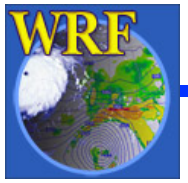
Post-processing Tools

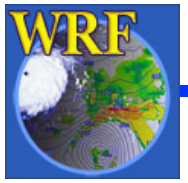
Cindy Bruyère



Graphical Packages

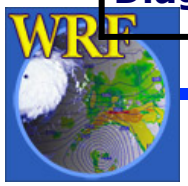
- **NCL** — UG: 9-2
  - Graphical package
- **ARWpost** — UG: 9-28
  - Converter (GrADS)
- **RIP4** — UG: 9-19
  - Converter and interface to graphical package NCAR Graphics
- **UPP** — UG: 9-35
  - Converter (GrADS & GEMPAK)
- **VAPOR** — UG: 9-58
  - Converter and graphical package
  - *Support: VAPOR*
- **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



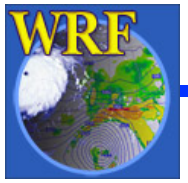
Graphical Packages

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

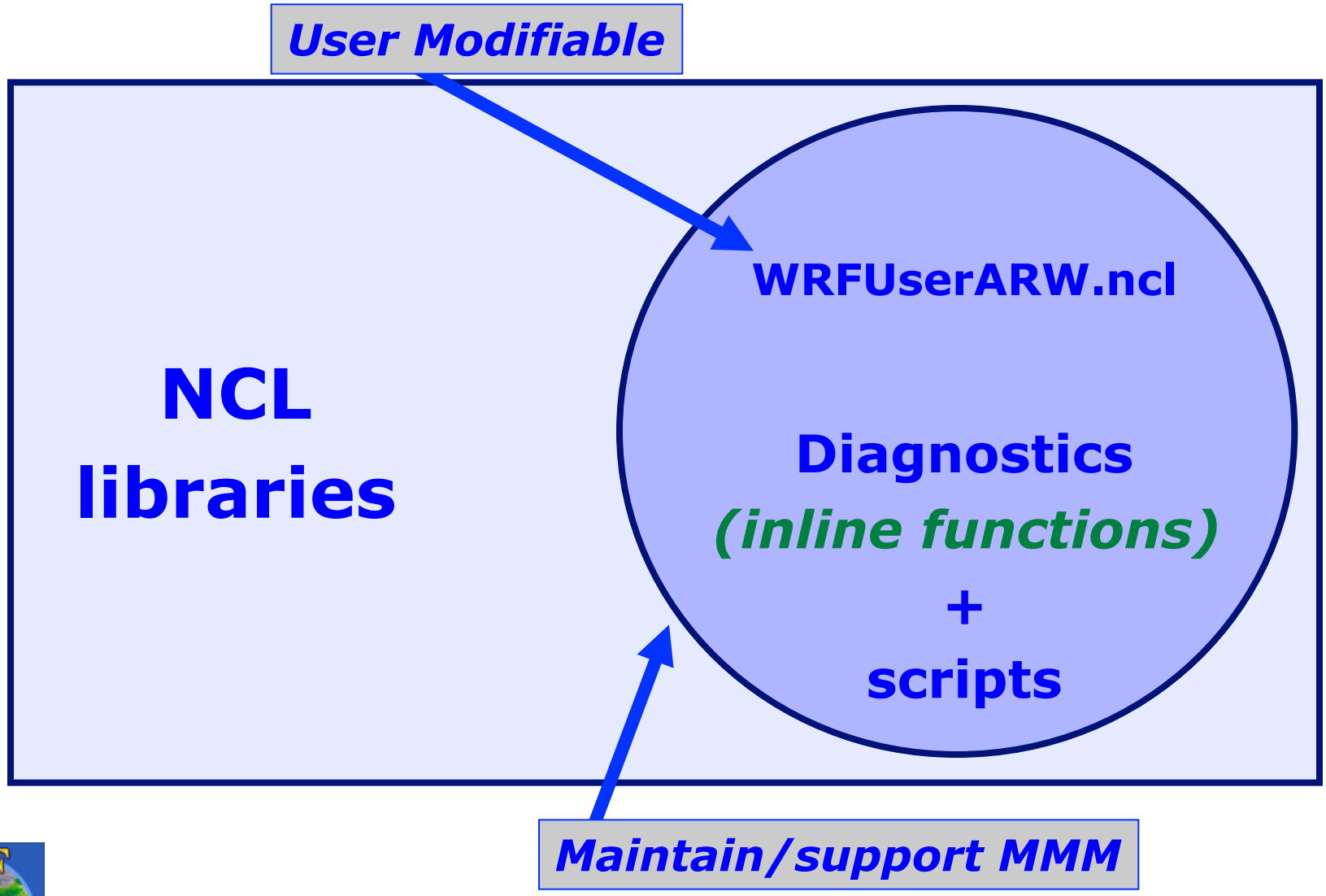


NCL

- NCAR Command Language
- http://www.ncl.ucar.edu
- Read WRF-ARW data directly
- Generate a number of graphical plots
  - Horizontal, cross-section, skewT, meteogram, panel
- 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 *(current v6.0.0)*
- NCARG_ROOT environment variable
  - setenv NCARG_ROOT /usr/local/ncl



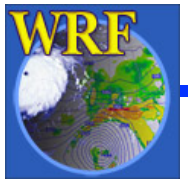
NCL & WRF


User Modifiable
WRFUserARW.ncl
NCL
libraries
Diagnostics
(inline functions)
+
scripts
Maintain/support MMM

Home Directory → ~/.hluresfile ← Very Important

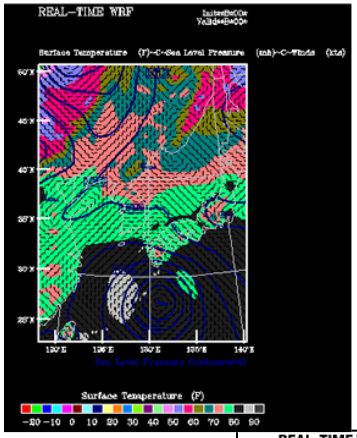
- Required by NCL libraries
- Must be in your "~/" directory *(home directory)*
- Control
  - color table ; font
  - white/black background
  - size of plot
  - control characters

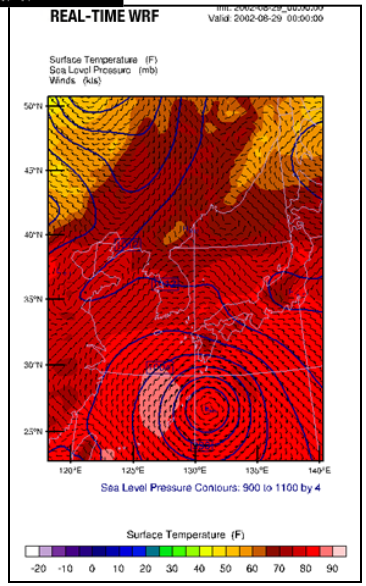
- http://www.ncl.ucar.edu/Document/Graphics/hlures.shtml



~/.hluresfile

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



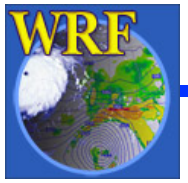



```
PLCHHQ - CHARACTER NUMBER 26 (.) IS NOT A LEGAL FUNCTION CODE
PLCHHQ - CHARACTER NUMBER 29 (t) IS NOT A LEGAL FUNCTION CODE
PLCHHQ - CHARACTER NUMBER 30 (o) IS NOT A LEGAL FUNCTION CODE
PLCHHQ - CHARACTER NUMBER 32 (.) IS NOT A LEGAL FUNCTION CODE
PLCHHQ - CHARACTER NUMBER 35 (b) IS NOT A LEGAL FUNCTION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

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

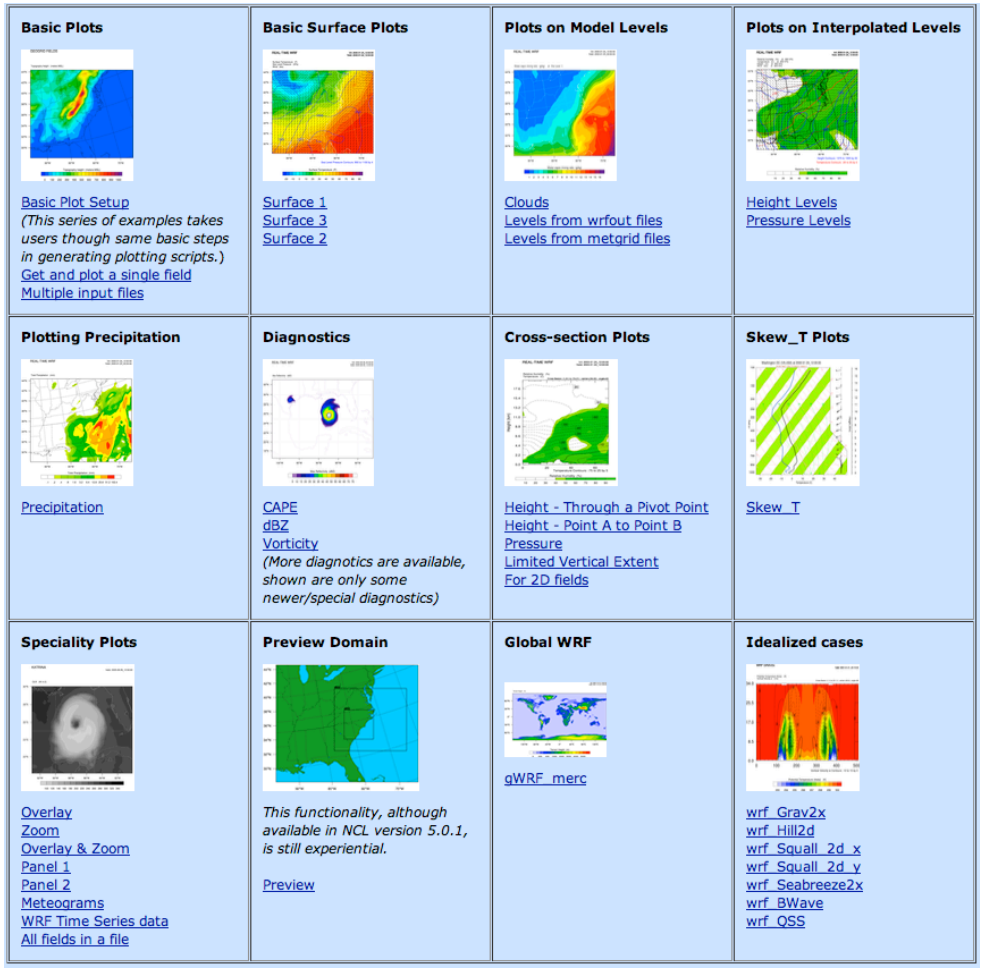
Generate Plots with NCL

- 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*



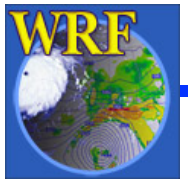
Generate Plots: *A good start - OnLine Tutorial*

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



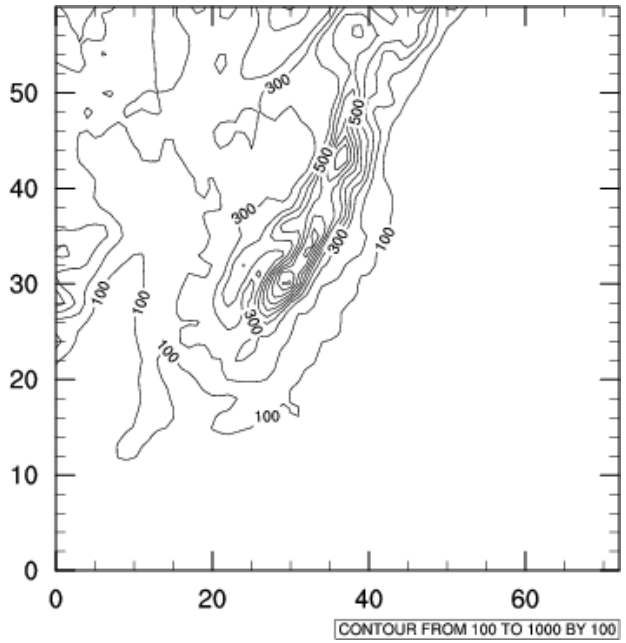
Creating a Plot : NCL script

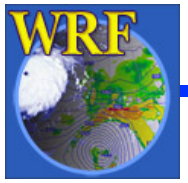
```
load ncl library scripts

begin

  ; Open graphical output
  ; Open input file(s)

  ; Read variables

  ; Set up plot resources & Create plots
  ; Output graphics

end
```



Generate Plots

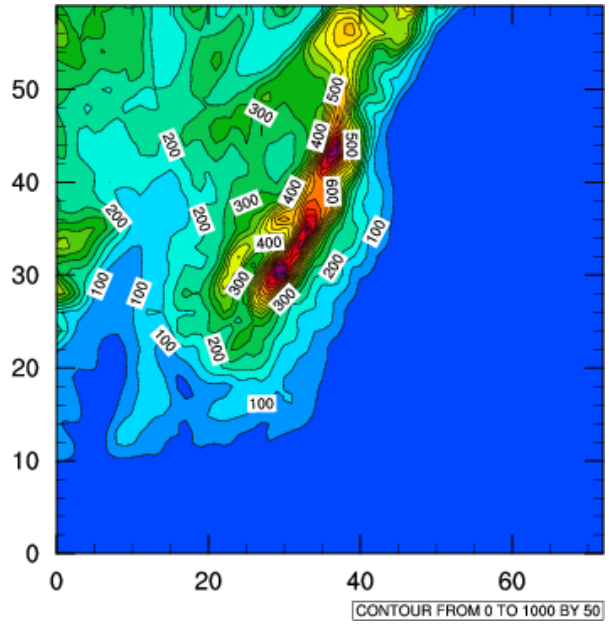
```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"

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

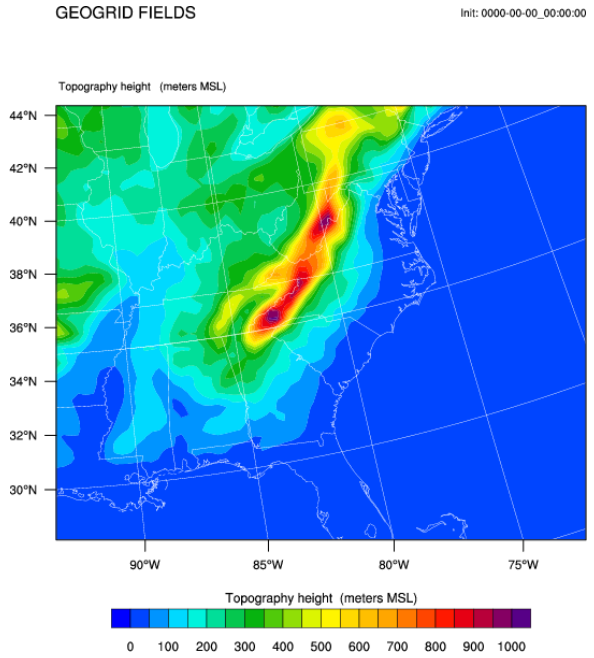
```
load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"

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.*

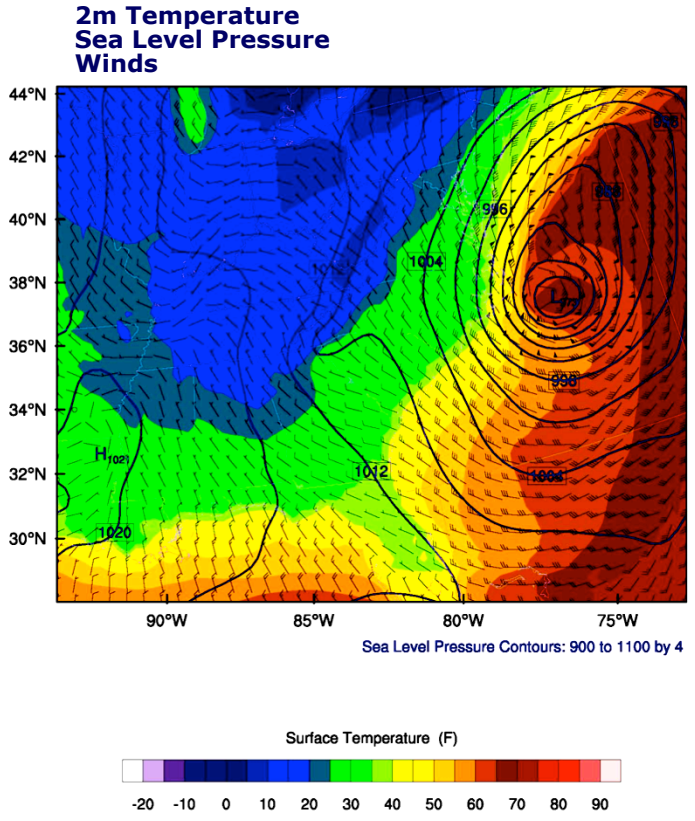
Generate Plots

```
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
```



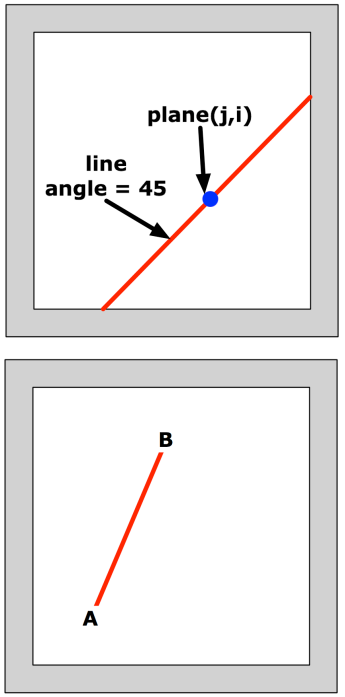
Generate Plots

```
slp = wrf_user_getvar(a,"slp",5)
t2  = wrf_user_getvar(a,"T2",5)
u10 = wrf_user_getvar(a,"U10",5)

v10 = wrf_user_getvar(a,"V10",5)

os@cnLineColor = "NavyBlue"
c_slp = wrf_contour(a,wks,slp,os)
ot@cnFillOn = True
c_tc = wrf_contour(a,wks,t2,ot)
ov@NumVectors = 47
vec = wrf_vector(a,wks,u10,v10,ov)

plot = wrf_map_overlays(a, wks, \
       (/c_tc,c_slp,vec/), pltres, mpres)
```



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

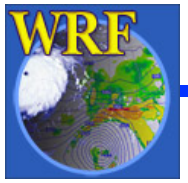
  ter = wrf_user_getvar(a,"HGT",0) {ter=a->HGT(0,:,:)}
  t2 = wrf_user_getvar(a,"T2",-1) {t2=a->T2}
  slp = wrf_user_getvar(a,"slp",1)

  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)

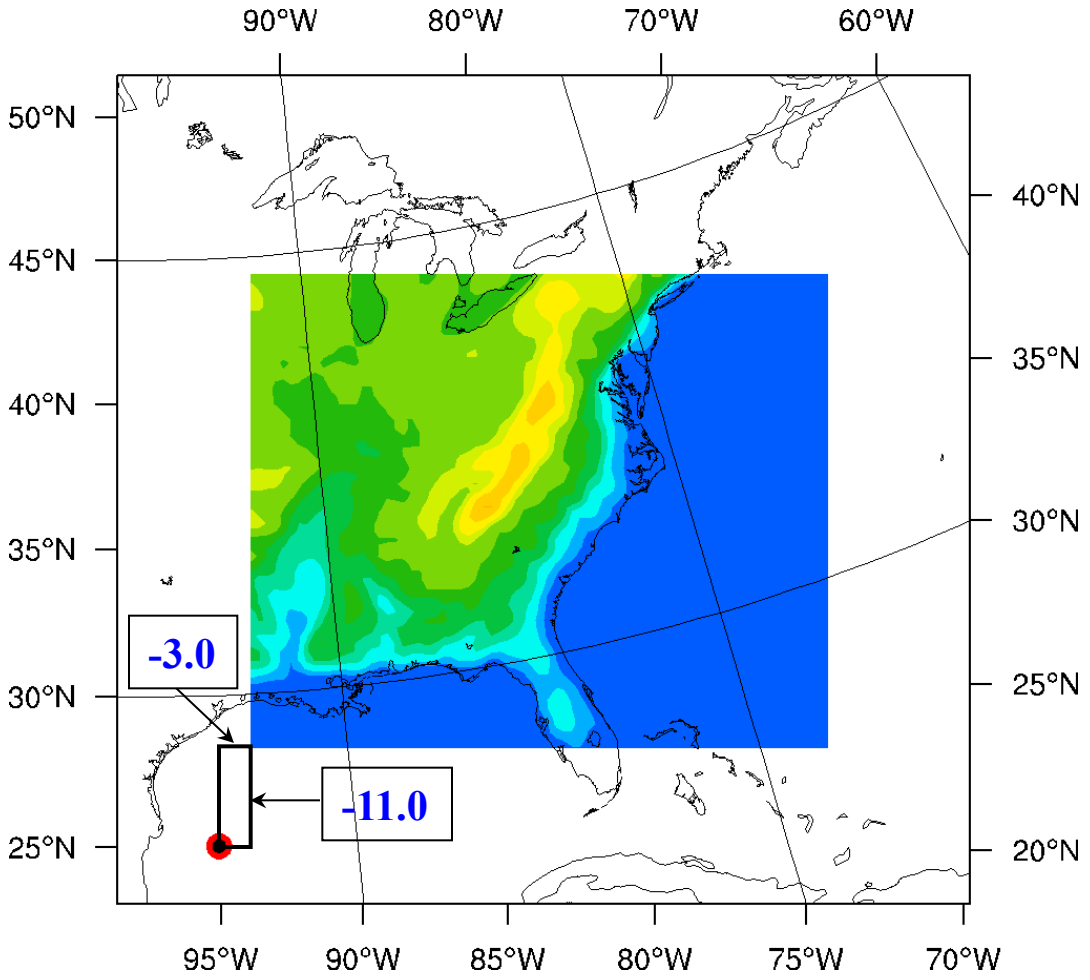
Diagnostics

```
PH      = nc_file->PH(5,:,:,:)
PHB     = nc_file->PHB(5,:,:,:)
   PH     = ( PH + PHB ) / 9.81
   z = wrf_user_unstagger(PH,PH@stagger)

P       = nc_file->P(5,:,:,:)
PB      = nc_file->PB(5,:,:,:)
   P = P + PB

T       = nc_file->T(5,:,:,:)
   T = T + 300.
   tk = wrf_tk( P , T )

QVAPOR = nc_file->QVAPOR(5,:,:,:)
   QVAPOR = QVAPOR > 0.000


slp = wrf_slp( z, tk, P, QVAPOR )
```

```
slp = wrf_user_getvar(nc_file,"slp",5)
```

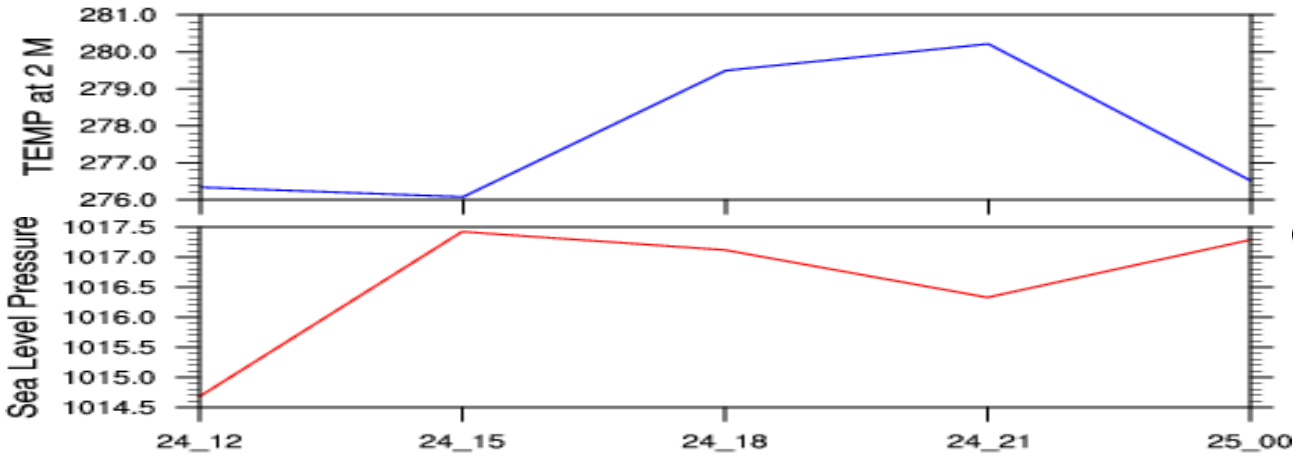


wrf_user_ll_to_ij



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)
```



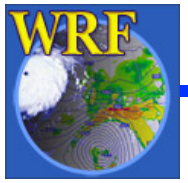
NCL and WRF_NCL

- 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)
```

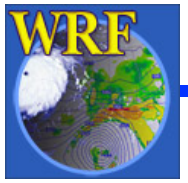


Resources

- **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

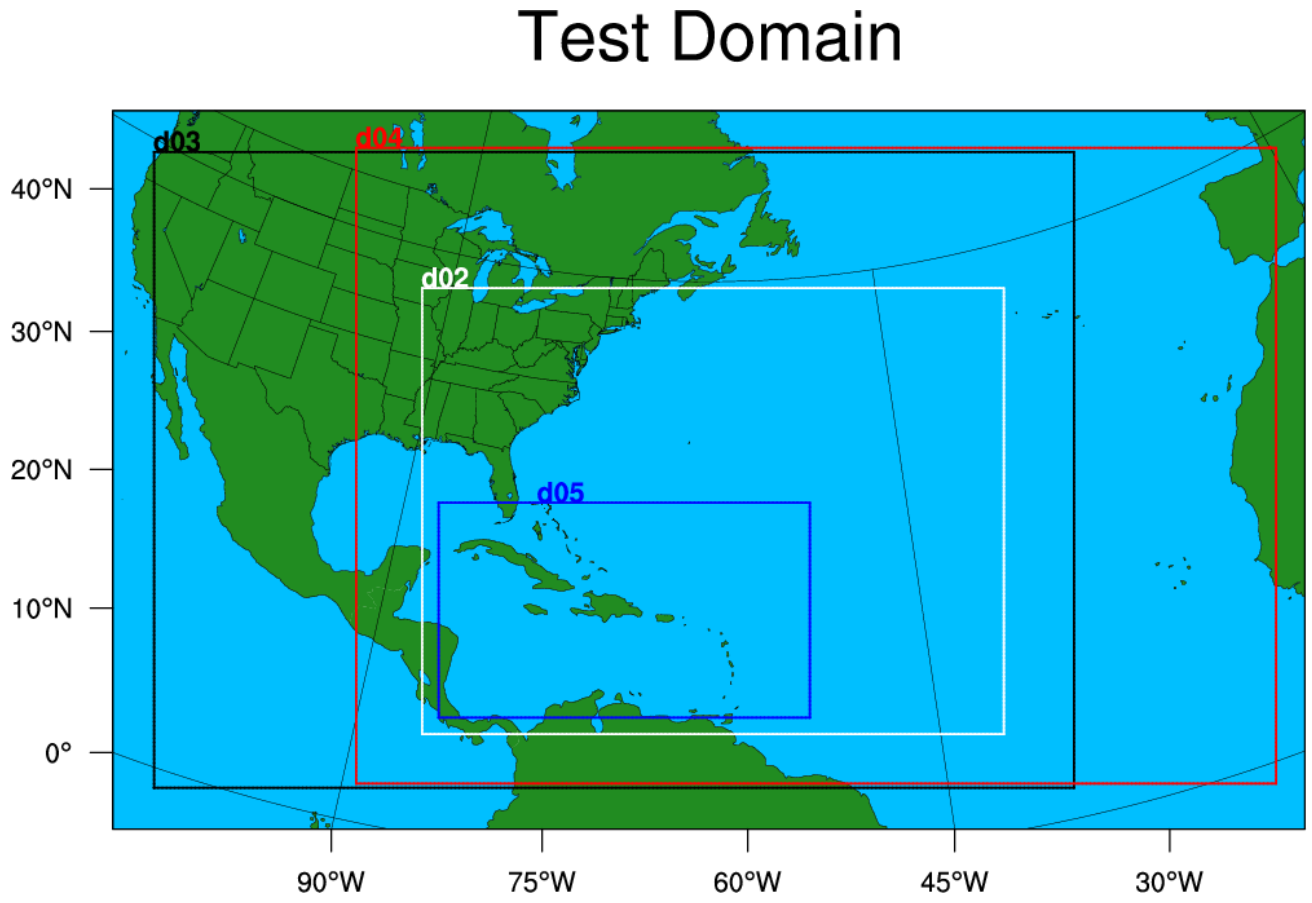
  - am (annotation manager)
  - app (app)
  - ca (coordinate array)
  - cn (contour)
  - ct (coordinate array table)
  - dc (data comm)
  - err (error)
  - gs (graphic style)
  - gsn (gsn high-level interfaces)
  - lb (label bar)
  - lg (legends)
  - mp (maps)
  - pm (plot manager)
  - pr (primitives)
  - 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)



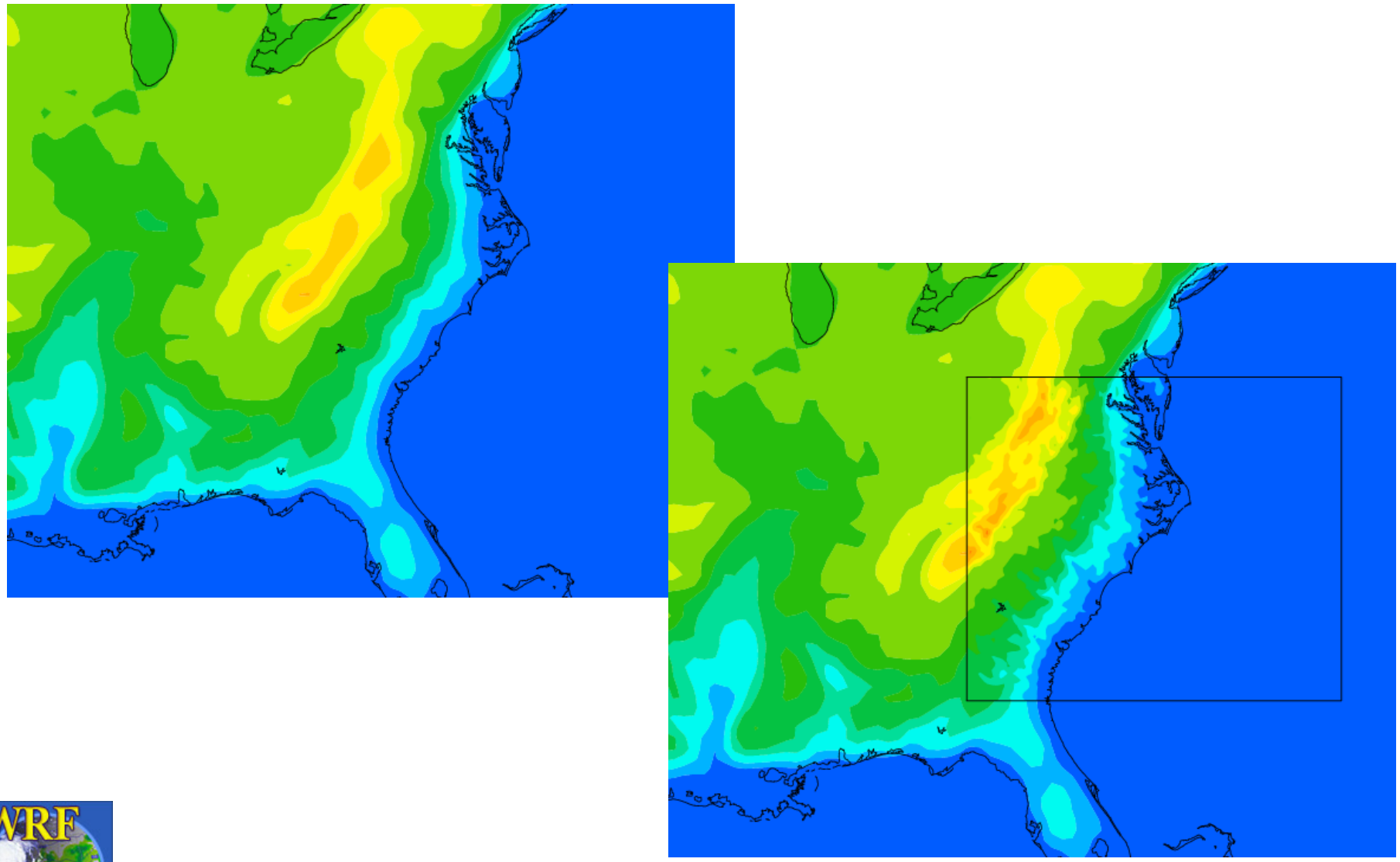
Domain Design

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

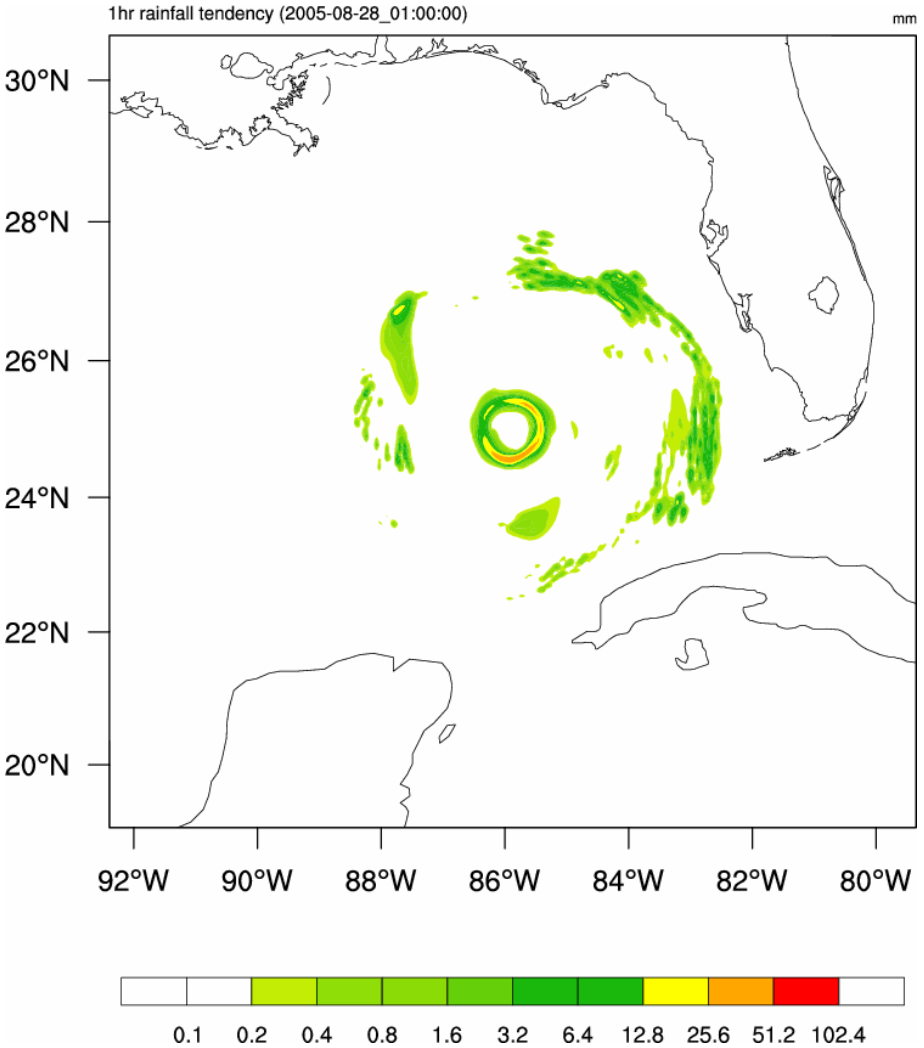
WPS/util/plotgrids.ncl



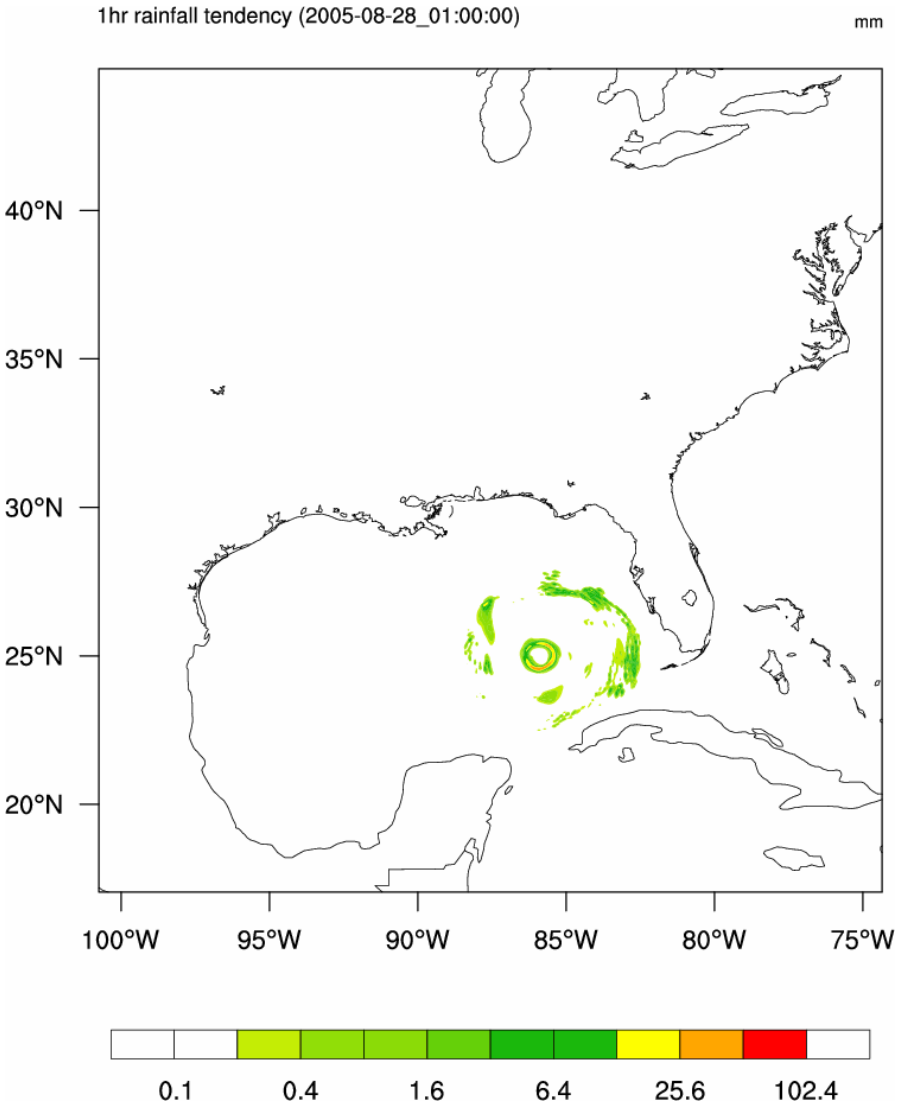
Over Domains



Moving Nests



Moving Nests

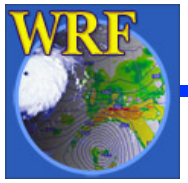


Calling FORTRAN code from NCL

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

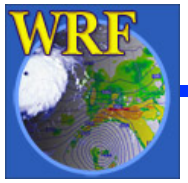
```
C NCLFORTSTART
C NCLEND
```

- 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
```

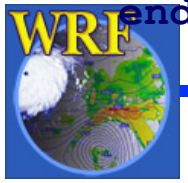


myTK.so - *Create & use in NCL script*

```
% WRAPIT myTK.f
```

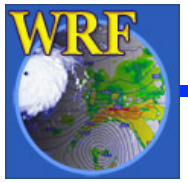
This will create a "myTK.so" file

```
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
```

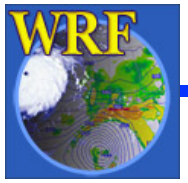


FORTRAN 90 code

- 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.

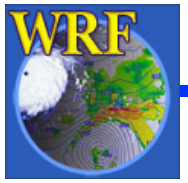


Other Post-processing Tools



RIP4

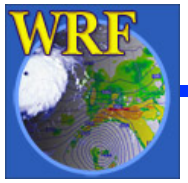
- Read Interpolate Plot version 4
- Develop by Mark Stoelinga (3TIER/UW/NCAR) & MMM/NCAR Staff
- Originally developed for the MM5 model
- Generate a number of graphical plots
  - Horizontal, cross-section, skewT
- Current Version: 4.6
  - configure / compile



RIP4 - General Information

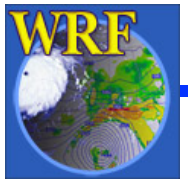
- **Documentation**
  - In program tar file under the Doc/ directory
  - *http://www.mmm.ucar.edu/wrf/users/docs/ripug.htm*
  - *http://www.dtcenter.org/wrf-nmm/users/docs/user_guide/RIP/ripug.htm*

- **OnLine Tutorial:**
  - *http://www.mmm.ucar.edu/wrf/users/graphics/RIP4/RIP4.htm*
  - *http://www.dtcenter.org/wrf-nmm/users/OnLineTutorial/NMM/RIP/index.php*

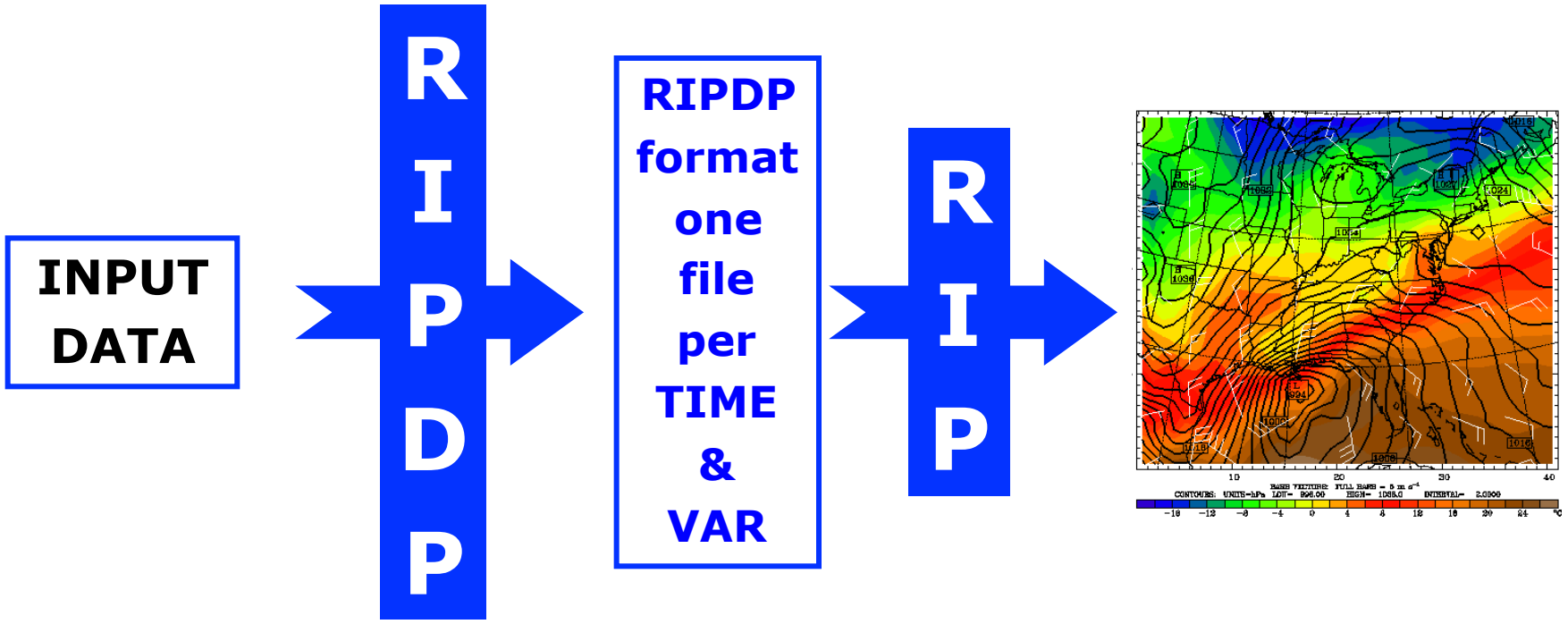


RIP4 - General Information

- Requires NCAR Graphics low-level routines
  - *http://ngwww.ucar.edu*
- NCL Version 5 / 6:
  - *http://www.ncl.ucar.edu*
  - Released November 2007
  - Combine NCL and NCAR Graphics
  - Open Source
  - Recommended
- Download Code:
  - *http://www.mmm.ucar.edu/wrf/users/download/get_source.html*
  - *http://www.dtcenter.org/wrf-nmm/users/downloads/index.php*



RIP4



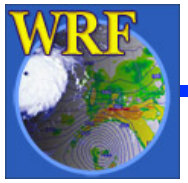
RIP4 on your computer

- **set environment variables**
  setenv RIP_ROOT /usr/$USER/RIP4 *(rip_root)*
  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)**



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

- **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

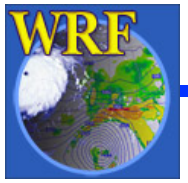
Example:

ripdp_wrfarw RIPDP/test all wrfout*

rip [-f] RIPDP/test rip_sample.in

output

[rip_sample.out]
rip_sample.TYPE



rip UIF

```
&userin
    .......
&end
```

Namelist controlling general parameters

```
&trajcalc
    .......
&end
```

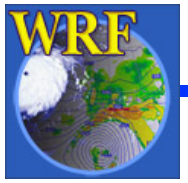
Namelist for trajectory calculations
Only used if itrajcalc=1, in userin namelist

```
=====================================
---------    Plot Specification Table   ----------
=====================================
feld= .........
feld= .........
=====================================
feld= .........
feld= .........
=====================================
```

Frame specification group (FSG)

Plot Specification Table (PST)

Plot specification line (PSL)

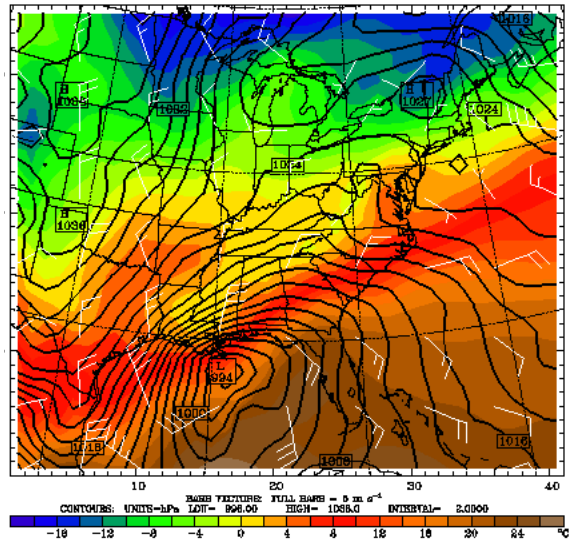


Creating a Plot

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**

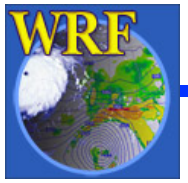
```
==========================================
feld=tmc; ptyp=hc; vcor=s; levs=1fb; >
   cint=2; cmth=fill; >
   cosq=32,light.violet,-16,blue, >
   0,yellow,16,orange,32,light.gray
feld=slp; ptyp=hc; cint=2; linw=2
feld=uuu,vvv; ptyp=hv; vcmx=1; >
    colr=white;intv=5
feld=map; ptyp=hb
feld=tic; ptyp=hb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



Common Error Message

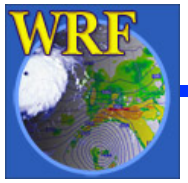
- **Most often this is NOT a graphics error.**
- **More often this is an error with the times you are asking RIP to process**
  - Check the ptimes in your .in file
  - Check the xtimes files created by RIPDP

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



ARWpost

- Converter
  - Requires GrADS to display data
- GrADS software only needed to display data, not needed to compile the code
- Generate a number of graphical plots
  - Horizontal, cross-section, skewT, meteogram, panel
- Version 2 (old – not recommended)
  - Could produce vis5d output
  - Needed WRFV3 complied



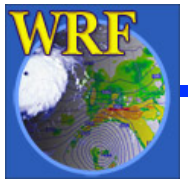
ARWpost - *converter*

- **Download Code** (http://www.mmm.ucar.edu/wrf/users)

- **OnLine Tutorial**
  http://www.mmm.ucar.edu/wrf/users/graphics/ARWpost/ARWpost.htm

- **Compile** ***(similar to WPS)***
  ./configure & ./compile
- Edit *namelist* and run *ARWpost.exe*
- Output is *.dat* and *.ctl* files

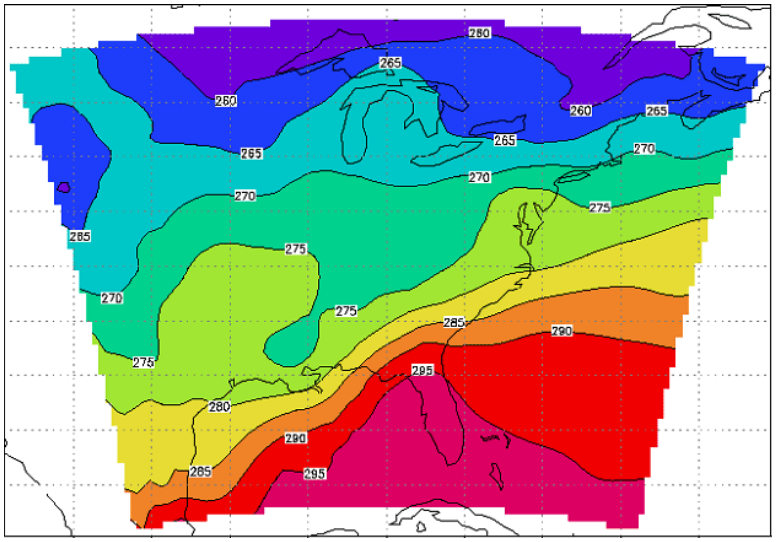
- **For GrADS output**
  - GrADS libraries only needed to display data *(freely available)*
  - http://grads.iges.org/grads/grads.html



GrADS specific notes

- **To display images**
  - Requires GrADS software
  - Freely available from: http://grads.iges.org/grads/grads.html
  - Documentation: http://grads.iges.org/grads/gadoc/index.html

- **Projection**
  - Data is plotted on model projection

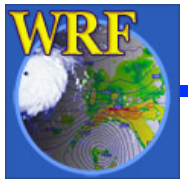


GrADS - .ctl file

```
dset ^test.dat
options byteswapped
undef 1.e37
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
```

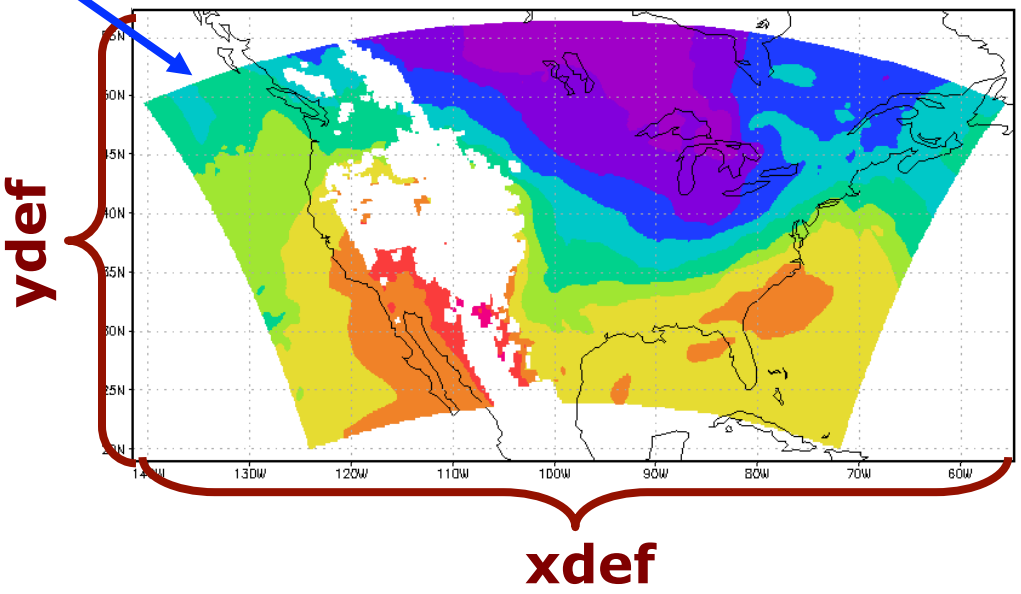
options byteswapped

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



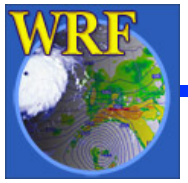
GrADS - .ctl file

```
dset ^test.dat
options byteswapped
title  OUTPUT FROM WRF V3.2 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
```



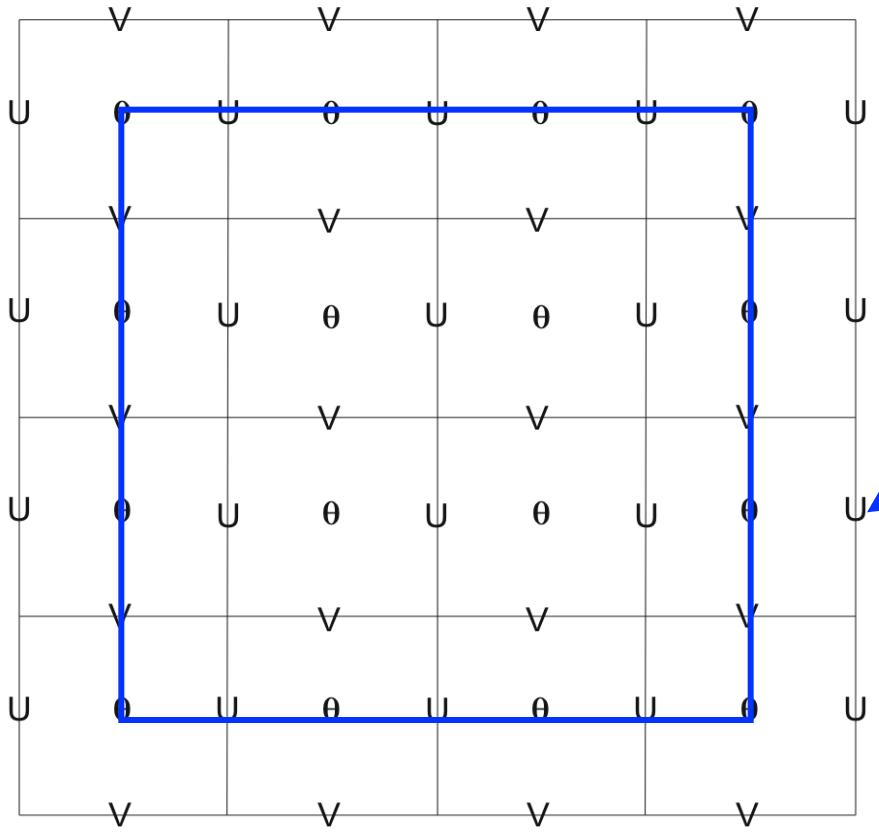
GrADS conversion - question

- **Why is a converter needed if GrADS can display netCDF files?**
  - Can only display model surface coordinates
  - Cannot interpolate to height or pressure levels
  - All diagnostics must be added via GrADS script files
  - GRIB1 model output can also be read directly by GrADS, but above issues are still valid
  - For GRIB1, there is also a stagger problem



GrADS conversion - question

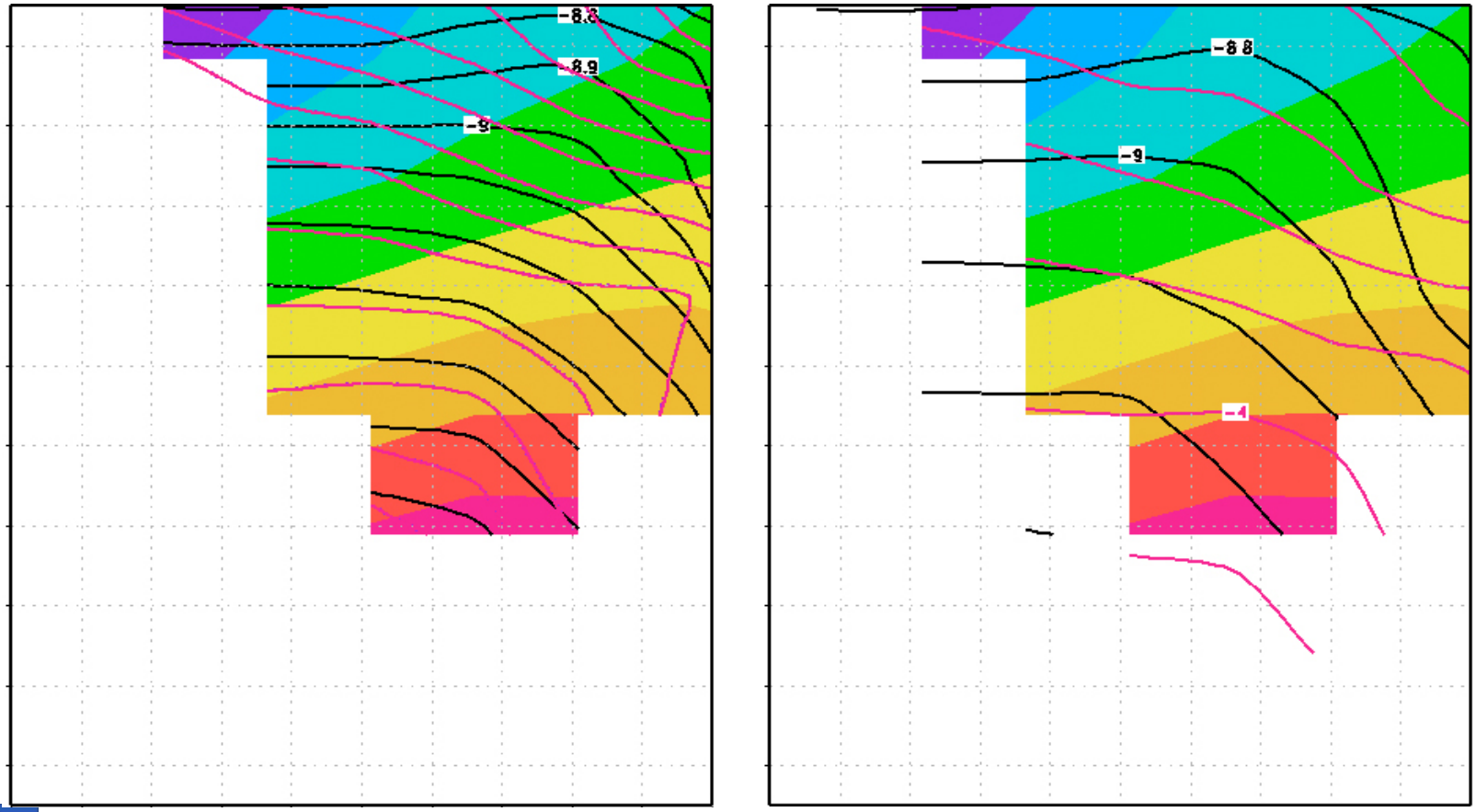
- Why is a converter needed if GrADS can display netCDF files?



WRF staggered grid

Staggering

shaded=T ; black=U ; red=V

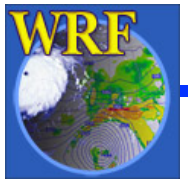


VAPOR

Computational and Information Systems Laboratory
National Center for Atmospheric Research

Visualization and Analysis Platform for Oceanic, atmospheric and solar Research

Alan Norton
alan@ucar.edu
vapor@ucar.edu
National Center for Atmospheric Research



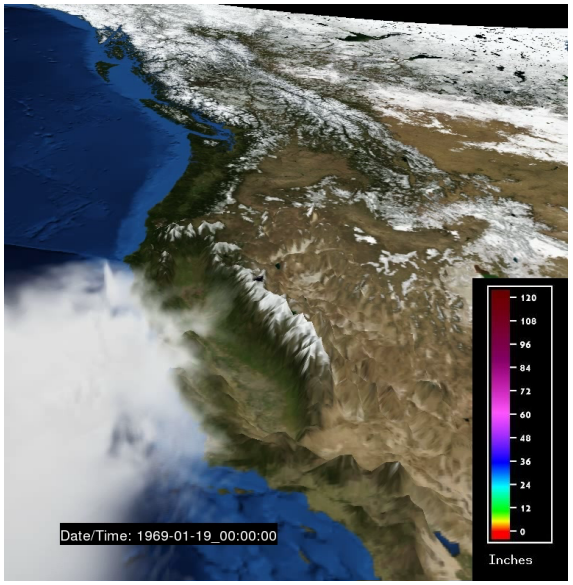
WRF in VAPOR

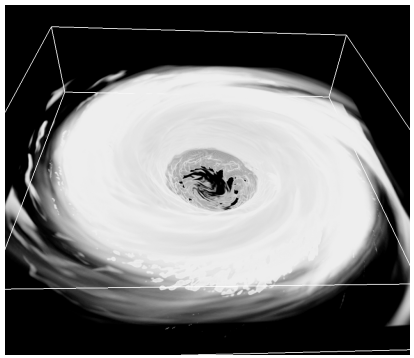
Computational and Information Systems Laboratory
National Center for Atmospheric Research

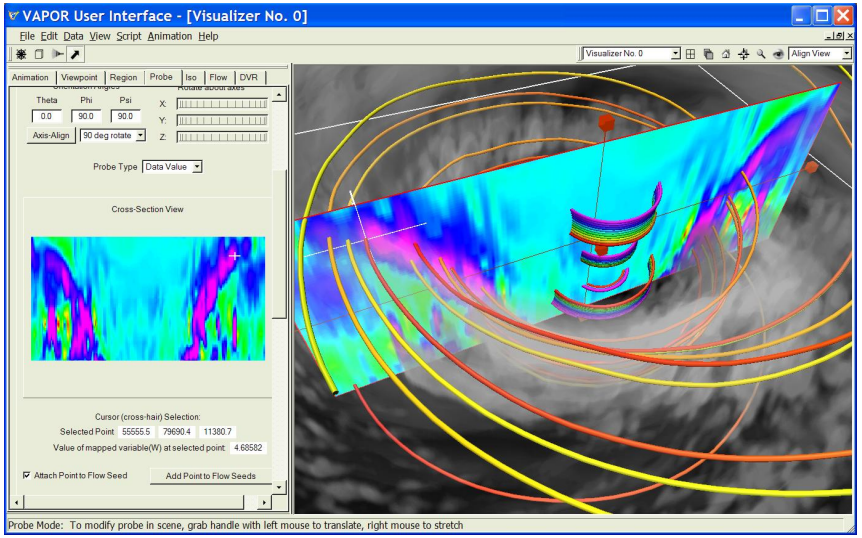
- Interactive 3D visualization of WRF-ARW data *(wrfout files only)*
- Available free on Linux, Windows, Mac
- Interactive rendering and animation (using GPU acceleration)
- Simple 2-step data conversion from WRF output to VAPOR
  - wrfvdfcreate & wrf2vdf
- Steady and unsteady flow integration
- Data probing
- Downloads, documentation, examples at: http://www.vapor.ucar.edu
- http://www.vapor.ucar.edu/doc/WRFsupport.pdf

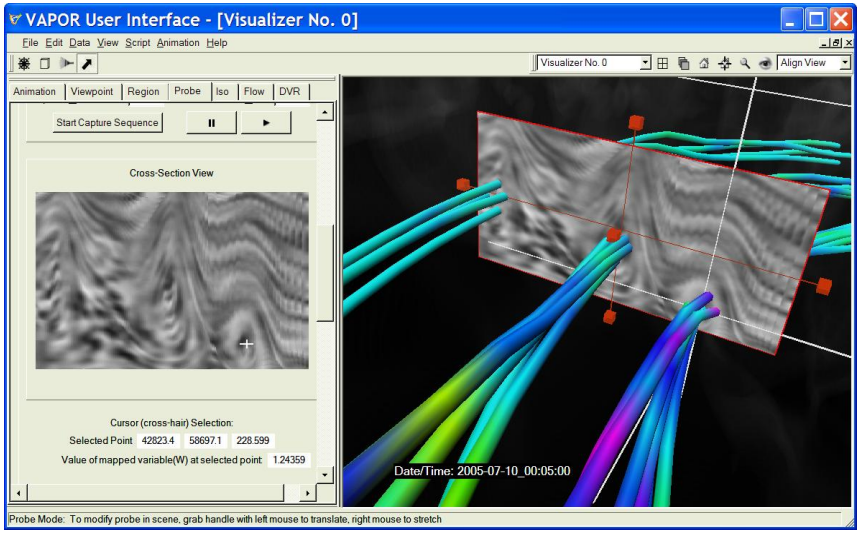
WRF in VAPOR

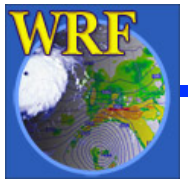
Computational and Information Systems Laboratory
National Center for Atmospheric Research











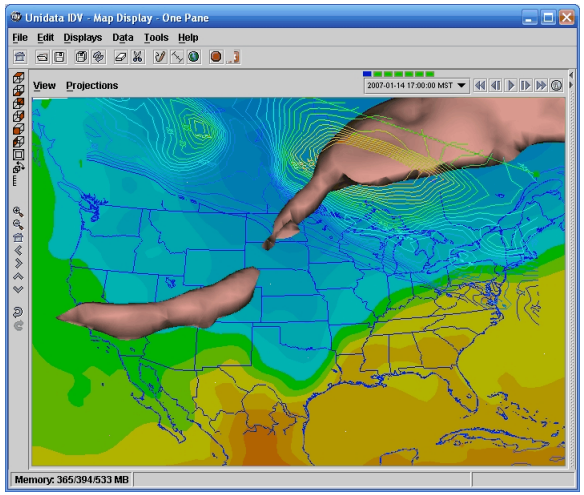
IDV

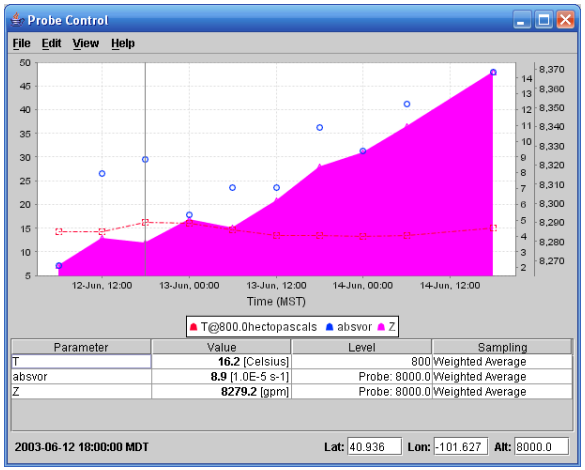
Integrated Data Viewer

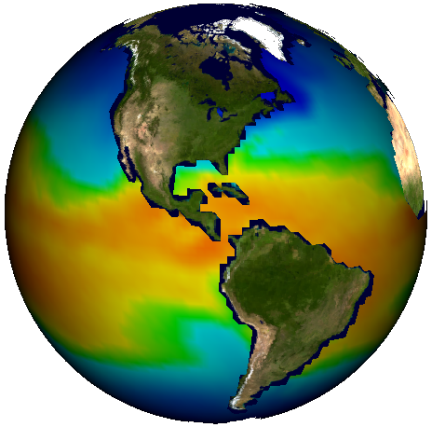
Yuan Ho and Julien Chastang
Unidata Program Center/UCAR

What is the IDV?

- Visualization and analysis tool for geoscience data developed and supported by Unidata
- Freely available Java™ framework and application
- Integrated 2D/3D displays of a wide range of data
- Built on VisAD library

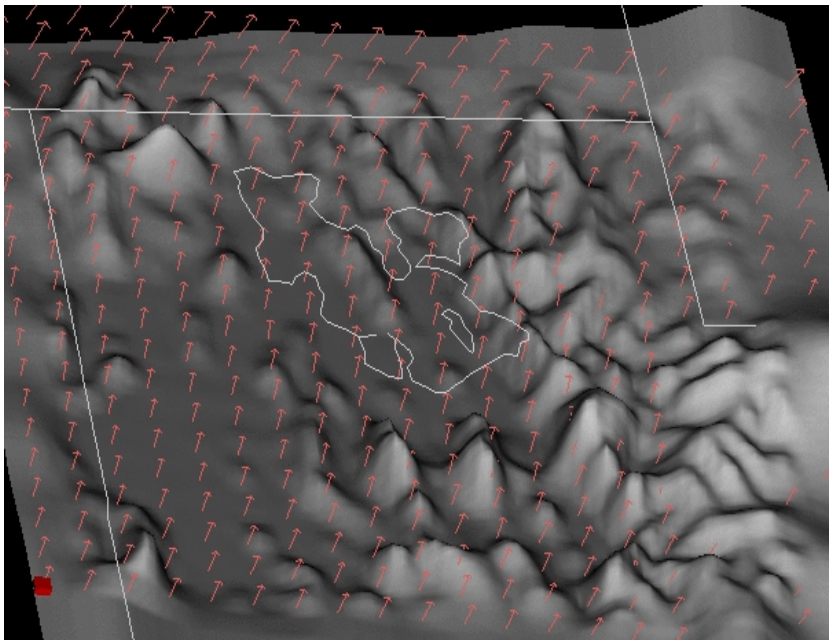






IDV Strengths

- Easy to download and install on any platform
- Remote and local access to datasets
- 2D/3D visualization
- Bundle mechanism
- Support for multi-disciplinary datasets integrated from a variety of sources
- Flexible framework supports customization (GEON-IDV, field projects, McIDAS-V)
- Extensive documentation
- Community driven development



Model simulation of wind, isentropic potential vorticity and low level moisture flow over the Great Salt Lake basin

Supported Data Sources

- **Data Types:**
  - Gridded model output
  - Satellite imagery
  - Radar data
  - Point observations
  - Balloon soundings
  - NOAA Profiler Network winds
  - Aircraft Tracks
  - Fronts
  - GIS data (WMS, shapefile)
  - Quick Time movies
  - Web Cams
- **Vertical Coordinates**
  - Pressure
  - Height/Depth
  - Other (2D only)
- **Sample of Supported Formats:**
  - netCDF
  - GRIB
  - Vis5D
  - KML
  - CSV
  - GEMPAK grid
  - ADDE
- **Access Methods:**
  - Local files
  - HTTP
  - ADDE, TDS and OPeNDAP servers
  - WMS

ADDE = Abstract Data Distribution Environment
TDS (THREDDS) = Thematic Realtime Environmental Distributed Data Services

For Further Information for IDV

- Integrated Data Viewer homepage
  - http://www.unidata.ucar.edu/software/idv
- RAMADDA homepage
  - http://www.unidata.ucar.edu/software/ramadda/
- VisAD homepage
  - http://www.ssec.wisc.edu/~billh/visad.html
- All IDV questions/comments
  - support-idv@unidata.ucar.edu