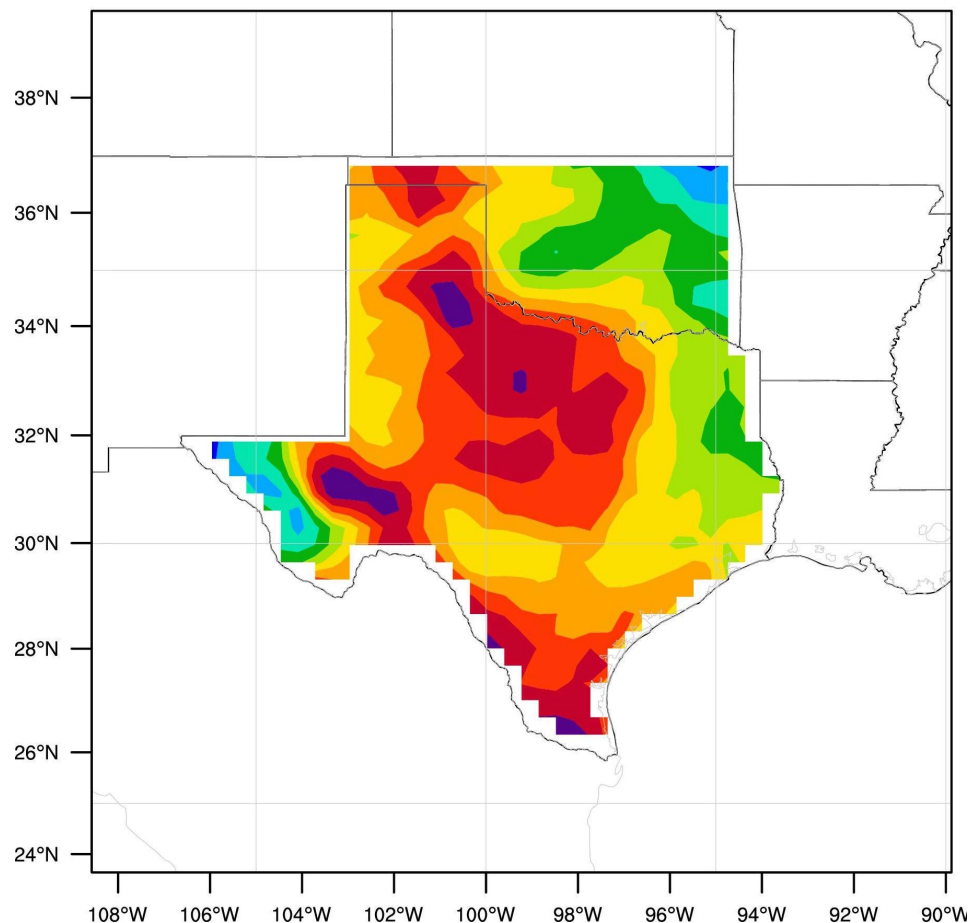




Post-Processing Tools: NCL

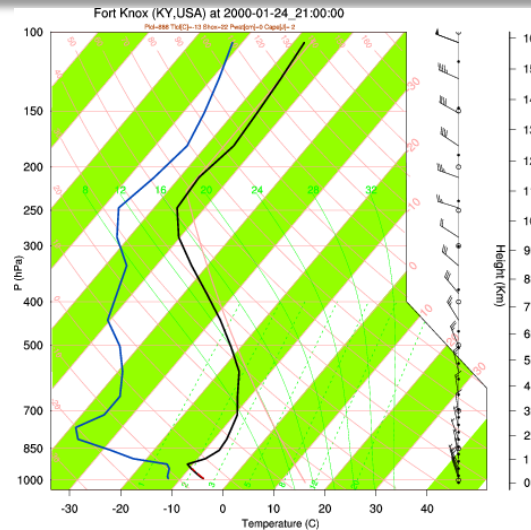
Abby Jaye

January 2020

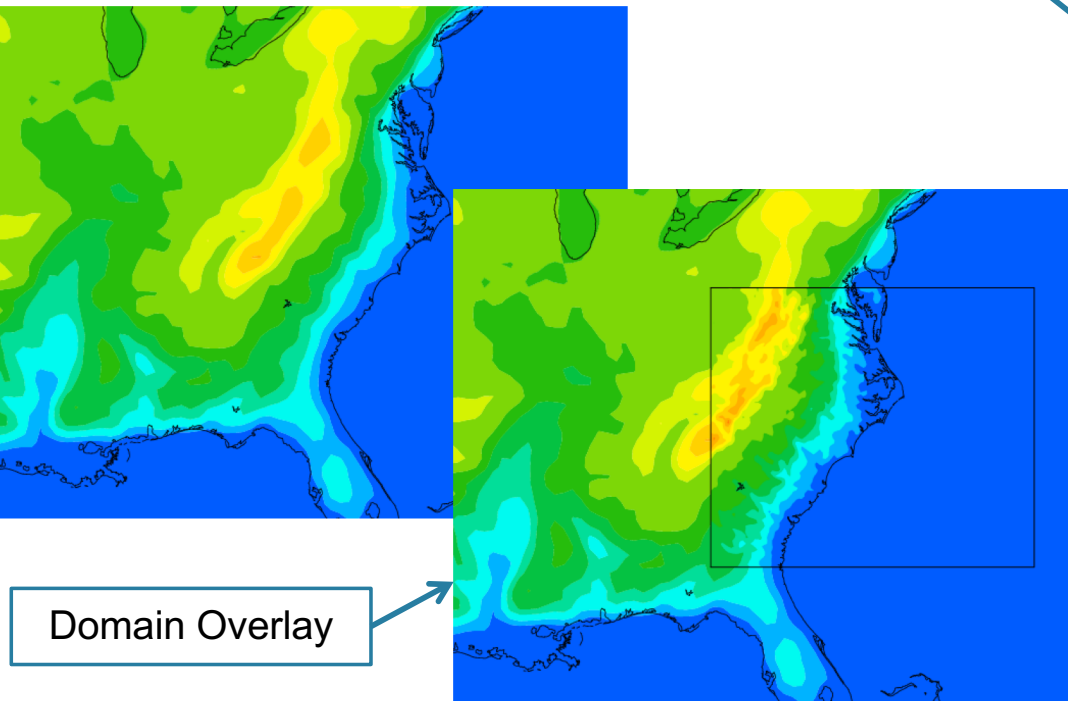
NCL

- 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

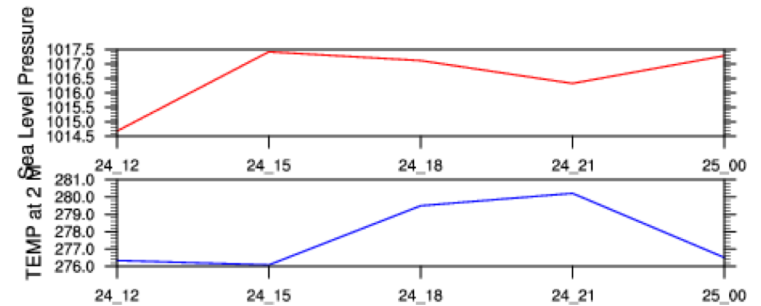
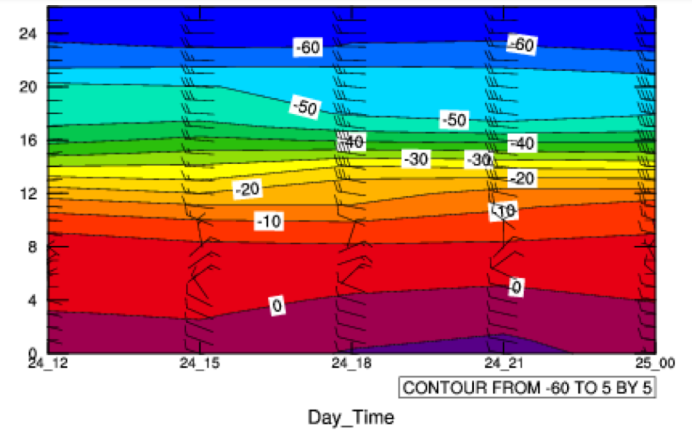


Skew-T Diagram



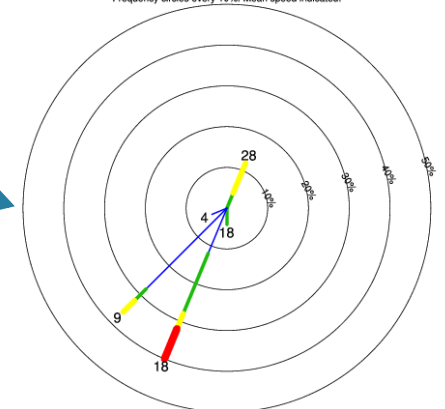
Domain Overlay

Time Series



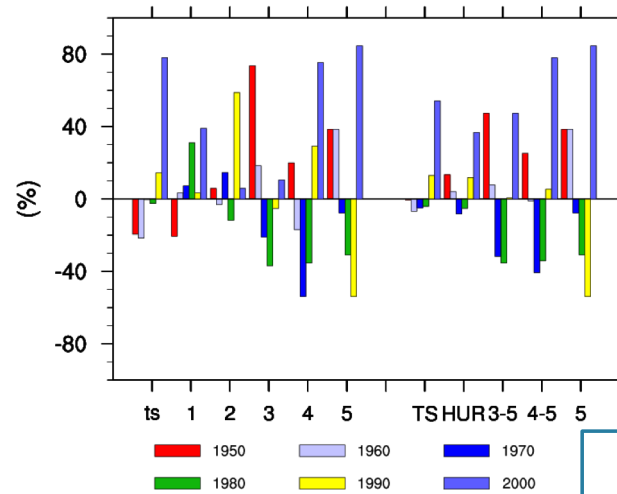
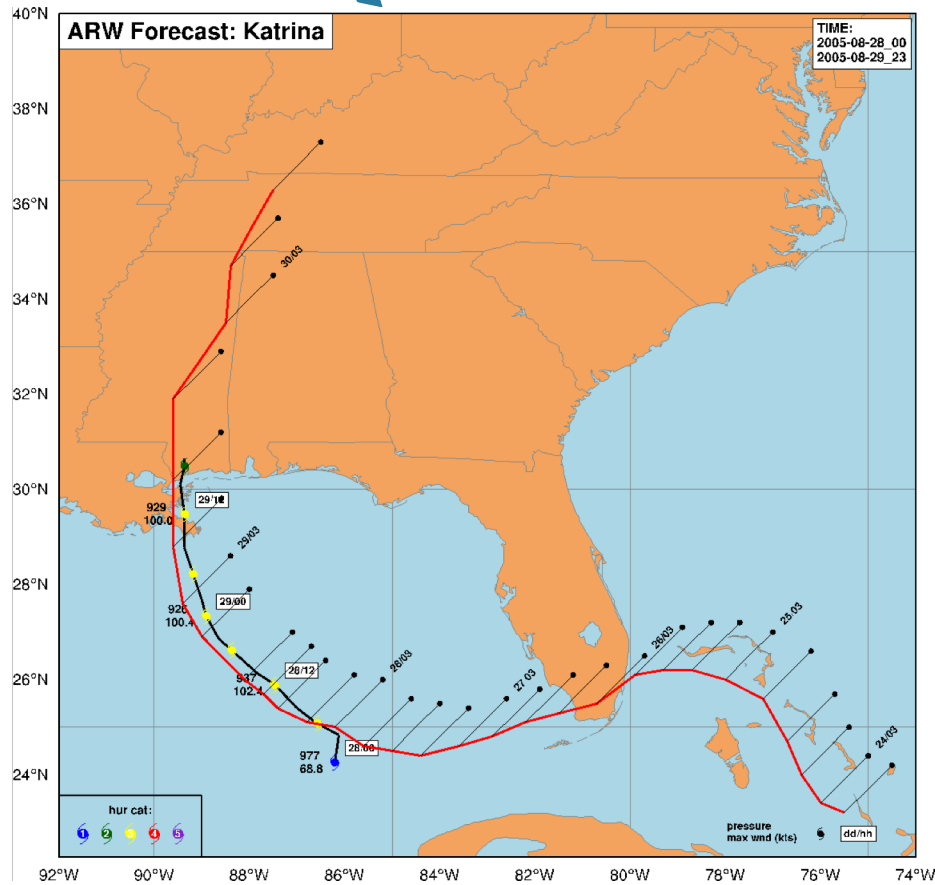
WRF: All Times: grid point [25.65 , -87.37]
SpdAve=16 SpdStd=11 DirAve=216 No Calm Reports Nwnd=25
Frequency circles every 10%. Mean speed indicator.

Wind Rose

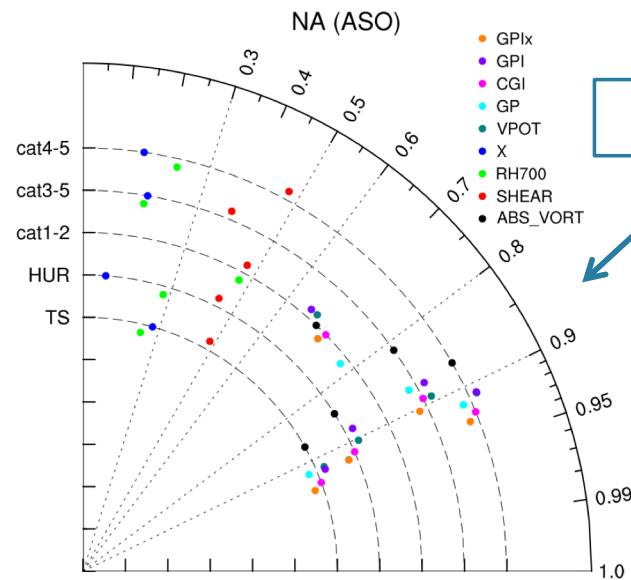


Example Plots

Storm Track

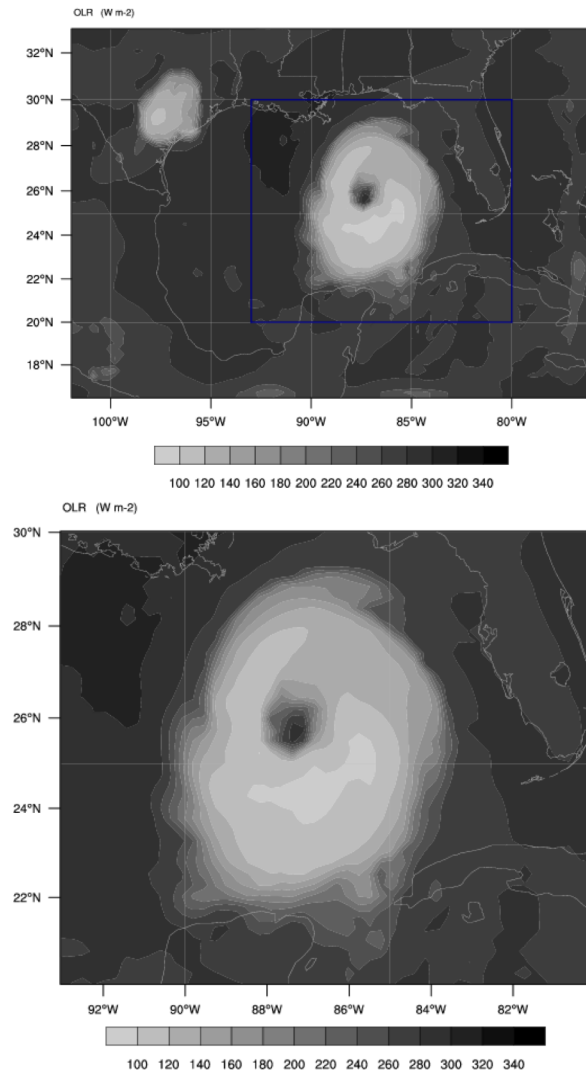


Taylor Diagram



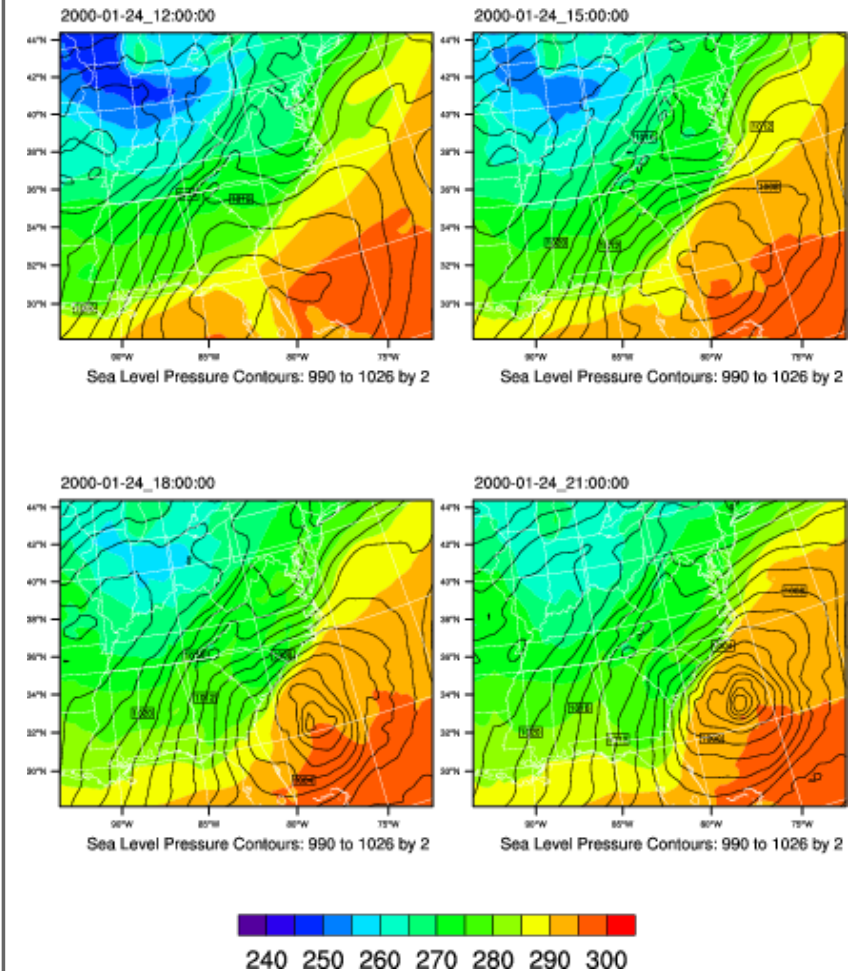
Example Plot Functions

Zooming



Panel Plots

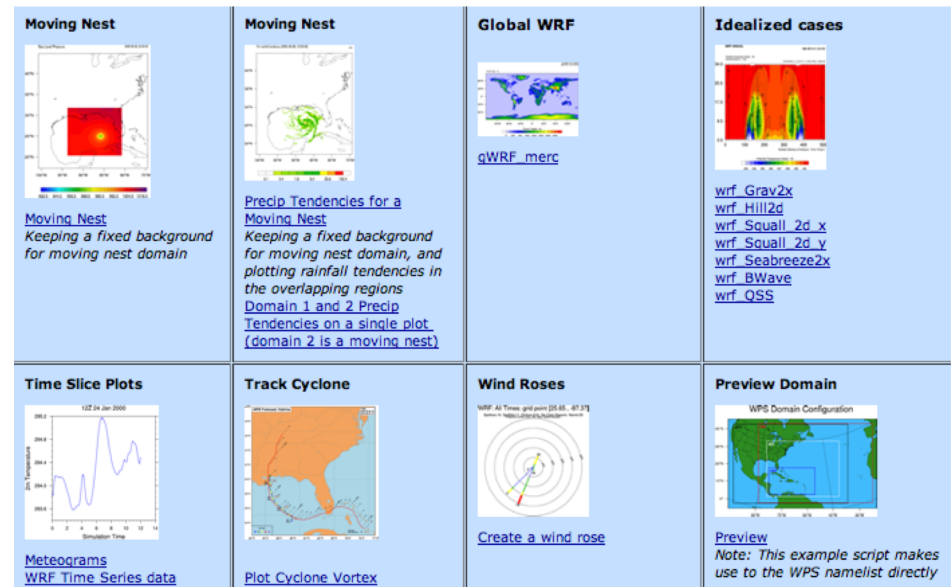
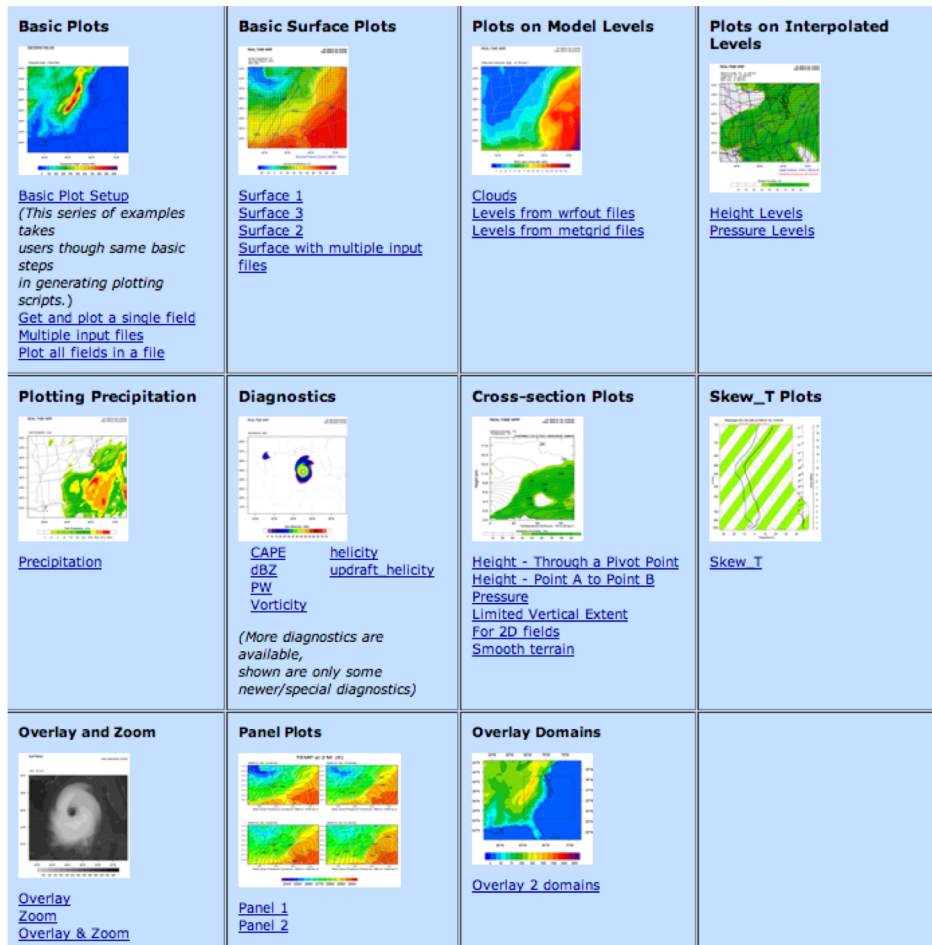
TEMP at 2 M (K)



Generating Plots

A good start: WRF Online Tutorial

http://www2.mmm.ucar.edu/wrf/OnLineTutorial/Graphics/NCL/NCL_examples.php



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: v6.6.2)

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

NCL & WRF

User Modifiable

NCL
libraries

Scripts
(*WRFUserARW.ncl*)

WRF
Specific
Diagnostics
(*like slp*)

Maintained/supported WRF Dev Group

~/.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



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/WRFUserA
RW.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
```

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

```
; 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
```



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 mcaps/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 a

```
load "$NCARG_ROOT/lib/ncarg
load "$NCARG_ROOT/lib/ncarg

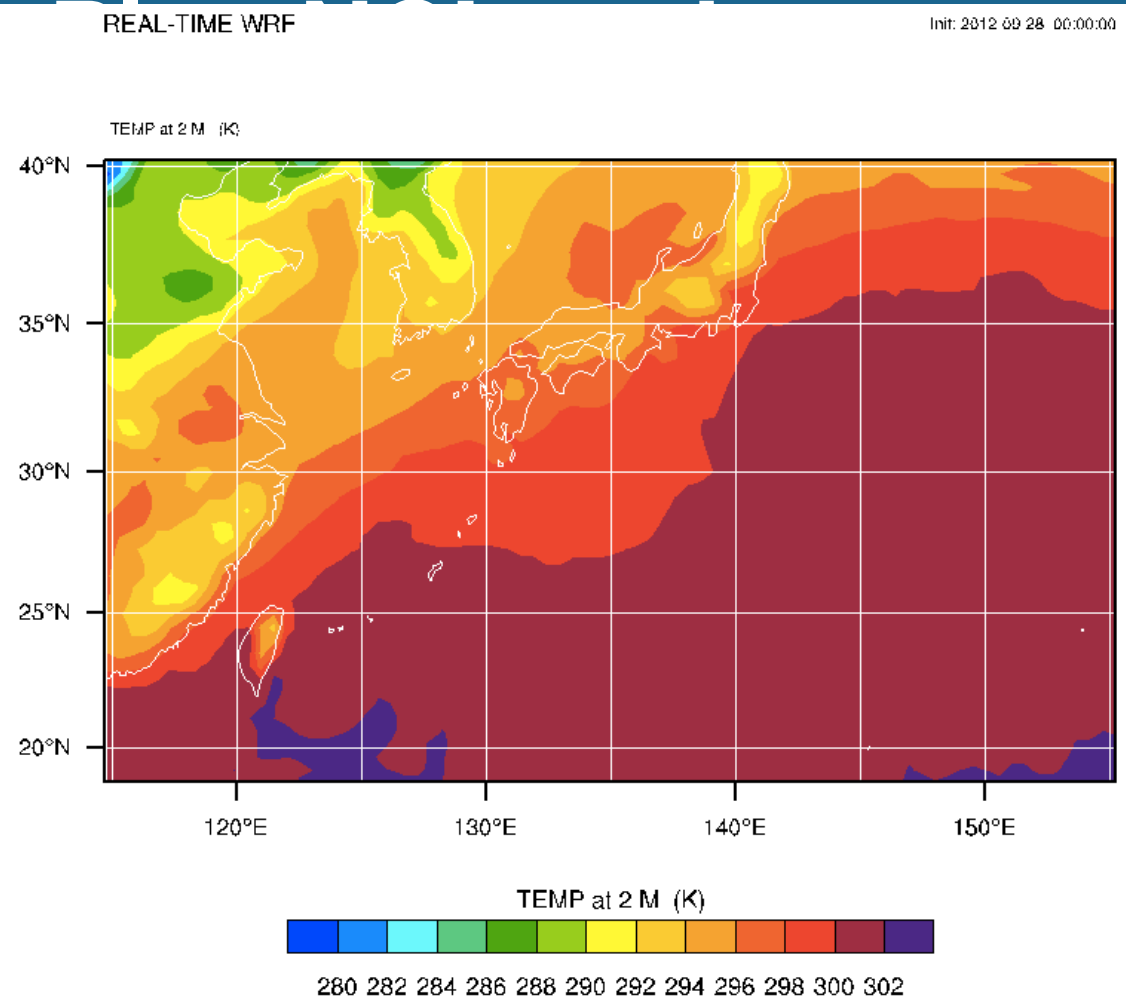
begin

a = addfile("./wrfout_d
wks = gsn_open_wks("X11",

T2 = wrf_user_getvar(a,

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,
```

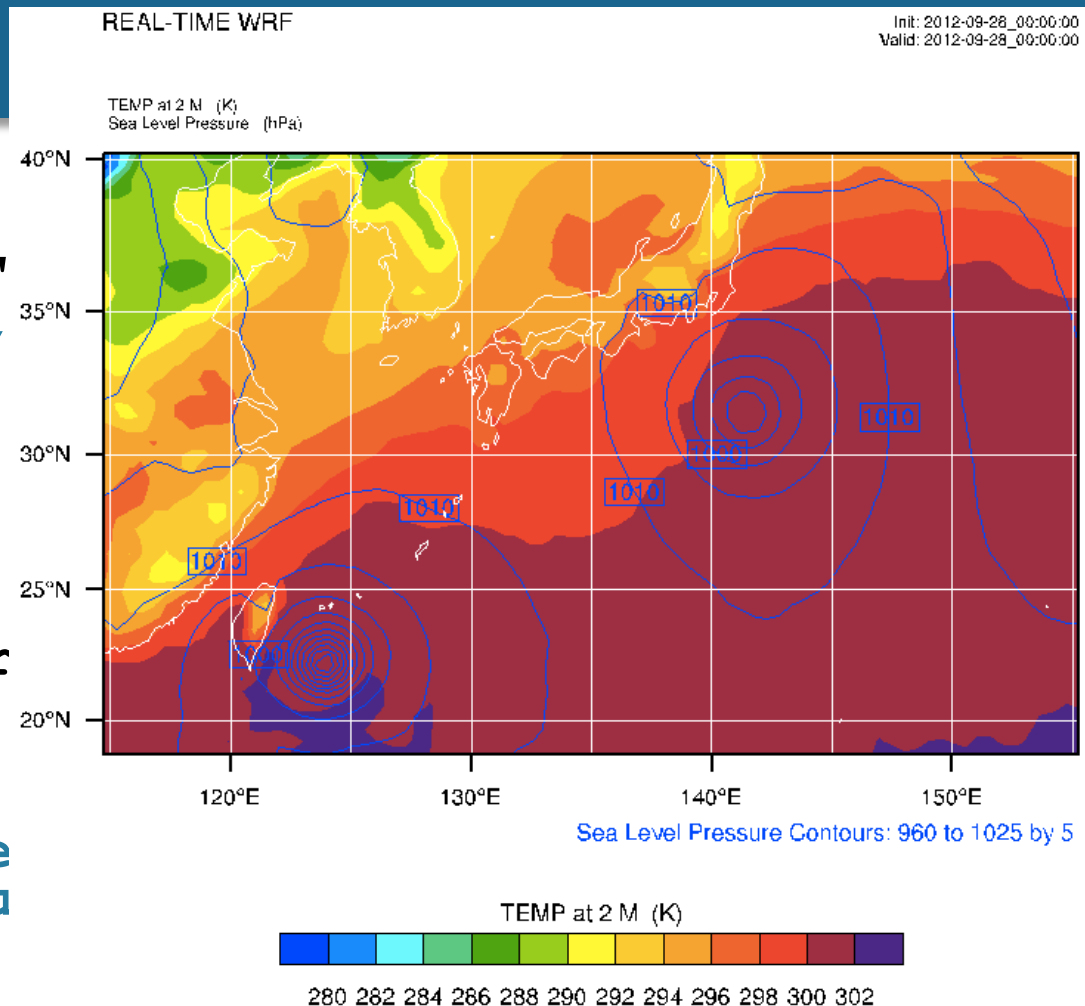
```
pltres = True  
mpres = True
```

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

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

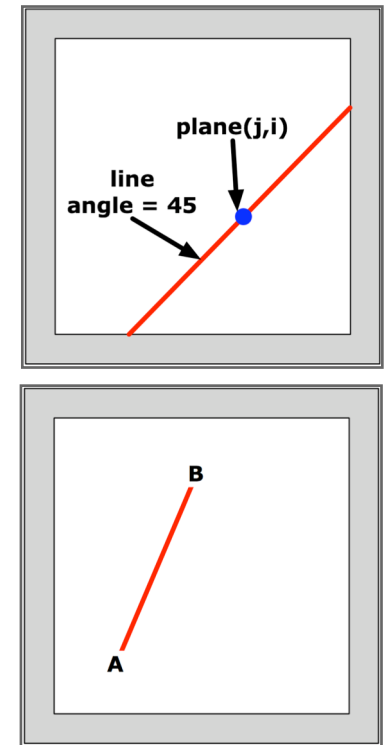
```
plot = wrf_map_overlays(a,wks, (/contour_t2,contour_slp/),  
pltres,mpres)
```

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`
Unstagger 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_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
```

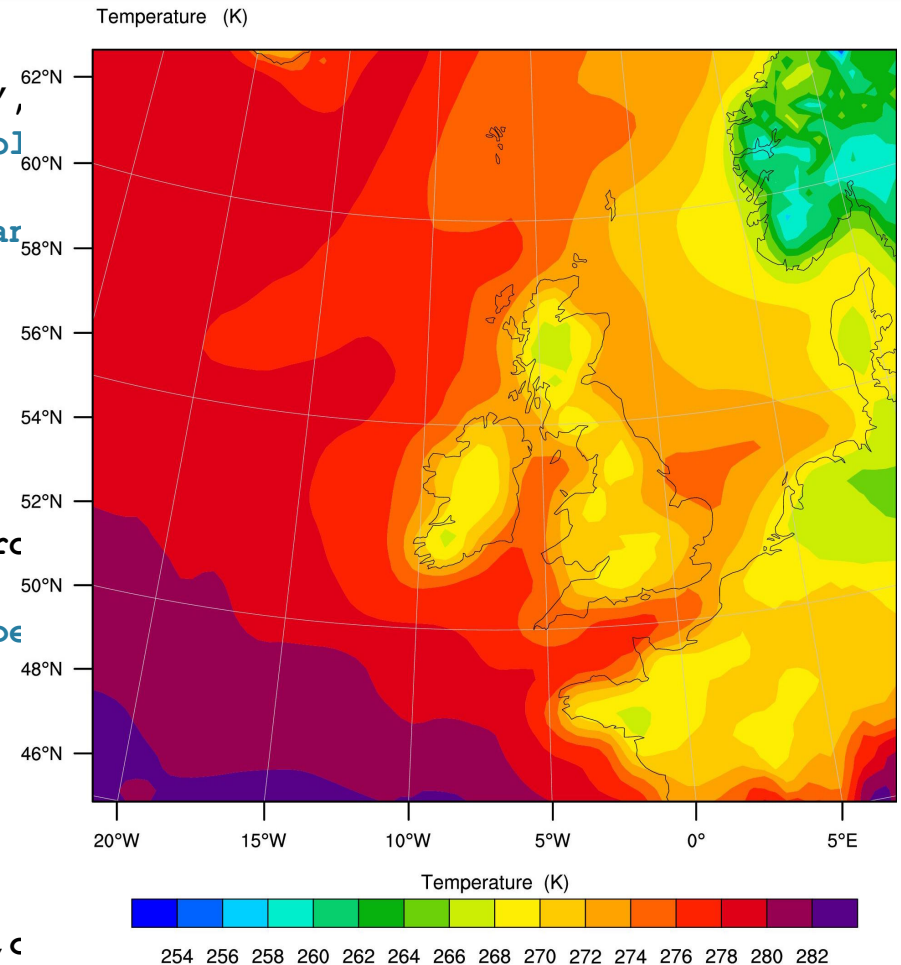
wrf_user_vert_interp

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

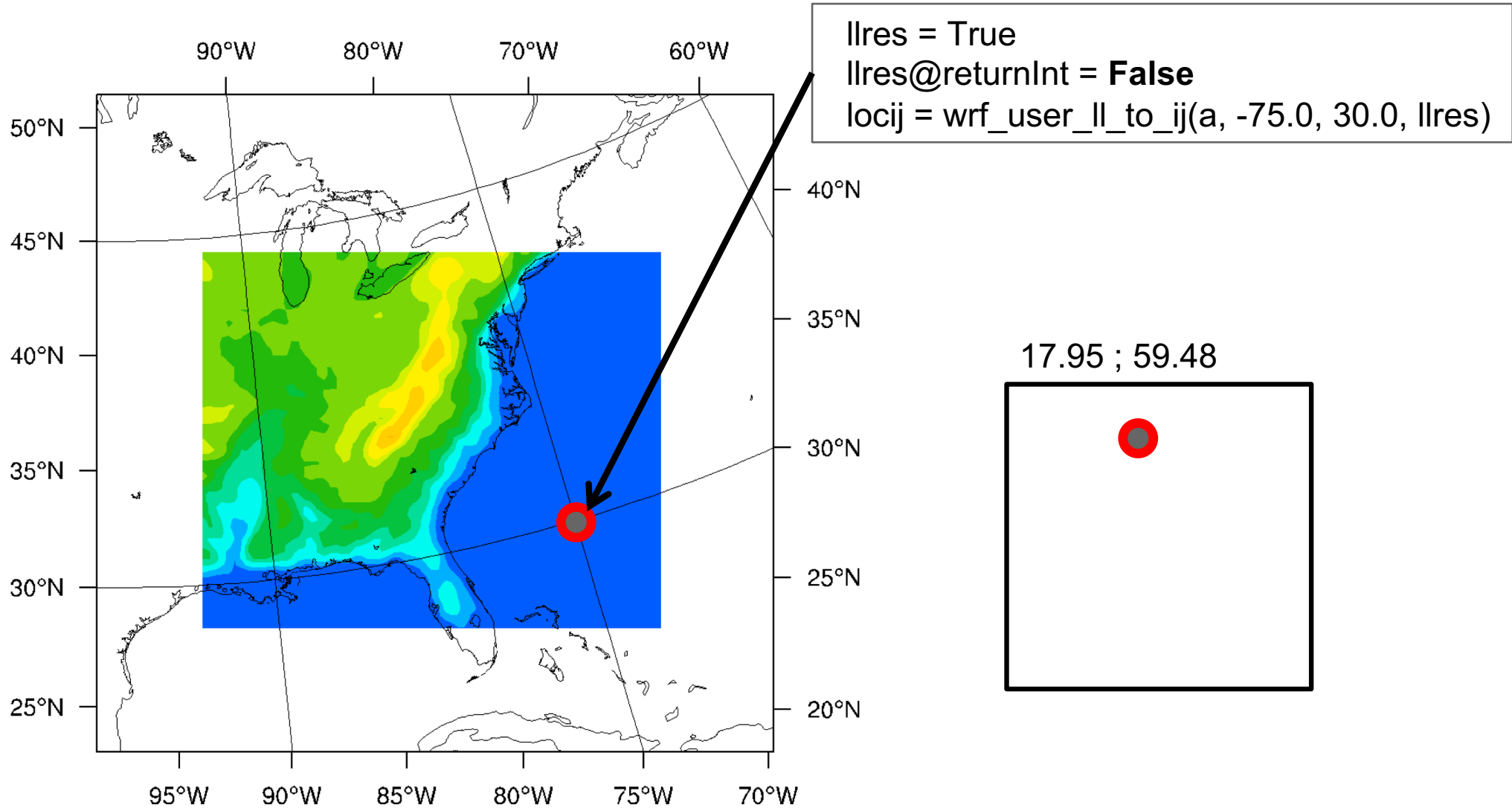
  vert_coord = "pressure" ; Set the surface we want
  interp_levels = (/200,300,500,1000/)

  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)

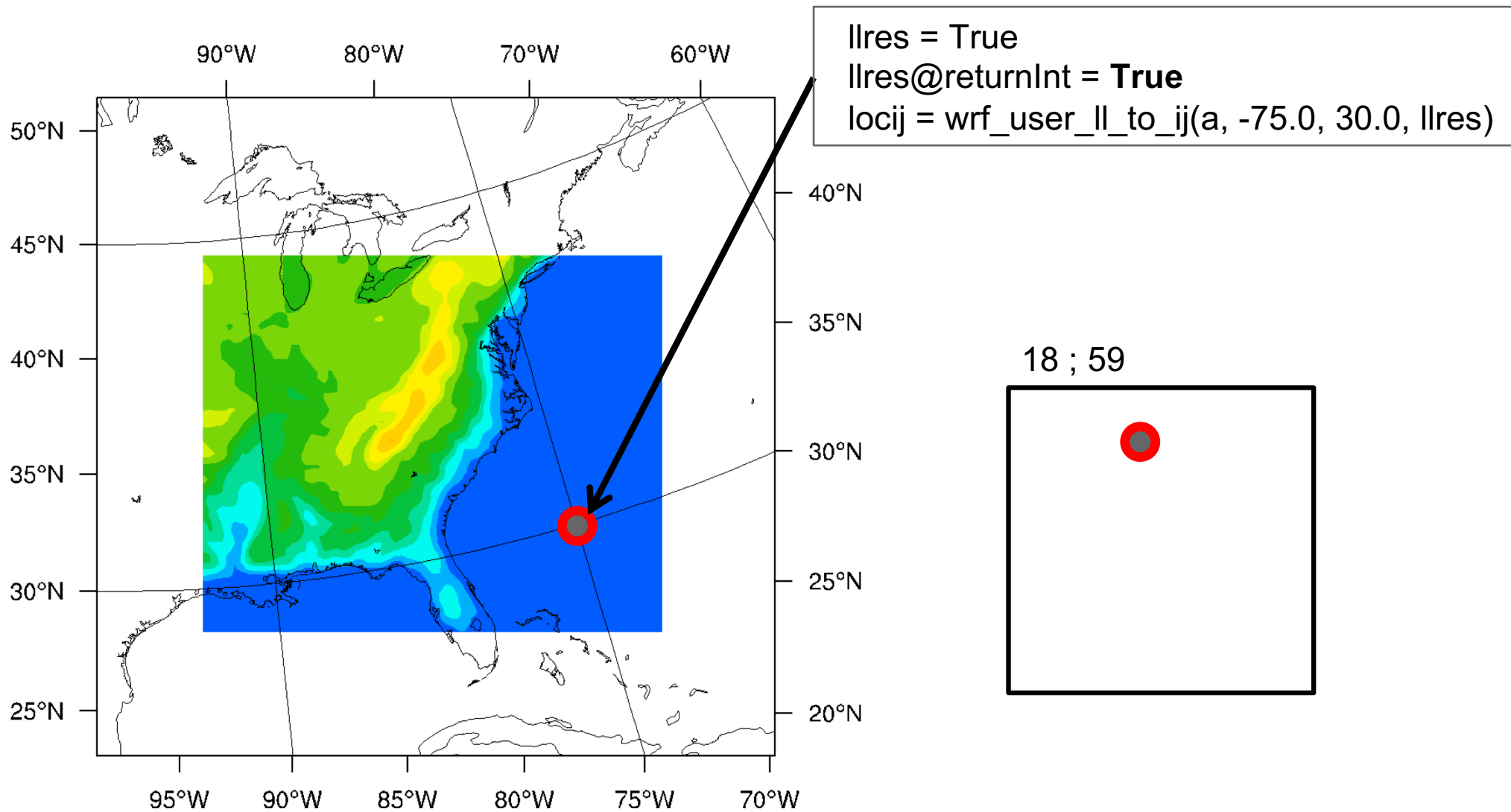
  wks = gsn_open_wks("X11","plot_tk_1000mb") ; open a window
  opts2 = True ; Set options for the plot
  opts2@cnFillOn = True
  pltres = True
  mpres = True
  mpres@mpGeophysicalLineColor = "Black"
  ; Make the contour and plot it over a map
  contour = wrf_contour(a,wks,tk_interp(0,3,:,:),c)
  plot =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



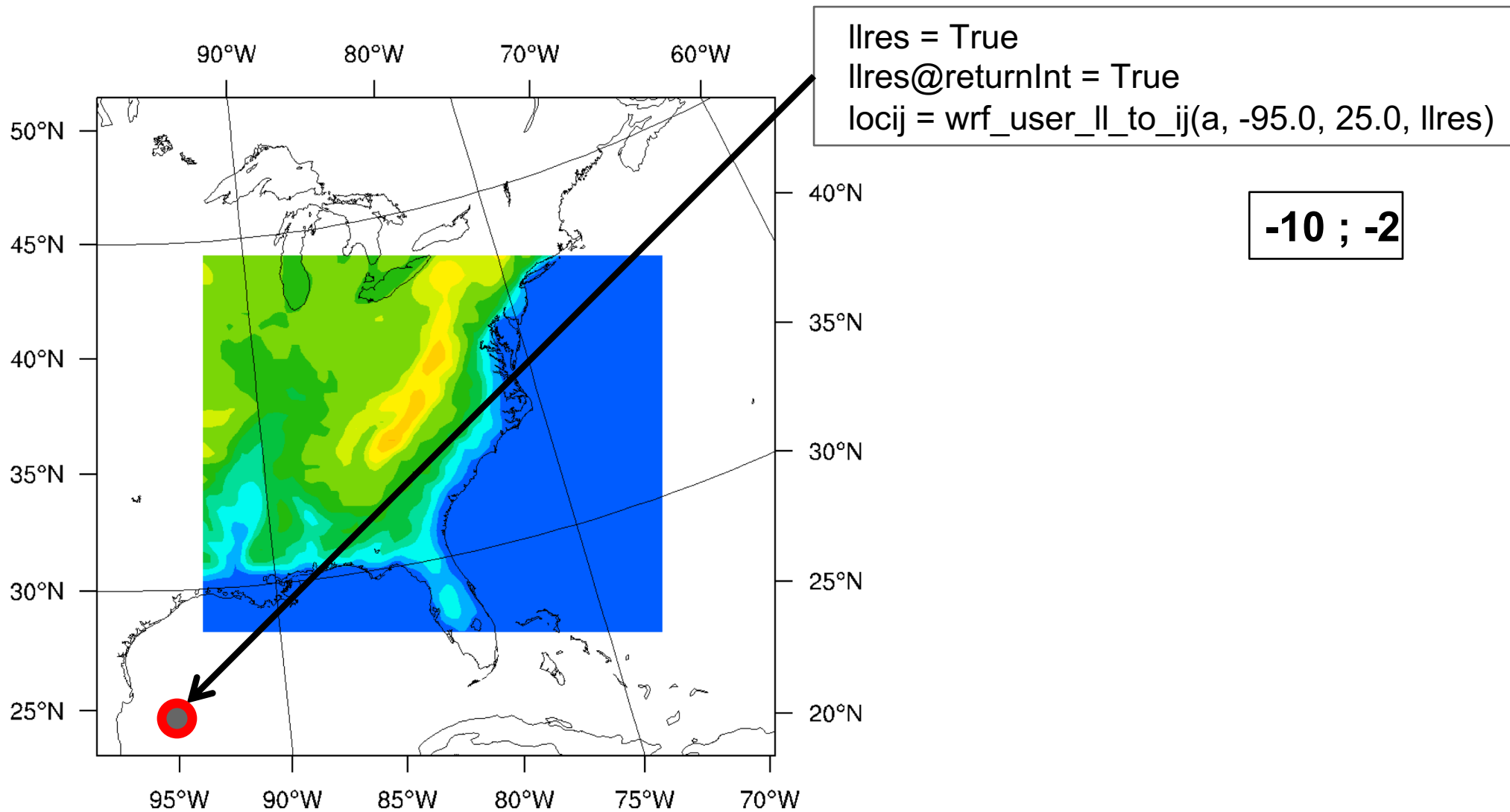
wrf_user_ll_to_ij



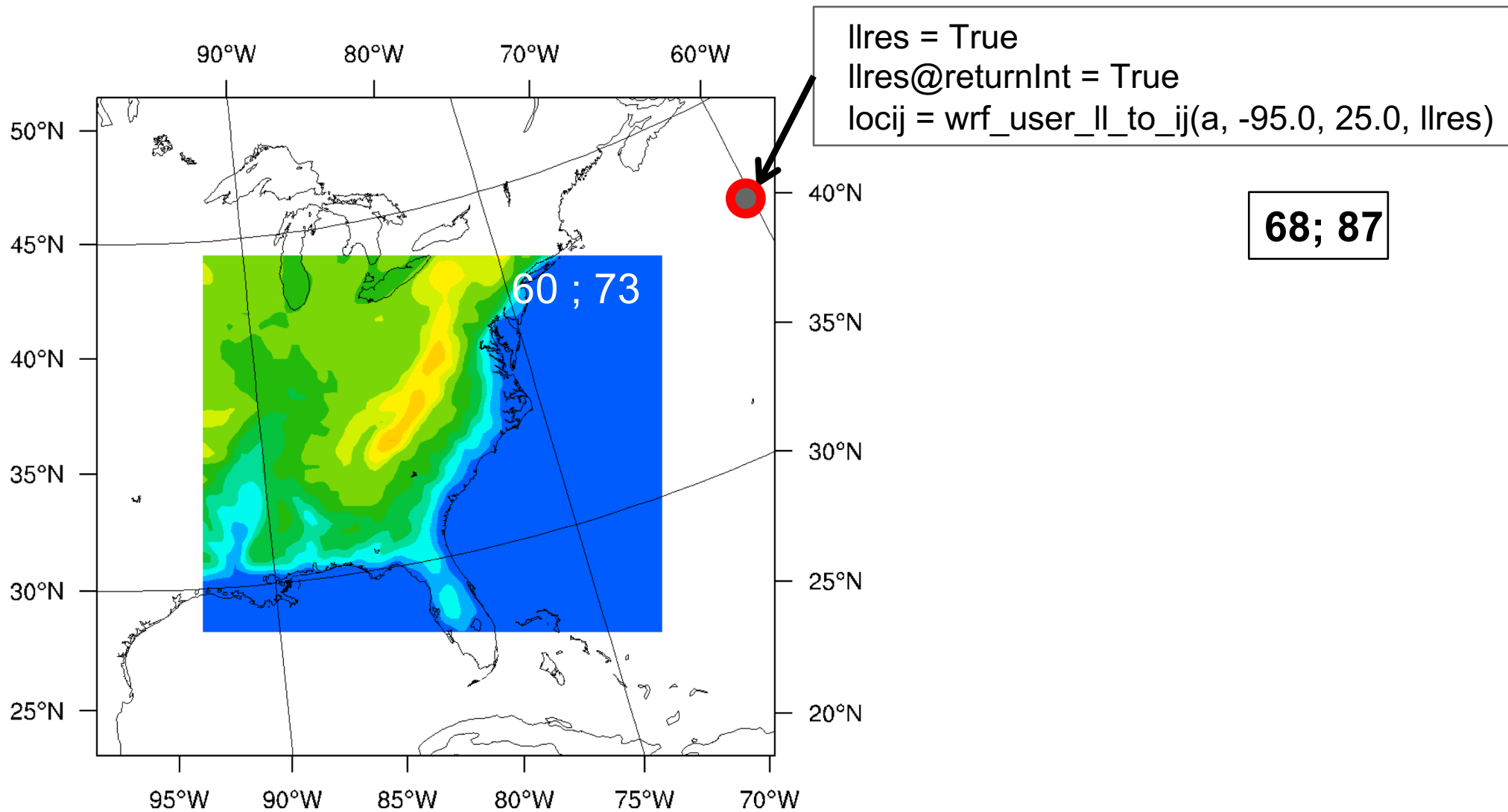
wrf_user_ll_to_ij



wrf_user_ll_to_ij



wrf_user_ll_to_ij

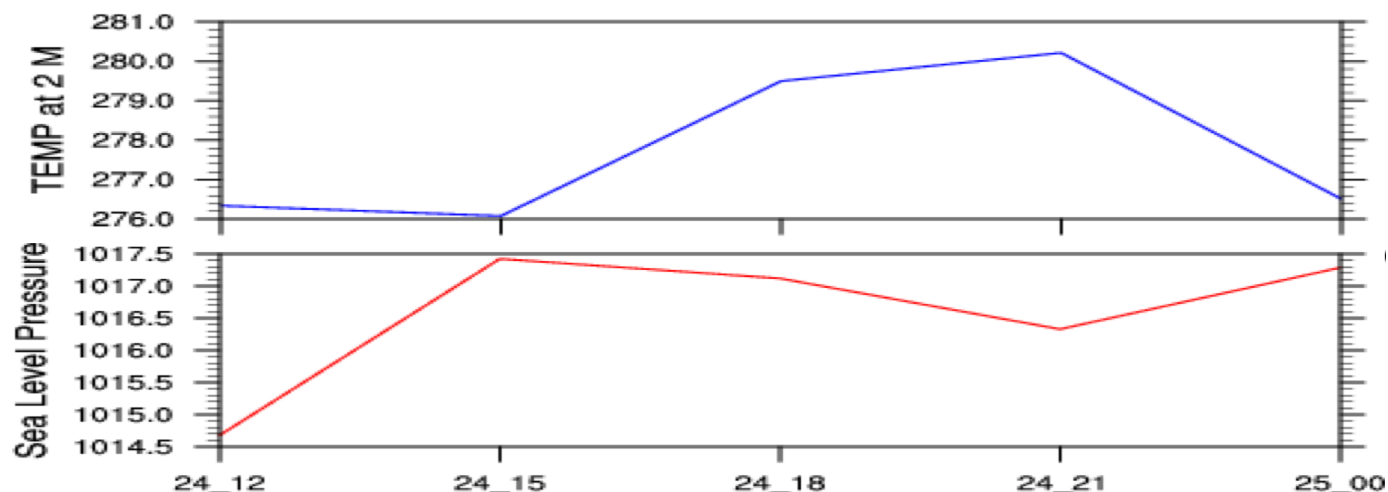


Time Series

```
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)
```

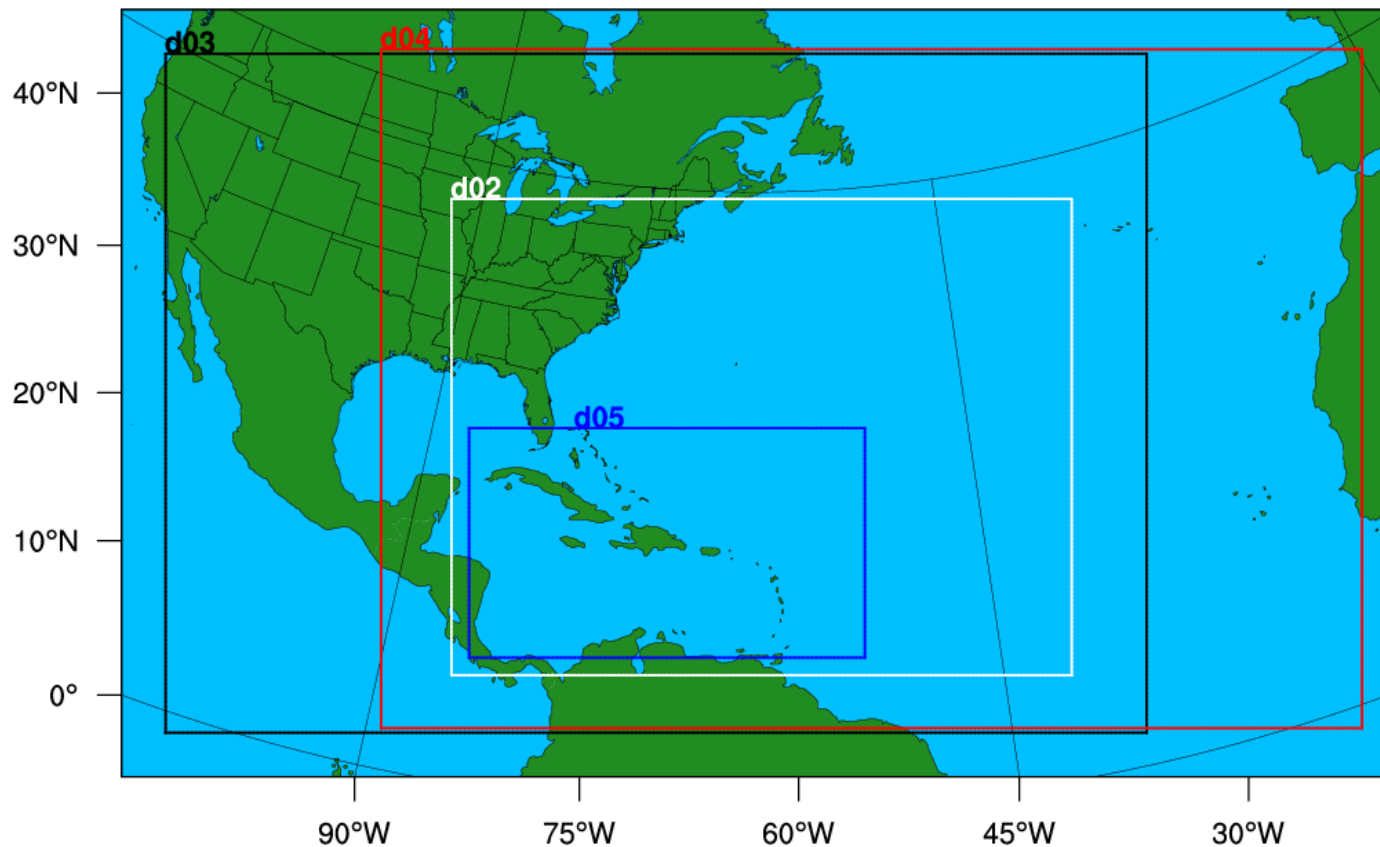
```
slp      = wrf_user_getvar(a,"slp",-1)
slp_point = slp(:,locY,locX)
slp_plot  = gsn_csm_xy(wks,taus,slp_point,t2_res)
```



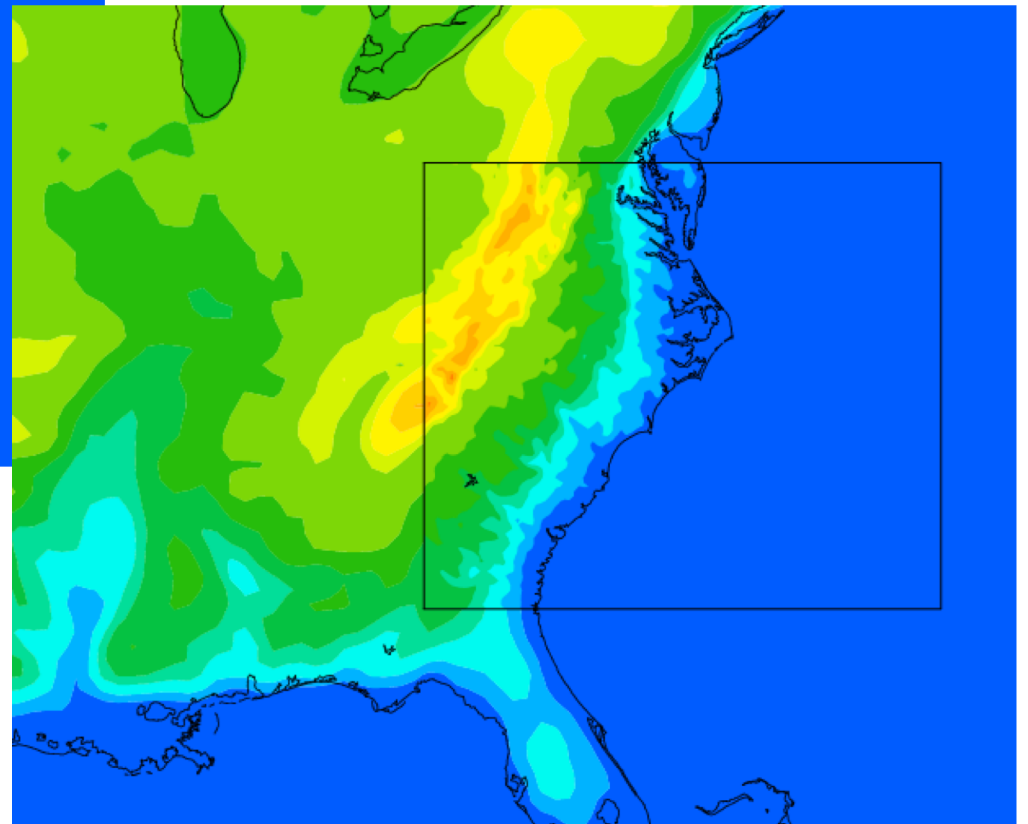
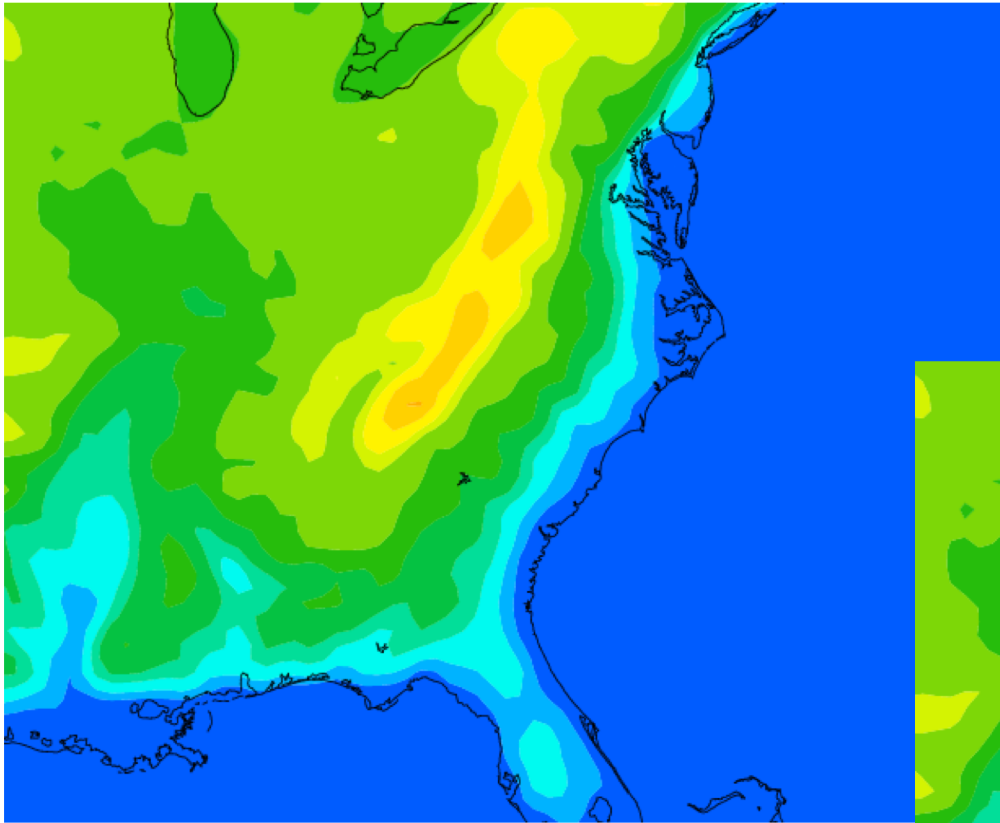
Domain Design

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

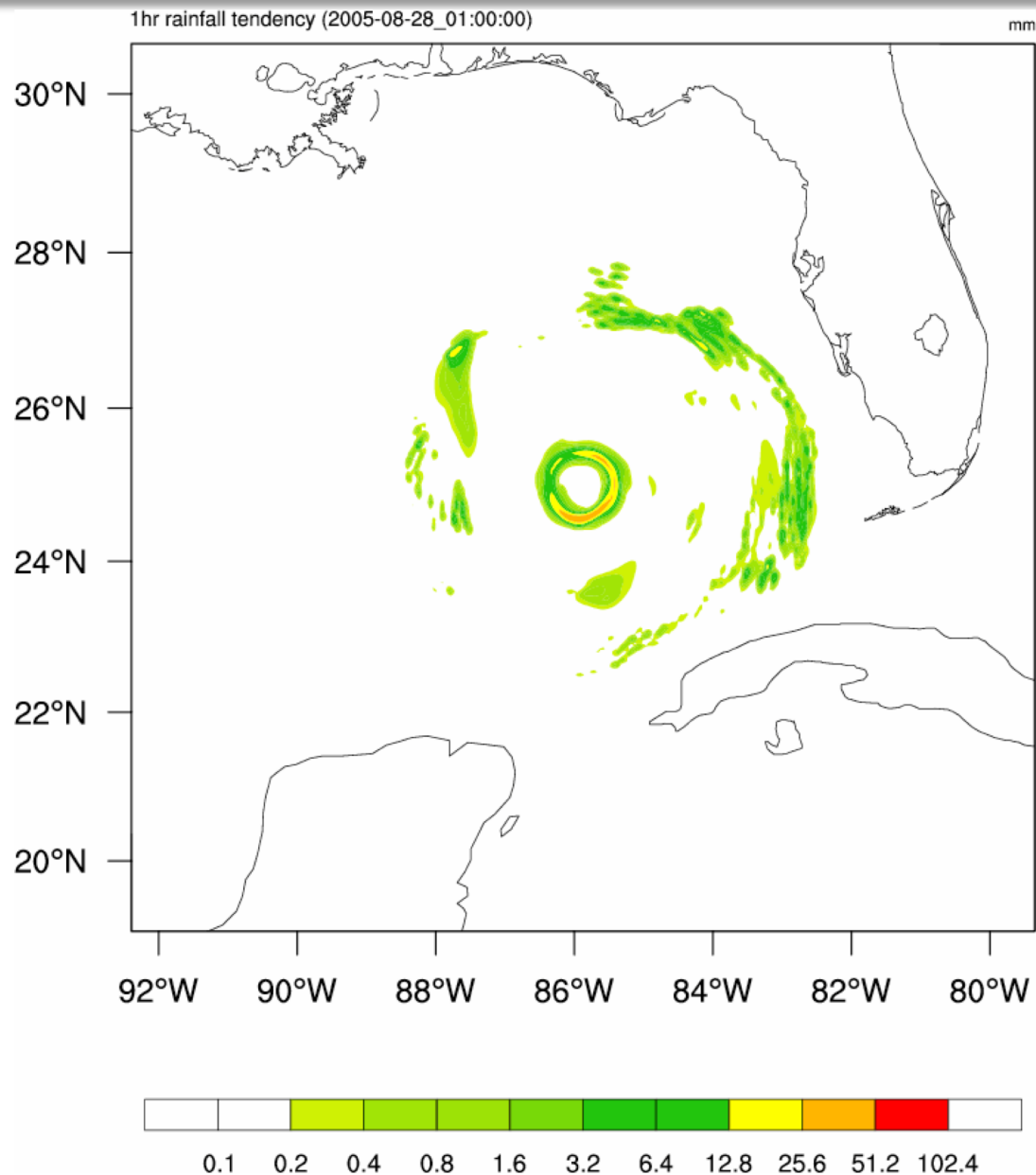
Test Domain



Overlay Domains



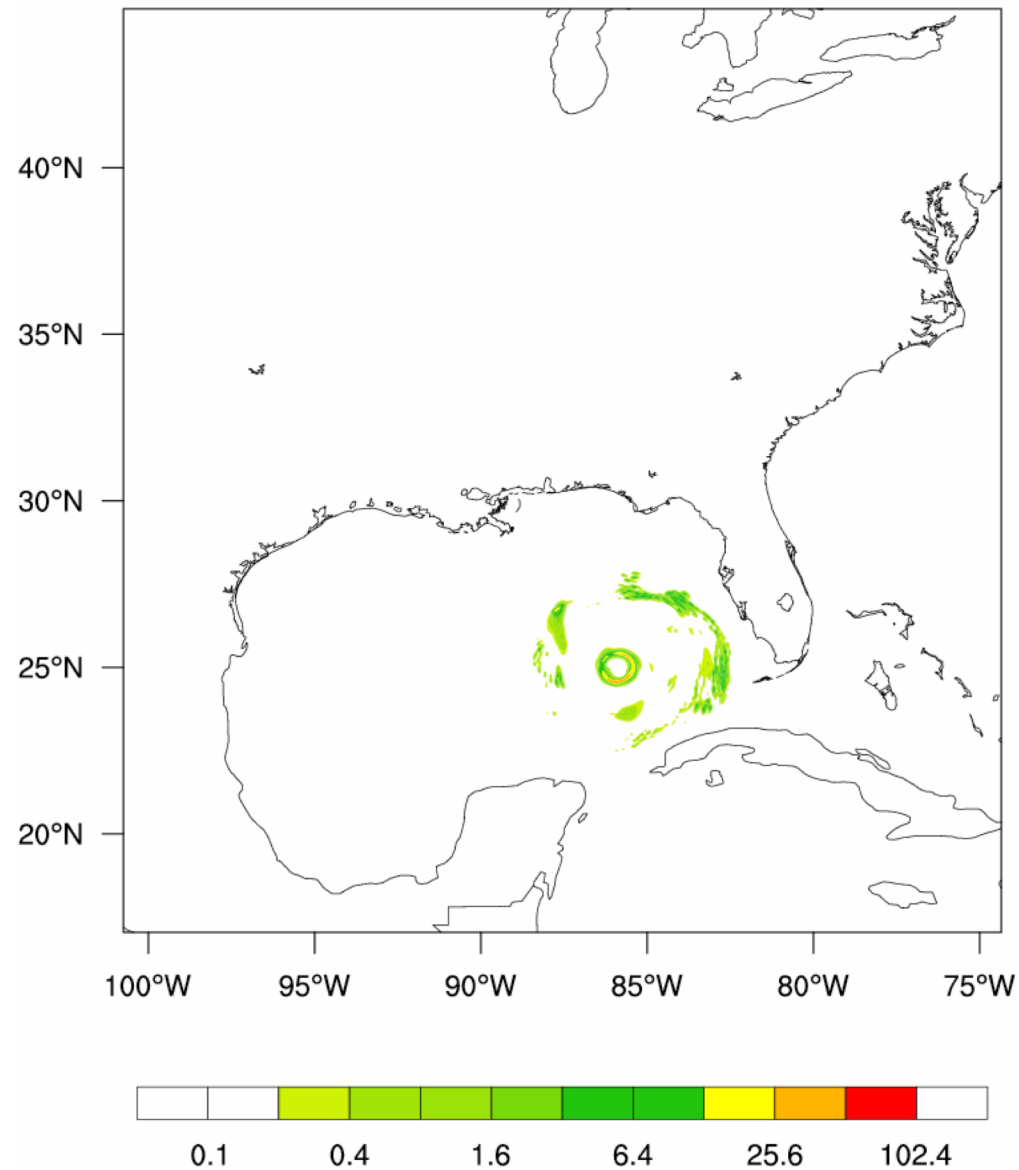
Moving Nests



Moving Nests

1hr rainfall tendency (2005-08-28_01:00:00)

mm



Change Fields in a 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
```

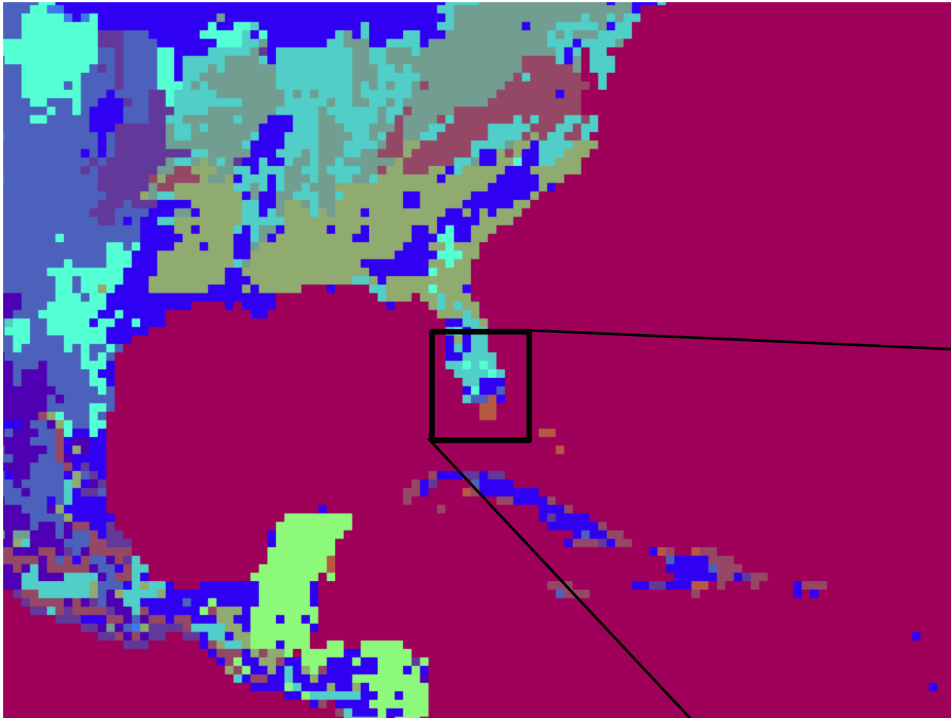
```
        sst = sst + 1     ; change the entire field
```

```
        a->SST = sst      ; write the field back to the file
```

```
    end do
```

```
end
```

Change Fields in a 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
```

									55
									56
									57
									58
									59
									60
									61
									62
									63
									64
									65
78	79	80	81	82	83	84	85	86	

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

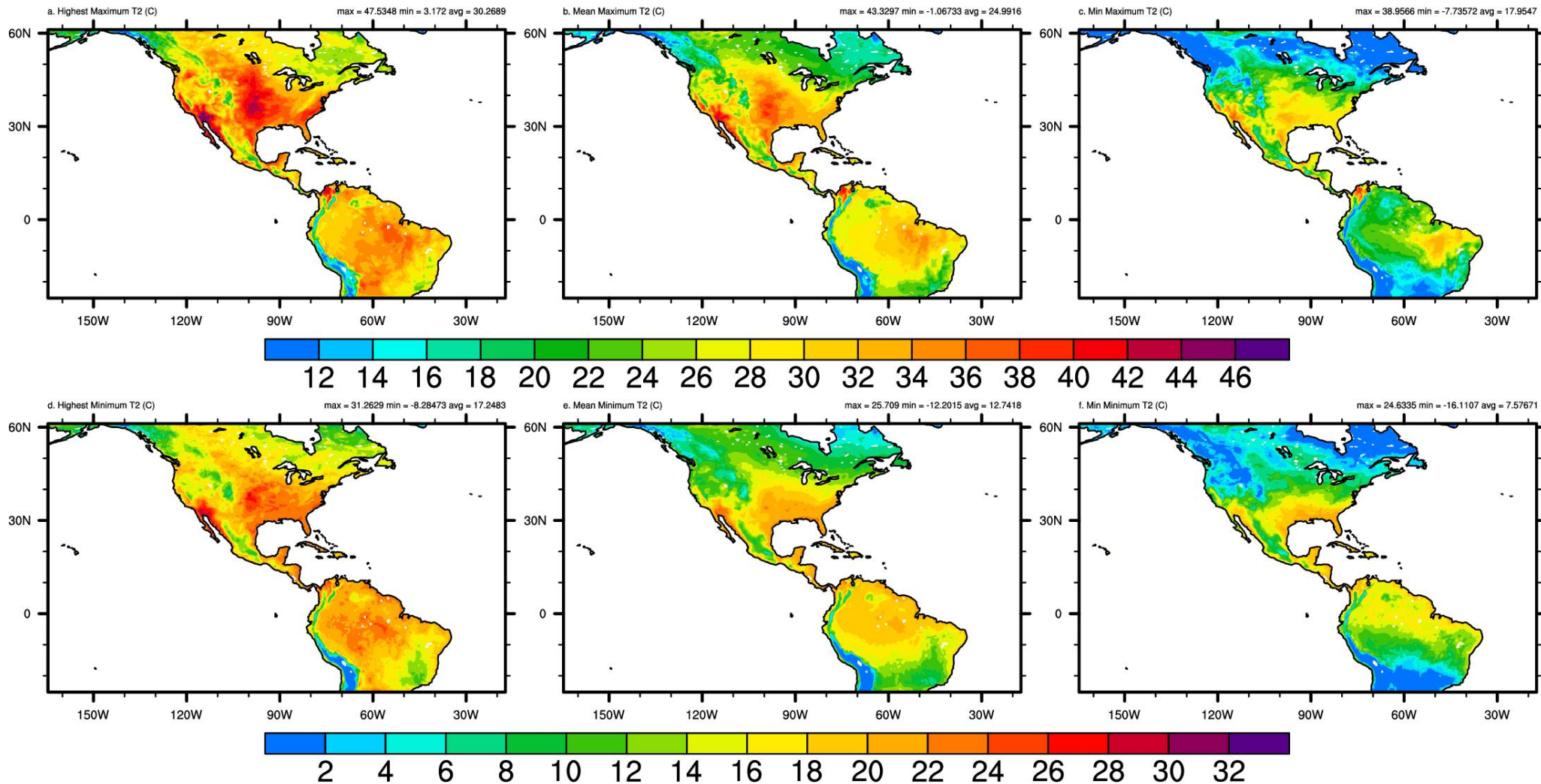
tmax = dim_max_n(t,0) ; Daily max
tlandmax = dim_max_n(tland,0) ; Daily max with ocean masked out

; 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)/)))
```

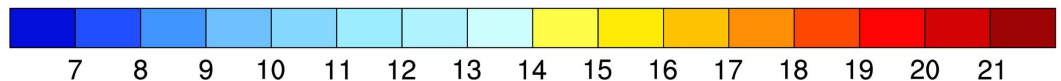
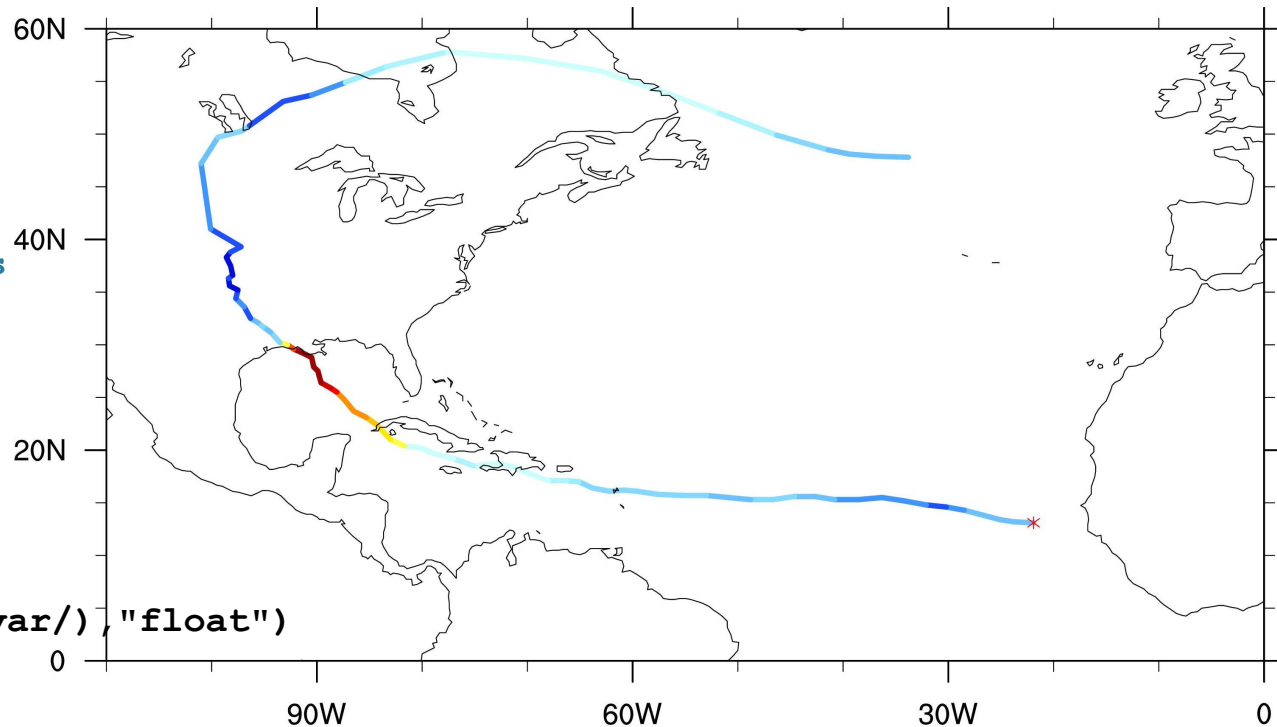
```
    gsn_polyline(wks,map, (/lon(j),lon(j+1)/), (/lat(j),lat(j+1)/), pres)
```

```
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)
```

```
asciwrite ("t2.txt" , t2_point)
asciwrite ("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
asciwrite ("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

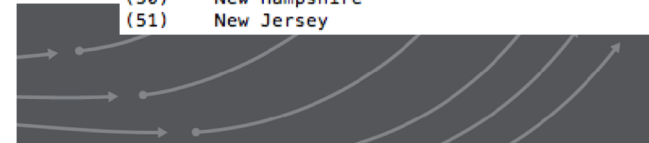
```
shp_filename = "cb_2014_us_state_20m.shp"
f = addfile(shp_filename, "r")
print(f)                ; print shapefile metadata

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

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_numPts : 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
```

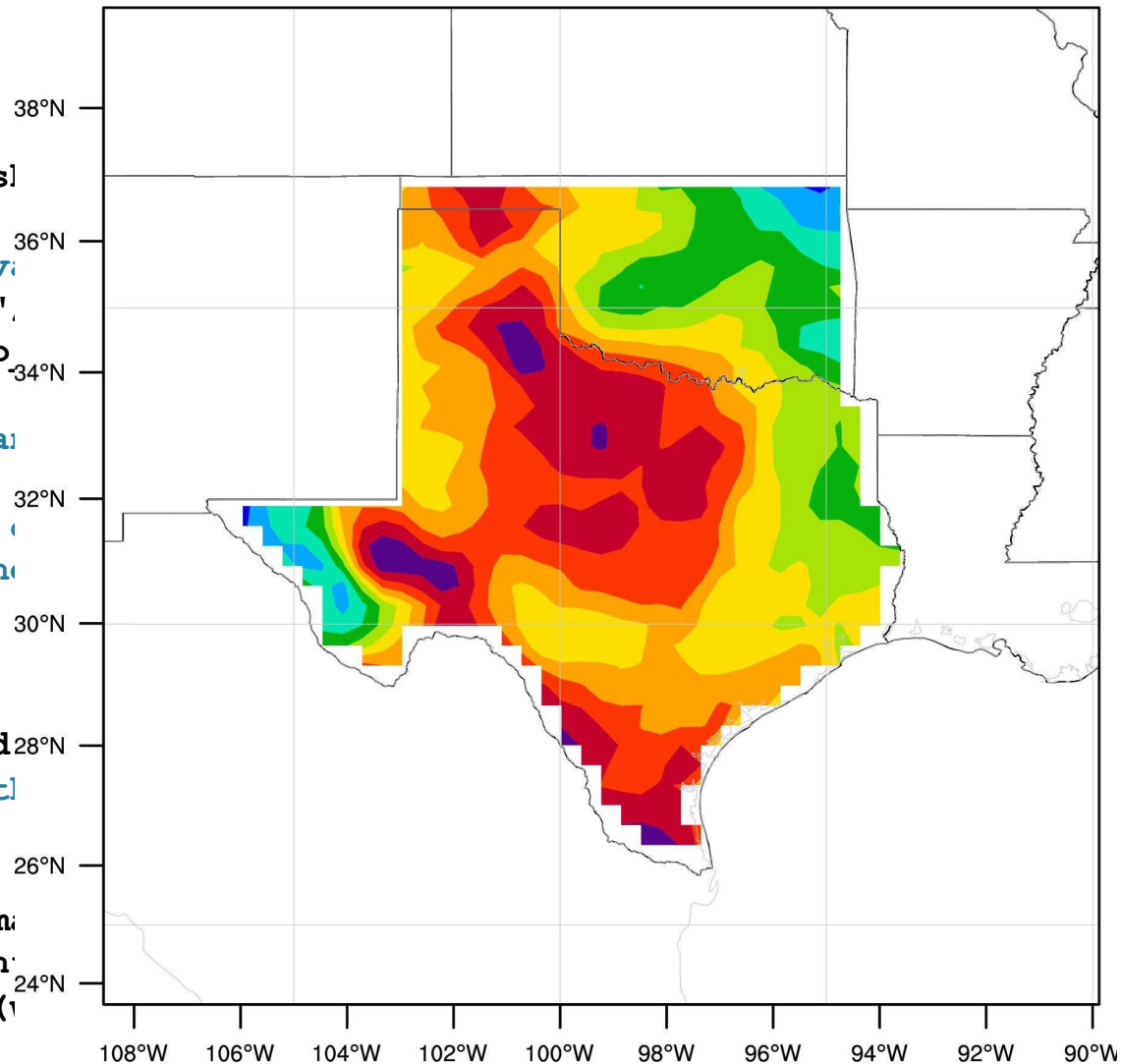


Shapefiles and WRF

```
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 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.shp"
opt = True
opt@shape_var = "NAME" ; We know the variable
opt@shape_names = (/"Oklahoma","Texas"/)
var_mask = shapefile_mask_data(var,shp_filename)
pltres = True
pltres@PanelPlot = True ; We need a panel
mpres = True
mpres@Zoomin = True ; We want to zoom in
mpres@Xstart = 150 ; Grid points of the map
mpres@Ystart = 135
mpres@Xend = 200
mpres@Yend = 185
var_mask_zoom = var_mask(y_start:y_end,x_start:x_end)
; Make contours, draw them on a map, title
opts = True
opts@cnFillOn = True
contour_mask = wrf_contour(a,wks,var_mask_zoom)
plot_mask = wrf_map_overlays(a,wks,contour_mask)
id_mask = gsn_add_shapefile_polygons(id_mask,shp_filename)
draw(plot_mask)
frame(wks)
```



Georeferenced 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 & user 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>



WRF/NCL Support

- Use the WRF & MPAS-A Support Forum
 - Use for all questions about using NCL with WRF
 - There is a NCL section in the Post-processing/Utilities folder

<http://forum.mmm.ucar.edu/phpBB3/index.php>

- For generic NCL questions use:
 - ncl-talk@ucar.edu

