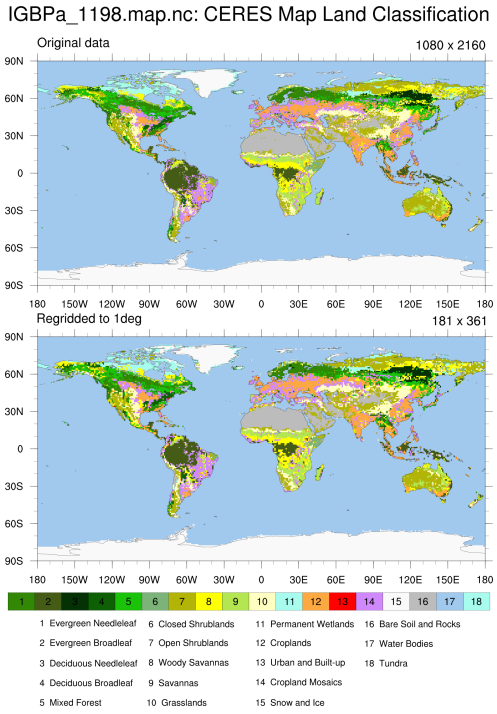


Introduction to NCL Graphics

Mary Haley

Sponsored by the National Science Foundation

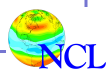
Goals for this lecture

- Get you comfortable with the basics of NCL Graphics
- Geared towards new users of NCL. . .but with tips for intermediate and advanced users
- Show you the most common things people do with NCL graphics
- Give you random tips for editing, debugging, creating nice graphics

For the lab: be thinking about what kind of graphics you want to create with NCL.

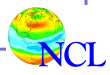
NCL lecture scripts can also be found on workshop page, or here:

http://www.ncl.ucar.edu/Training/Workshops/Scripts/



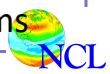
NCL Graphics topics

- Types of graphics you can create with NCL
- The basics
- Line-by-line examples
- Review
- Customizing NCL environment
- Common mistakes and problems
- Debugging tips
- Creating images for Web and/or presentations
- PyNGL/PyNIO Python modules



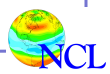
Types of graphics you can create with NCL

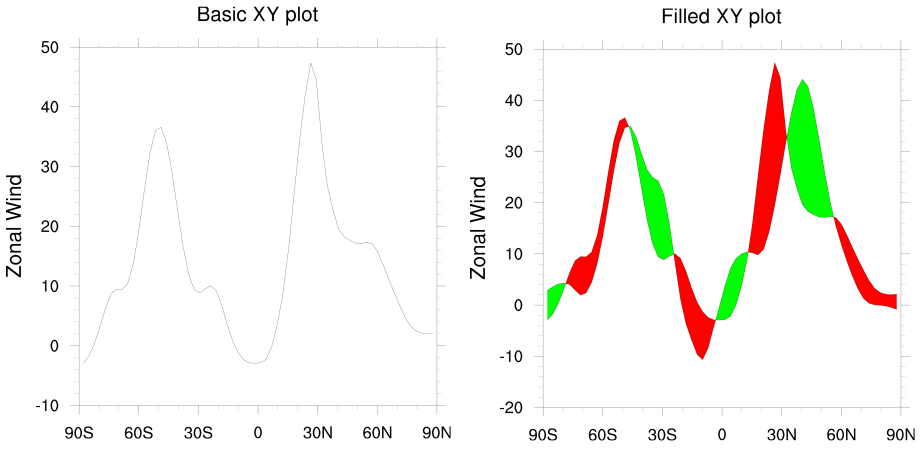
- XY
- Contour
- Vector
- Streamline
- Overlays
  - Contours over maps, vectors over contours, etc.
- Primitives
  - markers, polylines, polygons, text
- Specialized plots
  - bar charts, skew-T, wind roses, taylor diagrams

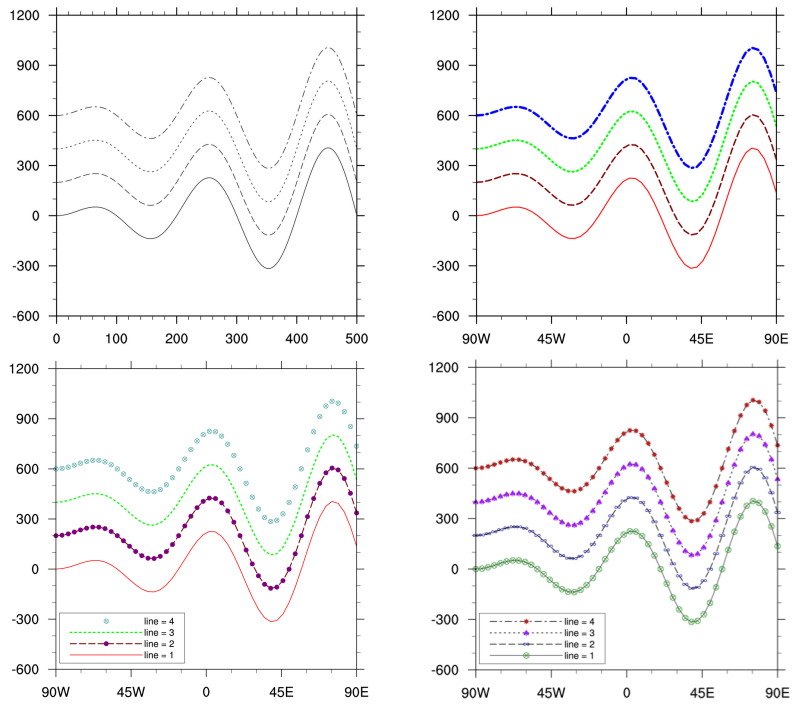


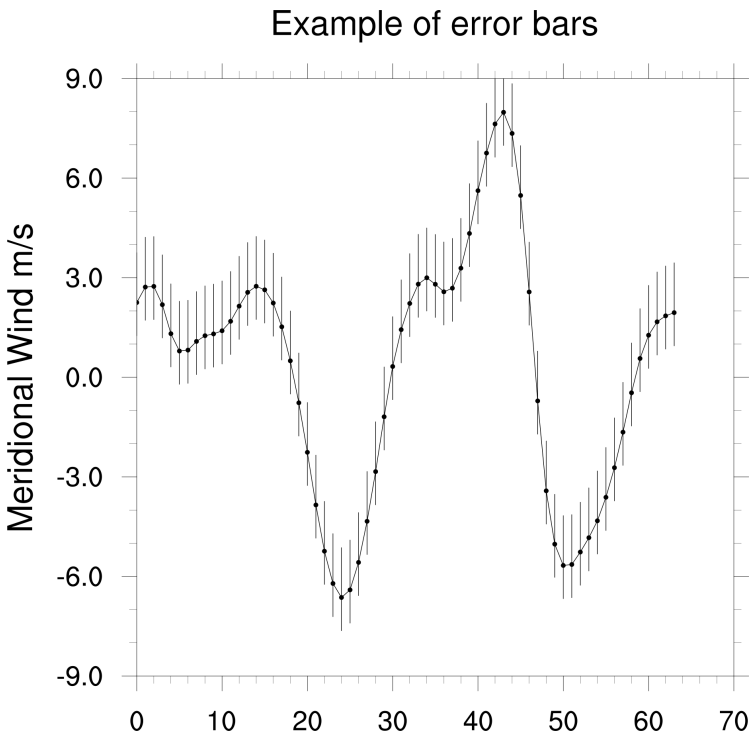
Types of graphics you can create with NCL

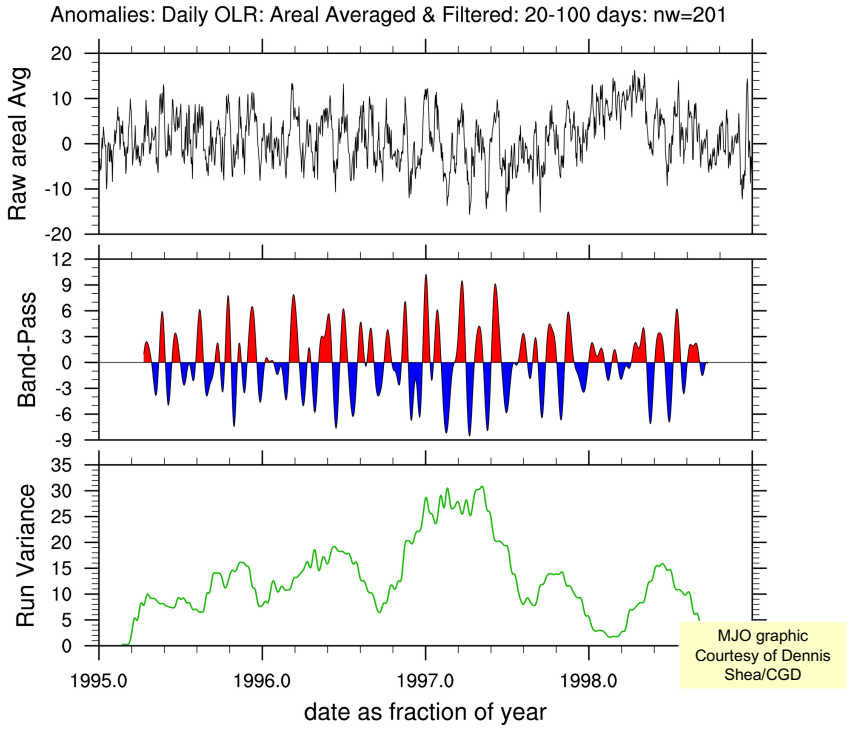
- XY
- Contour
- Vector
- Streamline
- Overlays
- Primitives
- Specialized plots

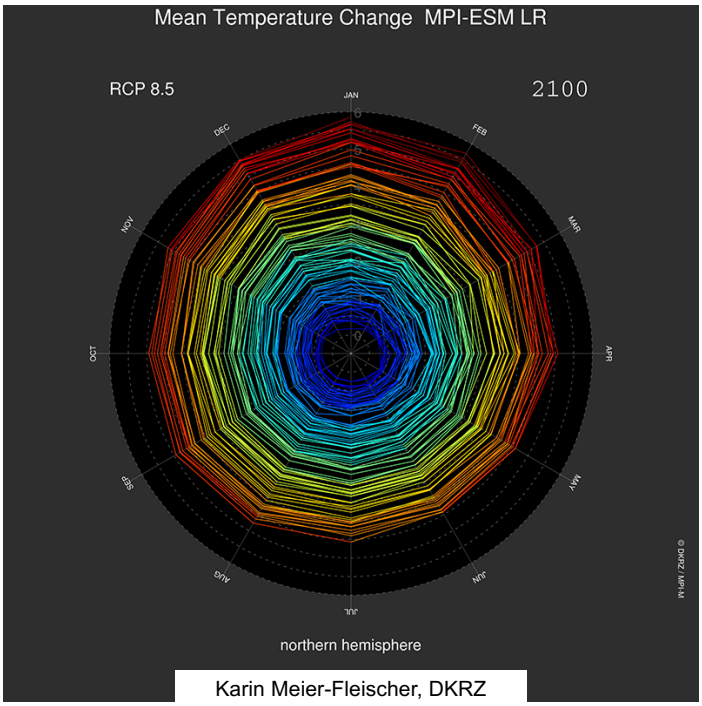



Basic XY plot
Filled XY plot
Zonal Wind
Zonal Wind
50
40
30
20
10
0
-10
-20
90S
60S
30S
0
30N
60N
90N


1200
900
600
300
0
-300
-600
0
100
200
300
400
500
90W
45W
0
45E
90E
line = 4
line = 3
line = 2
line = 1

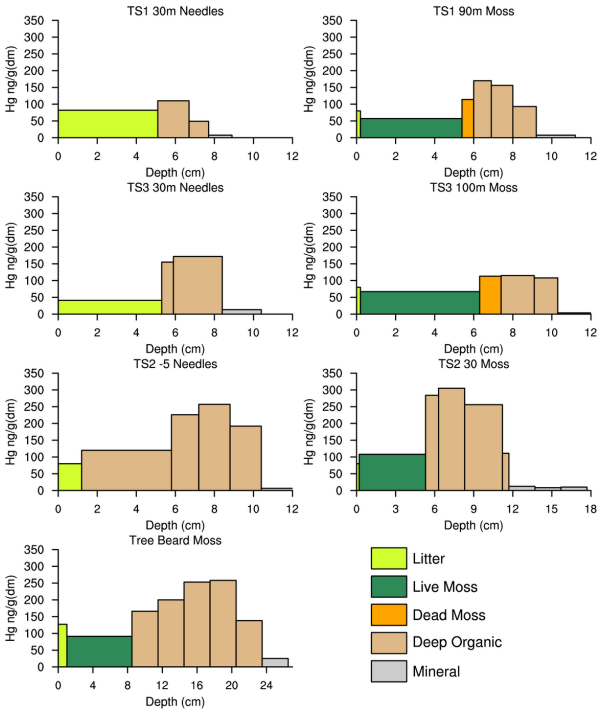

Example of error bars
Meridional Wind m/s
9.0
6.0
3.0
0.0
-3.0
-6.0
-9.0
0
10
20
30
40
50
60
70

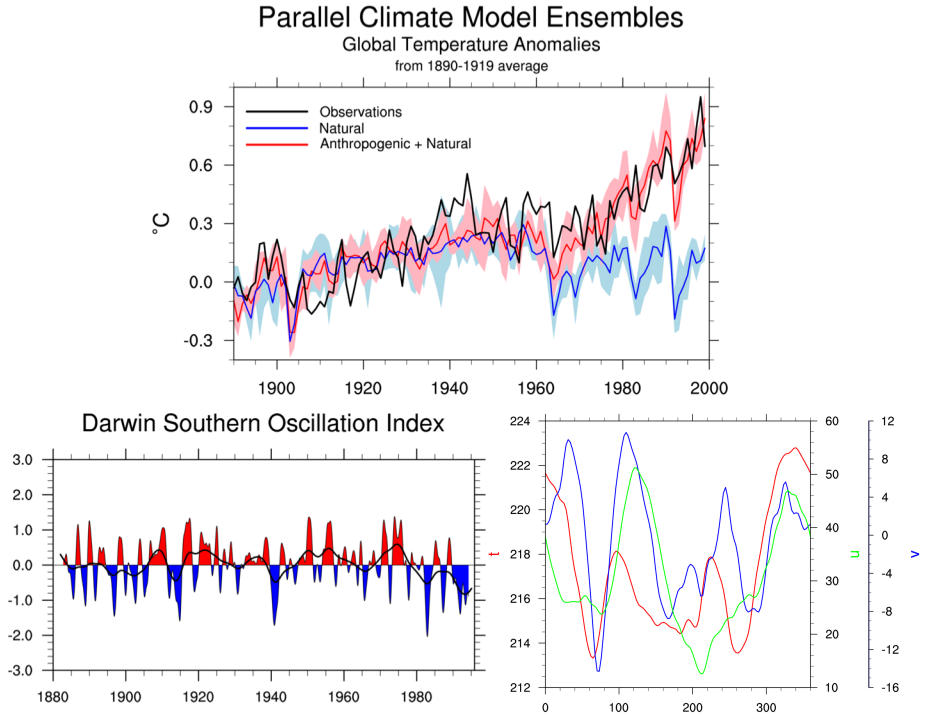

Anomalies: Daily OLR: Areal Averaged & Filtered: 20-100 days: nw=201
Raw areal Avg
Band-Pass
Run Variance
1995.0
1996.0
1997.0
1998.0
date as fraction of year
MJO graphic
Courtesy of Dennis
Shea/CGD



Karin Meier-Fleischer, DKRZ

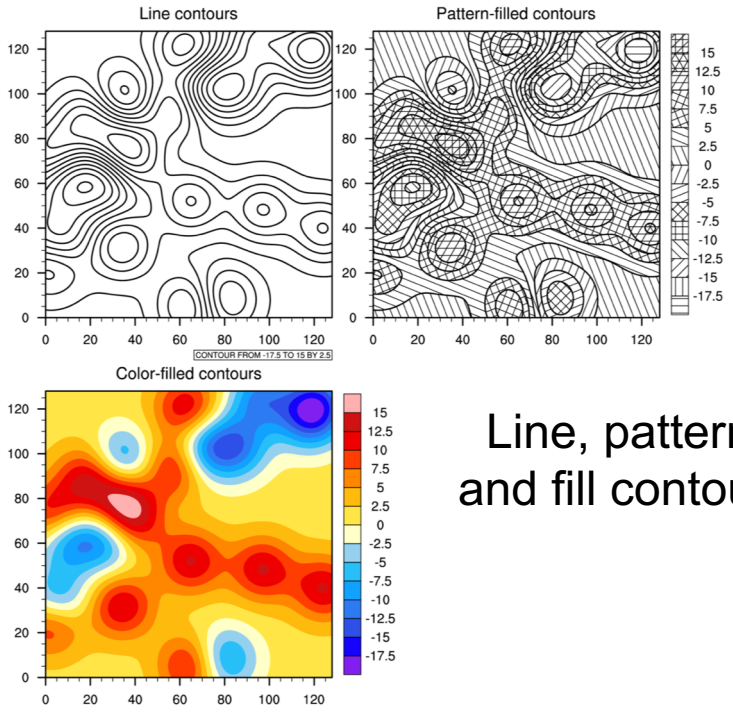
Bar chart



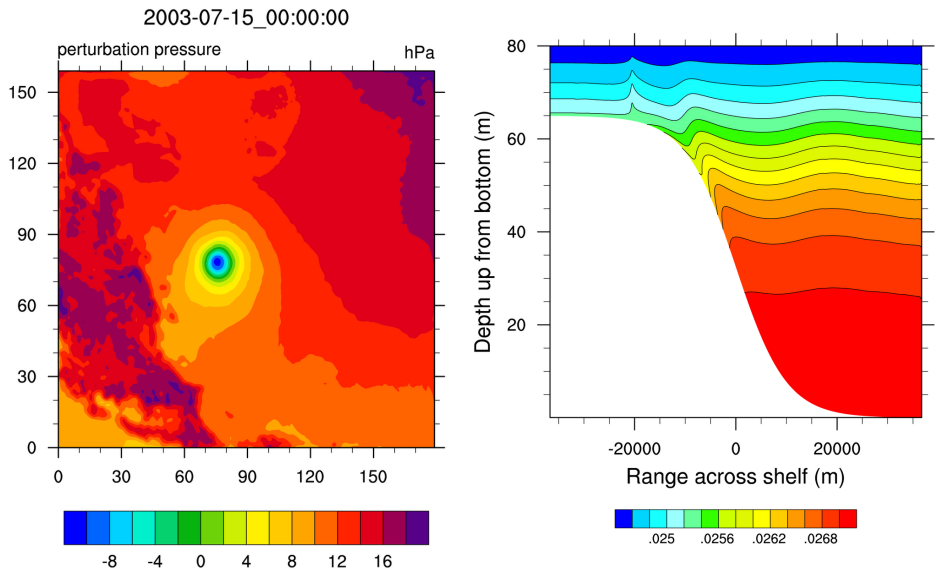


Types of graphics you can create with NCL

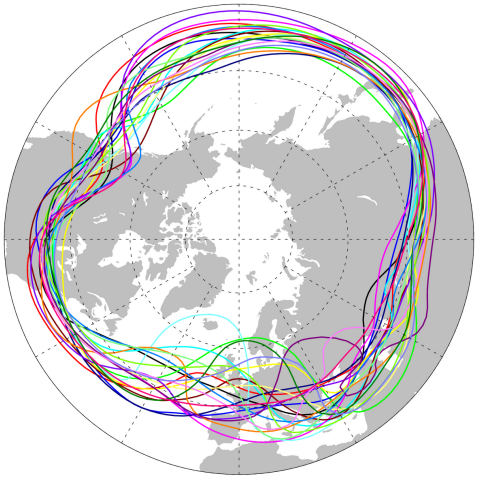
- XY
- Contour
- Vector
- Streamline
- Overlays
- Primitives
- Specialized plots

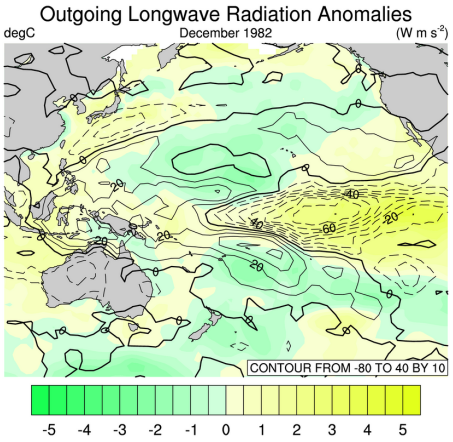


Line, pattern, and fill contours

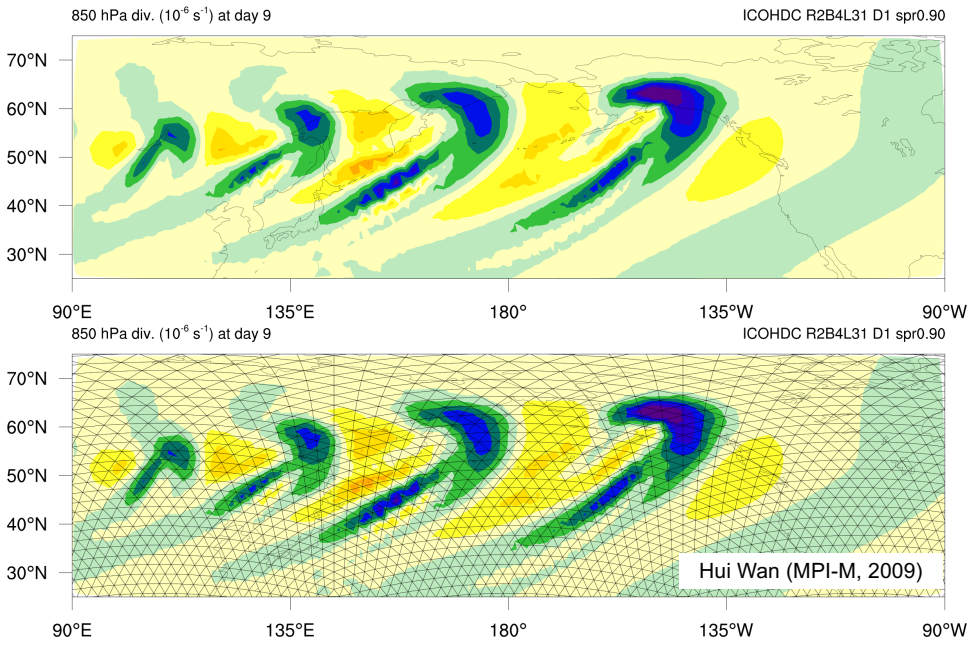


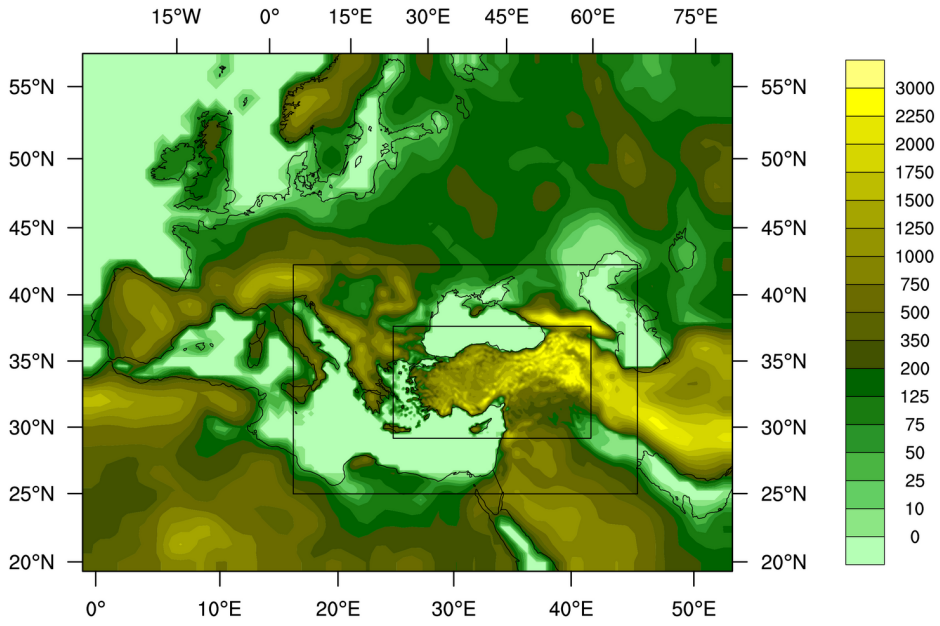
Contours (line and filled) over a map



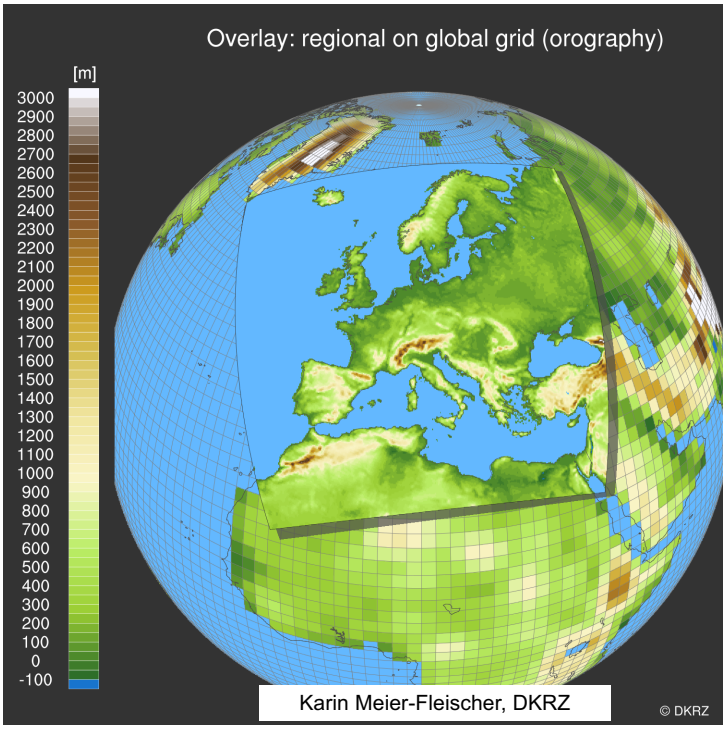


ICON plot with and without triangular mesh





Ufuk Turuncoglu, ITU
Turkey Climate Change Scenarios



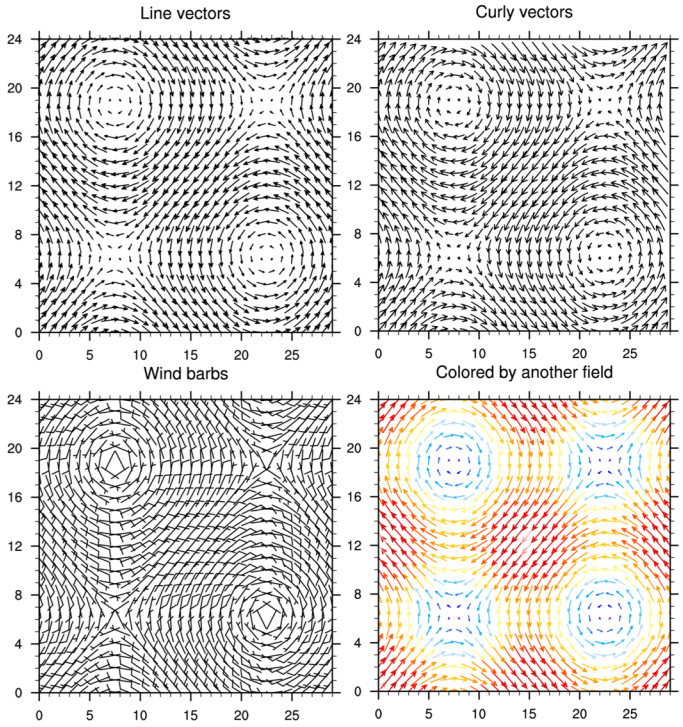
Types of graphics you can create with NCL

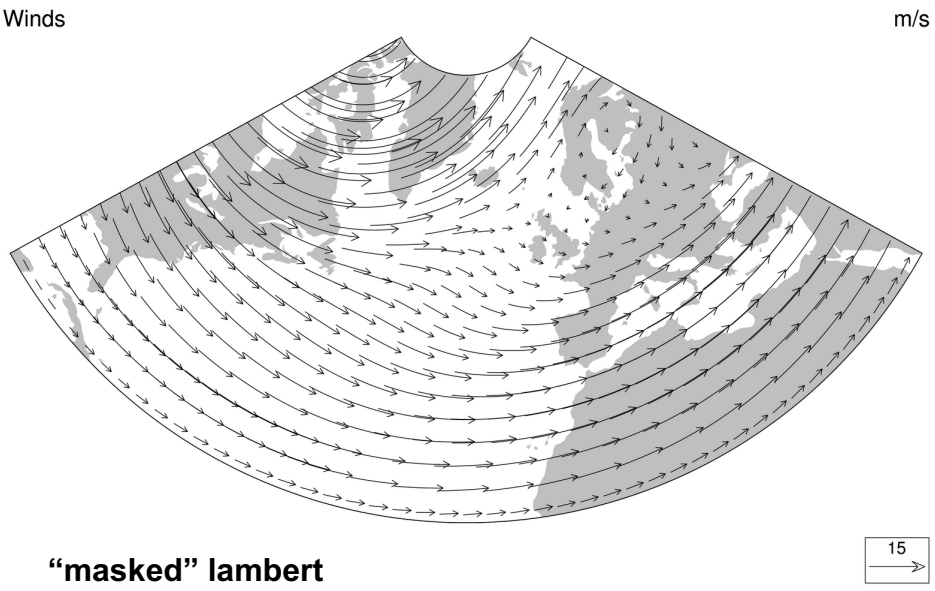
- XY
- Contour
- Vector
- Streamline
- Overlays
- Primitives
- Specialized plots

Vector types

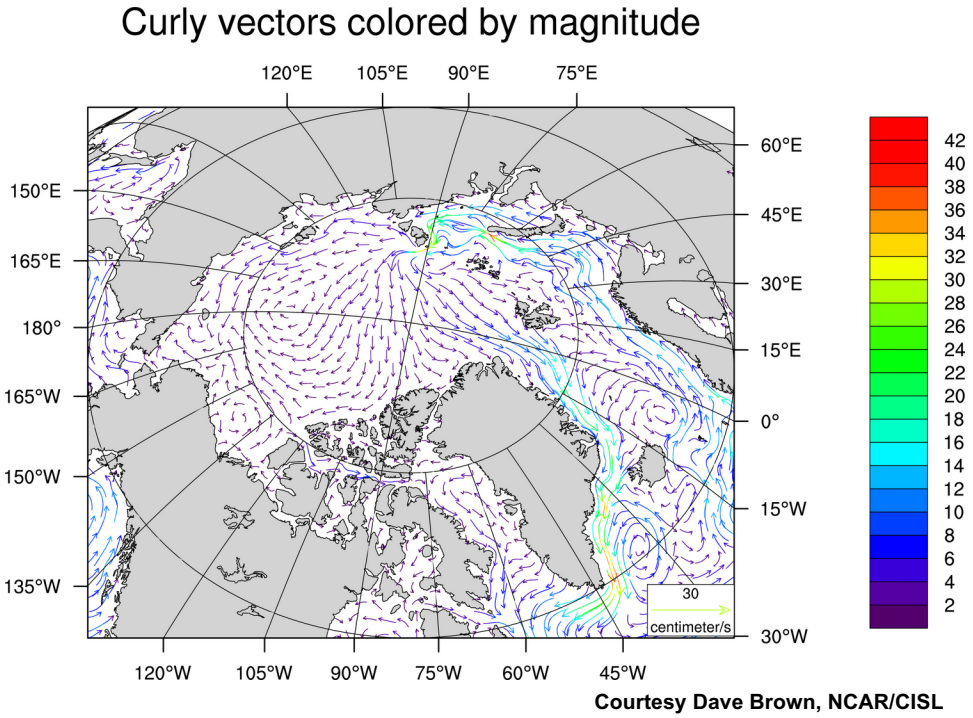
1. Line
2. Curly
3. Wind barb

Any one of these can be colored by another field (e.g. magnitude)





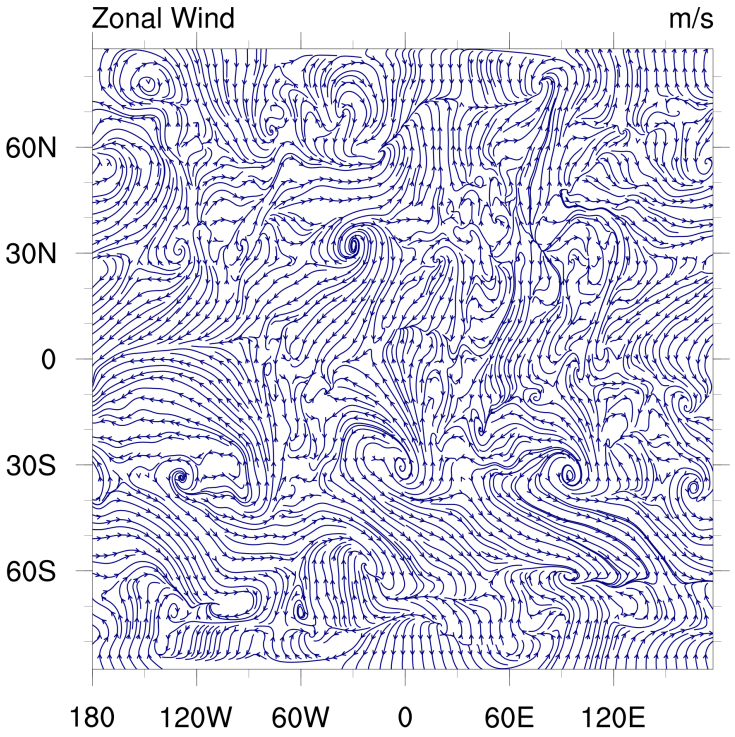
“masked” lambert conformal plot



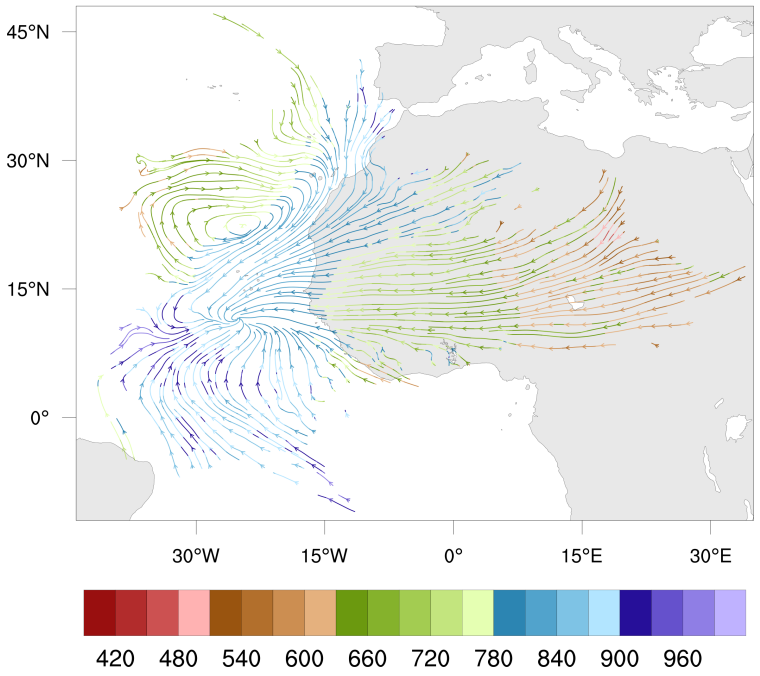
Courtesy Dave Brown, NCAR/CISL

Types of graphics you can create with NCL

- XY
- Contour
- Vector
- **Streamline**
- Overlays
- Primitives
- Specialized plots



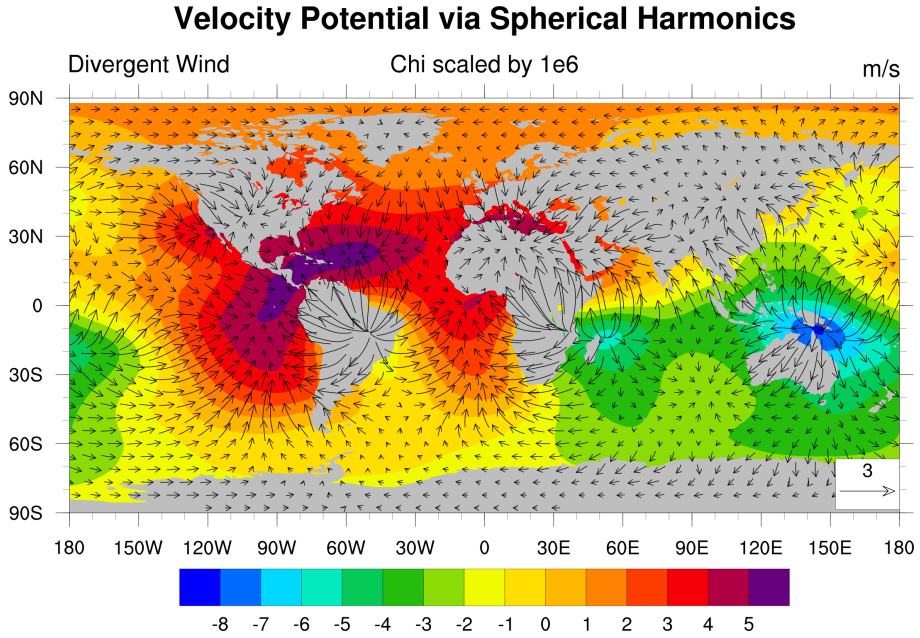
Streamlines colored by another field

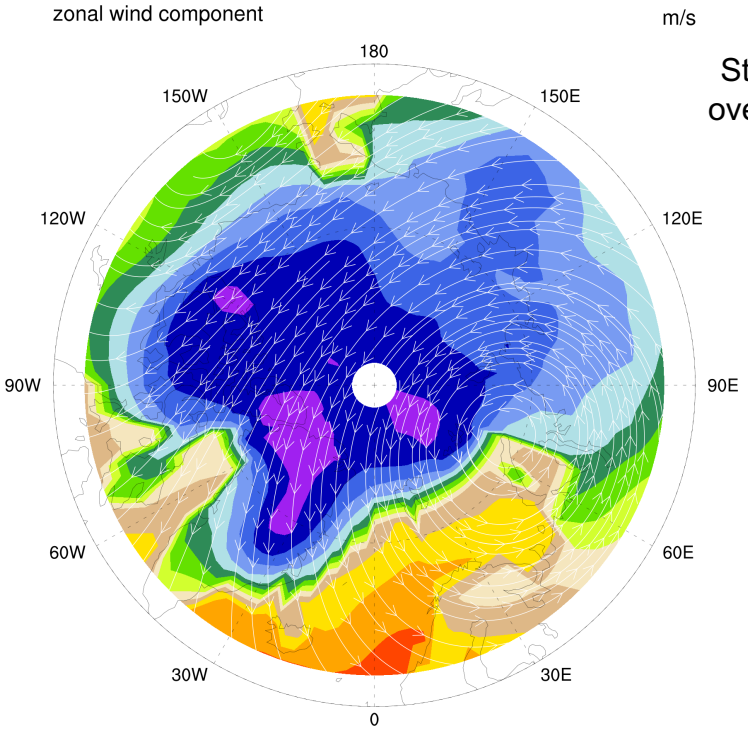


Types of graphics you can create with NCL

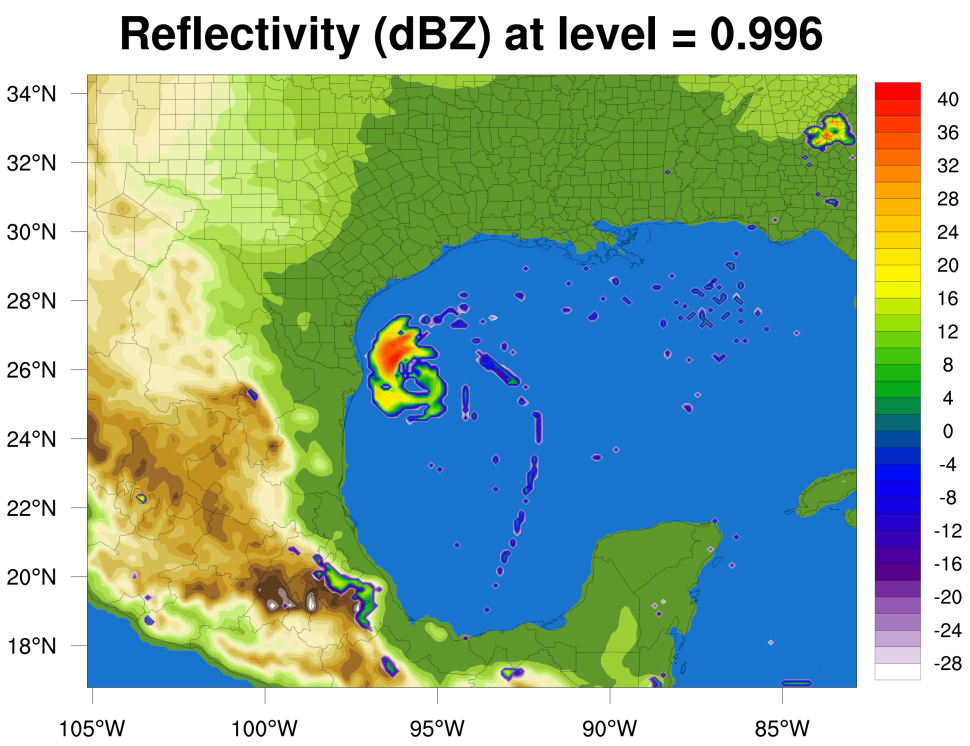
- XY
- Contour
- Vector
- Streamline
- Overlays
- Primitives
- Specialized plots

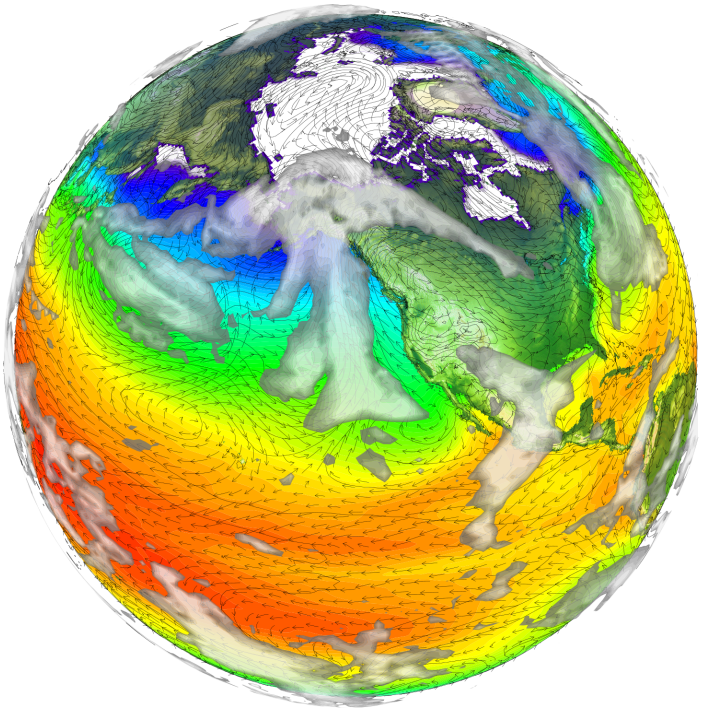
Multiple overlays (contours and vectors)





Streamlines over contours





CCSM4 data
Six fields overlaid:

Ice thickness
(filled contours)

Sea surface temperature
(filled contours)

Topo map
(filled contours)

Sea level pressure
(line contours)

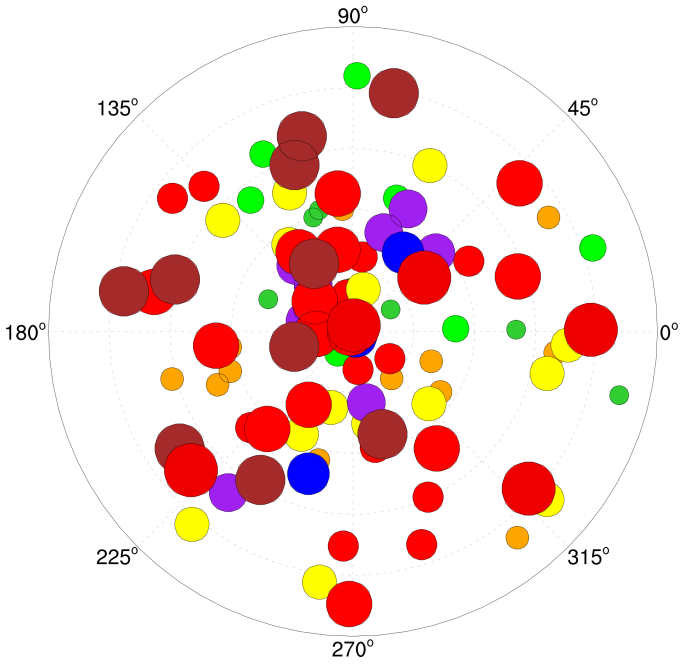
UV winds

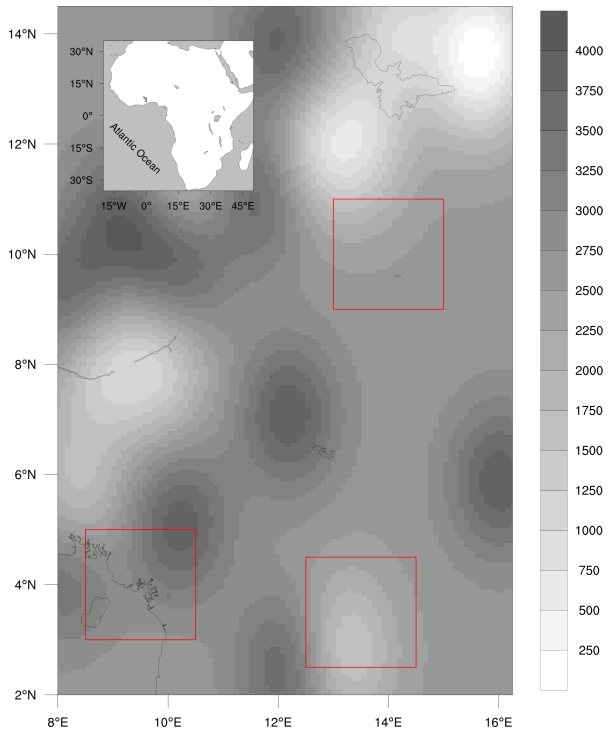
Vertically-integrated
clouds (partially
transparent
filled contours)

Types of graphics you can create with NCL

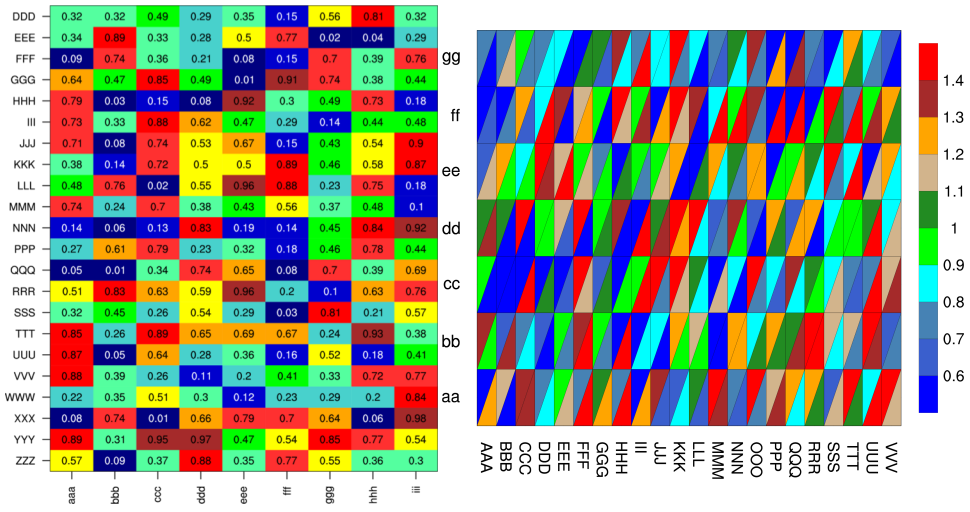
- XY
- Contour
- Vector
- Streamline
- Overlays
- Primitives (markers, lines, text, polygons)
- Specialized plots

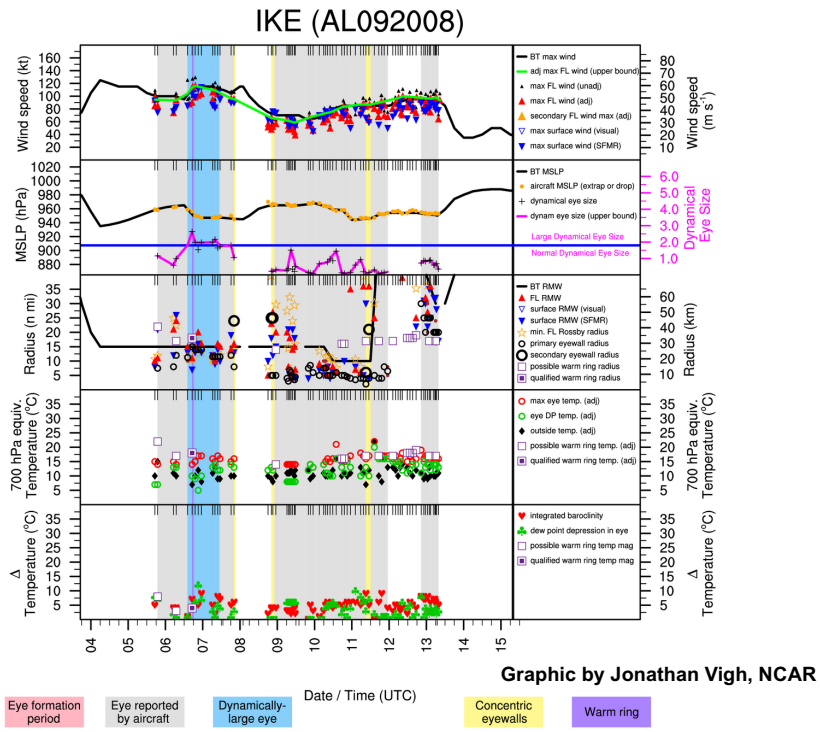
Radial background with markers





Filled polygons with text

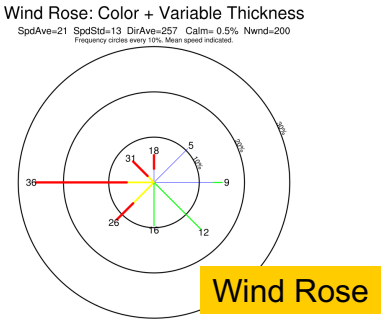




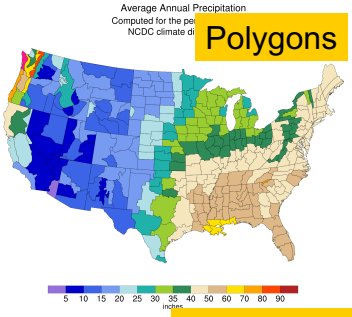
Types of graphics you can create with NCL

- XY
- Contour
- Vector
- Streamline
- Overlays
- Primitives
- Specialized plots
  - Skew T, WRF, wind roses, panels, shapefiles

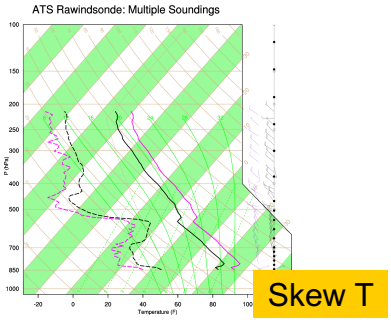
Special Templates and Scripts



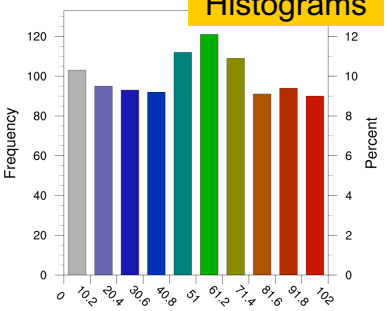
Wind Rose



Polygons

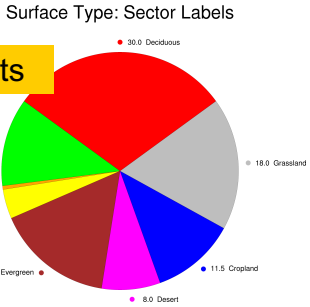


Skew T



Histograms

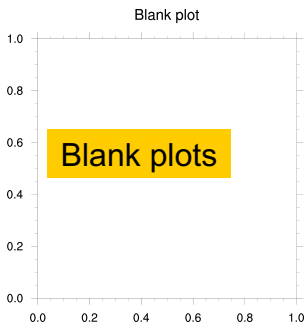
More Special Templates and Scripts



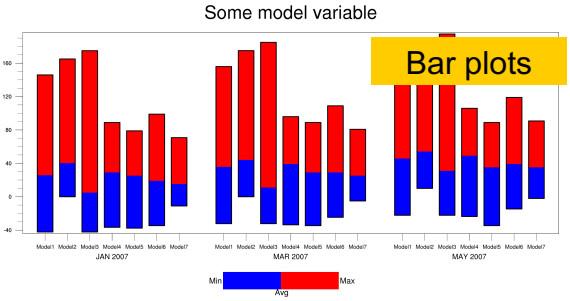
Pie Charts

CAM METRICS	Case A	Case B
	ANN	ANN
SLP_ERA40	1.230	1.129
Tsfc_ERA40	0.988	0.996
	1.092	1.016
	1.172	1.134
	1.064	1.023
SW_ERS	0.966	0.962
U300_ERA40	1.079	1.048
Guess_BOGUS	0.781	0.852
RH_NCEP	1.122	0.911
LHFLX_ERA40	1.000	0.835
TWP_ERA40	0.998	0.712
CLDTOT_NCEP	1.321	1.122
O3_NASA	0.842	0.956
Q_JMA	0.978	0.832
PBLH_JMA	0.998	0.900
Omega_CAS	0.811	1.311

Tables



Blank plots



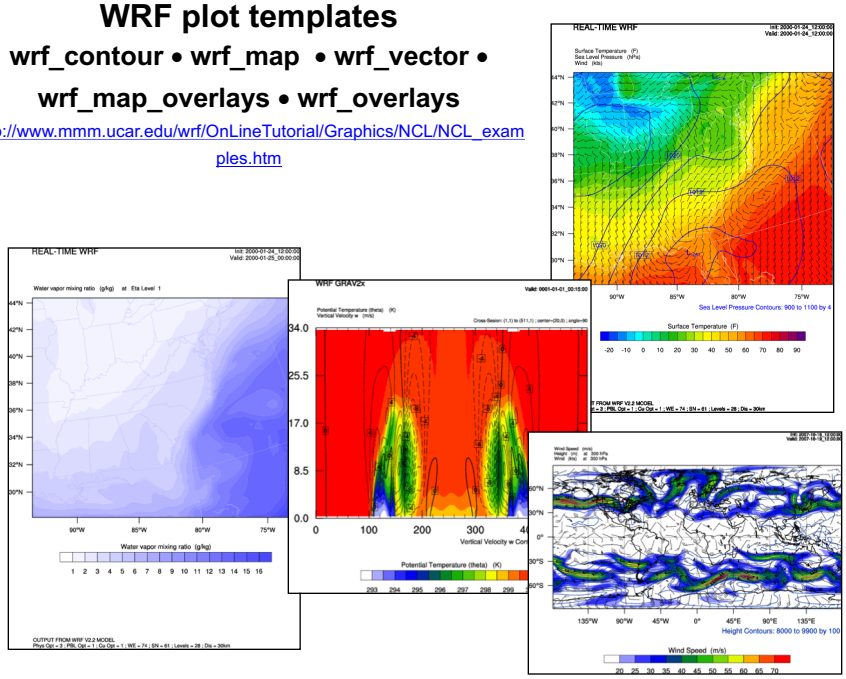
Bar plots

WRF plot templates

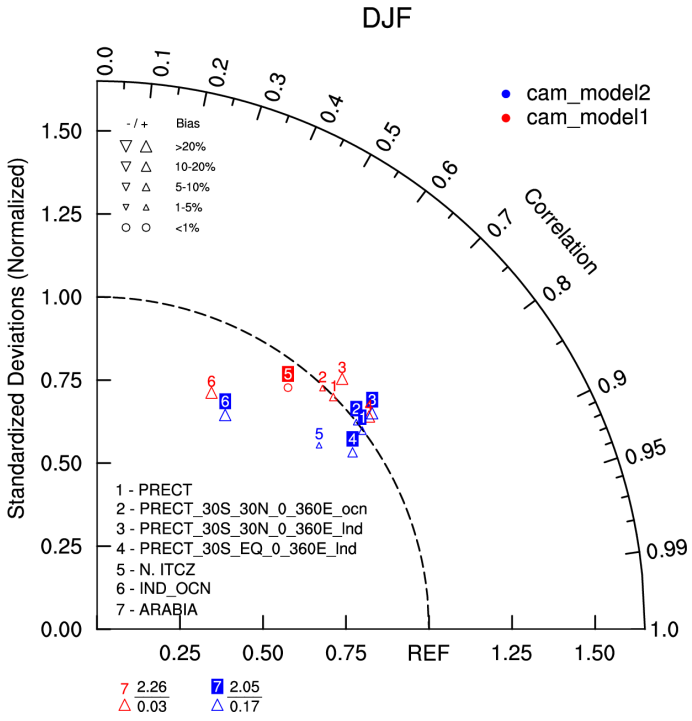
wrf_contour • wrf_map • wrf_vector •

wrf_map_overlays • wrf_overlays

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

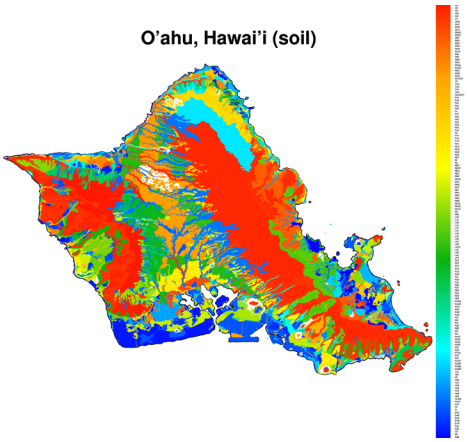


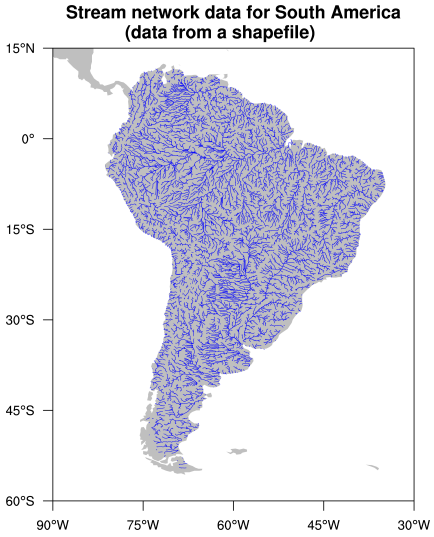
Taylor diagram Courtesy of Dennis Shea and Adam Phillips, CGD

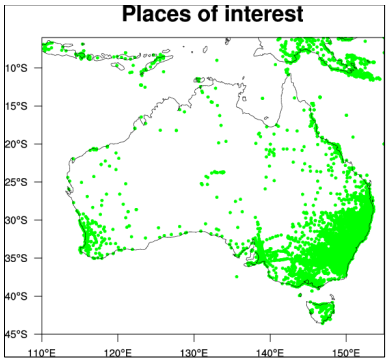


The ESRI Shapefile is a popular geospatial vector data format for GIS software.

Numerous (and free) shapefiles can be found by googling on the web.





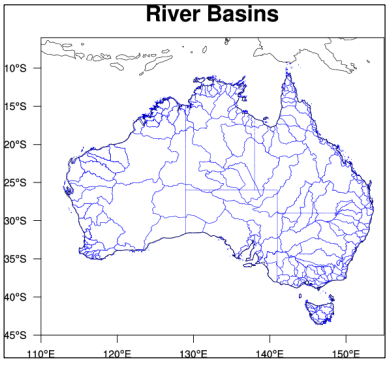


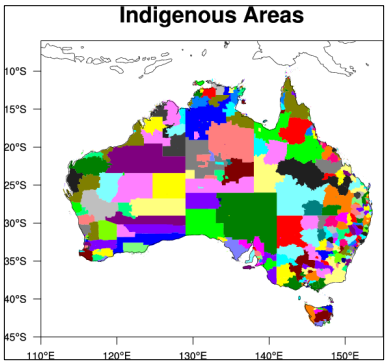
The three types of shapefiles supported by NCL:

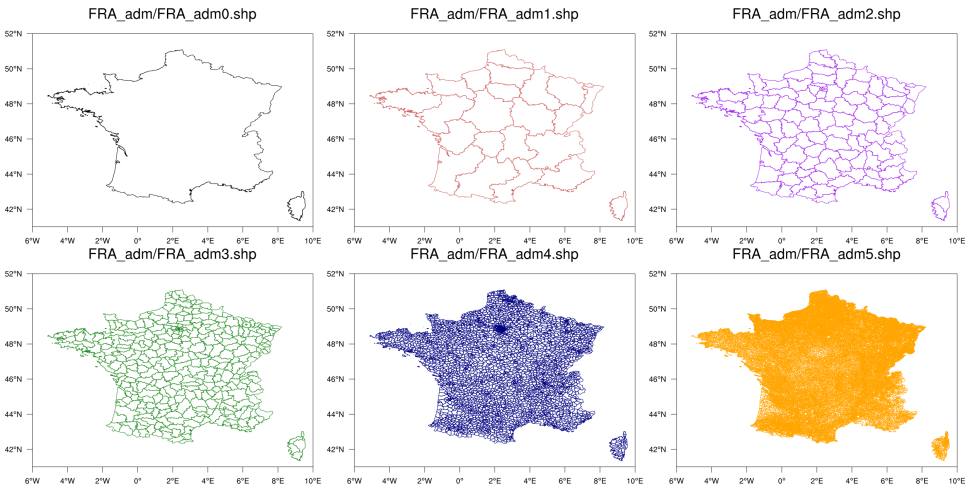
Point – locations of cities, population data

Line – rivers, roads, trails

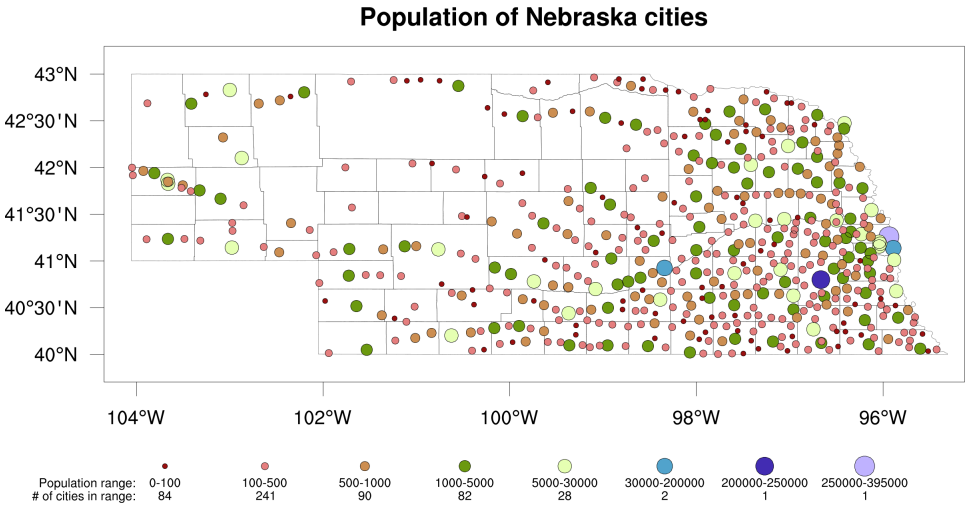
Polygon – counties, lakes



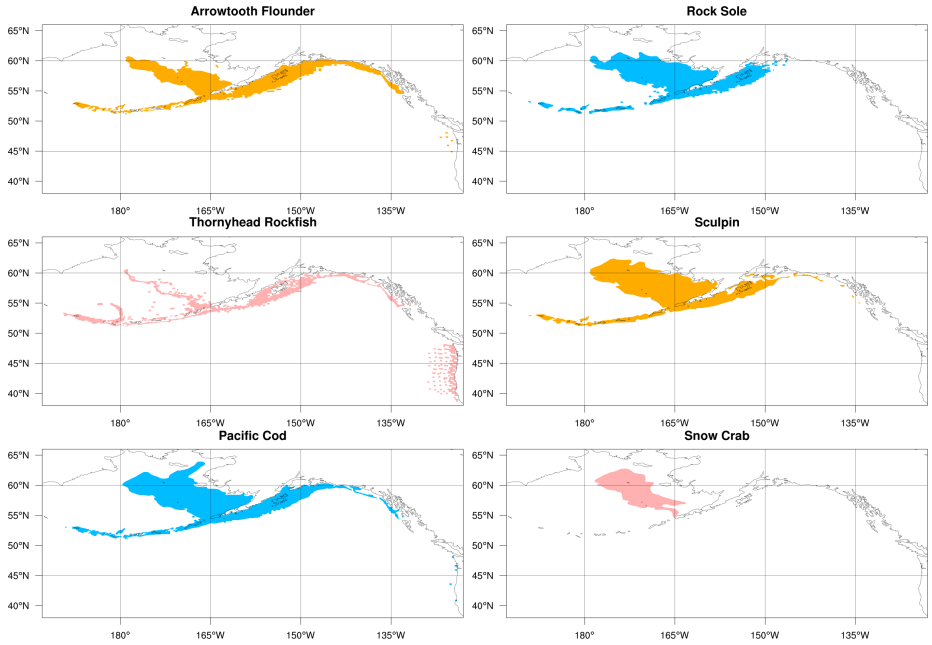




Global Administrative Areas database (http://www.gadm.org) offers consistent administrative boundaries at many levels. The level 0 database (nations) is good to use for global or mesoscale results, level 1 is the first level of sub-national administration (typically states/provinces and territories) while level 2 offers the second level of administration and is potentially useful for high-resolution plots.



Population data downloaded from Nebraska Department of Natural Resources
http://dnr.nebraska.gov/boundaries-plss



The "Alaska Essential Fish Habitat Species" shapefile was downloaded from:

http://alaskafisheries.noaa.gov/habitat/efh/efhshp/default.htm

Panel plots: multiple plots on a page

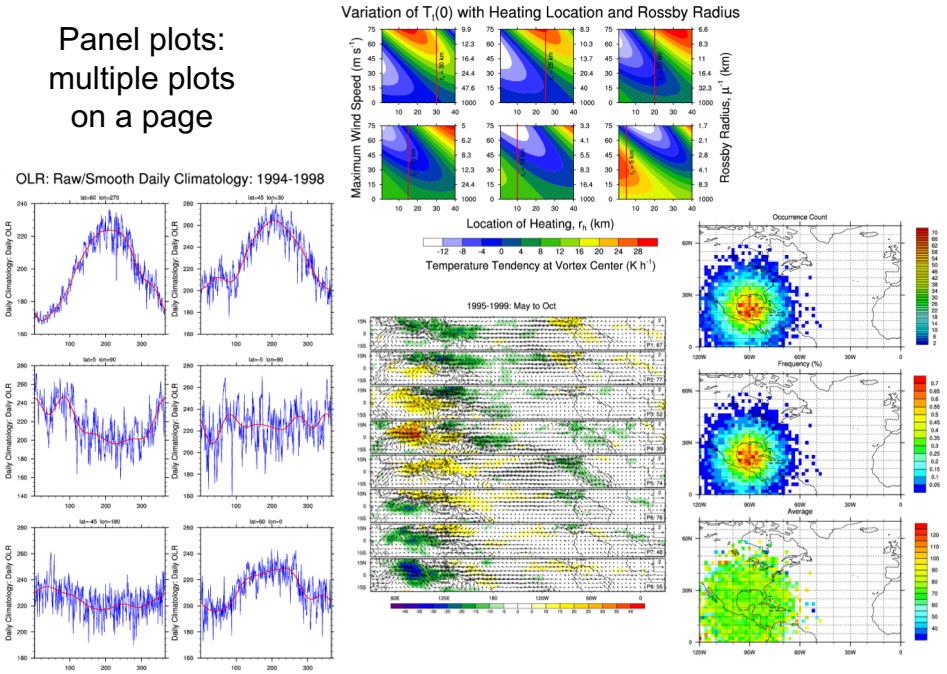
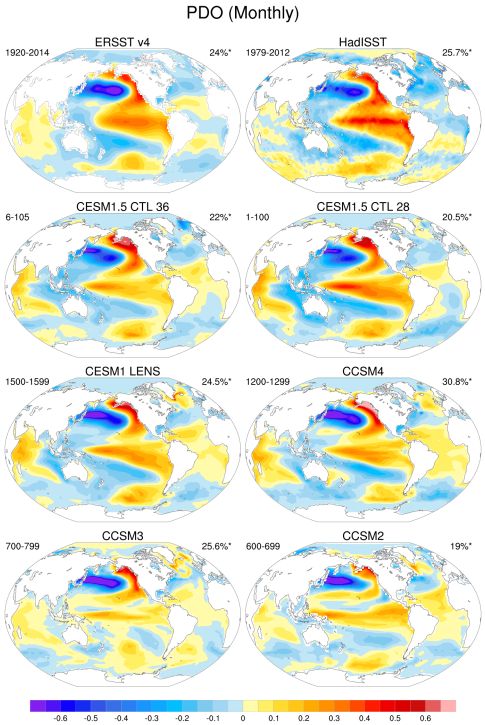


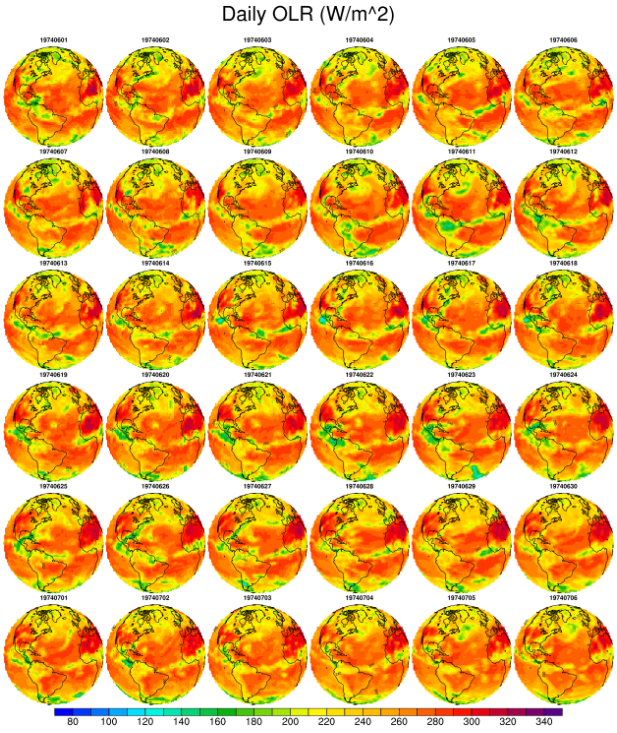
Image from Climate Variability Diagnostics Package.

Courtesy Adam Phillips

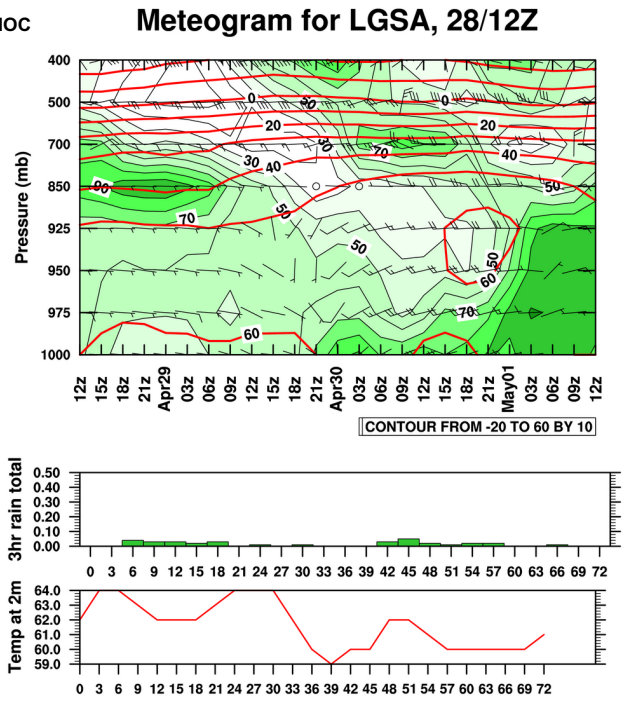
NCAR/CGD

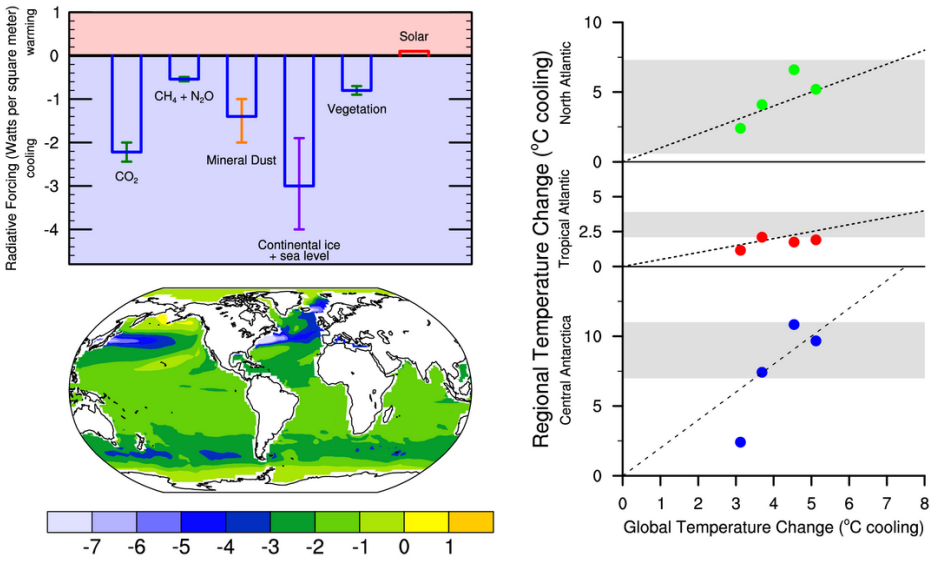


OLR file from Bob Setzenfand

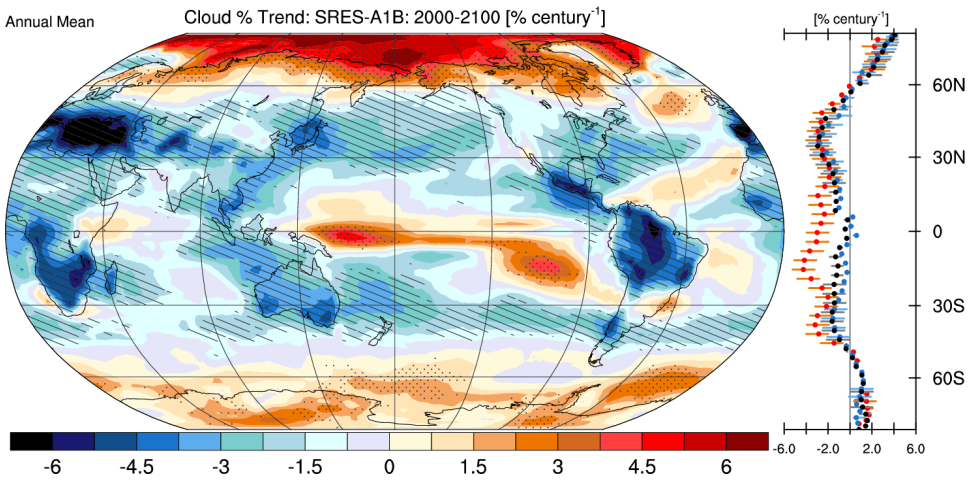


John Ertl, FNMOC





Courtesy Adam Phillips, NCAR CGD



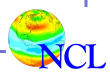
John Fasullo, NCAR/CGD

NCL Graphics topics

- Types of graphics you can create with NCL
- The basics
- Line-by-line examples
- Review
- Customizing NCL environment
- Common mistakes and problems
- Debugging tips
- Creating images for Web and/or presentations
- PyNGL/PyNIO Python modules

Four steps to create an NCL plot

1. Load the required scripts
2. Open a “workstation”
3. Set plot options (resources)
4. Call the appropriate plotting function



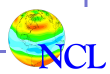
Step 1: Load the required scripts *

* Optional

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

begin                                                    optional
  ii = ispan(0,500,1)
  x  = 501 + .9 * ii * cos(0.0314*ii)
  y  = 501 + .9 * ii * sin(0.0314*ii)

  wks = gsn_open_wks(“png”, “spiral”)    ; spiral.png

  res                   = True           ; Plot options
  res@xyLineColor       = "RoyalBlue"    ; line color
  res@xyLineThicknessF  = 3.0            ; line thickness
  res@tiMainString      = "This is a title”

  plot = gsn_csm_xy(wks,x,y,res)
end                                                      optional
```

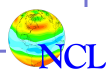
Step 1: Load the "gsn" scripts

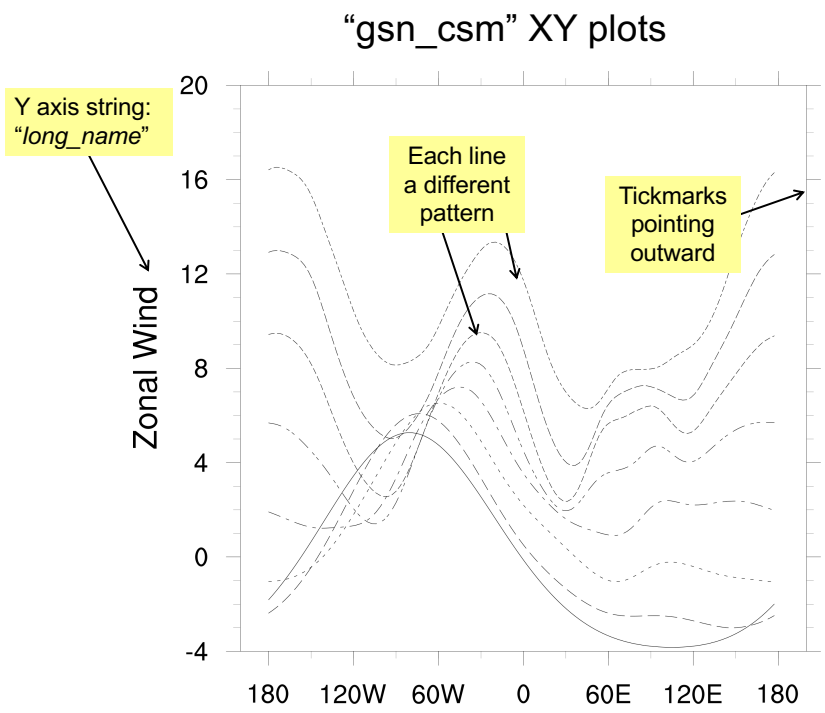
- The "gsn" scripts contain high-level graphical functions
- "gsn" stands for Getting Started using NCL
- "csm" stands for Climate System Model
- Graphical functions in gsn_csm.ncl" are "metadata aware"

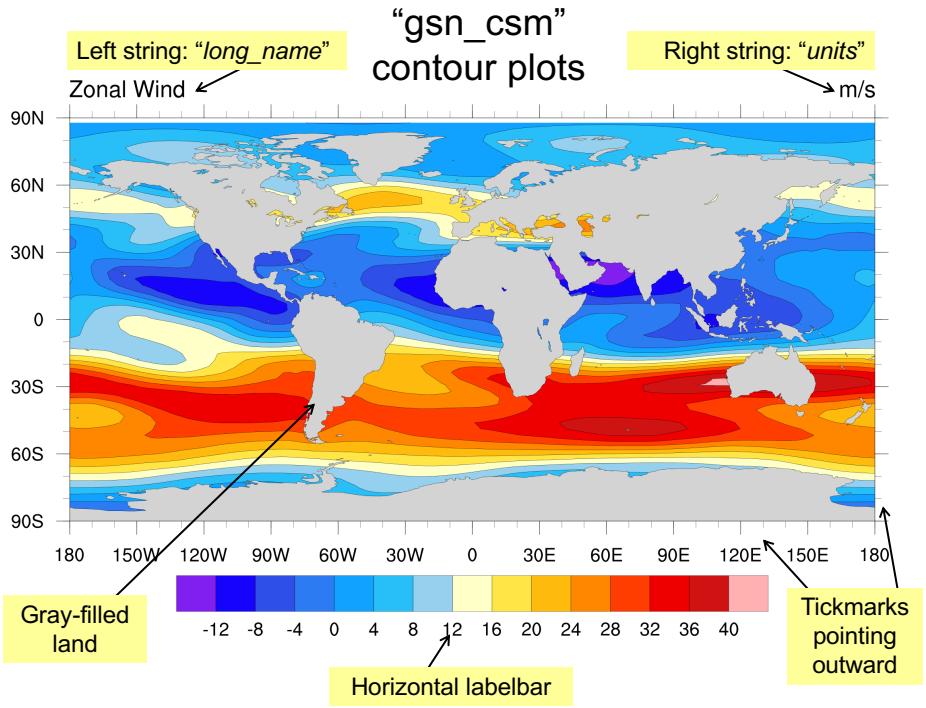


Metadata recognized by gsn_csm scripts

- _FillValue attribute recognized as missing value ("missing_value" is NOT)
- Data attributes such as "long_name" and "units" may be used for plot titles
- Coordinate arrays used for axes values
- If data has coordinate arrays and you are plotting over a map, then "units" attribute of "degrees_east" or "degrees_north" expected




"gsn_csm" XY plots
Y axis string: "long_name"
Each line a different pattern
Tickmarks pointing outward
Zonal Wind
20
16
12
8
4
0
-4
180
120W
60W
0
60E
120E
180


"gsn_csm" contour plots
Left string: "long_name"
Zonal Wind
Right string: "units"
m/s
90N
60N
30N
0
30S
60S
90S
180
150W
120W
90W
60W
30W
0
30E
60E
90E
120E
150E
180
-12
-8
-4
0
4
8
12
16
20
24
28
32
36
40
Gray-filled land
Horizontal labelbar
Tickmarks pointing outward

Step 1.5: Get some data!

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

begin
  ii = ispan(0,500,1)   ; ii = (/0,1,2, . . .,500/)
  x  = 501 + .9 * ii * cos(0.0314*ii)    ; 501 points
  y  = 501 + .9 * ii * sin(0.0314*ii)

  wks = gsn_open_wks("png", "spiral")   ; spiral.png

  res                   = True          ; Plot options
  res@xyLineColor       = "RoyalBlue"   ; line color
  res@xyLineThicknessF  = 3.0           ; line thickness
  res@tiMainString      = "This is a title"

  plot = gsn_csm_xy(wks,x,y,res)
end
```

Step 2: Open a workstation

- "Workstation" is where to send the graphics
  - PNG ("png" – what I used in this PowerPoint)
  - PostScript ("ps") ("eps" - only one image)
  - PDF ("pdf")
  - X11 window ("x11" – good for debugging)
  - SVG (scalable, good for web)
  - NCGM ("ncgm") – rarely used, but can use with "idt" for quick animations
- Has a default color map associated with it

Step 2: Open a “workstation”

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

begin
  ii = ispan(0,500,1)
  x  = 501 + .9 * ii * cos(0.0314*ii)    ; 501 points
  y  = 501 + .9 * ii * sin(0.0314*ii)

  wks = gsn_open_wks(“png”, “spiral”)   ; spiral.png

  res                   = True          ; Plot options
  res@xyLineColor       = "RoyalBlue"   ; line color
  res@xyLineThicknessF  = 3.0           ; line thickness
  res@tiMainString      = "This is a title”

  plot = gsn_csm_xy(wks,x,y,res)
end
```

Step 3: Set plot options (resources)

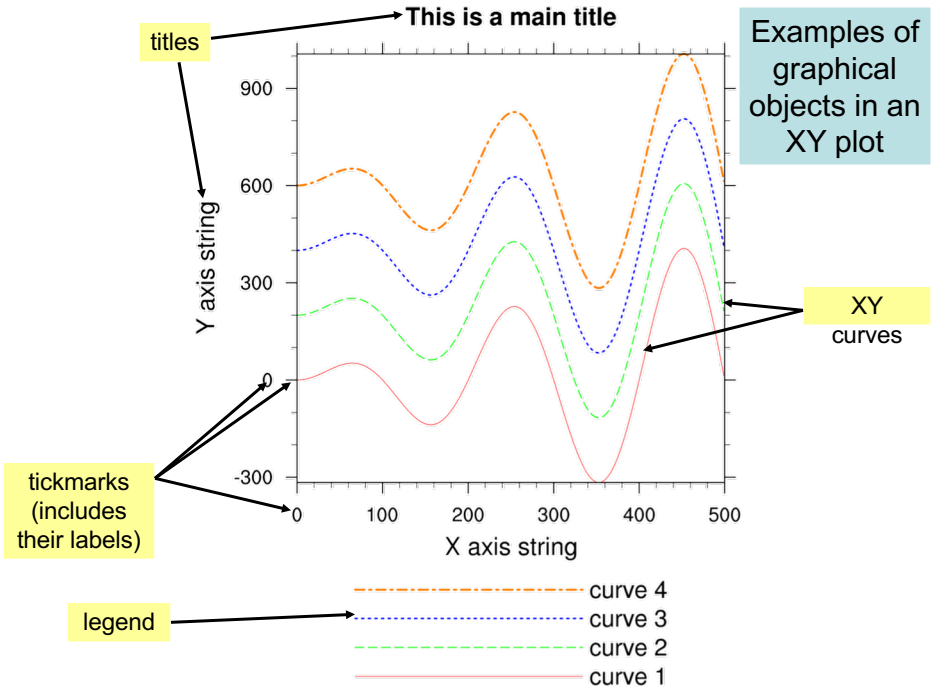
```
;load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_code.ncl"
;load "$NCARG_ROOT/lib/ncarg/nclscripts/csm/gsn_csm.ncl"

begin
  ii = ispan(0,500,1)
  x  = 501 + .9 * ii * cos(0.0314*ii)    ; 501 points
  y  = 501 + .9 * ii * sin(0.0314*ii)

  wks = gsn_open_wks(“png”, “spiral”)   ; spiral.png

  res                   = True          ; Plot options
  res@xyLineColor       = "RoyalBlue"   ; line color
  res@xyLineThicknessF  = 3.0           ; line thickness
  res@tiMainString      = "This is a title”

  plot = gsn_csm_xy(wks,x,y,res)
end
```

Step 3: Set plot options (resources)

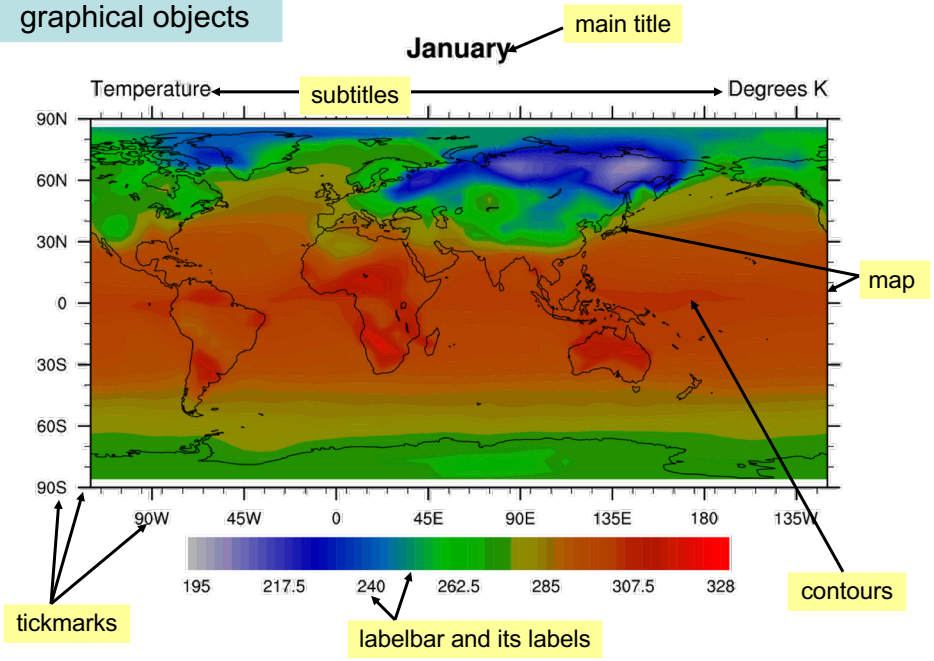
- Resources are how you customize your plots
- There are over 1,400 resources
- They are grouped by object type
- There are 11 "graphical" objects:

  contours *labelbars* *legends* *maps*
  primitives *streamlines* *text strings* *tickmarks*
  titles *vectors* XY *curves*

- There are other "special" resources too
- Most common resources listed in the very back of your workshop booklets



More examples of graphical objects



Description of resources

- Starts with 2 or 3 lower-case letters based on object it is associated with. Some examples:
  - "xy" - XY Plot
  - "cn" - Contour plot
  - "vc" - Vector plot
  - "ti" - Title
  - "tm" - Tickmark
  - "lb" - Labelbar
- Made up of full words; first letter capitalized:
  - xyLineColor
  - cnFillOn
  - tiMainString
  - vcRefMagnitudeF
  - gsnMaximize
- Some have an "**F**" on the end to indicate a floating point resource: "xyLineThickness**F**"
- "gsn" – special resources recognized by gsn scripts

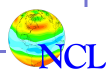
Description of resources (cont'd)

- Resources are set by attaching them as attributes to an NCL *logical* variable:

```
res = True          ; can name "res" whatever you want
res@mpMinLatF = 30  ; decimal not necessary
```

- Most have default values.
- There are many types:

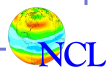
```
res@tiMainString          = "This is a title"
res@tmXBLabelFontHeightF = 0.01
res@cnLineLabelsOn        = True    ; False
res@xyLineColor           = "RosyBrown"
res@xyLineColors          = (/"red","green","blue"/)
res@lgLineThicknesses     = (/ 1.0, 2.0, 3/)
```

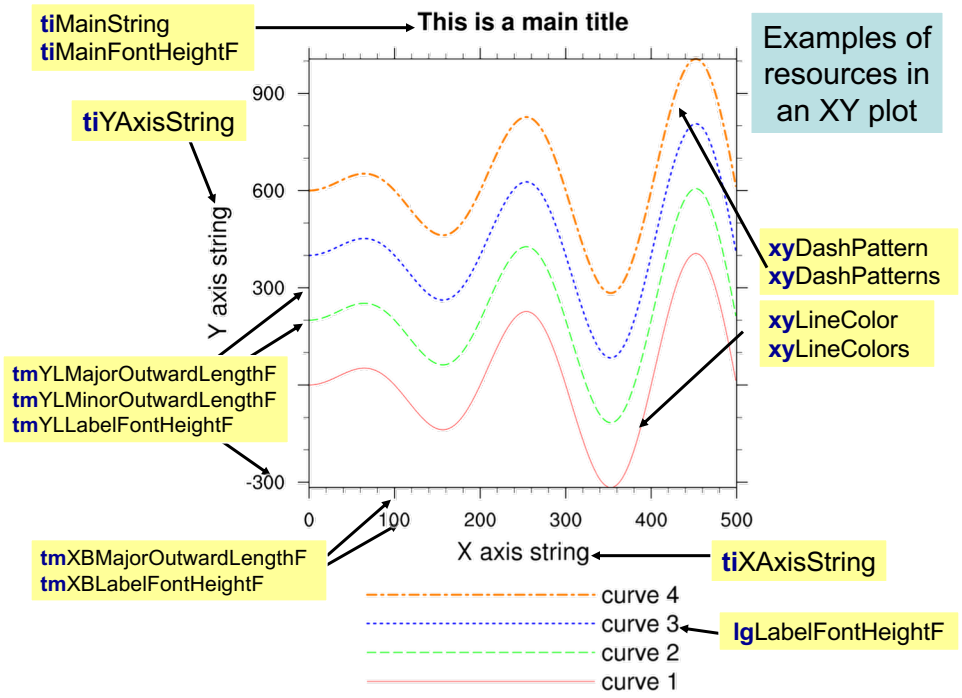



Description of resources (cont'd)

- Resources across objects are similarly named for easier recollection:

```
xyLineColor, cnLineColor, gsLineColor,
mpGridLineColor, tmBorderLineColor

tiMainFontHeightF, tmXBLabelFontHeightF,
lbLabelFontHeightF, cnLineLabelFontHeightF

xyDashPattern, mpPerimLineDashPattern,
lbBoxLineDashPattern, cnLineDashPattern

and so on…
```

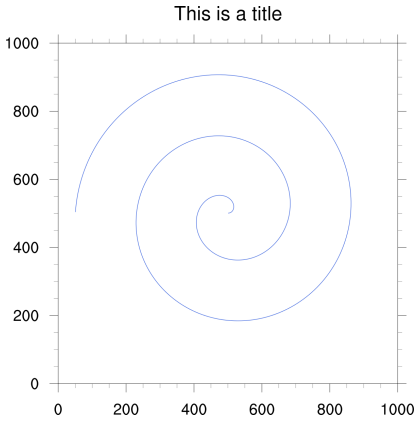





Step 4: Call appropriate plotting function

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

begin
  ii = ispan(0,500,1)
  x  = 501 + .9 * ii * cos(0.0314*ii)    ; 501 points
  y  = 501 + .9 * ii * sin(0.0314*ii)

  wks = gsn_open_wks("png", "spiral")   ; spiral.png

  res                   = True          ; Plot options
  res@xyLineColor       = "RoyalBlue"   ; line color
  res@xyLineThicknessF  = 3.0           ; line thickness
  res@tiMainString      = "This is a title"

  plot = gsn_csm_xy(wks,x,y,res)
end
```

```
ii = ispan(0,500,1)
x  = 501 + .9 * ii * cos(0.0314*ii)    ; 501 points
y  = 501 + .9 * ii * sin(0.0314*ii)

wks = gsn_open_wks("png", "spiral")   ; spiral.png

res                   = True          ; Plot options
res@xyLineColor       = "RoyalBlue"   ; line color
res@xyLineThicknessF  = 3.0           ; line thickness
res@tiMainString      = "This is a title"

plot = gsn_csm_xy(wks,x,y,res)
```

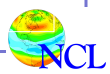


NCL Graphics topics

- Types of graphics you can create with NCL
- The basics
- Line-by-line examples
- Review
- Customizing NCL environment
- Common mistakes and problems
- Debugging tips
- Creating images for Web and/or presentations
- PyNGL/PyNIO Python modules

Line-by-line examples

- XY plots
- Use of "gsnMaximize" resource
- Contour plot
- Contour plot over map
- Contouring curvilinear and unstructured data
- WRF plots
- Vector plot
- Panel plots
- Shapefile outlines



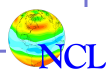
Line-by-line example scripts

Most NCL scripts that follow can be viewed and downloaded from the web:

http://www.ncl.ucar.edu/Training/Workshops/Scripts/

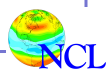
Scripts have names like *xy2a.ncl*, *xy2b.ncl*, …

The first one is usually one with no resources set, and each subsequent script adds a few more resources.



Example *xy2a.ncl*

- gsn_csm_y
- Read data from a NetCDF file
- XY plot with one curve
- No resources (plot options) set



ncl_filedump uv300.nc

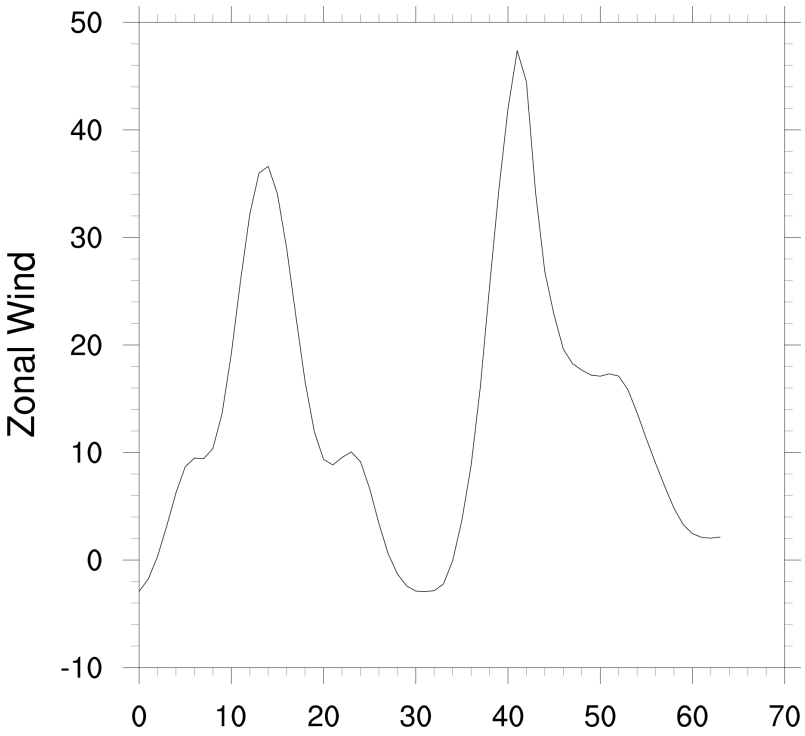
```
filename:   uv300
path: uv300.nc
   file global attributes:
      title : UV300: January and July
      source : Climate Analysis Section, NCAR
      history : Dataset uv300.hs from EZPLOT demo dataset
      Conventions : None
      creation_date : Mon Mar 29 09:24:57 MST 1999
  dimensions:
      lat = 64
      lon = 128
      time = 2
   variables:
      float U ( time, lat, lon )
         _FillValue :   -999
         long_name :    Zonal Wind
         short_name :   U
         units :  m/s
```

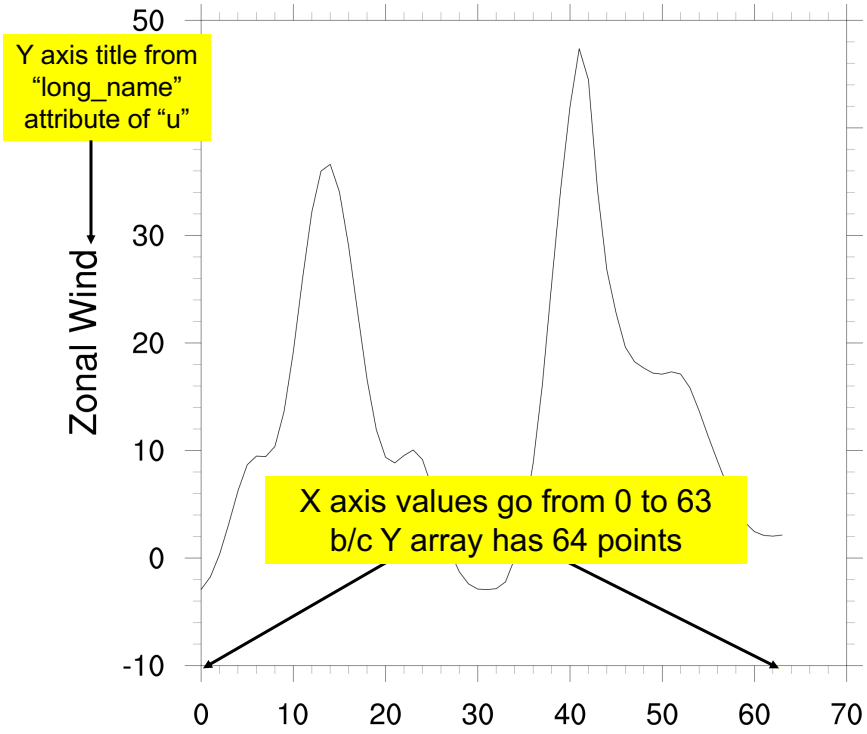
```
begin
  f = addfile (“uv300.nc”,”r”)
  u = f->U(0,:,{82})           ; Read “U” off the file
                               ; “u” will have 64 values

  printVarSummary(u)     ; Important: Look at your data!

  wks = gsn_open_wks (“x11”, “xy2a”)  ; “ps”, “png”

  res  = True                  ; No plot options yet.
  plot = gsn_csm_y (wks,u,res) ; Draw an XY plot
end
```

printVarSummary(u)

```
Variable: u
Type: float
Total Size: 256 bytes
            64 values
Number of Dimensions: 1
Dimensions and sizes:   [lat | 64]
Coordinates:
            lat: [-87.8638..87.8638]
Number Of Attributes: 6
  lon :      81.5625
  time :     1
  _FillValue :     -999
  long_name :      Zonal Wind
  short_name :     U
  units :    m/s
```

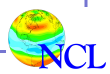
This will get used for Y axis title


Zonal Wind
50
40
30
20
10
0
-10
0
10
20
30
40
50
60
70

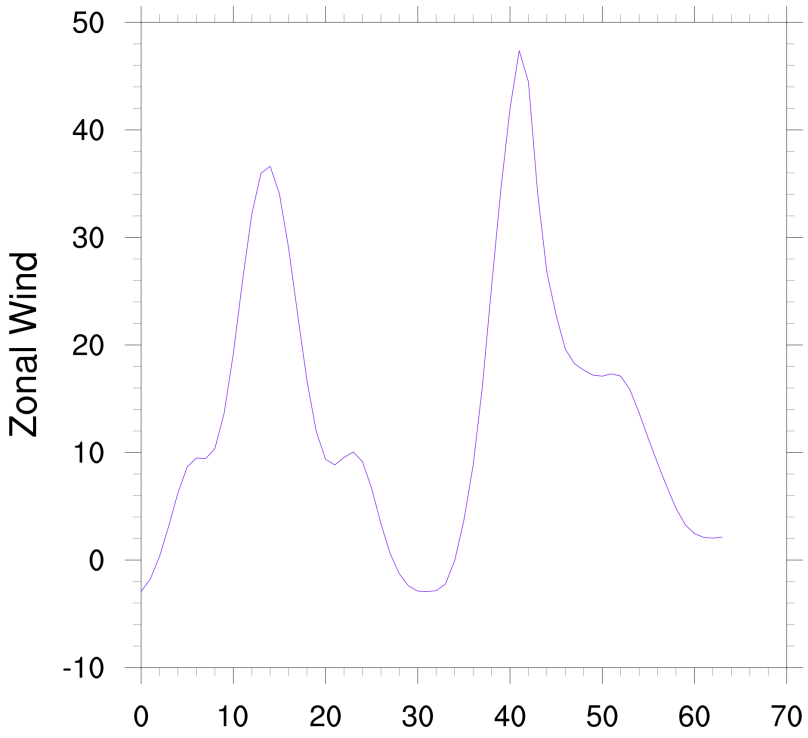

Y axis title from "long_name" attribute of "u"
Zonal Wind
X axis values go from 0 to 63 b/c Y array has 64 points
50
30
20
10
0
-10
0
10
20
30
40
50
60
70

Example *xy2b.ncl*

- Line color changed
  (using a color index value)
- Default color map discussed
- Resource introduced:
  - xyLineColor - sets line color




```
begin
  f = addfile ("uv300.nc","r")
  u = f->U(0,:,{82})           ; Read "U" off the file

  wks = gsn_open_wks ("x11", "xy2b")

  res             = True
  res@xyLineColor = 2      ; Old way of setting color

  plot = gsn_csm_y (wks,u,res) ; Draw an XY plot
end
```



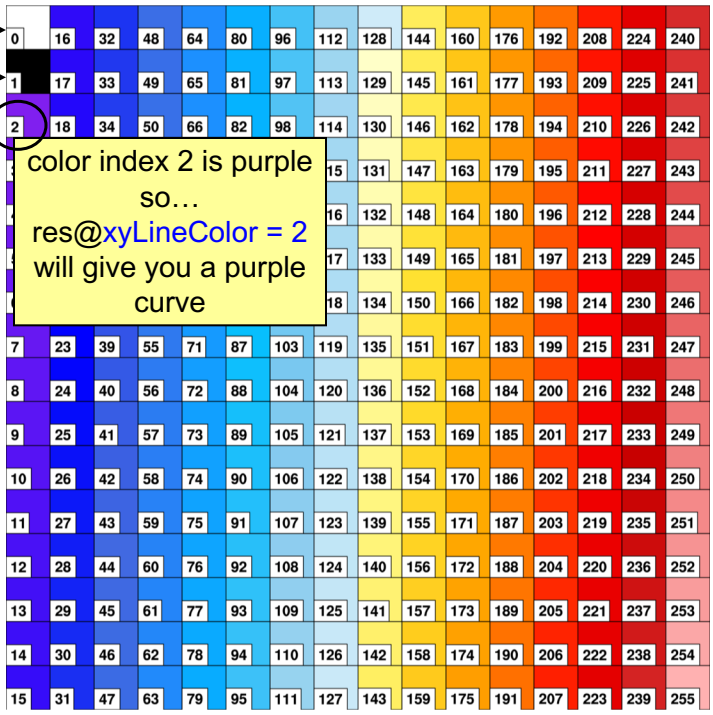
Index 0 is the background color

Index 1 is the foreground color

color index 2 is purple so… res@xyLineColor = 2 will give you a purple curve

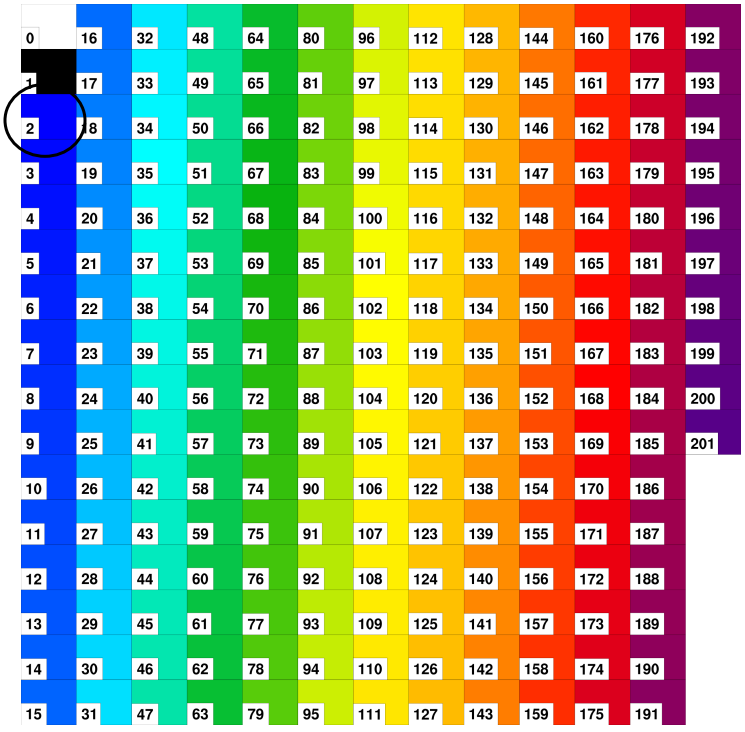
Integer values used with color resources will be indexes into the current color table

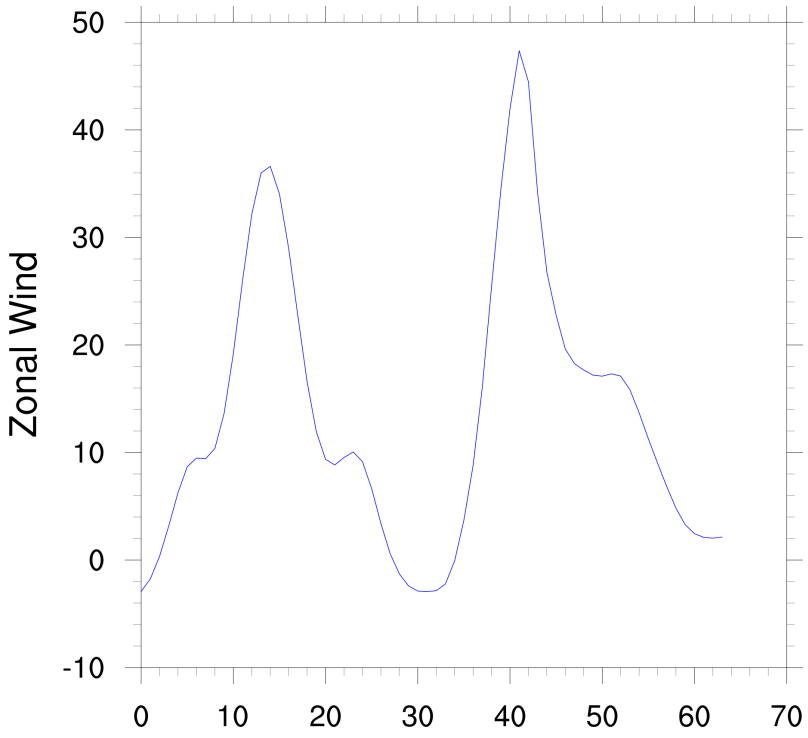
Default color table

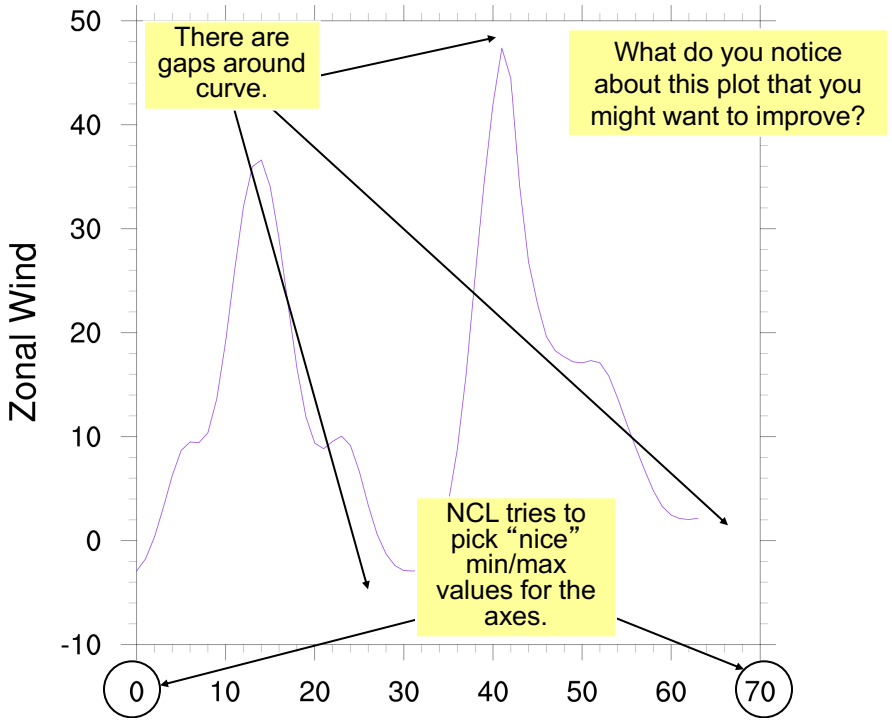


Side topic: how to change workstation color map

```
begin
  f = addfile (“uv300.nc”,”r”)
  u = f->U(0,:,{82})           ; Read “U” off the file

  wks = gsn_open_wks (“x11”, “xy2b_color1”)
  gsn_define_colormap(wks,”BlAqGrYeOrReVi200”)

  res             = True
  res@xyLineColor = 2     ; Old way of setting color

  plot = gsn_csm_y (wks,u,res)   ; Draw an XY plot
end
```

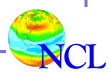



Zonal Wind
50
40
30
20
10
0
-10
0
10
20
30
40
50
60
70

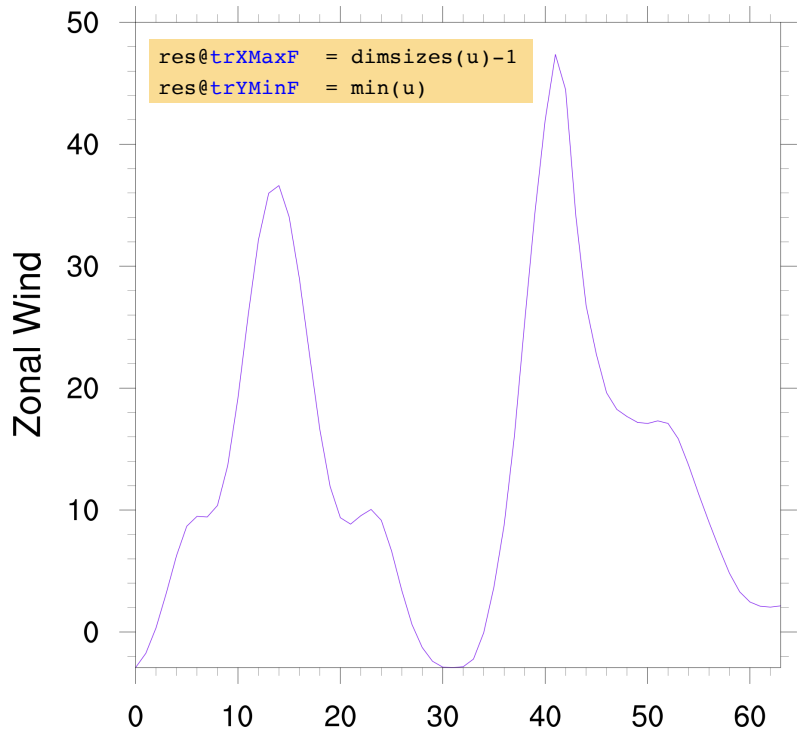

Zonal Wind
There are gaps around curve.
What do you notice about this plot that you might want to improve?
NCL tries to pick “nice” min/max values for the axes.
50
40
30
20
10
0
-10
0
10
20
30
40
50
60
70

Example *xy2b_zoom1.ncl*

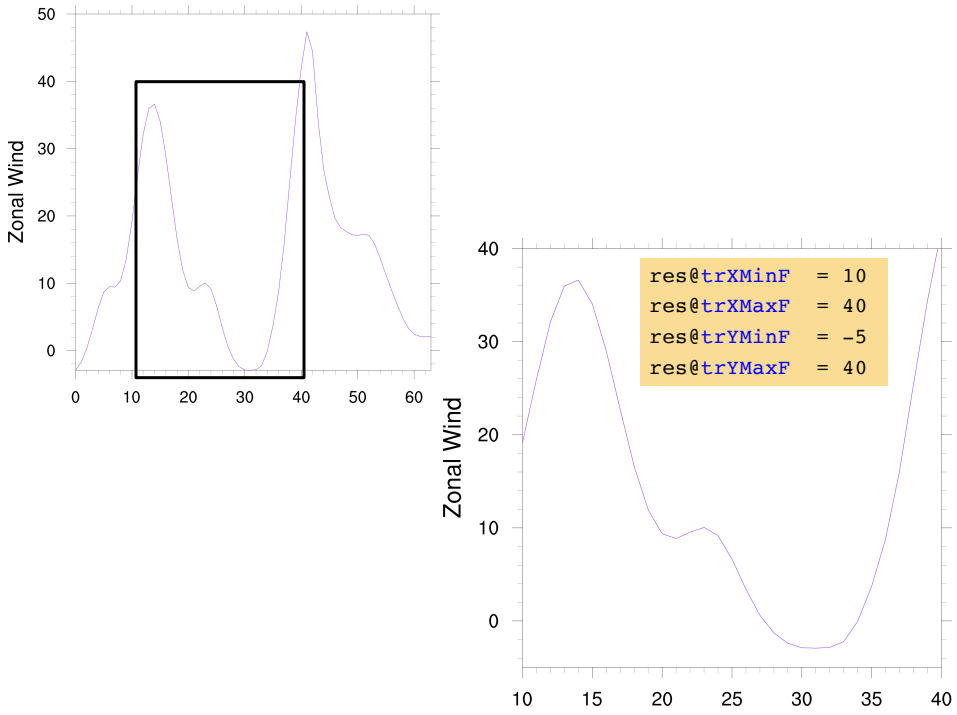
- Resources introduced:
  - trYMinF, trYMaxF, trXMinF, trXMaxF - sets min and max values for X and Y axes (**tr**ansformation resources)
- These resources can also be applied to contour, vector, and streamline plots
- Map plots have their own set of resources for zooming in on maps



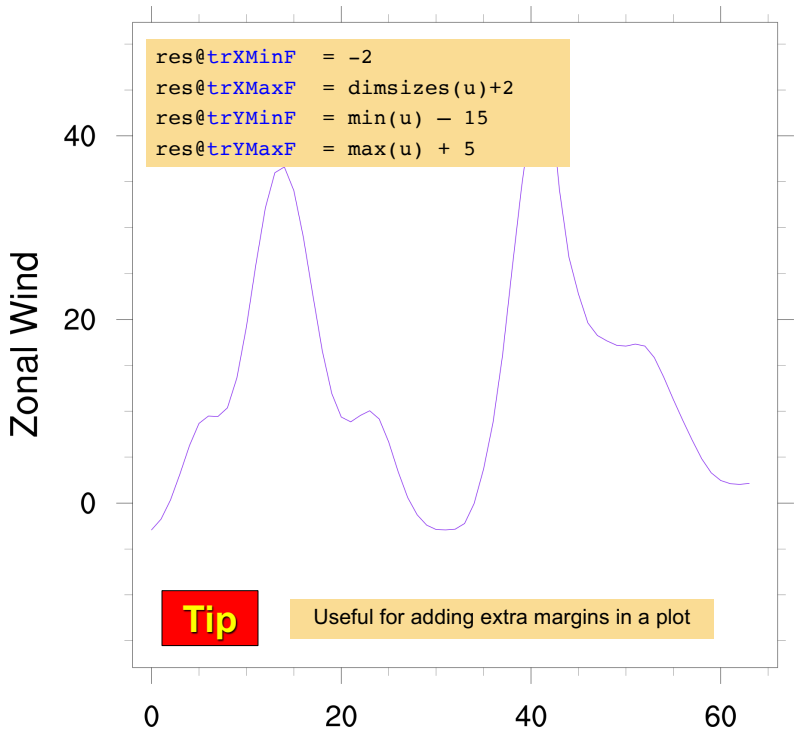

```
begin
  f = addfile ("uv300.nc","r")
  u = f->U(0,:,{82})           ; Read "U" off the file

  wks = gsn_open_wks ("x11", "xy2b_zoom1")

  res             = True
  res@xyLineColor = 2     ; Old way of setting color

  res@trXMaxF   = dimsizes(u)-1  ; Set X axis max to 63
  res@trYMinF   = min(u)         ; Set Y axis min to min(u)

  plot = gsn_csm_y (wks,u,res)   ; Draw an XY plot
end
```




```
begin
  f = addfile (“uv300.nc”,”r”)
  u = f->U(0,:,{82})           ; Read “U” off the file

  wks = gsn_open_wks (“x11”, “xy2b_zoom2”)

  res             = True
  res@xyLineColor = 2     ; Old way of setting color

  res@trXMinF     = 10        ; Zoom IN on a region
  res@trXMaxF     = 40
  res@trYMinF     = -5
  res@trYMaxF     = 40

  plot = gsn_csm_y (wks,u,res) ; Draw an XY plot
end
```




```
begin
  f = addfile ("uv300.nc","r")
  u = f->U(0,:,{82})           ; Read "U" off the file

  wks = gsn_open_wks ("x11", "xy2b_zoom3")

  res             = True
  res@xyLineColor = 2          ; Old way of setting color

  res@trXMinF     = -2           ; Zoom OUT of a region
  res@trXMaxF     = dimsizes(u)+2
  res@trYMinF     = min(u) - 15
  res@trYMaxF     = max(u) + 5

  plot = gsn_csm_y (wks,u,res) ; Draw an XY plot
end
```

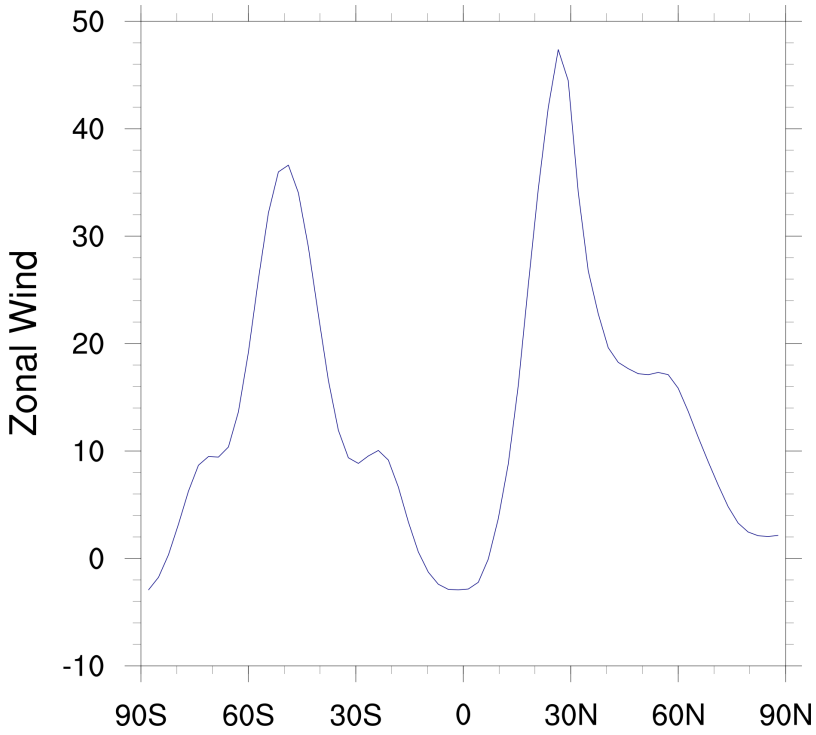


Example *xy2c.ncl*

- gsn_csm_xy
- Use "u&lat" for X axis
- Better way of setting line color (using "named" color)
- Line thickness increased
- Resource introduced:
  - xyLineThicknessF - sets line thickness

```
begin
  f = addfile (“uv300.nc”,”r”)
  u = f->U(0,:,{82})           ; Read “U” off the file

  wks = gsn_open_wks (“x11”, “xy2c”)    ; “ps”, “png”

  res                          = True
  res@xyLineColor      = “NavyBlue”    ; Named color
  res@xyLineThicknessF = 3             ; 3x as thick

  plot = gsn_csm_xy (wks,u&lat,u,res) ; Draw an XY plot
end
```

Order of resources not important

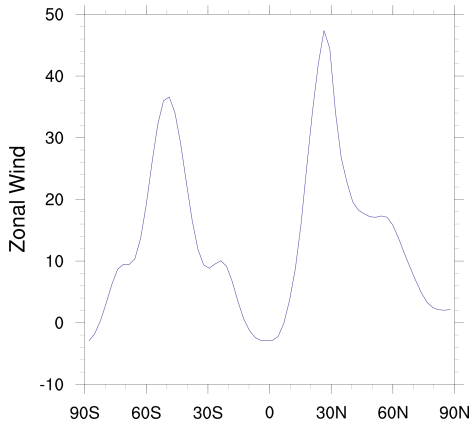


Samples of named colors

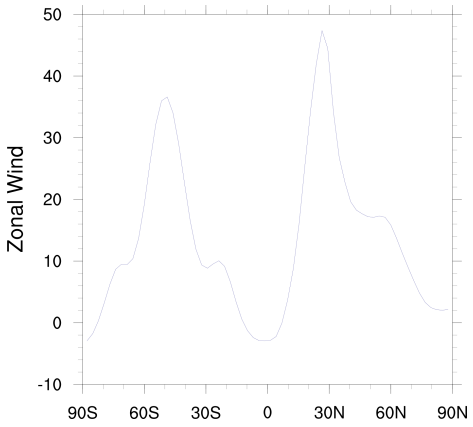
http://www.ncl.ucar.edu/Document/Graphics/named_colors.shtml

Why should I care about line thickness?

3x line thickness



Default line thickness



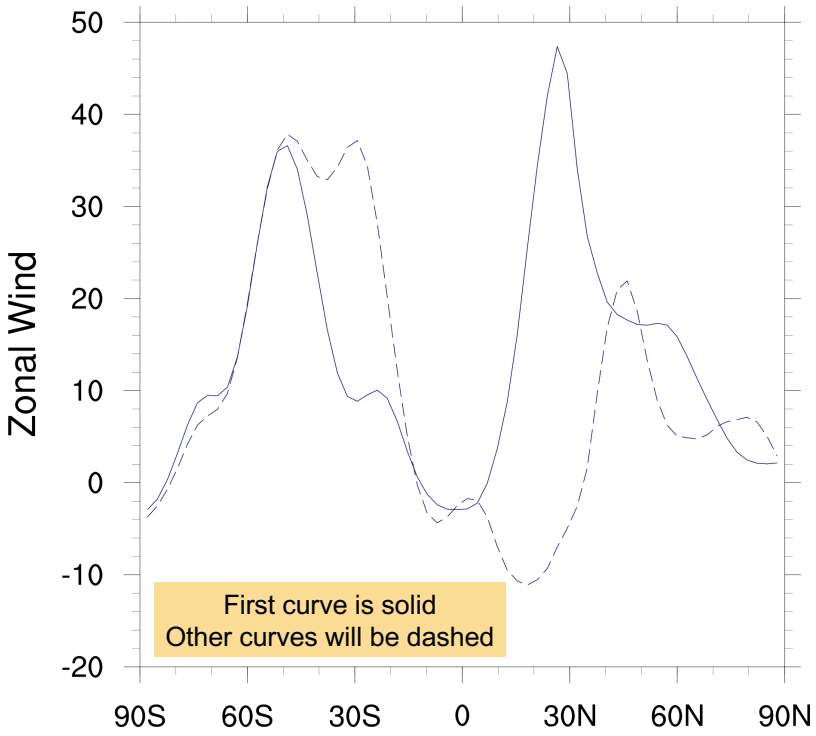
Tip Useful for improving the look of your graphics for large displays

Example *xy2d.ncl* – read both time steps

```
f = addfile ("uv300.nc","r")
u = f->U(:,:,{82})        ; Read "U" off the file
                          ; "u" will be 2 x 64 2D array
printVarSummary(u)

wks = gsn_open_wks ("x11", "xy2d")    ; "ps", "png"

res                  = True
res@xyLineColor      = "NavyBlue"   ; Named color
res@xyLineThickness  = 3            ; 3x as thick

plot = gsn_csm_xy (wks,u&lat,u,res) ; Draw an XY plot
```

printVarSummary(u)

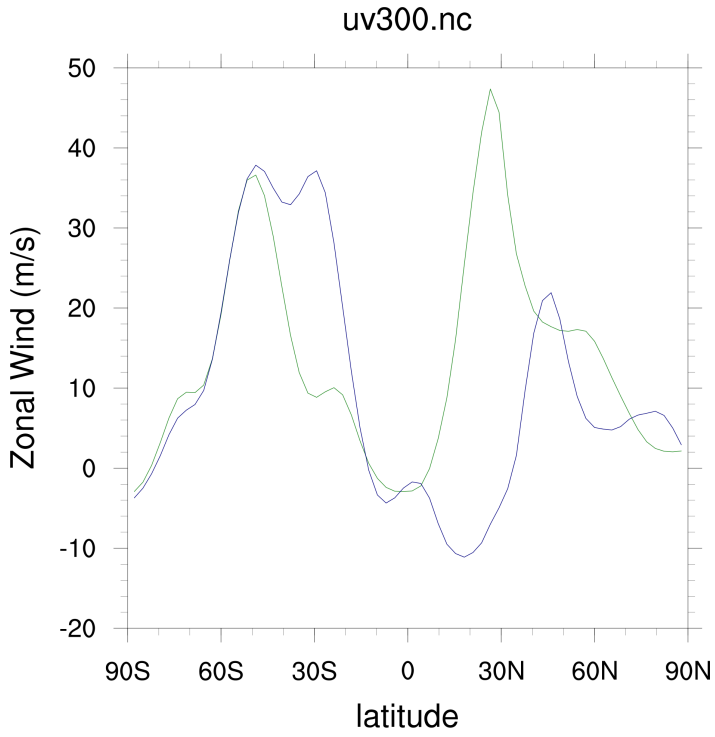
```
Variable: u
Type: float
Total Size: 512 bytes
            128 values
Number of Dimensions: 2
Dimensions and sizes:   [time | 2] x [lat | 64]
Coordinates:
            time: [1..7]
            lat: [-87.8638..87.8638]
Number Of Attributes: 5
  lon :     81.5625
  _FillValue :    -999
  long_name :     Zonal Wind
  short_name :    U
  units :   m/s
```



Example *xy2e.ncl*

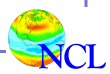
- Resources introduced:
  - xyDashPattern - sets dash pattern for curve
  - xyLineColors – sets colors for multiple curves
  - tiMainString, tiXAxisString, tiYAxisString - sets strings for axes and main title, can also be used for contour, vector, etc, plots
  - Note: tiXAxisString and tiYAxisString will override "long_name" attributes

```
begin
  fname = "uv300.nc"
  f     = addfile (fname,"r")
  u     = f->U(:,:,{82})        ; Read "U" off the file

  wks = gsn_open_wks ("x11", "xy2e")    ; "ps", "png"

  res                   = True

  res@xyLineThicknessF = 3           ; 3x thicker
  res@xyDashPattern    = 0           ; Solid curve
  res@xyLineColors     = (/"ForestGreen", "NavyBlue"/)

  res@tiMainString     = fname
  res@tiYAxisString    = u@long_name +" (" + u@units + ")"
  res@tiXAxisString    = u&lat@long_name

  plot = gsn_csm_xy (wks,u&lat,u,res) ; Draw an XY plot
end
```

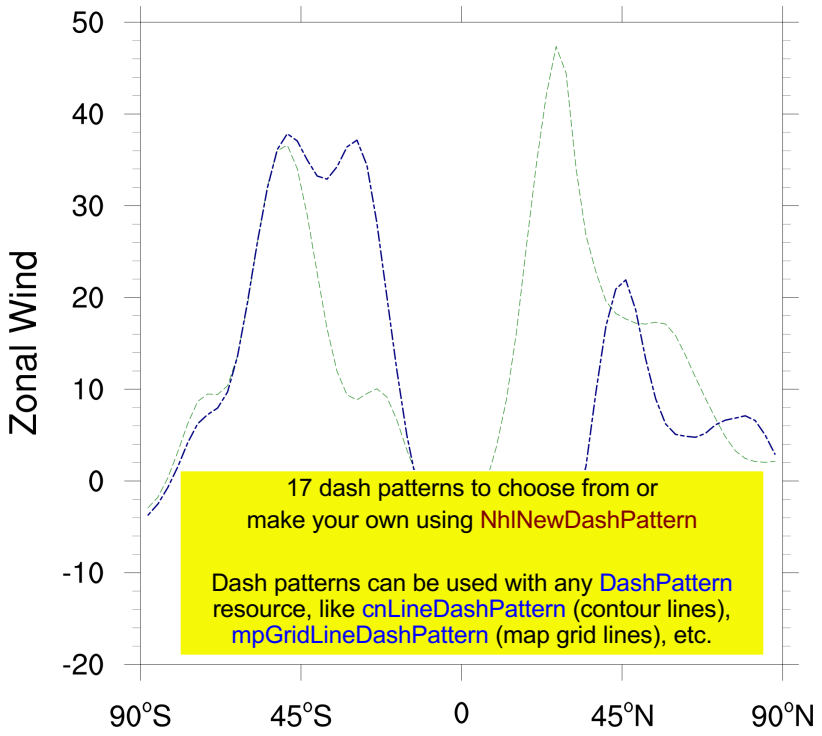


Example *xy2f.ncl*

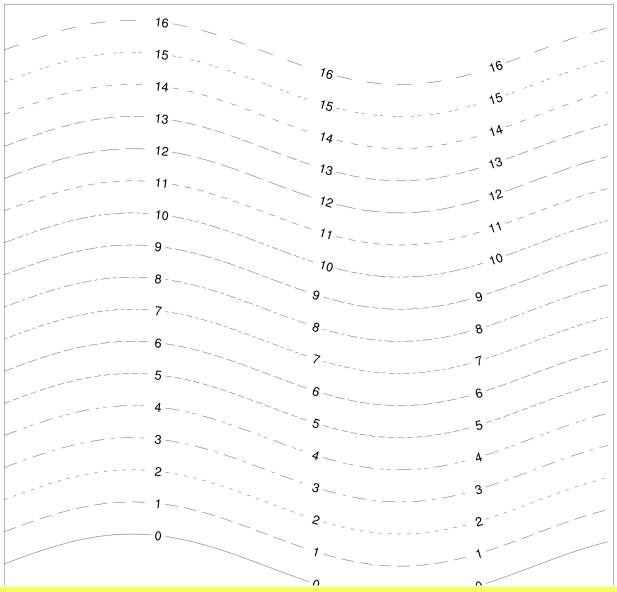
- Use dash patterns for both curves
- Explicitly set labels for bottom tickmarks
- Resources introduced:
  - xyDashPatterns
  - tmXBMode, tmXBValues, tmXBLabels – sets locations and strings for bottom tickmarks
- Note about tickmark resources: all four axes have their own set of tickmarks, for example:
  - tmYLMode (Y left), tmYRMode (Y right), tmXBMode (X bottom), tmXTMode (X top)



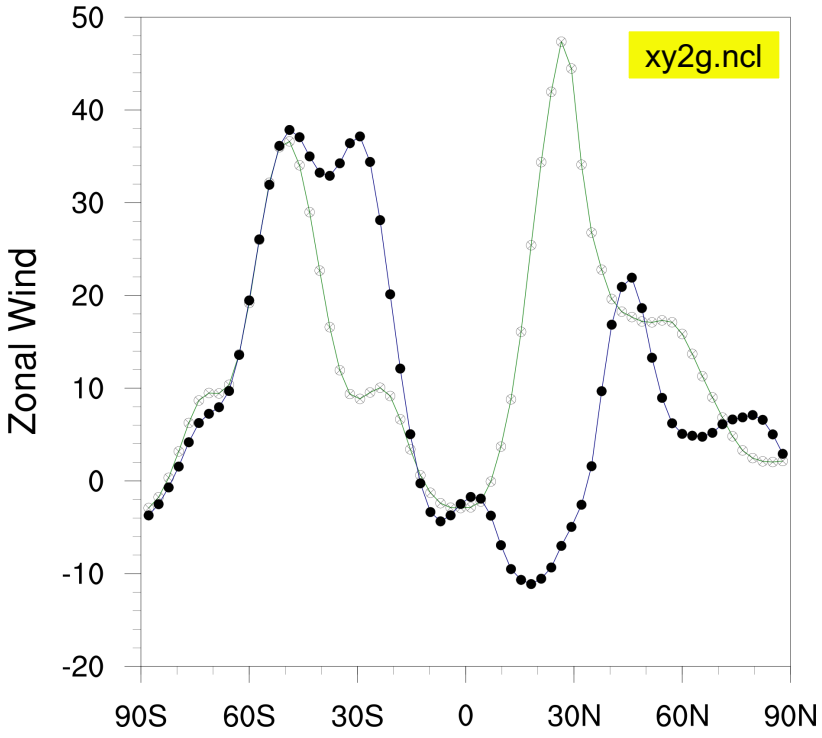

```
f = addfile ("uv300.nc","r")
u = f->U(:,:,{82})        ; Read "U" off the file

wks = gsn_open_wks ("x11", "xy2f")    ; "ps", "png"

res                   = True

res@xyDashPatterns    = (/5,8/)         ; Dashed patterns
res@xyLineColors      = (/"ForestGreen", "NavyBlue"/)
res@xyLineThicknesses = (/3,6.5/)  ; 3x and 6.5x thicker

res@tmXBMode          = "Explicit"    ; Label X axis
res@tmXBValues        = (/-90,-45,0,45,90/)
res@tmXBLabels        = (/"90~S~o~N~S", "45~S~o~N~S", \
                          "0","45~S~o~N~N","90~S~o~N~N"/)

res@tmXBMinorOn       = False   ; Turn off minor ticks

plot = gsn_csm_xy (wks,u&lat,u,res) ; Draw an XY plot
```



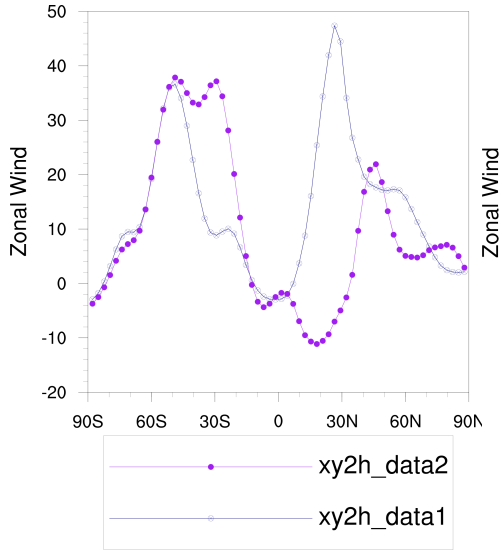
Predefined dash patterns and their index numbers



pre-defined dash patterns:
http://www.ncl.ucar.edu/Document/Graphics/dash_patterns.shtml

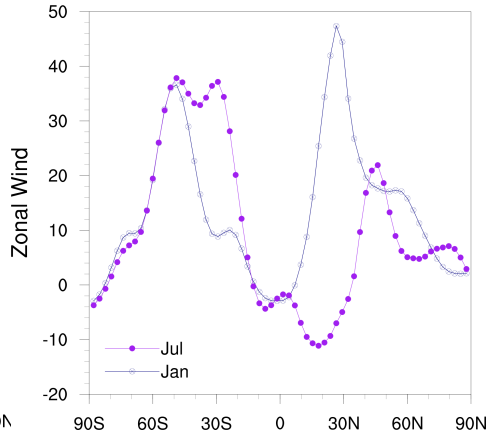


xy2h.ncl: default legend



NCL V6.4.0 now has a "simple_legend" function

xy2i.ncl: customized legend



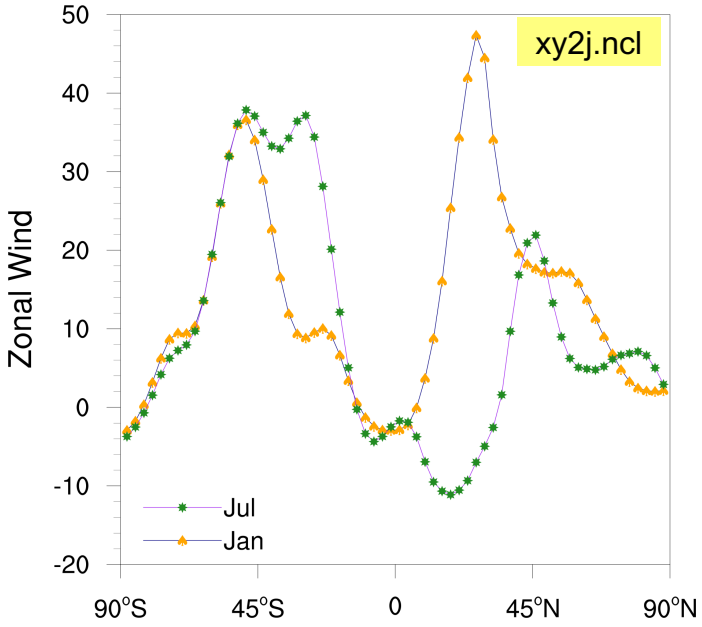
Tip

Default legend not pretty. Use:

```
xyExplicitLegendLabels
lgLabelFontHeightF
pmLegendWidthF
pmLegendHeightF
pmLegendOrthogonalPosF
pmLegendParallelPosF
```

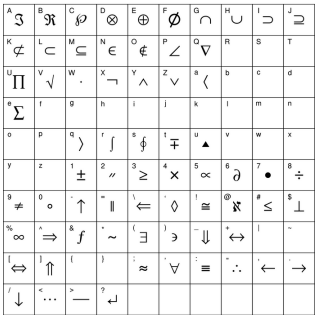
Tip

Customized markers: NhlNewMarker function

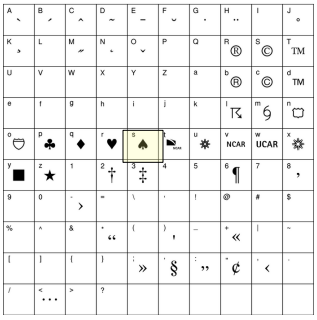



```
spade = NhlNewMarker(wks, "s", 35, 0, 0, 1, 1, 0)
res@xyMarker = spade
```

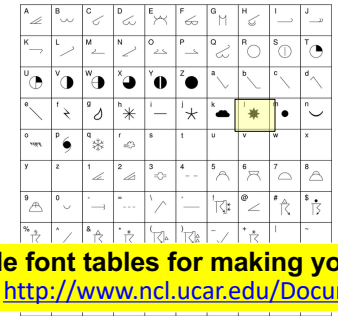
Font 34, math-symbols



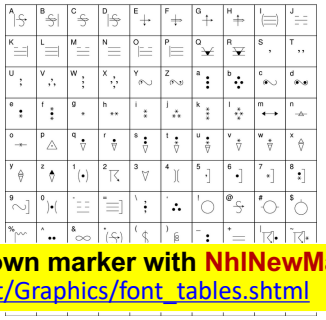
Font 35, text-symbols



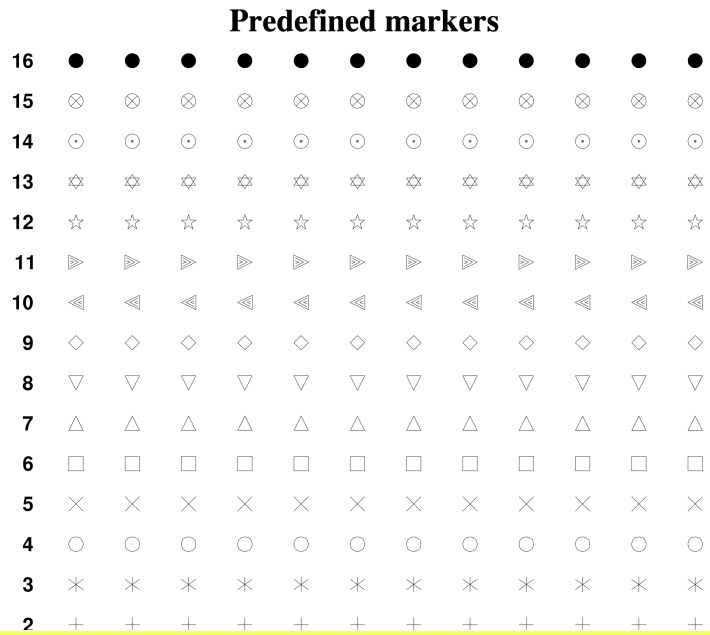
Font 37, weather2



Font 36, weather1



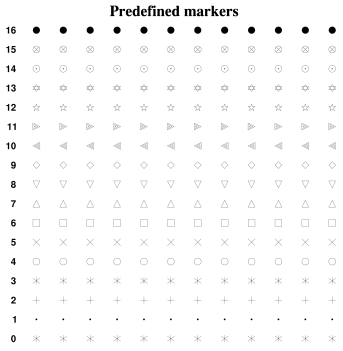
Sample font tables for making your own marker with NhlNewMarker
http://www.ncl.ucar.edu/Document/Graphics/font_tables.shtml

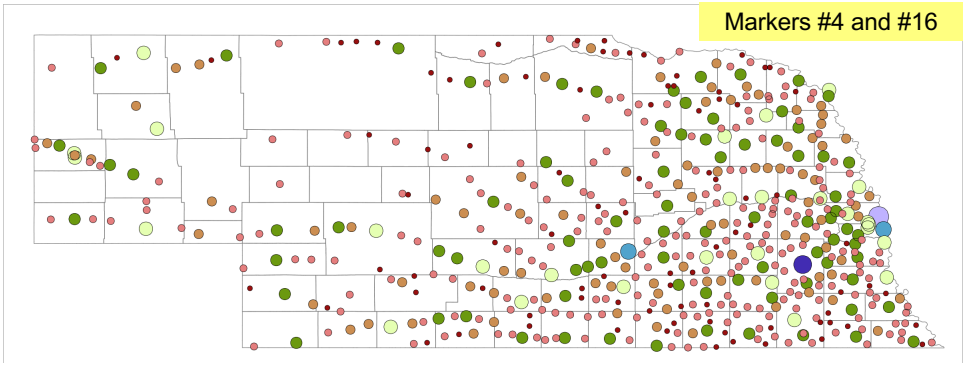


pre-defined markers:
http://www.ncl.ucar.edu/Document/Graphics/marker_styles.shtml

Tip

I like to use filled and hollow markers together to get an outline effect.

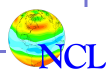




Markers #4 and #16

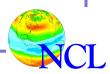
Advanced topics

- Two ways to add data to an existing XY plot
  http://www.ncl.ucar.edu/Applications/xy.shtml#ex25
  http://www.ncl.ucar.edu/Applications/xy.shtml#ex26
- Filling the area between two curves
  http://www.ncl.ucar.edu/Applications/xy.shtml#ex24
- Turning XY curves into individual bars
  http://www.ncl.ucar.edu/Applications/bar.shtml
- Changing an axis (log, irregular, linear)
  http://www.ncl.ucar.edu/Applications/axes.shtml#ex3



Line-by-line examples

- XY plots
- Use of "gsnMaximize" resource
- Contour plot
- Contour plot over map
- Contouring curvilinear and unstructured data
- WRF plots
- Vector plot
- Panel plots
- Shapefile outlines



Special topic: Maximizing plots in a frame

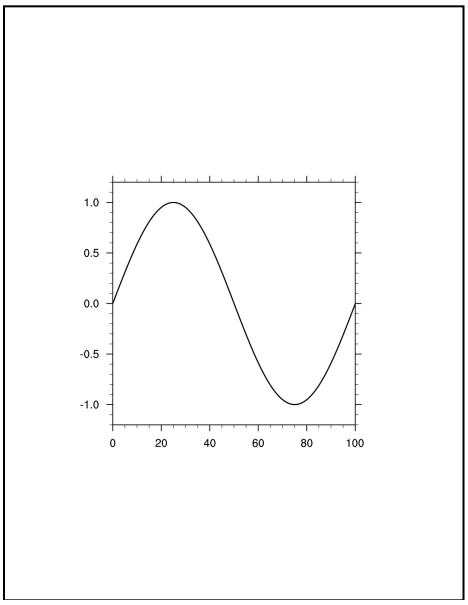
- By default, no maximization takes place
- If you want to maximize, set:

  res@gsnMaximize = True

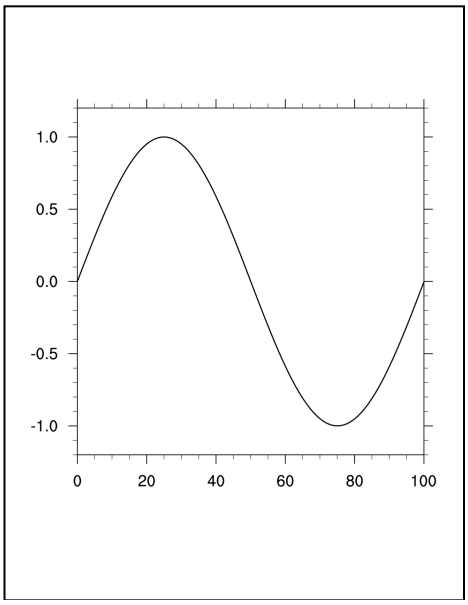
- Aspect ratio will be preserved, but positions and plot size will change
- By default, plot will be rotated for PS or PDF output if appropriate ("landscape")

How gsnMaximize affects output to PS file on an 8-1/2" x 11" page

res@gsnMaximize = False

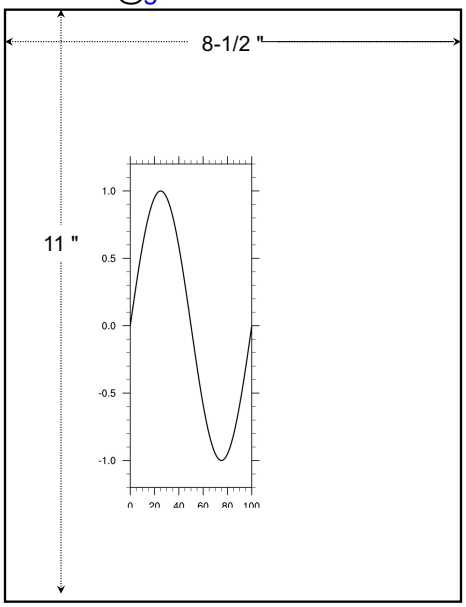


res@gsnMaximize = True

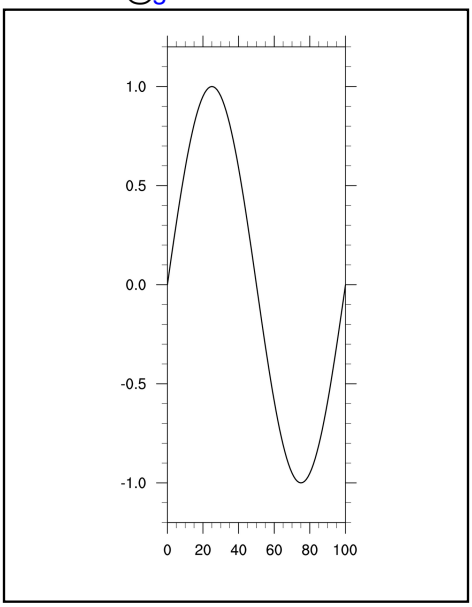


How gsnMaximize affects output to PS file on an 8-1/2” x 11” page

res@gsnMaximize = False



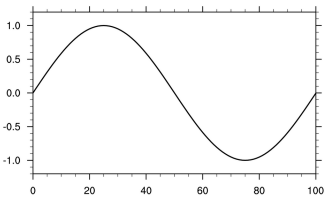
res@gsnMaximize = True



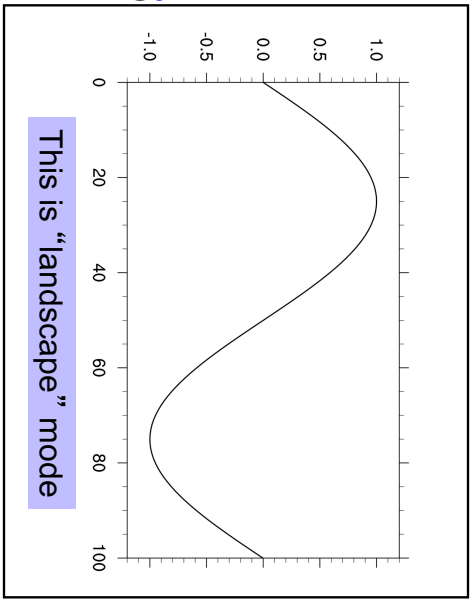
How gsnMaximize affects output to PS file on an 8-1/2” x 11” page

res@gsnMaximize = False

This is “portrait” mode

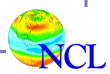


res@gsnMaximize = True



Line-by-line examples

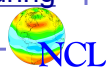
- XY plots
- Use of "gsnMaximize" resource
- Contour plot
- Contour plot over map
- Contouring curvilinear and unstructured data
- WRF plots
- Vector plot
- Panel plots
- Shapefile outlines



Example *contour1a.ncl*

- gsn_csm_contour
- Simple contour plot
- Data retrieved from a NetCDF file
- Data has _FillValue attribute, which is recognized by gsn_csm script as data to not plot
- No plot options (resources) set

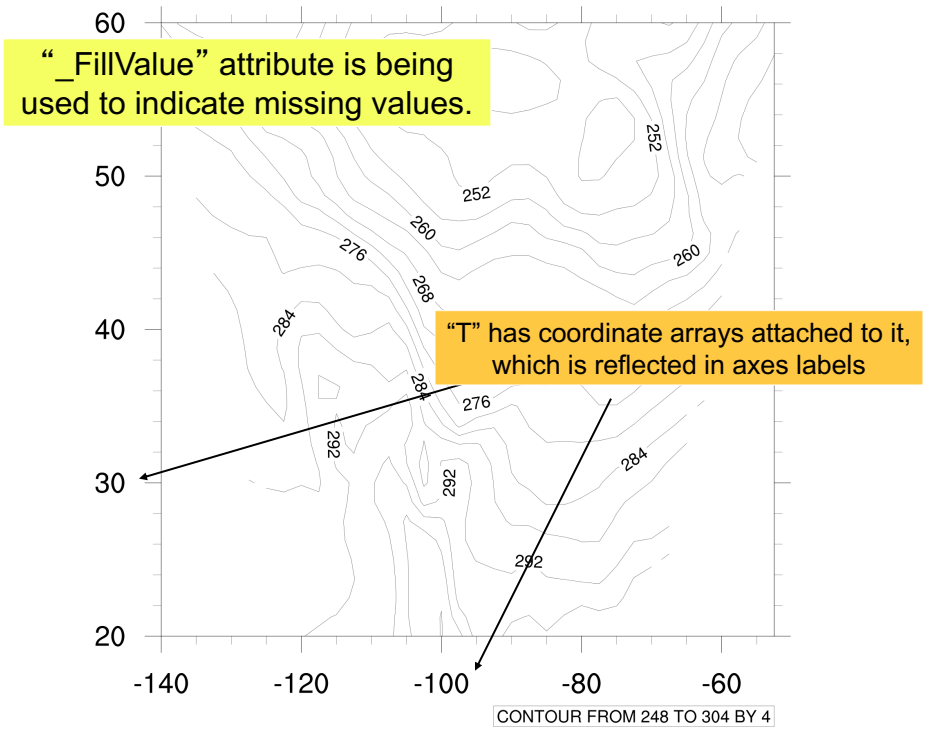
http://www.ncl.ucar.edu/Training/Workshops/Scripts/#Contouring



```
begin
  tf = addfile("Tstorm.cdf","r")
  T  = tf->t(0,:,:)     ; Read first time step
                        ; of temperature data.
  printVarSummary(T)    ; Look at T
  printMinMax(T,0)      ; min=245.152   max=304.152

  wks = gsn_open_wks("png","contour1a")

  res = True            ; No plot options set.
  plot = gsn_csm_contour(wks,T,res)
end
```

Output from "**printVarSummary**(T)"

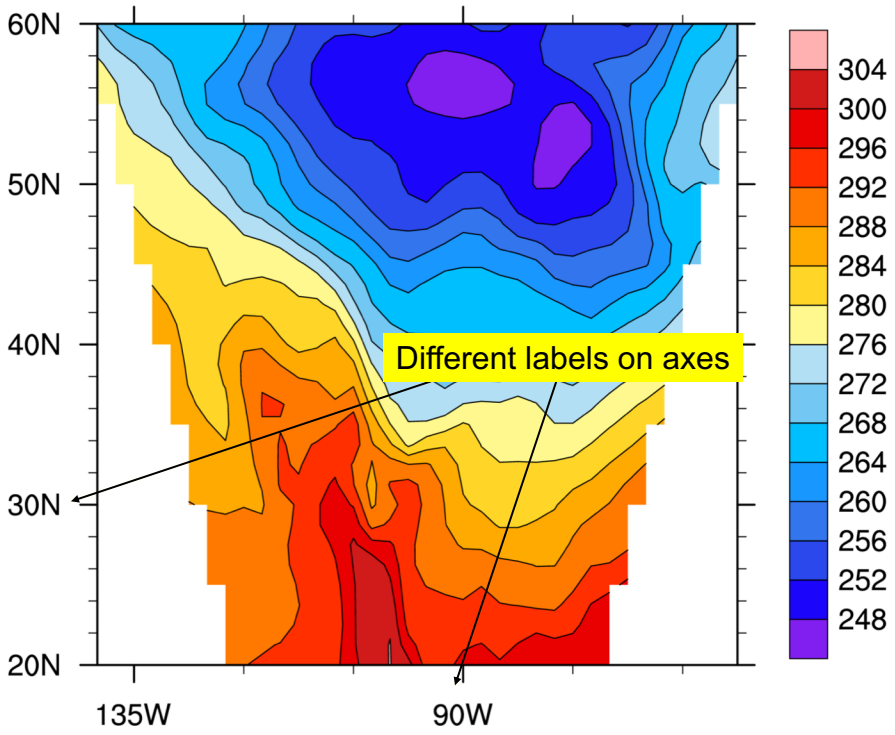
```
Variable: T
Type: float
Total Size: 4752 bytes
            1188 values
Number of Dimensions: 2
Dimensions and sizes:   [lat | 33] x [lon | 36]
Coordinates:
            lat: [20..60]
            lon: [-140..-52.5]
Number Of Attributes: 2
  timestep :    0
  _FillValue :  -9999
```



Example *contour1b.ncl*

- Adding "units" attribute to both lat and lon coordinate arrays
- Contour fill turned on
- Resources introduced:
  - cnFillOn - turn on contour fill
  - lbOrientation - change labelbar orientation

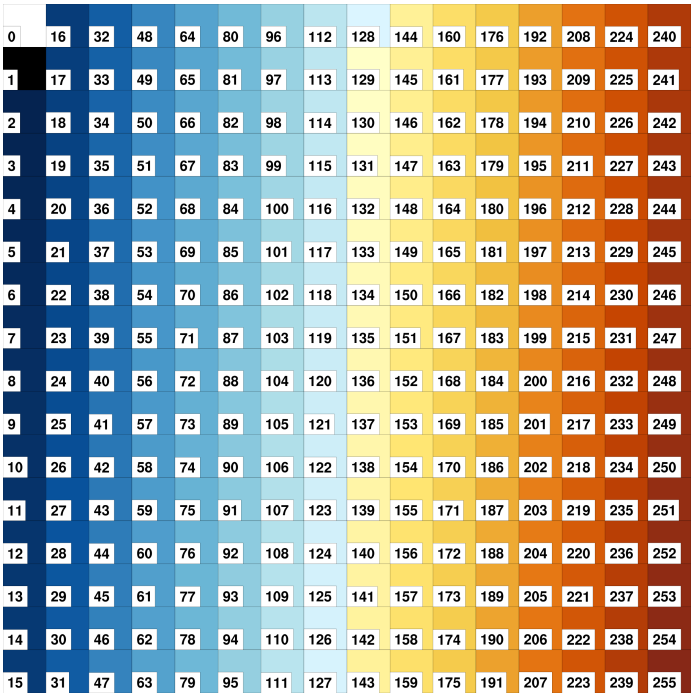
```
begin
  tf = addfile("Tstorm.cdf","r")
  T  = tf->t(0,:,:)              ; Get first time step
  T&lon@units = "degrees_east"  ; Add some units
  T&lat@units = "degrees_north"

  wks = gsn_open_wks("x11","contour1b")

  res                 = True
  res@cnFillOn        = True        ; Turn on contour fill
  res@lbOrientation = "Vertical"  ; Move labelbar

  plot = gsn_csm_contour(wks,T,res)
end
```



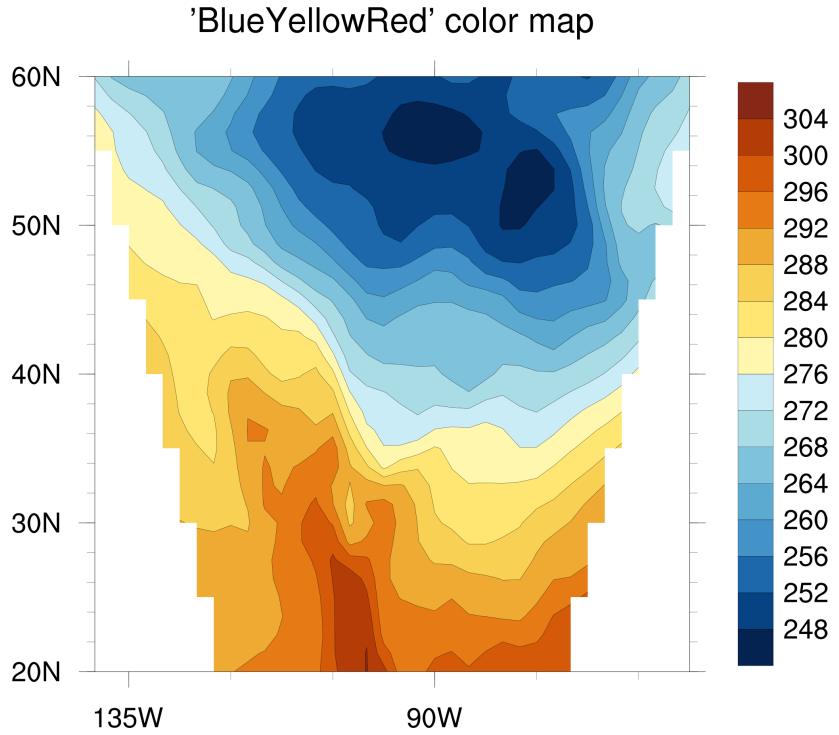
Example *contour1d.ncl*

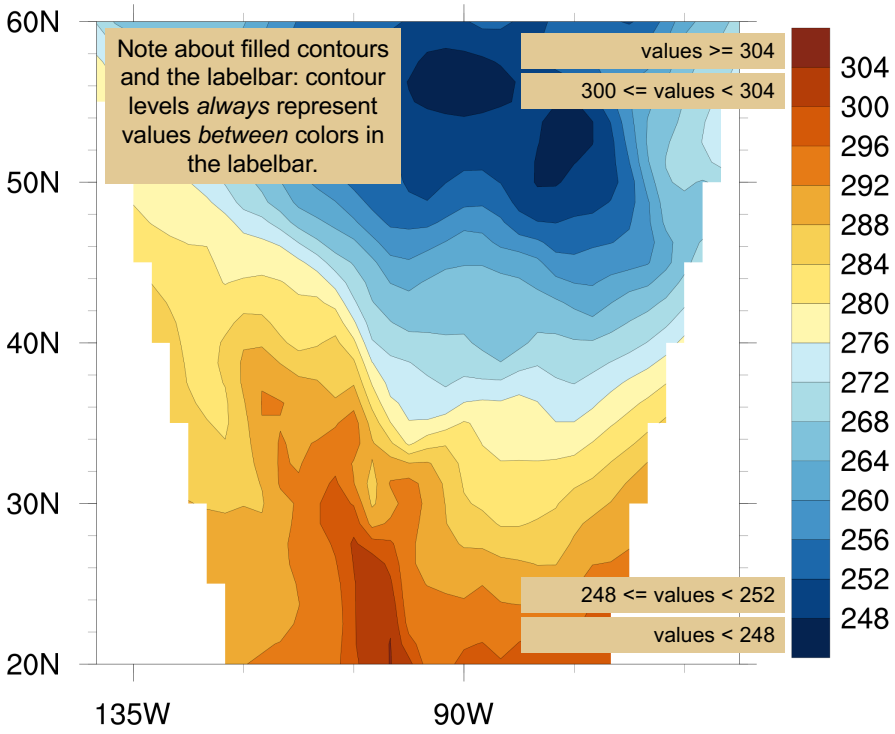
- Color map changed
- Title added
- Resources introduced:
  - cnFillPalette – change contour color map

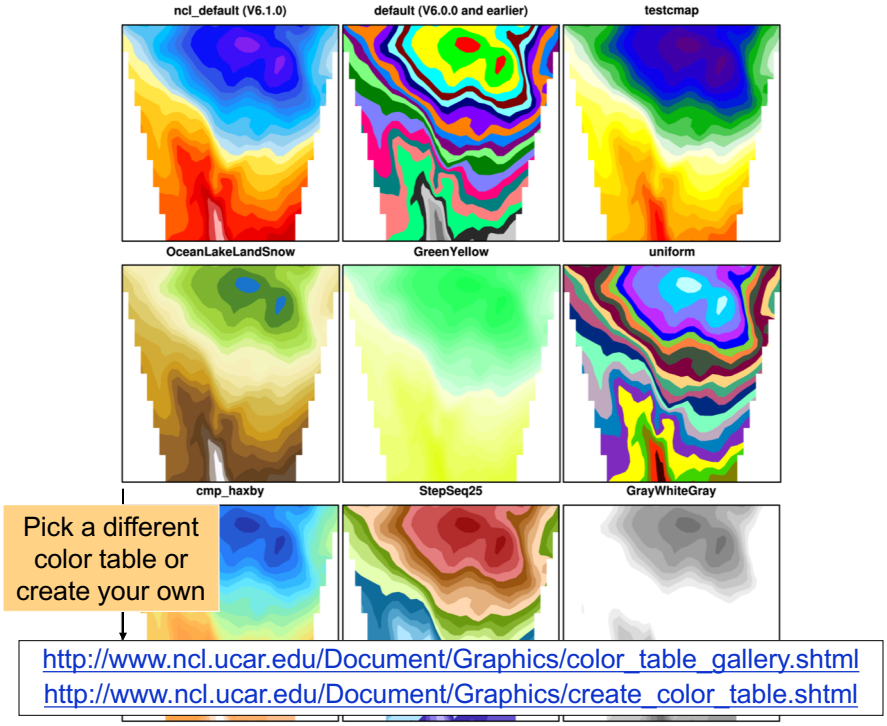
"**BlueYellowRed**" color map



```
begin
  tf = addfile("Tstorm.cdf","r")
  T  = tf->t(0,:,:)
  T&lon@units = “degrees_east”  ; Add some units
  T&lat@units = “degrees_north”

  wks = gsn_open_wks("x11","contour1d")

 ;gsn_define_colormap(wks,"BlueYellowRed") ; old way of changing
                                           ; colormap; NOT
                                           ; recommended!
  res                  = True
  res@cnFillOn         = True        ; Turn on contour fill
  res@lbOrientation    = “Vertical”  ; Move labelbar
  res@cnFillPalette    = “BlueYellowRed”

  plot = gsn_csm_contour(wks,T,res)
end
```



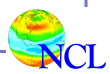

60N
50N
40N
30N
20N
135W
90W
Note about filled contours and the labelbar: contour levels *always* represent values *between* colors in the labelbar.
values >= 304
300 <= values < 304
248 <= values < 252
values < 248
304
300
296
292
288
284
280
276
272
268
264
260
256
252
248


ncl_default (V6.1.0)
default (V6.0.0 and earlier)
testcmap
OceanLakeLandSnow
GreenYellow
uniform
cmp_haxby
StepSeq25
GrayWhiteGray
Pick a different color table or create your own
http://www.ncl.ucar.edu/Document/Graphics/color_table_gallery.shtml
http://www.ncl.ucar.edu/Document/Graphics/create_color_table.shtml

Line-by-line examples

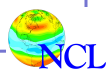
- XY plots
- Use of "gsnMaximize" resource
- Contour plot
- Contour plot over map
- Contouring curvilinear and unstructured data
- WRF plots
- Vector plot
- Panel plots
- Shapefile outlines

Live demo



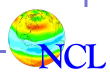
Time for a demo

- Contours over a map
- Follows the *contour2x.ncl* scripts (sort of)



Example *contour2a.ncl*

- gsn_csm_contour_map
- New data file used
- Contour plot overlaid on a cylindrical equidistant map
- Data must have lat/lon coordinates or you must know EXACT map projection ("plotting in a native projection)
- No resources introduced (yet)




```
begin
  f = addfile("uv300.nc","r")
  U = f->U(0,:,:)   ; U will have metadata attached.

  printVarSummary(U)      ; U has coord arrays and attrs
  printVarSummary(U&lon)  ; U&lon has "units" attribute

  wks = gsn_open_wks("png","contour2a")  ;"contour2a.png"

  res = True                ; No plot options set.
  plot = gsn_csm_contour_map(wks,U,res)
end
```

Output from "printVarSummary(U)"

```
Variable: U
Type: float
Total Size: 32768 bytes
            8192 values
Number of Dimensions: 2
Dimensions and sizes:   [lat | 64] x [lon | 128]
Coordinates:
            lat: [-87.8638..87.8638]
            lon: [-180..177.1875]
Number Of Attributes: 5
  time :    1
  _FillValue :    -999
  long_name :     Zonal Wind
  short_name :    U
  units :   m/s
```

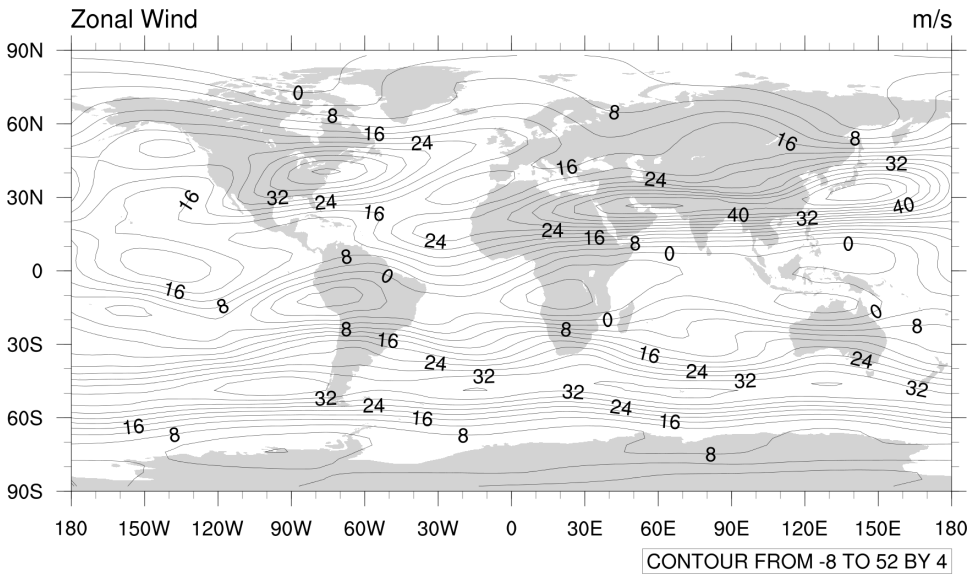
Longitude data is not cyclic. gsn_csm_xxxx scripts will try to add cyclic point by default.

Output from "printVarSummary(U&lon)"

```
Variable: lon (coordinate)
Type: float
Total Size: 512 bytes
            128 values
Number of Dimensions: 1
Dimensions and sizes:    [lon | 128]
Coordinates:
Number Of Attributes: 3
  units : degrees_east
  long_name  : longitude
  short_name : lon
  units      : degrees_east
```

"units" attribute is required for lat/lon arrays!!

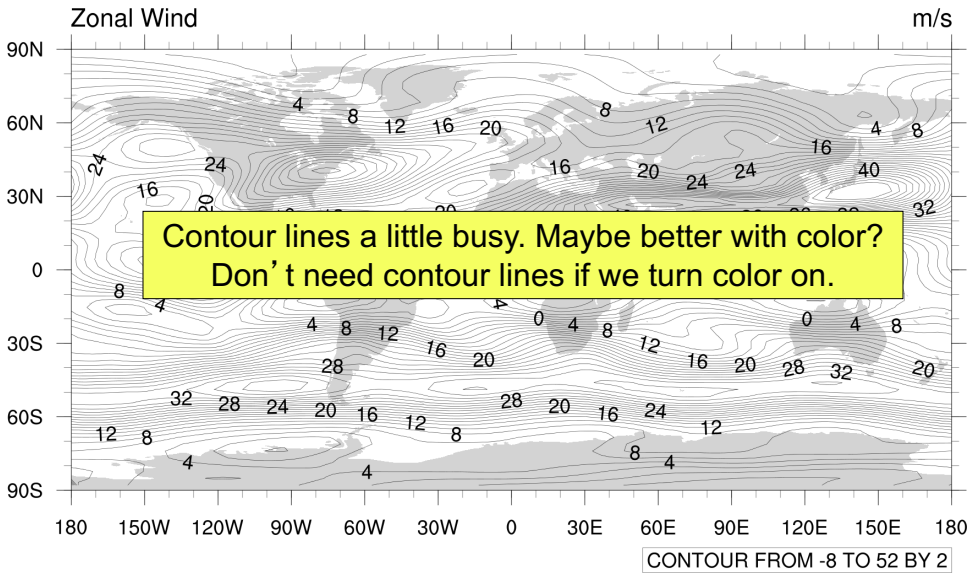
Similarly, U&lat has "degrees_north" attribute



Example *contour2b.ncl*

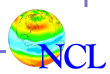
- Contour levels set manually
- Resources introduced:
  - cnLevelSelectionMode - mode for setting contour levels
  - If "cnLevelSelectionMode" is "ManualLevels", then use these 3 resources: cnMinLevelValF, cnMaxLevelValF, cnLevelSpacingF
- Can set "cnLevelSpacingF" by itself.

```
begin
  f = addfile("uv300.nc","r")
  T = f->U(0,:,:)

  wks = gsn_open_wks("png","contour2b")

  res                          = True
  res@cnLevelSelectionMode = "ManualLevels"
  res@cnMinLevelValF       = -8   ; Min contour
  res@cnMaxLevelValF       = 52   ; Max contour
  res@cnLevelSpacingF      =  2   ; Spacing

 ; res@cnLevelSelectionMode = "ExplicitLevels"
 ; res@cnLevels    = (/-8,0,10,15,20,30,45,50/)

  plot = gsn_csm_contour_map(wks,U,res)
end
```

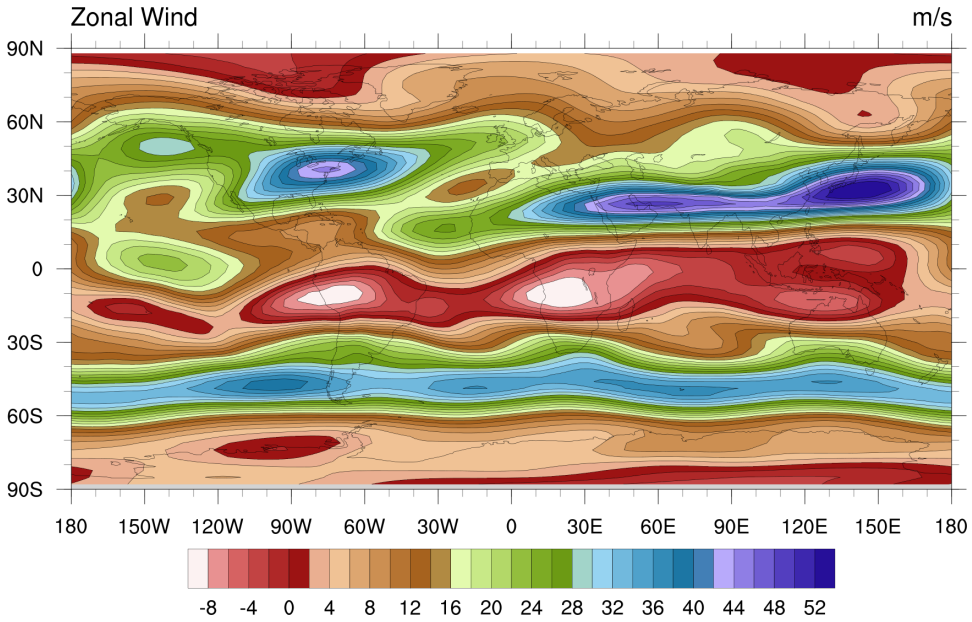


Example *contour2c.ncl*

- Contour fill turned on
- Color map changed

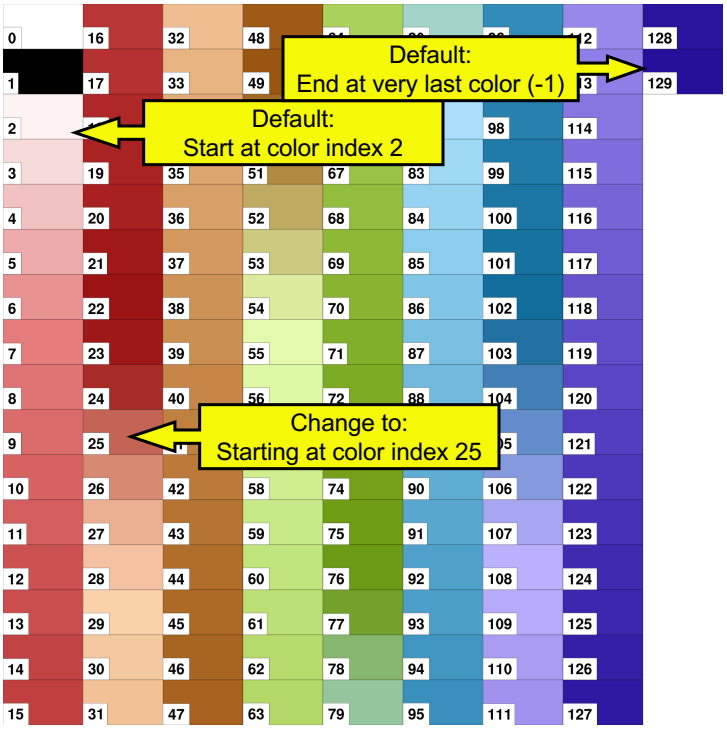



```
begin
  f = addfile("uv300.nc","r")
  T = f->U(0,:,:)

  wks = gsn_open_wks("png","contour2c")

  res                        = True
  res@cnLevelSelectionMode = "ManualLevels"
  res@cnMinLevelValF         = -8   ; Min contour
  res@cnMaxLevelValF         = 52   ; Max contour
  res@cnLevelSpacingF        =  2   ; Spacing

  res@cnFillOn               = True
  res@cnFillPalette          = "MPL_StepSeq"

  plot = gsn_csm_contour_map(wks,U,res)
end
```

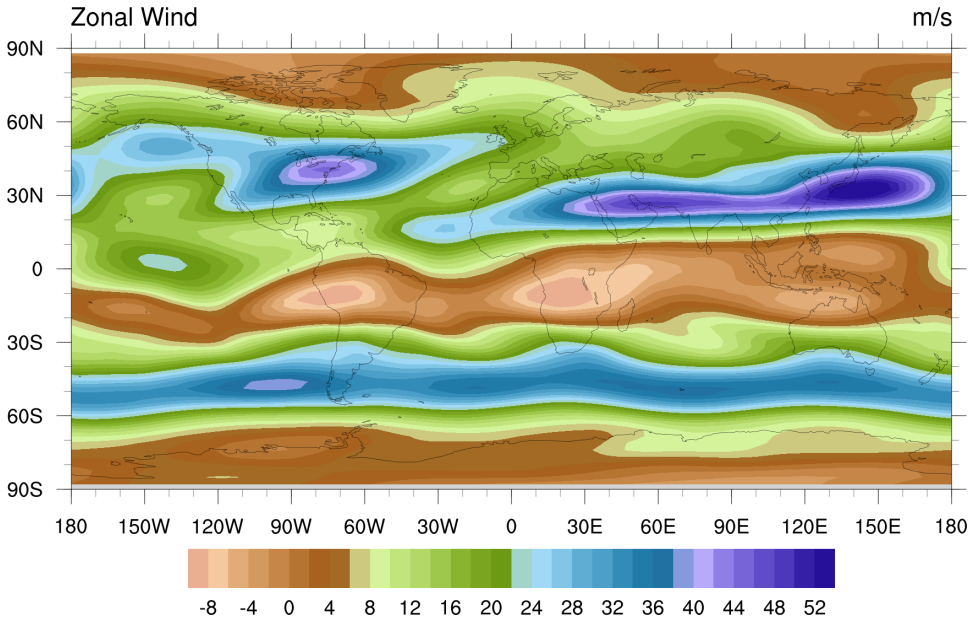


Example *contour2d.ncl*

- Contour and labelbar box lines turned off
- Only part of color map spanned
- Resources introduced:
  - cnLinesOn - turns contour lines on/off
  - lbBoxLinesOn - turns labelbar box lines on/off




```
begin
  f = addfile("uv300.nc","r")
  T = f->U(0,:,:)

  wks = gsn_open_wks("png","contour2d")

  res                      = True
  res@cnLevelSelectionMode = "ManualLevels"
  res@cnMinLevelValF       = -8   ; Min contour
  res@cnMaxLevelValF       = 52   ; Max contour
  res@cnLevelSpacingF      =  2   ; Spacing

  cmap = read_colormap_file("MPL_StepSeq")  ; 130 x 3 RGB array
  res@cnFillPalette         = cmap(25:,:)   ; Start at index 25
  res@cnLinesOn             = False         ; contour lines off
  res@lbBoxLinesOn          = False         ; labelbar box
                                            ; lines off

  plot = gsn_csm_contour_map(wks,U,res)
end
```



Contouring examples

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

Look for "contour" categories:

- Plotting data over a map **Useful**
- Contours: no map
- Contour effects
- Contour labels
- Labelbars

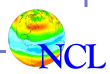
http://www.ncl.ucar.edu/Training/Workshops/Exercises/

Click on:

- Contour plot exercises
- Contours over map exercises (set 1)
- Contours over map exercises (set 2)

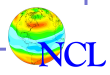
Advanced topics

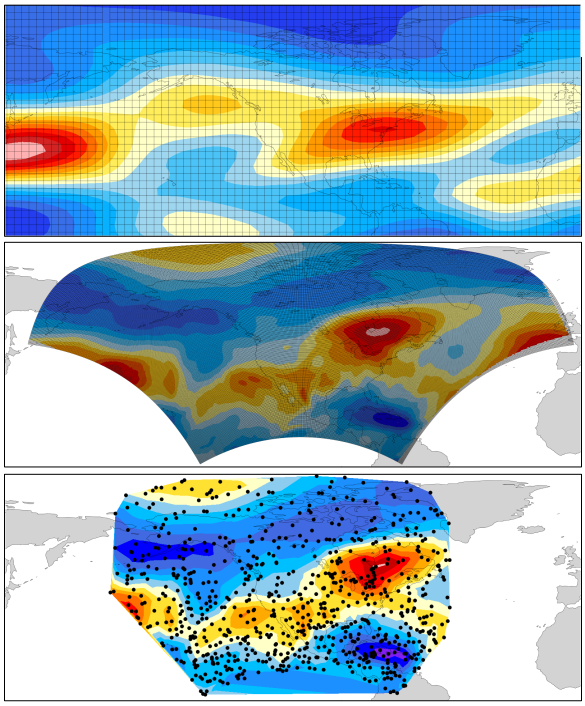
- Changing the labeling style of labelbars: http://www.ncl.ucar.edu/Applications/labelbar.shtml#ex14
- Setting contour levels to get "white in the middle": http://www.ncl.ucar.edu/Applications/color.shtml#ex15
- Controlling individual contours with shading (patterns): http://www.ncl.ucar.edu/Applications/overlay.shtml#ex5
- Controlling individual contour lines with color and/or thickness: http://www.ncl.ucar.edu/Applications/conOncon.shtml#ex7



Line-by-line examples

- XY plots
- Use of "gsnMaximize" resource
- Contour plot
- Contour plot over map
- Contouring curvilinear and unstructured data
- WRF plots
- Vector plot
- Panel plots
- Shapefile outlines





Types of lat/lon arrays

Rectilinear
(1D coordinate arrays)

Curvilinear
(2D lat/lon coordinates)

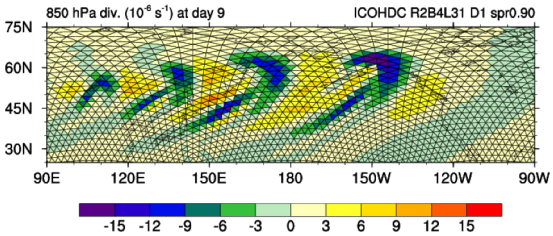
WRF, POP, NARR, are examples of curvilinear data

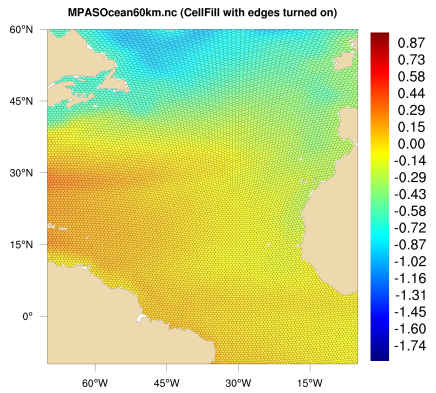
Unstructured

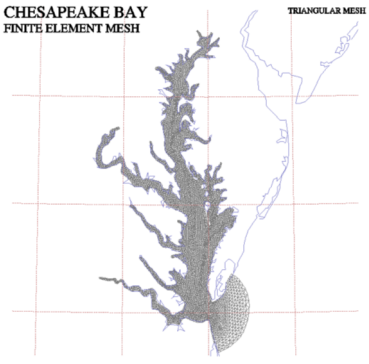
Usually “meshes”

Can be random points

More examples of unstructured grids and meshes







Special topic on contouring data

- Rectilinear: one-dimensional (1D) coordinate arrays
  - gsn_csm_xxxx scripts automatically look for coordinate arrays
- Curvilinear: 2D lon/lat arrays (WRF files, satellite data)
  - Use special lat2d/lon2d attributes (setting sfX/YArray also works)

    data@lat2d = lat res@sfXArray = lon

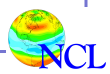
    data@lon2d = lon res@sfYArray = lat
- Unstructured (meshes or 1D X,Y,Z arrays)
  - Use sfXArray, sfYArray resources

    res@sfXArray = lon data@lat1d = lat

    res@sfYArray = lat data@lon1d = lon

Might need:
res@gsnAddCyclic = False

NCL V6.4.0



Curvilinear example: 2D lat/lon arrays

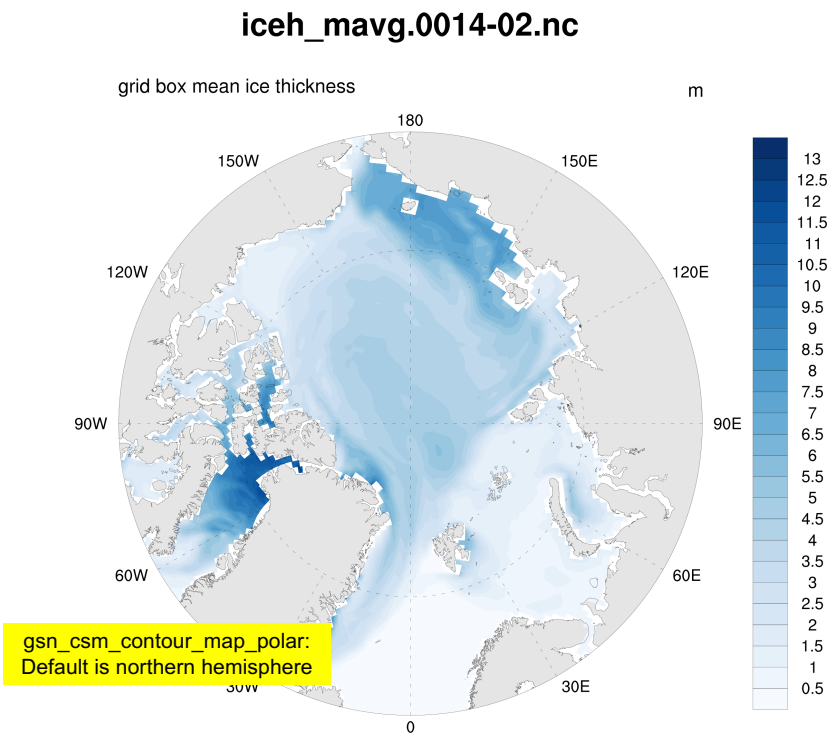
- Assume file is from sea ice model: "iceh_mavg.0014-02.nc"
- Has a variable "hi" w/no coordinate arrays

```
Dimensions and sizes:   [lat | 384] x [lon | 320]
Coordinates:
Number Of Attributes: 7
  time :         4804
  units :        m
  long_name :    grid box mean ice thickness
  coordinates :  i j time
  _FillValue :   1e+30
  time_rep :     averaged
```

- File does have two-dimensional lat/lon variables

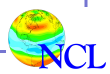
```
float TLON ( lat, lon )
         long_name :    grid center longitude
         units :        degrees_east
float TLAT ( lat, lon )
         long_name :    grid center latitude
         units :        degrees_north
```

```
f = addfile("iceh_mavg.0014-02.nc","r")
hi         = f->hi(0,:,:)            ; Ice coverage
printVarSummary(hi)                  ; No coordinate arrays!

hi@lat2d = f->TLAT     ; Use special "lat2d", "lon2d"
hi@lon2d = f->TLON     ; attributes to plot data correctly.

wks = gsn_open_wks("png","ice")             ; ice.png

res                   = True           ; Plot mods desired
res@gsnAddCyclic      = False          ; Turn off cyclic point
res@cnFillOn          = True           ; Turn on color fill
res@cnLinesOn         = False          ; Turn off contour lines
res@cnFillPalette     = "MPL_Blues"    ; Change color map
res@cnLevelSpacingF   = 0.5            ; NCL chose 1.0
res@lbOrientation     = "Vertical"

res@mpMinLatF         = 65             ; Minimum lat
res@tiMainString      = fname

plot = gsn_csm_contour_map_polar(wks,hi,res)
```



Example: one-dimensional x,y,z data

- File is from ARPEGE grid
- Has a variable "SUTOPRSU" with no coordinate arrays
- Variable has a time dimension
- Separate file contains 1D lat, lon arrays

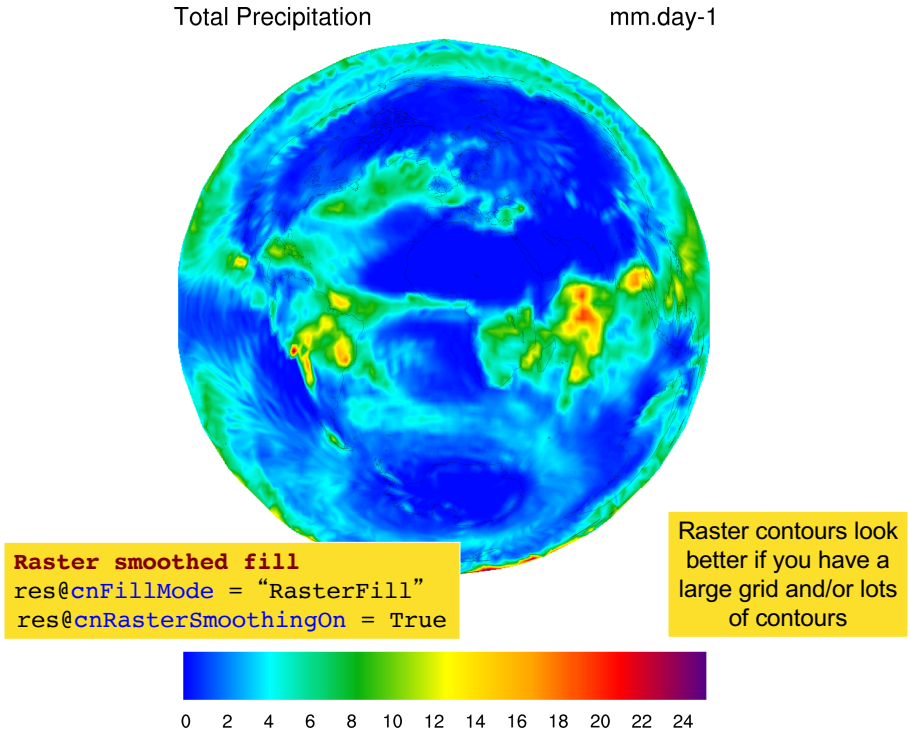


Example: 1D x,y,z data (cont'd)

```
Variable: SUTOPRSU (file variable)
Type: float
Number of Dimensions: 3
Dimensions and sizes:   [time_counter | 240] x [jpif | 6232]
Coordinates:
            time_counter: [   0.. 239]
Number Of Attributes: 4
  short_name :  PRECI TOTALE
  gridtype :    BT42REDU
  long_name :   Total Precipitation
  units :       mm.day-1

Variable: bt42_lat (file variable)
Type: double
Number of Dimensions: 2
Dimensions and sizes:  [jpif | 6232]

Variable: bt42_lon (file variable)
Type: double
Number of Dimensions: 2
Dimensions and sizes:   [jpif| 6232]
```

```
begin
  g = addfile("arpege_grd.nc","r")
  f = addfile("IBF_1m_000101_002012_SUTOPRSU.nc","r")
  data = f->SUTOPRSU(0,:)      ; read first time step
  data@lon1d = g->bt42_lon  ; 1D arrays, 6232 values
  data@lat1d = g->bt42_lat

  wks = gsn_open_wks("x11","arpege_raster_smooth")

  res                       = True

  res@cnFillOn              = True            ; Turn on contour fill
  res@cnFillMode            = "RasterFill"    ; Smooth raster contours
  res@cnRasterSmoothingOn   = True
  res@cnLinesOn             = False           ; Turn off contour lines

  res@cnLevelSelectionMode  = "ExplicitLevels"   ; Set contour levels.
  res@cnLevels              =  fspan(0,25,201)
  res@cnFillPalette         = "BlAqGrYeOrReVi200"

  res@lbBoxLinesOn          = False      ; Turn off box lines

  res@mpProjection          = "LambertEqualArea"  ; Map projection

  contour = gsn_csm_contour_map(wks,data,res)
end
```

"RasterFill" mode can be *much* faster.

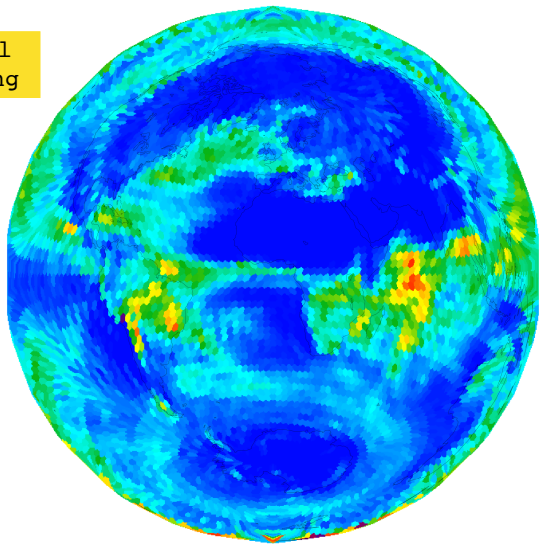


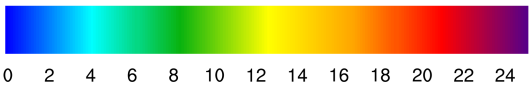
Raster smoothed fill
res@cnFillMode = "RasterFill"
res@cnRasterSmoothingOn = True

Raster contours look better if you have a large grid and/or lots of contours

Total Precipitation mm.day-1

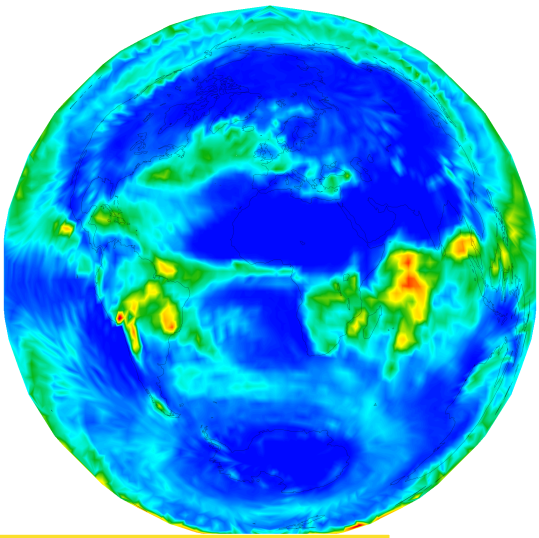
Raster fill
No smoothing





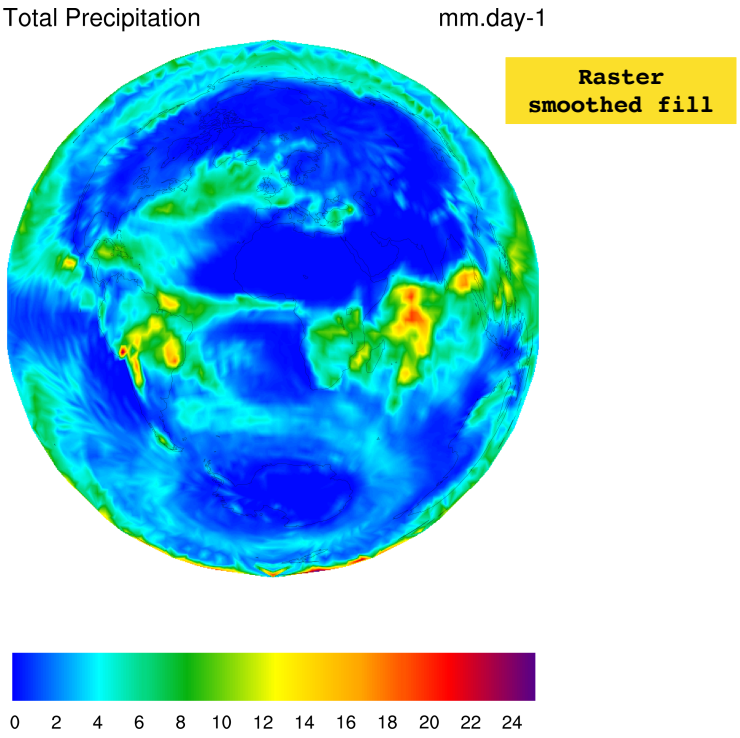
Total Precipitation mm.day-1

Area (smooth) fill



Timings

Area fill: 68.1 seconds
Raster fill: 1.7 seconds

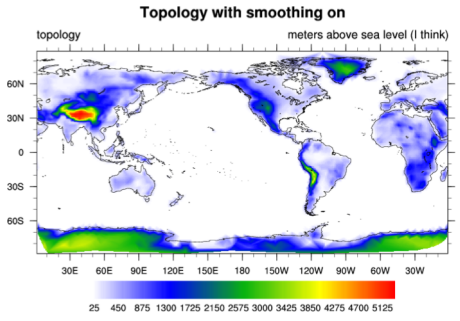


Examples of contouring unstructured data

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

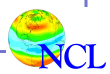
Click on “non-uniform grids/random data

- Adaptive grids
- ORCA grids
- MPAS
- Triangular meshes
- etc.



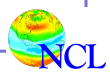
Line-by-line examples

- XY plots
- Use of "gsnMaximize" resource
- Contour plot
- Contour plot over map
- Contouring curvilinear and unstructured data
- WRF plots
- Vector plot
- Panel plots
- Shapefile outlines



Visualizing WRF data

- WRF data is curvilinear (2D lat/lon)
- Use wrf_user_getvar to extract data or calculate basic diagnostics
- Can extract data the usual way, too
- Use wrf_*xxxx* (WRF plot templates) to create "black box" graphics
- Use gsn_csm_*xxxx* if more (or less!) customization is needed



WRF plot templates

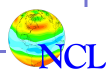
- Developed and maintained in NCAR/MMM

  http://www.ncl.ucar.edu/Applications/wrf.shtml

- Set of scripts for post-processing WRF-ARW data
- Scripts are "black-box"; if you need more/less customization, consider using gsn_csm scripts:

  http://www.ncl.ucar.edu/Applications/wrfgsn.shtml

If you need help with visualizing WRF data, this is a good lab activity to consider!



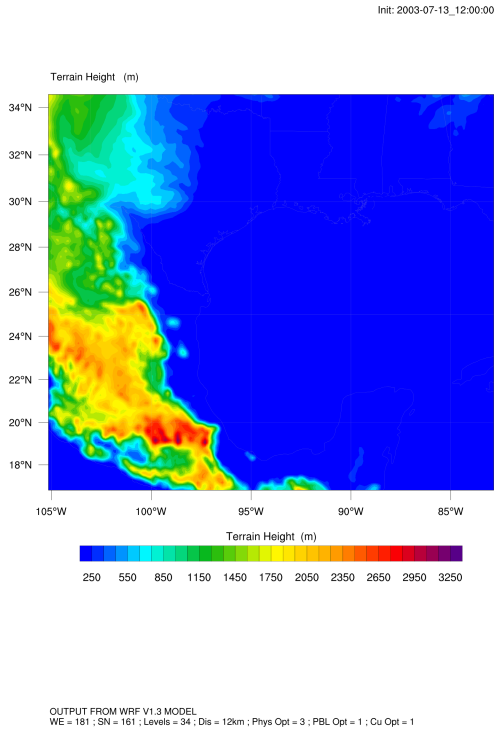
Plotting WRF data using wrf_*xxxx* functions

```
f = addfile ("wrfout_d01_2003-07-15_00:00:00.nc", "r")
hgt = wrf_user_getvar(f,"HGT",0)
; hgt = f->HGT(0,:,:)  ; Can extract the "usual" way

wks = gsn_open_wks("png" ,"wrfgeo_wrf")
gsn_define_colormap(wks,"BlAqGrYeOrReVi200")

res                     = True
res@cnFillOn            = True
res@ContourParameters = (/ 250., 3250., 100. /)

contour = wrf_contour(f,wks,hgt,res)

pltres = True
mpres  = True
plot   = wrf_map_overlays(f,wks,contour,pltres,mpres)
```

ContourParameters is a special wrf_contour resource

wrf_map_overlays looks at file to determine map projection

This image is tailored for an 8 ½ x 11 piece of paper

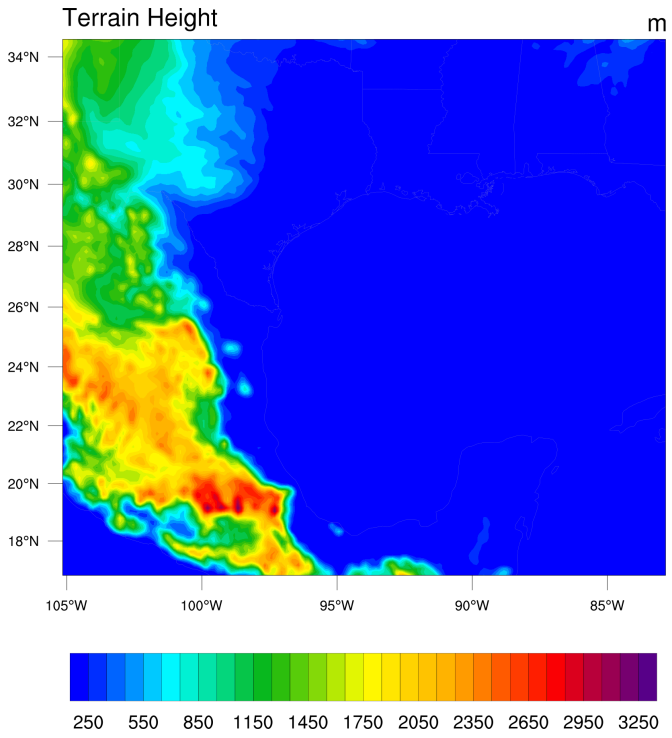


Plotting WRF data using gsn_csm_contour_map

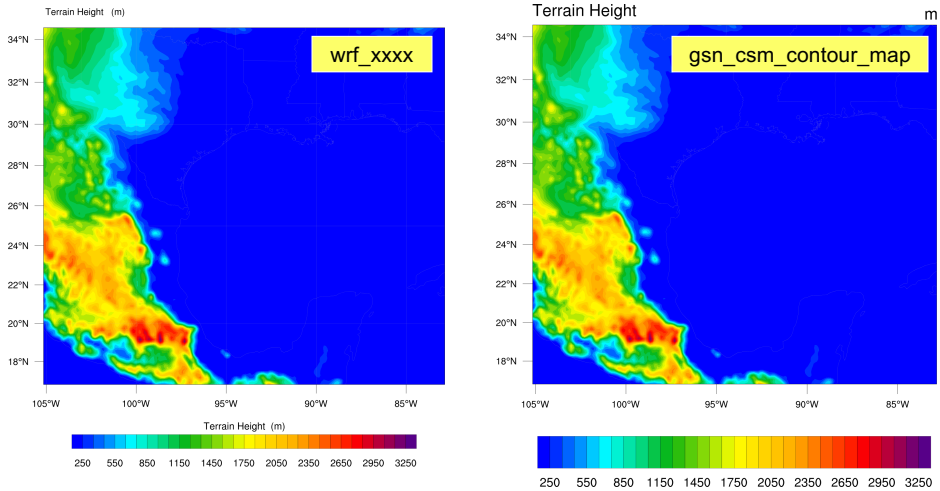
```
f = addfile ("wrfout_d01_2003-07-15_00:00:00.nc", "r")
hgt = wrf_user_getvar(f,"HGT",0)

wks = gsn_open_wks("png" ,"wrfgeo_gsn")

res                         = True
res@cnFillOn                = True
res@cnFillPalette           = "BlAqGrYeOrReVi200"
res@cnLinesOn               = False
res@cnLevelSelectionMode    = "ManualLevels"
res@cnMinLevelValF          =  250.
res@cnMaxLevelValF          = 3250.
res@cnLevelSpacingF         = 100.

res@tfDoNDCOverlay          = True
res@gsnAddCyclic            = False

res = wrf_map_resources(f,res)

plot = gsn_csm_contour_map(wks,hgt,res)
```

res@tfDoNDCOverlay = True tells NCL you know the exact map projection

wrf_map_resources sets resources for WRF map projection



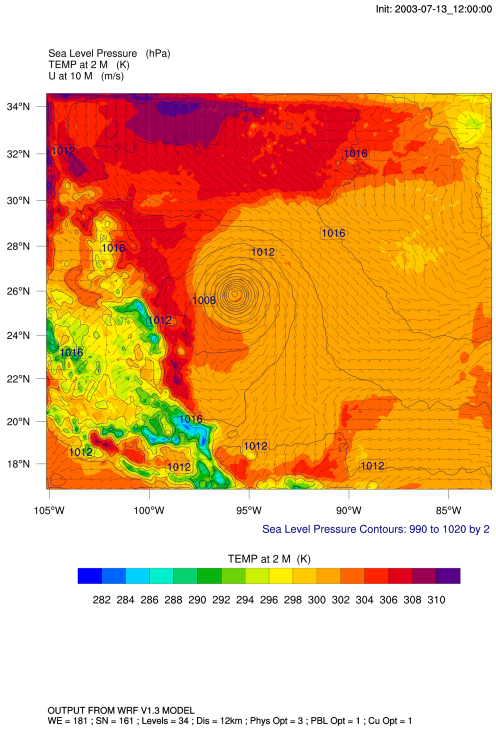
Init: 2003-07-13_12:00:00



OUTPUT FROM WRF V1.3 MODEL
WE = 181 ; SN = 161 ; Levels = 34 ; Dis = 12km ; Phys Opt = 3 ; PBL Opt = 1 ; Cu Opt = 1

WRF-NCL: line/fill contours, vectors

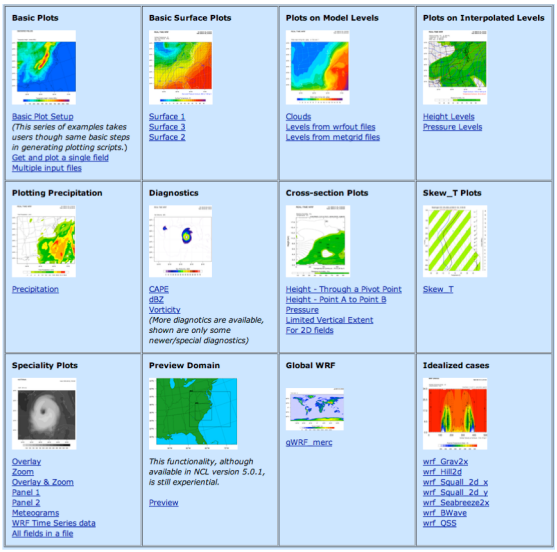
```
f   = addfile ("wrfout_d01_2003-07-15_00:00:00.nc", "r")
slp = wrf_user_getvar(f,"slp",0)
t2  = wrf_user_getvar(f,"T2",0)
u10 = wrf_user_getvar(f,"U10",0)
v10 = wrf_user_getvar(f,"V10",0)

wks = gsn_open_wks("png" ,"wrf_line_fill_vector")
gsn_define_colormap(wks,"BlAqGrYeOrReVi200")

; Line contours
lres                  = True
lres@cnLineColor      = "NavyBlue"
lres@cnLineThicknessF = 2.0
lcontour              = wrf_contour(f,wks,slp,lres)

; Filled contours
fres                  = True
fres@cnFillOn         = True
fres@cnLineThicknessF = 2.0
fcontour              = wrf_contour(f,wks,t2,fres)

; Vectors
vres            = True
vres@NumVectors = 47
vector          = wrf_vector(f,wks,u10,v10,vres)

; Overlay everything on a WRF map
pltres = True
mpres  = True
plot   = wrf_map_overlays(f,wks,(/lcontour,fcontour,vector/),pltres,mpres)
```



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

Scripts and full-sized images available.

For help: email **wrfhelp@ucar.edu**

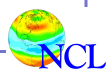


Line-by-line examples

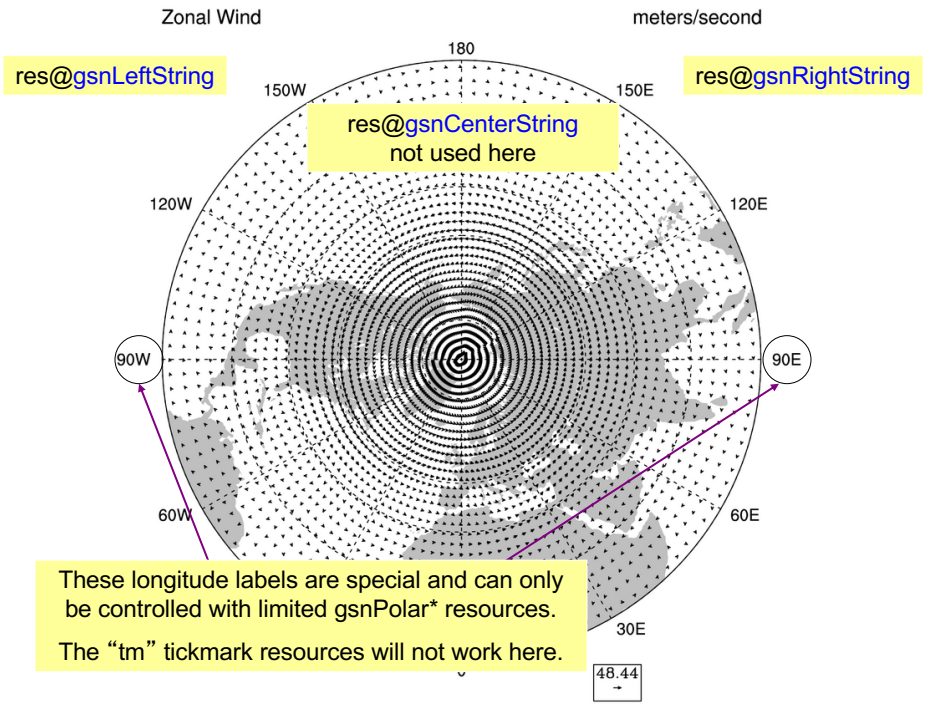
- XY plots
- Use of "gsnMaximize" resource
- Contour plot
- Contour plot over map
- Contouring curvilinear and unstructured data
- WRF plots
- Vector plot
- Panel plots
- Shapefile outlines

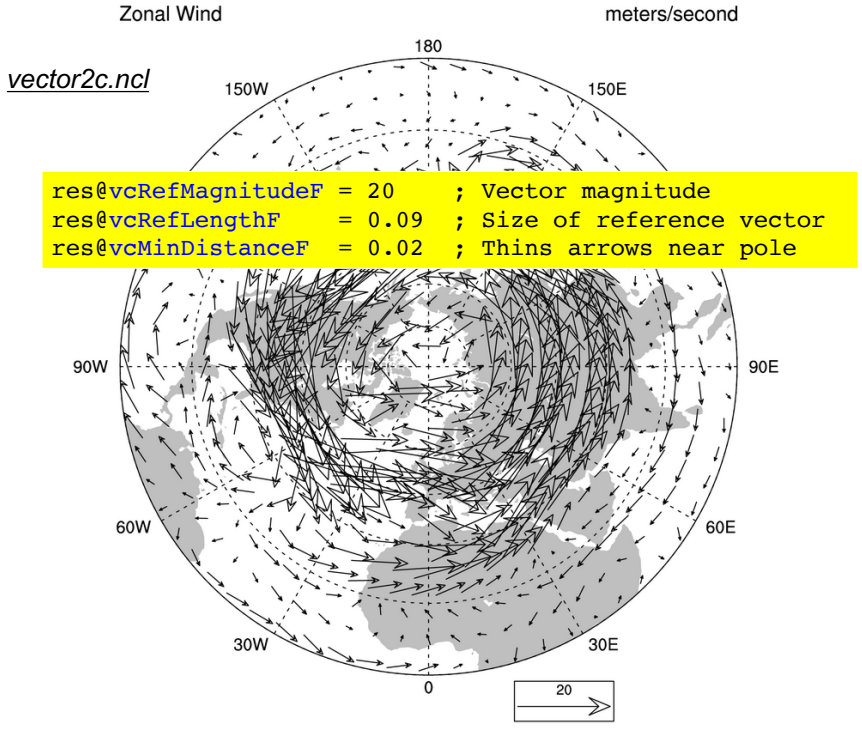
Example *vector2b.ncl*

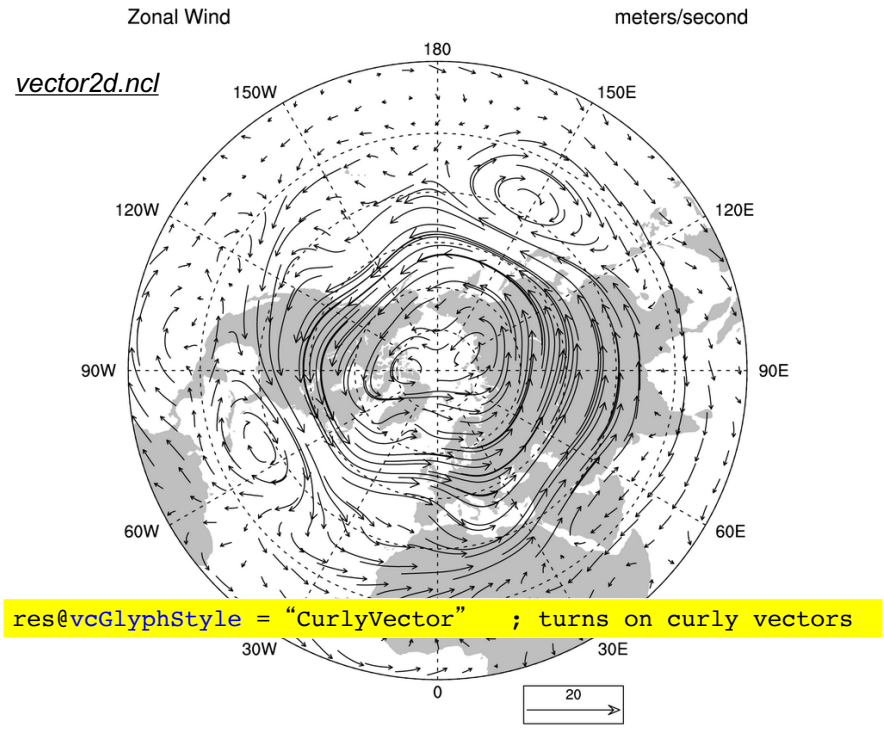
- gsn_csm_vector_map_polar
- Vectors over northern hemisphere polar stereographic map
- Special longitude labels
- Resources introduced:
  - gsnPolar, gsnLeftString, gsnRightString



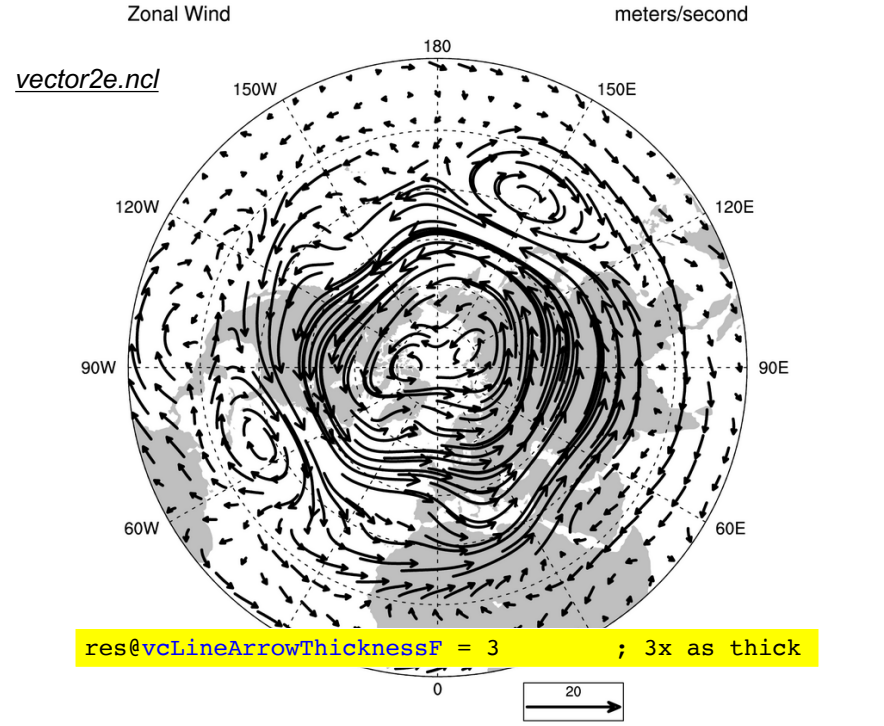

```
begin
  a = addfile("atmos.nc","r")        ; Read in first time step
  u = a->U(0,15,:,:)                 ; and 15th level of
  v = a->V(0,15,:,:)                 ; atmospheric data.

  wks = gsn_open_wks("png","vector2b")

  res                 = True

  res@gsnPolar        = "NH"             ; Northern hemisphere

  res@gsnLeftString   = "Zonal Wind"     ; Left subtitle
  res@gsnRightString  = "meters/second"  ; Right subtitle
; res@gsnCenterString = "something"      ; Not used here

  plot = gsn_csm_vector_map_polar(wks,u,v,res)
end
```


Zonal Wind
meters/second
res@gsnLeftString
res@gsnRightString
res@gsnCenterString
not used here
180
150W
150E
120W
120E
90W
90E
60W
60E
30E
48.44
These longitude labels are special and can only be controlled with limited gsnPolar* resources.
The “tm” tickmark resources will not work here.


Zonal Wind
meters/second
vector2c.ncl
res@vcRefMagnitudeF = 20 ; Vector magnitude
res@vcRefLengthF = 0.09 ; Size of reference vector
res@vcMinDistanceF = 0.02 ; Thins arrows near pole
180
150W
150E
90W
90E
60W
60E
30W
30E
0
20

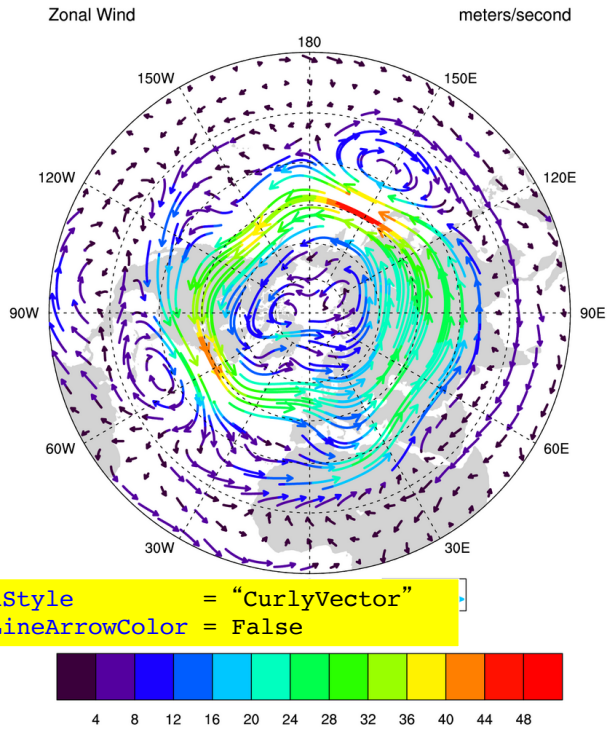



```
res@vcGlyphStyle = "CurlyVector"    ; turns on curly vectors
```




```
res@vcLineArrowThicknessF = 3          ; 3x as thick
```

vector2f.ncl
vector2g.ncl



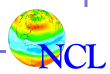

```
res@vcGlyphStyle          = "CurlyVector"
res@vcMonoLineArrowColor  = False
```

Vector examples

http://www.ncl.ucar.edu/Applications/vector.shtml
http://www.ncl.ucar.edu/Applications/veceff.shtml

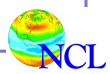
Line-by-line examples

- XY plots
- Use of "gsnMaximize" resource
- Contour plot
- Contour plot over map
- Contouring curvilinear and unstructured data
- WRF plots
- Vector plot
- Panel plots
- Shapefile outlines

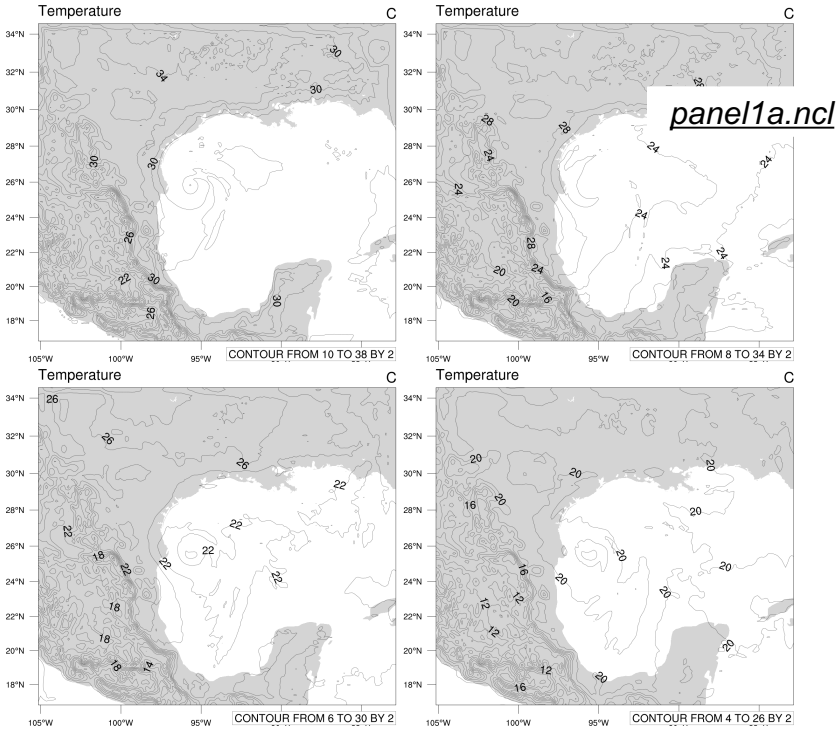


Example *panel1*.ncl*

- gsn_panel
- Special topic: paneling plots (multiple plots on a page)
- Plots must be the same size!
- Can have common labelbar, title, figure captions
- Resources mentioned again:
  - gsnDraw, gsnFrame
- If plots different sizes, use vpXF, vpYF, vpWidthF, vpHeightF resources instead

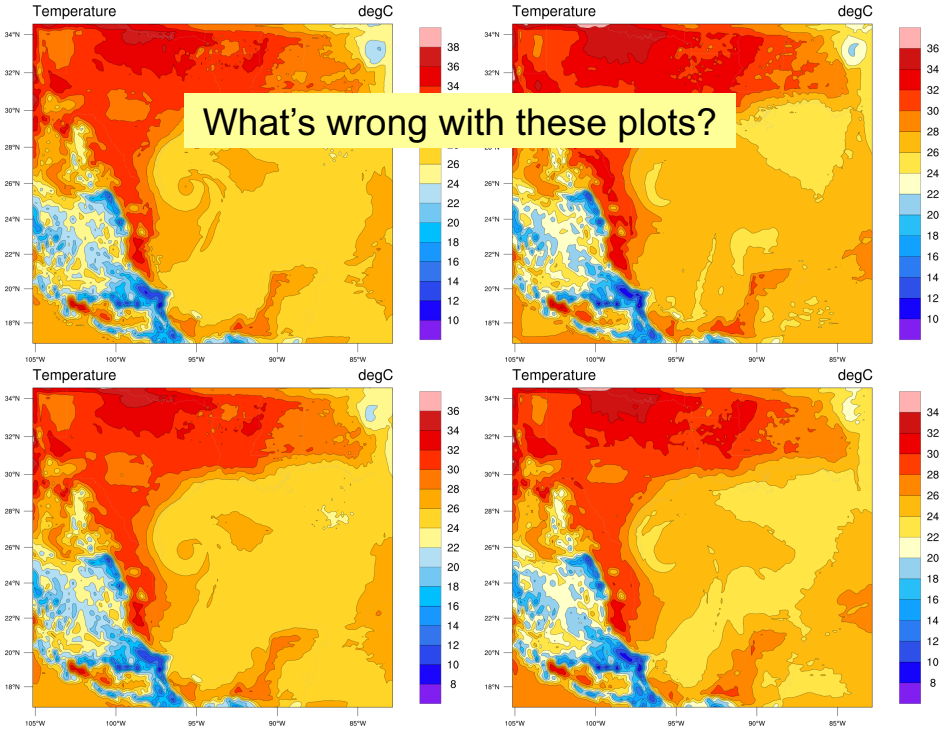


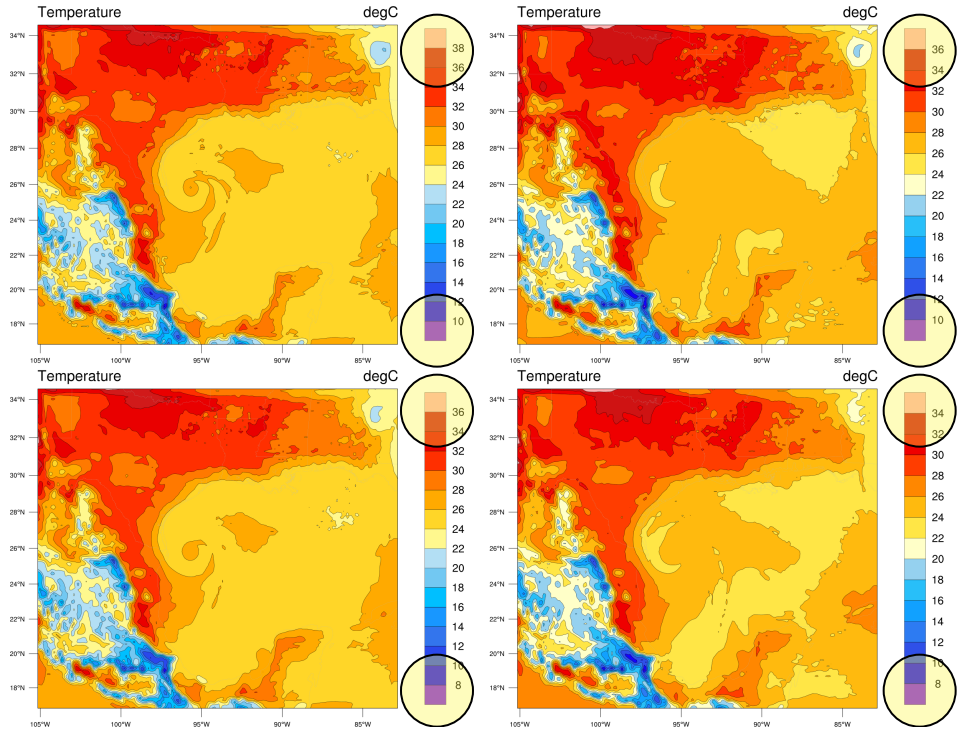
```
f = addfile ("wrfout_d01_2003-07-15_00:00:00.nc", "r")
tc = wrf_user_getvar(f,"tc",0)    ; temperature in degC

wks = gsn_open_wks("x11", "panel1a")

res                  = True
res@gsnDraw          = False      ; Don't draw plots
res@gsnFrame         = False      ; Don't advance frame

res@tfDoNDCOverlay = True         ; Use native WRF map projection
res@gsnAddCyclic   = False        ; Don't add longitude cyclic pt
res = wrf_map_resources(f,res)    ; Add resources for WRF map

plot1 = gsn_csm_contour_map(wks,tc(0,:,:),res) ; Create contour
plot2 = gsn_csm_contour_map(wks,tc(3,:,:),res) ; plots at
plot3 = gsn_csm_contour_map(wks,tc(5,:,:),res) ; different
plot4 = gsn_csm_contour_map(wks,tc(7,:,:),res) ; levels

;---Create a panel of plots with 2 rows and 2 columns.
gsn_panel(wks,(/plot1,plot2,plot3,plot4/),(/2,2/),False)
```



panel1a.ncl

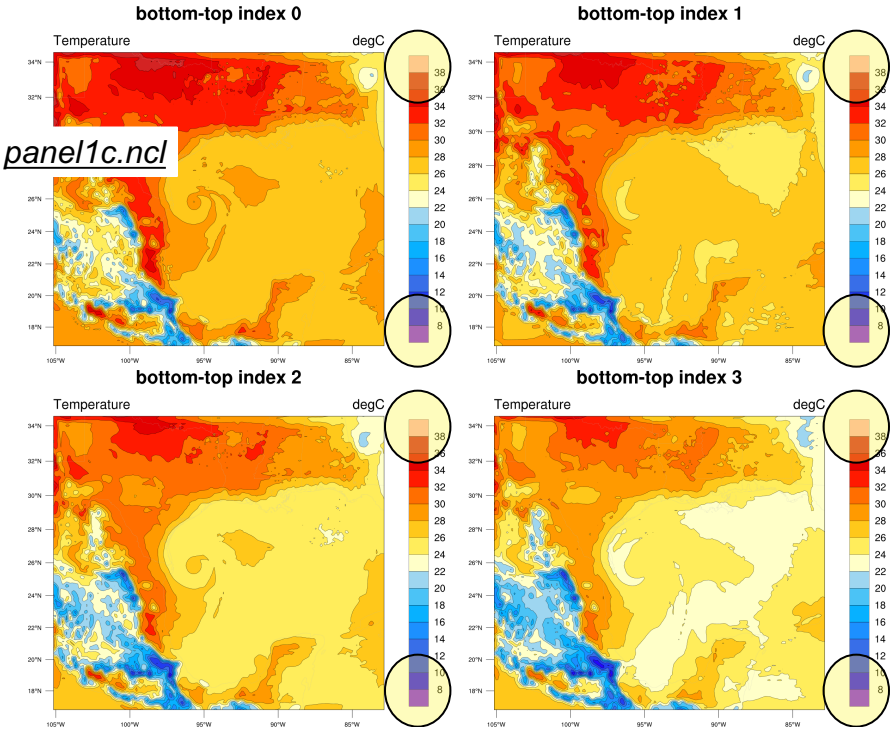
```
f = addfile ("wrfout_d01_2003-07-15_00:00:00.nc", "r")
tc = wrf_user_getvar(f,"tc",0)    ; temperature in degC
tc@units = "degC"                      ; better units

wks = gsn_open_wks("x11", "panel1b")

res                  = True
res@gsnDraw          = False      ; Don't draw plots
res@gsnFrame         = False      ; Don't advance frame

res@tfDoNDCOverlay = True         ; Use native WRF map projection
res@gsnAddCyclic   = False        ; Don't add longitude cyclic pt

res = wrf_map_resources(f,res)  ; Add resources for WRF map

res@cnFillOn         = True            ; Turn on color
res@lbOrientation    = "Vertical"      ; Vertical labelbar

;---Loop across four levels and create a contour plot.
plots = new(4,graphic)
do i=0,3
  plots(i) = gsn_csm_contour_map(wks,tc(i,:,:),res)
end do

;---Create a panel of plots with 2 rows and 2 columns.
gsn_panel(wks,plots,(/2,2/),False)
```



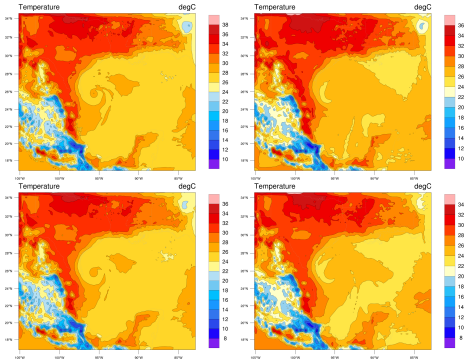


Set contour levels to same for all plots

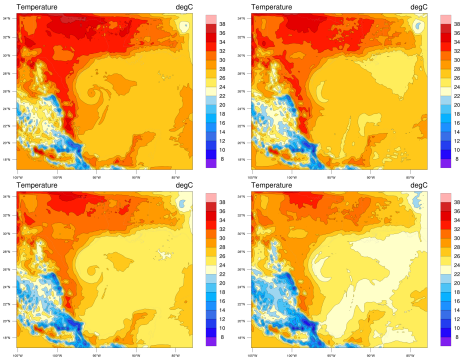
```
. . .
res                   = True           ; Plot options desired
. . .
res@cnLevelSelectionMode = "ManualLevels"
res@cnMinLevelValF       =  8     ; Set contour levels to
res@cnMaxLevelValF       = 38     ; the same for each plot.
res@cnLevelSpacingF      =  2

;---Loop across four levels and create a contour plot.
plots = new(4,graphic)
do i=0,3
  plots(i) = gsn_csm_contour_map(wks,tc(i,:,:),res)
end do

  . .
  gsn_panel(wks,plots,(/2,2/),False)
```



panel1c.ncl

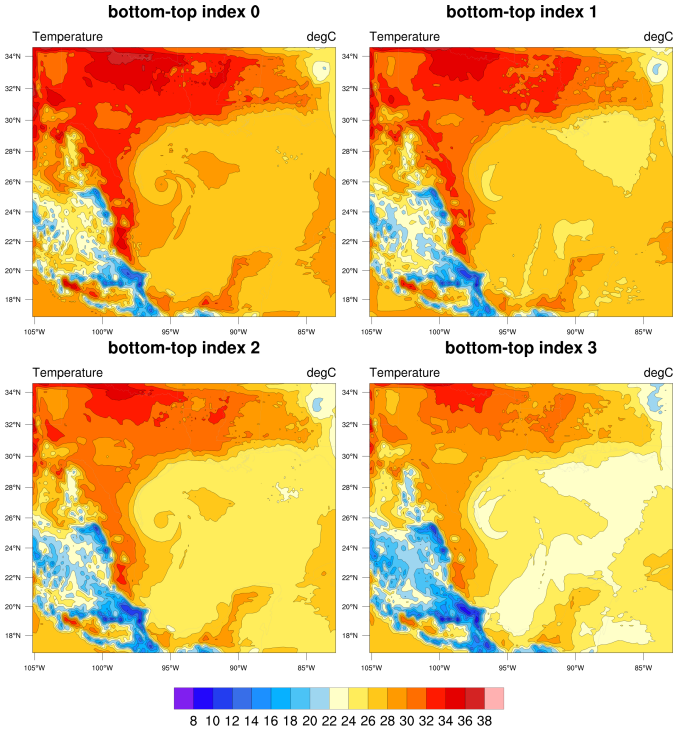


Before fixing contour levels

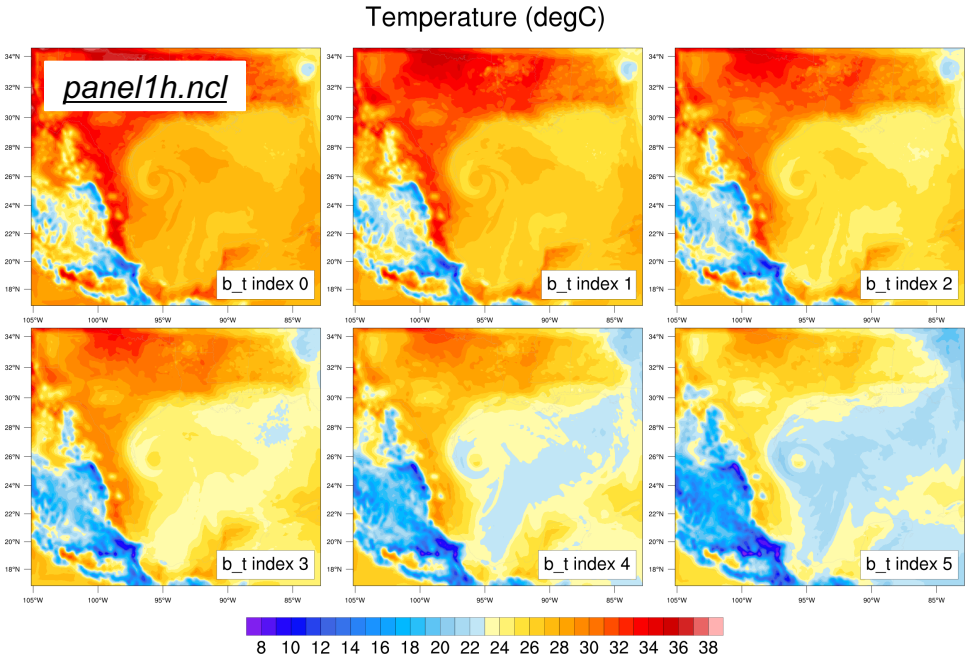


After fixing contour levels

```
. . .
  res                       = True           ; Plot options desired
  res@gsnDraw               = False          ; Don't draw plots
  res@gsnFrame              = False          ; Don't advance frames
  res@cnFillOn              = True           ; Turn on color
  res@cnLevelSelectionMode  = "ManualLevels"
  res@cnMinLevelValF        =   8            ; Set contour levels to
  res@cnMaxLevelValF        =  38            ; the same for each plot.
  res@cnLevelSpacingF       =   2
  res@lbOrientation         = "Vertical"     ; Vertical labelbar
  res@lbLabelBarOn          = False ; Turn off individual labelbar

  plots = new(4,graphic)
  do i=0,3
    res@tiMainString = "bottom-top index " + i
    plots(i)         = gsn_csm_contour_map(wks,tc(i,:,:),res)
  end do
; Create a panel of plots with 2 rows and 2 columns.

  pres                  = True          ; Set panel resources.
  pres@gsnMaximize      = True          ; Maximize plots in panel.
  pres@gsnPanelLabelBar = True          ; Turn on panel labelbar.

 gsn_panel(wks,plots,(/2,2/),pres)
end
```



panel1d.ncl
common
labelbar



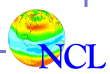
2 rows x 3 columns Longer labelbar Figure strings added

Problems with paneling?

- Are plots the same size? If not, maybe set res@gsnPanelScalePlotIndex
- Set res@gsnPanelDebug = True
  - Prints debug information about size and location of paneled plots
- Set res@gsnPanelBoxes = True
  - Draws bounding boxes around each plot element so you can see true size
- Set res@gsnPanelXF and/or res@gsnPanelYF
  - Can use these to force plots to line up
- Use vpXF/vpYF/vpWidthF/vpHeightF instead of gsn_panel

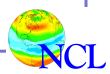
Panel examples

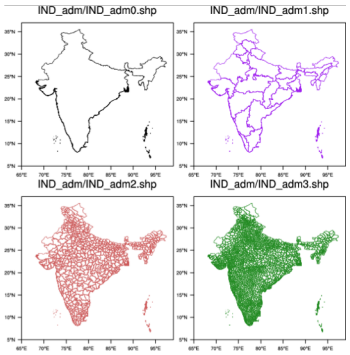
http://www.ncl.ucar.edu/Applications/panel.shtml

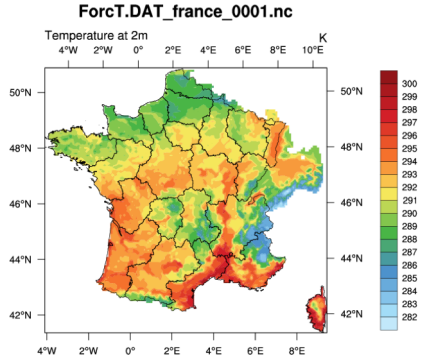


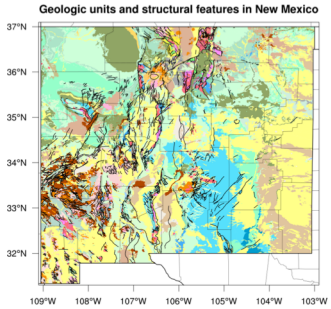
Line-by-line examples

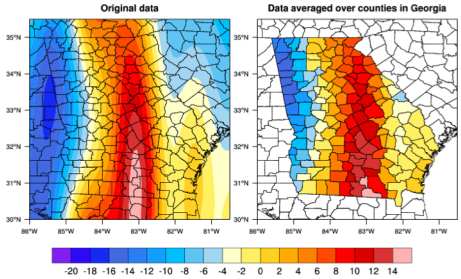
- XY plots
- Use of "gsnMaximize" resource
- Contour plot
- Contour plot over map
- Contouring curvilinear and unstructured data
- WRF plots
- Vector plot
- Primitives
- Panel plots
- Shapefile outlines
- Overlays



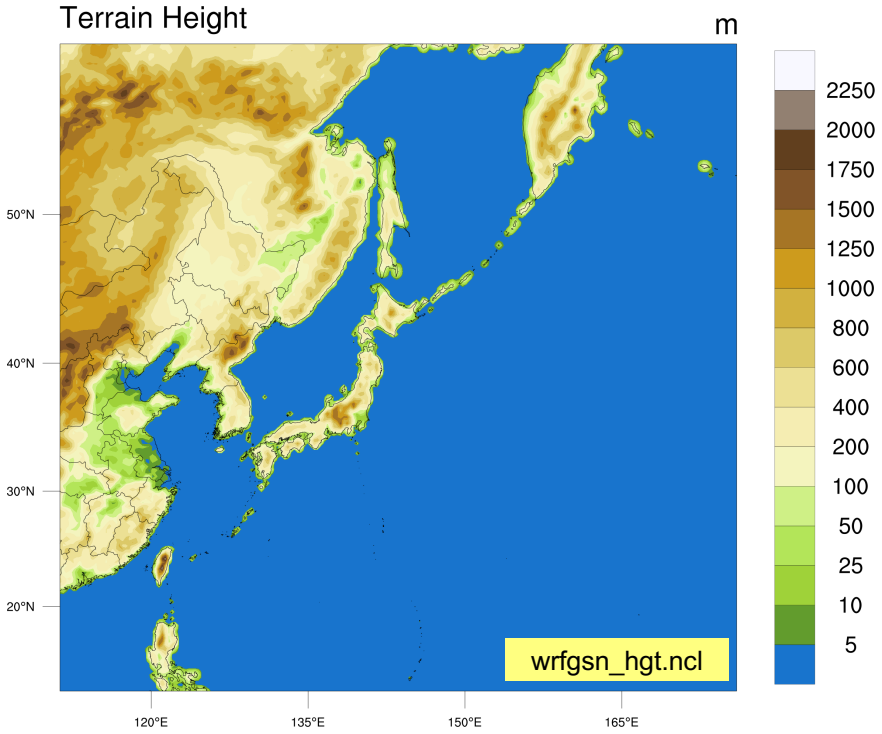




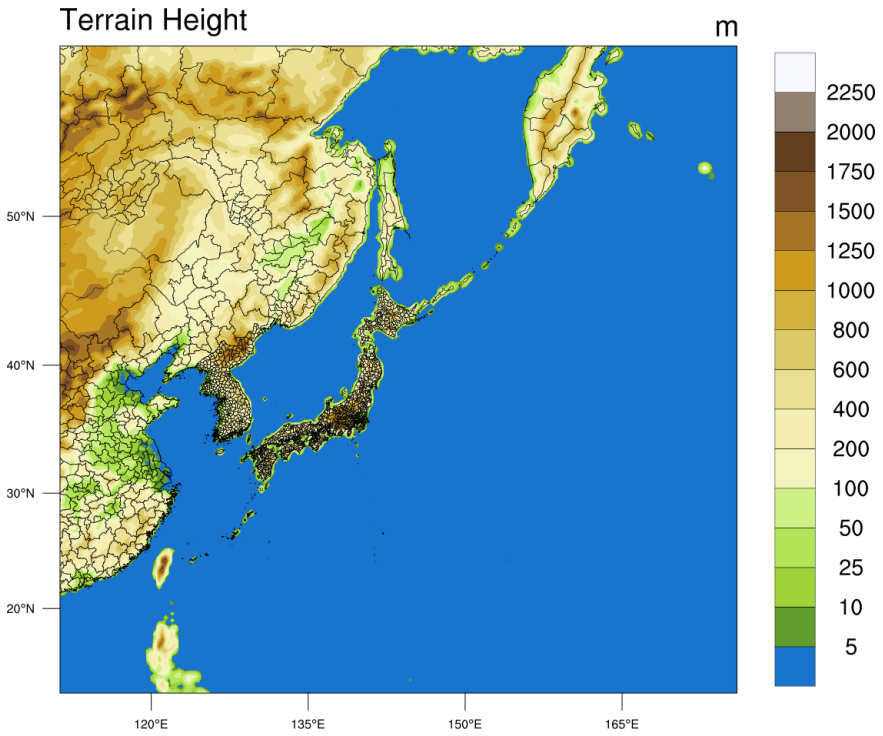





```
a    = addfile("wrfout_d01_2008-09-29_00:00:00","r")
hgt = wrf_user_getvar(a,"HGT",0)    ; Read height off WRF file

wks = gsn_open_wks("x11","wrfgsn_hgt")

res                    = True
res@gsnMaximize        = True

res@cnFillOn           = True         ; Turn on contour fill
res@cnLinesOn          = False        ; Turn off contour lines
res@cnFillPalette      = "OceanLakeLandSnow"
res@lbOrientation      = "Vertical"   ; Default is horizontal

res@cnLevelSelectionMode = "ExplicitLevels"
res@cnLevels             = (/5,10,25,50,100,200,400,600,800,1000, \
                             1250,1500,1750,2000,2250/)

res = wrf_map_resources(a,res)      ; Set map resources based on WRF file
res@tfDoNDCOverlay     = True       ; If using WRF native projection
res@gsnAddCyclic       = False      ; Don't add longitude cyclic point

res@mpGeophysicalLineColor      = "Black"  ; WRF default is gray
res@mpGeophysicalLineThicknessF = 2.0

res@cnLevelSelectionMode = "ExplicitLevels"   ; Change contour levels
res@cnLevels             = (/5,10,25,50,100,200,400,600,800,\
                             1000,1250,1500,1750,2000,2250/)

plot = gsn_csm_contour_map(wks,hgt,res)
```

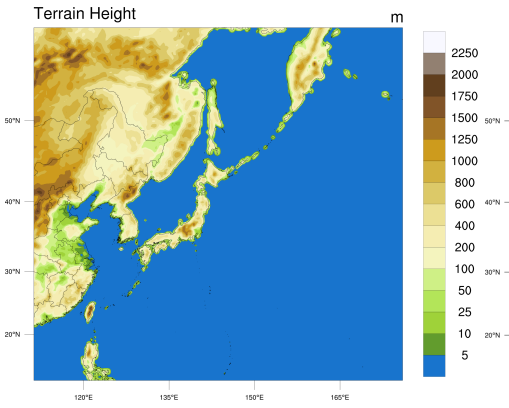



```
. . .
  res@mpOutlineOn       = False   ; Turn off NCL's map outlines and
  res@mpFillOn          = False   ; map fill.

  res@gsnDraw           = False   ; Don't draw plot or advance frame
  res@gsnFrame          = False   ; when gsn_csm_contour_map is called

  plot = gsn_csm_contour_map(wks,hgt,res)

;---Attach outlines from four country shapefiles from gadm.org
  lnres                  = True
  lnres@gsLineThicknessF = 2.0  ; Twice as thick

  jpn_id = gsn_add_shapefile_polylines(wks,plot,"JPN_adm1.shp",lnres)
  chn_id = gsn_add_shapefile_polylines(wks,plot,"CHN_adm2.shp",lnres)
  kor_id = gsn_add_shapefile_polylines(wks,plot,"KOR_adm2.shp",lnres)
  rus_id = gsn_add_shapefile_polylines(wks,plot,"RUS_adm2.shp",lnres)
  prk_id = gsn_add_shapefile_polylines(wks,plot,"PRK_adm2.shp",lnres)

  draw(plot)   ; Drawing plot will also draw attached shapefile outlines
  frame(wks)
```

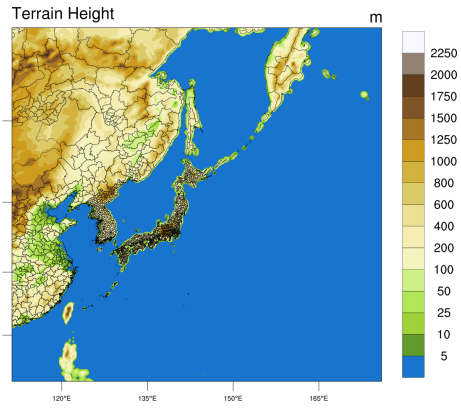


Compare

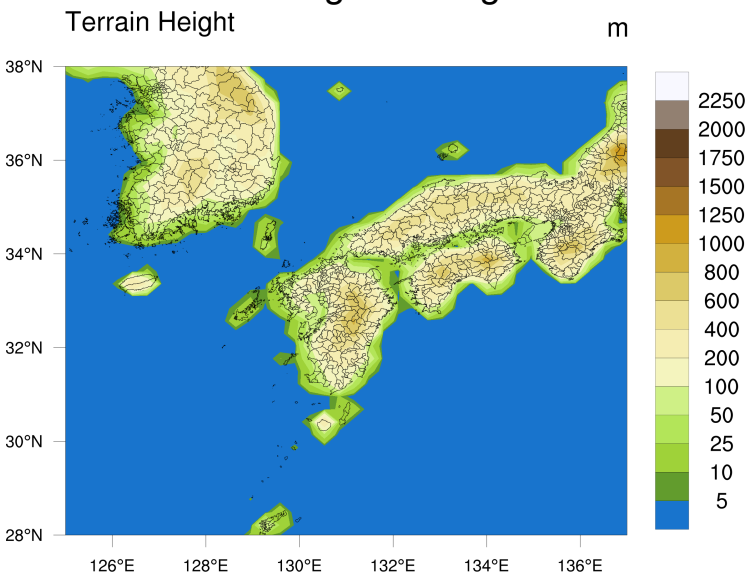
NCL outlines



Shapefile outlines



Zooming in on region



Using higher-res shapefiles: JPN_adm2 and KOR_adm2

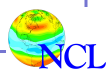
wrfgsn_hgt_shapefiles_zoom.ncl

Shapefile examples

http://www.ncl.ucar.edu/Applications/shapefiles.shtml

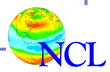
NCL Graphics topics

- Types of graphics you can create with NCL
- The basics
- Line-by-line examples
- Review
- Customizing NCL environment
- Common mistakes and problems
- Debugging tips
- Creating images for Web and/or presentations
- PyNGL/PyNIO Python modules



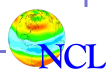
In review…

- Four main steps to create a plot
- Use X11 window while debugging script; move to PNG / PS / PDF later
- Hardest part are the resources: start simple
- Organize resources for easier debugging
- Start with an existing script if possible



NCL Graphics topics

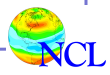
- Types of graphics you can create with NCL
- The basics
- Line-by-line examples
- Review
- Customizing NCL environment
- Common mistakes and problems
- Debugging tips
- Creating images for Web and/or presentations
- PyNGL/PyNIO Python modules



Customize your UNIX editor

- Users have contributed many nice enhanced UNIX editor features specifically for NCL scripts.
- Enhancements available for emacs, nedit, vim, TextMate, Aquamacs, Notepad++, NetBeans, to name a few.

  http://www.ncl.ucar.edu/Applications/editor.shtml
- Editor enhancements will highlight functions, graphical resource, comments, syntax, and other features in different colors.
- Makes debugging a little easier!



Sample editor enhancement for vi / vim

```
begin
;---Get data
 f     = addfile("atmos.nc","r")
 u     = f->V(0,0,:,:)

;---Open PNG file
 wks  = gsn_open_wks ("png", "coneff" )

;---Set plot options
 res                      = True             ; plot mods desired
 res@cnFillOn             = True             ; fill contour intervals
 res@lbLabelAutoStride    = True             ; nice label bar stride

 res@cnLevelSelectionMode = "ManualLevels"   ; manually specify contour levels
 res@cnMinLevelValF       = -35.             ; min level
 res@cnMaxLevelValF       =  35.             ; max level
 res@cnLevelSpacingF      =  10              ; contour interval

 res@cnMonoFillPattern    = False            ; want multiple patterns
 res@cnFillPatterns       = (/3,3,3,3,-1,17,17,17,17/) ; the patterns
 res@cnMonoFillScale      = False            ; want different densities
 res@cnFillScales         = (/.1,.2,.3,.4,1,1,1.5,2.0,2.5/) ; the densities

 res@tiMainString         = "Default dot size"
 plot = gsn_csm_contour(wks, u, res )
```

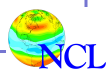
Sample editor enhancement for emacs

```
begin
;---Get data
 f     = addfile("atmos.nc","r")
 u     = f->V(0,0,:,:)

;---Open PNG file
 wks  = gsn_open_wks ("png", "coneff" )

;---Set plot options
 res                      = True             ; plot mods desired
 res@cnFillOn             = True             ; fill contour intervals
 res@lbLabelAutoStride    = True             ; nice label bar stride

 res@cnLevelSelectionMode = "ManualLevels"   ; manually specify contour levels
 res@cnMinLevelValF       = -35.             ; min level
 res@cnMaxLevelValF       =  35.             ; max level
 res@cnLevelSpacingF      =  10              ; contour interval

 res@cnMonoFillPattern    = False            ; want multiple patterns
 res@cnFillPatterns       = (/3,3,3,3,-1,17,17,17,17/) ; the patterns
 res@cnMonoFillScale      = False            ; want different densities
 res@cnFillScales         = (/.1,.2,.3,.4,1,1,1.5,2.0,2.5/) ; the densities

 res@tiMainString         = "Default dot size"
 plot = gsn_csm_contour(wks, u, res )
```

Customize your NCL graphics environment

- Optional, was highly recommended for older versions of NCL
- Download ".hluresfile" file, put in home directory
- Can be used to change default color map, font, function code, etc.
- **Can be used to change default size of X11 window that pops up, or default size of PNG image**

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

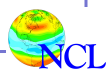


Sample ".hluresfile"

```
! Color map
*wkColorMap       : BlAqGrYeOrReVi200

! Make default X11 window larger
! (default is 512 x 512)
*windowWorkstationClass*wkWidth  : 1000
*windowWorkstationClass*wkHeight : 1000

! Make default PNG window larger
! (default is 1024 x 1024)
*imageWorkstationClass*wkWidth   : 2500
*imageWorkstationClass*wkHeight  : 2500

! Increase all line thicknesses, default is 1.
*LineThicknessF  : 8.5

! Default font is helvetica
*Font            : times-roman
```

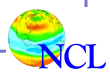
NCL Graphics topics

- Types of graphics you can create with NCL
- The basics
- Line-by-line examples
- Review
- Customizing NCL environment
- Common mistakes and problems
- Debugging tips
- Creating images for Web and/or presentations
- PyNGL/PyNIO Python modules



Common mistakes or problems

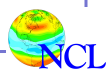
- "xyLineColour is not a resource in XyPlot at this time"
  - Misspelling a resource, "xyLineColour"
  - Using the wrong resource with the wrong plot (i.e. using "vcRefMagnitudeF" in a contour plot).
- "The units attribute of the Y coordinate array is not set to one of the allowable units values (i.e. 'degrees_north'). Your latitude labels may not be correct."
  - Lack of (or wrong) "units" attribute attached to your data's coordinate arrays



More common mistakes or problems

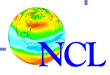
- Data values in plot look off-scale
  - Maybe "_FillValue" attribute not set or not correct.
- "_NhlCreateSplineCoordApprox: Attempt to create spline approximation for Y axis failed: consider adjusting trYTensionF value"
  - Data is too irregularly spaced in the X or Y direction. May need to subset it.

TIP http://www.ncl.ucar.edu/Document/Language/error_messages.shtml
http://www.ncl.ucar.edu/FAQ/#err_msgs



NCL Graphics topics

- Types of graphics you can create with NCL
- The basics
- Line-by-line examples
- Review
- Customizing NCL environment
- Common mistakes and problems
- Debugging tips
- Creating images for Web and/or presentations
- PyNGL/PyNIO Python modules



Debugging tips

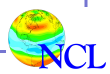
- Use interactive mode to try "quick" things that you don't understand in NCL.
- Otherwise, use "batch" script mode. That is, write a script and then run it:

```
ncl myscript.ncl
```

- Start with an existing script, if possible. You can use templates provided at:

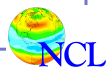
  http://www.ncl.ucar.edu/Applications/Templates/

- Start small, don't set a bunch of resources all at once
- Group resources by type



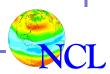
More debugging tips

- If graphics look wrong, comment out a bunch of resources and add them back slowly to see where problem is
- Use "printVarSummary" to examine variables
  - Missing coordinate arrays
  - No "_FillValue" or wrong "_FillValue"
- To further examine data, use:
  - printMinMax(x,0) ; Minimum/maximum of data
  - print(num(ismissing(x))) ; Count number of missing values
- Read errors and warnings carefully
- Use an enhanced UNIX editor! ☺



NCL Graphics topics

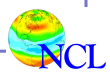
- Types of graphics you can create with NCL
- The basics
- Line-by-line examples
- Review
- Customizing NCL environment
- Common mistakes and problems
- Debugging tips
- Creating images for Web and/or presentations
- PyNGL/PyNIO Python modules



Creating images for web or PowerPoint

- Use direct “png” output and increase size of image through two resources:

```
wtype           = “png”
wtype@wkWidth   = 2500   ; Default is 1024
wtype@wkHeight  = 2500
wks = gsn_open_wks(wtype,”example”)
. . .
```

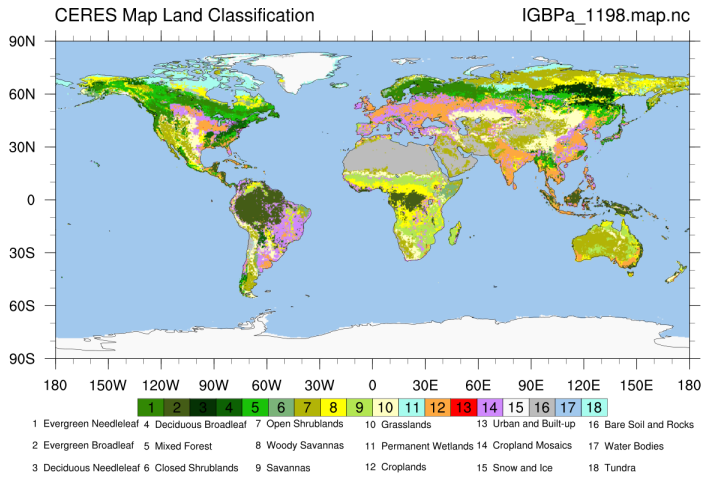
```
f = addfile ("uv300.nc","r")
u = f->U(0,:,{82})          ; Read "U" off the file
```

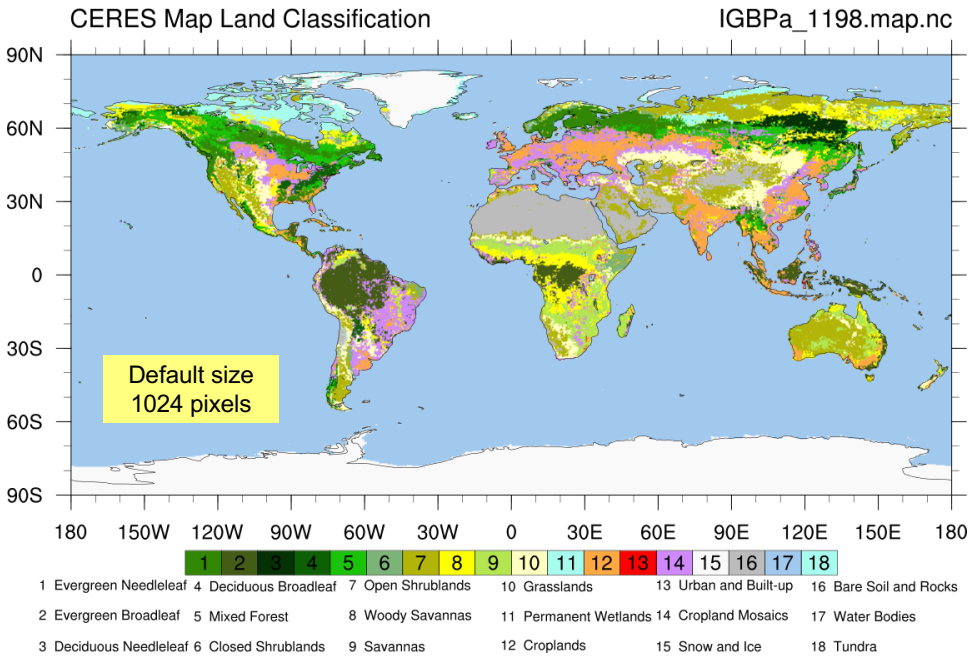
```
wtype          = "png"
wtype@wkWidth  = 2500
wtype@wkHeight = 2500

wks = gsn_open_wks (wtype, "xy")
```

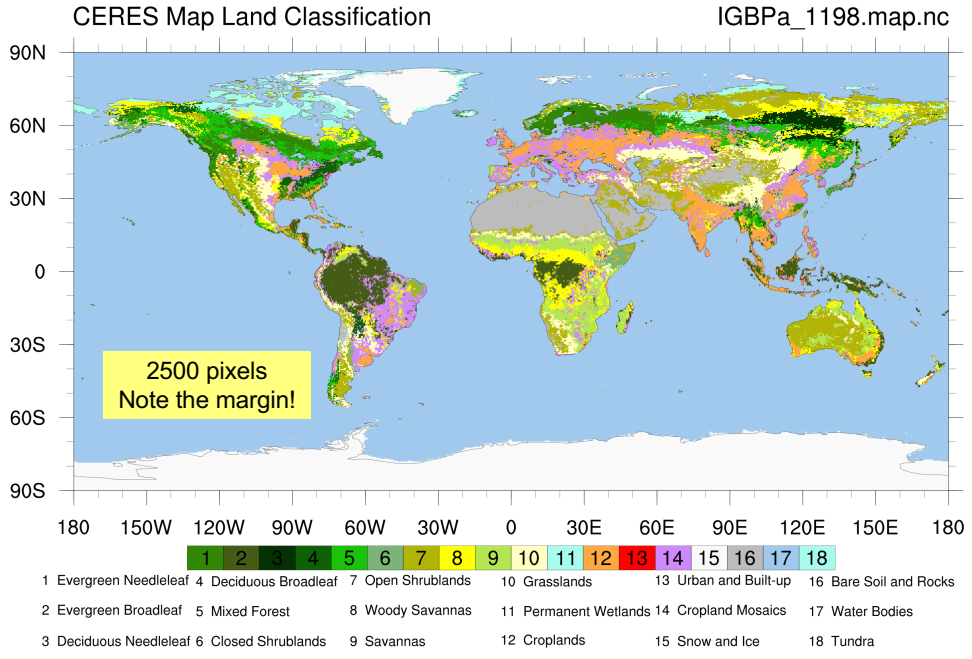
```
res                  = True
res@xyLineColor      = "NavyBlue"
res@xyLineThickness = 3

plot = gsn_csm_xy (wks,u&lat,u,res)
```

```
delete(wks)   ; Close the PNG image first

cmd = "convert –border 8 -bordercolor white –trim xy.png xy.png"
system(cmd)   ; Execute the command
```

Default size
1024 pixels
(no trimming)




CERES Map Land Classification
IGBPa_1198.map.nc
90N
60N
30N
0
30S
60S
90S
180 150W 120W 90W 60W 30W 0 30E 60E 90E 120E 150E 180
Default size
1024 pixels
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
1 Evergreen Needleleaf
2 Evergreen Broadleaf
3 Deciduous Needleleaf
4 Deciduous Broadleaf
5 Mixed Forest
6 Closed Shrublands
7 Open Shrublands
8 Woody Savannas
9 Savannas
10 Grasslands
11 Permanent Wetlands
12 Croplands
13 Urban and Built-up
14 Cropland Mosaics
15 Snow and Ice
16 Bare Soil and Rocks
17 Water Bodies
18 Tundra


CERES Map Land Classification
IGBPa_1198.map.nc
90N
60N
30N
0
30S
60S
90S
180 150W 120W 90W 60W 30W 0 30E 60E 90E 120E 150E 180
2500 pixels
Note the margin!
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
1 Evergreen Needleleaf
2 Evergreen Broadleaf
3 Deciduous Needleleaf
4 Deciduous Broadleaf
5 Mixed Forest
6 Closed Shrublands
7 Open Shrublands
8 Woody Savannas
9 Savannas
10 Grasslands
11 Permanent Wetlands
12 Croplands
13 Urban and Built-up
14 Cropland Mosaics
15 Snow and Ice
16 Bare Soil and Rocks
17 Water Bodies
18 Tundra

Creating images for web or PowerPoint

- Download "convert", part of free ImageMagick package

  http://www.imagemagick.org/script/index.php

- Mac users can use MacPorts:

```
port install imagemagick
```

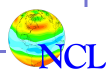
- Linux users:

```
yum install imagemagick
apt-get install imagemagick
```

- Use command like:

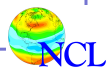
```
convert –geometry 2000x2000 -density 300 -trim xy.ps xy.png
```

- The "-density 300" option is what gives you higher-quality text. You may need to play with this number.
  For posters, use larger values for both the geometry and density.



NCL Graphics topics

- Types of graphics you can create with NCL
- The basics
- Line-by-line examples
- Review
- Customizing NCL environment
- Common mistakes and problems
- Debugging tips
- Creating images for Web and/or presentations
- PyNGL/PyNIO Python modules



PyNIO/PyNGL Python modules

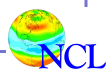
- PyNGL - Python interface to NCL's graphical library
- PyNIO - Python interface to NCL's file input/output library

http://www.pyngl.ucar.edu

PyNIO and PyNGL available under conda

"Python for the Atmospheric and Oceanic Sciences"

http://pyaos.johnny-lin.com/



Compare PyNGL/PyNIO and NCL/GSUN scripts

PyNGL/PyNIO

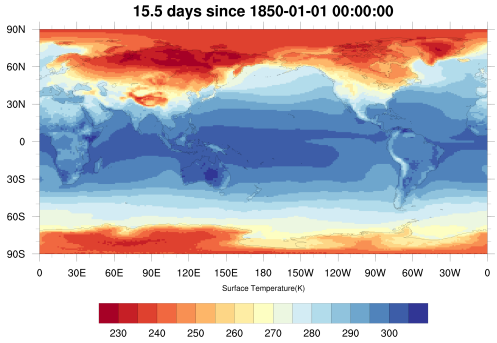
```
import Ngl, Nio

#  Open the NetCDF file.
nf = Nio.open_file("mtemp.cdf","r")

# Get lat/lon/temperature variables.
lat = nf.variables["lat"][:]
lon = nf.variables["lon"][:]
T   = nf.variables["t"][0,:,:]

# Open a PS workstation.
wks = Ngl.open_wks("ps","mecca")

# Contour & scalar field resources.
res                 = Ngl.Resources()
res.sfXArray        = lon
res.sfYArray        = lat
res.cnFillOn        = True

# Draw contour plot.
contour = Ngl.contour(wks,T,res)

Ngl.end()
```

NCL/GSUN

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

begin
;  Open the NetCDF file.
  nf = addfile("mtemp.cdf","r")

; Get lat/lon/temperature variables.
  lat  = nf->lat
  lon  = nf->lon
  T    = nf->t(0,:,:)

; Open a PS workstation.
  wks = gsn_open_wks("ps","mecca")

; Contour & scalar field resources.
  res                          = True
  res@sfXArray                 = lon
  res@sfYArray                 = lat
  res@cnFillOn                 = True
  res@pmLabelBarDisplayMode = "Always"

; Draw contour plot.
  contour = gsn_contour(wks,T,res)

end
```

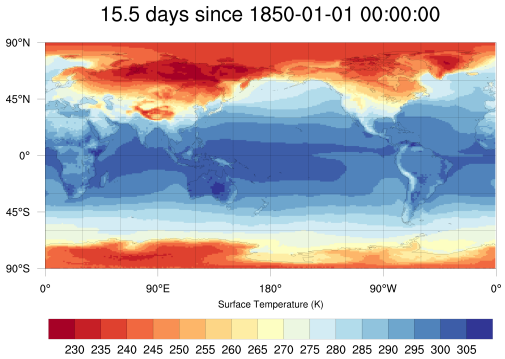
```
filename = "ts_Amon_CESM1-CAM5_historical_r1i1p1_185001-200512.nc"
a    = addfile(filename,"r")
ts   = a->ts
nt   = 0           ; time index to plot

wks = gsn_open_wks("png","compare_ncl")

res                   = True

res@gsnMaximize       = True     ; maximize plot in frame

res@cnFillOn          = True     ; turn on contour fill
res@cnLinesOn         = False
res@cnLineLabelsOn    = False
res@cnFillPalette     = "MPL_RdYlBu"
res@cnFillMode        = "RasterFill"    ; Faster than default "AreaFill"

res@mpCenterLonF      = 180           ; default is lon=0

res@tiMainString = ts&time(nt) + " " + ts&time@units
res@tiMainFontHeightF = 0.02         ; default is a big large

res@lbTitleString      = ts@long_name + "(" + ts@units + ")"
res@lbTitleFontHeightF = 0.01
res@pmLabelBarOrthogonalPosF = 0.12

res@gsnRightString    = ""
res@gsnLeftString     = ""

plot = gsn_csm_contour_map(wks,ts(nt,:,:),res)
```

```
import numpy, Nio, Ngl

fname = "ts_Amon_CESM1-CAM5_historical_r1i1p1_185001-200512.nc"
a     = Nio.open_file(fname)
ts    = a.variables["ts"]           # Get the "ts" variable object
lat   = a.variables["lat"][:]
lon   = a.variables["lon"][:]
time  = a.variables["time"]
nt    = 0     # time index to plot

wks = Ngl.open_wks("png","compare_pyngl")

res = Ngl.Resources()

res.cnFillOn          = True          # turn on contour fill
res.cnLinesOn         = False         # turn off contour lines
res.cnLineLabelsOn    = False         # turn off line labels
res.cnFillPalette     = "MPL_RdYlBu"
res.cnFillMode        = "RasterFill"  # Faster than default "AreaFill"

res.mpCenterLonF      = 180           # default is lon=0

res.lbTitleString      = "%s (%s)" % (ts.long_name,ts.units)
res.lbOrientation      = "horizontal"  # default is "vertical"

res.tiMainString = "%s %s" % (time[nt],time.units)
res.tiMainFontHeightF = 0.02         # default is a big large

res.sfXArray          = lon     # Additional resources needed for
res.sfYArray          = lat     # putting contours on map

plot = Ngl.contour_map(wks,ts[nt,:,:],res)
```



NCL plot

Python/PyNGL plot



Where to start

- Install one of the UNIX editor enhancements
- Download your own datasets (highly recommended) or use sample files
- Write simple scripts to read data and plot it. . . OR. . .work on existing scripts
- Browse "Examples" on NCL website
- Review graphical examples used in lecture
- Refer to index in back of book for web links
- Ask instructors questions!

Where to start – more stuff to do

- Try adding shapefile outlines to a map plot.
  - Use gadm.org to download country shapefiles
- Write a script that plots every "meaningful" variable from one of your own files.
- Try cleaning up an existing script to "modernize" it
  - Use "cnFillPalette" or "vcLevelPalette" instead of "gsn_define_colormap"
  - Use named colors (e.g. "blue") instead of indexed color (e.g. 5)
  - Remove unneeded do loops
- Improve your graphical images
  - Increase the size of your PNG images
  - Try different color maps or colors
  - Increase the thickness of lines