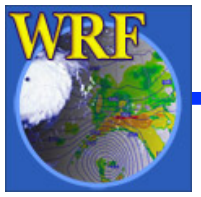
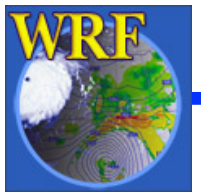
Pre & Post-Processing Tools

Cindy Bruyère

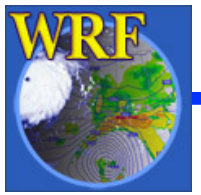


Topics

- **Pre-Processing**
  - OBSGRID
- **Post-Processing**
  - NCL graphical tool
  - RIP4 converter and graphical tool
  - ARWpost converter



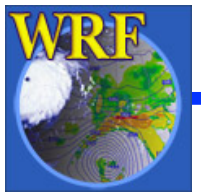
Objective Analysis (OBSGRID)



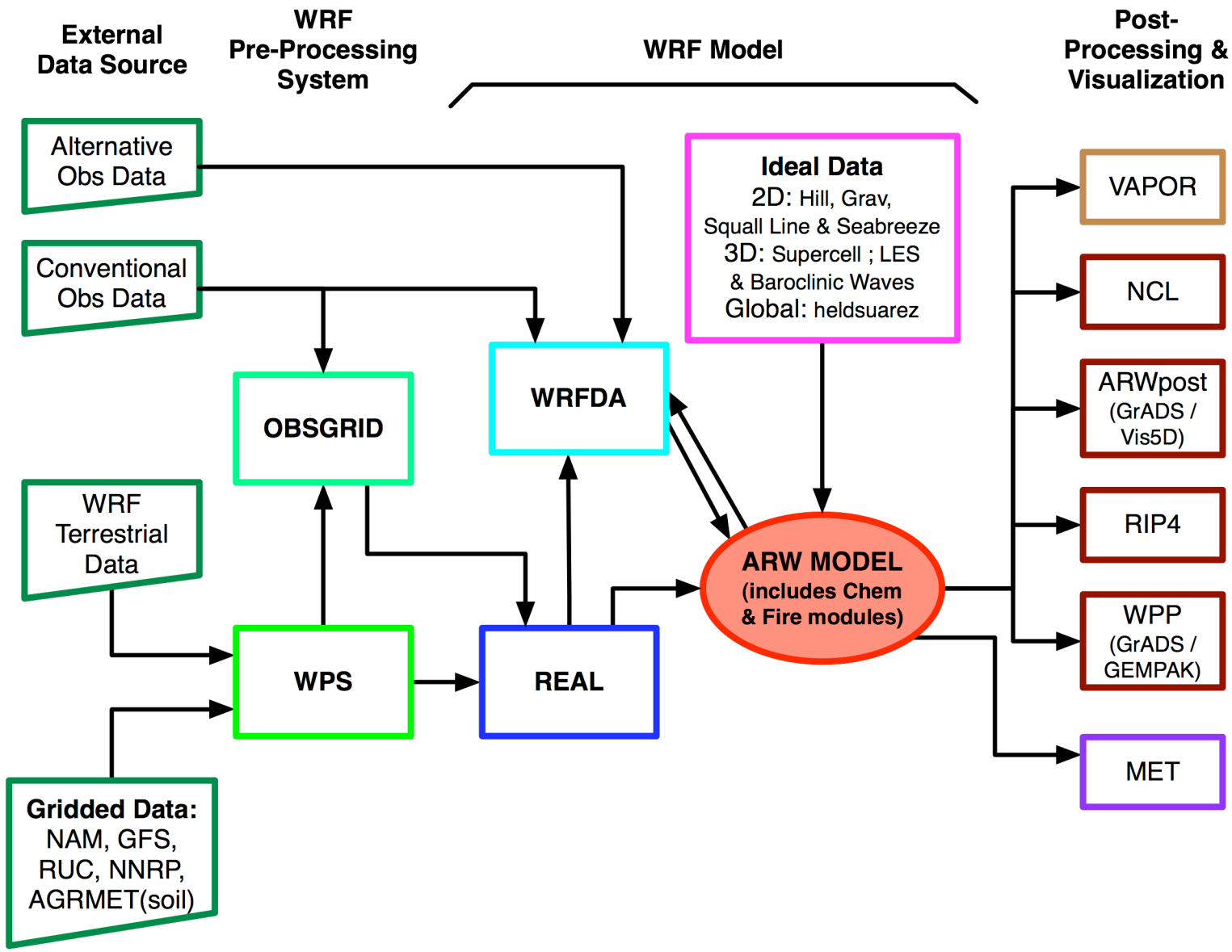
Objective Analysis

- **To improve a *first-guess* gridded analysis by incorporating additional observational information**
  - Traditionally, this first-guess analysis comes from low-resolution global analysis and forecast grids
  - These days, higher-resolution, regional scale analyses are more readily available
    - *These high-resolution analyses mean that in many cases, the objective analysis step is not essential when running WRF*

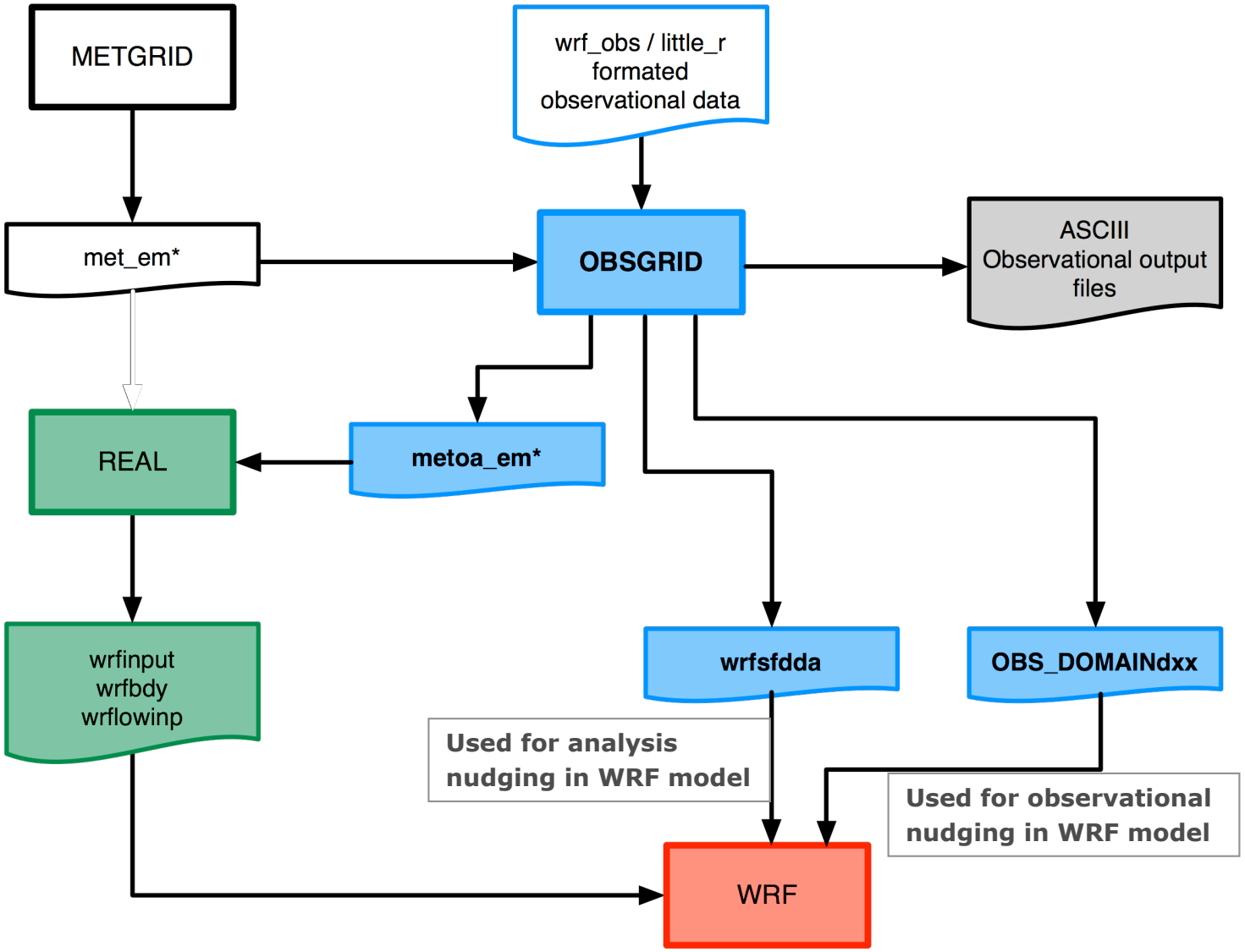
- **Methods**
  - **Traditional methods**
    - *Direct observations of T, U, V, RH at surface and on pressure levels (conventional observations)*
  - **Variational Analysis**
    - *Direct and indirect observations on model levels (conventional + alternative observations) - WRFDA*



WRF Modeling System Flow Chart

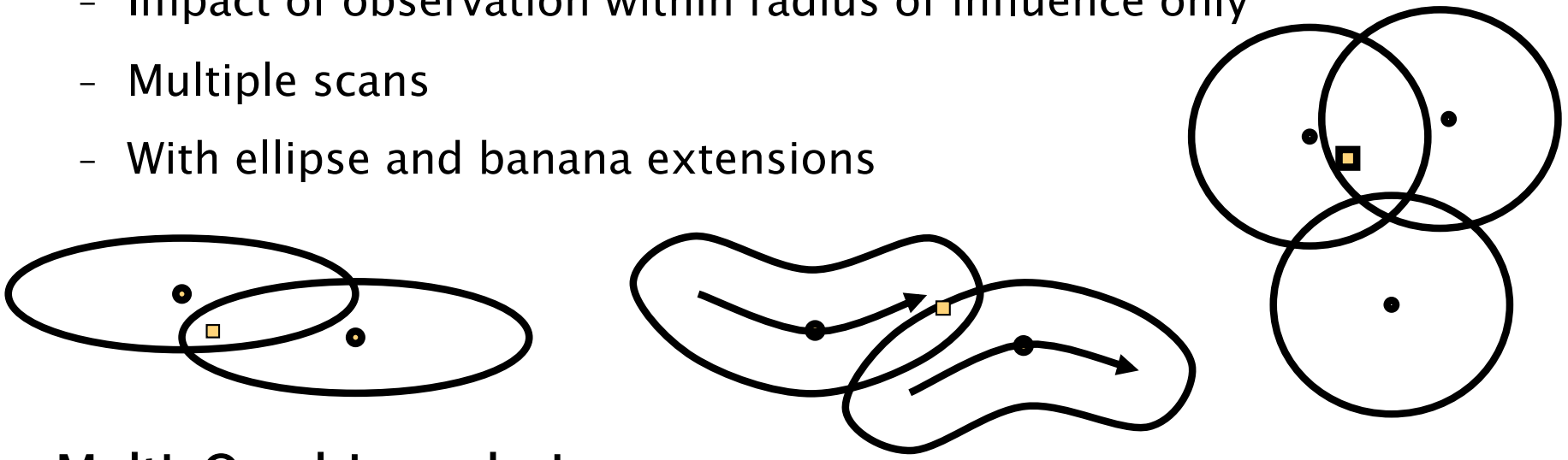


Program Flow



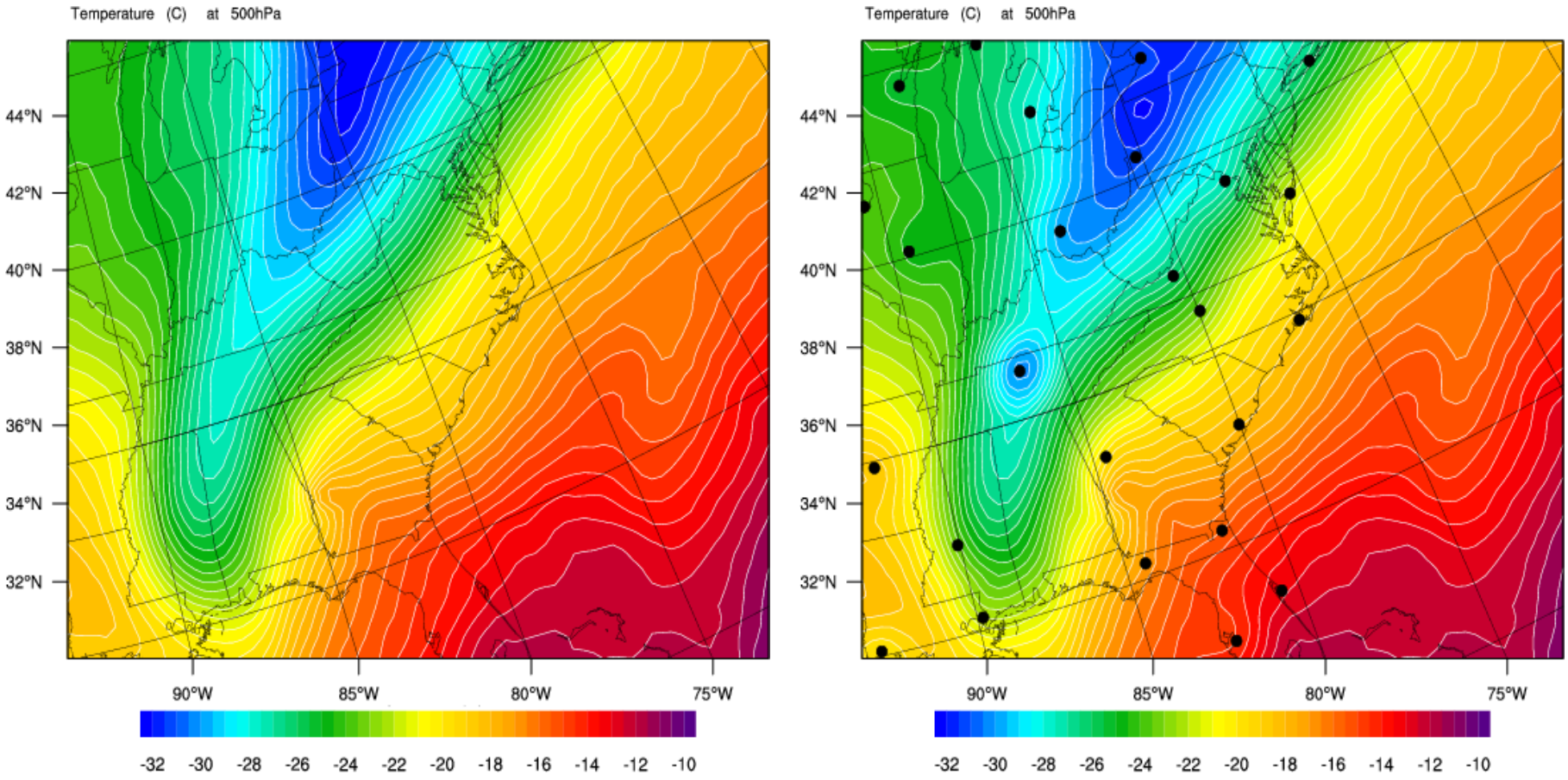
OA Techniques in OBSGRID

- **Cressman-based analysis**
  - Impact of observation within radius of influence only
  - Multiple scans
  - With ellipse and banana extensions

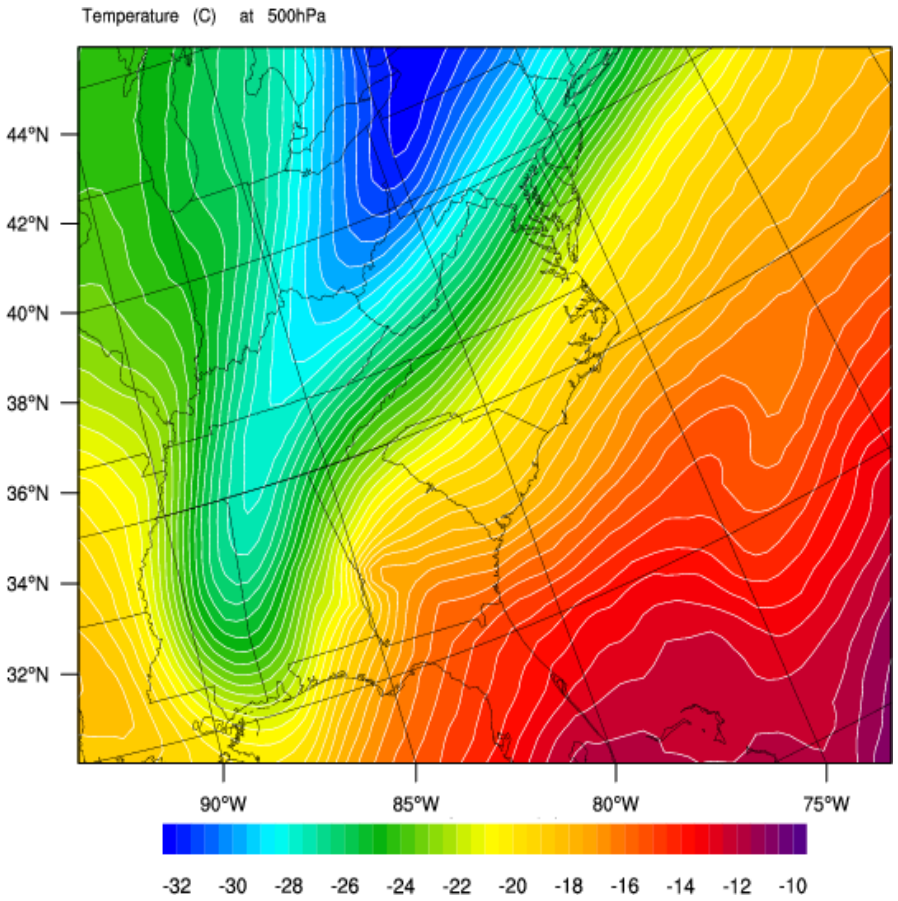


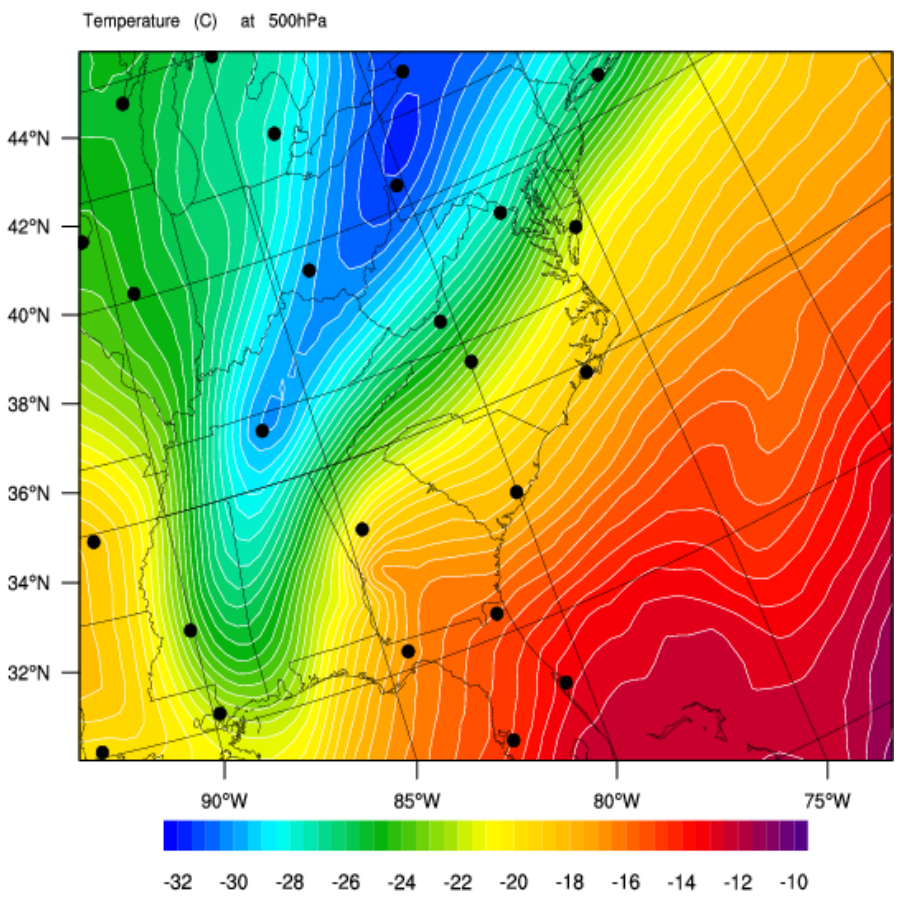
- **Multi-Quadric analysis**
  - Impact of observations are over the entire model domain
  - Scheme is sensitive to the data density distribution
  - Makes use of hyperboloid radial basis functions to interpolate observation corrections onto model grid

Cressman Scheme



Multi-Quadric Scheme





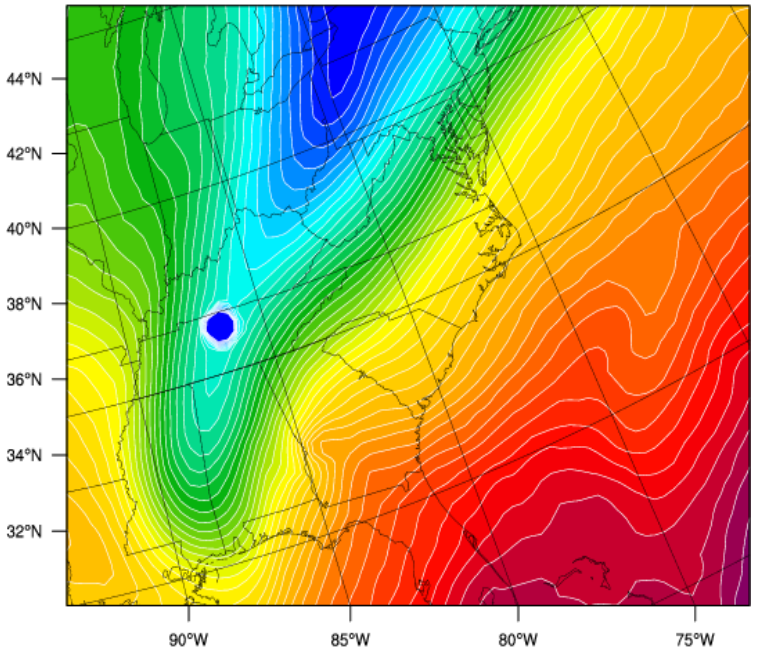
Quality Control for Observations

- **A critically important step**

  Bad observations = Bad objective analysis

- **Even a single bad observation can ruin initial conditions**

- **Default namelist settings should in most cases prevent this from happening**

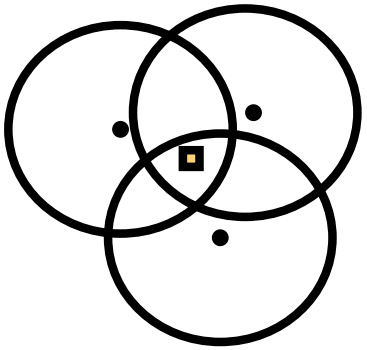


- **Tests performed in OBSGRID**
  - Tests on individual reports
    - *Only gross checks, no comparisons to other soundings or to first-guess field*
  - ERRMAX test
  - Buddy test

Tests performed in OBSGRID

- **ERRMAX test**
  - Observations are compared to first-guess field
  - If the difference between the observation and the first-guess exceeds a threshold, the observation is discarded
  - Threshold varies depending on field, level, time of day
  - Works well with good first-guess field

- **BUDDY test**
  - Observations are compared to the first guess and to nearby observations
  - If an observation deviates from the first guess in a manner inconsistent with the deviations of surrounding stations from the first guess, then that observation is discarded
  - Works well in regions of good data density



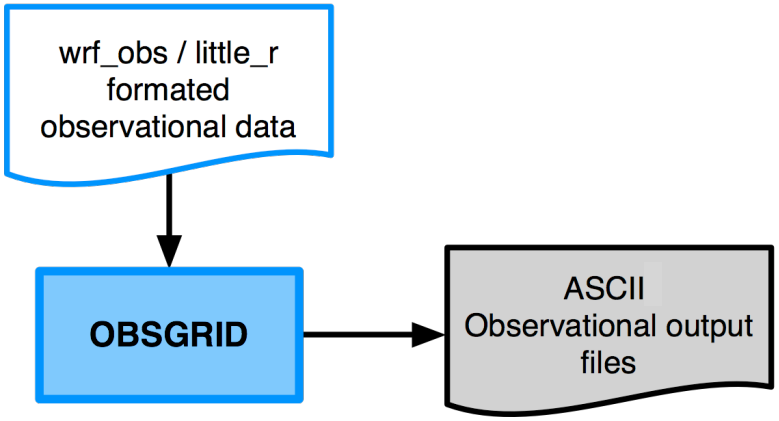
Observations

UG: 7-13

- **ASCII text files *(wrf_obs / little_r format)***
  - One entry per observation *(sfc or upper-air)*
    - Header ; Data ; End

- **Each time period is stored in a separate file**
  - Naming convention: *obs_filename*:<YYYY-MM-DD_HH>

- **OBSGRID combines reports, removes duplicates, interpolates to analysis levels**



Observations Format

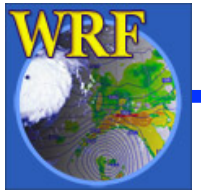
header

station_latitude, station_longitude, station_id,
station_name, type_of_observation, observation_source,
station_elevation, num_valid_fields, num_errors,
num_warnings, **report_number,** number_of_report_duplicates,
is_this_a_sounding, is_this_a_bogus_observation,
discard_the_report, time_of_report, julian_day,
report_time, slp, slp_qc, ref_pres, rpres_qc, ground_t,
t_qc, sst, sst_qc, psfc, psfc_qc, precip, precip_qc,
t_max, t_max_qc, t_min, t_min_qc, t_min_night,
t_min_night_qc, p_tend03, tend03_qc, p_tend24, tend24_qc,
cloud_cvr, cloud_qc, ceiling, ceiling_qc

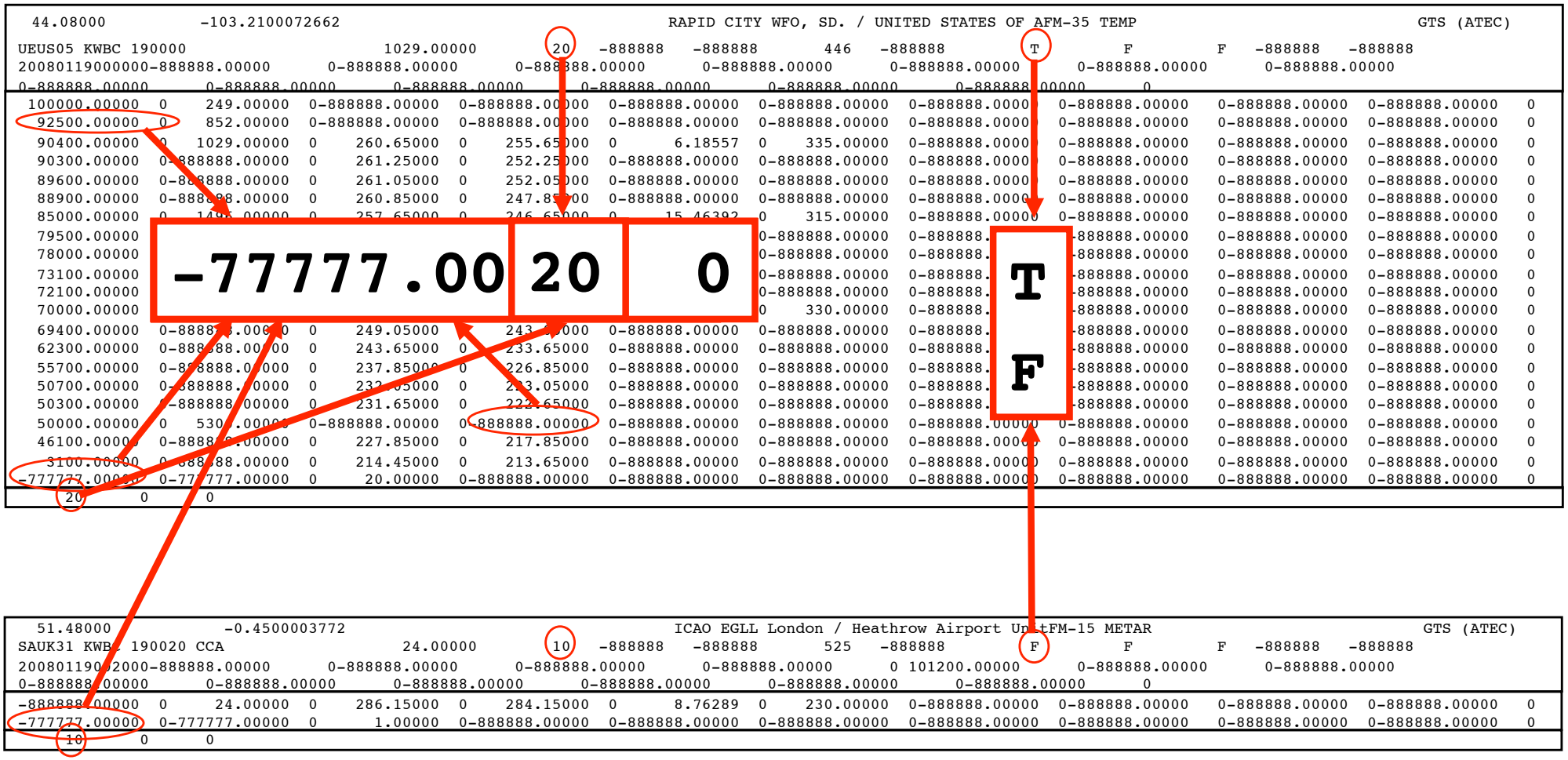
end data

pressure, p_qc, height, z_qc, temperature, t_qc,
dew_point, td_qc, speed, spd_qc, direction, dir_qc, u,
u_qc, v, v_qc, rh, rh_qc, thickness, tn_qc

num_valid_fields, num_errors, num_warnings



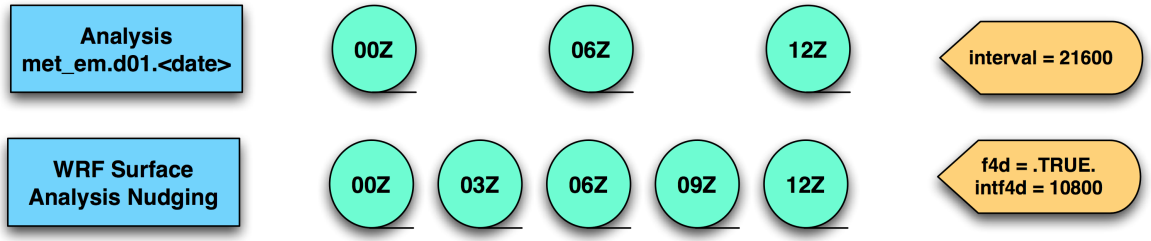
Observations Format



Namelist

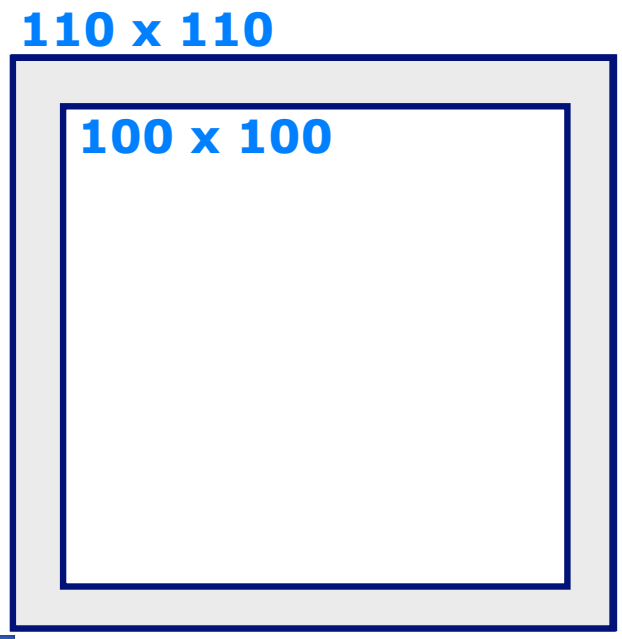
UG: 7-15

obs_filename	Root file name *(may include directory information)* of the observational files. All input files must have the format obs_filename:<YYYY-MM-DD_HH>. One file per time period Also used when doing surface fdda *(F4D=.TRUE.)*
interval	Interval between analysis times (*sec*) *i.e., interval between met_em.d0x.<date> files*
f4d	Turn on / off the creation of the surface analysis file *(wrfsfdda_d0x)*
intf4d	Time interval (*sec*) for surface analyses *(typically smaller than interval between analysis)*



Namelist

trim_domain	Set to .TRUE. if this domain must be cut down on output
trim_value	Value by which the domain will be cut down in each direction



- **Why do this:**
  This allows observations just outside the desired grid box of 100x100 grid points to be included in the OA

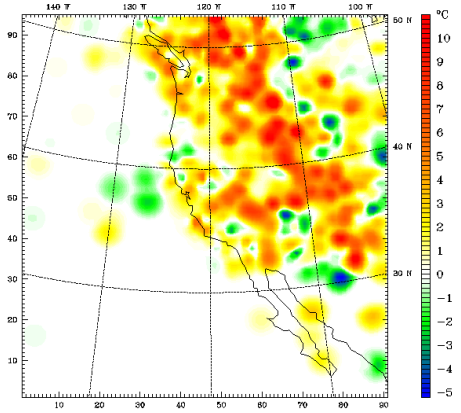
- ***geogrid* and *metgrid* run with a domain size of 110x110**

- **When running *obsgrid* set**
  trim_domain = .TRUE.
  trim_value = 5

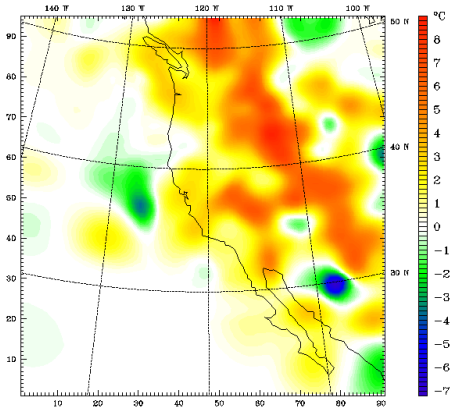
Namelist

- OA_type = "Cressman"
  - *All OA for surface and upper-air will be Cressman*
- radius_influence
  - *0*
    Code will automatically calculate number of grids to use of OA
    Will use 4 scans
  - *-5*
    Code will automatically calculate number of grids to use of OA
    Will use set number of scans (5 in the example)
  - *5, 4, 3, 2*
    Use 4 scans with grid points as set

RIN=5,4,3,2

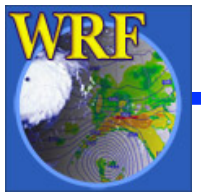


RIN=0



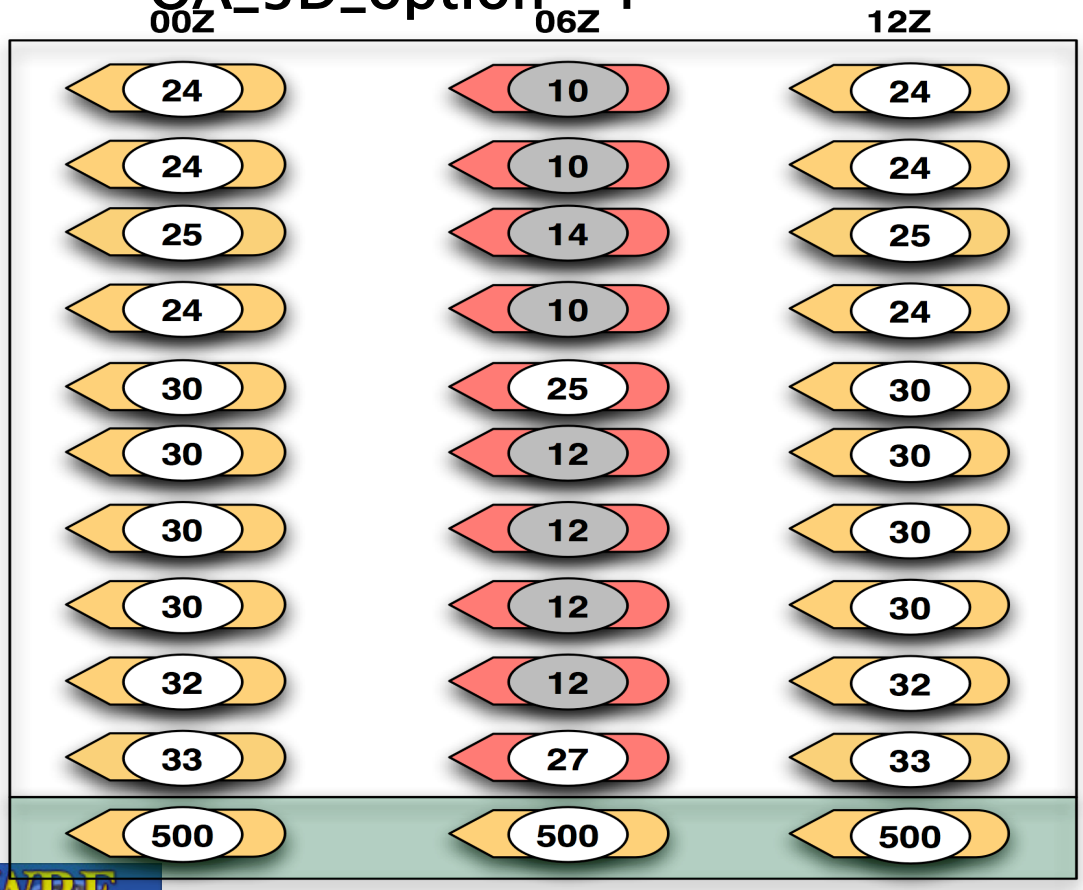
Namelist

- OA_type = "MQD"
  - *Will **attempt** to use MQD for all OA*
  - *Set radius_influence in case code has to switch to Cressman (recommend 0)*
- OA_max_switch (T/F) ; MQD_maximum_num_obs
  - *T ; Switch to Cressman if too many observations*
  - *F ; No OA if too many observations*
  - *Only needed if the code takes to long to execute due to large number of observations*
- OA_min_switch (T/F) ; MQD_minimum_num_obs
  - *T ; Switch to Cressman if too few observations*
  - *F ; No OA if too few observations*



Namelist

- OA_type = "MQD" ;
  OA_min_switch = T ; MQD_minimum_num_obs = 24
  OA_3D_option =1



MQD_minimum_num_obs = 24

Do not make this to low in order to force MQD ***(i.e., 9 in this case)***

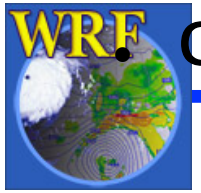
OA_3D_option = 1

All upper-air, at a given time
Especially good for f4d runs

MQD ; Cressman

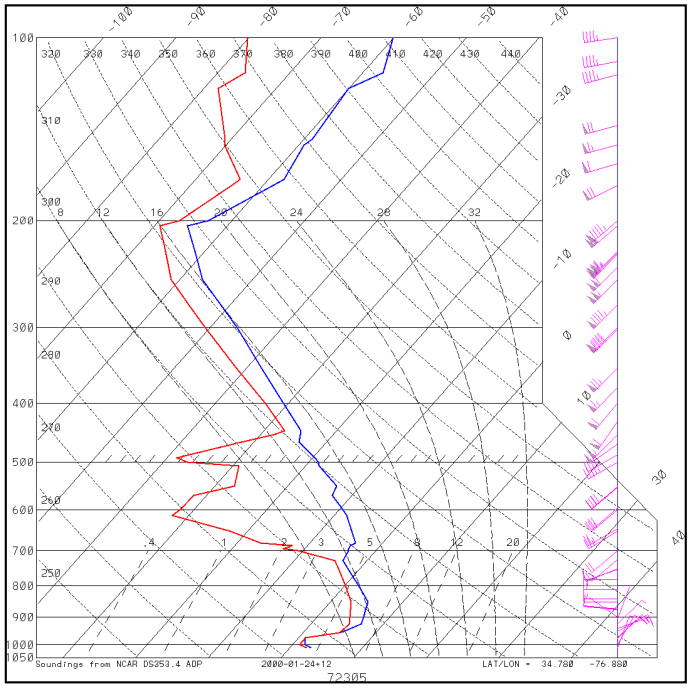
How to run OBSGRID

- Get the source code
- Compile *(./configure & ./compile)*
- Prepare observations files
  - Users need to generate this file *(some sample programs are available)*
- Edit the namelist
- Run the program
- Check your output

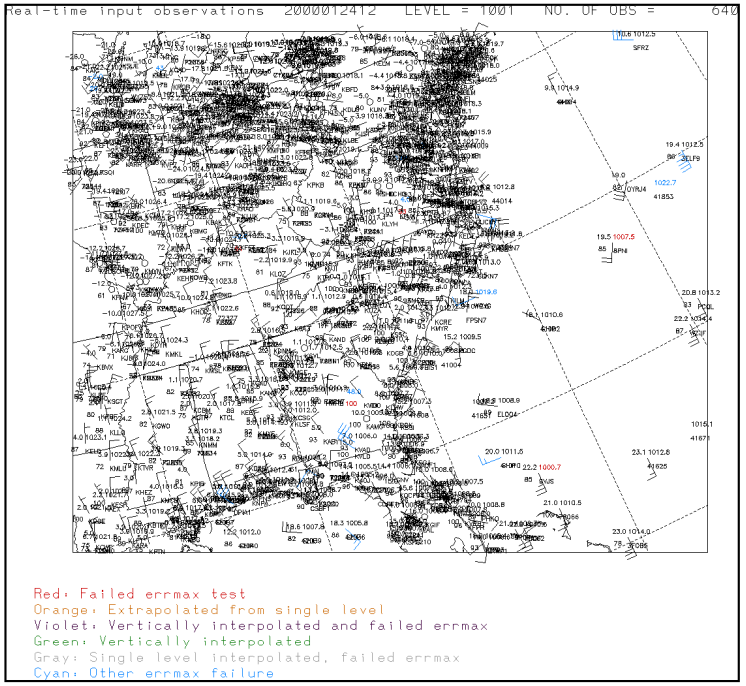


Utilities

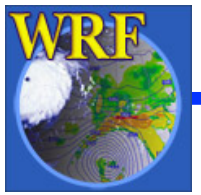
- **plot_sound**
  - Plot sounding from the 'raw' or 'used' wrf_obs / little_r output files
  - Namelist control: **&record1, &record2, &plot_sounding**



- **plot_level**
  - Plot data used on all levels
  - Namelist control: **&record1, &record2**



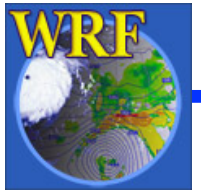
Post-processing Tools



Graphical Packages

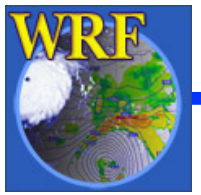
- **NCL** — UG: 9-2
  - Graphical package
- **ARWpost** — UG: 9-28
  - Converter (GrADS & vis5d)
- **RIP4** — UG: 9-19
  - Converter and interface to graphical package NCAR Graphics
- **WPP** — UG: 9-35
  - Converter (GrADS & GEMPAK)
- **VAPOR** — UG: 9-50
  - Converter and graphical package
  - *Support: VAPOR*
- **IDV** — unidata.ucar.edu
  - GRIB (from WPP)
  - GEMPAK (from wrf2gem)
  - vis5d (from ARWpost)
  - CF complaint data (from wrf_to_cf)
  - *Support: unidata*
- **GEMPAK**
  - Data from wrf2gem or WPP
  - *Support: unidata*

MatLab / IDL / R / ferret



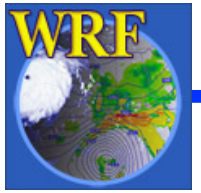
Graphical Packages

	NCL	RIP4	ARWpost (GrADS)
Directly ingest WRF data	***Y***	**N** *converter*	**N /** ***(Y)*** *converter*
Intermediate files	***N***	lots	large file
WPS DATA	***Y***	***Y***	***Y***
wrfinput data	***Y***	***Y***	***Y***
Idealized data files	***Y***	***Y***	***Y***
Input format	*netCDF*	*netCDF*	*netCDF / GRIB1*
Vertical Output Coordinate	eta pressure height	eta pressure height	eta pressure height
Software required ***(All binaries are free)***	*NCL*	*NCARG*	*GrADS*
Diagnostics	*some*	*> 100*	*some*



NCL

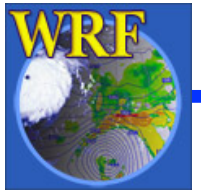
- NCAR Command Language
- http://www.ncl.ucar.edu
- Read WRF-ARW data directly
  - WRF specific libraries and scripts are distributed with NCL libraries
- Download
  - http://www.ncl.ucar.edu/Download
  - Fill out short registration form *(there is a short waiting period)*
  - Read and agree to OSI-based license
  - Get version 5.1.1 or later
- NCARG_ROOT environment variable
  - setenv NCARG_ROOT /usr/local/ncl



Home Directory → ~/.hluresfile ← Very Important

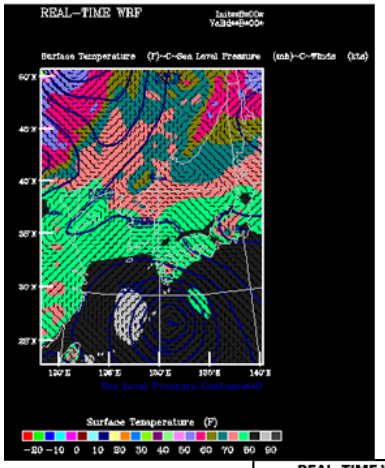
- Required by NCL libraries
- Must be in your "~/" directory *(home directory)*
- Control
  - color table ; font
  - white/black background
  - size of plot
  - control characters

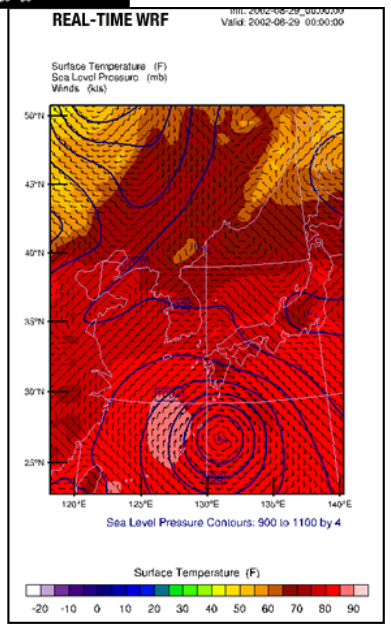
- http://www.ncl.ucar.edu/Document/Graphics/hlures.shtml



~/.hluresfile

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



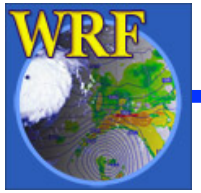



```
PLCHHQ - CHARACTER NUMBER 26 (.) IS NOT A LEGAL FUNCTION CODE
PLCHHQ - CHARACTER NUMBER 29 (t) IS NOT A LEGAL FUNCTION CODE
PLCHHQ - CHARACTER NUMBER 30 (o) IS NOT A LEGAL FUNCTION CODE
PLCHHQ - CHARACTER NUMBER 32 (.) IS NOT A LEGAL FUNCTION CODE
PLCHHQ - CHARACTER NUMBER 35 (b) IS NOT A LEGAL FUNCTION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

http://www.mmm.ucar.edu/wrf/OnLineTutorial/Graphics/NCL/.hluresfile

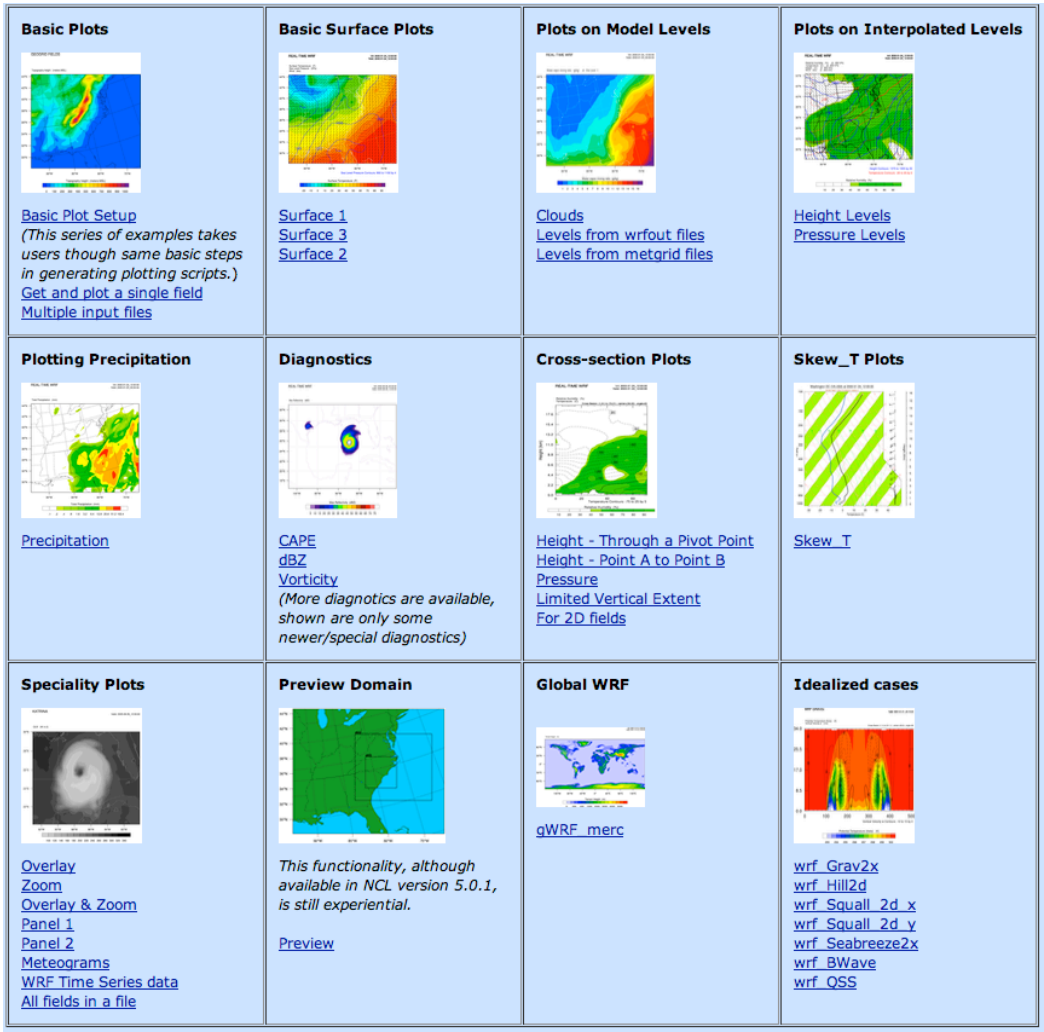
Generate Plots with NCL

- **Set NCARG_ROOT environment variable:**
  - *setenv NCARG_ROOT /usr/local/ncl*
- **Ensure you have an ~/.hluresfile file**
- **Create a script**
  - *wrf_real.ncl*
    (start with a sample script)
- **Run NCL script**
  - *ncl wrf_real.ncl*



Generate Plots: *A good start - OnLine Tutorial*

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



Basic Plots

Basic Plot Setup
(This series of examples takes users though same basic steps in generating plotting scripts.)
Get and plot a single field
Multiple input files

Basic Surface Plots

Surface 1
Surface 3
Surface 2

Plots on Model Levels

Clouds
Levels from wrfout files
Levels from metgrid files

Plots on Interpolated Levels

Height Levels
Pressure Levels

Plotting Precipitation

Precipitation

Diagnostics

CAPE
dBZ
Vorticity
(More diagnotics are available, shown are only some newer/special diagnostics)

Cross-section Plots

Height - Through a Pivot Point
Height - Point A to Point B
Pressure
Limited Vertical Extent
For 2D fields

Skew_T Plots

Skew_T

Speciality Plots

Overlay
Zoom
Overlay & Zoom
Panel 1
Panel 2
Meteograms
WRF Time Series data
All fields in a file

Preview Domain

This functionality, although available in NCL version 5.0.1, is still experiential.

Preview

Global WRF

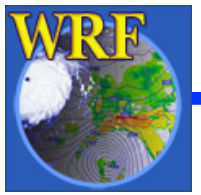
gWRF_merc

Idealized cases

wrf_Grav2x
wrf_Hill2d
wrf_Squall_2d_x
wrf_Squall_2d_y
wrf_Seabreeze2x
wrf_BWave
wrf_QSS

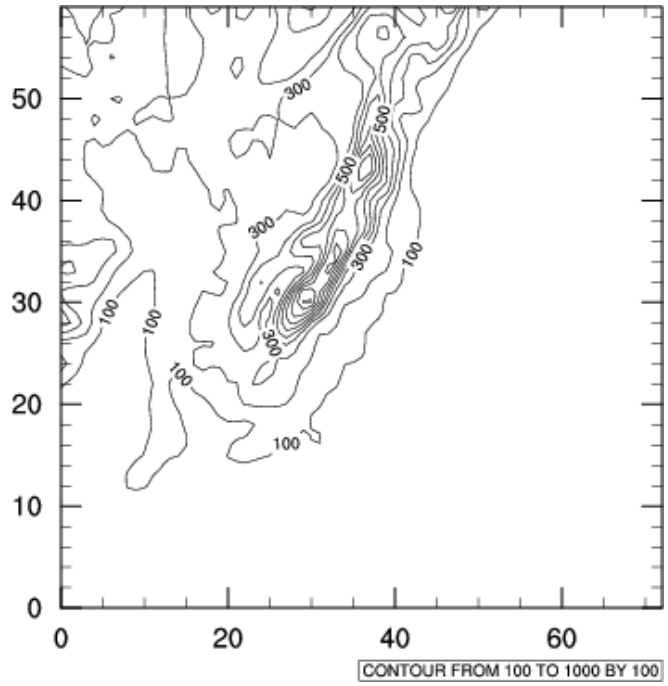
Creating a Plot : NCL script

```
load ncl library scripts

begin

  ; Open graphical output
  ; Open input file(s)

  ; Read variables

  ; Set up plot resources & Create plots
  ; Output graphics

end
```



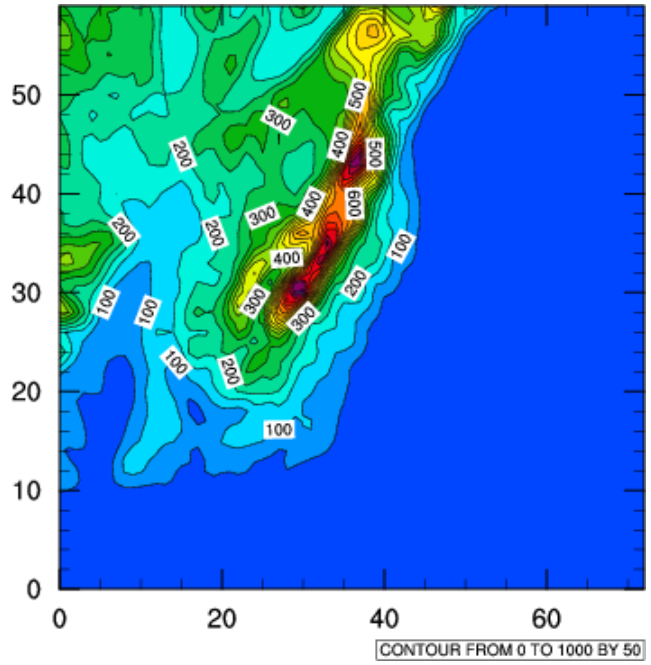
Generate Plots with NCL

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

begin
  a = addfile("./geo_em.d01.nc","r")
  wks = gsn_open_wks("pdf","plt_ter1")
  ter = a->HGT_M(0,:,:)
  plot = gsn_contour(wks,ter,True)
end
```



Generate Plots with NCL

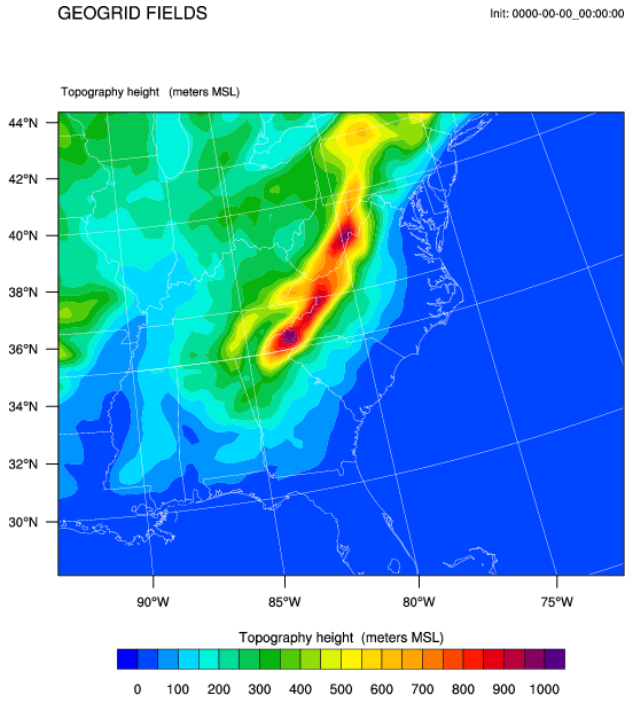
```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"

begin
  a = addfile("./geo_em.d01.nc","r")
  wks = gsn_open_wks("pdf","plt_ter1")
  ter = a->HGT_M(0,:,:)
  res = True
  res@cnFillOn = True
  res@gsnSpreadColors = True
  res@cnLevelSelectionMode = \
     "ManualLevels"
  res@cnMinLevelValF = 0.
  res@cnMaxLevelValF = 1000.
  res@cnLevelSpacingF = 50.
  plot = gsn_contour(wks,ter,res)
end
```



Basic NCL resources - *there are over 1400 controlling contours, labelbars, legends, maps, etc.*

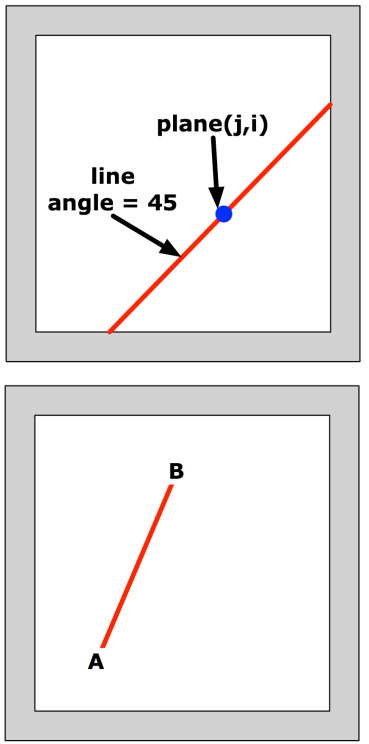
Generate Plots with NCL

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

begin
  a = addfile("./geo_em.d01.nc","r")
  wks = gsn_open_wks("pdf","plt_ter5")
  res = True
  res@MainTitle = "GEOGRID FIELDS"
  pltres = True
  mpres = True

  ter = wrf_user_getvar(a,"HGT_M",0)
  res@cnFillOn = True
  res@ContourParameters = (/0.,1000.,50./)
  contour = wrf_contour(a,wks,ter,res)
  plot = wrf_map_overlays(a,wks,(/contour/),\
    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_list_times**
  Get list if times available in input file
- **wrf_user_unstagger**
  Unstaggers an array

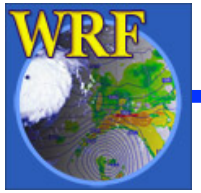


Special WRF NCL Functions

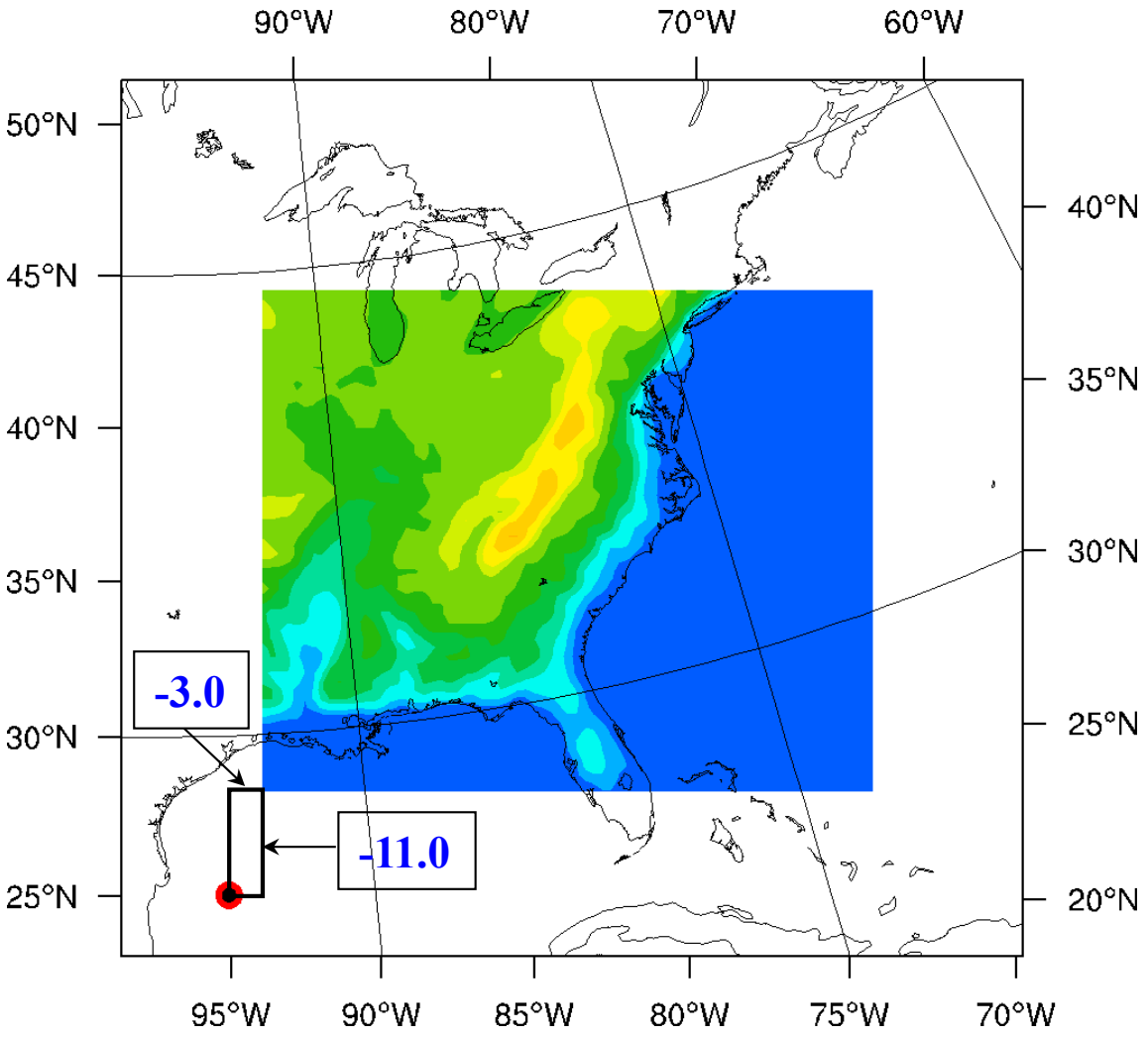
- **wrf_user_getvar**
  Get fields from input file

  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,
  cape_2d: 2D mcape/mcin/lcl/lfc,
  cape_3d: 3D cape/cin,
  dbz/mdbz: Reflectivity (3D and max),
  geopt/geopotential: Geopotential,
  p/pres/pressure: Pressure,
  rh/rh2: Relative Humidity (3D and 2m),
  slp: Sea Level Pressure,
  td/td2: Dew Point Temperature (3D and 2m),
  tc/tk: Temperature (C and F), th/theta: Potential Temperature,
  z/height: Height, ua/va/wa: wind on mass points,
  uvmet/uvmet10: wind rotated to earth coordinates (3D and 10m)



wrf_user_ll_to_ij



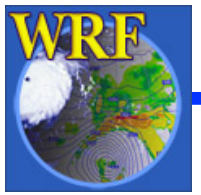
NCL and WRF_NCL

- Combine strength of WRF_NCL specific and NCL general capabilities

```
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)
```

```
res@cnLevelSelectionMode = \
     "ManualLevels"
res@cnMinLevelValF = 0.
res@cnMaxLevelValF = 1000.
res@cnLevelSpacingF = 50.
contour=wrf_contour(a,wks,ter,res)


res@ContourParameters = \
     (/0.,1000.,50./)
contour=wrf_contour(a,wks,ter,res)
```

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


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

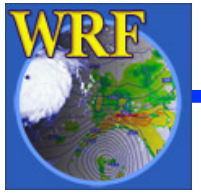


NCL Resources

- **The special WRF functions have unique resources:**
  http://www.mmm.ucar.edu/wrf/OnLineTutorial/Graphics/NCL/NCL_functions.htm

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

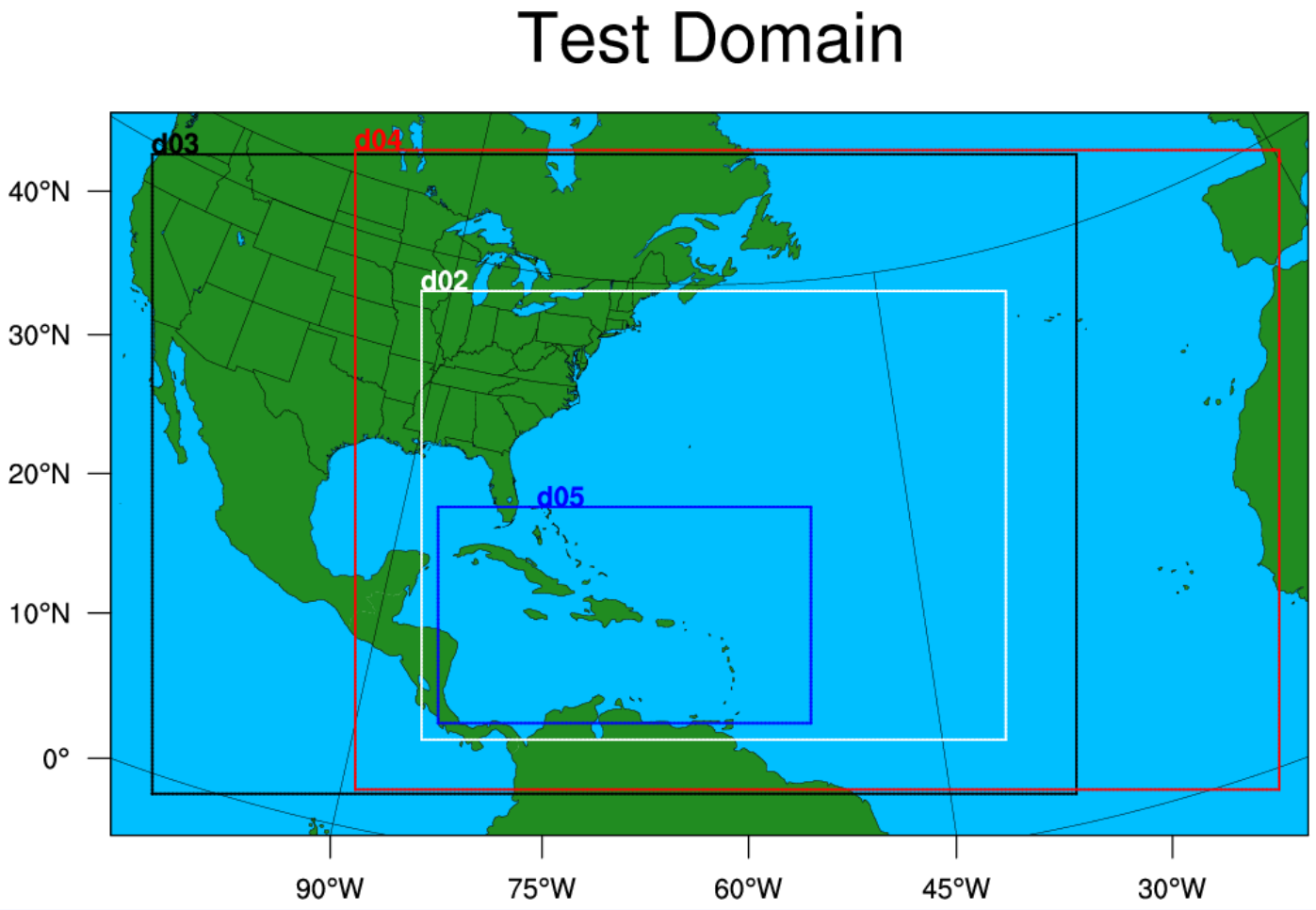
  - am (annotation manager)
  - app (app)
  - ca (coordinate array)
  - cn (contour)
  - ct (coordinate array table)
  - dc (data comm)
  - err (error)
  - gs (graphic style)
  - gsn (gsn high-level interfaces)
  - lb (label bar)
  - lg (legends)
  - mp (maps)
  - pm (plot manager)
  - pr (primitives)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



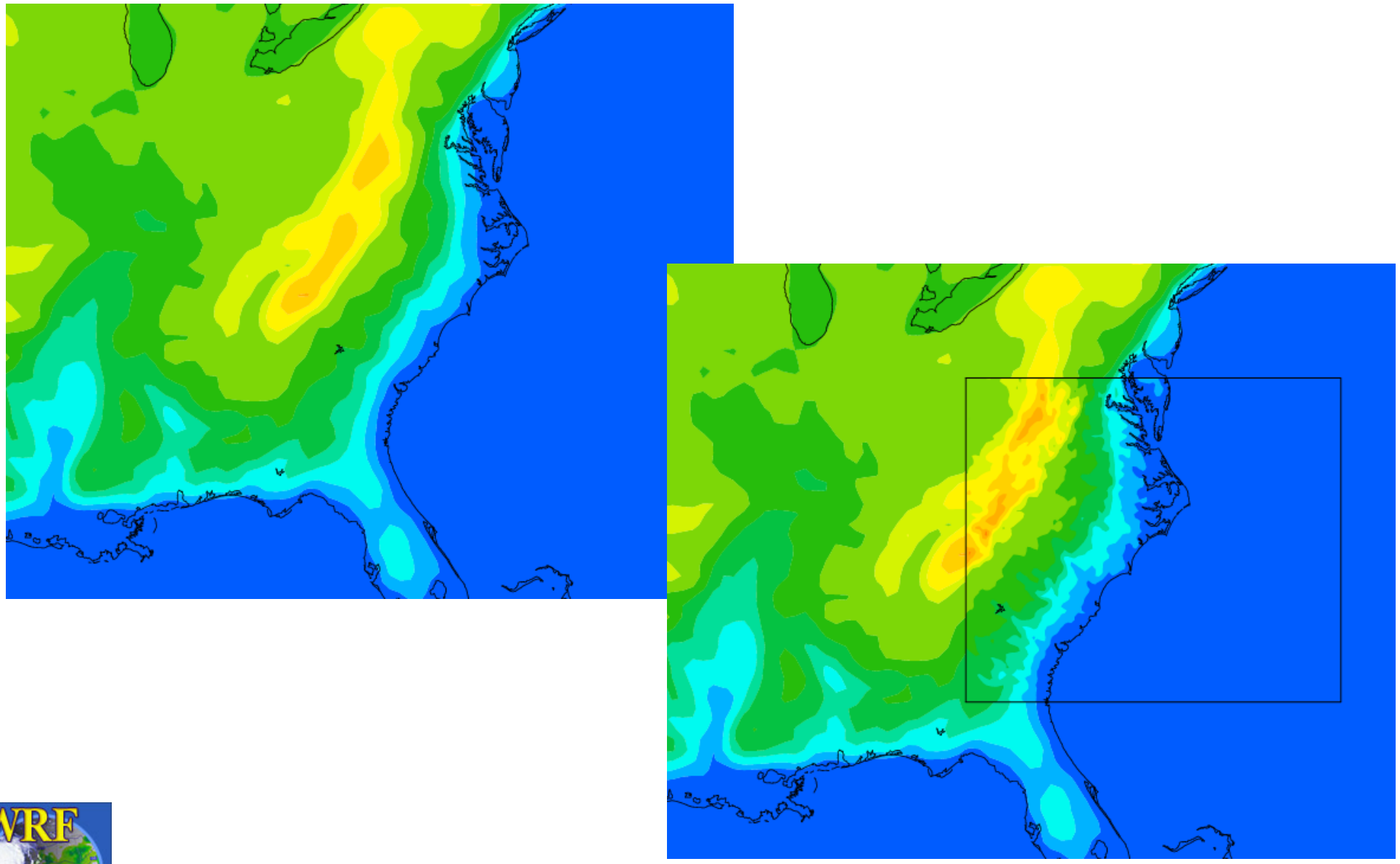
Domain Design

mp = wrf_wps_dom (wks, mpres, lnres, txres)

WPS/util/plotgrids.ncl



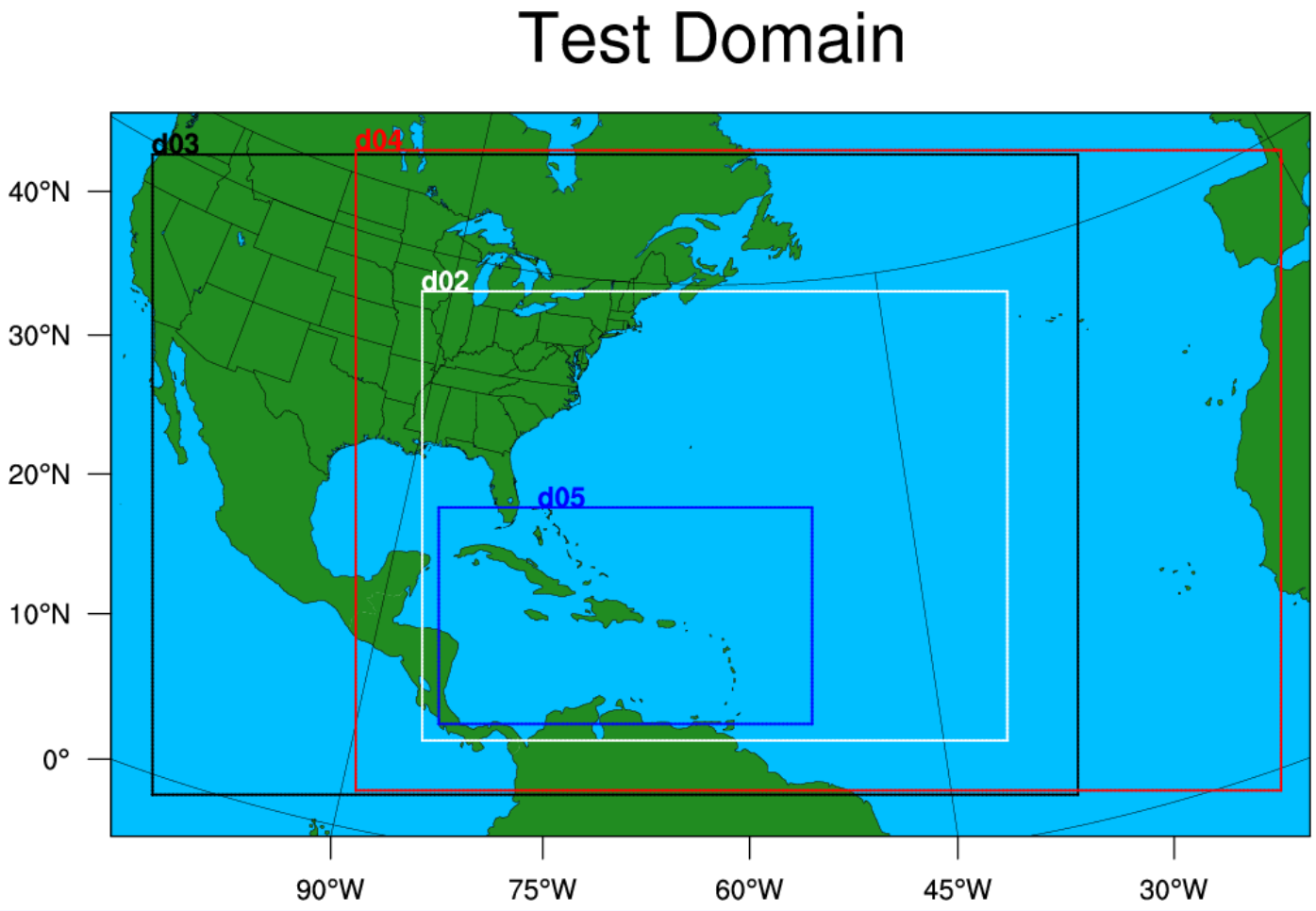
Overlay Domains



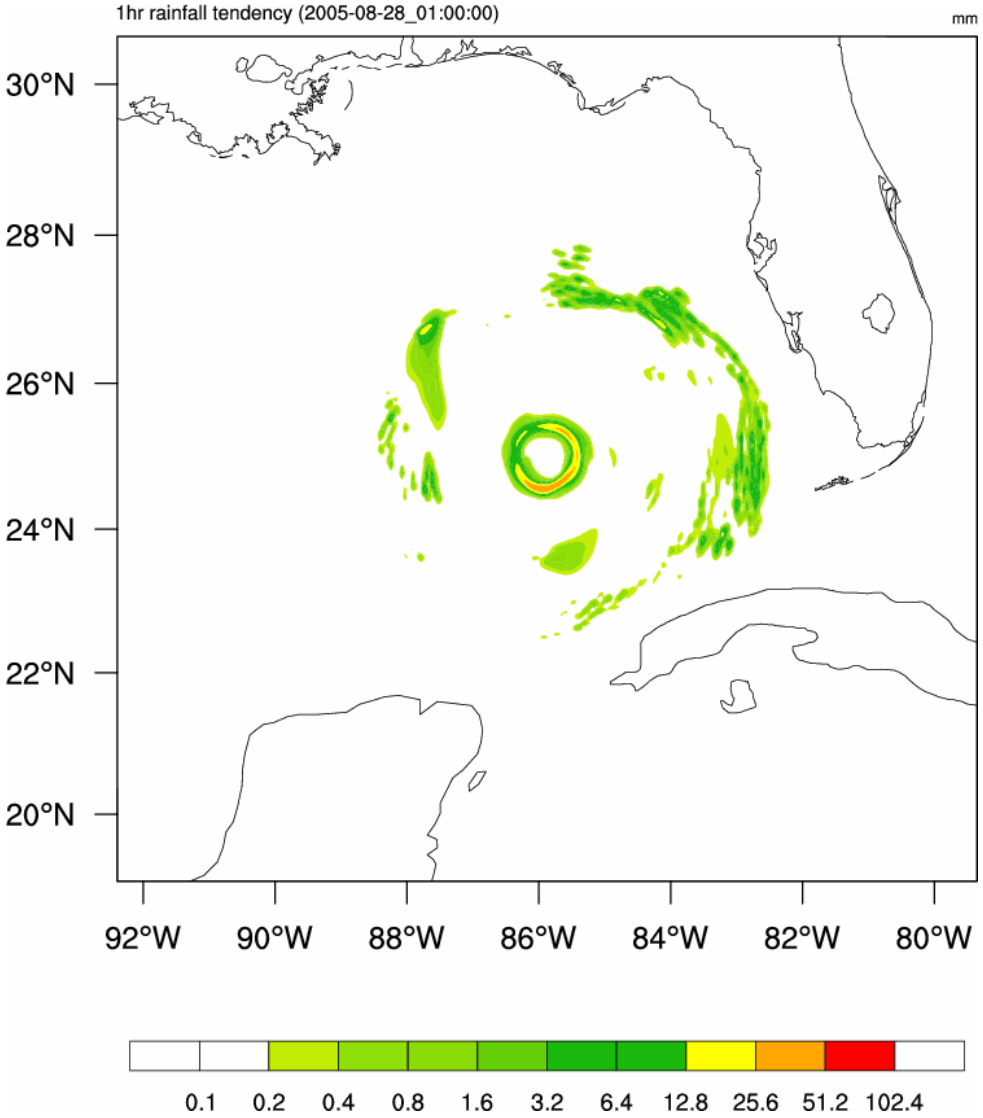
Domain Design

mp = wrf_wps_dom (wks, mpres, lnres, txres)

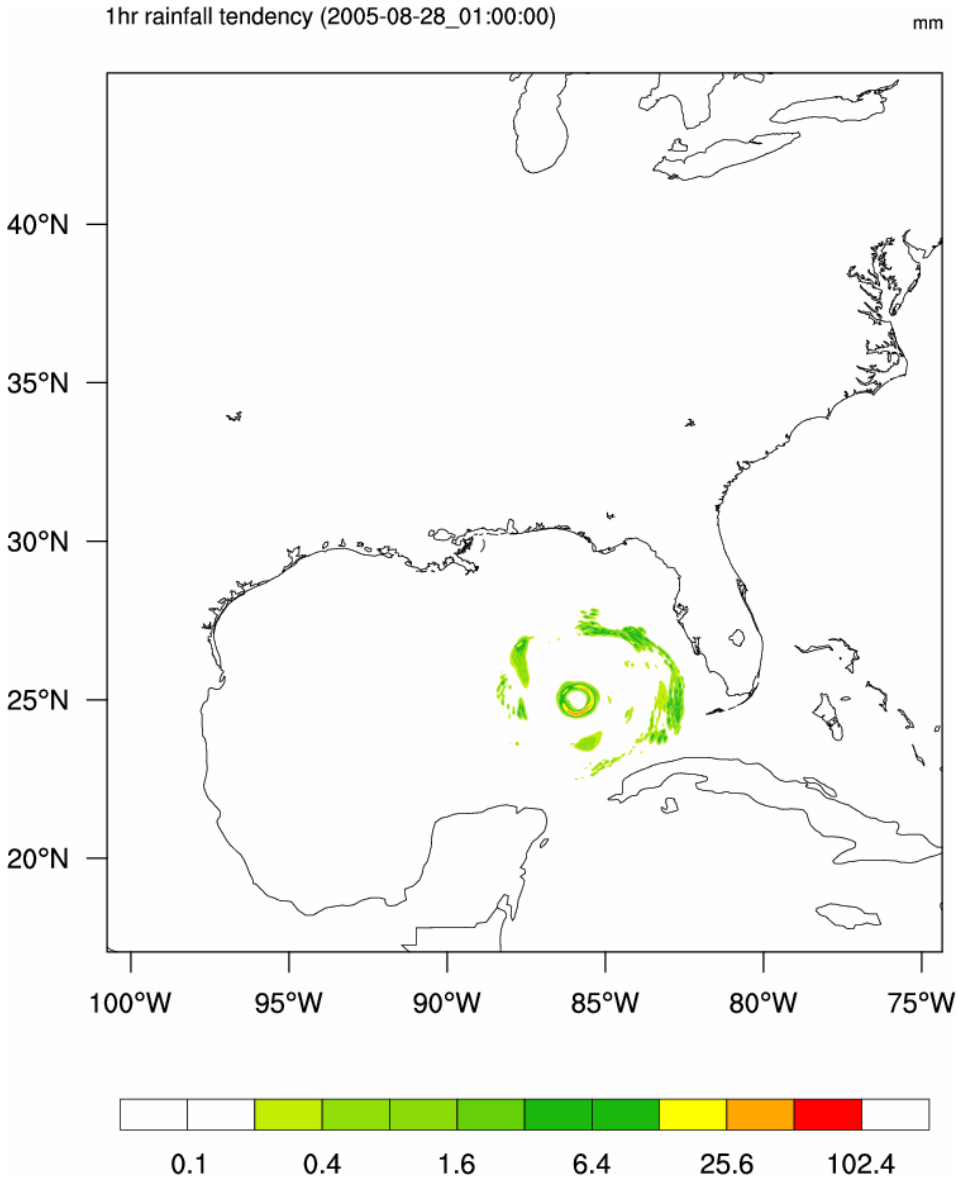
WPS/util/plotgrids.ncl



Moving Nests



Moving Nests

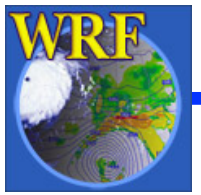


Calling FORTRAN code from NCL

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

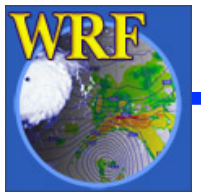
```
C NCLFORTSTART
C NCLEND
```

- Use a tool called `WRAPIT` to create a `*.so` file
- Load `*.so` file in NCL script with "`external`" statement
- Call Fortran function with special "`::`" syntax
- Must pre-allocate arrays!



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
```

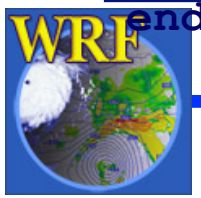


myTK.so - *Create & use in NCL script*

```
% WRAPIT myTK.f
```

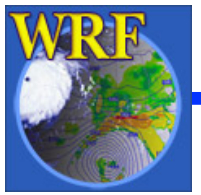
This will create a "myTK.so" file

```
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
```

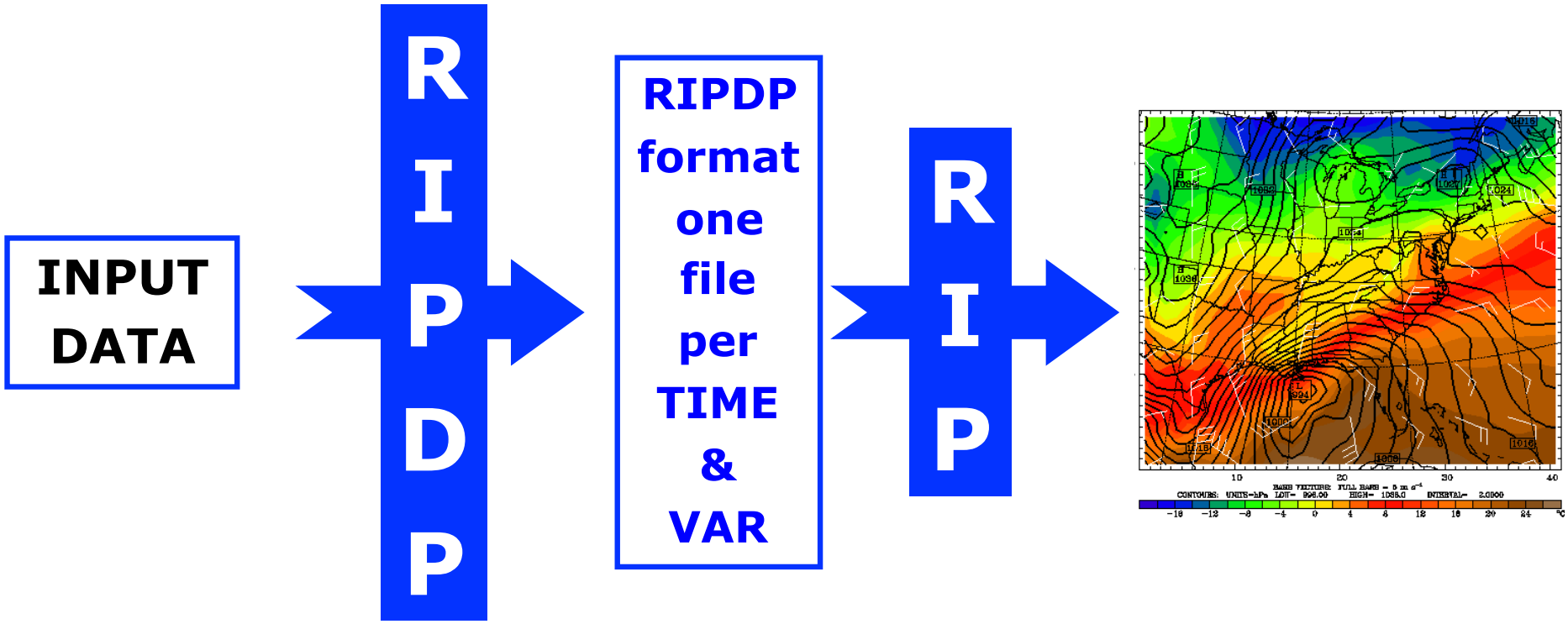


FORTRAN 90 code

- **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.

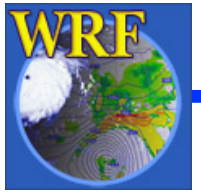


RIP4



RIP4 General Information

- Requires NCL Version 5 or NCAR Graphics low-level routines
  - *http://www.ncl.ucar.edu (Recommended)*
  - *http://ngwww.ucar.edu*
- Source Code:
  - *http://www.mmm.ucar.edu/wrf/users/download/get_source.html*
- Documentation
  - In program tar file under the Doc/ directory
  - *http://www.mmm.ucar.edu/wrf/users/docs/ripug.htm*
- OnLine Tutorial:
  - *http://www.mmm.ucar.edu/wrf/users/graphics/RIP4/RIP4.htm*



RIP4 on your computer

- **set environment variables**
  setenv RIP_ROOT /usr/$USER/RIP4
  setenv NCARG_ROOT /usr/local/ncl *(/usr/local/ncarg)*

- **Configure**
  ./configure
  (check configure.rip to ensure netCDF paths are correct)

- **Compile**
  ./compile

- **RIP4 has 2 parts (RIPDP and RIP)**

Compiling with ncl

ncl myscript.ncl

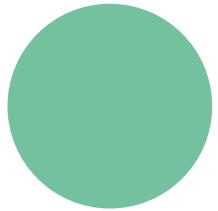
NCARG_ROOT

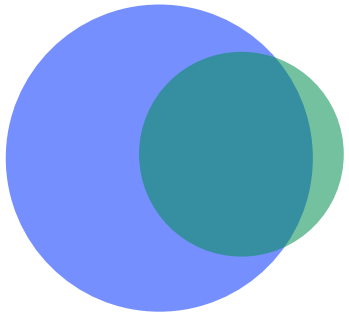
/usr/local/ncl

compile <util>

NCARG_ROOT

/usr/local/ncarg





ncl myscript.ncl

compile <util>

NCARG_ROOT

/usr/local/ncl

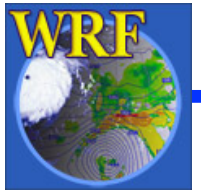
Compiling with ncl

```
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

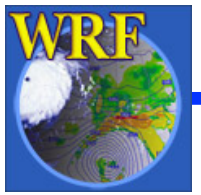


ripdp & rip

- **ripdp**
  - RIP Data Preparation
  - Converter
    RIPDP converts WRF formatted data (*netCDF*) into RIP format (*B – grid*)
  - Output
    RIPDP puts each Variable at each Time into a separate file
    Creates LOTS of files

mkdir RIPDP

- **rip**
  - Reads the output generated by *ripdp*
  - Makes use of a User Input File (UIF) *(rip_sample.in)* to control plots
  - Output: X11, pdf, ps, cgm



Running ripdp & rip

ripdp_wrf*xxx* [-n namelist-file]
<model_data_name> \ [basic/all]
<input_file(s)>

rip [-f] <model_data_name> rip-execution-name

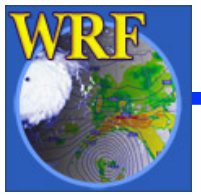
Example:

ripdp_wrfarw RIPDP/arw all wrfout*

rip [-f] RIPDP/arw rip_sample.in

output

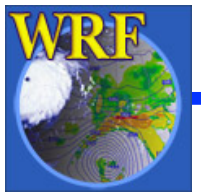
[rip_sample.out]
rip_sample.TYPE



RIP4 Common Error Message

- **Most often this is 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

```
GKS ERROR NUMBER   2 ISSUED FROM SUBROUTINE
GCLKS :--GKS NOT IN PROPER STATE: GKS SHALL BE IN
STATE GKOPFORTRAN STOP
```

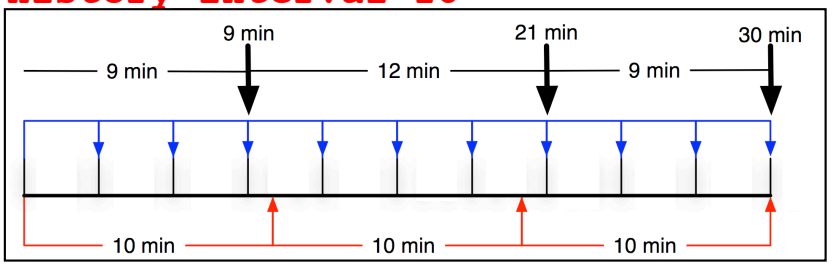


RIP4 namelist

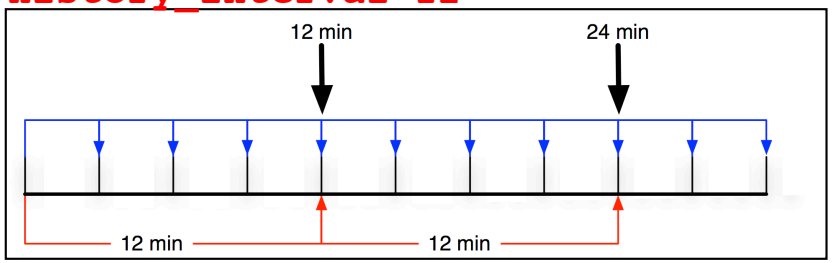
- **ptimes** *(times for ripdp to process)*
  - `0,1,2,3,4,5,6` (0,1,2,3,4,5,6)
  - `0,-6,1` (0,1,2,3,4,5,6)
  - `0,  2,-4,1,  6` (0, 2,3,4, 6)

- **tacc**: *input files not on exact times* [`time_step=180` (3 min)]

history_interval=10



history_interval=12

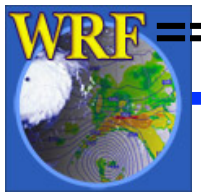


RIP4 User Input File

```
&userin
    .......
&end
&trajcalc
    .......
&end
=============================================
---------------      Plot Specification Table      ---------------
=============================================
feld= .....
feld= .....
=============================================
feld= .....
feld= .....
=============================================
```

Frame (overlay) of a number of fields

Plot a specific field

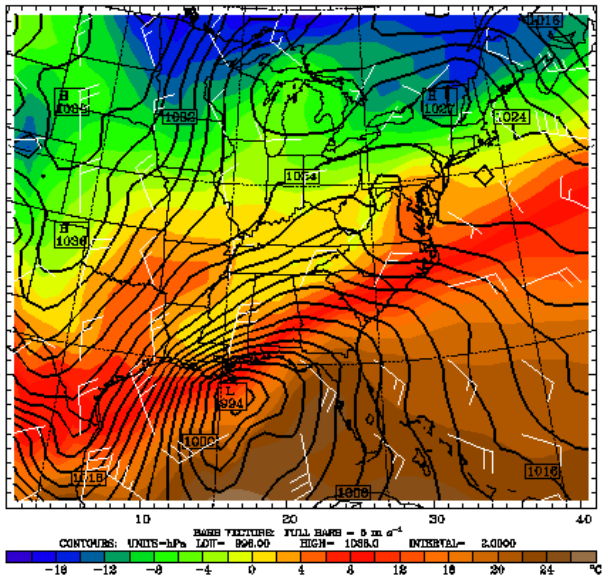


Creating a Plot with RIP4

feld=
diagnostics - *tmc*
native - *PSFC*

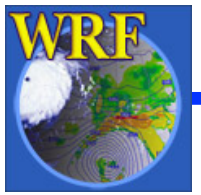
vcor=s; **levs=2fb**
vcor=s; **levs=1,2,3**
vcor=p; **levs=800,500**
vcor=p; **levs=800,-300,100**

```
==========================================
feld=tmc; ptyp=hc; vcor=s; levs=1fb; >
   cint=2; cmth=fill; >
   cosq=32,light.violet,-16,blue, >
   0,yellow,16,orange,32,light.gray
feld=slp; ptyp=hc; cint=2; linw=2
feld=uuu,vvv; ptyp=hv; vcmx=1; >
    colr=white;intv=5
feld=map; ptyp=hb
feld=tic; ptyp=hb
==========================================
```



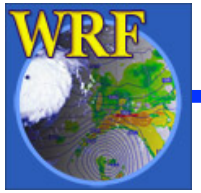
ARWpost - *converter*

- Download Code (http://www.mmm.ucar.edu/wrf/users)
- OnLine Tutorial
  http://www.mmm.ucar.edu/wrf/users/graphics/ARWpost/ARWpost.htm
- For GrADS output
  - GrADS libraries only needed to display data *(freely available)*
  - http://grads.iges.org/grads/grads.html
- Version 2 (old – not recommended)
  - Could produce vis5d output
  - Needed WRFV3 complied

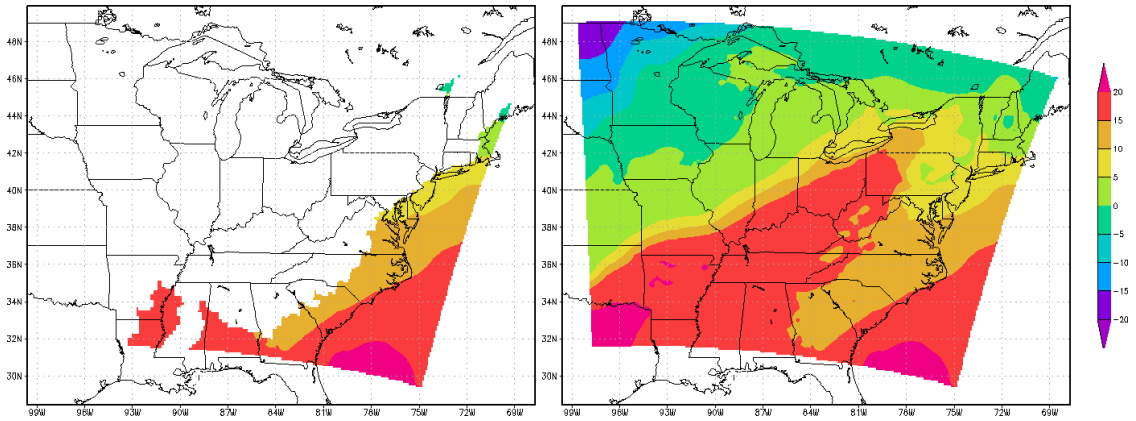


namelist.ARWpost

split_output	Split your GrADS output files into a number of smaller files *(a common .ctl file will be used for all .dat files).*
frames_per_outfile	If *split_output* is **.True.**, how many time periods are required per output (.dat) file.
plot	Which fields to process. (*all, list, all_list)* Order has no effect, i.e., "all_list" and "list_all" "**list**" – list variables in "**fields**"
fields	Fields to plot. Only used is list was used in the "plot" variable. **Must use to generate diagnostics.**
Available diagnostics: cape, cin, mcape, mcin, clfr, dbz, max_dbz, geopt, height, lcl, lfc, pressure, rh, rh2, theta ,tc, tk, td, td2, slp, umet, vmet, u10m, v10m, wdir, wspd, wd10, ws10	



namelist.ARWpost

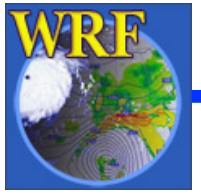
interp_method	0 = sigma levels, -1 = code defined "nice" height levels, 1 = user defined height or pressure levels
interp_levels	Only used if interp_method=1 Supply levels to interpolate to, in hPa (*pressure*) or km (*height above sea level*) Supply levels bottom to top
extrapolate	Extrapolate below ground *(default .false.)* 

GrADS - .ctl file

```
dset ^test.dat
options byteswapped
title  OUTPUT FROM WRF V2.2 MODEL
pdef  259 163 lcc  40.000  -98.000  130.000   82.000
60.00000  30.00000  -98.00000  22000.000  22000.000
xdef  877 linear -141.49254   0.09909910
ydef  389 linear   18.88639   0.09909910
```

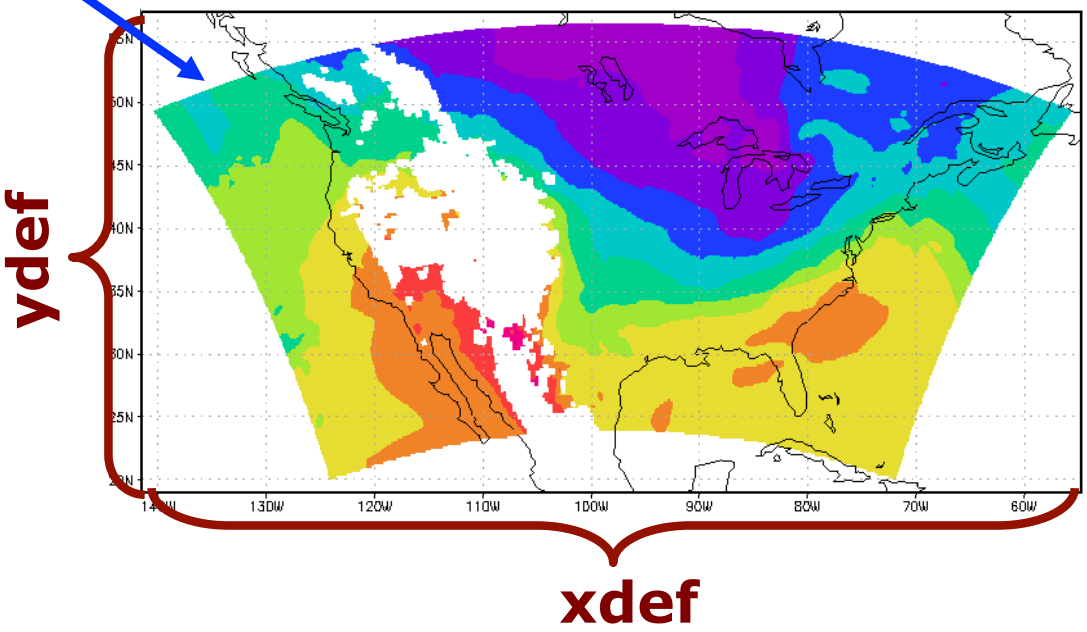
options byteswapped

Needed on some machines - if you get NaNs when you plot, ***remove*** *this line from .ctl file*



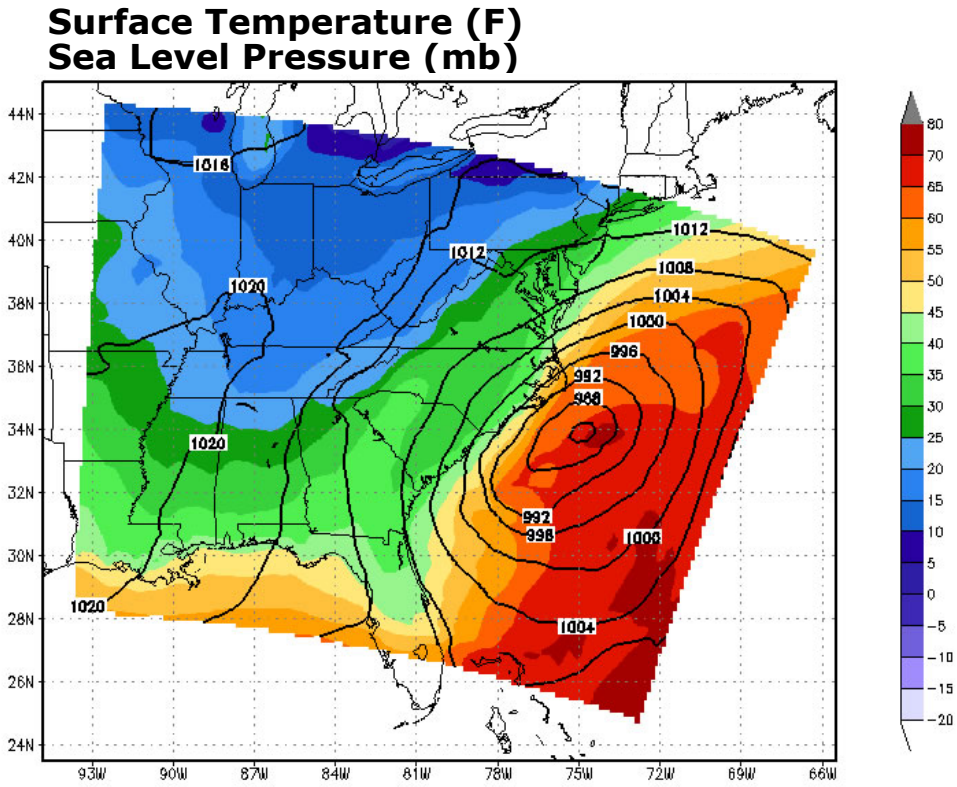
GrADS - .ctl file

```
dset ^test.dat
options byteswapped
title  OUTPUT FROM WRF V2.2 MODEL
pdef  259 163 lcc  40.000  -98.000  130.000   82.000
60.00000  30.00000  -98.00000  22000.000  22000.000
xdef  877 linear -141.49254   0.09909910
ydef  389 linear   18.88639   0.09909910
```



Creating a Plot with GrADS

```
open em_real.ctl
set mpdset hires
set display color white

define tf=1.8*tc + 32
set gxout shaded
set z 1
d tf
run cbar.gs

set gxout contour
set ccolor 1
set cint 4
d slvl
```



Documentation: http://grads.iges.org/grads/gadoc/index.html

How to add diagnostics

- **RIP4**
  - Create a subroutine (note RIP4 expects the code to be in "j/I/-k" orientation)
  - Add links to the RIP4/src/fields.f routine
  - Add new subroutine to RIP4/src/Makefile
- **ARWpost**
  - Create a subroutine
  - Add links to ARWpost/src/module_diagnostics.f90
  - Add new subroutine to ARWpost/src/Makefile
- **NCL**
  - Create an NCL script and load/call from other NCL scripts
  - Link in a FORTRAN / C program into an NCL script