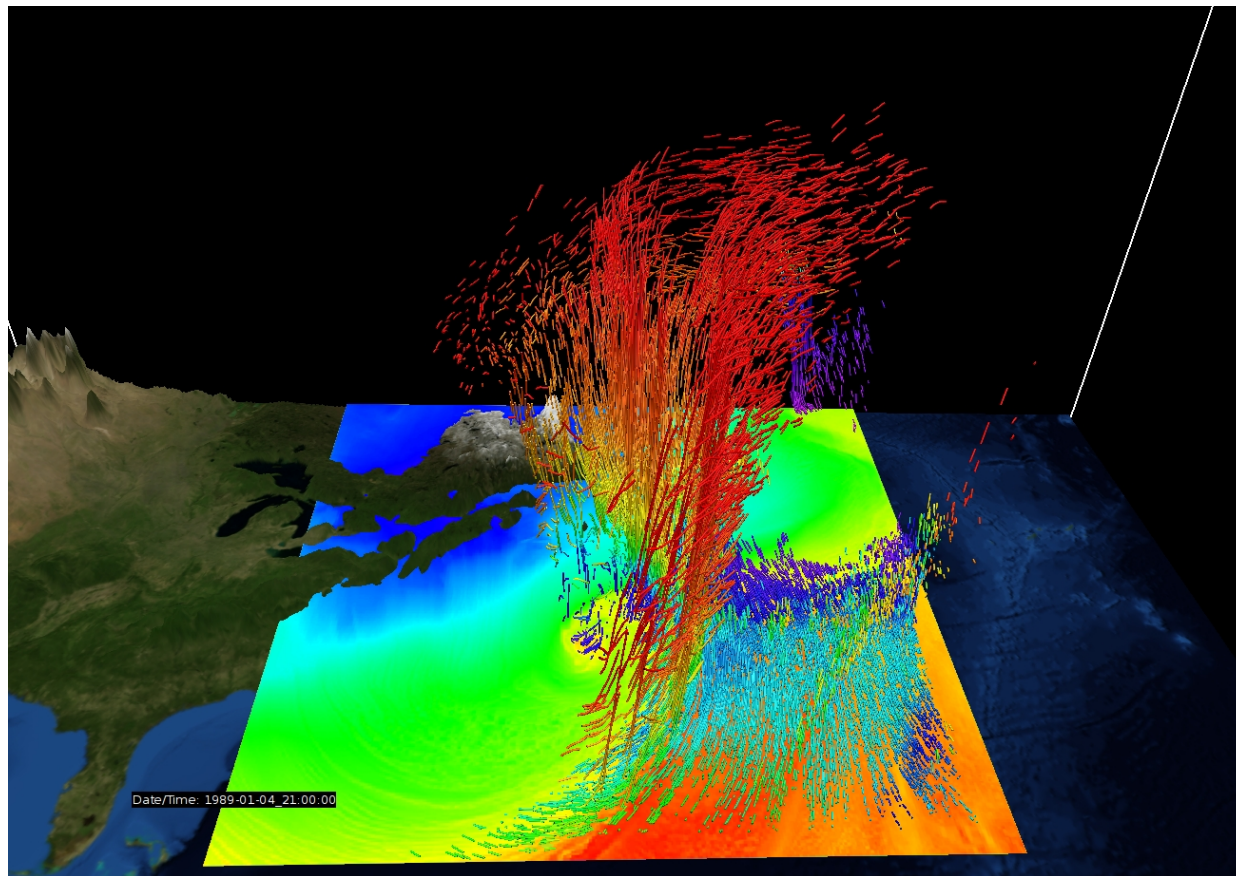


Post-processing Tools

Cindy Bruyère



Supported Post-Processing Packages

- **NCL** UG: 9-2
 - Graphical package
- **ARWpost** UG: 9-28
 - Converter (GrADS)
- **RIP4** UG: 9-19
 - Converter and interface to graphical package NCAR Graphics
- **UPP** UG: 9-35
 - Converter (GrADS & GEMPAK)
- **VAPOR** UG: 9-58
 - Converter and graphical package
 - *Support: VAPOR*
- **IDV** unidata.ucar.edu
 - GRIB (from UPP)
 - GEMPAK (from wrf2gem)
 - Vis5d
 - CF complaint data (from wrf_to_cf)
 - *Support: unidata*
- **GEMPAK**
 - Data from wrf2gem or UPP
 - *Support: unidata*

**MatLab; IDL; R; ferret;
panoply; python**



Picking a Post-Processing Tool

- Can it read your data
 - netCDF (CF), GRIB, ASCII, shapefiles
- Do you have to first pre-process the data
- Can it handle big datasets
- Is it purely a visualization tool, or can you do post-processing as well
- Which diagnostic / statistical functions does it have
- 3D or 2D visualization
- Can it deal with staggered grids
- How easy is it to add diagnostics
- How is data below ground handled
- Vertical grids
- How is model time stamps handled
- Easy of use
- Cost of package
- Support



Data Handing

	NCL	RIP4	GrADS	UPP	VAPOR	IDV
netCDF		ripdp	ARWpost	converter	converter	converter
GRIB						
ASCII						
shapefiles						
geogrid & metgrid output						
intermediate file format	V6.2.0					
wrfinput data						
Idealized data						
wrfoutput						
big data						



Post-Processing

	NCL	RIP4	GrADS	UPP	VAPOR	IDV
post-processing						
data output						
3D						
diagnostics	Some	Lots	Some	Some	Limited	Limited
add diagnostics	Very Easy	Easy	Easy	Relatively Easy	Less Easy	Less Easy
vertical output coordinate	model pressure height	model pressure height	model pressure height	pressure	model	model
extrapolate below ground						



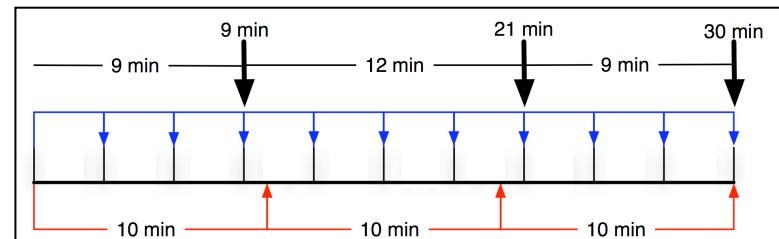
wrfout Timestamps

- *output files not on exact times*

- 2013-01-31_00
- 2013-01-31_09
- 2013-01-31_21
- 2013-01-31_30

- Delta Times =
9 ; 12 ; 9 minutes

history_interval=10 ; time_step=180 (3 min)

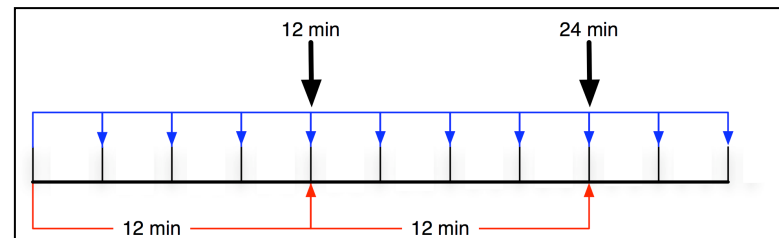


- *output files on exact times*

- 2013-01-31_00
- 2013-01-31_10
- 2013-01-31_20
- 2013-01-31_30

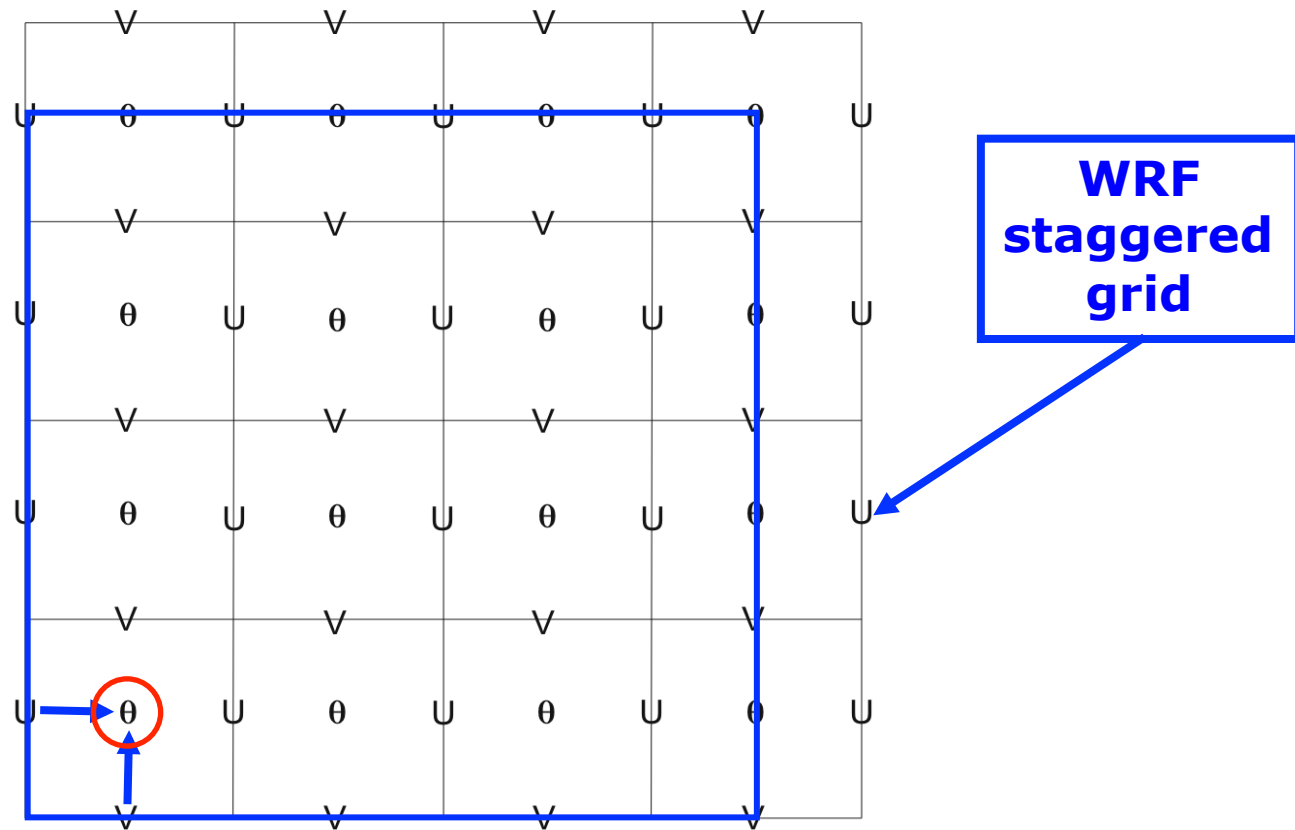
- Delta Times =
10; 10; 10minutes

history_interval=12 ; time_step=180 (3 min)



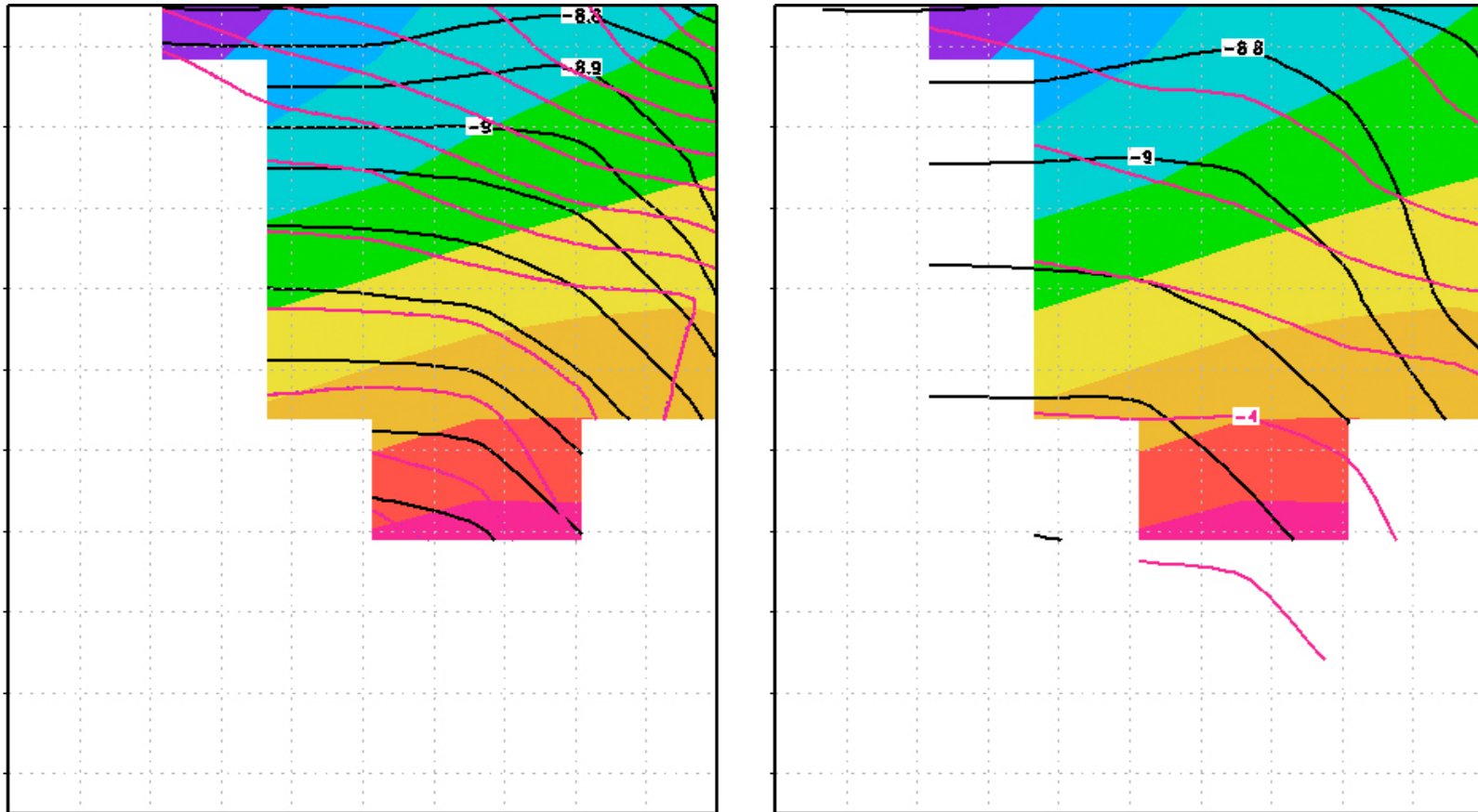
Model Staggering

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

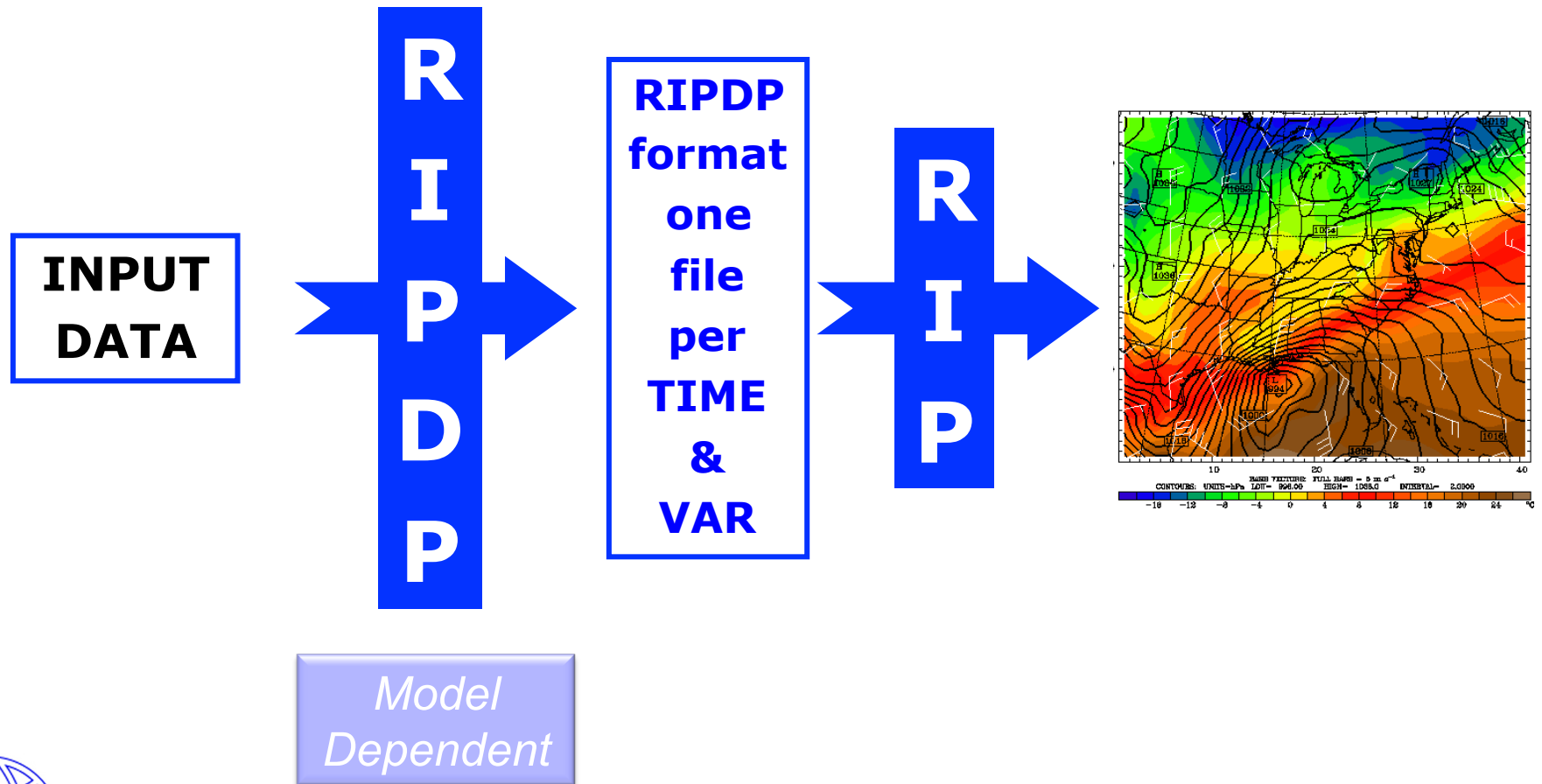


Model Staggering

shaded=T ; black=U ; red=V

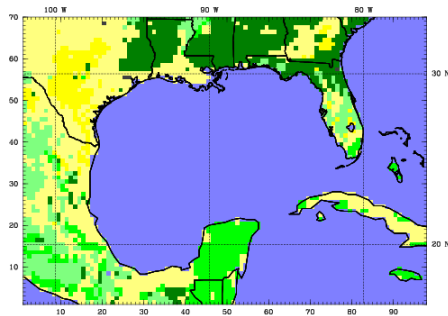


RIP4

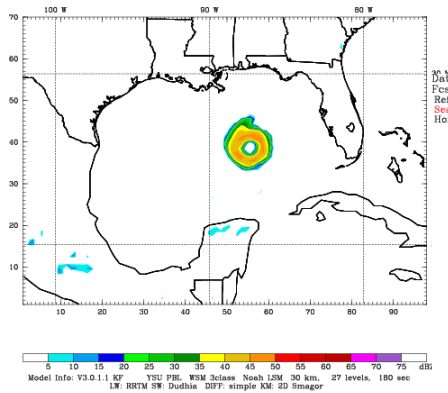


RIP4 - Examples

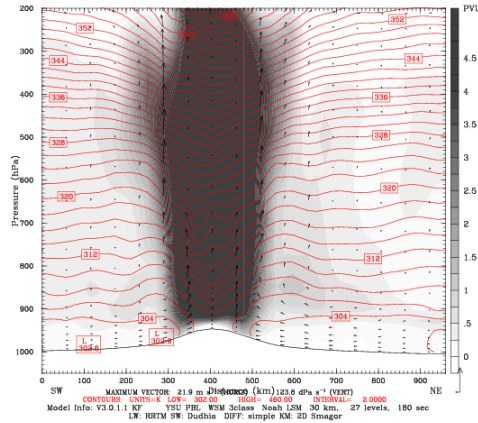
Dataset: katrina RIP: katrina Init: 0000 UTC Sun 28 Aug 05
Fest: 0.00 h Valid: 1200 UTC Sun 28 Aug 05 (0600 MDT Sat 27 Aug 05)
Land use category



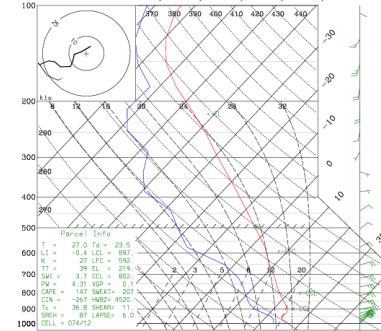
Dataset: katrina RIP: katrina Init: 0000 UTC Sun 28 Aug 05
Fest: 12.00 h Valid: 1200 UTC Sun 28 Aug 05 (0600 MDT Sun 28 Aug 05)
Reflectivity () at k-index = 27



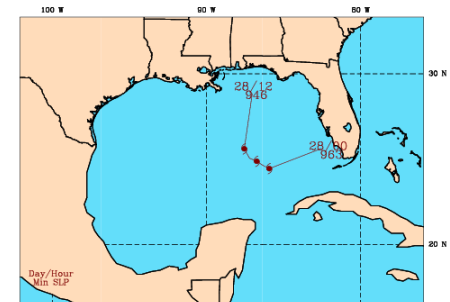
Dataset: katrina RIP: katrina Init: 0000 UTC Sun 28 Aug 05
Fest: 12.00 h Valid: 1200 UTC Sun 28 Aug 05 (0600 MDT Sun 28 Aug 05)
Potential vorticity XY= 45.0, 30.0 to 70.0, 50.0
Potential temperature XY= 45.0, 30.0 to 70.0, 50.0
Circulation vectors XY= 45.0, 30.0 to 70.0, 50.0



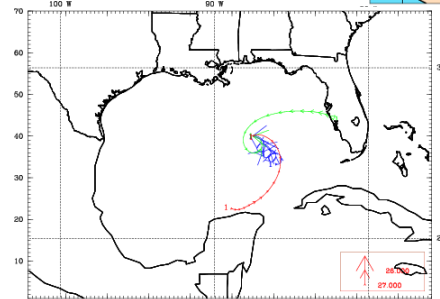
Dataset: katrina RIP: katrina Init: 0000 UTC Sun 28 Aug 05
Fest: 12.00 h Valid: 1200 UTC Sun 28 Aug 05 (0600 MDT Sun 28 Aug 05)
Temperature x,y= 44.86, 54.37 lat,lon= 29.99, -80.35 min-KM0772521
Dewpoint temperature x,y= 44.86, 54.37 lat,lon= 29.99, -80.35 min-KM0772521
Horizontal wind vectors x,y= 44.86, 54.37 lat,lon= 29.99, -80.35 min-KM0772521



Dataset: katrina RIP: typhoon
Fest: 0.00 h Valid: 0000 UTC Sun 28 Aug 05 (1800 MDT Sat 27 Aug 05)
Typhoon Track



Dataset: katrina RIP: traj plot Init:
Fest: 0.00 h Valid: 0000 UTC Sun 28 Aug 05 ()
Trajectories from hour 0.000 to 12.000
Trajectories from hour 0.000 to 12.000
Trajectories from hour 0.000 to 12.000



RIP4 General Information

- Requires NCL
 - <http://www.ncl.ucar.edu>
- Source Code:
 - http://www.mmm.ucar.edu/wrf/users/download/get_source.html
- Documentation
 - In program tar file under the Doc/ directory
 - <http://www.mmm.ucar.edu/wrf/users/docs/ripug.htm>
- OnLine Tutorial:
 - <http://www.mmm.ucar.edu/wrf/users/graphics/RIP4/RIP4.htm>



RIP4 on your computer

- set environment variables

```
setenv RIP_ROOT /usr/$USER/RIP4  
setenv NCARG_ROOT /usr/local/ncl
```

- Configure

./configure

*check configure.rip to ensure netCDF paths are correct
gfortran ; z and png libraries may be required*

- Compile

./compile

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

- RIP4 has 2 parts (RIPDP and RIP)

ripdp_mm5

ripdp_wrfarw
ripdp_wrfnmm



Running ripdp & rip

ripdp_wrfarw [-n namelist-file] *optional*
<model_data_name> [basic/all] <input_file(s)>

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

Example:

ripdp_wrfarw RIPDP/CaseX all wrfout*

rip [-f] RIPDP/CaseX rip_sample.in

output

***[rip_sample.out]
rip_sample.TYPE***



RIP4 User Input File

```
&userin  
.....  
/  
&trajcalc  
.....  
/
```

Namelists

=====

----- Plot Specification Table -----

=====

feld=

feld=

=====

feld=

feld=

=====

Graphics

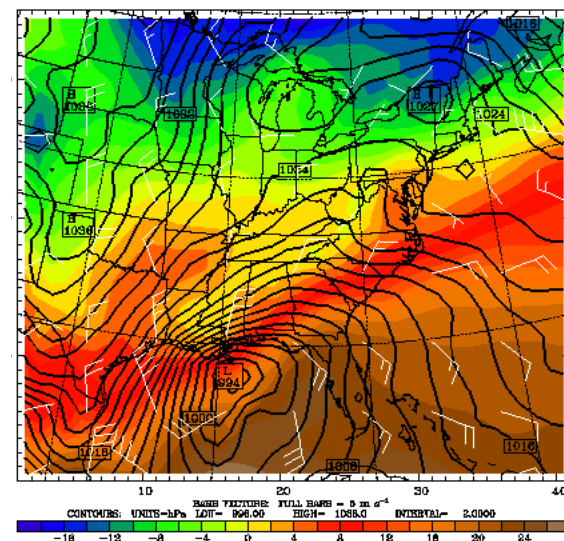


Creating a Plot with RIP4

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

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

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

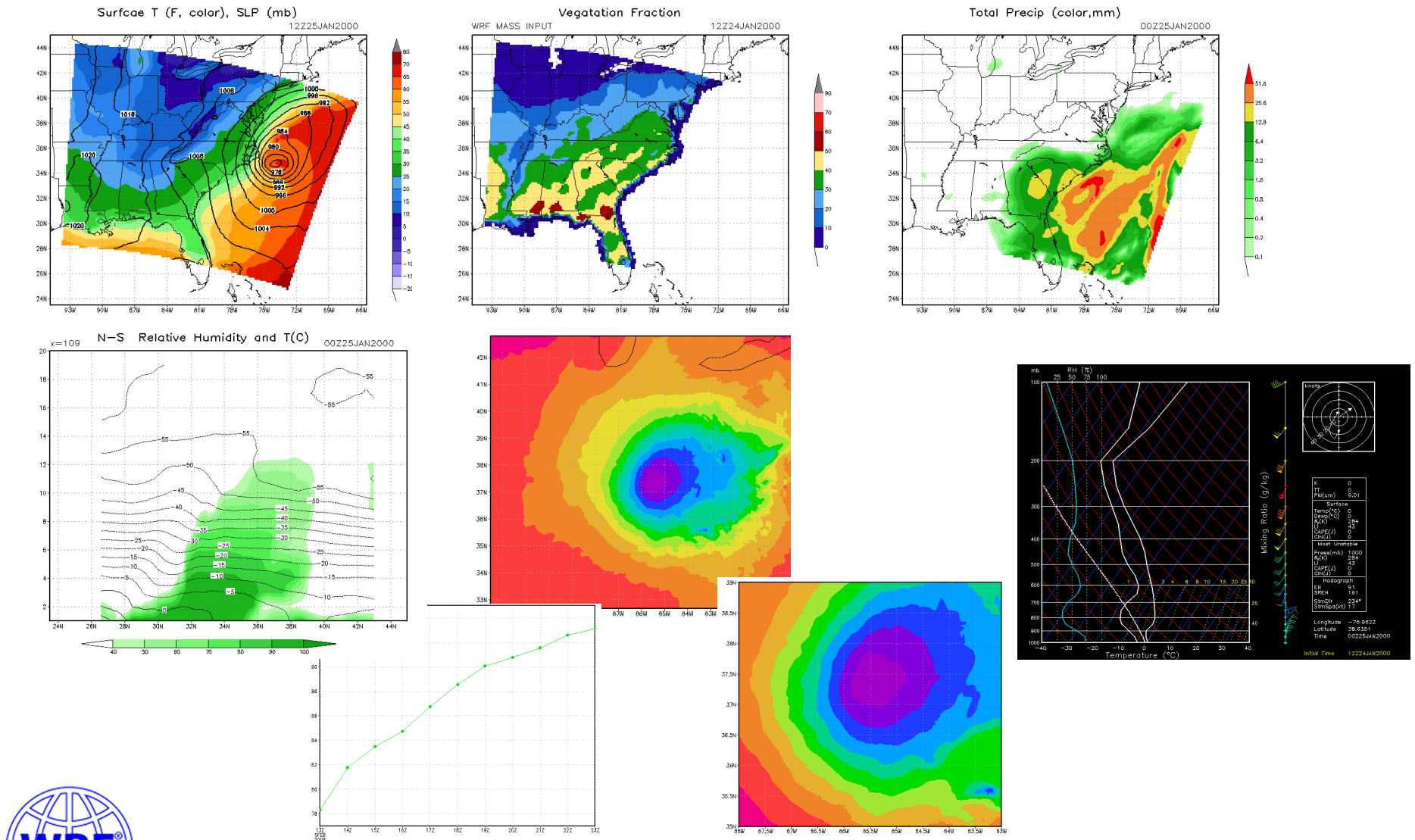


ARWpost

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



ARWpost - Examples



ARWpost - *converter*

- Download Code (<http://www.mmm.ucar.edu/wrf/users>)
- OnLine Tutorial
<http://www.mmm.ucar.edu/wrf/users/graphics/ARWpost/ARWpost.htm>
- Compile (*similar to WPS*)
./configure & ./compile
- For GrADS output
 - GrADS libraries only needed to display data (*freely available*)
 - <http://grads.iges.org/grads/grads.html>



namelist.ARWpost

<i>input_root_name</i>	Path and root name of files to use as input. <i>Do not only provide directory name.</i> Can use wild characters.
<i>output_root_name</i>	Output root name. output_root_name.dat & output_root_name.ctl
<i>split_output</i>	Split your GrADS output files into a number of smaller files (<i>a common .ctl file will be used for all .dat files</i>).
<i>frames_per_outfile</i>	If <i>split_output</i> is .True., how many time periods are required per output (.dat) file.

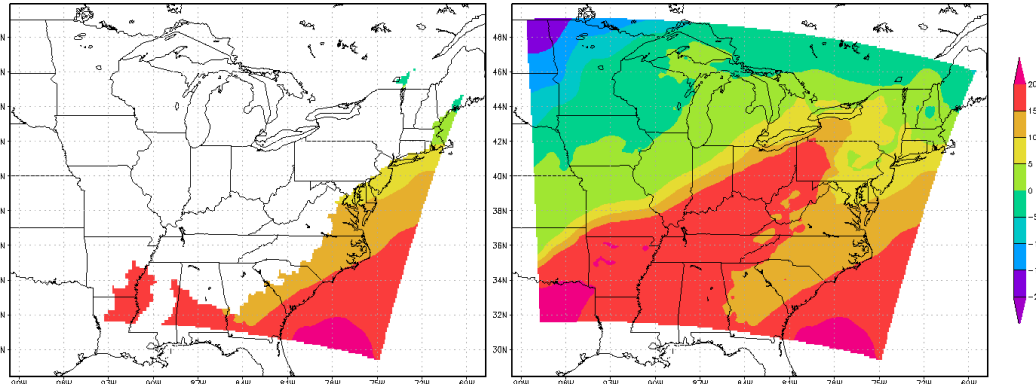


namelist.ARWpost

<i>plot</i>	Which fields to process. (<i>all</i> , <i>list</i> , <i>all_list</i>) Order has no effect, i.e., “all_list” and “list_all” “list” – list variables in “fields”
fields	Fields to plot. Only used if list was used in the “plot” variable. Must use to generate diagnostics. Can be used for model output.
Available diagnostics: cape, cin, mcape, mcin, clfr, dbz, max_dbz, geopt, height, lcl, lfc, pressure, rh, rh2, theta ,tc, tk, td, td2, slp, umet, vmet, u10m, v10m, wdir, wspd, wd10, ws10	



namelist.ARWpost

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



GrADS - .ctl file

```
dset ^test.dat
options byteswapped
undef 1.e37
title OUTPUT FROM WRF V2.2 MODEL
pdef 259 163 lcc 40.000 -98.000 130.000 82.000
      60.00000 30.00000 -98.00000 22000.000 22000.000
xdef 877 linear -141.49254 0.09909910
ydef 389 linear 18.88639 0.09909910
```

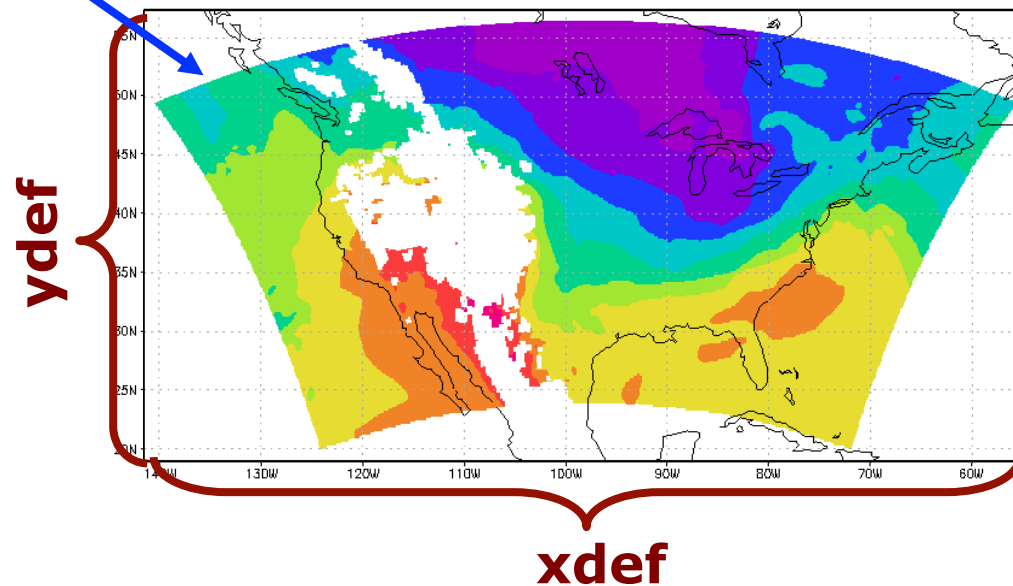
options byteswapped

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



GrADS - .ctl file

```
dset ^test.dat
options byteswapped
title OUTPUT FROM WRF V3.2 MODEL
pdef 259 163 lcc 40.000 -98.000 130.000 82.000
60.000000 30.000000 -98.000000 22000.000 22000.000
xdef 877 linear -141.49254 0.09909910
ydef 389 linear 18.88639 0.09909910
```



How to add diagnostics

- **RIP4**
 - Create a subroutine (note RIP4 expects the code to be in “j/l/-k” orientation)
 - Add links to the RIP4/src/fields.f routine
 - Add new subroutine to RIP4/src/Makefile
- **ARWpost**
 - Create a subroutine
 - Add links to ARWpost/src/module_diagnostics.f90
 - Add new subroutine to ARWpost/src/Makefile



VAPOR visualization of WRF-ARW data



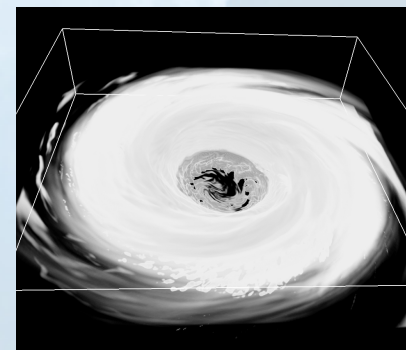
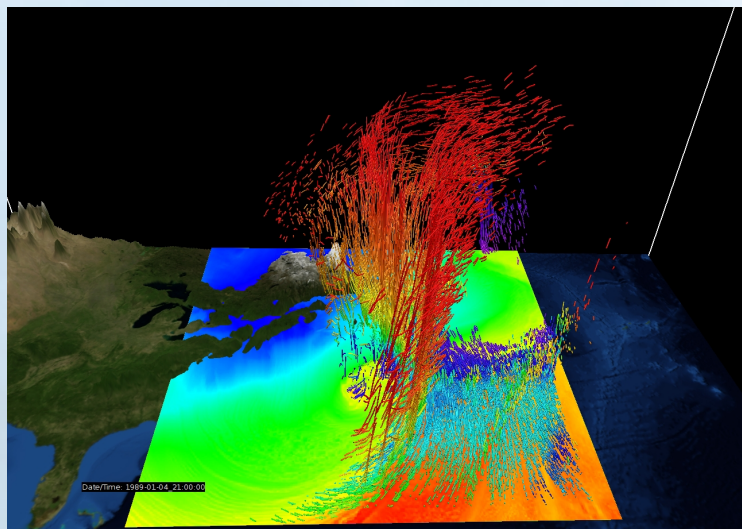
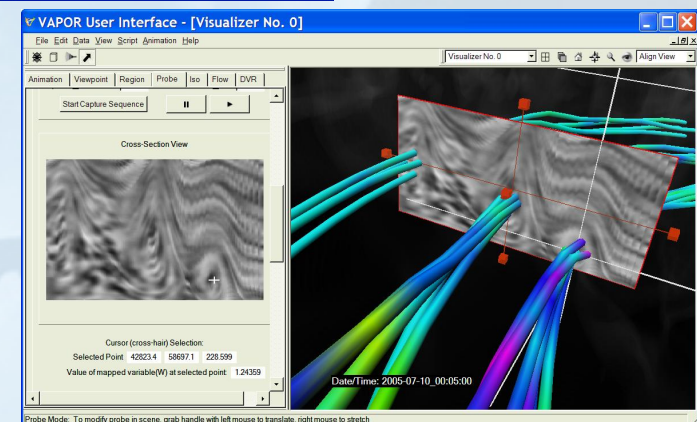
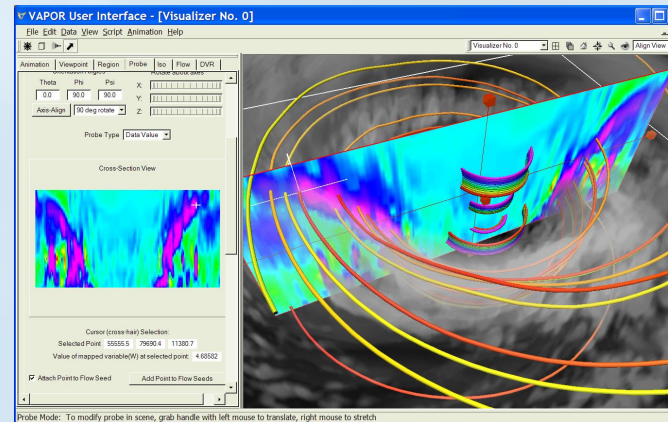
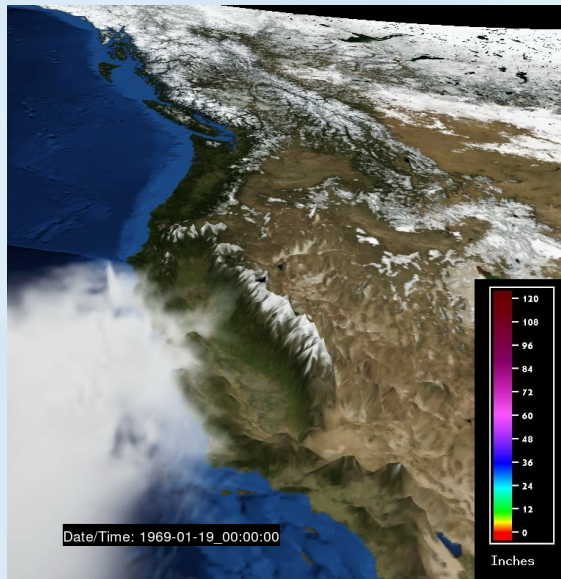
Visualization and Analysis Platform for Oceanic, atmospheric and solar Research

Alan Norton
alan@ucar.edu
vapor@ucar.edu

National Center for Atmospheric Research



VAPOR visualization of WRF-ARW data



VAPOR Installation



- Available for Linux, Windows, or Mac systems
- Should have a reasonably modern graphics card
 - nVidia, ATI or AMD graphics cards are good; others may not perform all visualizations.
- From the VAPOR website <http://www.vapor.ucar.edu>: Download appropriate binary installer from the VAPOR download page, follow the installation instructions.
- You will need Administrative privileges on Mac
- Note that on Linux and Mac you need to source vapor-install.csh in your shell before running any VAPOR commands.
- Run the vaporgui application to visualize your data

VAPOR visualization of WRF-ARW data



A short summary of VAPOR capabilities

1. Read or convert WRF-ARW output files
2. Apply geo-referenced images to the terrain
3. Calculate 2D and 3D derived variables in Python
4. Volume render 3D variables
5. Display isosurfaces of 3D variables
6. Display color-mapped 2D variables on planes or terrain-mapped.
7. Use wind barbs to show flow direction and speed
8. Display streamlines or path lines in scene
9. Insert contour planes, use them to position flow seeds.
10. Image-based flow shows flow motion in 2D slices
11. Create animated 3D sequences

Reading or converting WRF-ARW output files

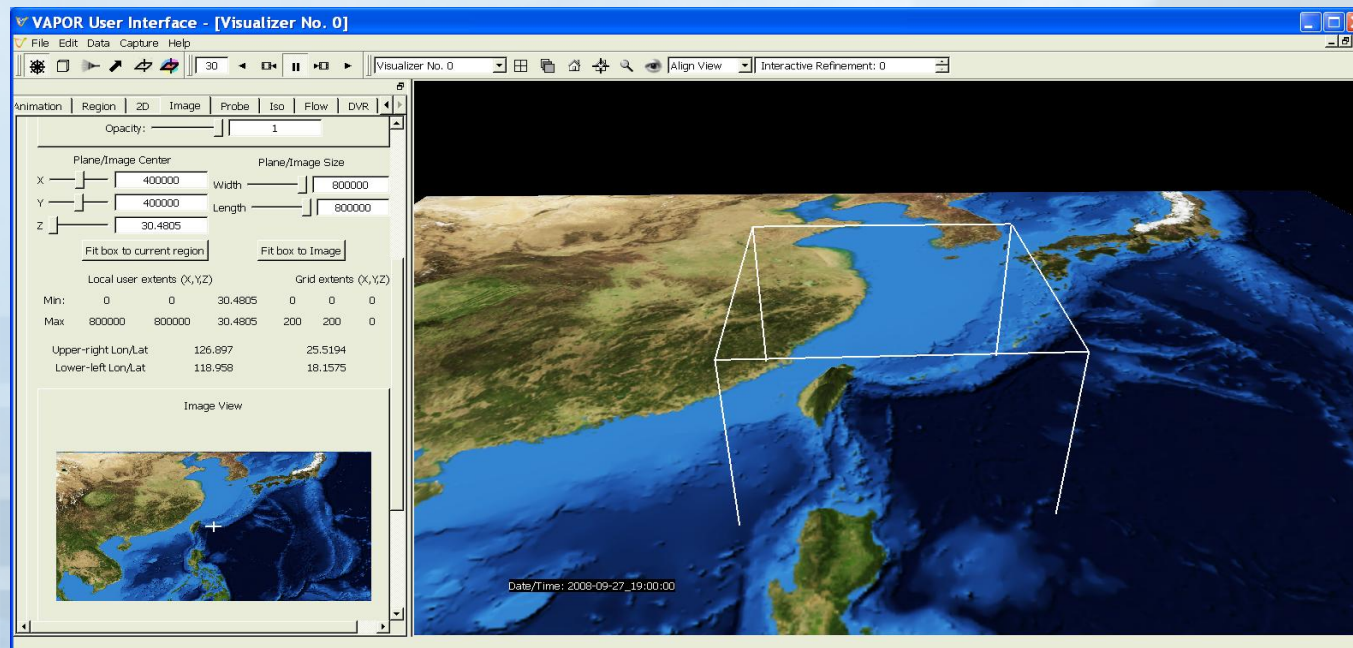


- To directly read WRF output:
 - Run vaporgui
 - All data must be on the same grid, using the same nesting level.
 - Specify “Import WRF-ARW output files” from the Data menu, and select all the wrfout files to visualize
- For interactive visualization of large WRF-ARW datasets, it’s best to convert WRF data to the VAPOR data format, using wrfvdfcreate and wrf2vdf utilities.
 - `wrfvdfcreate wrfoutfiles... vdf file.vdf`
creates a VAPOR metadata file “vdf file.vdf” that describes a set of wrfout files.
 - `wrf2vdf vdf file.vdf wrfoutfiles...`
converts the specified wrfout files to a vapor data collection
 - From the vaporgui Data menu, load the file “vdf file.vdf” to visualize the converted data

Apply images to use in the VAPOR scene



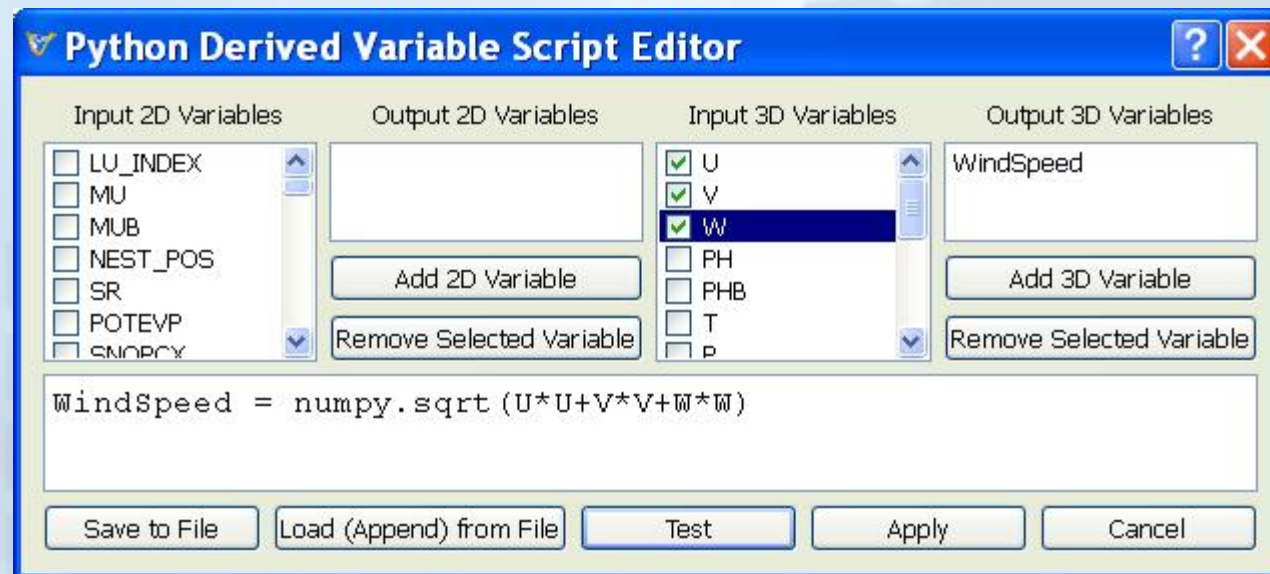
- Geo-referenced satellite images can be retrieved from the Web, and VAPOR will insert them at the correct world coordinates.
 - VAPOR provides a shell script “getWMSImage.sh” that can be used to retrieve Web Mapping Service images for a specified longitude/latitude rectangle
- Also, several useful images are installed with vapor; e.g. state or national boundary maps, NASA’s Blue Marble image of the earth.
- From the image panel, specify the image file, apply to terrain.



Create derived variables with Python



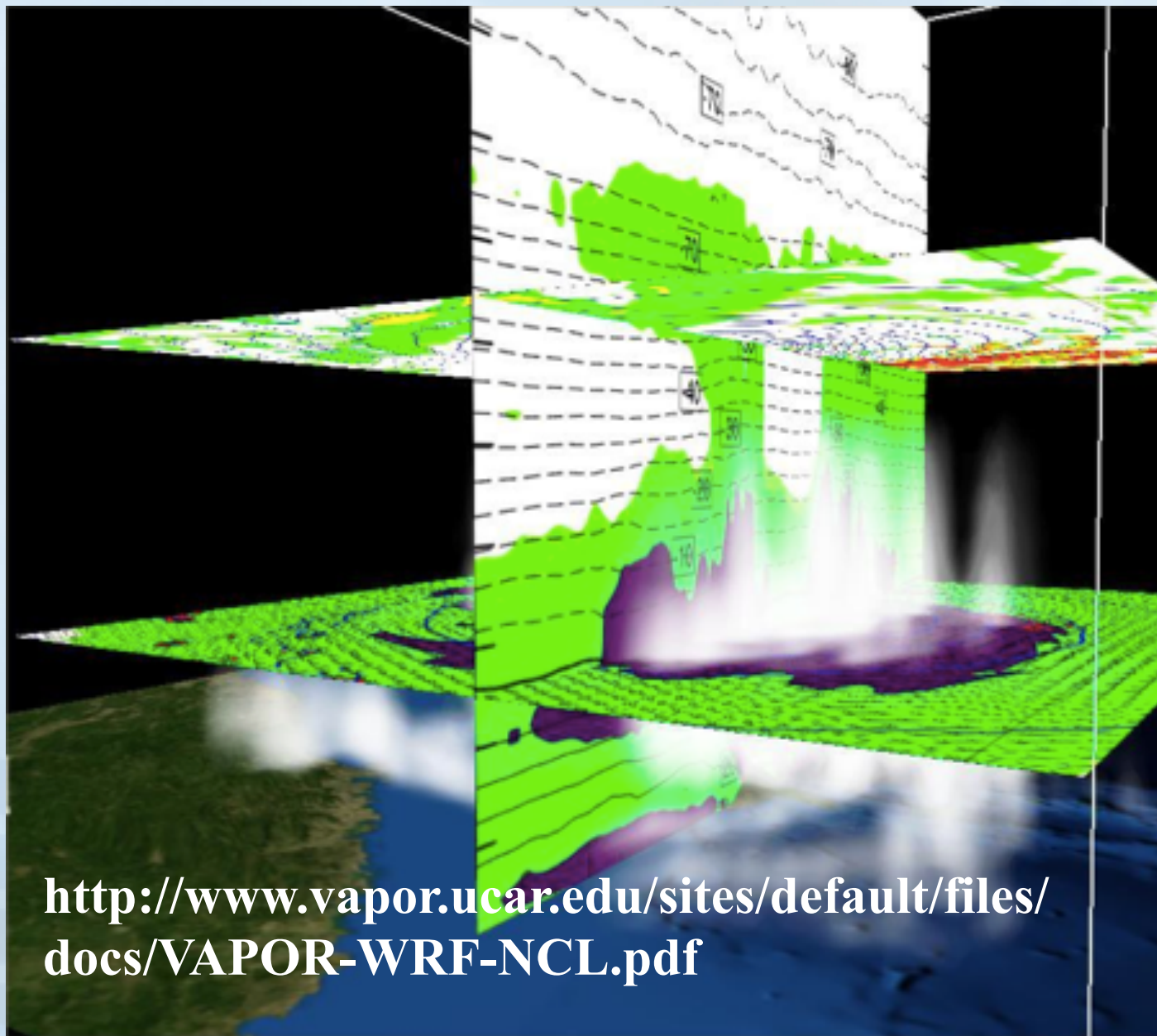
- From the Edit menu, “Edit Python program defining a new variable”
- Use Python script editor to define variables as arithmetic expressions of other variables.
- Variables are evaluated and cached as needed for visualization
- Python functions are also provided to derive several useful variables from WRF data; e.g. cloud-top temperature, relative humidity, potential vorticity, sea-level pressure, dewpoint temperature, radar reflectivity, equivalent potential temperature, wind shear, temperature in degrees Kelvin.



VAPOR / NCL



NCAR



<http://www.vapor.ucar.edu/sites/default/files/docs/VAPOR-WRF-NCL.pdf>



IDV

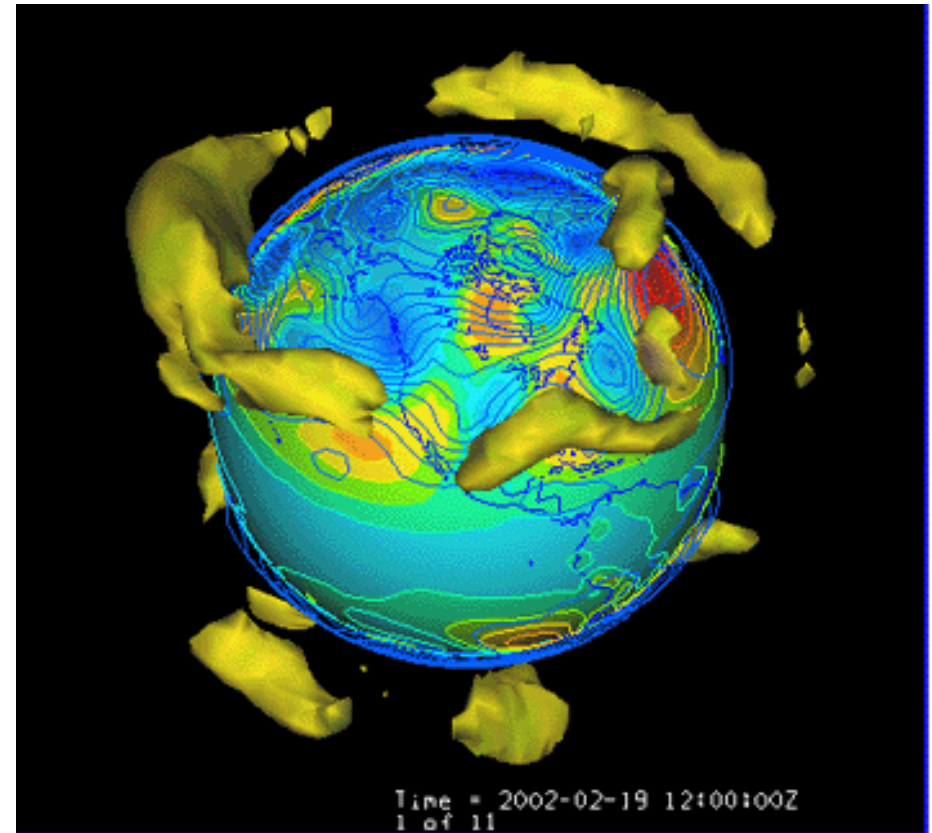
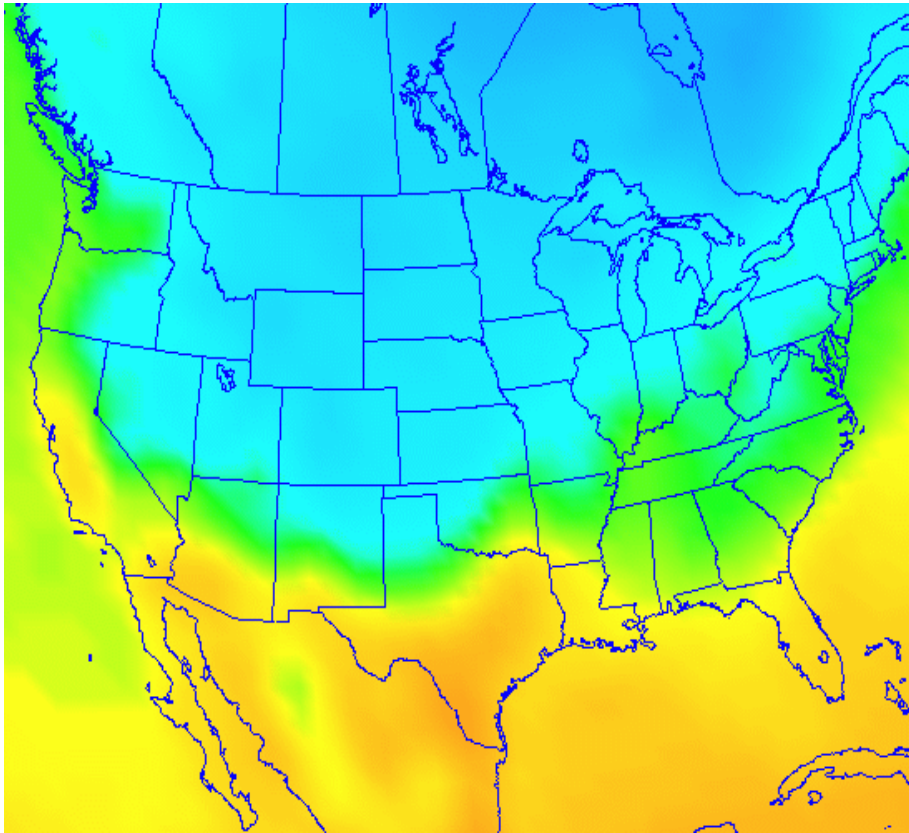
Integrated Data Viewer

Yuan Ho and Julien Chastang
Unidata Program Center/UCAR



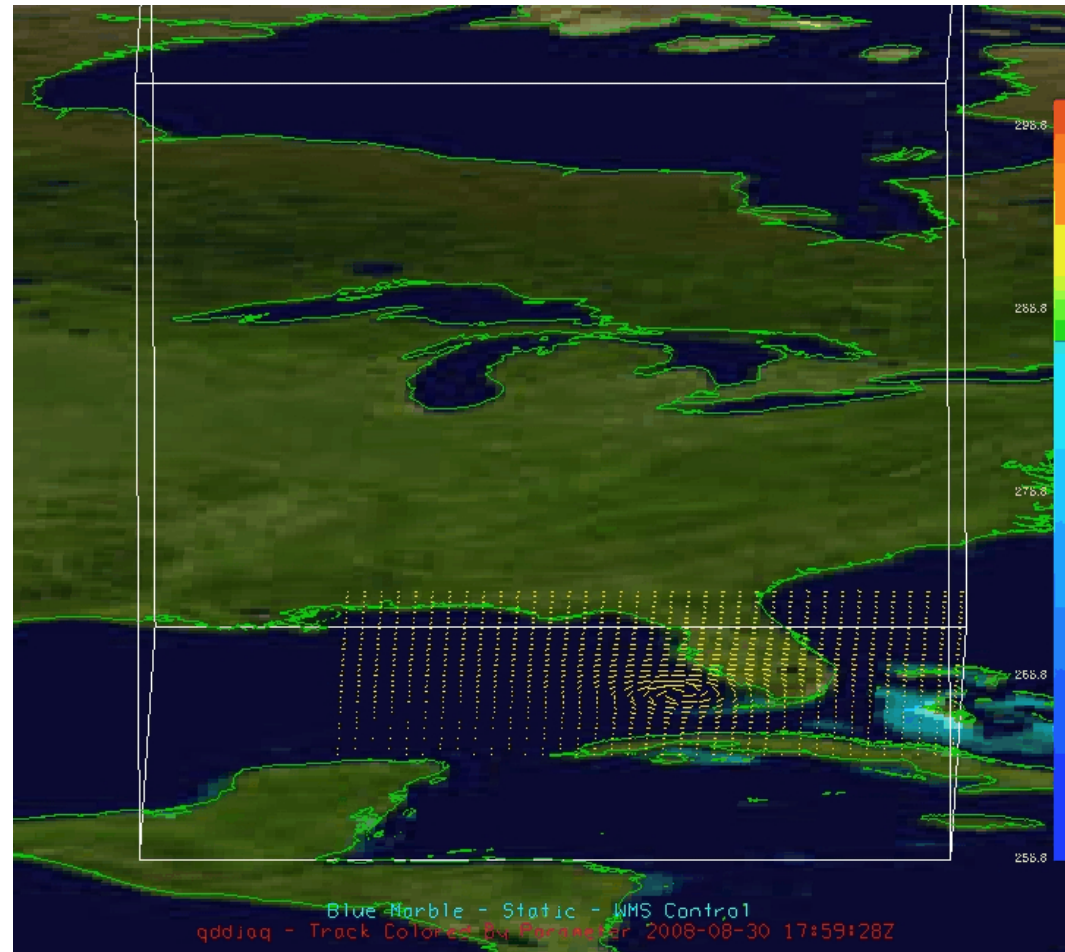


Unidata IDV – What can it do?





Unidata IDV – What can it do?

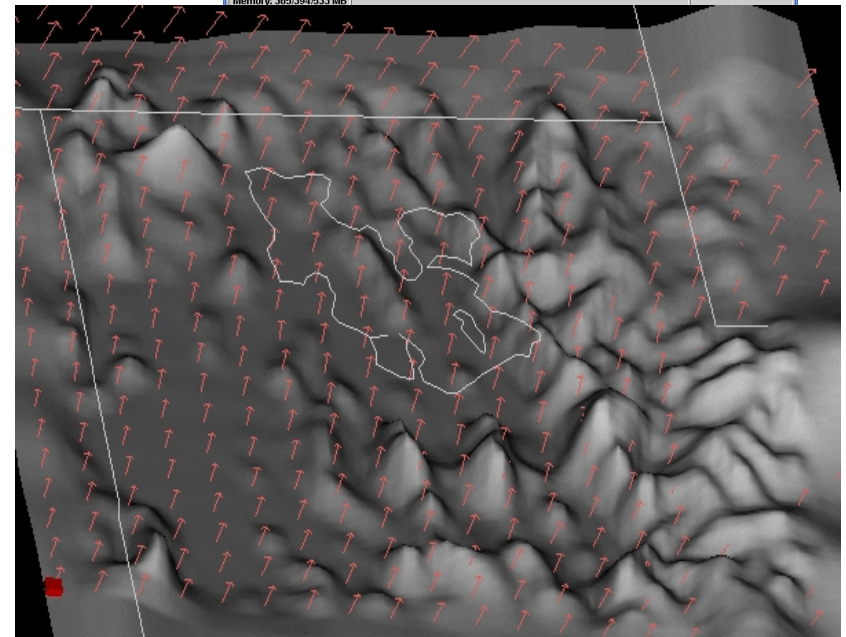
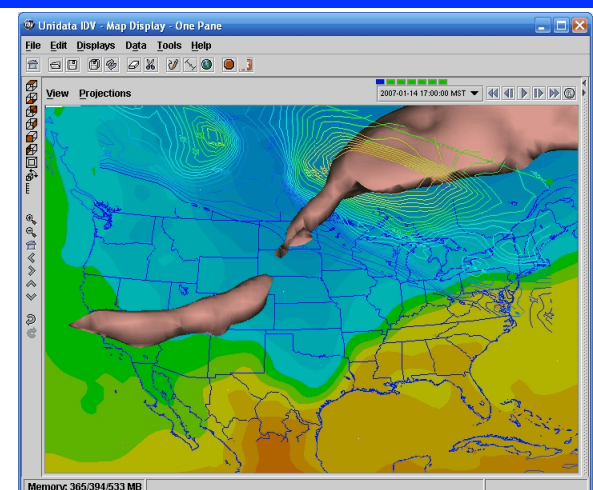




Unidata IDV – What is it?

- Unidata Integrated Data Viewer

- 2D and 3D visualization
 - GUI or scripting (Jython, ISL) interface
- Interactive probes for dataset exploration
 - Parameter readouts, vertical profiles, time/height displays, etc.
- A rich set of analysis capabilities
- Integrate model and observational data
- Access local and remote datasets
- Visualize and analyze post-processed WRF output
 - works best if grid is unstaggered





Supported Data Sources

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



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



Unidata IDV – Where to get it?

- <https://www.unidata.ucar.edu/software/idv>
 - Point-and-click installers
 - Windows (.exe), Mac (.dmg), and Linux (.sh) installers available for both 32 and 64-bit systems
- **System requirements:**
 - 2+ GB RAM
 - Java 1.6
 - *Latest* video card driver

