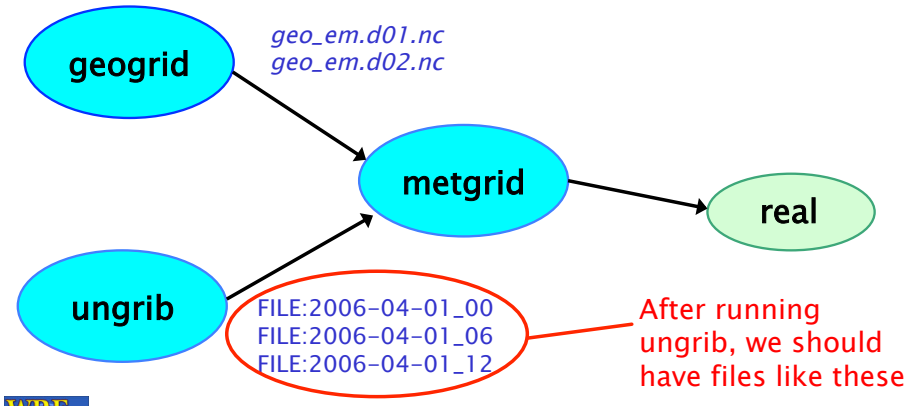
If ungrib ran successfully, this message should be printed:

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

If there was an error, check for error message in ungrib's printout or in the `ungrid.log` file.

Common errors are related to incorrect date specifications in the `&share` namelist, or because GRIB2 data was used with a version of WPS compiled without GRIB2 libraries.

Running ungrib



Running metgrid

STEP 1: Edit `namelist.wps`

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


Running metgrid

STEP 1: Edit `namelist.wps`

```
&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 range: Time range to process *for each domain*. Usually, only the initial time is needed for ARW nested domains.

See p. 3-17 and 3-37

Running metgrid

STEP 1: Edit `namelist.wps`

Intermediate file prefixes: Prefix (or prefixes) of intermediate files to interpolate to model domain. Should match prefix given to ungrib.

See p. 3-17 and 3-24

```
&metgrid
   fg_name = 'GFS',

   constants_name = 'SST:2006-04-01_00',

   io_form_metgrid = 2,
/
```

Constant fields: Optional name of an intermediate file with fields to be used for every time period.

Metgrid I/O format: Which I/O format to use for metgrid output? 2=netCDF is recommended.

See p. 3-17, and 3-41

Running metgrid

STEP 2: Run 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 constant files (given using `constants_name`) are processed before any time varying fields.

Metgrid processes all time period for one domain before processing for the next domain

Running metgrid

STEP 3: Check that metgrid ran successfully

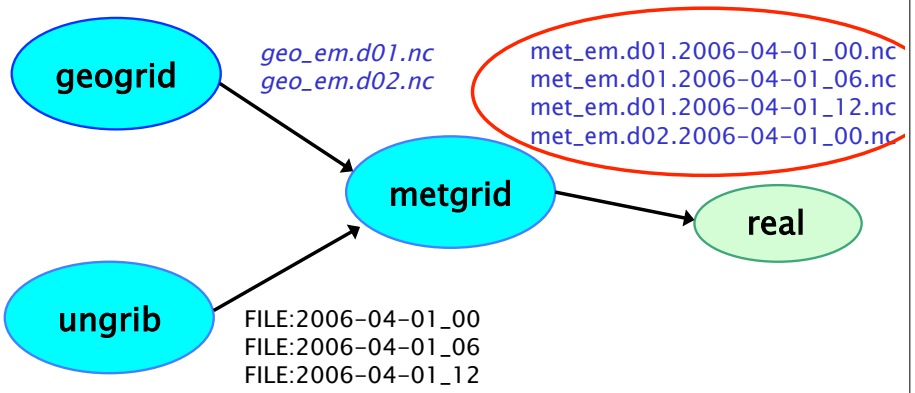
If metgrid ran successfully, this message should be printed:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!  Successful completion of metgrid.          !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

If there was an error, check for an ERROR or WARNING message in the `metgrid.log` file, or for a system error, like "Segmentation fault".

Running metgrid

After running metgrid, we should have files similar to these



Overview

- How to run through the WPS for basic cases
  - Basic steps for running WPS
    - Geogrid
    - Ungrib
    - Metgrid
- WPS utility programs
- Common WPS mistakes

WPS Utility Programs

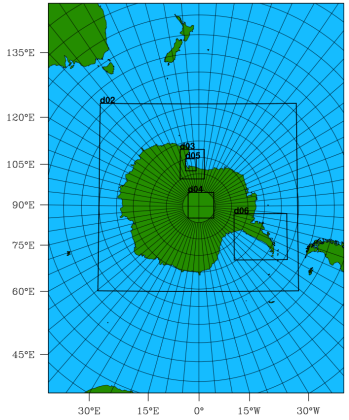
- Besides geogrid, ungrib, and metgrid, some simple utility programs are distributed with WPS:
  - For checking contents of intermediate format files
  - For listing contents of GRIB1 & GRIB2 files
  - To assist in locating domains
  - For computing 3d pressure field for ECMWF data
- Some programs use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

See p. 3-27

Utility: plotgrids.ncl

The *plotgrids.ncl* script plots the locations of grids defined in *namelist.wps*

- *plotgrids can be used to iteratively refine the locations of grids.*
- *plotgrids.ncl* uses the namelist.wps file only, so there is no need to run geogrid first!



Utility: rd_intermediate

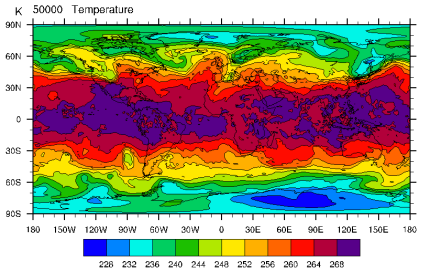
The rd_intermediate lists information about the fields found in an intermediate-format file

```
================================================
FIELD = TT
UNITS = K DESCRIPTION = Temperature
DATE = 2000-01-24_12:00:00 FCST = 0.000000
SOURCE = unknown model from NCEP GRID 212
LEVEL = 200100.000000
I,J DIMS = 185, 129
IPROJ = 1
  REF_X, REF_Y = 1.000000, 1.000000
  REF_LAT, REF_LON = 12.190000, -133.459000
  DX, DY = 40.635250, 40.635250
  TRUELAT1 = 25.000002
DATA(1,1)=295.910950
================================================
```

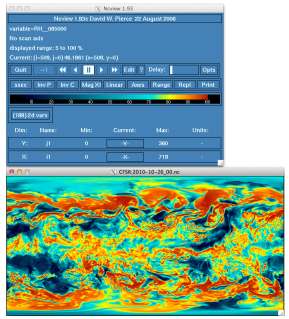

Utility: int2nc + plotfmt_nc.ncl

The int2nc program converts an ungrib intermediate file to a standard NetCDF file

- *Users may then visualize fields with ncview, NCL, or other graphical packages:*



Visualize NetCDF intermediate fields using plotfmt_nc.ncl script



Visualize NetCDF intermediate fields using ncview

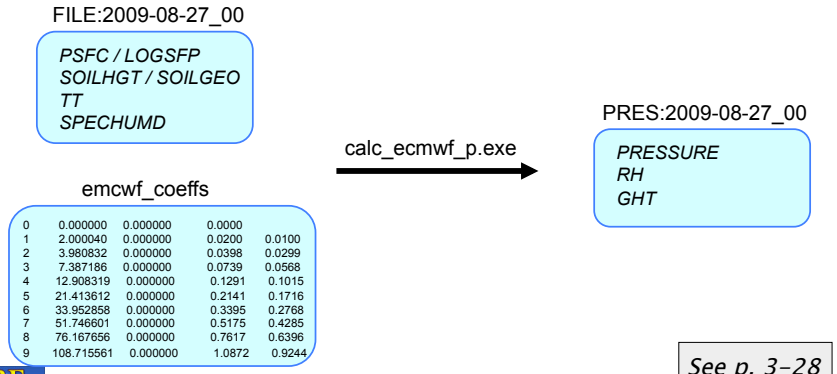
Utility: g1print and g2print

The *g1print* and *g2print* programs list the contents of a GRIB1 or GRIB2 file:

rec num	Prod Disc	Cat	Param num	Lvl code	Lvl one	Lvl two	Name	Time	Fcst hour
1	0	3	5	100	100000	0	HGT	2006-08-16_12:00:00	00
2	0	3	5	100	97500	0	HGT	2006-08-16_12:00:00	00
3	0	3	5	100	95000	0	HGT	2006-08-16_12:00:00	00
4	0	3	5	100	92500	0	HGT	2006-08-16_12:00:00	00
5	0	3	5	100	90000	0	HGT	2006-08-16_12:00:00	00
6	0	3	5	100	85000	0	HGT	2006-08-16_12:00:00	00
7	0	3	5	100	80000	0	HGT	2006-08-16_12:00:00	00
8	0	3	5	100	75000	0	HGT	2006-08-16_12:00:00	00
9	0	3	5	100	70000	0	HGT	2006-08-16_12:00:00	00
10	0	3	5	100	65000	0	HGT	2006-08-16_12:00:00	00

Utility: calc_ecmwf_p

The *calc_ecmwf_p* utility creates intermediate files with a pressure (and possibly GHT and RH) field



See p. 3-28

Overview

- How to run through the WPS for basic cases
  - Basic steps for running WPS
    - Geogrid
    - Ungrib
    - Metgrid
- WPS utility programs
- Common WPS mistakes

Common WPS Mistakes

1) All 3-d fields must have same number of levels in metgrid

```
WRF_DEBUG: Warning DIM              4 , NAME
 num_metgrid_levels REDIFINED  by var GHT            27
 26  in wrf_io.F90 line          2347
 ERROR: Error in ext_pkg_write_field
```

- This is usually corrected by ensuring that all 3-d meteorological fields have surface level data
- Try setting debug_level=1000 in &share namelist, and checking metgrid.log for a table showing which fields are available at each level

Common WPS Mistakes

2) When using a regional data set (e.g., NAM), ensure that model domain is completely covered by the data

- The metgrid program will stop if the model domain has grid points that are not covered by data

3) For native vertical coordinate data sets (e.g., RUCb, ECMWF), ensure that both pressure and geopotential height fields are available

Questions?

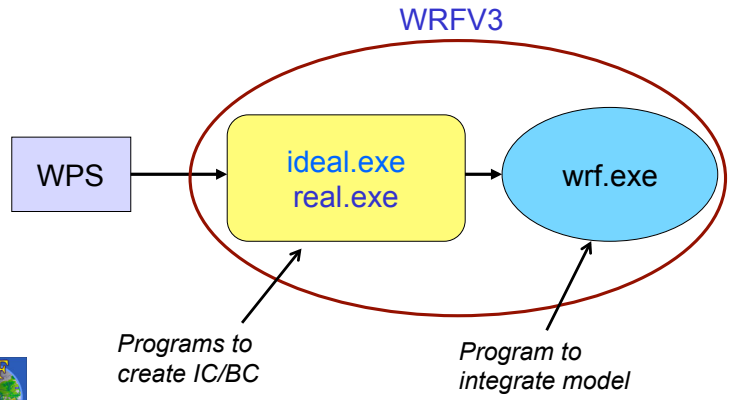
WRF:
Set-up and Run

Wei Wang

Set Up and Run WRF

Wei Wang
Nansha, Guangdong, China
December 2015

WRF System Flowchart



Outline

- Running WRF code
  - Things to check before you run..
  - Running real-data case
  - Running idealized case
- Basic runtime options (*namelist*)
- Check output
- Simple trouble shooting

Before You Run ..

- Make sure appropriate executables are created in **`WRFV3/main/`** directory (select 'basic nest' at compile time)
  - **`ideal.exe`** – *executable to create idealized IC*
  - **`real.exe`** – *executable to create IC/BC*
  - **`wrf.exe`** – *executable for model integration*
  - **`ndown.exe`** – *utility program for offline one-way nest*
  - **`tc.exe`** – *utility routine for TC bogusing*
- If you are working with real data, be sure that files for ***a few time periods*** for domain 1, and nest files for model start time from WPS are correctly generated:
  - **`met_em.d01.*`** (multiple time periods)
  - **`met_em.d0*`** (model start time)

WRF test case directories

You have these choices in **`WRFV3/test/`**
(made at compile time):

- **`em_real`** } *3-dimensional real-data – real.exe*
- **`em_quarter_ss`**
- **`em_b_wave`**
- **`em_les`**
- **`em_tropical_cyclone`** } *3d ideal*
- **`em_heldsuarez`**
- **`em_hill2d_x`**
- **`em_squall2d_x`**
- **`em_squall2d_y`**
- **`em_grav2d_x`** } *2d ideal*
- **`em_seabreeze2d_x`**
- **`em_scm_xy`** } *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 and times of the case
4. Run a initialization program (*ideal.exe* or *real.exe*)
5. Run model executable, *wrf.exe*

WRFV3/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`**
- **`ETAMPNEW_DATA`**
- **`tr49t67`**
- **`tr49t85`**
- **`tr67t85`**

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

- **`gribmap.txt`** } *for grib IO*
- **`grib2map.tbl`**

… (a few more)

WRFV3/run directory after compile

`LANDUSE.TBL`
`SOILPARM.TBL`
`VEGPARM.TBL`
`GENPARM.TBL`
`URBPARM.TBL`
`RRTM_DATA`
`RRTMG_SW_DATA`
`RRTMG_LW_DATA`
`ETAMPNEW_DATA`
`tr49t67`
`tr49t85`
`tr67t85`
…

An example after em_real case compile

`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)

Running a Real-Data Case

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 offline one-way nesting*
  - **tc.exe** - *program for TC bogusing*
- These executables are linked to:

  WRFV3/run

  and

  WRFV3/test/*em_real*

➔ One can go to either directory to run.

WRFV3/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
ETAMPNEW_DATA -> ../../run/ETAMPNEW_DATA
tr49t67 -> ../../run/tr49t67
tr49t85 -> ../../run/tr49t85
tr67t85 -> ../../run/tr67t85
…
namelist.input          - editing required
real.exe -> ../../main/real.exe
wrf.exe -> ../../main/wrf.exe
ndown.exe -> ../../main/ndown.exe
```

…. (a few more)

Running a Real-data Case

- One must successfully run WPS, and create **met_em.*** file for more than one time period
- Move or link WPS output files to the run directory:

```
cd test/em_real
ln -s ../../../WPS/met_em.d0* .
```

Running a 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 a MPI job

Running a Real-data Case

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

wrfinput_d01
wrfinput_d02
…..
wrfbdy_d01

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

N-1 time-level data for lateral boundaries, and only for domain 1

N: the number of time periods processed

ncdump –v Times wrfbdy_d01

Running a Real-data Case

- Typing **'ncdump –v Times wrfbdy_d01'** will produce boundary conditions for the following times, from 6 hourly data interval in a 24 h period:

  .. 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" ;
```

Running a Real-data Case

- Run the model executable by typing:

  ./wrf.exe >& wrf.out &

  or

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

Running a Real-data Case

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

  wrfout_d01_2005-08-28_00:00:00

  Based on start date set in namelist

  wrfout_d02_2005-08-28_00:00:00

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

wrfrst_d01_2005-08-28_12:00:00

wrfrst_d02_2005-08-28_12:00:00

Exact time at 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

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

  ideal.exe - ideal case initialization program

  wrf.exe - model executable

- These executables are linked to:

  WRFV3/run

  and

  WRFV3/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 physics data files to the currect directory:

```
./run_me_first.csh
```

Running an *Idealized* Case

Then run the ideal initialization program:

```
./ideal.exe
```

The input to this program is typically a sounding file (file named ***input_sounding***), or a pre-defined 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 conditions are set in the **namelist.input** file. For example, these are for options in east-west, or x direction:

```
periodic_x     = .false.,.false.,.false.,
symmetric_xs   = .false.,.false.,.false.,
symmetric_xe   = .false.,.false.,.false.,
open_xs        = .true., .false.,.false.,
open_xe        = .true., .false.,.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 (where ***N*** is the number of processors requested)
- Successful running of the model executable will create a model history file called **wrfout_d01_<date>**

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

  Based on start date set in namelist

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 in talk 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?

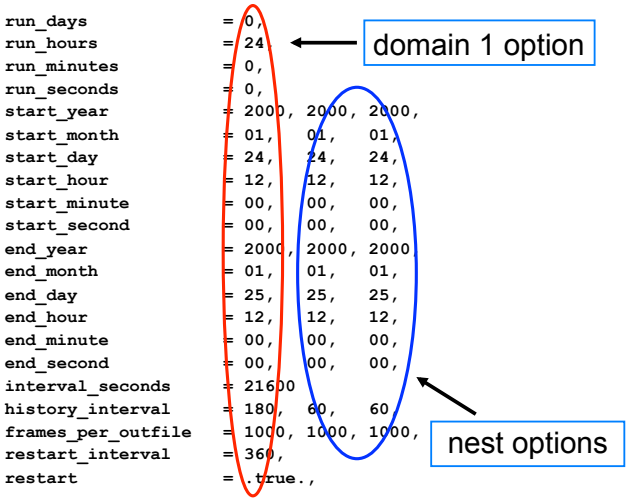
- A Fortran namelist contains a list of *runtime* options for the code to read in during its execution. Use of a namelist allows one to change runtime configuration without the need to recompile the source code.
- Fortran 90 namelist has very specific format, so edit with care:

  `&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



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
  - 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 in namelist

&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,     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,
```

Notes on &domains

- *time_step, time_step_fract_num, time_step_frac_den*:
  - Time step for model integration in seconds.
  - Fractional time step specified in separate integers of numerator and denominator.
  - 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.
- *num_metgrid_levels*:
  - Number of *metgrid* (input) data levels.
- *num_metgrid_soil_levels*:
  - Number of soil data levels in the input data

  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 Ptop)
- *eta_levels*:
  - Specify your own model levels from 1.0 to 0.0.
  - If not specified, program *real* will calculate a set of levels
  - Use a minimum of 30 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*

```
spec_bdy_width      = 5, (10)
spec_zone           = 1, (1)
relax_zone          = 4, (9)
specified           = .true., .false.,.false.,
nested              = .false., .true., .true.,
```

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

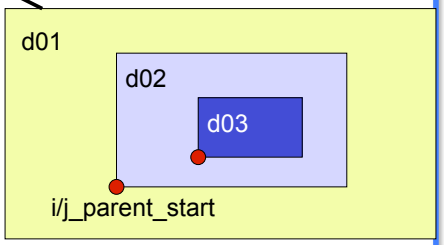
namelist record &domains

```
max_dom   = 3,
e_we      = 74, 112,94,
e_sn      = 61, 97, 91,
e_vert    = 30, 30, 30,
grid_id         = 1,  2,  3,
parent_id       = 0,  1,  2,
i_parent_start  = 0, 31, 30,
j_parent_start  = 0, 17, 30,
```

Activate nests: no. of domains to run

Dimensions of all domains; same as in WPS.

Make sure the nest domain parameters match those set in WPS



&domains

```
dx = 30000., 10000., 3333.33,
dy = 30000., 10000., 3333.33,
parent_grid_ratio = 1, 3, 3,
parent_time_step_ratio = 1,3,3,
```

For fractional grid distance, use at least 2 decimal places

All 4 variables must be specified. *Grid ratio* can be any integer (3 and 5 are recommended), and *time step ratio* can be different from grid ratio. Grid distance is in meters, even for lat/lon map projection.

Time steps for nested domains are specified by coarse domain time step and parent_time_step_ratio

&domains

```
feedback      = 1,
smooth_option = 2,
```

When feedback is on, this option can be selected to smooth the area in the parent domain where the nest is. Valid values are 0,1,2.

Whether a nest will overwrite parent domain results. Setting feedback=0 ➔ 'one-way' nesting in a concurrent run.

Other notes on namelists

- Use same physics options for all domains.
  - An exception is cumulus scheme. One may need to turn it off for a nest that has grid distance of a few kilometers.
- Also use same physics calling frequency (e.g. **`radt, cudt,`** etc.) in all domains.

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 p5-25 to 5-27 of the ARW User's Guide:
  - 2 or 4 km convection-permitting runs
  - 20 – 30 km, 2 – 3 day runs
  - Antarctic region
  - Tropical storm forecasting
  - Regional climate

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 and registry.io_boilerplate (for IO options) (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 **rsl**.* files
- Model history file(s):

  wrfout_d01_<date>

  wrfout_d02_<date>…
- Model restart file(s), optional

  wrfrst_d0*_<date>

Output from a multi-processor run

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

mpirun -np 4 ./wrf.exe ➔

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

There is one pair of files for each processor requested

What to Look for in a standard out File?

Check run log file by typing

tail wrf.out, or

tail rsl.out.0000

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

wrf: SUCCESS COMPLETE WRF

How to Check Model History File?

- Use **ncdump**:

  ncdump -v Times wrfout_d01_<date>

  to check output times. Or

  ncdump -v U wrfout_d01_<date>

  to check a particular variable (U)

- Use **ncview** (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
```

- Time taken to compute one model step:

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

- Time taken to write history and restart file:

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

- Any model error prints:

```
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

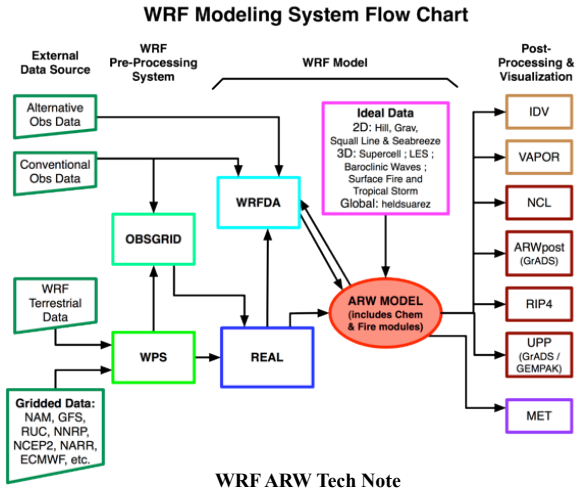
- Information on compiling and running WRF, and a more extensive list of namelist options and their definition / explanations can be found in the User's Guide, Chapter 5
- Also see '*Nesting Setup and Run*' in your slide book and *'Other Runtime Options'* talk.

WRF Dynamics and Numerics

Bill Skamarock

The Advanced Research WRF (ARW) Dynamics Solver

1. Terrain, vertical coordinate
2. Equations and variables
3. Time integration scheme
4. Grid staggering
5. Advection (transport) and conservation
6. Time step parameters
7. Filters
8. Map projections and global configuration



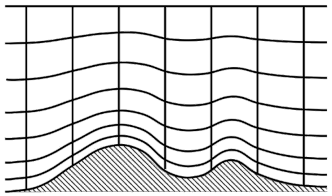
WRF ARW Tech Note
A Description of the Advanced Research WRF Version 3
http://www.mmm.ucar.edu/wrf/users/pub-doc.html

Hydrostatic pressure π

Column mass (per unit area) $\mu = \pi_s - \pi_t$

Vertical coordinate $\eta = \dfrac{(\pi - \pi_t)}{\mu}$

Layer mass (per unit area) $\mu \Delta\eta = \Delta\pi = g\rho\Delta z$



Conserved state (prognostic) variables:

$$\mu, \quad U = \mu u, \quad V = \mu v, \quad W = \mu w, \quad \Theta = \mu\theta$$

Non-conserved state variable: $\phi = gz$

Moist Equations:

$$\frac{\partial U}{\partial t} + \alpha\mu_d \frac{\partial p}{\partial x} + \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\frac{\partial \phi}{\partial x} = -\frac{\partial Uu}{\partial x} - \frac{\partial \Omega u}{\partial \eta}$$

$$\frac{\partial W}{\partial t} + g\left(\mu_d - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\right) = -\frac{\partial Uw}{\partial x} - \frac{\partial \Omega w}{\partial \eta}$$

$$\frac{\partial \mu_d}{\partial t} + \frac{\partial U}{\partial x} + \frac{\partial \Omega}{\partial \eta} = 0$$

$$\frac{\partial \Theta}{\partial t} + \frac{\partial U\theta}{\partial x} + \frac{\partial \Omega\theta}{\partial \eta} = \mu Q$$

$$\frac{d\phi}{dt} = gw$$

$$\frac{\partial(\mu_d q_{v,l})}{\partial t} + \frac{\partial(U q_{v,l})}{\partial x} + \frac{\partial(\Omega q_{v,l})}{\partial \eta} = \mu Q_{v,l}$$

Diagnostic relations: $\dfrac{\partial \phi}{\partial \eta} = -\alpha_d \mu_d, p = \left(\dfrac{R_d \Theta_m}{p_o \mu_d \alpha_d}\right)^{\gamma}, \ \Theta_m = \Theta\left(1 + \dfrac{R_v}{R_d} q_v\right)$

Dynamics: 2. Equations and variables – velocities

Vertical momentum eqn. $$\frac{\partial W}{\partial t} + g\left(\mu_d - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\right) = -\frac{\partial Uw}{\partial x} - \frac{\partial \Omega w}{\partial \eta}$$

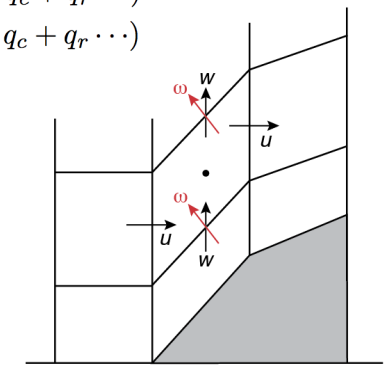
Subscript *d* denotes *dry*, and

$$\alpha_d = \frac{1}{\rho_d} \qquad \alpha = \alpha_d\,(1 + q_v + q_c + q_r \cdots)^{-1}$$
$$\rho = \rho_d\,(1 + q_v + q_c + q_r \cdots)$$

covariant (*u, ω*) and
contravariant *w* velocities

$$u = \frac{dx}{dt}, \quad w = \frac{dz}{dt}, \quad \omega = \frac{d\eta}{dt}$$

$$U = \mu u, \quad W\mu w, \quad \Omega = \mu\omega$$



Dynamics: 3. Time integration scheme

3rd Order Runge-Kutta time integration

advance $\phi^t \rightarrow \phi^{t+\Delta t}$

$$\phi^* = \phi^t + \frac{\Delta t}{3} R\left(\phi^t\right)$$
$$\phi^{**} = \phi^t + \frac{\Delta t}{2} R\left(\phi^*\right)$$
$$\phi^{t+\Delta t} = \phi^t + \Delta t\, R\left(\phi^{**}\right)$$

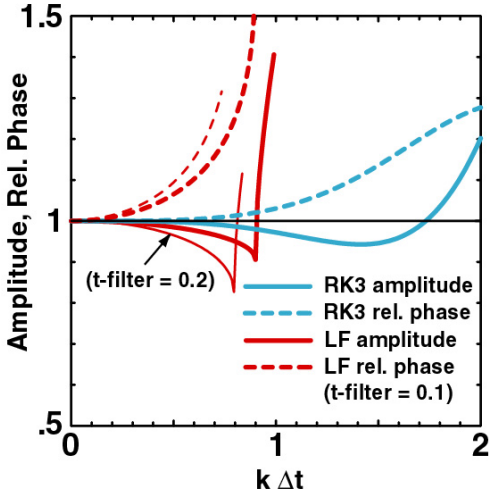
Amplification factor $\phi_t = ik\phi; \quad \phi^{n+1} = A\phi^n; \quad |A| = 1 - \frac{(k\Delta t)^4}{24}$

Dynamics: 3. Time integration scheme

Phase and amplitude errors for LF, RK3

Oscillation equation analysis

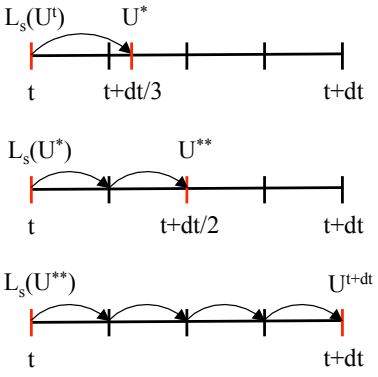
$$\psi_t = ik\psi$$



Dynamics: 3. Time integration scheme – time splitting

$$U_t = L_{fast}(U) + L_{slow}(U)$$

3rd order Runge-Kutta, 3 steps



- RK3 is 3rd order accurate for linear eqns, 2nd order accurate for nonlinear eqns.
- Stable for centered and upwind advection schemes.
- Stable for Courant number Udt/dx < 1.73
- Three $L_{slow}(U)$ evaluations per timestep.

Dynamics: 3. Time integration scheme - implementation

Begin time step

- Runge-Kutta loop (steps 1, 2, and 3)
  - (i) advection, p-grad, buoyancy using (ϕ^t, ϕ^*, ϕ^{**})
  - (ii) physics if step 1, save for steps 2 and 3
  - (iii) mixing, other non-RK dynamics, save…
  - (iv) assemble dynamics tendencies
  - Acoustic step loop
    - (i) advance U,V, then μ, Θ, then w, ϕ
    - (ii) time-average U,V, Ω
  - End acoustic loop
  - Advance scalars using time-averaged U,V, Ω
- End Runge-Kutta loop
- Adjustment physics (currently microphysics)

End time step

Dynamics: 3. Time integration scheme – perturbation variables

Introduce the perturbation variables:

$$\phi = \bar{\phi}(\bar{z}) + \phi', \mu = \bar{\mu}(\bar{z}) + \mu';$$
$$p = \bar{p}(\bar{z}) + p', \alpha = \bar{\alpha}(\bar{z}) + \alpha'$$

Note – $\phi = \bar{\phi}(\bar{z}) = \bar{\phi}(x, y, \eta)$,

likewise $\bar{p}(x, y, \eta), \bar{\alpha}(x, y, \eta)$

Reduces horizontal pressure-gradient errors.

For small time steps, recast variables as perturbations from time t

$$U' = U'^t + U'', \; V' = V'^t + V'', \; W' = W'^t + W'',$$
$$\Theta' = \Theta'^t + \Theta'', \; \mu' = \mu'^t + \mu'', \; \phi' = \phi'^t + \phi'';$$
$$p' = p'^t + p'', \; \alpha' = \alpha'^t + \alpha''$$

Allows vertical pressure gradient to be expressed in terms of ϕ".

Dynamics: 3. Time integration scheme – acoustic step

(Without expanding variables into perturbation form)

$$U^{\tau+\Delta\tau} \qquad \frac{\partial U}{\partial t} + \left(\mu_d \alpha \frac{\partial p}{\partial x} + \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\frac{\partial \phi}{\partial x}\right)^{\tau} = R_U^t$$

$$\mu_d^{\tau+\Delta\tau} \quad \Omega^{\tau+\Delta\tau} \qquad \frac{\partial \mu_d}{\partial t} + \frac{\partial U}{\partial x}^{\tau+\Delta\tau} + \frac{\partial \Omega}{\partial \eta}^{\tau+\Delta\tau} = 0$$

$$\Theta^{\tau+\Delta\tau} \qquad \frac{\partial \Theta}{\partial t} + \left(\frac{\partial U\theta^t}{\partial x} + \frac{\partial \Omega\theta^t}{\partial \eta}\right)^{\tau+\Delta\tau} = R_\Theta^t$$

$$W^{\tau+\Delta\tau} \qquad \frac{\partial W}{\partial t} + g\overline{\left(\mu_d - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\right)}^{\tau} = R_W^t$$

$$\phi^{\tau+\Delta\tau} \qquad \mu_d^t \frac{\partial \phi}{\partial t} + U^{\tau+\Delta\tau}\frac{\partial \phi}{\partial x}^t + \Omega^{\tau+\Delta\tau}\frac{\partial \phi}{\partial \eta}^t - g\overline{W}^{\tau} = R_\phi^t$$

- Forward-backward differencing on U, Θ, and μ equations
- Vertically implicit differencing on W and ϕ equations

Dynamics: 3. Time integration scheme – hydrostatic option

Instead of solving vertically implicit equations for W and ϕ

Integrate the hydrostatic equation to obtain p (π):

$$\frac{\partial p}{\partial \eta} = \left(\frac{\alpha_d}{\alpha}\right)^t \mu_d$$

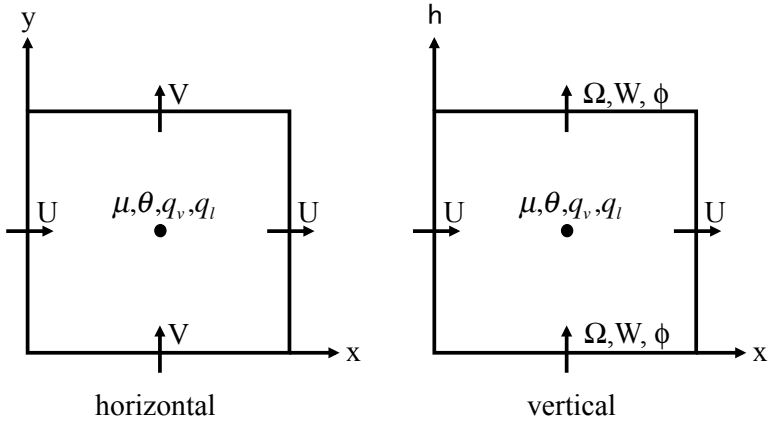
Recover α and ϕ from: $p = \left(\frac{R_d \Theta_m}{p_o \mu_d \alpha_d}\right)^{\gamma}$, $\Theta_m = \Theta\left(1 + \frac{R_v}{R_d} q_v\right)$,

and $\frac{\partial \phi}{\partial \eta} = -\mu_d \alpha_d$

W is no longer required during the integration.

Dynamics: 4. Grid staggering – horizontal and vertical

C-grid staggering



Dynamics: 5. Advection (transport) and conservation

2nd, 3rd, 4th, 5th and 6th order centered and upwind-biased schemes are available in the ARW model.

Example: 5th order scheme

$$\frac{\partial(U\psi)}{\partial x} = \frac{1}{\Delta x}\left(F_{i+\frac{1}{2}}(U\psi) - F_{i-\frac{1}{2}}(U\psi) \right)$$

where

$$F_{i-\frac{1}{2}}(U\psi) = U_{i-\frac{1}{2}}\left\{ \frac{37}{60}(\psi_i + \psi_{i-1}) - \frac{2}{15}(\psi_{i+1} + \psi_{i-2}) + \frac{1}{60}(\psi_{i+2} + \psi_{i-3}) \right\}$$

$$- sign(1,U)\,\frac{1}{60}\left\{ (\psi_{i+2} - \psi_{i-3}) - 5(\psi_{i+1} - \psi_{i-2}) + 10(\psi_i - \psi_{i-1}) \right\}$$

Dynamics: 5. Advection (transport) and conservation

For constant U, the 5th order flux divergence tendency becomes

$$\Delta t \left.\frac{\delta(U\psi)}{\Delta x}\right|_{5th} = \Delta t \left.\frac{\delta(U\psi)}{\Delta x}\right|_{6th}$$

$$- \underbrace{\left|\frac{U\Delta t}{\Delta x}\right| \frac{1}{60}\left(-\psi_{i-3} + 6\psi_{i-2} - 15\psi_{i-1} + 20\psi_i - 15\psi_{i+1} + 6\psi_{i+2} - \psi_{i+3}\right)}_{\frac{Cr}{60}\frac{\partial^6\psi}{\partial x^6} + H.O.T}$$

The odd-ordered flux divergence schemes are equivalent to the next higher ordered (even) flux-divergence scheme plus a dissipation term of the higher even order with a coefficient proportional to the Courant number.

Dynamics: 5. Advection (transport) and conservation

Maximum Courant Number for Advection

$$C_a = U\Delta t / \Delta x$$

Time Integration Scheme	*Advection Scheme*				
	2nd	*3rd*	*4th*	*5th*	*6th*
Leapfrog (g=0.1)	0.91	U	0.66	U	0.57
RK2	U	0.90	U	0.39	U
RK3	1.73	1.63	1.26	1.43	1.09

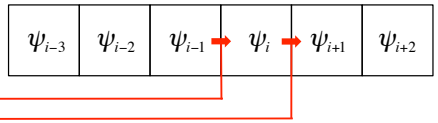
U = unstable

(Wicker & Skamarock, 2002)

Dynamics: 5. Advection (transport) and conservation – WENO scheme

Weighted Essentially Non-Oscillatory scheme

$$\psi_i^{t+\Delta t} - \psi_i^t = -\frac{\Delta t}{\Delta x}\left(flux_{i+1/2} - flux_{i-1/2}\right)$$



for $U_{i-1/2} \geq 0$

$$flux_{i-1/2} = U_{i-1/2}\left(\frac{w_0}{\overline{w}} f_0 + \frac{w_1}{\overline{w}} f_1 + \frac{w_2}{\overline{w}} f_2\right)$$

where $\overline{w} = w_0 + w_1 + w_2$

$$f_0 = \frac{1}{3}\psi_{i-3} - \frac{7}{6}\psi_{i-2} + \frac{11}{6}\psi_{i-1} \qquad w_0 = \frac{0.1}{(\varepsilon + \beta_0)^2}$$

$$f_1 = -\frac{1}{6}\psi_{i-2} + \frac{5}{6}\psi_{i-1} + \frac{1}{3}\psi_i \qquad w_1 = \frac{0.6}{(\varepsilon + \beta_1)^2}$$

$$f_2 = \frac{1}{3}\psi_{i-1} + \frac{5}{6}\psi_i - \frac{1}{6}\psi_{i+1} \qquad w_2 = \frac{0.3}{(\varepsilon + \beta_2)^2}$$

$$\beta_0 = \frac{13}{12}\left(\psi_{i-3} - 2\psi_{i-2} + \psi_{i-1}\right)^2 + \frac{1}{4}\left(\psi_{i-3} - 4\psi_{i-2} + 3\psi_{i-1}\right)^2$$

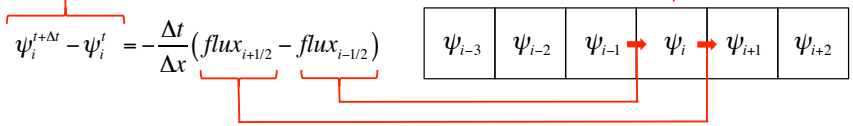
$$\beta_1 = \frac{13}{12}\left(\psi_{i-2} - 2\psi_{i-1} + \psi_i\right)^2 + \frac{1}{4}\left(\psi_{i-2} - 4\psi_{i-1} + 3\psi_i\right)^2$$

$$\beta_2 = \frac{13}{12}\left(\psi_{i-1} - 2\psi_i + \psi_{i+1}\right)^2 + \frac{1}{4}\left(\psi_{i-1} - 4\psi_i + 3\psi_{i+1}\right)^2$$

Dynamics: 5. Advection (transport) and conservation – WENO scheme

Weighted Essentially Non-Oscillatory scheme

$$\psi_i^{t+\Delta t} - \psi_i^t = -\frac{\Delta t}{\Delta x}\left(flux_{i+1/2} - flux_{i-1/2}\right)$$



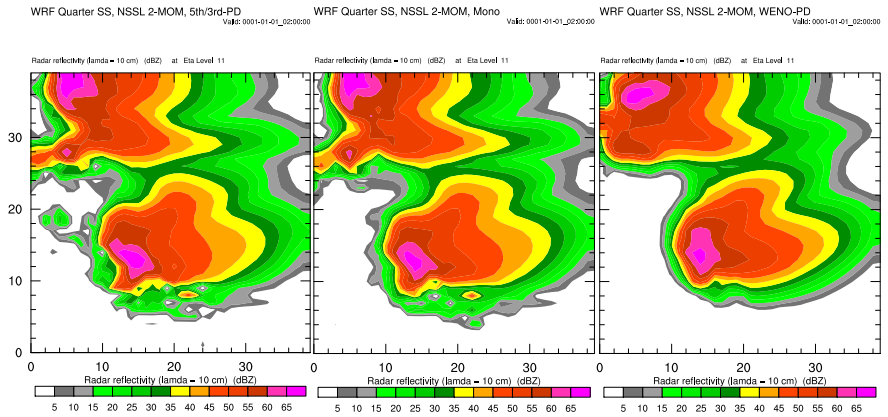
for $U_{i-1/2} \geq 0$

$$flux_{i-1/2} = U_{i-1/2}\left(\frac{w_0}{\overline{w}} f_0 + \frac{w_1}{\overline{w}} f_1 + \frac{w_2}{\overline{w}} f_2\right)$$

Positive-definite option uses same flux-renormalization as the standard ARW transport schemes.

Can be used on momentum in addition to scalars.

Dynamics: 5. Advection (transport) and conservation – WENO scheme

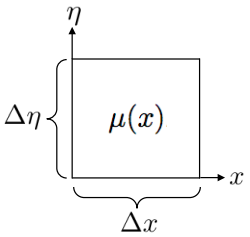


One should consider WENO for some double-moment microphysics scheme applications where noise is a problem.
Application to momentum may not provide much benefit.

Results courtesy of Ted Mansell (NOAA/NSSL)

Dynamics: 5. Advection (transport) and conservation – dry-air mass

control volume (2D example) →



Mass in a control volume is proportional to

$$(\Delta x \Delta \eta)(\mu)^t$$

since $\mu(x)\Delta\eta = \Delta\pi = -g\rho\Delta z$

Dynamics: 5. Advection (transport) and conservation – dry-air mass

Mass in a control volume $(\Delta x \Delta\eta)(\mu)^t$
2D example

Mass conservation equation

$$\Delta t^{-1}(\Delta x \Delta\eta)\cdot\left[(\mu)^{t+\Delta t}-(\mu)^t\right]=\left[(\mu u\Delta\eta)_{x-\Delta x/2,\eta}-(\mu u\Delta\eta)_{x+\Delta x/2,\eta}\right]+\left[(\mu\omega\Delta x)_{x,\eta-\Delta\eta/2}-(\mu\omega\Delta x)_{x,\eta+\Delta\eta/2}\right]$$

Change in mass over a time step

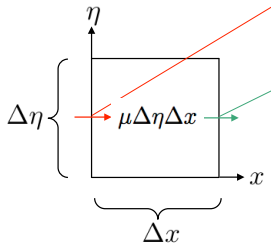
mass fluxes through control volume faces

Dynamics: 5. Advection (transport) and conservation – dry-air mass

Mass in a control volume $(\Delta x \Delta\eta)(\mu)^t$

Mass conservation equation

$$\Delta t^{-1}(\Delta x \Delta\eta)\cdot\left[(\mu)^{t+\Delta t}-(\mu)^t\right]=\left[(\mu u\Delta\eta)_{x-\Delta x/2,\eta}-(\mu u\Delta\eta)_{x+\Delta x/2,\eta}\right]+\left[(\mu\omega\Delta x)_{x,\eta-\Delta\eta/2}-(\mu\omega\Delta x)_{x,\eta+\Delta\eta/2}\right]$$



Horizontal fluxes through the vertical control-volume faces

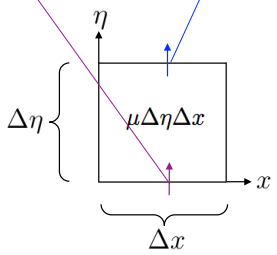
Dynamics: 5. Advection (transport) and conservation – dry-air mass

Mass in a control volume $(\Delta x \Delta\eta)(\mu)^t$

Mass conservation equation

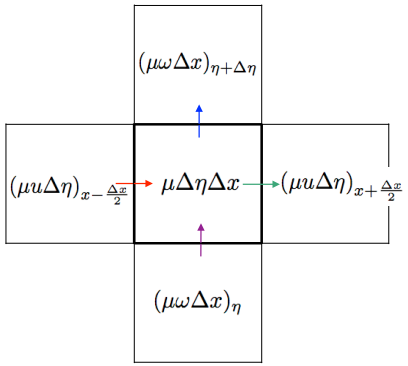
$$\Delta t^{-1}(\Delta x \Delta\eta)\cdot\left[(\mu)^{t+\Delta t}-(\mu)^t\right]=\left[(\mu u\Delta\eta)_{x-\Delta x/2,\eta}-(\mu u\Delta\eta)_{x+\Delta x/2,\eta}\right]+\left[(\mu\omega\Delta x)_{x,\eta-\Delta\eta/2}-(\mu\omega\Delta x)_{x,\eta+\Delta\eta/2}\right]$$

Vertical fluxes through the horizontal control-volume faces



Dynamics: 5. Advection (transport) and conservation – dry-air mass

The same mass fluxes are used for neighboring grid cells - hence mass is conserved locally and globally.



Dynamics: 5. Advection (transport) and conservation – scalars

Mass in a control volume	$(\Delta x \Delta \eta)(\mu)^t$
Scalar mass	$(\Delta x \Delta \eta)(\mu\phi)^t$

Mass conservation equation:

$$\Delta t^{-1}(\Delta x \Delta \eta) \cdot \left[(\mu)^{t+\Delta t} - (\mu)^t\right] = \left[(\mu u \Delta \eta)_{x-\Delta x/2,\eta} - (\mu u \Delta \eta)_{x+\Delta x/2,\eta})\right] + \left[(\mu \omega \Delta x)_{x,\eta-\Delta\eta/2} - (\mu \omega \Delta x)_{x,\eta+\Delta\eta/2})\right]$$

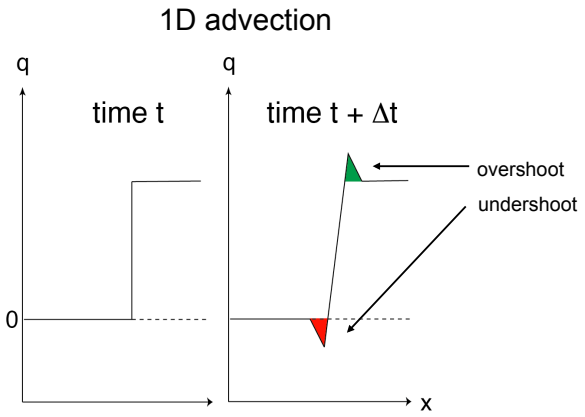
change in mass over a time step — mass fluxes through control volume faces

Scalar mass conservation equation:

$$\Delta t^{-1}(\Delta x \Delta \eta) \cdot \left[(\mu\phi)^{t+\Delta t} - (\mu\phi)^t\right] = \left[(\mu u \phi \Delta \eta)_{x-\Delta x/2,\eta} - (\mu u \phi \Delta \eta)_{x+\Delta x/2,\eta})\right] + \left[(\mu \omega \phi \Delta x)_{x,\eta-\Delta\eta/2} - (\mu \omega \phi \Delta x)_{x,\eta+\Delta\eta/2})\right]$$

change in tracer mass over a time step — tracer mass fluxes through control volume faces

Dynamics: 5. Advection (transport) and conservation – shape preserving



ARW transport is conservative,
but not positive definite nor monotonic.
Removal of negative q ■
results in spurious source of q ■ .

Dynamics: 5. Advection (transport) and conservation – shape preserving

Scalar update, last RK3 step

$$(\mu\phi)^{t+\Delta t} = (\mu\phi)^t - \Delta t \sum_{i=1}^{n} \delta_{x_i}[f_i] \qquad (1)$$

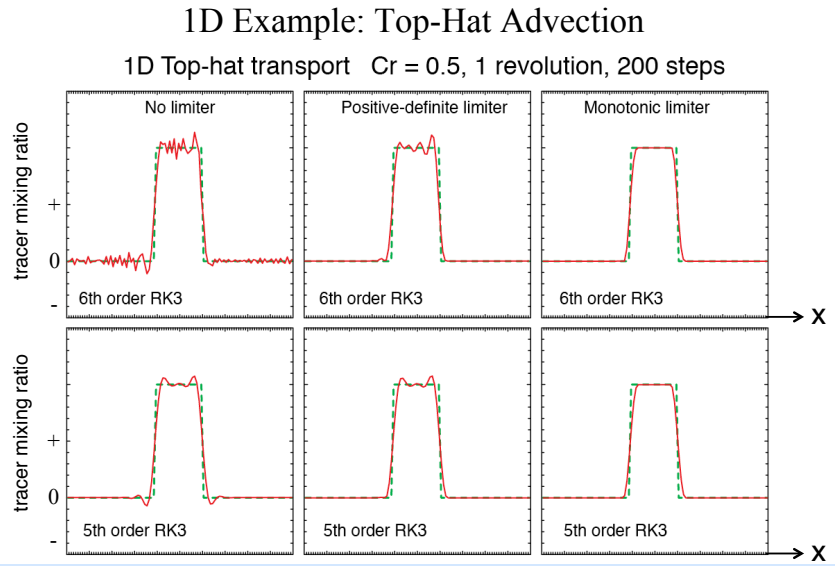
(1) Decompose flux: $f_i = f_i^{upwind} + f_i^c$

(2) Renormalize high-order correction fluxes f_i^c such that solution is positive definite or monotonic: $f_i^c = R(f_i^c)$

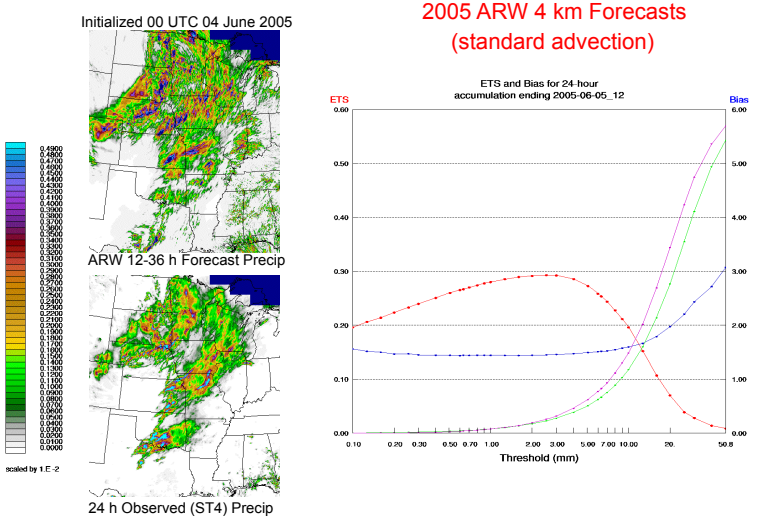
(3) Update scalar eqn. (1) using $f_i = f_i^{upwind} + R(f_i^c)$

Skamarock, MWR 2006, 2241-2250

Dynamics: 5. Advection (transport) and conservation – examples

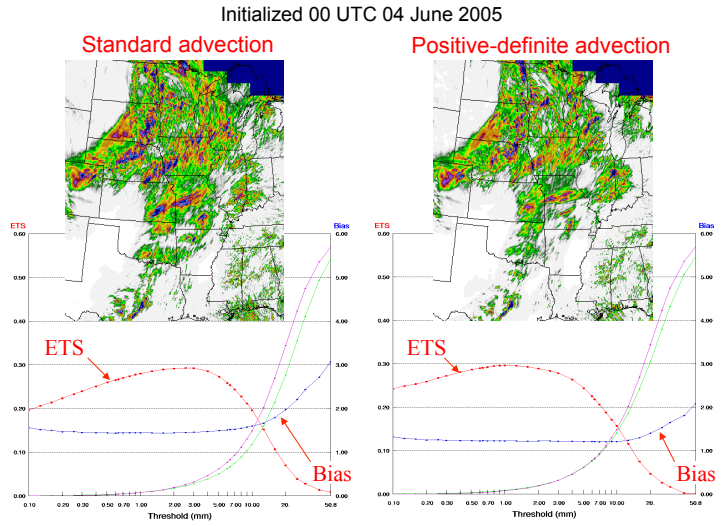


Dynamics: 5. Advection (transport) and conservation – examples



2005 ARW 4 km Forecasts
(standard advection)

Dynamics: 5. Advection (transport) and conservation – examples



Dynamics: 5. Advection (transport) and conservation

Where are the transport-scheme parameters?

The namelist.input file:

&dynamics

h_mom_adv_order
v_mom_adv_order
h_sca_adv_order
v_sca_adv_order

→ scheme order (2, 3, 4, or 5)
defaults:
horizontal (h_*) = 5
vertical (v_*) = 3

momentum_adv_opt →
= 1 standard scheme
= 3 5th order WENO
default: 1

moist_adv_opt
scalar_adv_opt
chem_adv_opt
tracer_adv_opt
tke_adv_opt

→ options:
= 1, 2, 3 : no limiter,
positive definite (PD),
montonic
= 4 : 5th order WENO
= 5 : 5th order PD WENO

Dynamics: 6. Time step parameters

3rd order Runge-Kutta time step Δt_{RK}

Courant number limited, 1D: $C_r = \frac{U\Delta t}{\Delta x} < 1.43$

Generally stable using a timestep approximately twice as large as used in a leapfrog model.

Where?

The namelist.input file:

&domains

time_step (integer seconds)
time_step_fract_num
time_step_fract_den

Dynamics: 6. Time step parameters

3rd order Runge-Kutta time step Δt_{RK} (&domains *time_step*)

Acoustic time step

2D horizontal Courant number limited: $C_r = \frac{C_s \Delta\tau}{\Delta h} < \frac{1}{\sqrt{2}}$

$\Delta\tau_{sound} = \Delta t_{RK} / (\text{number of acoustic steps})$

Where?
The namelist.input file:
&dynamics
time_step_sound (integer)

Dynamics: 6. Time step parameters

3rd order Runge-Kutta time step Δt_{RK} (&domains *time_step*)

Acoustic time step [&dynamics *time_step_sound* (integer)]

Guidelines for time step

Δt_{RK} in seconds should be about 6*Δx (grid size in kilometers). Larger Δt can be used in smaller-scale dry situations, but *time_step_sound* (default = 4) should increase proportionately if larger Δt is used.

If ARW blows up (aborts) quickly, try:

Decreasing Δt_{RK} (that also decreases Δt_{sound}),

Or increasing time_step_sound (that decreases Δt_{sound} but does not change Δt_{RK})

Dynamics: 7. Filters – divergence damping

Purpose: filter acoustic modes (3-D divergence, $D = \nabla \cdot \rho\mathbf{V}$)

$$\left\{ \frac{\partial \rho\mathbf{V}}{\partial t} + \nabla p + \ldots = \gamma_d' \nabla D \right\}$$

$$\nabla \cdot \{\ \} \quad \rightarrow \quad \frac{\partial D}{\partial t} + \nabla^2 p + \ldots = \gamma_d' \nabla^2 D$$

From the pressure equation: $p_t \simeq c^2 D$

$$\frac{\partial \rho\mathbf{V}}{\partial t} + \nabla[p_\tau + \gamma_d (p^\tau - p^{\tau-\Delta\tau})] + \ldots = 0$$

$\gamma_d = 0.1$ recommended (default) (&dynamics *smdiv*)

(Illustrated in height coordinates for simplicity)

Dynamics: 7. Filters – time off-centering the vertical acoustic modes

Purpose: damp vertically-propagating acoustic modes

$$\frac{\partial W}{\partial t} + g\overline{\left(\mu_d - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\right)}^{\tau} = \ldots$$

$$\frac{\partial \phi}{\partial t} - \frac{g}{\mu_d^t}\overline{W}^{\tau} = \ldots$$

$$\overline{(\ )}^{\tau} = \frac{1+\beta}{2}\overline{(\ )}^{\tau+\Delta\tau} + \frac{1-\beta}{2}\overline{(\ )}^{\tau}$$

Slightly forward centering the vertical pressure gradient damps 3-D divergence as demonstrated for the divergence damper

$\beta = 0.1$ recommended (default) [&dynamics *epssm*]

Dynamics: 7. Filters – external mode filter

Purpose: filter the external mode

Vertically integrated horizontal divergence, $D_h = \int_1^0 (\nabla_\eta \cdot \mu \mathbf{V}_h) d\eta$

$$\left\{ \frac{\partial \mu \mathbf{V}_h}{\partial t} + \ldots = -\gamma_e \nabla_\eta D_h \right\}$$

$$\int_1^0 \nabla_\eta \cdot \left\{ \quad \right\} d\eta \quad \rightarrow \quad \frac{\partial D_h}{\partial t} + \ldots = \gamma_e \nabla^2 D_h$$

Continuity equation: $\frac{\partial \mu}{\partial t} = -\nabla_\eta \cdot \mu \mathbf{V}_h - \frac{\partial \mu \dot{\eta}}{\partial \eta} = D_h$

$$\frac{\partial \mu \mathbf{V}_h}{\partial \tau} + \ldots = -\gamma_e \frac{\Delta x^2}{\Delta \tau^2} \nabla_\eta (\mu^\tau - \mu^{\tau - \Delta\tau})$$

γ_e = 0.01 recommended (default) [&dynamics *emdiv*]

(Primarily for real-data applications)

Dynamics: 7. Filters – vertical velocity damping

Purpose: damp anomalously-large vertical velocities

(usually associated with anomalous physics tendencies)

Additional term:

$$\partial_t W = \ldots - \mu_d \, \mathrm{sign}(W) \gamma_w (Cr - Cr_\beta)$$

$$Cr = \left| \frac{\Omega dt}{\mu d\eta} \right|$$

Cr_β = 1.0 typical value (default)
[share/module_model_constants.F *w_beta*]

γ_w = 0.3 m/s^2 recommended (default)
[share/module_model_constants.F *w_alpha*]

[&dynamics *w_damping* 0 (off; default) 1 (on)]

Dynamics: 7. Filters – 2D Smagorinsky

2nd-Order Horizontal Mixing,
Horizontal-Deformation-Based K_h

Purpose: mixing on horizontal coordinate surfaces
(real-data applications) [&dynamics *diff_opt=1, km_opt=4*]

$$K_h = C_s^2 \, l^2 \left[0.25 (D_{11} - D_{22})^2 + \overline{D_{12}^2}^{xy} \right]^{\frac{1}{2}}$$

where $l = (\Delta x \Delta y)^{1/2}$

$$D_{11} = 2\, m^2 \left[\partial_x (m^{-1} u) - z_x \partial_z (m^{-1} u) \right]$$
$$D_{22} = 2\, m^2 \left[\partial_y (m^{-1} v) - z_y \partial_z (m^{-1} v) \right]$$
$$D_{12} = m^2 \left[\partial_y (m^{-1} u) - z_y \partial_z (m^{-1} u) + \partial_x (m^{-1} v) - z_x \partial_z (m^{-1} v) \right]$$

C_s = 0.25 (Smagorinsky coefficient, default value)
[&dynamics *c_s*]

Dynamics: 7. Filters – gravity-wave absorbing layer

Implicit Rayleigh *w* Damping Layer for Split-Explicit
Nonhydrostatic NWP Models (gravity-wave absorbing layer)

Modification to small time step:

- Step horizontal momentum, continuity, and potential temperature equations to new time level: $U^{\tau+\Delta\tau}$ $\mu^{\tau+\Delta\tau}$ $\Omega^{\tau+\Delta\tau}$ $\Theta^{\tau+\Delta\tau}$
- Step vertical momentum and geopotential equations (implicit in the vertical): $W^{*\tau+\Delta\tau}$ $\phi^{*\tau+\Delta\tau}$
- Apply implicit Rayleigh damping on *W* as an adjustment step: $W^{\tau+\Delta\tau} = W^{*\tau+\Delta\tau} - \Delta\tau R_w(\eta) W^{\tau+\Delta\tau}$
- Update final value of geopotential at new time level: $\phi^{\tau+\Delta\tau}$

Dynamics: 7. Filters – gravity-wave absorbing layer

Implicit Rayleigh w Damping Layer for Split-Explicit Nonhydrostatic NWP Models (gravity-wave absorbing layer)

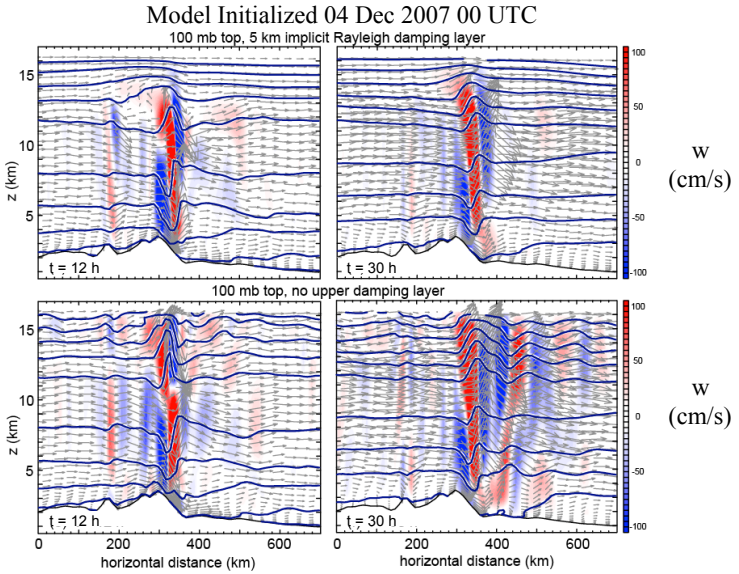
$$W^{\tau+\Delta\tau} = W^{*\tau+\Delta\tau} - \Delta\tau R_w(\eta) W^{\tau+\Delta\tau}$$

$$R_w(\eta) = \begin{cases} \gamma_r \sin^2\left[\frac{\pi}{2}\left(1 - \frac{z_{top}-z}{z_d}\right)\right] & \text{for } z \geq (z_{top} - z_d); \\ 0 & \text{otherwise,} \end{cases}$$

$R_w(\eta)$- damping rate (t^{-1})
z_d - depth of the damping layer
γ_r - damping coefficient

[&dynamics *damp_opt = 3* (default = 0)]
[&dynamics *damp_coef = 0.2* (recommended, = 0. default)]
[&dynamics *zdamp = 5000.* (z_d *(meters)*; default); height below model top where damping begins]

Dynamics: 7. Filters – gravity-wave absorbing layer example



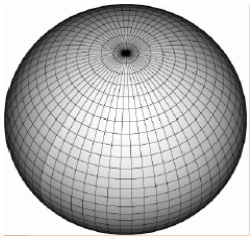
Dynamics: 8. Map projections and global configuration

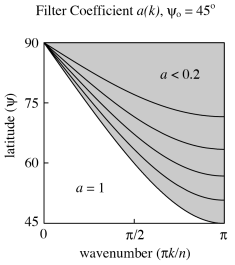
ARW Model: projection options

1. Cartesian geometry:
   idealized cases
2. Lambert Conformal:
   mid-latitude applications
3. Polar Stereographic:
   high-latitude applications
4. Mercator:
   low-latitude applications
5. Latitude-Longitude global, regional

Projections 1-4 are isotropic ($m_x = m_y$)
Latitude-longitude projection is anistropic ($m_x \neq m_y$)

Dynamics: 8. Map projections and global configuration





Global ARW – Polar filters

Converging gridlines severely limit timestep.
The polar filter removes this limitation.

Filter procedure - Along a grid latitude circle:
1. Fourier transform variable.
2. Filter Fourier coefficients.
3. Transform back to physical space.

$$\hat{\phi}(k)_{filtered} = a(k)\,\hat{\phi}(k), \quad \text{for all } k$$

$$a(k) = \min\left[1., \max\left(0., \left(\frac{\cos\psi}{\cos\psi_o}\right)^2 \frac{1}{\sin^2(\pi k/n)}\right)\right]$$

$k =$ dimensionless wavenumber
$\hat{\phi}(k) =$ Fourier coefficients from forward transform
$a(k) =$ filter coefficients
$\psi =$ latitude $\psi_o =$ polar filter latitude, filter when $|\psi| > \psi_o$

ARW integration with polar filtering

Begin time step

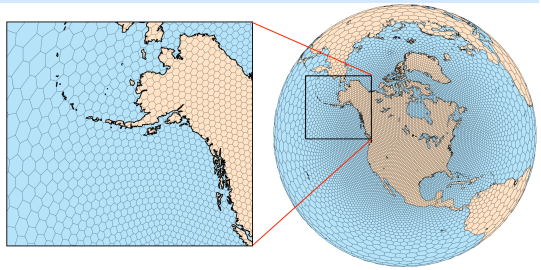
- Runge-Kutta loop (steps 1, 2, and 3)
  - (i) advection, p-grad, buoyancy, physics
  - Acoustic step loop
    - (i) advance U,V (Fourier Filter U,V)
    - (ii) advance μ, Θ (Fourier Filter μ, Θ)
    - (iii) advance w, ϕ (Fourier Filter w, ϕ)
  - End acoustic loop
  - Advance scalars (Fourier Filter Scalars)
- End Runge-Kutta loop
- Adjustment physics (currently microphysics) (Fourier Filter Sc)

End time step

Timestep limited by minimum Δx outside of polar-filter region.
Monotonic and PD transport is not available for global model.

An alternative to global ARW…

MPAS
Model for Prediction Across Scales



- Global, nonhydrostatic, C-grid Voronoi mesh
- Numerics similar to WRF; WRF-NRCM physics
- No pole problems
- Variable-resolution mesh – no nested BC problems

Available at: http://mpas-dev.github.io/

Los Alamos NATIONAL LABORATORY EST. 1943 | U.S. DEPARTMENT OF ENERGY | Office of Science | NSF | NESL

ARW Model: Boundary Condition Options

Lateral boundary conditions

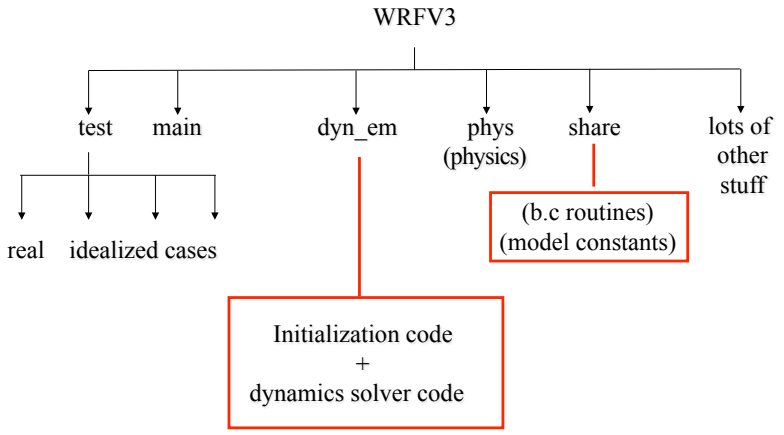
1. Specified (Coarse grid, real-data applications).
2. Open lateral boundaries (gravity-wave radiative).
3. Symmetric lateral boundary condition (free-slip wall).
4. Periodic lateral boundary conditions.
5. Nested boundary conditions (specified).

Top boundary conditions

1. Constant pressure.

Bottom boundary conditions

1. Free slip.
2. Various B.L. implementations of surface drag, fluxes.



WRF ARW Tech Note
A Description of the Advanced Research WRF Version 3 (June 2008, 2012 update)
http://www.mmm.ucar.edu/wrf/users/pub-doc.html

WRF Physics

Jimy Dudhia

NCAR MMM

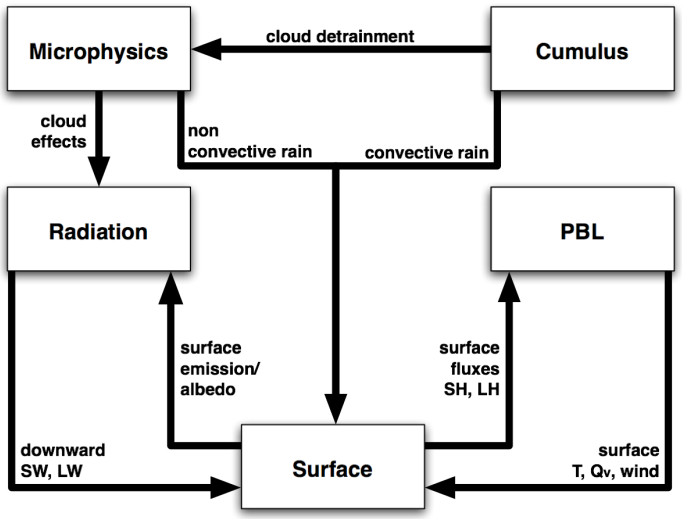
Overview of WRF Physics

Jimy Dudhia
NCAR/MMM

WRF Physics

- Radiation
  - Longwave (ra_lw_physics)
  - Shortwave (ra_sw_physics)
- Surface
  - Surface layer (sf_sfclay_physics)
  - Land/water surface (sf_surface_physics)
- PBL (bl_pbl_physics)
- Turbulence/Diffusion (diff_opt, km_opt)
- Cumulus parameterization (cu_physics)
- Microphysics (mp_physics)

Direct Interactions of Parameterizations

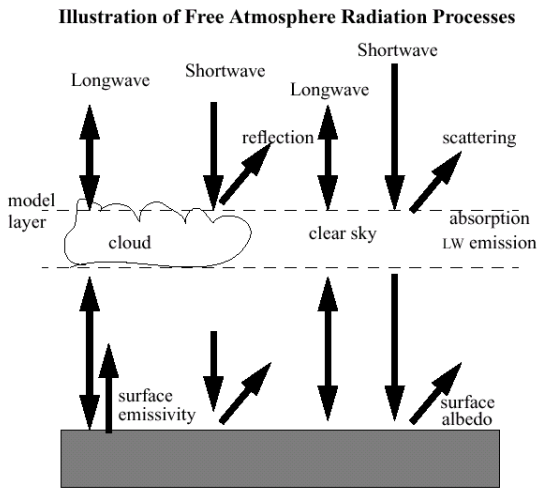


Radiation

Provides

Atmospheric temperature tendency profile

Surface radiative fluxes



WRF Longwave Radiation Schemes (ra_lw_physics)

- Compute clear-sky and cloud upward and downward radiation fluxes
  - Consider IR emission from layers
  - Surface emissivity based on land-type
  - Flux divergence leads to cooling in a layer
  - Downward flux at surface important in land energy budget
  - IR radiation generally leads to cooling in clear air (~2K/day), stronger cooling at cloud tops and warming at cloud base

Longwave Radiation schemes

ra_lw_physics	Scheme	Reference	Added
1	RRTM	Mlawer et al. (1997, JGR)	2000
3	CAM	Collins et al. (2004, NCAR Tech. Note)	2006
4	RRTMG	Iacono et al. (2008, JGR)	2009
5	New Goddard	Chou and Suarez (2001, NASA Tech Memo)	2011
7	FLG (UCLA)	Gu et al. (2011, JGR), Fu and Liou (1992, JAS)	2012
31	Held-Suarez		2008
99	GFDL	Fels and Schwarzkopf (1981, JGR)	2004

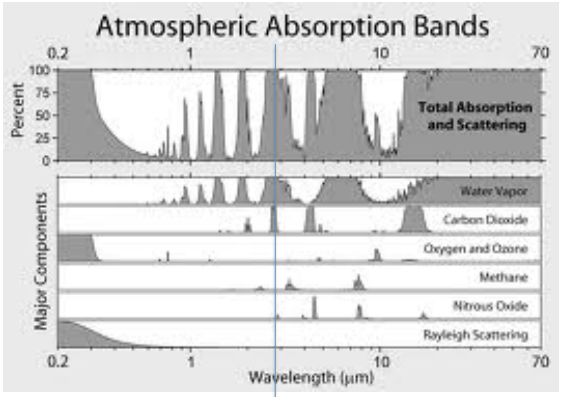
Longwave Radiation in V3.7

ra_lw_physics	Scheme	Cores+Chem	Microphysics Interaction	Cloud Fraction	GHG
1	RRTM	ARW NMM	Qc Qr Qi Qs Qg	1/0	constant or yearly GHG
3	CAM	ARW	Qc Qi Qs	Max-rand overlap	yearly CO2 or GHG
4	RRTMG	ARW +Chem(τ)	Qc Qr Qi Qs	Max-rand overlap	constant or yearly GHG
5	New Goddard	ARW	Qc Qr Qi Qs Qg	1/0	constant
7	FLG (UCLA)	ARW	Qc Qr Qi Qs Qg	1/0	constant
31	Held-Suarez	ARW	none	none	none
99	GFDL	ARW NMM	Qc Qr Qi Qs	Max-rand overlap	constant

Clear Sky: IR-active Gases

- H2O – from model prognostic vapor
- CO2 – well-mixed, specified constant in whole atmosphere (CAM has yearly values)
  - For CAM, RRTM and RRTMG, GHG input file can update CO2, N2O and CH4 (new in V3.5)
- O3 – schemes have own climatologies
  - CAM has monthly, zonal, pressure-level data and RRTMG has this as an option (V3.5)
  - Others use single profiles (Goddard has 5 profiles to choose from)

Radiation effects in clear sky



shortwave longwave

Spectral Bands

- Schemes divide IR spectrum into bands dominated by different absorption gases
- Typically 8-16 bands are used
- Computations use look-up tables for each band
  - Tables were generated from results of line-by-line calculations (LBLRTM models)

Clouds

- All schemes interact with resolved model cloud fields allowing for ice and water clouds and precipitating species
  - Only Thompson passes its own particle sizes to RRTMG radiation (since 3.5.1): other combinations only use mass info and assume effective sizes
- Clouds strongly affect IR at all wavelengths (considered "grey bodies") and are almost opaque to it

Cloud Fractions

- Cloud fraction for microphysics clouds
  - icloud=1: Xu and Randall method
  - icloud=2: simple 1/0 method
  - Icloud=3: new Thompson option in V3.7
- Cloud fraction for unresolved convective clouds
  - cu_rad_feedback = .true.
  - Only works for GF, G3, GD and KF options
  - ZM separately provides cloud fraction to radiation

Cloud Fraction

- Overlap assumptions needed with multiple layers of varying fraction
  - Random overlap
  - Maximum overlap (clouds stacked as much as possible)
  - Maximum-random overlap (maximum for neighboring cloudy layers, random for layers separated by clear air)
- WRF schemes use max-random overlap

WRF Shortwave Radiation Options (ra_sw_physics)

- Compute clear-sky and cloudy solar fluxes
- Include annual and diurnal solar cycles
- Most schemes consider downward and upward (reflected) fluxes
  - Dudhia scheme only has downward flux
- Primarily a warming effect in clear sky
- Important component of surface energy balance

Shortwave Radiation schemes

ra_sw_physics	Scheme	Reference	Added
1	Dudhia	Dudhia (1989, JAS)	2000
2	Goddard	Chou and Suarez (1994, NASA Tech Memo)	2000
3	CAM	Collins et a. (2004, NCAR Tech Note)	2006
4	RRTMG	Iacono et al. (2008, JGR)	2009
5	New Goddard	Chou and Suarez (1999, NASA TM)	2011
7	FLG (UCLA)	Gu et al. (2011, JGR), Fu and Liou (1992, JAS)	2012
99	GFDL	Fels and Schwarzkopf (1981, JGR)	2004

Shortwave Radiation in V3.7

ra_lw_ physics	Scheme	Cores+Chem	Microphysics Interaction	Cloud Fraction	Ozone
1	Dudhia	ARW NMM + Chem(PM2.5)	Qc Qr Qi Qs Qg	1/0	none
2	GSFC	ARW +Chem(τ)	Qc Qi	1/0	5 profiles
3	CAM	ARW	Qc Qi Qs	Max-rand overlap	Lat/ month
4	RRTMG	ARW +Chem(τ), NMM	Qc Qr Qi Qs	Max-rand overlap	1 profile or lat/ month
5	New Goddard	ARW	Qc Qr Qi Qs Qg	1/0	5 profiles
7	FLG (UCLA)	ARW	Qc Qr Qi Qs Qg	1/0	5 profiles
99	GFDL	ARW NMM	Qc Qr Qi Qs	Max-rand overlap	Lat/date

Clear Sky

- Main effect in troposphere is water vapor absorption (CO2 minor effect)
- Aerosols would be needed for additional scattering (WRF-Chem interacts with Goddard and RRTMG shortwave)
  - Dudhia scheme has tunable scattering
  - RRTMG has climatological aerosol input option (since V3.5)
    - aer_opt=1 (Tegen global monthly climatology)
    - aer_opt=2 user-specified properties and/or AOD map

Ozone

- Ozone heating maintains warm stratosphere
- Important for model tops above about 20 km (50 hPa)
- Usually specified from profiles as with longwave options
  - Dudhia scheme has no ozone effect
  - CAM, RRTMG have zonal climatology
- CAM, RRTMG, Goddard can also handle trace gases mainly N2O and CH4 (set constant)

Spectral Bands

- 11-19 spectral bands used by CAM, RRTMG and Goddard schemes
- Look-up tables

Clouds and Cloud Fraction

- Similar considerations to longwave
- Interacts with model resolved clouds and in some cases cumulus schemes
- Fraction and overlap assumptions
- Cloud albedo reflection
- Surface albedo reflection based on land-surface type and snow cover

Slope effects on shortwave

- In V3.2 available for all shortwave options
- Represents effect of slope on surface solar flux accounting for diffuse/direct effects
- slope_rad=1: activates slope effects - may be useful for complex topography and grid lengths < 2 km.
- topo_shading=1: shading of neighboring grids by mountains - may be useful for grid lengths < 1 km.

radt

ARW only

Radiation time-step recommendation

- Radiation is too expensive to call every step
- Frequency should resolve cloud-cover changes with time
- radt=1 minute per km grid size is about right (e.g. radt=10 for dx=10 km)
- Each domain can have its own value but recommend using same value on all 2-way nests

Surface Shortwave Fluxes

New in V3.5.1 (for all shortwave options)

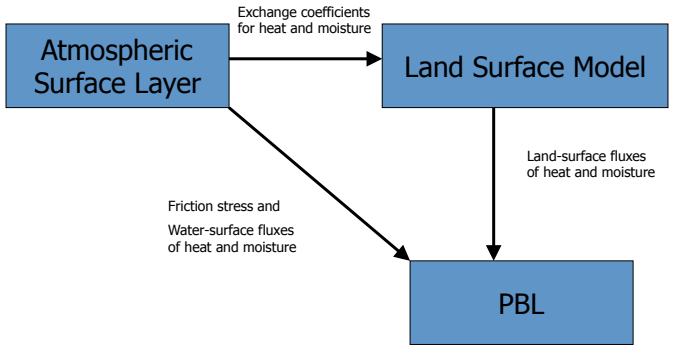
- swint_opt=1 provides a smooth surface downward flux over time (interpolates using cosine zenith angle)
  - This also allows smoother variation of ground variables and fluxes (eliminates steps in time series)
- Diffuse, direct, and direct normal shortwave components are now output (swddir, swddif, swddni)

Surface schemes

Surface layer of atmosphere diagnostics (exchange/transfer coeffs)

Land Surface: Soil temperature /moisture / snow prediction /sea-ice temperature

Surface Physics Components



Surface Fluxes

- Heat, moisture and momentum

$$H = \rho c_p u_* \theta_* \qquad E = \rho u_* q_* \qquad \tau = \rho u_* u_*$$

$$u_* = \frac{kV_r}{\ln(z_r / z_0) - \psi_m} \qquad \theta_* = \frac{k\Delta\theta}{\ln(z_r / z_{0h}) - \psi_h} \qquad q_* = \frac{k\Delta q}{\ln(z_r / z_{0q}) - \psi_h}$$

Subscript *r* is reference level (lowest model level, or 2 m or 10 m)
Δ refers to difference between surface and reference level value
z_0 are the roughness lengths
k is the von Karman constant (0.4)

Roughness Lengths

- Roughness lengths are a measure of the "initial" length scale of surface eddies, and generally differ for velocity and scalars
- Roughness length depends on land-use type
- Some schemes use smaller roughness length for heat than for momentum
- For water points roughness length is a function of surface wind speed

Exchange Coefficient

- C_{hs} is the exchange coefficient for heat, defined such that

$$H = \rho c_p C_{hs} \Delta\theta$$

It is related to the roughness length, stability function and u* by

$$C_{hs} = \frac{k u_*}{\ln\left(\frac{z}{z_0}\right) - \psi_h}$$

WRF Surface Layer Options (sf_sfclay_physics)

- Use similarity theory to determine exchange coefficients and diagnostics of 2m T and q and 10 m winds
- Provide exchange coefficient to land-surface models
- Provide friction velocity to PBL scheme
- Provide surface fluxes over water points
- Schemes have variations in stability functions, roughness lengths

Hurricane Options

ARW only

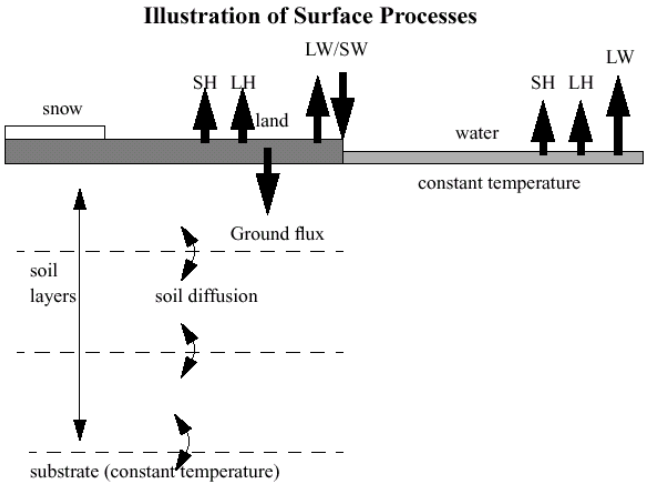
- Ocean Mixed Layer Model (sf_ocean_physics=1)
  - 1-d slab ocean mixed layer (specified initial depth)
  - Includes wind-driven ocean mixing for SST cooling feedback
- 3d PWP ocean (Price et al.) (sf_ocean_physics=2)
  - 3-d multi-layer (~100) ocean, salinity effects
  - Fixed depth
- Alternative surface-layer options for high-wind ocean surface (isftcflx=1,2)
  - Use with sf_sfclay_physics=1
  - Modifies Charnock relation to give less surface friction at high winds (lower Cd)
  - Modifies surface enthalpy (Ck, heat/moisture) either with constant z0q (isftcflx=1), Garratt formulation (option 2)

Fractional Sea Ice

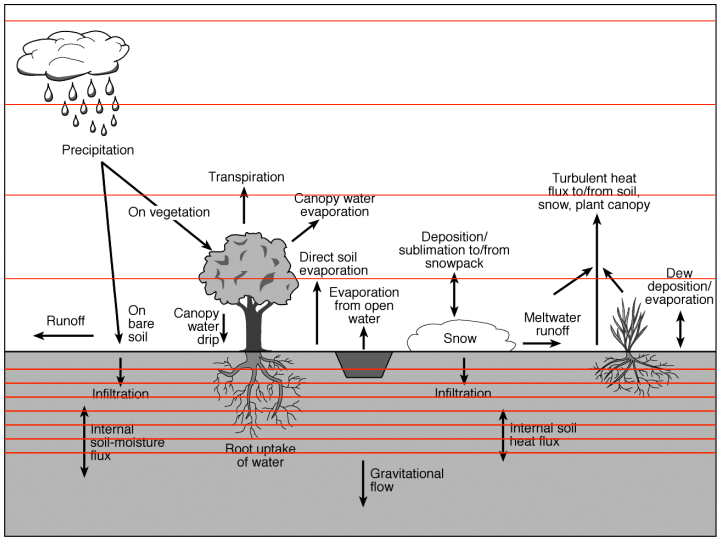
- fractional_seaice=1 - with input sea-ice fraction data can partition land/water fluxes within a grid box
- Can be used with nearly all surface-layer schemes

WRF Land-Surface Model Options (sf_surface_physics)

- Simple 5-layer soil model
  - No vegetation or snow cover prediction, just thermal diffusion in soil layers
- Noah LSM, NoahMP, RUC LSM, PX LSM, CLM4, SSiB land-surface models
  - Sophisticated vegetation model and snow cover prediction



Land-Surface Model Processes



Land-Surface Model

- Driven by surface energy and water fluxes
- Predicts soil temperature and soil moisture in layers (4 for Noah and NoahMP, 9 for RUC, 2 for PX and 3 for SSiB)
- Predicts snow water equivalent on ground. May be in layers (NoahMP, RUC, SSiB)
- May predict canopy moisture only (Noah, RUC) or temperature only (SSiB) or both (NoahMP)

Land-Surface Options

- 5-layer thermal diffusion
- Noah LSM (also with mosaic option in V3.6)
- RUC LSM
- Pleim-Xiu LSM
- NoahMP (new in V3.4)
- SSiB (new in V3.4)
- CLM4 (new in V3.5)

Vegetation and Soil

- Processes include evapotranspiration, root zone and leaf effects
- Vegetation fraction varies seasonally
- Considers vegetation categories (e.g. cropland, forest types, etc.)
- Considers soil categories (e.g. sandy, clay, etc.) for drainage and thermal conductivity

Snow Cover

- LSMs include fractional snow cover and predict snow water equivalent development based on precipitation, sublimation, melting and run-off
  - Single-layer snow (Noah, PX)
  - Multi-layer snow (RUC, NoahMP, SSiB,CLM4)
  - 5-layer option has no snow prediction
- Frozen soil water also predicted (Noah, NoahMP, RUC)

Urban Effects

- Urban category in LSM is usually adequate for larger-scale studies
- Or can use an urban model (sf_urban_physics) with Noah LSM
  - Urban Canopy Model
  - Building Environment Parameterization (multi-layer model)
  - Building Energy Model (adds heating/AC to BEP)
  - NUDAPT detailed map data for 40+ US cities

LSM Tables

- Properties can be changed in text files (tables)
- VEGPARM.TBL used by Noah and RUC for vegetation category properties
  - Albedo, roughness length, emissivity, vegetation properties
- SOILPARM.TBL used by Noah and RUC for soil properties
- LANDUSE.TBL used by 5-layer model
- URBPARM.TBL used by urban models

Initializing LSMs

- Noah and RUC LSM require additional fields for initialization
  - Soil temperature
  - Soil moisture
  - Snow liquid equivalent
- These are in the Grib files, but are not from observations
- They come from "offline" models driven by observations (rainfall, radiation, surface temperature, humidity wind)

Initializing LSMs

- There are consistent model-derived datasets for Noah and RUC LSMs
  - Eta/GFS/AGRMET/NNRP for Noah (although some have limited soil levels available)
  - RUC for RUC
- But, resolution of mesoscale land-use means there will be inconsistency in elevation, soil type and vegetation
- The only adjustment for soil temperature (done in real.exe) is for elevation differences between the original elevation and model elevation (SOILHGT used)
- Inconsistency leads to spin-up as adjustments occur in soil temperature and moisture at the beginning of the simulation
- This spin-up can only be avoided by running offline model on the same grid (e.g. HRLDAS for Noah) – may take months to spin up soil moisture
- Cycling land state between forecasts also helps, but may propagate errors (e.g in rainfall effect on soil moisture)

Sub-grid Mosaic

- Default behavior is one dominant vegetation and soil type per grid cell
- Noah (sf_surface_mosaic) and RUC (mosaic_lu and mosaic_soil) allow multiple categories within a grdi cell

sst_update=1

ARW only

Reads lower boundary file periodically to update the sea-surface temperature (otherwise it is fixed with time)

- For long-period simulations (a week or more)
- wrflowinp_d0n created by real
- Sea-ice can be updated since Version 3.0
- Vegetation fraction update is included
  - Allows seasonal change in albedo, emissivity, roughness length in Noah LSM
- usemonalb=.true. to use monthly albedo input

Regional Climate Options

- tmn_update=1 - updates deep-soil temperature for multi-year future-climate runs
- sst_skin=1 - adds diurnal cycle to sea-surface temperature
- bucket_mm and bucket_J - a more accurate way to accumulate water and energy for long-run budgets (see later)
- output_diagnostics=1 (addes in 3.3.1) – ability to output max/min/mean/std of surface fields in a specified period

Lake Model

- Added in V3.6 (from CLM climate physics)
- 10-layer lake model (sf_lake_physics=1)
- We have global bathymetry data for most large lakes (added from geogrid)
- Also can predict lake ice
- Can be used with any LSM
- Preprocessing allows diurnal averaging methods to initialize lake temperatures where not resolved by SST analysis

WRF-Hydro

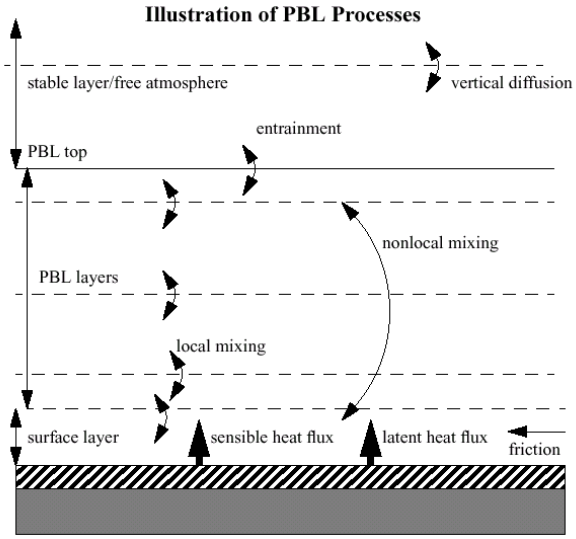
- New in V3.5
- Coupling to hydrological model available
- Streamflow prediction, etc.
- Sub-grid tiling to ~100 m grid

Planetary Boundary Layer

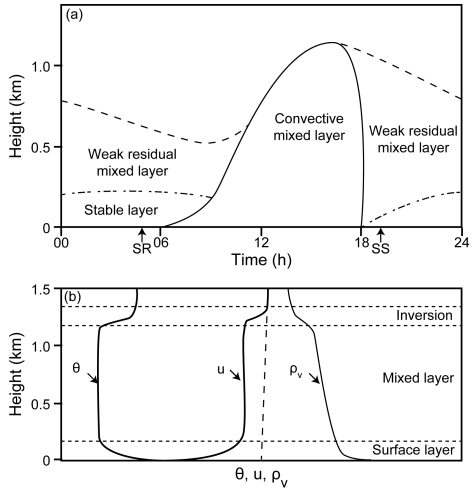
Provides

Boundary layer fluxes (heat, moisture, momentum)

Vertical diffusion in whole column



Planetary Boundary Layer



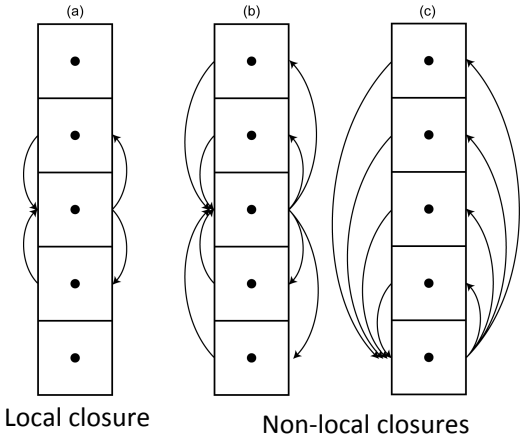
WRF PBL Options (bl_pbl_physics)

- Purpose is to distribute surface fluxes with boundary layer eddy fluxes and allow for PBL growth by entrainment
- Classes of PBL scheme
  - Turbulent kinetic energy prediction (Mellor-Yamada Janjic, MYNN, Bougeault-Lacarrere, TEMF, QNSE, CAM UW)
  - Diagnostic non-local (YSU, GFS, MRF, ACM2)
- Above PBL all these schemes also do vertical diffusion due to turbulence

PBL schemes in V3.7

bl_pbl_physics	Scheme	Reference	Added
1	YSU	Hong, Noh and Dudhia (2006, MWR)	2004
2	MYJ	Janjic (1994, MWR)	2000
3	GFS	Hong and Pan (1996, MWR)	2005
4	QNSE-EDMF	Sukoriansky, Galperin and Perov (2005, BLM), Pergaud, Masson, Malardel et al. (2009, BLM)	2012
5	MYNN2	Nakanishi and Niino (2006, BLM)	2009
6	MYNN3	Nakanishi and Niino (2006, BLM)	2009
7	ACM2	Pleim (2007, JAMC)	2008
8	BouLac	Bougeault and Lacarrere (1989, MWR)	2009
9	UW	Bretherton and Park (2009, JC)	2011
10	TEMF	Angevine, Jiang and Mauritsen (2010, MWR)	2011
11	SH	Shin and Hong (2014, MWR)	2015
12	GBM	Grenier and Brethertion (2001, MWR)	2013
99	MRF	Hong and Pan (1996, MWR)	2000

Different approaches



TKE schemes

- Solve for TKE in each column
  - Buoyancy and shear production
  - Dissipation
  - Vertical mixing $\frac{\partial}{\partial z} K_v \frac{\partial}{\partial z} \theta$
- TKE and length-scale are used to determine the Kv for local vertical mixing
- Schemes differ most in diagnostic length-scale computations

Nonlocal Schemes

- Diagnose a PBL top (either stability profile or Richardson number)
- Specify a K profile $\frac{\partial}{\partial z} K_v \left(\frac{\partial}{\partial z} \theta + \Gamma \right)$
- YSU, MRF, GFS include a non-gradient term (Γ)
- ACM2, TEMF, EDMF include a mass-flux profile, M, which is an additional updraft flux

$$\frac{\partial}{\partial z} \left(K_v \frac{\partial}{\partial z} \theta + M(\theta_u - \theta) \right)$$

Vertical Mixing Coefficient

- Several schemes also output exch_h which is Kv for scalars that is used by WRF-Chem
- WRF can now (V3.6) do scalar and tracer vertical mixing with PBL K-coefficients
  - scalar_pblmix=1, tracer_pblmix=1
- PBL schemes themselves only mix limited variables: momentum, heat, vapor and some specific cloud variables

PBL Schemes with Shallow Convection

- Some PBL schemes include shallow convection as part of their parameterization
- These use mass-flux approaches either
  - through the whole cloud-topped boundary layer (QNSE-EDMF and TEMF)
  - only from cloud base (GBM and UW PBL)
- YSU has top-down mixing option for turbulence driven by cloud-top radiative cooling which is separate from bottom-up surface-flux-driven mixing

Grey-Zone PBL

- "Grey Zone" is sub-kilometer grids
  - PBL and LES assumptions not perfect
- New Shin-Hong PBL based on YSU designed for sub-kilometer transition scales (200 m – 1 km)
  - Nonlocal mixing term reduces in strength as grid size gets smaller and local mixing increases
- Other schemes may work in this range but will not have correctly partitioned resolved/sub-grid energy fractions

PBL schemes in V3.7

bl_pbl_physics	Scheme	Cores	sf_sfclay_physics	Prognostic variables	Diagnostic variables	Cloud mixing
1	YSU	ARW NMM	1,91		exch_h	QC,QI
2	MYJ	ARW NMM	2	TKE_PBL	EL_PBL, exch_h	QC,QI
3	GFS(hwrf)	NMM	3			QC,QI
4	QNSE-EDMF	ARW NMM	4	TKE_PBL	EL_PBL, exch_h, exch_m	QC,QI
5	MYNN2	ARW	1,2,5,91	QKE	Tsq, Qsq, Cov, exch_h, exch_m	QC
6	MYNN3	ARW	1,2,5,91	QKE, Tsq, Qsq, Cov	exch_h, exch_m	QC
7	ACM2	ARW	1,7,91			QC,QI
8	BouLac	ARW	1,2,91	TKE_PBL	EL_PBL, exch_h, exch_m	QC
9	UW	ARW	1,2,91	TKE_PBL	exch_h, exch_m	QC
10	TEMF	ARW	10	TE_TEMF	*_temf	QC, QI
11	SH	ARW	1,91		Exch_h	QC, QI
12	GBM	ARW	1,91	TKE_PBL	EL_PBL,exch_h, exch_m	QC, QI
99	MRF	ARW NMM	1,91			QC,QI

PBL Scheme Options

PBL schemes can be used for most grid sizes when surface fluxes are present

- Lowest level should be in the surface layer (0.1h)
  - Important for surface (2m, 10m) diagnostic interpolation
- With ACM2, GFS and MRF PBL schemes, lowest full level should be .99 or .995 not too close to 1
- TKE schemes can use thinner surface layers
- Assumes that PBL eddies are not resolved
- At grid size dx << 1 km, this assumption breaks down
  - Can use 3d diffusion instead of a PBL scheme in Version 3 (coupled to surface physics)
  - Works best when dx and dz are comparable

Large-Eddy Simulation

- For grid sizes of up to about 100 m, LES is preferable
- LES treats turbulence three-dimensionally instead of separate vertical (PBL) and horizontal diffusion schemes
- TKE and 3d Smagorinsky options exist for the sub-grid turbulence

Large-Eddy Simulation

- To run LES mode
  - Use bl_pbl_physics=0 and diff_opt=2 with km_opt=2 or 3
  - This scheme can also use real surface fluxes from the surface physics (heat, moisture, momentum stress) or idealized constant values

LES schemes

Unified horizontal and vertical mixing (for dx~dz).
Typically needed for dx<~200 m. Also use mix_isotropic=1.

bl_pbl_physics	diff_opt	km_opt	Scheme	Cores	sf_sfclay_physics	isfflx	Prognostic variables
0	2	2	tke	ARW	0,1,2	0,1,2	tke
0	2	3	3d Smagorinsky	ARW	0,1,2	0,1,2	

Namelist isfflx controls surface flux methods

isfflx	sf_sfclay_physics	Heat flux	Drag	Real/Ideal
0	0	From namelist tke_heat_flux	From namelist tke_drag_coefficient	Ideal
1	1,2	From LSM/sfclay physics (HFX, QFX)	From sfclay physics (UST)	Real
2	1,2	From namelist tke_heat_flux	From sfclay physics (UST)	Ideal

Other Options

- Gravity-wave drag can be added for low resolution (> 10 km) runs to represent sub-grid orographic gravity-wave vertical momentum transport (gwd_opt=1)
- Wind-farm model has been added to investigate wind-farm effects on the environment (extra stress and turbulence generation)
- topo_wind=1,2: wind-bias correction methods for terrain effects
- Fog: grav_settling=2 (Katata)

bldt

ARW only

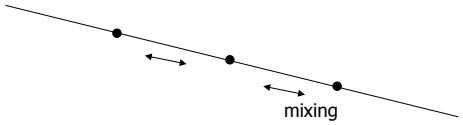
- Minutes between boundary layer/LSM calls
- Typical value is 0 (every step)

Turbulence/Diffusion

Sub-grid eddy mixing effects on all fields, e.g.

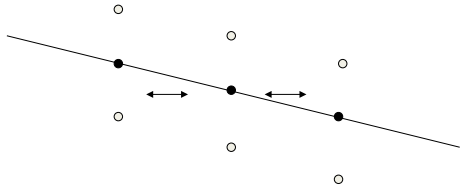
$$\frac{\partial}{\partial x} K_h \frac{\partial}{\partial x}\theta + \frac{\partial}{\partial y} K_h \frac{\partial}{\partial y}\theta + \frac{\partial}{\partial z} K_v \frac{\partial}{\partial z}\theta$$

Difference between diff_opt 1 and 2



diff_opt=1
Horizontal diffusion acts along model levels
Simpler numerical method with only neighboring points on the same model level

Difference between diff_opt 1 and 2



diff_opt=2
Horizontal diffusion acts on horizontal gradients
Numerical method includes vertical correction term using more grid points

ARW only

km_opt

- km_opt selects method for computing K coefficient
  - km_opt=1: constant (use khdif and kvdif to specify – idealized)
  - km_opt=2: 3d tke prediction used to compute K (requires diff_opt=2)
  - km_opt=3: 3d Smagorinsky diagnostic K (requires diff_opt=2)
  - km_opt=4: 2d Smagorinsky for horizontal K (to be used with PBL or kvdif for vertical K)

ARW only

sfs_opt

- Sub-filter-scale stress model for LES applications impacting momentum mixing (Kosovic, Mirocha)
  - sfs_opt=0 (default) off
  - sfs_opt=1 Nonlinear Backscatter and Anisotropy (NBA) option 1: using diagnostic stress terms (km_opt=2,3)
  - sfs_opt=2 NBA option 2: using tke-based stress terms (km_opt=2 only)
  - Also m_opt=1 for added outputs of SGS stresses

Diffusion Option Choice

- Real-data case with PBL physics on
  - Best is diff_opt=1, km_opt=4
  - From V3.6 diff_opt=2 can be used with km_opt=4 (was unstable with complex terrain before this version)
  - This complements vertical diffusion done by PBL scheme
- High-resolution real-data cases (~100 m grid)
  - No PBL
  - diff_opt=2; km_opt=2,3 (tke or Smagorinsky scheme)
- Idealized cloud-resolving (dx =1-3 km) modeling (smooth or no topography, no surface heat fluxes)
  - diff_opt=2; km_opt=2,3
- Complex topography with no PBL scheme
  - diff_opt=2 is more accurate for sloped coordinate surfaces, and prevents diffusion up/down valley sides but still sometimes unstable with complex terrain
- Note: WRF can run with no diffusion (diff_opt=0)

ARW only

diff_6th_opt

- 6th order optional added horizontal diffusion on model levels
  - Used as a numerical filter for 2*dx noise
  - Suitable for idealized and real-data cases
  - Affects all advected variables including scalars
- diff_6th_opt
  - 0: none (default)
  - 1: on (can produce negative water)
  - 2: on and prohibit up-gradient diffusion (better for water conservation)
- diff_6th_factor
  - Non-dimensional strength (typical value 0.12, 1.0 corresponds to complete removal of 2*dx wave in a time-step)

ARW only

Upper damping (damp_opt)

Purpose is to prevent unrealistic reflections of waves from model top. Can be important over high topography.

Options

- 1: Upper level diffusive layer
- 2: Rayleigh damping (idealized only – needs input sounding)
- 3: w-Rayleigh damping (damps w only)

All options use

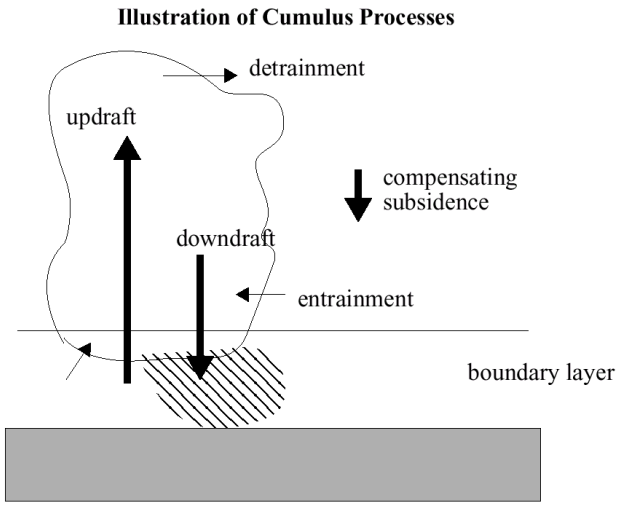
- Cosine function of height
- Additional parameters
  - zdamp: depth of damping layer
  - dampcoef: nondimensional maximum magnitude of damping

Cumulus Parameterization

Provides

Atmospheric heat and moisture/cloud tendency profiles

Surface sub-grid-scale (convective) rainfall



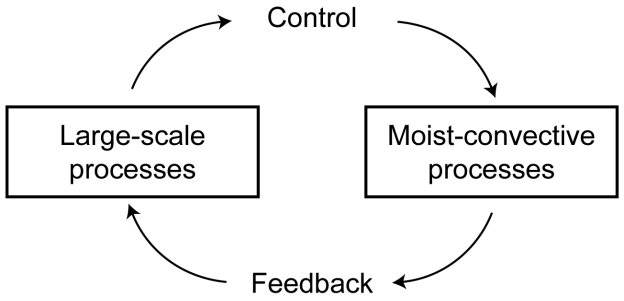
Cumulus Schemes

- Use for grid columns that completely contain convective clouds
- Re-distribute air in column to account for vertical convective fluxes
  - Updrafts take boundary layer air upwards
  - Downdrafts take mid-level air downwards
- Schemes have to determine
  - When to trigger a convective column
  - How fast to make the convection act

Deep Convection

- Schemes work in individual columns that are considered convectively unstable
- Mass-flux schemes transport surface air to top of cloud and include subsidence
- Subsidence around cloud warms and dries troposphere removing instability over time
- Additionally downdrafts may cool PBL

Parameterizations of cumulus convection



WRF Cumulus Parameterization Options

- Cumulus schemes fall into two main classes
  - Adjustment type (Betts-Miller-Janjic)
    - Relaxes towards a post-convective (mixed) sounding
  - Mass-flux type (all others in WRF)
    - Determines updraft (and often downdraft) mass flux and other fluxes (sometimes including momentum transport)

Cumulus schemes in V3.7

cu_physics	Scheme	Reference	Added
1	Kain-Fritsch	Kain (2004, JAM)	2000
2	Betts-Miller-Janjic	Janjic (1994, MWR; 2000, JAS)	2002
3	Grell-Freitas	Grell and Freitas (2013, to be published)	2013
4	Old Simplified Arakawa-Schubert	Grell et al. (1994, MM5 NCAR Tech Note)	2002/ 2011
5	Grell-3	Grell and Devenyi (2002, GRL)	2008
6,16	Tiedtke	Tiedtke (1989, MWR), Zhang, Wang and Hamilton (2011, MWR)	2011, 2015
7	Zhang-McFarlane	Zhang and McFarlane (1995, AO)	2011
11	Multi-Scale KF	Alapaty and Herwehe	2015
14	New SAS	Han and Pan (2010,...)	2011
84	New SAS (HWRF)	Han and Pan (2010,...)	2012
93	Grell-Devenyi	Grell and Devenyi (2002, GRL)	2002
99	Old Kain-Fritsch	Kain and Fritsch (1990, JAS; 1993 Meteo. Monogr.)	2000

Triggers

- Clouds only activate in columns that meet certain criteria
  - Presence of some convective available potential energy (CAPE) in sounding
  - Not too much convective inhibition (CIN) in sounding (cap strength)
  - Minimum cloud depth from parcel ascent

Closures

- Closure determine cloud strength (mass-flux) based on various methods
  - Clouds remove CAPE over time
    - Specified CAPE-removal time scale (KF, Tiedtke, ZM, BMJ)
    - Quasi-equilibrium (Arakawa-Schubert) with large-scale destabilization d(CAPE)/dt (SAS, NSAS)
    - Column moisture convergence
    - Low-level large-scale ascent (mass convergence)

Ensemble methods

- GF, G3 and GD use ensemble of triggers and closures possibly with varying parameters (up to 144 members)
- Take mean of ensemble to feed back to model
- In principle, can be tuned to emphasize various members under different conditions

Shallow Convection

- Non-precipitating shallow mixing dries PBL, moistens and cools above
- This can be done by an enhanced mixing approach (SAS, GRIMS) or mass-flux approach (KF, NSAS, Tiedtke, G3, GF)
- May be useful at grid sizes that do not resolve shallow cumulus clouds (> 1 km)

Shallow Convection

- Cumulus schemes may include shallow convection (KF, SAS schemes, G3, GF, BMJ, Tiedtke)
- Standalone shallow schemes
  - UW Park-Bretherton (shcu_physics=2)
  - GRIMS shallow scheme (shcu_physics=3)
- Part of PBL schemes with mass-flux method
  - TEMF PBL option (bl_bl_physics=10)
  - GBM PBL option (bl_bl_physics=12)
  - QNSE-EDMF PBL (bl_bl_physics=4)

Momentum Transport

- Some cumulus parameterizations also have momentum transport (SAS, NSAS, Tiedtke, ZM)
- Most schemes transport momentum as a passive scalar but ZM and NSAS include a convective pressure gradient term

Cloud Detrainment

- Most schemes detrain cloud and ice at cloud top (except BMJ)
- KF schemes also detrain snow and rain
- These are then used by the microphysics

ARW only

cudt

- Time between cumulus scheme calls
- Typical value is 5 minutes
  - Note: for KF scheme this is also used for averaging time for vertical velocity trigger
  - Not used by G3 or GD schemes

Cumulus schemes in V3.7

cu_physics	Scheme	Cores	Moisture Tendencies	Momentum Tendencies	Shallow Convection
1	Kain-Fritsch Eta	ARW NMM	Qc Qr Qi Qs	no	yes
2	Betts-Miller-Janjic	ARW NMM	-	no	yes
3	Grell-Freitas	ARW	Qc Qi	no	yes
4	Old Simplified Arakawa-Schubert	ARW NMM	Qc Qi	yes (NMM)	yes (ARW)
5	Grell-3	ARW	Qc Qi	no	yes
6,16	Tiedtke	ARW	Qc Qi	yes	yes
7	Zhang-McFarlane	ARW	Qc Qi	yes	no
14	New SAS	ARW	Qc Qi	yes	yes
84	New SAS (HWRF)	ARW NMM	Qc Qi	yes (NMM)	yes
93	Grell-Devenyi	ARW	Qc Qi	no	no
99	Old Kain-Fritsch	ARW	Qc Qr Qi Qs	no	no

Cumulus scheme

Recommendations about use

- For dx ≥ 10 km: probably need cumulus scheme
  - These release instability gradually (prevent grid-point storms)
- For dx ≤ 3 km: probably do not need scheme (resolved/ permitted by dynamics)
  - However, there are cases where the earlier triggering of convection by cumulus schemes help
- For dx=3-10 km, scale separation is a question
  - Few schemes are specifically designed with this range of scales in mind
  - G3 has an option to spread subsidence in neighboring columns
  - GF automatically phases out deep convection at fine grid size
- Issues with 2-way nesting when physics differs across nest boundaries (seen in precip field on parent domain)
  - best to use same physics in both domains or 1-way nesting or make nested domain large enough to keep parent effects away from interior

Microphysics

Provides

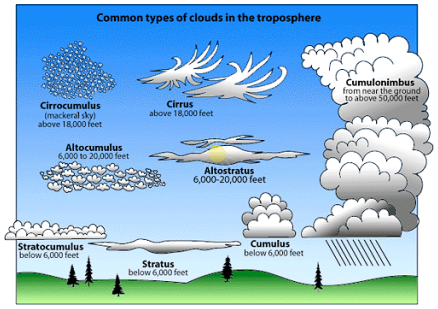
Atmospheric heat and moisture tendencies

Microphysical rates

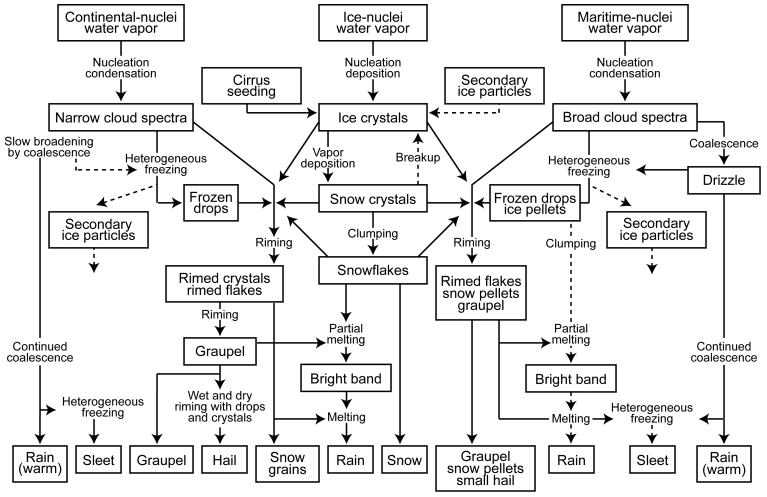
Surface resolved-scale rainfall

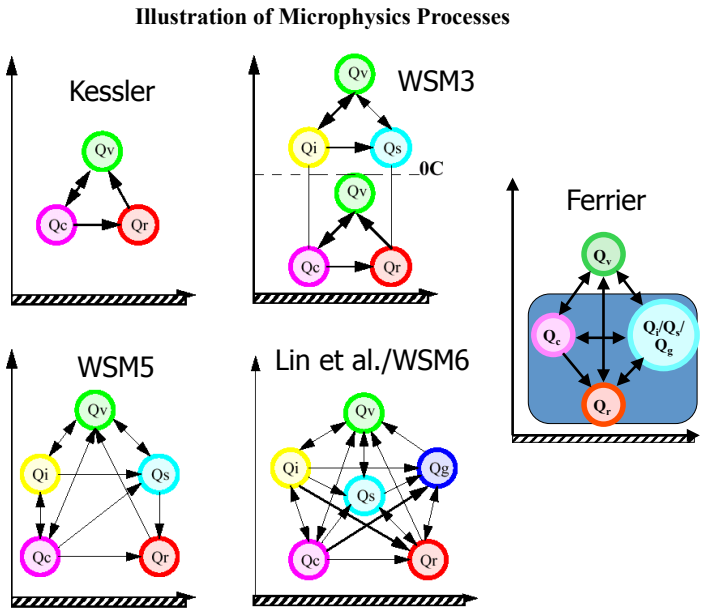
Resolved clouds

- Formed by radiative, dynamical or convective processes
- Model only considers grid-scale average so will not resolve fine-scale structures



Microphysics Parameterization





WRF Microphysics Options (mp_physics)

- Range of levels of sophistication
  - Warm rain (i.e. no ice) – Kessler (idealized)
  - Simple ice (3 arrays) – WSM3
  - Mesoscale (5 arrays, no graupel) – WSM5
  - Cloud-scale single-moment (6 arrays, graupel) – WSM6, Lin, Goddard, SBU, Eta-Ferrier
  - Double-moment (8-13 arrays) – Thompson, Morrison, Milbrandt-Yau, WDM5, WDM6
  - Spectral Bin (120-240 arrays)

Microphysics schemes in V3.7

mp_physics	Scheme	Reference	Added
1	Kessler	Kessler (1969)	2000
2	Lin (Purdue)	Lin, Farley and Orville (1983, JCAM)	2000
3	WSM3	Hong, Dudhia and Chen (2004, MWR)	2004
4	WSM5	Hong, Dudhia and Chen (2004, MWR)	2004
5	Eta (Ferrier)	Rogers, Black, Ferrier et al. (2001)	2000
6	WSM6	Hong and Lim (2006, JKMS)	2004
7	Goddard	Tao, Simpson and McCumber (1989, MWR)	2008
8	Thompson (+old)	Thompson et al. (2008, MWR)	2009
9	Milbrandt 2-mom	Milbrandt and Yau (2005, JAS)	2010
10	Morrison 2-mom	Hong and Pan (1996, MWR)	2008
11	CESM 1.0	Morrison and Gettelman (2008, JC)	2013
13	SBU-Ylin	Lin and Colle (2011, MWR)	2011
14	WDM5	Lim and Hong (2010, MWR)	2009
16	WDM6	Lim and Hong (2010, MWR)	2009
17	NSSL 2-mom	Mansell, Ziegler and Bruning (2010, JAS)	2012
18	NSSL 2-mom + ccn	Mansell, Ziegler and Bruning (2010, JAS)	2012

Microphysics schemes in V3.7

mp_physics	Scheme	Reference	Added
19	NSSL 7-class	Mansell, Ziegler and Bruning (2010, JAS)	2013
21	NSSL 6-class	Mansell, Ziegler and Bruning (2010, JAS)	2013
22	NSSL 6-class 2-mom	Mansell, Ziegler and Bruning (2010, JAS)	2015
28	Thompson aero	Thompson and Eidhammer (2014, JAS)	2014
30	SBM fast	Hong, Dudhia and Chen (2004, MWR)	2014
32	SBM full	Rogers, Black, Ferrier et al. (2001)	2014

Microphysics

- Latent heat release from
  - Condensation, evaporation, deposition, sublimation, freezing, melting
- Particle types
  - Cloud water, rain drops, ice crystals, snow, graupel (also hail in some)
    - Total mass contributes to liquid loading in dynamics
- Processes
  - Aggregation, accretion, growth, fall-out

Microphysics: Single and Double Moment Schemes

- Single-moment schemes have one prediction equation for mass (kg/kg) per species (Qr, Qs, etc.) with particle size distribution being derived from fixed parameters
- Double-moment (DM) schemes add a prediction equation for number concentration (#/kg) per DM species (Nr, Ns, etc.)
  - DM schemes may only be double-moment for a few species
  - DM schemes allow for additional processes such as size-sorting during fall-out and sometimes aerosol (CCN) effects

Spectral Bin Schemes

- New in V3.6 (Hebrew University of Jerusalem, Khain and Lynn scheme)
- Size distribution resolved by doubling mass bins (typically 32 for each particle type)
- Many added advected arrays (expensive)
  - Options have 4x32 or 8x32 arrays

Microphysics: Fall terms

- Microphysics schemes handle fall terms for particles (usually everything except cloud water has a fall term)
- For long time-steps (such as mesoscale applications dt ~ 60 s, Vt= 5 m/s), drops may fall more than a grid level in a time-step
- This requires splitting the time-step (most schemes) or lagrangian numerical methods (WSM and WDM schemes) to keep the scheme numerically stable

Interaction with Aerosols

- WRF-Chem can provide aerosols to some options (Lin, Morrison, CESM)
- WDM, an NSSL option, and spectral bin schemes can advect idealized CCNs which affect cloud droplet number
- Thompson "aerosol-aware" scheme (new in V3.6) can use its own aerosol climatology

Interaction with Radiation

- Several schemes now pass their own ice, snow, cloud-water particle sizes to RRTMG radiation
  - Thompson, WSM, WDM, NSSL 2-mom schemes
  - Other schemes do not and radiation uses internal assumptions about particle sizes

Microphysics schemes in V3.7

* Advects only total condensate Nn= CCN number

mp_physics	Scheme	Cores	Mass Variables	Number Variables
1	Kessler	ARW	Qc Qr	
2	Lin (Purdue)	ARW (Chem)	Qc Qr Qi Qs Qg	
3	WSM3	ARW	Qc Qr	
4	WSM5	ARW NMM	Qc Qr Qi Qs	
5	Eta (Ferrier)	ARW NMM	Qc Qr Qs (Qt*)	
6	WSM6	ARW NMM	Qc Qr Qi Qs Qg	
7	Goddard	ARW	Qc Qr Qi Qs Qg	
8	Thompson	ARW NMM	Qc Qr Qi Qs Qg	Ni Nr
9	Milbrandt 2-mom	ARW	Qc Qr Qi Qs Qg Qh	Nc Nr Ni Ns Ng Nh
10	Morrison 2-mom	ARW (Chem)	Qc Qr Qi Qs Qg	Nr Ni Ns Ng
11	CESM 1.0	ARW (Chem)	Qc Qr Qi Qs	Nc Nr Ni Ns
13	SBU-YLin	ARW	Qc Qr Qi Qs	
14	WDM5	ARW	Qc Qr Qi Qs	Nn Nc Nr
16	WDM6	ARW	Qc Qr Qi Qs Qg	Nn Nc Nr
17	NSSL 2-mom	ARW	Qc Qr Qi Qs Qg Qh	Nc Nr Ni Ns Ng Nh
18	NSSL2-mom+ccn	ARW	Qc Qr Qi Qs Qg Qh	Nc Nr Ni Ns Ng Nh Nn

Microphysics schemes in V3.7

* Advects only total condensate Nn= CCN number

mp_physics	Scheme	Cores	Mass Variables	Number Variables
19	NSSL 7-class	ARW	Qc Qr Qi Qs Qg Qh	VOLg
21	NSSL 6-class	ARW	Qc Qr Qi Qs Qg	
22	NSSL 6-class 2-mom	ARW	Qc Qr Qi Qs Qg	Nn Nc Nr Ni Ns Ng VOLg
28	Thompson aero	ARW	Qc Qr Qi Qs Qg	Nc Ni Nr Nn Nni
30	HUJI fast SBM	ARW	Qc Qr Qi Qs Qg	Nn Nc Nr Ni Ns Ng
32	HUJI full SBM	ARW	Qc Qr Qic Qip Qid Qs Qg Qh (outputs aggregated from bins)	Nn Nc Nr Nic Nip Nid Ns Ng Nh

Microphysics Options

Recommendations about choice

- Probably not necessary to use a graupel scheme for dx > 10 km
  - Updrafts producing graupel not resolved
  - Cheaper scheme may give similar results
- When resolving individual updrafts, graupel scheme should be used
- All domains use same option

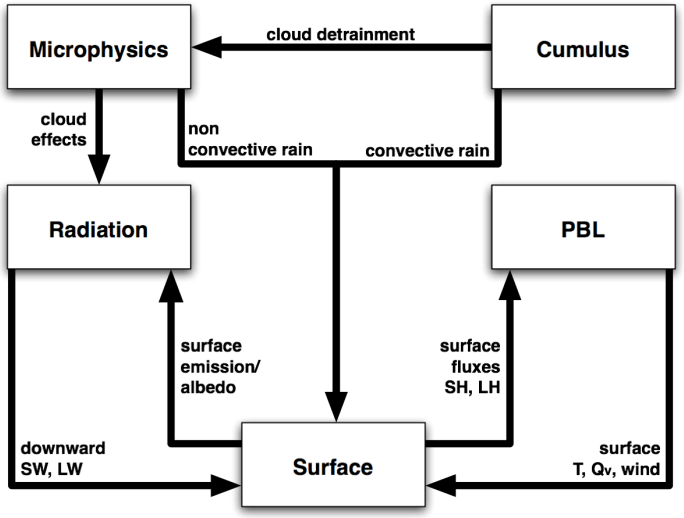
ARW only

Rainfall Output

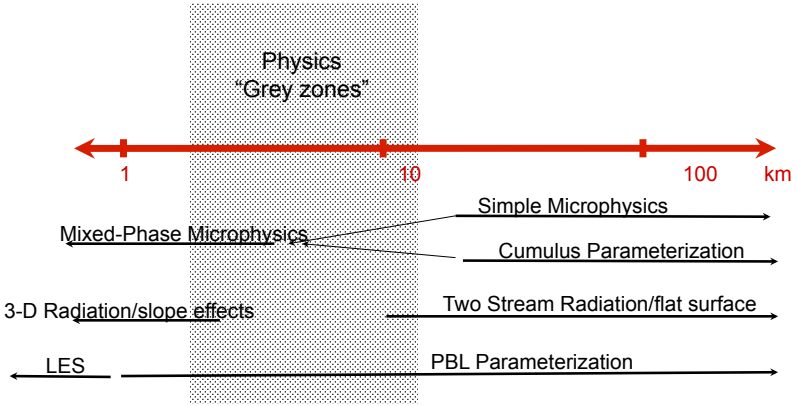
- Cumulus and microphysics can be run at the same time
- ARW outputs rainfall accumulations since simulation start time (0 hr) in mm
- RAINC comes from cumulus scheme
- RAINNC comes from microphysics scheme
- Total is RAINC+RAINNC
  - RAINNCV is time-step value
  - SNOWNC/SNOWNCV are snow sub-set of RAINC/RAINNCV (also GRAUPELNC, etc.)

Physics Interactions

Direct Interactions of Parameterizations



Physics in Multiscale NWP Model

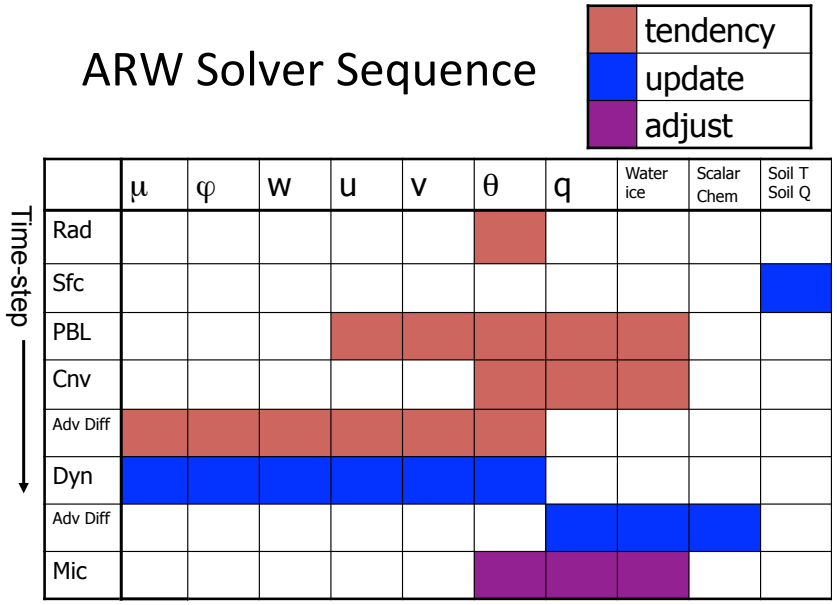


Solver Calling Sequence (ARW example)

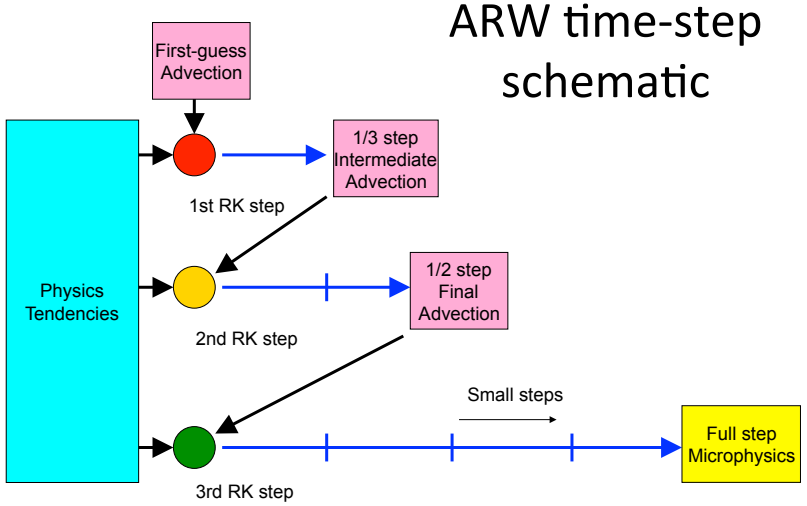
Call to solver advances one domain by one model time-step

- Physics tendencies
  - Radiation, surface, land-state update, PBL, cumulus, grid-fdda, obs-fdda
- Dynamics tendencies
  - Diffusion, advection, dynamics terms (for 3d momentum, theta, geopotential, surface pressure)
- Acoustic steps
  - Update 3d momentum, theta, surface pressure, height
- Scalar dynamics tendencies and update
  - Advection, diffusion of moist (qv,qc, etc.), scalar, tracer, tke, (and chemistry) variables
- Microphysics update

ARW Solver Sequence



ARW time-step schematic



&physics

Seven major physics categories:

- **mp_physics**: 0,1,2,3,…
- **ra_lw_physics**: 0,1,3,…
- **ra_sw_physics**: 0,1,2,3,…
- **sf_sfclay_physics**: 0,1,2, …
- **sf_surface_physics**: 0,1,2,3,… (set before running **real** or **ideal**, need to match with **num_soil_layers** variable)
  - sf_urban_physics: 0, 1, 2, 3
- **bl_pbl_physics**: 0,1,2,…
- **cu_physics**: 0,1,2,3,…

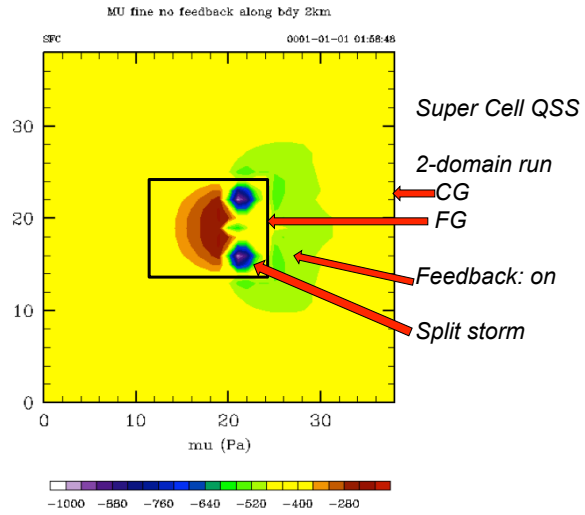
End

Nesting in WRF

Dave Gill

Nesting in WRF

Dave Gill



Overview

- Nesting: Journalism 101: Who, what, why, when, where
- Domains
  - OK *vs* semi-OK *vs* not OK at all
  - Variable staggering CG to FG
  - Lateral forcing
  - Feedback
  - Masked interpolation
  - Time stepping for multi-domain
- Concurrent *vs* Offline Nesting
- Registry
  - U D F S
  - i2
- Some suggestions
  - Performance
  - Location, location, location
  - Inside out, start with inner domain
  - Go big or go home
  - Map factors, stability, time step

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

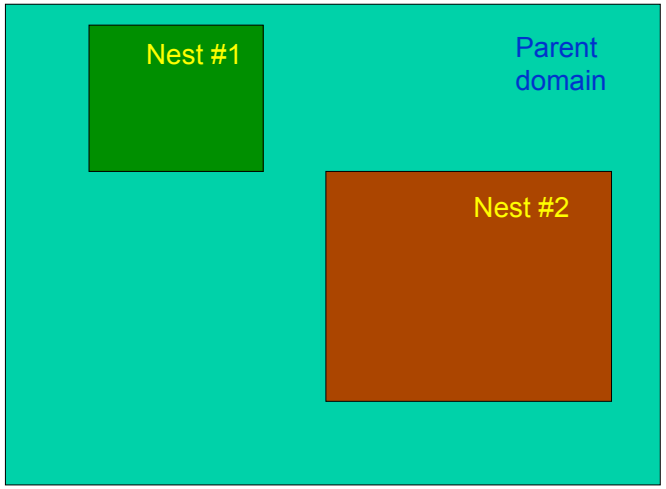
Nesting Basics

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

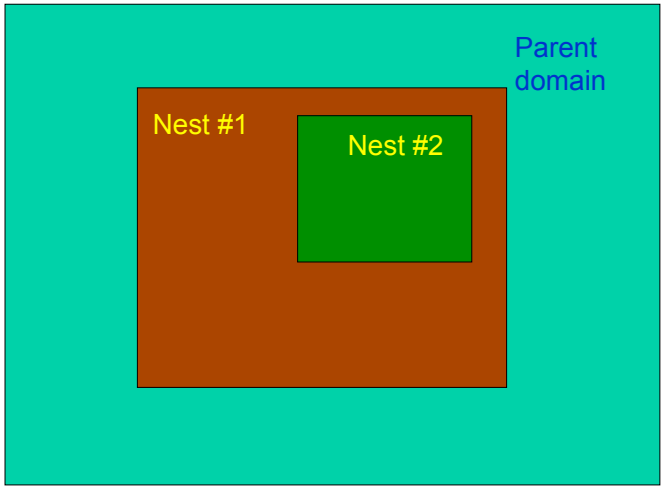
- One-way/two-way nesting with a single input file, all fields interpolated from the coarse grid
- One-way/two-way nesting with multiple input files, each domain with a full input data file
- One-way/two-way nesting with the coarse grid data including all meteorological fields, and the fine-grid domains 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 (usually) 700 hPa low tracking

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

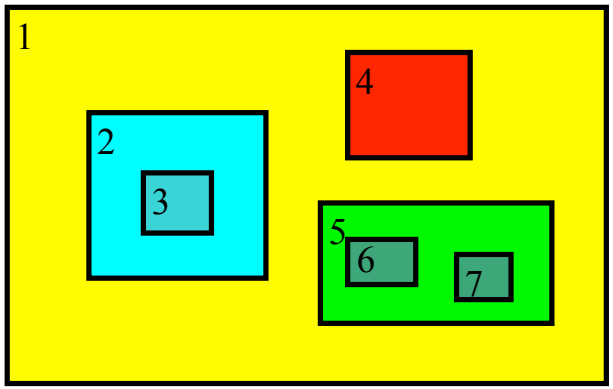


Two levels of nests, with nest #1 acting as the parent for nest #2



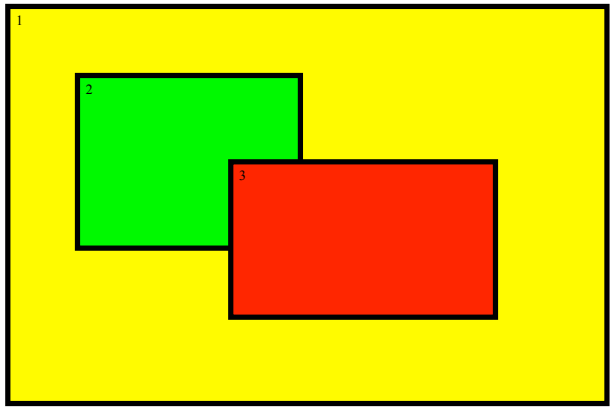
These are all OK

Telescoped to any depth
Any number of siblings

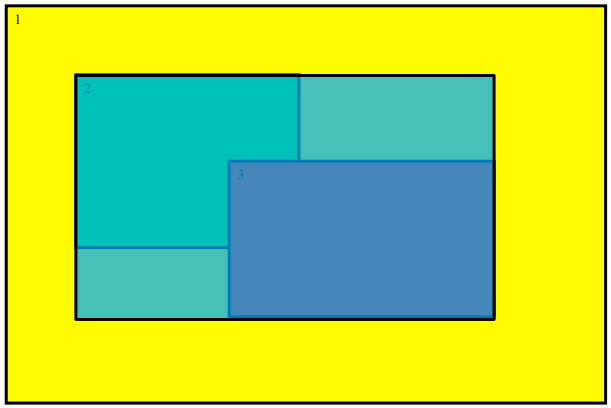


Not OK for 2-way

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

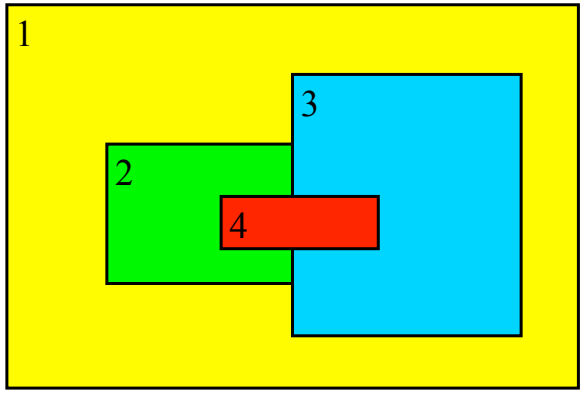


Preferred – One big Nest



Not OK either

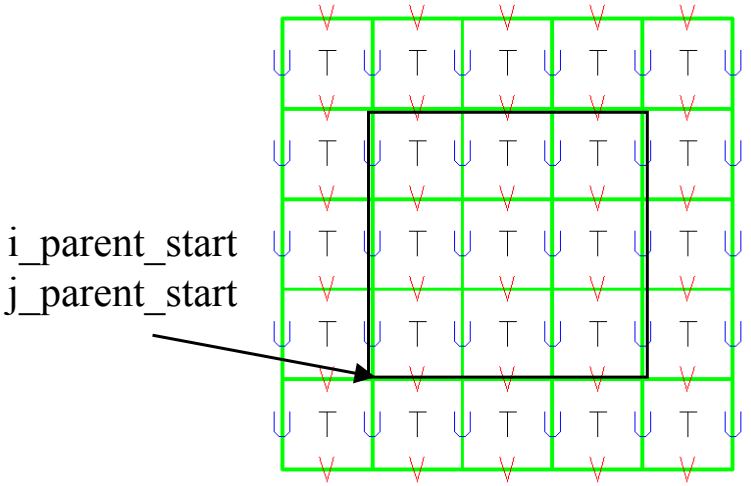
Domains have one, and only one, parent -
(domain 4 is NOT acceptable even with 1-way nesting)



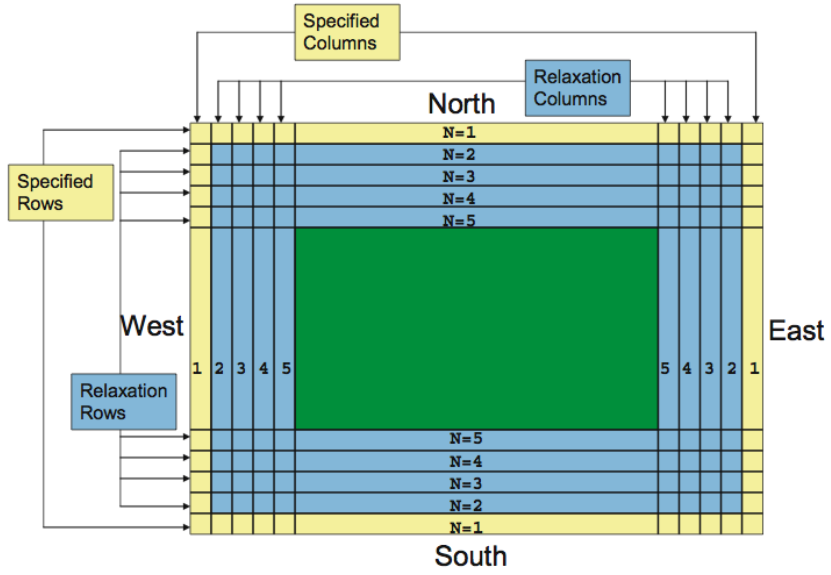
WRF Coarse-Fine Overlap

- The rectangular fine grid is coincident with a portion of the high-resolution grid that covers the entire coarse grid cell
- 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.

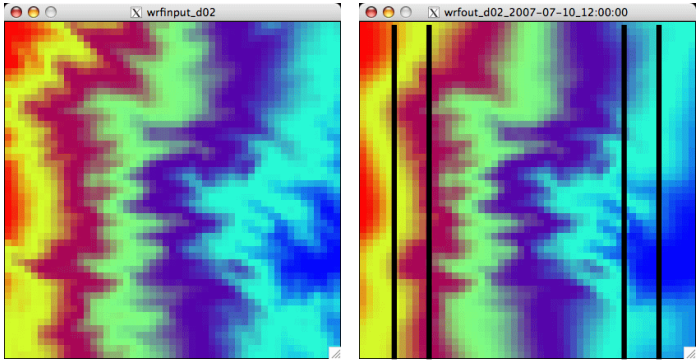
Coarse Grid Staggering



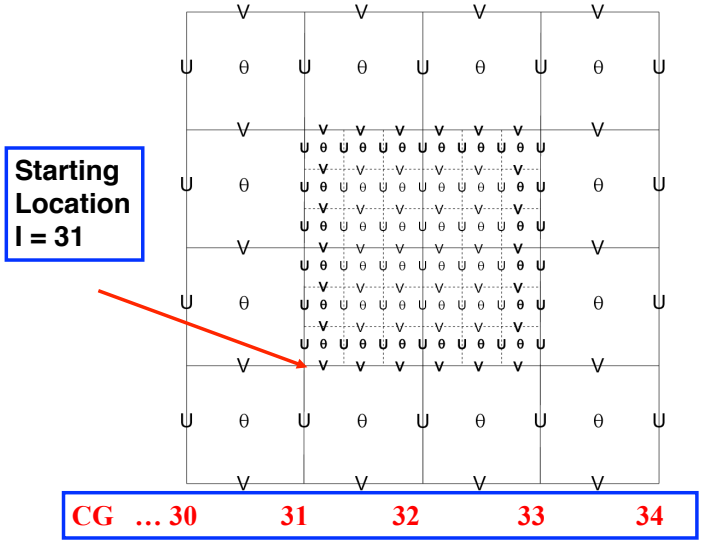
Real-Data Lateral Boundary Condition: Location of Specified and Relaxation Zones



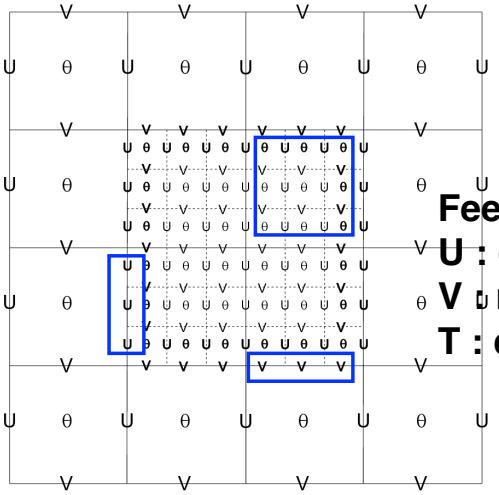
Lateral Smoothing



Coarse Grid Staggering 3:1 Ratio



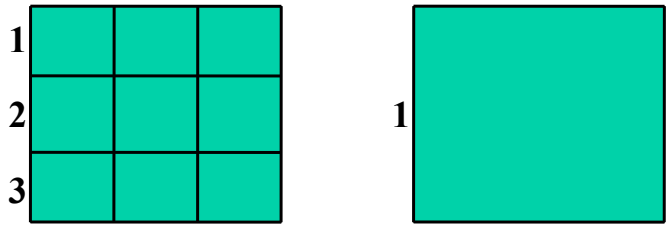
Coarse Grid Staggering 3:1 Ratio



Feedback:
U : column
V : row
T : cell

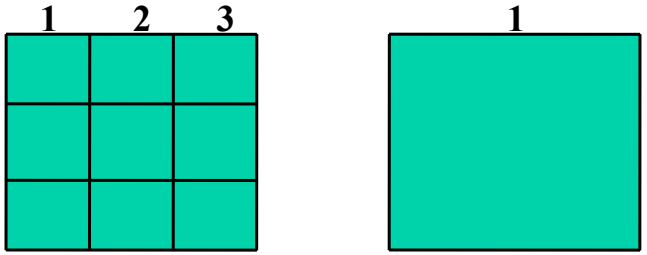
Coarse Grid Staggering 3:1 Ratio

Feedback:
U : column
V : row
T : cell



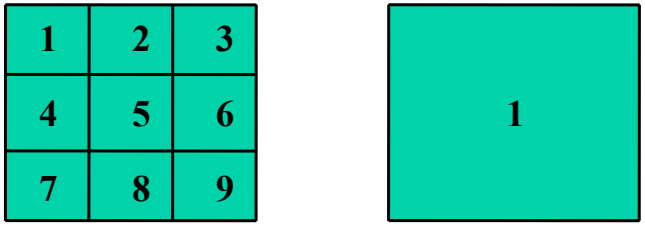
Coarse Grid Staggering 3:1 Ratio

Feedback:
U : column
V : row
T : cell

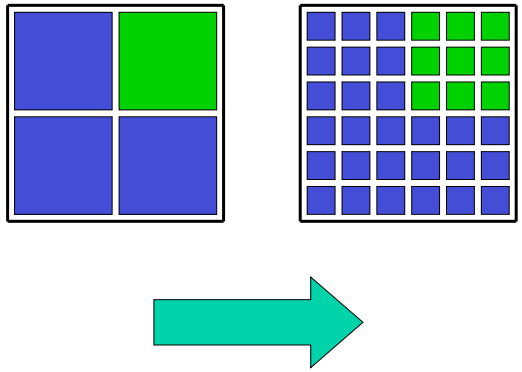


Coarse Grid Staggering 3:1 Ratio

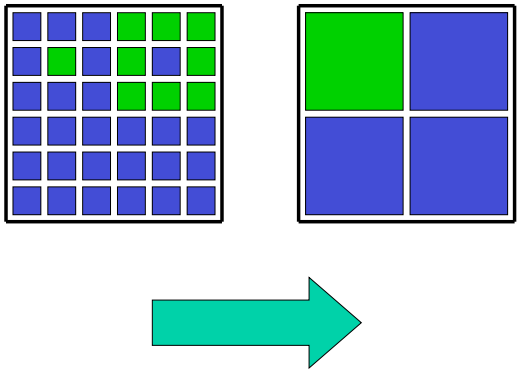
Feedback:
U : column
V : row
T : cell



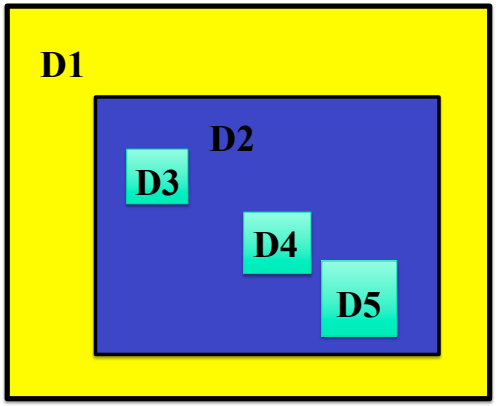
Masked Interpolation



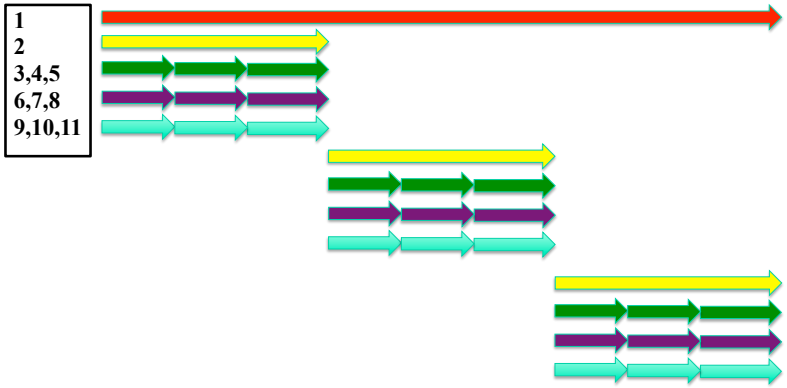
Masked Feedback



WRF 5-domain run: Domain 1 (a single 3 min dt), then Domain 2 (a single 1 min dt). Then Domain 3, in 20 s pieces up to 1 min. Then Domain 4, in 20 s pieces up to 1 min, and same with Domain 5.



WRF 5-domain run: Domain 1 (a single 3 min dt), then Domain 2 (a single 1 min dt). Then Domain 3, in 20 s pieces up to 1 min. Then Domain 4, in 20 s pieces up to 1 min, and same with Domain 5.



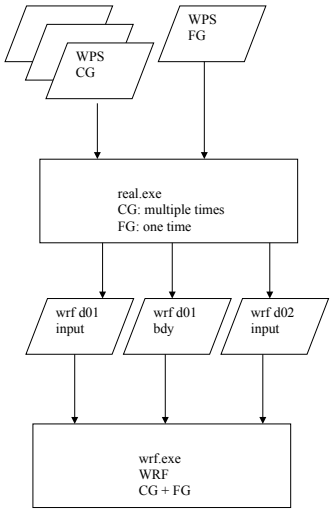
Concurrent Nesting with *n* Inputs

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



Concurrent Nesting with *n* Inputs

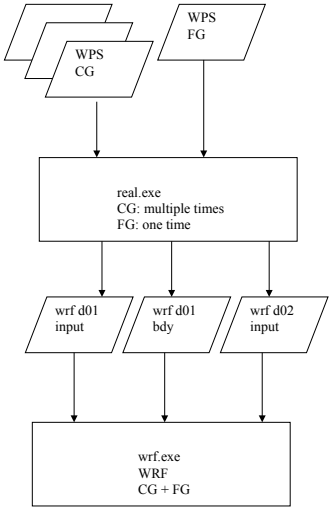
Restricted vertical nesting

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

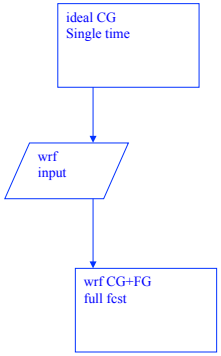
The grid distance ratio is not strictly tied to the time step ratio

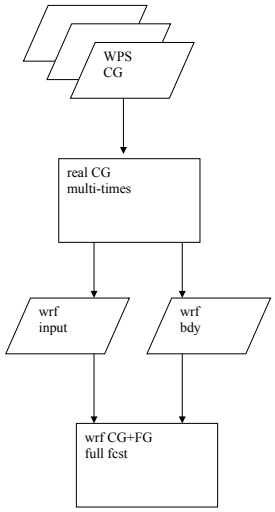
Topography smoothly ramps from coarse grid to the fine grid along the interface along the nest boundary

All fine grids must use the nested lateral boundary condition



Concurrent Nesting with 1 Input

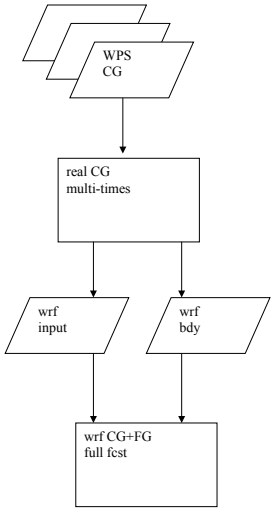




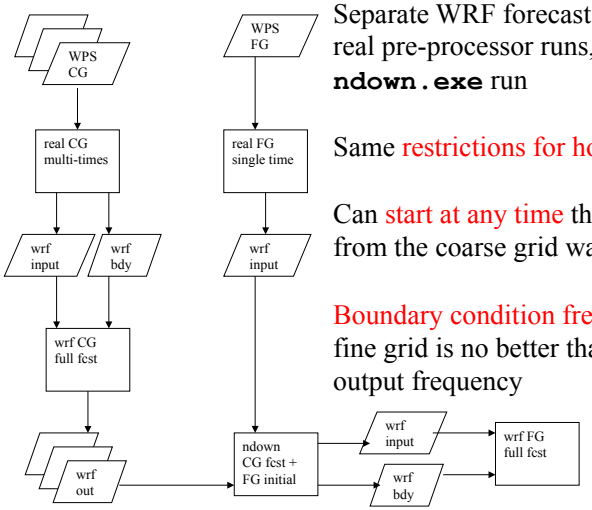
Concurrent Nesting with 1 Input

A single namelist column entry is tied to each domain

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



ndown: Offline Nesting



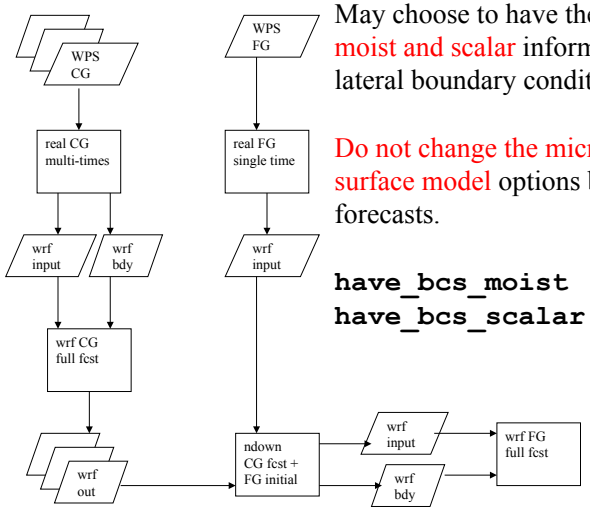
Separate WRF forecast runs, separate real pre-processor runs, intervening **`ndown.exe`** run

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 no better than the coarse grid output frequency

ndown: Offline Nesting

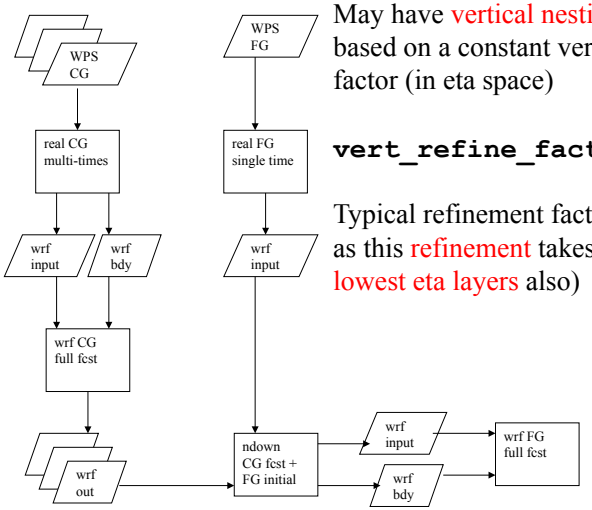


May choose to have the parent WRF model moist and scalar information used for the lateral boundary conditions.

Do not change the microphysics or land surface model options between the WRF forecasts.

have_bcs_moist
have_bcs_scalar

ndown: Offline Nesting



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

vert_refine_fact

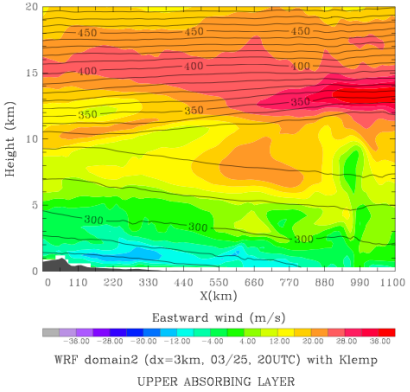
Typical refinement factors 2-5 (be careful, as this refinement takes place in the lowest eta layers also)

West East Cross section

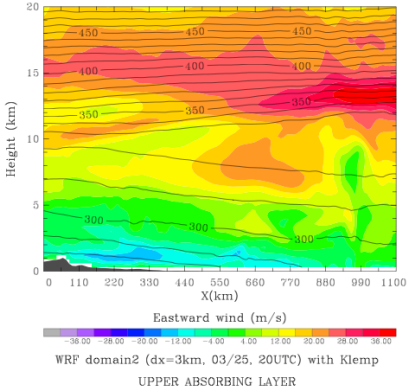
Shaded: v; Contour: theta

6-h Forecast, from Mohamed Moustaoui

Standard Levels



3x Refinement



Vertical Nesting

Starting with V3.7, the model allows different numbers of vertical levels on different domains. Usually, this is to include more levels on the inner domains.

This is a new feature and needs to be handled cautiously.

Restrictions:

- Only RRTM and RRTMG radiation schemes are available
- Real data cases only
- Static nest locations only

Vertical Nesting

```
&domains
max_dom             = 2,
e_vert              = 35,    45,
vert_refine_method  = 0,     2,
```

Vertical Nesting

```
eta_levels( 1:35) = 1., 0.993, 0.983, 0.97, 0.954, 0.934,
                    0.909, 0.88, 0.8406663, 0.8013327,
                    0.761999, 0.7226653, 0.6525755,
                    0.5877361, 0.5278192, 0.472514,
                    0.4215262, 0.3745775, 0.3314044,
                    0.2917579, 0.2554026, 0.2221162,
                    0.1916888, 0.1639222, 0.1386297,
                    0.1156351, 0.09525016, 0.07733481,
                    0.06158983, 0.04775231, 0.03559115,
                    0.02490328, 0.0155102, 0.007255059, 0.
```

Vertical Nesting

```
eta_levels(36:81) = 1.0000, 0.9946, 0.9875, 0.9789, 0.9685,
                    0.9562, 0.9413, 0.9238, 0.9037, 0.8813,
                    0.8514, 0.8210, 0.7906, 0.7602, 0.7298,
                    0.6812, 0.6290, 0.5796, 0.5333, 0.4901,
                    0.4493, 0.4109, 0.3746, 0.3412, 0.3098,
                    0.2802, 0.2524, 0.2267, 0.2028, 0.1803,
                    0.1593, 0.1398, 0.1219, 0.1054, 0.0904,
                    0.0766, 0.0645, 0.0534, 0.0433, 0.0341,
                    0.0259, 0.0185, 0.0118, 0.0056, 0.
```

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)

Called at beginning of each set of child time steps, has parent and child information available – could be used with SST.

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

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 – **smoothing can always be turned off without introducing any problems**

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" ""
```

Nesting Suggestions

- 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 Suggestions

- Example: assume 3:1 nest ratio

If the nest has the same number of grid cells, then the **amount of CPU** to do a single time step for a coarse grid (CG) and a fine grid step (FG) is **approximately the same**.

Since the fine grid (3:1 ratio) has 1/3 the grid distance, it requires 1/3 the model time step. Therefore, the FG requires 3x the CPU to catch up with the CG domain.

Nesting Suggestions

- Example: assume 3:1 nest ratio

If you try to cover the SAME area with a FG domain as a CG domain, you need **(ratio)^2 grid points**.

With the associated FG time step ratio, you require a **(ratio)^3**.

With a 3:1 ratio, a FG domain covering the same area as a CG domain **requires 27x CPU**.

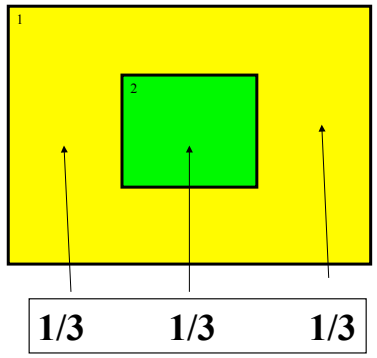
Nesting Suggestions

- Example: assume **10:1 nest ratio**

To change your test case from 50-km resolution to a finer 5-km resolution would be **1000x more** expensive.

Nesting Suggestions

- 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



Nesting Suggestions

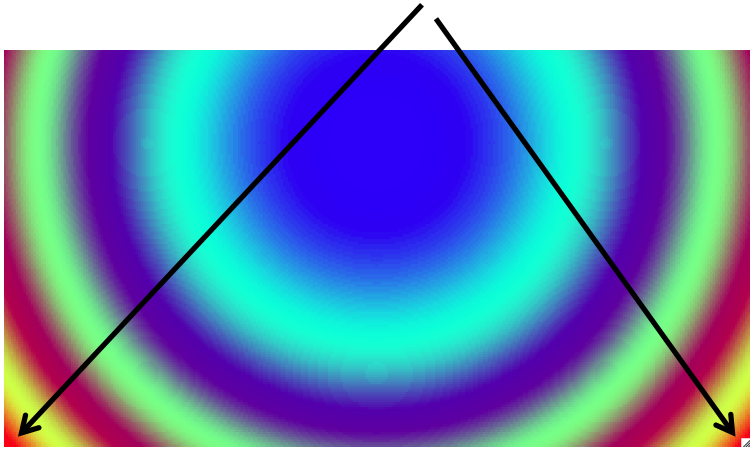
- Start with designing your inner-most domain. For a traditional forecast, you want everything important for that forecast to be entirely contained inside the domain.
- Then start adding parent domains at a 3:1 or 5:1 ratio. A parent should not have a smaller size (in grid points). Keep adding domains until the most coarse WRF grid has a no more than a 3:1 to 5:1 ratio to the external model (first guess) data.

Nesting Suggestions

- Larger domains tend to be better than smaller domains.
- A 60 m/s parcel moves at > 200 km/h. A 2-km resolution grid with 100x100 grid points could have most of the upper-level initial data swept out of the domain within a couple of hours.

Nesting Suggestions

Map factors > 1.6



Nesting Suggestions

- The most-coarse domain may have a geographic extent that causes large map factors.

```
time_step                = 300 (BLOWS UP)
dx                       = 45000,15000,5000
grid_id                  = 1,    ,2     ,3
parent_id                = 0,    ,1     ,2
parent_grid_ratio        = 1,    ,3     ,3
parent_time_step_ratio   = 1,    ,3     ,3
```

Nesting Suggestions

- Reducing the time step so that the coarse grid is stable makes the model too expensive. 1.6x

```
time_step                = 180 (STABLE, PRICEY)
dx                       = 45000,15000,5000
grid_id                  = 1,    ,2     ,3
parent_id                = 0,    ,1     ,2
parent_grid_ratio        = 1,    ,3     ,3
parent_time_step_ratio   = 1,    ,3     ,3
```

Nesting Suggestions

- Only reduce the time step on the coarse grid, and keep the fine grid time steps at their approx original values.

```
time_step                = 180 (STABLE, CHEAP)
dx                       = 45000,15000,5000
grid_id                  = 1,    ,2     ,3
parent_id                = 0,    ,1     ,2
parent_grid_ratio        = 1,    ,3     ,3
parent_time_step_ratio   = 1,    ,2     ,3
```

Nesting Suggestions

- Model time step is always proportional to the time step of the most coarse grid.
- The coarse grid is the only grid impacted with large map factors: **`dt(s) = 6*dx(km)`** but the nominal grid distance needs to be scaled:

  dt(s) = 6*dx(km) / MAX (map factor in domain)
- Reducing the coarse grid time step does not significantly reduce model performance if you can tweak the time step ratio.

Nesting Suggestions

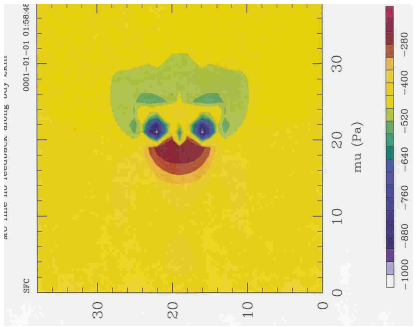
- The time step ratio and grid distance ratio are not necessarily identical, and may used effectively when large map factors in the coarse grid domain force a time step reduction for stability.
- If map factors are causing stability troubles, it is usually only the most coarse grid that is impacted since the fine grid is usually in the middle of the domain.

Nesting Suggestions

- Set up domain first to provide good valid forecast, then deal with efficiency
- Selecting a set of domains with the reason "it is all I can afford" gets you into trouble
- Numerically stable and computationally expedient do not imply scientifically or physically valid

Review

- Nesting: Journalism 101: Who, what, why, when, where
- Domains
  - OK *vs* semi-OK *vs* not OK
  - Variable staggering CG to FG
  - Lateral forcing
  - Feedback
  - Masked interpolation
  - Time stepping for multi-domain
- Concurrent *vs* Offline Nesting
- Registry
  - U D F S
  - i2
- Some suggestions
  - Performance
  - Location, location, location
  - Inside out, start with inner domain
  - Go big or go home
  - Map factors, stability, time step



Nesting PART 2

- Nesting steps inside of WRF
- Available source code options
- Choosing the nested interpolation type
- Building automatically accessed routines

Nesting Sequence Inside of WRF

- The WRF model always has the parent domain integrate a single time step, then the code checks to see if a child domain exists (valid time)
- The parent has current (t+dt) information stored in the _2 variables and the information from the previous time step stored in the _1 variables (for example t_1, t_2, etc).
- These two time levels of data allow the lateral boundary conditions for the fine grid to be handled similarly to that of the coarse grid: an initial value (the old time, _1) and a tendency to get to the next time are required

Nesting Sequence Inside of WRF

- The initial value and the tendency from the parent domain are horizontally interpolated onto the child domain
- For a nest ratio of 3:1, then three child time steps are required to get to the parent current time. The tendency during these three child time steps along the lateral boundaries remains constant
- At the end of the last child time step required to get to the parent's current time, for a two-way nest, the child information feedsback to the parent domain

Available Source Code Options

- The nesting inside of WRF requires a few types of routines:
  - Horizontally interpolate the parent to the child
  - Generate the lateral boundary conditions for the child
  - Feed information from the child back to the parent
  - Optionally smooth the area in the parent covered by the child domain after feedback
- All of these options are selected through the Registry

Available Source Code Options

- In the Registry, the nesting options are located with the I/O flags

```
Latitude: du=(copy_fcnm)
U: usdf=(bdy_interp:dt)
TSK: d=(interp_mask_field:lu_index,iswater)u=(copy_fcnm)
LANDMASK: d=(interp_fcnm_imask)u=(copy_fcnm)
SST: d=(interp_mask_field:lu_index,iswater)
```

- but could be as complicated as

```
SST: d=(interp_mask_field:lu_index,iswater)\
f=(p2c_mask:lu_index,tslb,num_soil_layers,iswater)
```

Available Source Code Options

- The syntax for horizontal interpolation from the parent to the child is "d" for "down"

```
d=(subroutine_name: optional arguments, comma separated)
```

- Default is interp_fcn
- The "d" option is handled only once per domain, at initialization

Available Source Code Options

- The syntax for feedback from the child back to the parent is "u" for "up"

```
d=(subroutine_name: optional arguments, comma separated)
```

- Default is copy_fcn
- The "u" is processed in the WRF model after the last required fine grid time step brings the child domain up to the same time as the parent

Available Source Code Options

- The syntax for the lateral boundary tendency computation is "f" for LBC "forcing"

```
f=(subroutine_name: optional arguments – typically time step)
```

- Default is interp_bdy (but specified because the time step argument is always used)
- The lateral boundary condition shares the "f" option. Any domain that would like to have the child domain given information at the end of each parent time step (such as lateral boundaries), may use the "f=()" Registry option. Some developers have subroutines that interpolate a child domain from the parent at EACH parent time step (SSTs and perturbations from SKEBS are examples)

Choosing the Nested Interpolation Type

- At run-time, the user may select the order of the horizontal interpolation to be used

```
&domains
interp_method_type = 1: bilinear
                     2: sint
                     3: nearest neighbor
                     4: quadratic
/
```

- The same order/type of interpolator is used the initial horizontal interpolation and the subsequent lateral boundary interpolation

Building Automatically Accessed Routines

- The registry program manufactures a default template for the subroutine call.

```
SUBROUTINE interp_fcn                          &
   ( cfld,                                     &
     cids, cide, ckds, ckde, cjds, cjde,       &
     cims, cime, ckms, ckme, cjms, cjme,       &
     cits, cite, ckts, ckte, cjts, cjte,       &
     nfld,                                     &
     nids, nide, nkds, nkde, njds, njde,       &
     nims, nime, nkms, nkme, njms, njme,       &
     nits, nite, nkts, nkte, njts, njte,       &
     shw,                                      &
     imask,                                    &
     xstag, ystag,                             &
     ipos, jpos,                               &
     nri, nrj                                  )
```

Building Automatically Accessed Routines

- The lateral boundary routines (the "f=()" option) always get the eight boundary arrays appended (total of 16 arrays, 8 for parent, 8 for child).
- Any extra variables are ALWAYS tagged on to the end of the subroutine, and always in pairs: parent and child (for example: time step, land mask, etc).
- The user may place the new routine (called by the name given in the Registry file) in the interp_fcn.F file

Nesting PART 2

- Nesting steps inside of WRF
- Available source code options
- Choosing the nested interpolation type
- Building automatically accessed routines

WRF Nudging

Jimy Dudhia

WRF Four-Dimensional Data Assimilation (FDDA)

Jimy Dudhia

FDDA

ARW only

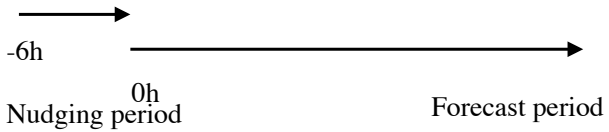
- Method of nudging model towards observations or analysis
- May be used for
  - Dynamical initialization (pre-forecast period)
  - Creating 4D meteorological datasets (e.g. for air quality model)
  - Boundary conditions (outer domain nudged towards analysis)

Method

- Model is run with extra nudging terms for horizontal winds, temperature and water vapor
- In analysis nudging, these terms nudge point-by-point to a 3d space- and time-interpolated analysis field
- In obs-nudging, points near observations are nudged based on model error at obs site
- The nudging is a relaxation term with a user-defined time scale around an hour or more
- Nudging will work with nesting and restarts

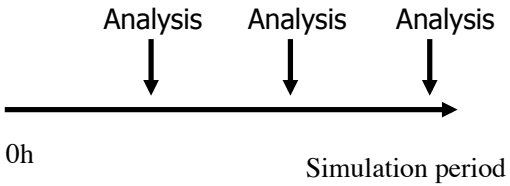
Dynamic Initialization

- Model domains are nudged towards analysis in a pre-forecast period of 6-12 hours
- This has benefit of smooth start up at forecast time zero



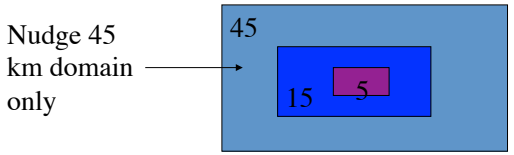
Four-Dimensional Met Analysis

- Produces analyses between normal analysis times
- High-resolution balanced and mass-continuity winds can be output to drive off-line air quality models



Boundary Conditions

- Nudge an outer domain towards analysis through forecast
- This has benefit of providing smoother boundary conditions to domain of interest than if 15 km domain is the outer domain with interpolated-analysis boundary conditions



FDDA Methods

- Two Methods
  - Grid or analysis nudging (suitable for coarse resolution)
  - Observation or station nudging (suitable for fine-scale or asynoptic obs)
- Nudging can be applied to winds, temperature, and water vapor

Note: nudging terms are fake sources, so avoid FDDA use in dynamics or budget studies

Analysis Nudging (grid_fdda=1)

- Each grid-point is nudged towards a value that is time-interpolated from analyses

$$\frac{\partial p^* \alpha}{\partial t} = F(\alpha, \mathbf{x}, t) + G_\alpha \cdot W_\alpha \cdot \epsilon_\alpha(\mathbf{x}) \cdot p^*(\hat{\alpha}_0 - \alpha)$$

In WRF p* is $\mu\upsilon$ and α is u,v,T or q

F includes all the regular WRF terms

Analysis Nudging

$$\frac{\partial p^* \alpha}{\partial t} = F(\alpha, \mathbf{x}, t) + G_\alpha \cdot W_\alpha \cdot \epsilon_\alpha(\mathbf{x}) \cdot p^*(\hat{\alpha}_0 - \alpha)$$

- G is nudging inverse time scale
- W is vertical weight (upper air and surface)
- ε is a horizontal weight for obs density (not implemented yet)

Analysis Nudging

- 3d analysis nudging uses the WRF input fields at multiple times that are put in wrffdda_d01 file by program real when run with grid_fdda=1
  - With low time-resolution analyses, it is recommended not to use 3d grid-nudging in the boundary layer, especially for temperature
- Surface (2d) analysis nudging available in Version 3.1
  - Nudges surface and boundary layer only

Analysis-Nudging namelist options

Can choose

- Frequency of nudging calculations (fgdt in minutes)
- Nudging time scale for each variable (guv, gt, gq in inverse seconds)
- Which variables not to nudge in the PBL (if_no_pbl_nudging_uv, etc.)
- Model level for each variable below which nudging is turned off (if_zfac_uv, k_zfac_uv, etc.)
- Ramping period over which nudging is turned off gradually (if_ramping, dt_ramp_min)

Surface Analysis Nudging

- In Version 3.1 added 2d (surface) nudging (grid_fdda=1 and grid_sfdda=1) for surface analyses
  - wrfsfdda_d01 file created by obsgrid.exe
  - Weights given by guv_sfc, gt_sfc, and gq_sfc
  - Note: grid_fdda=1 must be used to activate this. If upper-air nudging not wanted, set upper weights guv, gt, gq =0.

Spectral Nudging

- In Version 3.1 added spectral nudging (grid_fdda=2) to do 3d nudging of only selected larger scales
- This may be useful for controlling longer wave phases for long simulations. Compensates for error due to low-frequency narrow lateral boundaries
  - Top wavenumber nudged is selected in namelist (xwavenum, ywavenum, e.g. =3)
    - Typically choose so that (domain size)/(wavenumber)=~1000 km in each direction
  - Nudges u, v, theta, geopotential (not q)
  - Can nudge in all levels or use ramp above a specified model level (if_zfac_ph, k_zfac_ph, dk_zfac_ph, etc.)

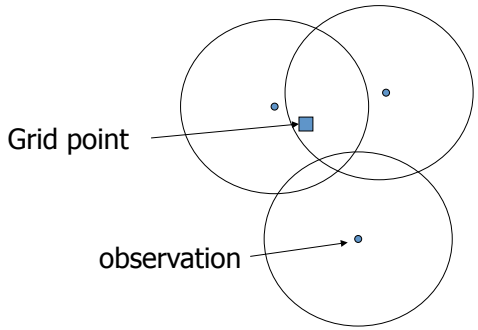
Obs Nudging (obs_nudge_opt=1)

- Each grid point is nudged using a weighted average of differences from observations within a radius of influence and time window

$$\frac{\partial p^* \alpha}{\partial t} = F(\alpha, \mathbf{x}, t) + G_\alpha \cdot p^* \frac{\sum_{i=1}^{N} W_i^2(\mathbf{x}, t) \cdot \gamma_i \cdot (\alpha_o - \hat{\alpha})_i}{\sum_{i=1}^{N} W_i(\mathbf{x}, t)}$$

$$W(\mathbf{x}, t) = w_{xy} \cdot w_\sigma \cdot w_t$$

Obs Nudging



Note: errors at obs sites are weighted by distance for nudging

Obs Nudging

$$w_{xy} = \frac{R^2 - D^2}{R^2 + D^2} \qquad 0 \le D \le R$$

$$w_{xy} = 0 \qquad D > R,$$

- R is radius of influence
- D is distance from ob modified by elevation difference

Obs Nudging

$$w_t = 1 \qquad |t - t_0| < \tau/2$$

$$w_t = \frac{\tau - |t - t_0|}{\tau/2} \qquad \tau/2 \leq |t - t_0| \leq \tau$$

- t is the specified time window for the obs
- This is a function that ramps up and down

Obs Nudging

- w_σ is the vertical weighting – usually the vertical influence is set small (0.005 eta-difference) so that data is only assimilated on its own eta level
- obs input file is a special ascii file (OBS_DOMAIN101) with obs sorted in chronological order
  - each record is the obs (u, v, T, Q) at a given model position and time
  - Utility programs exist to convert data to this format from other common formats
  - In V3.1 obsgrid.exe can create this file from standard observations that are in little_r format

Obs-Nudging namelist options

Can choose

- Frequency of nudging calculations (iobs_ionf)
- Nudging time scale for each variable (obs_coef_wind, etc.)
- Horizontal and vertical radius of influence (obs_rinxy, obs_rinsig)
- Time window (obs_twindo)
- Ramping period over which nudging is turned off gradually (obs_idynin, obs_dtramp)

Vertical weighting functions

- Added flexibility options for advanced usage of obs-nudging with surface observations (switches in run/README.namelist, e.g. obsnudgezfullr1_uv, etc.)
  - These allow specifying how variables are nudged in a profile with their full weight and/or ramp down function relative to the surface or PBL top in different regimes (stable or unstable).
  - Defaults are set to reasonable values, so these can be left out of namelist unless needed.

FDDA Summary

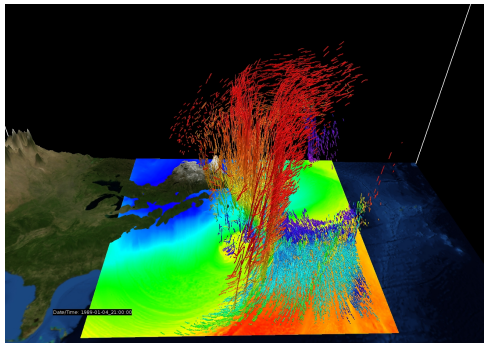
- FDDA grid nudging is suitable for coarser grid sizes where analysis can be better than model-produced fields
- Obs nudging can be used to assimilate asynoptic or high-frequency observations
- Grid and obs nudging can be combined
- FDDA has fake sources and sinks and so should not be used on the domain of interest and in the time period of interest for scientific studies and simulations

End

Post-processing Tools

Kelly Werner

Post-processing Tools



Kelly Keene

December 2015

Supported Post-processing Packages

Package	Users' Guide Page #	Information
NCL	9-2	Graphical package Supported by NCAR/CISL (wrfhelp@ucar.edu and ncl-talk@ucar.edu)
ARWpost	9-29	Converter (GrADS)
RIP4	9-20	Converter and interface to graphical Package, NCAR graphics
UPP	9-36	Converter (GrADS & GEMPAK)
VAPOR	9-38	Converter and graphical package Supported by VAPOR (vapor@ucar.edu)
IDV	None – see unidata.ucar.edu	GRIB (from UPP) GEMPAK (from wrf2gem) Vis5d CF compliant data (from wrf_to_cf) Supported by unidata (support@unidata.ucar.edu)
GEMPAK	None - see: unidata.ucar.edu/software/gempak	Data from wrf2gem or UPP Supported by unidata (support@unidata.ucar.edu)

Choosing the Right Tool

- Can it read your data?
- Will you need to pre-process the data first?
- Is it purely a visualization tool, or does it include post-processing?
- Can it handle big datasets?
- Which diagnostic/statistical functions does it have?
- 3D or 2D visualization?
- Can it handle staggered grids?
- How easy is it to add diagnostics?
- How is data below the ground handled?
- Vertical grids?
- How are model time stamps handled?
- Easy to use?
- Cost of package?
- How well supported is it?

Data Handling

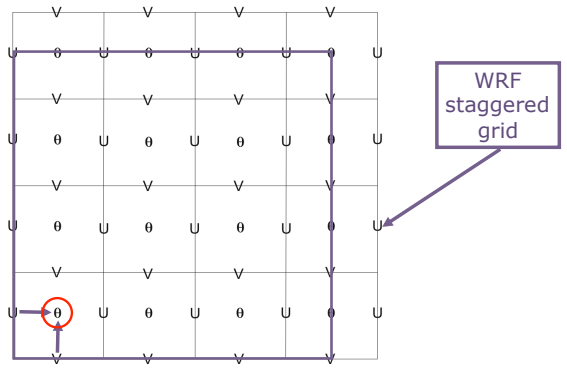
	NCL	RIP4	GrADS	UPP	VAPOR	IDV
netCDF		ripdp	ARWpost	converter	converter	converter
GRIB						
ASCII						
shapefiles						
geogrid & metgrid output						
intermediate file format	***V6.2.0 V6.3.0***					
wrfinput data						
Idealized data						
wrfoutput						
big data						

Post-processing

	NCL	RIP4	GrADS	UPP	VAPOR	IDV
Post-processing						
Data output						
3D						
diagnostics	some	a lot	some	some	limited	limited
Add diagnostics	Very easy	easy	easy	Relatively easy	Not as easy	Not as easy
Vertical output Coordinate	Model pressure height	Model pressure height	Model pressure height	pressure	model	model
Extrapolate Below ground						

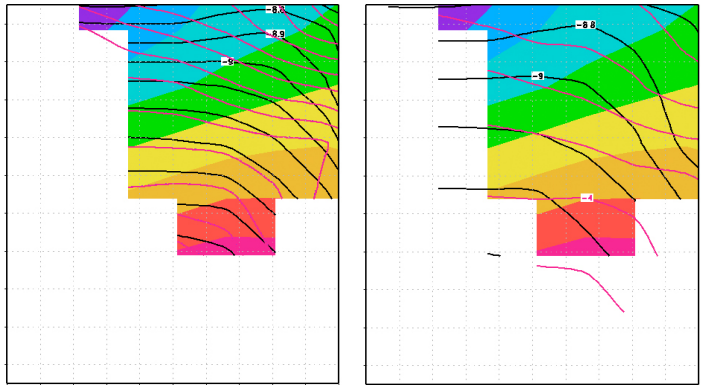
Model Staggering

Why is a converter necessary if a package can display netCDF files?



Model Staggering

Shaded = temp; black lines = U; red = V



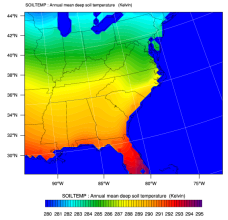
NCL

NCL: General Information

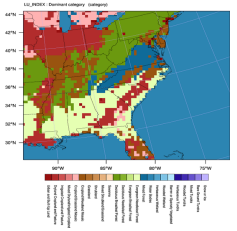
- NCAR Command Language (NCL)
- Website: http://www.ncl.ucar.edu
- Reads WRF-ARW data directly
- Can generate many types of graphical plots
  - Horizontal
  - Cross-section
  - SkewT
  - Meteogram
  - Panel

NCL: Example Plots

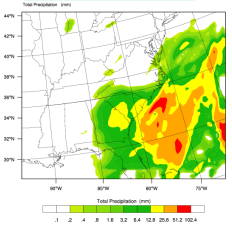
Soil Temperature



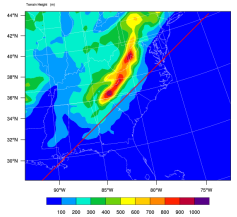
LU-INDEX



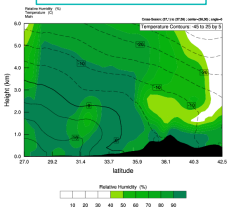
Total Precipitation



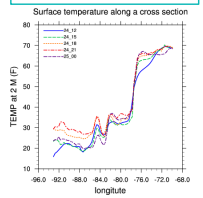
Terrain Height



Relative Humidity Vertical X-section

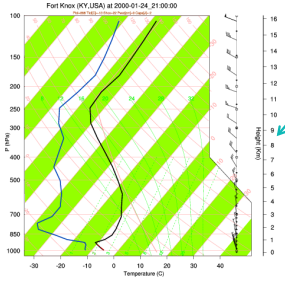


Sfc Temperature Horizontal X-section

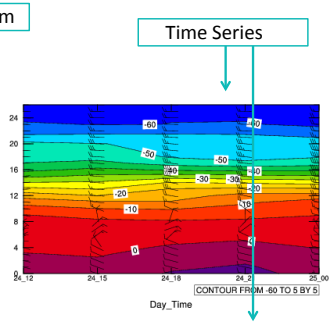


NCL: Example Plots

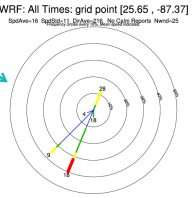
Skew-T Diagram

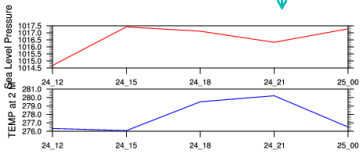


Time Series



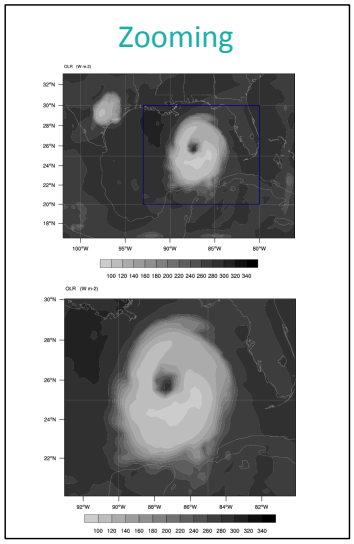
Wind Rose



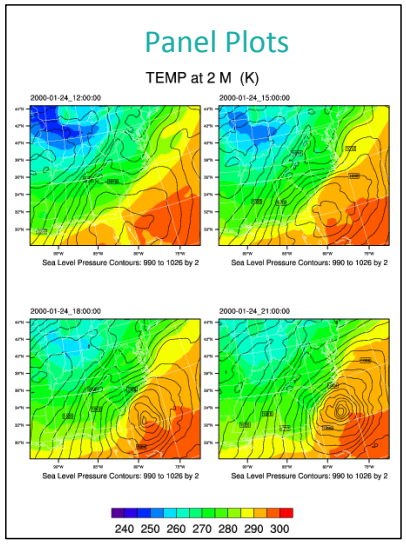


NCL: Example Plot Functions

Zooming



Panel Plots



NCL: Downloading

http://www.ncl.ucar.edu/Download

– Fill out short registration form (short waiting period)

– Read and agree to OSI-based license

– Get version 6 or LATER (current: v6.3.0)

****Always download binary code instead of source code****

NCL: Libraries

- Scripts
  - User modifiable – can alter for your own purpose
  - WRFUserARW.ncl – this script contains most functions, diagnostics, etc. used by the sample script
- Diagnostics
  - E.g., sea-level pressure
  - Written in Fortran and attached to code
- wrfhelp@ucar.edu : all questions about WRF and NCL
- ncl-talk@ucar.edu : generic NCL questions

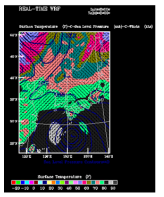
NCL: ~/.hluresfile

- Very important for NCL versions earlier than v6
  - http://www.ncl.ucar.edu/Document/Graphics/hlures.shtml
- Required by NCL libraries
- Must be placed in your home directory
- Will control:
  - Color table; font
  - White/black background
  - Size of plot
  - Characters

NCL: ~/.hluresfile

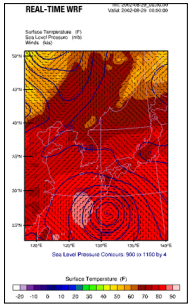
Without it:

```
PLCHHQ - CHARACTER NUMBER 26 (.) IS NOT A LEGAL FUNCTION CODE
PLCHHQ - CHARACTER NUMBER 29 (t) IS NOT A LEGAL FUNCTION CODE
PLCHHQ - CHARACTER NUMBER 30 (o) IS NOT A LEGAL FUNCTION CODE
PLCHHQ - CHARACTER NUMBER 32 (.) IS NOT A LEGAL FUNCTION CODE
PLCHHQ - CHARACTER NUMBER 35 (b) IS NOT A LEGAL FUNCTION CODE
PLCHHQ - CHARACTER NUMBER 36 (y) IS NOT A LEGAL FUNCTION CODE
```



With it:

```
*wkColorMap           : BlAqGrYeOrReVi200
*wkBackgroundColor    : white
*wkForegroundColor    : black
*FuncCode             : ~
*TextFuncCode         : ~
*Font                 : helvetica
*wkWidth              : 900
*wkHeight             : 900
```



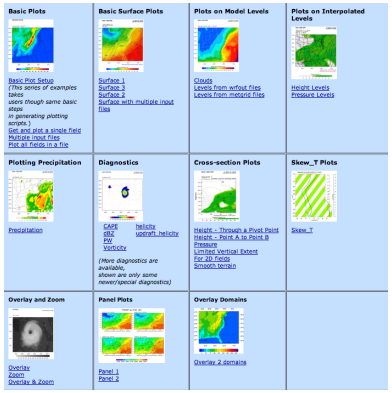
NCL: Generating Plots

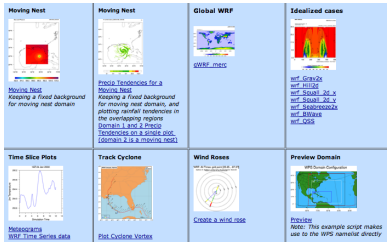
- Set NCARG_ROOT environment variable
  setenv NCARG_ROOT /usr/local/ncl ← **for example**
- Ensure you have a ~./hluresfile file
- Create a script
  - Start with a sample script
  - Most script routines are called from:
    http://www2.mmm.ucar.edu/wrf/OnLineTutorial/Graphics/NCL/NCL_examples.htm
- Run NCL script
  ncl your_script.ncl

NCL: Generating Plots

A good start: WRF Online Tutorial

http://www2.mmm.ucar.edu/wrf/OnLineTutorial/Graphics/NCL/index.html





NCL: Basic Set-up to Create a Plot

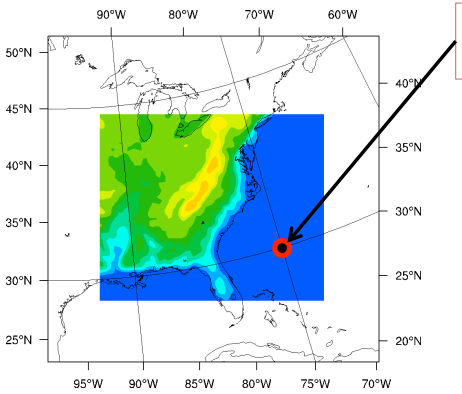
```
; load ncl library scripts
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
Load "$NCARG_ROOT/lib/ncarg/nclscripts/wrf/WRFUserARW.ncl"

Begin
     ;open input file(s)
     a = addfile("./wrfout_d01_2012-09-28_00:00:00.nc","r")
     ;open graphical output
     wks = gsn_open_wks("X11","plt_Surface")

     ;Read variables
     T2 = wrf_user_getvar(a,"T2",0)

     ;Set-up plot resources and create plots
     pltres = True       ;plotting resource, like overlays
     mpres = True        ;map resources - map resolution and zooming options
     opts = True         ;resources associated with each individual plot
     opts@cnFillOn = True
     contour_t2 = wrf_contour(a,wks,T2,opts)
     contour_slp = wrf_contour(a,wks,slp,opts)


     ;Output graphics
     plot = wrf_map_overlays(a,wks,(/contour_t2,contour_slp/),pltres,mpres)

end
```

NCL: Special WRF Functions

- **wrf_user_getvar**
  – *Get native and diagnostic variables*
- **wrf_contour / wrf_vector**
  – *Create line/shaded & vector plots*
- **wrf_map_overlays / wrf_overlays**
  – *Overlay plots created with wrf_contour and wrf_vector*
  – *X-section or idealized*
- **wrf_wps_read_int / wrf_wps_write_int**
  - For reading/writing in intermediate format

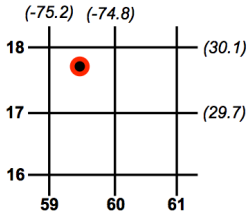
NCL: Special WRF Functions

- wrf_user_ll_to_ij / wrf_user_ij_to_ll
  - *Converts lat/lon coordinates to i/j coordinates (and vice versa)*




```
llres = True
llres@returnInt = False
locij = wrf_user_ll_to_ij(a, 30.0, -75.0, llres)
```

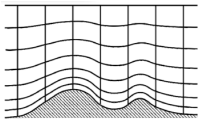
I = 59.45 ; J = 17.75



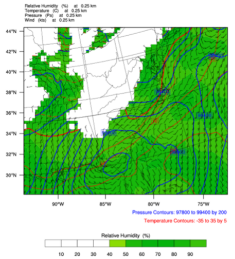
NCL: Special WRF Functions

- wrf_user_intrp3d / wrf_user_intrp2d
  - *Interpolates horizontally to a given pressure/height*
  - *3d data only – interpolates vertically along a given line*

- `rh_plane = wrf_user_intrp3d(rh,z,"h",250.,.0.,False)`
  - *Interpolates "rh" to "z" (height above mean sea level)*



```
Z_AGL = Z
dims = dimsizes(z)
do k=0,dims(0)-1
  z_AGL(k,:,:) = z_AGL(k,:,:) – ter(:,:)
end do
```



NCL: Special WRF Functions

- wrf_user_vert_interp

  NEW – in progress

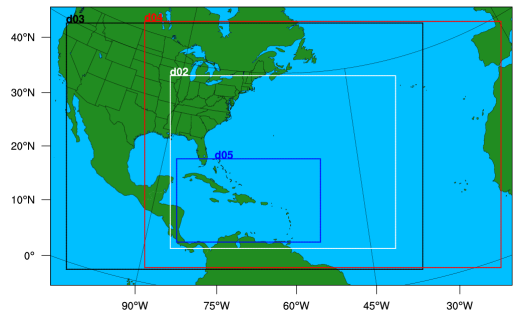
- Interpolates to:
  - "pressure","pres" – pressure [hPa]
  - "ght_msl" – grid point height msl [km]
  - "ght_agl" – grid point height agl [km]
  - "theta" – potential temperature [K]
  - "theta-e" – equivalent potential temperature [K]

- Extrapolates below the ground

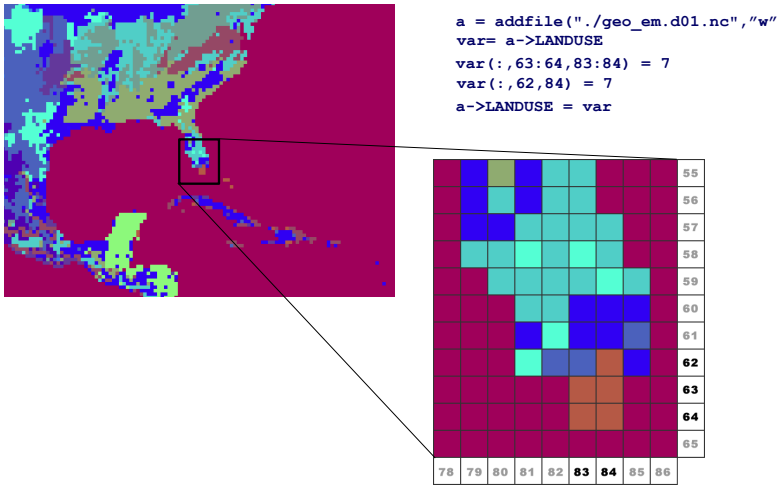
NCL: Domain Design

- plotgrids.ncl – can be found in WPS/util/ directory
  - *Reads namelist.wps and plots domains*
  - *Can modify code to add/change things specific to your plot*

Test Domain



NCL: Change Fields in netCDF File



NCL: Linking Fortran/C Code

- Link Fortran/C code to NCL scripts
  - Create a library from your Fortran/C code
  - Link to NCL script
- Link low-level NCL (NCAR Graphics) to Fortran code
  - Add calls to code inside Fortran code
  - Compile Fortran code with NCL libraries

NCL: Resources

- The special WRF functions have unique resources, found at:
http://www2.mmm.ucar.edu/wrf/OnLineTutorial/Graphics/NCL/NCL_functions.htm

- All general NCL resources can also be used to control the plot:
http://www.ncl.ucar.edu/Document/Graphics/Resources

am (anotation manager)
app (app)
ca (coordinate array)
cn (contour)
ct (coordinate array table)
dc (data comm)
err (error)
gs (graphic style)
gsn (gsn high-level interfaces)
lb (label bar)
lg (legends)
mp (maps)
pm (plot manager)
pr (primitives)
sf (scalar field)
st (streamline)
tf (transformation)
ti (title)
tm (tickmark)
tf (irregular transformation)
tx (text)
vc (vectors)
vf (vector fields)
vp (view port)
wk (workstation)
ws (workspace)
xy (xy plots)

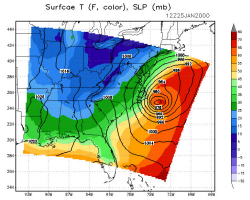
ARWpost

ARWpost: General Information

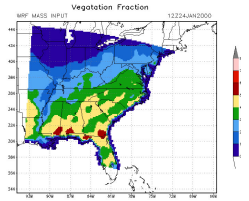
- Converter
  - Reads in wrf-arw model data, creates GrADS output files
  - Requires GrADS to display
- GrADS software is only needed to display data, not needed to compile the code
  - http://www.iges.org/grads/grads.html
- Generate a number of graphical plots
  - Horizonal
  - Cross-section
  - skewT
  - Meteogram
  - Panel
- Download Code
  - http://www2.mmm.ucar.edu/wrf/users/download/get_sources.html
- Online Tutorial
  - http://www2.mmm.ucar.edu/wrf/users/graphics/ARWpost/ARWpost.htm

ARWpost: Example Plots

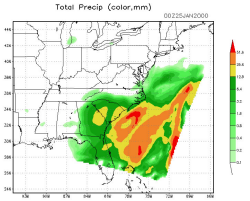
Surface Temp



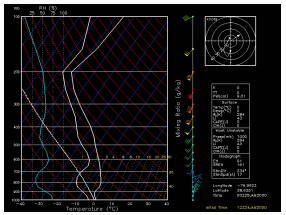
Vegetation Fraction



Total Precipitation

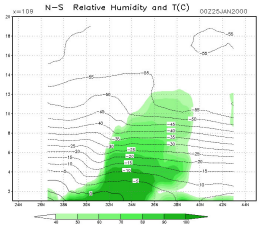


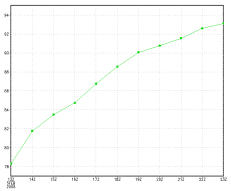
Skew-T Diagram



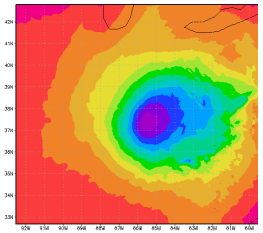
ARWpost: Example Plots

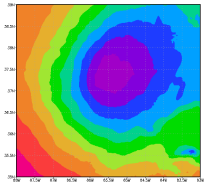
Cross-Sections





Zooming





ARWpost: Diagnostics

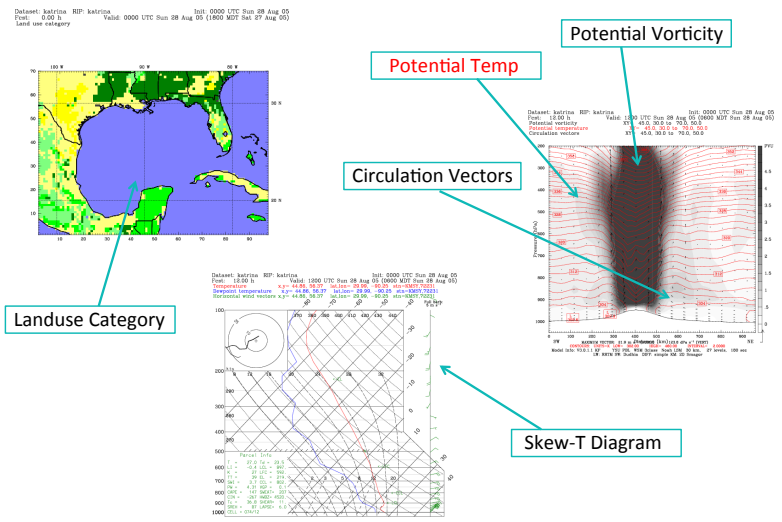
- cape – 3d cape
- cin – 3d cin
- mcape – maximum cape
- mcin – minimum cin
- clfr – low/middle/high cloud fraction
- dbz – 3d reflectivity
- max_dbz – maximum reflectivity
- geopt – geopotential
- height – model height in km
- lcl – lifting condensation level
- lfc – level of free convection
- pressure – full model pressure in hPa
- rh – relative humididy
- rh2 – 2 m relative humidity
- theta – potential temperature
- tc – temperature in degrees C
- tk – temperature in degrees K
- td – dew point temperature in degrees C
- td2 – 2m dew point temperature in degrees C
- slp – sea level pressure
- umet & vmet – winds rotated to Earth coordinates
- u10m & v10m – 10 m winds rotated to Earth coordinates
- wdir – wind direction
- wspd – wind speed coordinates
- wd10 – 10 m wind direction
- ws10 – 10 m wind speed

ARWpost: Scripts

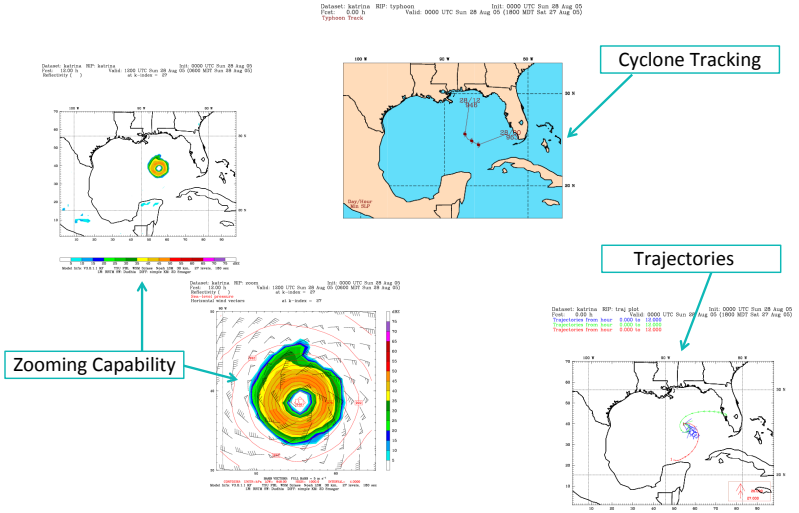
Script Name	Description
cbar.gs	Plots a color bar on shaded plots
rgbset.gs	Allows you to add/change colors from color # 20 – 99
skew.gs	Program to plot a skewT
plot_all.gs	Automatically finds all .ctl files in the directory and lists them so the user can pick when to use, will plot all fields chosen
rain.gs (real data only)	Plots total rainfall (must have data that contain fields RAINC and RAINNC
cross_z.gs (real data only)	Plots a NS and EW cross section of RH and T (C)

RIP4

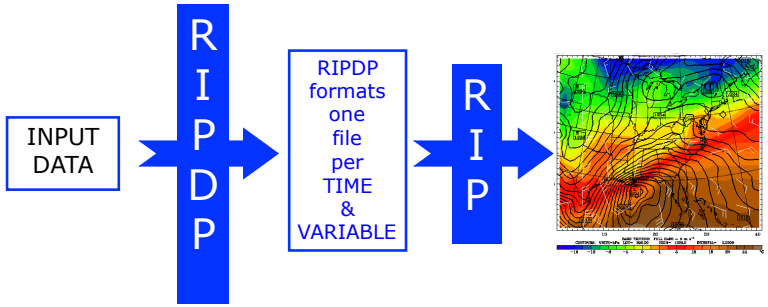
RIP4: Example Plots



RIP4: Example Plots



RIP4: Program Flow



RIP4: General Information

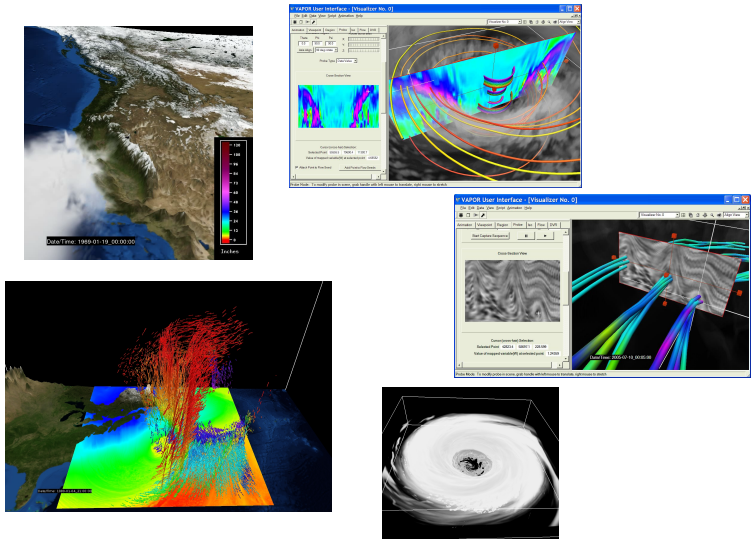
- Requires NCAR Graphics Libraries
  - http://www.ncl.ucar.edu
- Source Code
  - http://www2.mmm.ucar.edu/wrf/users/download/get_source.html
- Documentation
  - Included in program's tar file (in Doc/ directory)
  - http://www2.mmm.ucar.edu/wrf/users/docs/ripug.htm
- Online Tutorial
  - http://www2.mmm.ucar.edu/wrf/users/graphics/RIP4/RIP4.htm

VAPOR

VAPOR: General Information

- Visualization and Analysis Platform for Oceanic, atmospheric, and solar Research
- 3D visualization tool
- Developed and supported by:
  - Alan Norton and Colleagues (NCAR)
  - vapor@ucar.edu

VAPOR: Example Plots



VAPOR Installation

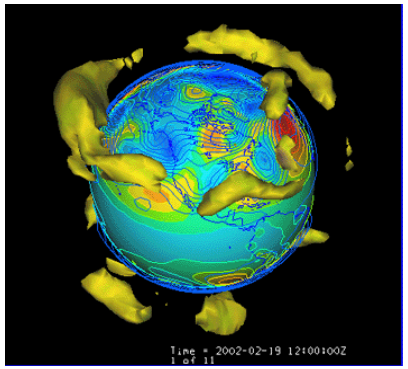
- Available for Linux, Windows, and Mac systems
- Need a reasonably modern graphics card
  - E.g., nVidia, ATI, AMD
- VAPOR website: http://www.vapor.ucar.edu
  - Download appropriate binary installer from VAPOR download page, follow installation instructions
  - *Note: You will need administrative privileges on a Mac
- On Linux and Mac, need to source vapor install.csh in your shell script before running any VAPOR commands

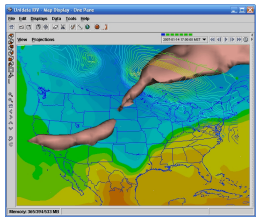
IDV

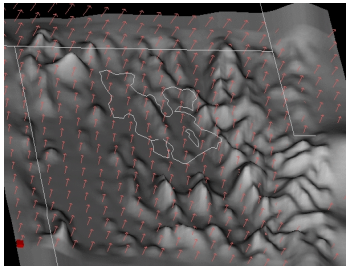
IDV: General Information

- Integrated Data Viewer
- 2D and 3D visualization tool
- Integrate model and observational data
- Visualize and analyze post-processed WRF output
- Developed and supported by:
  - Yuan Ho and Julien Chastang (NCAR)
  - support@unidata.ucar.edu

IDV: Example Plots







IDV: Downloading

- https://www.unidata.ucar.edu/software/idv
  - Point-and-click installers
  - Windows (.exe), Mac (.dmg), and Linux (.sh) installers available for both 32 and 64-bit systems
- System requirements
  - 2+ GB of Ram
  - Java 1.6
  - *Latest* video card driver

NCO Tools

NCO Tools

http://nco.sourceforge.net

- ncdiff
  - Shows the differences between 2 files

```
ncdiff input2.nc input2.nc output.nc
```

- ncrcat (nc cat)
  - Writes specified variables/times to a new file

```
ncrcat -v RAINNC wrfout* RAINNC.nc
ncrcat -d Time,0,231 -v RAINNC wrfout* RAINNC.nc
```

- ncra (nc average)
  - Averages variables and writes to a new file

```
ncra -v OLR wrfout* -o OLR.nc
```

- ncks (nc kitchen sink)
  - Combination of all NCO tools in 1
  - Specifically nice for splitting files

```
ncks -d Time,1,1 wrfout -o wrfout1.nc
```

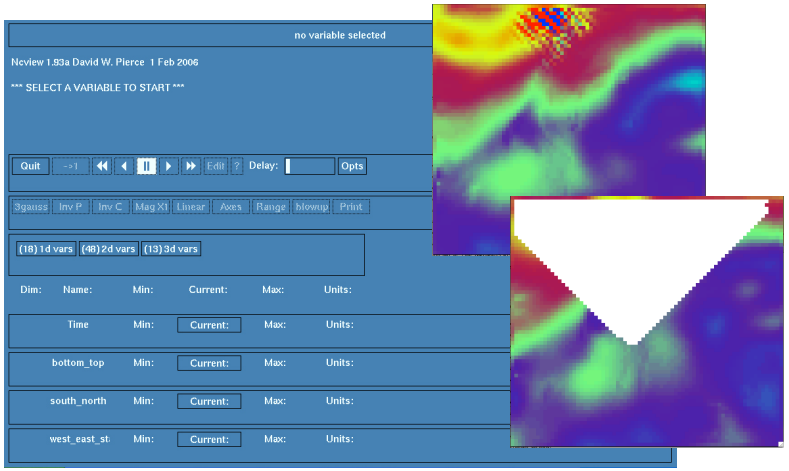
NCO Tools: Other Available Operators

- **ncap2:** arithmetic processor
- **ncatted:** ATTribute editor
- **ncbo:** binary operator (includes ncadd, ncsubtract, ncmultiply, ncdivide)
- **ncea:** ensemble averager
- **ncecat:** ensemble conCATenator
- **ncflint:** FiLe INTerpolator
- **ncpdq:** permute dimensions quickly, pack data quietly
- **ncrename:** RENAME-er
- **ncwa:** weighter averager

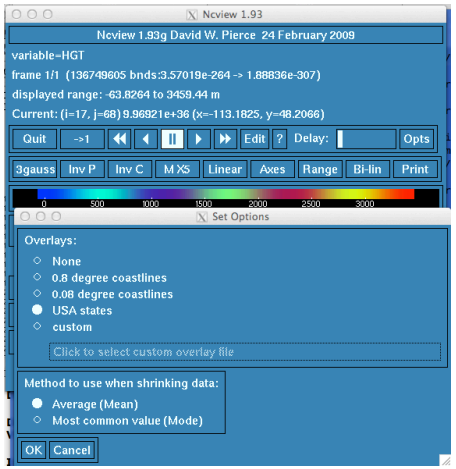
ncview

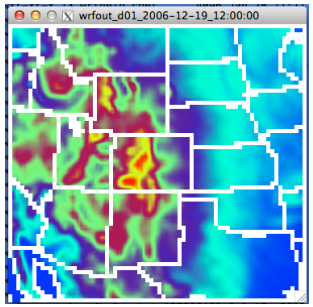
ncview

http://meteora.ucsd.edu/~pierce/ncview_home_page.html



ncview





Questions?

WRF Registry

Dave Gill

How to Use the WRF Registry

John Michalakes, NCEP

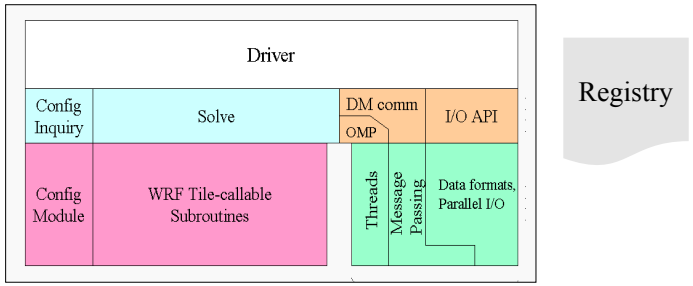
Dave Gill, NCAR

WRF Software Architecture Working Group

Outline

- What is the WRF Registry
- Keyword syntax
- The BIG Three
- Examples
  - Runtime I/O mods
  - Adding a variable to the namelist
  - Adding an array to WRF
  - Compute a diagnostic
  - New physics scheme
  - Passive tracer

WRF Software Architecture



Registry

Text based file for real and WRF
Active data dictionary
Used with cpp to auto generate source
Controls/defines
- Variables (I/O, comms, nesting)
- Communications
- namelist options

About 300k lines added to source
Easy – 3x the size since initial release
Compile-time option
- ./clean
- ./configure
- ./compile

Registry.EM_COMMON (else lost changes)

Registry Keywords

- Currently implemented as a text file: Registry/Registry.EM_COMMON
- Types of entry:
  - *Dimspec* – Describes dimensions that are used to define arrays in the model
  - *State* – Describes state variables and arrays in the domain structure
  - *I1* – Describes local variables and arrays in solve
  - *Typedef* – Describes derived types that are subtypes of the domain structure

Registry Keywords

- Types of entry:
  - *Rconfig* – Describes a configuration (e.g. namelist) variable or array
  - *Package* – Describes attributes of a package (e.g. physics)
  - *Halo* – Describes halo update interprocessor communications
  - *Period* – Describes communications for periodic boundary updates
  - *Xpose* – Describes communications for parallel matrix transposes
  - *include* – Similar to a CPP #include file

Registry State Entry

```
#      Type Sym  Dims   Use    Tlev Stag      IO      Dname       Descrip
state  real  tsk   ij    misc    1    -     i01rhusdf   "TSK"    "SKIN TEMP"
```

- Elements
  - *Entry*: The keyword "state"
  - *Type*: The type of the state variable or array (real, double, integer, logical, character, or derived)
  - *Sym*: The symbolic name of the variable or array
  - *Dims*: A string denoting the dimensionality of the array or a hyphen (-)
  - *Use*: A string denoting association with a solver or 4D scalar array, or a hyphen
  - *NumTLev*: An integer indicating the number of time levels (for arrays) or hypen (for variables)

Registry State Entry

```
#      Type Sym  Dims   Use    Tlev Stag      IO      Dname       Descrip
state  real  tsk   ij    misc    1    -     i01rhusdf   "TSK"    "SKIN TEMP"
```

- Elements
  - *Stagger*: String indicating staggered dimensions of variable (X, Y, Z, or hyphen)
  - *IO*: String indicating whether and how the variable is subject to various I/O and Nesting
  - *DName*: Metadata name for the variable
  - *Units*: Metadata units of the variable
  - *Descrip*: Metadata description of the variable

State Entry: Defining a variable-set for an I/O stream

- Fields are added to a variable-set on an I/O stream in the Registry

```
#      Type Sym  Dims   Use    Tlev  Stag    IO      Dname       Descrip
state  real  tsk   ij    misc    1     -    i01rhusdf   "TSK"    "SKIN TEMP"
```

- ***IO*** is a string that specifies if the variable is to be available to initial, restart, or history I/O. The string may consist of '**h**' (subject to history I/O), '**i**' (initial dataset), '**r**' (restart dataset).
- The 'h', 'r', and 'i' specifiers may appear in any order or combination.

State Entry: Defining a variable-set for an I/O stream

- Fields are added to a variable-set on an I/O stream in the Registry

```
#       Type Sym  Dims   Use    Tlev  Stag    IO      Dname       Descrip
state   real  tsk   ij    misc   1     -     i01rhusdf  "TSK"   "SKIN TEMP"
```

- The ‘h’ and ‘i’ specifiers may be followed by an optional integer string consisting of ‘0’, ‘1’, … , ‘9’
- Zero denotes that the variable is part of the principal input or history I/O stream.
- The characters ‘1’ through ‘9’ denote one of the auxiliary input or history I/O streams.

State Entry: Defining a variable-set for an I/O stream

- Fields are added to a variable-set on an I/O stream in the Registry

```
#       Type Sym  Dims   Use    Tlev  Stag    IO      Dname       Descrip
state   real  tsk   ij    misc   1     -     i01rhusdf  "TSK"   "SKIN TEMP"
```

usdf refers to nesting options: **u = UP, d = DOWN, s = SMOOTH, f = FORCE**

u – at end of each set of child time steps
d – at instantiation of child domain
f – at beginning of each set of child time steps
s – after each feedback

State Entry: Defining a variable-set for an I/O stream

- Fields are added to a variable-set on an I/O stream in the Registry

```
#       Type Sym  Dims   Use    Tlev  Stag    IO      Dname       Descrip
state   real  tsk   ij    misc   1     -     i01rhusdf  "TSK"   "SKIN TEMP"
```

Only variables involved with I/O, communications, packages are required to be state

Local variables inside of physics packages are not controlled by the Registry

Rconfig Entry

```
#          Type       Sym              How set           Nentries  Default
rconfig    integer spec_bdy_width  namelist,bdy_control    1          1
```

- This defines namelist entries
- Elements
  - *Entry*: the keyword “rconfig”
  - *Type*: the type of the namelist variable (integer, real, logical, string)
  - *Sym*: the name of the namelist variable or array
  - *How set*: indicates how the variable is set: e.g. namelist or derived, and if namelist, which block of the namelist it is set in

Rconfig Entry

```
#         Type       Sym              How set          Nentries  Default
rconfig   integer spec_bdy_width  namelist,bdy_control    1         1
```

- This defines namelist entries
- Elements
  - *Nentries*: specifies the dimensionality of the namelist variable or array. If 1 (one) it is a variable and applies to all domains; otherwise specify max_domains (which is an integer parameter defined in module_driver_constants.F).
  - *Default*: the default value of the variable to be used if none is specified in the namelist; hyphen (-) for no default

Package Entry

- Elements
  - *Entry*: the keyword "package",
  - *Package name*: the name of the package: e.g. "kesslerscheme"
  - *Associated rconfig choice*: the name of a rconfig variable and the value of that variable that choses this package

```
# specification of microphysics options
package   passiveqv      mp_physics==0     -   moist:qv
package   kesslerscheme  mp_physics==1     -   moist:qv,qc,qr
package   linscheme      mp_physics==2     -   moist:qv,qc,qr,qi,qs,qg
package   ncepcloud3     mp_physics==3     -   moist:qv,qc,qr
package   ncepcloud5     mp_physics==4     -   moist:qv,qc,qr,qi,qs


# namelist entry that controls microphysics option
rconfig  integer   mp_physics   namelist,physics   max_domains    0
```

Package Entry

- Elements
  - *Package state vars*: unused at present; specify hyphen (-)
  - *Associated* variables: the names of 4D scalar arrays (moist, chem, scalar) and the fields within those arrays this package uses, and the state variables (state:u_gc, ...)

```
# specification of microphysics options
package   passiveqv      mp_physics==0     -   moist:qv
package   kesslerscheme  mp_physics==1     -   moist:qv,qc,qr
package   linscheme      mp_physics==2     -   moist:qv,qc,qr,qi,qs,qg
package   ncepcloud3     mp_physics==3     -   moist:qv,qc,qr
package   ncepcloud5     mp_physics==4     -   moist:qv,qc,qr,qi,qs


# namelist entry that controls microphysics option
rconfig  integer   mp_physics   namelist,physics   max_domains    0
```

Outline

- Examples
  - 1) Add output without recompiling
  - 2) Add a variable to the namelist
  - 3) Add an array
  - 4) Compute a diagnostic
  - 5) Add a physics package
  - 6) Tracer

Example 1: Add output without recompiling

- Edit the namelist.input file, the time_control namelist record

```
iofields_filename = "myoutfields.txt" (MAXDOM)
io_form_auxhist24 = 2 (choose an available stream)
auxhist24_interval = 10 (MAXDOM, every 10 minutes)
```

- Place the fields that you want in the named text file `myoutfields.txt`

```
+:h:24:RAINC,RAINNC
```

- Where "+" means ADD this variable to the output stream, "h" is the history stream, and "24" is the stream number

Example 1: Zap output without recompiling

- Edit the namelist.input file, the time_control namelist record

```
iofields_filename = "myoutfields.txt"
```

- Place the fields that you want in the named text file `myoutfields.txt`

```
-:h:0:W,PB,P
```

- Where "–" means REMOVE this variable from the output stream, "h" is the history stream, and "0" is the stream number (standard WRF history file)

Outline

- Examples
  - 1) Add output without recompiling
  - 2) Add a variable to the namelist
  - 3) Add an array
  - 4) Compute a diagnostic
  - 5) Add a physics package
  - 6) Tracer

Example 2: Add a variable to the namelist

- Use the examples for the rconfig section of the Registry
- Find a namelist variable similar to what you want
  - Integer *vs* real *vs* logical *vs* character
  - Single value *vs* value per domain
  - Select appropriate namelist record
- Insert your mods in all appropriate Registry files

Example 2: Add a variable to the namelist

- Remember that ALL Registry changes require that the WRF code be cleaned and rebuilt

```
./clean -a
./configure
./compile em_real
```

Example 2: Add a variable to the namelist

- Adding a variable to the namelist requires the inclusion of a new line in the Registry file:

```
rconfig integer my_option_1  namelist,time_control  1 0 - "my_option_1" "test namelist option"
rconfig integer my_option_2  namelist,time_control  max_domains 0
```

- Accessing the variable is through an automatically generated function:

```
USE module_configure
INTEGER :: my_option_1 , my_option_2

CALL nl_get_my_option_1( 1, my_option_1 )
CALL nl_set_my_option_2( grid%id, my_option_2 )
```

Example 2: Add a variable to the namelist

- You also have access to the namelist variables from the grid structure …

```
SUBROUTINE foo ( grid , ... )

  USE module_domain
  TYPE(domain) :: grid

  print *,grid%my_option_1
```

Example 2: Add a variable to the namelist

- … and you also have access to the namelist variables from config_flags

```
SUBROUTINE foo2 ( config_flags , ... )

  USE module_configure
  TYPE(grid_config_rec_type) :: config_flags

  print *,config_flags%my_option_2
```

Example 2: Add a variable to the namelist

- What your variable looks like in the namelist.input file

```
&time_control
run_days                            = 0,
run_hours                           = 0,
run_minutes                         = 40,
run_seconds                         = 0,
start_year                          = 2006, 2006, 2006,
my_option_1                         = 17
my_option_2                         = 1, 2, 3
```

Outline

- Examples
  - 1) Add output without recompiling
  - 2) Add a variable to the namelist
  - 3) Add an array
  - 4) Compute a diagnostic
  - 5) Add a physics package
  - 6) Tracer

Example 3: Add an Array

- Adding a state array to the solver, requires adding a single line in the Registry
- Use the previous Registry instructions for a state or I1 variable

Example 3: Add an Array

- Select a variable similar to one that you would like to add
  - 1d, 2d, or 3d
  - Staggered (X, Y, Z, or not "-", *do not leave blank*)
  - Associated with a package
  - Part of a 4d array
  - Input (012), output, restart
  - Nesting, lateral forcing, feedback

Example 3: Add an Array

- Copy the "similar" field's line and make a few edits
- Remember, no Registry change takes effect until a "clean -a" and rebuild

```
state  real  h_diabatic  ikj  misc  1  -      r                    \
       "h_diabatic"  "PREVIOUS TIMESTEP CONDENSATIONAL HEATING"

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

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

state  real  ht_input     ij   misc  1  -     -                     \
       "HGT_INPUT"   "Terrain Height from FG Input File"

state  real  TSK_SAVE     ij   misc  1  -     -                     \
       "TSK_SAVE"    "SURFACE SKIN TEMPERATURE"  "K"
```

Example 3: Add an Array

- Always modify Registry.*core_name*_COMMON or Registry.*core_name*, where *core_name* might be EM

```
state  real  h_diabatic  ikj  misc  1  -      r                    \
       "h_diabatic"  "PREVIOUS TIMESTEP CONDENSATIONAL HEATING"

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

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

state  real  ht_input     ij   misc  1  -     -                     \
       "HGT_INPUT"   "Terrain Height from FG Input File"

state  real  TSK_SAVE     ij   misc  1  -     -                     \
       "TSK_SAVE"    "SURFACE SKIN TEMPERATURE"  "K"
```

Example 3: Add an Array

- Add a new 3D array that is sum of all moisture species, called all_moist, in the Registry.EM_COMMON
  - Type: real
  - Dimensions: 3D and ikj ordering, not staggered
  - Supposed to be output only: h
  - Name in netCDF file: ALL_MOIST

```
state    real   all_moist       ikj       \
dyn_em       1       -      h             \
"ALL_MOIST"                               \
"sum of all of moisture species"          \
"kg kg-1"
```

Example 3: Add an Array

- Registry state variables become part of the derived data structure usually called grid inside of the WRF model.
- WRF model top → integrate → solve_interface → solve
- Each step, the grid construct is carried along for the ride
- No source changes for new output variables required until below the solver routine

Example 3: Add an Array

- Top of solve_em.F
- grid is passed in
- No need to declare any new variables, such as all_moist

```
!WRF:MEDIATION_LAYER:SOLVER

  SUBROUTINE solve_em ( grid , &

  config_flags , &
```

Example 3: Add an Array

- The solve routine calls first_rk_step_part1
- grid is passed in
- No need to pass any variables, such as all_moist

```
!WRF:MEDIATION_LAYER:SOLVER

  CALL first_rk_step_part1( grid , &

  config_flags , &
```

Example 3: Add an Array

- Top of first_rk_step_part1.F
- grid is passed in
- No need to declare any new variables, such as all_moist

```
!WRF:MEDIATION_LAYER:SOLVER

MODULE module_first_rk_step_part1

CONTAINS

  SUBROUTINE first_rk_step_part1 ( grid , &

  config_flags , &
```

Example 3: Add an Array

- In first_rk_step_part1, add the new array to the call for the microphysics driver
- Syntax for variable=local_variable is an association convenience
- All state arrays are contained within grid, and must be de-referenced

```
CALL microphysics_driver(           &
     QV_CURR=moist(ims,kms,jms,P_QV), &
     QC_CURR=moist(ims,kms,jms,P_QC), &
     QR_CURR=moist(ims,kms,jms,P_QR), &
     QI_CURR=moist(ims,kms,jms,P_QI), &
     QS_CURR=moist(ims,kms,jms,P_QS), &
     QG_CURR=moist(ims,kms,jms,P_QG), &
     QH_CURR=moist(ims,kms,jms,P_QH), &
     all_moist=grid%all_moist       , &
```

Example 3: Add an Array

- After the array is re-referenced from grid and we are inside the microphysics_driver routine, we need to
  - Pass the variable through the argument list
  - Declare our passed in 3D array

```
  ,all_moist &




REAL, DIMENSION(ims:ime ,kms:kme ,jms:jme ), &
      INTENT(OUT) ::  all_moist
```

Example 3: Add an Array

- After the array is re-referenced from grid and we are inside the microphysics_driver routine, we need to
  - Zero out the array at each time step

```
!  Zero out moisture sum.

 DO j = jts,MIN(jde-1,jte)
 DO k = kts,kte
 DO i = its,MIN(ide-1,ite)
    all_moist(i,k,j) = 0.0
 END DO
 END DO
 END DO
```

Example 3: Add an Array

- After the array is re-referenced from grid and we are inside the microphysics_driver routine, we need to
  - At the end of the routine, for each of the moist species that exists, add that component to all_moist

```
DO j = jts,MIN(jde-1,jte)
  DO k = kts,kte
  IF ( f_QV ) THEN
     DO i = its,MIN(ide-1,ite)
        all_moist(i,k,j) = all_moist(i,k,j) + &
                           qv_curr(i,k,j)
      END DO
  END IF
```

Outline

- Examples
  - 1) Add output without recompiling
  - 2) Add a variable to the namelist
  - 3) Add an array
  - 4) Compute a diagnostic
  - 5) Add a physics package
  - 6) Tracer

Example 4: Compute a Diagnostic

- Problem: Output global average and global maximum and lat/lon location of maximum for 10 meter wind speed in WRF
- Steps:
  - Modify solve to compute wind-speed and then compute the local sum and maxima at the end of each time step
  - Use reduction operations built-in to WRF software to compute the global qualities
  - Output these on one process (process zero, the "monitor" process)

Example 4: Compute a Diagnostic

- Compute local sum and local max and the local indices of the local maximum

```
   --- File: dyn_em/solve_em.F  (near the end) ---

! Compute local maximum and sum of 10m wind-speed
   sum_ws = 0.
   max_ws = 0.
   DO j = jps, jpe
     DO i = ips, ipe
       wind_vel = sqrt( grid%u10(i,j)**2+ grid%v10(i,j)**2 )
       IF ( wind_vel .GT. max_ws ) THEN
          max_ws = wind_vel
          idex = i
          jdex = j
       ENDIF
       sum_ws = sum_ws + wind_vel
     ENDDO
   ENDDO
```

Example 4: Compute a Diagnostic

- Compute global sum, global max, and indices of the global max (WRF intrinsics)

```
! Compute global sum
   sum_ws = wrf_dm_sum_real ( sum_ws )

! Compute global maximum and associated i,j point
   CALL wrf_dm_maxval_real ( max_ws, idex, jdex )
```

Example 4: Compute a Diagnostic

- On the process that contains the maximum value, obtain the latitude and longitude of that point; on other processes set to an artificially low value.
- The use parallel reduction to store that result on every process

```
   IF ( ips .LE. idex .AND. idex .LE. ipe .AND.  &
        jps .LE. jdex .AND. jdex .LE. jpe ) THEN
      glat = grid%xlat(idex,jdex)
      glon = grid%xlong(idex,jdex)
   ELSE
      glat = -99999.
      glon = -99999.
   ENDIF

! Compute global maximum to find glat and glon
   glat = wrf_dm_max_real ( glat )
   glon = wrf_dm_max_real ( glon )
```

Example 4: Compute a Diagnostic

- Output the value on process zero, the "monitor"

```
! Print out the result on the monitor process
   IF ( wrf_dm_on_monitor() ) THEN
      WRITE(outstring,*)'Avg. ',sum_ws/((ide-ids+1)*(jde-jds+1))
      CALL wrf_message ( TRIM(outstring) )
      WRITE(outstring,*)'Max. ',max_ws,' Lat. ',glat,&
                                       ' Lon. ',glon
      CALL wrf_message ( TRIM(outstring) )
   ENDIF
```

Example 4: Compute a Diagnostic

- Output from process zero of a multi-process run

```
    --- Output file: rsl.out.0000 ---
    . . .
  Avg.    5.159380
  Max.    15.09370     Lat.    37.25022     Lon.   -67.44571
Timing for main: time 2000-01-24_12:03:00 on domain   1:    8.96500 elapsed secs.
  Avg.    5.166167
  Max.    14.97418     Lat.    37.25022     Lon.   -67.44571
Timing for main: time 2000-01-24_12:06:00 on domain   1:    4.89460 elapsed secs.
  Avg.    5.205693
  Max.    14.92687     Lat.    37.25022     Lon.   -67.44571
Timing for main: time 2000-01-24_12:09:00 on domain   1:    4.83500 elapsed secs.
    . . .
```

Outline

- Examples
  - 1) Add output without recompiling
  - 2) Add a variable to the namelist
  - 3) Add an array
  - 4) Compute a diagnostic
  - 5) Add a physics package
  - 6) Tracer

Example 5: Input periodic SSTs

- Add a new physics package with time varying input source to the model
- This is how we could supply a time varying value to the model for a field that is traditionally fixed
- Example is sea surface temperature

Example 5: Input periodic SSTs

- Problem: adapt WRF to input a time-varying lower boundary condition, e.g. SSTs, from an input file for a new surface scheme
- Given: Input file in WRF I/O format containing 12-hourly SST's
- Modify WRF model to read these into a new state array and make available to WRF surface physics

Example 5: Input periodic SSTs

- Steps
  - Add a new state variable and definition of a new surface layer package (that will use the variable) to the Registry
  - Add to variable stream for an unused Auxiliary Input stream
  - Adapt physics interface to pass new state variable to physics
  - Setup namelist to input the file at desired interval

Example 5: Input periodic SSTs

- Add a new state variable to Registry/Registry.EM_COMMON and put it in the variable set for input on Auxiliary Input Stream #4

```
#     type  symbol dims use  tl stag  io     dname      description        units
state real  nsst   ij   misc  1  -    i4h   "NEW_SST" "Time Varying SST" "K"
```

- Also added to History and Restart
- Result:
  - 2-D variable named grid%**nsst** defined and available in solve_em
  - Dimensions: ims:ime, jms:jme
  - Input and output on the AuxInput #4 stream will include the variable under the name NEW_SST

Example 5: Input periodic SSTs

- Pass new state variable to surface physics

```
       --- File: dyn_em/module_first_rk_step_part1.F ---

  CALL surface_driver(                                                  &
             . . .
! Optional
 &        ,QV_CURR=moist(ims,kms,jms,P_QV), F_QV=F_QV                   &
 &        ,QC_CURR=moist(ims,kms,jms,P_QC), F_QC=F_QC                   &
 &        ,QR_CURR=moist(ims,kms,jms,P_QR), F_QR=F_QR                   &
 &        ,QI_CURR=moist(ims,kms,jms,P_QI), F_QI=F_QI                   &
 &        ,QS_CURR=moist(ims,kms,jms,P_QS), F_QS=F_QS                   &
 &        ,QG_CURR=moist(ims,kms,jms,P_QG), F_QG=F_QG                   &
 &        ,NSST=grid%nsst                                               & ! new
 &        ,CAPG=grid%capg, EMISS=grid%emiss, HOL=hol,MOL=grid%mol       &
 &        ,RAINBL=grid%rainbl,SR=grid%em_sr                             &
 &        ,RAINNCV=grid%rainncv,REGIME=regime,T2=grid%t2,THC=grid%thc   &
             . . .
```

Example 5: Input periodic SSTs

- Add new variable nsst to Physics Driver in Mediation Layer

```
      --- File: phys/module_surface_driver.F ---

SUBROUTINE surface_driver(                                          &
       . . .
          !  Other optionals (more or less em specific)
  &          ,nsst                                                  &
  &          ,capg,emiss,hol,mol                                    &
  &          ,rainncv,rainbl,regime,t2,thc                          &
  &          ,qsg,qvg,qcg,soilt1,tsnav                              &
  &          ,smfr3d,keepfr3dflag                                   &
      . . .
                                                                    ))
      . . .
REAL, DIMENSION( ims:ime, jms:jme ), OPTIONAL, INTENT(INOUT)::   nsst
```

- By making this an "Optional" argument, we preserve the driver's compatibility with other cores and with versions of WRF where this variable hasn't been added.

Example 5: Input periodic SSTs

- Add call to Model-Layer subroutine for new physics package to Surface Driver

```
      --- File: phys/module_surface_driver ---

!$OMP PARALLEL DO   &
!$OMP PRIVATE ( ij, i, j, k )
   DO ij = 1 , num_tiles
     sfclay_select: SELECT CASE(sf_sfclay_physics)

       CASE (SFCLAYSCHEME)
         . . .
       CASE (NEWSFCSCHEME)  ! <- This is defined by the Registry "package" entry

         IF (PRESENT(nsst))  THEN
            CALL NEWSFCCHEME(                                       &
                nsst,                                               &
                ids,ide, jds,jde, kds,kde,                          &
                ims,ime, jms,jme, kms,kme,                          &
                i_start(ij),i_end(ij), j_start(ij),j_end(ij), kts,kte    )
         ELSE
           CALL wrf_error_fatal('Missing argument for NEWSCHEME in surface driver')
         ENDIF
         . . .
     END SELECT sfclay_select
   ENDDO
!$OMP END PARALLEL DO
```

- Note the PRESENT test to make sure new optional variable nsst is available

Example 5: Input periodic SSTs

- Add definition for new physics package NEWSCHEME as setting 4 for namelist variable sf_sfclay_physics

```
rconfig   integer  sf_sfclay_physics   namelist,physics   max_domains    0

package   sfclayscheme   sf_sfclay_physics==1        -            -
package   myjsfcscheme   sf_sfclay_physics==2        -            -
package   gfssfcscheme   sf_sfclay_physics==3        -            -
package   newsfcscheme   sf_sfclay_physics==4        -            -
```

- This creates a defined constant NEWSFCSCHEME and represents selection of the new scheme when the namelist variable sf_sfclay_physics is set to '4' in the namelist.input file
- `clean -a` and recompile so code and Registry changes take effect

Example 5: Input periodic SSTs

- Setup namelist to input SSTs from the file at desired interval

```
      --- File: namelist.input ---

&time_control
   . . .
 auxinput4_inname       = "sst_input"
 auxinput4_interval_h   = 12
 . . .
/

   . . .
&physics
 sf_sfclay_physics   = 4, 4, 4
   . . .
/
```

- Run code with sst_input file in run-directory

Outline

- Examples
  - 1) Add output without recompiling
  - 2) Add a variable to the namelist
  - 3) Add an array
  - 4) Compute a diagnostic
  - 5) Add a physics package
  - 6) Tracer

Tracer Example

Modify Registry for new fields.

Use the "tracer" array with a new 3D component

Use existing NML option

Initialize data in real.

Identify (i,j) location

Spread in "PBL"

Set values in solver.

"Release" per time step

Tracer Example

Registry/Registry.EM add our new field "PLUME" as part of "TRACER" array.

```
#     New tracer for example
state  real  plume  ikjftb  tracer \
     1  -  irhusdf=(bdy_interp:dt) \
     "PLUME"  "Fukushima Tracer"  " "

#     4D arrays need an associated package
package  tracer_test3  tracer_opt==3  - \
     tracer:plume
```

Tracer Example

Modify the real and WRF programs to initialize and continuously re-supply the "PLUME" array

dyn_em/module_initialize_real.F (initial value from real.exe)

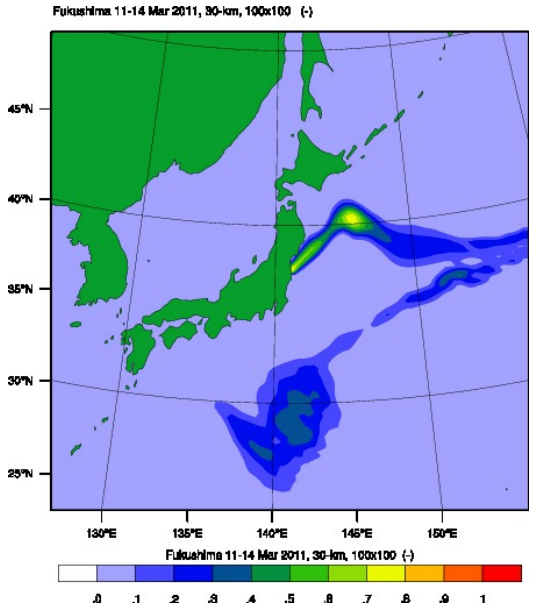
dyn_em/solve_em.F (continuous plume in wrf.exe)

```
!  Add in the Fukushima initial venting.

   IF ( ( its .LE. 50 ) .AND. ( ite .GE. 50 ) .AND. &
        ( jts .LE. 50 ) .AND. ( jte .GE. 50 ) ) THEN
      tracer(50,1:5,50,P_plume) = 1.
   END IF
```

Tracer Example

- Modify the test/em_real/namelist.input file
- Include the new settings for the tracer option required from the Registry file

```
&dynamics
 tracer_opt  = 3, 3, 3,
```



Outline

- What is the WRF Registry
- Keyword syntax
- The BIG Three
- Examples
  - Runtime I/O mods
  - Adding a variable to the namelist
  - Adding an array to WRF

WPS:
Advanced Usage

Michael Duda

Advanced Features of the WRF Preprocessing System

Michael Duda

*NCAR is sponsored by the National Science Foundation

Outline

- The GEOGRID.TBL file
  - What is the GEOGRID.TBL file?
  - Ingesting new static fields
  - Examples: Using high-resolution land use and topography data
- The METGRID.TBL file
  - What is the METGRID.TBL file?
  - Example: Defining interpolation options for a new field
  - Example: Using the METGRID.TBL file for a real-time system
- Utility programs example: fixing "hot lakes"

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

The GEOGRID.TBL File

- 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   # Houston, TX urban data
       priority    = 1
       dest_type   = categorical
       z_dim_name  = land_cat
       interp_option = 30s:nearest_neighbor
       abs_path = 30s:/users/duda/Houston/
==============================
```

For a complete list of possible keywords *See p. 3-46*

The GEOGRID.TBL File

- 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*

New Fields in GEOGRID.TBL

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 South Korea

1) Completely new fields

Completely new fields:

For a new field, simply add an entry in GEOGRID.TBL for that field.

```
==============================
name = MY_NEW_FIELD_NAME
   priority = 1
   dest_type  = continuous
   interp_option = four_pt
   abs_path      = /data/duda/mydata/
==============================
```

Name of field that this entry is for

Priority of this data source compared with other sources for same field

How to interpolate this field

Where on disk to find the data for this field

See p. 3-46

2) Different resolution data set

Different resolution data sets for an existing field:

Specify the path to the new data set and which interpolation methods 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

Alternative sources for a field that must be used in addition to an existing source :

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/
==============================
```


Preparing new geogrid data sets

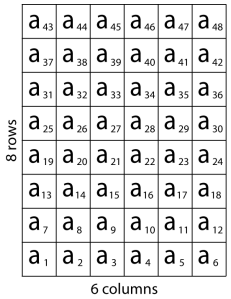
To add a new data source, we need to

1) Write the data in the proper binary format
   - See Chapter 3: "Writing Static Data to the Geogrid Binary Format"
   - Can make use of read_geogrid.c and write_geogrid.c
2) Create an "index" metadata file for the data set
   - This tells geogrid about the projection, coverage, resolution, type, and storage representation of the data set
3) Add/edit entry for the data in the GEOGRID.TBL file
   - The change to GEOGRID.TBL will follow one of the three cases mentioned before

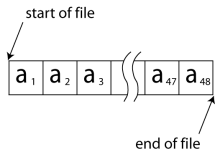
The Geogrid Data Format

The geogrid format is a simple binary raster

- Elements of a rectangular array of data are written, row by row, to a file
- No record markers or any type of metadata are written to this file



See p. 3-37



*A file containing a N×M array, with each element represented using K bytes, should have size exactly N*M*K bytes!*

The Geogrid Data Format

Since the contents of the file contain only the values from the array, *care must be taken if using Fortran to write the array*

- Fortran unformatted writes add *record markers* to the beginning and end of each record
- So, rather than $X_1X_2X_3\ldots X_{n-1}X_n$ we get $RX_1X_2X_3\ldots X_{n-1}X_nR$, where R is a record marker

Instead of Fortran, the C routines `read_geogrid.c` and `write_geogrid.c` may be used to read and write binary files

- these may be called from either Fortran or C

The Geogrid Data Format

The filenames of geogrid binary files should have the form:

xxxxx-XXXXX.yyyyy-YYYYY

where

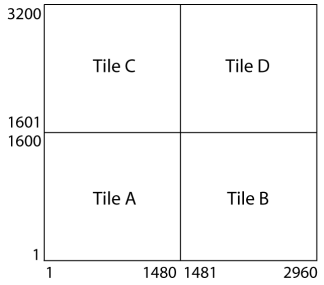
xxxxx	is the starting x-index
XXXXX	is the ending x-index
yyyyy	is the starting y-index
YYYYY	is the ending y-index

E.g., For a binary file containing an array with 500 columns and 750 rows, the file name would be 00001-00500.00001-00750

The Geogrid Data Format

If the data are not available in a single tile (array), multiple files may be used to store the data

- All tiles must have the same x-dimension
- All tiles must have the same y-dimension
- If necessary, a tile can be "padded" with missing values to expand it to the same size as other tiles in the data set

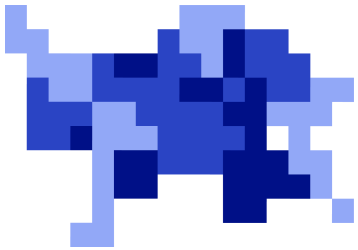


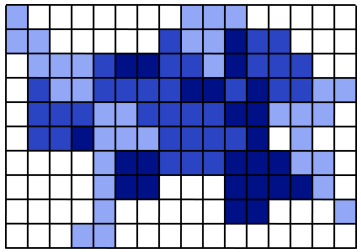
Tile A named 00001-01480.00001-01600
Tile B named 01481-02960.00001-01600
Tile C named 00001-01480.01601-03200
Tile D named 01481-02960.01601-03200

The Geogrid Data Format

If the data do not cover a rectangular region, areas with no data are simply filled with a missing value so that the overall data set is rectangular

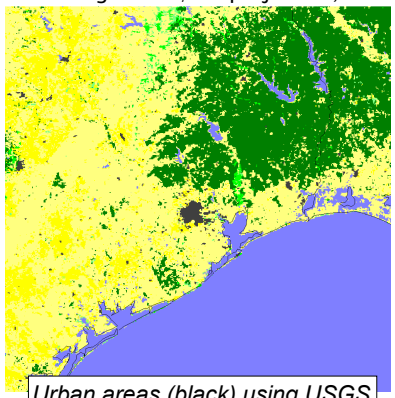
- The particular missing value used in the data set is specified in the `index` metadata file for the data set



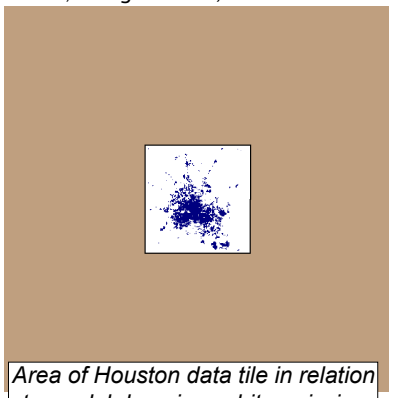


Example: Houston LU Data Set

- Given dataset for new Houston urban land use categories
  - Regular lat/lon projection, 30" resolution; categories 31, 32 & 33



Urban areas (black) using USGS 24-category data set



Area of Houston data tile in relation to model domain – white=missing data and blue=valid data

Example: Houston LU Data Set

To make use of the new data, we do the following:

1) Write the data to the binary format used by geogrid

2) Create an index file for the data

```
type=categorical
category_min=31; category_max=33
projection=regular_ll
dx=0.00833333; dy=0.00833333
known_x=1.0;   known_y=1.0
known_lat=29.3375
known_lon=-95.9958333
wordsize=1
tile_x=157; tile_y=143; tile_z=1
missing_value = 0.
units="category"
description="3-category urban LU"
```

Data set has categories 31 through 33

30 arc second resolution

Geographic location of data set

Treat 0 as "no data"

See p. 3-49

Example: Houston LU Data Set

3) Define an entry for the data in GEOGRID.TBL

```
==============================
name=LANDUSEF
        priority    = 2
        dest_type   = categorical
        z_dim_name  = land_cat
        interp_option = default:nearest_neighbor
        abs_path    = default:/users/duda/Houston/
==============================
```

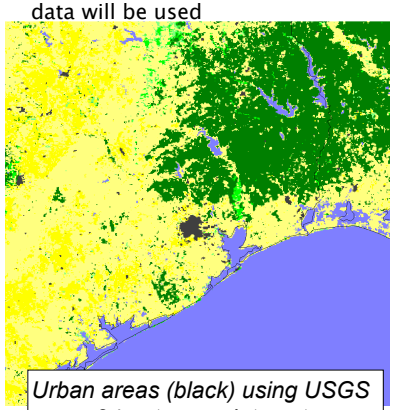
Give this data source priority over default data sources

How to interpolate this data source, and where to find it on disk

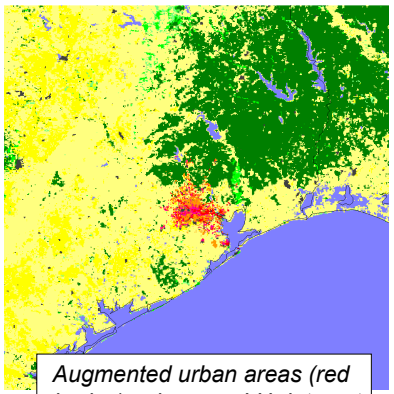
Example: Houston LU Data Set

4) Run geogrid.exe

Any gridpoints covered by Houston data will use it; otherwise default USGS data will be used



Urban areas (black) using USGS 24-category data set



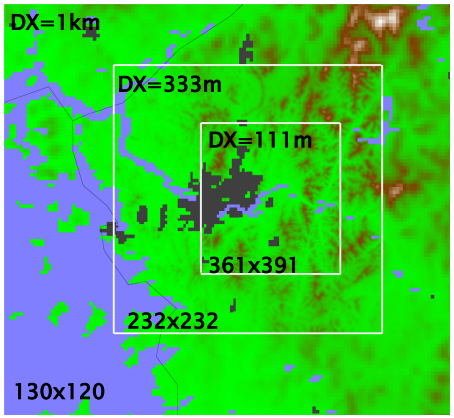
Augmented urban areas (red shades) using new LU data set

Example: South Korea

Shuttle Radar Topography Mission (SRTM) 3 arc second topography data

We would like to use the SRTM data, especially for domains 2 and 3.

Follow steps for adding a new resolution for an existing data set (case 2)



Example: Seoul

To use the SRTM topography data, we

1) Write data to geogrid binary format

2) Create an index file for the data set

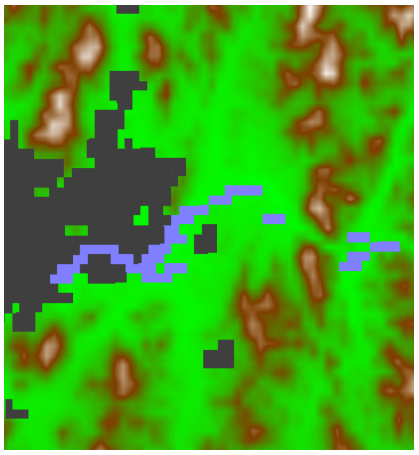
3) Modify the GEOGRID.TBL entries for HGT_M, HGT_U, and HGT_V

```
==============================
name = HGT_M
     priority = 1
     dest_type = continuous
     interp_option   =   30s:special(4.0)+four_pt
     interp_option   =   SRTM:four_pt
     rel_path    =   30s:topo_30s/
     rel_path    =   SRTM:SRTM/
==============================
```

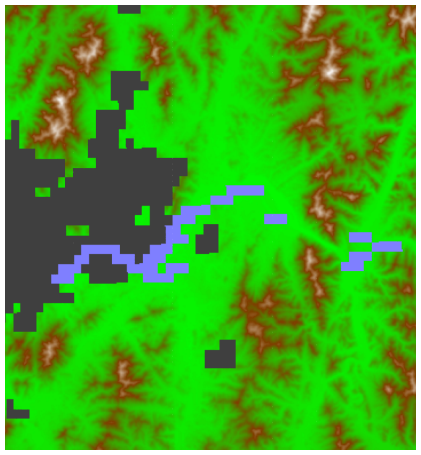
4) Specify that we should interpolate from SRTM in namelist by setting

```
geog_data_res = '30s','SRTM+30s','SRTM+30s'
```


Example: Seoul



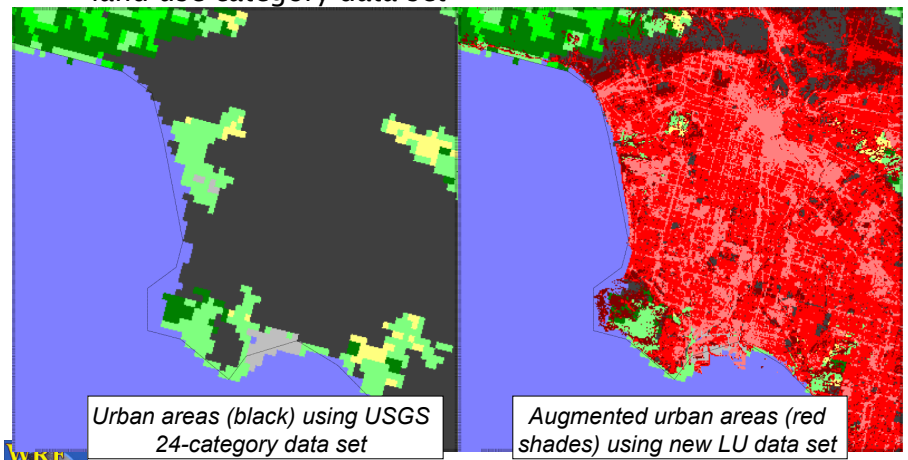
Domain 3 (DX=111m) using default 30" USGS topography



Domain 3 (DX=111m) using 3" SRTM topography

Another Example: Los Angeles

For Los Angeles, we have a 30-meter resolution, 3 urban land use category data set



Urban areas (black) using USGS 24-category data set

Augmented urban areas (red shades) using new LU data set

Outline

- The GEOGRID.TBL file
  - What is the GEOGRID.TBL file?
  - Ingesting new static fields
  - Examples: Using high-resolution land use and topography data
- The METGRID.TBL file
  - What is the METGRID.TBL file?
  - Example: Defining interpolation options for a new field
  - Example: Using the METGRID.TBL file for a real-time system
- Utility programs example: fixing "hot lakes"

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

The METGRID.TBL File

- 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

The METGRID.TBL File

- Example METGRID.TBL entry (for "soil moisture 0-10 cm")

```
========================================
name=SM000010
interp_option=sixteen_pt+four_pt+average_4pt
masked=water
interp_mask=LANDSEA(0)
fill_missing=1.
flag_in_output=FLAG_SM000010
========================================
```

Example: A new METGRID.TBL entry

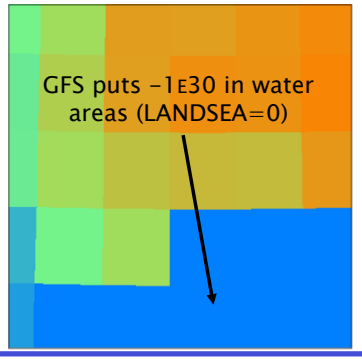
- Suppose we have a 1000x1000 domain over Houston (dx=500 m)
  - This is the same domain as in the urban land use example
- Meteorological data come from 1-degree GFS
  - *Note that we will be interpolating 1-degree data onto a 500-m grid!*
- We want to create an entry for a new soil moisture field, SM000010

Example: A new METGRID.TBL entry

- Initially, we run metgrid.exe and get the message:

```
INFORM: Entry in METGRID.TBL not found for field SM000010.
  Default options will be used for this field!
```

- The resulting SM000010 field looks very coarse
- We need to create a METGRID.TBL entry so metgrid will know how to interpolate this field!



Example: A new METGRID.TBL entry

- We add an initial entry in METGRID.TBL for SM000010:

```
==========================
name = SM000010
masked = water
interp_mask = LANDSEA(0)
interp_option = sixteen_pt + nearest_neighbor
fill_missing = 0.
==========================
```

Specify that the field should *not* be interpolated to model water points

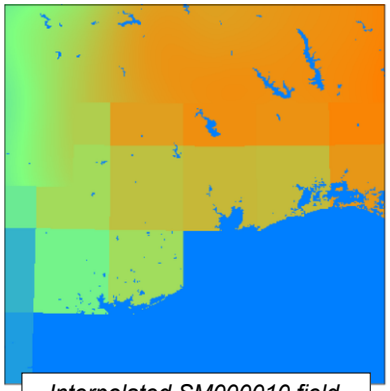
Specify that metgrid should not use points in source where LANDSEA field equals 0

Fill model points that don't receive an interpolated value (like water) to 0

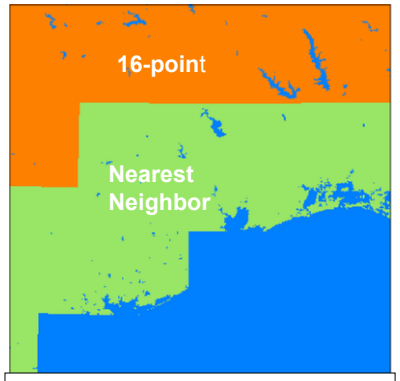
For a complete list of possible keywords *See p. 3-52*

Example: A new METGRID.TBL entry

- Now, after running metgrid.exe again, the SM000010 field looks like



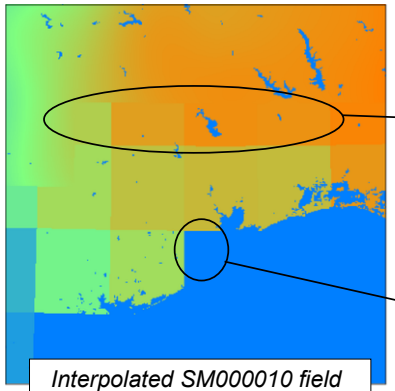
Interpolated SM000010 field (sixteen_pt + nearest_neighbor)



Which interpolator was used at each model grid point

Example: A new METGRID.TBL entry

- But, the interpolated field still looks bad near the coastline



Interpolated SM000010 field (sixteen_pt + nearest_neighbor)

Should be sufficient data to use 4-point interpolation in these areas

Model grid points here should be adjacent to at least one valid GFS point (though not nearest)

Example: A new METGRID.TBL entry

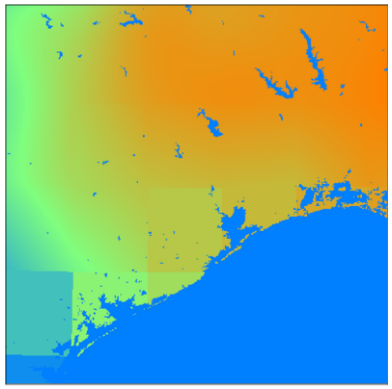
- Update the METGRID.TBL entry for SM000010

```
==========================
name = SM000010
masked = water
interp_mask = LANDSEA(0)
interp_option = sixteen_pt + four_pt + average_4pt
fill_missing = 0.
==========================
```

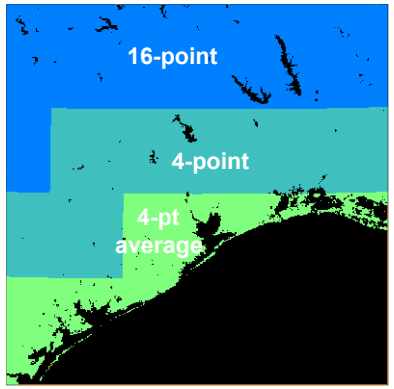
- If 16-pt doesn't work, then try 4-pt before reverting to a 4-point average
  - Note that 4-point average will work anywhere nearest_neighbor would (missing/masked values not counted in the average)

Example: A new METGRID.TBL entry

- The resulting field, below-left:



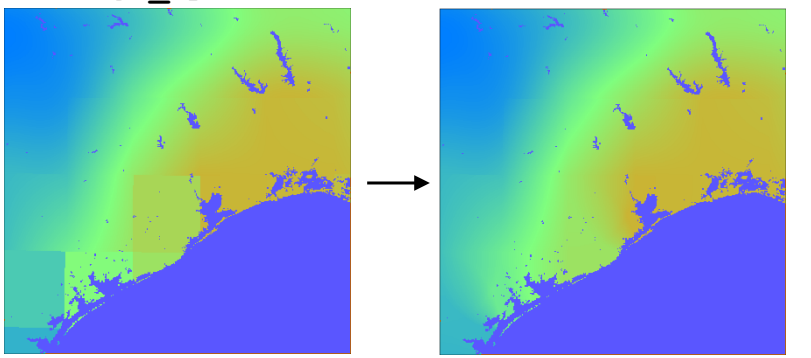
Interpolated SM000010 field (sixteen_pt + four_pt + average_4pt)



Which interpolator was used at each model grid point

Example: A new METGRID.TBL entry

- By using **`wt_average_4pt`** instead of **`average_4pt`**:



sixteen_pt + four_pt + average_4pt

sixteen_pt + four_pt + wt_average_4pt

METGRID.TBL: Real-time System Example

- Suppose we have a real-time system that:
  - Uses GFS for initial and boundary conditions
  - When possible (i.e., if the files are available soon enough) uses *soil moisture* and *soil temperature* fields from AGRMET
- In our system, it may occasionally happen that the AGRMET files are not ready when we want to start our WRF run
  - Because system is real-time, we want to proceed using just the GFS land surface fields!

METGRID.TBL: Real-time System Example

- We already know how to run ungrib on multiple sources of data to get

  GFS:YYYY-MM-DD_HH

  and

  AGRMET:YYYY-MM-DD_HH

  intermediate files, and specify

  fg_name = 'GFS', 'AGRMET',

  in the `&metgrid` namelist record to use both sources

See p. 3-24

METGRID.TBL: Real-time System Example

Without further changes, what happens if:

Only GFS data are available when we run metgrid

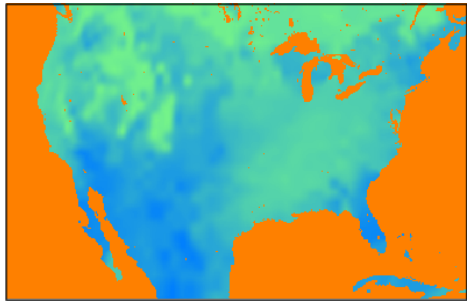
Metgrid runs and warns that no AGRMET data files were found:

```
Processing 2012-04-01_00
      GFS
      AGRMET
 WARNING: Couldn't open file AGRMET:2012-04-01_00 for
 input.
```

Metgrid will finish, but will only use GFS data!

METGRID.TBL: Real-time System Example

And the 0-10 cm soil moisture field (SM000010) looks like:

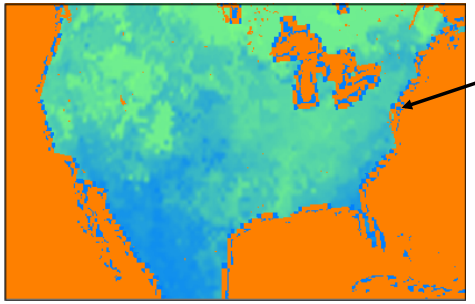


METGRID.TBL: Real-time System Example

However, what happens if:

Both GFS and AGRMET files are available when we run metgrid?

Our SM000010 field looks like:



We get unreasonable values with magnitude ~1E30 near land-water boundaries!

METGRID.TBL: Real-time System Example

Why are there bad values near coastlines? What went wrong?

In both Vtable.GFS and Vtable.AGRMET, the land-sea mask field is named LANDSEA

- In METGRID.TBL, our entry for SM000010 says:

```
========================================
name=SM000010
interp_option=sixteen_pt+four_pt+wt_average_4pt+search
masked=water
interp_mask=LANDSEA(0)
fill_missing=1.
flag_in_output=FLAG_SM000010
========================================
```


METGRID.TBL: Real-time System Example

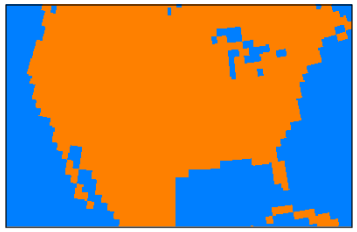
```
========================================
name=SM000010
interp_option=sixteen_pt+four_pt+wt_average_4pt+search
masked=water
interp_mask=LANDSEA(0)
fill_missing=1.
flag_in_output=FLAG_SM000010
========================================
```

After metgrid reads in LANDSEA from GFS file *to use as an interpolation mask*, it ignored the LANDSEA field from AGRMET *for use as a mask*.

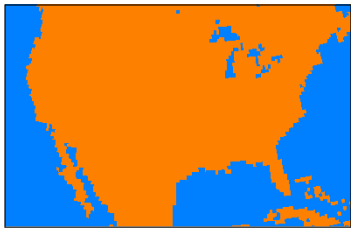
- So, metgrid used the GFS LANDSEA mask even when interpolating AGRMET data!

METGRID.TBL: Real-time System Example

When metgrid interpolated SM000010, it used the GFS landmask for a field masked by the AGRMET landmask!



GFS LANDSEA field



AGRMET LANDSEA field

Note the disagreement between the two data sources near coastlines.

METGRID.TBL: Real-time System Example

Solution:

- Rename LANDSEA to *AGR_LAND* in Vtable.AGRMET
- Rename LANDSEA to *GFS_LAND* in Vtable.GFS
- Create separate entries in METGRID.TBL

  one for GFS SM000010 field

  another for AGRMET SM000010 field

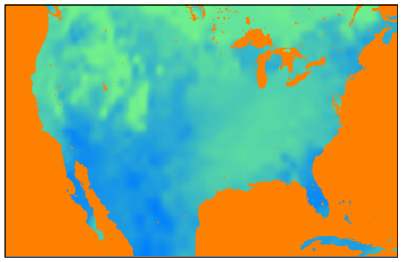
METGRID.TBL: Real-time System Example

```
=======================================
name=SM000010; from_input=GFS
interp_option=sixteen_pt+four_pt+wt_average_4pt+search
masked=water
interp_mask=GFS_LAND(0)
fill_missing=1.
flag_in_output=FLAG_SM000010
=======================================
```

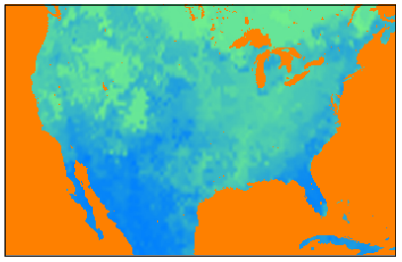
```
=======================================
name=SM000010; from_input=AGRMET
interp_option=sixteen_pt+four_pt+wt_average_4pt+search
masked=water
interp_mask=AGR_LAND(-1.E30)
fill_missing=1.
flag_in_output=FLAG_SM000010
=======================================
```


METGRID.TBL: Real-time System Example

With modified Vtables and METGRID.TBL:



The SM000010 field when only GFS files are available



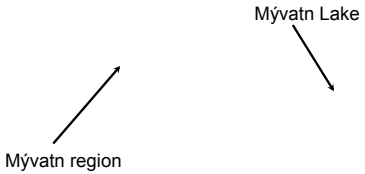
The SM000010 field when both GFS and AGRMET files are available

Outline

- The GEOGRID.TBL file
  - What is the GEOGRID.TBL file?
  - Ingesting new static fields
  - Examples: Using high-resolution land use and topography data
- The METGRID.TBL file
  - What is the METGRID.TBL file?
  - Example: Defining interpolation options for a new field
  - Example: Using the METGRID.TBL file for a real-time system
- Utility programs example: fixing "hot lakes"

Motivating Problem

The "Hot Lake" problem: Inland water bodies that are not resolved by SST data sets often receive extrapolated values from nearby oceans or other resolved water bodies.



Above left: Skin temperature field (TSK) for Iceland and surrounding ocean on 26 January 2011 1200 UTC from NCEP GFS and RTG SST data.

Above right: TSK in the Mývatn region. SST for Mývatn Lake is ~277 K!

Approach

In WRF v3.3 and later, let the *real* preprocessor know which water points are inland water bodies, and provide it a more accurate estimate of SST to be used only over these water bodies.

1) Identify inland water bodies in the land cover data set

2) Provide a suitable proxy for SST field over inland water bodies
   - E.g., Average surface air temperature for X days prior, 273 K for frozen lakes, etc.

3) Modify the SST field in the WRF input file
   - Use new capability in v3.3 real.exe program

Identifying Lakes

Some data sets already identify lakes with separate categories
- MODIS, CORINE

For others, we need a way to do this
- Should be automated
  - don't want to spend long hours clicking on pixels for each data set
- Should be tunable
  - what constitutes a lake will naturally depend on what our SST data set is able to resolve
- Ideally, would not require auxiliary data

In *namelist.wps*, set:
- `geog_data_res = "usgs_lakes+30s"` for USGS land use (16=ocean, 28=lake)
- `geog_data_res = "modis_lakes+30s"` for MODIS land use (17=ocean, 21=lake)

Creating a Proxy SST Field

The *avg_tsfc.exe* utility program may be used to compute the average 2-m air temperature field for any number of full diurnal cycles

- Number of cycles determined by available intermediate files and date range in namelist
- The resulting TAVGSFC intermediate file may be provided to the metgrid program

Test case: Lake Mývatn

To confirm that everything is working as expected, try correcting the temperature for Lake Mývatn in the winter

Grid ID	Resolution	Size
1	16 km	99x99
2	4 km	208x172
3	1 km	136x128
4	250 m	160x160

Ics + BCs from NCEP GFS

Sea surface temperatures from RTG SST

Initial time: 26 January 2011, 1200 UTC

Test case: Lake Mývatn

26 January 2011, 12 UTC

~277.5 K

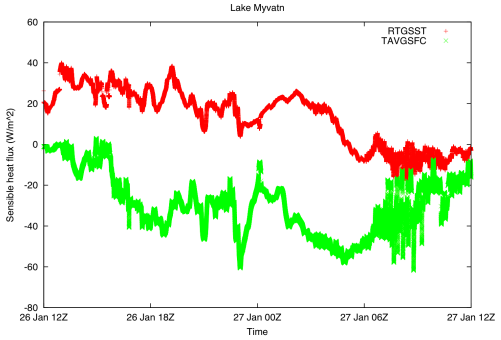
~273.8 K

Initial skin temperature field using extrapolated SST values

Initial skin temperature field using previous 5-day average of 2-m air temperature for lake SST

Test case: Lake Mývatn

Time series of sensible heat flux in the center of the lake show a significant decrease when using a more realistic SST (TAVGSFC)



Latent heat flux time series from simulation using TAVGSFC for SST also shows a decrease from RTG SST time series as well

Summary

- In this lecture, we've seen
  - What the GEOGRID.TBL and METGRID.TBL files do
  - How to use new geographical data sources in the WPS
    - High-resolution land use and topography data
  - How to use the METGRID.TBL file to correct two types of interpolation-related problems
  - How utility programs can be used to improve simulations
- For other features of the WPS, see Chapter 3 of the User's Guide
- For more information about using high-resolution topography data or urban land use data (over the U.S.), see http://www2.mmm.ucar.edu/people/duda/files/how_to_hires.html

Questions?

Other Run-time Options

Wei Wang

WRF:
More Runtime Options

Wei Wang
Nansha, Guangdong, China
December 2015

More options

- Some useful runtime options:
  - Vertical interpolation options
  - IO options
  - Base state parameters
  - Options for long simulations
  - Adaptive-time step
  - Digital filter
  - Global runs
  - Moving nest
  - TC options
  - Tracer / trajectory
  - Optional output
  - Stochastic kinetic-energy backscatter scheme (SKEB)
  - IO quilting
- Time series output (surface and profile)

namelist.input

general namelist records:

- `&time_control`
- `&domains`
- `&physics`
- `&dynamics`
- `&bdy_control`
- `&namelist_quilt`

specialized namelist records:

- `&dfi_control`
- `&fdda`
- `&grib2`
- `&scm`
- `&tc`
- `&noah_mp`

Look for these in **`examples.namelist`**

Vertical interpolation options (1)

Program **`real`** only, optional, &domains:

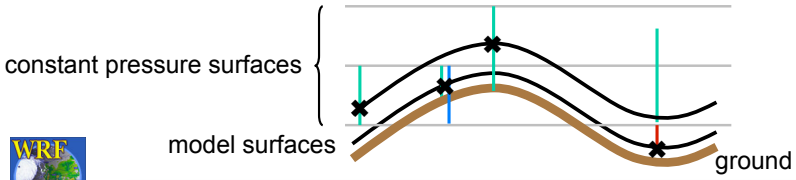
- `use_surface`: whether to use surface observations
- `use_levels_below_ground`: whether to use data below the ground
- `lowest_lev_from_sfc`: logical, whether surface data is used to fill the lowest model level values
- `force_sfc_in_vinterp`: number of levels to use surface data, default is 1
- `extrap_type`: how to do extrapolation: 1 - use 2 lowest levels; 2 - constant
- `t_extrap_type` : extrapolation option for temperature: 1 - isothermal; 2 - 6.5 K/km; 3 - adiabatic

Vertical interpolation options (2)

Program **real** for ARW only, optional:

- **interp_type:in** pressure or log pressure
- **lagrange_order:** linear or quadratic
- **zap_close_levels:** Δp where a non-surface pressure level is removed in vertical interpolation

related namelists: **examples.namelist**



IO Control (1)

History output control in **&time_control**

history_interval:	used often, unit in minutes
history_interval_h:	history output interval in hours
history_interval_s:	history output interval in seconds
history_begin_h:	history output beginning time in hours
history_begin_d:	history output beginning time in days

Look for listing in **Registry/ registry.io_boilerplate**

IO Control (2)

Optional history output in **&time_control**

1. Change Registry.EM and **recompile**:

```
state integer rainc ij misc 1 - h03 "RAINC" ""
  "ACCUMULATED TOTAL CUMULUS PRECIPITATION"
state integer rainnc ij misc 1 - h03 "RAINC" ""
  "ACCUMULATED TOTAL GRID SCALE PRECIPITATION"
```

2. Edit namelist.input to output these variables:

```
auxhist3_outname = "rainfall_d<domain>"
auxhist3_interval = 10, 10,
frames_per_auxhist3 = 1000, 1000,
io_form_auxhist3 = 2
```


IO Control (3)

Starting in V3.2, there is an alternative to add/remove additional output at **runtime** (state variables in Registry only)

1. new namelists in &time_control:

```
iofields_filename(max_dom) = 'my_output.txt',
ignore_iofields_warning = .true.
```

2. prepare a text file ('my_output.txt') to select io fields:

```
+:h:3:rainc,rainnc
```
← syntax in the file

3. set other namelists under **&time_control**:

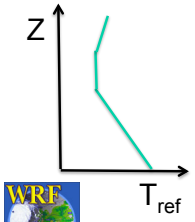
```
auxhist3_outname = "rainfall_d<domain>"
auxhist3_interval = 10, 10,
frames_per_auxhist3 = 1000, 1000,
io_form_auxhist3 = 2
```

See 'Run-Time IO' section in Chapter 5, User's Guide

Base State Parameters

The following could be varied:

`base_temp`	Base state surface temperature
`iso_temp`	Base state stratosphere temperature
`base_pres_strat`	Pressure at which the stratosphere temperature lapse rate changes (since 3.6.1)



Help to improve simulations when model top is higher than 20 km (~ 50 mb)

Options for long simulations (1)

Lower boundary update control: allow SST, seaice, monthly vegetation fraction, and albedo to be updated regularly during a model run:

`sst_update:` 0 – no update
1 – update all above fields

Set before running **`real.exe`**, and this will create additional output files: wrflowinp_d01, wrflowinp_d02, ..

Other namelists required in **`&time_control`**:

`auxinput4_inname` = "wrflowinp_d<domain>"
`auxinput4_interval` = 360, 360,
`io_form_auxinput4` = 2 (netCDF)

See 'Using sst_update Option' in Chapter 5, User's Guide

Options for long simulations (2)

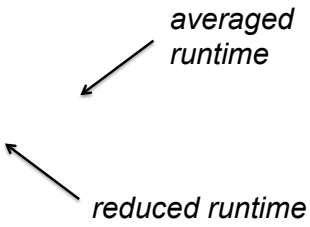
`sst_skin`	diurnal water temp update
`tmn_update`	deep soil temp update, used with lagday
`lagday`	averaging time in days
`bucket_mm`	bucket reset value for rainfall (e.g. **`rainc=i_rainc*bucket_mm+rainc`**)
`bucket_j`	bucket reset value for radiation fluxes
`spec_exp`	exponential multiplier for boundary zone ramping (set in *real*). Usually used with wider boundary zone

Adaptive time steps (1)

- Adaptive-time-step is a way to maximize the model time step while keeping the model numerically stable.
- New in V3. Good to use for real-time run.
- May not work in combination with other options.

Also see 'Using Adaptive Time Stepping' section in Chapter 5, UG

Adaptive time steps (2): an example



On average, forecasts finish in 60 min (50-73min) as compared to 79 min standard runtime

Adaptive time steps (3)

Namelist control: **`&domains`** * USE WITH CARE

`use_adaptive_time_step`	logical switch
`step_to_output_time`	whether to write at exact history output times
`target_cfl`	maximum cfl allowed (1.2)
`max_step_increase_pct`	percentage of time step increase each time; set to 5, 51, 51 (larger value for nest)
`starting_time_step`	in seconds; e.g. set to 4*DX
`max_time_step`	in seconds; e.g. set to 8*DX
`min_time_step`	in seconds; e.g. set to 4*DX

Digital filter initialization (1)

Digital filter initialization is a simple way to remove initial model imbalance:

- May be introduced by simple interpolation, different topography, or by objective analysis, or data assimilation
- It may generate spurious gravity waves in the early simulation hours, which could cause erroneous precipitation, numerical instability and degrade subsequent data assimilation

Digital filter initialization (2)

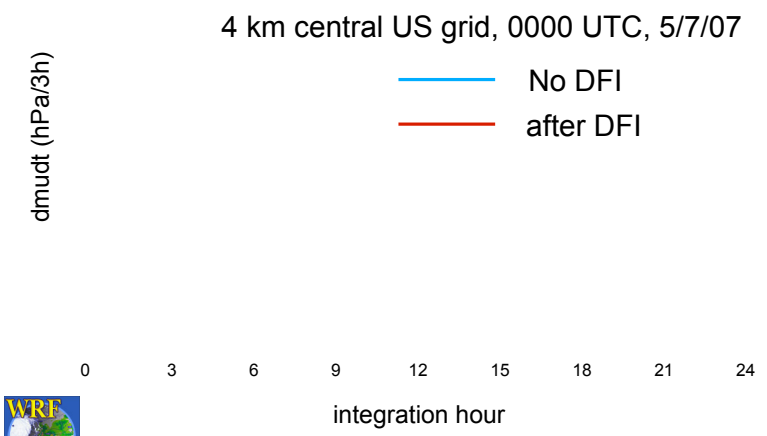
Using DFI

- can construct consistent model fields which do not exist in the initial conditions, e.g. vertical motion, cloud variables
- may reduce the spin-up problem in early simulation hours
- Useful if the first few hours of forecast is the interest

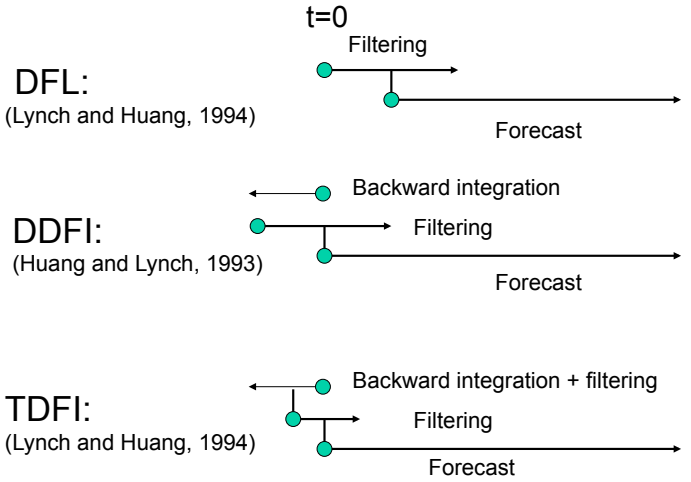
DFI is done after program **`real`**, or data-assimilation step, just before model integration

See 'Using Digital Filter Initialization', Chap 5, UG.

Digital filter initialization (3)



Digital filter initialization (4)



Digital filter inilialization (5)

Namelist control: **`&dfi`**

- `dfi_opt`: dfi options: 0: no DFI; 1: DFL; 2: DDFI; 3: TDFI (recommended)
- `dfi_nfilter`: filter options 0 - 8, recommended: 7
- `dfi_cutoff_seconds` : cutoff period
- `dfi_write_filtered_input` : whether to write filtered IC
- `dfi_bckstop_*` : stop time for backward integration
- `dfi_fwdstop_*` : stop time for forward integration

related namelists: **`examples.namelist`**

To get pressure tendency data, set `diag_print=1` or `2`

Global application

Setup mostly done in WPS:

- `map_proj = 'lat-lon'`
- `e_we, e_sn:` **`geogrid`** will compute dx, dy
- See template `'namelist.wps.global'`

In the model stage:

- `fft_filter_lat`: default value is 45 degrees
- Caution: some options do not work, or have not been tested with global domain. Start with template `'namelist.input.global'`

See 'Global Run' section, Chap 5, UG

Automatic moving nest options

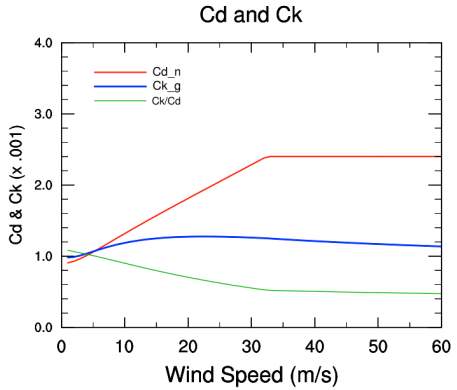
Tropical cyclone / typhoon / hurricane applications:

- `vortex_interval:` time interval when vortex location is estimated
- `max_vortex_speed:` used to compute the search radius for vortex location
- `corral_dist:` how far the vortex can move near the parent domain boundary (number of grids)
- `track_level:` e.g. 700 or 500 mb
- `time_to_move:` hold nests still until this time

See '**Moving Nested Run**', Chap 5, UG

TC options (1)

`isftcflx:` alternative C_d (Donelan) and C_k (=2, Garratt) formulation for TC application



TC options (2)

`sf_ocean_physics=1:` simple ocean mixed layer

`oml_hml0:` initial ocean mixed layer depth

`oml_gamma:` lapse rate in deep water

The ocean mixed layer model can also be initialized with real-data, e.g. HYCOM. More info can be found at

http://www2.mmm.ucar.edu/wrf/users/hurricanes/wrf_ahw.html

TC options (3)

`sf_ocean_physics = 2:`

3D Price-Weller-Pinkel (PWP) ocean model based on Price et al. (1994). It has full ocean process (e.g. advection, pressure-gradient force, and mixing). It doesn't have ocean topography. Only simple initialization is provided in Version 3.5.

tracer option

Add the following in **&dynamics** to activate tracer option (default no. is 8: with array names **tr17_1, tr17_2, …, tr17_8**):

tracer_opt = 2,

One would need some way to initialize the tracer. A simple initialization can be found in program real (**dyn_em/module_initialize_real.F**)

trajectory option

Add the following in **&physics** to activate trajectory option:

traj_opt = 1,

And set the number of trajectories in **&domains**:

num_traj = 25, (default value)

Output: **traj_i(num_traj), traj_j, traj_k, traj_lat, traj_long**

To change initial launch points, edit code in initialization program **real.exe** (**dyn_em/module_initialize_real.F**)

Stochastic kinetic-energy backscatter scheme

This is a way to stochastically perturb forecasts.

stoch_force_opt: = 1, activate the scheme

nens: = N, an integer that controls the random number stream; a different integer will give a differently perturbed forecast

perturb_bdy: = 1, use SKEB pattern; = 2, use user-provided pattern (new in 3.5)

Also see '**Option to stochastically perturb forecasts**' section in Chap 5, UG

(Berner et al. 2011, MWR)

Additional Output Option (1)

prec_acc_dt = 60.:

Output precipitation in a time interval (e.g. 60 min):

PREC_ACC_C, for convective rain

PREC_ACC_NC, for explicit rain

SNOW_ACC_NC, for explicit snow

(May not suitable for use in long runs)

Additional Output Option (2)

Since V3.4.1:

```
&diags
 p_lev_diag = 1.
 num_press_levels = 4,
 press_levels = 85000,70000,50000,20000
```

Output a few met fields on pressure levels :

U_PL, V_PL, S_PL, T_PL, TD_PL, RH_PL, GHT_PL,

Output goes to auxiliary stream 23, so need to set

`auxhist23_outname, io_form_auxhist23,`
`auxhist23_interval, frames_per_auxhist23`

Additional Output Option (3)

`output_diagnostics = 1:`

output max, min, time of max and min, mean value, standard deviation of the mean for 8 surface variables (T2, Q2, TSK, U10, V10, 10 m wind speed, RAINCV, and RAINNCV [time step rain])

`auxhist3_outname =“wrfxtrm_d<domain>_<date>”`
`io_form_auxhist3 = 2`
`auxhist3_interval = 1440, 1440,`
`frame_per_auxhist3 = 10, 10,`

Additional Output Option (4)

`nwp_diagnostics = 1:`

Output max 10 m wind speed, max helicity in 2 – 5 km layer, max w in updraft and downdraft below 400 mb, mean w in 2 – 5 km layer, and max column graupel in a time window between history output times.

Data goes to history file.

Additional Output Option (5)

`do_radar_ref = 1:`

Compute radar reflectivity using parameters used by different microphysics. Works for options mp_physics = 2,4,6,7,8,10,14,16. Option 9, NSSL mp also produce radar reflectivity output.

Data goes to history file.

Additional Output Option (6)

`do_avgflx_em = 1:`

output history-time-averaged, column-pressure-coupled u, v and w:
AVGFLX_RUM, AVGFLX_RVM, AVGFLX_RWM
– useful for driving downstream transport model

Additional Output Option (7) *(extra)*

`afwa_*_opt = 1:` (with sub-options)

output over 60 diagnostic variables to history file (for example, MSLP, precipitable water, cloud cover, etc.)

See Registry/registry.afwa for full listing.

Data goes to history as well as auxhist2 file.

IO quilting: &namelist_quilt

Parallel I/O control:

`nio_tasks_per_group (>0):` allow IO to be done on separate processors. Performance improvement for large domain runs. A value of 2 to 4 works well.

`io_groups (>1):` number of I/O streams that the quilting applies.

See 'Using IO Quilting' section, Chap 5, UG

Other ways to improve IO: 1) p-netCDF; 2) use netCDF4 compression option; 3) use io_form_history=102 to output patches of data

Time Series Output (1)

- It is a special output in text format with file name like
  `prefix.d<domain>.TS`
- It outputs 14 surface variables at every time step:
  e.g. 10 m u/v, 2 m T/qv, precipitation, radiation fluxes, surface fluxes
- One file per location (e.g. at weather station), per domain

Time Series Output (2)

- It also outputs profiles of U, V, Th, Qv, PH (levels set by **max_ts_level**, default 15):

  ***prefix*.d<domain>.UU**

  ***prefix*.d<domain>.VV**

  ***prefix*.d<domain>.TH**

  ***prefix*.d<domain>.QV**

  ***prefix*.d<domain>.PH**

- One file per location (e.g. at weather station), per domain.

Time Series Output (3)

- Not a namelist option
- If output more than 5 locations, use namelist **max_ts_locs**
- Depends the presence of a file called '**tslist**' (a sample of the file is available in **WRFV3/run/**

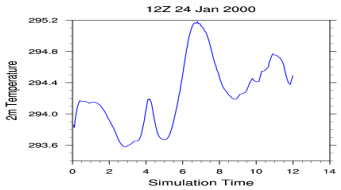
```
#-----------------------------------------------#
# 24 characters for name | pfx |  LAT  |   LON  |
#-----------------------------------------------#
Cape Hallett             hallt -72.330  170.250
McMurdo Station          mcm   -77.851  166.713
```

- This file provides a list of locations where you would like to output time series
- More information in **run/README.tslist** and 'Output Time Series' section, Chapter 5, UG

Time Series Output (4)

Content in `hallt.d01.TS`:

```
Cape Hallett                1  1 hallt ( 36.710, -79.000) (  41,  38)
    ( 36.600, -79.142)  159.6 meters
 1     0.050000     1   41    38     275.47397       0.00288
   3.52110      -2.34275   99988.76563     244.81276
 0.00000     -29.94841       4.09765     273.90295     278.20197
   0.00000       0.00000       0.00000
 1     0.100000     1   41    38     275.56287       0.00282
   3.14414      -2.05875   99956.98438     244.81276
 0.00000     -25.64095       4.18446     273.78323     278.18314
   0.00000       0.00000       0.00000
```



Recommended

Start with the namelist template in a particular test directory, and the options specified in the file, and make modifications.

Chapter 5 of ARW User's Guide, pages 5-32 – 5-34: examples for various applications.

For special applications in ARW, look for related namelists in the file *examples.namelist* in *test/em_real/* directory.

For more information on global extension, DFI and adaptive time step, read Tech Note, and User's Guide.

WRF Software

Dave Gill

WRF Software: Code and Parallel Computing

John Michalakes, Head WRF Software Architecture

Dave Gill

Outline

- WRF architecture – driver, mediation, model
- Need and design for parallelism
- Communication patterns to support parallelism
- Directory structure and file location overview
- Model layer interface
  - The "grid" struct
  - Indices
  - Dereferencing
- I/O

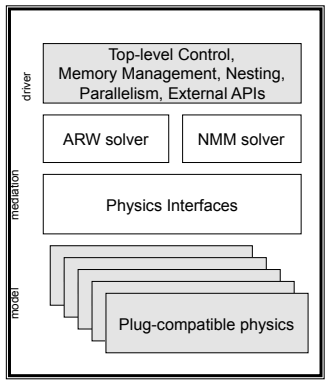
Introduction – WRF Software Characteristics

- Developed from scratch beginning around 1998, primarily Fortran and C
- Requirements emphasize flexibility over a range of platforms, applications, users, performance
- WRF develops rapidly. First released Dec 2000
- Supported by flexible efficient architecture and implementation called the WRF Software Framework

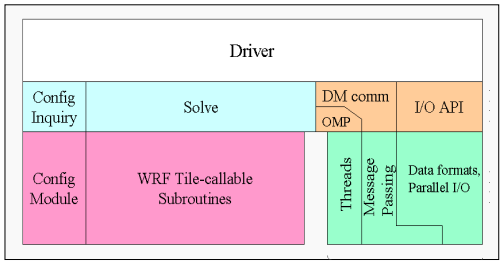
Introduction - WRF Software Framework Overview

- Implementation of WRF Architecture
  - Hierarchical organization
  - Multiple dynamical cores
  - Plug compatible physics
  - Abstract interfaces (APIs) to external packages
  - Performance-portable
- Designed from beginning to be adaptable to today's computing environment for NWP

http://mmm.ucar.edu/wrf/WG2/bench/



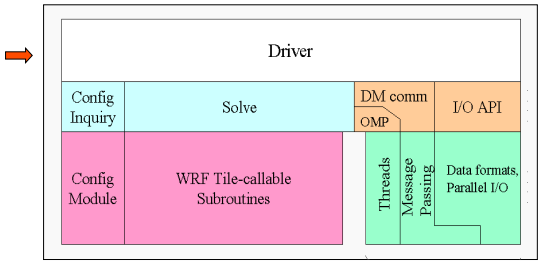
WRF Software Architecture



Registry

- Hierarchical software architecture
  - Insulate scientists' code from parallelism and other architecture/ implementation-specific details
  - Well-defined interfaces between layers, and external packages for communications, I/O, and model coupling facilitates code reuse and exploiting of community infrastructure, e.g. ESMF.

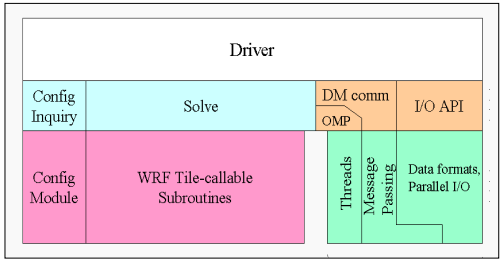
WRF Software Architecture



Registry

- Driver Layer
  - **Domains**: Allocates, stores, decomposes, represents abstractly as single data objects
  - **Time loop**: top level, algorithms for integration over nest hierarchy

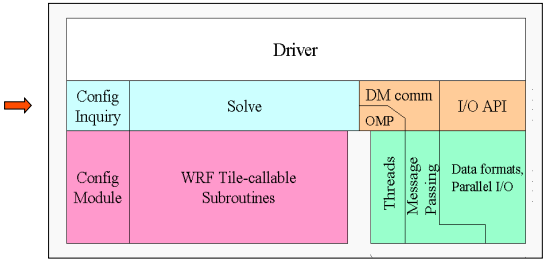
WRF Software Architecture



Registry

- Mediation Layer
  - Solve routine, takes a domain object and advances it one time step
  - Nest forcing, interpolation, and feedback routines

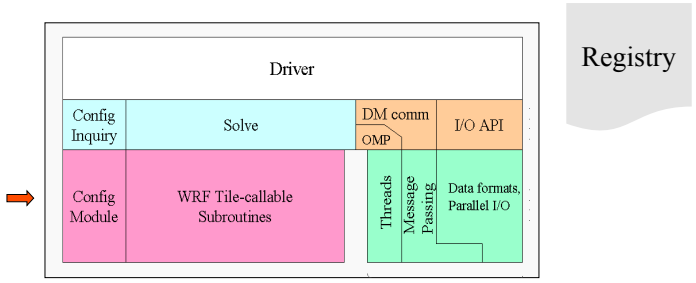
WRF Software Architecture



Registry

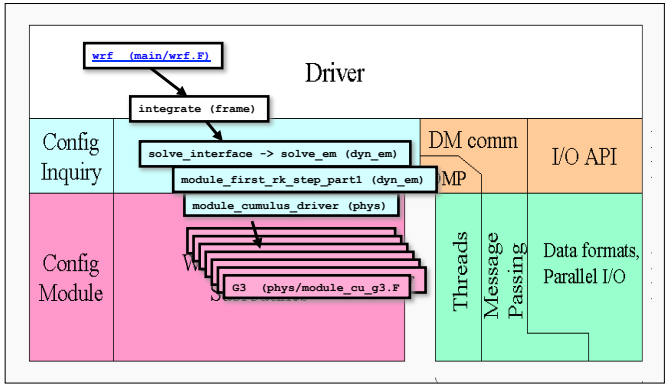
- Mediation Layer
  - The sequence of calls for doing a time-step for one domain is known in Solve routine
  - Dereferences fields in calls to physics drivers and dynamics code
  - Calls to message-passing are contained here as part of Solve routine

WRF Software Architecture



- Model Layer
  - **Physics and Dynamics**: contains the actual WRF model routines are written to perform some computation over an arbitrarily sized/ shaped, 3d, rectangular subdomain

Call Structure Superimposed on Architecture



Hardware: The Computer

- The 'N' in NWP
- Components
  - Processor
    - A program counter
    - Arithmetic unit(s)
    - Some scratch space (registers)
    - Circuitry to store/retrieve from memory device
    - Cache
  - Memory
  - Secondary storage
  - Peripherals
- The implementation has been continually refined, but the basic idea hasn't changed much

Hardware has not changed much…

A computer in 1960

IBM 7090

6-way superscalar
36-bit floating point precision
~144 Kbytes

~50,000 flop/s
48hr 12km WRF CONUS in 600 years

A computer in 2013

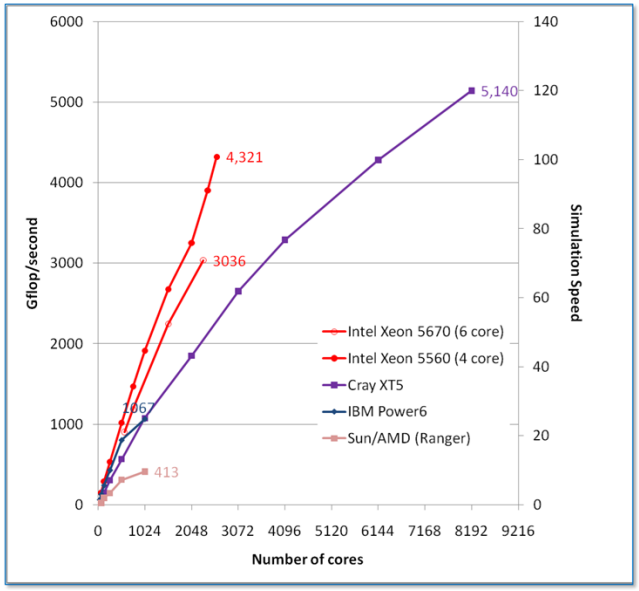
Dual core, 2.6 GHz chip
64-bit floating point precision
20 MB L3

~5,000,000,000 flop/s
48 12km WRF CONUS in 26 Hours

…how we use it has

- Fundamentally, processors haven't changed much since 1960
- Quantitatively, they haven't improved nearly enough
  - 100,000x increase in peak speed
  - 100,000x increase in memory size
- We make up the difference with parallelism
  - Ganging multiple processors together to achieve 10^{11-12} flop/second
  - Aggregate available memories of 10^{11-12} bytes

~1,000,000,000,000 flop/s ~2500 procs
48-h,12-km WRF CONUS in under 15 minutes



January 2000 Benchmark

- 74x61 grid cells
- 1 hour forecast, 3 minute time step, 20 time step average
- IO exlcuded

Decomposed domain sizes proc count: I-dim x J-dim

1: 74x61 2: 74x31 4: 37x31 8: 37x16

16: 19x16 32: 19x8 64: 10x8

January 2000 Benchmark

Processor Count	SM – OpenMP % Efficiency	DM – MPI % Efficiency
1 74x61	100	100
2 74x31	72	98
4 37x31	65	91
8 37x16	31	83
16 19x16	16	70
32 19x8	8	56
64 10x8	3	40

January 2000 Benchmark

- WRF timing estimates may be obtained from the model print-out
- Serial

```
Timing for main on domain   1:   32.16074 elapsed seconds
```

- OpenMP

```
Timing for main on domain   1:    8.56216 elapsed seconds
```

- MPI

```
Timing for main on domain   1:    7.36243 elapsed seconds
```

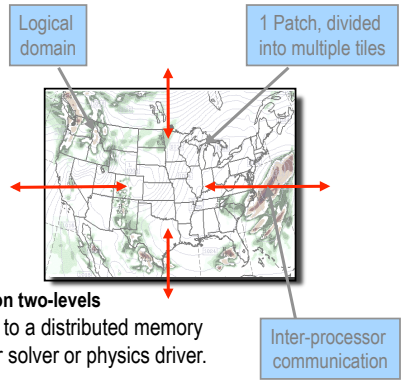
- Get enough time steps to include "day-time" radiation, and to have the microphysics "active" for better estimates

Application: WRF

- WRF can be run serially or as a parallel job
- WRF uses ***domain decomposition*** to divide total amount of work over parallel processes

Parallelism in WRF: Multi-level Decomposition

- **Single version of code for efficient execution on:**
  - Distributed-memory
  - Shared-memory (SMP)
  - Clusters of SMPs
  - Vector and microprocessors



Model domains are decomposed for parallelism on two-levels

Patch: section of model domain allocated to a distributed memory node, this is the scope of a mediation layer solver or physics driver.

Tile: section of a patch allocated to a shared-memory processor within a node; this is also the scope of a model layer subroutine.

Distributed memory parallelism is over patches; shared memory parallelism is over tiles within patches

Distributed Memory Communications

When Needed?

Communication is required between patches when a horizontal index is incremented or decremented on the right-hand-side of an assignment.

Why?

On a patch boundary, the index may refer to a value that is on a different patch.

Following is an example code fragment that requires communication between patches

Signs in code

Note the tell-tale +1 and –1 expressions in indices for **rr**, **H1,** and **H2** arrays on right-hand side of assignment.

These are ***horizontal data dependencies*** because the indexed operands may lie in the patch of a neighboring processor. That neighbor's updates to that element of the array won't be seen on this processor.

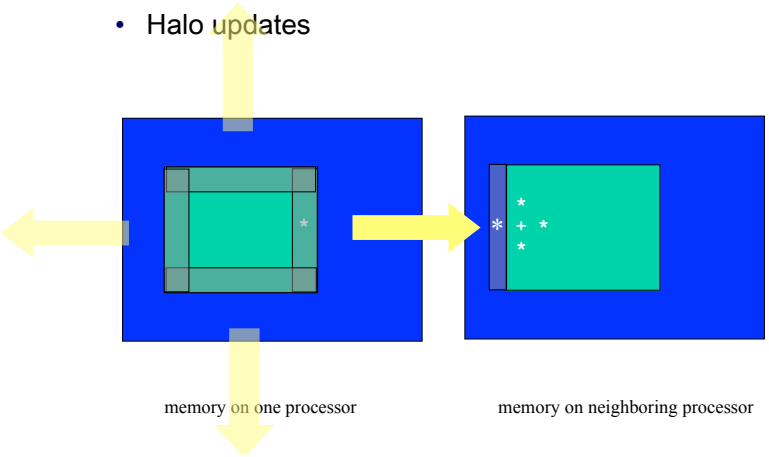
Distributed Memory Communications

(module_diffusion.F)

```
SUBROUTINE horizontal_diffusion_s (tendency, rr, var, . . .
. . .
   DO j = jts,jte
   DO k = kts,ktf
   DO i = its,ite
      mrdx=msft(i,j)*rdx
      mrdy=msft(i,j)*rdy
      tendency(i,k,j)=tendency(i,k,j)-                              &
           (mrdx*0.5*((rr(i+1,k,j)+rr(i,k,j))*H1(i+1,k,j)-          &
                      (rr(i-1,k,j)+rr(i,k,j))*H1(i   ,k,j))+        &
            mrdy*0.5*((rr(i,k,j+1)+rr(i,k,j))*H2(i,k,j+1)-          &
                      (rr(i,k,j-1)+rr(i,k,j))*H2(i,k,j   ))-        &
            msft(i,j)*(H1avg(i,k+1,j)-H1avg(i,k,j)+                 &
                       H2avg(i,k+1,j)-H2avg(i,k,j)                  &
                             )/dzetaw(k)                            &
          )
   ENDDO
   ENDDO
   ENDDO
. . .
```

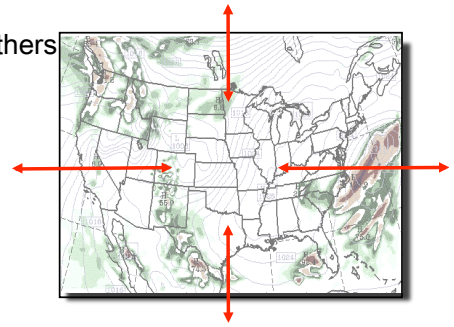
Distributed Memory MPI Communications

- Halo updates



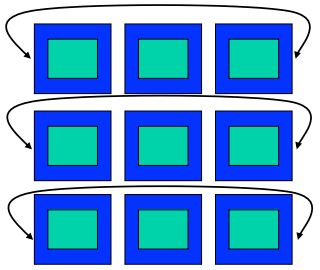
Distributed Memory (MPI) Communications

- Halo updates
- Periodic boundary updates
- Parallel transposes
- Nesting scatters/gathers

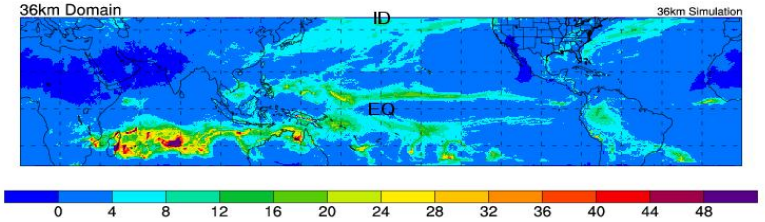


Distributed Memory (MPI) Communications

- Halo updates
- Periodic boundary updates
- Parallel transposes
- Nesting scatters/gathers

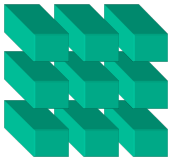


Average Daily Total rainfall (mm) - March 1997



Distributed Memory (MPI) Communications

- Halo updates
- Periodic boundary updates
- **Parallel transposes**
- Nesting scatters/gathers



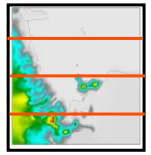
all y on patch

all z on patch

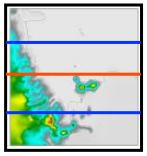
all x on patch

Distributed Memory (MPI) Communications

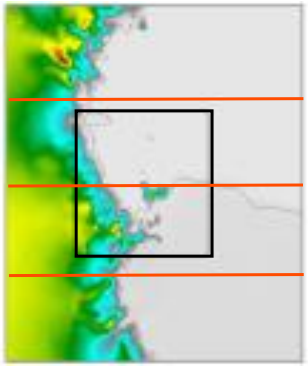
- Halo updates
- Periodic boundary updates
- Parallel transposes
- **Nesting scatters/gathers**



NEST:2.22 km



INTERMEDIATE: 6.66 km



COARSE
Ross Island
6.66 km

WRF Model Top-Level Directory Structure

[WRF Design and Implementation Doc, p 5](#)

DRIVER
MEDIATION
MODEL

```
Makefile
README
README_test_cases
clean                 }
compile               }  build scripts
configure             }
Registry/             CASE input files
arch/                 machine build rules
dyn_em/               }
dyn_nnm/              }
external/             }
frame/                }  source code directories
inc/                  }
main/                 }
phys/                 }
share/                }
tools/                }
run/                  }  execution directories
test/                 }
```

Where are WRF source code files located?

```
$(RM) $@

$(CPP) -I$(WRF_SRC_ROOT_DIR)/inc \
  $(CPPFLAGS) $(OMPCPP) $*.F  > $*.f90

$(FC) -o $@ -c $(FCFLAGS) $(MODULE_DIRS) \
  $(PROMOTION) $(FCSUFFIX) $*.f90
```

Where are WRF source code files located?

```
cpp –C –P file.F  > file.f90
gfortran –c file.f90
```

Where are WRF source code files located?

- The most important command is the "find" command. If there is an error in the model output, you can find that location in the source code with the **find** command.

```
cd WRFV3
find . -name \*.F -exec grep -i "Flerchinger" {} \; -print
```

Where are WRF source code files located?

- All of the differences between the .F and .f90 files are due to the included pieces that are manufactured by the Registry.
- These additional pieces are all located in the WRFV3/inc directory.
- For a serial build, almost 450 files are manufactured.
- Usually, most developers spend their time working with physics schemes.

Where are WRF source code files located?

- The "main" routine that handles the calls to all of the physics and dynamics:
  - WRFV3/dyn_em/solve_em.F
- This "solver" is where the tendencies are initialized to zero, some pre-physics terms are computed, and the time stepping occurs
- The calls to most of the physics schemes are made from a further call down the call tree
  - dyn_em/module_first_rk_step_part1.F

Where are WRF source code files located?

- Inside of solve_em and first_rk_step_part1, all of the data is located in the "grid" structure: grid%ht.
- The dimensions in solve_em and first_rk_step_part1 are "d" (domain), and "m" (memory):

  ids, ide, jds, jde, kds, kde

  ims, ime, jms, jme, kms, kme
- The "t" (tile) dimensions are computed in first_rk_step_part1 and passed to all drivers.
- WRF uses global indexing

Where are WRF source code files located?

- If you are interested in looking at physics, the WRF system has organized the files in the WRFV3/phys directory.
- In WRFV3/phys, each type of physics has a driver:

module_cumulus_driver.F	cu
module_microphysics_driver.F	mp
module_pbl_driver.F	bl
module_radiation_driver.F	ra
module_surface_driver.F	sf

Where are WRF source code files located?

- The subgrid-scale precipitation (*_cu_*.F)

module_cu_bmj.F	module_cu_camzm.F
module_cu_g3.F	module_cu_gd.F
module_cu_kf.F	module_cu_kfeta.F
module_cu_nsas.F	module_cu_osas.F
module_cu_sas.F	module_cu_tiedtke.F

Where are WRF source code files located?

- Advection

  WRFV3/dyn_em/module_advect_em.F

- Lateral boundary conditions

  WRFV3/dyn_em/module_bc_em.F

Where are WRF source code files located?

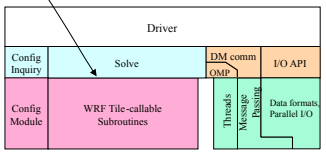
- Compute various RHS terms, pressure gradient, buoyancy, w damping, horizontal and vertical diffusion, Coriolis, curvature, Rayleigh damping
  WRFV3/dyn_em/module_big_step_utilities_em.F
- All of the sound step utilities to advance u, v, mu, t, w within the small time-step loop
  WRFV3/dyn_em/module_small_step_em.F

WRF Model Layer Interface – The Contract with Users

All state arrays passed through argument list as simple (not derived) data types

Domain, memory, and run dimensions passed unambiguously in three dimensions

Model layer routines are called from mediation layer (physics drivers) in loops over tiles, which are multi-threaded



WRF Model Layer Interface – The Contract with Users

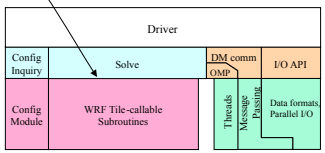
Restrictions on Model Layer subroutines:

No I/O, communication

No stops or aborts
Use wrf_error_fatal

No common/module storage of decomposed data

Spatial scope of a Model Layer call is one "tile"



WRF Model Layer Interface

```
SUBROUTINE driver_for_some_physics_suite (
    . . .
!$OMP DO PARALLEL
   DO ij = 1, numtiles
      its = i_start(ij) ; ite = i_end(ij)
      jts = j_start(ij) ; jte = j_end(ij)
      CALL model_subroutine( arg1, arg2, . . .
           ids , ide , jds , jde , kds , kde ,
           ims , ime , jms , jme , kms , kme ,
           its , ite , jts , jte , kts , kte )
   END DO
    . . .

 END SUBROUTINE
```

WRF Model Layer Interface

template for model layer subroutine

```
SUBROUTINE model_subroutine ( &
  arg1, arg2, arg3, … , argn,   &
  ids, ide, jds, jde, kds, kde, &  ! Domain dims
  ims, ime, jms, jme, kms, kme, &  ! Memory dims
  its, ite, jts, jte, kts, kte  )  ! Tile dims

IMPLICIT NONE

! Define Arguments (State and I1) data
REAL, DIMENSION (ims:ime,kms:kme,jms:jme) :: arg1, . . .
REAL, DIMENSION (ims:ime,jms:jme)         :: arg7, . . .
 . . .
! Define Local Data (I2)
REAL, DIMENSION (its:ite,kts:kte,jts:jte) :: loc1, . . .
 . . .
```

WRF Model Layer Interface

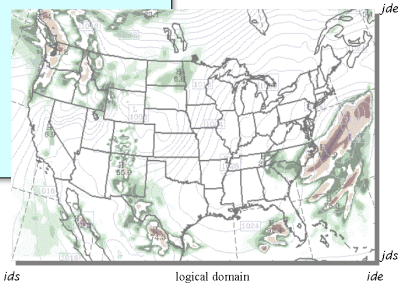
template for model layer subroutine

```
. . .
 ! Executable code; loops run over tile
 ! dimensions
 DO j = jts, MIN(jte,jde-1)
   DO k = kts, kte
     DO i = its, MIN(ite,ide-1)
       loc1(i,k,j) = arg1(i,k,j) + …
     END DO
   END DO
 END DO
```

template for model layer subroutine

```
SUBROUTINE model ( &
  arg1, arg2, arg3, … , argn,   &
  ids, ide, jds, jde, kds, kde, &  ! Domain dims
  ims, ime, jms, jme, kms, kme, &  ! Memory dims
  its, ite, jts, jte, kts, kte  )  ! Tile dims

IMPLICIT NONE

! Define Arguments (S and I1) data
REAL, DIMENSION (ims:ime,kms:kme,jms:jme) :: arg1, . . .
REAL, DIMENSION (ims:ime,jms:jme)         :: arg7, . . .
 . . .
! Define Local Data (I2)
REAL, DIMENSION (its:ite,kts:kte,jts:jte) :: loc1, . . .
 . . .
! Executable code; loops run over tile
! dimensions
DO j = MAX(jts,jds), MIN(jte,jde-1)
  DO k = kts, kte
    DO i = MAX(its,ids), MIN(ite,ide-1)
      loc1(i,k,j) = arg1(i,k,j) + …
    END DO
  END DO
END DO
```

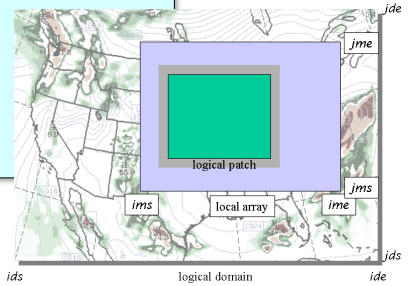
- Domain dimensions
  - Size of logical domain
  - Used for bdy tests, etc.



template for model layer subroutine

```
SUBROUTINE model ( &
  arg1, arg2, arg3, … , argn,   &
  ids, ide, jds, jde, kds, kde, &  ! Domain dims
  ims, ime, jms, jme, kms, kme, &  ! Memory dims
  its, ite, jts, jte, kts, kte  )  ! Tile dims

IMPLICIT NONE

! Define Arguments (S and I1) data
REAL, DIMENSION (ims:ime,kms:kme,jms:jme) :: arg1, . . .
REAL, DIMENSION (ims:ime,jms:jme)         :: arg7, . . .
 . . .
! Define Local Data (I2)
REAL, DIMENSION (its:ite,kts:kte,jts:jte) :: loc1, . . .
 . . .
! Executable code; loops run over tile
! dimensions
DO j = MAX(jts,jds), MIN(jte,jde-1)
  DO k = kts, kte
    DO i = MAX(its,ids), MIN(ite,ide-1)
      loc1(i,k,j) = arg1(i,k,j) + …
    END DO
  END DO
END DO
```

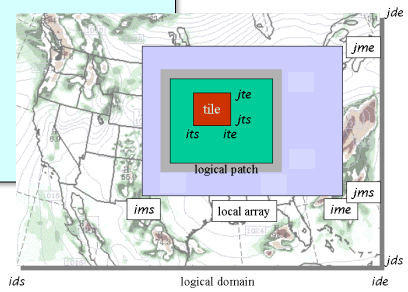
- Domain dimensions
  - Size of logical domain
  - Used for bdy tests, etc.
- Memory dimensions
  - Used to dimension dummy arguments
  - Do not use for local arrays



template for model layer subroutine

```
SUBROUTINE model ( &
  arg1, arg2, arg3, … , argn,   &
  ids, ide, jds, jde, kds, kde, &  ! Domain dims
  ims, ime, jms, jme, kms, kme, &  ! Memory dims
  its, ite, jts, jte, kts, kte  )  ! Tile dims

IMPLICIT NONE

! Define Arguments (S and I1) data
REAL, DIMENSION (ims:ime,kms:kme,jms:jme) :: arg1, . . .
REAL, DIMENSION (ims:ime,jms:jme)         :: arg7, . . .
 . . .
! Define Local Data (I2)
REAL, DIMENSION (its:ite,kts:kte,jts:jte) :: loc1, . . .
 . . .
! Executable code; loops run over tile
! dimensions
DO j = MAX(jts,jds), MIN(jte,jde-1)
  DO k = kts, kte
    DO i = MAX(its,ids), MIN(ite,ide-1)
      loc1(i,k,j) = arg1(i,k,j) + …
    END DO
  END DO
END DO
```

- Domain dimensions
  - Size of logical domain
  - Used for bdy tests, etc.
- Memory dimensions
  - Used to dimension dummy arguments
  - Do not use for local arrays
- Tile dimensions
  - Local loop ranges
  - Local array dimensions

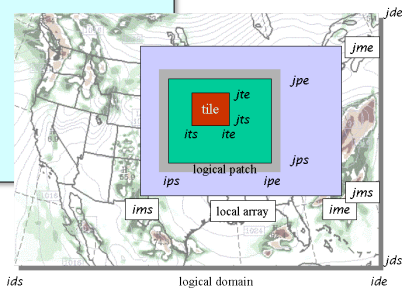


template for model layer subroutine

```
SUBROUTINE model ( &
  arg1, arg2, arg3, … , argn,   &
  ids, ide, jds, jde, kds, kde, &  ! Domain dims
  ims, ime, jms, jme, kms, kme, &  ! Memory dims
  its, ite, jts, jte, kts, kte  )  ! Tile dims

IMPLICIT NONE

! Define Arguments (S and I1) data
REAL, DIMENSION (ims:ime,kms:kme,jms:jme) :: arg1, . . .
REAL, DIMENSION (ims:ime,jms:jme)         :: arg7, . . .
 . . .
! Define Local Data (I2)
REAL, DIMENSION (its:ite,kts:kte,jts:jte) :: loc1, . . .
 . . .
! Executable code; loops run over tile
! dimensions
DO j = MAX(jt,jds), MIN(jte,jde-1)
  DO k = kts, kte
    DO i = MAX(its,ids), MIN(ite,ide-1)
      loc1(i,k,j) = arg1(i,k,j) + …
    END DO
  END DO
END DO
```

- Domain dimensions
  - Size of logical domain
  - Used for bdy tests, etc.
- Memory dimensions
  - Used to dimension dummy arguments
  - Do not use for local arrays
- Tile dimensions
  - Local loop ranges
  - Local array dimensions

- Patch dimensions
  - Start and end indices of local distributed memory subdomain
  - Available from mediation layer (solve) and driver layer; not usually needed or used at model layer



WRF I/O

- Streams: pathways into and out of model
- Can be thought of as files, though that is a restriction
  - History + auxiliary output streams (10 and 11 are reserved for nudging)
  - Input + auxiliary input streams (10 and 11 are reserved for nudging)
  - Restart, boundary, and a special DA in-out stream
  - Currently, 24 total streams
  - Use the large values and work down to stay away from "used"

WRF I/O

- Attributes of streams
  - Variable set
    - The set of WRF state variables that comprise one read or write on a stream
    - Defined for a stream at compile time in Registry
  - Format
    - The format of the data outside the program (e.g. NetCDF), split
    - Specified for a stream at run time in the namelist

WRF I/O

- Attributes of streams
  - Additional namelist-controlled attributes of streams
    - Dataset name
    - Time interval between I/O operations on stream
    - Starting, ending times for I/O (specified as intervals from start of run)

WRF I/O

- Attributes of streams
  - Mandatory for stream to be used:
    - Time interval between I/O operations on stream
    - Format: io_form

Outline

- WRF architecture – driver, mediation, model
- Need and design for parallelism
- Communication patterns to support parallelism
- Directory structure and file location overview
- Model layer interface
  - The "grid" struct
  - Indices
  - Dereferencing
- I/O

Best Practices (Part I)

Wei Wang

Considerations for Designing an Numerical Experiment

Wei Wang

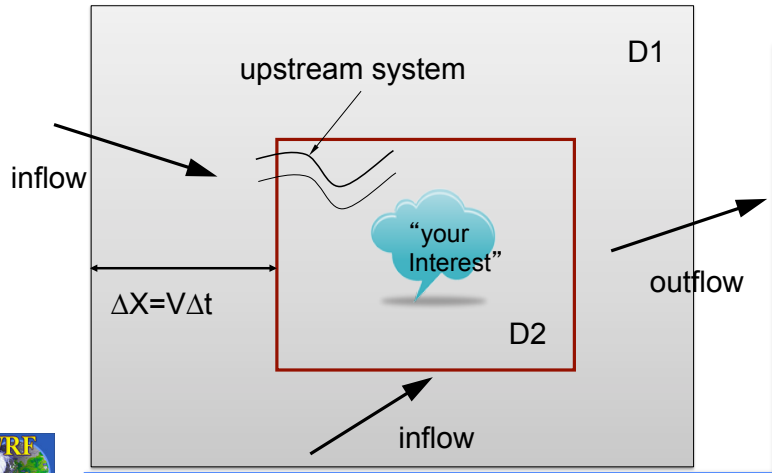
Nansha, Guangdong, China

December 2015

Domains

- In general,
  - IC is more important for simulations of a few days;
  - BC is more important for longer simulations.
- How large do they need to be?
  - Should not be too small, otherwise solution will be determined by forcing data
  - No less than 100x100 (at least 10 grid points are in the boundary zone)
- Where to place my lateral boundaries?
  - Avoid steep topography
  - Away from the area of interest

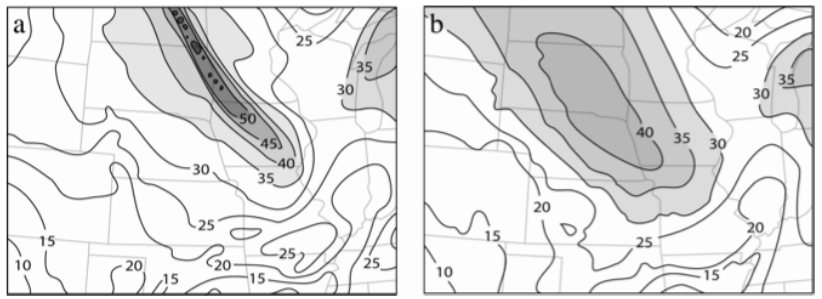
Note on Configuring Domains: Horizontal



Note on Configuring Domains: *Effect of domain sizes*

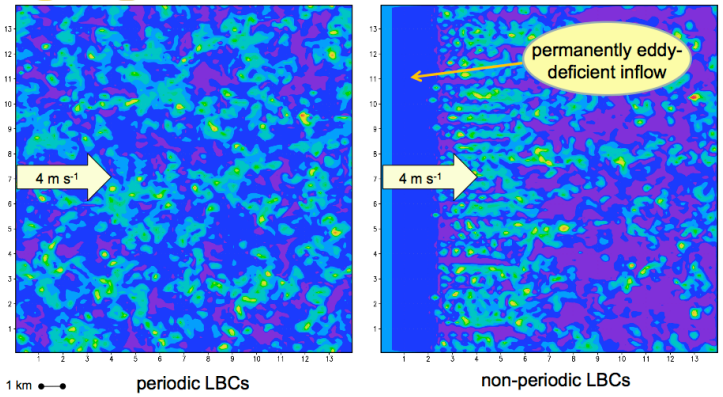
Large regional domain

Smaller regional domain



(From Warner, 2011)

Note on Configuring Domains:
Effect of lateral boundary conditions

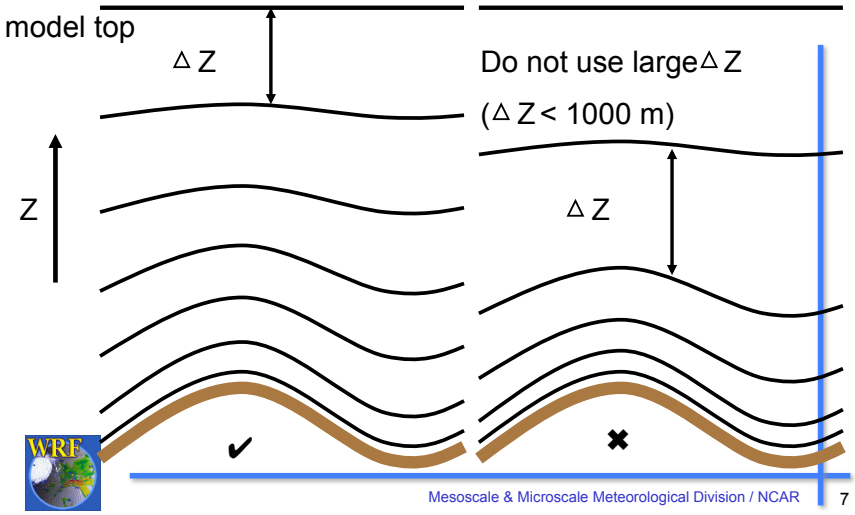


(From Gaudet et al. WRF Users' Workshop 2012, talk 3.5)

Domains

- How many vertical levels should I use?
  - At least 30 or more levels for model top at 50 mb
  - 50 mb model top is recommended
  - Vertical grid distance should not be larger than 1000 m:
    - Radiation, microphysics, less accurate lateral BC
  - Related to horizontal grid size too: if finer horizontal grid size is used, consider adding a few more levels in the vertical
  - Make sure dz < dx

Note on Configuring Domains:
Vertical levels



Domains

- Consider the placement of your domains:
  - What map projection to use?
  - Check the range of the map scale factor after running *geogrid*
    - Values should be close to 1

* Placement of the domain will affect the time step used in the model.

Nests:

- When should I use nests?
  Some of the reasons may be:
  - Input data resolution is too coarse
  - Input data may not be adequate as LBC
  - There isn't sufficient computing resources
- Nest domain sizes should not be too small;
- Nest boundary should be kept away from coarse domain boundary, and steep topography;
- If you use a nest, do not save on coarse domain – it's cheap (and may scale better when using large number of processors)

Input Data

- Check land data:
  e.g. landuse: *does it represent my area well?*
- Know about the data: *how good are the data?*
  - Forecast data
  - Reanalysis data
  - Climate model data
- How frequent do I need to have boundary conditions?
  - More frequent is better

* Good data will go a long way to ensure good outcome.

Model Options

- What do I start with?
  - What other people have success with?
    - References, papers
    - Consider well-tested options first
  - Simple options first:
    For example,
    - Graupel may not be important if dx >> 10 km
    - mixed layer ocean model may not be needed if the modeled track isn't correct
    - Use analyses from weather centers before trying to create your own (via either *obsgrid* or DA) for both initial and lateral boundary conditions
    - Single domain first, before using many nests

Model Options

  - Choose physics for appropriate grid sizes
    - Use a cumulus scheme if grid size > 10 km
    - A cumulus scheme isn't needed when grid size < 4 km
    - Avoid grid sizes 5 – 10 km
    - Use a PBL for grid size > 500 m
    - Use LES options for grid size < 100 m
  - Consider other options:
    For example,
    - Upper level damping over topography
    - Gravity-wave drag if resolution is coarse
    - Slope effect on radiation when grid size < 2 km

Verification:

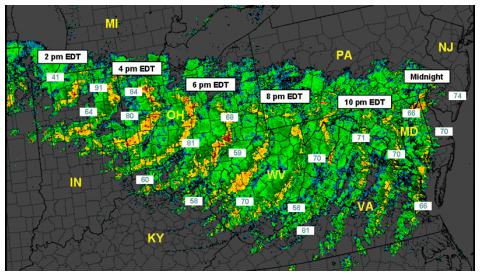
- Important to verify:
  - Knowing where model is biased can be very useful
- Verifying high-resolution model can be tricky:
  e.g. phase error, which punishes higher resolution model more
  - Neighborhood method more appropriate

A forecast example

-- What can we learn from this example?

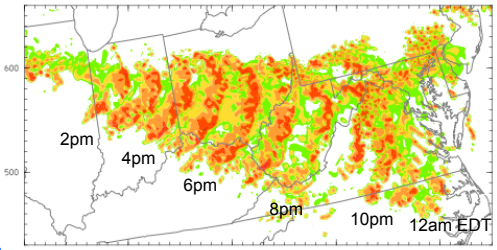
Derecho forecast from NCAR's 2012 RT

Observed radar composite

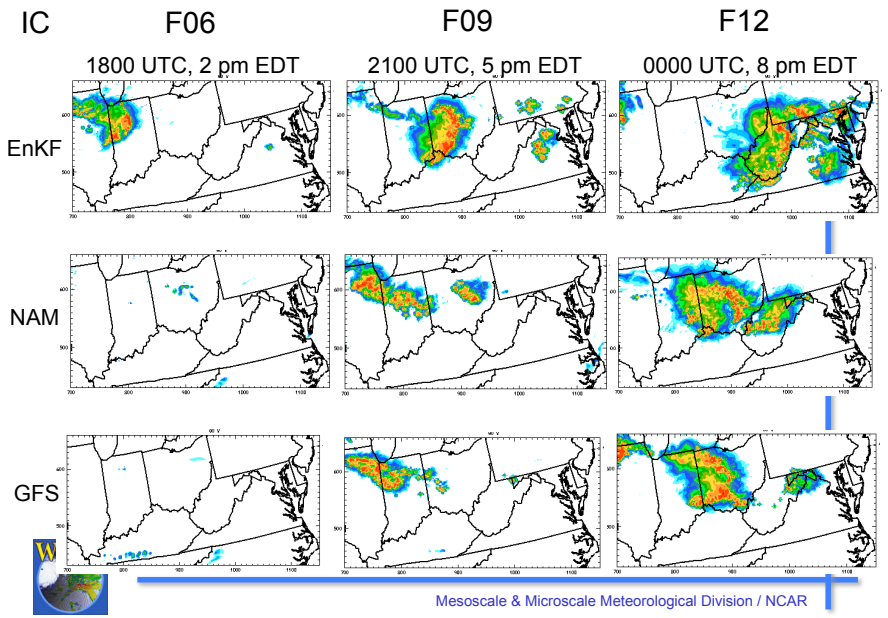


(from NOAA/SPC)

Forecast max-column reflectivity from 3 km model, starting from 1200 UTC June 29

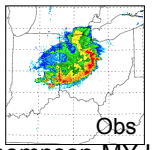


IC: Fully cycled analysis starting from late April using WRF-DART

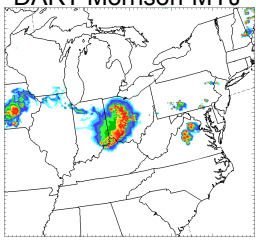


Sensitivity to physics and initial conditions:

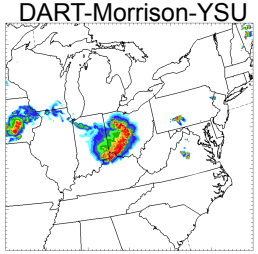
29 June 2012
Derecho



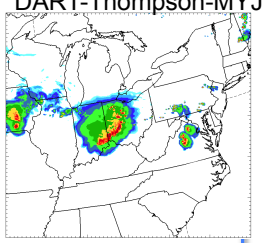
DART-Morrison-MYJ



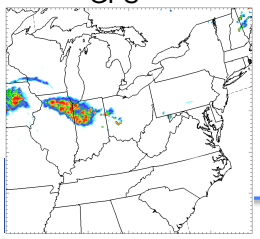
DART-Morrison-YSU



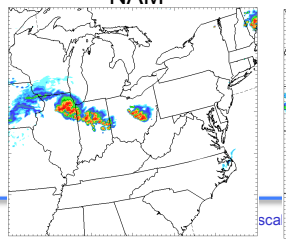
DART-Thompson-MYJ



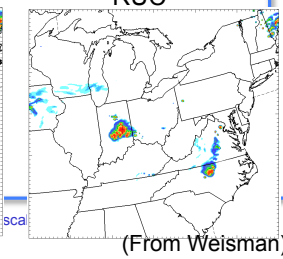
GFS



NAM



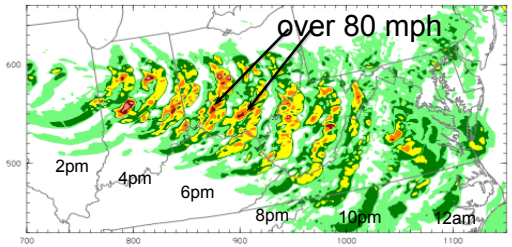
RUC



(From Weisman)

Resolution Differences: simulated max winds

3 km results



15 km results

NCAR Real-time Forecast Domain (2013)

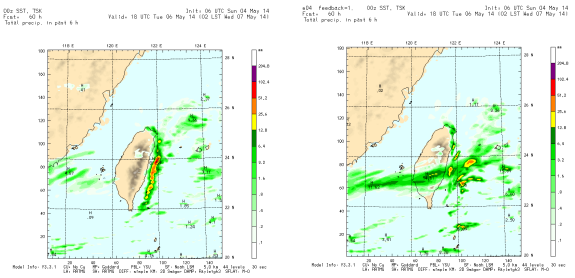
15 km

3 km

What this case show:

- Initial conditions are important
- Different initial conditions will likely give different solutions
- Compared to model runs using different physics options, changing initial conditions is likely to have larger impact
- Model resolution matters

An example of nest feedback



feedback = 0 feedback = 1

60-h forecast of the 6-h rainfall for the period ending 18 UTC 6 May 2014 (courtesy of J. Bresch)

Bottomline..

- Model results can be affected by many choices:
  - Domain configuration, both horizontal and vertical;
  - Input data;
  - Initial and lateral boundary conditions.
- Model has limitations:
  - Physics: biases, may not represent certain process well, etc.
  - Limitation of the lateral boundaries
- **Always check the output after each program**

***Other Best Practice Reading**:*

- *"12 steps toward improving the outcome" by C. Davis:* http://www2.mmm.ucar.edu/wrf/users/workshops/WS2012/ppts/discussion1.pdf
- *"WRF Advanced usage and Best Practices" by Dudhia and Wang:* http://www2.mmm.ucar.edu/wrf/users/workshops/WS2014/ppts/best_prac_wrf.pdf

***References**:*

Numerical Weather and Climate Prediction, 2011. By Thomas Warner, *Cambridge University Press*.

Warner, T., 2011. Quality assurance in atmospheric modeling. *Bull. Amer. Met. Soc. Dec. issue, p1601 – 1611.*

Stensrud, D., 2007. Parameterization Schemes: Keys to Understanding Numerical Weather Prediction Models. *Cambridge University Press*.

Haltiner G. and R. Williams, 1980. Numerical Prediction and Dynamic Meteorology. *Wiley*.

Best Practices (Part II)

Jimy Dudhia

WRF Advanced Usage and Best Practices

Jimy Dudhia
Wei Wang
NCAR/MMM

Motivation

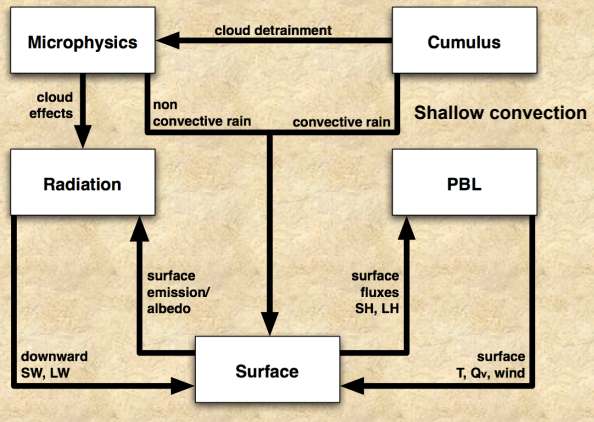
- This talk is motivated by our user support questions
  - We often find problems are due to wrong usages of the model
  - Many questions on how to do various advanced applications
  - We hope to address some here
  - Can't be comprehensive (questions can be asked later)

Topics

- Physics: So many options to choose from
- Complex terrain
- Nesting, resolution and domain sizes
- Model levels and high tops
- Nudging options: use or not
- Initialization and spin-up issues
- Damping and advection options

Physics

Direct Interactions of Parameterizations



Physics

- All WRF options enable the basic interactions outlined in previous figure (no "wrong" combination in that sense)

However

- Consider tried and trusted schemes first
  - see papers on similar uses of WRF
  - See example in Users' Guide
- Consider what remains unresolved or unrepresented – WRF may have options to help
  - Subgrid cloud effects, aerosol effects on clouds/ radiation, radiation-microphysics coupling

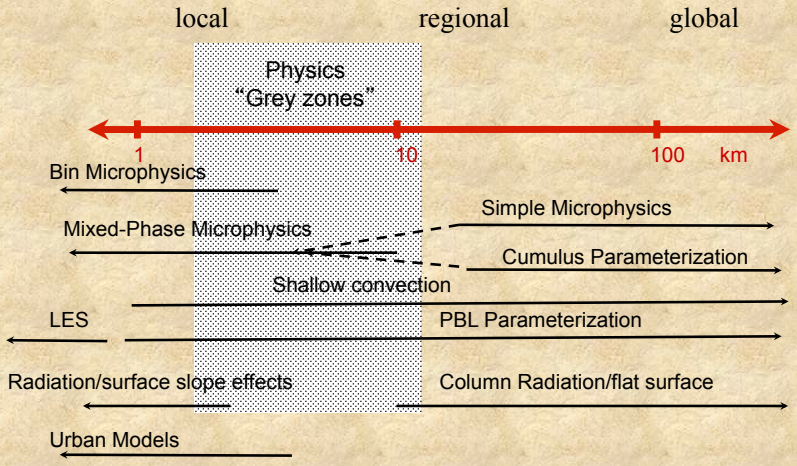
Physics Interactions

- Not all physics options fully interact with each other, notably
  - All microphysics but only some cumulus schemes provide cloud fraction to radiation schemes
  - Only some radiation schemes have aerosol, ozone, or greenhouse gas climatology or variation
  - Only some radiation, PBL, cumulus and microphysics schemes interact with WRF-Chem
  - Only some radiation schemes accept cloud particle size information from some microphysics schemes
  - Note: RRTMG is used as our standard and has most of these interactions added before other radiation options

Special Options

- WRF has many specialized options:
  - Hurricanes, moving nests, surface fluxes and slab ocean
  - Fractional sea ice
  - Urban physics
  - Mosaic sub-grid land categories
  - Windfarm drag
  - Solar outputs (diffuse/direct) and inputs (specified aerosols)
  - Bin microphysics
  - 3d multi-layer ocean
  - Lake model
  - Trajectories and tracers
  - Stochastic perturbations (SKEBS)
  - Single-column model

Physics in Multiscale Model

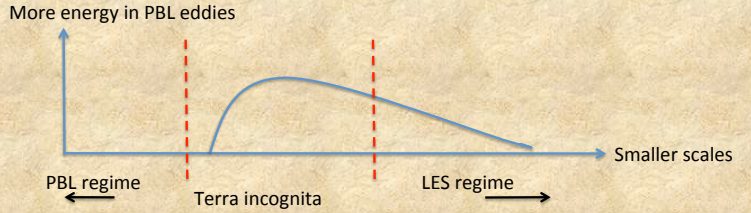


Physics

- Consider grid size when choosing sophistication of microphysics
  - Don't need complex scheme for 10 km grid
  - Do need at least graupel for convection-resolving grids
- When to use cumulus parameterization
  - Grid size > 10 km - yes
  - Grid size < 4 km - probably not
  - Perhaps best to avoid grid sizes 5-10 km for convective cases

LES Modeling

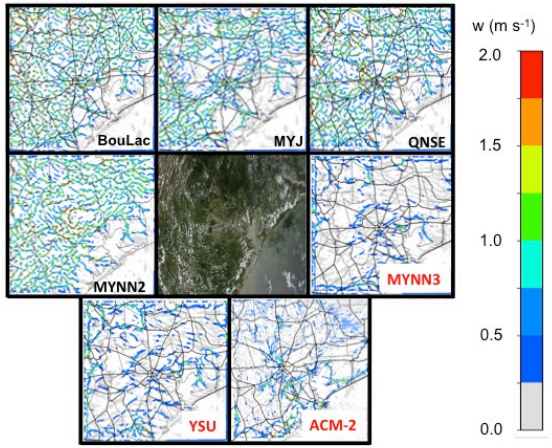
- "Terra Incognita" range of grid sizes where main PBL eddies are partially resolved
  - PBL assumes all eddies are unresolved
  - LES assumes eddies are well resolved



Boundary-Layer Rolls

Mesoscale simulations are sensitive to choice of PBL parameterization options

Ching et al. 2014



WRF simulations of vertical velocity in the PBL (125-m, Level 10) for 20 UTC August 4, 2006 over Houston-Galveston Texas area

Satellite image is from Terra 17:20 UTC, 500-m pixels:

PBL schemes 1: Vertical fluxes are proportional to local vertical gradients:
- BouLac
- MYJ
- QNSE
- MYNN2

PBL schemes 2: Vertical fluxes using non-local closure schemes:
- MYNN3
- YSU
- ACM-2

Physics

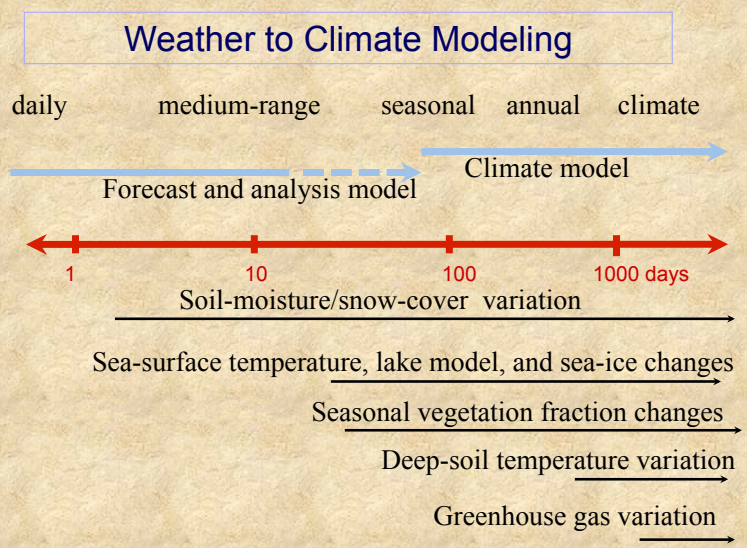
- When to use LES
  - Grid size > 500 m – use PBL
  - Grid size < 100 m – use LES
  - Grid size 100-500 m – either may work to some extent
- Important note: keep dz < dx
  - Particularly applies to LES with real data where model levels stretch with height
  - Can lead to significant noise at top if not done
- When to use slope radiation effects
  - When slope is resolved and significant (dx < 2 km probably)

PBL and LES

- In WRF the distinction is that
  - PBL is one-dimensional mixing/turbulence in a column, independent of horizontal diffusion, suitable for dx >> dz
  - LES uses three-dimensional sub-grid turbulence scheme, unifying horizontal and vertical mixing strengths, suitable for dx ≈ dz
  - So, when PBL option is set to zero, vertical diffusion is called, and when it is non-zero vertical diffusion is not called
  - km_opt=2,3 are suitable choices for LES 3d diffusion

Climate runs

- WRF physics is suitable for climate runs
  - Drive with reanalyses or global climate model outputs for boundary conditions
- Spectral nudging may be suitable for large domains with reanalysis boundaries, but not necessarily beneficial with GCM boundaries
  - Main purpose is to keep weather systems in correct phase with analysis when doing timewise verification
- Extra diagnostic packages are provided for max/min daily temp, etc.
- Select physics appropriately



Physics

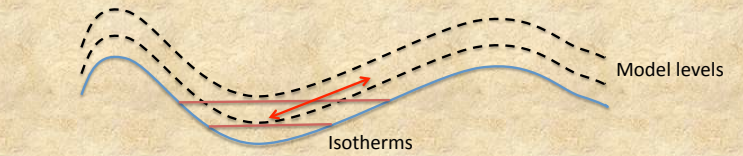
- Regional climate physics
  - Use land model with soil moisture and evolving snow
  - Use sst_update for evolving vegetation fraction and seasonal cycle too (albedo, roughness length)
  - Lake model is available
    - If not using lake model make sure SST input has reasonable temperatures for unresolved lakes
  - Longer simulations may need
    - Deep soil temperature update option
    - Greenhouse gas update option

Complex Terrain

- Steep terrain (> 45 degrees) may cause numerical stability problems – some things to try
- For immediate blow-ups try increasing epssm from default 0.1 to 0.5 or even 1.0
  - This is a sound-wave damper that can stabilize slope treatment by dynamics (little other effect)
- For significant slopes, diff_opt=1 is less realistic than diff_opt=2, but diff_opt=2 was often unstable
  - Since V3.6 WRF now has a diff_opt=2, km_opt=4 option with improved numerical stability

Horizontal Diffusion

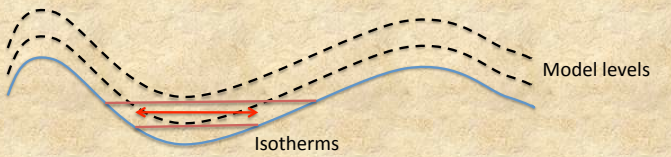
- diff_opt=1



Mixing along levels: Not correct

Horizontal Diffusion

- diff_opt=2



Mixing horizontally: correct

Complex Terrain

- LES in complex terrain remains challenging
- Can now nest down to LES (e.g., 1 km PBL, 333 m LES) with V3.6 since diff_opt, km_opt are now domain dependent
  - Inflow boundary may need to develop rolls
- TKE option (km_opt=2) appears more stable than 3d Smagorinsky (km_opt=3)

Nesting, Resolution and Domain Size

- Nesting is probably needed if your target resolution is much less than your analysis resolution
- Use outer domain(s) to keep low-resolution analysis well upstream of domain of interest
- Usually makes no sense to use less than 100x100 points in a domain on computers these days
- Outer domain grid size could be about 1/3-1/10 analysis (or boundary-data) resolution.
- Makes no sense to have a 1 degree analysis drive a 1 km domain. Use nests to gradually change resolution to domain of interest.
- Keep interior nest boundaries away from each other
- Recommend 3:1 nest ratio
- 5:1 also appears acceptable but be cautious of keeping boundary far from area of interest to allow hi-res adjustment
- Use two nest levels rather than large dx jump with a single nest

Nesting, Resolution and Domain Size

- Try to keep all physics options constant across nest boundaries
  - Cumulus schemes on/off differences can lead to spurious rainfall gradient at nest boundary (e.g., convective rain outside, clear inside)
    - Solved by using 1-way nesting or no feedback or same or no cu_physics on both domains
  - Another common exception is PBL/LES where you can change to LES at hi-res but may see gradients
    - Should use large enough nest area to keep boundary gradients away from region of interest

Model Levels and High Tops

- Not setting eta_levels gives default stretching near ground and uniform Δz higher up
  - Be aware that matching of level thicknesses may be discontinuous, so you may want to use this as a starting point and edit your own levels in the namelist.
  - If you choose too few levels for model top pressure, real.exe will stop because its default dz is not allowed to exceed 1 km (a good rule to follow to prevent noise)
- Choosing base state appropriate to domain surface temperatures (base_temp=270,280,290) may help reduce pressure-gradient force error (keeps p' smaller)
- Hypsometric option 2 is now the default – improves vertical interpolations and integrations with geopotential linear in log p assumption

Model Levels and High Tops

- For high tops < 50 hPa use the (default) stratosphere option for the base state (e.g. iso_temp=200 K)
  - This prevents base state from becoming unrealistically cold at high levels
  - In V3.6.1 we will allow a stratospheric positive lapse rate
- For tops near 1 hPa (45-50 km), may need 60 or more levels
- Some studies (Evan) show 500 m vertical resolution is needed if studying gravity waves in stratosphere
- RRTM and RRTMG radiation include code to prevent cold bias at model top (Cavallo) by estimating downward radiation above model top with extra layers
- Ozone climatology becomes important for tops above about 30 hPa that include some or all of the ozone layer
  - CAM monthly ozone is best we have and is now available for RRTMG

Nudging Options: Use or not

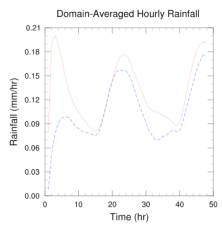
- Four-Dimensional Data Assimilation (Nudging) has specific purposes
  - Adding data during a model run (dynamic analysis)
  - Helping with dynamic initialization (nudged pre-forecast)
  - Keeping an outer domain on track (BCs)
- Nudging introduces fake terms so not recommended for case studies of dynamics and physics effects in events
- Spectral Nudging only affects larger scales (>500-1000 km typically) and may be useful in very large domains if timing of weather systems needs to be accurate in areas far from boundaries (e.g. reanalysis)
  - Can be seen as an interior correction for lateral-boundary distortion of long waves especially by linear interpolation in time

Initialization and Spin-Up Issues

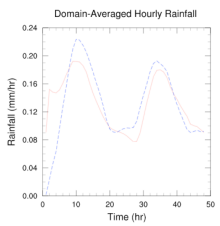
- Model problems often caused by poor initial condition
  - Poor soil temperature or moisture
  - Inappropriate water temperatures or missing masking at coastlines when creating SST in pre-processors
  - Check inputs carefully including soil temperatures, sea-surface temperature
- In first few hours, expect noise in pressure fields
  - Mostly sound waves adjusting winds to terrain
  - This disappears in about the time-scale for sound waves to leave the domain area and has no harmful lasting effects
  - For large domains this is longer (~1 hour per 1000 km)
  - If interested in the first hour or two (e.g. short-period cycling) consider Digital Filter Initialization that effectively filters high-frequencies out from the beginning

Initialization and Spin-Up Issues

- Convection Spin-Up
  - Model will take time to develop deep convection (e.g. 00Z initialization in central US)
  - This delay may be followed by a high bias when convection finally spins up
  - Example of NCAR's 3km convective runs from 2009



00Z



12Z

RUC initialized (red), GFS (blue)

Initialization and Spin-Up Issues

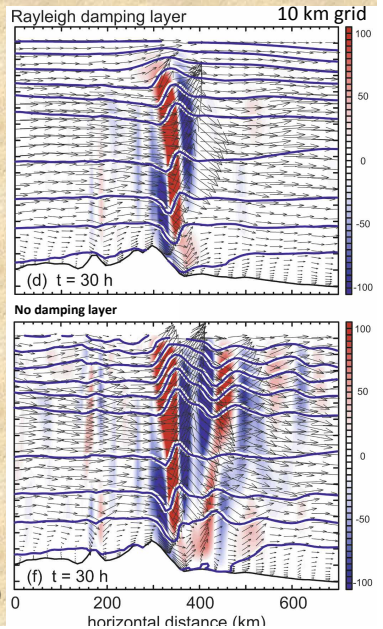
- Land Model
  - Soil moisture and temperature analysis come from generally much coarser offline analyses
  - Soil-data resolution and terrain don't match WRF
    - We handle elevation adjustment for soil temperature using SOILHGT data from source model
    - Cannot handle landuse/soil differences in hi-res domain which means adjustments may occur in soil moisture
    - This adjustment is slow and only way to prevent it is an offline land analysis on the same grid (HRLDAS for Noah)

Damping Options

- Convective instabilities (CFL)
  - w_damping is an artificial negative buoyancy added to updrafts if they approach the CFL stability limit
    - Only recommended for those doing long runs or mass-production/operational runs where they don't want to individually handle blow-ups with re-runs using a short time-step
    - Generally has no effect other than inside strong updrafts
  - Alternative is adaptive time-step option that automatically adjusts time step based on CFL criteria

Damping Options

- Model-top reflection of mountain waves is best solved with damp_opt=3 (Rayleigh damping of w) for real-data cases
  - This very effective at producing proper wave tilts consistent with no reflection

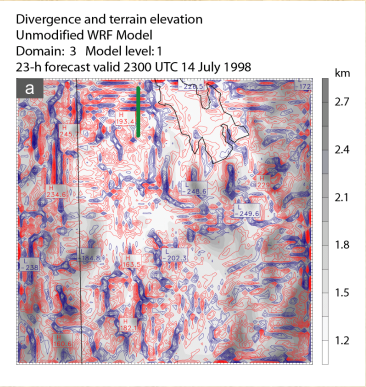


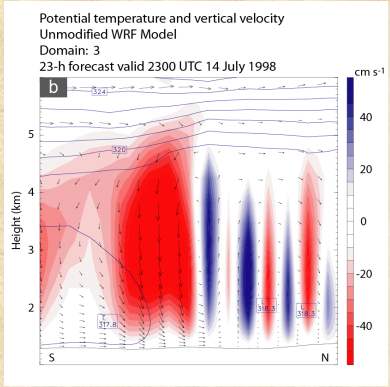
Klemp et al., (2008 MWR)

Damping Options

- diff_6th_opt
  - Selective filter to remove poorly resolved structures (off by default)
  - Most common example is 2Δx waves in boundary layer with weak wind and grid sizes in the 1-4 km range
    - Note that in weak winds odd-order advection damping is less able to smooth the result, so problem appears less with strong enough wind
  - diff_6th_opt=2 (positive definite option) should be used
    - Acts on all advected fields including moisture and option 1 creates negative water that, when zeroed out, becomes a significant non-conserving source

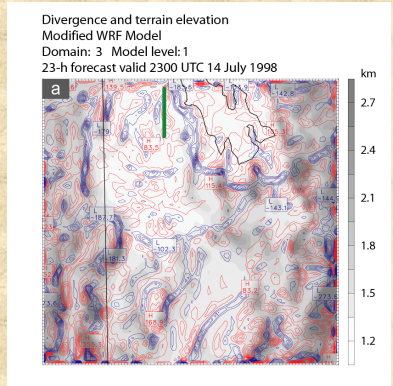
Example of case study: noisy boundary layer

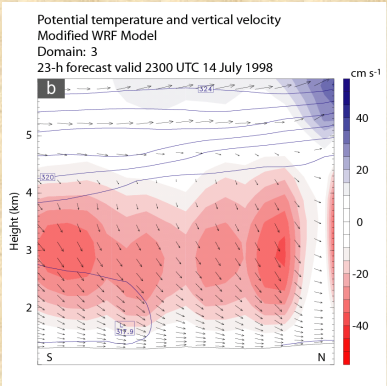




- Reduced diffusion in weak wind allows grid-scale noise to grow in daytime boundary layers

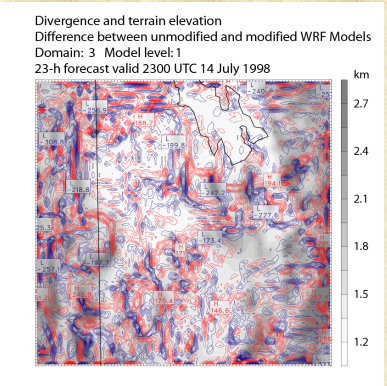
Example of case study: noisy boundary layer

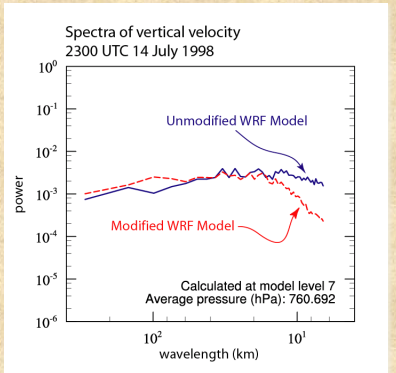




- Adding 6th-order, monotonic, numerical diffusion removes most of the grid-scale noise

Example of case study: noisy boundary layer





- Added diffusion acts mainly on wavelengths less than 6 times grid interval

Advection Options

- 5th order horizontal, 3rd order vertical by default
  - cleaner than even-ordered schemes
  - If using even-ordered maybe diff_6th_opt is helpful
- Positive definite is the default (required for water conservation)
- Monotonic is available (reduces overshoot in maxima), perhaps good for chemistry
- WENO is designed to reduce oscillations at cloud edges

Further Best Practices Reading

- Chris Davis' best practices talk: http://www2.mmm.ucar.edu/wrf/users/workshops/WS2012/ppts/discussion1.pdf
- Wei's tutorial best practices talk: http://www2.mmm.ucar.edu/wrf/users/tutorial/201401/best-practices_wang.pdf

***Reference**:*

Warner, T., 2011. Quality assurance in atmospheric modeling. *Bull. Amer. Met. Soc. Dec. issue, p1601 – 1611.*

Introduction to WRF-Chem

Georg Grell

Introduction to WRF-Chem

Georg Grell

Steven E. Peckham, Stuart A. McKeen, Jan Kazil, R. Ahmadov + others from **NOAA/ESRL**

Jerome Fast, William Gustafson jr., P.L. Ma, B. Singh+ many others from **PNNL**

+ Alma Hodzic, Christine Wiedinmyer, Gabi Pfister, Mary Barth and many others from **NCAR**
other **University contributions**
+ Saulo Freitas (**CPTEC, BRAZIL**)
+new stuff from **NCSU** (Yang Zhang)

+ many more national and international collaborators

WRF-Chem web site - http://wrf-model.org/WG11

WRF-Chem

Community effort

Largest contributing groups: ESRL, PNNL, NCAR

Other significant contributions from: National and international Universities, CPTEC Brazil, NASA, AFWA, NCSU

Structure of Talk

1. Brief description of only the *general features* of WRF-Chem
2. Some applications of what the model may be used for are mixed in

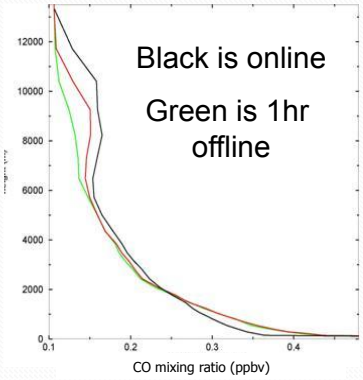
There are more than 50 chemistry options for the main gas phase chemistry and aerosol modules!

WRF-Chem

- Chemistry is online, completely embedded within WRF CI
- Consistent: all transport done by meteorological model
  - Same vertical and horizontal coordinates (no horizontal and vertical interpolation)
  - Same physics parameterization for subgrid scale transport
  - No interpolation in time
- Easy handling (Data management)
- Ideally suited to study feedbacks between chemistry and meteorology
- Ideally suited for air quality forecasting on regional to cloud resolving scales

Why Online?

- Offline modeling introduces errors for air quality applications
  - Error for offline modeling is increasing with increasing horizontal resolution
  - Power spectrum analysis can show the amount of information that is lost in offline runs
- 2-way feedback in-between chemistry and meteorology
  - Process studies relevant for global climate change
  - Ultimately it should lead to improved data assimilation (meteorology) and improved weather forecasts



Grell and Baklanov, 2011, AE

What is needed for this type of modeling system?

1. Advection and diffusion (all done by WRF)
2. Sub-grid scale transport (WRF parameterizations, PBL, convection)
3. Some processes that are specific for chemical constituents, but need meteorology: emissions (biogenic, fire, sea salt, dust, volcanic, anthropogenic), dry deposition, wet scavenging
4. Treatment of chemical reactions, aqueous phase chemistry, gas phase species and aerosols
5. "Chemical" radiation routines (photolysis routines) that provide photolysis rates necessary for (4)
6. Capability of feedback from chemistry to meteorology (meteorological radiation and microphysics parameterizations, possibly also convective parameterizations)

What is needed for this type of modeling system?

1. Advection and diffusion (all done by WRF)
2. Sub-grid scale transport (WRF parameterizations, PBL, convection)
3. Some processes that are specific for chemical constituents, but need meteorology: emissions (biogenic, fire, sea salt, dust, volcanic, anthropogenic), dry deposition, wet scavenging
4. **Treatment of chemical reactions, aqueous phase chemistry, gas phase species and aerosols**
5. "Chemical" radiation routines (photolysis routines) that provide photolysis rates necessary for (4)
6. Capability of feedback from chemistry to meteorology (meteorological radiation and microphysics parameterizations, possibly also convective parameterizations)

Gas Phase Chemistry Packages

Very complex part of the modeling system: Many additional species that are fully prognostic variables and require transport

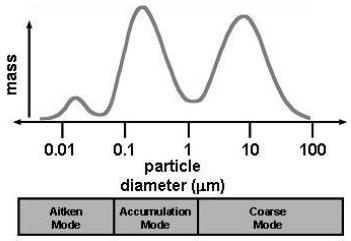
- Hard coded: chemical mechanism from RADM2
- Hard coded: Carbon Bond (CBM-Z) based chemical mechanism
- ***K***inetic ***P***re***P***rocessor (KPP) – Many different equations files exist (also for RADM2 and CBM-Z). KPP will generate the modules from equation files. These generated modules will then be used by WRF-Chem
- IN V3.5.1/ V3.6: CRIMech gas phas scheme (U. of Manchester, 240 species, 652 reactions)
- V3.7: CB05

Photolysis Packages – all coupled to aerosols and hydrometeors

- Madronich Photolysis
- Madronich F-TUV
- Fast-j photolysis scheme

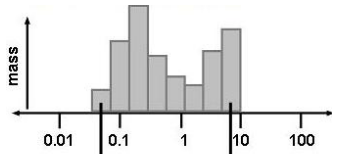
Available aerosol modules

(1) Modal



composition
sulfate
nitrate
ammonium
chloride
carbonate
sodium
calcium
other inorganics
organic carbon
elemental carbon

(2) Sectional



(3) Bulk: Sections for dust and sea salt, otherwise total mass only

Aerosols may have a significant impact on weather forecasts through interaction with radiation (sometimes also called “direct effect”) and microphysics (sometimes also called “indirect effect”)

Aerosols may also impact meteorological data assimilation

For NWP a bulk scheme is very attractive: GOCART (Currently used in real-time high resolution global (dx=30km) and regional modeling (up to dx=3km) at ESRL

- Much simpler than the sectional and modal schemes
  - Calculates only with the total mass of the aerosol components
  - Provides no information on
    - Particle size
    - Particle concentration
  - E.g., when particles grow, the aerosol mass increases but we don't know how their size/number changes
- Numerically very efficient
- Coupled with radiation (Mie scattering and extinction calculations)
- Will be coupled to microphysics in future versions

For research on aerosol direct and indirect effects modal and sectional approaches are more attractive

Less assumptions are made when coupled to atmospheric radiation and/or microphysics

Interaction processes are very complex, they will not work for every radiation and microphysics scheme in WRF ! (takes time to implement)

Examples of available Aerosol Modules

- Bulk: GOCART
- Modal: MADE/SORGAM (3 modes)
- MAM (Modal Aerosol Model from NCAR Climate model
- MOSAIC (Sectional)

Selection of radiation parameterizations for aerosol "direct effect"

Since V3.5 all aerosol modules were hooked up to Goddard short wave radiation, and RRTMG short and long wave scheme, CAM radiation.

Selection of microphysics parameterizations for aerosol "indirect effect"

Since V3.6

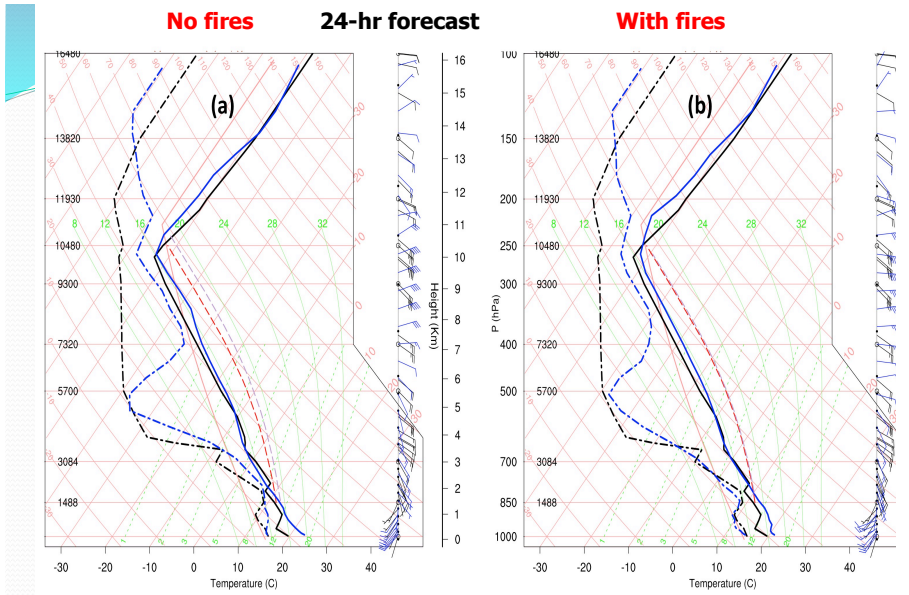
Modal and sectional schemes only can be used in combination with a version of the Lin et al. Microphysics scheme as well as the Morrison scheme

Special physics choice available when using NCAR community climtae model physics

"indirect effect" is a result of the interaction aerosols/microphysics

How is the meteorological forecast affected by aerosol?

- Large importance for climate simulations is recognized (when integrating models over 100's of years, small differences in the earth's energy budget are extremely important)
- Weather forecasting for only a few days?
  - Much research needed, but chemistry may positively influence forecasts when strong signals exist
  - Influence on meteorological data assimilation



Observed (black) and predicted (blue) sounding for Fairbanks, Alaska, on July 4, 0000UTC.

Biogenic emissions

- May be calculated "online" based on USGS landuse
  - Easy to use
- May be input
- BEISv3.13 (offline reference fields, online modified)
  - Good choice, but difficult to use
- **Use of MEGAN**
  - **Best choice!!**

***M*odel of *E*missions of *G*ases and *A*erosols from *N*ature (MEGAN)**

Global, high resolution biogenic emissions

Out of available biogenic emissions modules only BEIS and MEGAN are actively being worked on (developed)

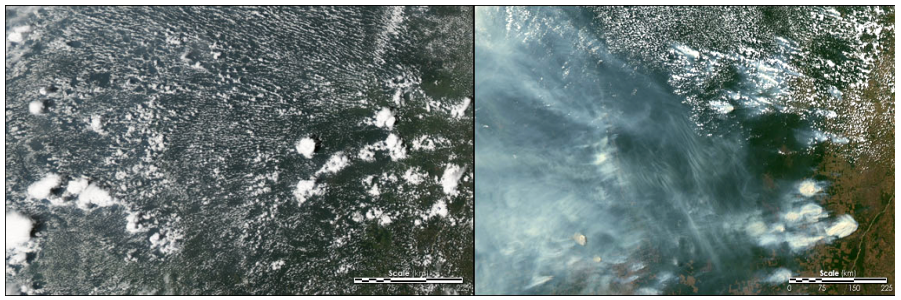
Preprocessor for MEGAN exists and can be downloaded from NCAR

Fire Plumerise

1-D Cloud model used in WRF-Chem to determine injection height, wind shear effects are included

Satellite information (other aerial and ground observations may also be used) to determine fire location and fire properties

Emissions preprocessing may be done by (1) CPTEC preprocessor, or (2) NCAR's FINN preprocessor



Impact of Volcanoes

- Ash-fall near eruption
- Transport of fine ash in high concentrations for long distances
- Impact on weather, climate, and air quality

The plume of the 30 Sept/1 Oct 1994 eruption of Kliuchevskoi Volcano, Kamchatka taken from the space shuttle STS-68 mission (Russia)

10 size bins for prediction of ash-fall and transport of volcanic ash

Particle Size Bin	Phi	Percentage of mass
1 – 2mm	-1 – 0	2
0.5 – 1 mm	0 – 1	4
0.25 – 0.5 mm	1 – 2	11
125 – 250 µm	2 – 3	9
62.5 – 125 µm	3 – 4	9
31.25 – 62.5 µm	4 – 5	13
15.625 – 31.25 µm	5 – 6	16
7.8125 – 15.625 µm	6 – 7	16
3.9065 – 7.8125 µm	7 – 8	10
< 3.9 µm	> 8	10

- Options for transport only (4 bins or 10 bins +so2) – aerosol direct effect may be included
- Coupled with chemistry/aerosol modules (only using up to three bins – depending on size)

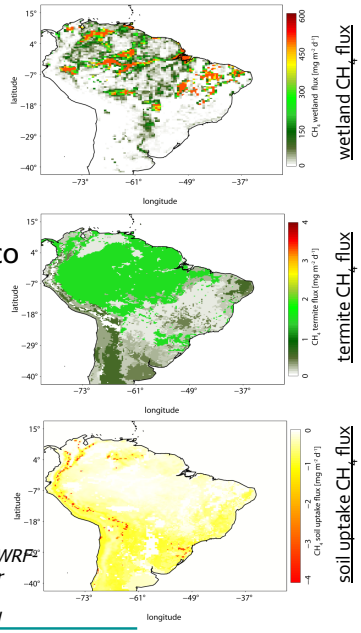
4 size bins for prediction if transport only is of interest

Particle Size Bin	Phi	Percentage of mass
15.625 – 31.25 µm	5 – 6	16
7.8125 – 15.625 µm	6 – 7	16
3.9065 – 7.8125 µm	7 – 8	10
< 3.9 µm	> 8	10

3 size bins for coupling with other aerosol modules

WRF-Chem Greenhouse Gas Packages *(chem_opt =17)-new in WRF-ChemV3.4*

- **Online calculation of biospheric CH_4 fluxes**
  wetland – Kaplan (2002)
  termite – Sanderson (1996)
  soil uptake – Ridgwell et al. (1999)
- **Passive tracer simulations for CO_2, CH_4, and CO** (including all options of CO_2 tracer package, chem_opt=16)
- **Tuning of wetland fluxes** through namelist options wpeat and wflood possible
- **Separate biomass burning option** for CO_2, CH_4, and CO including plumerise calculation (biomass_burn_opt = 5)
- **Detailed description**
  Beck et al., (2011): The WRF Greenhouse Gas Model (WRF-GHG) Technical Report No. 25, Max Planck Institute for Biogeochemistry, Jena, Germany, available online at http://www.bgc-jena.mpg.de/bgc-systems/index.shtml



Direct connection to NCAR's climate modeling system: Implementation of the Community Atmosphere Model version 5 (CAM5) Physics/Chemistry

Pacific Northwest NATIONAL LABORATORY
Proudly Operated by Battelle Since 1965

- **Includes different physics options for deep and shallow convection, microphysics, boundary layer**
- **Aerosols:** *Liu et al*. (GMD, 2012), Modal Aerosol Model (MAM)
- **Gas-Phase Chemistry:** MOZART used by "CAM-Chem" already implemented in WRF-Chem by NCAR
- **PNNL** has coupled MAM with CBM-Z photochemistry in WRF-Chem

overview paper of CAM5 and coupling of these parameterizations (Rasch et al., 2013)

This Climate Model package also includes aerosol direct and indirect effect, but is limited on combinations with other packages

Several dust and sea-salt models, used for bulk, modal, and sectional approaches

Lightning parameterization for NOx emissions

Aerosol interaction with convective parameterization is currently evaluated in Grell-Freitas scheme (maybe released in V3.7.1)

University of Manchester: completed developments (Lowe et al.) was added in WRF-Chem 3.6

- Common Represetative Intermediate Mechanism (CRIMech) (CRIv2-R5; 240 species, 652 rxns) (Watson et al., 2008)
- N_2O_5 heterogeneous chemistry in WRF-Chem sectional aerosol (Bertram & Thornton, 2009)
- Sea-spray emission scheme with organics (Fuentes et al., 2011)
- Organic Partial Derivative Fitted Taylor Expansion (PD-FiTE) added to MOSAIC sectional aerosol (Topping et al., 2009; 2012)

Douglas Lowe, Steven Utembe*, Scott Archer-Nicholls, David Topping, Mark Barley, Gordon McFiggans

Chemical data assimilation

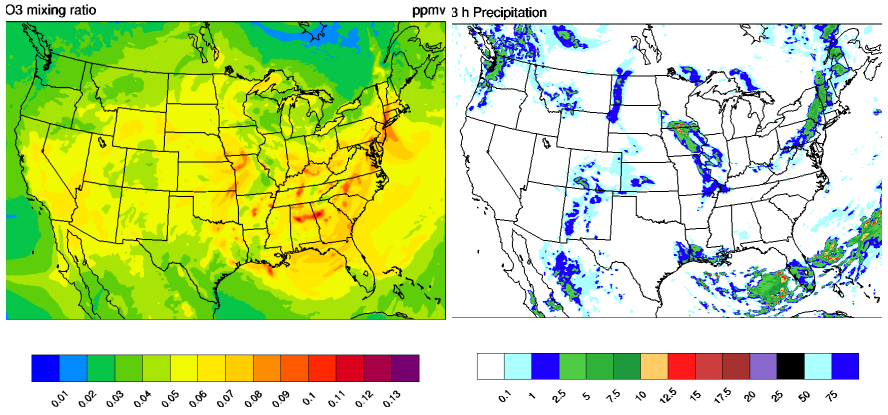
- NCEP's Grid Point Statistical Interpolation (GSI, 3DVAR) assimilation system can be used with surface chemical data as well as with AOD: Significant improvements in forecasts.
- EnKF assimilation system has been used for WRF-Chem
- Work is on-going with hybrid EnKF/GSI system (ESRL and NCAR)
- Work is also ongoing with WRF-Chem adjoint development (project lead by Greg Carmichael)

These approaches are not released to community yet, but one approach with WRF DART system maybe openly available

Otherwise, if you need chemical data assimilation to help develop or use, email wrfchemhelp for contact information

Real-time AQ forecasting with WRF-Chem

http://ruc.noaa.gov/wrf/WG11_RT/

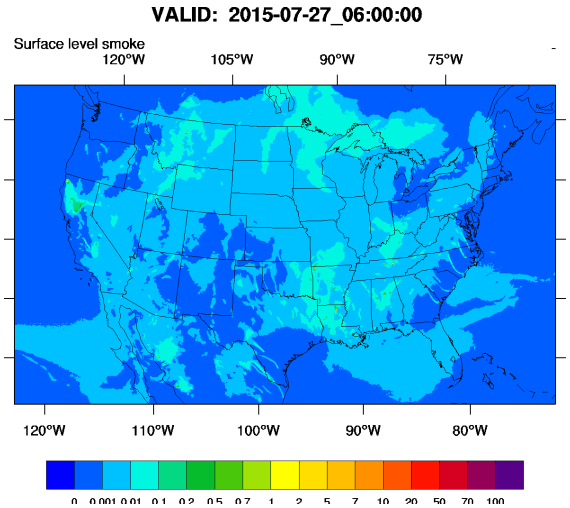


Ozone at 21Z, July 27

3-hr precip at 21Z

WRF-Chem using MADE/VBS/RACM on Rapid Refresh Domain, DX=13km

HRRR-Smoke: 3km horizontal resolution



Resources

- WRF project home page
  - http://www.wrf-model.org
- WRF users page (linked from above)
  - http://www.mmm.ucar.edu/wrf/users
- WRF users help desk
  - wrfhelp@ucar.edu
- WRF-Chem users help desk
  - wrfchemhelp.gsd@noaa.gov
- Publications (please send us yours)
  - http://ruc.noaa.gov/wrf/WG11/References/WRF-Chem.references.htm

Inter-journal special issue on WRF-Chem now also opened: ACP and GMD

Thank you for your interest, Questions?

Aerosol Modeling with WRF-Chem

Georg Grell

Aerosol modeling with WRF/Chem

Jan Kazil

University of Colorado / NOAA Earth System Research Laboratory

WRF/Chem Tutorial, 3 August 2015
(WRF/Chem 3.7)

Part I - Introduction

- **Overview of ...**
  - Aerosol
  - Aerosol processes and life cycle
  - Model treatment of aerosol
  - WRF/Chem aerosol schemes

Part II – The details

- **Representing the aerosol size distribution**
- **Walk through the WRF/Chem aerosol schemes**
  - How they work and what they do
  - Coupling to other processes
    - Gas phase chemistry
    - Aqueous chemistry
    - ...
- **Hint on how to tell WRF/Chem what to do**
- **Resources**

Part I – Introduction

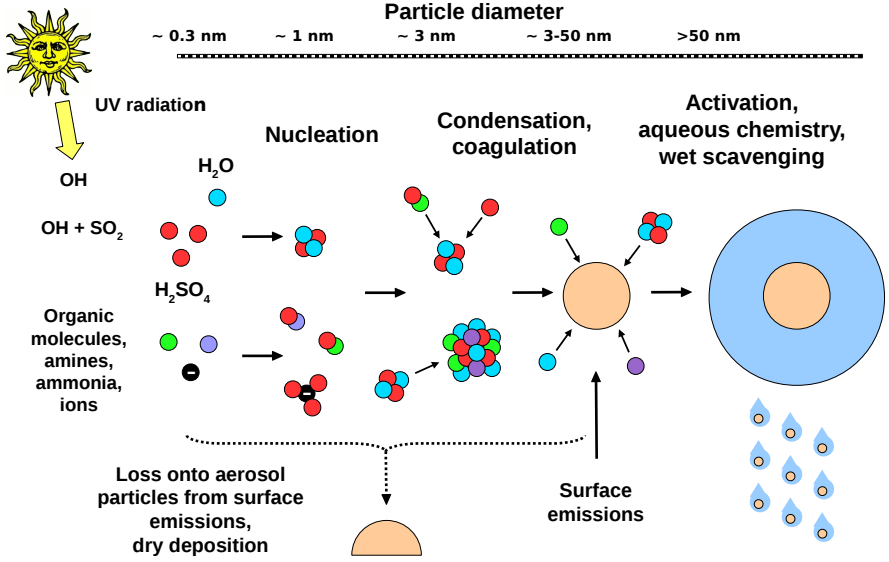
Aerosol



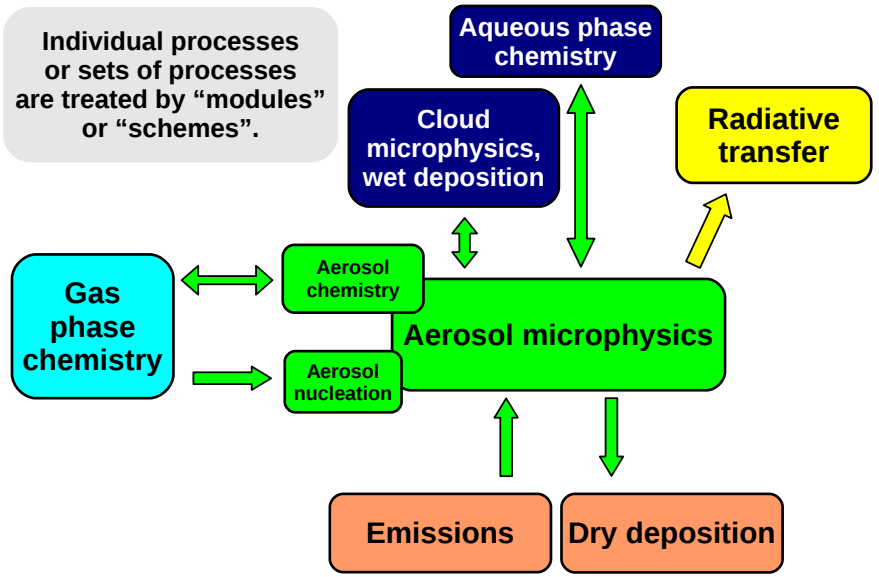




Aerosol life cycle and processes



Model treatment of aerosol



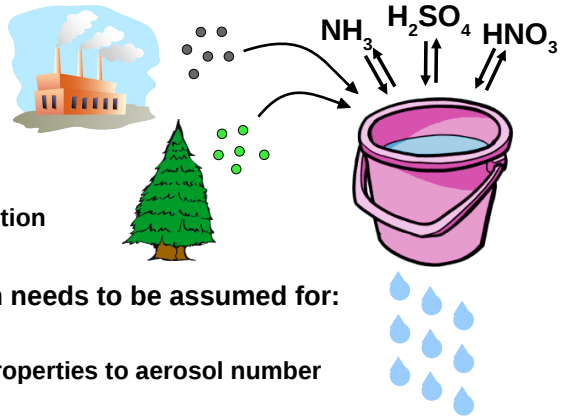
WRF/Chem aerosol schemes

- **An efficient aerosol scheme from the GOCART model**
  - No size information for sulfate, BC, OC
  - Size information for dust and sea salt
  - No secondary organic aerosol (SOA)
- **Modal Aerosol Dynamics Model for Europe – MADE**
  - 3 log-normal modes
  - Inorganic, organic aerosol, SOA
- **Model for Simulating Aerosol Interactions and Chemistry (MOSAIC)**
  - Sectional aerosol scheme, 4 or 8 bins
  - Inorganic, organic aerosol, SOA
- **MAM – Modal Aerosol Model from CAM5**
  - 3 or 7 log-normal modes
  - Inorganic, organic aerosol, SOA, sea salt, BC, mineral dust
- **Simple sectional (bin) scheme for volcanic ash aerosol**

Part II – The details

Bulk aerosol schemes

- Only total mass of aerosol compounds is known



- No information on
  - Particle number
  - Aerosol size distribution

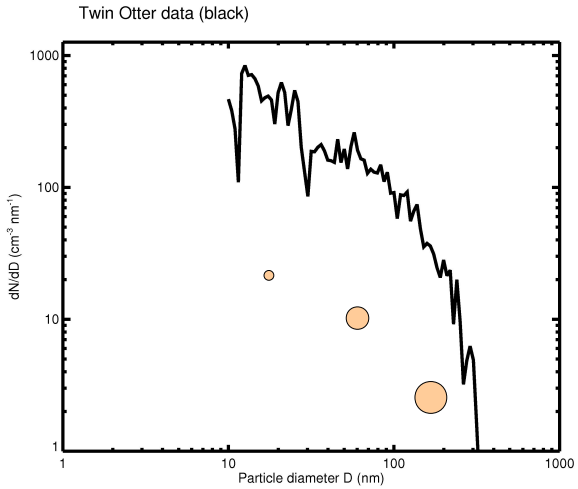
Aerosol size distribution needs to be assumed for:

- radiative transfer
- response of cloud properties to aerosol number

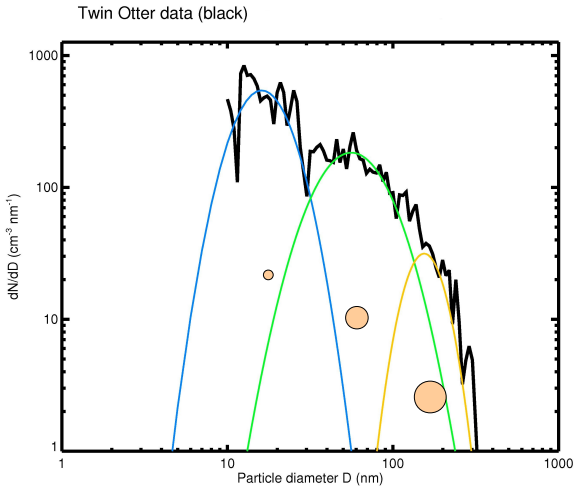
- Numerically efficient
- Useful when focus is on complex gas phase chemistry

→ GOCART (+ size resolved dust and sea salt)

Modal aerosol schemes



Modal aerosol schemes



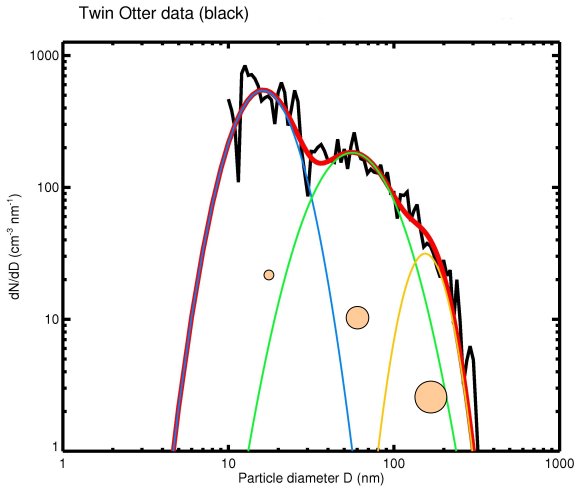
$$\frac{dN}{dD} = \frac{N}{\sqrt{2\pi}\ln(\sigma)D} e^{-\frac{1}{2}\left[\frac{\ln(D/\mu)}{\ln(\sigma)}\right]^2}$$

$\frac{dN}{dD}$ → $N = 8195\ cm^{-3}$
$\mu = 18.22\ nm$
$\sigma = 1.42$

$\frac{dN}{dD}$ → $N = 12732\ cm^{-3}$
$\mu = 68.44\ nm$
$\sigma = 1.57$

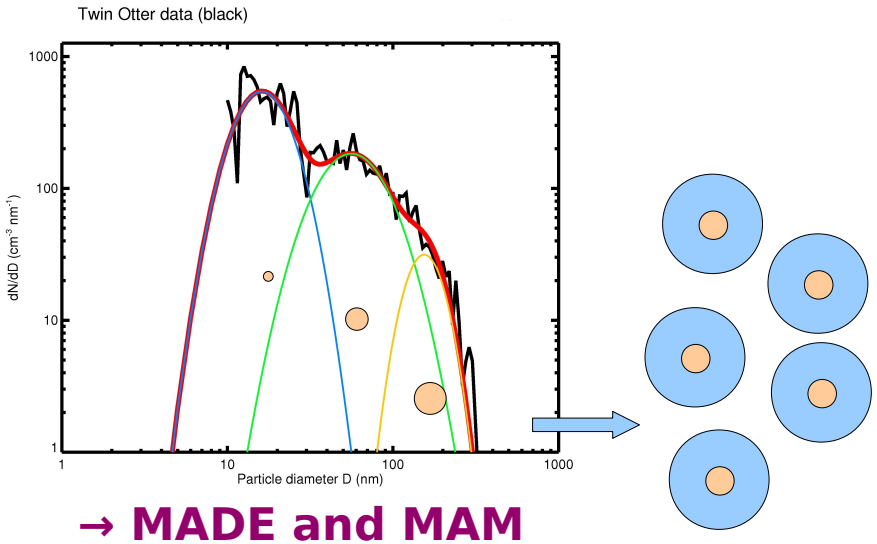
$\frac{dN}{dD}$ → $N = 3140\ cm^{-3}$
$\mu = 164.41\ nm$
$\sigma = 1.28$

Modal aerosol schemes



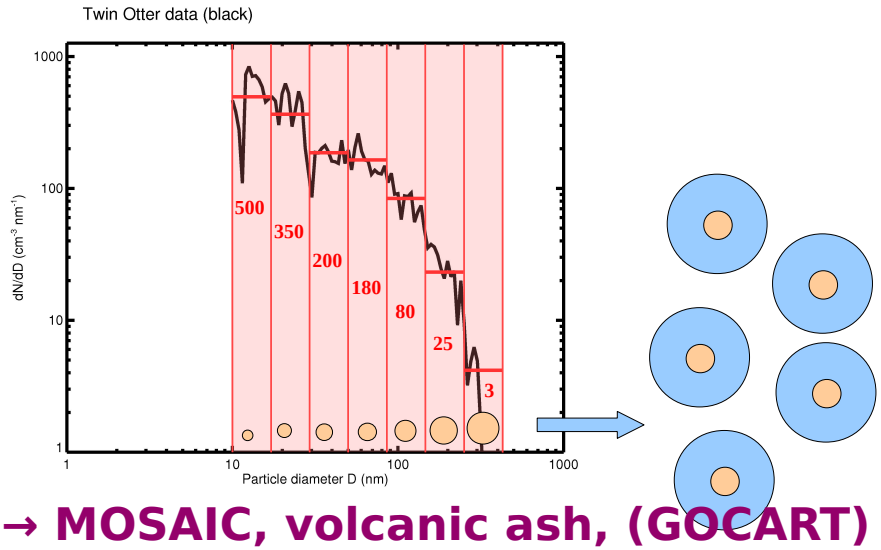
$$\frac{dN}{dD} = \frac{dN}{dD} + \frac{dN}{dD} + \frac{dN}{dD}$$

Modal aerosol schemes



→ MADE and MAM

Sectional aerosol schemes



→ MOSAIC, volcanic ash, (GOCART)

GOCART aerosol module

- Georgia Tech/Goddard Global Ozone Chemistry Aerosol Radiation and Transport model (Chin et al., JGR, 2000)
  - Bulk aerosol:
    - Hydrophobic black carbon (fresh soot)
    - Hydrophilic black carbon (aged/coated soot)
    - Hydrophobic organic carbon (fresh burnt biomass)
    - Hydrophilic organic carbon (aged/coated burnt biomass)
      - Fresh → aged conversion time 2.5 days
    - Other GOCART primary PM2.5
    - Other GOCART primary PM10
    - Sulfate (only secondary aerosol species)
  - Sectional scheme for dust and sea salt:
    - Dust: 0.5, 1.4, 2.4, 4.5, 8.0 µm effective radius
    - Sea salt: 0.3, 1.0, 3.2, 7.5 µm effective radius

GOCART aerosol module

GOCART comes with sulfur gas phase chemistry:

- DMS + OH → SO_2 + ...
- DMS + OH → MSA + ...
- DMS + NO_3 → SO_2 + ...
- SO_2 + OH → $SO_4^=$ + ...

Extended gas phase chemistry can be used:

- MOZART (with KPP)
- RACM (with KPP)
- RADM (with and without KPP)

GOCART aerosol module

- Interaction with radiation:
  - Direct effect for some model setups
  - Effect on photochemistry
- Interaction with clouds:
  - Aqueous chemistry
    - $SO_2 + H_2O_2 \rightarrow SO_4^=$
    - $SO_2 + O_3 \rightarrow SO_4^=$
- No secondary organic aerosol (SOA)

MADE aerosol module

Modal **A**erosol **D**ynamics Model for **E**urope
(Ackermann et al., Atm. Env., 1998)

- 3 log-normal aerosol modes: Aitken, accumulation, coarse
- Mode width σ is fixed
- Aerosol number and mass variable
- Interaction with radiation:
  - Direct aerosol effect
  - Effect on photolysis
- Interaction with clouds:
  - Aerosol number determines cloud drop number and size
  - Radiative response → 1st indirect aerosol effect
    - only for resolved clouds (Sc)
  - Aqueous chemistry
  - Wet removal (scavenging)

MADE aerosol module

Aitken and accumulation modes:

- $SO_4^=$, NH_4^+, NO_3^-, H_2O
- NaCl (sea salt)
- Anthropogenic SOA from oxidation of ...
  - Alkanes
  - Alkenes
  - Aromatics
- Biogenic SOA from oxidation of ...
  - Alpha-pinene
  - Limonene
  - Isoprene
- Anthropogenic POA
- Elemental carbon (soot)
- Primary PM2.5

MADE aerosol module

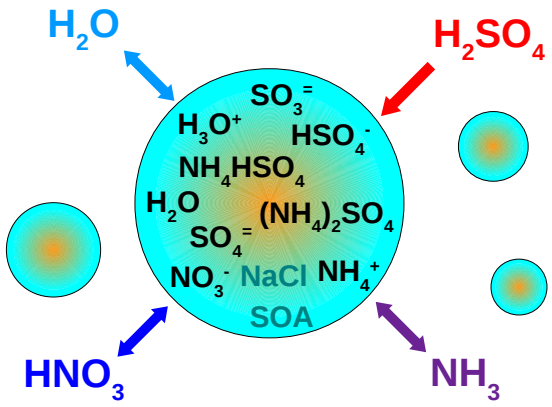
Coarse mode:

- Anthropogenic primary aerosol – e.g. from
  - Coal combustion
  - Cement manufacturing
  - Metallurgy
  - Waste incineration
- Sea salt
- Soil derived particles (mineral dust)

MADE aerosol coupling with chemistry

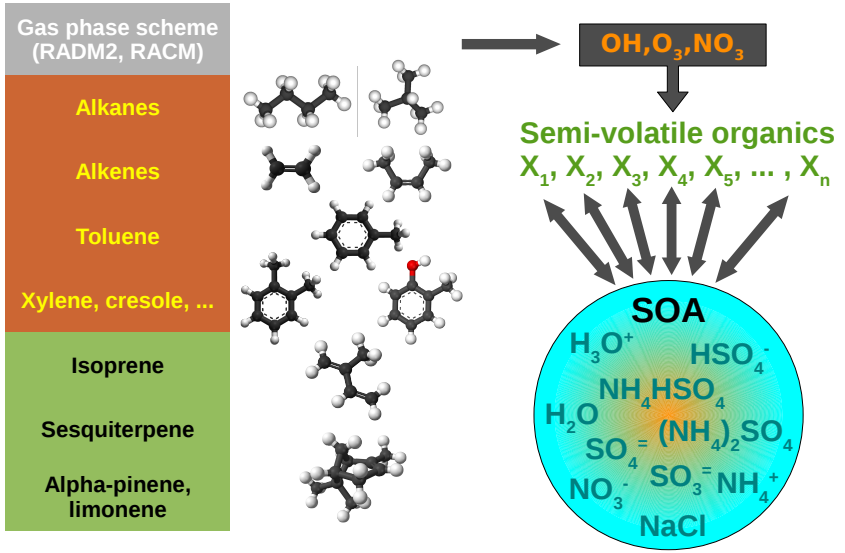
- **Gas phase chemistry:**
  - RADM2 (Regional Acid Deposition Model version 2)
  - RACM (Regional Atmospheric Chemistry Mechanism)
  - RACM NOAA/ESRL version
  - CBMZ (Carbon-Bond Mechanism version Z)
- **Gas phase/particle partitioning (aerosol chemistry):**
  - MARS (Model for an Aerosol Reacting System)
  - SORGAM (Secondary Organic Aerosol Model)
  - VBS (Volatility Basis Set)
- **Aqueous chemistry:**
  - (CMU aqueous chemistry)
  - CMAQ (EPA) aqueous chemistry
  - Only for Aitken and accumulation mode
  - Only for selected gas phase chemistry options

MADE and MARS: Inorganic aerosol chemistry



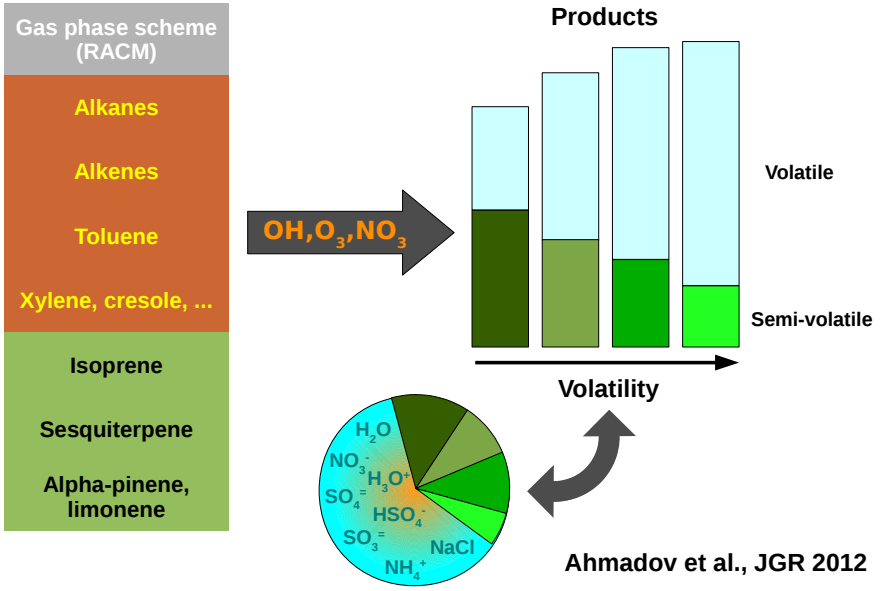
MARS (Model for an Aerosol Reacting System), Saxena et al., Atm. Env., 1986

MADE/SORGAM



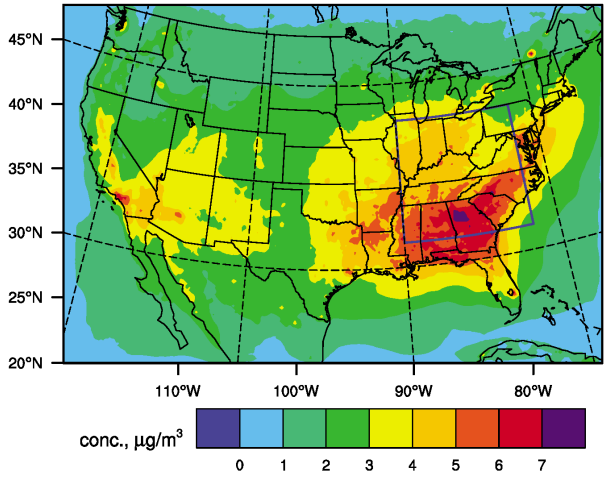
SORGAM (Secondary Organic Aerosol Model), Schell et al., JGR, 2001

MADE/VBS (Volatility Basis Set)



Ahmadov et al., JGR 2012

MADE/VBS (Volatility Basis Set)

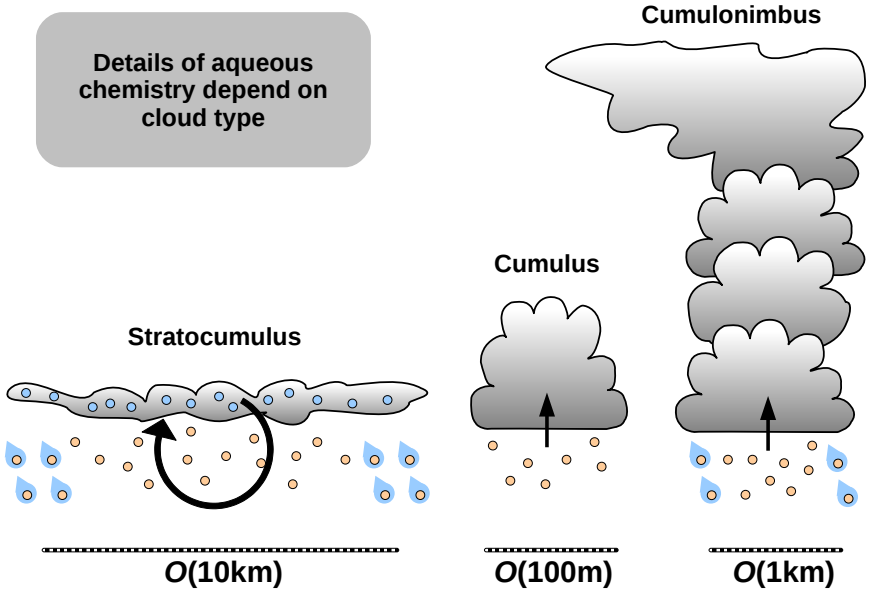


Organic aerosol mass in the surface layer
(August - September 2006)

Ahmadov et al., JGR 2012

How clouds are simulated in WRF(Chem)

Details of aqueous chemistry depend on cloud type



MADE and aqueous chemistry

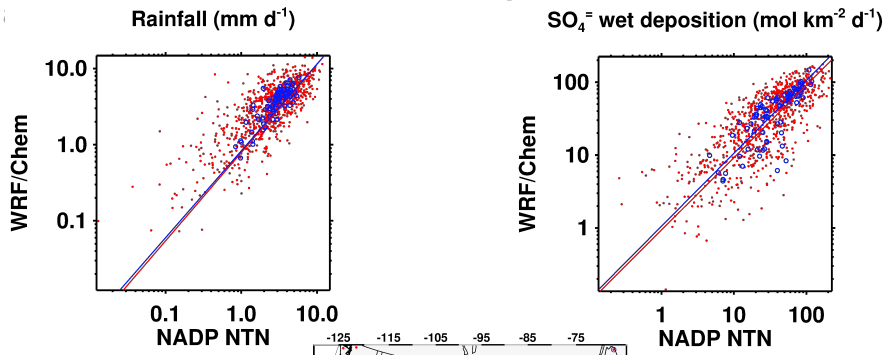
- CMU aqueous chemistry
  - (Fahey & Pandis, Atm. Env., 2001)
  - Slow, only for resolved clouds (Sc)
- CMAQ aqueous chemistry
  - (Walcek & Taylor, JAS, 1986)
  - Relatively fast
  - In both resolved (Sc) and parameterized (Cu) clouds
  - Can generate wet deposition of $SO_4^=$ and NO_3^-
  - In file “registry.chem”, set "h" (history) flag of
    - wd_so4_sc, wd_so4_cu
    - wd_no3_sc, wd_no3_cu
    - Recompile

MADE and aqueous chemistry

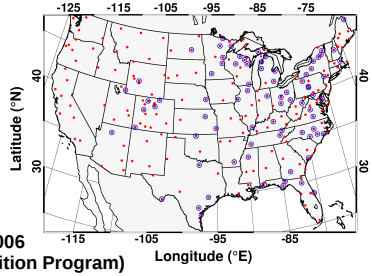
CHEM_OPT	Stratocumulus	Cumulus
1		AQCHEM **
11	CMU *	AQCHEM **
12	CMU *	AQCHEM **
105		AQCHEM **
106		AQCHEM **
107		AQCHEM **
108		AQCHEM **
41	AQCHEM *	AQCHEM **
42	AQCHEM *	AQCHEM **
42	AQCHEM *	AQCHEM **

```
*  cloud_chem_onoff = 1 : Aqueous chemistry in Sc clouds on
** conv_tr_aqchem   = 1 : Aqueous chemistry in Cu clouds on
** wetscav_onoff    = 1 : Wet scavenging on in Sc clouds
** conv_tr_wetscav  = 1 : Wet scavenging in Cu clouds
```

Wet deposition example



Rainfall	
r	model/obs.
0.80	1.24



$SO_4^=$ wet dep.	
r	model/obs.
0.89	1.04

May-September 2006
(National Atmospheric Deposition Program)

MOSAIC aerosol module

Model for Simulating Aerosol Interactions and Chemistry (Zaveri et al., JGR, 2008)

- Modern aerosol scheme in WRF/Chem
- 4 or 8 aerosol size sections (bins) 39 nm – 10 μm
- Interaction with radiation:
  - Direct aerosol effect
  - Effect on photolysis
- Interaction with clouds:
  - Aerosol number determines cloud drop number and size
  - Radiative response → 1st indirect aerosol effect
  - Aqueous chemistry
  - Wet removal (scavenging)

MOSAIC aerosol module

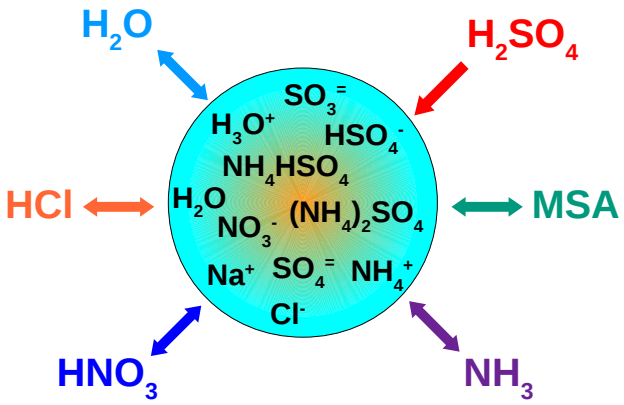
Aerosol composition

- $SO_4^=$, NH_4^+, NO_3^-, H_2O
- NaCl (sea salt)
- CH_3SO_3 (methanesulfonate)
- carbonate (CO_3)
- calcium (Ca)
- black carbon (BC)
- primary organic mass (OC)
- other inorganic mass (minerals, trace metals)

MOSAIC aerosol coupling with chemistry

- Gas phase chemistry:
  - CBMZ (Carbon-Bond Mechanism version Z)
    - "Standard" gas phase chemical scheme for MOSAIC
  - SAPRC99 (extensive VOC chemistry)
    - Works with the VBS SOA scheme
  - MOZART (Model for Ozone and Related chem. Tracers)
    - Works with the VBS SOA scheme
- Gas phase/particle partitioning (aerosol chemistry):
  - MTEM (Multicomponent Taylor Expansion Method)
  - MESA (Multicomponent Equilibrium Solver for Aerosols)
  - VBS (Volatility Basis Set)
- Aqueous chemistry:
  - CMU aqueous chemistry, only for resolved clouds (Sc)

MOSAIC, MTEM, and MESA



MTEM calculates activity coefficients
MESA solves ion-equilibria in the liquid phase
For SOA: VBS (Volatility Basis Set) scheme

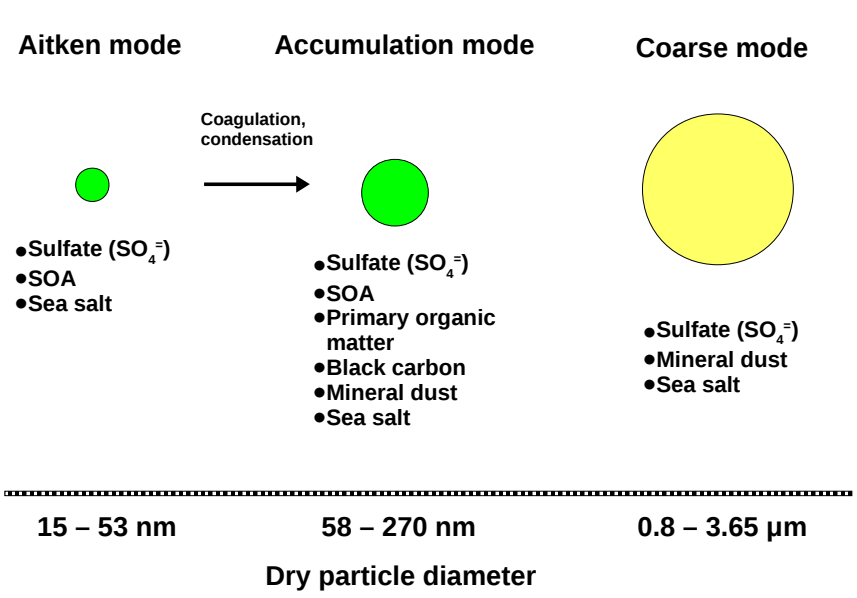
MTEM (Multicomponent Taylor Expansion Method), Zaveri et al., JGR 2005a
MESA (Multicomponent Equilibrium Solver for Aerosols), Zaveri et al., JGR 2005b

MAM aerosol module

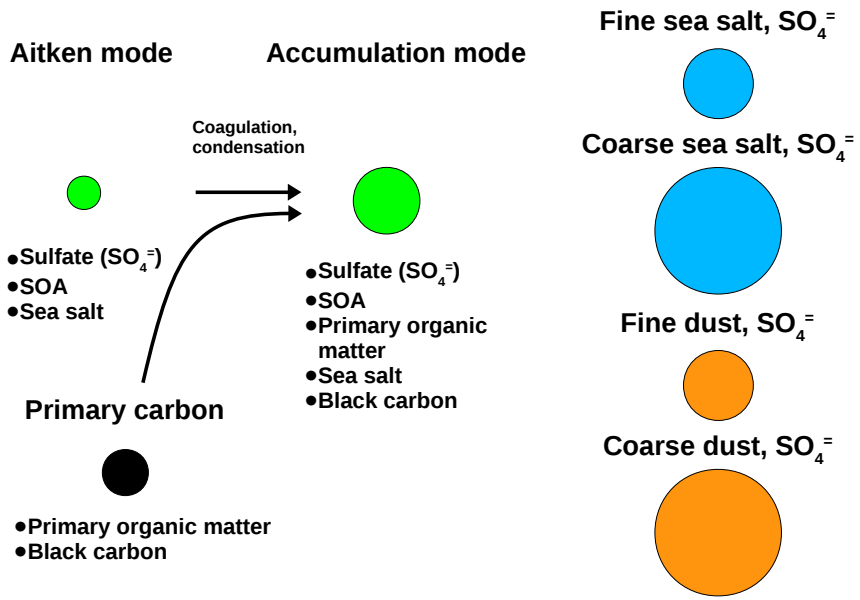
Modal Aerosol Model from CAM5

- **3 or 7 log-normal aerosol modes: MAM3 and MAM7**
- **Mode width σ is fixed**
- **Aerosol number and mass variable**
- **Liu et al., Geosci. Model Dev., 5, 709-739, 2012**

MAM 3



MAM 7



MAM aerosol module

- Currently only one gas phase chemistry scheme
  - CBMZ (Carbon-Bond Mechanism version Z)
- Interaction with radiation:
  - Coupled to RRTMG radiation → Direct aerosol effect

As in CAM5:

- Gas phase/particle partitioning (aerosol chemistry):
  - Condensation of water vapor and of the 4 inorganic trace gase species: NH_3, H_2SO_4, HNO_3, HCl
- Interaction with clouds only resolved clouds (Sc):
  - Coupled to Morrison & Gettelman cloud microphysics
  - Radiative response → 1st indirect aerosol effect
  - Wet removal (scavenging)
  - Aqueous chemistry
- Dry deposition

Volcanic ash

- 10 bins for volcanic ash aerosol
- Transport, settling, dry deposition
- Currently no other aerosol
- Single active volcano
- Database of 1535 volcanoes (latitude, longitude, height)

How to tell WRF/Chem what to do

../WRFV3/test/em_real/real.exe
../WRFV3/test/em_real/namelist.input
../WRFV3/test/em_real/...
../WRFV3/test/em_real/...

```
  ...
  ...
 &chem
 chem_opt        = 43
 photdt          = 0.25
 chemdt          = 0
  ...
aerchem_onoff    = 1
  ...
conv_tr_aqchem = 1
```

MADE/SORGAM, RACM-ESRL, CMAQ aqueous chemistry

Switches all aerosol processes on/off

CMAQ aqueous chemistry on in Cu

Resources

- WRF/Chem User's Guide
  - Model options (namelist parameters)
  - Combinations of physical/chemical schemes
  - ...
- Papers referenced in the WRF/Chem User's Guide
- WRF/Chem source code

- WRF/Chem Help (wrfchemhelp.gsd@noaa.gov)
- Yours truly (jan.kazil@noaa.gov)

Best Practices for Applying WRF-Chem

Georg Grell

Best Practices for Applying WRF-Chem

NOAA/Earth System Research Laboratory
Steven Peckham, Georg Grell, Mariusz Pagowski, Stuart McKeen, and Ravan Ahmadov

NCAR/Atmospheric Chemistry Division
Mary Barth, Christine Wiedinmyer, Gabriele Pfister, Rajesh Kumar, Christoph Knote, Alma Hodzic, and Alex Guenther

Pacific Northwest National Laboratory
Jerome Fast, William Gustafson, Richard Easter, Manish Shrivastava, Rahul Zaveri, and James Barnard

Univ. of Manchester
Douglas Lowe, Gordon McFiggans, and Scott Archer-Nicholls

As well as many other WRF-Chem developers

Overview

- General Considerations
- Highlight WRF-Chem Options
- Advise From Developers
  - NOAA/ESRL
  - NCAR/ACD
  - PNNL
  - Others

General Considerations

Ask yourself

- **Am I following the WRF best practices for my domain?**
- **What chemistry option should I use for my project?**
  - Research or forecasting?
    - What is my computational power?
    - How quickly do I need a solution?
  - What horizontal and vertical resolution may be required to resolve important features?
    - Meteorology and chemistry might be different
  - Do I have appropriate emissions inventory (if necessary)?
    - Speciation for chemical mechanism correct?
    - Adequate spatial and temporal resolution?

General Considerations

- **What chemistry option should I use for my project?**
  - What process do I want to simulate?
    - Dispersion?
      - may include any tracer (dust, smoke, hazardous constituents,…), computing costs from a little more to about 2x plain WRF run
    - Air Quality?
      - usually requires gas-phase chemistry as well as aerosol modules, computing costs more than 5x plain WRF run with little upper limit
    - Fully interactive weather/climate and chemistry/aerosols?
      - Interaction of aerosols with radiation (direct and resulting semi-direct effect) could be anywhere between 2x to more than 100x plain WRF run;
      - indirect effect 10x to more than 100x plain WRF run.
  - Do I have appropriate initial and boundary conditions?
    - Spin-up time for chemistry (enough computational power)?

General Considerations

Special interest for air quality and weather forecasting: aerosols

- Aerosols may have a significant impact on weather forecasts through:
  - interaction with radiation (sometimes also called “direct effect”)
  - microphysics (sometimes also called “indirect effect”)
- Aerosols may also impact meteorological data assimilation

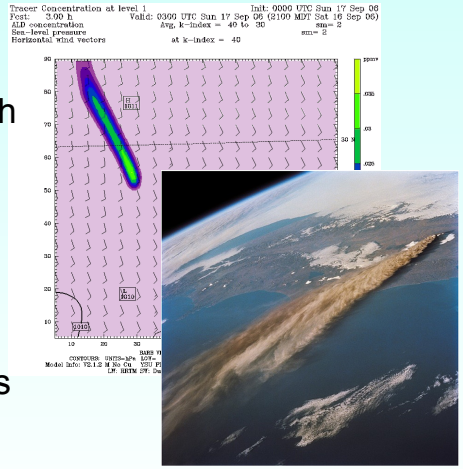
Overview

- General Considerations
- Highlight WRF-Chem Options
- Advise From Developers
  - NOAA/ESRL
  - NCAR/ACD
  - PNNL
  - Others

Available Aerosol Modules

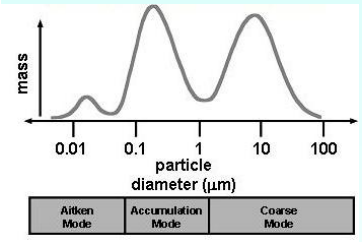
(1) Tracers: Dust, Smoke, Volcanic ash

(2) Bulk: Sections for dust and sea salt, otherwise total mass for a few species



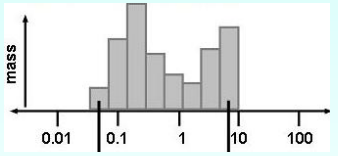
Available Aerosol Modules

(3) Modal



composition
sulfate
nitrate
ammonium
chloride
carbonate
sodium
calcium
other inorganics
organic carbon
elemental carbon

(4) Sectional



Modal and sectional schemes may be too expensive in operational NWP settings – but important in research and for air quality forecasting

Dust only: Consideration

- Dust emissions has a strong dependence on:
  - soil moisture
  - erodable fraction
  - Dust not included from other sources (e.g., roads, farms)
- dust_opt=3, or 4 are currently preferred to dust_opt=1, since they includes more updated physics
- When using the MADE modal aerosol scheme, 3d settling is limited for dust_opt 1, 3, or 4. Same for sectional sea salt option

Dust only: Consideration

- dust_opt=4 has not be as well tested
  - dust_schme = 1 Shao (2001)
  - dust_schme = 2 Shao (2004)
  - dust_schme = 3 Shao (2011)
    - simplification of dust_schme =2

Passive Tracers: Consideration

- Passive tracers can be run separately or along with chemistry
  - Tracer initialization and BCs: module_input_tracer.F
  - When using tracer_opt with WRF-Chem, tracers are advected (chem_adv_opt in namelist.input), vertically mixed (vertmix_onoff) and also mixed by cumulus parameterization (chem_conv_tr). Deposition and emission could be included with some code modification.
- Old tracer option (chem_opt=13 to 15) to be removed in future release
- Passive tracer options for greenhouse gases (chem_opt=16, 17)
  - no chemical reactions considered
  - Includes anthropogenic emissions, and biospheric fluxes of carbon dioxide CO_2 and methane (CH_4)

Options for Gas Phase Chemical Mechanism

- **Regional Acid Deposition Model, 2nd generation (RADM2)**
- **Regional Atmospheric Chemistry Mechanism (RACM)**
- **RACM - Mainz Isoprene Mechanism (RACM-MIM)**
- **RACM - Earth System Research Laboratory (RACM-ESRL), *updated RACM-MIM***
- **Carbon-Bond Mechanism version Z (CBM-Z)**
- **Carbon-Bond Mechanism (CB05)**
- **Model of Ozone and Related Chemical Tracers (MOZART)**
- **Statewide Air Pollution Research Center (SAPRC99)**
- **Common Reactive Intermediates gas-phase Mechanism (CRIMech)**

different implementations, coupled to different aerosol schemes and aqueous chemistry, suitable for different applications ranging from regional air quality to global atmospheric chemistry simulations

Comments About Chemistry Options in WRF-Chem

- **RADM2 based options:**
  - Most computationally efficient options for Ozone chemistry
  - for air quality modeling, not all biogenic VOCs are included
- **RACM based options:**
  - more updated version of RADM, more biogenic VOCs are included, more coupling options
  - When coupled with modal aerosol and Secondary Organic Aerosol (SOA) module – is the top of the line setup for real-time air quality forecasting (at ESRL)
- **Carbon Bond based options:**
  - Widely used for research applications with sectional aerosol module
  - Studies using CAM5 physics and modal aerosols
  - Warning: CBM4 and NMHC9 are not yet fully functional in WRF-Chem

Comments About Chemistry Options in WRF-Chem

- **MOZART based options:**
  - Used in some of global atmospheric chemistry models (NCAR, GFDL)
  - Can run alone, or with GOCART for gas-phase experiments
  - Run with MOSAIC aerosols for global aerosol studies
- **SAPRC99 based options:**
  - Widely used for research applications with sectional aerosol module and VBS for organic aerosols
- **CRIMech based options:**
  - Very detailed chemistry mechanism based on Master Chemical Mechanism
  - Suitable for detailed chemistry analysis; If you have anthropogenic emissions and measurements to run and evaluate this scheme
  - Computationally very expensive!
  - Geosci. Model Dev. Discuss., 7, 871–929, 2014

Comments About Chemistry Options in WRF-Chem

KPP in WRF-Chem

Kinetic PreProcessor (KPP) reads chemical reactions and rate constants from ASCII input files and automatically generates code for chemistry integration using the Rosenbrok solver

Advantages:

No KPP for aerosols!

- less time consuming than manual coding
- less error prone
- numerically efficient
- flexibility in updating mechanism with additional species and equations
- suitable for adjoint code development

References:

- Damian, V., et al. (2002), The kinetic preprocessor KPP - a software environment for solving chemical kinetics, *Comput. Chem. Eng., 26(11), 1567-1579.*
- Sandu, A., and R. Sander (2006), Technical note: Simulating chemical systems in Fortran90 and Matlab with the Kinetic PreProcessor KPP-2.1, *Atmos. Chem. Phys., 6, 187-195.*
- Verwer, J., Spee, E., Blom, J. G., and Hunsdorfer, W. (1999), A second order Rosenbrock method applied to photochemical dispersion problems, *SIAM Journal on Scientific Computing, 20, 1456–1480.*
- www.mpch-mainz.mpg.de/~salzmann/my_home/sub/wkc.html

Chemistry Mechanisms

Chemical mechanisms	*Fixed versions*	*KPP*	*Coupled aerosol schemes*	*Indirect effect*
RADM2	Yes	Yes	MADE/SORGAM, GOCART	MADE/SORGAM with aqueous phase chemistry
RACM	-	Yes	MADE/SORGAM, GOCART	MADE/SORGAM with aqueous phase chemistry
RACM-MIM	-	Yes	-	-
RACM-ESRL	-	Yes	MADE/SORGAM, MADE/ SOA_VBS	-
CBM-Z	Yes	Yes	MADE/SORGAM, MOSAIC, MAM	4bin/8bin with aqueous phase chemistry; CAM5 -MAM and aqueous phase chemistry
CB05	-	Yes	MADE/SORGAM	with aqueous phase chemistry
MOZART	-	Yes	GOCART, MOSAIC	-
SAPRC99	-	Yes	MOSAIC	8bin with aqueous phase chemistry
CRIMech	-	Yes	MOSAIC	4bin/8bin with with aqueous phase chemistry

Additional Considerations

- Photolysis: Madronich TUV photolysis is the only option that is aware of parameterized convection.
  - FAST-J and FTUV photolysis are not impacted by parameterized convection
- Direct effect: radiation is impacted by aerosol when using:
  - Goddard short wave,
  - RRTMG (both short and long wave)
  - CAM5
- Not all microphysics can be used with indirect effect.
  - Lin et al and Morrison are needed for indirect effect
    - Can use only the modal and sectional approaches.
  - Only double moment for resolved clouds (Qc) when used with WRF-Chem
    - These schemes in WRF (not WRF-Chem) have cloud droplet number (qndrop) as single moment.

Additional Considerations

- For now, indirect effect needs aqueous phase chemistry to be turned on at the same time
- Don't use too many processors when running real.exe. Some systems prefer a smaller number of nodes.
- Check for consistency between the netCDF file attributes with all input data files
  - e.g., mminlu should be the same for: wrfinput, wrfbdy, wrffdda, wrfchemi, wrffirechemi, wrfbiochemi, exo_coldens, and wrf_seasons)

Anthropogenic Emissions

- One can generate anthropogenic emissions input data using any desired methodology
  - Most familiar tools work with RETRO/EDGAR or NEI emissions data sets
    - PREP_CHEM_SRC (Freitas et al., 2011) V1.5 includes HTAP emissions
    - Emiss_v03
  - Tools available from NCAR/ACD
    - anthro_emiss
      - works with data on lat/lon grid
    - Aircraft emissions
  - Some groups use Sparse Matrix Operator Kernel Emissions (SMOKE) model to generate input emissions

Biogenic Emissions

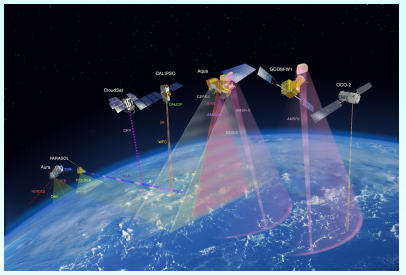
- Generate emissions input data using any desired methodology
  - Preprocessor tools available
    - PREP_CHEM_SRC (Freitas et al., 2011)
    - MEGAN (Guenther et al., 2012)
- Model has several options once static data is produced
  - Gunther,
  - BEIS,
  - MEGAN
- Recommend MEGAN (Model of Emissions of Gases and Aerosols from Nature)
  - Estimate emission of VOCs, Nox and CO from vegetation
    - Driving variables include land cover, Leaf Area Index, weather and atmospheric chemical composition
    - The land cover differs between MEGAN and WRF at this time.

Fire Emissions

- Biomass burning – use any tool available
  - Two choices available from developers:
    - PREP_CHEM_SRC (Freitas et al. 2011)
    - FINN (Wiedinmyer et al., GMD, 2011)
  - Fire location/intensity based upon satellite observations
    - MODIS
    - WF-ABBA

Wildfire Satellite Data

- Moderate-Resolution Imaging Spectroradiometer (MODIS) instruments on NASA's Aqua and Terra satellites.
  - Daily global 1km fire data
  - Available in real-time
    - https://earthdata.nasa.gov/data/near-real-time-data/firms
  - Data archives can be found online
  - Reads text (.txt) data files



- GOES, MET, MTSAT
  - Automated Biomass Burning Algorithm (ABBA) fire products (http://wfabba.ssec.wisc.edu)
  - Derived from radiances from bands 1 (visible), 2 (3.9 micron), and 4 (11 micron)
  - Available in real time (GOES: North & South America domain, MET-9: Africa, MTSAT: Austrailia)

Fire Emissions

- PREP_CHEM_SRC
  - Needs user provided satellite data files
  - Can be used in real time based upon user download ability
  - Can be used with archived observation data
  - Speciates emissions for RADM/RACM chemical mechanism
  - Provides binary intermediate file that needs to be converted to WRF input
- FINN
  - Products are available in near real-time (~ 1 day delay)
  - Smoldering emissions are expected as input (follows PREP_CHEM_SRC). Thus one needs to set:
    - **scale_fire_emiss** controls separation between smoldering and flaming emissions (Set to .true.)
      - Assumes the total emission strength is provided
    - **biomass_burn_opt** must be >= 2
  - Archived for previous years (2002 – 2013)
  - Speciate emissions for MOZART, SAPRC99 and GEOS-Chem
  - Utility program needed to convert data for use with WRF-Chem
  - http://www.acd.ucar.edu/acresp/forecast/fire-emissions.shtml

Chemical Boundary Conditions

- Default BCs not recommended due to poor upper troposphere/stratosphere representation
- Can use any method to modify the lateral boundary conditions and initial fields
  - User needs to consider the chemistry moving from GCM to their particular WRF simulation
- Available lateral boundary condition modifying routines:
  - wrfchembc
  - mozbc (Recommended), MOZART data available for download
  - Set have_bcs_chem = .true.
- Available ways to modify chemical Initial conditions
  - Recycle chemistry from previous simulation (auxinput12)
  - mozbc
    - Uses MOZART, or MACC data
  - Set chem_in_opt= 1
- Upper boundary condition
  - Initial profile does not adequately describe stratosphere

Overview

- General Considerations
- Highlight WRF-Chem Options
- Advise From Developers
  - NOAA/ESRL
  - NCAR/ACD
  - PNNL
  - Others

WRF-Chem Web Page

http://wrf-model.org/WG11

← Community leaders

← Mission, collaboration

← Real time forecast link

← WRF community news

← Model status, evaluations

← Reference/Publications list

← Online tutorial

← Contact WRF-Chem help

Other Resources/Information

- Publication list online

  http://ruc.noaa.gov/wrf/WG11/References/WRF-Chem.references.html
  - Please use this list to find papers to read and cite.
  - Please send us your publications too!
  - Inter-journal special issue on WRF-Chem; Geoscientific Model Development (GMD) and Atmospheric Chemistry and Physics (ACP)
    - GMD is restricted to WRF-Chem community version,
    - ACP open to all WRF-Chem applications
  - Add any older GMD and ACP WRF-Chem articles to special issue as well, can be done even if already in other special issues

- WRF-Chem tutorials and presentations

- WRF repository additions
  - Please make it easy for us If you plan to provide development work back to the community, (provide documentation, follow coding standards)

- Check WRF-Chem web page for model information
  http://www.wrf-model.org/WG11

Advise from NOAA/ESRL

- Chemical data assimilation – Not yet officially released
  - NCEP's Grid Point Statistical Interpolation (GSI, 3DVAR)
  - Shows significant improvement in forecast

- Ensemble Kalman Filter (enKF) has also been used, adjoint in development

- Email WRF-Chem help if:
  - You want to get in contact with developers
  - Find out the do's and do not's

Advise from NOAA/ESRL

- Focus: Research and real time forecasts; Collaboration inside ESRL
- In practice:
  - Large domain – much bigger than location, or area of interest
  - Use more detailed MEGAN biogenics
  - Use Global Chemical Model data for boundary conditions (RAQMS, MACC, etc.)
  - Use RADM/RACM chemistry with MADE/VBS
  - Use Grell-Freitas Cumulus parameterization
  - Use RRTMG radiation
  - Cycle chemistry rather than restart
- When building a nested domain
  - Use NDOWN first
    - computationally significantly cheaper
    - Build emissions an debug each domain individually
  - Using restart may require some debugging for each chemistry option
    - All fields are not in the restart files. Need to add/remove on a case by case basis

Advise From NCAR/ACD

- Focus: Research MOZART chemistry, lightning and several other emission sources
- MOZART User's Guide
  - Describes options related to using MOZART in WRF-Chem
  - http://www.acd.ucar.edu/wrf-chem/MOZART_UsersGuide.pdf
- Also provide several tools for users
  - Biogenic emissions: MEGAN
  - Anthropogenic emissions: anthro_emiss, aircraft
  - FINN fire emissions

Advise from PNNL: Best Practices for MOSAIC

Chem_opt=8 (CBM-Z coupled with MOSAIC) is the most used and tested version of MOSAIC, but users need to be aware of its limitations that depend on the application

- Interstitial aerosols using 8 size bins, but no cloud-aerosol interactions or wet removal
  - *No wet removal is an important sink that is neglected, so this version should be used only for cases with little precipitation*
- Other MOSAIC options include more detailed chemistry and species (Ca, CO_3, DMS, MSA) that will only be imporant for certain applications and locales
  - *Ca and CO_3 used for heteorgeneous reactions associated with dust and HNO_3.*
  - *DMS (biological emission) and MSA (aerosol product) useful for more complete representations of SO_4 for marine applications.*

  ] users need to define emissions
- Does not include SOA, so simulations will likely underpredict total organic matter

Chem_opt=10 is more complete and more appropriate for more applications

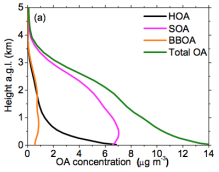
- Includes both interstitial and cloud-borne aerosols using 8 size bins, cloud aerosol interactions, aqueous chemistry, and wet removal, but still lacks SOA

4 size bin versions of these 2 options are available that reduce computational cost. If users are interested in only PM mass they may be viable options. But there may be differences in behavior of 4 and 8 size bins when computing effects of aerosols on radiation and clouds that are not fully tested.

Advise from PNNL: "Best" version of MOSAIC

Chem_opt=198 (SAPRC99 coupled with MOSAIC) is the most complete version available in the repository.

- Interstitial aerosols using 4 size bins, but no cloud-aerosol interactions or wet removal
- Volatility Basis Set (VBS) approach for SOA
- Includes Ca and CO_3 for heteorgeneous reactions associated with dust and HNO_3



Shrivastava et al., ACP, 2011

But the best / more complete versions are not yet in the public version of WRF-Chem:

- With 8 size bins — + *Fast et al., ACPD, 2014*
- With cloud-borne aerosols, cloud-aerosol interactions, aqueous chemistry, wet removal (shrivastava et al.,) — + *Shrivastava et al., JGR, 2013*
- With cloud-borne aerosols, cloud-aerosol interactions, aqueous chemistry, wet removal for both resolved clouds and parameterized shallow and deep convective clouds — + *Berg et al., GMDD, 2014*
- VBS that treats SOA as non-volatile, non-absorbing semi-solid and includes first-order gas-phase fragmentation reactions — *Shrivastava et al., JGR, 2013*

Contact PNNL investigators for packages not in public version

Considerations for Cloud-Aerosol Interactions

- PNNL's implementation of cloud-aerosol interactions have been done in either the Lin microphysics (it was modified to a 2-moment approach) or the Morrison microphysics
  - *Recommend Morrison scheme*
- Care must be taken in drawing conclusions regarding impact of cloud-aerosol interactions on other quantities (e.g. precipitation, cloud properties) based on the domain size
  - *For "large" grid spacings (Δx > 10 km) parameterized convection becomes relatively more important (depending on the region), so those simulations will be "incomplete"*
  - *If the user wants to focus on cloud-aerosol interactions, suggest Δx < 5 km or less*
- **Most common mistake** is taking the differenece of WRF (no chemistry) and a simulation using MOSAIC or MADE/SORGAM with cloud-aerosol interactions, and calling that the "indirect effect". That is not the case in the strict sense of the definition.
  - *Both microphysics schemes are only double moment for Qc when using WRF-Chem*
  - *In that case, could run the same cloud-aerosol interaction configuration with lower aerosol concentrations (either prescribing, lowering emissions, or lowering background values)*
  - *Contact WRF-Chem help, or PNNL developers for assistance*
- **Another common mistake** is that users are not aware that the autoconversion scheme changes between progn=0 & 1 when using Lin microphysics, so there is another factor affecting the results besides changing the inclusion of aerosols

CAM5 Physics Package: Recommended Settings

To run CAM5 physics in WRF that is **closest to the global CAM5 model**, use mp_physics=11, cu_physics=7, shcu_physics=2, bl_pbl_physics=9, chem_opt=503 (CBMZ_CAM5_MAM3)

- Morrison-Gettleman (M&G) microphysics, Zhang-McFarlane deep convection, UW shallow convection, UW boundary layer, and Modal Aerosol Module (MAM)
- Recommend using RRTMG radiation options, which are also used in global CAM5
- Most WRF surface layer schemes work with CAM5. **Note:** the Community Land Model (CLM, sf_surface_physics=5) has been ported to WRF, but this version is not the same as in CAM5 and has not been tested when coupled to CAM5 physics in WRF-Chem.
- CBM-Z photochemistry is used to couple with MAM. **Note:** the global CAM5 model either uses prescribe oxidants or MOZART. PNNL has coupled MAM with MOZART, but this version is not in repository.
- For emissions use, emiss_opt=9, emiss_input_opt=102 or 104.
  - *102 uses RADM/SORGAM type of emissions then converts them to what CBM-Z and MAM needs. On-line sea-salt and dust emissions can be used in this case.*
  - *For 104, the user will have to create CBM-Z and MAM specific emission species that include prescribed sea-salt and dust emissions.*

Do not switch CAM5 options with other WRF options. The M&G microphysics assumes prognostic aeorsols from MAM when computing cloud-aerosol interactions.

CAM5 Physics Package: Alternative

Depending on the application, one can use CBMZ_CAM5_MAM (chem_opt = 501):

- This version does not include cloud-aerosol interactions and wet scavenging, and runs somewhat faster since cloud-borne aerosols are not included
- In this case, prescribed aerosols are used in the Morrison & Gettleman microphysics
- This also gives users more freedom to mix and match WRF and CAM5 physics, for example:
  - *One could use Zhang-McFarlane deep convection scheme with WRF options for all other parameterizations. This is useful to compare performance of parameterizations in a series of sensitivity simulations.*
  - *Morrison & Gettleman microphysics should only be coupled to a TKE-based boundary layer scheme*

Depending on the application, one can use no chemistry (chem_opt = 0):

- In this case, users can use the meteorological CAM5 physics options without chemistry or aerosols

Note: The 7-mode aerosol model MAM7 does not work in WRF-Chem – it is a placeholder for now. Only the 3-mode version has been tested.

Note: gaschem_onoff, aerchem_onoff, wetscav_onoff, cldchem_onoff, vertmix_onoff switches have not been tested with CAM5 physics options, so use with caution to verify output

Advise For CRIMech

- Be careful about when switching to CRIMech!
  - Need to get the non-methane volatile organic compounds (NMVOC) gas-phase emissions right
  - Added the "ecrimech" and "ecrimechtno" namelist options
    - provide the right emitted species, but the emission files were generated using our own tool rather than the emissions tool provided with WRF-Chem.
  - Table 5 in their publication provides the VOC fractionation
    - Also a map for emissions to CBM-Z
    - S. Archer-Nicholls, D. Lowe, S. Utembe et al. , GMD, 2014
  - Contact Univ. of Manchester for more information/help
- MEGAN recommended for biogenic emissions
  - the older Guenther scheme to work for CRIMech too, but the speciation in MEGAN species is better.

Advise For CRIMech

- Getting the chemical boundary conditions right is important.
  - MOZART provides the quickest solution
  - The speciation is listed in Table 6.
    - S. Archer-Nicholls, D. Lowe, S. Utembe et al. , GMD, 2014
- Otherwise the standard settings for radiation schemes, etc., should work
- A word of warning about methane
  - We have included methane as a reactive species in our scheme, but haven't included it in our emitted species
    - With a large domain there's enough at the boundaries, depositional losses are relatively small
    - For smaller sized domains is would be wise to either include emissions of methane or to modify the deposition routines (and chemical mechanisms) to stop the loss terms.

Final Words of Advise

- Contact WRF-Chem help, or the developers directly if you want specifics regarding any particular feature in WRF-Chem

- Appendix

- Definitions
  - DMS = Dimethyl sulfide $(CH_3)_2$
  - MSA = Methanesulfonic Acid
    - forms from oxidation of DMS
  - Ca = Calcium and Carbonate particles (CO_3) are both found in mineral dust.
  - HNO_3 = Nitric Acid
  - SOA = Secondary Organic Aerosols
  - CAM5 = NCAR Community Atmosphere Model version 5

Running WRF-Chem with the MOZART chemistry options

MOZART

MOZART-GOCART

MOZART-MOSAIC

Thanks to those who contributed their namelist options and advice:

Rajesh Kumar, Christoph Knote, Megan Bela, Gabriele Pfister, Alma Hodzic, Mary Barth

Introduction

MOZART User's Guide

http://www.acd.ucar.edu/wrf-chem/MOZCART_UsersGuide.pdf

Describes options to select when running with the MOZART-GOCART chemical option, chem_opt = 112

Is being updated to describe MOZART-MOSAIC chemical option

MOZART appropriate for

simulations of pure gas-phase

MOZART-GOCART appropriate for

simulations of months-years,

simulations focused on trace gas chemistry

MOZART-MOSAIC appropriate for

short-term simulations or aerosol-climate studies

detailed analysis of trace gas and aerosol processes

Emissions

- Anthropogenic Emissions:
  - **emiss_opt = 7/8/10**
    for gases only/gases & GOCART/gases & MOSAIC
  - Maps hydrocarbons to MOZART mechanism
- Biogenic Emissions: **bio_emiss_opt = 3**
  - MEGAN biogenic emissions with speciation to MOZART
- Fire Emissions
  - Can use the fire plumerise module
  - **scale_fire_emiss** controls whether the total fire emission strength is split into smoldering and flaming parts
  - If FINN is used for fire emissions, then the total emission strength is provided and **scale_fire_emiss** should be **.true**.
  - **biomass_burn_opt** must be >= 2

Emissions

- Aircraft Emissions: **aircraft_emiss_opt = 1**
  - SO_2, NO, CO and CH_4 only; applied every time step
  - **kemit_aircraft** = number of model levels to apply aircraft emissions
  - Need wrfaircraftchemi_* input files for each domain

Lightning-NOx Emissions

- Lightning NO Emissions
  - Namelist variables in both *phys* and *chem* sections

  phys section
  - **lightning_option** sets what method to use and will produce 4 new 2D arrays: ic_flashrate, cg_flashrate, ic_flashcount, cg_flashcount
  - **iccg_method** sets how to get IC to CG ratio (more details in User's Guide: http://ruc.noaa.gov/wrf/WG11/Users_guide.pdf)
  - **lightning_dt** = time interval for calling lightning parameterization, should be a multiple of model time step
  - **lightning_start_seconds** = start time for calling lightning parameterization (recommend at least 10 minutes)
  - **flashrate_factor** = factor to multiply predicted number of flashes (recommend set to 1, then adjust to improve result)

Emissions

- Lightning NO Emissions (cont'd)
  - Namelist variables in both *phys* and *chem* sections

  chem section
  - **lnox_option** sets what method to vertically distribute NO emissions (see User's guide)
  - **N_IC, N_CG** sets number of moles of NO emitted per IC or CG flash
  - **lnox_passive = .true.** Emits only passive tracers. Chemistry will not be affected. (set to false to emit to NO mixing ratio)
  - **ltng_temp_upper, ltng_temp_lower** sets vertical location of maximum NO emissions for IC and CG flashes in the cloud-resolving lightning-NOx scheme
  - http://ruc.noaa.gov/wrf/WG11/Users_guide.pdf

Photolysis, Dry Deposition, etc

- Photolysis Rates: **photo_opt = 3**
  - Fast TUV code, which reads in climatological O3 and O2 overhead columns instead of using a fixed value
  - Need additional input file that can be downloaded from NCAR: http://www.acd.ucar.edu/wrf-chem/download.shtml
- Dry Deposition: **gas_drydep_opt = 1**
  - Wesely scheme requires additional input files for seasonal changes. Fortran code to make these files can be downloaded from NCAR.
- CH_4, H_2, and N_2O mixing ratios are held constant at values specified in initial conditions

Upper Boundary Condition

- Upper boundary conditions for selected trace gases **have_bcs_upper = .true.**
  - O_3, NO, NO_2, HNO_3, CH_4, CO, N_2O, N_2O_5 mixing ratios from model top to tropopause overwritten
  - **fixed_upper_bc** controls pressure level of where trace gases are set to a climatological fixed value (between model top and **fixed_upper_bc** pressure level, default = 50 hPa)
  - Between **fixed_upper_bc** pressure level and tropopause values are relaxed between WRF-Chem and climatology
  - Two input data files needed:
  - clim_p_trop.nc is a climatology of tropopause levels
  - File containing the stratosphere mixing ratios. Name is set by **fixed_ubc_inname** Climatologies from WACCM are available
  - Additional output of tropopause model level, pressure, height: tropo_lev, tropo_p, tropo_z

  http://www.acd.ucar.edu/wrf-chem/processors.shtml

Aerosols

- GOCART: Bulk aerosol scheme
- MOSAIC 4-bin: Sectional scheme
  - with simple SOA parameterization
  - with SOA volatility basis set parameterization and explicit aqueous-phase (in WRF-Chem V3.6.1)

Wet Scavenging

- Wet scavenging for resolved precipitation that is determined by the cloud physics schemes

  MOZART uses Neu and Prather (2012) scheme for trace gases: **wetscav_onoff = 1**
  http://www.mmm.ucar.edu/wrf/users/workshops/WS2011/WorkshopPapers.php
  - ➢ works with only 2 cloud physics schemes:

    Thompson scheme and CAM scheme

    WRF-Chem V3.6.1 will also have Morrison double-moment

- Convective wet scavenging **conv_tr_wetscav = 1**
  - Use with Grell convection scheme
  - WRF-Chem V3.6.1 will also have aqueous chemistry, set **conv_tr_aqchem** = 1

Special Output

- By setting up "tracking output" can get vertical profiles of meteorological and chemical species at prescribed times and lat/lons written to a special output file: **wrfout_track_d<nn>**
  - **track_loc_in** = number of track locations in input file
  - **wrfinput_track.txt** is input file containing time, latitude, longitude points
  - For **track_loc_in > 0**, get z, p, t, u, v, w, alt, qcloud, qrain, qice, qsnow, qgraup, and qvapor
  - **track_chem_num** and **track_chem_name** need to be set. Total number of chemical species must be <= 100.
  - Example on next page

Tracking Output Example

```
&domains
track_loc_in = 2,
/
&chem
track_chem_num = 2
track_chem_name = 'co', 'o3',
/
```

wrfinput_track.txt contains, for example:
2010-08-10_00:12:00 41.450 -87.300
2010-08-10_00:36:00 41.510 -87.390

The indicated output times need to be given in multiples of model time steps or else no output is produced.

Fortran format for the lines in the wrfinput_track.txt file is (A19,1X,F7.3,1X,F7.3)
Latitudes and longitudes use WRF conventions as shown above.

MOZART/GOCART namelist example

```
&dynamics
rk_ord                  = 3,
w_damping               = 1,
diff_opt                = 1,
km_opt                  =4,
base_temp               = 290.
damp_opt                = 0,
zdamp                   = 5000., 5000., 5000.,
dampcoef                = 0.01,  0.01,  0.01
diff_6th_opt            = 0,
diff_6th_factor         = 0.12,
khdif                   = 0,   0,   0,
kvdif                   = 0,   0,   0,
non_hydrostatic         = .true., .true., .true.,
moist_adv_opt           = 2,   0,   0,
scalar_adv_opt          = 2,   0,   0,
chem_adv_opt            = 2,   0,   0,
tke_adv_opt             = 2,   0,   0,
time_step_sound         = 4,   4,   4,
h_mom_adv_order         = 5,   5,   5,
v_mom_adv_order         = 3,   3,   3,
h_sca_adv_order         = 5,   5,   5,
v_sca_adv_order         = 3,   3,   3,
/

&physics
mp_physics              = 8,   3,   3,
progn                   = 0,
ra_lw_physics           = 4,   1,   1,
ra_sw_physics           = 4,   1,   1,
radt                    = 30,  30,  30,
sf_sfclay_physics       = 1,   1,   1,
sf_surface_physics      = 2,   2,   2,
bl_pbl_physics          = 1,   1,   1,
bldt                    = 0,   0,   0,
cu_physics              = 1,   1,   0,
cudt                    = 0,   5,   5,
surface_input_source    = 1,
num_soil_layers         = 4,
sf_urban_physics        = 0,   0,   0,
mp_zero_out             = 2,
mp_zero_out_thresh      = 1.e-8,
maxlens                 = 1,
maxens                  = 3,
maxens2                 = 3,
maxens3                 = 16,
ensdim                  = 144,
cu_rad_feedback         = .false.,
/
```

```
&chem
kemit                   = 1,
kemit_aircraft          = 0,
chem_opt                = 112,
bioemdt                 = 3.00,
photdt                  = 30,
chemdt                  = 3.00,
io_style_emissions      = 2,
emiss_opt               = 8,     3,
emiss_inpt_opt          = 111,
aircraft_emiss_opt      = 0,
chem_in_opt             = 0,     0,
phot_opt                = 3,     0,
gas_drydep_opt          = 1,     1,
aer_drydep_opt          = 1,
bio_emiss_opt           = 3,     0,
gas_bc_opt              = 1,     0,
gas_ic_opt              = 1,     0,
aer_bc_opt              = 1,     0,
aer_ic_opt              = 1,     0,
gaschem_onoff           = 1,     0,
wetscav_onoff           = 1,     0,
cldchem_onoff           = 0,     0,
vertmix_onoff           = 1,     0,
biomass_burn_opt        = 2,     0,
plumerisefire_frq       = 30,    0,
dust_opt                = 1,
seas_opt                = 1,
aer_op_opt              = 1,
opt_pars_out            = 1,
aer_ra_feedback         = 1,
scale_fire_emiss        = .true.
have_bcs_chem           = .true., .false., .false.,
chemdiag                = 0,
chem_conv_tr            = 1,     0,
ne_area                 = 81,
/
```

MOZART/GOCART LNOx-aircraft namelist example

```
&time_control
...
...
io_form_auxinput5           = 2,
frames_per_auxinput5        = 1, 1, 1,
auxinput5_inname            = 'wrfchemi_d<domain>_<date>',
auxinput5_interval_m        = 60, 60, 60,
io_form_auxinput6           = 2,
frames_per_auxinput6        = 1, 1, 1,
auxinput6_inname            = 'wrfbiochemi_d<domain>',
io_form_auxinput7           = 2,
frames_per_auxinput7        = 1, 1, 1,
auxinput7_inname            = 'wrffirechemi_d<domain>_<date>',
auxinput7_interval_m        = 60, 60, 60,
io_form_auxinput14          = 2,
frames_per_auxinput14       = 1, 1, 1,
auxinput14_inname           = 'wrfaircraftchemi_d<domain>_<date>',
auxinput14_interval_m       = 60, 60, 60,
iofields_filename           = "output.txt", "output_10min.txt",
ignore_iofields_warning     = .true.,
debug_level                 = 0
/


&physics
mp_physics                  = 10, 10,
do_radar_ref                = 1,
ra_lw_physics               = 4, 4,
ra_sw_physics               = 4, 4,
radt                        = 10,10,
sf_sfclay_physics           = 1, 1,
sf_surface_physics          = 2, 2,
bl_pbl_physics              = 1, 1,
bldt                        = 0, 0,
cu_physics                  = 5, 5,
cudt                        = 0, 0,
cugd_avedx                  = 1,
cu_rad_feedback             = .true.,.true.,
cu_diag                     = 1,
isfflx                      = 1,
ifsnow                      = 0,
icloud                      = 1,
surface_input_source        = 1,
num_soil_layers             = 4,
mp_zero_out                 = 2,
num_land_cat                = 20,
lightning_option            = 11,1,
iccg_method                 = 2,2,
lightning_dt                = 75,75,
cellcount_method            = 0,1,
lightning_start_seconds     = 600,600,
flashrate_factor            = 5.0,1.0,
cldtop_adjustment           = 0,0,
/
```

```
&chem
kemit                       = 11,
chem_opt                    = 112,   112,
bioemdt                     = 0,     0,
photdt                      = 15,    15,
chemdt                      = 0.,    0.,
io_style_emissions          = 2,
emiss_inpt_opt              = 111,   111,
emiss_opt                   = 8,     8,
emiss_opt_vol               = 0,
chem_in_opt                 = 0,     0,
phot_opt                    = 3,     3,
gas_drydep_opt              = 1,     1,
aer_drydep_opt              = 1,     1,
bio_emiss_opt               = 3,     3,
ne_area                     = 118,
gas_bc_opt                  = 112,   112,
gas_ic_opt                  = 112,   112,
aer_bc_opt                  = 112,   112,
aer_ic_opt                  = 112,   112,
gaschem_onoff               = 1,     1,
aerchem_onoff               = 1,     1,
wetscav_onoff               = 1,     1,
cldchem_onoff               = 0,     0,
vertmix_onoff               = 1,     1,
chem_conv_tr                = 1,     1,
conv_tr_wetscav             = 1,     1,
conv_tr_aqchem              = 1,     1,
seas_opt                    = 1,
dust_opt                    = 1,
dmsemis_opt                 = 1,
biomass_burn_opt            = 2,     2,
plumerisefire_frq           = 30,    30,
have_bcs_chem               = .true., .true.,
aer_ra_feedback             = 1,     1,
aer_op_opt                  = 1,
opt_pars_out                = 1,
diagnostic_chem             = 1,
track_chem_name             =
"co","o3","rb","no2","so2","hcho","hno3","h2o2","isopr","tol","bigalk","bc1","bc2","oc1","oc2","p
25","p10","dust_1","dust_2","dust_3","dust_4","seas_1","seas_2","seas_3","seas_4","dms",
track_chem_num              = 26,
kemit_aircraft              = 40,
aircraft_emiss_opt          = 1,1,
lnox_opt                    = 1,2,
N_IC                        = 500., 500.,
N_CG                        = 500., 500.,
/
have_bcs_upper              = .false., .false.,
fixed_ubc_press             = 50.,   50.,
fixed_ubc_inname            = 'ubvals_b40.20th.track1_1996-2005.nc',
```

MOZART/GOCART LNOx-aircraft namelist example

```
&time_control
...
...
io_form_auxinput5           = 2,
frames_per_auxinput5        = 1, 1, 1,
auxinput5_inname            = 'wrfchemi_d<domain>_<date>',
auxinput5_interval_m        = 60, 60, 60,
io_form_auxinput6           = 2,
frames_per_auxinput6        = 1, 1, 1,
auxinput6_inname            = 'wrfbiochemi_d<domain>',
io_form_auxinput7           = 2,
frames_per_auxinput7        = 1, 1, 1,
auxinput7_inname            = 'wrffirechemi_d<domain>_<date>',
auxinput7_interval_m        = 60, 60, 60,
io_form_auxinput14          = 2,
frames_per_auxinput14       = 1, 1, 1,
auxinput14_inname           = 'wrfaircraftchemi_d<domain>_<date>',
auxinput14_interval_m       = 60, 60, 60,
iofields_filename           = "output.txt", "output_10min.txt",
ignore_iofields_warning     = .true.,
debug_level                 = 0
/


&physics
mp_physics                  = 10,
progn                       = 1,
ra_lw_physics               = 4,
ra_sw_physics               = 4,
radt                        = 30,
sf_sfclay_physics           = 5, ! Ravans suggestion
sf_surface_physics          = 2, ! NOAH
bl_pbl_physics              = 5, ! MYNN 2.5
bldt                        = 0,
cu_physics                  = 5, ! Grell 3D
cudt                        = 0,
cugd_avedx                  = 1, ! set to 3 for 4km run, 1 for
30km
isfflx                      = 1,
ifsnow                      = 1,
icloud                      = 1,
surface_input_source        = 1,
num_soil_layers             = 4,
sf_urban_physics            = 1,
sst_update                  = 1,
mp_zero_out                 = 2,
mp_zero_out_thresh          = 1.e-8,
cu_rad_feedback             = .true.,
cu_diag                     = 1,
slope_rad                   = 0,
topo_shading                = 0,
/
```

```
&chem
kemit                       = 8,
chem_opt                    = 999,
bioemdt                     = 30,
photdt                      = 30.,
chemdt                      = 6.0,
io_style_emissions          = 2,
emiss_inpt_opt              = 102,
emiss_opt                   = 10,
chem_in_opt                 = 1,
phot_opt                    = 3,
gas_drydep_opt              = 1,
aer_drydep_opt              = 1,
bio_emiss_opt               = 3,
gas_bc_opt                  = 1,
gas_ic_opt                  = 1,
aer_bc_opt                  = 1,
aer_ic_opt                  = 1,
gaschem_onoff               = 1,
aerchem_onoff               = 1,
wetscav_onoff               = 1,
cldchem_onoff               = 1,
vertmix_onoff               = 1,
chem_conv_tr                = 1,
conv_tr_wetscav             = 1,
conv_tr_aqchem              = 1,
seas_opt                    = 2,
dust_opt                    = 3,
dmsemis_opt                 = 1,
biomass_burn_opt            = 0,
aer_ra_feedback             = 1,
ne_area                     = 140,
opt_pars_out                = 1,
have_bcs_chem               = .true.,
have_bcs_upper              = .false.,
/
```

CRIMech namelist example

```
&physics
mp_physics                          = 2,
progn                               = 1,
ra_lw_physics                       = 4,
ra_sw_physics                       = 2,
radt                                = 5,
sf_sfclay_physics                   = 2,
sf_surface_physics                  = 2,
bl_pbl_physics                      = 2,
bldt                                = 0,
cu_physics                          = 5,
cu_diag                             = 1,
cugd_avedx                          = 1,
cudt                                = 0,
isfflx                              = 1,
ifsnow                              = 1,
icloud                              = 1,
surface_input_source                = 1,
num_soil_layers                     = 4,
sf_urban_physics                    = 1,
mp_zero_out                         = 2,
mp_zero_out_thresh                  = 1.e-12
maxiens                             = 1,
maxens                              = 3,
maxens2                             = 3,
maxens3                             = 16,
ensdim                              = 144,
cu_rad_feedback                     = .true.,
sst_update                          = 1,
sst_skin                            = 1,
usemonalb                           = .true.,
tmn_update                          = 1,
/

&chem
kemit                               = 5,
chem_opt                            = 601,
bioemdt                             = 30,
photdt                              = 30,
chemdt                              = 1,
io_style_emissions                  = 2,
emiss_inpt_opt                      = 121,
emiss_opt                           = 20,
emiss_opt_vol                       = 0,
chem_in_opt                         = 0,
phot_opt                            = 2,
gas_drydep_opt                      = 1,
aer_drydep_opt                      = 1,
bio_emiss_opt                       = 3,
gas_bc_opt                          = 0,
gas_ic_opt                          = 0,
aer_bc_opt                          = 0,
aer_ic_opt                          = 0,
gaschem_onoff                       = 1,
aerchem_onoff                       = 1,
wetscav_onoff                       = 1,
cldchem_onoff                       = 1,
vertmix_onoff                       = 1,
chem_conv_tr                        = 1,
seas_opt                            = 3,
dust_opt                            = 0,
dmsemis_opt                         = 2,
biomass_burn_opt                    = 0,
have_bcs_chem                       = .true.,
aer_ra_feedback                     = 1,
chemdiag                            = 1,
n2o5_hetchem                        = 2,
ne_area                             = 250,
/
```