WRF Modeling System Overview

Jimy Dudhia

WRF Modeling System Overview

Jimy Dudhia

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?

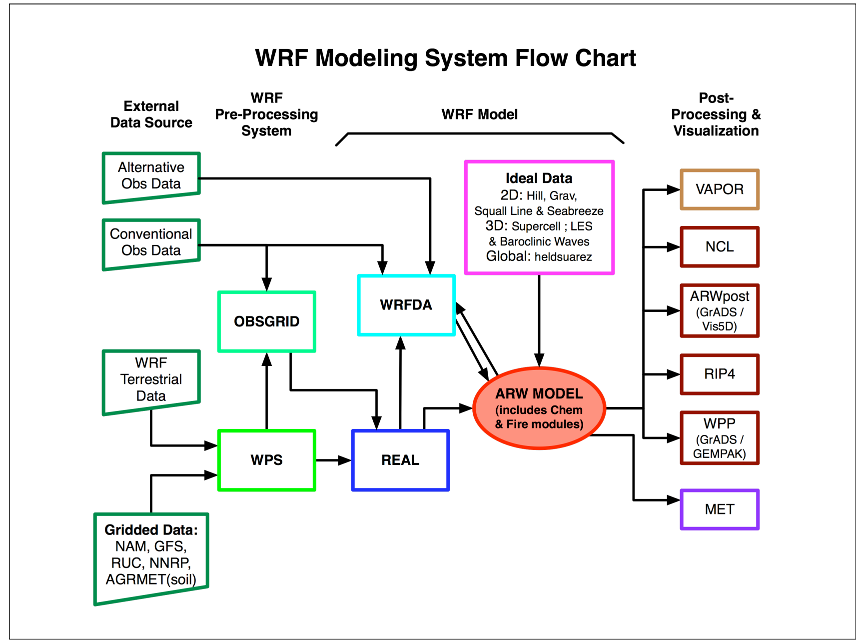
- WRF has two dynamical cores: The Advanced Research WRF (ARW) and Nonhydrostatic Mesoscale Model (NMM)
  - Dynamical core includes mostly advection, pressure-gradients, Coriolis, buoyancy, filters, diffusion, and time-stepping
- Both are Eulerian mass dynamical cores with terrain-following vertical coordinates
- ARW support and development are centered at NCAR/MMM
- NMM development is centered at NCEP/EMC and support is provided by NCAR/DTC (operationally now only used for HWRF)
- 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 Community Model

- Version 1.0 WRF was released December 2000
- Version 2.0: May 2004 (add nesting)
- Version 3.0: April 2008 (add global ARW version)
- … (major releases in April, minor releases in summer)
- Version 3.8: April 2016
- Version 3.8.1: August 2016
- Version 3.9: April 2017
- Version 3.9.1(.1) (August 2017)

What can WRF be used for?

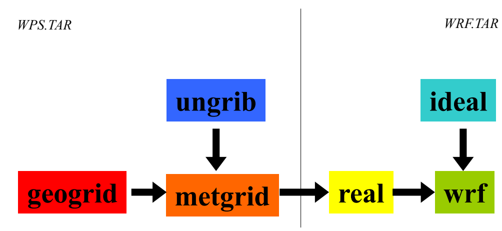
- ARW and NMM
  - Atmospheric physics/parameterization research
  - Case-study research
  - Real-time NWP and forecast system research
  - Data assimilation research
  - Teaching dynamics and NWP
- ARW only
  - Regional climate and seasonal time-scale research
  - Coupled-chemistry applications
  - Global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



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 and NMM dynamical cores)
  - Initialization programs for real and (for ARW) idealized data (real.exe/ideal.exe)
  - Numerical integration program (wrf.exe)
- Graphics and verification tools including MET
- WRFDA (separate tutorial)
- WRF-Chem (separate tutorial)
- WRF-Hydro – hydrology model coupled to WRF
- 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)
- GEOGRID program
  - Map projection information
    - 2d gridded latitude, longitude, Coriolis parameter, map-scale factors, etc.
  - Topographic information
    - 2d gridded elevation, vegetation and soil categories, etc.

Mesoscale & Microscale Meteorology Laboratory / NCAR

Real-Data Applications

- Need time-dependent information
- Initial conditions (initial analysis time)
- Boundary conditions (later times)
  - except if running WRF globally
- UNGRIB and METGRID programs
  - 3d fields of horizontal wind, temperature, geopotential height, relative humidity
  - 2d fields of surface or sea-level pressure, surface temperature, relative humidity, horizontal winds
  - Time-sensitive land-surface fields: snow-cover, soil temperature, soil moisture

Mesoscale & Microscale Meteorology Laboratory / NCAR

Real-Data Applications

- Regional domains need *specified* lateral boundary conditions at later times (e.g. every 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

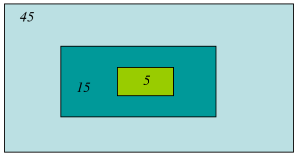
Mesoscale & Microscale Meteorology Laboratory / NCAR

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 4 interior points
- Lower Boundary Condition (step-wise)
  - New SSTs are read in and overwritten at each analysis time from *wrflowinp* file

Nesting

- Running multiple domains with increasing resolution in nested areas
- Parent has *specified* boundary conditions from *wrfbdy* fie
- *Nested* boundary conditions come from parent



Nesting (Two-Way)

- Lateral boundary condition is provided by parent domain at every parent step
- Method is same as for outer domain (specified and relaxation zones)
- Additional fields include vertical motion and microphysics species
- Feedback: Interior of nest overwrites overlapped parent area

One-Way Nesting

- As two-way nesting but no feedback
- Can also be done with NDOWN program to take a previous WRF run output and provide nest boundary conditions at parent output frequency
  - Uses parent WRF run instead of analysis for initial and lateral boundary conditions

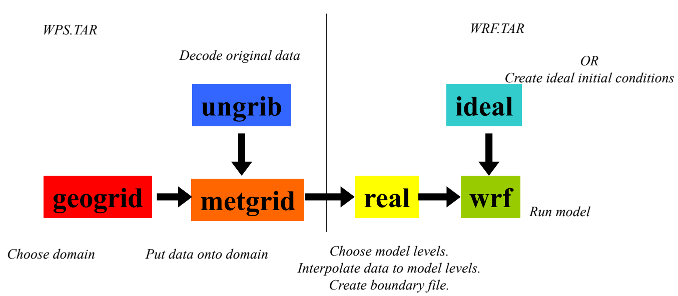
WPS Functions

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

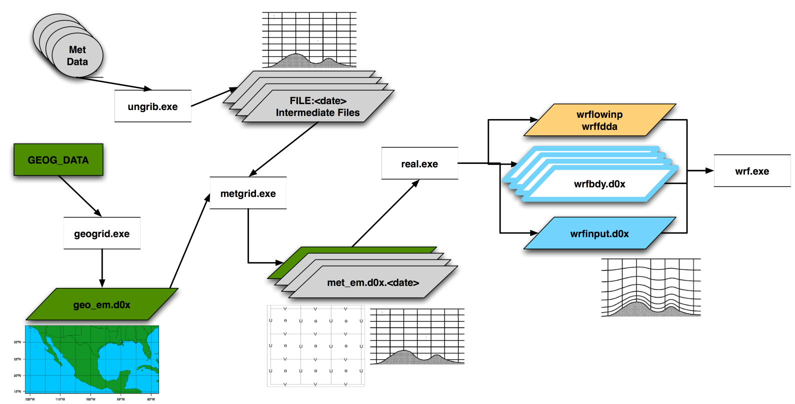
WPS Data

- 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)

WPS and WRF Program Flow



Data Flow



WRF real and ideal functions

- 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
- IDEAL (ARW only)
  - Programs for setting up idealized case
  - Simple physics and usually single sounding
  - Initial conditions and dynamic balance

WRF Model

- WRF
  - Dynamical core (ARW or NMM) is compile-time selectable
  - Uses initial conditions from REAL or IDEAL (ARW)
  - 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

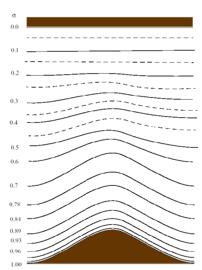
ARW Dynamics

Key features:

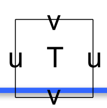
- Fully compressible, non-hydrostatic (with hydrostatic option)
- Mass-based terrain following coordinate, η

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

where π is hydrostatic pressure, μ is column mass

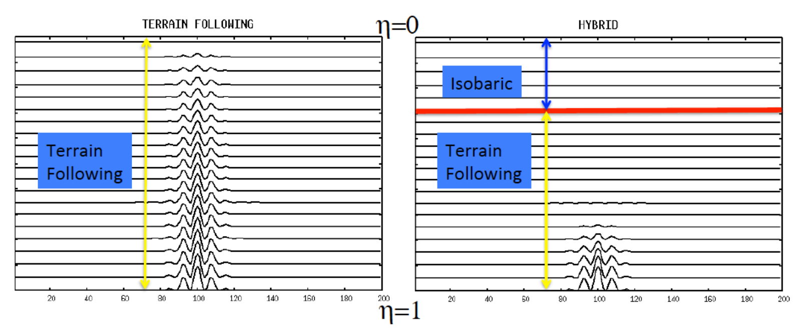


- Arakawa C-grid staggering



ARW Model

- New hybrid vertical coordinate option in V3.9
- Isobaric at top means less noise in upper-air output over mountains



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

Graphics and Verification Tools

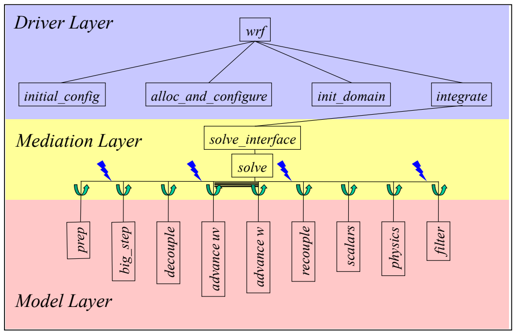
- ARW and NMM
  - RIP4 (Read, Interpolate and Plot)
  - Unified Post-Processor (UPP)
    - Conversion to GriB (for GrADS and GEMPAK)
  - MET (Model Evaluation Toolkit)
- ARW
  - NCAR Graphics Command Language (NCL)
  - ARWpost
    - Conversion program for GrADS
  - VAPOR (3D visualization tool)
  - IDV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

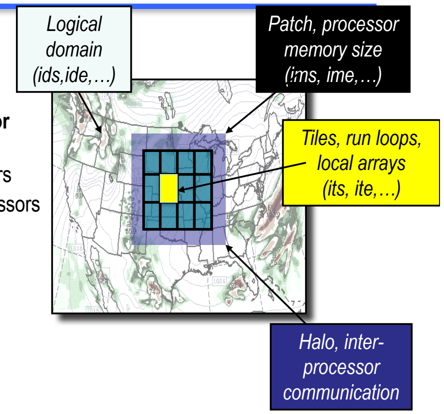
WRF Hierarchical Software Architecture

- Driver Layer
  - Memory allocation, nest starting, time-stepping, I/O
- Mediation Layer
  - Solver
- Model Layer
  - Dynamics, physics



WRF Two-Layer Domain Decomposition (patches, tiles, halo)

- **Single version of code enabled for efficient execution on:**
  - Shared-memory multiprocessors
  - Distributed-memory multiprocessors
  - Distributed clusters of SMPs
  - Vector and scalar processors



Registry File

- Input for automatic code generation
- Designed to make adding arrays or new namelist parameters easy
- Allocates, passes, and declares, listed arrays for nesting, i/o and "solver" routines
    - Solver advances one domain by one time step
  - From solver, it can be passed to parts of the low-level code via argument lists
- Also can add them to "halo" for MPI communications (only sometimes needed)

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 special compilation option

User Support

- Email: wrfhelp@ucar.edu
- User Web pages:

  ARW:http://www.mmm.ucar.edu/wrf/users/

  NMM:http://www.dtcenter.org/wrf-nmm/users/

  – Latest update for the modeling system
  – WRF software download
  – Various documentation
    - Users' Guides (both cores)
    - Technical Note (ARW Description)
    - Technical Note (NMM Description)

Examples of WRF Forecasts

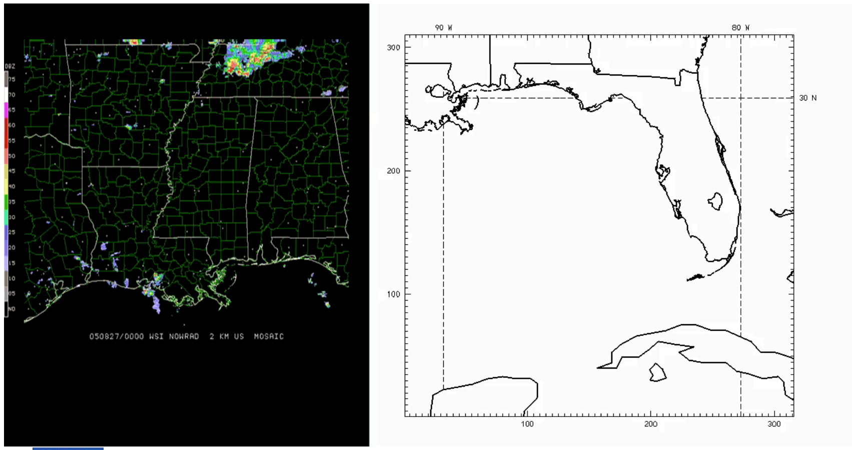
(1) Hurricane Katrina (August, 2005)

- Moving 4 km nest in a 12 km outer domain

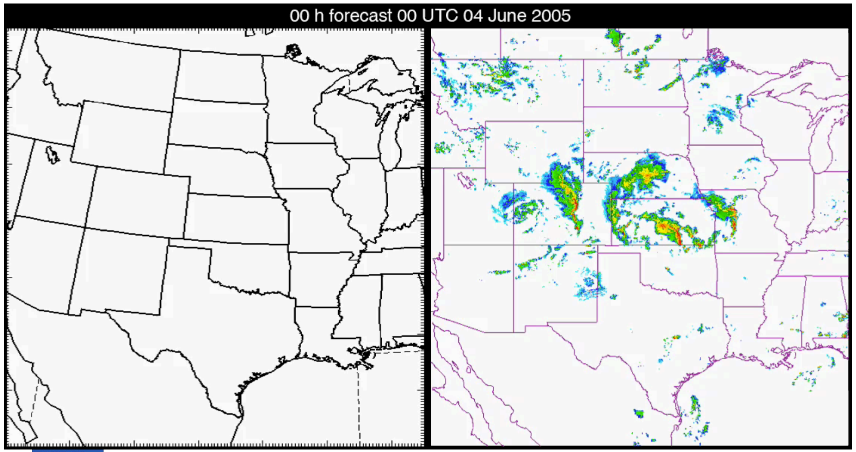
(2) US Convective System (June, 2005)

- Single 4 km central US domain

ARW Hurricane Katrina Simulation (4km)



ARW Convective-scale Forecasting (4km)



The WRF Preprocessing System (WPS): Fundamental Capabilities

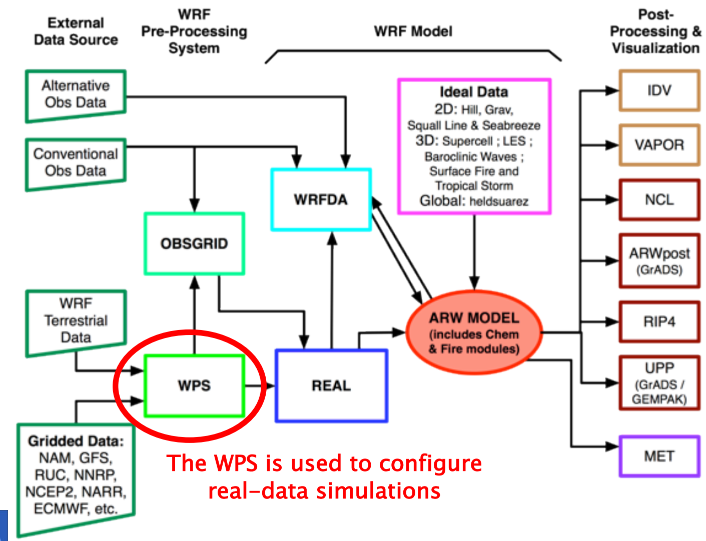
Michael Duda

The WRF Preprocessing System (WPS): Fundamental Capabilities

Michael Duda

*NCAR is sponsored by the National Science Foundation

WRF Modeling System Flow Chart



Overview

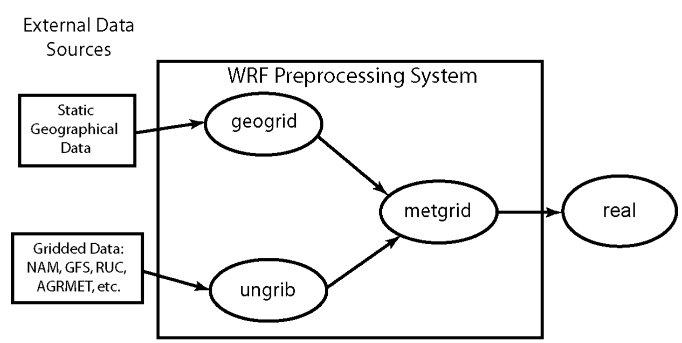
This lecture focuses on the basic use of the WPS to:

- Define a single simulation domain
  - The setup of *nested domains* is covered in a later talk
- Preprocess time-varying atmospheric and land-surface datasets
- Horizontally interpolate datasets for use as initial and boundary conditions for WRF

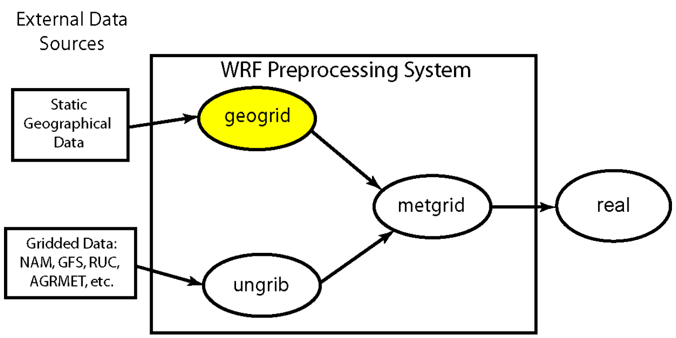
– Practical details of *actually running* the WPS are covered this afternoon and in a live demo tomorrow

– *Advanced features* of the WPS are described on Thursday

WPS Program Flowchart



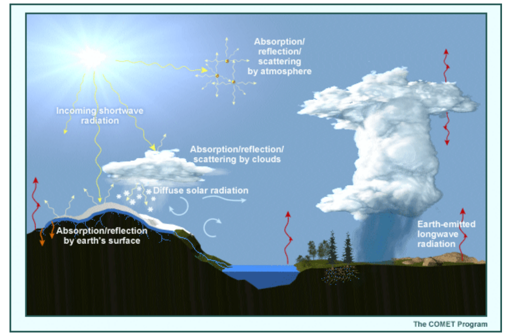
The *geogrid* program



geogrid: think geographical

The *geogrid* program

Let's suppose we wish to perform a simulation for the domain below…



- Where is this domain located?
- What area does the domain cover?
- How well do we resolve the atmosphere and land surface (horizontally)?
- What sources of data do we use for topography, vegetation categories, and soil categories?

Using the geogrid program, we answer these questions from the perspective of the WRF model.

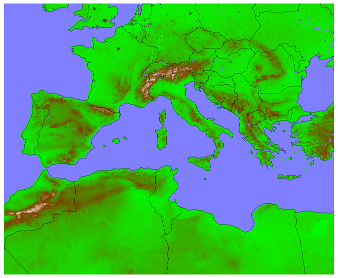
The *geogrid* program

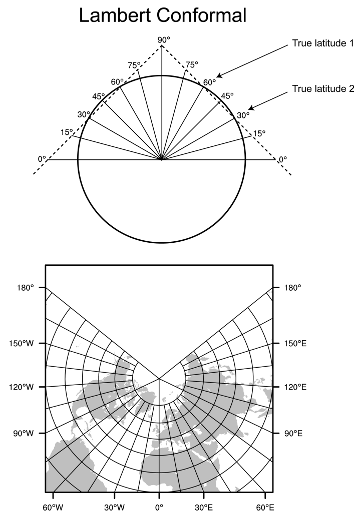
- We use the geogrid program to define:
  - Map projection (all domains must use the same projection)
  - Geographic location of domains
  - Dimensions of domains
  - Horizontal resolution of domains
- Geogrid provides values for static (time-invariant) fields at each model grid point
  - Compute latitude, 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) from global datasets

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!)

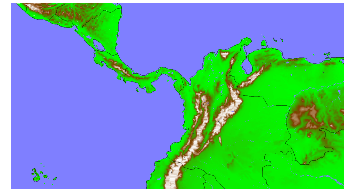
ARW Projections: Lambert Conformal

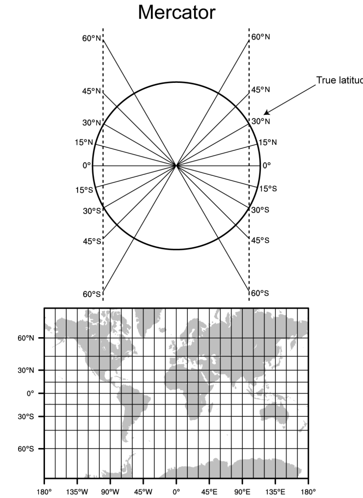




- Well-suited for mid-latitudes
- Domain cannot contain either pole
- Domain cannot be periodic in west-east direction
- Either one or two *true latitudes* may be specified
  - If two are given, the order doesn't matter

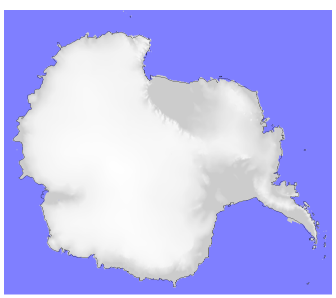
ARW Projections: Mercator

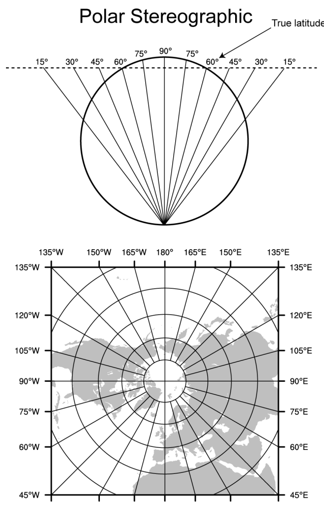




- Well-suited for low-latitudes
- May be used for "channel" domain (periodic domain in west-east direction)
- A single true latitude is specified
  - Cylinder intersects the earth's surface at +/- truelat

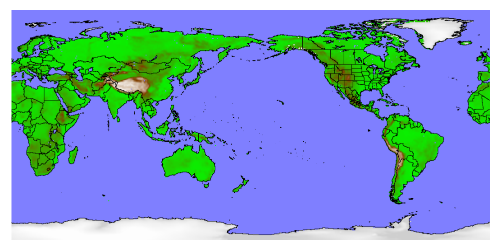
ARW Projections: Polar Stereographic

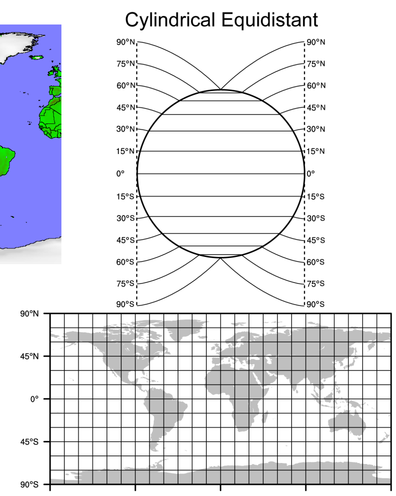




- Good for high-latitude domains, especially if domain must contain a pole
- A single true latitude is specified

ARW Projections: Cylindrical Equidistant





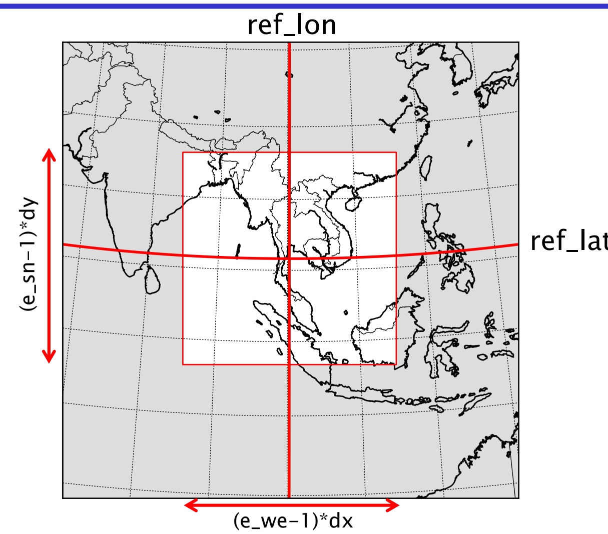
- Required for global domains
- May be used for regional domains
- Can be used in its normal or rotated aspect

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: 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

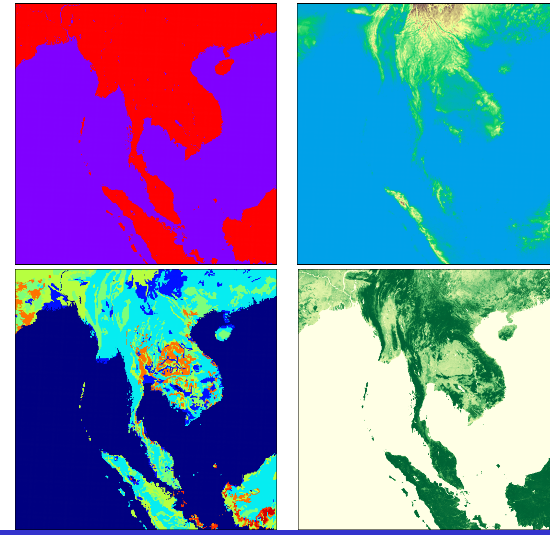
Geogrid: Program Output

- 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



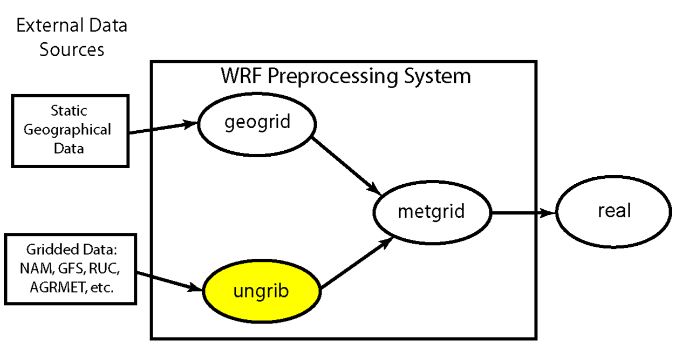
Land-water mask

Topography height

Top-layer dominant soil type

Climatological vegetation fraction (January)

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: <u>V</u>ariable <u>tables</u>)

- 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
  - *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)

See p. 3-33

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:

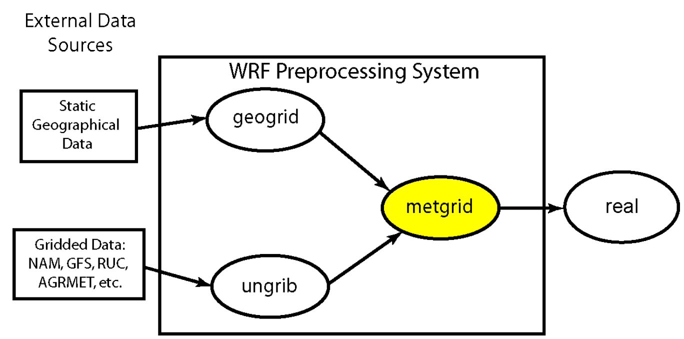
  FILE:2007-07-24_00

  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



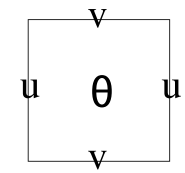
metgrid: think meteorological

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!*)

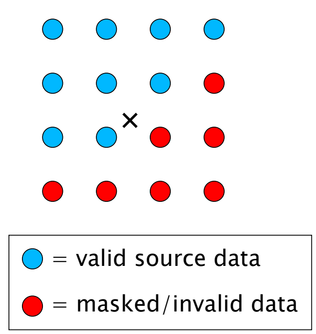


A single ARW grid cell, with "u", "v", and "θ" points labeled.

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



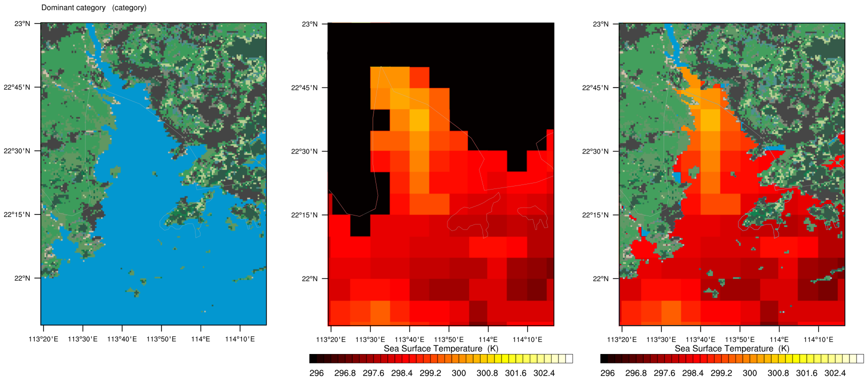
Suppose we need to interpolate to point X

- 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.



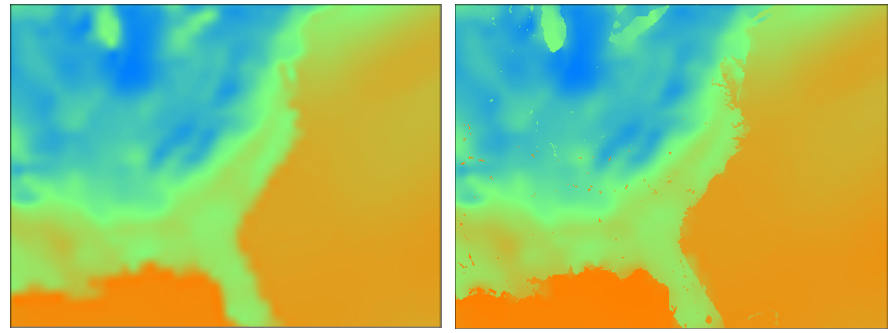
A high-resolution WRF domain centered on Pearl River Estuary.

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.



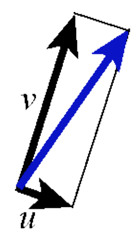
Skin temperature field interpolated from GFS 0.5-deg field with no mask using a sixteen-point interpolator.

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

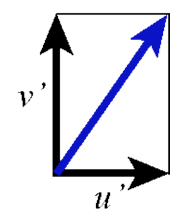
Metgrid: Wind Rotation

- 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'`

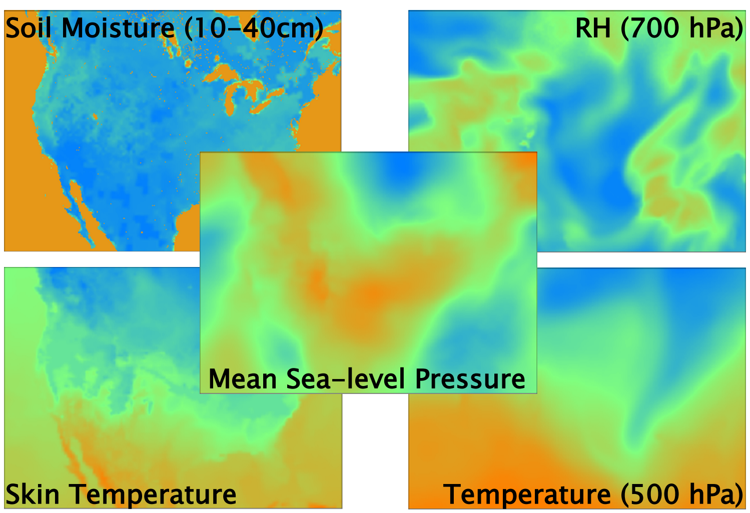
Metgrid: Program Output

- 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:

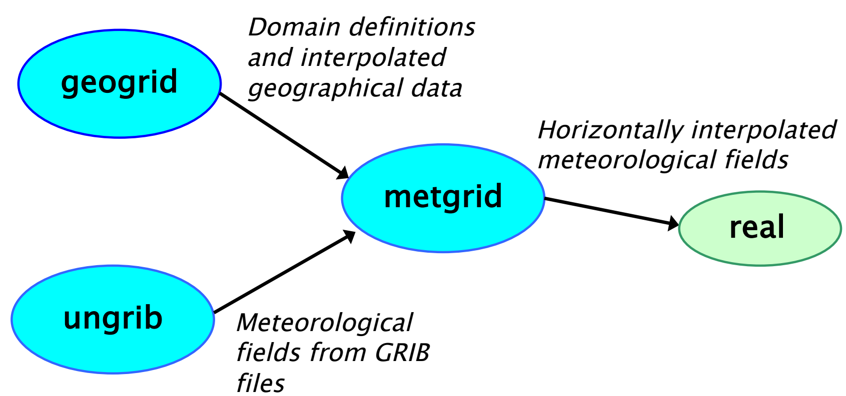
  ARW: `met_em.d0`*`n`*`.`*`YYYY-MM-DD_HH:mm:ss`*`.nc`

  (where *n* is the domain ID number)

Metgrid: Example Output

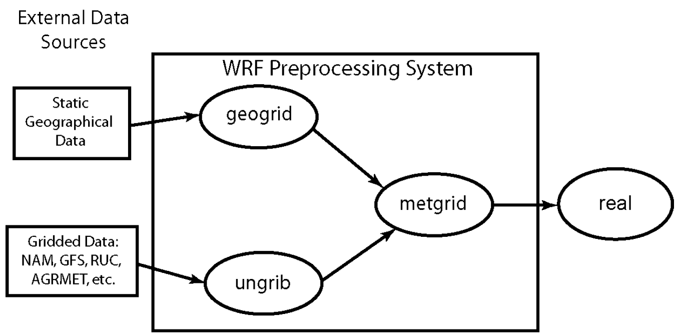


WPS Summary



And finally…

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



Questions?

Extra slides

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

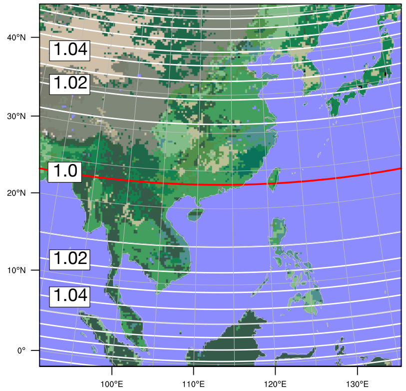
- Can easily check min/max map scale factor using, e.g., ncview!

Why do map projections matter?

Example:

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

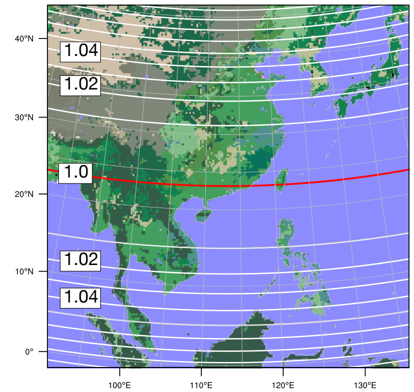
Choosing both true latitudes in the center of the WRF domain leads to maximum map scale factors of 1.0975, corresponding to a *minimum physical grid distance of 27/1.0975 = 24.6 km.*



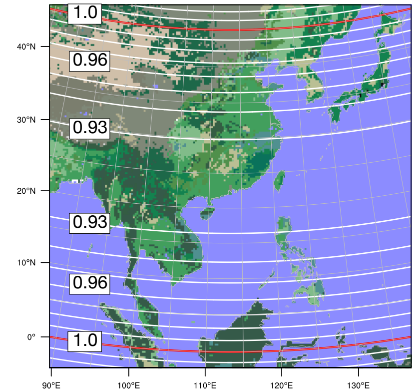
Above: Contours of map scale factor (white; interval 0.01) with true latitudes (red).

Why do map projections matter?

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



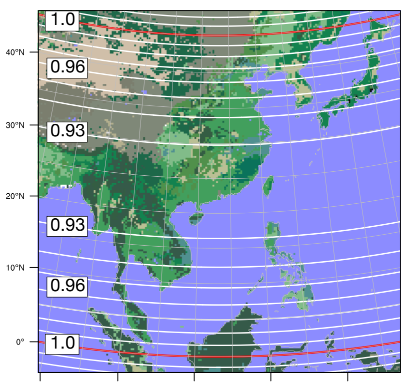
Above: Reference projection as on previous slide; maximum grid distance is 27 km (at true latitude).



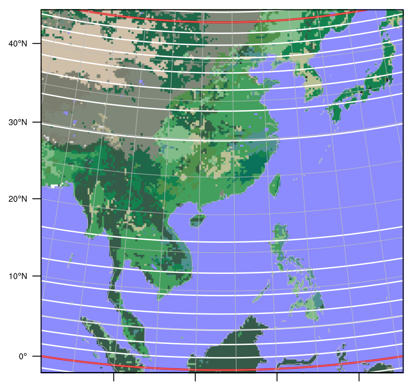
Above: The maximum map scale factor is 1.03, but the minimum is 0.924, corresponding to a physical distance of 29.2 km.

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.

Program Real: Description of General Functions

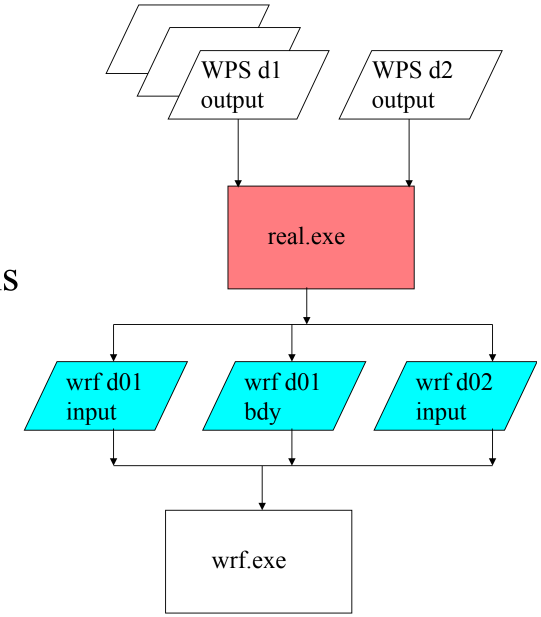
Dave Gill

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

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

Function

- **Run-time options**
  - specified in the Fortran namelist file (namelist.input for real and WRF)
- **Compile-time options**
  - Changes inside of the source code
  - Compiler flags
  - CPP ifdefs
  - Modifications to the Registry file

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 monotonic*** 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***

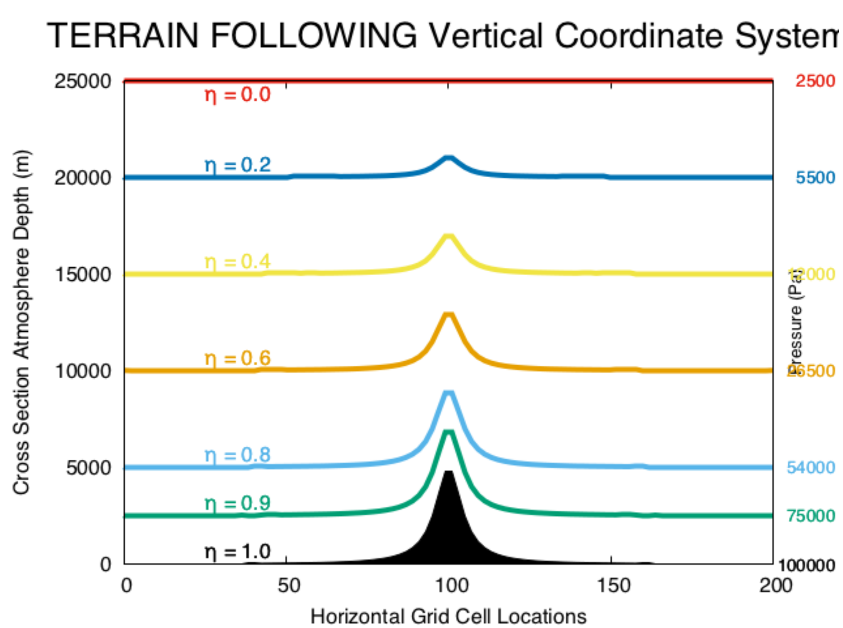
Hybrid Vertical Coordinate

- New in WRF v3.9 is the capability to have a hybrid vertical coordinate: a terrain following coordinate near the surface and relaxing to isobaric surfaces aloft
- A compile-time option is required:

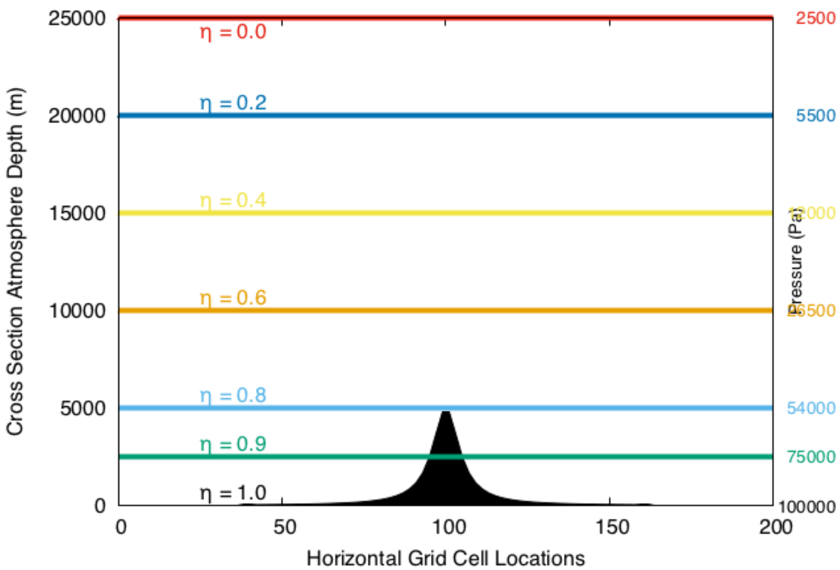
```
configure -hyb
```

- A run-time option is required:

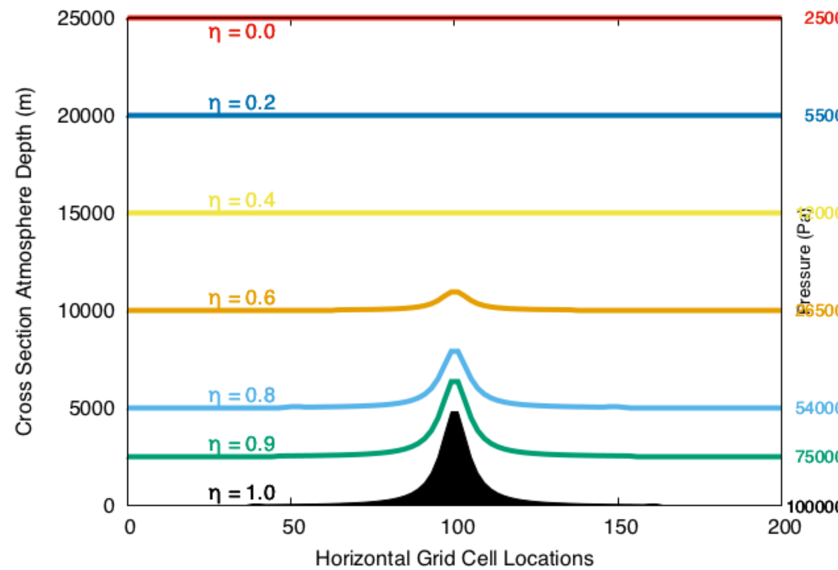
```
&dynamics
hybrid_opt = 2
/
```



ISOBARIC Vertical Coordinate System



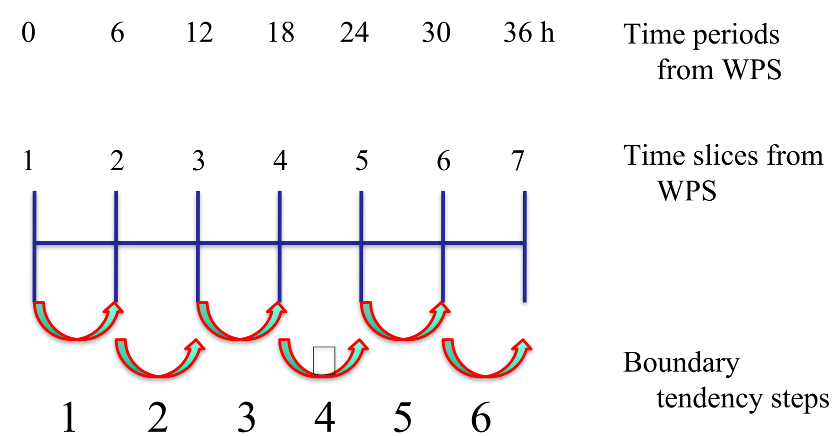
HYBRID Vertical Coordinate System



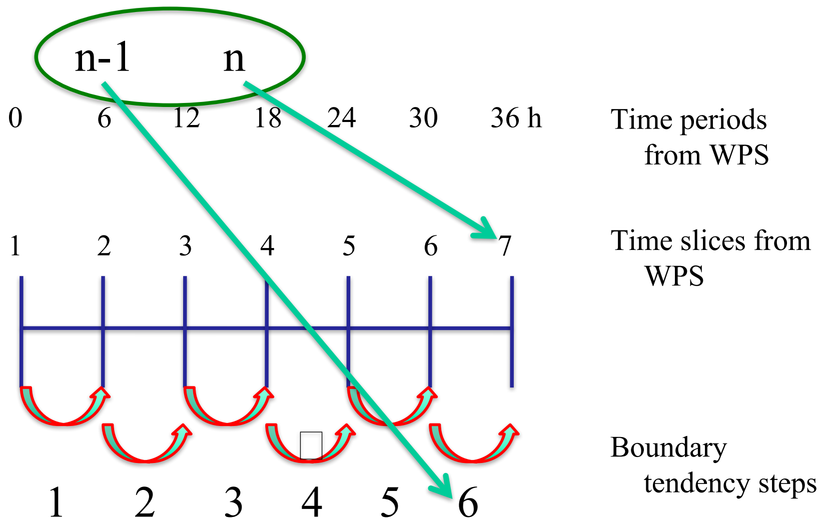
Standard Generated Output

- 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 (look at MPAS for global simulations)
- 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

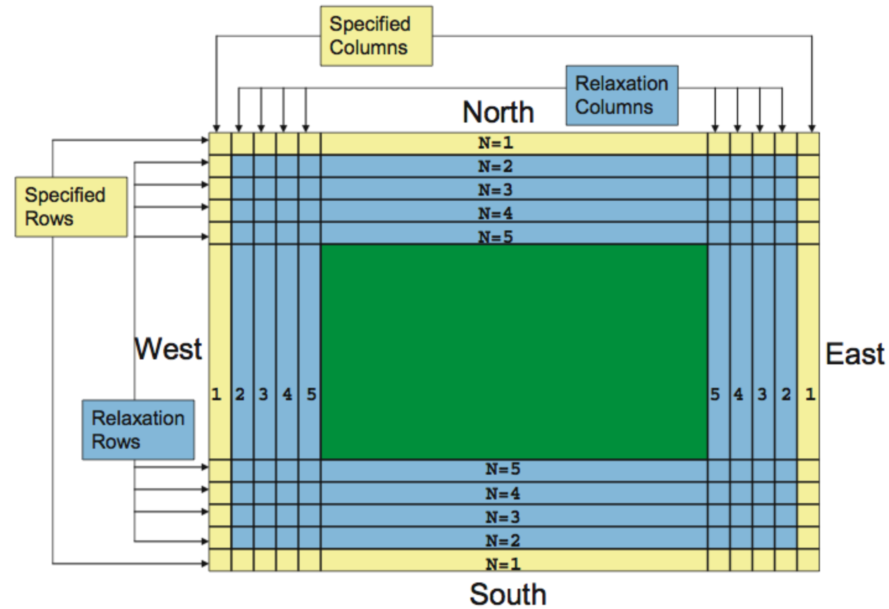
Lateral Boundary Condition Times



Lateral Boundary Condition Times



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



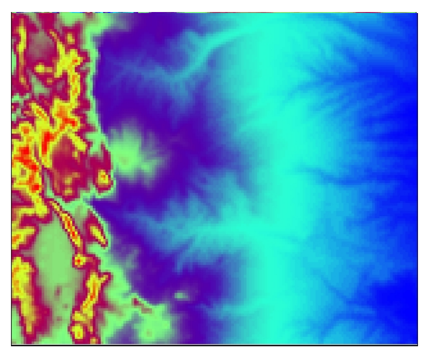
Vertical Interpolation

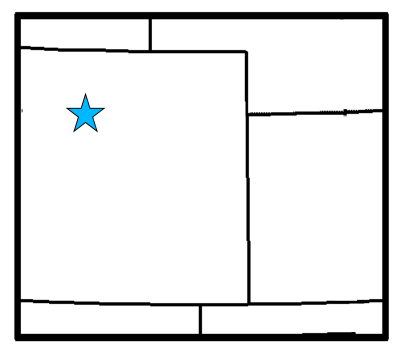
- A number of vertical interpolation options are available to users
- The options can have a significant impact on the initial conditions passed to the model
- More information is contained in the info file ***README.namelist*** in the ***run*** directory
- Options are located in the ***&domains*** namelist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

Vertical Interpolation

- Impact: *Expected region of changes*
- *Non-standard setting*
- Which level is being viewed

- Topography and domain for difference plots, 160x140, 4 km, input = 40 km NAM

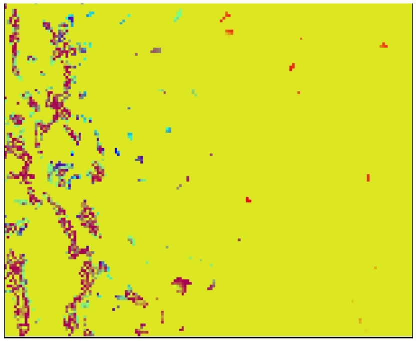


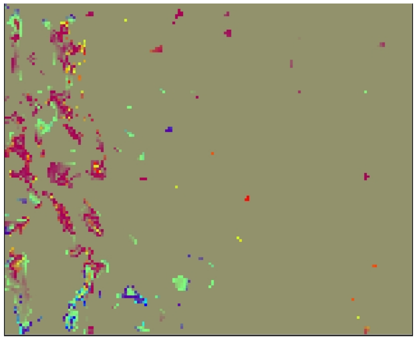


Vertical Interpolation

- Impact: few lowest levels only
- force_sfc_in_vinterp = 0
- η level 1

- Theta (-8 K blue, 0 K yellow)
- U (-3 m/s blue, 2 m/s red)

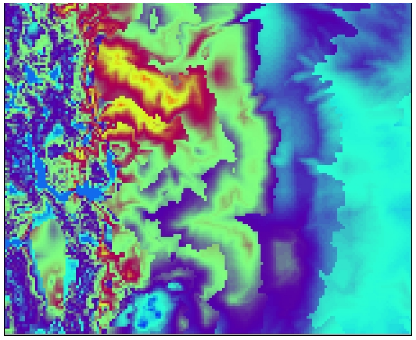


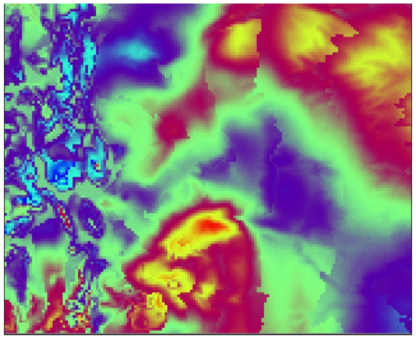


Vertical Interpolation

- Impact: few lowest levels only
- force_sfc_in_vinterp = 6
- η level 4

- Theta (0 K blue, 10 K red)
- U (-5 m/s blue, 6 m/s red)

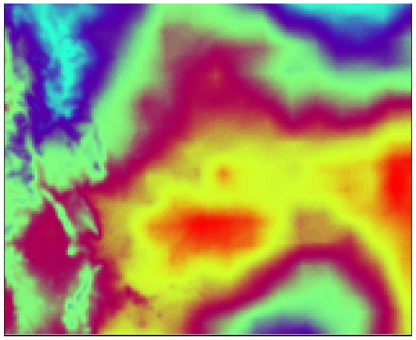


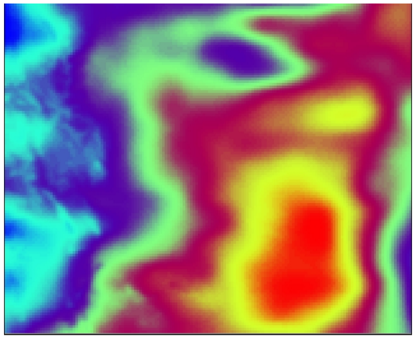


Vertical Interpolation

- Impact: above first 4 levels, most near tropopause
- lagrange_order = 2
- η level TOP

- Theta (0.7 K blue, 1.6 K red)
- U (0.4 m/s blue, 1.4 m/s red)

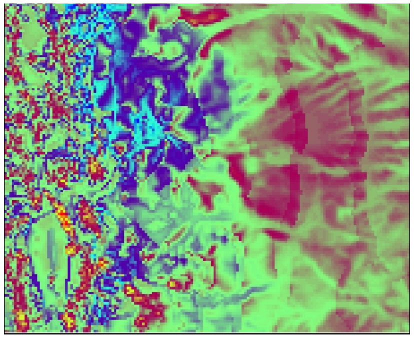


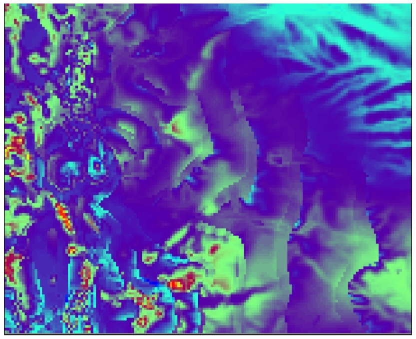


Vertical Interpolation

- Impact: lowest level only
- lowest_lev_from_sfc = T
- η level 1

- Theta (-10 K blue, 8 K red)
- U (-3 m/s blue, 7 m/s red)





Vertical Interpolation

- Impact: outer few rows and column, amplitude damps upward
- smooth_cg_topo = T
- η level 1

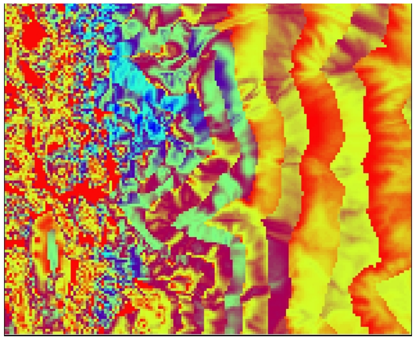
- Theta (-10 K blue, 9 K red)

- U (-6 m/s blue, 6 m/s red)

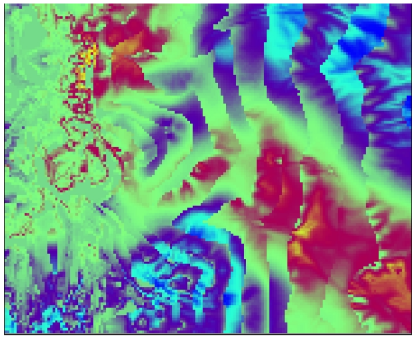
Vertical Interpolation

- Impact: lowest few levels
- use_surface = F
- η level 1

- Theta (-11 K blue, 0 K red)



- U (-3 m/s blue, 4 m/s red)



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 ***ετα surfaces*** may be computed or selected

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

Vertical Interpolation

- Select reasonable ετα 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 ετα values***
  - Initially, ***select the number*** of ετα levels
  - ***Plot profiles*** of the resultant heights
  - ***Adjust the η levels*** accordingly

- A few namelist options, the terrain elevation, and eta levels completely define the model coordinate for the WRF code

Vertical Interpolation

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

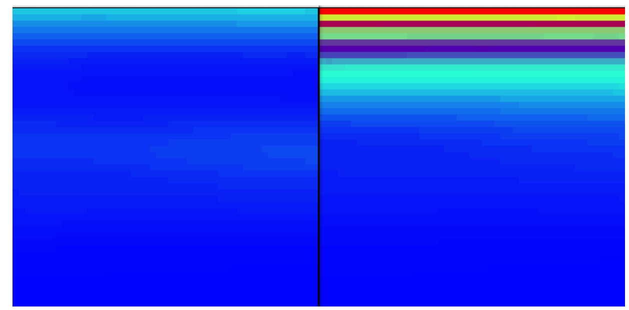
```
&domains
e_vert            = 50,    50,   50
p_top_requested   = 1000,

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

Vertical Interpolation

Vertical cross sections of THICKNESS of each model layer, with 50 vertical levels above the PBL, ptop = 10 hPa.

Uniform layers   Exaggerated Stretching

720-820 m



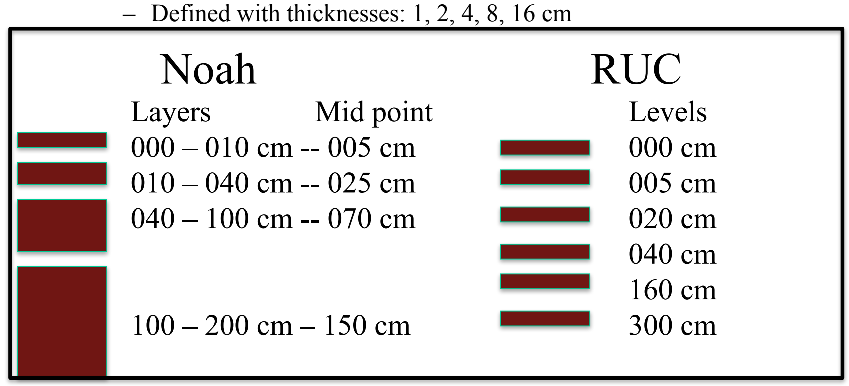
2000 m
1000 m
800 m
575 m

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

Noah

Layers Mid point

000 – 010 cm -- 005 cm
010 – 040 cm -- 025 cm
040 – 100 cm -- 070 cm
100 – 200 cm – 150 cm

RUC

Levels

000 cm
005 cm
020 cm
040 cm
160 cm
300 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

Noah

Layers Mid point

000 – 010 cm -- 005 cm
010 – 040 cm -- 025 cm
040 – 100 cm -- 070 cm
100 – 200 cm – 150 cm

RUC

Levels

000 cm
005 cm
020 cm
040 cm
160 cm
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

Noah

Layers Mid point

000 – 010 cm -- 005 cm
010 – 040 cm -- 025 cm
040 – 100 cm -- 070 cm
100 – 200 cm – 150 cm

RUC

Levels

000 cm
005 cm
020 cm
040 cm
160 cm
300 cm

Real program in a nutshell

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

Real namelist

- The real program and WRF model SHARE a namelist file: **namelist.input**
- Some entries are used by both programs: dimensions, starting date/time, LBC, FDDA
- Many entries are only for one of the programs: real does not care about advection, diffusion, DFI

```
WRFV3
     /test
          /em_real
               /namelist.input
```

Real namelist

- The **namelist.input** file is separated into separate namelist records, usually with related areas

&time_control – dates, time, I/O
&domains – domain grid sizes, REAL options
&physics – land scheme and layers, land cats
&dynamics – base state, GWD
&bdy_control – lateral boundary conditions

Real namelist

```
&time_control
start_year          = 2017, 2008,
start_month         = 05,   01,
start_day           = 30,   01,
start_hour          = 12,   00,
end_year            = 2017, 2008,
end_month           = 05,   01,
end_day             = 30,   01,
end_hour            = 12,   00,
interval_seconds    = 21600
input_from_file     = .t.,  .t.,
io_form_input       = 2
io_form_boundary    = 2
```

Real namelist

```
&time_control
start_year          = 2017, 2008,
start_month         = 05,   01,
start_day           = 30,   01,
start_hour          = 12,   00,
end_year            = 2017, 2008,
end_month           = 05,   01,
end_day             = 30,   01,
end_hour            = 12,   00,
interval_seconds    = 21600
input_from_file     = .t.,  .t.,
io_form_input       = 2
io_form_boundary    = 2
```

START:
Same as metgrid

Real namelist

```
&time_control
start_year          = 2017, 2008,
start_month         = 05,   01,
start_day           = 30,   01,
start_hour          = 12,   00,
end_year            = 2017, 2008,
end_month           = 05,   01,
end_day             = 30,   01,
end_hour            = 12,   00,
interval_seconds    = 21600
input_from_file     = .t.,  .t.,
io_form_input       = 2
io_form_boundary    = 2
```

END:
Same as metgrid

Real namelist

```
&time_control
start_year          = 2017, 2008,
start_month         = 05,   01,
start_day           = 30,   01,
start_hour          = 12,   00,
end_year            = 2017, 2008,
end_month           = 05,   01,
end_day             = 30,   01,
end_hour            = 12,   00,
interval_seconds    = 21600
input_from_file     = .t.,  .t.,
io_form_input       = 2
io_form_boundary    = 2
```

Interval:
Same as metgrid

Real namelist

```
&time_control
start_year          = 2017, 2008,
start_month         = 05,   01,
start_day           = 30,   01,
start_hour          = 12,   00,
end_year            = 2017, 2008,
end_month           = 05,   01,
end_day             = 30,   01,
end_hour            = 12,   00,
interval_seconds    = 21600
input_from_file     = .t.,  .t.,
io_form_input       = 2
io_form_boundary    = 2
```

How many domains from geogrid and metgrid

Real namelist

```
&time_control
start_year          = 2017, 2008,
start_month         = 05,   01,
start_day           = 30,   01,
start_hour          = 12,   00,
end_year            = 2017, 2008,
end_month           = 05,   01,
end_day             = 30,   01,
end_hour            = 12,   00,
interval_seconds    = 21600
input_from_file     = .t.,  .t.,
io_form_input       = 2
io_form_boundary    = 2
```

Leave default = 2
NETCDF

Real namelist

```
&domains
max_dom                    = 1,
e_we                       = 74,     112,
e_sn                       = 61,     97,
e_vert                     = 30,     30,
p_top_requested            = 5000,
num_metgrid_levels         = 27,
num_metgrid_soil_levels    = 4,
dx                         = 30000, 10000,
dy                         = 30000, 10000,
grid_id                    = 1,      2,
parent_id                  = 0,      1,
i_parent_start             = 1,      31,
j_parent_start             = 1,      17,
parent_grid_ratio          = 1,      3,
```

Real namelist

```
&domains
max_dom                    = 1,
e_we                       = 74,     112,
e_sn                       = 61,     97,
e_vert                     = 30,     30,
p_top_requested            = 5000,
num_metgrid_levels         = 27,
num_metgrid_soil_levels    = 4,
dx                         = 30000, 10000,
dy                         = 30000, 10000,
grid_id                    = 1,      2,
parent_id                  = 0,      1,
i_parent_start             = 1,      31,
j_parent_start             = 1,      17,
parent_grid_ratio          = 1,      3,
```

REAL: Total number of domains on INPUT

Real namelist

```
&domains
max_dom                    = 1,
e_we                       = 74,     112,
e_sn                       = 61,     97,
e_vert                     = 30,     30,
p_top_requested            = 5000,
num_metgrid_levels         = 27,
num_metgrid_soil_levels    = 4,
dx                         = 30000, 10000,
dy                         = 30000, 10000,
grid_id                    = 1,      2,
parent_id                  = 0,      1,
i_parent_start             = 1,      31,
j_parent_start             = 1,      17,
parent_grid_ratio          = 1,      3,
```

Domain size in grid cells (u,v,w)

Real namelist

```
&domains
max_dom                    = 1,
e_we                       = 74,     112,
e_sn                       = 61,     97,
e_vert                     = 30,     30,
p_top_requested            = 5000,
num_metgrid_levels         = 27,
num_metgrid_soil_levels    = 4,
dx                         = 30000, 10000,
dy                         = 30000, 10000,
grid_id                    = 1,      2,
parent_id                  = 0,      1,
i_parent_start             = 1,      31,
j_parent_start             = 1,      17,
parent_grid_ratio          = 1,      3,
```

Model lid (Pa) 5000 Pa (20 km), No lower

Real namelist

```
&domains
max_dom                   = 1,
e_we                      = 74,     112,
e_sn                      = 61,     97,
e_vert                    = 30,     30,
p_top_requested           = 5000,
num_metgrid_levels        = 27,
num_metgrid_soil_levels = 4,
dx                        = 30000, 10000,
dy                        = 30000, 10000,
grid_id                   = 1,      2,
parent_id                 = 0,      1,
i_parent_start            = 1,      31,
j_parent_start            = 1,      17,
parent_grid_ratio         = 1,      3,
```

Consistent with dimensions from metgrid

Real namelist

```
&domains
max_dom                   = 1,
e_we                      = 74,     112,
e_sn                      = 61,     97,
e_vert                    = 30,     30,
p_top_requested           = 5000,
num_metgrid_levels        = 27,
num_metgrid_soil_levels = 4,
dx                        = 30000, 10000,
dy                        = 30000, 10000,
grid_id                   = 1,      2,
parent_id                 = 0,      1,
i_parent_start            = 1,      31,
j_parent_start            = 1,      17,
parent_grid_ratio         = 1,      3,
```

Grid distance (m)

dx=dy, except for lat/lon domains

Real namelist

```
&domains
max_dom                   = 1,
e_we                      = 74,     112,
e_sn                      = 61,     97,
e_vert                    = 30,     30,
p_top_requested           = 5000,
num_metgrid_levels        = 27,
num_metgrid_soil_levels = 4,
dx                        = 30000, 10000,
dy                        = 30000, 10000,
grid_id                   = 1,      2,
parent_id                 = 0,      1,
i_parent_start            = 1,      31,
j_parent_start            = 1,      17,
parent_grid_ratio         = 1,      3,
```

Parent/child information, same as metgrid

Real namelist

```
&domains
max_dom                   = 1,
e_we                      = 74,     112,
e_sn                      = 61,     97,
e_vert                    = 30,     30,
p_top_requested           = 5000,
num_metgrid_levels        = 27,
num_metgrid_soil_levels = 4,
dx                        = 30000, 10000,
dy                        = 30000, 10000,
grid_id                   = 1,      2,
parent_id                 = 0,      1,
i_parent_start            = 1,      31,
j_parent_start            = 1,      17,
parent_grid_ratio         = 1,      3,
smooth_cg_topo            = .t.
```

With high topo on CG boundaries, turn this ON

Real namelist

```
&physics
sf_surface_physics  = 2,      2,
num_soil_layers     = 4,
num_land_cat        = 21,
sf_urban_physics    = 0,      0,
```

Real namelist

```
&physics
sf_surface_physics  = 2,      2,
num_soil_layers     = 4,
num_land_cat        = 21,
sf_urban_physics    = 0,      0,
```

Real and WRF have to be consistent with the surface layer scheme due to the dimensions of the soil temp and moisture

Real namelist

```
&physics
sf_surface_physics  = 2,      2,
num_soil_layers     = 4,
num_land_cat        = 21,
sf_urban_physics    = 0,      0,
```

The dimensions of the land categories must match the selected data source from geogrid

Real namelist

```
&physics
sf_surface_physics  = 2,      2,
num_soil_layers     = 4,
num_land_cat        = 21,
sf_urban_physics    = 0,      0,
```

Arrays between real and WRF need to be consistently dimensioned

Real namelist

```
&dynamics
base_temp       = 290.
gwd_opt         = 1,
```

Real namelist

```
&dynamics
base_temp       = 290.
gwd_opt         = 1,
```

Atmospheric temperature (K) at sea level (NOT SST) in the middle of your domain

Must not change between real and WRF

Real namelist

```
&dynamics
base_temp       = 290.
gwd_opt         = 1,
```

GWD allocates space which must be consistent between real and WRF

Real namelist

```
&bdy_control
spec_bdy_width   = 5,
spec_zone        = 1,
relax_zone       = 4,
specified        = .t., .f.,
nested           = .f., .t.,
```

Real namelist

```
&bdy_control
spec_bdy_width  = 5,
spec_zone       = 1,
relax_zone      = 4,
specified       = .t., .f.,
nested          = .f., .t.,
```

spec_bdy_width = spec_zone + relax_zone

spec_zone = 1

Real namelist

```
&bdy_control
spec_bdy_width  = 5,
spec_zone       = 1,
relax_zone      = 4,
specified       = .t., .f.,
nested          = .f., .t.,


spec_bdy_width  = 10,
relax_zone      = 9,
spec_exp        = 0.33
```

Can choose larger relaxation zone since domain is big

Can choose exponential decay

Real namelist

```
&bdy_control
spec_bdy_width  = 5,
spec_zone       = 1,
relax_zone      = 4,
specified       = .t., .f.,
nested          = .f., .t.,
```

REAL:
d01 always specified=T

All other domains have specified=F

Real namelist

```
&bdy_control
spec_bdy_width  = 5,
spec_zone       = 1,
relax_zone      = 4,
specified       = .t., .f.,
nested          = .f., .t.,
```

d01 always nested=F

All other domains have nested=T

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 refinement**, 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,
```

The 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
========================================
```

The 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 ;
```

The 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**

The 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 (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.
/
```

Options

- **Time varying fields** for longer simulations are available from the technique set up for "SST Update"
- A new field will be automatically added to the input file to the WRF model (provided by the real program) with a few changes to the Registry file (**`Registry.EM_COMMON`**), specifying **stream 4**

```
state  real my_new_field  ij misc 1  - \
i024rhdu "MY_NEW_FIELD" \
"SOME DESCRIPTION" "SOME UNITS"
```

Options

- Information for **using time varying data** is specified at run-time in the namelist file

```
&time_control
 auxinput4_inname    = "wrflowinp_d<domain>"
 auxinput4_interval = 360
 io_form_auxinput4  = 2

&physics
 sst_update          = 1
```

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

Real program in a nutshell: PART 2

- Access to everything — The Derived Data Type: grid
- Eta levels
- Metgrid flags — Example: grid%sst
- Adding a variable for vertical interpolation
- Vertical interpolation
- Tracers
- Trajectories
- Options

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

Completely user defined

May be different per domain

Be careful of the thicknesses

Tightly coupled with the model lid

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

The metgrid program provides flags for some internal communication real to metgrid

These flags are defined inside the METGRID.TBL file (for WPS) and in the file share/module_optional_input.F (real)

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

Requires new code inside real

Examples are easily available

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

Always in dry pressure

Input vertical coordinate neutral

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

Simple way to initialize passive scalars

Users should provide info for which tracers in the Registry, and select the accompanying option in the namelist

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

A simple (i,j,k) initialization for the starting locations of trajectory points is available

Choose the number of trajectory points

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

Users may smooth the outer rows and columns so that the topography on the coarse grid and the external model are consistent

Users may add variables to streams easily, an example is that the SST update option could have a new field included (for example, soil moisture)

Running the WPS

Michael Duda

Running the WRF Preprocessing System

Michael Duda

*NCAR is sponsored by the National Science Foundation

Overview

- How to run through the WPS for a single-domain case
  - 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 = 1
/
```

```
&geogrid
  map_proj   = 'lambert'
  truelat1   = 45.0
  truelat2   = 30.0
  stand_lon  = -105.25
  ref_lat    = 40.0
  ref_lon    = -105.25
  e_we       =  220
  e_sn       =  175
  dx         = 15000
  dy         = 15000
  geog_data_res = 'default'
  geog_data_path = '/data/static/geog/'
/
```


Running geogrid

STEP 1: Edit `namelist.wps`

```
&share
  wrf_core = 'ARW'
  max_dom = 1
/
```

wrf_core: Which WRF core? For ARW, set to 'ARW'. For NMM, set to 'NMM'

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

See p. 3-8 and 3-37

Running geogrid

STEP 1: Edit `namelist.wps`

```
&geogrid
   ...
   map_proj    = 'lambert'
   truelat1    = 45.0
   truelat2    = 30.0
   stand_lon   = -105.25
   ...
/
```

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

See p. 3-9 and 3-40

Running geogrid

STEP 1: Edit `namelist.wps`

```
&geogrid
   ...
   ref_lat          = 40.0
   ref_lon          = -105.25

   e_we             = 220
   e_sn             = 175
   dx               = 15000
   dy               = 15000

   geog_data_res    = 'default'
   geog_data_path = '/data/static/geog/'

   ...
/
```

Domain location: Where on Earth is the center of the domain?

Domain size: 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? Where to find data on the filesystem? *(See "Extra slides"...)*

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

Running geogrid

STEP 2: Run geogrid.exe

```
Parsed 11 entries in GEOGRID.TBL
Processing domain 1 of 1
  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

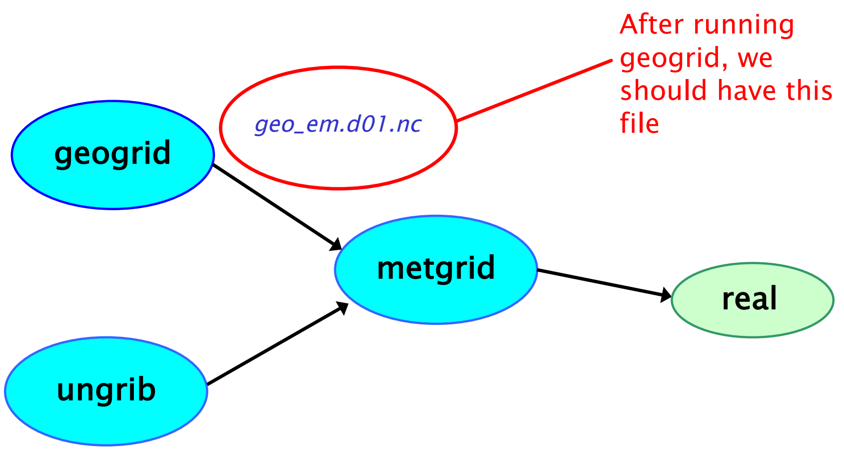
STEP 3: Check that geogrid ran successfully

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".

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 = 1
   start_date  = '2006-04-01_00:00:00'
   end_date    = '2006-04-01_12:00:00'
   interval_seconds = 21600
/

&ungrib
   prefix = 'GFS'
/
```


Running ungrib

STEP 1: Edit `namelist.wps`

```
&share
   wrf_core = 'ARW'
   max_dom = 1

   start_date  = '2006-04-01_00:00:00'
   end_date    = '2006-04-01_12:00:00'

   interval_seconds = 21600

/
```

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
   prefix = 'GFS'
/
```

Intermediate file names: Gives prefix for intermediate files.
Prefix can include a path.
E.g., 'XYZ' 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

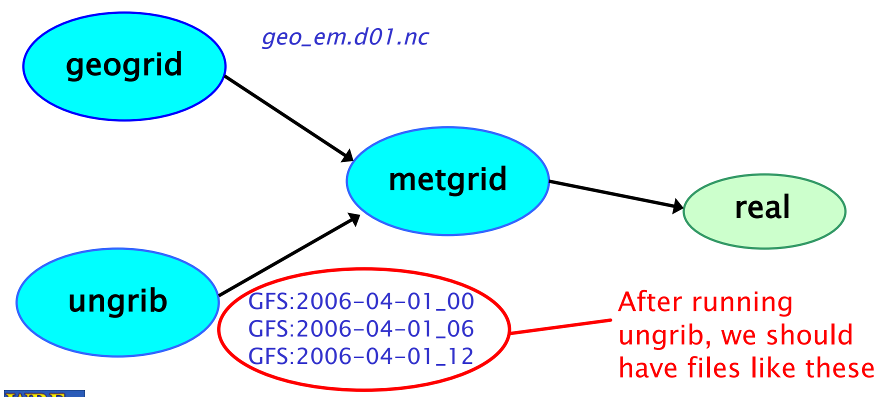
If ungrib ran successfully, this message should be printed:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!  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 = 1
  start_date  = '2006-04-01_00:00:00'
  end_date    = '2006-04-01_12:00:00'
  interval_seconds = 21600
/
```

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


Running metgrid

STEP 1: Edit `namelist.wps`

```
&share
  wrf_core = 'ARW'
  max_dom = 1

  start_date  = '2006-04-01_00:00:00'
  end_date    = '2006-04-01_12:00:00'

  interval_seconds = 21600
/
```

Data time range: Time range to process.

Interval between intermediate files created by ungrib

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'
/
```

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

See p. 3-17, and 3-41

Running metgrid

STEP 2: Run metgrid.exe

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

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

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

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


Running metgrid

STEP 3: Check that metgrid ran successfully

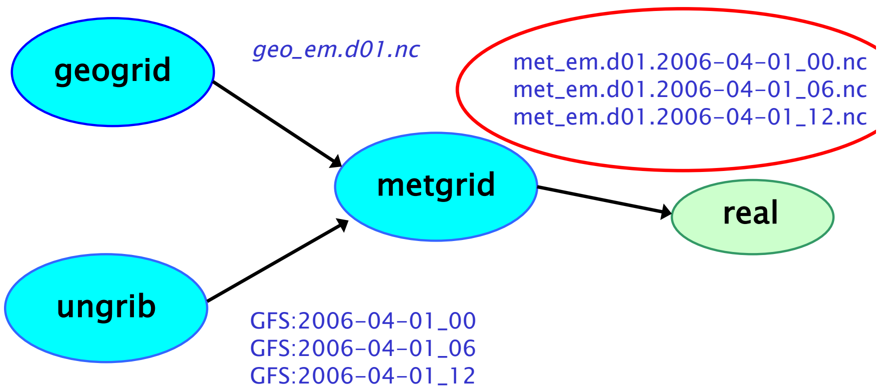
If metgrid ran successfully, this message should be printed:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!  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

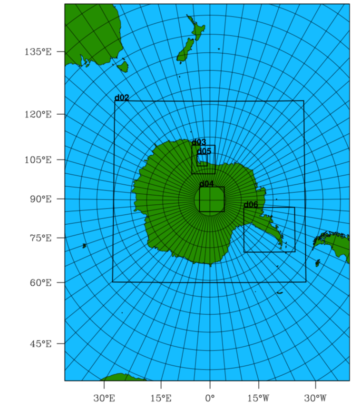
- Besides geogrid, ungrib, and metgrid, some simple utility programs are distributed with WPS:
  - For checking contents of intermediate format files
  - For listing contents of GRIB1 & GRIB2 files
  - To assist in locating domains
  - For computing 3d pressure field for ECMWF data
- Some programs use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

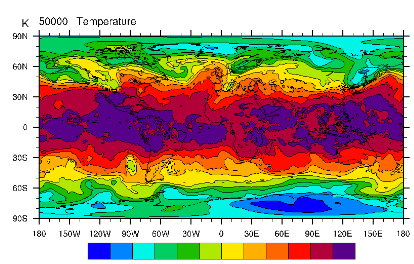
The rd_intermediate lists information about the fields found in an intermediate-format file

```
================================================
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
================================================
```

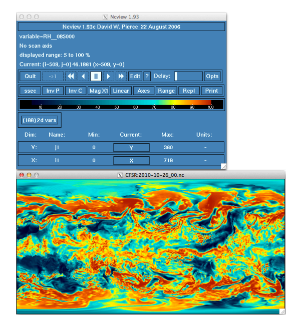

Utility: int2nc + plotfmt_nc.ncl

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

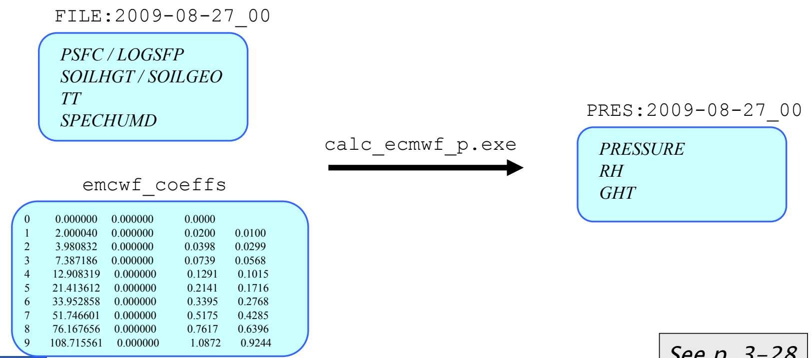
Utility: g1print and g2print

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



0	0.000000	0.000000	0.0000	
1	2.000040	0.000000	0.0200	0.0100
2	3.980832	0.000000	0.0398	0.0299
3	7.387186	0.000000	0.0739	0.0568
4	12.908319	0.000000	0.1291	0.1015
5	21.413612	0.000000	0.2141	0.1716
6	33.952858	0.000000	0.3395	0.2768
7	51.746601	0.000000	0.5175	0.4285
8	76.167656	0.000000	0.7617	0.6396
9	108.715561	0.000000	1.0872	0.9244

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?

Extra slides

Choosing Static Datasets

WPS v3.9 supports several land cover datasets and 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

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.

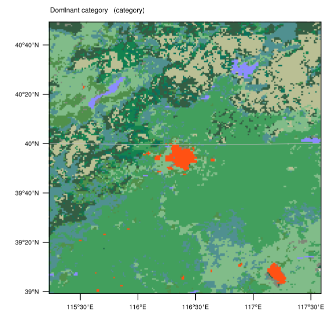
E.g.,

```
geog_data_res = 'nlcd2011_9s+default'
```

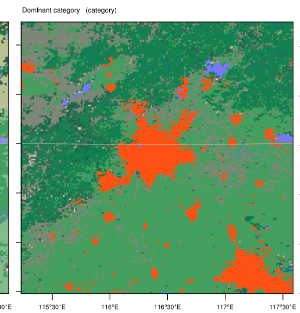
to use NLCD 2011 9-arc-second land cover, and default resolution for other static fields.

Global Land Cover Datasets

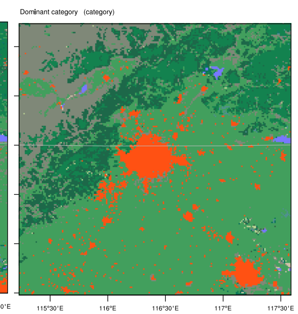
Consider an example 1-km domain centered over Beijing:



USGS 30-arc-second resolution, from ~1993 data; select using `'usgs_30s'`



MODIS 30-arc-second resolution, from 2001(?) data; the MODIS data are used by default



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.

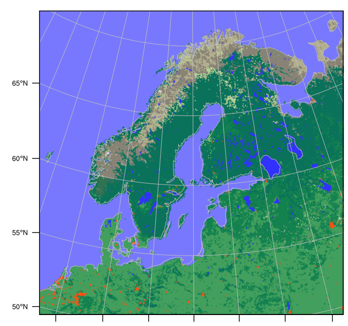
<u>MODIS 30-arc-second</u>:

- Selected using `'modis_lakes'`

<u>USGS 30-arc-second</u>:

- 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).

<u>NLCD 2011 9-arc-second</u>:

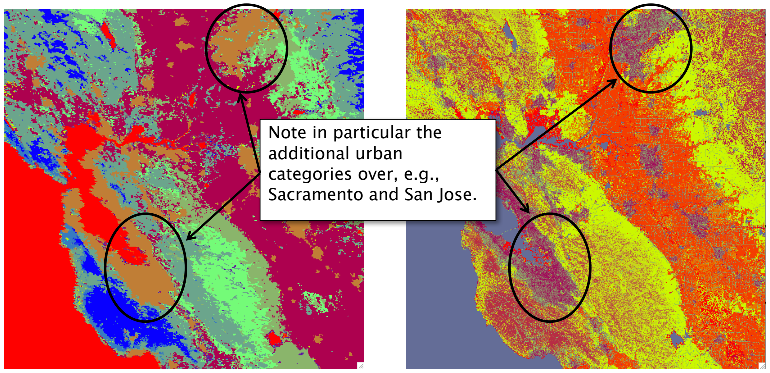
- Selected using `'nlcd2011_9s'`

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)

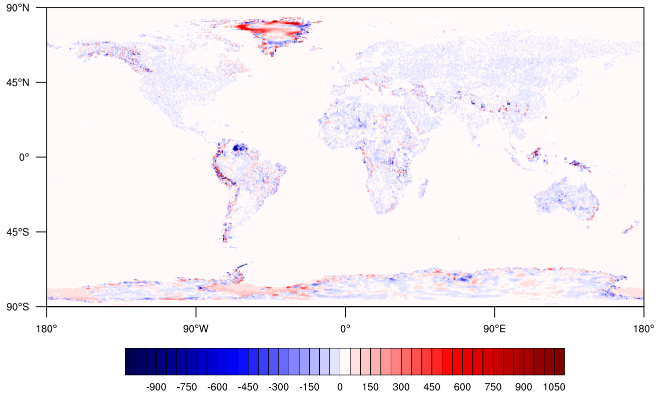
For the WRF domains over the Continental U.S., one can use high-resolution land cover from the National Land Cover Database (NLCD).



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

WPS v3.8 and newer replace the GTOPO30 dataset with 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

WRF:
Set-up and Run

Wei Wang

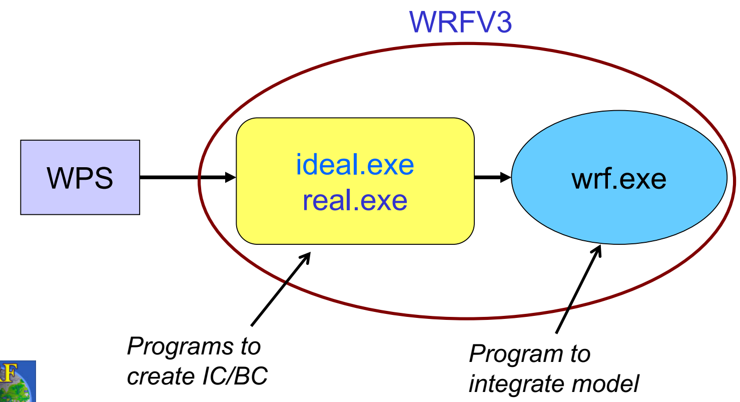
Set Up and Run WRF

(*real* and *Ideal* data)

Wei Wang

January 2018

WRF System Flowchart



Outline

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

Before You Run ..

- Make sure appropriate executables are created in **`WRFV3/main/`** directory:
  - **`ideal.exe`** – *executable to create idealized IC*
  - **`real.exe`** – *executable to create IC/BC*
  - **`wrf.exe`** – *executable for model integration*
  - **`ndown.exe`** – *utility*
  - **`tc.exe`** – *utility routine for TC bogusing*
- If you are working with real data, be sure that files for ***a few time periods*** from WPS are correctly generated:
  - **`met_em.d01.*`**

WRF test case directories

You have these choices in **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**, **em_heldsuarez** } *3d ideal*
- **em_hill2d_x**, **em_squall2d_x**, **em_squall2d_y**, **em_grav2d_x**, **em_seabreeze2d_x** } *2d ideal*
- **em_scm_xy** } *1d ideal*

(3d ideal, 2d ideal, 1d ideal } *ideal.exe*)

Steps to Run

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

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
aerosol.formatted
aerosol_lat.formatted
aerosol_lon.formatted
aerosol_plev.formatted

} *These are model physics data files: they are used to either initialize physics variables, or make physics computation faster*
** Some of these files are text files, hence editable*

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

…. (a total of 60 files)

WRFV3/run directory after compile

An example after em_real case compile

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

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 doing 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
ozone.formatted -> ../../run/ozone.formatted
ozone_lat.formatted -> ../../run/ozone_lat.formatted
ozone_plev.formatted -> ../../run/ozone_plev.formatted
…
namelist.input               - editing required
real.exe -> ../../main/real.exe
wrf.exe -> ../../main/wrf.exe
ndown.exe -> ../../main/ndown.exe
```

…. (many more)

Running a Real-data Case

- One must successfully run WPS to prepare data required, and create **`met_em.*`** files for *multiple time periods* for initial and boundary conditions

- Move or link WPS/metgrid output files to the run directory:

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

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` for a MPI job

  where ***N*** is the number of processors requested.

Running a Real-data Case

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

`wrfinput_d01` ← *Single time level data at model's start time*

`wrfbdy_d01` ← *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 give you, for a 24 hour period, 6 hourly data interval:

.. a bunch of prints and at the end:

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

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

Running a Real-data Case

- Run the model executable by typing:

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

  or

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

- Successfully running the model will a create model <u>*history*</u> file:

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

  Based on start date set in namelist

  and a <u>*restart*</u> file if **`restart_interval`** is set to a time within the range of the forecast time:

  `wrfrst_d01_2005-08-28_12:00:00`

  Exact time at a restart

Running a Real Data Case

wrfout_d01_2005-08-28_00:00:00

Based on start date set in namelist

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


Running an Idealized Case

Running an *Idealized* Case

- An idealized case refers to data in the initial condition file (no need to run WPS)
- If you have compiled an ideal case, you should have:
  - **`ideal.exe`** – program to create idealized initial condition
  - **`wrf.exe`** - model executable
- These executables are linked to:

  `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 relevant physics data files to the current directory:

`./run_me_first.csh`

Running an *Idealized* Case

Then run the ideal initialization program:

./ideal.exe

The input to this program is typically a sounding file (file named ***input_sounding***), or a pre-defined 2D input (e.g. ***input_jet*** in **em_b_wave** case).

Running **ideal.exe** *only* creates WRF initial condition file: **wrfinput_d01**

Running an *Idealized* Case

Note that wrfbdy file is not needed for idealized cases.

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

```
periodic_x     = .true.,.false.,.false.,
symmetric_xs   = .false.,.false.,.false.,
symmetric_xe   = .false.,.false.,.false.,
open_xs        = .false.,.false.,.false.,
open_xe        = .false.,.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 (3D cases only)
- Successful running of the model executable will create a model history file called **wrfout_d01_*<date>***

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

  Based on start date set in namelist

Running an *Idealized* Case

wrfout_d01_0001-01-01_00:00:00

Based on start date set in namelist

```
start_year     = 0001, 0001, 0001,
start_month    = 01,   01,   01,
start_day      = 01,   01,   01,
start_hour     = 00,   00,   00,
start_minute   = 00,   00,   00,
start_second   = 00,   00,   00,
end_year       = 0001, 0001, 0001,
end_month      = 01,   01,   01,
end_day        = 01,   01,   01,
end_hour       = 00,   00,   00,
end_minute     = 120,  120,  120,
end_second     = 00,   00,   00,
```

Running an *Idealized* Case

- Edit **`namelist.input`** file to change options.
- For your own case, you may provide a different sounding.
- You may also edit **`dyn_em/module_initialize_<case>.F`** to change other aspects of the initialization (*more on Thur.*)

Note:

- For 2D cases and baroclinic wave case, **`ideal.exe`** must be run serially
- For all 2D cases, **`wrf.exe`** must be run serially or with SMP
- For the 1D case, compile and run serially

Basic namelist Options

What is a namelist?

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

  `&namelist-record` - start

  `/` - end
- As a general rule:
  - Multiple columns: domain dependent
  - Single column: value valid for all domains
- A namelist file may contain a number of records

namelist record **`&time_control`**

```
run_days              = 0,
run_hours             = 24,
run_minutes           = 0,
run_seconds           = 0,
start_year            = 2000, 2000, 2000,
start_month           = 01,   01,   01,
start_day             = 24,   24,   24,
start_hour            = 12,   12,   12,
start_minute          = 00,   00,   00,
start_second          = 00,   00,   00,
end_year              = 2000, 2000, 2000,
end_month             = 01,   01,   01,
end_day               = 25,   25,   25,
end_hour              = 12,   12,   12,
end_minute            = 00,   00,   00,
end_second            = 00,   00,   00,
interval_seconds      = 21600
history_interval      = 180,  60,   60,
frames_per_outfile    = 1000, 1000, 1000,
restart_interval      = 360,
restart               = .true.,
```

domain 1 option

for nests

Notes on &time_control

- *run_** time variables:
  - Model simulation length: *wrf.exe* and domain 1 only
- *start_** and *end_** time variables:
  - Program *real* will use WPS output between these times to produce lateral (and lower) boundary file
  - They can also be used to specify the start and end of simulation times for the coarse grid if *run_** variables are not set (or set to 0)

Notes on &time_control

- *interval_seconds*:
  - Time interval between WPS output times, and lateral BC (and lower BC) update frequency
- *history_interval*:
  - Time interval in minutes when a history output is written (*note* output is instantaneous)
  - If the **time_step** cannot be evenly divided by **history_interval**, then nearest time step output is used
  - The time stamp in a history file name is the time when the history file is first written, and multiple time periods may be written in one file. e.g. a history file for domain 1 that is first written for 1200 UTC Jan 24 2000 is

    wrfout_d01_2000-01-24_12:00:00

Notes on &time_control

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

    wrfrst_d01_2000-01-25_00:00:00
- *restart:*
  - whether this is a restart run

Notes on &time_control

Example 1: all output times are in a single file

```
history_interval   = 180,  60,   60,
frames_per_outfile = 1000, 1000, 1000,
```

wrfout_d01_2000-01-24_12:00:00

Example 2: each output file only contains a single time

```
history_interval   = 180, 60,  60,
frames_per_outfile = 1,   1,   1,
```

wrfout_d01_2000-01-24_12:00:00
wrfout_d01_2000-01-24_15:00:00
wrfout_d01_2000-01-24_18:00:00

Notes on *restart*

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

&time_control

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

IO format options:
= 1, binary
= 2, netcdf (most common)
= 4, PHDF5
= 5, Grib 1
=10, Grib 2
=11, pnetCDF

For large files:
io_form_restart = 102 :
write output in patch sizes: fast for large grid and useful for restart file

namelist record &domains

```
time_step                = 180
time_step_fract_num      = 0,
time_step_fract_den      = 1,
max_dom                  = 1,
e_we                     = 74,   112,   94,
e_sn                     = 61,   97,    91,
e_vert                   = 28,   28,    28,
num_metgrid_levels       = 32,
num_metgrid_soil_levels  = 4
dx                       = 30000, 10000, 3333,
dy                       = 30000, 10000, 3333,
eta_levels               = 1.0,0.996,0.99,0.98,… 0.0
p_top_requested          = 5000,
```

nest options

Notes on &domains

- *time_step, time_step_fract_num, time_step_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 ore more levels with 5000 Pa model top to limit vertical grid distance < 1 km. Use more vertical levels when decreasing horizontal grid sizes.

namelist record **&bdy_control**

typical *optional*

```
spec_bdy_width     = 5, (10)
spec_zone          = 1, (1)      <- do not change
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

Other namelists

&physics:
- Model physics options

&dynamics:
- Damping, diffusion options
- Advection options

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 p5-36 to 5-38 of the ARW User's Guide:
  - 2 or 4 km microphysics-only runs
  - 20 – 30 km, 2 – 3 day runs
  - Antarctic region
  - Tropical storm forecasting
  - Regional climate
  - Try physics suites (since V3.9)

Where do I start?

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

To run a job in a different directory..

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

Check Output

Output After a Model Run

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

Output from a multi-processor run

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

mpirun -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
WRF V3.9 MODEL
```

- Time taken to compute one model step:

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

- Time taken to write history and restart file:

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

- Any model error prints:

```
5 points exceeded cfl=2 in domain 1 at time 4.200000 MAX AT i,j,k: 123 48 3
     cfl,w,d(eta)= 4.165821
```

→ An indication the model has become numerically unstable

Simple Trouble Shooting

Often-seen runtime problems

– `module_configure: initial_config: error reading namelist: &dynamics`

> Typos or erroneous namelist variables exist in namelist record *&dynamics* in *namelist.input* file

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

> Grid dimensions in error

Often-seen runtime problems

– `Segmentation fault (core dumped)`

> Often typing '**`unlimit`**' or '**`ulimit -s unlimited`**' or equivalent can help when this happens quickly in a run, and on a small computer

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

```
121 points exceeded cfl=2 in domain 1 at time
4.200000 MAX AT i,j,k: 123 48 3 cfl,w,d(eta)=
4.165821
```

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

References

- Information on compiling and running WRF, and a more extensive list of namelist options and their definition / explanations can be found in the User's Guide, Chapter 5
- Also see nesting talk and demonstration tomorrow.

WRF and WPS: Compilation Process

Kelly Werner

WRF & WPS: Compilation Process

Kelly Werner

NCAR/MMM

January 2018

Installing Steps

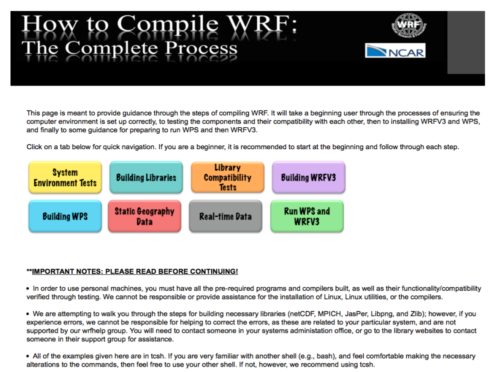
- *Check system requirements*
- Installing libraries
- Download source data
- Compile WRFV3
- Compile WPS
- Download initial/BC datasets

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

Check System Requirements

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



How to Compile WRF:
The Complete Process

This page is meant to provide guidance through the steps of compiling WRF. It will take a beginning user through the processes of ensuring the computer environment is set up correctly, to testing the components and their compatibility with each other, then to installing WRFV3 and WPS, and finally to some guidance for preparing to run WPS and then WRFV3.

Click on a tab below for quick navigation. If you are a beginner, it is recommended to start at the beginning and follow through each step.

System Environment Tests | Building Libraries | Library Compatibility Tests | Building WRFV3 | Building WPS | Static Geography Data | Real-time Data | Run WPS and WRFV3

****IMPORTANT NOTES: PLEASE READ BEFORE CONTINUING!**

- In order to use personal machines, you must have all the pre-required programs and compilers built, as well as their functionality/compatibility verified through testing. We cannot be responsible or provide assistance for the installation of Linux, Linux utilities, or the compilers.
- We are attempting to walk you through the steps for building necessary libraries (netCDF, MPICH, JasPer, Libpng, and Zlib); however, if you experience errors, we cannot be responsible for helping to correct the errors, as these are related to your particular system, and are not supported by our wrfhelp group. You will need to contact someone in your systems administation office, or go to the library websites to contact someone in their support group for assistance.
- All of the examples given here are in tcsh. If you are very familiar with another shell (e.g., bash), and feel comfortable making the necessary alterations to the commands, then feel free to use your other shell. If not, however, we recommend using tcsh.

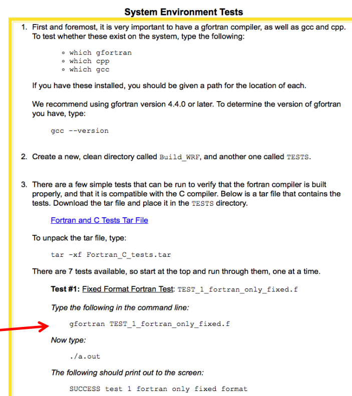
Check System Requirements

- 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)
  - netCDF Version 3 or 4 are acceptable
  - If using netCDF4 capabilities
    http://www2.mmm.ucar.edu/wrf/users/building_netcdf4.html
- Optional libraries for GRIB2 meteorological 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

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 ..
```


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 ..
```


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 ..
```

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 ..
```

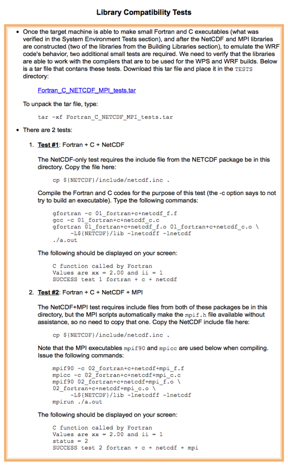

Installing Libraries: JasPer

- *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 contans 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

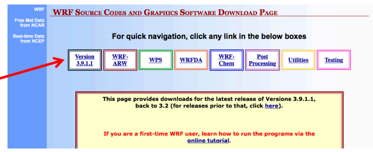
- Download WRF & WPS source code from:

  http://www2.mmm.ucar.edu/wrf/users/download/get_source.html

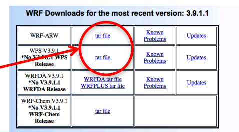
  – Click 'New User,' register and download, or

  – Click 'Returning User,' enter your email, and download

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



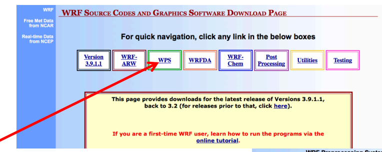
Step 2: Click on tar files to download



Download Static Geographical Data

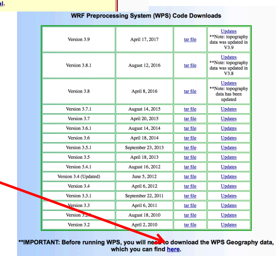
- From the WRF Download page:

  http://www2.mmm.ucar.edu/wrf/users/download/get_sources.html



Step 1: Click 'WPS' box

Step 2: Click 'here' to get geography data



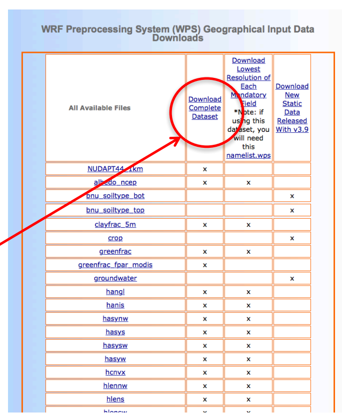
Download Static Geographical Data

- 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 Options for WRFV3

Debugging Options

- `./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, GNU, and PGI

Large File Support

- `setenv WRFIO_NCD_LARGE_FILE_SUPPORT 1`
  - > 2GB
  - Before configuring
  - Built-in since V3.9

Hybrid Coordinate Option

- `./configure -hyb`

Parallel Compile Option for WRFV3

- To build WRF in parallel
  - *setenv J "-j 2"*

# 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

* This test done with GNU compiler

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 environment variable PNETCDF identically to how NETCDF is set (point to the PNETCDF top-level directory)

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)

3d Ideal:
- em_quarter_ss
- em_b_wave
- em_les
- em_heldsuarez
- em_tropical_cyclone
- em_convrad

2d Ideal:
- em_hill2d_x
- em_squall2d_x
- em_squall2d_y
- em_grav2d_x
- em_seabreeze2d_x

em_scm_xy (1d ideal)

Compilation should take ~30 mins

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.

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'
  - Reconfigure: If you need to make changes to the configure.wrf file, do this after issuing ./configure, and then save the edited file.
  - Recompile
- Contact wrfhelp@ucar.edu

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

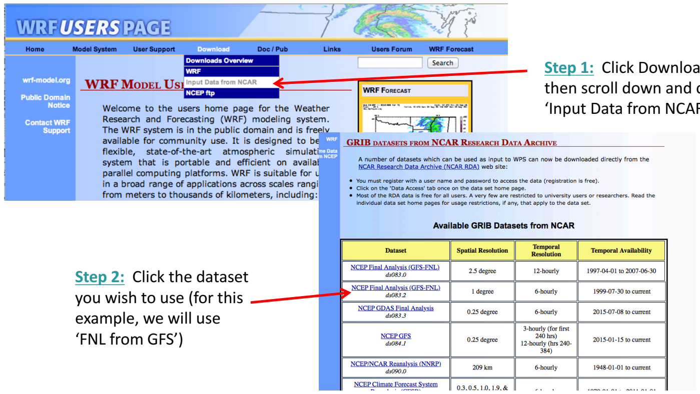
- The './clean –a' command is something that should be used when you have made corrections to your configure.wrf file, configure.wps file, or any changes to the registry. If you have made any of these changes, or if you plan to recompile your code from scratch, you must issue a 'clean –a' before recompiling.
- 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'

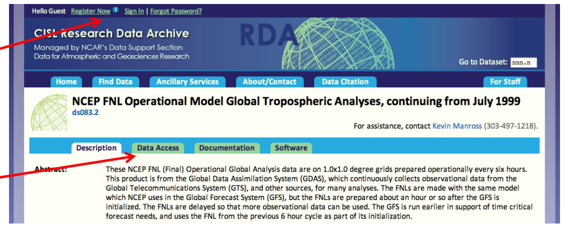
Step 2: Click the dataset you wish to use (for this example, we will use 'FNL from GFS')

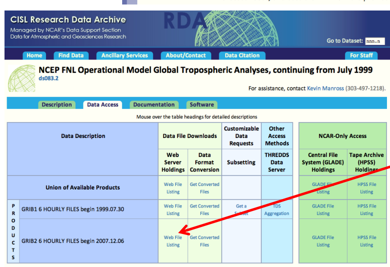
*Note: The NOMADS site has several types of useful data: http://nomads.ncdc.noaa.gov

Download Datasets (continued)

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

Step 4: Click 'Data Access'

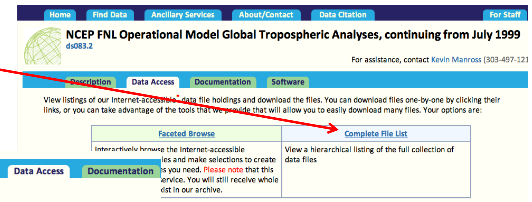


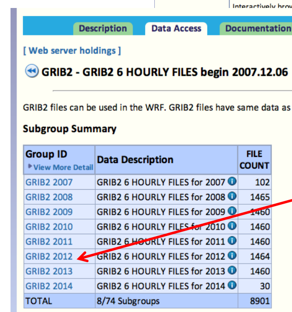


Step 5: Click 'Web File Listing' for the span of years you need

Download Datasets (continued)

Step 6: Click 'Complete File List'

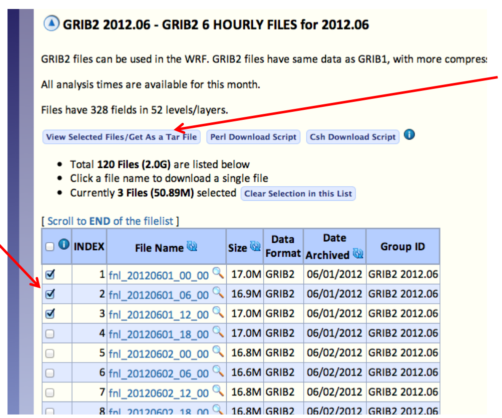




Step 7: Click the year you need. After this, You will click the month you need (not shown)

Download Datasets (continued)

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



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?

Nesting in WRF

Kelly Werner

NESTING IN WRF

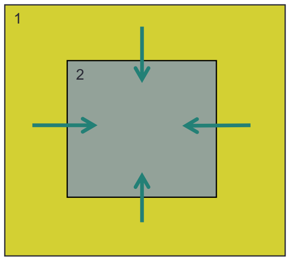
Kelly Werner & Wei Wang
January 2018

What is a nest?

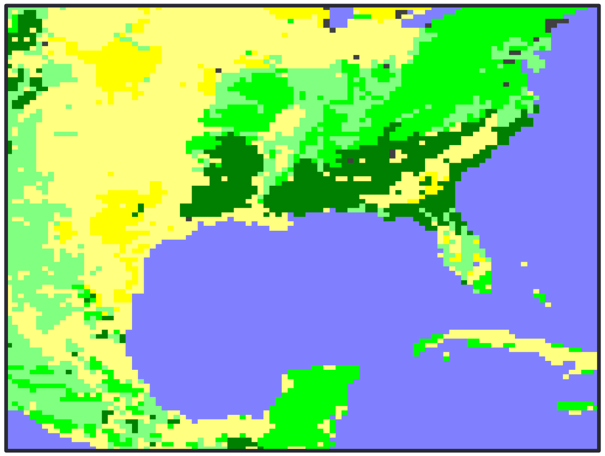
- A *finer-resolution* domain embedded in a coarser resolution domain, and run together with the coarser resolution domain
- Enables running at a higher-resolution without:
  - Uniformly high-resolution over a large domain – VERY expensive
  - High resolution for a very small domain, with mismatched time and spatial lateral boundary conditions

What is a nest?

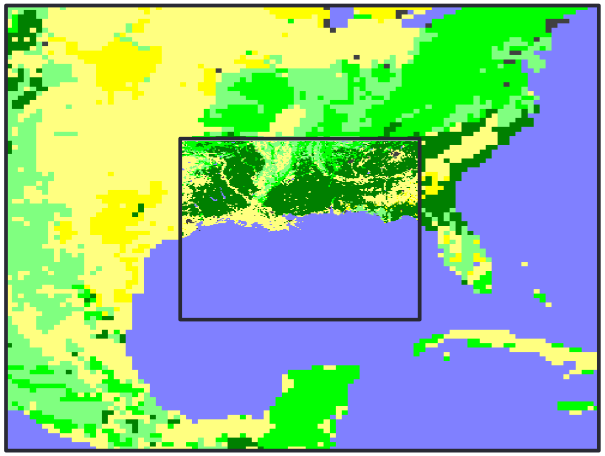
- Covers a portion of the parent domain, and is fully contained by the parent domain
- Driven along its lateral boundaries by the parent domain
- May feedback the computed values back to the parent domain



When Should I Use Nests?



When Should I Use Nests?



When Should I Use Nests?

- Need to simulate localized phenomena: convection, topography, landuse-forced, etc.
  - What resolution is necessary to resolve what you are interested in?
  - Input data resolution is too coarse by more than a factor of 5-10x
  - Would like to provide better boundary conditions for the area of interest
    - BC's for external sources are typically 3-6 hours and do not have tendencies for all predicted fields
  - Computing resources not available for uniform coverage

Types of Nesting

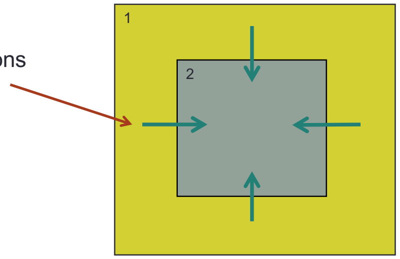
- Using a single input domain (met_em.d01*)
  - No met_em.d02* files are used
  - All fields are interpolated from the model coarse grid
  - Only recommended if nest is over the ocean
- Using multiple input domains
  - Each domain contains full input data files (including topography, landuse, etc.)
- Specified move
  - Must specify every move
  - Can use, but tedious to set-up
- Automatic move
  - Build WRF with "3=vortex following"
  - Only for tropical cyclone tracking
  - Expensive for single large nest
- ndown.exe
  - Use coarser WRF model output to drive finer resolution domains (i.e. 'downscaling')
  - If you have run a long coarse domain simulation (years) and later decide you want to have a nest with higher resolution.

Types of Nesting

One-way/two-way nesting

- Determined by the namelist parameter "feedback"
  - **feedback = 0 (turned off/one-way)**

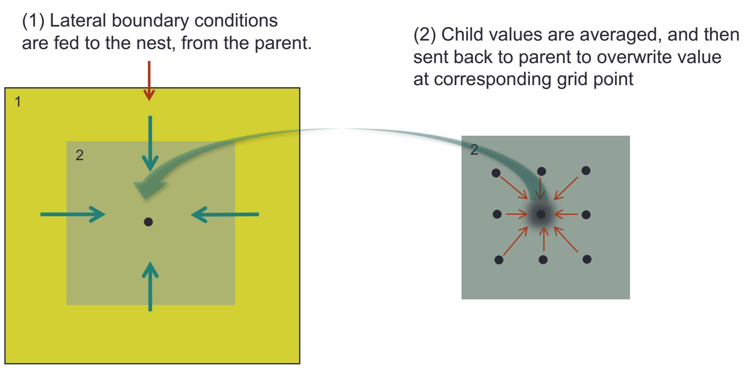
Lateral boundary conditions are fed to the nest, from the parent.



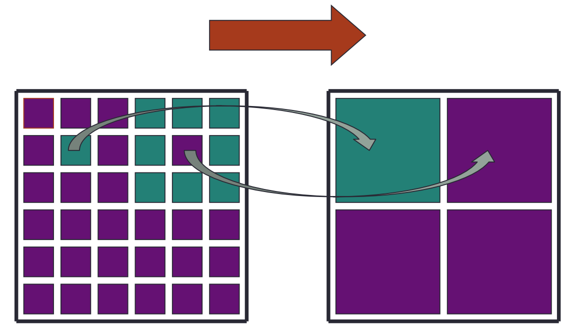
Types of Nesting

One-way/**two-way nesting**

- Determined by the namelist parameter “feedback”
  - **feedback = 1 (turned on/two-way)**

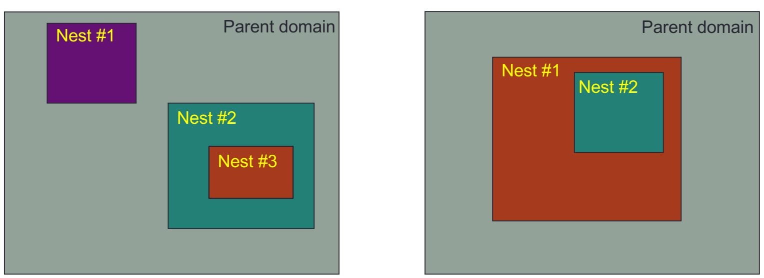


Masked Feedback

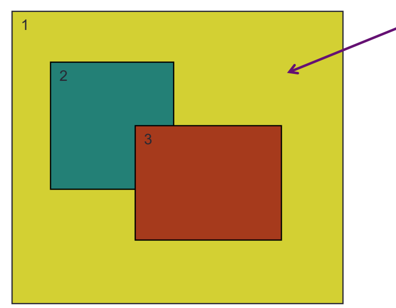


Single grid value feedback for categorical and masked data

Nests that are OK

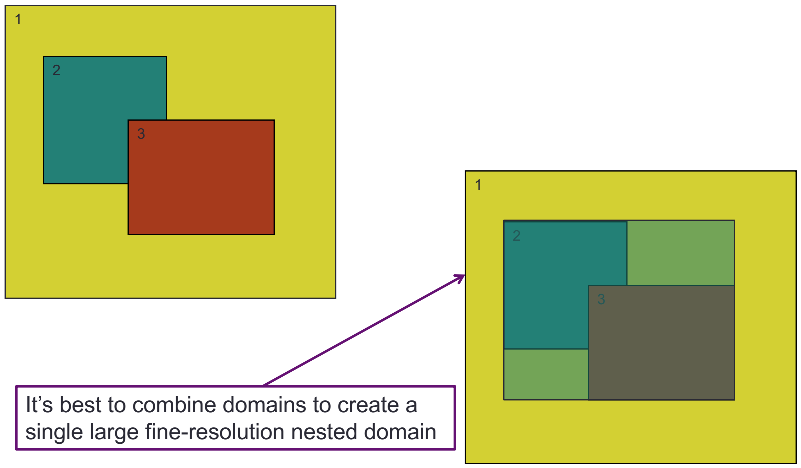


Nests that are NOT OK

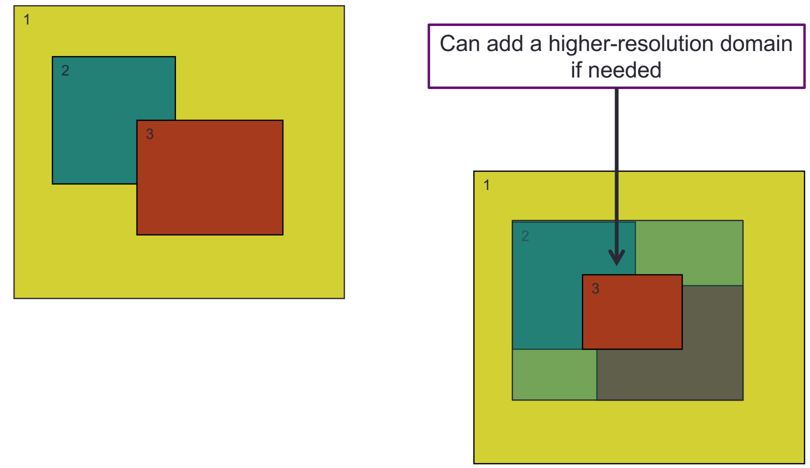


Child domains *may not* have overlapping points in the parent domain (possible if Feedback is off).

Nests that are NOT OK



Nests that are NOT OK



Nesting Set-up and Run

Compiling for Nesting (WRF)

```
------------------------------------------------------------------------
Please select from among the following Darwin ARCH options:

  1. (serial)   2. (smpar)   3. (dmpar)   4. (dm+sm)   PGI (pgf90/pgcc)
  5. (serial)   6. (smpar)   7. (dmpar)   8. (dm+sm)   INTEL (ifort/icc)
  9. (serial)  10. (smpar)  11. (dmpar)  12. (dm+sm)   INTEL (ifort/clang)
 13. (serial)               14. (dmpar)                GNU (g95/gcc)
 15. (serial)  16. (smpar)  17. (dmpar)  18. (dm+sm)   GNU (gfortran/gcc)
 19. (serial)  20. (smpar)  21. (dmpar)  22. (dm+sm)   GNU (gfortran/clang)
 23. (serial)               24. (dmpar)                IBM (xlf90_r/cc)
 25. (serial)  26. (smpar)  27. (dmpar)  28. (dm+sm)   PGI (pgf90/pgcc): -f90=pgf90

Enter selection [1-28] : 9
------------------------------------------------------------------------
Compile for nesting? (0=no nesting, 1=basic, 2=preset moves, 3=vortex following) [default 0]:
```

Compile with nesting option (1=basic)

*Note: Unless compiling for a moving nest, or 2D idealized case, there's no reason to not always choose "basic." It takes no longer to build.

namelist.wps - WPS

namelist.*wps* set-up: *&share*

To edit the namelist.wps file, make sure you are in the WPS/ directory

```
&share
 wrf_core = 'ARW',
 max_dom = 2,
 start_date = '2012-01-27_00:00:00',' 2012-01-27_00:00:00'
 end_date = '2012-01-28_00:00:00', 2012-01-27_00:00:00'
 interval_seconds = 21600
 io_form_geogrid = 2,
/
```

real.exe program only requires initial time for fine domain (unless doing nudging or SST-update in the nest)

Make sure to edit start/end dates for all domains!

namelist.*wps* set-up: *&geogrid*

```
&geogrid

 parent_id         =  1,       1,
 parent_grid_ratio =  1,       3,
 i_parent_start    =  1,      70,
 j_parent_start    =  1,      67,

 e_we              =  175,   181,
 e_sn              =  145,   181,
 geog_data_res     =  'default', 'default',

 dx                = 30000,
 dy                = 30000,
 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/'
/
```

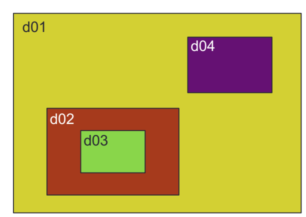
Used for nesting purposes
- What is the grid ratio for each nest?
- Where is it located inside its parent?
- parent_grid_ratio: integer ratio required

Domain sizes: How many grid points does each domain have?

namelist.*wps* set-up: *&geogrid*

```
&geogrid

 parent_id         =  1,       1,
 parent_grid_ratio =  1,       3,
 i_parent_start    =  1,      70,
 j_parent_start    =  1,      67,

 e_we              =  175,   181,
 e_sn              =  145,   181,
 geog_data_res     =  'default', 'default',

 dx                = 30000,
 dy                = 30000,
 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/'
/
```

parent_id:
The domain # of the nest's parent



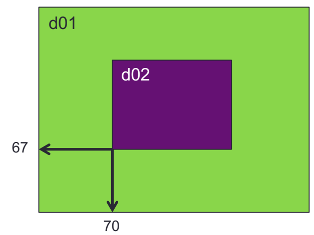
parent_id = 1, 1, 2, 1

namelist.*wps* set-up: *&geogrid*

```
&geogrid
  parent_id          =  1,       1,
  parent_grid_ratio  =  1,       3,
  i_parent_start     =  1,      70,
  j_parent_start     =  1,      67,

  e_we               =  175,   181,
  e_sn               =  145,   181,
  geog_data_res      =  'default', 'default',

  dx                 = 30000,
  dy                 = 30000,
  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/'
/
```

parent_grid_ratio:
recommended ratios are 3:1 or 5:1
(odd ratios, less than 7)

i/j_parent_start:

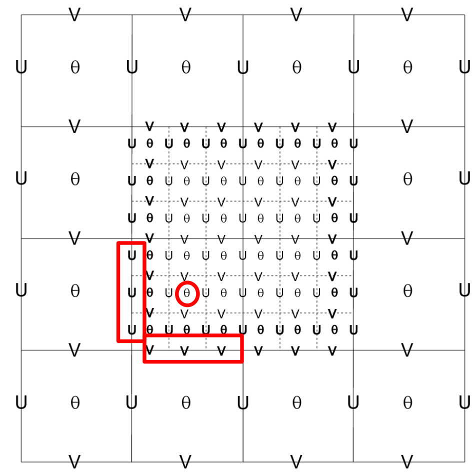


Feedback 3:1 Ratio

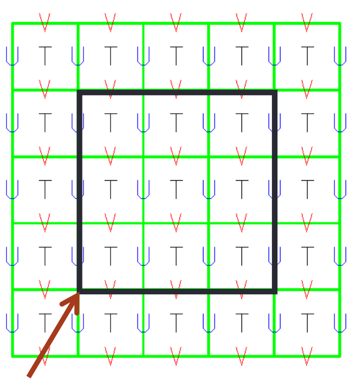
When using feedback, conditions are fed back to the parent domain from the child along the rows and columns, and at the mass points (center)

U: east-west velocities
V: south-north velocities
Θ: all other meteorological data

Averaging is performed



WRF Parent-nest Domain Overlap



i_parent_start
j_parent_start

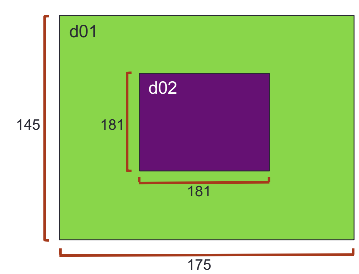
- The nested domain can be placed *anywhere* within the parent domain and the nested grid cells will exactly overlap the parent cells at the coincident cell boundaries

- Coincident parent/nest grid points eliminate the need for complex, generalized remapping calculations, and enhances model performance and portability.

namelist.*wps* set-up: *&geogrid*

```
&geogrid
  parent_id          =  1,       1,
  parent_grid_ratio  =  1,       3,
  i_parent_start     =  1,      70,
  j_parent_start     =  1,      67,

  e_we               =  175,   181,
  e_sn               =  145,   181,
  geog_data_res      =  'default', 'default',

  dx                 = 30000,
  dy                 = 30000,
  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/'
/
```

e_we and e_sn:
Each domain's full west-east and south-north dimensions



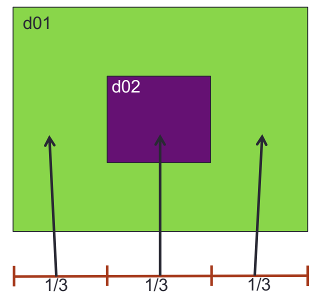
Notes:
- Domains should be no smaller than about 100x100
- Avoid placing any boundaries over complex terrain
- Keep nest away from coarse domain

namelist.*wps* set-up: *&geogrid*

```
&geogrid

 parent_id          =  1,        1,
 parent_grid_ratio  =  1,        3,
 i_parent_start     =  1,       70,
 j_parent_start     =  1,       67,

 e_we               =  175,    181,
 e_sn               =  145,    181,
 geog_data_res      =  'default', 'default',


 dx                 = 30000,
 dy                 = 30000,
 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/'
/
```

Minimum distance between nest boundary and parent boundary:

- 4 grid cells
- need MUCH larger buffer zone



- Good practice to have ~1/3 of coarse-grid surrounding each side of nest
- Nest can be placed a bit downstream of the inflow boundary

namelist.*wps* set-up: *&geogrid*

```
&geogrid

 parent_id          =  1,        1,
 parent_grid_ratio  =  1,        3,
 i_parent_start     =  1,       70,
 j_parent_start     =  1,       67,

 e_we               =  175,    181,
 e_sn               =  145,    181,
 geog_data_res      =  'default', 'default',


 dx                 = 30000,
 dy                 = 30000,
 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/'
/
```

dx and dy:
Only need the coarse domain resolution. The geogrid program calculates the nest resolution(s) using the "parent_grid_ratio"

***Note:**
No changes need to be made to the &ungrib and &metgrid namelists records for nesting purposes

namelist.input (WRFV3)

namelist.*input* set-up: *&time_control*

```
&time_control
 run_days            = 0,
 run_hours           = 24,
 run_minutes         = 0,
 run_seconds         = 0,
 start_year          = 2012,  2012,   2012,
 start_month         = 01,    01,     01,
 start_day           = 27,    27,     27,
 start_hour          = 00,    00,     00,
 start_minute        = 00,    00,     00,
 start_second        = 00,    00,     00,
 end_year            = 2012,  2012,   2012,
 end_month           = 01,    01,     01,
 end_day             = 28,    28,     28,
 end_hour            = 00,    00,     00,
 end_minute          = 00,    00,     00,
 end_second          = 00,    00,     00,
 interval_seconds    = 10800
 input_from_file     = .true., .true., .true.
 history_interval    = 360,    60,     60
 frames_per_outfile  = 1000,   1,      1
 restart             = .false.
 restart_interval    = 180
 io_form_history     = 2
 io_form_restart     = 2
```

** To edit the namelist.input file, make sure you are in the *WRFV3/test/em_real/* (or *WRFV3/run/*) directory

start/end date/times:
These values *typically* will be the same for all domains

history_interval:
May choose to have more frequent output time for nests

frames_per_outfile:
May choose to have all history outputs in a single file, or in multiple files
- to display geographic boundaries in newer verions of ncview, it's necessary to have 1 file per time period.

namelist.*input* set-up: *&domains*

```
&domains
 time_step                    = 180,
 time_step_fract_num          = 0,
 time_step_fract_den          = 1,
 max_dom                      = 2,
 e_we                         = 175,   181,   94,
 e_sn                         = 145,   181,   91,
 e_vert                       = 36,    36,    36,
 p_top_requested              = 5000,
 num_metgrid_levels           = 32,
 num_metgrid_soil_levels      = 4,
 dx                           = 30000, 10000, 3333.33,
 dy                           = 30000, 10000, 3333.33,
 grid_id                      = 1,     2,     3,
 parent_id                    = 0,     1,     2,
 i_parent_start               = 1,     70,    30,
 j_parent_start               = 1,     67,    30,
 parent_grid_ratio            = 1,     3,     3,
 parent_time_step_ratio       = 1,     3,     3,
 feedback                     = 1,
 smooth_option                = 0
/
```

max_dom:
Activate nests - # of domains to run

e_we and e_sn:
should match namelist.wps values

e_vert:
All columns usually have the same value

dx/dy:
must set values for each domain.
make sure values correspond with "parent_grid_ratio"
- for non-integer grid resolutions, use at least two decimal places

namelist.*input* set-up: *&domains*

```
&domains
 ..........
 grid_id                      = 1,   2,   3,
 parent_id                    = 0,   1,   2,
 i_parent_start               = 1,   70,  30,
 j_parent_start               = 1,   67,  30,
 parent_grid_ratio            = 1,   3,   3,
 parent_time_step_ratio       = 1,   3,   3,
 feedback                     = 1
 smooth_option                = 0
/
```

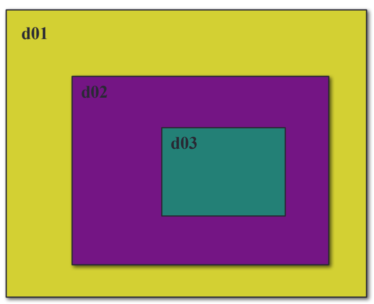
All must be set to the same values used in namelist.wps

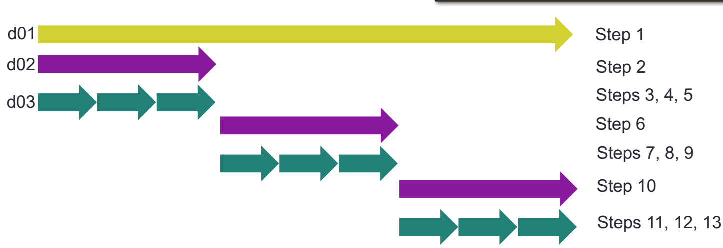
feedback:
Whether a nest will overwrite parent results
- 2-way nesting: feedback = 1
- 1-way nesting: feedback = 0

parent_time_step_ratio:
See next slide!

Nested 3:1 Time Step Ratio

- Example: 3-domain nested run
  - D01: a single 3 min dt
  - D02: a single 1 min dt
  - D03: 20 second intervals, up to 1 min





namelist.*input* set-up: *&physics*

- You should use the same physics options for all domains for all schemes
  - **Exceptions**:
    - cumulus_scheme (cu_physics): may need to be turned off for a nest that has a grid distance of only a few kilometers
    - may turn off PBL scheme for resolutions close to 100 m
- Use same values for physics calling frequency parameters (for each domain)
  - radt: radiation time step
  - bldt: boundary layer time step
  - cudt: cumulus scheme time step

Computationally inexpensive – no reason to not always set to zero (run every time step); NOTE: radt=15 => run radiation every 15 min

Nesting in real.exe

- *real* program reads & processes multiple domain input files from *metgrid* (met_em_d0*)
- *real* program does vertical interpolation only
- There are no consistency check between domains (this is handled in the feedback step for the WRF model)
- *real.exe* must be re-run if you make changes to:
  - Date/time
  - Domain – size, location, quantity
  - A number of physics options (those related to input fields)
  - Input data

Where do I start?

- Always start with a *namelist* template provided in the WRFV3/test/em_real (or WRFV3/run/) directory
- Use documents/websites to guide your namelist modifications
  - WRFV3/run/*README.namelist*
  - WRFV3/test/em_real/*examples.namelist*
  - Users' Guide, Chapter 5
    - http://www2.mmm.ucar.edu/wrf/users/docs/user_guide_V3.9/users_guide_chap5.htm
  - Namelist Best Practice web pages:
    - WPS: http://www2.mmm.ucar.edu/wrf/users/namelist_best_prac_wps.html
    - WRFV3: http://www2.mmm.ucar.edu/wrf/users/namelist_best_prac_wrf.html
- Not all namelist options are domain dependent. If in doubt:
  - Check WRFV3/Registry/*Registry.EM_COMMON* or *registry.io_boilerplate* (grep for parameter names)
  - Check WRFV3/run/*README.namelist* (grep for parameter names)
  - Rule of thumb: If default namelist only has 1 column, don't add values for other columns!

Steps to run with a nest

- WPS: Identical to single domain run:
  1) Make sure you are in the WPS/ directory
  2) Make necessary changes to the *namelist.wps* file
  3) Run geogrid.exe, ungrib.exe, and metgrid.exe

```
./geogrid.exe
./ungrib.exe
./metgrid.exe
```

- WRFV3: Identical to single domain run:
  1) Make sure you are in the *WRFV3/test/em_real* (or *WRFV3/run/) directory*
  2) Move or link WPS output files (*met_em.d0**) to your running directory

```
ln -sf ../../../WPS/met_em* .
```

  3) Edit *namelist.input* file for the appropriate grid and times of the case
  4) Run initialization program (assuming a dmpar compile):

```
mpirun -np n ./real.exe
```

  - "n": number of processors used

  1) Run model executable (assuming a dmpar compile):

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

Successful *real.exe* Run

- If *real.exe* was successful, you should see this at the end of your rsl.error.0000 file (assuming a dmpar compile):
  - `tail rsl.error.0000`
  - SUCCESS COMPLETE REAL_EM INIT
- You should have these files in your running directory:
  - wrfbdy_d01 :
    - Lateral boundary data for all times (domain 01 only)
  - wrfinput_d01, wrfinput_d02, ….
    - Single time-level data at the model's start time (for all domains)
    - 1 file per domain

Successful *wrf.exe* Run

- If *wrf.exe* was successful, you should see this at the end of your rsl.error.0000 file (assuming a dmpar compile):
  - `tail rsl.error.0000`
  - SUCCESS COMPLETE WRF
- You should have these files in your running directory:
  - wrfout_d01_2005-08-28_00:00:00
  - wrfout_d02_2005-08-28_00:00:00
    - One for each domain, for each history time (depending on how you set 'frames_per_outfile')
  - wrfrst_d01_2005-08-28_00:00:00
  - wrfrst_d02_2005-08-28_00:00:00
    - If "restart_interval" is less than or equal to the integration time

Summary

- Decide what is the best strategy to do the simulation
- If nesting is required, design your nest configuration
  - Design the coarse domain first
  - Determine the beginning and ending indices of the nest on the coarse domain
- Choose the appropriate nesting strategy:
  - one-way, two-way, or one-way via *ndown*

Questions?

Fundamentals in Atmospheric Modeling

Song-You Hong

Fundamentals in Atmospheric Modeling

Song-You Hong

(KIAPS: Korea Institute of Atmospheric Prediction Systems)

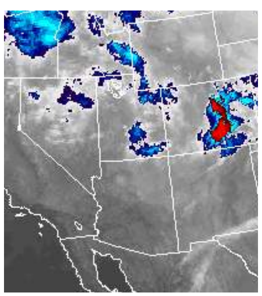
(Also NCAR affiliate scientist)

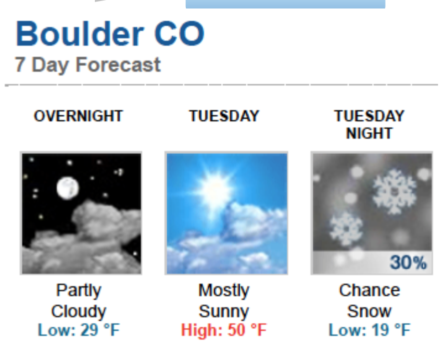
List of presentations

- Concept of modeling
- Structure of models
- Predictability
- Regional modeling

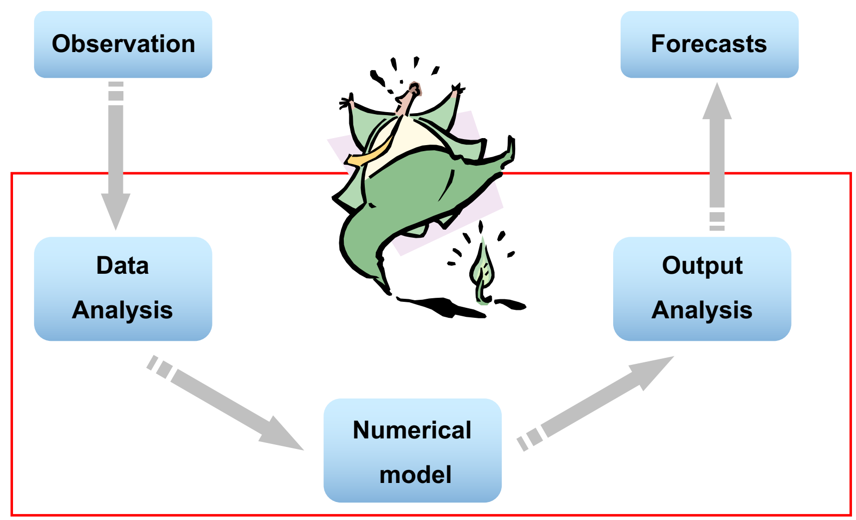
How were the today's forecasts made ?

Observation → **Forecasts**





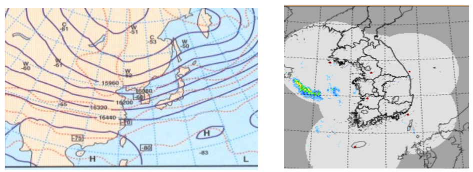
Numerical model is a crucial component



Then, how ?

Step1:
Observation

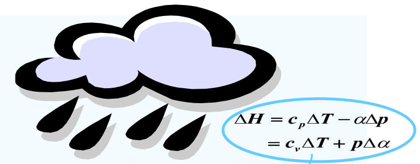
Step2:
Data analysis



Theory of NWP

Thermodynamics

Heat = Energy + Work

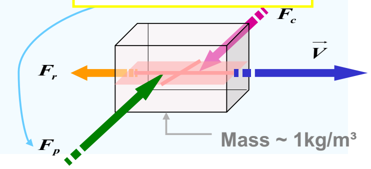


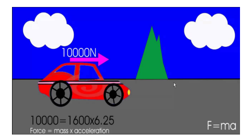
Nonlinear interaction

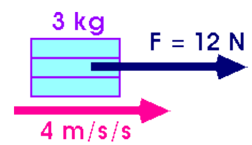
Dynamics

Force = Mass × Acceleration

- Mass ≒ 1 kg/m³
- Force: PGF, Coriolis, Friction…







$\Rightarrow v = v_0 + at$

$\Rightarrow x = v_0 t + \frac{1}{2} a t^2$

Theory of NWP : Atmosphere is conserved

- **Momentum** $F = ma$ — Force = mass x acceleration
- **Mass** $\frac{1}{M}\frac{dM}{dt} = 0$ — Mass of a fluid is conserved
- **Moisture** $\frac{dq}{dt} = E - C$ — Moisture change = evaporation - condensation
- **Energy** $Q = C_v \frac{dT}{dt} + p\frac{d\alpha}{dt}$ — Heat = internal energy change − work done
- **Ideal gas** $p\alpha = RT$ — Pressure x specific volume = gas constant x temperature

The governing equations

__V. Bjerknes__ (1904) pointed out for the first time that there is a complete set of __7 equations with 7 unknowns__ that governs the evolution of the atmosphere:

$$\frac{d\mathbf{v}}{dt} = -\alpha\nabla p - \nabla\phi + \mathbf{F} - 2\Omega\times\mathbf{v} \quad (1\text{-}3)$$ East-west, North-south, and vertical

$$\frac{\partial\rho}{\partial t} = -\nabla.(\rho\mathbf{v}) \quad (4)$$

$$p = \rho R T \quad (5)$$

$$\frac{ds}{dt} = C_p \frac{1}{\theta}\frac{d\theta}{dt} = \frac{Q}{T} \quad (6)$$

$$\frac{dq}{dt} = E - C \quad (7)$$

7 equations, 7 unknown (u,v,w,T, p, den and q)

solvable

History of numerical weather forecasts

1904 : Norwegian **V.** Bjerknes (1862-1951) :
Setup the governing equations

1922 : British **L. F.** Richardson (1881-1953) :
Integrate model ➔ failed

1939 : Swedish **C.-G. Rossby** :

1948, 1949, J. G. Charney (1917-1981)

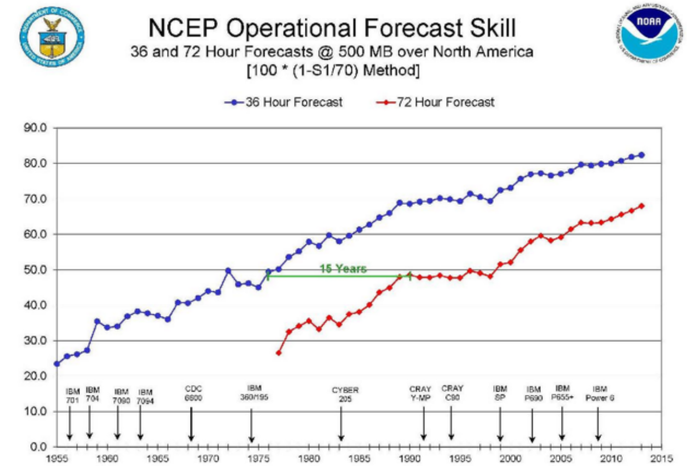
1950 : **Princeton Group**
(Charney, Fjortoft,
von Newman)
ENIAC
(Electrical Numerical
Integrator and Computer)
➔ **first success**

Computer Age (1946~)

- von Neumann and Charney
  - Applied ENIAC to weather prediction

- Carl-Gustaf Rossby
  - The Swedish Institute of Meteorology
  - First routine real-time numerical weather forecasting. (1954)
  - (US in 1958, Japan in 1959)

History of NWP skill : NCEP GFS



1day / 10year

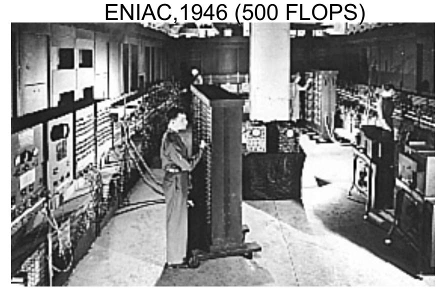
1987: 48 hr fcst skill= 2017: 120 hr fcst skill

Factors for the improvement (Kalnay 2002)

- Supercomputers
- Physical processes
- Initial conditions

Super-computer for weather models

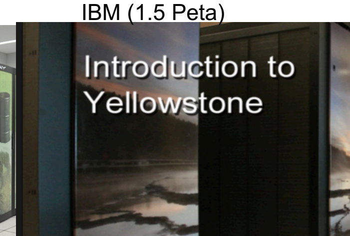
ENIAC,1946 (500 FLOPS)



Sunway (125 Peta=10**15FLOPS)

XC40 (2.9 Peta)

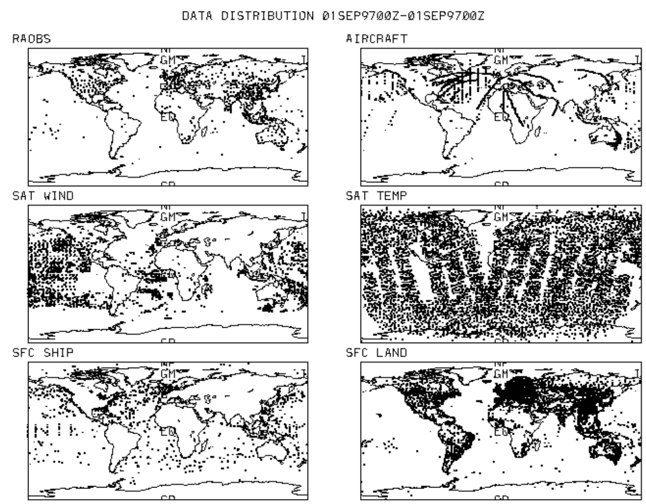
IBM (1.5 Peta)



K-computer (11.2 Peta)

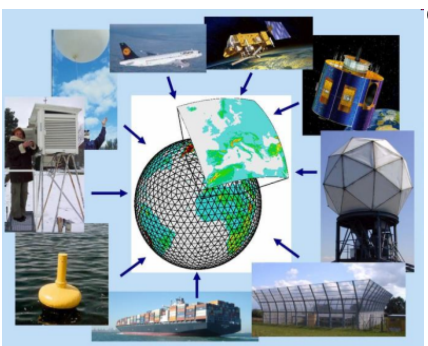
Initial condition (data assimilation)

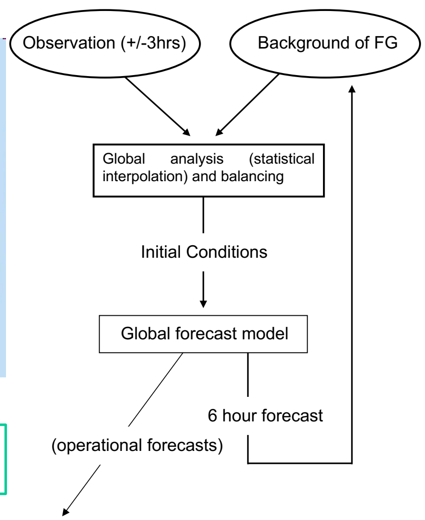
Various observations



Heterogeneous in space and time….

Data Assimilation





Data assimilation best combines observations and a model

Model

- Dynamics : Identity (Speed)
- Physics : Components (Predictability)

Step3: Integration

Dynamics : Grid system

$$u_t + uu_x + vu_y + wu_z = -\frac{1}{\rho}p_x + \left(f + \frac{u}{a\tan\phi}\right)v + F_x$$

$$v_t + uv_x + vv_y + wv_z = -\frac{1}{\rho}p_y - \left(f + \frac{u}{\tan\phi}\right)u + F_y$$

$$w_t + uw_x + vw_y + ww_z = -\frac{1}{\rho}p_z - g + F_z$$

$$\rho_t + u\rho_x + v\rho_y + w\rho_z = -\rho\left(u_x + v_y + w_z\right)$$

$$T_t + uT_x + vT_y + wT_z - \frac{1}{\rho C_p}\left(p_t + up_x + vp_y + wp_z\right) = \boxed{\frac{1}{C_p}Q}$$

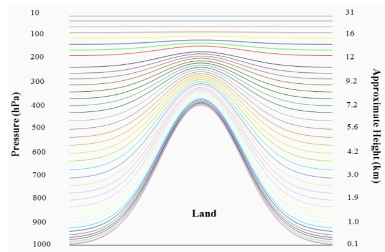
$$q_t + uq_x + vq_y + wq_z = \boxed{M}$$

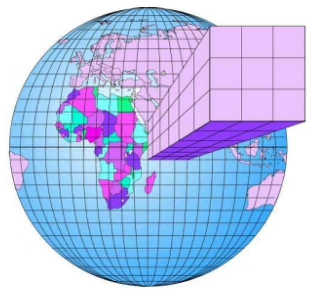
$$p = \rho R T$$

PHYSICS

unknown : $[u, v, w, \rho, T(\theta), q, p]$

If we consider O_3, $C_t + uC_x + vC_y + wC_z = \boxed{O_3}$





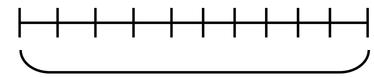
Dynamics : Numerical method (spatial)

Finite difference method (FDM) :
Spectral method (SPM) :
Finite element method (FEM) :

Ex) $\frac{\partial\phi}{\partial t} = -c\frac{\partial\phi}{\partial x}$; advection eq.

1) FDM (Finite difference)

$$\frac{\Delta\phi}{\Delta t} = \frac{\phi_2 - \phi_1}{t_2 - t_1}$$



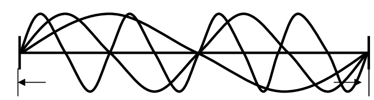
L M : # of grids

2) Spectral Method

- Determine basis function to get $H(\phi(x))$

$e_m(x)$ (basis funct), $m = m_1 \cdots m_n$.. → infinite

$$\Rightarrow \phi(x,t) = \sum_{m=m_1}^{M}\phi_m(t)e_m(x)$$



L M : # of waves

* Resolution Increases: $\Delta x \rightarrow decreases$ $(\Delta x = L/M)$ $M \rightarrow increases$

Dynamics : Numerical method (temporal)

a) $\frac{u^{n+1} - u^{n-1}}{2\Delta t} = F(u^n)$: **leap-frog** **good for hyperbolic** **unstable for parabolic**

b) $\frac{u^{n+1} - u^n}{\Delta t} = F(u^n)$: **Euler-forward** **good for diffusion** **unstable for hyperbolic**

c) $\frac{u^{n+1} - u^n}{\Delta t} = F\left(\frac{u^n + u^{n+1}}{2}\right)$: **Crank-Nicholson**

d) $\frac{u^{n+1} - u^n}{\Delta t} = F(u^{n+1})$: **Fully implicit, backward**

e) $\frac{u^* - u^n}{\Delta t} = F(u^n)$; $\frac{u^{n+1} - u^n}{\Delta t} = F(u^*)$: **Euler-backward (Matzuno)**

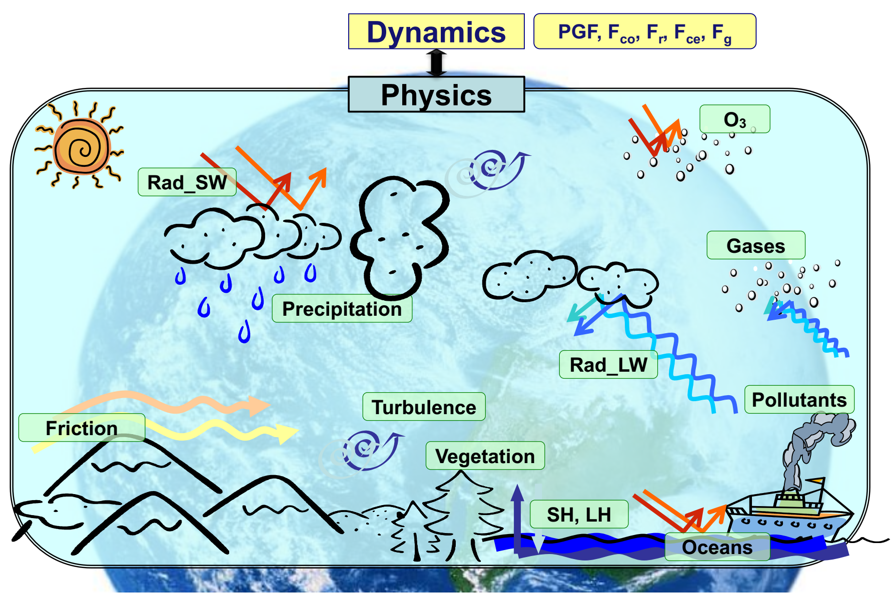
f) $\frac{u^{n+\frac{1}{2}*} - u^n}{\Delta t/2} = F(u^n)$; $\frac{u^{n+1*} - u^n}{\Delta t} = F\left(u^{n+\frac{1}{2}*}\right)$

$\frac{u^{n+1} - u^n}{\Delta t} = \frac{1}{6}\left[F(u^n) + 4F\left(u^{n+\frac{1}{2}*}\right) + F\left(u^{n+1*}\right)\right]$: **RK(Runge-Kuta)-3rd order**

g) $\frac{u^{n+1} - u^{n-1}}{2\Delta t} = F_1(u^n) + F_2\left(\frac{u^{n+1} - u^{n-1}}{2}\right)$: **Semi-Implicit**

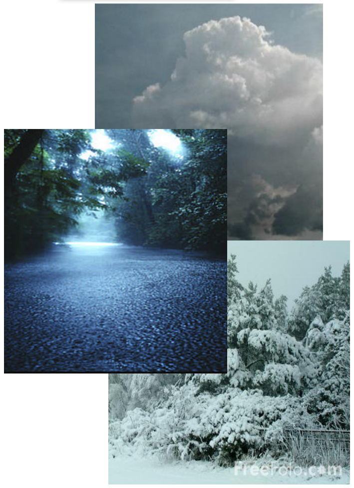
h) $\frac{u^* - u^n}{\Delta t} = F_1(u^n)$; $\frac{u^{n+1} - u^*}{\Delta t} = F_2(u^*)$: **Fractional steps**

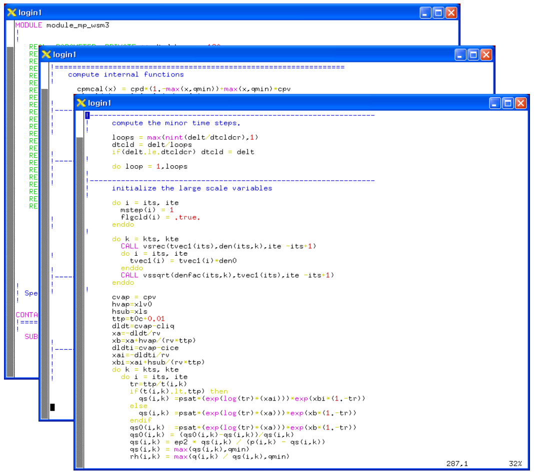
Physics modules : Branches of atmospheric sciences



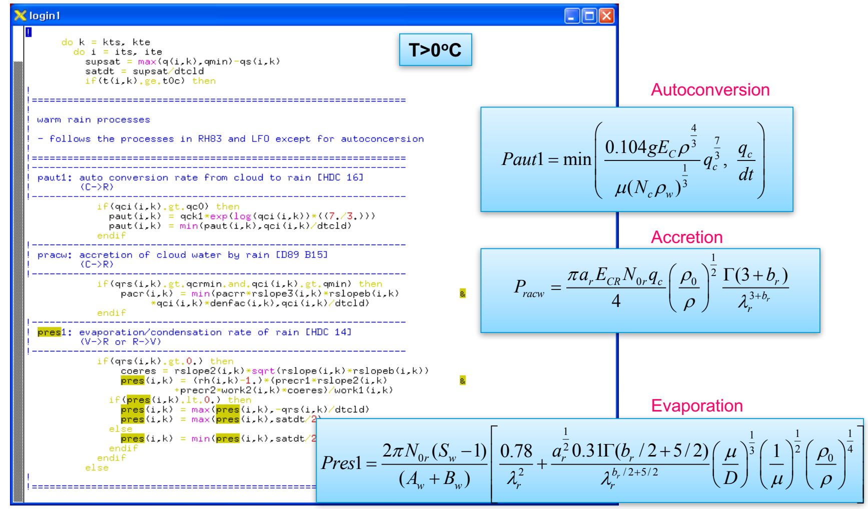
Physics module (example): Cloud and precipitation

Real atmosphere → Theory / Formulation → Model (computer program)





Physics module (example): Cloud and precipitation



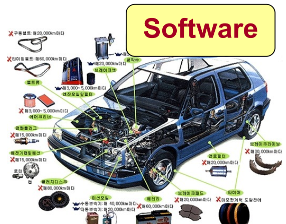
$$Paut1 = \min\left(\frac{0.104 g E_C \rho^{\frac{4}{3}}}{\mu (N_c \rho_w)^{\frac{1}{3}}} q_c^{\frac{7}{3}}, \frac{q_c}{dt}\right)$$

$$P_{racw} = \frac{\pi a_r E_{CR} N_{0r} q_c}{4}\left(\frac{\rho_0}{\rho}\right)^{\frac{1}{2}} \frac{\Gamma(3+b_r)}{\lambda_r^{3+b_r}}$$

$$Presl = \frac{2\pi N_{0r}(S_w - 1)}{(A_w + B_w)}\left[\frac{0.78}{\lambda_r^2} + \frac{a_r^{\frac{1}{2}} 0.31 \Gamma(b_r/2 + 5/2)}{\lambda_r^{b_r/2+5/2}}\left(\frac{\mu}{D}\right)^{\frac{1}{3}}\left(\frac{1}{\mu}\right)^{\frac{1}{2}}\left(\frac{\rho_0}{\rho}\right)^{\frac{1}{4}}\right]$$

Car and model

Dynamics

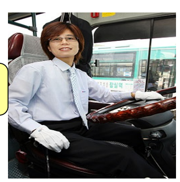


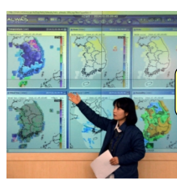
Software

Physics

Data assimilation

Driver





Forecaster

Classification of models

- **Dynamic core**

Hydrostatic	Non-hydrostatic
Large-scale	Small-scale (heavy rainfall, complex mountain)

- **Scale**

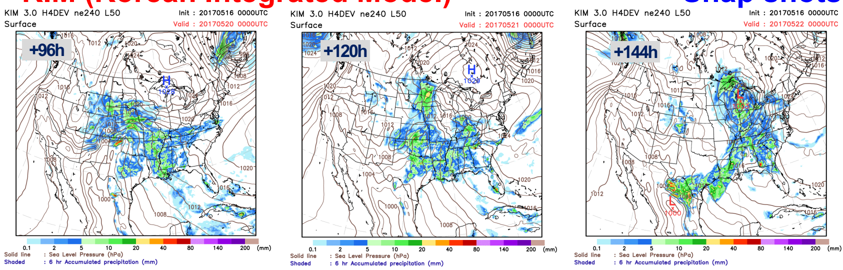
Global	Regional
10 km – 100 km (NWP – Climate)	1 km - 10 km (NWP-Climate)

- **Purpose**

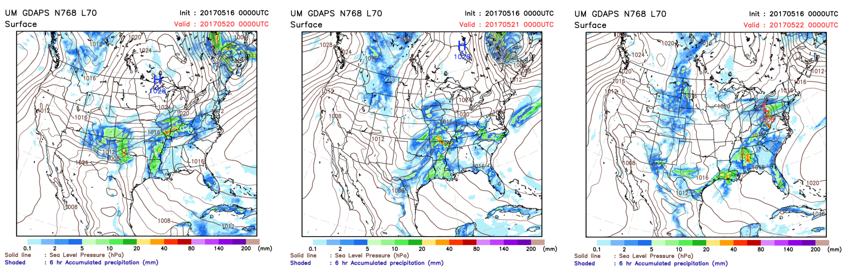
Initial data-> FORECAST	Forcing → RESPONSE
NWP : upto 2 weeks	**GCM** (General circulation model)

KIAPS real time forecasts (INIT 2017051600) : 12 km global NWP

KIM (Korean Integrated Model) **Snap shots**



KMA UM



Predictability

Chaos theory (Lorenz)

Charney (1951) : Uncertainties in initial condition and model

↓

Lorenz (1962,1963) : Unstable nature of atmosphere

↓

Purpose : NWP is better than statistical forecast
Tool : 4 K memory computer
Model : 12 variables (heating and dissipation forcing)
Results : differences -> non-periodicity

↓

Initial condition (3 decimal point) : different after 2 month

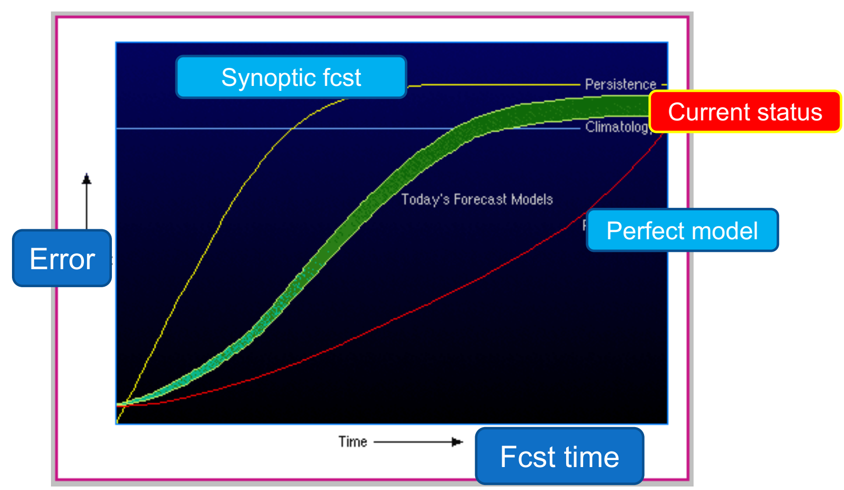
↓

Round-off error -> cause of non-periodcity

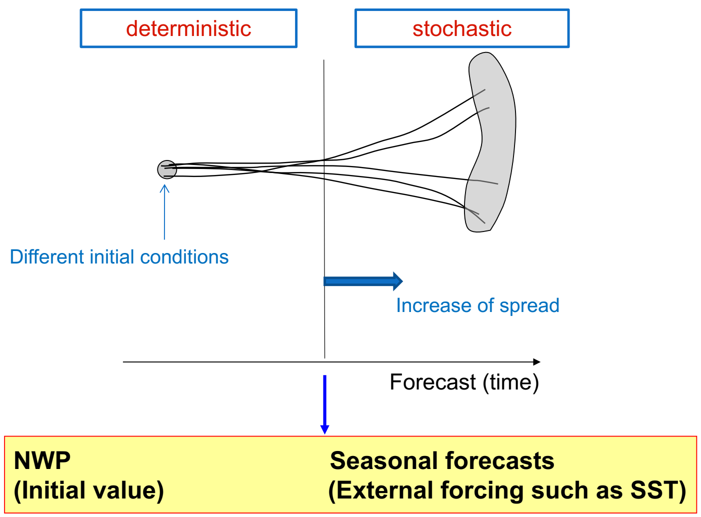
↓

Chaos theory– two weeks for NWP

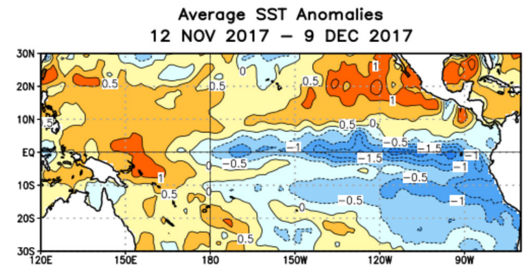
Predictability : Atmosphere is chaotic



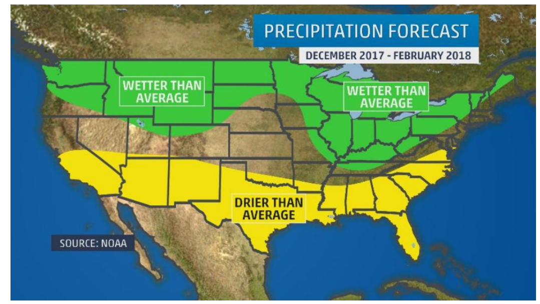
Ensemble forecasts : Seasonal and beyond



Ensemble forecasts : Seasonal and beyond



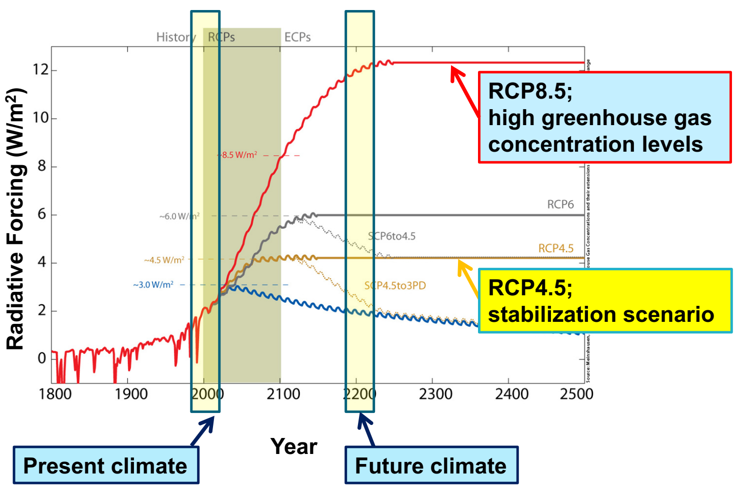
SST anomaly in November 2017



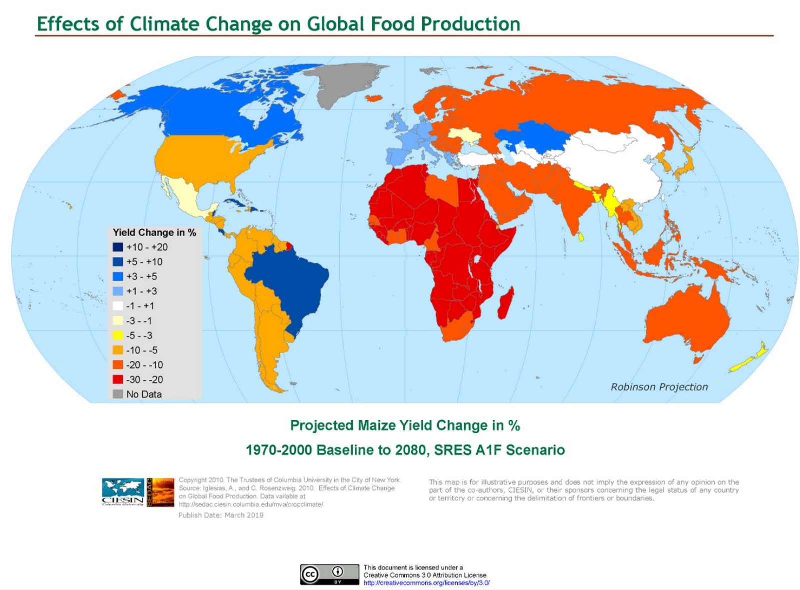
Precipitation anomaly in winter 2017/18

Climate prediction : For given RCP scenarios

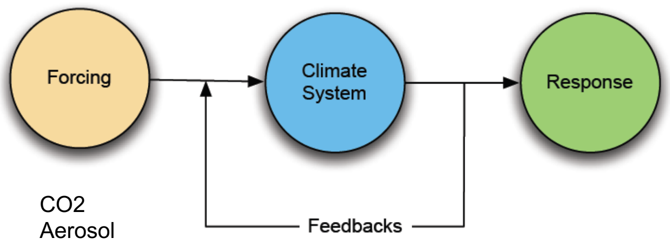
Climate changes = future minus present



Climate prediction : For given RCP scenarios



Climate prediction : Climate system sensitivity

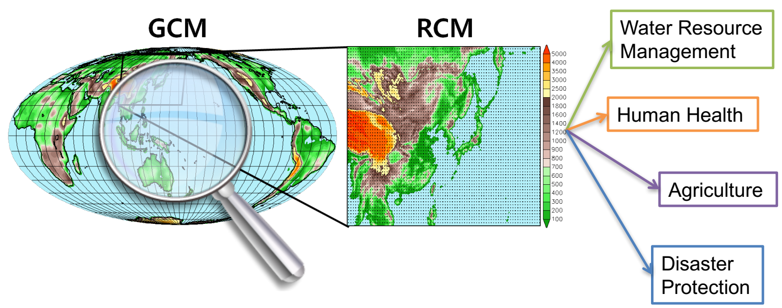


CO_2
Aerosol
Volcanic

Water vapor feedback
Ice-albedo feedback
Vegetation feedbacks
Cloud (radiative) feedback
(Great debate!, Mostly still uncertain)
…

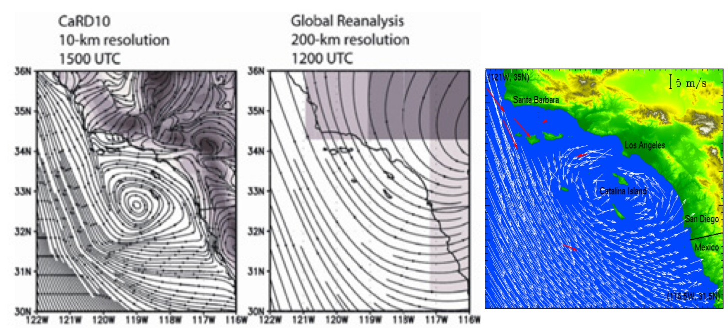
Global vs Regional

Regional modeling



Regional model is a magnifying glass

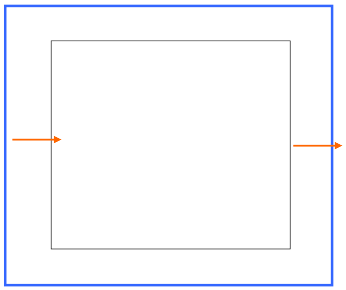
High resolution benefit ? ---- Very clear !



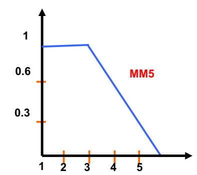
10 km 200 km Observed

Another inherent issue in regional modeling
: lateral boundary treatment is empirical

Buffer zone



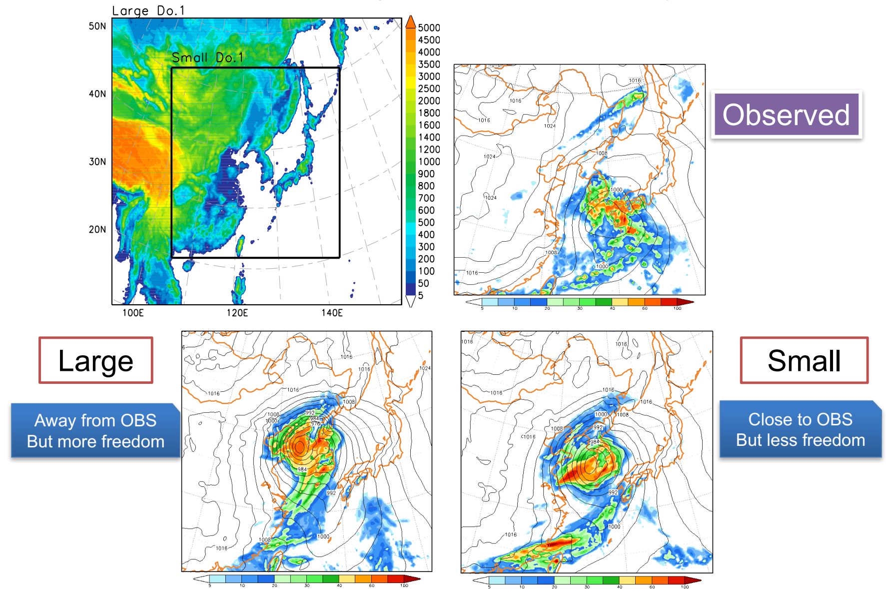
F(n) : weighting of global



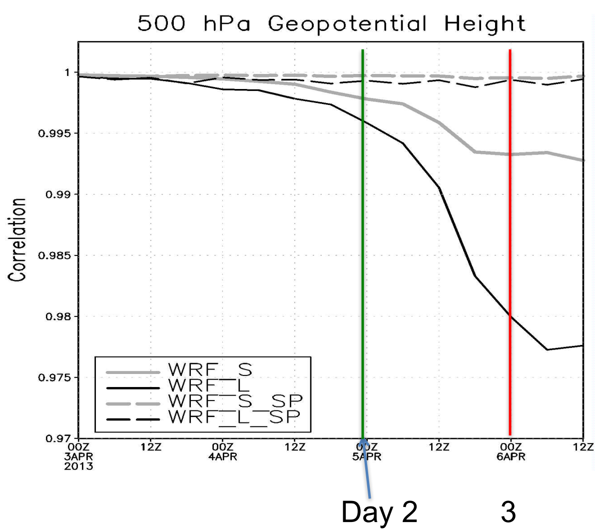
$$\frac{\partial A}{\partial t}\Big|_n = F(n)F_1(A_{CM} - A_{FM}) - F(n)F_2\nabla^2(A_{CM} - A_{FM})$$

So, empirical

Domain size sensitivity : A mid-latitude cyclone



Domain size sensitivity : Pattern correlation with global



Fundamental limit of the regional model : low resolution global and mathematically ill-posed setup

Small domain keeps the large-scale from the global but loses its freedom

Spectral nudging keeps the large-scale, but may lose the regional details

Thanks for your attention !
songyouhong@gmail.com

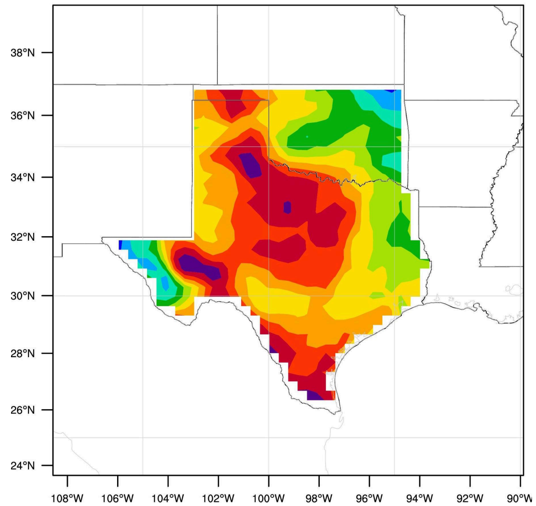
Hong, S.-Y., and M. Kanamitsu, 2014: Dynamical downscaling: Fundamental issues from an NWP point of view and recommendations. *Asia-Pac. J. Atmos. Sci.*, **50,** 83-104, doi: 10.1007/s13143-014-0029-2.

Dudhia, J., 2014: A history of mesoscale model Development. *Asia-Pac. J. Atmos. Sci.*, **50,** 121-131.

Post-processing Tools (1): NCL

Abby Jaye

Post-processing Tools: NCL



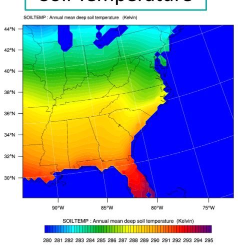
Abby Jaye

NCL

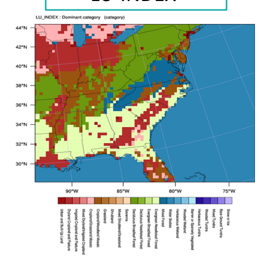
- NCAR Command Language
- Website: http://www.ncl.ucar.edu
- Reads WRF-ARW data directly
- Can generate many types of graphical plots
  - Horizontal
  - Cross-section
  - SkewT
  - Meteogram
  - Panel

Example Plots

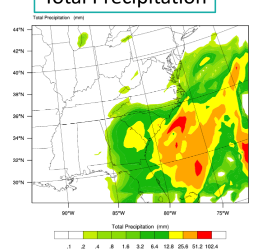
Soil Temperature



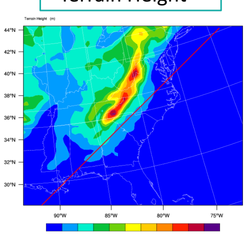
LU-INDEX



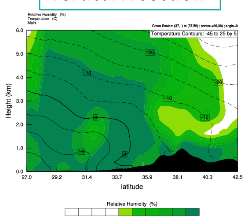
Total Precipitation



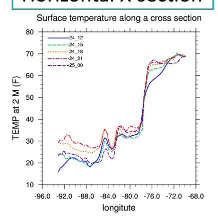
Terrain Height



Relative Humidity Vertical X-section

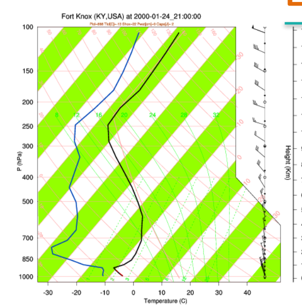


Sfc Temperature Horizontal X-section

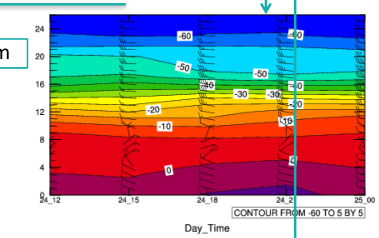


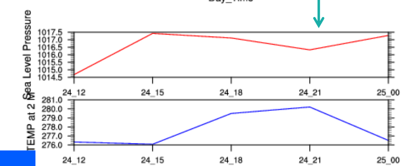
Example Plots

Skew-T Diagram

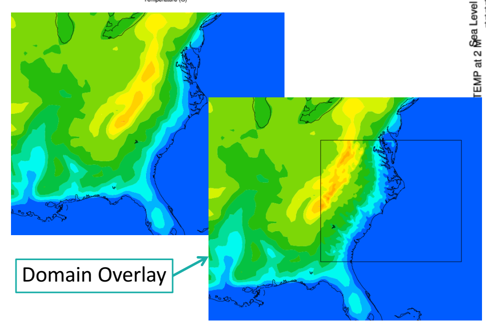


Time Series

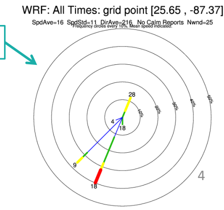




Domain Overlay

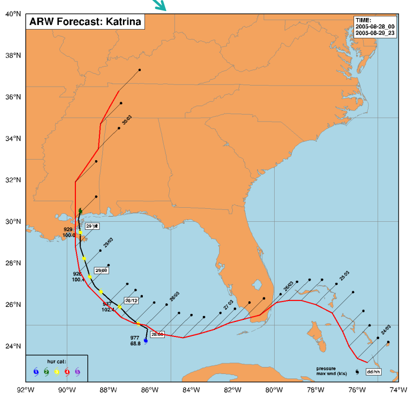


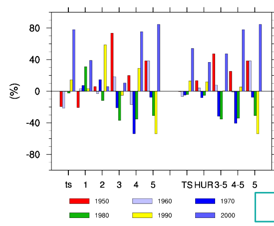
Wind Rose



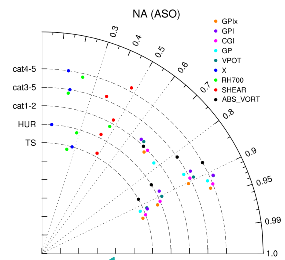
Example Plots

Storm Track





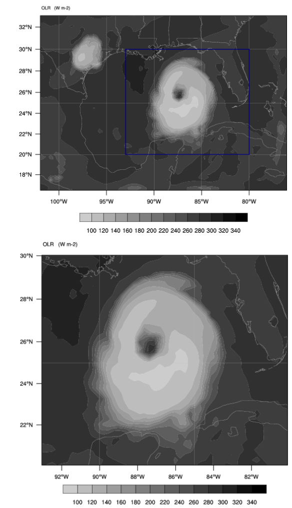
Bar Chart



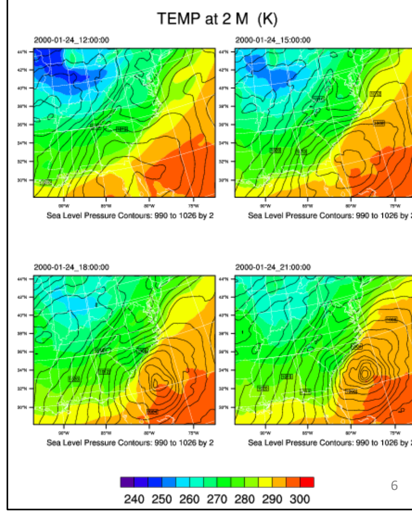
Taylor Diagram

Example Plot Functions

Zooming



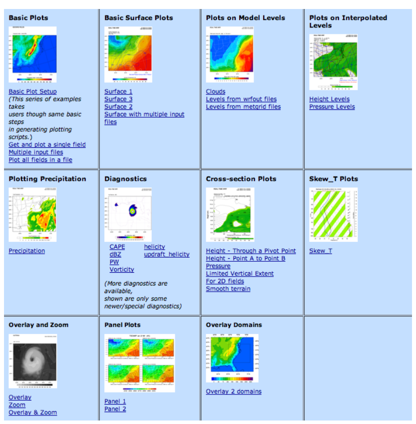
Panel Plots

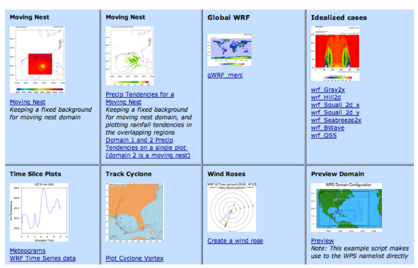


Generating Plots

A good start: WRF Online Tutorial

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





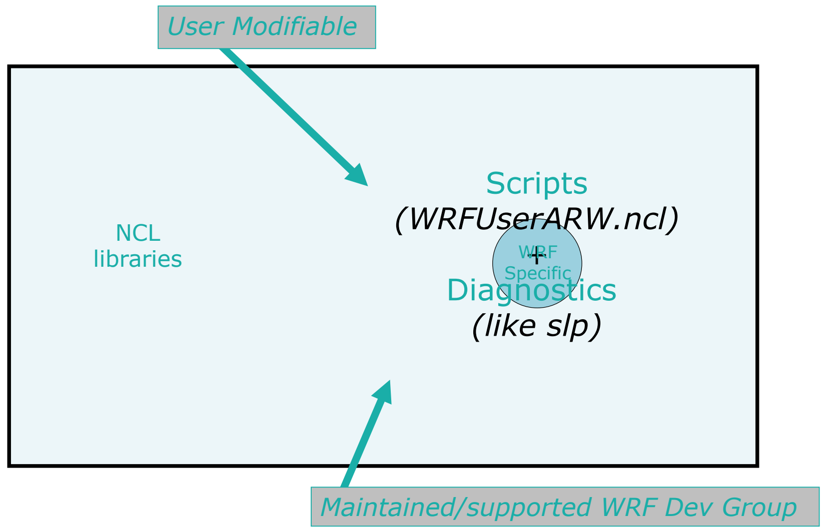
NCL Download

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: v6.4.0)

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

NCL & WRF



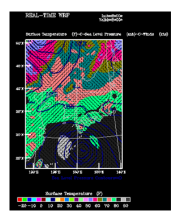
~/.hluresfile

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

~/.hluresfile

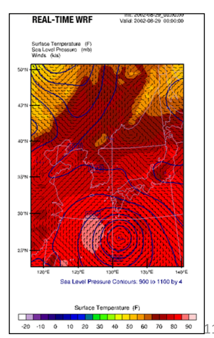
Without it:

```
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           : ncl_default
*wkBackgroundColor    : white
*wkForegroundColor    : black
*FuncCode             : ~
*TextFuncCode         : ~
*Font                 : helvetica
*wkWidth              : 1000
*wkHeight             : 1000
```



Generating Plots

- Set NCARG_ROOT environment variable
  setenv NCARG_ROOT /usr/local/ncl ← **for example**
- Ensure you have a ~./hluresfile file
- Create a script
  - *wrf_real.ncl*
    (start with a sample script)
    Most of the WRF script routines are called from
    "$NCARG_ROOT/lib/ncarg/nclscripts/wrf/WRFUserARW.ncl"
    Feel free to add or change this script
- Run NCL script
  ncl wrf_real.ncl

Creating a Plot: NCL script

```
load ncl library scripts

begin

 ; Open input file(s)
 ; Open graphical output

 ; Read variables

 ; Set up plot resources & Create plots
 ; Output graphics

end
```


Creating a Plot: NCL script

```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/wrf/WRFUserARW.ncl"

begin

 ; Open input file(s)
 ; Open graphical output

 ; Read variables

 ; Set up plot resources & Create plots
 ; Output graphics

end
```

```
Load not required for NCL version 6.3
load "/mydir/myWRFUserARW.ncl"
```


Creating a Plot: NCL script

```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/wrf/WRFUserARW.ncl"

begin

 ; Open input file(s)
 ; Open graphical output

 ; Read variables

 ; Set up plot resources & Create plots
 ; Output graphics

end
```


Creating a Plot: NCL script

```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/wrf/WRFUserARW.ncl"

begin

 a = addfile("./wrfout_d01_2012-09-28_00:00:00.nc","r")
 ; Open graphical output

 ; Read variables

 ; Set up plot resources & Create plots
 ; Output graphics

end
```

```
a = addfile("./wrfout_d01_2005-10-08_00:00:00.nc","r")
Can be "r", "w", "c"
```

```
> ls wrfout*
  wrfout_d01_2005-10-08_00:00:00

a = addfile("./wrfout_d01_2005-10-08_00:00:00.nc","r")
```

Creating a Plot: NCL script

```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/wrf/WRFUserARW.ncl"

begin

 a = addfile("./wrfout_d01_2012-09-28_00:00:00.nc","r")
 wks = gsn_open_wks("X11","plt_Surface")

 ; Read variables

 ; Set up plot resources & Create plots
 ; Output graphics

end
```

Can output either on the screen (X11), or as pdf, eps, ps, png, cgm

Creating a Plot: NCL script

```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/wrf/WRFUserARW.ncl"

begin

 a = addfile("./wrfout_d01_2012-09-28_00:00:00.nc","r")
 wks = gsn_open_wks("X11","plt_Surface")

 T2 = wrf_user_getvar(a,"T2",0)

 ; Set up plot resources & Create plots
 ; Output graphics

end
```

```
T2 = wrf_user_getvar(a,"T2",0)
T2 = a->T2(0,:,:)

T2 = wrf_user_getvar(a,"T2",-1)
T2 = a->T2
```


Special WRF NCL Functions

wrf_user_getvar
Get fields from input file

ter = wrf_user_getvar(a,"HGT",0) {ter=a->HGT(0,:,:)}
t2 = wrf_user_getvar(a,"T2",-1) {t2=a->T2}
slp = wrf_user_getvar(a,"slp",1)

avo/pvo: Absolute/Potential Vorticity, eth: Equivalent Potential Temperature, cape_2d: 2D mcape/mcin/lcl/lfc, cape_3d: 3D cape/cin, ctt: cloud top temperature, dbz/mdbz: Reflectivity (3D and max), geopt/geopotential: Geopotential, lat/lon: latitude/longitude
helicity/updraft_helicity: Storm Relative Helicity/Updraft helicity, omg: Omega, p/pres/pressure: Pressure, pw: Precipitable Water, rh/rh2: Relative Humidity (3D and 2m), slp: Sea Level Pressure, times: Time as a string [(*Times: Time as characters)*], td/td2: Dew Point Temperature (3D and 2m), ter: terrain, tc/tk: Temperature (C and F), th/theta: Potential Temperature, tv: Virtual Temperature, twb: Wetbulb Temperature, z/height: Height, ua/va/wa: wind on mass points, uvmet/uvmet10: wind rotated to earth coordinates (3D and 10m)

Creating a Plot: NCL script

```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/wrf/WRFUserARW.ncl"

begin

 a = addfile("./wrfout_d01_2012-09-28_00:00:00.nc","r")
 wks = gsn_open_wks("X11","plt_Surface")

 T2 = wrf_user_getvar(a,"T2",0)

 ; Set up plot resources & Create plots
 ; Output graphics

end
```

Creating a Plot: NCL script

```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/wrf/WRFUserARW.ncl"

begin

 a = addfile("./wrfout_d01_2012-09-28_00:00:00.nc","r")
 wks = gsn_open_wks("X11","plt_Surface")

 T2 = wrf_user_getvar(a,"T2",0)

 pltres = True
 mpres = True
 opts = True
 opts@cnFillOn = True
 ; Output graphics

end
```

```
pltres: Plotting resources – like overlays
mpres:  Map resources – like map resolution
    and zooming option
```

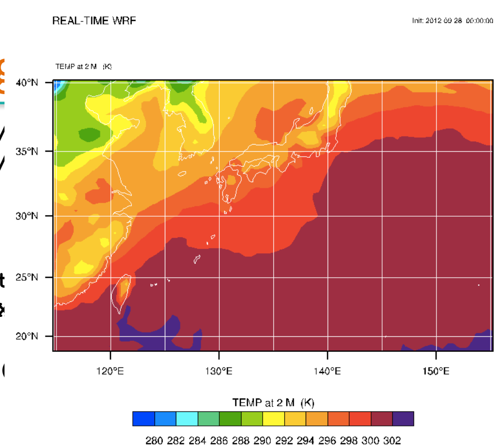
```
opts:   Resources associated with each
    individual plot
```


Creating a Plot: NCL script

```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/wrf/WRFUserARW.ncl"

begin

 a = addfile("./wrfout_d01_2012-09-28_00:00:00.nc","r")
 wks = gsn_open_wks("X11","plt_Surface")

 T2 = wrf_user_getvar(a,"T2",0)

 pltres = True
 mpres = True
 opts = True
 opts@cnFillOn = True
 contour_t2 = wrf_contour(a,wks,T2,opts)
 plot= wrf_map_overlays(a,wks,(/contour_t2/),pltres,mpres)

end
```


Creating

```
load "$NCARG_ROOT/lib/                                  "
load "$NCARG_ROOT/lib/                                  cl"

begin

 a = addfile("./wrfout
 wks = gsn_open_wks("X

 T2 = wrf_user_getvar(

 pltres = True
 mpres = True
 opts = True
 opts@cnFillOn = True
 contour_t2 = wrf_contour(a,wks,T2,opts)
 plot= wrf_map_overlays(a,wks,(/contour_t2/),pltres,mpres)

end
```

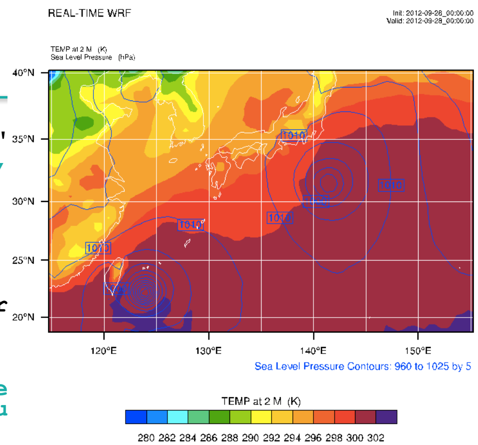


Creating a Plot: NCL script

```
  T2 = wrf_user_getvar(a,"T2",0)
  slp = wrf_user_getvar(a,"slp",0)

  pltres = True
  mpres = True

  opts = True
  opts@cnFillOn = True
  contour_t2 = wrf_contour(a,wks,T2,opts)
  delete(opts)

  opts = True
  opts@cnLineColor = "Blue"
  contour_slp = wrf_contour(a,wks,slp,opts)
  delete(opts)

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

end
```

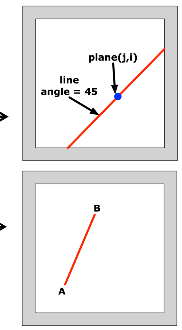
Creating a

```
T2 = wrf_user_getvar(a,"
slp = wrf_user_getvar(a,

pltres = True
mpres = True

opts = True
opts@cnFillOn = True
contour_t2 = wrf_contour
delete(opts)

opts = True
opts@cnLineColor = "Blue
contour_slp = wrf_contou
delete(opts)

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

end
```



Special WRF NCL 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
- wrf_user_intrp3d / wrf_user_intrp2d
  Interpolate horizontally to a given pressure/height
  (3d data only)
  Interpolate vertically along a given line
- wrf_user_ll_to_ij / wrf_user_ij_to_ll
  Convert: lat/lon ⟷ ij
- wrf_user_unstagger
  Unstaggers an array
- wrf_user_vert_interp
- wrf_wps_read_int / wrf_wps_write_int



NCL and WRF_NCL

- Combine strength of WRF_NCL specific and NCL general capabilities

```
plot = wrf_map_overlays \
  (a,wks,(/contour/),pltres,mpres)

mpres@mpGridSpacingF = 45
 plot = wrf_map_overlays \
  (a,wks,(/contour/),pltres,mpres)

 mpres@mpGeophysicalLineColor
 mpres@mpGridLineColor
 mpres@mpNationalLineColor
 mpres@mpUSStateLineColor

 mpres@mpOutlineBoundarySets
    "NoBoundaries"  ; "Geophysical"
    "National"      ; "USStates"
    "GeophysicalAndUSStates"
    "AllBoundaries"
```

```
a = addfile("./wrfout.d01.nc","r")

t2 = a->T2(5,:,:)
t2 = wrf_user_getvar(a,"T2",5)

qv = a->QVAPOR(5,:,:,:)
qv = wrf_user_getvar(a,"QVAPOR",5)

t2 = a->T2
t2 = wrf_user_getvar(a,"T2",-1)

t2 = wrf_user_getvar(a,"T2",
(/0,10,2/))
t2 = wrf_user_getvar(a,"T2",
(/1,2,3,4,5/))
```

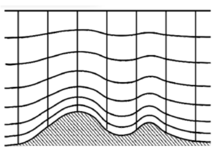

NCL Resources

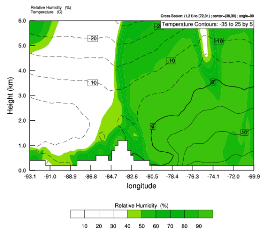
- The special WRF functions have unique resources:
  http://www.ncl.ucar.edu/Document/Functions/wrf.shtml
- All general NCL resources can also be used to control the plot:
  http://www.ncl.ucar.edu/Document/Graphics/Resources

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)

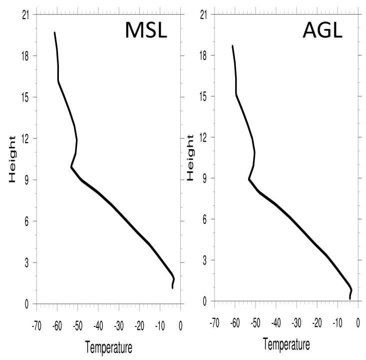
wrf_user_intrp3d

- rh_plane = wrf_user_intrp3d(rh,z,"v",plane,angle,opts)
  - Interpolate “rh” to “z”, which by definition is **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
```



wrf_user_vert_interp

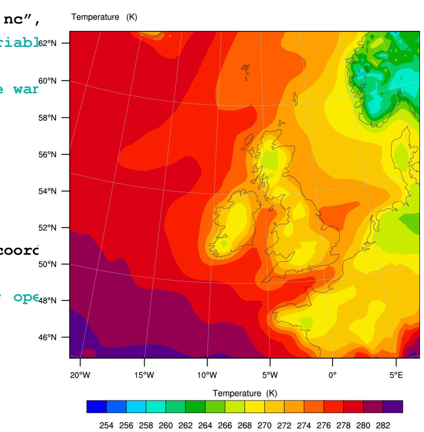
- Interpolate to:
  - "pressure", "pres" - pressure [hPa]
  - "ght_msl" - grid point height msl [km]
  - "ght_agl" - grid point height agl [km]
  - "theta" - potential temperature [K]
  - "theta-e" - equivalent potential temperature [K]
- Extrapolate below the ground
  - **Resource - opts@extrapolate**

wrf_user_vert_interp

```
begin
  a = addfile("wrfout_d01_1991-01-01_00:00:00.nc","r")
  tk = wrf_user_getvar(a,"tk",0) ; Get our variable

  vert_coord = "pressure" ; Set the surface we want to interpolate to and which levels
  interp_levels = (/200,300,500,1000/)

  opts = True ; Set options for the function
  opts@extrapolate = True
  opts@field_type = "t"
  opts@logP = True
  tk_interp = wrf_user_vert_interp(a,tk,vert_coord,interp_levels,opts)

  wks = gsn_open_wks("X11","plot_tk_1000mb") ; open the workstation
  opts2 = True ; Set options for the plot
  opts2@cnFillOn = True
  pltres = True
  mpres = True
  mpres@mpGeophysicalLineColor = "Black"
  ; Make the contour and plot it over a map
  contour = wrf_contour(a,wks,tk_interp(0,3,:,:),opts2) ; Plot at time 0 and level 3
  plot = wrf_map_overlays(a,wks,(/contour/),pltres,mpres)
end
```

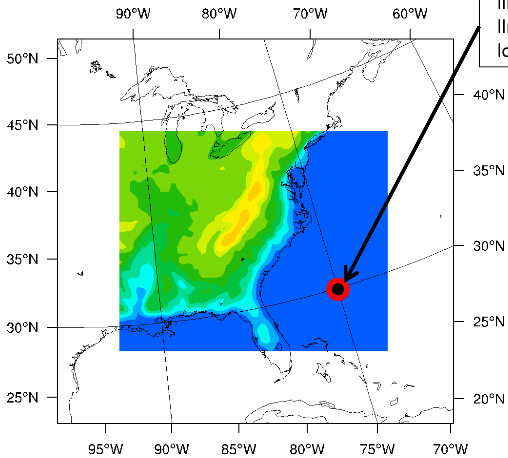

wrf_user_vert_interp

```
begin
  a = addfile("wrfout_d01_1991-01-01_00:00:00.nc",
  tk = wrf_user_getvar(a,"tk",0) ; Get our variabl

  vert_coord = "pressure" ; Set the surface we war
  interp_levels = (/200,300,500,1000/)

  opts = True ; Set options for the function
  opts@extrapolate = True
  opts@field_type = "t"
  opts@logP = True
  tk_interp = wrf_user_vert_interp(a,tk,vert_coorc

  wks = gsn_open_wks("X11","plot_tk_1000mb") ; ope
  opts2 = True ; Set options for the plot
  opts2@cnFillOn = True
  pltres = True
  mpres = True
  mpres@mpGeophysicalLineColor = "Black"
  ; Make the contour and plot it over a map
  contour = wrf_contour(a,wks,tk_interp(0,3,:,:),opts2) ; Plot at time 0 and level 3
  plot = wrf_map_overlays(a,wks,(/contour/),pltres,mpres)
end
```

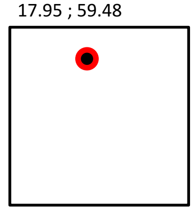


wrf_user_ll_to_ij

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

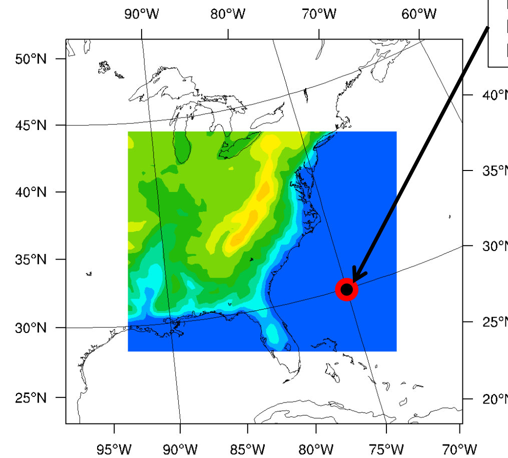


17.95 ; 59.48

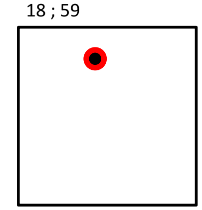


wrf_user_ll_to_ij

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

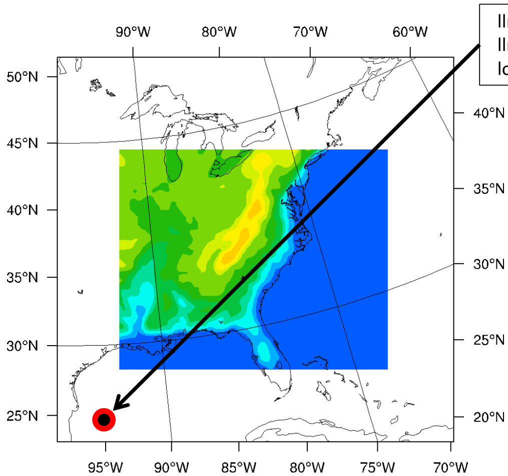


18 ; 59



wrf_user_ll_to_ij

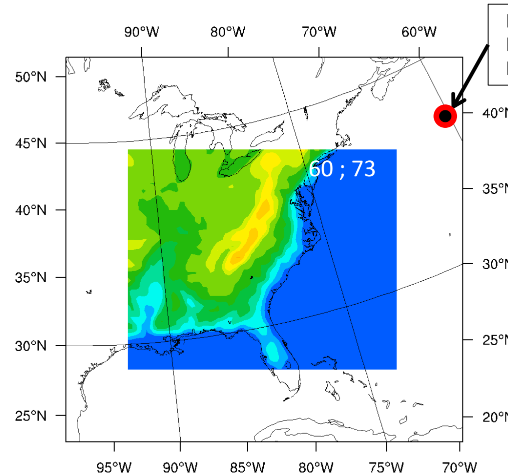
llres = True
llres@returnInt = True
locij = wrf_user_ll_to_ij(a, -95.0, 25.0, llres)



-10 ; -2

wrf_user_ll_to_ij

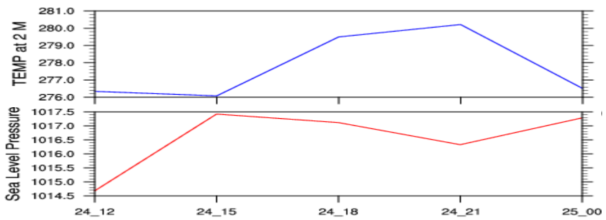
llres = True
llres@returnInt = True
locij = wrf_user_ll_to_ij(a, -95.0, 25.0, llres)



68; 87

Time Series

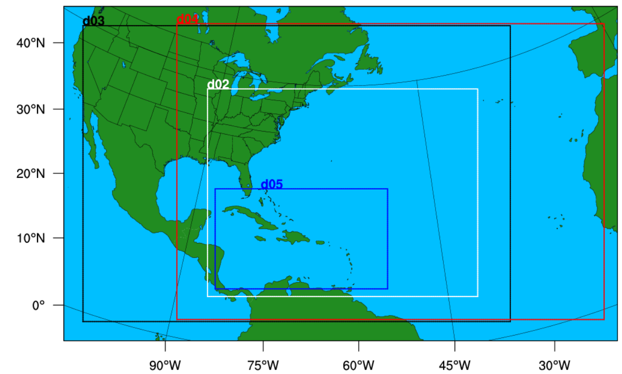
```
locij = wrf_user_ll_to_ij(a, -87., 32.5, llres)
locij = locij - 1
locX  = locij(0)
locY  = locij(1)

t2_point  = a->T2(:,locY,locX)
t2_plot   = gsn_csm_xy(wks,taus,t2_point,t2_res)

slp       = wrf_user_getvar(a,"slp",-1)
slp_point = slp(:,locY,locX)
slp_plot  = gsn_csm_xy(wks,taus,slp_point,t2_res)
```



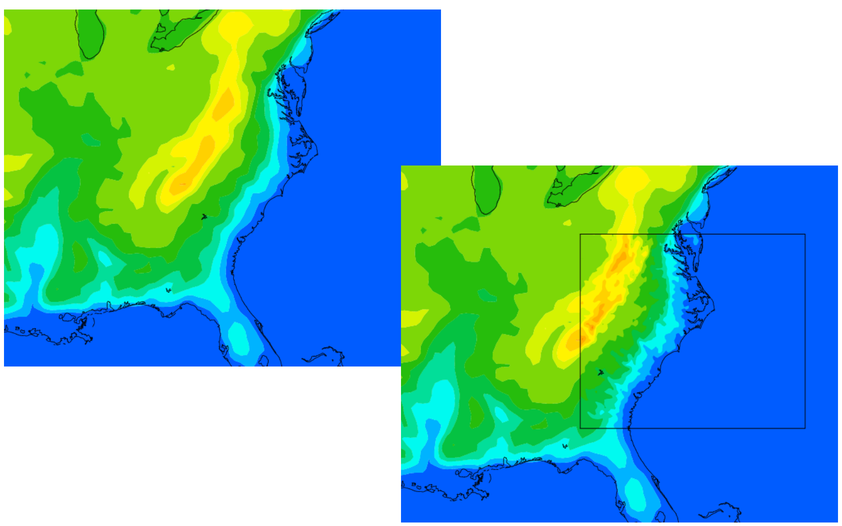
Domain Design

mp = wrf_wps_dom (wks, mpres, lnres, txres)
WPS/util/plotgrids.ncl

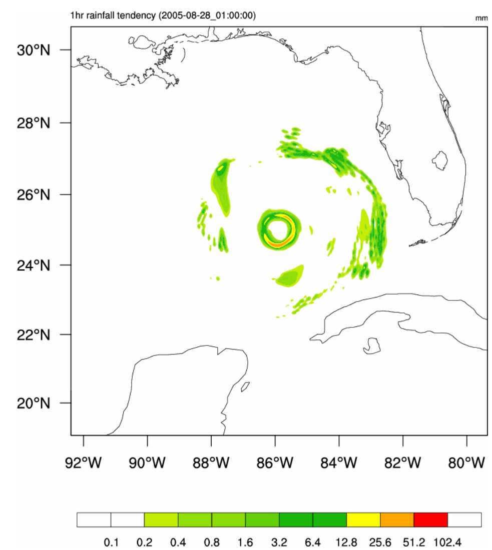
Test Domain



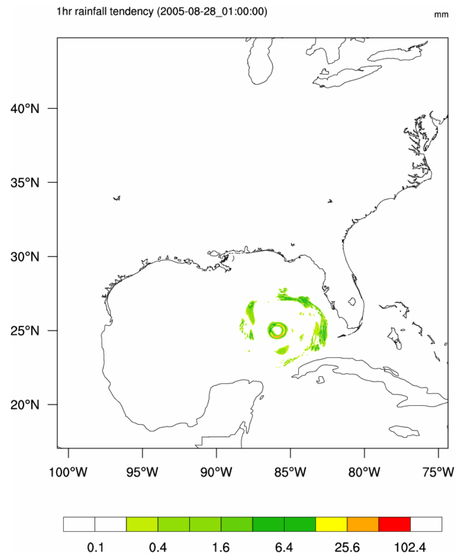
Overlay Domains



Moving Nests



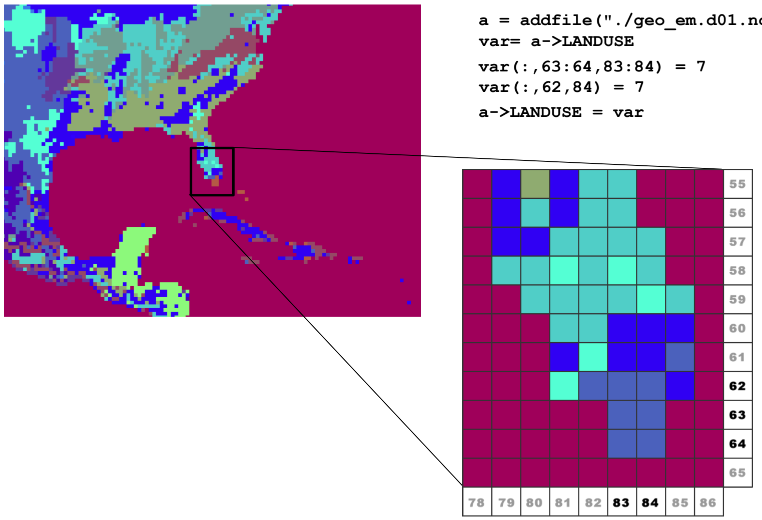
Moving Nests



Change Fields in netCDF File

```
begin

  DATADir = "./"
  FILES = systemfunc (" ls -1 " + DATADir + "met_em.d01* ")
  numFILES = dimsizes(FILES)

  do i=0,numFILES-1
    a = addfile(FILES(i),"w")
    sst = a->SST    ; read the field
    sst = sst + 1   ; change the entire field
    a->SST = sst    ; write the field back to the file
  end do

end
```


Change Fields in netCDF File

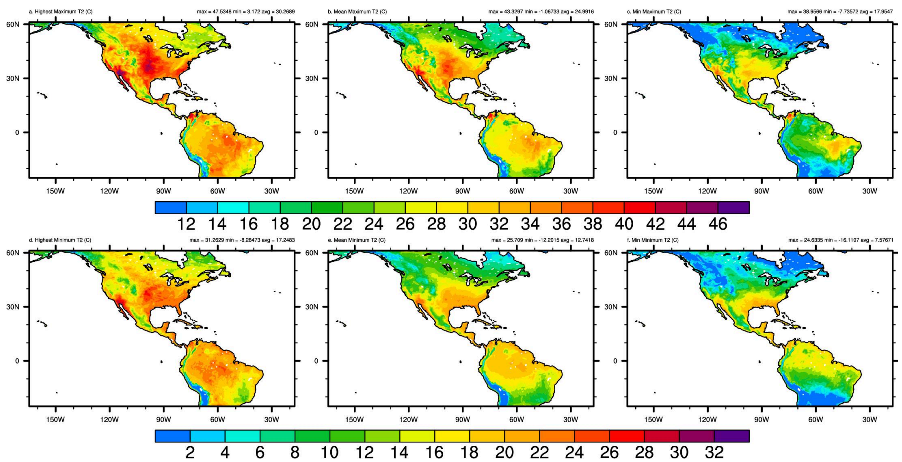



```
a = addfile("./geo_em.d01.nc","w")
var= a->LANDUSE
var(:,63:64,83:84) = 7
var(:,62,84) = 7
a->LANDUSE = var
```


Data Manipulation in NCL

```
begin

    out = addfile("t2_dailymax_1993-08-20.nc","c") ; Create new netCDF file
    filedimdef(out,"Time",-1,True) ; Make Time unlimited

    a = addfile("wrfout_d01_1993-08-20_00:00:00.nc","r") ;File has 24 time steps
    fileattdef(out,a) ; Transfer attributes to new file
    t = a->T2-273.15
    landmask = a->LANDMASK(0,:,:)
    lat = a->XLAT(0,:,:)
    lon = a->XLONG(0,:,:)
    times = a->Times(0,0:9)

    tland = mask(t,landmask,1) ; Mask out the ocean

    tmax = dim_max_n(t,0) ; Daily max
    tlandmax = dim_max_n(tland,0) ; Daily max with ocean masked out

    ; Write attributes for the variable
    tlandmaxday!0 = "Time"
    tlandmaxday!1 = "south_north"
    tlandmaxday!2 = "west_east"
    tlandmaxday@units = "C"
    tlandmaxday@coordinates = "XLONG XLAT"
    tlandmaxday@description = "DAILY MAX TEMP at 2 M (masked)"

    ; Write out data
    out->XLAT = lat
    out->XLONG = lon
    out->LANDMASK = landmask
    out->T2MAX = tlandmaxday

end
```

Data Manipulation in NCL



Reading ASCII Data

```
begin
   fname = "/mydir/ascii.txt"
   foo = asciiread(fname,(/21,6/),"integer")
end
```

```
Variable: foo
Dimensions and sizes:
 [Data|21] x [Columns|6]

(0,0) 1950
(0,1) 13
(0,2) 11
(0,3) 3
(0,4) 8
(0,5) 3
(1,0) 1951
(1,1) 10
```

/mydir/ascii.txt

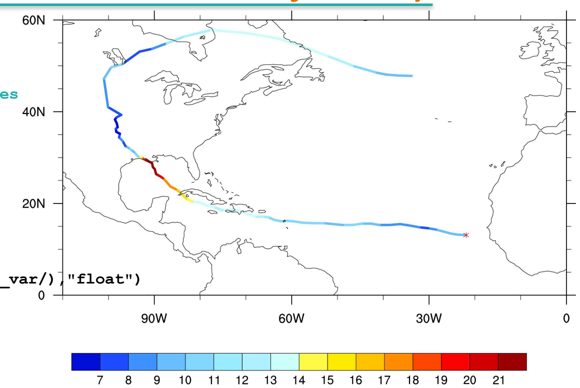
1950	13	11	3	8	3
1951	10	8	3	5	2
1952	7	6	3	3	1
1953	14	6	2	4	1
1954	11	8	6	2	1
1955	12	9	3	6	2
1956	8	4	2	2	1
1957	8	3	1	2	2
1958	10	7	2	5	3
1959	11	7	5	2	1
1960	7	4	2	2	2
1961	11	8	1	7	4
1962	5	3	2	1	0
1963	9	7	5	2	1
1964	12	6	0	6	4
1965	5	4	3	1	1
1966	11	7	4	3	1
1967	8	6	5	1	1
1968	8	5	5	0	0
1969	18	12	7	5	1
1970	10	5	3	2	0

Reading ASCII Data: Trajectory

```
begin

  flnm = "track-1995_0723-0812-380.txt"
  n_col = numAsciiCol(flnm)

  n_var = n_col
  cnLevels = ispan(7, 21, 1)

  wks  = gsn_open_wks("x11", "traj")
  gsn_define_colormap (wks, "GMT_panoply")
  cmap = gsn_retrieve_colormap(wks)

  res = True  ; mapping resources
  res@mpLimitMode = "LatLon"
  res@mpMaxLatF = 60
  res@mpMinLatF = 0
  res@mpMinLonF = -110
  res@mpMaxLonF = 0
  res@mpCenterLonF = -65

  pres = True  ; polyline resources
  pres@gsLineThicknessF = 6.0   ; line thickness

  mres  = True  ; marker resources
  first = True
  first@gsMarkerSizeF = 9.0
  first@gsMarkerColor = "red"
```

19950804	12	31.2	265.6	1013.8	16.3	11.2	9.0	32.8	3.9	-4.2	4.6
19950804	18	31.7	264.9	1013.9	17.2	13.6	10.5	27.8	5.7	-2.6	5.5
19950805	00	32.1	264.4	1011.5	19.7	12.5	11.0	37.5	3.9	2.0	4.9
19950805	06	32.5	263.7	1012.7	16.3	10.5	8.9	36.5	4.9	-1.2	3.7
19950805	12	33.6	263.1	1013.7	13.7	7.7	7.0	27.9	5.3	-2.9	3.5
19950805	18	34.1	262.6	1013.3	14.2	10.0	9.1	28.5	5.6	-2.2	5.2
19950806	00	34.4	262.3	1011.6	17.8	10.2	8.8	32.8	4.4	2.6	4.1
19950806	06	35.2	262.5	1013.6	16.1	7.3	6.5	30.5	4.5	1.6	3.1
19950806	12	35.6	261.7	1015.3	11.8	6.6	5.9	21.4	4.9	-6.2	2.9
19950806	18	36.3	261.6	1015.5	11.7	8.4	7.9	31.1	4.9	-6.8	4.7
19950807	00	36.6	262.0	1013.9	13.2	10.3	7.5	34.2	4.7	-5.5	4.2
19950807	06	37.5	261.8	1016.0	13.2	5.4	3.6	23.7	3.3	-0.3	3.9
19950807	12	38.3	261.4	1016.6	10.3	4.4	3.5	28.0	3.9	-1.9	2.8
19950807	18	38.8	261.8	1016.1	10.0	7.8	7.5	24.5	3.4	-4.3	4.2
19950808	00	39.3	262.8	1014.1	11.8	9.4	7.6	21.2	3.4	-3.3	4.4
19950808	06	41.0	259.9	1014.9	21.1	8.8	8.0	1.9	3.0	5.6	2.8
19950808	12	47.2	259.0	1008.7	17.3	9.1	8.4	20.0	6.1	-11.7	-1.6
19950808	18	49.7	260.6	1007.7	19.7	11.5	10.9	4.0	7.3	-14.1	-1.7
19950809	00	50.3	262.9	1007.2	19.3	12.9	11.1	6.4	5.3	-14.4	-1.4
19950809	06	50.8	263.6	1006.9	18.9	10.7	9.8	16.7	5.6	-15.7	-0.8
19950809	12	53.1	266.8	1006.6	17.4	6.5	5.9	5.1	4.7	-17.8	-0.8
19950809	18	53.7	269.5	1003.1	17.9	8.3	8.3	9.5	5.2	-17.7	-0.6
19950810	00	54.9	272.7	999.4	18.3	12.0	9.5	13.3	4.9	-12.0	-0.8
19950810	06	56.4	276.6	997.2	17.8	14.6	13.0	12.4	3.9	-13.2	-1.9
19950810	12	57.8	282.4	997.3	19.0	21.9	11.1	18.7	1.8	-16.0	0.2
19950810	18	57.2	289.7	994.3	25.6	16.8	15.8	34.7	4.1	-20.6	1.7
19950811	00	56.0	296.9	996.1	30.9	14.6	12.1	41.5	3.9	-19.7	2.8
19950811	06	53.8	303.5	997.1	30.8	16.0	14.3	47.2	6.2	-17.0	1.2

Reading ASCII Data: Trajectory




```
 mres  = True  ; marker resources
 first = True
 first@gsMarkerSizeF = 9.0
 first@gsMarkerColor = "red"

 map = gsn_csm_map_ce(wks,res)

 n_pt = numAsciiRow(flnm)
 data = asciiread(flnm,(/n_pt,n_var/),"float")
 lat = data (:,2)
 lon = data (:,3)
 var = data (:,7)

 do j = 0, n_pt - 2
   ; assign a color based upon an input scalar variable
   pres@gsLineColor = GetFillColor(cnLevels,cmap,avg((/var(j),var(j+1)/)))
   gsn_polyline(wks,map,(/lon(j),lon(j+1)/),(/lat(j),lat(j+1)/),pres)
 end do
 gsn_polymarker(wks,map,lon(0),lat(0),first)  ; draw start of trajectory

 draw(map)
end
```

Writing ASCII Data

```
t2_point  = a->T2(:,locY,locX)
q2_point  = a->Q2(:,locY,locX)

asciiwrite ("t2.txt" , t2_point)
asciiwrite ("q2.txt" , sprintf("%9.3f", q2_point))

npts = dimsizes(t2_point)
data  = new( npts, "string")
  do npt=0,npts-1
    data(npt) = sprintf("%7.1f ", t2_point(npt))
    data(npt) = data(npt) + sprintf("%7.1f ", q2_point(npt))
  end do
asciiwrite ("t2_q2.txt", data)

data2 = new((/npts,2/),"float")
data2(:,0) = t2_point
data2(:,1) = q2_point
write_matrix(data2,"2f7.3",True)
```

write_matrix

Shapefiles and WRF

- A geospatial vector data format for GIS systems software
- We can use it to mask data to specific regional or state borders, rather than drawing a box over an area
- Shapefiles can have three different types of data:
  - Point (locations of cities or places of interest, population data, election data)
  - Polyline (non-closed boundaries like rivers and roads)
  - Polygon (closed geographic boundaries like countries, states, provinces, territories, and lakes)
  - Only one data type per shapefile!

Shapefiles and WRF

```
shp_filename = "cb_2014_us_state_20m.shp"
f = addfile(shp_filename, "r")
print(f)        ; print shapefile metadata

id = f->NAME    ; we know we want the states,
                ; so read and print the metadata

print(id)
```

```
Variable: f
Type: file
filename:       cb_2014_us_state_20m
path:   /glade/p/p66770001/DNV_Ensembles/T2/ShapeFiles/ShapeFiles_US/cb_2014_us_state_20m.shp
   file global attributes:
      layer_name : cb_2014_us_state_20m
      geometry_type : polygon
      geom_segIndex : 0
      geom_numSegs : 1
      segs_xyzIndex : 0
      segs_numPnts : 1
   dimensions:
      geometry = 2
      segments = 2
      num_features = 52  // unlimited
      num_segments = 132
      num_points = 13785
   variables:
      integer geometry ( num_features, geometry )

      integer segments ( num_segments, segments )

      double x ( num_points )

      double y ( num_points )

      double z ( num_points )

      string STATEFP ( num_features )

      string STATENS ( num_features )

      string AFFGEOID ( num_features )

      string GEOID ( num_features )

      string STUSPS ( num_features )

      string NAME ( num_features )

      string LSAD ( num_features )

      double ALAND ( num_features )

      double AWATER ( num_features )
```

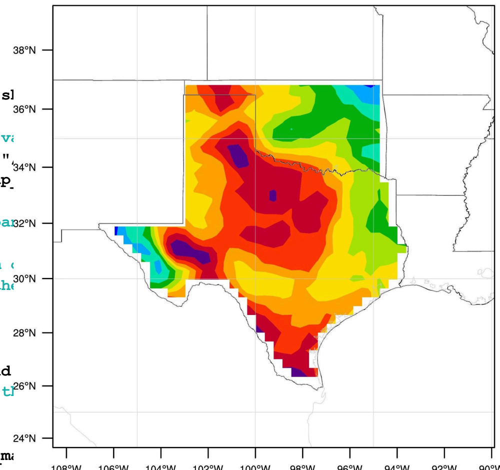
```
Variable: id
Type: string
Total Size: 416 bytes
            52 values
Number of Dimensions: 1
Dimensions and sizes:   [num_features | 52]
Coordinates:
Number Of Attributes: 0
(0)     California
(1)     District of Columbia
(2)     Florida
(3)     Georgia
(4)     Idaho
(5)     Illinois
(6)     Iowa
(7)     Kentucky
(8)     Louisiana
(9)     Maryland
(10)    Michigan
(11)    Minnesota
(12)    Missouri
(13)    New York
(14)    Oregon
(15)    Tennessee
(16)    Texas
(17)    Virginia
(18)    Wisconsin
(19)    Alaska
(20)    Arizona
(21)    Arkansas
(22)    Colorado
(23)    Indiana
(24)    Connecticut
(25)    Hawaii
(26)    Nebraska
(27)    New Mexico
(28)    North Carolina
(29)    Ohio
(30)    Maine
(31)    Massachusetts
(32)    Mississippi
(33)    Montana
(34)    Oklahoma
(35)    South Carolina
(36)    South Dakota
(37)    Utah
(38)    Washington
(39)    West Virginia
(40)    Wyoming
(41)    Delaware
(42)    Rhode Island
(43)    Alabama
(44)    North Dakota
(45)    Pennsylvania
(46)    Vermont
(47)    Puerto Rico
(48)    Kansas
(49)    Nevada
(50)    New Hampshire
(51)    New Jersey
```


Shapefiles and WRF

```
a = addfile("wrfout_d01_2000-07-17_00:00:00.nc","r")
wks = gsn_open_wks("X11","OK_TX")
var = a->T2(0,:,:)
var@lat2d = a->XLAT(0,:,:)
var@lon2d = a->XLONG(0,:,:)
shp_filename = "cb_2014_us_state_20m.shp" ; Shapefile from internet
opt = True
opt@shape_var = "NAME" ; We know the variable name
opt@shape_names = (/"Oklahoma","Texas"/) ; The states we want to mask
var_mask = shapefile_mask_data(var,shp_filename,opt) ; Mask the data
pltres = True
pltres@PanelPlot = True ; We need a panel plot to plot more than one state
mpres = True
mpres@Zoomin = True ; We want to zoom on on our area of interest
mpres@Xstart = 150 ; Grid points of the zoomed in area
mpres@Ystart = 135
mpres@Xend = 200
mpres@Yend = 185
var_mask_zoom = var_mask(y_start:y_end,x_start:x_end) ; Zoomed in variable
; Make contours, draw them on a map, then draw the outline of the polygons
opts = True
opts@cnFillOn = True
contour_mask = wrf_contour(a,wks,var_mask,opts)
plot_mask = wrf_map_overlays(a,wks,contour_mask,pltres,mpres)
id_mask = gsn_add_shapefile_polylines(wks,plot_mask,shp_filename,True)
draw(plot_mask)
frame(wks)
```

Shapefiles and WRF

```
a = addfile("wrfout_d01_2000-07-17_00:00:00.nc","r")
wks = gsn_open_wks("X11","OK_TX")
var = a->T2(0,:,:)
var@lat2d = a->XLAT(0,:,:)
var@lon2d = a->XLONG(0,:,:)
shp_filename = "cb_2014_us_state_20m.sl
opt = True
opt@shape_var = "NAME"  ; We know the va
opt@shape_names = (/"Oklahoma","Texas"
var_mask = shapefile_mask_data(var,shp_
pltres = True
pltres@PanelPlot = True  ; We need a pa
mpres = True
mpres@Zoomin = True  ; We want to zoom
mpres@Xstart = 150  ; Grid points of th
mpres@Ystart = 135
mpres@Xend = 200
mpres@Yend = 185
var_mask_zoom = var_mask(y_start:y_end
; Make contours, draw them on a map, th
opts = True
opts@cnFillOn = True
contour_mask = wrf_contour(a,wks,var_ma
plot_mask = wrf_map_overlays(a,wks,contour_mask,pltres,mpres)
id_mask = gsn_add_shapefile_polylines(wks,plot_mask,shp_filename,True)
draw(plot_mask)
frame(wks)
```



Geo Referenced Graphics


```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/wrf/WRFUserARW.ncl"

begin
 a = addfile("./wrfout_d01_2012-09-28_00:00:00.nc","r")
 wks = gsn_open_wks("X11","plt_Surface")

 T2 = wrf_user_getvar(a,"T2",-1)
 times = wrf_user_getvar(a,"times",-1)
 ntimes = dimsizes(times)

 mpres = True
 pltres = True

 do it=0,ntimes-1
  opts = True
  opts@cnFillOn = True
  contour_t2 = wrf_contour(a,wks,T2(it,:,:),opts)
  plot = wrf_map_overlays(a,wks,(/contour_t2/),pltres,mpres)

 end do

 end
```



```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/wrf/WRFUserARW.ncl"
load "$VAPOR_HOME/share/examples/NCL/wrf2geotiff.ncl"

begin
 a = addfile("./wrfout_d01_2012-09-28_00:00:00.nc","r")
 wks = gsn_open_wks("ps","plt_Surface")
 wrf2gtiff = wrf2geotiff_open(wks)

 T2 = wrf_user_getvar(a,"T2",-1)
 times = wrf_user_getvar(a,"times",-1)
 ntimes = dimsizes(times)

 mpres = True
 pltres = True
 pltres@gsnFrame = False

 do it=0,ntimes-1
  opts = True
  opts@cnFillOn = True
  contour_t2 = wrf_contour(a,wks,T2(it,:,:),opts)
  plot = wrf_map_overlays(a,wks,(/contour_t2/),pltres,mpres)
  wrf2geotiff_write(wrf2gtiff,a,times(it), wks, plot, False)
  frame(wks)
 end do

 wrf2geotiff_close(wrf2gtiff, wks)
 end
```

Linking NCL to 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
  - Example: WPS/utils/plotfmt.exe
  - *Older way of creating plots – not recommended*

Linking NCL to Fortran/C Code

- Easier to use F77 code, but works with F90 code
- Need to isolate definition of input variables and wrap them with special comment statements:

```
C   NCLFORTSTART
C   NCLEND
```

- Use a tool called **`WRAPIT`** to create a `*.so` file

```
> WRAPIT myTK.f
```

- Load the `*.so` file into the NCL script with "**`external`**" statement
- Call Fortran function with special "`::`" syntax
- You must pre-allocate for arrays!

Linking NCL to Fortran/C Code: myTK.f

```
C NCLFORTSTART
      subroutine compute_tk (tk,pressure,theta, nx, ny, nz)
         implicit none
         integer nx,ny,nz
         real pi, tk(nx,ny,nz)
         real pressure(nx,ny,nz), theta(nx,ny,nz)
C NCLEND
         integer i,j,k

         do k=1,nz
          do j=1,ny
          do i=1,nx
            pi=(pressure(i,j,k) / 1000.)**(287./1004.)
            tk(i,j,k) = pi*theta(i,j,k)
          enddo
          enddo
         enddo

         end
```


myTK.so – *Create & use in NCL Script*

```
% WRAPIT myTK.f
```

This will create a "**`myTK.so`**" file

```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
load "$NCARG_ROOT/lib/ncarg/nclscripts/wrf/WRFUserARW.ncl"
external myTK "./myTK.so"

begin
    t = wrf_user_getvar(a,"T",5)
    t = t + 300
    p = wrf_user_getvar(a,"pressure",5)

    ; Must preallocate space for output arrays
    dim = dimsizes(t)
    tk = new( dimsizes(t), typeof(t) )

    ; Remember, Fortran/NCL arrays are ordered differently
        myTK :: compute_tk (tk,p,t,dim(2),dim(1),dim(0))
end
```

FORTRAN 90 Code

- Can use simple FORTRAN 90 code
- Your FORTRAN 90 program may not contain any of the following features:
  - pointers or structures as arguments,
  - missing/optional arguments,
  - keyword arguments, or
  - recursive procedure.

Compiling with NCL

```
In function `write_png':
  undefined reference to `png_create_write_struct'
  undefined reference to `png_create_info_struct'
  undefined reference to `png_destroy_write_struct'
  undefined reference to `png_destroy_write_struct'
```

```
-L<path_to_png_lib> -lpng -L<path_to_z_lib> -lz
```

```
/usr/local/ncl/lib/libncarg.a(agcurv.o): In function `agcurv_':
agcurv.f:(.text+0x69): undefined reference to `_gfortran_copy_string'
/usr/local/ncl/lib/libncarg.a(aggtch.o): In function `aggtch_':
aggtch.f:(.text+0x3e): undefined reference to `_gfortran_copy_string'
aggtch.f:(.text+0x7b): undefined reference to `_gfortran_copy_string'
```

```
-L<path_to_gfortran_lib> -lgfortran
```


WRF-Python

- A collection of diagnostic and interpolation routines for use with WRF-ARW
- Functionality is very similar to what is provided by the WRF NCL functions
- When coupled with either matplotlib or PyNGL you can create plots very similar to what you make with NCL

https://github.com/NCAR/wrf-python

Practice Session

- If you want to use an NCL script that needs a wrfout file with multiple time steps:

ncrcat wrfout* wrfout_all.nc

NCL Support

- wrfhelp@ucar.edu

all questions about WRF and NCL

- ncl-talk@ucar.edu

generic NCL questions

WRF Physics

Jimy Dudhia

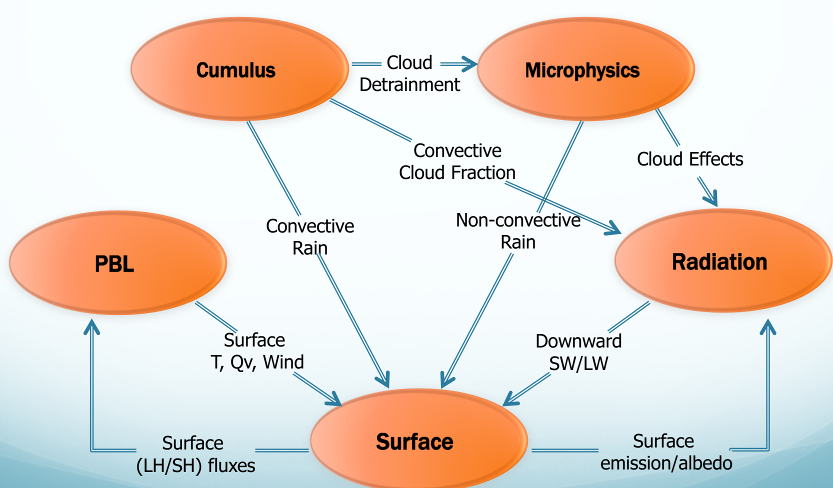
Overview of WRF Physics

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)

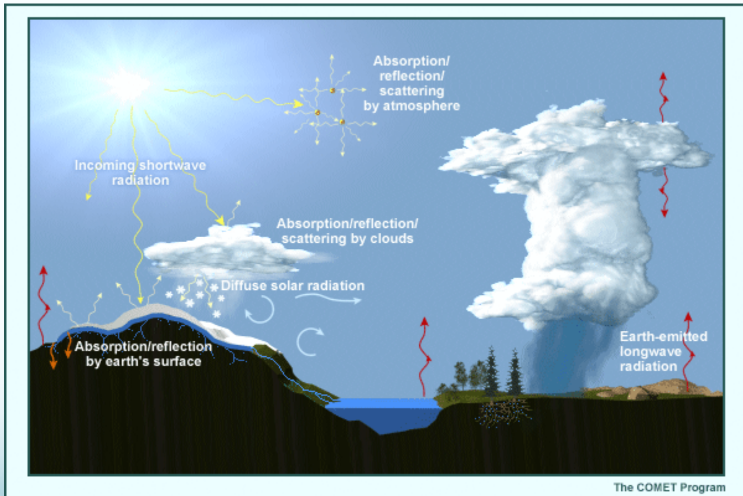
Direct Interactions of Parameterizations



Radiation

Provides
Atmospheric temperature tendency profile
Surface radiative fluxes

Free Atmosphere Radiation Processes



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

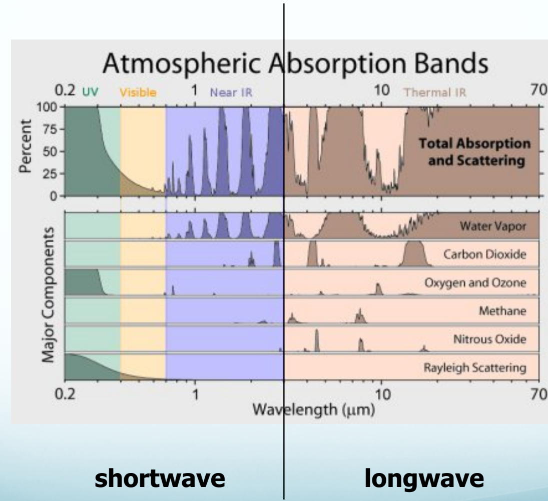
Longwave Radiation schemes

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
- O3 – schemes have own climatologies
  - CAM has monthly, zonal, pressure-level data and RRTMG has this as an option
  - Others use single profiles (Goddard has 5 profiles to choose from)

Radiation Effects in Clear Sky



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
  - Some microphysics options pass own particle sizes to RRTMG radiation: 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 (improved in V3.8)
- 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)
- Different WRF schemes may use different cloud overlapping assumption. For example, RRTMG, CAM 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

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/mont h
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 and Aerosols

- Main gas 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 options
    - aer_opt=1 Tegen (EC) global monthly climatology
    - aer_opt=2 user-specified properties and/or AOD map
    - aer_opt=3 Thompson microphysics nuclei (V3.8)

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

- Many schemes use multiple spectral bands
  - As with longwave, bands are ranges of wavelengths usually dominated by different gases
- Look-up tables
  - Also as with longwave

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

- Available for all shortwave options
- Represents effect of slope on surface solar flux accounting for diffuse/direct effects
- Two levels of detail (namelist options):
  - slope_rad: activates slope effects - may be useful for complex topography and grid lengths < 2 km.
    - topo_shading: shading of neighboring grids by mountains - may be useful for grid lengths < 1 km.

radt

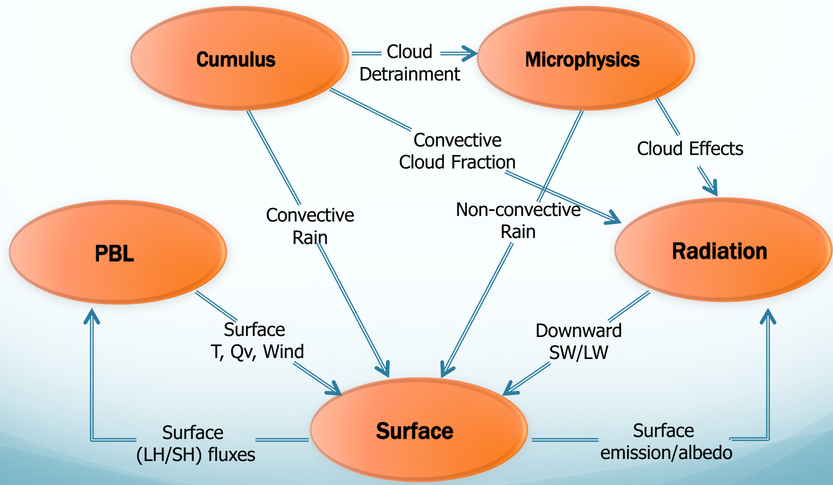
Radiation time-step recommendation

- 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

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

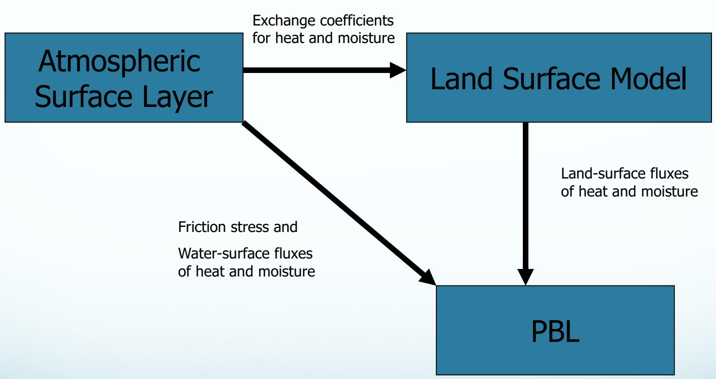
Direct Interactions of Parameterizations



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

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

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

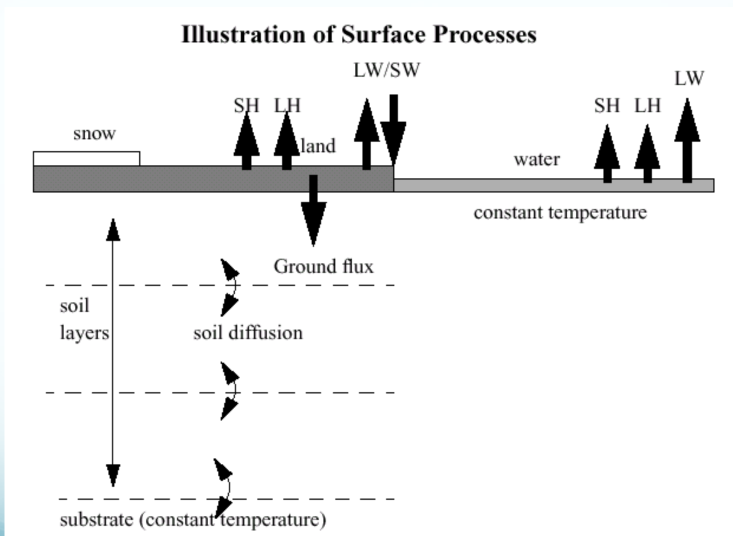
- Ocean Mixed Layer Model (sf_ocean_physics=1)
  - 1-d slab ocean mixed layer (specified initial depth)
  - Includes wind-driven ocean mixing for SST cooling feedback
- 3d PWP ocean (Price et al.) (sf_ocean_physics=2)
  - 3-d multi-layer (~100) ocean, salinity effects
  - Fixed depth
- Alternative surface-layer 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

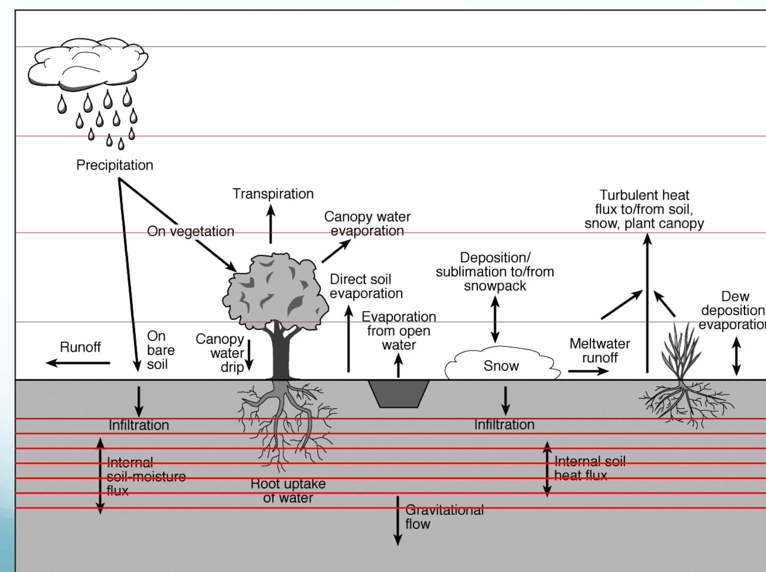
- fractional_seaice=1 - with input sea-ice fraction data can partition land/water fluxes within a grid box
- Can be used with nearly all surface-layer schemes

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, 10 for CLM4)
- Predicts snow water equivalent on ground. May be in layers (NoahMP, RUC, SSiB,CLM4)
- May predict canopy moisture only (Noah, RUC) or temperature only (SSiB) or both (NoahMP, CLM4)

Land-Surface Options

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

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,CLM4)

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
- MPTABLE.TBL used by NoahMP
- 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)

Initializing LSMs

- 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 grid cell

sst_update=1

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_d0*n* created by *real*
- Sea-ice can be updated
- 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 – ability to output max/min/mean/std of surface fields in a specified period

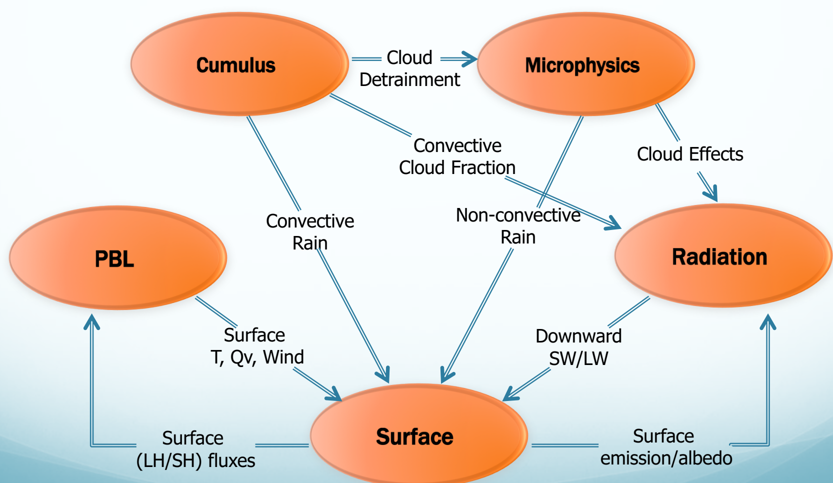
Lake Model

- 10-layer lake model from CLM (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
- WPS preprocessing allows diurnal averaging methods to initialize lake temperatures where not resolved by SST analysis (TAVGSFC)

WRF-Hydro

- Coupling to hydrological model available
- Streamflow prediction, etc.
- Sub-grid tiling to ~100 m grid
- Requires special initialization for hydrological datasets

Direct Interactions of Parameterizations

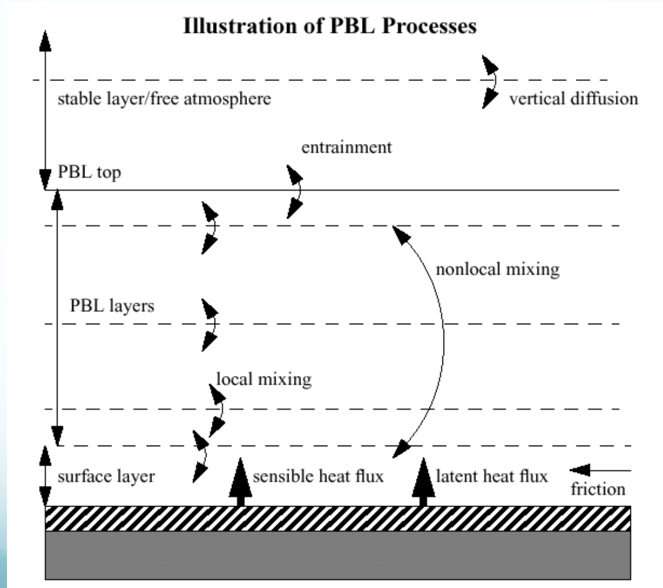


Planetary Boundary Layer

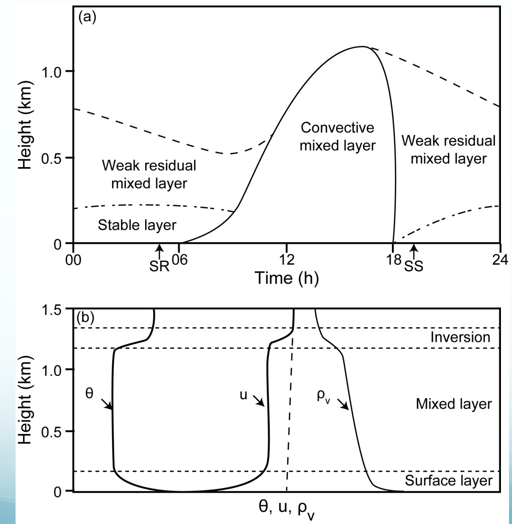
Provides

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

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

Nonlocal PBL schemes

Non-local schemes have two main components

$$\overline{w'\phi'}^{\Delta} = \underbrace{-K_{\phi}\frac{\partial \overline{\phi}^{\Delta}}{\partial z}}_{(1)} + \underbrace{F^{NL}_{w\phi}}_{(2)}$$

(1) Term for local (L) transport by small eddies

(2) Term for nonlocal (NL) transport by large eddies

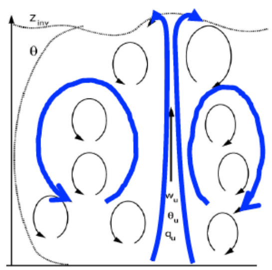


FIG. 1. Sketch of a convective updraft embedded in a turbulent eddy structure.

Explicitly included in **nonlocal PBL parameterizations**

(i.e., Mass-flux term or counter-gradient gamma)

Figure is taken from Siebesma et al. (2007, JAS)

5

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 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

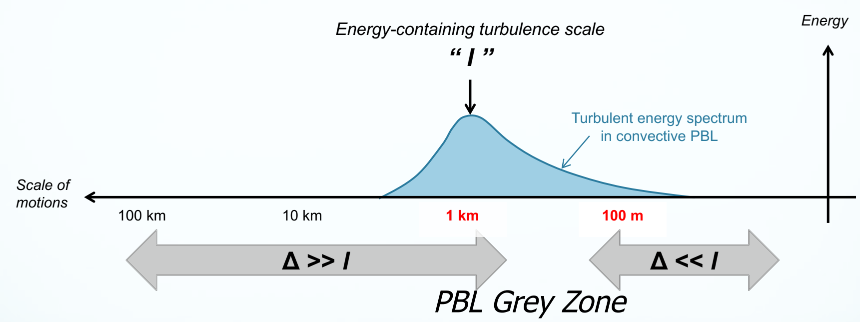
PBL schemes

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 (coupled to surface physics)
  - Works best when dx and dz are comparable

Model Grid Spacing: PBL and LES



For coarse grid spacing

- ✓ **PBL schemes have been designed for Δ >> l**
- ✓ ***All eddies are sub-grid***
- ✓ ***1d column schemes handle sub-grid vertical fluxes***

For fine grid spacing

- ✓ **LES schemes have been designed for Δ << l**
- ✓ ***All major eddies are resolved***
- ✓ ***3d turbulence schemes handle sub-grid 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 (gamma) term reduces in strength as grid size gets smaller and resolved mixing increases
- Other schemes may work in this range but will not have correctly partitioned resolved/sub-grid energy fractions

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

- For YSU
  - topo_wind=1,2: wind-bias correction methods for terrain effects
  - ysu_topdown_pblmix=1: cloud-top cooling-driven mixing
- For MYNN
  - Wind-farm model has been added to investigate wind-farm effects on the environment (extra stress and turbulence generation)
- Gravity-wave drag can be added for low resolution (> 5 km) runs to represent sub-grid orographic gravity-wave vertical momentum transport (gwd_opt=1)
- Fog: grav_settling=2 (Katata)

bldt

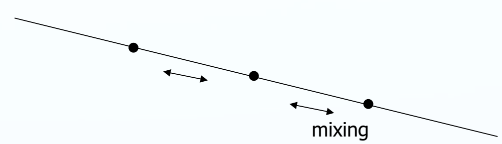
- Minutes between boundary layer/LSM calls
- Typical value is 0 (every step)
- CLM LSM is expensive, so may consider bldt in that case

Turbulence/Diffusion

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

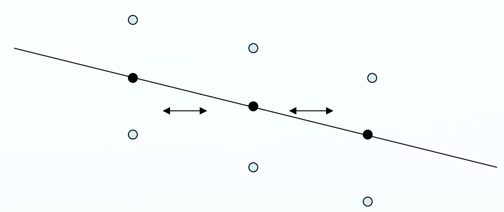
$$\frac{\partial}{\partial x} K_h \frac{\partial}{\partial x}\theta + \frac{\partial}{\partial y} K_h \frac{\partial}{\partial y}\theta + \frac{\partial}{\partial z} K_v \frac{\partial}{\partial z}\theta$$

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

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)

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)

Diffusion Option Choice

- 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)

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)

Upper damping (damp_opt)

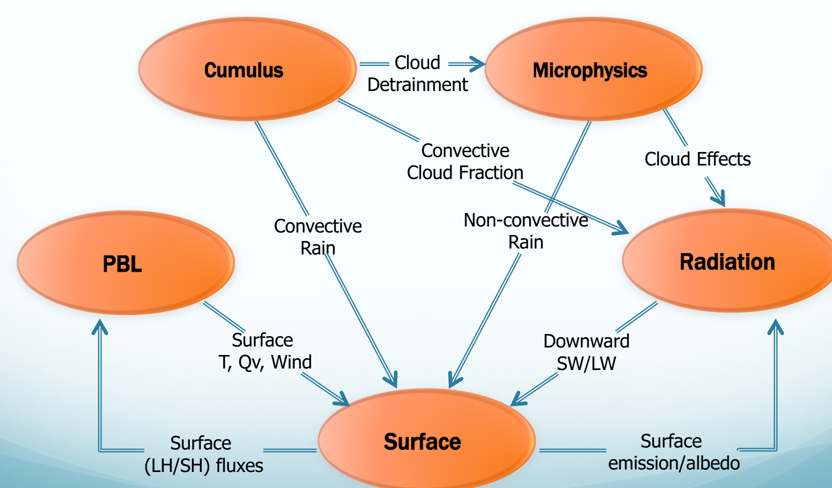
Purpose is to prevent unrealistic reflections of waves from model top. Can be important over high topography.

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

Direct Interactions of Parameterizations



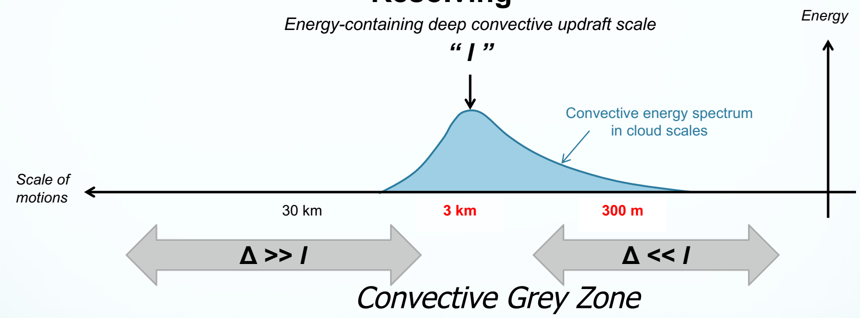
Cumulus Parameterization

Provides

Atmospheric heat and moisture/cloud tendency profiles

Surface sub-grid-scale (convective) rainfall

Model Grid Spacing: Cumulus Parameterization and Cloud-Resolving



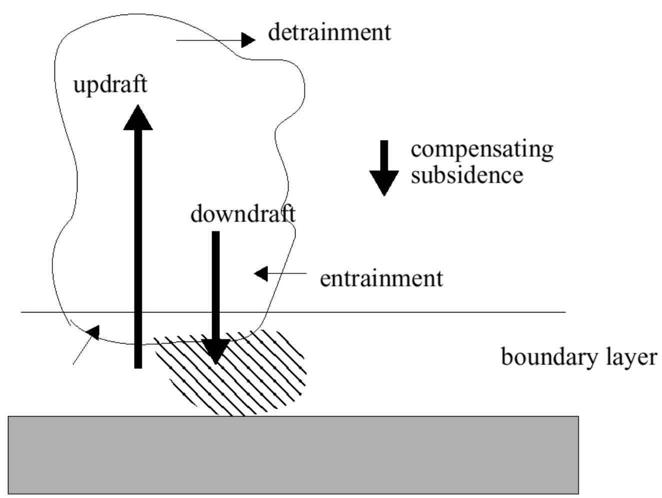
For coarse grid spacing

- ✓ **Cumulus parameterization schemes have been designed for Δ >> l**
- ✓ ***All updrafts and downdrafts are sub-grid***
- ✓ ***1d column schemes handle sub-grid vertical fluxes***

For fine grid spacing

- ✓ **Resolved dynamics and microphysics work for Δ << l**
- ✓ ***Updrafts and downdrafts are resolved***
- ✓ ***PBL and/or diffusion schemes handle local sub-grid mixing***

Illustration of Cumulus Processes



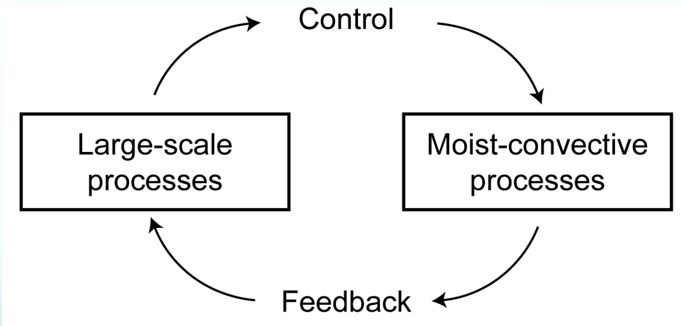
Cumulus Schemes

- Use for grid columns that completely contain convective clouds (typically dx > 10 km)
- 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 environmental subsidence around clouds
  - Note: schemes have no net mass flux – subsidence compensates cloud mass fluxes exactly
  - Environmental subsidence around cloud warms and dries troposphere removing instability over time
    - Dynamics may produce mean vertical motion in grid cell in response to scheme's heating profile
- 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

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
10	KF CuP	Berg and Stull (2004, 2005, JAS)	2016
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, ZM, Tiedtke, BMJ)
    - Quasi-equilibrium (Arakawa-Schubert) with large-scale destabilization d(CAPE)/dt (SAS, NSAS)
    - 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

Radiation Interaction

- The Grell schemes, KF and MSKF interact using cu_rad_feedback=1 which allows them to provide a cloud fraction and amount in active grid columns
- Zhang-McFarlane is part of the CESM suite that also provides a part of the cloud fraction (used with CESM (or MG) microphysics and UW shallow scheme)
- If using the GFDL radiation scheme there is a Slingo method of cloud fraction from many schemes using precip rate, top and bottom, to compute a cloud fraction

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

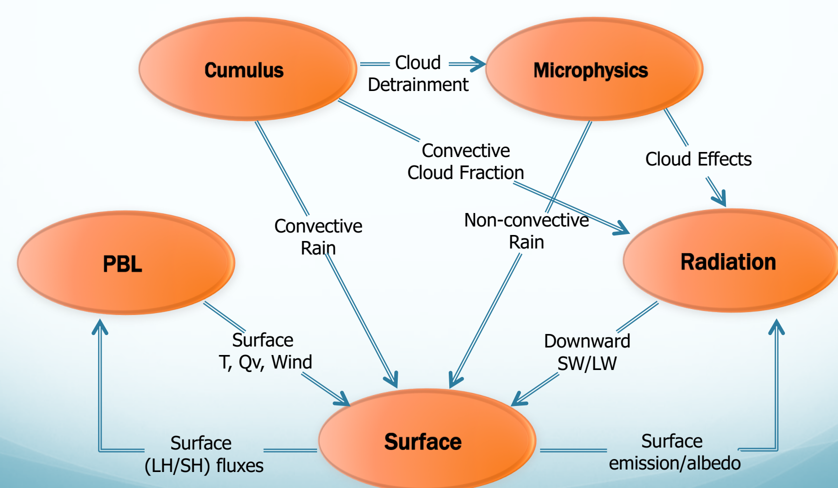
Cumulus schemes

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

Cumulus scheme: Recommendations

- dx ≥ 10 km:
  - probably need cumulus scheme
  - These release instability gradually (prevent grid-point storms)
- 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
- 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 and MSKF 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

Direct Interactions of Parameterizations



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

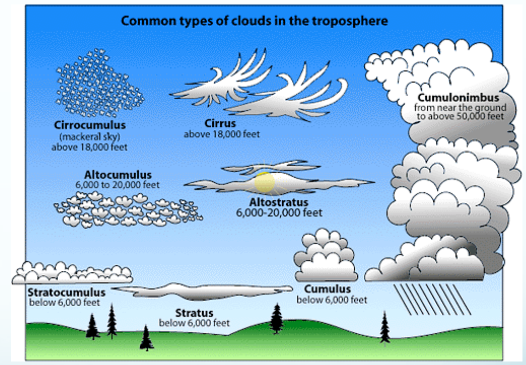
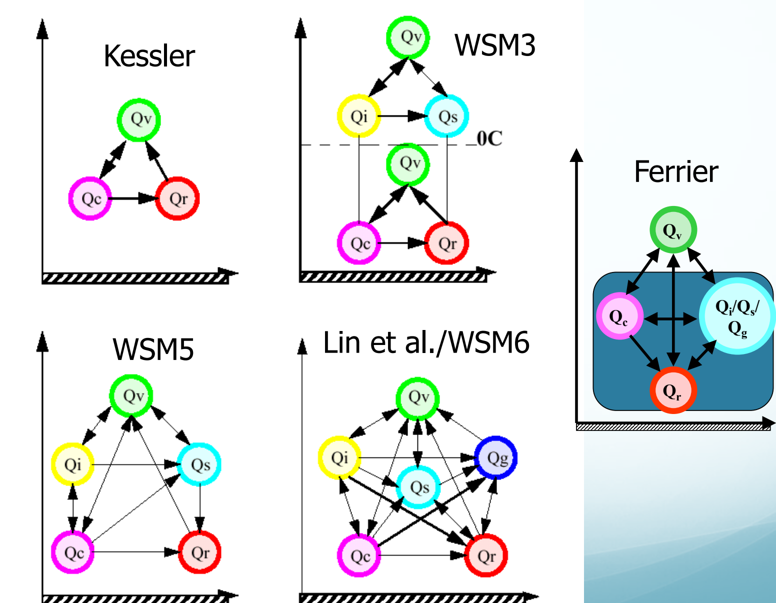


Illustration of Microphysics Processes



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

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	Morrison et al. (2009, 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

mp_physics	Scheme	Reference	Added
19	NSSL 7-class	Mansell, Ziegler and Bruning (2010, JAS)	2013
21	NSSL 6-class	Gilmore, Straka and Rasmussen (2004, MWR)	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	Khain, Lynn and Dudhia (2010, JAS)	2014
32	SBM full	Khain et al. (2004, JAS)	2014
50	P3	Morrison and Milbrandt (2015, JAS)	2017
51	P3-nc	Morrison and Milbrandt (2015, JAS)	2017

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

- 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 (fast scheme) or 8x32 (full scheme) 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

Particle Densities

- Some schemes allow variable densities especially for riming rather than discrete densities for snow, graupel and hail
  - WSM6/WDM6 schemes simply combine snow and graupel for purposes of computing fallspeed – rimed fraction is qg/(qs+qg)
  - NSSL schemes compute volume of graupel as a density variable
  - P3 computes growth by riming and deposition to compute density for ice/snow/graupel combined particles

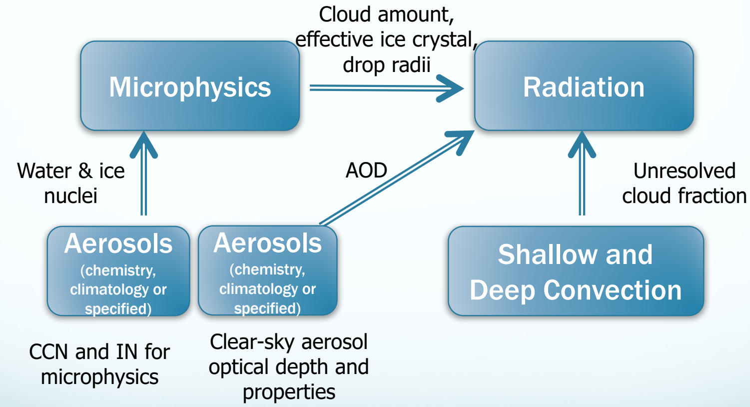
Interaction with Aerosols

- 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 can use its own aerosols (water and ice nuclei) initialized from climatology, advected

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
  - This represents so-called indirect effect on radiation due to drop size variation
  - Other schemes do not and radiation uses internal assumptions about particle sizes

Cloud-Aerosol-Radiation Interaction



Note: aerosols not always unified (CCN and AOD may come from different sources)

Microphysics schemes

* 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

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
50	P3	ARW	Qc Qr Qi	Nr Ni Ri Bi
51	P3-nc	ARW	Qc Qr Qi	Nc Nr Ni Ri Bi

- Nn = CCN number
- VOLg = graupel volume
- Ri = rimed ice mass   Bi = rimed ice volume

Microphysics Options: Recommendations

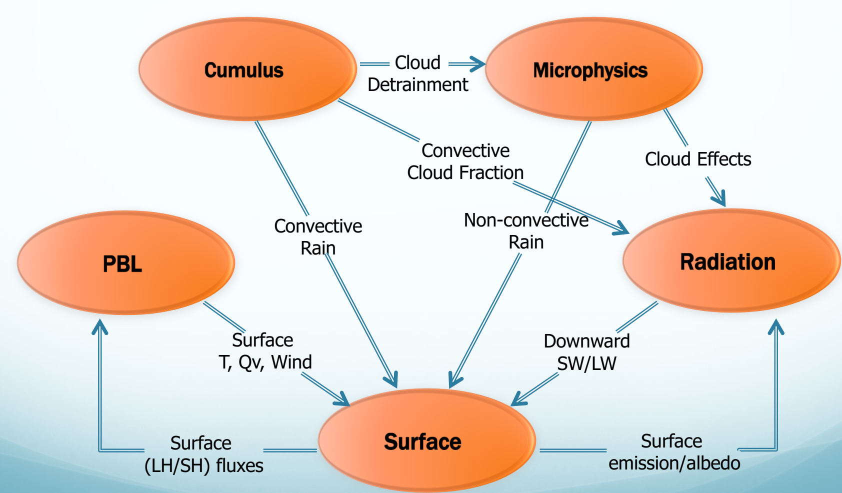
- Probably not necessary to use a graupel scheme for dx > 10 km
  - Updrafts producing graupel not resolved
  - Cheaper scheme may give similar results
- When resolving individual updrafts, graupel scheme should be used
- All domains use same option

Rainfall Output

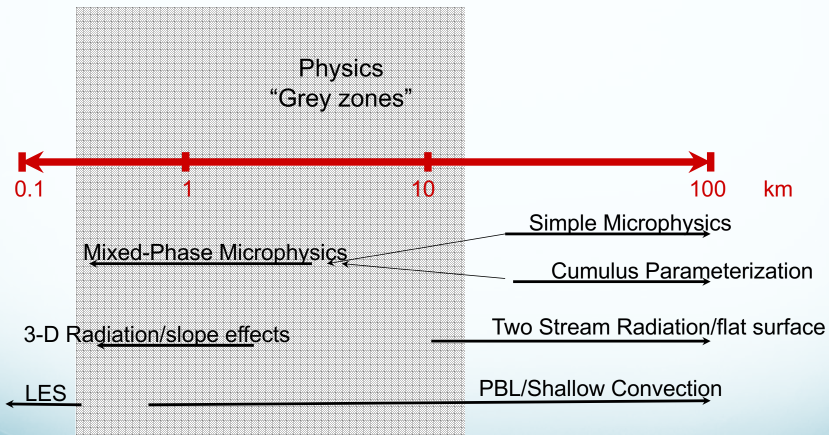
- Cumulus and microphysics can be run at the same time
- ARW outputs rainfall accumulations since simulation start time (0 hr) in mm
- RAINC comes from cumulus scheme
- RAINNC comes from microphysics scheme
- Total is RAINC+RAINNC
  - RAINNCV is time-step value
  - SNOWNC/SNOWNCV are snow sub-set of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

Physics Interactions

Direct Interactions of Parameterizations



Physics in Multiscale NWP Model

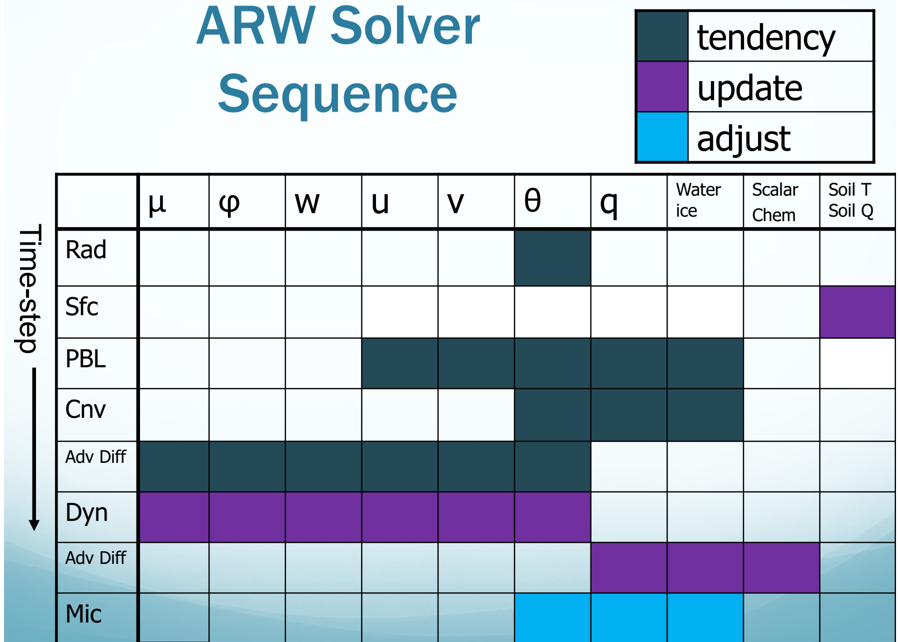


Solver Calling Sequence (ARW example)

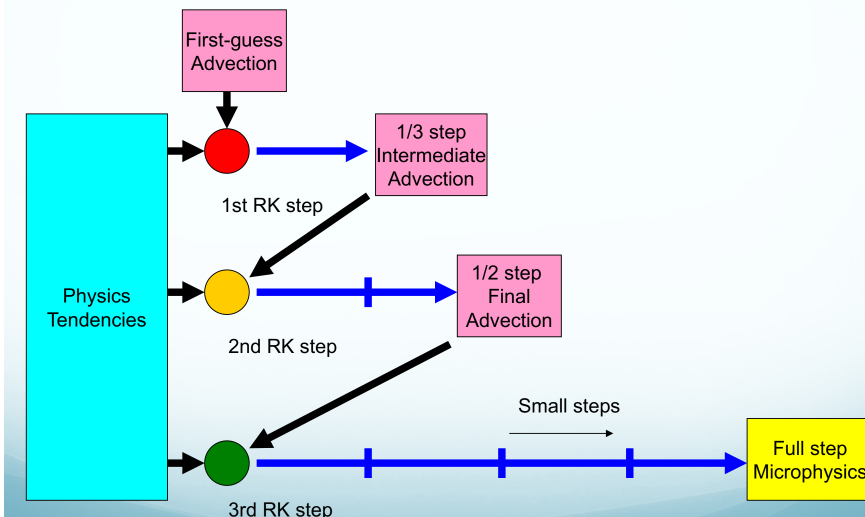
Call to solver advances one domain by one model time-step

- 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 (namelist.input)

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

WRF-ARW Dynamics Solver

Bill Skamarock

Dynamics: Introduction

The Advanced Research WRF (ARW) Dynamics Solver

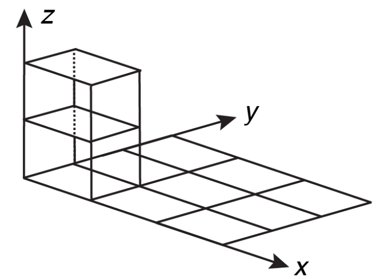
1. What is a dynamics solver?
2. Variables and coordinates
3. Equations
4. Time integration scheme
5. Grid staggering
6. Advection (transport) and conservation
7. Time step parameters
8. Filters
9. Map projections and global configuration
10. Boundary condition options

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

Dynamics: 1. What is a *dynamics solver?*

A *dynamical solver* (or a *dynamical core*, or *dycore*) performs a time (t) and space (x,y,z) integration of the equations of motion.

Given the 3D atmospheric state at time t, $S(x,y,z,t)$, we integrate the equations forward in time from $t \longrightarrow T$, i.e. we run the model and produce a forecast.



The equations cannot be solved analytically, so we *discretize* the equations on a *grid* and compute *approximate* solutions.

The accuracy of the solutions depend on the numerical method and the mesh spacing (grid).

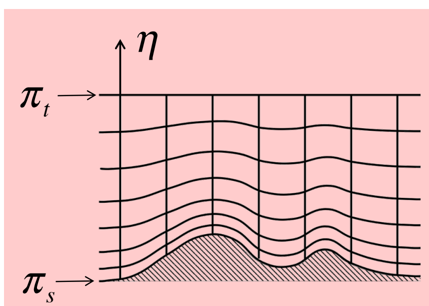
Dynamics: 2. Variables and coordinates

Vertical coordinates: (1) Traditional terrain-following mass coordinate

Dry hydrostatic pressure π_d

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

Vertical coordinate $\eta = \dfrac{(\pi_d - \pi_t)}{\mu_d}$



Layer mass (per unit area) $\mu_d \Delta\eta = \Delta\pi_d = -g\rho_d \Delta z$, Pressure $\pi_d(\eta) = \eta\mu_d + \pi_t$

Dynamics: 2. Variables and coordinates

Vertical coordinates: (2) Hybrid terrain-following mass coordinate

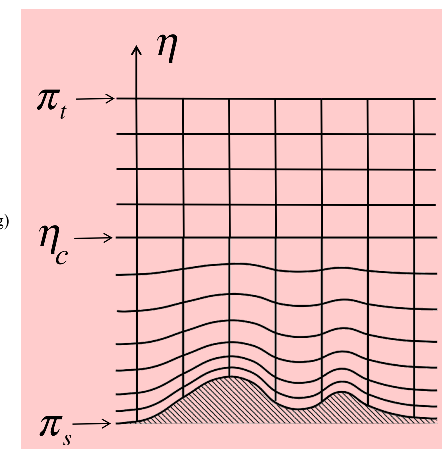
Isobaric coordinate (constant pressure):

$$\eta = \frac{\pi_d}{\pi_0 - \pi_t}$$

Hybrid terrain-following coordinate:

$$\pi_d(\eta) = B(\eta)\mu_d + \pi_t \text{ (Terrain-following)}$$
$$+[\eta - B(\eta)](\pi_0 - \pi_t) \text{ (Isobaric)}$$

η_c level at which B $\rightarrow$ 0, i.e. transition between isobaric and terrain-following coordinate.



Dynamics: 2. Variables and coordinates

Variables:

Grid volume mass (per unit area):

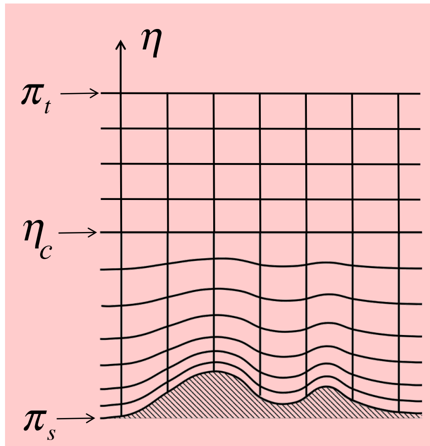
$$\mu_d = \frac{\partial \pi_d}{\partial \eta} = B_\eta(\pi_s - \pi_t) + (1 - B_\eta)(\pi_0 - \pi_t)$$

Conserved state (prognostic) variables:

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

Non-conserved state variable:

$$\phi = gz$$



Dynamics: 2. Variables and coordinates

Vertical momentum eqn.

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

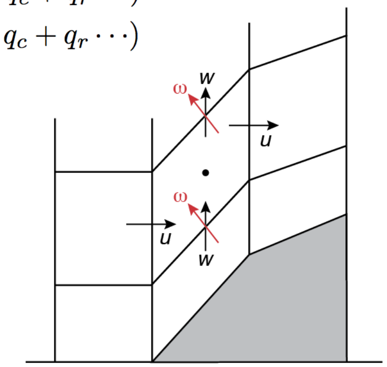
Subscript d denotes *dry*, and

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

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. Equations

$$\frac{\partial U}{\partial t} =$$
$$\frac{\partial V}{\partial t} =$$
$$\frac{\partial W}{\partial t} =$$
$$\frac{\partial \mu_d}{\partial t} =$$
$$\frac{\partial \Theta}{\partial t} =$$
$$\frac{\partial \mu_d q_j}{\partial t} =$$
$$\frac{\partial \phi}{\partial t} =$$

Dynamics: 3. Equations

transport

$$\frac{\partial U}{\partial t} = -\frac{\partial U u}{\partial x} - \frac{\partial V u}{\partial y} - \frac{\partial \Omega u}{\partial \eta}$$
$$\frac{\partial V}{\partial t} = -\frac{\partial U v}{\partial x} - \frac{\partial V v}{\partial y} - \frac{\partial \Omega v}{\partial \eta}$$
$$\frac{\partial W}{\partial t} = -\frac{\partial U w}{\partial x} - \frac{\partial V w}{\partial y} - \frac{\partial \Omega w}{\partial \eta}$$
$$\frac{\partial \mu_d}{\partial t} = -\frac{\partial U}{\partial x} - \frac{\partial V}{\partial y} - \frac{\partial \Omega}{\partial \eta}$$
$$\frac{\partial \Theta}{\partial t} = -\frac{\partial U \theta}{\partial x} - \frac{\partial V \theta}{\partial y} - \frac{\partial \Omega \theta}{\partial \eta}$$
$$\frac{\partial \mu_d q_j}{\partial t} = -\frac{\partial U q_j}{\partial x} - \frac{\partial V q_j}{\partial y} - \frac{\partial \Omega q_j}{\partial \eta}$$
$$\frac{\partial \phi}{\partial t} = -u\frac{\partial \phi}{\partial x} - v\frac{\partial \phi}{\partial y} - \omega\frac{\partial \phi}{\partial \eta}$$

Dynamics: 3. Equations

transport | pressure gradient

$$\frac{\partial U}{\partial t} = -\frac{\partial Uu}{\partial x} - \frac{\partial Vu}{\partial y} - \frac{\partial \Omega u}{\partial \eta} - \alpha\mu_d\frac{\partial p}{\partial x} - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\frac{\partial \phi}{\partial x}$$
$$\frac{\partial V}{\partial t} = -\frac{\partial Uv}{\partial x} - \frac{\partial Vv}{\partial y} - \frac{\partial \Omega v}{\partial \eta} - \alpha\mu_d\frac{\partial p}{\partial y} - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\frac{\partial \phi}{\partial y}$$
$$\frac{\partial W}{\partial t} = -\frac{\partial Uw}{\partial x} - \frac{\partial Vw}{\partial y} - \frac{\partial \Omega w}{\partial \eta} - g\left(\mu_d - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\right)$$
$$\frac{\partial \mu_d}{\partial t} = -\frac{\partial U}{\partial x} - \frac{\partial V}{\partial y} - \frac{\partial \Omega}{\partial \eta}$$
$$\frac{\partial \Theta}{\partial t} = -\frac{\partial U\theta}{\partial x} - \frac{\partial V\theta}{\partial y} - \frac{\partial \Omega\theta}{\partial \eta}$$
$$\frac{\partial \mu_d q_j}{\partial t} = -\frac{\partial Uq_j}{\partial x} - \frac{\partial Vq_j}{\partial y} - \frac{\partial \Omega q_j}{\partial \eta}$$
$$\frac{\partial \phi}{\partial t} = -u\frac{\partial \phi}{\partial x} - v\frac{\partial \phi}{\partial y} - \omega\frac{\partial \phi}{\partial \eta}$$

Dynamics: 3. Equations

transport | pressure gradient

$$\frac{\partial U}{\partial t} = -\frac{\partial Uu}{\partial x} - \frac{\partial Vu}{\partial y} - \frac{\partial \Omega u}{\partial \eta} - \alpha\mu_d\frac{\partial p}{\partial x} - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\frac{\partial \phi}{\partial x} + R_u + Q_u$$
$$\frac{\partial V}{\partial t} = -\frac{\partial Uv}{\partial x} - \frac{\partial Vv}{\partial y} - \frac{\partial \Omega v}{\partial \eta} - \alpha\mu_d\frac{\partial p}{\partial y} - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\frac{\partial \phi}{\partial y} + R_v + Q_v$$
$$\frac{\partial W}{\partial t} = -\frac{\partial Uw}{\partial x} - \frac{\partial Vw}{\partial y} - \frac{\partial \Omega w}{\partial \eta} - g\left(\mu_d - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\right) + R_w + Q_w$$
$$\frac{\partial \mu_d}{\partial t} = -\frac{\partial U}{\partial x} - \frac{\partial V}{\partial y} - \frac{\partial \Omega}{\partial \eta}$$
$$\frac{\partial \Theta}{\partial t} = -\frac{\partial U\theta}{\partial x} - \frac{\partial V\theta}{\partial y} - \frac{\partial \Omega\theta}{\partial \eta} + R_\theta + Q_\theta$$
$$\frac{\partial \mu_d q_j}{\partial t} = -\frac{\partial Uq_j}{\partial x} - \frac{\partial Vq_j}{\partial y} - \frac{\partial \Omega q_j}{\partial \eta} + R_{q_j} + Q_{q_j}$$
$$\frac{\partial \phi}{\partial t} = -u\frac{\partial \phi}{\partial x} - v\frac{\partial \phi}{\partial y} - \omega\frac{\partial \phi}{\partial \eta} + gw$$

↑ ← numerical filters, physics, projection terms

← geopotential eqn term

Dynamics: 3. Equations

transport | pressure gradient

$$\frac{\partial U}{\partial t} = -\frac{\partial Uu}{\partial x} - \frac{\partial Vu}{\partial y} - \frac{\partial \Omega u}{\partial \eta} - \alpha\mu_d\frac{\partial p}{\partial x} - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\frac{\partial \phi}{\partial x} + R_u + Q_u$$
$$\frac{\partial V}{\partial t} = -\frac{\partial Uv}{\partial x} - \frac{\partial Vv}{\partial y} - \frac{\partial \Omega v}{\partial \eta} - \alpha\mu_d\frac{\partial p}{\partial y} - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\frac{\partial \phi}{\partial y} + R_v + Q_v$$
$$\frac{\partial W}{\partial t} = -\frac{\partial Uw}{\partial x} - \frac{\partial Vw}{\partial y} - \frac{\partial \Omega w}{\partial \eta} - g\left(\mu_d - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\right) + R_w + Q_w$$
$$\frac{\partial \mu_d}{\partial t} = -\frac{\partial U}{\partial x} - \frac{\partial V}{\partial y} - \frac{\partial \Omega}{\partial \eta}$$
$$\frac{\partial \Theta}{\partial t} = -\frac{\partial U\theta}{\partial x} - \frac{\partial V\theta}{\partial y} - \frac{\partial \Omega\theta}{\partial \eta} + R_\theta + Q_\theta$$
$$\frac{\partial \mu_d q_j}{\partial t} = -\frac{\partial Uq_j}{\partial x} - \frac{\partial Vq_j}{\partial y} - \frac{\partial \Omega q_j}{\partial \eta} + R_{q_j} + Q_{q_j}$$
$$\frac{\partial \phi}{\partial t} = -u\frac{\partial \phi}{\partial x} - v\frac{\partial \phi}{\partial y} - \omega\frac{\partial \phi}{\partial \eta} + gw$$

↑ ← numerical filters, physics, projection terms

← geopotential eqn term

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

Dynamics: 4. Time integration scheme

3rd Order Runge-Kutta time integration

$$\frac{\partial U}{\partial t} = RHS_u \qquad \frac{\partial V}{\partial t} = RHS_v \qquad \frac{\partial W}{\partial t} = RHS_w \quad \ldots$$

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

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

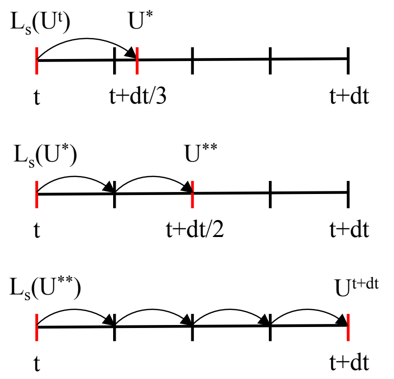
Amplification factor $\phi_t = ik\phi;\quad \phi^{n+1} = A\phi^n;\quad |A| = 1 - \frac{(k\Delta t)^4}{24}$

Dynamics: 4. Time integration scheme – time splitting

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

fast: acoustic and gravity wave terms.
slow: everything else.

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: 4. Time integration scheme – acoustic step

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

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

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

$W^{\tau+\Delta\tau}$, $\phi^{\tau+\Delta\tau}$ $$\begin{cases} \dfrac{\partial W}{\partial t} + g\overline{\left(\mu_d - \dfrac{\alpha}{\alpha_d}\dfrac{\partial p}{\partial \eta}\right)}^{\tau} = R_W^t \\ \mu_d^t \dfrac{\partial \phi}{\partial t} + U^{\tau+\Delta\tau}\dfrac{\partial \phi}{\partial x}^t + \Omega^{\tau+\Delta\tau}\dfrac{\partial \phi}{\partial \eta}^t - g\overline{W}^{\tau} = R_\phi^t \end{cases}$$

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

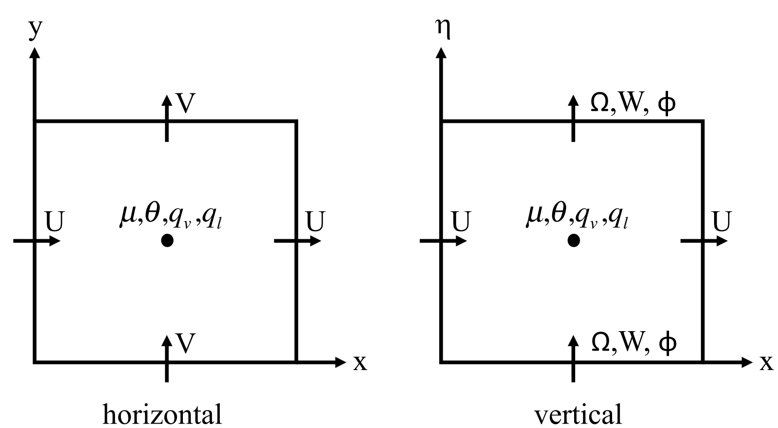
Dynamics: 4. Time integration scheme - implementation

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: 5. Grid staggering – horizontal and vertical

C-grid staggering



Dynamics: 6. Advection (transport) and conservation – dry-air mass

transport | pressure gradient

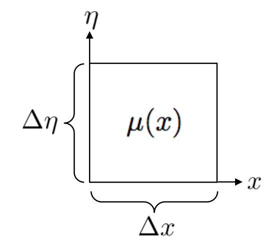
$$
\begin{aligned}
\frac{\partial U}{\partial t} &= -\frac{\partial Uu}{\partial x} - \frac{\partial Vu}{\partial y} - \frac{\partial \Omega u}{\partial \eta} - \alpha\mu_d\frac{\partial p}{\partial x} - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\frac{\partial \phi}{\partial x} + R_u + Q_u \\
\frac{\partial V}{\partial t} &= -\frac{\partial Uv}{\partial x} - \frac{\partial Vv}{\partial y} - \frac{\partial \Omega v}{\partial \eta} - \alpha\mu_d\frac{\partial p}{\partial y} - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\frac{\partial \phi}{\partial y} + R_v + Q_v \\
\frac{\partial W}{\partial t} &= -\frac{\partial Uw}{\partial x} - \frac{\partial Vw}{\partial y} - \frac{\partial \Omega w}{\partial \eta} - g\left(\mu_d - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\right) + R_w + Q_w \\
\frac{\partial \mu_d}{\partial t} &= -\frac{\partial U}{\partial x} - \frac{\partial V}{\partial y} - \frac{\partial \Omega}{\partial \eta} \\
\frac{\partial \Theta}{\partial t} &= -\frac{\partial U\theta}{\partial x} - \frac{\partial V\theta}{\partial y} - \frac{\partial \Omega\theta}{\partial \eta} + R_\theta + Q_\theta \\
\frac{\partial \mu_d q_j}{\partial t} &= -\frac{\partial Uq_j}{\partial x} - \frac{\partial Vq_j}{\partial y} - \frac{\partial \Omega q_j}{\partial \eta} + R_{q_j} + Q_{q_j} \\
\frac{\partial \phi}{\partial t} &= -u\frac{\partial \phi}{\partial x} - v\frac{\partial \phi}{\partial y} - \omega\frac{\partial \phi}{\partial \eta} + gw
\end{aligned}
$$

Next:
Dry-air mass conservation in WRF

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

Dynamics: 6. 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: 6. 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

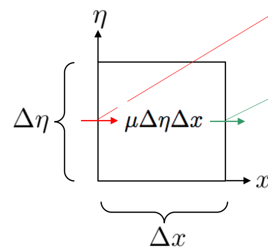
mass fluxes through control volume faces

Dynamics: 6. 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

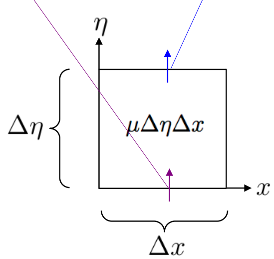
Dynamics: 6. Advection (transport) and conservation – dry-air mass

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

Mass conservation equation

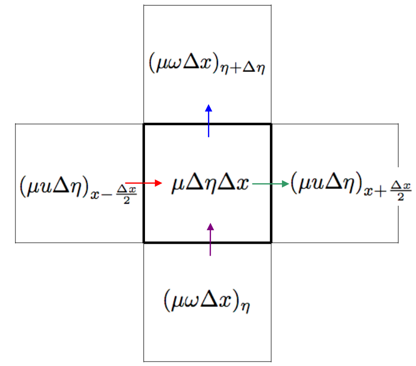
$$\Delta t^{-1}(\Delta x \Delta \eta)\cdot\left[(\mu)^{t+\Delta t}-(\mu)^t\right] = \left[(\mu u \Delta\eta)_{x-\Delta x/2,\eta} - (\mu u \Delta\eta)_{x+\Delta x/2,\eta}\right] + \left[(\mu\omega\Delta x)_{x,\eta-\Delta\eta/2} - (\mu\omega\Delta x)_{x,\eta+\Delta\eta/2}\right]$$

Vertical fluxes through the horizontal control-volume faces



Dynamics: 6. 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: 6. Advection (transport) and conservation

transport pressure gradient

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

$$\frac{\partial V}{\partial t} = -\frac{\partial Uv}{\partial x} - \frac{\partial Vv}{\partial y} - \frac{\partial \Omega v}{\partial \eta} - \alpha\mu_d\frac{\partial p}{\partial y} - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\frac{\partial \phi}{\partial y} + R_v + Q_v$$

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

$$\frac{\partial \mu_d}{\partial t} = -\frac{\partial U}{\partial x} - \frac{\partial V}{\partial y} - \frac{\partial \Omega}{\partial \eta}$$

$$\frac{\partial \Theta}{\partial t} = -\frac{\partial U\theta}{\partial x} - \frac{\partial V\theta}{\partial y} - \frac{\partial \Omega\theta}{\partial \eta} + R_\theta + Q_\theta$$

$$\frac{\partial \mu_d q_j}{\partial t} = -\frac{\partial Uq_j}{\partial x} - \frac{\partial Vq_j}{\partial y} - \frac{\partial \Omega q_j}{\partial \eta} + R_{q_j} + Q_{q_j}$$

$$\frac{\partial \phi}{\partial t} = -u\frac{\partial \phi}{\partial x} - v\frac{\partial \phi}{\partial y} - \omega\frac{\partial \phi}{\partial \eta} + gw$$

Entropy and scalar mass conservation in WRF

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

Dynamics: 6. 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: 6. Advection (transport) and conservation

transport — pressure gradient

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

$$\frac{\partial V}{\partial t} = -\frac{\partial U v}{\partial x} - \frac{\partial V v}{\partial y} - \frac{\partial \Omega v}{\partial \eta} - \alpha\mu_d\frac{\partial p}{\partial y} - \frac{\alpha}{\alpha_d}\frac{\partial p}{\partial \eta}\frac{\partial \phi}{\partial y} + R_v + Q_v$$

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

$$\frac{\partial \mu_d}{\partial t} = -\frac{\partial U}{\partial x} - \frac{\partial V}{\partial y} - \frac{\partial \Omega}{\partial \eta}$$

$$\frac{\partial \Theta}{\partial t} = -\frac{\partial U\theta}{\partial x} - \frac{\partial V\theta}{\partial y} - \frac{\partial \Omega\theta}{\partial \eta} + R_\theta + Q_\theta$$

$$\frac{\partial \mu_d q_j}{\partial t} = -\frac{\partial U q_j}{\partial x} - \frac{\partial V q_j}{\partial y} - \frac{\partial \Omega q_j}{\partial \eta} + R_{q_j} + Q_{q_j}$$

$$\frac{\partial \phi}{\partial t} = -u\frac{\partial \phi}{\partial x} - v\frac{\partial \phi}{\partial y} - \omega\frac{\partial \phi}{\partial \eta} + gw$$

Transport schemes: flux divergence (transport) options in WRF

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

Dynamics: 6. 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: 6. Advection (transport) and conservation

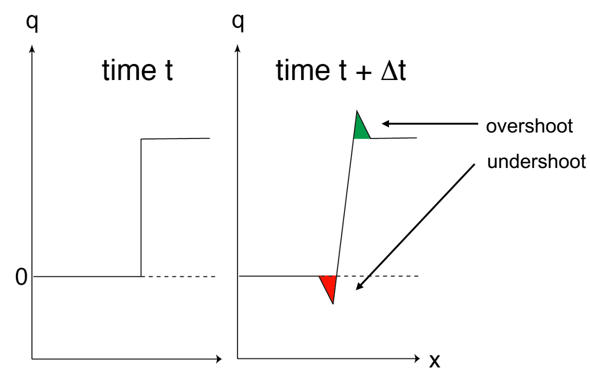
For constant U, the 5th order flux divergence tendency becomes

$$\Delta t \left.\frac{\delta (U\psi)}{\Delta x}\right|_{5th} = \Delta t \left.\frac{\delta (U\psi)}{\Delta x}\right|_{6th} - \underbrace{\left|\frac{U\Delta t}{\Delta x}\right| \frac{1}{60}\left(-\psi_{i-3} + 6\psi_{i-2} - 15\psi_{i-1} + 20\psi_i - 15\psi_{i+1} + 6\psi_{i+2} - \psi_{i+3}\right)}_{\frac{Cr}{60}\frac{\partial^6 \psi}{\partial x^6} + H.O.T}$$

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: 6. Advection (transport) and conservation – shape preserving

1D advection



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

Dynamics: 6. 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: $f_i = f_i^{upwind} + f_i^c$

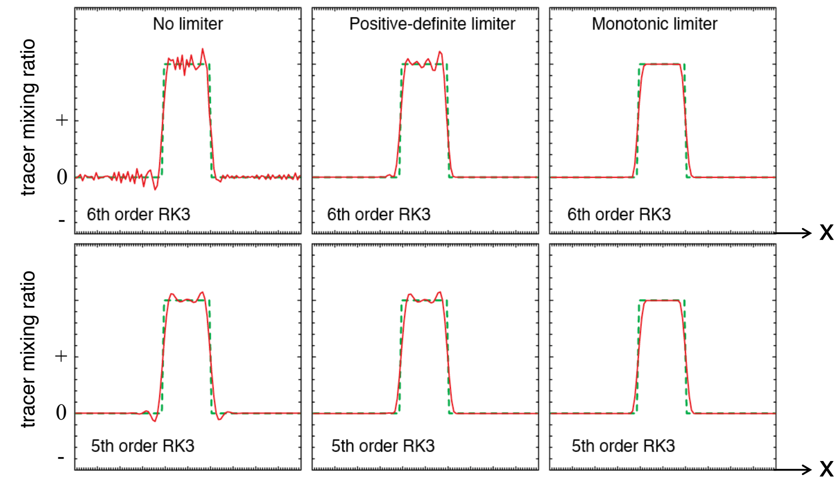
(2) Renormalize high-order correction fluxes f_i^c such that solution is positive definite or monotonic: $f_i^c = R(f_i^c)$

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

Dynamics: 6. Advection (transport) and conservation – examples

1D Example: Top-Hat Advection

1D Top-hat transport Cr = 0.5, 1 revolution, 200 steps



Dynamics: 6. 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: 7. Time step parameters

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

Courant number limited, 1D: $C_r = \frac{U\Delta t}{\Delta x} < 1.43$ (5th order adv.)

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: 7. Time step parameters

3[rd] 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: 7. Time step parameters

3[rd] 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: 8. Filters – divergence damping

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

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

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

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

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

γ_d = 0.1 recommended (default) (&dynamics *smdiv*)

(Illustrated in height coordinates for simplicity)

Dynamics: 8. 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

β = 0.1 recommended (default) [&dynamics *epssm*]

Dynamics: 8. Filters – external mode filter

Purpose: filter the external mode

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

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

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

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

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

$\gamma_e = 0.01$ recommended (default) [&dynamics *emdiv*]

(Primarily for real-data applications)

Dynamics: 8. Filters – vertical velocity damping

Purpose: damp anomalously-large vertical velocities

(usually associated with anomalous physics tendencies)

Additional term:

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

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

$Cr_\beta = 1.0$ typical value (default)

[share/module_model_constants.F *w_beta*]

$\gamma_w = 0.3$ m/s^2 recommended (default)

[share/module_model_constants.F *w_alpha*]

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

Dynamics: 8. 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 $\quad l = (\Delta x \Delta y)^{1/2}$

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

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

Dynamics: 8. Filters – gravity-wave absorbing layer

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

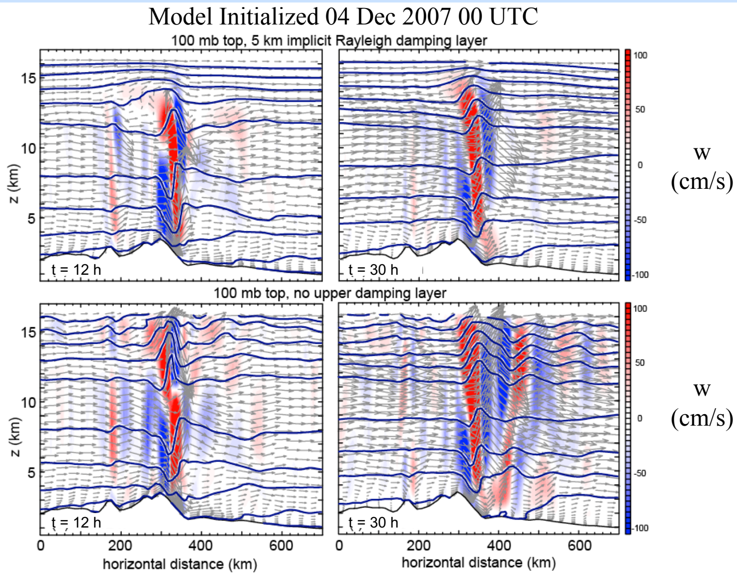
$$W^{\tau+\Delta\tau} = W^{*\tau+\Delta\tau} - \Delta\tau R_w(\eta) W^{\tau+\Delta\tau}$$

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

$R_w(\eta)$ - 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: 8. Filters – gravity-wave absorbing layer example



Dynamics: 9. Map projections and global configuration

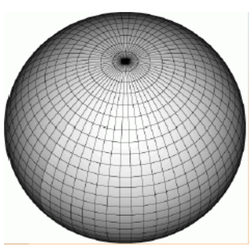
ARW Model: projection options

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 ($m_x \neq m_y$)

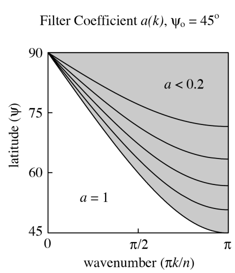
Dynamics: 9. Map projections and global configuration

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.



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

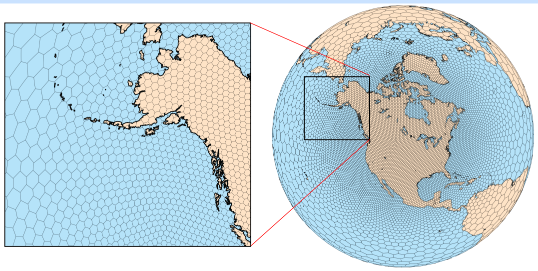
$$a(k) = \min\left[1., \max\left(0., \left(\frac{\cos\psi}{\cos\psi_o}\right)^2 \frac{1}{\sin^2(\pi k/n)}\right)\right]$$

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

Dynamics: 9. Map projections and global configuration

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; U.S. Department of Energy Office of Science; NSF; NESL

ARW Model: Boundary Condition Options

Lateral boundary conditions

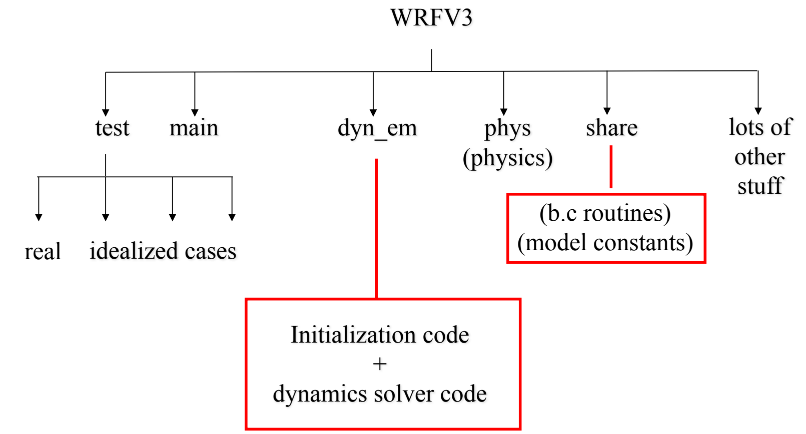
1. Specified (Coarse grid, real-data applications).
2. Open lateral boundaries (gravity-wave radiative).
3. Symmetric lateral boundary condition (free-slip wall).
4. Periodic lateral boundary conditions.
5. Nested boundary conditions (specified).

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

Post-processing Tools (2): UPP

Kate Fossell

NCEP's UNIFIED POST PROCESSOR (UPP)

Presented by Kate Fossell

January 24, 2018

Outline

- Overview
- Components and Functions
- Sample fields generated
- Installing UPP
- Running *unipost*
  - Required input files
  - Controlling output generation
    - grib1 and grib2 formats
- Running *copygb*
  - Specifying target grid
- Visualization

UPP Overview

- UPP is one of the many post processing packages available
- NCEP Developed & Supported Operationally
  - GFS, GEFS, NAM, SREF, RAPR, HRRR, HWRF, etc.
- NCAR Supports community code for WRF Post Processing

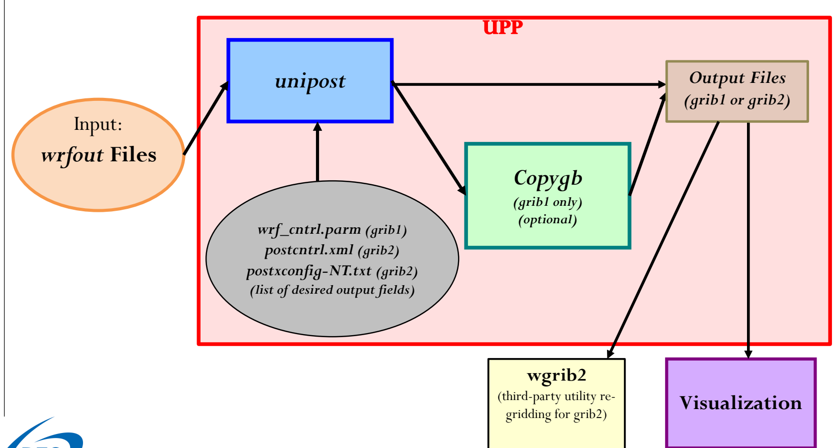
Why would you want to use UPP?

- Generates output in GRIB1 and GRIB2 format.
- Produces hundreds of products like those used operationally on same operational grids.
- Enables product generation on any output grid.

  E.g. MET: Regrid model data to match a observational grid for verification
- Processes model output from the WRF-ARW dynamical core
- Produces requested diagnostics and fields, but does not plot or visualize data.
- MPI parallelized code
- UPP Supports WRFV3.9 new vertical hybrid coordinate

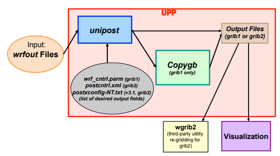
Components of the UPP

UPP has two components: 1) unipost 2) copygb (grib1 only)



Unipost Functions & Features

- Performs **vertical** interpolation from model levels/surfaces onto isobaric, height, and other levels/surfaces
- Computes **diagnostic** fields
- Destaggers wind onto mass points
- An MPI-parallel code



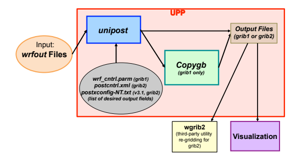
Copygb (grib1 format only) Functions & Features

- Performs ***optional*** **horizontal** interpolation to a defined output grid – grib1 format only

i.e. Creates an output grid different than the model integration domain

- e.g. Convert to operational grid: 221 (NAM, RAP, SREF)
- e.g. Lambert ⟶ Lat-Lon
- e.g. convert to observational grid

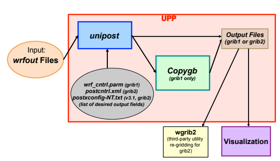
wgrib2 Third-party utility for re-gridding grib2



Ingesting WRF model output

Input: *wrfout* Files

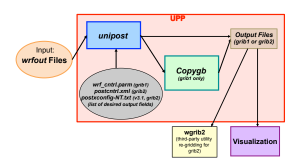
- The unipost ingests WRF model output in netCDF using the WRF I/O package.
  - One time per output file is best w/ sample UPP run scripts (frames_per_outfile=1 in WRF model namelist).
  - By default UPP tries to read a set list of fields in wrfout files.
    - Should contain necessary fields for basic diagnostics
    - Could impact UPP if you change registry or wrfout fields



Fields generated by the UPP

Output Files (Grib1 or Grib2)

- **The UPP currently outputs hundreds of possible fields.**
  - Complete list in the Post Processing Utilities Chapter of the user guide
  - Fields are output in Grib1 or Grib2 format
- **Sample fields generated by UPP:**
  1) T, Z, humidity, wind, cloud water, cloud ice, rain, and snow on isobaric levels
  2) SLP + shelter level T, humidity, and wind fields
  3) Precipitation-related fields
  4) PBL-related fields
  5) Diagnostic products (i.e. RH, radar reflectivity, CAPE)
  6) Radiative/Surface fluxes
  7) Cloud related fields
  8) Aviation products
  9) Synthetic satellite products



Outputting fields on different vertical coordinates

- *unipost* outputs on several vertical coordinates:
  - Native model levels
  - 47 isobaric levels: Default: 2, 5, 7, 10, 20, 30, 50, 70, then every 25 hPa from 75-1000 hPa.
  - 15 flight/wind energy levels: 30, 50, 80, 100, …, 2743, 3658, 4572, 6000 m (above ground or above MSL)
  - 6 PBL layers: each averaged over a 30 hPa deep layer
  - 2 AGL radar levels: 1000 & 4000
- Except for AGL radar and isobaric levels, vertical levels are listed from the ground surface up in *wrf_cntrl.parm (postcntrl.xml).*

UPP download and compile

UPP Dependencies & Required Libraries

- UPP build relies on the existence of a built WRF source directory. Uses WRF i/o routines.
- UPPV2.1+ depends on WRFV3.5 or later releases.
- UPPV3.0+ depends on WRFV3.7+ for Ferrier Physics
- Libraries required:
  - netCDF
  - JasPer
  - PNG
  - Zlib
  - WRF i/o libs

Downloading the UPP source code

- The UPP source code can be obtained from:
  http://www.dtcenter.org/upp/users/downloads/index.php
  - The latest version available is: UPPV3.2.tar.gz
- Unpack the downloaded file:
  tar –zxvf UPPV3.2.*tar.gz*
- *cd* to newly created UPPV3.2/ directory
  - Important Directories:
    - **scripts/**: sample scripts for running UPP and generating graphics
    - **parm/**: contains the files used to request output fields when running the unipost (i.e. wrf_cntrl.parm, postcntrl.xml) *wrf_cntrl.parm (grib1) postcntrl.xml (grib2)*
    - **clean, configure, compile**: scripts used in the build process

Compile source codes

- The build mechanism follows the WRF model build paradigm:

  ./configure : respond to screen prompts about target computing platform

- *./compile >& compile_upp.log*

Compile source codes (cont.)

- If compilation is successful, these three executables will be present in bin/ :

  copygb.exe
  ndate.exe
  unipost.exe

- Currently have build options established for IBM and Linux (PGI/Intel/Gnu compilers)

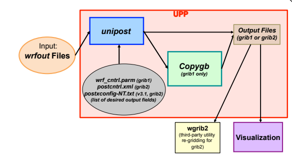
- The arch/configure.defaults file has compilation options for various platforms, and is where new computers or compilers might be added.

Developmental Testbed Center

Running unipost and copygb

Developmental Testbed Center

Running UPP



* Use sample scripts as a template or guide to run UPP *

Run Script: ./run_unipost >& script_output.log &

- run_unipost is a korn shell script that runs UPP end to end: unipost + copygb (if desired)
- User edits model core, paths, date, time, command syntax (serial vs. parallel) in script.
- Links all required files, loops over times/files and processes fields requested fields from wrf_cntrl.parm or postcntrl.xml, runs copygb if requested.
- Unipost.exe output/error messages is redirected to log files, e.g. unipost_d01.00.out. Hint: Look in these files for information about errors.

Unipost Running unipost.exe

** Requires 2 input files to run + a few extra data files **

1) *itag*: 4-5 line text file that details WRF model output to process. Also referred to as the namelist.

wrfout_d01_2010-06-27_00:00:00	← *WRF history filename*
netcdf	← *WRF output format (netcdf/binary)*
grib2	← *extra line only if writing GRIB2*
2010-06-27_00:00:00	← *validation time*
NCAR	← *model name: "NCAR" for WRF-ARW*

2) *wrf_cntrl.parm (grib1)* : control file specifying fields/levels to output in GRIB1 (*text file*)

- or -

postxconfig-NT.txt (grib2) : control file specifying fields/levels to output in GRIB2 *(text file generated from xml postcntrl.xml)*

wrf_cntrl.parm postcntrl.xml postxconfig-NT.txt

3) *extra data files: e.g. eta_micro_lookup.dat, coefficient files for satellite, etc.*

*** In the sample scripts/run_unipost* scripts, these files are automatically generated (itag) or linked (wrf_cntrl.parm & eta_micro_lookup.dat, etc).

unipost control file:

* file that lists desired fields and levels that unipost reads directly

wrf_cntrl.parm postcntrl.xml postxconfig-NT.txt

Grib1 Format:

wrf_cntrl.parm (text file)

Grib2 Format:

v3.1+: postxconfig-NT.txt (text file, but not formatted for easy read/write)

generated from postcntrl.xml & post_avblflds.xml (xml files)

unipost control file for **grib1** :

wrf_cntrl.parm (Grib1)

- User controlled and modified text file that lists fields and level(s) of fields to output; each product described by 2 lines (Examples next slides)
- The included parm/*wrf_cntrl.parm* file has entries for most output fields. ** Use this as template! ** (Text file fixed width format)
- The users' guide "Fields produced by *unipost*" (Table 1) more fully explains the character string abbreviations used in the wrf_cntrl.parm file.

unipost control file: *wrf_cntrl.parm*

wrf_cntrl.parm (Grib1)

- Each field described by 2 lines: product description and levels

```
(PRESS ON MDL SFCS    ) SCAL=(6.0)
L=(11000 00000 00000 00000 00000 00000 00000…
(HEIGHT ON MDL SFCS ) SCAL=(6.0)
L=(11000 00000 00000 00000 00000 00000 00000…
```

← GRIB packing precision**

Levels to output: Each column represents a single model/isobaric level: "1" (or "2" - special case) = output, "0" = no output

Product description – unipost code keys on these character strings.

** larger values → more precision, but larger GRIB files.

Examples

wrf_cntrl.parm (Grib1)

➢ Output T every 50 hPa from 50 hPa to 1000 hPa:

```
 (TEMP ON PRESS SFCS   ) SCAL=( 4.0)
L=(00000   01001    01...)
        2  5 7 10 20    30 50 70 75 100    125 150
```

*** Isobaric levels increase from left to right:
2, 5, 7, 10, 20, 30, 50, 70, then every 25 hPa from 75-1000 hPa.
(Default/standard – can manually change code for different pressure levels)

Isobaric levels every 50 hPa:

```
L=(00000 01001 01010 10101 01010 10101 01010 10101 01010 10000 00000 00000 00000 00000)
```

Isobaric levels every 25 hPa:

```
L=(00000 01011 11111 11111 11111 11111 11111 11111 11111 10000 00000 00000 00000 00000)
```


Examples

wrf_cntrl.parm (Grib1)

➢ Output instantaneous surface sensible heat flux:

```
(INST SFC SENHEAT FX ) SCAL=( 4.0)
L=(10000 00000 00000 00000 00000 00000 00000 00000 00000 00000…)
```

➢ Output the U-wind component at the 5 lowest model levels:

```
(U WIND ON MDL SFCS   ) SCAL=( 4.0)
L=(11111 00000 00000 00000 00000 00000 00000 00000 00000 00000…)
```

➢ Output U-wind component at 30, 50, and 80 m AGL:

```
(U WIND AT FD HEIGHT) SCAL=( 4.0)
L=(22200 00000 00000 00000 00000 00000 00000 00000 00000 00000…)
```

For the flight/wind energy level fields:
- "2" requests AGL.
- "1" requests above mean sea level.

unipost control file for **grib2** :

postcntrl.xml postxconfig-NT.txt (Grib2)

➢ User controlled xml file that lists desired fields to be output by UPP.

➢ The included parm/*postcntrl.xml* file has examples of the template to follow.

➢ The included parm/*post_avblcntrl.xml* file has a listing of all available field names, shortnames, parameter info.

➢ New V3.1+: the included parm/*postxconfig-NT.txt* file is read by unipost and includes fields from default postcntrl.xml and can be used directly

unipost control file: *post_avblflds.xml*

postcntrl.xml postxconfig-NT.txt (Grib2)

- **Lists all available fields and details for grib2 tables/output**
- **Does not need to be modified unless adding new variables or modifying from default.**

```
<param>
    <post_avblfldidx>2</post_avblfldidx>
    <shortname>TMP_ON_HYBRID_LVL</shortname>
    <pname>TMP</pname>
    <fixed_sfc1_type>hybrid_lvl</fixed_sfc1_type>
    <scale>4.0</scale>
</param>

<param>
    <post_avblfldidx>70</post_avblfldidx>
    <shortname>DPT_ON_SPEC_PRES_ABOVE_GRND</shortname>
    <pname>DPT</pname>
    <fixed_sfc1_type>spec_pres_above_grnd</fixed_sfc1_type>
    <scale>3.0</scale>
</param>
```

- **UPP ID**
- **Character name describing the product/field**
- **Field type abbreviation used by grib2 libraries**
- **Vertical coordinate type**
- **Grib precision packing**

unipost control file: *postcntrl.xml*

postcntrl.xml
postxconfig-NT.txt
(Grib2)

- User modified xml file to list all desired grib2 fields to be output by UPP
- **Use provided file as a guide**

<param>
<shortname> **TMP_ON_SPEC_HGT_LVL_ABOVE_GRND_2m**</shortname>
<scale>**4.0**</scale>
</param>

<param>
<shortname> **UGRD_ON_ISOBARIC_SFC**</shortname>
<scale> **4.0** </scale>
<level> **50000. 70000. 85000. 92500. 100000.** </level>
</param>

*** Formatting important, use provided file as a guide**

- **Character name describing the product/field**
- **Grib precision packing**
- **Vertical coordinate levels desired**

Developmental Testbed Center

New for V3.1+ when outputting grib2 format:

postcntrl.xml
postxconfig-NT.txt
(Grib2)

* unipost requires to read a text file called: **postxconfig-NT.txt**
(operationally motivated to speed up unipost)

Additional pre-processing step required to convert xml fie to a flat file:

ONLY IF : user wants to add/modify fields or levels of fields – must modify postcntrl.xml and then convert xml to text file:

1) >> cd UPPV3.1/parms
2) *edit the postcntrl.xml to add/remove fields/levels*
3) >> make

- make calls a perl program that does that conversion based on postcntrl.xml and post_avblflds.xml
- Detailed instructions in the UPP Users Guide
- output is " **postxconfig-NT.txt** "

ELSE: Can use default postxconfig-NT.txt with default fields and levels (new steps only necessary if requesting new fields to output)

Regridding UPP grib output (optional):

- **grib1** : *Copygb* *(grib1 format only)* (included in UPP package)
- **grib2** : *wgrib2* (third-party; examples in Users Guide and on online tutorial)

Developmental Testbed Center

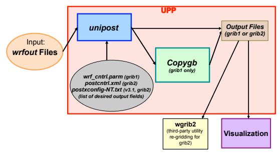
Copygb *(grib1 format only)* target grid definition

➢ The generic command to run copygb and horizontally interpolate onto a new grid is:

copygb.exe –xg"${grid}" in.grb out.grb

➢ Two options on how to specify the target $grid:

1. Pre-defined NCEP standard grid number
2. User-defined grid definition



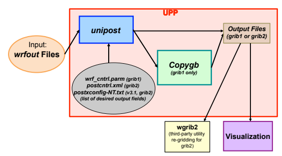
Developmental Testbed Center

Run *copygb* – Option 1

Copygb *(grib1 format only)*

- Interpolate to a pre-defined NCEP standard grid (restrictive but simple)
  - For example, to interpolate onto NCEP grid 212:
    copygb.exe –xg212 in.grb out.grb

Descriptions of NCEP grids are available online:
http://www.nco.ncep.noaa.gov/pmb/docs/on388/tableb.html



Developmental Testbed Center

Run *copygb* – Option 2a

Copygb *(grib1 format only)*

- Create a user-defined ***Lambert Conformal*** grid by specifying a full set of grid parameters (complicated but flexible).

255 indicates user-defined grid; map type (3=LC); # of points; SW corner (millidegrees); Proj cent lon (millidegrees)

copygb.exe –xg"255 3 NX NY STARTLAT STARTLON 8 CENLON DX DY
0 64 TRUELAT1 TRUELAT2 " in.grb out.grb

Proj true latitudes (millidegrees); horizontal spacing (meters)

copygb –xg"255 3 185 129 12190 -133459 8 -95000 40635 40635
0 64 25000 25000" in.grb out.grb

Developmental Testbed Center

Run *copygb* – Option 2b

Copygb *(grib1 format only)*

- Create a user-defined ***Polar Stereographic*** *grid* by specifying a full set of grid parameters (complicated but flexible).

map type (5=STR)

copygb.exe –xg"255 5 NX NY STARTLAT STARTLON 8 CENLON DX DY
0 64" in.grb out.grb

Center flag (0=NH ; 128=SH)

copygb –xg"255 5 580 548 10000 -128000 8 -105000 15000 15000
0 64" in.grb out.grb

Developmental Testbed Center

Run *copygb* – Option 2c

Copygb *(grib1 format only)*

- Create a user-defined ***Latitude-Longitude*** *grid* by specifying a full set of grid parameters (complicated but flexible).

map type (0=LTLN)

copygb.exe –xg"255 0 NX NY STARTLAT STARTLON 136 ENDLAT ENDLON
DLAT DLON 64" in.grb out.grb

grid spacing (millidegrees); NE lat (millidegrees); NE lon (millidegrees)

copygb –xg"255 0 401 401 10000 -130000 136 50000 -90000
100 100 64" in.grb out.grb

Developmental Testbed Center

Visualization of UPP output

- Gempak - sample run script with UPP package
- GrADS - sample run script with UPP package
- NCL
- Python
- Ncview - (after conversion to netcdf)
- Matlab
- IDL
- Many Others

Visualization: GEMPAK

- The GEMPAK utility "nagrib" reads GRIB files from any non-staggered grid and generates GEMPAK-binary files that are readable by GEMPAK plotting programs
- GEMPAK can plot horizontal maps, vertical cross-sections, meteograms, and sounding profiles.
- Package download and user guide are available online:

  http://www.unidata.ucar.edu/software/gempak/index.html

- A sample script named *run_unipostandgempak* is included in scripts/ that can be used to run *unipost*, *copygb*, and then plot various fields using GEMPAK.
- Further details on this script and using GEMPAK are available in the user's guide.

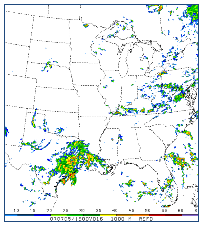
Visualization: GrADS

- GrADS also has utilities to read GRIB1 and GRIB2 files on any non-staggered grids and generate GrADS "control" files.
- The utilities grib2ctl (grib1), g2ctl (grib2), and gribmap are available via:
  http://www.cpc.ncep.noaa.gov/products/wesley/grib2ctl.html
  http://www.cpc.ncep.noaa.gov/products/wesley/g2ctl.html
- Package download and user guide for GrADS are available online:
  http://cola.gmu.edu/grads/
- A sample script named *run_unipostandgrads* is included in scripts/ that can be used to run *unipost*, *copygb*, and then plot various fields using GrADS.

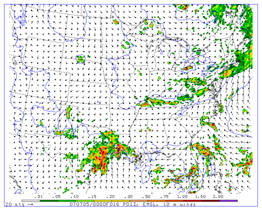
Example of Forecasts Plotted in:

Gempak

Derived Radar Reflectivity

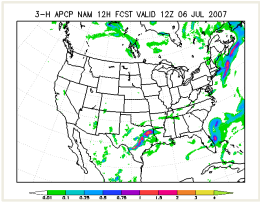


Observed Reflectivity

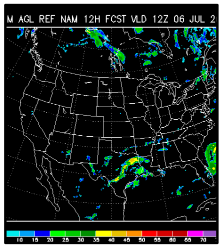


GrADS

Accumulated Precipitation



Observed Reflectivity



Future plans

- Continue adding new products to the released UPP code as they are developed, and expand code portability.
- Improved documentation for users to add custom fields

Helpful Links:

UPP Users Website:
http://www.dtcenter.org/upp/users/index.php

New UPP Online Tutorial
https://dtcenter.org/upp/users/support/online_tutorial/UPPv3.2/index.php

UPP Users' Guide available at:
http://www.dtcenter.org/upp/users/docs/user_guide/V3/upp_users_guide.pdf

UPP FAQ's Page:
http://www.dtcenter.org/upp/users/overview/upp_faqs.php

UPP Questions Contact: upp-help@ucar.edu

Questions???

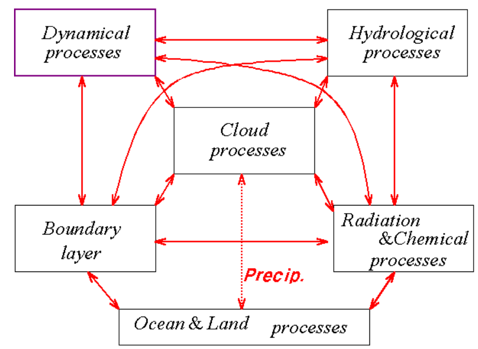
Overview of Physical Parameterizations

Song-You Hong

Overview of Physical Parameterizations

Song-You Hong
(KIAPS: Korea Institute of Atmospheric Prediction Systems)
(Also, an NCAR affiliate scientist)

1) Concept



*** Physical processes in the atmosphere**
: Specification of heating, moistening and frictional terms in terms of dependent variables of prediction model
→ Each process is a specialized branch of atmospheric sciences.

*** Parameterization**

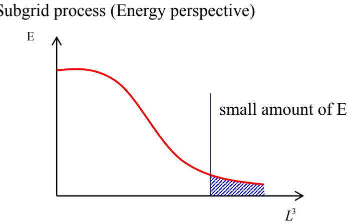
The formulation of physical process in terms of the model variables as parameters, i.e., constants or functional relations.

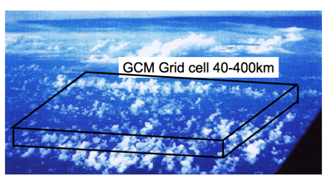
1) Concept…continued

Subgrid scale process

Any numerical model of the atmosphere must use a finite resolution in representing continuum certain physical & dynamical phenomena that are smaller than computational grid.

- Subgrid process (Energy perspective)





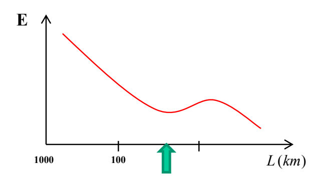
$\Delta x \rightarrow 0$, the energy dissipation takes place by molecular viscosity

Objective of subgrid-scale parameterization

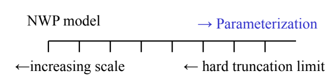
"To design the physical formulation of energy sink, withdrawing the equivalent amount of energy comparable to cascading energy down at the grid scale in an ideal situation."

1) Concept…continued

※ Parameterization that are only somewhat smaller than the smallest resolved scales.

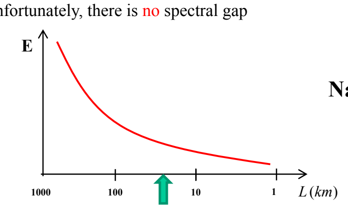


Model



where truncation limit ; spectral gap

Unfortunately, there is no spectral gap



Nature

2) Subgrid scale process & Reynolds averaging

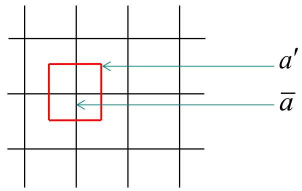
Consider prognostic water vapor equation

$$\frac{\partial \rho q}{\partial t} = -\frac{\partial \rho u q}{\partial x} - \frac{\partial \rho v q}{\partial y} - \frac{\partial \rho w q}{\partial z} + \rho E - \rho C \qquad \cdots(1)$$

In the real atmosphere,

$$u = \bar{u} + u', \qquad q = \bar{q} + q' \qquad \left(\begin{array}{l} ※\bar{a} : grid-resolvable \\ a' : subgrid\ scale\ perturbation \end{array} \right)$$

ρ' is neglected



2) Subgrid scale process & Reynolds averaging…continued

*** Rule of Reynolds average :** $\overline{q'} = 0,\ \overline{u'\bar{q}} = 0,\ \overline{\bar{u}\,\bar{q}} = \bar{u}\,\bar{q}$

then eq.(1) becomes

$$\frac{\partial \rho \bar{q}}{\partial t} = \underbrace{-\frac{\partial \rho \bar{u}\bar{q}}{\partial x} - \frac{\partial \rho \bar{v}\bar{q}}{\partial y} - \frac{\partial \rho \bar{w}\bar{q}}{\partial z}}_{①} \underbrace{- \frac{\partial \rho \overline{u'q'}}{\partial x} - \frac{\partial \rho \overline{v'q'}}{\partial y} - \frac{\partial \rho \overline{w'q'}}{\partial z}}_{②} + \rho E - \rho C \quad \cdots(2)$$

① grid-resolvable advection (dynamical process)
② turbulent transport

*** How to parameterize the effect of turbulent transport**

a) $-\rho \overline{w'q'} = 0$: 0th order closure

b) $-\rho \overline{w'q'} = K \frac{\partial \bar{q}}{\partial z}$: 1st order closure (K-theory)

c) obtain a prognostic equation for $\overline{w'q'}$ from (1), (2)

$$\frac{\partial \rho w q}{\partial t} = -\frac{\partial \rho u w q}{\partial x} + \cdots$$

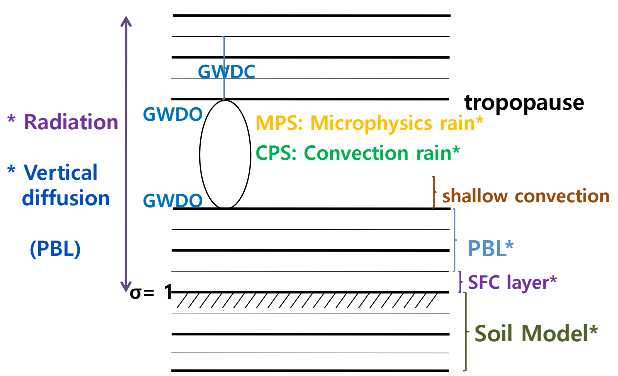
taking Reynolds averaging,

$$\frac{\partial \rho \overline{w'q'}}{\partial t} = \frac{\partial \rho \overline{w'w'q'}}{\partial z}$$

$$-\rho \overline{w'w'q'} = K' \frac{\partial \rho \overline{w'q'}}{\partial z} \quad : \text{2nd order closure}$$

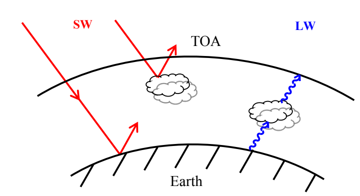
3) Schematic of physics algorithm : In modeled atmosphere : 7* (essential) ~10

$$\frac{d \ln\theta}{dt} = \frac{H}{c_p T}, \quad \frac{dq}{dt} = S, \quad \frac{d\vec{u}}{dt} = \nabla_z \vec{\tau}$$



1. Radiation

1.1 Concept



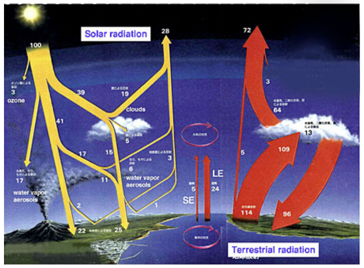
TOA (top of the atmos) : S = 1360 Wm^{-2}

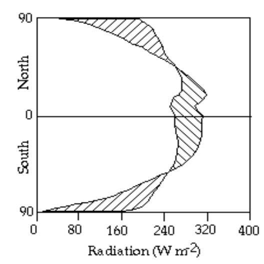
Mean Flux : $\frac{S}{4} = 340\ Wm^{-2}$ → Energy source for Earth

30% : reflected from the atmosphere clouds

→ Back to space by terrestrial infrared radiation

25% : absorbed in the atmosphere
45% : absorbed at the earth surface



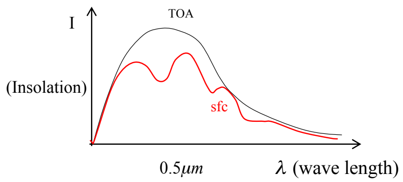


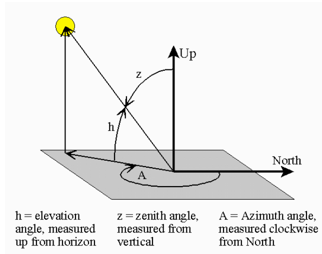
Energy loss

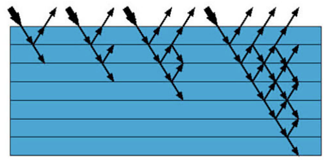
Energy gain

Energy loss

1.2 Solar radiative transfer







- At TOA,

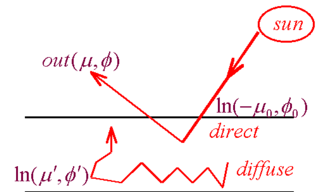
$$F = S(\frac{dm}{d})^2 \cos\theta_0 \quad (\theta_0 \text{ :Zenith angle})$$

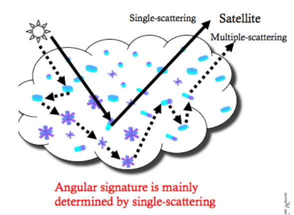
- Basic equations $\mu = \cos\theta$

$$\mu \frac{dI(\tau,\mu,\phi)}{d\tau} = I(\tau,\mu,\phi) - J(\tau,\mu,\phi)$$

absorption source emission

$$d\tau = -k_\nu \rho_a dz \qquad \tau\text{(optical depth)} = \int_z^{z_\infty} k_\nu(z')\rho_a(z')dz' = \int_0^p k_\nu(p')q(p')\frac{dp'}{g}$$





$$J = J(\tau,\mu,\phi) = \frac{\tilde{\omega}}{4\pi}\int_0^{2\pi}\int_{-1}^{1} I(\tau,\mu',\phi')P(\mu,\phi;\mu',\phi')d\mu'd\phi \text{ [diffuse (multiple) scattering]}$$

$$+\frac{\tilde{\omega}}{4\pi}F_0 P(\mu,\phi;-\mu_0,\phi_0)e^{-\frac{\tau}{\mu_0}} \text{ [single(direct) scattering]}$$

P : Scattering phase function : redirects (μ',ϕ') (μ,ϕ)

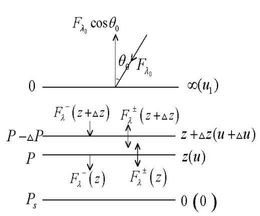
$\tilde{\omega} = \frac{\sigma_s}{\sigma_e}$: Scatteing albedo

scattering cross section/extinction(scattering + absorption) cross section

* remove ϕ dependency using $P(\cos\theta)$ function

* P, $\tilde{\omega}$, Albedo depend on λ, particle size & shape.

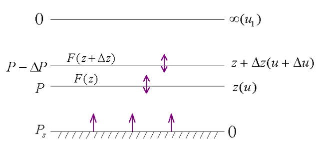
$$P(\cos\phi) = \sum_{l=0}^{N}\tilde{\omega}_l P_l(\cos\phi) \quad \text{: Legendre Polynomial}$$



Radiative transfer equation solver.

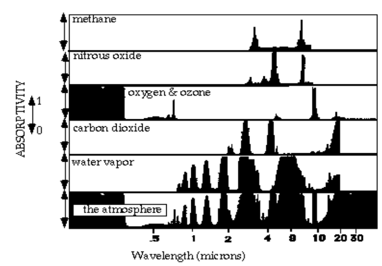
→ Discrete - ordinates method
Two - Stream and Eddington's approximation
Delta - function adjustment and similarity principle
δ - Four stream approximation

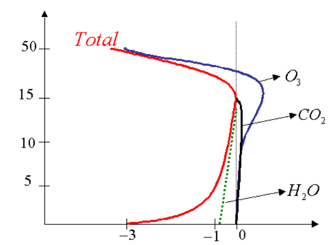
1.3 Terrestrial radiation



$$F(z) = F^\uparrow(z) - F^\downarrow(z)$$
$$\Delta F = F(z+\Delta z) - F(z)$$

$$\left.\frac{\partial T}{\partial t}\right|_{IR} = -\frac{1}{c_P\rho}\frac{\Delta F}{\Delta P} = -\frac{g}{c_P}\frac{\Delta F}{\Delta u}$$



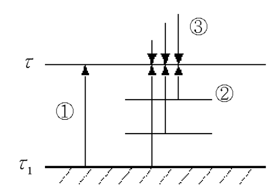


In spectral bands (monochromatic)

$$\uparrow \quad \mu\frac{dI_\nu(\tau,\mu)}{d\tau} = I_\nu(\tau,\mu) - B_\nu(T)$$

$$\downarrow \quad -\mu\frac{dI_\nu(\tau,-\mu)}{d\tau} = I_\nu(\tau,-\mu) - B_\nu(T)$$

B.C. $\begin{cases} SFC\ (\tau=\tau_1),\ I_\nu(\tau,\mu) = B_\nu(T_s) \\ TOP\ (\tau=0),\ I_\nu(0,-\mu) = 0 \end{cases}$



$$F^\uparrow(\tau) = 2\pi B_\nu(T_s)\int_0^1 e^{-\frac{(\tau_1-\tau)}{\mu}}\mu d\mu + 2\int_0^1\int_\tau^{\tau_1}\pi B_\nu[T(\tau')]e^{-\frac{(\tau'-\tau)}{\mu}}d\tau'd\mu$$

$$F^\downarrow(\tau) = 2\int_0^1\int_0^\tau \pi B_\nu[T(\tau')]e^{-\frac{(\tau-\tau')}{\mu}}d\tau'd\mu$$

$$d\tau = -k_\nu \rho dz = \int_0^{u_1} k_\nu du, \quad u_1 = \int_0^\infty \rho dz \text{ (path length)}$$

LW is time consuming !

1.4 Cloud fraction

i) Ealier method (Slingo)

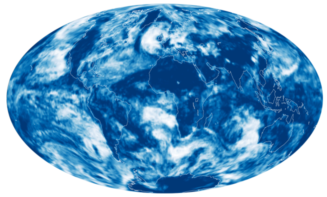
$f = f_c$(convective cloud, CPS) + f_l(large-scale cloud, MPS)

f_c : depends on precipitation, p_{top}, p_{bottom}

f_l : depends on $RH = 1 - \left[\frac{1-RH}{1-RH_0}\right]^{0.5}$

where RH_0 is the critical value of RH,

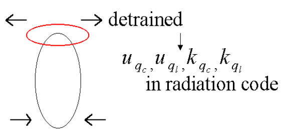
which is optimized based on observations



ii) Advanced method (Chou)

- inclusion of ice, liquid hydrometeors as prognostic water substances
- consistent treatment of water substance for both precipitation & radiative properties.

f_c : uses information of detrained water substances from sub-grid scale clouds in convective parameterization scheme (CPS)



f_l use the hydrometeor information from microphysics routine (MPS), $q_c, q_s, q_i, \cdots$

iii) Radiation properties

$\tau_i^c = cwp[a_i + \frac{b_i}{r_{ei}}] f_{ice}$ (optical thickness)

$w_i^c = 1 - c_i - d_i r_{ei}$ (co-albedo)

$g_i^c = e_i - f_i r_{ei}$ (asymmetry factor)

$f_i^c = (g_i)^2$ (forward peak fraction)

a-f : coeff : depends upon band and k-

$\overline{\tau_c} = \sum_i \tau_i$ $\quad$ i : each gas

(The effective optical thickness for each spectral band)

- The long wave cloud emissivity (E_{cld})

$c'_f = E_{cld} c_f$

$E_{cld} = 1 - e^{-Dk_{abc}cwp}$

where $D = 1.66$: diffusivity factor

$k_{abc} = k_l(1 - f_{ice}) + k_i f_{ice}$: absorptivity coefficient

- For diagnostic microphysics scheme,

- cloud water scale height

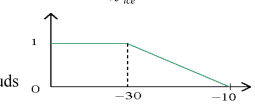
$h_l = a \ln(1.0 + \frac{b}{g}\int_{P_t}^{P_s} q dp)$

- cloud droplet size,

r_{ee} $\begin{cases} = 10\mu m & \text{over ocean} \\ < 10\mu m & \text{over land} \end{cases}$: warm clouds

r_{ei}: $10\mu m$ (low) ~ $30\mu m$: (high) ice clouds

- ice fraction, f_{ice}



- For prognostic microphysics scheme,

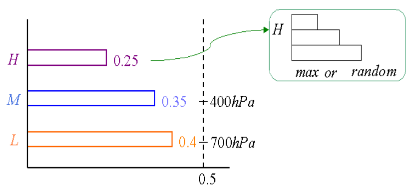
: Broadband radiation : Radiative properties are explicitly computed from prognostic water substances

: Simplified radiation : Dudhia (1989)

α_p(absorption coefficient) $= \frac{1.66}{2000}(\frac{\pi N_0}{\rho_{rs}^3})^{\frac{1}{4}} \quad m^2 g^{-1} = \begin{cases} 2.34\times10^{-3} \quad m^2g^{-1} & \text{for snow} \\ 0.33\times10^{-3} \quad m^2g^{-1} & \text{for rain} \end{cases}$

u_p (effective water path length) $= (\rho q_{rs})^{\frac{3}{4}} \Delta z \times 1000 \quad gm^{-2} \rightarrow \tau_p$ (transmission) = $\exp(-\alpha_p u_p)$

iv) Cloud overlapping



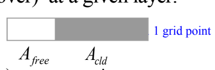
Maximum overlapping : 0.4

Minimum overlapping : 1.0

Random overlapping : $H + (1-H)M + \{1 - H - (1-H)M\}L = 0.6$

- Computation :

τ is scaled by A_c (cloud cover) at a given layer.



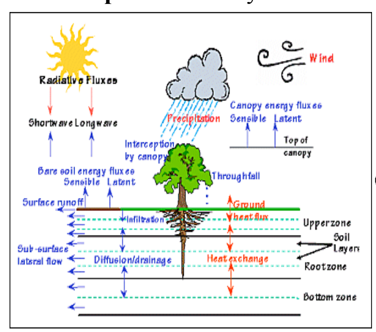
- Flux for each of $A_c, (1 - A_c)$ → summation

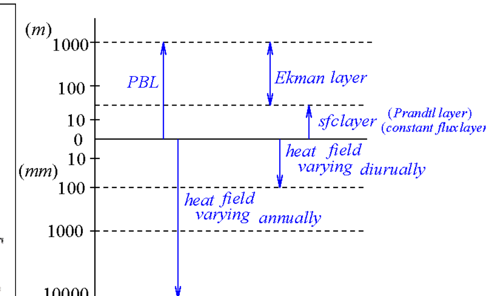
- Issues : $A_c = 0$ or 1 ← in WRF versus partial cloudiness in GFS : WRF has partial cloudiness recently

- Interaction : Radiation ← *reflect SW, LW* — Cloud; Radiation → *enhance or reduce cloud activity* → Cloud

2. Land-surface processes

2.1 Concept : Surface layer + soil model





Atmospheric surface layer : the lowest part of the atmospheric boundary layer (typically about a tenth of the height of the BL) where mechanical (shear) generation of turbulence exceeds buoyant generation or consumption. Turbulent fluxes and stress are nearly constant with height.

➔ **In atmospheric models**, it is defined the height of the lowest model level.

2.2 Surface layer parameterization

Surface layer schemes calculate friction velocities and exchange coefficients that enable the calculation of surface heat and moisture fluxes by the land-surface models. These fluxes provide a lower boundary condition for the vertical transport done in the PBL Schemes.

Over water surfaces, the surface fluxes and surface diagnostic fields are computed in the surface layer scheme itself. Sea surface temperature can be predicted by the surface energy budget and mixed layer mixing

1) Bulk method (before 1990, MM4)

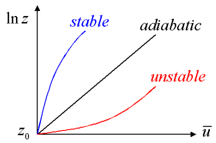
$$H_0 = \rho C_p C_H |\vec{V}_a| \Delta T$$
$$E_0 = \rho L C_H |\vec{V}_a| \Delta q M_a$$
$$\vec{\tau}_0 = \rho C_D |V_a| \vec{V}_a$$

C_D, C_H : prescribed
$= 0.01$ over land, $= 0.001$ over water

2) Monin-Obukov similarity

$$\frac{k_z}{u_*}\frac{\partial u}{\partial z} = \phi_m(z/L), \quad \frac{k_z}{u_*}\frac{\partial \theta}{\partial z} = \phi_t(z/L)$$

Integrate, $F_m = \int_{z_0}^{h_s} \frac{dz}{z}\phi_m dz = \ln(\frac{h_s}{z_0}) - \psi_m(h_s, z_0, L)$



$\bar{u} = C_u[\ln\frac{z}{z_0} + \Phi]$

Φ : curving factor

※ **Profile function** : ϕ_m and ϕ_t

Dyer and Hicks formula for similarity

- unstable (L < 0)

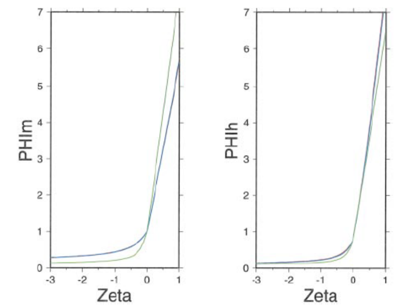
$\phi_m = (1-16\frac{0.1h}{L})^{-\frac{1}{4}}$ for u, v

$\phi_t = (1-16\frac{0.1h}{L})^{-\frac{1}{2}}$ for θ, q

- stable (L > 0)

$\phi_m = \phi_t = (1+5\frac{0.1h}{L})$

where $L = u_*^2\bar{\theta}/(kg\theta_*) = -\frac{\rho C_p \theta_0 u_*^3}{kgH_0}$



Kantha (2003)

$\frac{h_s}{L} = \frac{\phi_m^2(hs/L)}{\phi_t(hs/L)} Ri = \zeta$ (Zeta=hs/L)

where $Ri = \frac{g}{\theta}\frac{\partial\theta}{\partial z}(\frac{\partial u}{\partial z})^{-2}$

※ **Useful relation** :

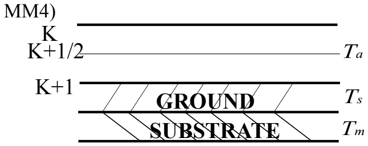
Given the F_m, F_H $C_D = k^2/F_m^2, C_Q = C_H = k^2/(F_m F_t)$ $u_* = kU/F_m$

$$\tau_0 = \rho k_m \frac{du}{dz} = -\overline{u'w'} = \rho C_d U^2$$
$$H_0 = -\rho C_p k_h \frac{d\theta}{dz} = \rho C_p \overline{\theta' w'} = -\rho C_p C_H U \Delta\theta$$
$$E_0 = -\rho L \overline{q' w'} = -\rho L C_q U \Delta q$$

2.3 Soil model

1) Slab model : force-restore method (before 1990 : MM4)

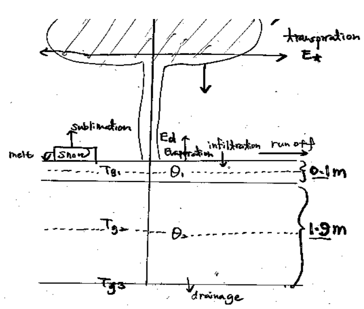
$$\frac{\partial T_s}{\partial t} = \lambda_T(R_n - LE - H) - \frac{2\pi}{\tau}(T_s - T_a)$$
$$\frac{\partial T_s}{\partial t} = \frac{1}{\tau}(T_s - T_m) \quad \textbf{: } T_m\textbf{, daily mean}$$



2) Multi-layer model : OSU method

$$\frac{\partial T_s}{\partial t} = \lambda_T(R_n - LE - H) \quad \textbf{: surface T}$$
$$(\rho C)_i \frac{\partial T_g}{\partial t} = \frac{\partial}{\partial z}(\lambda T_g \frac{\partial T_g}{\partial z}) \quad \textbf{: soil T}$$
$$\frac{\partial \Theta}{\partial t} = \frac{\partial}{\partial z}(D\frac{\partial\Theta}{\partial z}) + K\frac{\partial\Theta}{\partial z} + F_\Theta \quad \textbf{: soil moisture}$$

- NOAH, SIB, PLACE, VIC, CLM, etc

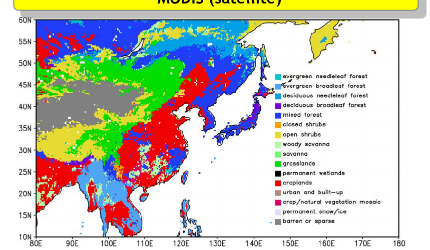


RA analysis: 2-layer soil mode l

2.4 Vegetation type → z_0, Albedo

Vegetation types

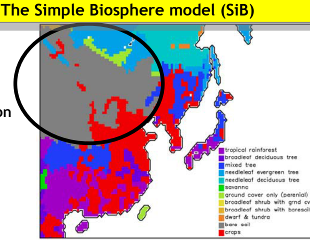
MODIS (satellite)



13 type data set
1 degree

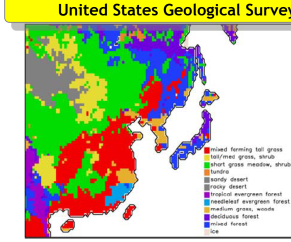
12 type data set
20 min

The Simple Biosphere model (SiB)

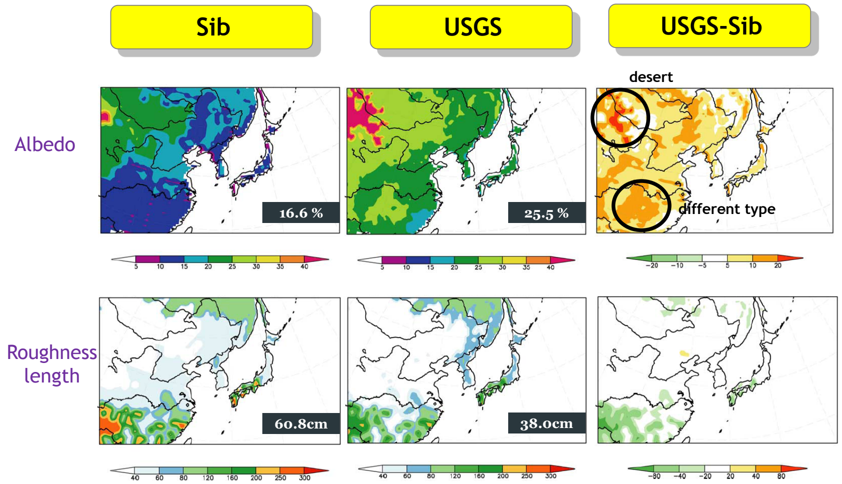


quite broad desert region

United States Geological Survey's (USGS)



Albedo and roughness length (z_0)

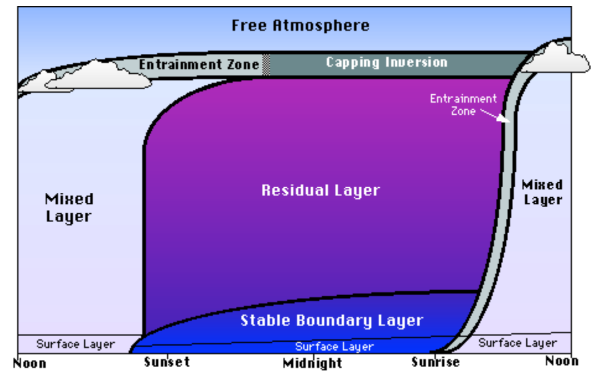


Much uncertainties in calibration/optimization !!!

3. Vertical diffusion (PBL)

3.1 Concept

- computes the parameterized effects of vertical turbulent eddy diffusion of momentum, water vapor and sensible heat fluxes



Structure of Planetary Boundary Layer
adapted from Stull, 1988.

3.2 Planetary Boundary Layer Structure : schematic

Daytime profiles

Daytime flux profiles

Nighttime flux profiles

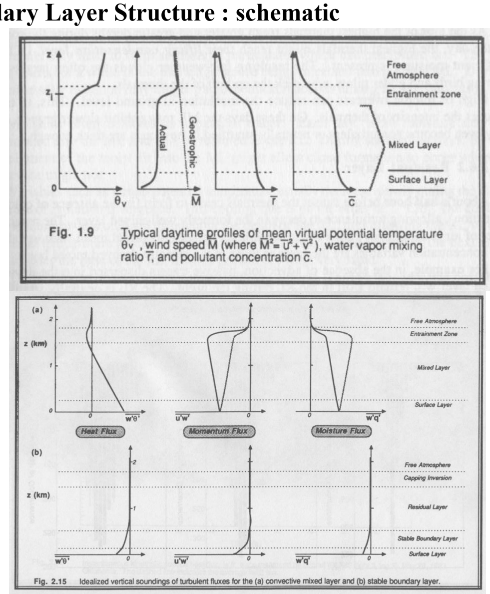


Fig. 1.9 Typical daytime profiles of mean virtual potential temperature $\bar{\theta}_v$, wind speed $\bar{M}$ (where $\bar{M}^2 = \bar{U}^2 + \bar{V}^2$), water vapor mixing ratio $\bar{r}$, and pollutant concentration $\bar{c}$.

Fig. 2.15 Idealized vertical soundings of turbulent fluxes for the (a) convective mixed layer and (b) stable boundary layer.

Stull (1988)

3.3 Classifications : how to determine, k_c

i) Local diffusion (Louis 1979)

$$\frac{\partial c}{\partial t} = \frac{\partial}{\partial z}(-\overline{wc}) = \frac{\partial}{\partial z}(k_c \frac{\partial c}{\partial z}) \quad k_c: \text{ diffusivity}, \quad k_m, k_t = l^2 f_{m,t}(Ri) \left| \frac{\partial U}{\partial z} \right|$$

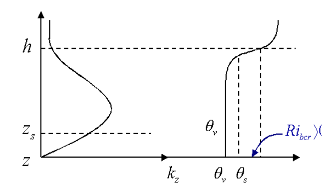
Local Richardson number

ii) Nonlocal diffusion (Troen and Mahrt 1986)

$$\frac{\partial c}{\partial t} = \frac{\partial}{\partial z}(-\overline{wc}) = \frac{\partial}{\partial z}(k_c(\frac{\partial c}{\partial z} - \gamma_c))$$

$$k_{zm} = k w_s z (1 - \frac{z}{h})^p, \; h = R_{ibcr} \frac{\theta_m}{g} \frac{U^2(h)}{(\theta_v(h) - \theta_s)}$$

$$\theta_s = \theta_{va} + \theta_T \; (= b \frac{\overline{(\theta_v ' w')_0}}{w_s}), \; w_s = u_* \phi_m^{-1}$$



iii) Eddy mass-flux diffusion (Siebesma et al. 2007)

$$\frac{\partial c}{\partial t} = \frac{\partial}{\partial z}(-\overline{wc}) = -\frac{\partial}{\partial z}[-k_c \frac{\partial \bar{c}}{\partial z} + M(c_u - \bar{c})]$$

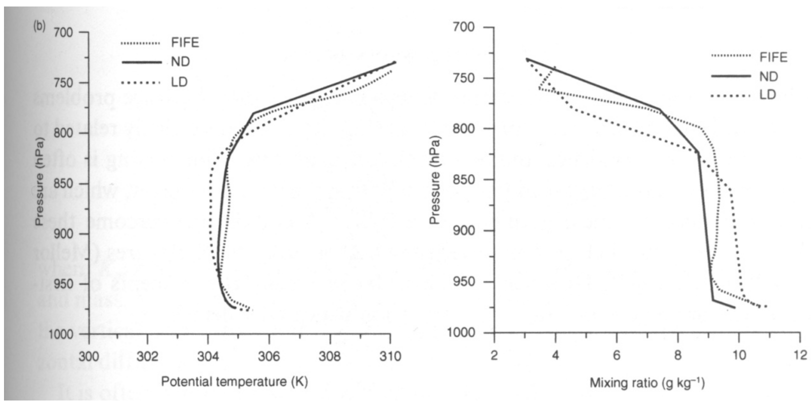
small eddies   strong updrafts

iv) TKE (Turbulent Kinetic Energy) diffusion (Mellor and Yamada 1982)

$$\text{TKE equation}: \quad \frac{\partial \overline{u_i u_j}}{\partial t} + u_j \frac{\partial \overline{u_i u_j}}{\partial x_j} = -\frac{\partial}{\partial x_k}[\overline{u_i u_j u_k} + \frac{1}{\rho} \cdots]$$

$$\overline{u_i u_j} \Longrightarrow k_z = \text{fn (TKE)}$$

3.4 Local versus nonlocal



Local scheme (LD) typically produces unstable mixed layer in order to transport heat upward. But, observation (FIFE) shows nearneutral or slightly stable BL in the upper part of BL

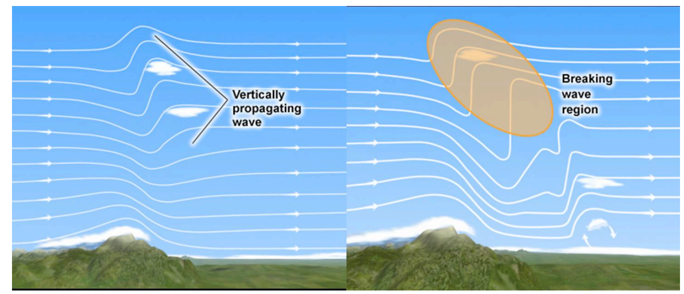
Hong and Pan (1996)

4. Gravity Wave Drag

- GWDO : GWD induced by sub-grid scale orography
- GWDC : GWD induced by precipitating deep convection

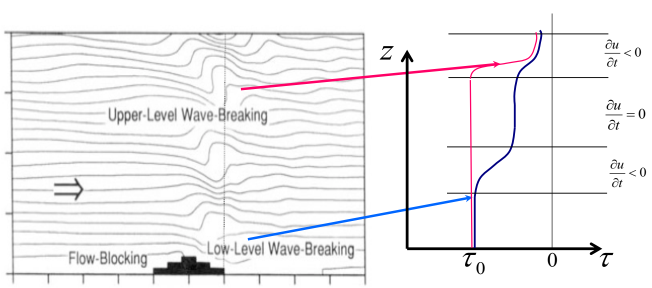
4.1 Concept

This scheme (GWDO) includes the effect of mountain induced gravity wave drag from sub-grid scale orography including convective breaking, shear breaking and the presence of critical levels. Effects are strong in the presence of strong vertical wind shear and thermally stable layer.



* In smoothed model orography, momentum stress near mountain cannot be generated

4.2 Enhanced lower tropospheric gravity wave drag (Kim and Arakawa 1995)



$$\frac{\partial u}{\partial t} < 0 \qquad \frac{\partial u}{\partial t} = 0 \qquad \frac{\partial u}{\partial t} < 0$$

$$\frac{\partial u}{\partial t} = -\frac{1}{\rho}\frac{\partial \tau_x}{\partial z}$$

$$\frac{\partial v}{\partial t} = -\frac{1}{\rho}\frac{\partial \tau_y}{\partial z}$$

Stress at reference level : $\tau_0 = -E\frac{1}{\Delta x}\frac{\rho_0 U_0^3}{N_0}\frac{Fr^2}{Fr^2 + 0.5/OC}, \; U_0 = \frac{1}{h}\int_{k=1}^{k=kpbl} U dz$

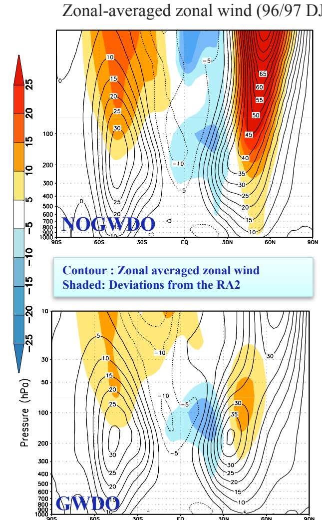
Reference level (KA95) : Max (2, KPBL)

Earlier method : the conventional Ri number-based wave- breaking mechanism using the saturation hypothesis, which works mainly in the upper atmosphere

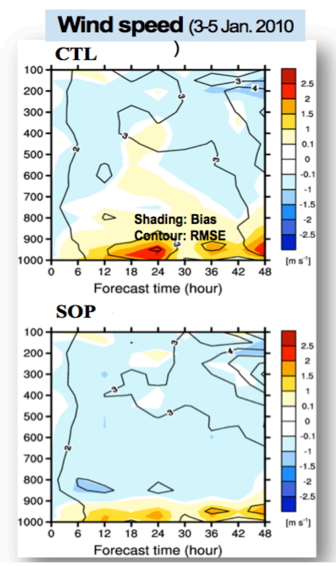
Advanced: the higher-moment orographic statistics-based wave- breaking mechanism using half-theory (Scorer parameter ~ BVF**2 / U**2) and half-empiricism obtained from mesoscale mountain wave simulations, which breaks in the lower atmosphere as well as upper layer, together with flow blocking

4.3 Impact of GWDO

Zonal-averaged zonal wind (96/97 DJF)



➔ Improves the upper level jets (Kim and Hong, 2009)

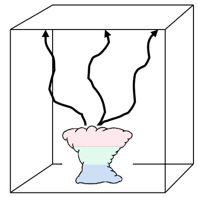


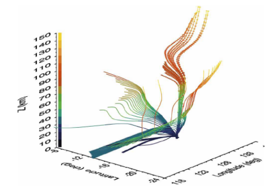
Flow blocking ➔ Improves the low-level winds (Choi and Hong, 2015)

4.4 Convective GWD (CGWD) or non-orographic GWD

CGWD parameterizations

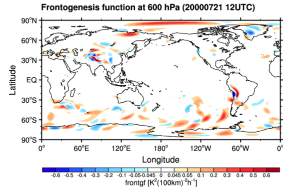
- Chun and Baik (1998, 2002): The momentum flux spectrum for the CGWD parameterization was **first analytically formulated**
- Song and Chun (2005): **Nonstationary parts** of convective GWs were included in the parameterization
- Song and Chun (2008): **Ray-based parameterization** that can represent a three-dimensional propagation of GWs was developed
- Choi and Chun (2011): Two free parameters, the **moving speed of the convective source** and **wave-propagation direction**, were determined





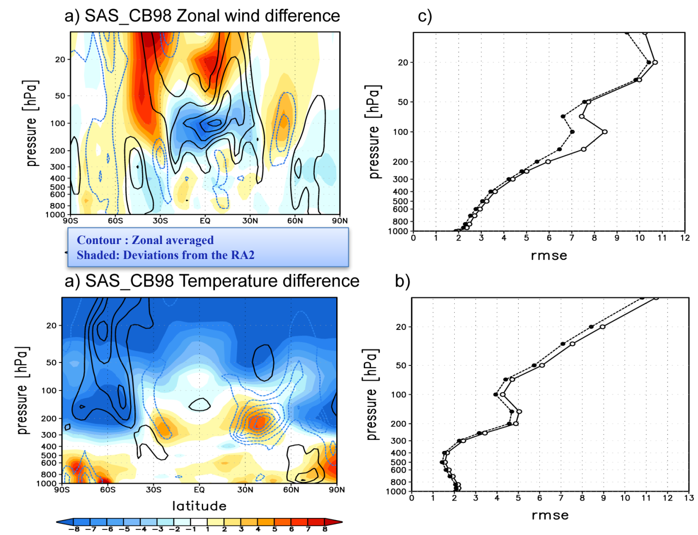
Jet/Front GWD parameterizations

- At this moment, there is **no source-level momentum flux formulation** of jet/front GWs, due primarily to the lack of complete understanding of generation mechanisms
- Charron and Manzini (2002), Richter et al. (2010): **Frontogenesis function** was adopted to diagnose the generation of frontal GWs



* It is very rare to have this CGWD or no_mtn_GWD

4.5 Improvement by GWDC (Chun and Baik 1998, Jeon et al. 2010)



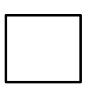
Wind and temperature RMSE are reduced in upper atmosphere

Precipitation Processes

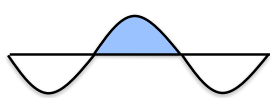
Concept : precipitation algorithms (CPS and MPS)

- In real atmosphere, dynamical motion -> RH > 1 => clouds form -> produces rain
- In modeled atmosphere, RH < 1

  But generate clouds by sub-grid scale motion → requires parameterized process
  (it is often related to the deep convection processes)
  Deep convection : 2~10 km



$\overline{RH} < 1$



$RH = 1$

Δx ==> 0 , more grid-resolvable precipitation

Thus, we need the cumulus parameterization scheme to account for releasing conditional instability due to subgrid scale motion

- Grid-resolvable (Microphysics scheme : MPS) : Supersaturation → clouds
- Subgrid scale (Cumulus parameterization scheme : CPS) : CAPE removal → clouds

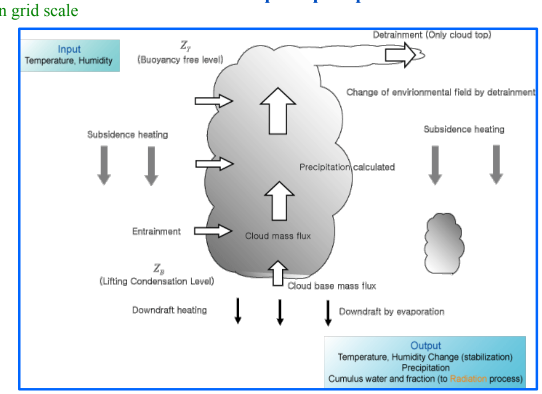
5. Cumulus parameterization scheme

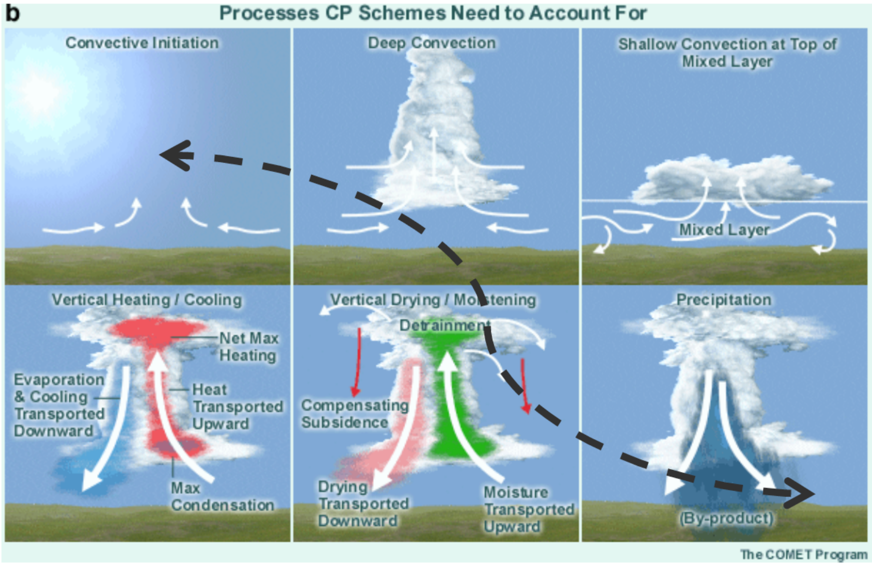
5.1 Concept

- represents deep precipitating convection and feedback to large-scale
- must formulate the collective effects of subgridscale clouds in terms of the prognostic variable in grid scale

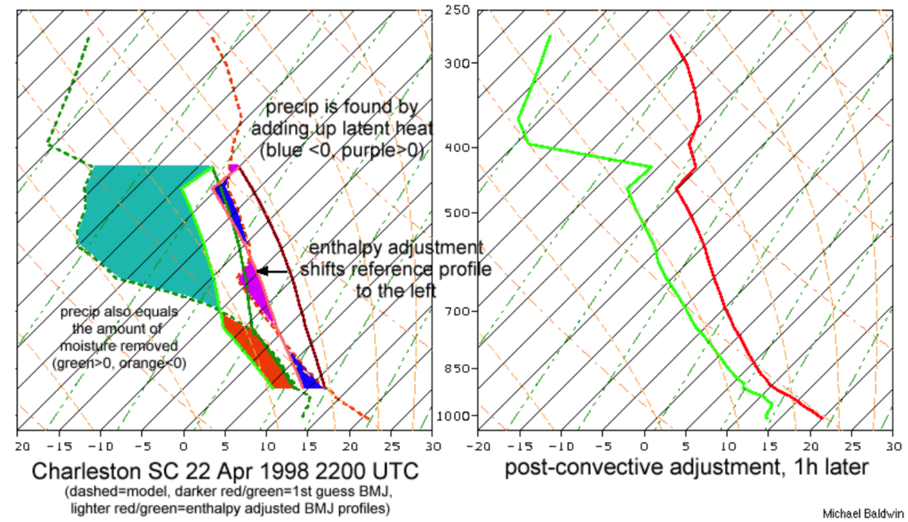
Parameterized convection
Deep convection
Subgridscale precipitation
Implicit precipitation

Jul 25, 2010, Seoul





CPS does not consider the detailed evolution of convection !



CPS consider changes in profile before and after convection

5.2 Kuo scheme (1965)

: Cloud is formed in proportional to column-integrated moisture convergence

$$M_t = -\frac{1}{g}\int_0^{P_s} \nabla\cdot(vq)dp + F_{g_s}$$

$$\int_0^{P_s} \frac{\partial q}{\partial t}dp = gbM_t\ ,\quad \text{b : moistening factor}$$

$$\int_0^{P_s} \theta_c dp = gL(1-b)M_t \qquad \frac{\theta_c}{\pi} = \frac{\theta_a - \theta}{\tau}$$

- Modified Kuo scheme

Krishnamurti el al. (1980, 1983) : proportional to vertical advection of moisture

$$M_t = -\frac{1}{g}\left\{\int_0^{P_s} \omega\frac{\partial q}{\partial p}dp + F_{q_s}\right\}$$

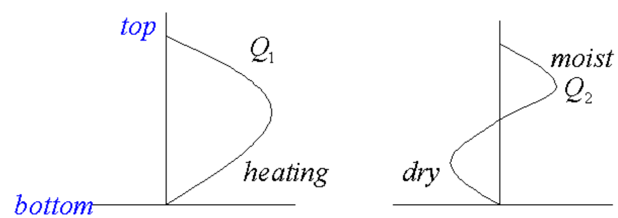
Anthes (AK : 1977) : a revised moistening factor and parcel buoyancy

$$b = \left(\frac{1-RH}{1-RH_c}\right)^n$$

- Heating and moistening profiles (prescribed)

$$\frac{d\theta}{dt} = \frac{1}{\pi}\left[gL(1-b)M_t Q_1 + Q_r\right]$$

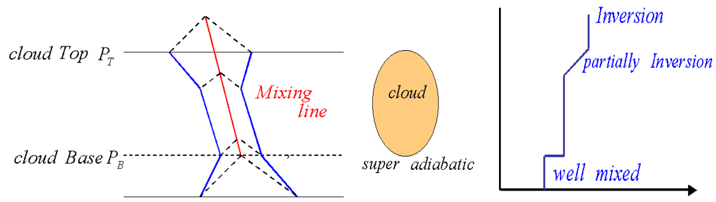
$$\frac{dq}{dt} = -g(1-b)M_t Q_2 \qquad \int_0^{P_s} Q_1 dp = \int_0^{P_s} Q_2 dp = 1$$



- Kuo type scheme has been widely used before 1990

5.3 Betts-Miller scheme (1986)

: adjust toward reference profiles that are based on observational evidence of convective equilibrium



Reference profile to be adjusted : originally based on tropical cyclones

$$\theta_R'(P)=\bar{\theta}(P_B)+\beta\, M_\theta(P-P_B)$$

$$\frac{\partial q}{\partial p}=\beta\left(\frac{\partial q}{\partial p_*}\right)_M \qquad \beta=\frac{\partial p^*}{\partial p} \qquad p^* \text{ : saturation pressure (= 1.2 for example)}$$

$$M_\theta=0.85\left(\frac{\partial \theta^*}{\partial P^*}\right)_M \qquad \text{M : Mixing line}$$

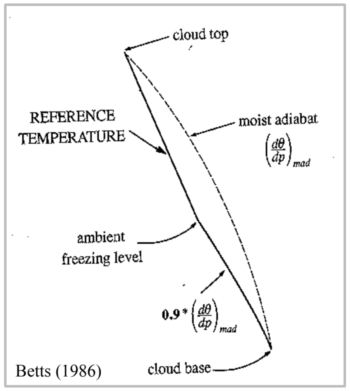
Energy Constraints : $\int_{P_B}^{P_{T+1}} C_P\left(T_R-\bar{T}\right)dp=\int_{P_B}^{P_{T+1}}\left(q_R-\bar{q}\right)dp=0$

- Convective tendencies & Precipitation

$$\left(\frac{\partial \bar{T}}{\partial t}\right)_{Cu}=\frac{T_R-\bar{T}}{\tau}$$

$$\left(\frac{\partial \bar{q}}{\partial t}\right)_{Cu}=\frac{q_R-\bar{q}}{\tau}$$

$$\text{Precip}=\int_{P_0}^{P_T}\left(\frac{q_R-\bar{q}}{\tau}\right)\frac{dP}{g}=-\frac{C_P}{L}\int_{P_0}^{P_T}\left(\frac{T_R-\bar{T}}{\tau}\right)\frac{dp}{g}$$



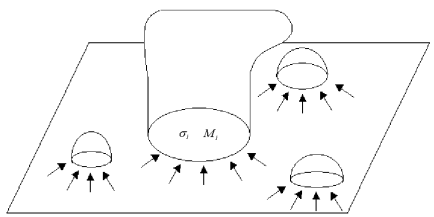
- Remarks : -Manabe (hard adjustment : toward a moist adiabat)
-Kuo, BM (soft adjustment : toward reference profiles

5.4 Mass-flux schemes : Arakawa-Schubert (1974)

i) Concept

-- Mass flux approach, cloud ensemble, quasi-equilibrium
-- Theoretical frame work for CPS

- Area is large enough so that cloud ensemble can be a statistical entity
- Area is small enough so that cloud environment is approximately uniform horizontally



M_i : vertical mass flux through ith cloud

σ_i : fractional area covered by ith cloud

$M_c=\sum_i M_i$: total vertical mass flux

$\rho M=M_c+\tilde{M}$ (environment)

: net mass flux/unit large-scale horizontal area

ii) Quasi-equilibrium : cloud forcing ~ large-scale adjustment

: CPS computes the warming (cooling) in the grid box due to adiabatic descent (ascent), rather than computing latent heat release in cloud models

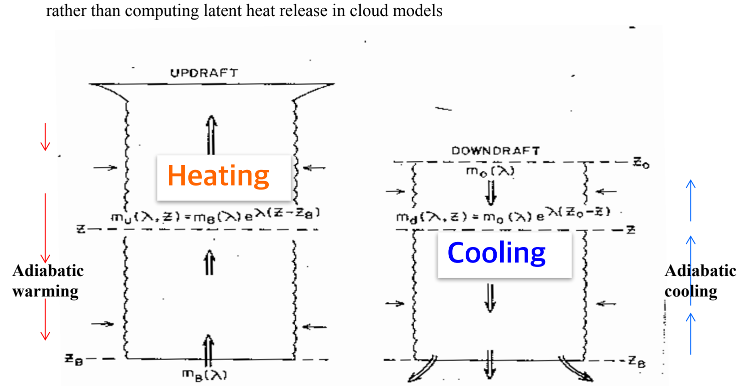


FIG. 4.10. Model for updraft and downdraft of cloud type λ (from Johnson 1976).

iii) Energy budget equations

Large-scale flux across grid box — Exchange of S between environment and clouds

$$\frac{\partial}{\partial t}\rho\left(1-\sigma_c\right)\tilde{s} = -\bar{\nabla}\cdot\overline{\left(\rho\tilde{V}S\right)} - \frac{\partial}{\partial Z}\left(\tilde{M}\tilde{S}\right) - \sum_i\left(\frac{\partial M_i}{\partial Z}+\rho\frac{\partial \sigma_i}{\partial t}\right)S_{i_b} - LE + \tilde{Q}_R \quad \text{: Environment MSE}$$

$$\frac{\partial}{\partial t}\rho\sum\sigma_i s_i = -\frac{\partial}{\partial z}\left(\sum_i M_i s_i\right) + \sum_i\left(\frac{\partial M_i}{\partial z}+\rho\frac{\partial \sigma_i}{\partial t}\right)S_{ib} + \sum_x\left(LC_i + Q_{Ri}\right) \quad \text{: in-cloud MSE}$$

$S_i : C_pT + gz$ (dry static energy) of i[th] cloud

$S_{ib} : C_pT + gz$ of the air entraining into or detraining from the i[th] cloud

C_i : condensation in the i[th] cloud

E : evaporation of liquid water in the environment

Q_r : Radiative heating

- Entrainment : $\frac{\partial M_i}{\partial z} + \rho\frac{\partial \sigma_i}{\partial t} > 0, \quad S_{ib} = \tilde{S}$
- Detrainment : $\frac{\partial M_i}{\partial z} + \rho\frac{\partial \sigma_i}{\partial t} < 0, \quad S_{ib} = S_i$

iv) Approximation

- Assume $\sigma_c \ll 1$, $\bar{s} \simeq \tilde{s}$

$$\frac{\partial}{\partial t}\rho\bar{s} = -\nabla\cdot(\rho\overline{v}\overline{s}) - \frac{\partial}{\partial z}(\rho\overline{w}\overline{s}) - \nabla\cdot\overline{(\rho\bar{v}s - \rho\overline{v}\overline{s})}$$

$$+ M_c\frac{\partial\bar{s}}{\partial z} - \sum_{dc}\left(\frac{\partial M_i}{\partial z} + \rho\frac{\partial\sigma_i}{\partial t}\right)(\delta_i - \bar{s}) - LE + \tilde{Q}_R$$

Adiabatic warming due to hypothetical subsidence between the clouds

Detraining clouds

detrainment, entrainment

$$\frac{\partial}{\partial t}\rho\bar{q} = -\nabla\cdot(\rho\overline{v}\overline{q}) - \frac{\partial}{\partial z}(\rho\overline{w}\overline{q}) - \nabla\cdot\overline{(\rho\bar{v}q - \rho\overline{v}\overline{q})}$$

$$+ M_c\frac{\partial\bar{q}}{\partial z} - \sum_{dc}\left(\frac{\partial M_i}{\partial z} + \rho\frac{\partial\sigma_i}{\partial t}\right)(q_i - \bar{q}) - E$$

- Spectral cloud ensemble :

$$M_c(z) = \int_0^{\lambda_{max}} m(z,\lambda)\,d\lambda$$

Sub-ensemble
mass flux of between λ and $d\lambda+\lambda$

$$= \int_0^{\lambda_{max}} m_B(\lambda)\eta(z,\lambda)\,d\lambda$$

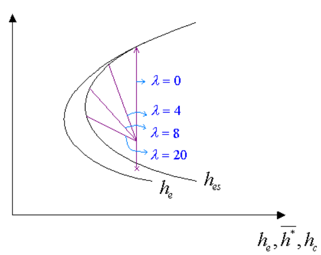
Mass flux at cloud base

$$\eta(z,\lambda) \equiv \frac{m(z,\lambda)}{m_B(\lambda)}$$; normalized subensemble mass flux

v) Closure

$$\frac{\partial m(z,\lambda)}{\partial z} = \mu(z,\lambda)\eta(z,\lambda)$$

$$\eta(z,\lambda) = e^{\lambda(z-z_B)}$$; mass flux profile



- Cloud work function : measure of buoyancy

$$A(\lambda) = \int_{z_B}^{z_D(\lambda)} \eta(z,\lambda)g\frac{T_c(z,\lambda)-\bar{T}(z)}{\bar{T}}dz$$

- Q-G equilibrium

$$\frac{dA(\lambda)}{dt} = \left.\frac{dA(\lambda)}{dt}\right|_{LS} + \left.\frac{dA(\lambda)}{dt}\right|_{C} \simeq 0$$

Large-scale forcing
>0 : destabilized

Adjustment
<0 : stabilization

Kernel : Cloud scheme kinetic energy

$$K_{ij} = \frac{A_i' - A_i}{(m_B\Delta t)} \qquad \sum_j^{i_{max}} K_{ij}(m_B\Delta t)_j + F_i = 0 \;\Rightarrow m_B$$

⟶ compute $\frac{\partial\bar{s}}{\partial t}, \frac{\partial\bar{q}}{\partial t}$ with η, m_B

5.5 Other schemes

*** Arakawa-Schubert type mass flux schemes**

Grell scheme (1993) : Updraft/downdraft couplet without lateral mixing to find the deepest cloud
Simplified AS (SAS, Han and Pan 2011): revised cloud physics from the Grell
Relaxed AS (RAS, Moorthi and Suarez 1992): linearized profile function

*** Other mass flux schemes : Low-level control convective schemes** (Stensrud 2007)

Kain and Fritsch (2004) : CAPE based sophisticated convective plume model
Emanuel (1991) : Stochastic mixing cloud model
Tiedtke (1989) : Large-scale moisture convergence (KUO) based mass flux (ECMWF IFS model)
Gregory-Rowntree (1990): Parcel buoyancy based turbulence in cloud model (UK model)

6. Shallow Convection

6.1 Concept

more vigorous vertical mixing of q and T above the mixed layer top. With the enhanced vertical eddy transport between LCL and inversion level, this process does not allow the excess moisture trapped near the surface in synoptically inactive regions (non-precipitating convection).

➔ Cooling and moistening above LCL and heating and warming below.

6.2 Classification

- Moist adjustment type : Betts and Miller (1993), Lock et al. (2000), Tiedtke (1983)
- Mass flux type : Kain (2004), Park and Bretherton (2009), Han and Pan (2011)

Tiedtke (1983)

$$\frac{\partial T}{\partial t} = \frac{1}{\rho}\frac{\partial}{\partial z}\left(\rho K\left[\frac{\partial T}{\partial z} + \Gamma\right]\right)$$

$$\frac{\partial q}{\partial t} = \frac{1}{\rho}\frac{\partial}{\partial z}\left(\rho K\frac{\partial q}{\partial z}\right)$$

Han and Pan (2011)

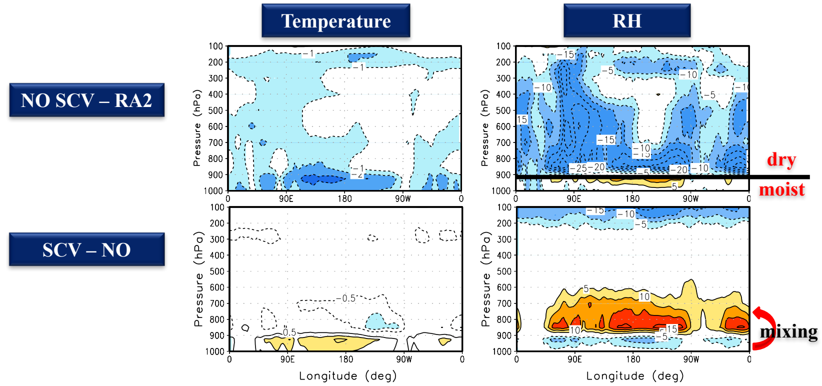
$$\frac{1}{\eta}\frac{\partial \eta}{\partial z} = \varepsilon - \delta$$

$$\frac{\partial(\eta s)}{\partial z} = (\varepsilon\bar{s} - \delta s)\eta$$

$$\frac{\partial[\eta(q_v + q_l)]}{\partial z} = \eta[\varepsilon\bar{q}_v - \delta(q_v + q_l) - r]$$

6.3 Impact of the shallow convection scheme (SCV)

: JJA 1996 simulation in a GCM



* SCV plays crucial role over the oceans.

7. Microphysics scheme

large-scale precipitation
grid-resolvable scale precipitation
explicit moisture scheme
cloud scheme
non-convective precipitation scheme

7.1 Concept

- Remove supersaturation after deep and shallow convection, and feedback to large-scale

7.2 Classification : according to the complexity in microphysics

i) Diagnostic : condensation, evaporation of falling precipitation

ii) Bulk microphysics : hydrometeors with size distribution in inverse-exponential function
- Single moment : predict mixing ratios of hydrometeors
- Double moment : + number concentrations
- Triple moment : + reflectivity

iii) Bin microphysics : divides the particle distribution into a number of finite size or mass categories.

7.3 Precipitate size distributions

Marshall and Palmer(1948) : exponential law
Heymsfield and Platt (1984) : Power law

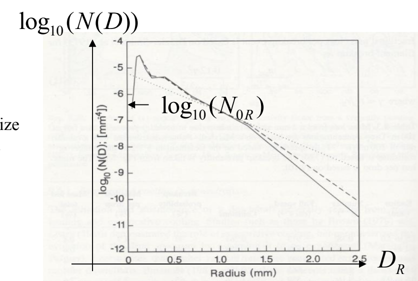
$$N_R(D_R) = aD_R^{\ b}$$

The rain and snow particles are assumed to follow the size distribution derived by Marshall and Palmer(1948), and Gunn and Marshall(1958), respectively. The size distributions for both rain and snow are formulated according to an inverse-exponential distribution and its formula for rain can be expressed by

$$N_R(D_R) = N_{0R}\exp(-\lambda_R D_R)$$

for rain, where N_{0R} is the intercept parameter of the rain distributions. Slope parameter is

$$\lambda_R = \left(\frac{\pi\rho_w N_{0R}}{\rho q_R}\right)^{1/4}$$



$$\Gamma(x) = \int_0^\infty t^{x-1}\exp(-t)dt$$

$$\int_0^\infty D_R^{\ 4-1}\exp(-\lambda_R D_R)dD_R$$

$$= \Gamma(4)/\lambda_R^{\ 4}$$

Due to the size distribution in exponential manner (integration of precip for whole size results in constant), we can apply the bulk property microphysics terms.

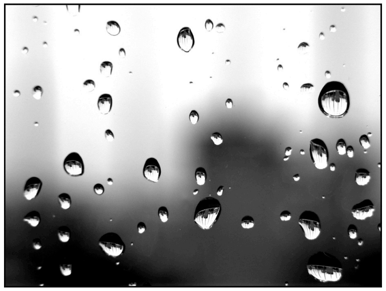
7.4 Bulk Method : 1-Moment versus 2-Moment

Mixing Ratio (1-moment/ 2-moment scheme)

$$\left(\int \frac{dM(D_R)}{dt} dN_{DR}\right) / \rho = \frac{dq}{dt} (kgkg^{-1}s^{-1})$$

Number concentration (2-moment scheme)

$$\left(\int \frac{d\,\mathrm{Pr}ob(D_R)}{dt} dN_{DR}\right) = \frac{dN}{dt} (m^{-3}s^{-1})$$



Single moment scheme

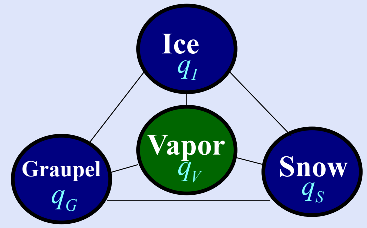
$$dN_{DR} = N_{0R} \exp(-\lambda_R D_R) dD_R$$

Double moment scheme

$$dN_{DR} = N_R \lambda_R^2 (N_R) D_R \exp(-\lambda_R D_R) dD_R$$

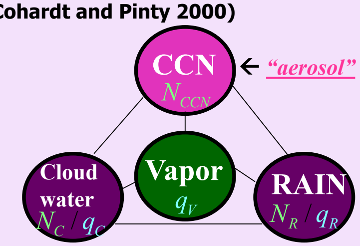
7.5 Bulk Method : 1-Moment (WSM) versus 2-Moment (WDM)

Cold rain processes :
(Hong et al. 2004; Hong and Lim 2006)



q for 4 hydrometeors will be predicted (Single Moment)

Warm rain processes :
(Khairoutdinov and Kogan 2000; Cohardt and Pinty 2000)



N,q for 2 hydrometeors will be predicted (Double Moment)

N: Cloud water, Rain, CCN

q: Cloud water, Rain, Ice, Snow, Graupel, Vapor

WDM6 (Lim and Hong, 2010)

7.6 WDM6 code

**** Warm rain processes** (Lim and Hong 2010)

```
! warm rain processes
!
! - follows the double-moment processes in Lim and Hong
!
!=================================================================
!
    do k = kts, kte
      do i = its, ite
        supsat = max(q(i,k),qmin)-qs(i,k,1)
        satdt = supsat/dtcld
!-----------------------------------------------------------------
! praut: auto conversion rate from cloud to rain  [LH 9] [CP 17]
!        (QC->QR)
!-----------------------------------------------------------------
        lencon  = 2.7e-2*den(i,k)*qci(i,k,1)*(1.e20/16.*rslopec2(i,k)   &
                  *rslopec2(i,k)-0.4)
        lenconcr = max(1.2*lencon, qcrmin)
        if(avedia(i,k,1).gt.di15) then
          taucon = 3.7/den(i,k)/qci(i,k,1)/(0.5e6*rslopec(i,k)-7.5)
          praut(i,k) = lencon/taucon
          praut(i,k) = min(max(praut(i,k),0.),qci(i,k,1)/dtcld)
!-----------------------------------------------------------------
! nraut: auto conversion rate from cloud to rain [LH A6] [CP 18 & 19]
!        (NC->NR)
!-----------------------------------------------------------------
          nraut(i,k) = 3.5e9*den(i,k)*praut(i,k)
          if(qrs(i,k,1).gt.lenconcr)                                   &
          nraut(i,k) = ncr(i,k,3)/qrs(i,k,1)*praut(i,k)
          nraut(i,k) = min(nraut(i,k),ncr(i,k,2)/dtcld)
        endif
!-----------------------------------------------------------------
! pracw: accretion of cloud water by rain     [LH 10] [CP 22 & 23]
!        (QC->QR)
! nracw: accretion of cloud water by rain     [LH A9]
!        (NC->)
!-----------------------------------------------------------------
        if(qrs(i,k,1).ge.lenconcr) then
          if(avedia(i,k,2).ge.di100) then
            nracw(i,k) = min(ncrk1*ncr(i,k,2)*ncr(i,k,3)*(rslopec3(i,k)   &
                         + 24.*rslope3(i,k,1)),ncr(i,k,2)/dtcld)
            pracw(i,k) = min(pi/6.*(denr/den(i,k))*ncrk1*ncr(i,k,2)      &
                         *ncr(i,k,3)*rslopec3(i,k)*(2.*rslopec3(i,k)     &
                         + 24.*rslope3(i,k,1)),qci(i,k,1)/dtcld)
          else
            nracw(i,k) = min(ncrk2*ncr(i,k,2)*ncr(i,k,3)*(2.*rslopec3(i,k)   &
                         *rslopec3(i,k)+5040.*rslope3(i,k,1)             &
                         *rslope3(i,k,1)),ncr(i,k,2)/dtcld)
            pracw(i,k) = min(pi/6.*(denr/den(i,k))*ncrk2*ncr(i,k,2)      &
                         *ncr(i,k,3)*rslopec3(i,k)*(6.*rslopec3(i,k)     &
                         *rslopec3(i,k)+5040.*rslope3(i,k,1)*rslope3(i,k,1))   &
                         ,qci(i,k,1)/dtcld)
          endif
        endif
```

***Auto conversion from cloud to rain [C→ R]**

$$\mathrm{Praut}\ [\mathrm{kgkg^{-1}s^{-1}}] = L/\tau \qquad L = 2.7\times10^{-2}\rho_a q_C\left(\frac{10^{20}}{16\lambda_C^4} - 0.4\right)$$

$$\tau = 3.7\frac{1}{\rho_a q_C}\left(\frac{0.5\times10^6}{\lambda_C} - 7.5\right)^{-1}$$

$$\mathrm{Nraut}[\mathrm{m^{-3}s^{-1}}] = 3.5\times10^9 \frac{\rho_a L}{\tau}$$

***Accretion of cloud water by rain [C→ R]**

$D_R \geq 100$ μm

$$\mathrm{Pracw}\ [\mathrm{kgkg^{-1}s^{-1}}] = \frac{\pi}{6}\frac{\rho_W}{\rho_a}K_1\frac{N_C N_R}{\lambda_C^3}\left\{\frac{2}{\lambda_C^3} + \frac{24}{\lambda_R^3}\right\}$$

$$\mathrm{Nracw}[\mathrm{m^{-3}s^{-1}}] = -K_1 N_C N_R\left\{\frac{1}{\lambda_C^3} + \frac{24}{\lambda_R^3}\right\}$$

$D_R < 100$ μm

$$\mathrm{Pracw}\ [\mathrm{kgkg^{-1}s^{-1}}] = \frac{\pi}{6}\frac{\rho_W}{\rho_a}K_2\frac{N_C N_R}{\lambda_C^3}\left\{\frac{6}{\lambda_C^6} + \frac{5040}{\lambda_R^6}\right\}$$

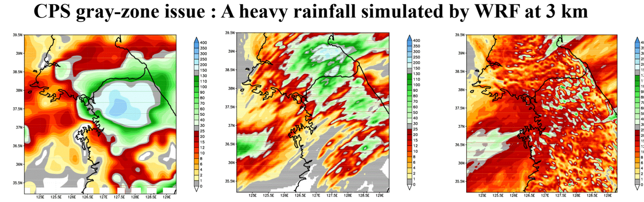
$$\mathrm{Nracw}[\mathrm{m^{-3}s^{-1}}] = -K_2 N_C N_R\left\{\frac{2}{\lambda_C^6} + \frac{5040}{\lambda_R^6}\right\}$$

Resolution Dependency

Cut-off horizontal grid length: current status in research communities

- PBL : ~50 m (Mirocha, 2008 WRF workshop)
- GWDO : ~ 3 or 1 km (hydrostatic approximation)
- GWDC: ~ 3 or 1 km (go with CP)
- Cumulus parameterization : ~ 3 or 1 km (cloud resolving scale)

CPS gray-zone issue : A heavy rainfall simulated by WRF at 3 km



OBS **No CPS** **With CPS**

Gray-zone : partly resolved and partly parameterized (Hong and Dudhia 2012)

- CPS (1 km~10 km) : Gerard, Grell and Freitas, Arakawa and Ming, Pan and Han, Kwon and Hong
- PBL (100 m~1 km) : Honnert, Boutle, Shin and Hong
- Other processes such as shallow convection may also consider gray-zone

References

Anthes, R. A., 1977: A cumulus parameterization scheme utilizing a one-dimensional cloud model. *Mon. Wea. Rev.,* 105, 270-286.

Arakawa, A., and C.-M. Wu, 2013: A unified representation of deep moist convection in numerical modeling of the atmosphere. Part 1. *J. Atmos. Sci.*, gttps://doi.org/10.1175/JAS-D-12-0330.

Arakawa, A., and W. H. Schubert, 1974: Interaction of a cumulus cloud ensemble with the large-scale environment, Part I. *J. Atmos. Sci.,* 31, 674-701.

Betts, A. K., 1986: A new convective adjustment scheme. Part I: Observational and theoretical basis. *Quart. J. Roy. Meteor. Soc.,* 112, 677 692.

Betts, A. K., and M. J. Miller, 1986: A new convective adjustment scheme. Part II: Single column tests using GATE wave, BOMEX, ATEX and arctic air-mass data sets. *Quart. J. Roy. Meteor. Soc.*, 112, 693-709.

Betts, A. K., and M. J. Miller, 1993: The Betts-Miller scheme. The representation of cumulus convection in numerical model pp107-121(book).

Boutle, I. A., R. J. Beare, S. E. Belcher, A. R. Brown, and R. S. Plant,2010: The moist boundary layer under a mid-latitude weather system. *Boundary-layer Meteor.,* 134:367-386, DOI 10.1007/s10546-009-9452-9.

Charron, M., and E. Manzini, 2002: Gravity waves from fronts: Parameterization and middle atmosphere response in a general circulation model. *J. Atmos. Sci.,* 59, 923–941.

Chen, F., and J. Dudhia, 2001: Coupling an advanced land-surface/hydrology model with the Penn State/NCAR MM5 modeling system. Part I: Model description and implementation. *Mon. Wea. Rev.,* 129, 569-585.

Choi, H.-J., and H.-Y. Chun, 2011: Momentum flux spectrum of convective gravity waves. Part I: An update of a parameterization using mesoscale simulations. *J. Atmos. Sci.*, 68, 739–759.

Choi, H.-j., S.-Y. Hong, 2015: An updated subgrid orographic parameterization for global atmospheric forecast models. *J. Geophys. Res.: Atmospheres*, 120, doi:10.1002/2015JD024230.

Chun, H.-Y., and J.-J. Baik, 1998: Momentum flux by thermally induced internal gravity waves and its approximation for large-scale models. *J. Atmos. Sci.*, 55, 3299-3310

Chun, H. Y., and J. J. Baik, 2002: An updated parameterization of convectively forced gravity wave drag for use in large-scale models. *J. Atmos. sci*., 59, 1006-1017.

Chou, C., and J. Dabid Neelin, 1999: Cirrus detrainment-temperature feedback. *Geophys. Res. Lett*., 26, 9, 1295-1295.

Cohard, J.-M., and J.-P. Pinty, 2000: A comprehensive two-moment warm microphysical bulk scheme. I: Description and tests. *Quart. J. Roy. Meteor. Soc*., 126, 1815-1842.

Dudhia, J., 1989: Numerical Study of Convection Observed during the Winter Monsoon Experiment Using a Mesoscale Two-Dimensional Model. *J. Atmos. Sci.*, 46, 3077–3107

Dyer, A. J., and B. B. Hicks, 1970: Flux-gradient relationships in the constant flux layer. *Quart. J. Roy. Meteor. Soc*., 96, 715-721.

Emanual, K. A., 1991: A scheme for representing cumulus convection in large-scale models. *J. Atmos. Sci*., 48, 2123-2335.

Gerard, L., J. M. Piriou, R. Brožková, J. F. Geleyn, and D. Banciu, 2009: Cloud and precipitation parameterization in a meso-gamma-scale operational weather prediction model. *Mon. Wea. Rev*., 137, 3960-3977.

Gregory, D., and P. R. Rowntree, 1990: A mass flux convection scheme with representation of cloud ensemble characteristics and stability-dependent closure. *Mon. Wea. Rev*., 118, 1483-1506.

Grell, G. A., 1993: Prognostic evaluation of assumptions used by cumulus parameterizations. *Mon. Wea. Rev*., 121, 764-787.

Grell, G. A. and Freitas, S. R., 2014: A scale and aerosol aware stochastic convective parameterization for weather and air quality modeling. *Atmos. Chem. Phys*., 14, 5233-5250.

Gunn, K. L. S., and J. S. Marshall, 1958: The distribution with size of aggregate snowflakes. *J. Meteor.*, 15, 452-461.

Han, J., and H.-L. Pan, 2011: Revision of convection and vertical diffusion schemes in the NCEP global forecast system. *Wea. and Forecasting*, 26, 520-533.

Heymsfield, A. J., and C. M. R. Platt, 1984: A parameterization of the particle size spectrum of ice clouds in terms of the ambient temperature and the ice water content. *J. Atmos. Sci*., 41, 846- 855

Hong, S.-Y., and H.-L. Pan, 1996: Nonlocal boundary layer vertical diffusion in a Medium-Range Forecast model. *Mon. Wea. R*ev., 124, 2322-2339.1

Hong, S.-Y., and J.-O. J. Lim, 2006: The WRF single-moment 6-class microphysics scheme (WSM6). *J. Korean Meteor. Soc.,* 42, 129-151.

Hong, S.-Y., and J. Dudhia, 2012: Next-Generation Numerical Weather Prediction: Bridging parameterization, explicit clouds, and large eddies. *Bull. Amer. Meteor. Soc*., 93(1), ES6–ES9, doi: 10.1175/2011BAMS3224.1.

Hong, S.-Y., J. Dudhia, and S.-H. Chen, 2004: A revised approach to ice microphysical processes for the bulk parameterization of clouds and precipitation. *Mon. Wea. Rev*., 132, 103-120.

Honnert, R., V. Masson, and F. Couvreux, 2011: A diagnostic for evaluating the representation of turbulence in atmospheric models at the kilometric scale. *J. Atmos. Sci*., 68, doi: 10.1175/JAS-D-11-061.1

Jeon, J. H., S. Y., Hong, H. Y. Chun, and I. S. Song, 2010: Test of a convectively forced gravity wave drag parameterization in a general circulation model. *Asia-Pac. J. Atmos. Sci*., 46, 1-10.

Johnson, R. H., 1976: The Role of Convective-Scale Precipitation Downdrafts in Cumulus and Synoptic-Scale Interactions. *J. Atmos. Sci*., 33, 1890–1910.

Kain, J. S., 2004: The Kain-Fritsch convective parameterization : An update. J. Appl. Meteor, 43, 170-181.

Kang, H. S., and S.-Y. Hong, 2008: Sensitivity of the simulated East Asian summer monsoon climatology to four convective parameterization schemes. *J. Geophys. Res.: Atmos*. (1984-2012), 113(D15)

Kantha, L. H., 2003: On an Improved Model for the Turbulent PBL. *J. Atmos. Sci*., 60, 2239–2246.

Khairoutdinov, M., and Y. Kogan, 2000: A new cloud physics parameterization in a large-eddy simulation model of marine stratocumulus. *Mon. Wea. Rev*., 128, 229-243.

Kim, Y.-J., and A. Arakawa, 1995: Improvement of orographic gravity-wave parameterization using a mesoscale gravity-wave model. *J. Atmos. Sci*., 52, 1875-1902.

Kim, Y.-J., and S.-Y. Hong, 2009: Interaction between the orography-induced gravity wave drag and boundary layer processes in a global atmospheric model. *Geophys. Res. Lett*., 36, L12809, doi:10.1029/2008GL037146.

Krishnamurti, T. N., L.-N. Simon, and R. J. Pasch, 1983: Cumulus parameterization and rainfall rates I. *Mon. Wea. Rev.,* 111, 815-4828.

Krishnamurti, T. N., Y. Ramanathan, H. L. Pan, R. J. Pasch, and J. Molinari, 1980: Cumulus parameterization and rainfall rates I. *Mon. Wea. Rev*., 108, 465-472.

Kuo, H.-L., 1965: On the formation and intensification of tropical cyclones through latent heat release by cumulus convection. *J. Atmos. Sci*., 22, 40-63.

Kwon, Y.-C., and S.-Y. Hong, 2017 : A mass-flux cumulus parameterization scheme across gray-zone resolutions. *Mon. Wea. Rev*., 145, 583-598.

Lim, K.-S. S., and S.-Y. Hong, 2010: Development of an effective double-moment cloud microphysics scheme with prognostic cloud condensation nuclei (CCN) for weather and climate models. *Mon. Wea. Rev*., 138, 1587-1612.

Lock, A. P., A. R. Brown, M. R. Bush, G. M. Martin, and R. N. B. Smith, 2000: A new boundary layer mixing scheme. Part I: Scheme description and single-column model tests. *Mon. Wea. Rev*., 128, 3187-3199.

Louis, J. F., 1979: A parametric model of vertical eddy fluxes in the atmosphere. *Bound.-Layer Meteor*., 17, 187-202.

Marshall, J. S., and W. MaK. Palmer, 1948: The distribution of raindrops with size. *J. Meteor.,* 5, 165-166.

Mellor, G., and T. Yamada, 1982: Development of a turbulence closure model for geophysical fluid problems. *Rev. Geophys. Space Phys*., 20, 851-875.

Mirocha, J. D., J. K. Lundquist, F. K. Chow, and K. A. Lundquist, 2008: Demonstration of an improved subgrid stress closure for WRF. The 8th WRF user's workshop, Boulder, CO, NCAR, 5-4.

Monin, A. S., and A. M. Obukhov, 1954: Basic laws of turbulent mixing in the surface layer of the atmosphere. Trudy Geofiz. Inst. Akad. Nauk SSSR, 24, 163-187.

Moorthi, S., and M. J. Suarez, 1992: Relaxed Arakawa-Schubert : A parameterization of moist convection for general circulation models. *Mon. Wea. Rev*., 120, 978-1002.

Park, S., and C. S. Bretherton, 2009: The University of Washington shallow convection and moist turbulence schemes and their impact on climate simulations with the Community Atmosphere Model. *J. Climate*, 22, 3449-3469.

Richter, J. H., F. Sassi, and R. R. Garcia, 2010: Towards a physically based gravity wave source parameterization in a general circulation model. *J. Atmos. Sci*., 67, 136–156.
Shin, H. H., and S.-Y. Hong, 2015: Representation of the Subgrid-Scale Turbulent Transport in Convective Boundary Layers at Gray-Zone Resolutions. *Mon. Wea. Rev*., DOI: 10.1175/MWR-D-14-00116.1

Siebesma, A. P., P. M. M. Soares, and J. Teixeira, 2007: A Combined Eddy-Diffusivity Mass-Flux Approach for the Convective Boundary Layer. J. Atmos. Sci., 64, 1230-1248.
Slingo, J. M., 1987: The development and verification of a cloud prediction scheme for the ECMWF model. *Quart. J. Roy. Meteor. Soc*., 113, 899-927.
Soares, P. M., P. M. Miranda, J. Teixeira, and A. P. Siebesma 2008: An Eddy-diffusivity/Mass-flux Boundary Layer Parameterization Based on the TKE Equation: a Dry Convection Case Study. Física de la Tierra, 19, 147-161.
Song, I.-S., and H.-Y. Chun, 2005: Momentum flux spectrum of convectively forced internal gravity waves and its application to gravity wave drag parameterization. Part I: *Theory. J. Atmos. Sci*., 62, 107–124.
Song, I.-S., and H.-Y. Chun, 2008: A Lagrangian spectral parameterization of gravity wave drag induced by cumulus convection. *J. Atmos. Sci*., 65, 1204–1224.
Stensrud, D. J., 2007: Parameterization schemes: keys to understanding numerical weather prediction models. Cambridge University Press.
Stull, R. B., 1988: An introduction to boundary layer meteorology. Kluwer Academic Publishers, The Netherlands, 666pp.
Tiedtke, M, 1989: A comprehensive mass flux scheme for cumulus parameterization in large-scale models. *Mon. Wea. Rev*., 117, 1779-1800.
Tiedtke, M., 1983: The sensitivity of the time-mean large-scale flow to cumulus convection in the ECMWF model. Proceedings of the ECMWF Workshop on Convection in Large-Scale Models, Reading, United Kingdom, ECMWF, 297-316.
Troen, I. B., and L. Mahrt, 1986: A simple model of the atmospheric boundary layer; sensitivity to surface evaporation. *Bound.-Layer Meteor*, 37, 129-148.
Yamada, T., and G. Mellor, 1975: A simulation of the Wangara atmospheric boundary layer data. *J. Atmos.*

Thanks for your attention !

Modeling is to understand what is happening in nature !

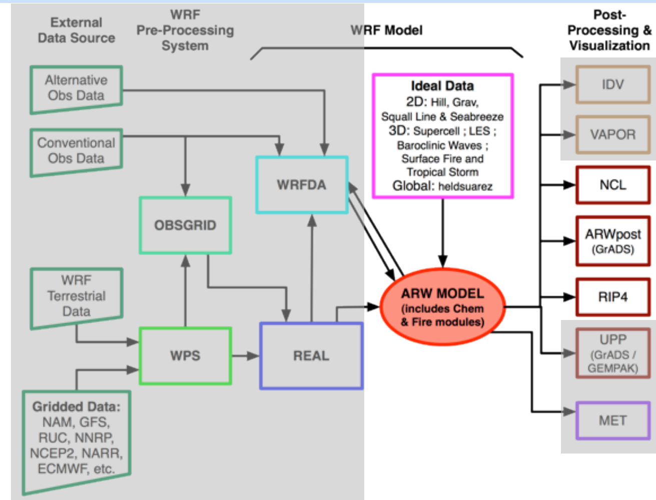
Initialization for Idealized Cases

Bill Skamarock

Initialization for Idealized Cases

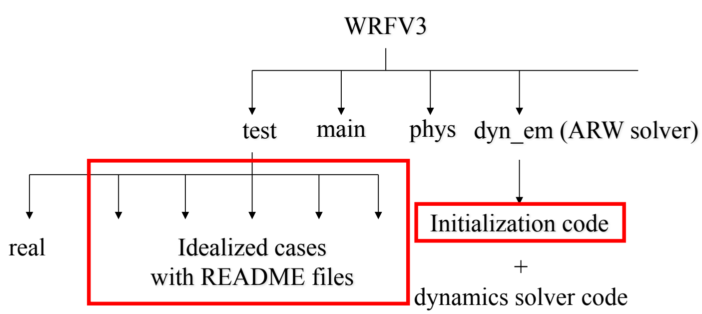
Why do we provide idealized cases?

1. The cases provide simple tests of the dynamics solver for a broad range of space and time scale:
   LES - Δx meters, Δt < second;
   Baroclinic waves - Δx 100 km, Δt = 10 minutes.
2. The test cases reproduce known solutions (analytic, converged, or otherwise).
3. The cases provide a starting point for other idealized experiments.
4. They can be used to test physics development.
5. These tests are the easiest way to test the solver.



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

WRF ARW code



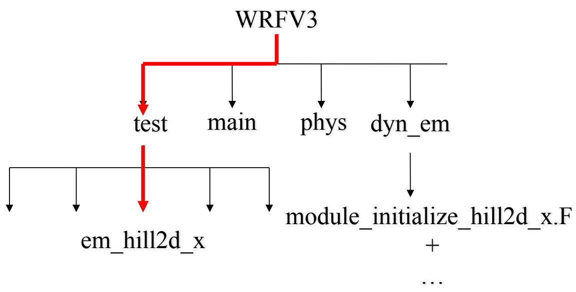
Idealized Test Cases for the WRF ARW Model V3.7

- 2D flow over a bell-shaped mountain – *WRFV3/test/em_hill2d_x*
- 2D squall line (x, z ; y, z) – *WRFV3/test/em_squall2d_x, em_squall2d_y*
- 2D gravity current – *WRFV3/test/em_grav2d_x*
- 2D sea-breeze case – *WRFV3/test/em_seabreeze2d_x*
- 3D large-eddy simulation case – *WRFV3/test/em_les*
- 3D quarter-circle shear supercell thunderstorm – *WRFV3/test/em_quarter_ss*
- 3D tropical cyclone – *WRFV3/test/em_tropical_cyclone*
- 3D baroclinic wave in a channel – *WRFV3/test/em_b_wave*
- 3D global: Held-Suarez case – *WRFV3/test/em_heldsuarez*
- 1D single column test configuration – *WRFV3/test/em_scm_xy*
- 3D fire model test cases – *WRFV3/test/em_fire*
- 3D convective radiative equilibrium test – *WRFV3/test/em_convrad*

Running a test case: *em_hill2d_x* example

2D Flow Over a Bell-Shaped Mountain

Initialization module: dyn_em/module_initialize_hill2d_x.F
Case directory: test/em_hill2d_x



From the WRFV3 main directory:

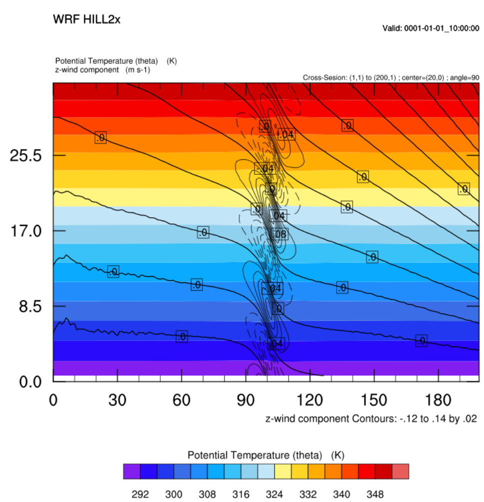
> configure (choose the *no nesting* option)
> compile em_hill2d_x

Move to the test directory:

> cd test/em_hill2d_x
> ideal.exe (this produces the ARW initial conditions)
> wrf.exe (executes ARW)

Finish by plotting output using scripts downloaded from the ARW website (wrf_Hill2d.ncl)

(dx = 2km, dt=20s, T=10 h, wrf_Hill2d.ncl)



What happens during the initialization

Initialization code: *WRFV3/dyn_em/module_initialize_hill2d_x.F*

- Model levels are set within the initialization: code in initialization exist to produce a stretched η coordinate (close to equally spaced z), or equally spaced η coordinate.
- Terrain is set in the initialization code
- A single sounding (z, θ, Q_v, u and v) is read in from *WRFV3/test/em_hill2d_x/input_sounding*
- Sounding is interpolated to the ARW grid, equation of state and hydrostatic balance used to compute the full thermodynamics state.
- Wind fields are interplolated to model η levels.

3D meshes are always used, even in 2D (x,z; y,z) cases. The third dimension contains only 5 planes, the boundary conditions in that dimension are periodic, and the solutions on the planes are identical in the initial state and remain so during the integration.

Idealized Cases: 2d flow over a bell-shaped mountain

Setting the terrain heights

In *WRFV3/dyn_em/module_initialize_hill2d_x.F*

```
SUBROUTINE init_domain_rk ( grid, &

...
  hm = 100.          <---- mountain height and half-width
  xa = 5.0

  icm = ide/2        <---- mountain position in domain
                           (center gridpoint in x)
...

DO j=jts,jte
DO i=its,ite   ! flat surface
  grid%ht(i,j) = hm/(1.+(float(i-icm)/xa)**2)     <---- Set height field
  grid%phb(i,1,j) = g*grid%ht(i,j)
  grid%php(i,1,j) = 0.   <---- lower boundary condition
  grid%ph0(i,1,j) = grid%phb(i,1,j)
ENDDO
ENDDO
```


Idealized Cases: 2d flow over a bell-shaped mountain

Sounding File Format

File: *WRFV3/test/em_quarter_ss/input_sounding*

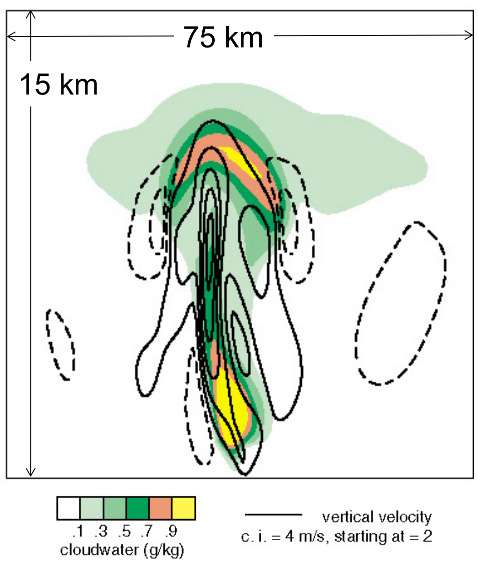
line 1 → surface Pressure (mb), surface potential Temperature (K), surface vapor mixing ratio (g/kg)

each successive line is a point in the sounding

height (m)	potential temperature (K)	vapor mixing ratio (g/kg)	U (west-east) velocity (m/s)	V (south-north) velocity (m/s)
1000.00	300.00	14.00		
250.00	300.45	14.00	-7.88	-3.58
750.00	301.25	14.00	-6.94	-0.89
1250.00	302.47	13.50	-5.17	1.33
1750.00	303.93	11.10	-2.76	2.84
2250.00	305.31	9.06	0.01	3.47
2750.00	306.81	7.36	2.87	3.49
3250.00	308.46	5.95	5.73	3.49
3750.00	310.03	4.78	8.58	3.49
4250.00	311.74	3.82	11.44	3.49
4750.00	313.48	3.01	14.30	3.49

Idealized Cases: 2d squall line

Squall-line simulation
T = 3600 s
$\Delta x = \Delta z = 250$ meters
$\upsilon = 300\ m^2/s$



Idealized Cases: 2d squall line

squall2d_x is (x,z), *squall2d_y* is (y,z); both produce the same solution.

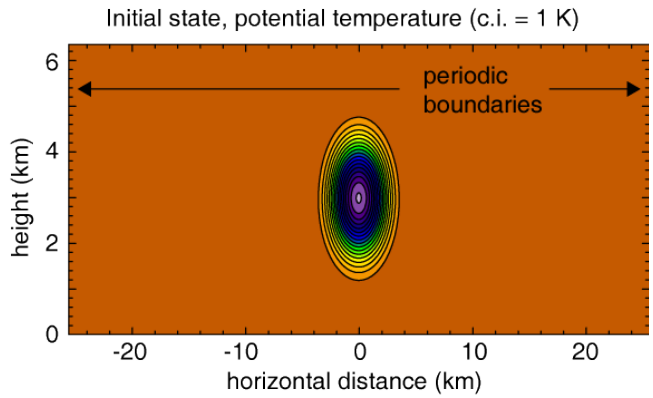
Initialization codes are in
WRFV3/dyn_em/module_initialize_squall2d_x.F
WRFV3/dyn_em/module_initialize_squall2d_y.F
This code also introduces the initial perturbation.

The thermodynamic soundings and hodographs are in the ascii input files
WRFV3/test/em_squall2d_x/input_sounding
WRFV3/test/em_squall2d_y/input_sounding

Idealized Cases: 2d gravity (density) current

(Straka et al, IJNMF, 1993)

2D channel (x , z ; 51.2 x 6.4 km)
Initial state: theta = 300 K (neutral) + perturbation (max = 16.2 K)
Eddy viscosity = 75 m**2/s (constant)

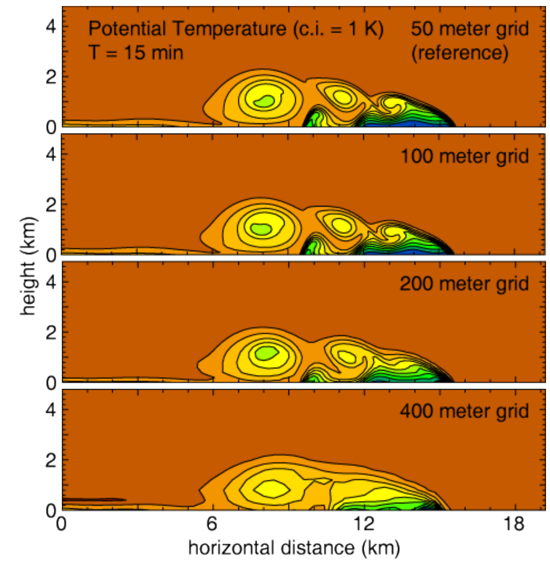


Idealized Cases: 2d gravity (density) current

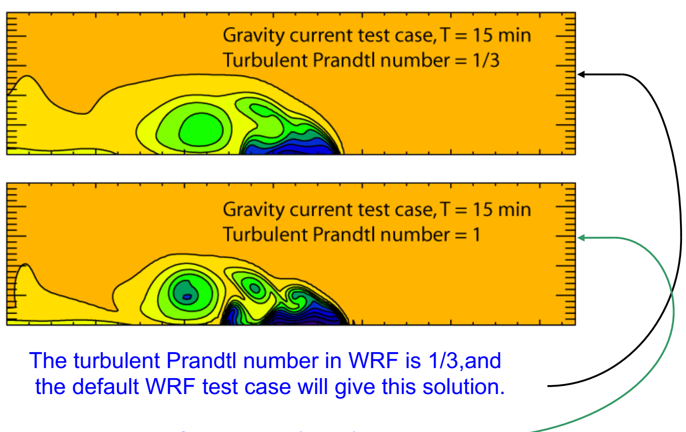
Default case, dx = 100 m,
5th order upwind advection,
uses namelist.input.100m

dx = 200 m,
5th order upwind advection,
use namelist.input.200m

dx = 400 m,
5th order upwind advection,
use namelist.input.400m



Idealized Cases: 2d gravity (density) current

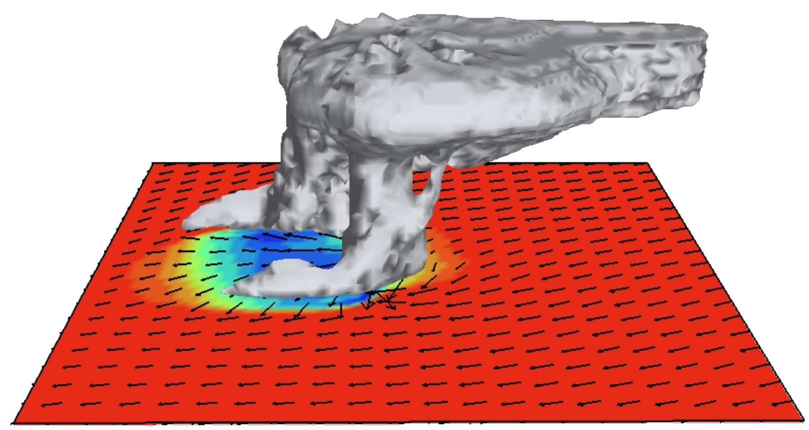


The turbulent Prandtl number in WRF is 1/3,and the default WRF test case will give this solution.

To recover the Straka et al (1993) solution, change the parameter *Prandtl* to 1 (from 1/3) in *WRFV3/share/module_model_constants.F*

Idealized Cases: 3d supercell thunderstorm

Height coordinate model
(dx = dy = 2 km, dz = 500 m, dt = 12 s, 160 x 160 x 20 km domain)
Surface temperature, surface winds and cloud field at 2 hours



Idealized Cases: 3d Large Eddy Simulation (LES)

Initialization code is in
WRFV3/dyn_em/module_initialize_les.F

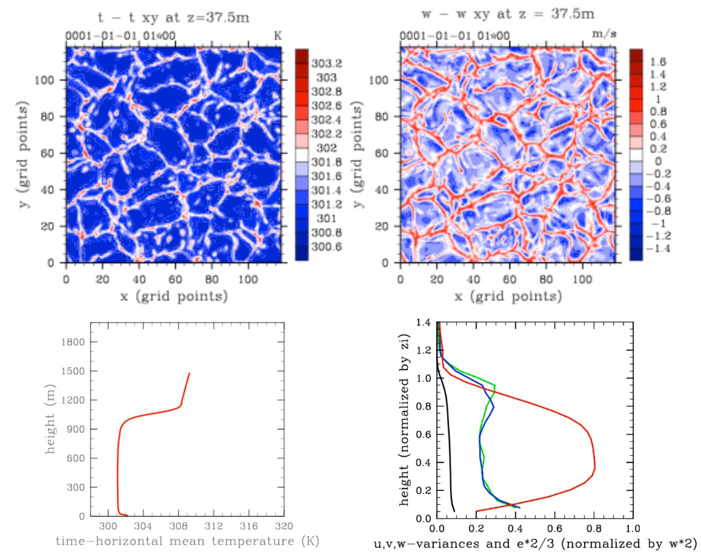
Test case directory is in
WRFV3/test/em_les

The default case is a large-eddy simulation of free convective boundary layer with no winds. The turbulence of the free CBL is driven and maintained by namelist-specified surface heat flux.

An initial sounding with mean winds is also provided.

Reference: Moeng et al. 2007 MWR

Idealized Cases: 3d Large Eddy Simulation (LES)



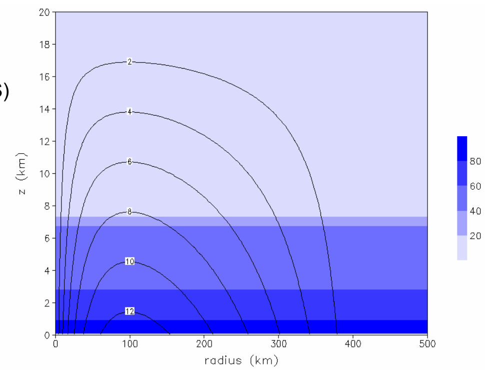
Idealized Cases: 3d tropical cyclone

Default vortex:

- weak (12.9 m/s) axisymmetric analytic vortex (Rotunno and Emanuel, 1987, JAS)
- placed in center of domain
- in "module_initialize_tropical_cyclone.F" users can modify initial size and intensity (see parameters r0, rmax, vmax, zdd)

Default environment:

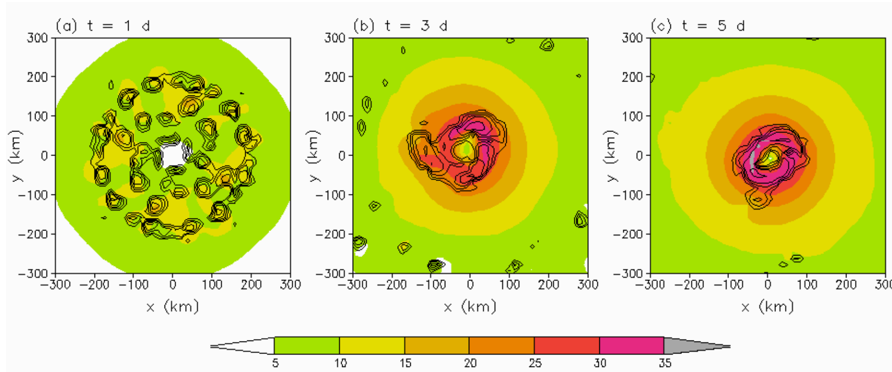
- mean hurricane sounding from Jordan (1958, J. Meteor.)
- SST = 28 degrees C
- f = 5e-5 s^{-1} (20 degrees North)



colors = relative humidity (%)
contours = azimuthal velocity (m/s)

Default domain:
- 3000 km x 3000 km x 25 km domain
- default dx,dy is only 15 km: useful for quick tests of new code (i.e., new physics schemes); research-quality studies should use smaller dx,dy

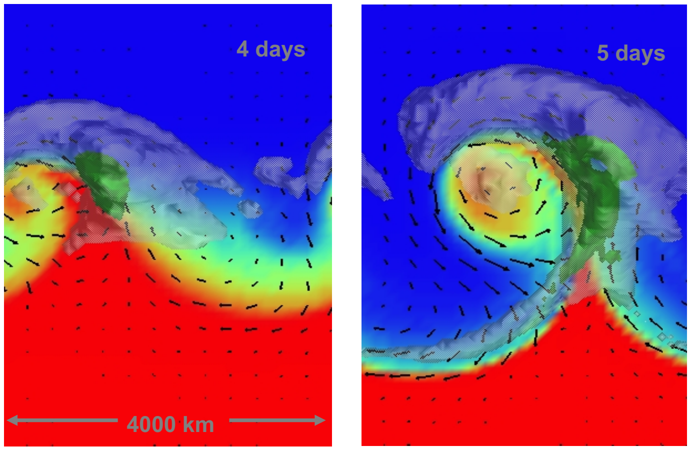
Idealized Cases: 3d tropical cyclone



colors = 10-m windspeed (m/s)
contours = reflectivity (every 10 dBZ)

Idealized Cases: baroclinic wave in a channel

Height coordinate model (dx = 100 km, dz = 250 m, dt = 600 s)
Surface temperature, surface winds, cloud and rain water



Idealized Cases: baroclinic wave in a channel

Initialization code is in
WRFV3/dyn_em/module_initialize_b_wave.F

The initial jet (y,z) is read from the binary input file
WRFV3/test/em_b_wave/input_jet

The initial perturbation is hardwired in the initialization code.

Idealized Cases: baroclinic wave in a channel

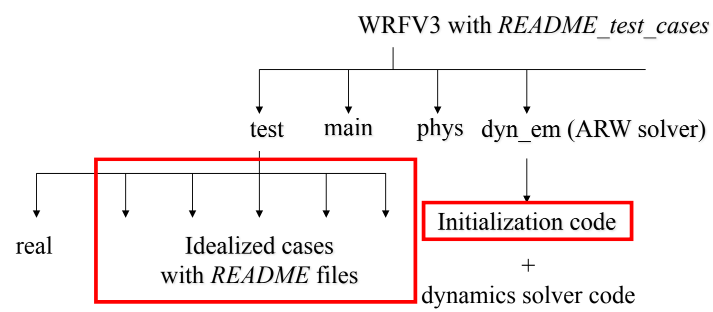
Default configuration in
WRFV3/test/em_b_wave/namelist.input
runs the dry jet in a periodic channel with dimension (4000 x 8000 x 16 km) (x,y,z).

Turning on any microphysics
(mp_physics > 0 in namelist.input) puts moisture into the model state.

The initial jet only works for dy = 100 km and 81 grid points in the y (south-north) direction.

Idealized Cases: More information

Descriptions:
WRFV3/README_test_cases
WRFV3/test/em_*/README



Idealized Test Cases for the WRF ARW Model V3.9

- 2D flow over a bell-shaped mountain – *WRFV3/test/em_hill2d_x*
- 2D squall line (x, z ; y, z) – *WRFV3/test/em_squall2d_x, em_squall2d_y*
- 2D gravity current – *WRFV3/test/em_grav2d_x*
- 2D sea-breeze case – *WRFV3/test/em_seabreeze2d_x*
- 3D large-eddy simulation case – *WRFV3/test/em_les*
- 3D quarter-circle shear supercell thunderstorm – *WRFV3/test/em_quarter_ss*
- 3D tropical cyclone – *WRFV3/test/em_tropical_cyclone*
- 3D baroclinic wave in a channel – *WRFV3/test/em_b_wave*
- 3D global: Held-Suarez case – *WRFV3/test/em_heldsuarez*
- 1D single column test configuration – *WRFV3/test/em_scm_xy*
- 3D fire model test cases – *WRFV3/test/em_fire*
- 3D convective radiative equilibrium test – *WRFV3/test/em_convrad*

Advanced Usage of the WPS

Michael Duda

Advanced Usage 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 field to be produced by geogrid
  - 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   = 2
       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 <u>in the existing entry for that field</u>.

```
===============================
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

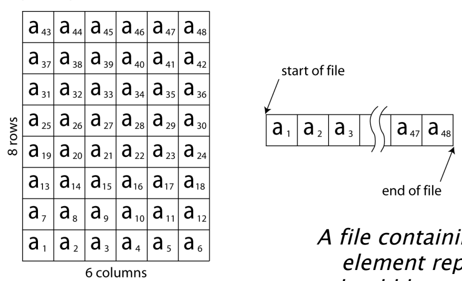
To add a new data source, we need to

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

From python, one can use

```
numpy.fromfile(file, dtype=dt)
```

to read the geogrid binary files, and

```
numpy.ndarray.tofile(file)
```

to write the geogrid binary files.

The `dtype` argument and `numpy.ndarray.astype` may be used to match the *wordsize* and *endianness* used in the binary file!

- Values are always represented as integers

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

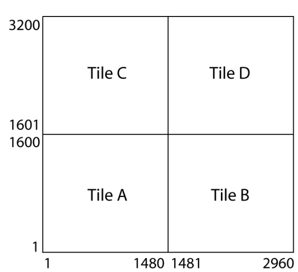
E.g., For a binary file containing an array with 500 columns and 750 rows, the file name would be

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

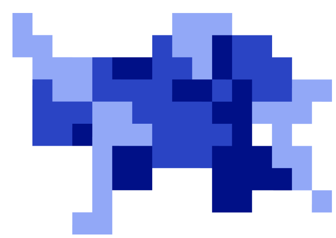


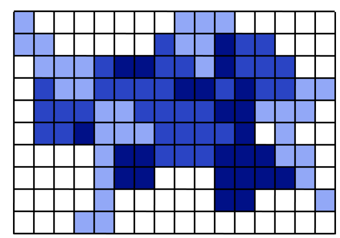
Tile A named	00001-01480.00001-01600
Tile B named	01481-02960.00001-01600
Tile C named	00001-01480.01601-03200
Tile D named	01481-02960.01601-03200

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

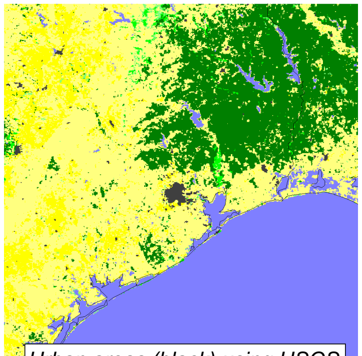
- The particular missing value used in the data set is specified in the `index` metadata file for the data set



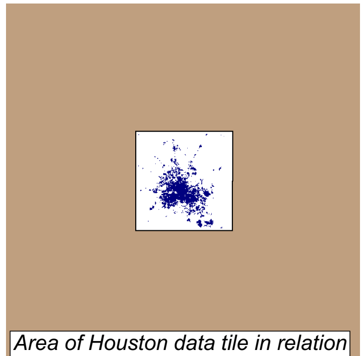


Example: Houston LU Data Set

- 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/
===============================
```

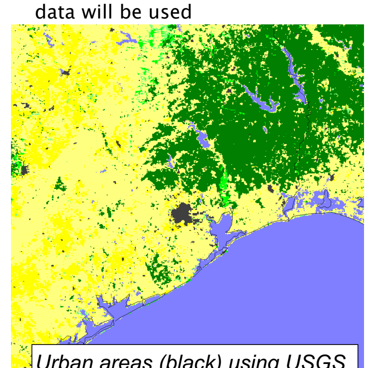
Give this data source priority over default data sources

How to interpolate this data source, and where to find it on disk

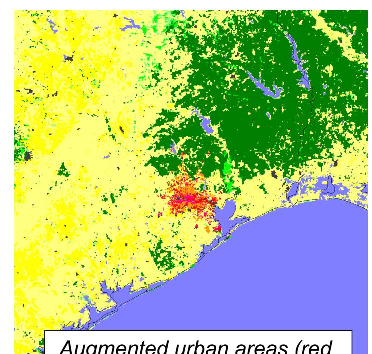
Example: Houston LU Data Set

4) Run geogrid.exe

Any gridpoints covered by Houston data will use it; otherwise default USGS data will be used



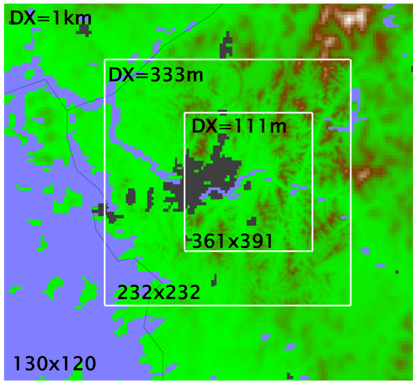
Urban areas (black) using USGS 24-category data set



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

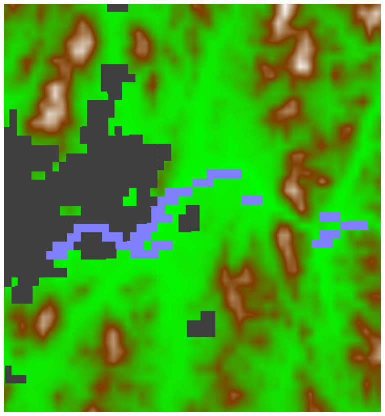
To use the SRTM topography data, we

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

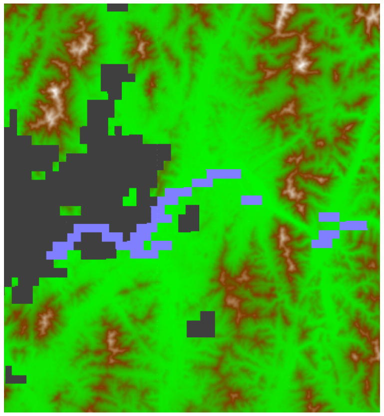
```
===============================
name = HGT_M
     priority = 1
     dest_type = continuous
     interp_option   =   30s:special(4.0)+four_pt
     interp_option   =   SRTM:four_pt
     rel_path    =   30s:topo_30s/
     rel_path    =   SRTM:SRTM/
===============================
```

4) Specify that we should interpolate from SRTM in namelist by setting
   `geog_data_res = '30s','SRTM+30s','SRTM+30s'`

Example: Seoul



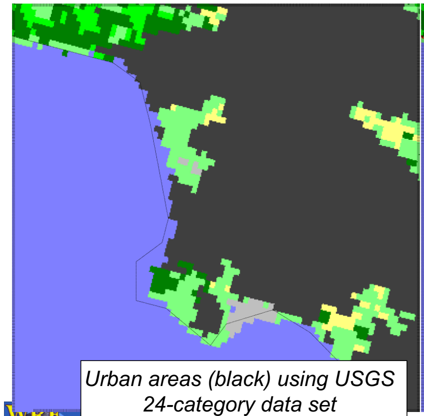
Domain 3 (DX=111m) using default 30" USGS topography



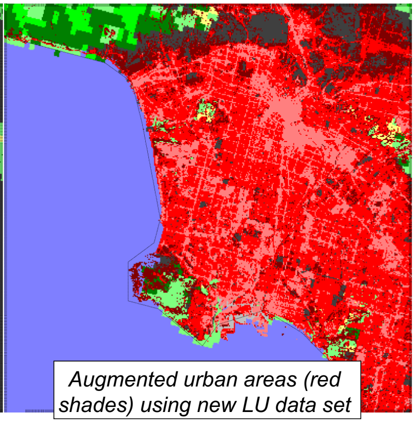
Domain 3 (DX=111m) using 3" SRTM topography

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 (<u>nearest neighbor</u>) 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

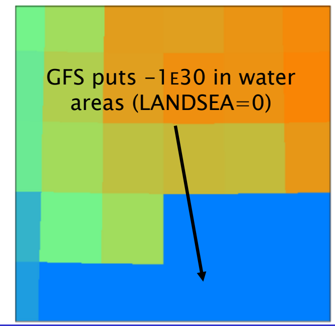
- Suppose we have a 1000x1000 domain over Houston (dx=500 m)
  - This is the same domain as in the urban land use example
- Meteorological data come from 1-degree GFS
  - *Note that we will be interpolating 1-degree data onto a 500-m grid!*
- We want to create an entry for a new soil moisture field, 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 (masked = water)

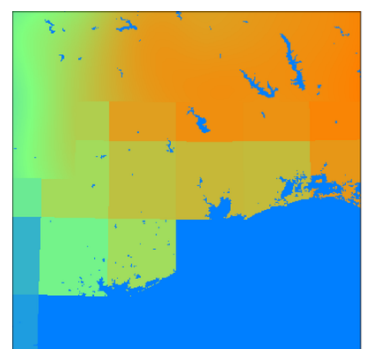
Specify that metgrid should not use points in source where LANDSEA field equals 0 (interp_mask = LANDSEA(0))

Fill model points that don't receive an interpolated value (like water) to 0 (fill_missing = 0.)

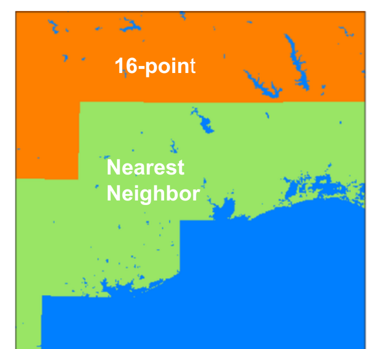
For a complete list of possible keywords *See p. 3-52*

Example: A new METGRID.TBL entry

- Now, after running metgrid.exe again, the SM000010 field looks like



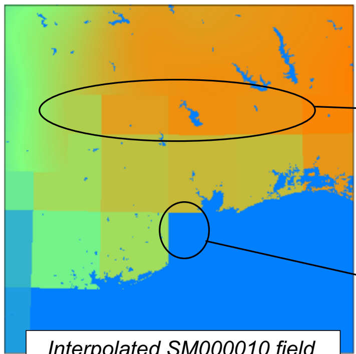
Interpolated SM000010 field (sixteen_pt + nearest_neighbor)



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



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)

Interpolated SM000010 field (sixteen_pt + nearest_neighbor)

Example: A new METGRID.TBL entry

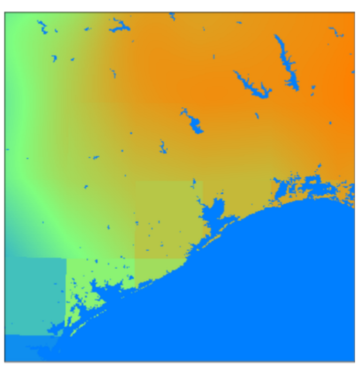
- Update the METGRID.TBL entry for SM000010

```
===========================
name = SM000010
masked = water
interp_mask = LANDSEA(0)
interp_option = sixteen_pt + four_pt + average_4pt
fill_missing = 0.
===========================
```

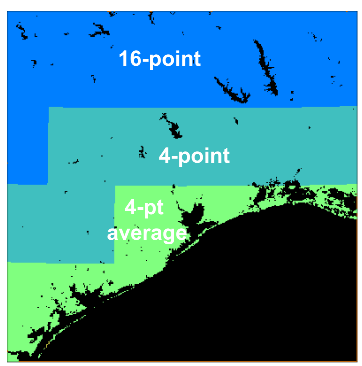
- If 16-pt doesn't work, then try 4-pt before reverting to a 4-point average
  - Note that 4-point average will work anywhere nearest_neighbor would (missing/masked values not counted in the average)

Example: A new METGRID.TBL entry

- The resulting field, below-left:



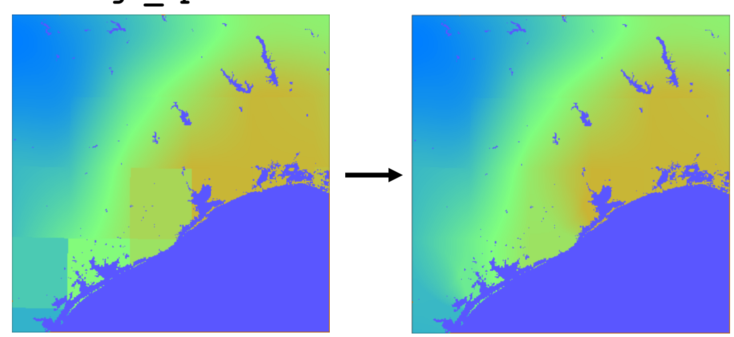
Interpolated SM000010 field (sixteen_pt + four_pt + average_4pt)



Which interpolator was used at each model grid point

Example: A new METGRID.TBL entry

- By using **wt_average_4pt** instead of **average_4pt**:



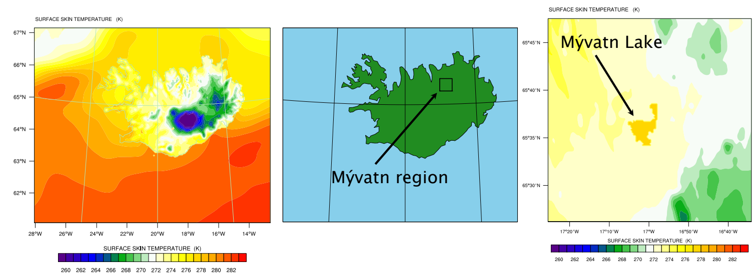
*sixteen_pt + four_pt + **average_4pt*** *sixteen_pt + four_pt + **wt_average_4pt***

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.



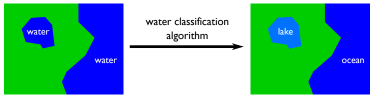
Above left: Skin temperature field (TSK) for Iceland and surrounding ocean on 26 January 2011 1200 UTC from NCEP GFS and RTG SST data.

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



1) 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.
2) 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

This is the default as of WPS v3.9

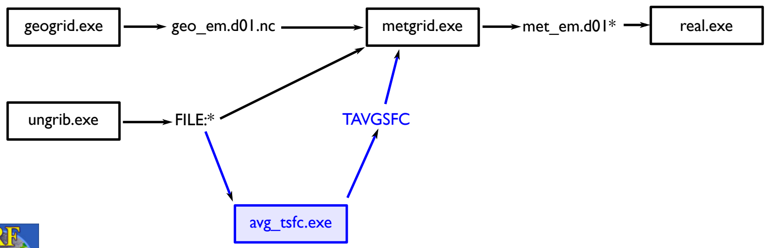
In *namelist.wps*, set:

- `geog_data_res = "usgs_lakes+default"` for USGS land use (16=ocean, 28=lake)
- `geog_data_res = "modis_30s_lake+default"` for MODIS land use (17=ocean, 21=lake)

Creating a Proxy SST Field

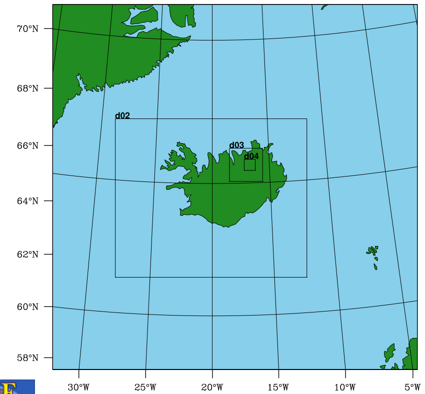
The *avg_tsfc.exe* utility program may be used to compute the average 2-m air temperature field for any number of full diurnal cycles

- 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

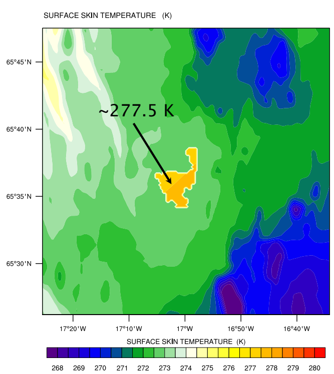
ICs + BCs from NCEP GFS

Sea surface temperatures from RTG SST

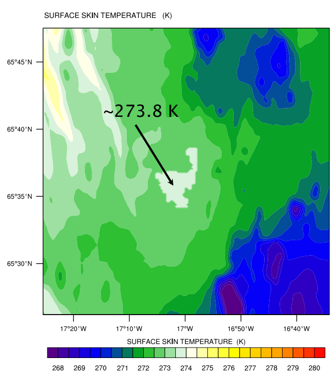
Initial time: 26 January 2011, 1200 UTC

Test case: Lake Mývatn

26 January 2011, 12 UTC



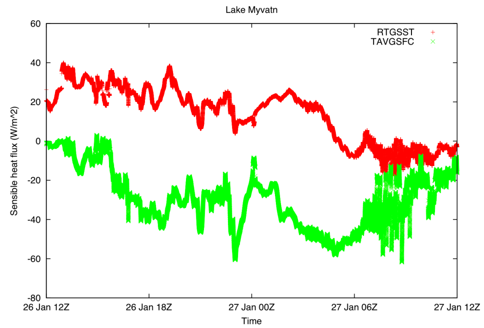
Initial skin temperature field using extrapolated SST values



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 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?

Bonus slides: A second METGRID.TBL example

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

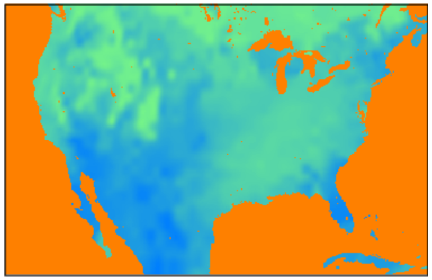
Metgrid runs and warns that no AGRMET data files were found:

```
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:

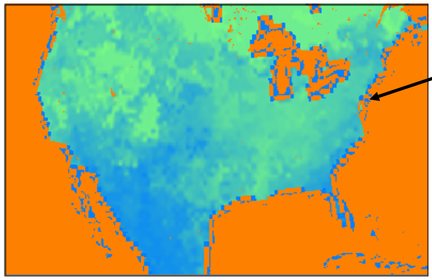


METGRID.TBL: Real-time System Example

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

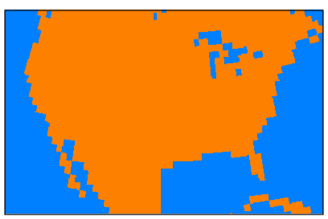
```
========================================
name=SM000010
interp_option=sixteen_pt+four_pt+wt_average_4pt+search
masked=water
interp_mask=LANDSEA(0)
fill_missing=1.
flag_in_output=FLAG_SM000010
========================================
```

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*.

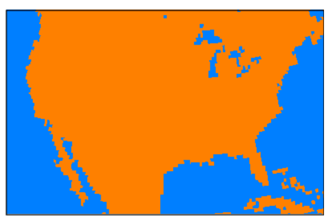
- So, metgrid used the GFS LANDSEA mask even when interpolating AGRMET data!

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

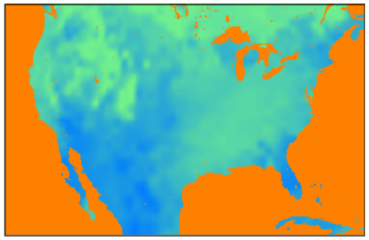
METGRID.TBL: Real-time System Example

```
========================================
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
========================================
```

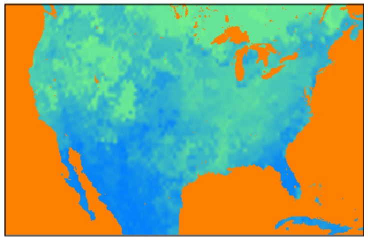
```
========================================
name=SM000010; from_input=AGRMET
interp_option=sixteen_pt+four_pt+wt_average_4pt+search
masked=water
interp_mask=AGR_LAND(-1.E30)
fill_missing=1.
flag_in_output=FLAG_SM000010
========================================
```


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

WRF Four-dimensional Data Assimilation (FDDA)

Jimy Dudhia

WRF Four-Dimensional Data Assimilation (FDDA)

Wei Wang and Jimy Dudhia

NCAS/NCAR Tutorial, Durham, UK

October 2017

FDDA

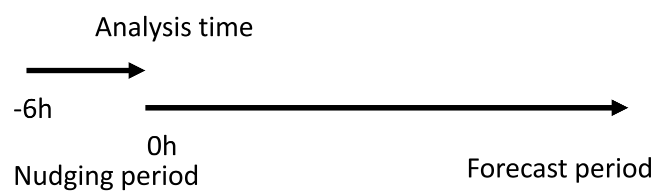
- Method of nudging model towards observations or analysis (gridded data)
- May be used for
  – Dynamic initialization (pre-forecast period)
  – Create 4-dimensional meteorological datasets (e.g. for air-quality models)
  – Boundary conditions (with outer domain nudged towards analysis)

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

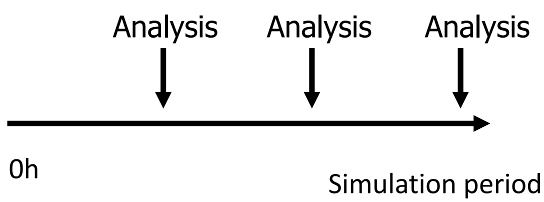
Dynamic Initialization

- 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



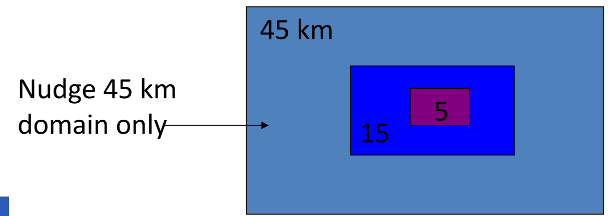
Four-Dimensional Met Analysis

- 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

- Three Methods
  - Grid or analysis nudging (suitable for coarse resolution)
  - Observation or station data nudging (suitable for fine-scale or asynoptic obs)
  - Spectral nudging on selective scales
- Nudging can be applied to winds, temperature, water vapor (first two methods), and geopotential (spectral)

Note: nudging terms are fake sources, so avoid FDDA use in dynamics or budget studies. Also data may be linearly interpolated.

Analysis Nudging (grid_fdda=1)

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

From MM5: Stauffer and Seaman (1990 MWR, 1994 JAM)

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

In WRF p* is μ 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)

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
  - 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 <u>not</u> 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

- 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.
- In Version 3.8 we have FASDAS (grid_sfdda=2)
  - Flux-Adjusted Surface Data Assimilation System
  - This is a special option to also correct the soil state
    - Only works with YSU PBL and Noah LSM

Spectral Nudging (grid_fdda=2)

- Spectral nudging does 3d nudging of only selected larger scales using gridded fields
  - Allows model small scales to evolve with no nudging
- This may be useful for controlling longer wave phases for long analysis-driven simulations (e.g. months to years)
  - 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, temperature, 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.)

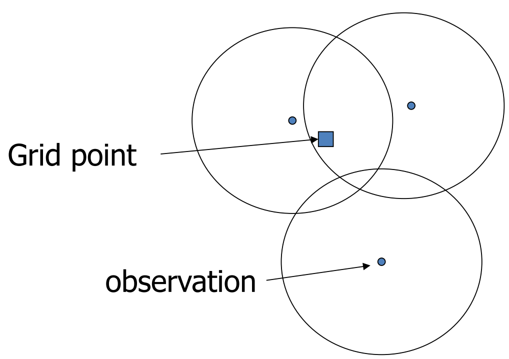
Obs Nudging (obs_nudge_opt=1)

- 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

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

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

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

Obs Nudging

- w_s 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
- Spectral nudging may be used to control large scale flows
- 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

How to Use the WRF Registry

Dave Gill

How to Use the WRF Registry

John Michalakes, NRL

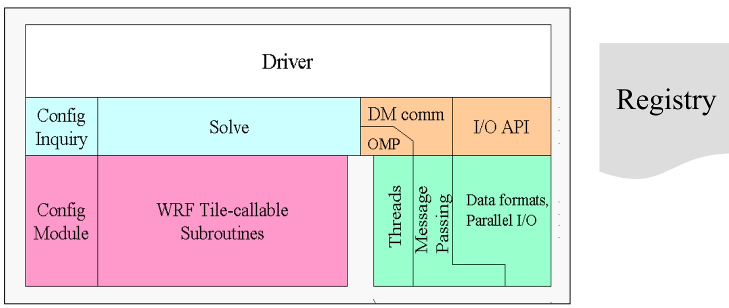
Dave Gill, NCAR

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



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    -    i01rhud    "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    -    i01rhud    "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     -    i01rhud   "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	-	i01rhud	"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.
- Double digit streams require "{}" braces: **i01{19}{24}**

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	-	i01rhud	"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

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)

Example 1: What streams can I use?

- Generally history streams 10 – 24 are OK
- Avoid 22, 23
- Need LOTS more streams?
  - Edit WRFV3/arch/preamble_new

  MAX_HISTORY = 25 <- - - *right now*

  - clean –a, configure, compile, re-run real and wrf

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     \
misc          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 when dereferenced by first_rk_step_part1 for the physics drivers

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

- In solve_em, 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)

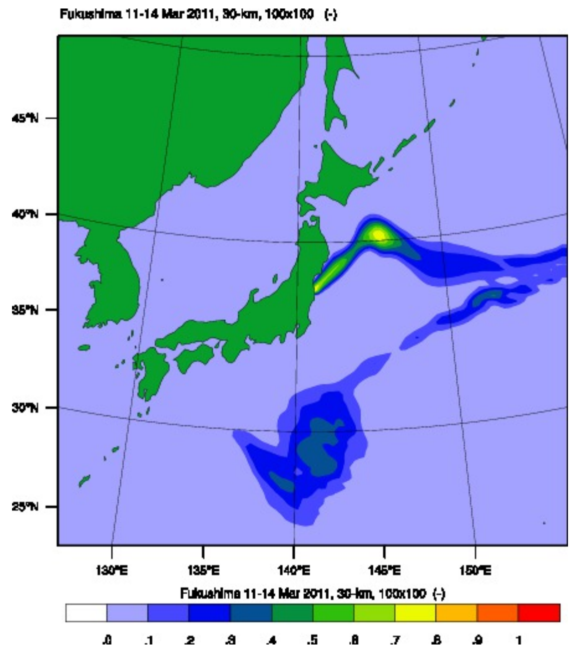
dyn_em/solve_em.F (continuous plume in wrf.exe)

```
!  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

WRF:
More Run-time Options

Wei Wang

WRF:
More Runtime Options

Wei Wang
January 2018

More options

- Some useful *runtime* options:
  - Vertical interpolation options (program real.exe)
  - Options to use hybrid vertical coordinate
  - IO options
  - Base state parameters
  - Options for long simulations
  - Adaptive-time step
  - Digital filter
  - Global runs
  - Moving nest
  - TC options
  - Tracer / trajectory
  - Stochastic kinetic-energy backscatter scheme (SKEB)
  - Optional output
  - 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:

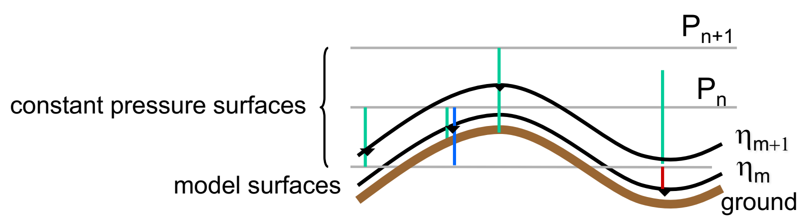
- `&dfi_control`
- `&fdda`
- `&grib2`
- `&scm`
- `&tc`
- `&noah_mp`

Look for these in **`examples.namelist`**

Vertical interpolation options (1)

Program **`real`** only, &domains:

- `interp_type:` in pressure or log pressure
- `lagrange_order:` linear or quadratic
- `use_surface:` whether to use surface level data



Vertical interpolation options (2)

Program **real** only, &domains:

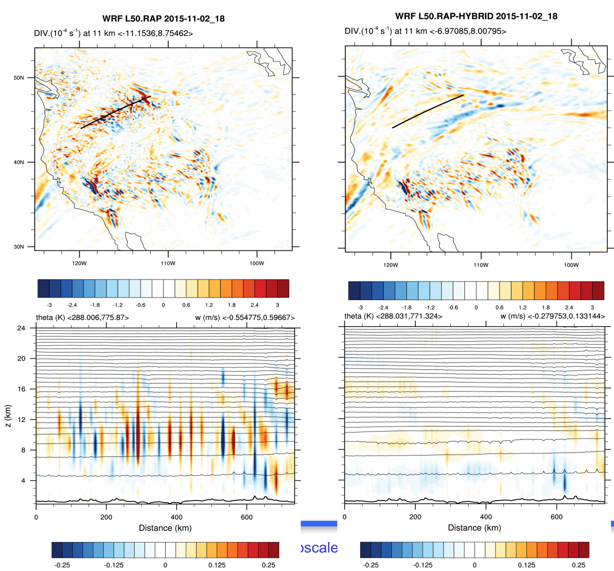
- **use_levels_below_ground**: whether to use data below the ground
- **lowest_lev_from_sfc**: logical, whether surface data is used to fill the lowest model level values
- **force_sfc_in_vinterp**: number of levels to use surface data, default is 1
- **extrap_type**: how to do extrapolation: 1 - use 2 lowest levels; 2 - constant
- **t_extrap_type** : extrapolation option for temperature: 1 - isothermal; 2 - 6.5 K/km; 3 - adiabatic

Look for these in **examples.namelist**

Hybrid Vertical Coordinate Option

- This is a compile-time option in V3.9:
  configure -hyb
- Decision made when running program **real.exe**, by setting these namelists in &dynamics
  hybrid_opt = 2 (0 turns it off)
  eta_c = 0.2 (default)
- New in V3.9

Hybrid Vertical Coordinate Options



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 the list in
Registry/registry.io_boilerplate

IO Control (2)

Specify input and output files explicitly in **&time_control**

```
auxinput1_inname = "/mydata/met_em.d<domain>.<date>"
```

: explicitly specify input file (it name and directory)

```
history_outname = "/mydata/wrfout_d<domain>_<date>"
```

: explicitly specify history output file (its name and directory)

Look for these in **Registry/registry.io_boilerplate**

IO Control (3)

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 (4)

Starting in V3.2, there is an alternative to add/remove output fields 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

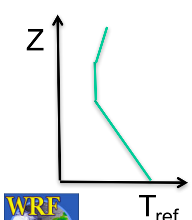
Base State Parameters

The following could be varied (set in program *real*):

base_temp Base state surface temperature

iso_temp Base state stratosphere temperature (default 200 K)

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)

Use of physics suite

Since 3.9, physics can be selected as a suite. These represents well-tested physics. Two are available:

```
physics_suite = 'tropical'
        ↓
mp_physics = 6, 6,
cu_physics = 16, 16,
ra_lw_physics = 4, 4,
ra_sw_physics = 4, 4,
bl_pbl_physics = 1, 1,
sf_sfclay_physics = 91, 91,
sf_surface_physics = 2, 2,
```

```
physics_suite = 'CONUS'
        ↓
mp_physics = 8, 8,
cu_physics = 6, 6,
ra_lw_physics = 4, 4,
ra_sw_physics = 4, 4,
bl_pbl_physics = 2, 2,
sf_sfclay_physics = 2, 2,
sf_surface_physics = 2, 2,
```


Use of physics suite

To turn an option off for a particular domain:

```
physics_suite = 'tropical'
        +
cu_physics = -1, 0,
```

To overwrite one or more with other options:

```
physics_suite = 'CONUS'
        +
cu_physics = 16, 16,
bl_pbl_physics = 1, 1,
sf_sfclay_physics = 1, 1,
```


Options for long simulations (1)

Update control for lower boundary fields: 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)

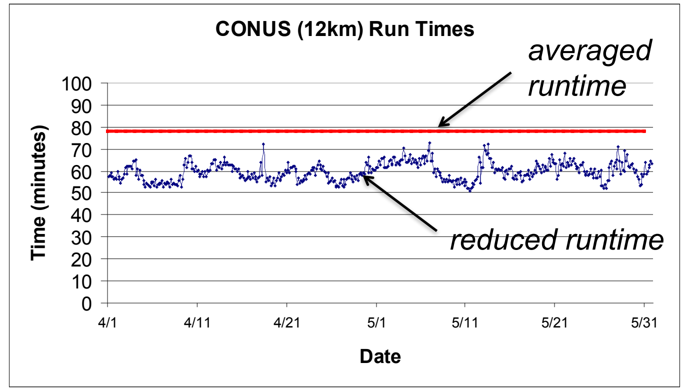
sst_skin	diurnal water temp update
tmn_update	deep soil temp update, used with lagday
lagday	averaging time in days
bucket_mm	bucket reset value for rainfall (e.g. **rainc=i_rainc*bucket_mm+rainc**)
bucket_j	bucket reset value for radiation fluxes
spec_exp	exponential multiplier for boundary zone ramping (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.
- 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

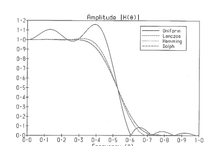
Adaptive time steps (3)

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 (DFI) (1)

- DFI is a way to use a low-pass filter to improve model initial conditions
- Useful for short-range model runs (1-6 hours)
- Imbalances in model IC
  - May be introduced by interpolation, different topography, or by objective analysis, and data assimilation
  - 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)

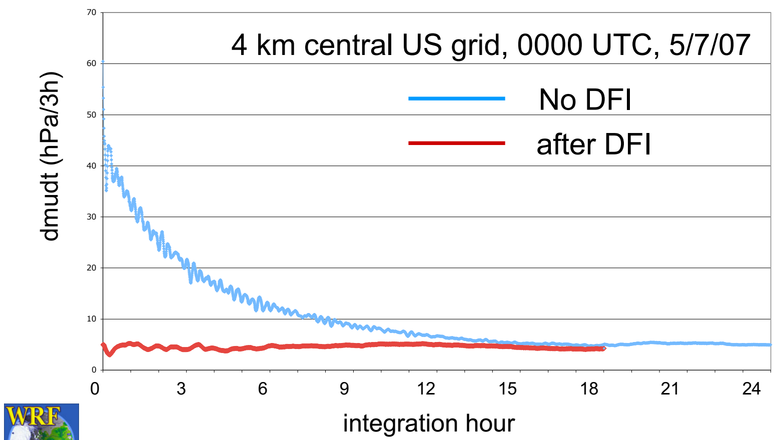
Using DFI
- can construct consistent model fields which do not exist in the initial conditions, e.g. vertical motion, cloud variables
- may reduce the spin-up problem in early simulation hours
- Useful for short-range (1-6 h) forecasts and cycling with data assimilation

DFI is done after program **`real`**, or data-assimilation step

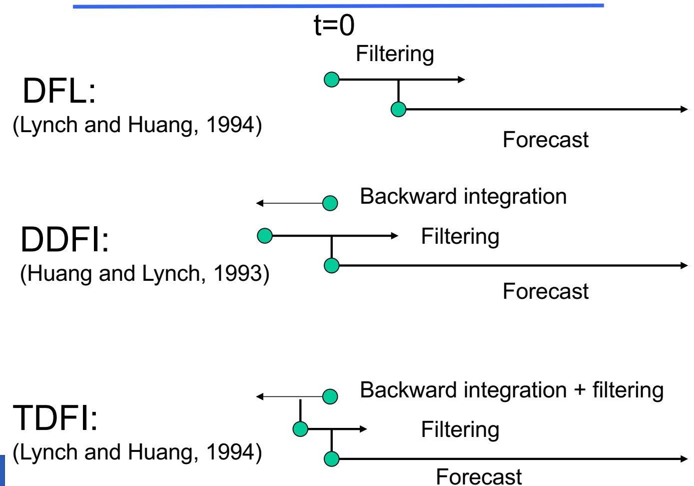
See 'Using Digital Filter Initialization', Chap 5, UG.

Digital filter initialization (3)

Use of DFI helps to damp high pressure tendencies in early forecast



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 in WPS:
  `map_proj = 'lat-lon'`
  `e_we, e_sn:` **geogrid** will compute dx, dy
  See template `'namelist.wps.global'`
- Requires only one-time period data
- 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)

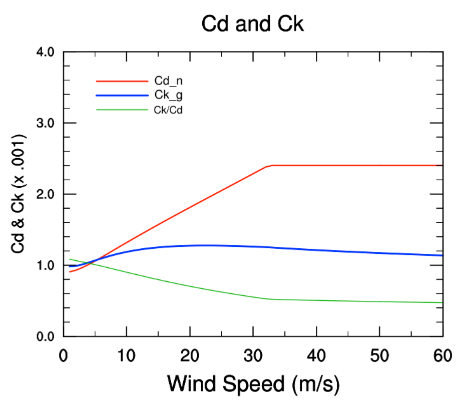
`track_level:` e.g. 700 or 500 mb

`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

`oml_relaxation_time:` time scale to relax ocean temperature back to initial value

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 bathymetry (or ocean depth). Only simple initialization is provided in the model (added 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 = 1000, (default value)

New in V3.9: it can output meteorological variables, as well as chemistry ones, along the trajectories.

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)
- **sppt:** = 1, activate stochastically parameterized pert tendencies
- **spp:** = 1, activate stochastic perturbed parameters in physics

Also see 'Option to stochastically perturb forecasts' section in Chap 5, UG

Also see http://www.cgd.ucar.edu/~berner/skebs.html

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

(Caution: *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)

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.

Additional Output Option (8)

More climate output (from RASM):
mean_diag = 1: (with interval options)
diurnal_diag = 1

Output time-step and diurnal averaging of a number of surface variables and radiative fluxes at surface and top of atmosphere

See run/README.rasm_diag for details, and Registry/registry.rasm_diag for full listing.

Data goes to auxhist5 file.

IO quilting: &namelist_quilt

I/O quilting 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 to turn it on
- 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/**

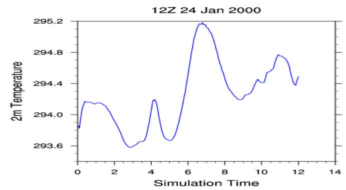
```
#-----------------------------------------------#
# 24 characters for name | pfx |  LAT  |   LON  |
#-----------------------------------------------#
Cape Hallett             hallt -72.330  170.250
McMurdo Station          mcm   -77.851  166.713
```

- 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 5-34 – 5-36: 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:
Best Practices

Ming Chen

Best Practices of WRF

Wei Wang Jimy Dudhia Ming Chen
National Center for Atmospheric Research

Best Practices of WRF

- WRF is well-tested and documented. It can be used by people who have no experiences or formal training.
- However, in spite of advanced parameterization schemes in WRF and high-resolutions permitted by faster computers, correct choice of options is still a prerequisite for successful application of WRF

Best Practices of WRF

- A Thorough Analysis of the Research Topic
  - Conclusions and approaches in previous studies? Questions not answered? Incomplete knowledge? Important processes (convection, radiation, surface forcing, etc.?)
  - extensive literature review
- Your Scientific or Practical Objectives?
  - Scientific questions you want to answer
  - What can you do with WRF? Where and how WRF simulations may be helpful

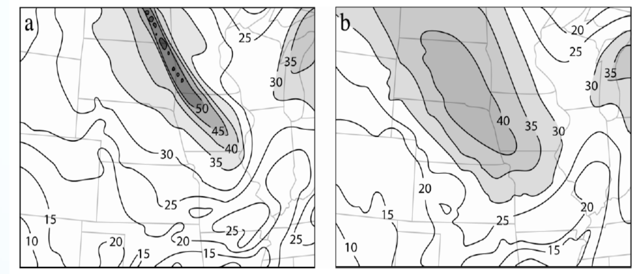
Best Practices of WRF

- The Model Configuration
  - Domain – often have profound influences
  - Resolution (horizontal and vertical)
  - Time and method of initialization
    - Cold start?
    - Variational data assimilation?
    - Spinup time?
  - Lateral Boundary Locations
  - Physics/dynamics options

How to determine the model domain

- 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

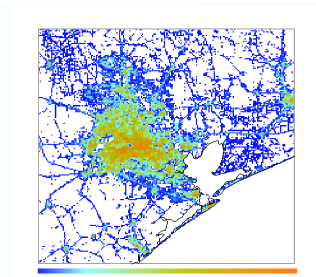
Importance of domain



12-hour simulations of 250-hPa winds (m s-1) from the 40-km grid increment Eta Model initialized at 1200 UTC 3 August 1992, based on experiments that used a large (a) and a small (b) computational domain. (Warner, 2011)

Initialization and Spin-up Issues

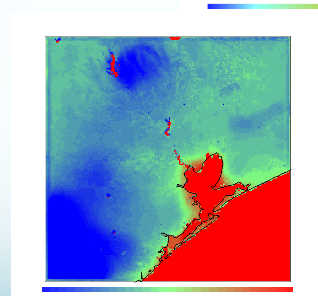
- Model problems often arise from poor initial condition
  - Appropriate initial time
  - Quality of initial condition
    - Check land data:
      e.g. landuse: *does it represent my area well?*
    - Know about the data: *how good are the data?*
      - Forecast data
      - Reanalysis data
      - Climate model data
- In the first few hours, expect noise in pressure fields
  - Mostly sound waves adjusting winds to terrain. No harmful lasting effects

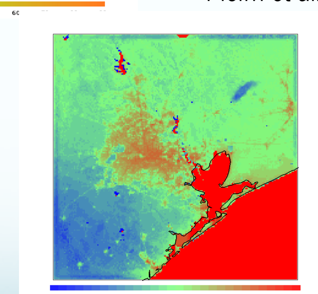


Impervious fraction (%)

Skintemp simulated with and without Impervious (Aug 26, 2006, 10Z)

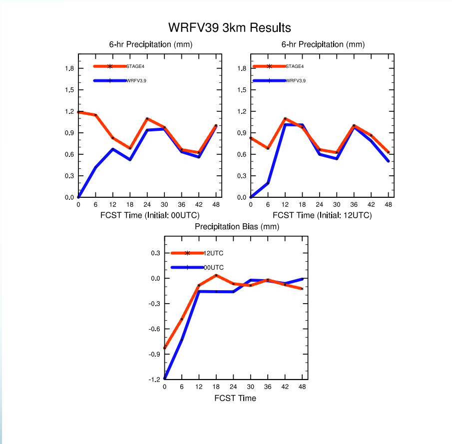
Pleim et al., 2012





Initialization and Spin-Up

Convective Spin-Up: An example of NCAR's 3-km convective runs

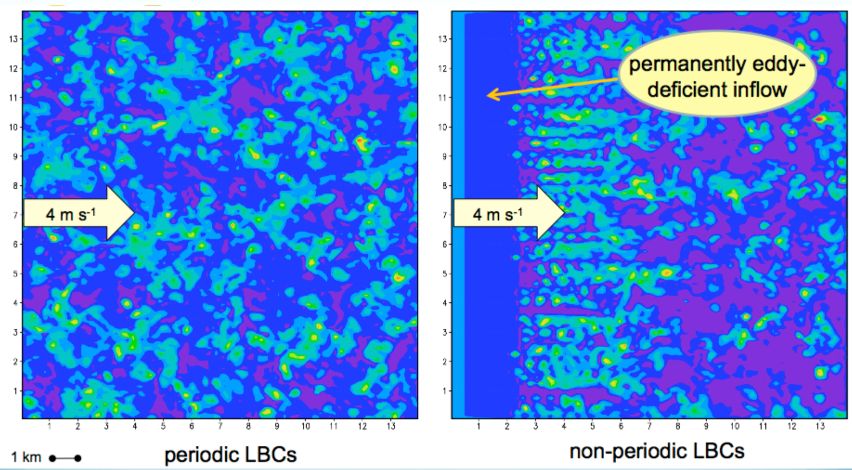


Red: StageIV

Blue: WRFV3.9

Lateral Boundary Condition

- A basic and potentially serious limitation to regional model simulation, including WRF
- Possible negative effects of LBC
- How to minimize the negative LBC impact on forecast quality: guidelines and cautions
  - Strong forcing should be avoided at lateral boundaries
  - Resolution-consistent input data should be used
  - More frequent is better
  - Interactive boundaries should be employed when possible



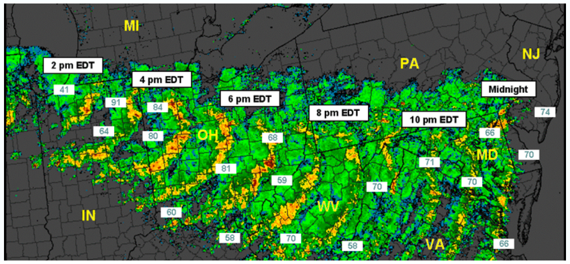
Gaudet et al. (2012)

Grid Size and Impact

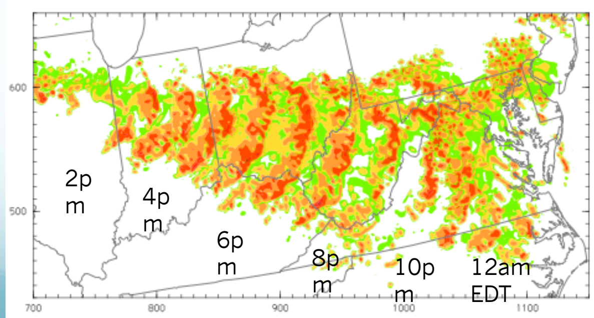
- Extreme weather event forecast
  - The Derecho of 29-30 June 2012
- $\Delta \approx 3$ km: Traditional cloud-permitting resolution
  - No need for deep-convective parameterization
- $\Delta \approx 30$ m: Traditional large-eddy simulation (LES) resolution
  - No need for a planetary boundary layer (PBL) parameterization
  - Turbulent eddies (i.e., thermals, rolls, etc.) are handled by the model's governing equations [plus surface-layer and subgrid turbulence schemes]
- 100 m < Δ < 1 km
  - A PBL scheme will still be needed for most cases
  - Shallow cumulus probably can be turned off (not for Δ > 500 m)
  - Advection Scheme: better use a monotonic/non-osciallaory option (adv_opt ≥ 2)

(Bryan, 2014)

Case Study: The Derecho of 29-30 June 2012

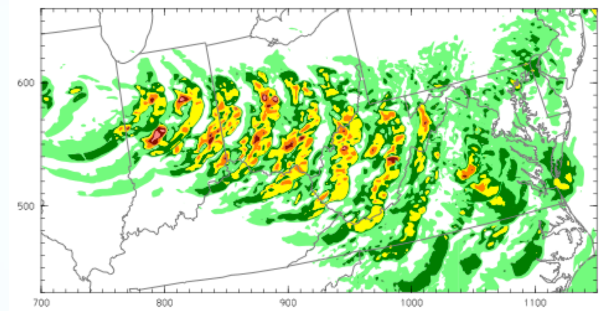


Radar composite reflectivity



WRF simulation of maximum reflectivity, DX=3km, initialized at 1200 UTC 29 June

Simulated maximum wind



3-km run

15-km run

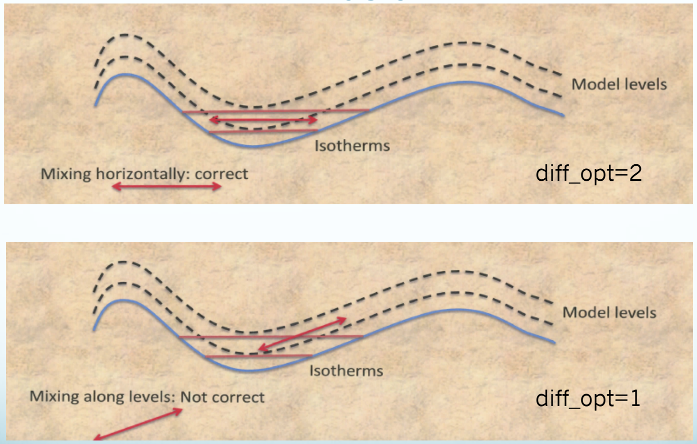
Model Levels and High Tops

- At least 30 or more levels for a model top at 50 mb
  - For high tops < 50 hPa
    - Stratosphere option for base state: Iso_temp=200 K. This prevents base state from becoming unrealistically cold.
    - Since V3.6.1, a positive lapse rate is allowed in stratosphere
  - For tops near 1 hPa (45-50km), 60 or more levels are required.
  - Ozone climatology becomes important above 30 hPa, where some or all of the ozone layer are included
    - Use RRTMG since CAM monthly ozone is available in RRTMG
- Vertical grid distance should not be larger than 1000 m (Radiation, microphysics, less accurate lateral BC)
- If finer horizontal grid size is used, more levels will be needed in the vertical
- Make sure dz < dx

Complex Terrain

- Steep terrain (> 45 degrees) may cause numerical stability problems.
  - Increasing epssm (0.1->0.5 or even larger)
    - This is a sound wave damper that can stabilize slope treatment by dynamics
  - For large slopes, set diff_opt=2
    - diff_opt=1 is less realistic than diff_opt=2, and diff_opt=2 used to be less stable but becomes more stable in recent versions
  - For V3.6 and later version, diff_opt=2 and km_opt=4 can be used together to improve stability

Diffusion



Dudhia (2014)

Selecting Model Physics

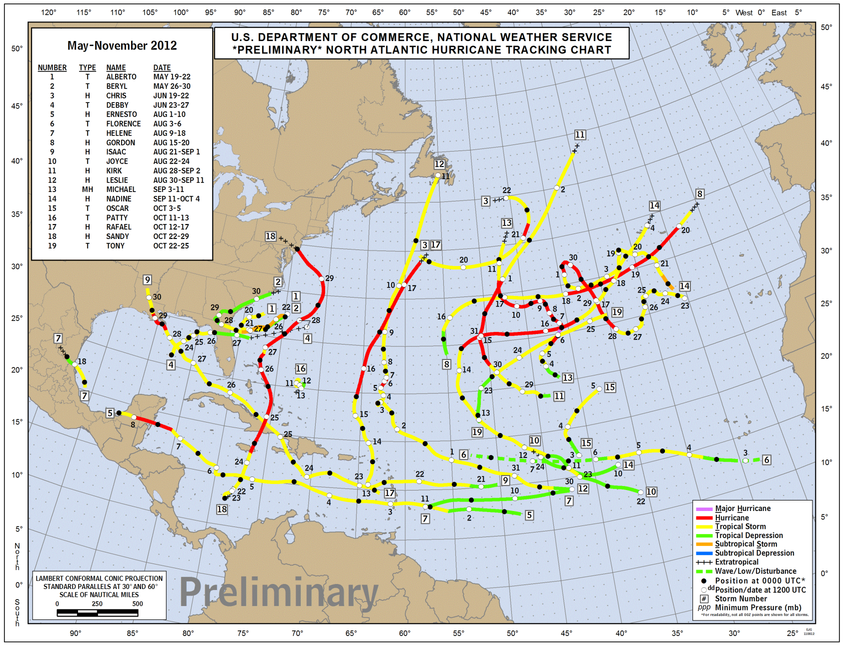
- Many options = more works
  - http://www2.mmm.ucar.edu/wrf/users/phys_references.html
  - http://www2.mmm.ucar.edu/wrf/users/docs/wrf-phy.html
- Testing of multiple options for a particular application
  - A given set of physics will perform differently depending on domain size, location, initialization and phenomenon of interest
  - Certain combinations better tested than others, but still no guarantee for better performance

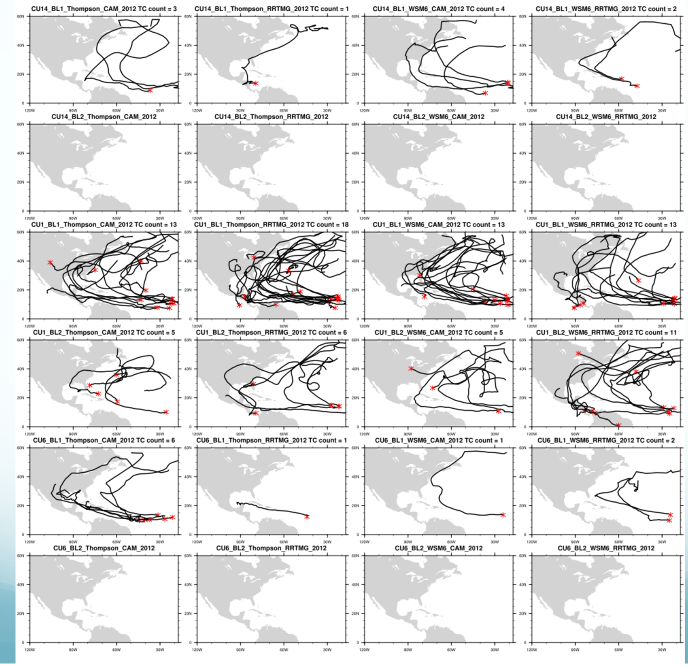
Physics in multi-scale model

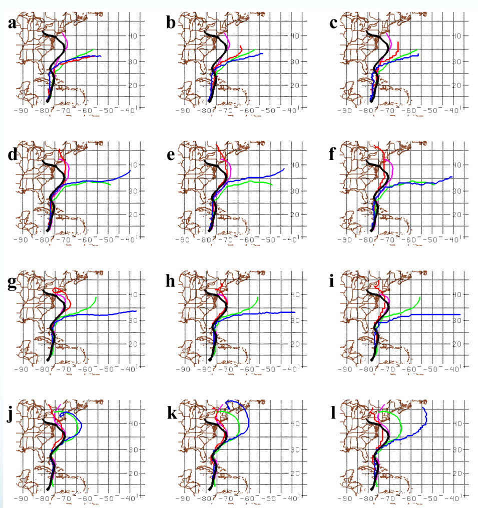
- Grid size and cumulus
  - DX > 10km, yes
  - DX < 4km, probably not
  - Grey Zone: 5-10km, no consensus, may try to use scale-aware cumulus scheme, such as GF, MSKF.
- Grid size and microphysics
  - For DX > 10km, no complex scheme is necessary
  - For DX <4km (convection-resolving), need at least graupel

Physics in Multi-scale Model

- Grid Size and PBL
  - PBL assumes all eddies are unresolved
    - DX > 500 m, PBL should be activated
  - LES assumes eddies are well resolved
    - DX < 100 m, LES should be applied
  - For DX 100-500 m, either may work to some extent
  - Terra incognita: resolved CISCs, violation of PBL assumption, and unresolved interaction between CISC and smaller scale turbulence.







- ECMWF(pink)
- GFS(green)
- TWRF(red, Tiedtke)
- SWRF(blue, SAS)

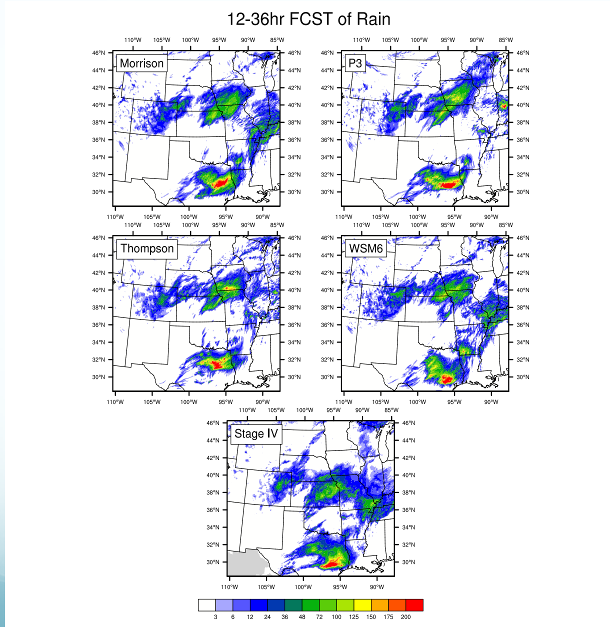
(Grid interval from left to right: 30, 60, 90-km;
Top two: initialized at 0000 and 1200 UTC 23 Oct.;
Bottom two: initialize at 0000 and 1200 UTC 24 Oct.)

Simulation of Hurricane Sandy: why such a large difference?

Bassill (2014)

Test of Sandy Simulation

- *For this case*, cumulus parameterization is the dominant driver of forecast track accuracy
- Poor track forecasts by the GFS/GEFS are not due to 'inappropriate' initial conditions, nor are they consequences of the differences in model resolution
- These types of examples serve to emphasize the importance of parameterization development as a necessary condition for forecast improvement

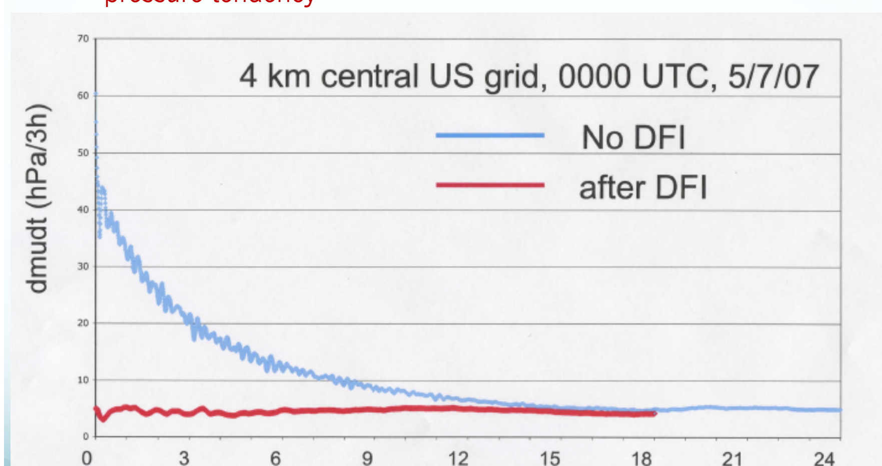


Other Options That May Be Considered

Example:

- Upper level damping over topography
- Gravity-wave drag if resolution is coarse
- Digital Filter Initialization
- Horizontal Diffusion
- Spectral Nudging

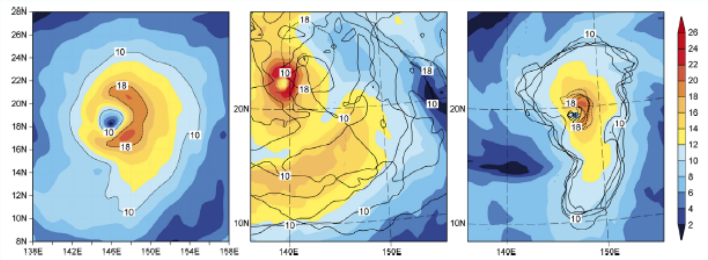
Domain average 3-hourly dry-hydrostatic column pressure tendency



Forecast Hours

Spectral Nudging

- It is useful for controlling longer wave phases. Compensates for errors due to low-frequency narrow lateral boundaries
- The “spectral nudging” method imposes time-variable large-scale atmospheric states on a regional atmospheric model
- Spectral nudging may be seen as a suboptimal and indirect data assimilation technique.
  - Wave number is selected so that domain size/wavenumber =~1000km in X and Y direction
  - Nudge U, V, THETA, Geopotential (not QV, since it has no wave pattern)
  - Can nudge in all levels or use ramp above a specified model level (if_zfac_ph, k_zfac_ph, etc.)
- However, strong nudging may reduce or filter out extreme events since nudging pushes the model toward a relatively smooth, large-scale state.



Horizontal 10 m wind speed fields (m s–1) for typhoon Songda (200418), on 1 September 2004, 0:00.
From left: CFSR reanalysis, CCLM-NN, CCLM-SN.

(Frauke Feser1 and Monika Barcikowska, Environmental Research Letters, 2012)

WRF Data, Utilities and Post-processing

Kelly Werner

WRF Data, Utilities & Post-processing

Kelly Werner

January 2018

Input Data

Input Data: Mandatory Fields

- **3D Data (data on pressure levels, for example)**
  - Temperature
  - U and V components of wind
  - Geopotential Height
  - Relative Humidity/Specific Humidity
- **2D Data**
  - Surface pressure
  - Mean sea-level pressure
  - Skin temperature/SST
  - 2 meter temperature and relative humidity
  - 10 meter U and V components of wind
  - Soil data (temperature and moisture) and soil height
- **Recommended Fields**
  - LANDSEA mask field for input data
  - Water equivalent snow depth
  - SEAICE
  - Additional SST data

External Data Sources: Global

Name	Resolution	Coverage	Temporal Availability	Website
NCEP/NCAR Reanalysis (R1/NNRP)	209 km 6-hourly	Global	Jan 1948 – present	http://rda.ucar.edu/datasets/ds090.0
NCEP/DOE Reanalysis (R2)	209 km 6-hourly	Global	Jan 1979 – present	http://rda.ucar.edu/datasets/ds091.0
ERA Interim Data	1.125° - 0.703° 6-hourly	Global	Jan 1979 – present	http://rda.ucar.edu/datasets/ds627.0
ECMWF's Operational Model Analysis	Varying		Jan 2011 – present	http://rda.ucar.edu/datasets/ds113.0
NCEP GDAS/FNL Reanalysis	0.25° 6-hourly	Global	July 2015 – present	http://rda.ucar.edu/datasets/ds083.3
GFS Real-time	1°	Global		ftp://ftpprd.ncep.noaa.gov/pub/data/nccf/com/gfs
NCEP GFS/FNL Reanalysis	1° 6-hourly	Global	Aug 1999 – present	http://rda.ucar.edu/datasets/ds083.2
GFS Gridded Model Data	0.5° 24-hourly	Global	Dec 2002 – present	http://rda.ucar.edu/datasets/ds335.0
NCEP GFS 0.25°	0.25° 3-hourly & 12-hourly	Global	Jan 2015 – present	http://rda.ucar.edu/datasets/ds084.1

External Data Sources: North America

Name	Resolution	Coverage	Temporal Availability	Website
NAM Real-time	32/12 km 6-hourly	North America		ftp://ftpprd.ncep.noaa.gov/pub/data/nccf/com/nam
NAM Analysis	12 km 6-hourly	North America	Jan 2012 – present	http://rda.ucar.edu/datasets/ds609.0
GCIP NCEP Eta	40 km 3-hourly & 6-hourly	North America	April 1995 – present	http://rda.ucar.edu/datasets/ds609.2
NCEP NARR	32 km 3-hourly	North America	Nov 1979 – present	http://rda.ucar.edu/datasets/ds608.0

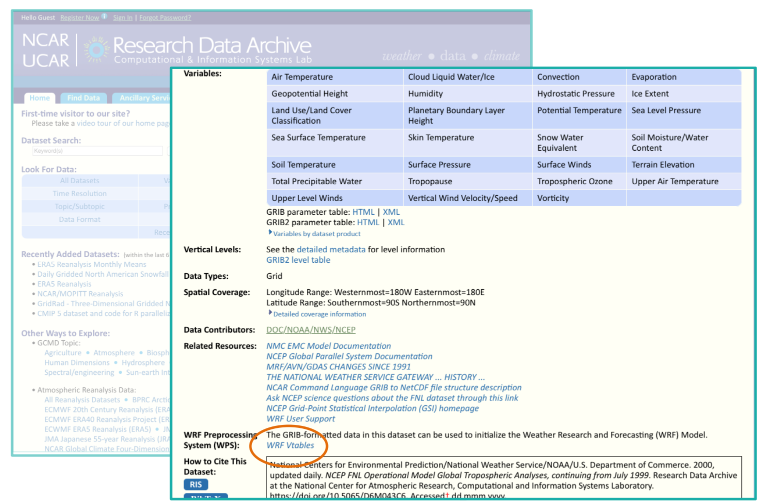
External Data Sources: Climate

Name	Resolution	Coverage	Temporal Availability	Website
NCEP Climate Forecast System Reanalysis (CFSR)	0.3° to 2.5° 6-hourly	Global	Jan 1979 – Dec 2010	http://rda.ucar.edu/datasets/ds093.0
NCEP Climate Forecast System Reanalysis II (CFSv2)	0.2° to 2.5° 6-hourly	Global	Jan 2011 – present	http://rda.ucar.edu/datasets/ds094.0
NCAR CESM CMIP5 data (netCDF format)	6-hourly	Global	Jan 1950 – 2100	http://rda.ucar.edu/datasets/ds316.0
NCAR CESM CMIP5 data (IM – Bias Corrected)	6-hourly	Global	Jan 1951 – 2100	http://rda.ucar.edu/datasets/ds316.1
SST DATA				
NCEP SST Analysis	1° - 1/12°	Global		http://polar.ncep.noaa.gov/sst
NOMAD3 SST	1° - 0.25°	Global	Jan 1854 – present (depending which product)	http://nomads.ncdc.noaa.gov/data.php
NCEP & NCDC Reconstructed SST	1° - 2°	Global	Jan 1854 – Dec 2015	http://rda.ucar.edu/datasets/ds277.0

External Data Sources: RDA

http://rda.ucar.edu

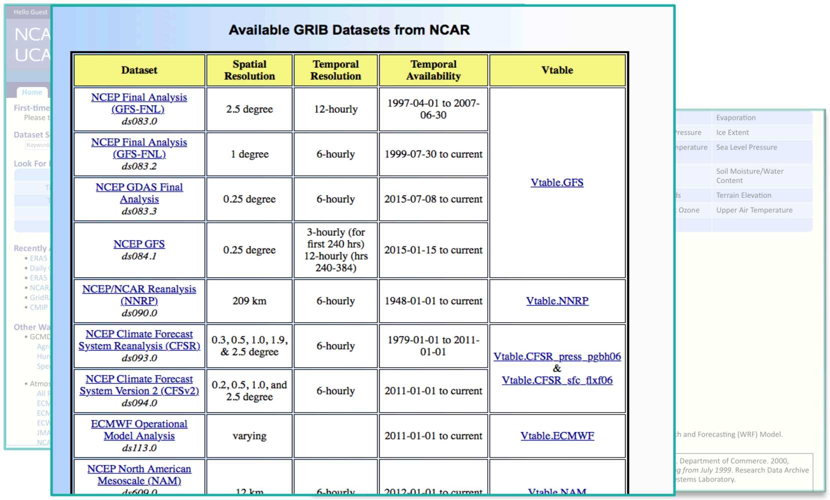
External Data Sources: RDA



https://rda.ucar.edu/datasets/ds083.2/

External Data Sources: RDA

http://www2.mmm.ucar.edu/wrf/users/download/free_data.html

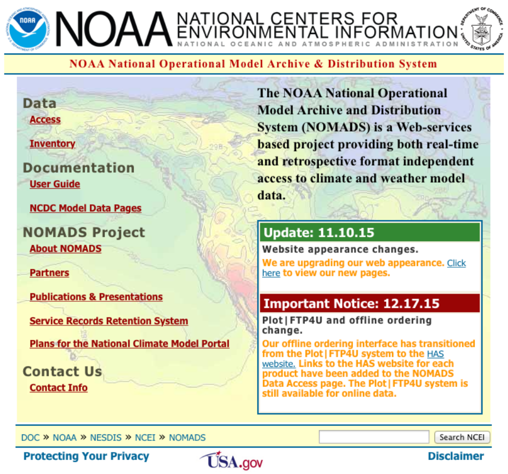


Available GRIB Datasets from NCAR

Dataset	Spatial Resolution	Temporal Resolution	Temporal Availability	Vtable
NCEP Final Analysis (GFS-FNL) *ds083.0*	2.5 degree	12-hourly	1997-04-01 to 2007-06-30	Vtable.GFS
NCEP Final Analysis (GFS-FNL) *ds083.2*	1 degree	6-hourly	1999-07-30 to current	
NCEP GDAS Final Analysis *ds083.3*	0.25 degree	6-hourly	2015-07-08 to current	
NCEP GFS *ds084.1*	0.25 degree	3-hourly (for first 240 hrs) 12-hourly (hrs 240-384)	2015-01-15 to current	
NCEP/NCAR Reanalysis (NNRP) *ds090.0*	209 km	6-hourly	1948-01-01 to current	Vtable.NNRP
NCEP Climate Forecast System Reanalysis (CFSR) *ds093.0*	0.3, 0.5, 1.0, 1.9, & 2.5 degree	6-hourly	1979-01-01 to 2011-01-01	Vtable.CFSR_press_pgbh06 & Vtable.CFSR_sfc_flxf06
NCEP Climate Forecast System Version 2 (CFSv2) *ds094.0*	0.2, 0.5, 1.0, and 2.5 degree	6-hourly	2011-01-01 to current	
ECMWF Operational Model Analysis *ds113.0*	varying		2011-01-01 to current	Vtable.ECMWF
NCEP North American Mesoscale (NAM) *ds609.0*	12 km	6-hourly	2012-01-01 to current	Vtable.NAM

External Data Sources: NOMADS

http://nomads.ncdc.noaa.gov



NAM
GFS
RUC
CFS
NARR
R1/R2
SST

Utilities

- Grib and Intermediate Data
- Designing a model domain
- netCDF tools
- Other Utilities
- ImageMagick
- Special WRF Output Variables
- OBSGRID
- MET

GRIB Data Handling

- **Documents**
  - **https://rda.ucar.edu/index.html#gribdoc (GRIB1 data)**
  - **https://rda.ucar.edu/index.html#grib2doc (GRIB2 data)**
- **Decoders**
  - ***wgrib, wgrib2, unpackgrib2.c, grib2to1.c***
    http://rda.ucar.edu/#!GRIB
    http://www.cpc.ncep.noaa.gov/products/wesley/wgrib.html
    http://www.cpc.ncep.noaa.gov/products/wesley/wgrib2
  - ***g1print.exe and g2print.exe***
    - Show data available in GRIB1 and GRIB2 files
    - Available from util/ directory in WPS
- **grib2ctl.pl**
  - Create .ctl and .idx files, so that you can plot GRIB files with GrADS
  - http://www.cpc.ncep.noaa.gov/products/wesley/grib2ctl.html
- **ncl_convert2nc**
  - Converts from grib format to netcdf format
    http://www.ncl.ucar.edu/Document/Tools/ncl_convert2nc.shtml

Writing Intermediate File Format

- http://www2.ucar.edu/wrf/users/docs/user_guide_V3/users_guide_chap3.htm#_Writing_Meteorological_Data
- **wrf_wps_write_int**

```
FIELD = "SST"
UNITS = "K"
DESC = "Sea Surface Temperature"

opt = True
opt@map_source              = "ERA-I Data"
opt@projection              = 0
opt@startloc                = "SWCORNER"
opt@startlon                = 0.0
opt@startlat                = -90.0
opt@deltalon                = 1.25
opt@deltalat                = 0.942408
opt@is_wind_earth_relative  = False
opt@date                    = "2015-07-26_00:00:00"
opt@level                   = 200100.

wrf_wps_wrtie_int(IM_name,FIELD,UNITS,DESC,VAR(:,:),opt)
```

Reading Intermediate Format Files

- **wrf_wps_read_int**

! opens file
istatus = wrf_wps_open_int(*filename)*

! reads header
wrf_wps_rdhead_int(istatus,head_real,field,hdate, \
units,map_source,desc)

! reads slab
Slab = wrf_wps_rddata_int(istatus,nx,ny)

! Loop until reaching the end of the file

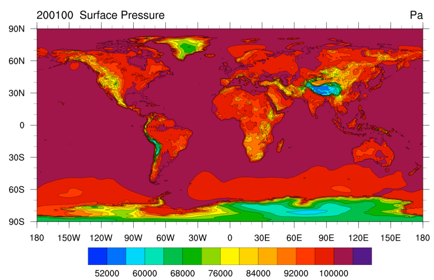
- **rd_intermediate**

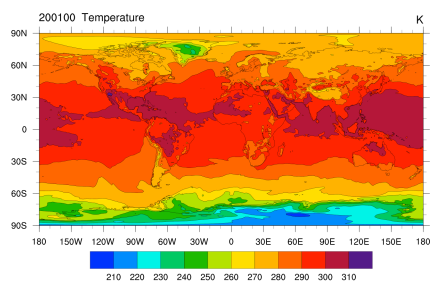
```
================================================
FIELD = TT
UNITS = K DESCRIPTION = TEMPERATURE
DATE = 2000-01-24_12:00:00 FCST = 0.000000
SOURCE = unknown model from NCEP GRID 212
LEVEL = 200100.000000
I,J DIMS = 185, 129
IPROJ = 1
   REF_X, REF_Y = 1.000000, 1.000000
   REF_LAT, REF_LON = 12.190000,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

Utility: plotfmt

- The plotfmt program plots the fields in the ungribbed intermediate files

`ncl plotfmt.ncl 'filename="FNL:2007-09-15_00"'`



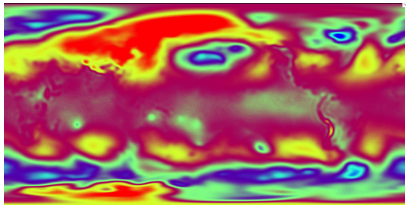


Plotting Intermediate Files in netCDF Format

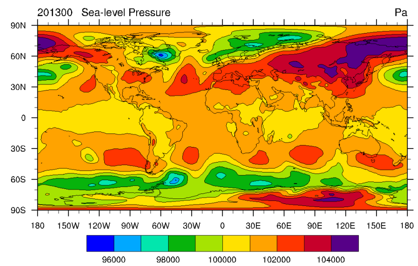
- Use the utility **int2nc.exe**
  - Converts intermediate files created by ungrib.exe to netcdf format
  - `./int2nc.exe FILE:`*`yyyy-mm-dd_hh`*
- To plot: **plotfmt_nc.ncl**

`ncl plotfmt_nc.ncl 'inputFILE="FNL:2007-09-15_00.nc"'`

Plot Using ncview



Plot Using plotfmt_nc.ncl

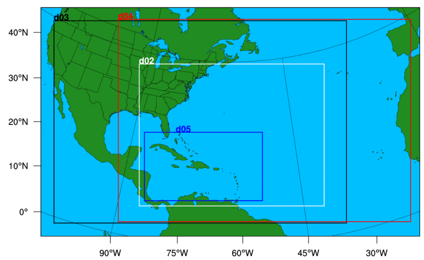


Model Domain Design

```
mpres@mpFillColors  =
 (/"background","DeepSkyBlue",
 "ForestGreen","DeepSkyBlue",
 "transparent"/)

mpres@mpGridSpacingF   = 45

lnres@domLineColors    = (/
 "white", "Red" , "Red" , "Blue" /)

mpres@mpOutlineBoundarySets
 "NoBoundaries"  ; "Geophysical"
 "National"      ; "USStates"
 "GeophysicalAndUSStates"
 "AllBoundaries"
pares = True
 pmres@gsMarkerColor = "White"
 pmres@gsMarkerIndex = 16
 pmres@gsMarkerSizeF = 0.01
 gsn_polymarker(wks,mp,-
 77.26,38.56,
                pmres)
```

- plotgrids.ncl
  - *WPS/util/plotgrids.ncl*
  - Reads namelist information to generate plot
  - X11, png, pdf

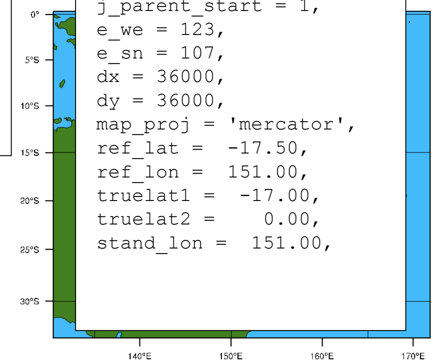
Test Domain



Model Domain Design

```
DOMS = 1
DX = 36.
MAP = "mercator"
LAT1 = (/ -35.0, -45., -27. /)
LAT2 = (/    0., -20., -23. /)
LON1 = (/ 131., 121., 125./)
LON2 = (/ 171., 159., 131./)
parent_id = (/ 0, 1, 2 /)
parent_grid_ratio = (/ 1, 3, 3 /)
```

design_grids.ncl




```
Suggested namelist options
 parent_id = 0,
 parent_grid_ratio = 1,
 i_parent_start = 1,
 j_parent_start = 1,
 e_we = 123,
 e_sn = 107,
 dx = 36000,
 dy = 36000,
 map_proj = 'mercator',
 ref_lat =  -17.50,
 ref_lon =  151.00,
 truelat1 =  -17.00,
 truelat2 =    0.00,
 stand_lon =  151.00,
```


netCDF Tools

NCO Tools

http://nco.sourceforge.net

- netCDF Operators are command-line programs that take netCDF (HDF and/or DAP) files as input, then operate (e.g., derive new data, compute stats, print, manipulate metadata) and output to the screen or files in various formats (text, binary, netCDF, etc.)

- ncdiff
  - Shows the differences between 2 files

  ncdiff input1.nc input2.nc diff.nc

- ncrcat (nc cat)
  - Writes specified variables/times to a new file, or concatenates files

  ncrcat -d file1.nc file2.nc combined.nc
  ncrcat -d Time,0,231 -v RAINNC wrfout* RAINNC.nc

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

  ncra -v T2 file1.nc file2.nc -o T2.nc
  ncra -v T2 wrfout* -o T2.nc

NCO Tools (continued)

http://nco.sourceforge.net

- ncrename
  - Renames variables, dimensions, attributes

```
ncrename -v LANDUSE,LAND -a missing_value,_FillValue
file.nc
```

- ncks (nc kitchen sink)
  - Combination of several NCO tools to allow cutting/pasting subsets of data into a new file
    - Extracting a specific variable

```
ncks -v RAINNC wrfout_d01_2015-06-01_00:00:00 RAINNC.nc
```

    - Splitting files

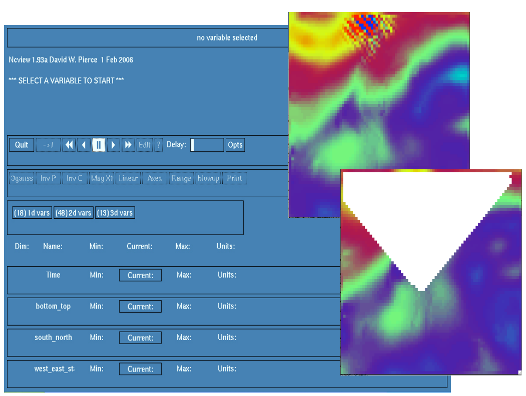
```
ncks -d Time,1,1 wrfout -o wrfout1.nc
```

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
- **ncwa:** weighted averager

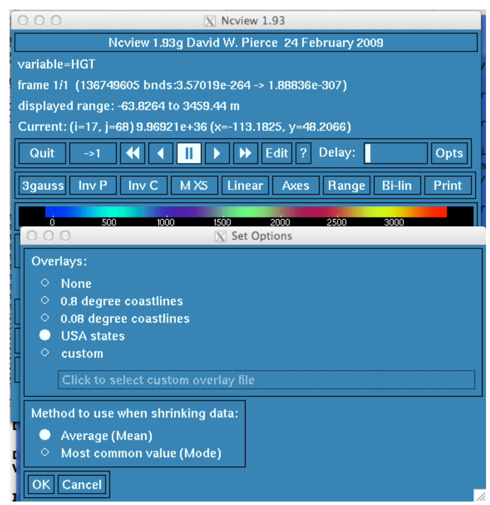
ncview

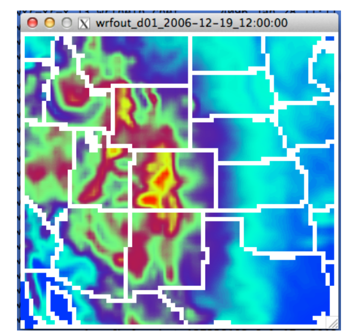
http://meteora.ucsd.edu/~pierce/ncview_home_page.html



- A graphical interface that allow quick viewing of netCDF files
  - All variables found in file
  - Detect where things go wrong
- Other options
  - Time series
  - Vertical Cross Section
- WRF/WPS files
  - Any netCDF format file
    - geo_em.d0*, met_em.d0*, wrfinput_d0*, wrfout.d0*, wrfrst.d0*

ncview

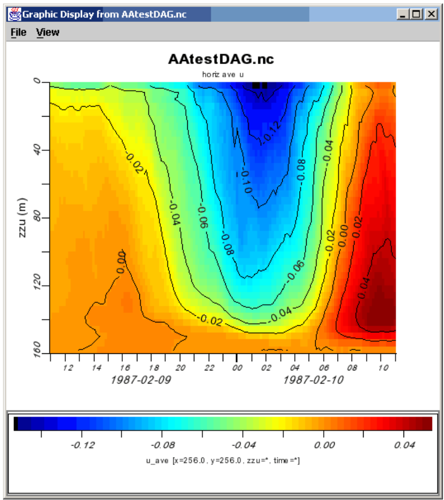




- Beginning V3.7
- Works with wrfinput* and wrfout* files
- Must have 1 time period per file

ncBrowse

http://www.epic.noaa.gov/java/ncBrowse/



ncdump

- Reads a netCDF dataset and prints information from that dataset
- `ncdump -h file`
  - Prints header (inclusive list of variables in the file)
- `ncdump -v VAR file`
  - Prints specific data for the variable 'VAR'
- `ncdump -v Times file`
  - Prints the times that are included in the file

ncdump –v Times

```
netcdf wrfout_d01_2000-01-24_12:00:00 {
dimensions:
        Time = UNLIMITED ; // (3 currently)
        DateStrLen = 19 ;
        west_east = 73 ;
        south_north = 60 ;
        west_east_stag = 74 ;
        bottom_top = 27 ;
        south_north_stag = 61 ;
        bottom_top_stag = 28 ;
variables:
        char Times(Time, DateStrLen) ;
        float LU_INDEX(Time, south_north, west_east) ;
                LU_INDEX:FieldType = 104 ;
                LU_INDEX:MemoryOrder = "XY " ;
        LU_INDEX:description = "LAND USE CATEGORY" ;
                LU_INDEX:units = "" ;
                LU_INDEX:stagger = "" ;
………
………

global attributes:

                :TITLE = " OUTPUT FROM WRF V3.4.1 MODEL";
                :START_DATE = "2000-01-24_12:00:00" ;
                :WEST-EAST_GRID_DIMENSION = 74 ;
                :SOUTH-NORTH_GRID_DIMENSION = 61 ;
                :BOTTOM-TOP_GRID_DIMENSION = 28 ;
                :DX = 30000.f ;
                :DY = 30000.f ;
………
………

data:
 Times =
  "2000-01-24_12:00:00",
  "2000-01-24_18:00:00",
  "2000-01-25_00:00:00"
```

Other Utilities

- Additional utilities
  - **read_wrf_nc**: reads WRF netCDF file, outputs various data
  - **iowrf**: extracts a box from WRF netCDF files, thin or destagger data
  - **wrf_interp**: interpolates WRF output files to pressure, height-agl, height-msl, potential temp, and equivalent potential temp, and can perform underground extrapolation
  - **p_interp**: converts wrfout data to pressure levels
  - **v_interp**: adds vertical levels in WRF input and boundary files
  - **diffwrf**: performs several functions, including making comparisons of two WRF files
  - For more details on the above utilities, see:

  http://www2.mmm.ucar.edu/wrf/users/utilities/util.htm
- To download utilities:
  http://www2.mmm.ucar.edu/wrf/users/download/get_sources.html

ImageMagick

http://www.imagemagick.org

- Converts graphical files from one format to another
  convert *file*.pdf *file*.png
  convert *file*.png *file*.bmp
- Many options available
  - Rotate frames, trim white space, etc.
  - 2 ways to use
    1) *display plot.png*
    2) *Convert –trim +repage –background white –flatten plot.pdf plot.png*
- Can make movies
  - Can create individual frames for each image
- Maintains high resolution – great for publishing!
- Cannot deal with .ncgm files

Special WRF Output Variables

- The WRF model outputs the state variables defined in the Registry file, and these state variables are used in the model's prognostic equations. Some of these variables are perturbation fields and therefore, the following definitions for reconstructing meteorological variables are necessary:

Total geopotential	PH + PHB
Total geopotential height in m	(PH + PHB) / 9.81
Total potential temp in K	T + 300
Total pressure in mb	(P + PB) * 0.01
Wind components, grid relative	U, V
Surface pressure in Pa	Psfc
Surface winds, grid relative	U10, V10 (valid at mass points)
Surface temp and mixing ratio	T2, Q2

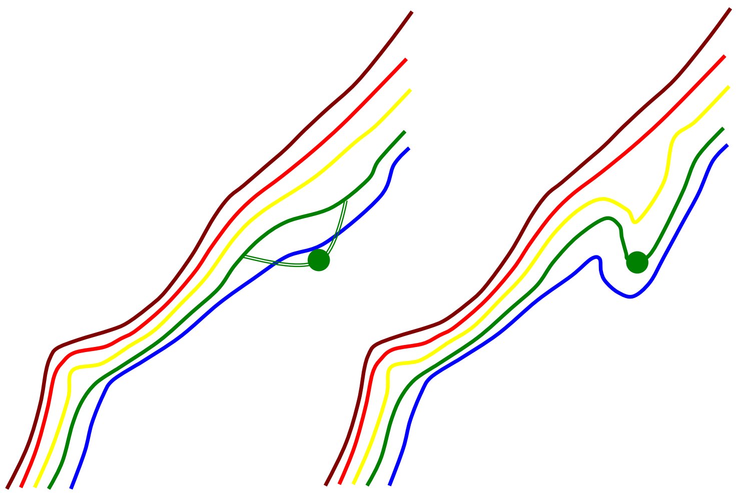
See WRFV3/Registry/Registry.EM_COMMON for description of variables

OBSGRID

OBSGRID

- To improve a first-guess gridded analysis by incorporating additional observational information
  - Traditionally first-guess analysis came from low-resolution global analysis and forecast grids
  - These days, higher-resolution, regional scale analyses are more readily available
- When is this method useful?
  - When using very coarse resolution first-guess input data
  - If you conducted a field campaign and have acquired very high-resolution station data (for example)

OBSGRID: Basic Concept



OBSGRID: How to Run

- Get the source code
  http://www2.mmm.ucar.edu/wrf/users/downloads.html
- Compile
- Prepare observation files
- Edit the namelist.oa
- Link in met_em* files from WPS
- Run the program
  – ./obsgrid.exe
- Check your output

See the WRF Users' Guide for detailed information
http://www2.mmm.ucar.edu/wrf/users/docs/user_guide_V3.8/users_guide_chap7.htm

OBSGRID: How to Use to Run WRF

- Link the 'metoa_em*' files to WRF running directory
  ln –sf ../../../OBSGRID/metoa_em.d01.* .
- Add the following to the &time_control section of the namelist
  auxinput1_inname = "metoa_em.d<domain>.<date>"
- Run real.exe
- Run wrf.exe

OBSGRID - Grid Nudging - Surface

- If you are interested in doing surface analysis nudging
- OBSGRID creates a file called wrfsfdda_d0*
- How to use this:
  – In &fdda, set grid_fdda = 1 and grid_sfdda = 1
  – Run real.exe and get a file called wrffdda_d01, and use with wrfsfdda_d01, wrfinput_d01, and wrfbdy_d01
  – Run wrf.exe
- For more information, refer to Jimy Dudhia's ARW Nudging talk

OBSGRID – Observation Nudging

- Allows for input observation data & quality control
- Used if you have a large number of extra observations, and a single case study (not recommended for climate studies)
- Can get obs data from CISL (little R format)
- How to use this
  - OBSGRID creates files called OBSDOMAIN_XXX (can concatenate files into 1: OBSDOMAIN_101)
  - In &fdda, add obs_nudge_opt = 1
  - In &time_control, add auxinput11_interval_s = *180,* auxinput11_end_h = *24*
  - Will need OBSDOMAIN_101, wrfinput_d01 and wrfbdy_d01 files
  - Run real.exe and wrf.exe as usual
- For more information, see http://www2.mmm.ucar.edu/wrf/users/wrfv3.1/How_to_run_obs_fdda.html and Jimy Dudhia's ARW Nudging talk

MET Verification Software

- Model Evaluation Tools (MET)
- Provides all the basics (e.g., RMSE, bias, skill scores)
- Provides
  - Advanced spatial methods (wavelets, objects)
  - Confidence intervals
- Download it
  http://www.dtcenter.org/met/users/downloads/
- Support
  met_help@ucar.edu
- Documentation
  http://www.dtcenter.org/met/users/docs/overview.php

Post-processing

- Supported Packages
- ARWpost
- RIP4

Supported Post-processing Packages

http://www2.mmm.ucar.edu/wrf/users/docs/user_guide_V3/contents.html

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) (wrfhelp@ucar.edu)
RIP4	9-20	Converter and interface to graphical Package, NCAR graphics (wrfhelp@ucar.edu)
UPP	9-36	Converter (GrADS & GEMPAK) (upp-help@ucar.edu)
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?
- How easy is it to add diagnostics?
- 3D or 2D visualization?
- Can it handle staggered grids?
- 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

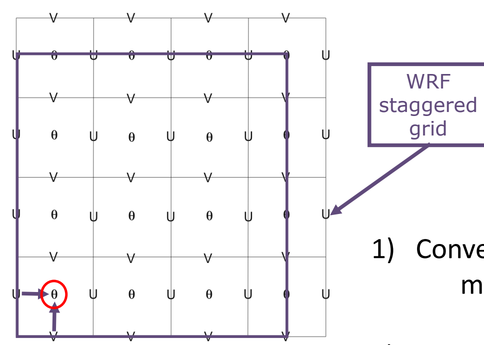
	NCL	RIP4	GrADS	UPP	VAPOR	IDV
netCDF		ripdp	ARWpost	converter	converter	converter
GRIB						
ASCII						
shapefiles						
geogrid & metgrid output						
intermediate file format	*V6.2.0* *V6.3.0*					
wrfinput data						
Idealized data						
wrfoutput						
big data						

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?



1) Converter co-locates data to mass points

2) Converter translates variables - e.g., “T” is not really temp. Must add 300 for actual temp (K)

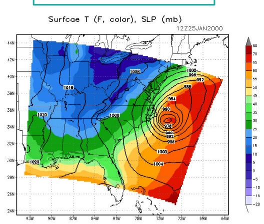
ARWpost

ARWpost: General Information

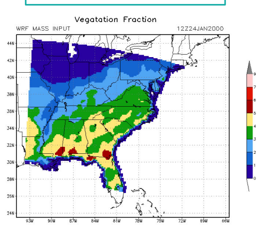
- Converter
  - Reads in wrf-arw model data, creates GrADS output files
  - Requires GrADS to display
- GrADS software is only needed to display data, not needed to compile the code
  - http://www.iges.org/grads/grads.html
- Generate a number of graphical plots
  - Horizonal
  - Cross-section
  - skewT
  - 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

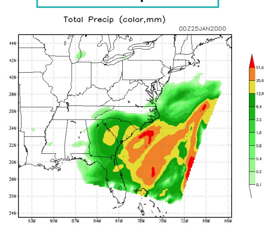
Surface Temp



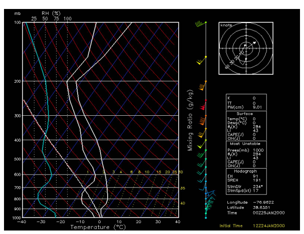
Vegetation Fraction



Total Precipitation

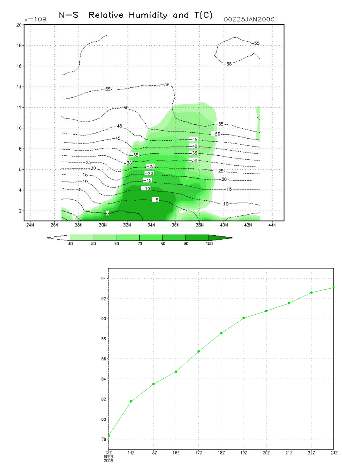


Skew-T Diagram

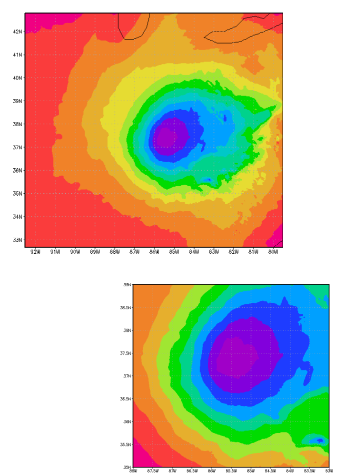


ARWpost: Example Functions

Cross-Sections



Zooming



ARWpost: Diagnostics

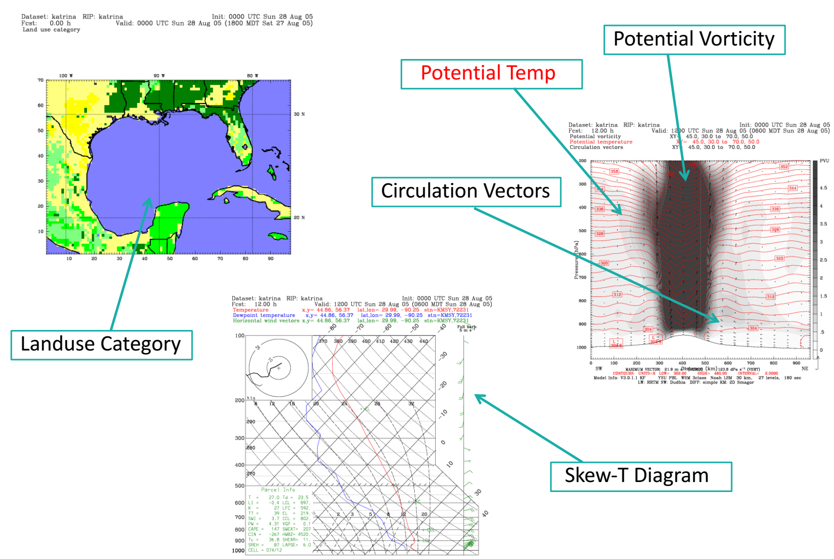
- cape – 3d cape
- cin – 3d cin
- mcape – maximum cape
- mcin – minimum cin
- clfr – low/middle/high cloud fraction
- dbz – 3d reflectivity
- max_dbz – maximum reflectivity
- geopt – geopotential
- height – model height in km
- lcl – lifting condensation level
- lfc – level of free convection
- pressure – 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

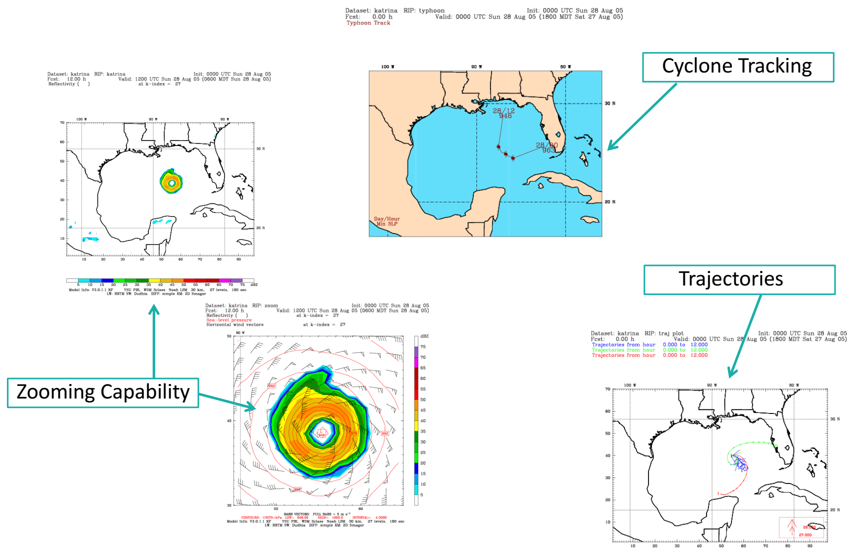
Script Name	Description
cbar.gs	Plots a color bar on shaded plots
rgbset.gs	Allows you to add/change colors from color # 20 – 99
skew.gs	Program to plot a skewT
plot_all.gs	Automatically finds all .ctl files in the directory and lists them so the user can pick when to use, will plot all fields chosen
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

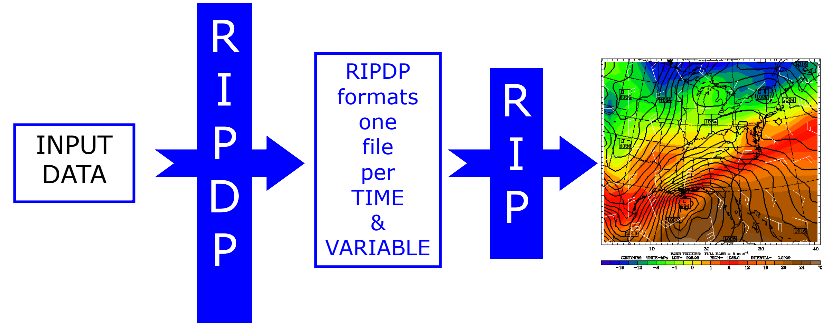
RIP4: Example Plots



RIP4: Example Plots



RIP4: Program Flow



RIP4: Namelist (*&userin)*

- Use namelist to control
  - processing times, intervals, title information, text quality on a plot
  - whether to do time series, trajectory, or to write output for Vis5D
  - *Full explanation for namelist variables is available in the user document*

- ptimes, ptimeunits – times to process
- tacc – tolerance for processing data
- iusedaylightrule – 1 applied, 0 not applied
- idotser – generate time series output
- icgmsplit – split metacode into several files
- itrajcalc – 0, 1 ONLY when doing trajectory calculations
- rip_root - override RIP_ROOT
- ncarg_root - output type: X11, cgm, pdf, ps

RIP4: Common Error Message

GKS ERROR NUMBER 2 ISSUED FROM SUBROUTINE GCLKS :--GKS NOT IN PROPER STATE: GKS SHALL BE IN STATE GKOPFORTRAN STOP

- Usually NOT a graphics error.
- More often this is an error with the times you are asking RIP to process
  - Check the ptimes in your .in file
  - Check the xtimes files created by RIPDP

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

Questions?

Verification of WRF Simulations

Ming Chen

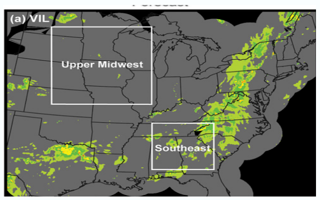
Verification of WRF Simulations

Ming Chen

National Center for Atmospheric Research

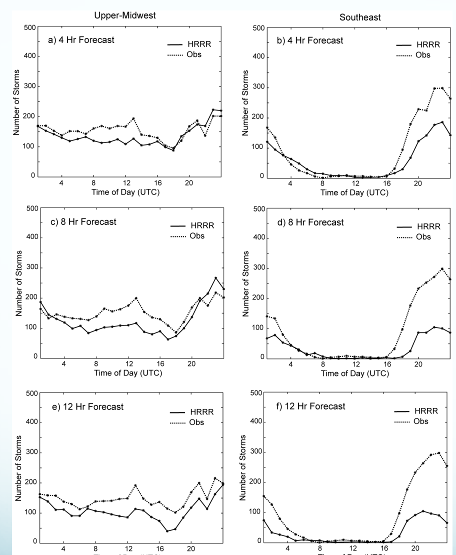
Verification of WRF Simulations

- Operational Forecasting
  - We need to monitor forecast quality – how accurate are the forecast?
- Research
  - Compare the performance of different schemes/ scheme combinations
  - To what extent does one scheme or one set of scheme combination give better simulation than another, and in what ways is that scheme better?
- Evaluation of WRF performance
  - Help users identify model weaknesses, strengths --- important for further improvement
  - We need to know what is wrong before we can improve



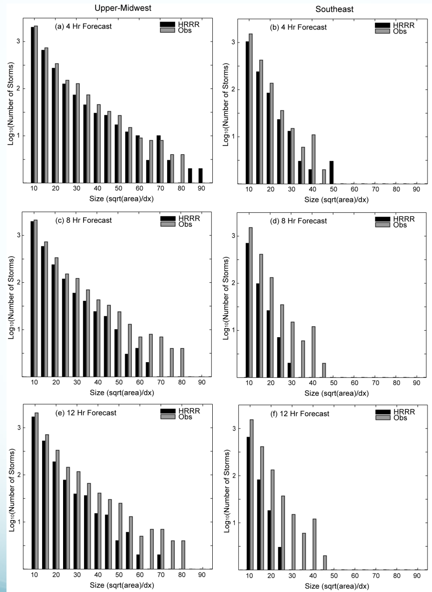
Model domains for convective storm forecasts by the High Resolution Rapid Refresh (HRRR) in the summer of 2010

(HRRR is a WRF- ARW based forecast system. See Weygandt et al. 2009; Benjamin et al.2011)



total number of storms as a function of time of the day

(Cai et al., 2015)



total number of storms as a function of the storm size

(Cai et al., 2015)

What can we find based on the verification?

- The diurnal variation of the total number of storms in the Southeast is stronger than that of the upper Midwest --- different forcing mechanisms are responsible for the storm initiation and evolution in these two subdomains.
- All forecasts for the upper Midwest showed almost simultaneous increases in the total number of storms compared to the observations starting at 1800 UTC --- fairly good timing of storm initiation
- All HRRR forecasts for the Southeast exhibited a significant delay or lack of new storms starting at 1700 UTC --- fewer new storms initialized in the model
- For longer forecast lead times the model tended to have fewer large storms compared with the observations in both the Midwest and the southwest --- large storms were not realistically maintained in the model

(Cai et al., 2015)

Verification of WRF Simulations

- Introduce methods for WRF simulation verification. The methods range from traditional statistics to methods for more detailed verification
- Give examples for each method
- Provide links and references for further information
- Does not provide source codes (details can be found in Model Evaluation Tools http://www.dtcenter.org/met/users/)

Recommendations on the Verification of WRF Simulations

- Types of forecast variable
  - Continuous
    - Temperature,
    - Precipitation
    - Winds, humidity, etc.
  - Categorical:
    - Rain vs no rain;
    - Strong winds vs no strong winds;
    - Fog vs no fog; clouds vs no clouds, etc.

Recommendations on the Verification of WRF Simulations – Continuous Variables

- **Mean Error (Bias):** a simplest and most familiar score to provide average direction of error

$$\text{Mean Error} = \frac{1}{n}\sum_{i=1}^{n}(f_i - o_i) = \bar{f} - \bar{o}$$

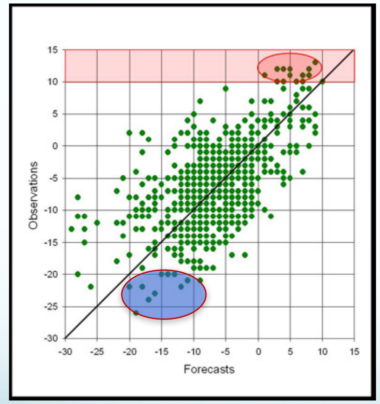
- **MAE:** average of the magnitude of errors (always view the ME and MAE simultaneously)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|f_i - o_i|$$

- **MSE (RMSE):** sensitive to large errors.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(f_i - o_i)^2 \qquad RMSE = \sqrt{MSE} = \frac{1}{n}\sum_{i=1}^{n}(f_i - o_i)^2$$

Verification of Continuous Variables: Scatterplot



All points with observed temperatures above the diagonal mean they are forecast too cold.

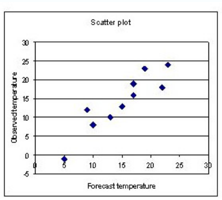
All the forecast is too low for T above +10?

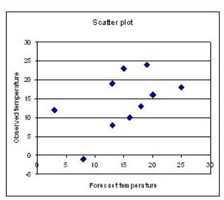
All the observed T below -20 are forecast too high except one

(http://www.eumetcal.org/)

Verification of Continuous Variables: Scatterplot

Below are two scatter plots representing two different sets of forecasts. The observations are the same in both cases. Can we say that these two sets of forecasts is positively correlated with the observations?





(http://www.eumetcal.org/)

Verification of Continuous Variables

- **Model-generated vertical profiles of variables**
  - Profiles of meteorological variables can be extracted from the WRF output files and placed on the desired location and time
    - use a sounding from the nearest grid point (i.e. no interpolation) to the desired location,
    - or use bilinear /inverse distance weight interpolation to horizontally interpolate WRF to the desired location
  - General rule for vertical interpolation: the pressure level intervals shouldn't be too large; for the vertical height levels, the layers can be very thin for close examination and allowed to be be thicker for regions of less detailed study
  - sources of comparison data may come from, for example, radar profilers and lidar for wind, microwave radiometers for temperature and moisture, and radio acoustic sounding systems for virtual temperature. Nevertheless radiosondes have remained the primary source of comparison data above the near surface layer

Sources of Observation Data

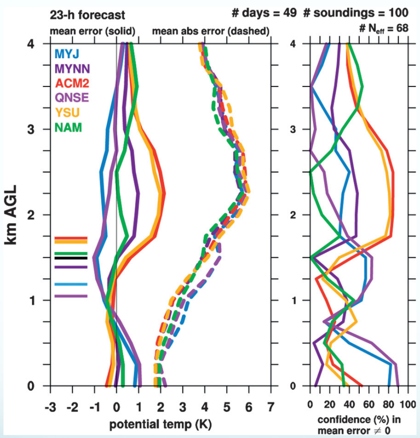
- Soundings
  http://www.weather.uwyo.edu/upperair/sounding.html
  This site contains WMO soundings in several formats
  http://www.esrl.noaa.gov/raobs)
  This site provides WMO sounding data, but requires different processing in the input function

- Verifications
  NCEP (http://www.emc.ncep.noaa.gov/gmb/STATS_vsdb/),
  ECMWFhttp://www.ecmwf.int/en/forecasts/charts/medium/monthly-wmo-scores-against-radiosondes)
  Worldwide comparisons are available for deterministic forecasts at
  http://apps.ecmwf.int/wmolcdnv/
  and for ensemble forecasts at the Japan Meteorological Agency (JMA)
  (http:/epsv.kishou.go.jp/EPSv/).

72469 DNR Denver Observations at 12Z 04 May 2016

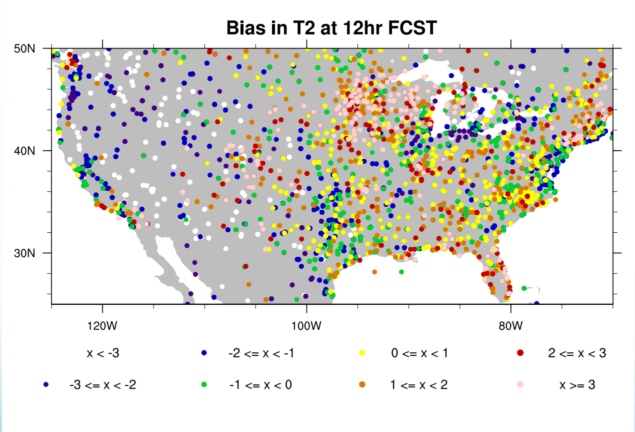
PRES hPa	HGHT m	TEMP C	DWPT C	RELH %	MIXR g/kg	DRCT deg	SKNT knot	THTA K	THTE K	THTV K
1000.0	160									
925.0	832									
850.0	1549									
844.0	1625	6.4	-0.6	61	4.36	175	5	293.4	306.4	294.2
842.0	1644	7.6	-1.4	53	4.12	180	6	294.9	307.3	295.6
834.0	1722	9.2	-2.8	43	3.75	199	10	297.4	308.8	298.1
824.0	1822	12.4	-2.6	35	3.85	223	15	301.8	313.8	302.5
823.3	1829	12.4	-2.7	35	3.84	225	15	301.9	313.8	302.6
802.0	2046	12.4	-4.6	30	3.40	204	6	304.1	314.9	304.8
793.5	2134	11.6	-4.3	33	3.52	195	3	304.2	315.3	304.8
765.0	2435	8.8	-3.2	43	3.97	210	4	304.4	316.8	305.1
764.8	2438	8.8	-3.2	43	3.96	210	4	304.4	316.8	305.1
749.0	2608	7.8	-5.2	39	3.48	268	3	305.1	316.2	305.8
736.7	2743	6.7	-4.8	44	3.65	315	2	305.4	316.9	306.1
724.0	2884	5.6	-4.4	49	3.83	313	4	305.7	317.8	306.4
708.0	3064	4.8	-8.2	38	2.92	311	7	306.8	316.2	307.3
700.0	3156	4.0	-9.0	38	2.78	310	9	306.9	315.8	307.4
678.0	3415	1.8	-10.2	41	2.61	305	12	307.2	315.7	307.7
657.8	3658	0.7	-13.6	33	2.05	300	15	308.7	315.5	309.1
643.0	3841	-0.1	-16.1	29	1.70	309	18	309.8	315.5	310.1
633.3	3962	-0.8	-20.5	21	1.19	315	20	310.4	314.4	310.6
620.0	4132	-1.7	-26.7	13	0.70	321	16	311.2	313.7	311.3
609.4	4267	-2.8	-27.0	14	0.69	325	13	311.4	313.9	311.6
586.0	4572	-5.3	-27.7	15	0.67	335	14	312.0	314.4	312.1
563.6	4877	-7.9	-28.5	17	0.65	330	16	312.5	314.8	312.6
533.0	5313	-11.5	-29.5	21	0.63	316	15	313.2	315.4	313.3
521.1	5486	-12.6	-34.1	15	0.41	310	14	313.9	315.4	313.9
516.0	5561	-13.1	-36.1	13	0.34	312	14	314.2	315.4	314.2
500.0	5800	-15.3	-35.3	16	0.38	320	12	314.3	315.7	314.4
480.5	6096	-17.9	-35.6	20	0.38	325	12	314.8	316.2	314.8
457.0	6470	-21.1	-36.1	25	0.39	329	14	315.2	316.7	315.3
438.0	6782	-22.5	-40.5	18	0.26	332	16	317.3	318.3	317.4
424.5	7010	-24.6	-41.1	20	0.25	335	17	317.5	318.5	317.6
411.0	7245	-26.7	-41.7	23	0.24	327	16	317.7	318.7	317.8
400.0	7440	-27.5	-45.5	16	0.16	320	16	319.2	319.8	319.2
392.0	7585	-28.5	-48.5	13	0.12	312	17	319.7	320.2	319.7
390.0	7620	-28.8	-48.7	13	0.12	310	17	319.8	320.3	319.8
357.4	8230	-33.7	-52.0	14	0.09	295	15	321.3	321.7	321.4
327.5	8839	-38.5	-55.2	16	0.07	310	12	322.8	323.0	322.8
315.0	9110	-40.7	-56.7	16	0.06	292	13	323.4	323.6	323.4
313.4	9144	-40.9	-56.9	16	0.06	290	13	323.5	323.7	323.5
300.0	9440	-42.9	-58.9	16	0.05	280	15	324.8	325.0	324.8
299.6	9449	-43.0	-59.0	16	0.04	280	15	324.8	325.0	324.8
281.0	9880	-45.5	-61.5	15	0.03	278	15	327.2	327.3	327.2
250.0	10650	-52.1	-71.1	8	0.01	275	14	328.5	328.5	328.5
242.0	10860	-53.7	-72.7	8	0.01	277	15	329.1	329.2	329.1
226.6	11278	-57.5	-71.1	16	0.01	280	18	329.6	329.6	329.6
225.0	11324	-57.9	-70.9	17	0.01	279	18	329.6	329.7	329.6
207.0	11847	-60.9	-70.9	26	0.01	266	15	332.9	332.9	332.9
205.7	11887	-61.3	-71.3	25	0.01	265	15	332.9	333.0	332.9
200.0	12060	-62.9	-72.9	25	0.01	270	19	333.0	333.1	333.0
189.0	12405	-65.3	-74.3	28	0.01	285	14	334.6	334.6	334.6
186.2	12497	-65.5	-74.6	27	0.01	285	14	335.8	335.8	335.8

Example: vertical profile verification against radiosondes



Forecasts at evening time --- (Coniglio et al., 2013, Weather and Forecasting)

Example : verification against station observations

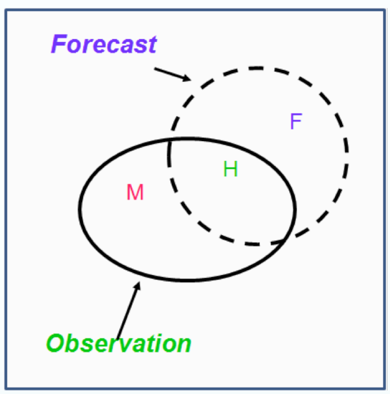


Sources of Observation Data

- Station Observations: GDAS prebufr format data
  NCEP FTP Site: ftp://ftpprd.ncep.noaa.gov/pub/data/nccf/com/gfs/prod
  BUFRLIB User Guide: http://www.nco.ncep.noaa.gov/sib/decoders/BUFRLIB/)
  - UPPER-AIR
  - AIRCRAFT REPORTS
  - SATELLITE-DERIVED WIND REPORTS
  - WIND PROFILER AND ACOUSTIC SOUNDER (SODAR) REPORTS
  - SURFACE LAND (SYNOPTIC, METAR) AND SURFACE MARINE (SHIP, BUOY, C-MAN PLATFORM) REPORTS

Verification of WRF Simulations – Categorical Variables

- Contingency table
- Several commonly used measures:
  - Accuracy
  - Frequency bias
  - Probability of detection
  - False alarm rate
  - Critical success index (Threat Score)
  - Gilbert Skill Score (ETS)
  - Heidke Skill Score



H: Hit M: Missed F: False Alarm

(NSSL 2012 Spring Forecast Experiment)

Verification of WRF Simulations – Categorical Variables

Contingency table in terms of counts: precipitation

Forecast	Observation		Total
	Yes	No	
Yes	Hits (YY)	False Alarm (YN)	YY+YN
No	Misses (NY)	Correct (NN)	NY+NN
total	YY+NY	YN+NN	T=YY+YN+NY+NN

Categorical Variables

Forecast	Observation		Total
	Yes	No	
Yes	Hits (YY)	False Alarm (YN)	YY+YN
No	Misses (NY)	Correct (NN)	NY+NN
total	YY+NY	YN+NN	T=YY+YN+NY+NN

Accuracy= (YY+NN)/(YY+YN+NY+NN)
what fraction of the forecasts were correct
Range: 0 to 1. Perfect score: 1

Threat Score (Critical Success Index)
CSI=TS=YY/(YY+NY+YN)
How well did the forecast "yes" events correspond to the observed "yes" events
Range: 0-1, 0 indicates no skill, 1 represents perfect score

Equitable Threat Score (Gilbert Skill Score)
$GSS=ETS=(YY - YY_{random})/(YY + NY + YN - YY_{random})$
How well did the forecast "yes" events correspond to the observed "yes" events (accounting for hits that would be expected by chance
Range: -1/3 – 1, 0 indicates no skill, 1 is perfect score

Where
$YY_{random}=(YY+YN)*(YY+NY)/(YY + YN + NY + NN)$
It is the number of hits for random forecasts

Bias (Or frequency Bias):
Bias=(YY+YN)/(YY+NY)
How similar were the frequencies of Yes forecasts and Yes observations? **Range:** 0 to infinity.
Perfect score: 1
When Bias is greater than 1, the event is overforecast; less than 1, underforecast

Verification of WRF Simulations – Categorical Variables

Forecast	Observation		Total
	Yes	No	
Yes	Hits (YY)	False Alarm (YN)	YY+YN
No	Misses (NY)	Correct (NN)	NY+NN
total	YY+NY	YN+NN	T=YY+YN+NY+NN

Probability of Detection (Hit Rate):
POD=YY/(YY+NY) (hits/(hits+misses))

False Alarm Ratio:
FAR=YN/(YY+YN) (False Alarm/(Hits+False Alarm))

False Alarm Rate (Probability of False Detection):
PODF=YN/(YN + NN) (False Alarm/(False Alarm+Correct))

Recommendations on the Verification of WRF Simulations –Categorical Variables

Example: daily rain forecasts and observations over 1-year period

Forecast	Observation		Total
	Yes	No	
Yes	82	38	**120**
No	23	222	**245**
total	105	260	**365**

(WCRP 2015)

Example:

Accuracy = (82+222)/365 = 0.83

Bias=(82+38)/(82+23)=1.14

POD=82/(82+23)=0.78

FAR=38/(82+38)=0.32

TS=82/(82+23+38)=0.57

ETS=(82-34)/(82+23+38-34)=0.44

Forecast	Observation		Total
	Yes	No	
Yes	82	38	**120**
No	23	222	**245**
total	105	260	**365**

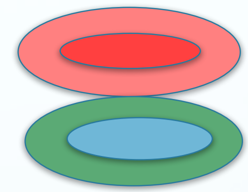
Verification of WRF Simulations – Categorical Variables

- Problems in traditional statistical measures -- scale-dependent
  - Warm season precipitation has significant small-scale variability
  - High-resolution models are becoming practical
  - Traditional scores are worse for detailed forecast --- double penalty
- Continuous, neighborhood method– a more sophisticated metrics to accurately quantify the realism of detailed forecast
  - Stage I: model forecast and observational fields are transformed into fraction grids
  - Stage II: Fractions are compared using the fractions skill score (FSS)

➔ The result is a measure of forecast skill against spatial scale for each selected threshold.

Verification of WRF Simulations – Categorical Variables

- Problems in traditional statistical measures -- scale-dependent
  - Warm season precipitation has significant small-scale variability
  - High-resolution models are becoming practical
  - Traditional scores are worse for detailed forecast --- double penalty

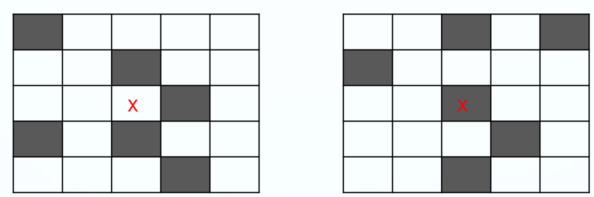


Verification of WRF Simulations – Categorical Variables

- A more sophisticated metrics to accurately quantify the realism of detailed forecast --- continuous, neighborhood method
  - Stage I: model forecast and observational fields are transformed into fraction grids
  - Stage II: Fractions are compared using the fractions skill score (FSS)

➔ The result is a measure of forecast skill against spatial scale for each selected threshold.

Recommendations on the Verification of WRF Simulations –Categorical Variables



Forecast Observation

A schematic example of fractional creation for a forecast and the corresponding observation. The precipitation exceeds the accumulation threshold in the shaded boxes.

At the central grid: NP_F=0, NP_O=1 ➔ FCST wrong
Over 3 x 3 grids: NP_F=3/9, NP_O=2/9 ➔ FCST over-forecast
Over 5 x 5 grids: NP_F=6/25, NP_O=6/25 ➔ FCST correct

Recommendations on the Verification of WRF Simulations –Categorical Variables

- Fraction of occurrences within a sample area:

$$FBS = \frac{1}{N_v}\sum_{i=1}^{N_v}[NP_{F(i)} - NP_{O(i)}]^2$$ (Fraction Brier Score)

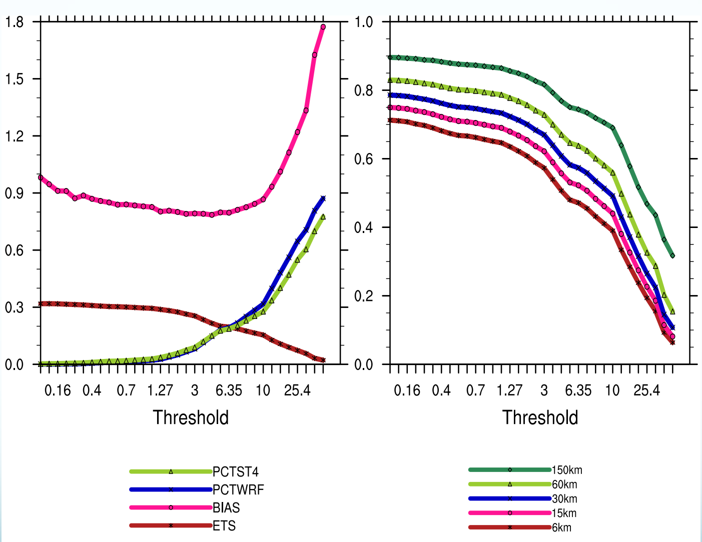
$$FBS_{worst} = \frac{1}{N_v}\left[\sum_{i=1}^{N_v} NP^2_{F(i)} + \sum_{i=1}^{N_v} NP^2_{O(i)}\right]$$ (The Worst FBS: no overlap of nonzero fractions)

$$FSS = 1 - \frac{FBS}{FBS_{worst}}$$ (Fractions Skill Score)

$NP_{F(i)}$ and $NP_{O(i)}$ are the neighborhood probabilities at the ith grid box in the model forecast and observed fraction fields, respectively. N is the number of grids in the verification area.

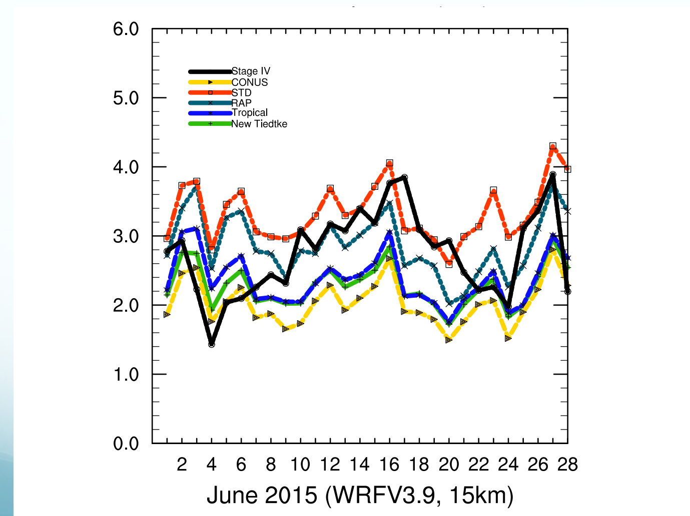
Verification of WRF Simulations –Categorical Variables (continue)

- FBS is negatively oriented
  - 0: perfect performance
  - Large FBS: poor correspondence between FCST and OBS
  - FBSworst: no overlap of nonzero fractions
  - FBS strongly depends on the frequency of the event
- FSS is defined to compare the FBS to the low-accuracy reference forecast (FBSworst)
  - FSS range (0,1): 1 for perfect forecast and 0 indicates no skill
  - As the number of grid boxes increases, FSS improves



WRFV3.9, 3km Test Cases in Summer 2016

Domain-average daily precipitation



WRF Software: Code and Parallel Computing

Dave Gill

WRF Software:
Code and Parallel Computing

John Michalakes, WRF Software Architect

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

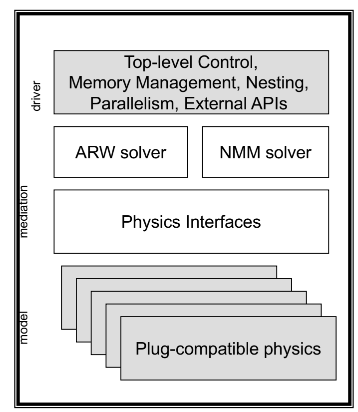
Introduction – WRF Software Characteristics

- 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

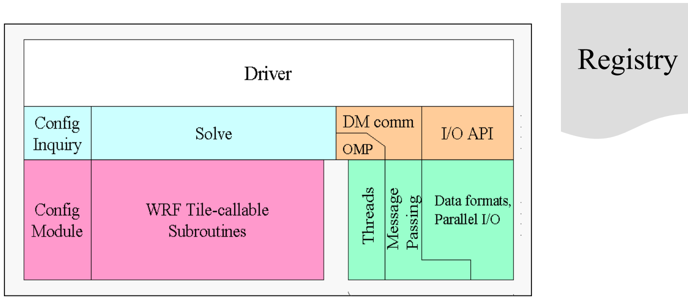
Introduction - WRF Software Framework Overview

- 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/

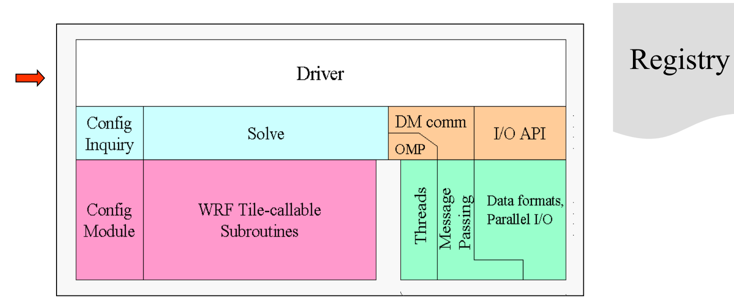


WRF Software Architecture



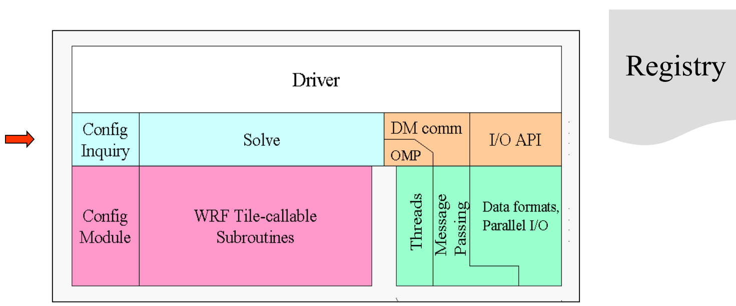
- Hierarchical software architecture
  - Insulate scientists' code from parallelism and other architecture/implementation-specific details
  - Well-defined interfaces between layers, and external packages for communications, I/O, and model coupling facilitates code reuse and exploiting of community infrastructure, e.g. ESMF.

WRF Software Architecture



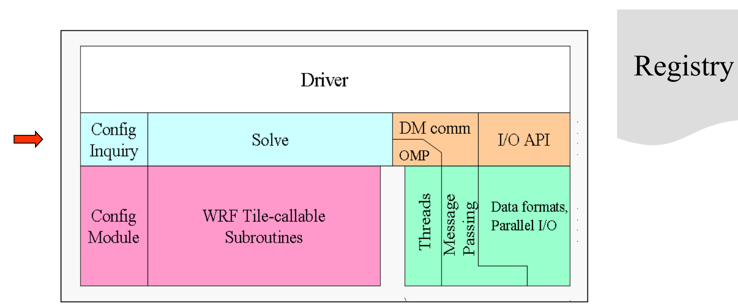
- Driver Layer
  - **Domains**: Allocates, stores, decomposes, represents abstractly as single data objects
  - **Time loop**: top level, algorithms for integration over nest hierarchy

WRF Software Architecture



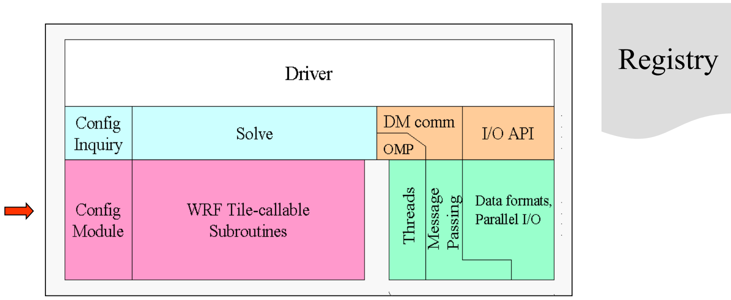
- Mediation Layer
  - Solve routine, takes a domain object and advances it one time step
  - Nest forcing, interpolation, and feedback routines

WRF Software Architecture



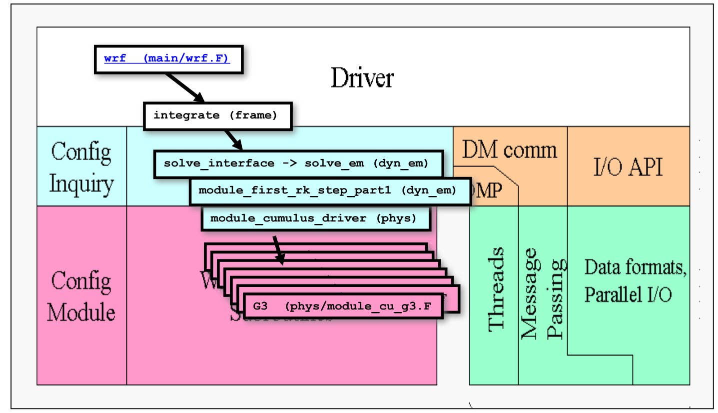
- Mediation Layer
  - The sequence of calls for doing a time-step for one domain is known in Solve routine
  - Dereferences fields in calls to physics drivers and dynamics code
  - Calls to message-passing are contained here as part of Solve routine

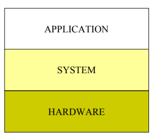
WRF Software Architecture



- 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

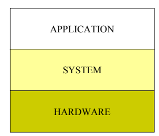
Call Structure Superimposed on Architecture





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

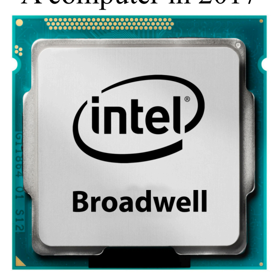
6-way superscalar

36-bit floating point precision

~144 Kbytes

~50,000 flop/s
48hr 12km WRF CONUS in 600 years

A computer in 2017



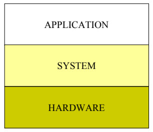
Dual core, 2.3 GHz chip

16 Flops/clock

64-bit floating point precision

20 MB L3

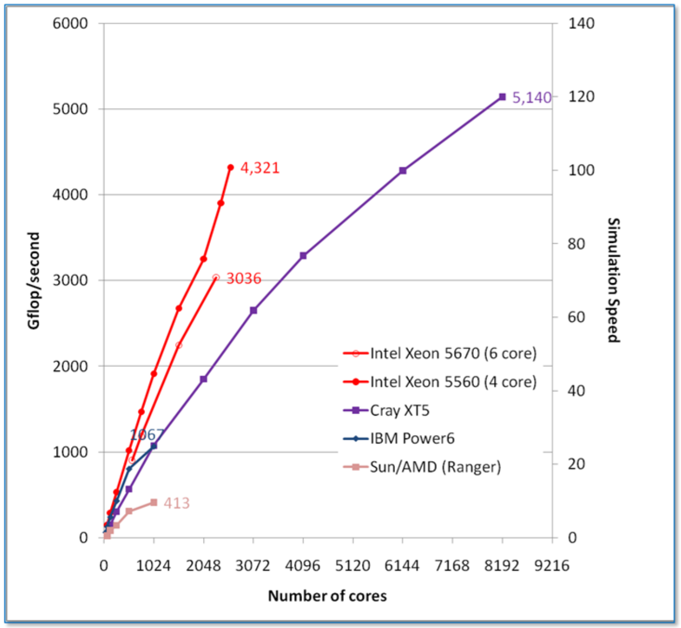
~5,000,000,000 flop/s
48 12km WRF CONUS in 26 Hours



…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

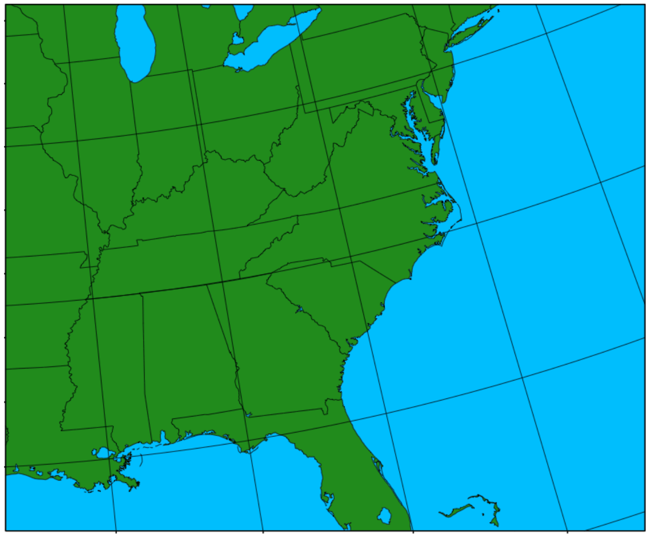
~1,000,000,000,000 flop/s ~2500 procs
48-h, 12-km WRF CONUS in under 15 minutes



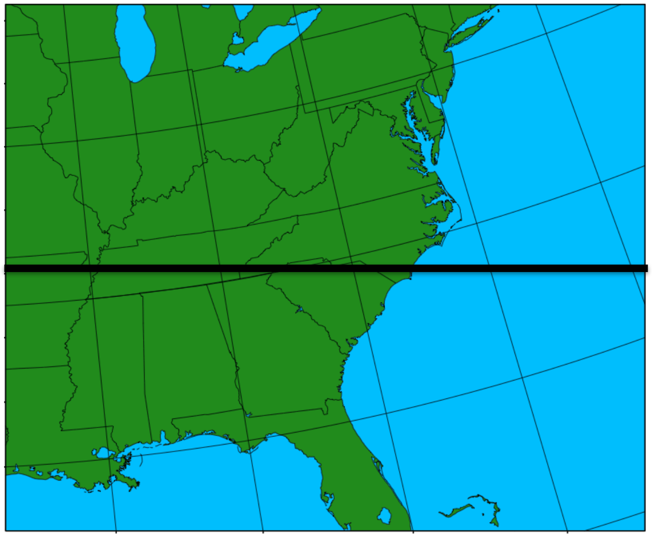
WRF Domain Decomposition

- The WRF model decomposes domains horizontally
- For *n* MPI tasks, the two nearest factors (*n= k * m*)are selected; the larger is used to decompose the y-direction, the smaller is used to decomposed the x-direction

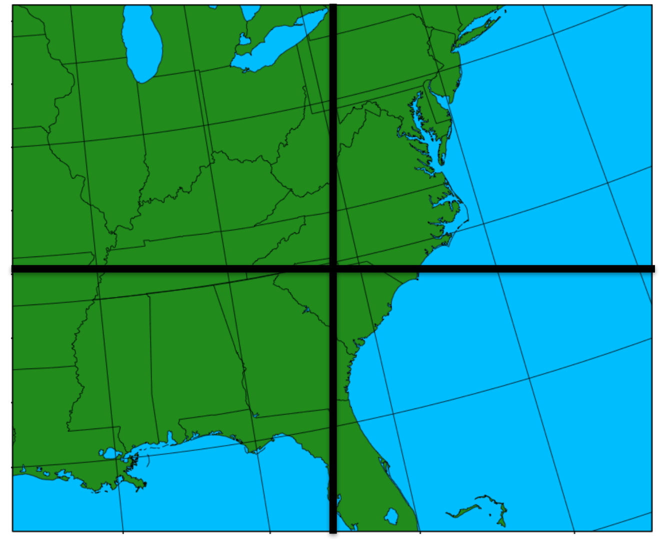
January 2000 Benchmark – 1 task: 74x61



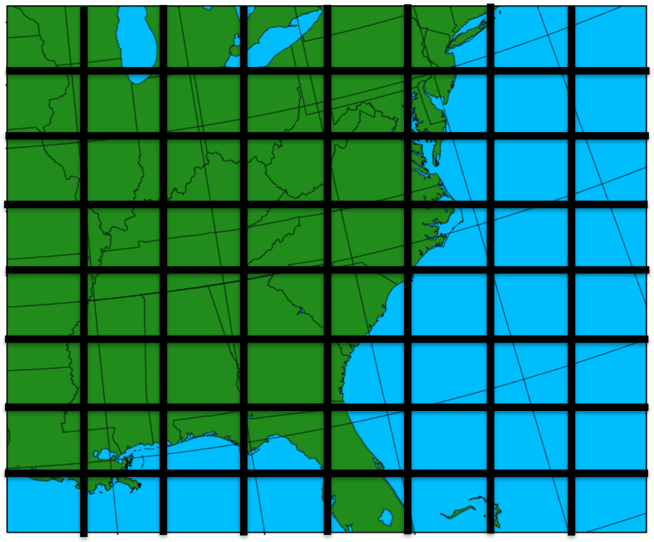
January 2000 Benchmark – 2 tasks: 74x31



January 2000 Benchmark – 4 tasks: 37x31



January 2000 Benchmark – 64 tasks: 10x8

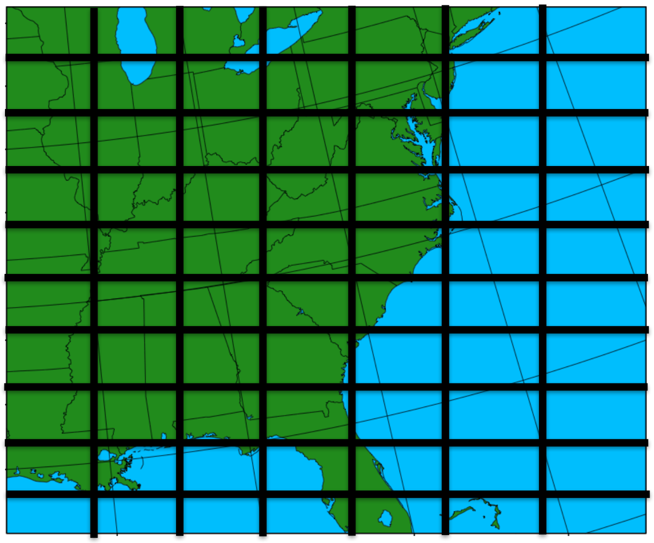


WRF Domain Decomposition

- Users may choose a preferred decomposition (nproc_x, nproc_y)

```
&domains
 nproc_x                =  7
 nproc_y                = 10
```

January 2000 Benchmark – 70 tasks



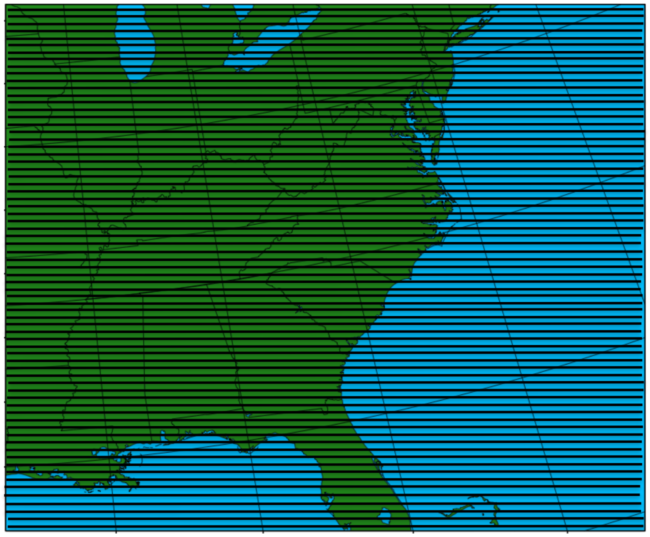
WRF Domain Decomposition

- Users may choose a preferred decomposition (nproc_x, nproc_y)

```
&domains
 nproc_x              =  7
 nproc_y              = 10
```

- Prime numbers and composites with large prime factors are usually to be avoided
- The behavior of 70 vs 71 is quite different

January 2000 Benchmark – 71 tasks



WRF Domain Decomposition

- As you **increase the number of total MPI tasks**, you **reduce the amount of work** inside of each MPI task
- The amount of time to process **communication** between MPI tasks tends to be ***at best* constant**
- As more MPI tasks are involved, more contention for hardware resources due to communication is likely increase
- As the computation time gets smaller compared to the communications time, **parallel efficiency suffers**

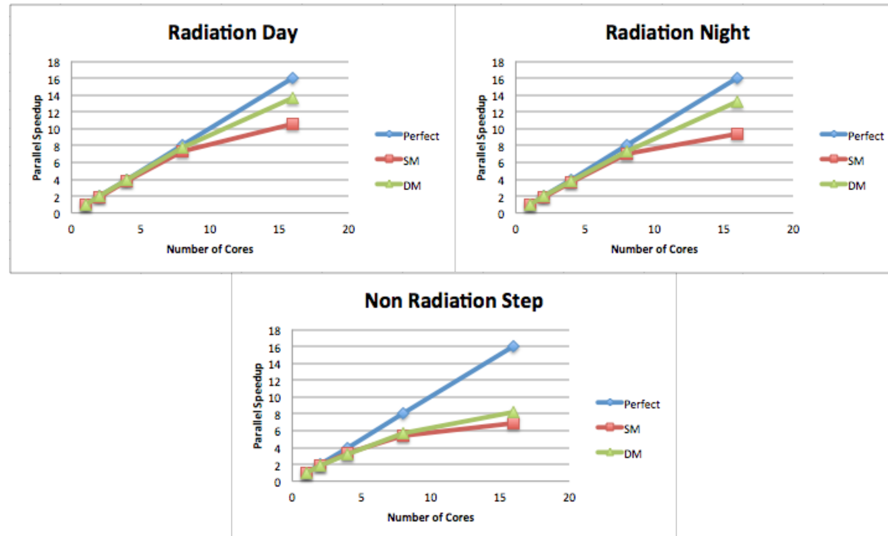
January 2000 Benchmark

- 74x61 grid cells, 24 hour forecast, 3 minute time step
- IO excluded
- Timing partitioned
  - Local DAY Radiation step (17 time periods)
  - Local NIGHT Radiation step (24 time periods)
  - Not a Radiation step (432 time periods)

Decomposed domain sizes   proc count: I-dim x J-dim

1: 74x61   2: 74x31   4: 37x31   8: 37x16   16: 19x16

January 2000 Benchmark



January 2000 Benchmark

Radiation Day

Core Count		SM Efficiency	DM Efficiency
1	74x61	100	100
2	74x31	97	100
4	37x31	93	97
8	37x16	91	96
16	19x16	65	85

Avg 5.76 s

Std 0.019 s

n = 17

Radiation Night

Core Count		SM Efficiency	DM Efficiency
1	74x61	100	100
2	74x31	97	100
4	37x31	93	95
8	37x16	88	92
16	19x16	59	83

Avg 2.16 s

Std 0.005 s

n = 24

Not Radiation Timestep

Core Count		SM Efficiency	DM Efficiency
1	74x61	100	100
2	74x31	94	97
4	37x31	84	80
8	37x16	68	71
16	19x16	43	52

Avg 0.39 s

Std 0.012 s

n = 432

January 2000 Benchmark

- WRF timing estimates may be obtained from the model print-out

Serial – 1 core, Day radiation step

```
Timing for main on domain   1:    5.77810 elapsed seconds
```

OpenMP – 8 cores, Day radiation step

```
Timing for main on domain   1:    0.83044 elapsed seconds
```

MPI – 16 cores, Day radiation step

```
Timing for main on domain   1:    0.39633 elapsed seconds
```

- Get enough time steps to include "day-time" radiation, and to have the microphysics "active" for better estimates

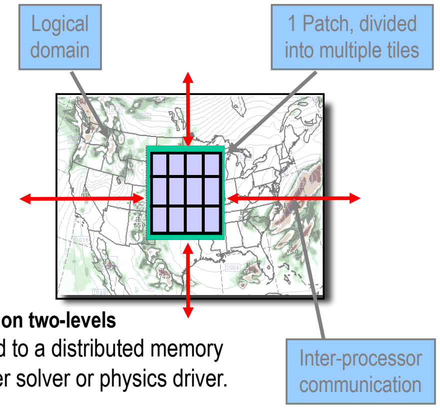
APPLICATION
SYSTEM
HARDWARE

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

APPLICATION
SYSTEM
HARDWARE

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.

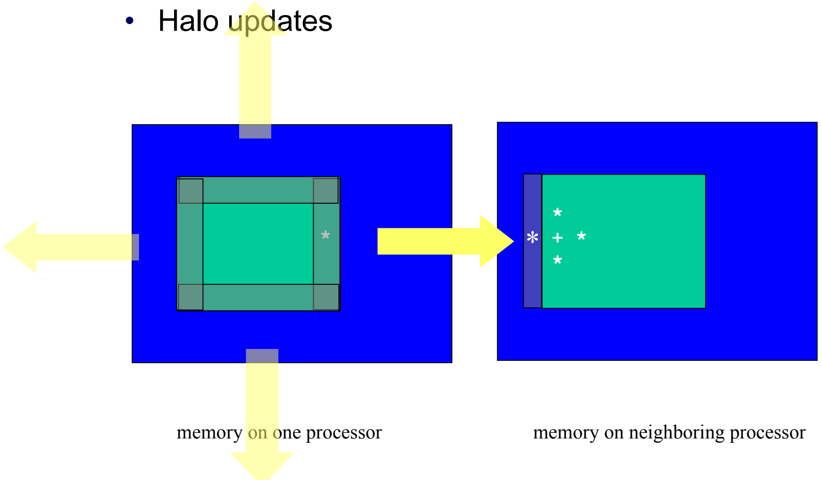
Distributed Memory Communications

```
                              (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
 . . .
```

APPLICATION
SYSTEM
HARDWARE

Distributed Memory MPI Communications

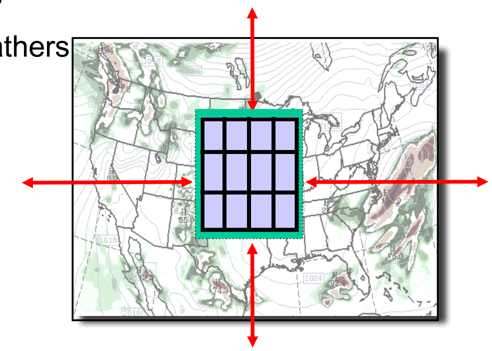
- Halo updates



APPLICATION
SYSTEM
HARDWARE

Distributed Memory (MPI) Communications

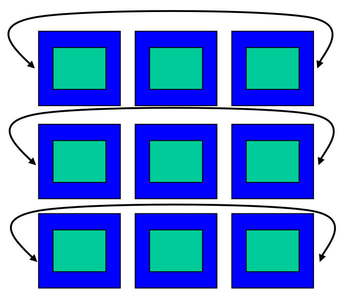
- Halo updates
- Periodic boundary updates
- Parallel transposes
- Nesting scatters/gathers



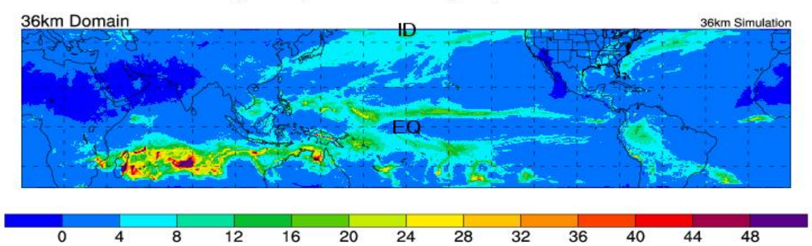
APPLICATION
SYSTEM
HARDWARE

Distributed Memory (MPI) Communications

- Halo updates
- Periodic boundary updates
- Parallel transposes
- Nesting scatters/gathers



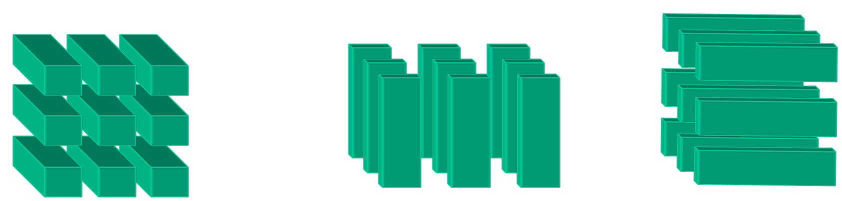
Average Daily Total rainfall (mm) - March 1997



APPLICATION
SYSTEM
HARDWARE

Distributed Memory (MPI) Communications

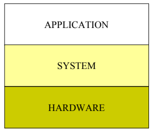
- Halo updates
- Periodic boundary updates
- Parallel transposes
- Nesting scatters/gathers



all y on patch

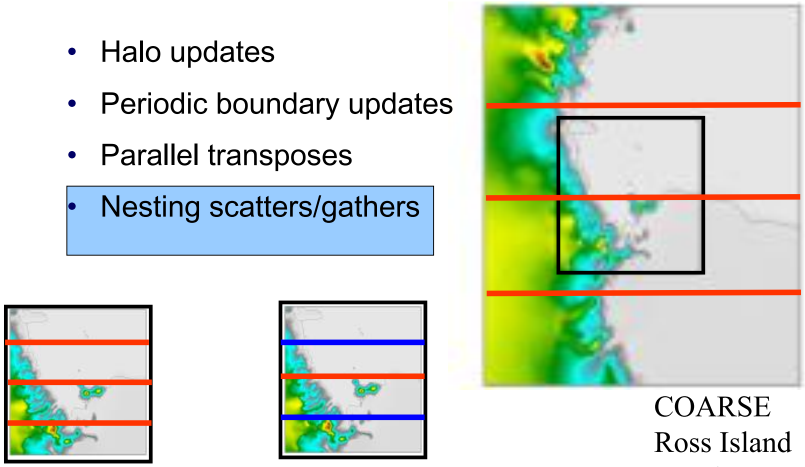
all z on patch

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 ●(red)
MEDIATION ●(blue)
MODEL ●(yellow)

```
Makefile
README
README_test_cases
clean        }
compile      }  build scripts
configure    }
Registry/       CASE input files
arch/           machine build rules
● dyn_em/      }
● dyn_nmm/     }
  external/    }
● frame/       }
  inc/         }  source code directories
● 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

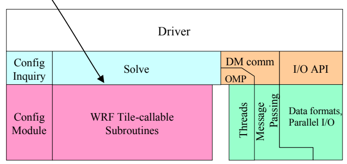
  WRFV3/dyn_em/module_small_step_em.F

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

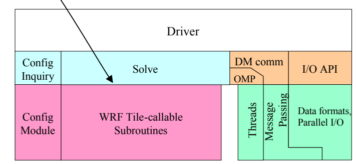
Restrictions on Model Layer subroutines:

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

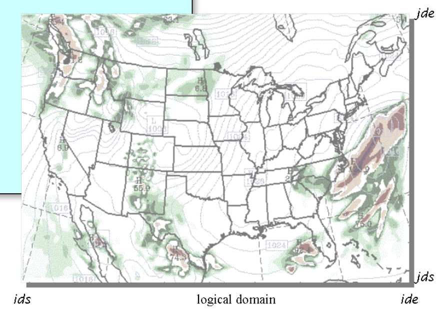
template for model layer subroutine

```
. . .
! 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
```

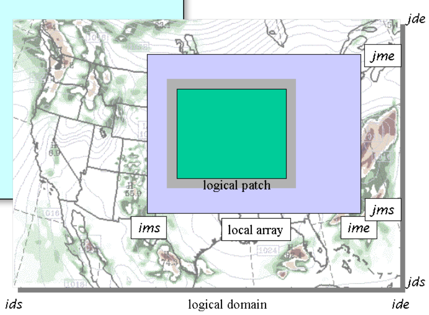
- Domain dimensions
  - Size of logical domain
  - Used for bdy tests, etc.



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
```

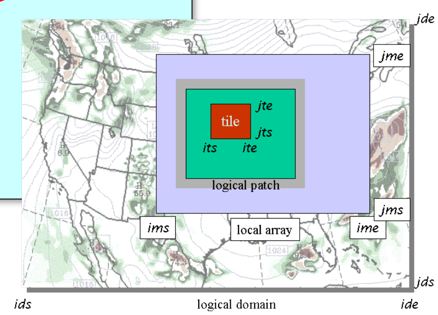
- Domain dimensions
  - Size of logical domain
  - Used for bdy tests, etc.
- Memory dimensions
  - Used to dimension dummy arguments
  - Do not use for local arrays



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

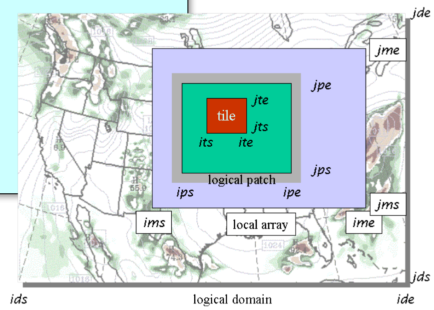


template for model layer subroutine

```
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 (similar to Fortran units): 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"
  – Non-chemistry: use history streams 13-22, 24
  – Chemistry: use history streams 20, 21, 22, 24

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

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

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