PV_ATMOS: Atmospheric and Oceanic Data Visualization with ParaView

Martin Jucker

Courant Institute for Mathematical Sciences, New York University

github.com/mjucker/pv_atmos
3 in 1: Science, Beauty, Fun
ParaView: Generalities

Installation and geekyness:

- Free, open source software
- Binaries available for any platform
  - No installation required
  - No python required (comes with it)
- Geekyness up to the user
  - No coding, just mouse and GUI, or
  - Pure coding, as python module, or
  - Anything in-between
Why ParaView?

Development and application:

• Very powerful, general viz software, based on VTK and Python
• Active development by Kitware, support from Sandia, LANL, many more
• Used in plasma physics, aerodynamics, medicine, biology, geosciences, particle physics, engineering, ...

ParaView is here to stay!
www.paraview.org
Why ParaView?

Development and application:

- Very powerful, general viz software, based on VTK and Python
- Active development by Kitware, support from Sandia, LANL, many more
- Used in plasma physics, aerodynamics, medicine, biology, geosciences, particle physics, engineering, ...

ParaView is here to stay!

www.paraview.org
pv_atmos: specializing PV

pv_atmos.basic: Data manipulation, helper tools

- Load netCDF data
- Convert lat-lon to sphere
- Logarithmic coordinates (→ pressure coordinates)
- Adjust aspect ratio

```python
LoadData( fileName, ncDims=['lon','lat','pfull'],
          aspectRatios=[1,1,20], logCoords=2, basis=[1000] )

Make3D( expandVar, expandDir='z' )

Cart2Spherical()

HideAll(), ShowAll(), DeleteAll()
```
pv_atmos: specializing PV

pv_atmos.grids: Axes, Labels, Grids, Slices

- Axes with or without labels
- Labels at specific places
- Grid and/or data planes at the correct location in physical space.

```python
AddGrid ( xlevels, ylevels, zlevels,
          AxisNames=["lon","lat","pressure [hPa]" ])
AddGridPlane ( dim, dimVal )
AddGridLabel ( dim, dimVal, LabelSize=5.0 )
```
(output_nc, Coor) = LoadData( fileName, ncDims=['lon', 'lat', 'pfull'] )
(output_nc, Coor) = LoadData( fileName, ncDims=['lon', 'lat', 'pfull'] )
Globe = Cart2Spherical(1.0, Coor)
(output_nc, Coor) = LoadData( fileName, ncDims=['lon','lat','pfull'] )
Globe = Cart2Spherical(1.0, Coor)
Cont = Contour(ContourBy='ucomp', Isosurfaces=[25])
(output_nc, Coor) = LoadData( fileName, ncDims=['lon', 'lat', 'pfull'] )
Globe = Cart2Spherical(1.0, Coor)
Cont = Contour(ContourBy='ucomp', Isosurfaces=[25])
Shells = SphericalShells( radius, src=Coor, shellValues=[10,1] )
(output_nc, Coor) = LoadData( fileName, ncDims=['lon','lat','pfull'] )
Globe = Cart2Spherical(1.0, Coor)
Cont = Contour(ContourBy='ucomp', Isosurfaces=[25])
Shells = SphericalShells( radius, src=Coor, shellValues=[10,1] )
< File -> Save Animation >
(output_nc,Coor) = LoadData(fileName, ncDims=['lon','lat','pfull'])
Globe = Cart2Spherical(1.0,Coor)
Cont = Contour(ContourBy='ucomp',Isosurfaces=[25])
Shells = SphericalShells( radius,src=Coor,shellValues=[10,1] )
< File -> Save Animation >
(depth_out, depth_coors) = loadData( fileName, ncDims=['lon', 'lat'] )
(depth_out, depth_coors) = LoadData( fileName, ncDims=["lon", "lat"] )
bathy = Make3D( expandVar='Z', expandDir='z', aspRat, logCoords=[] )
(depth_out, depth_coors) = LoadData( fileName, ncDims=['lon', 'lat'] )
bathy = Make3D( expandVar='Z', expandDir='z', aspRat, logCoords=[] )
(o2_out, o2_coors) = LoadData( dataFile, ncDims=['x', 'y', 's'], aspRat )
(depth_out, depth_coor) = LoadData( fileName, ncDims=['lon','lat'] )
bathy = Make3D( expandVar='Z', expandDir='z', aspRat, logCoords=[] )
(o2_out, o2_coor) = LoadData( dataFile, ncDims=['x','y','s'], aspRat )
o2_cont = Contour( o2_coor, ContourBy=['POINTS','o2'], Isosurfaces=[8e-5] )
(depth_out, depth_coor) = LoadData( fileName, ncDims=[['lon','lat']] )
bathy = Make3D( expandVar='Z', expandDir='z', aspRat, logCoords=[] )
(o2_out, o2_coor) = LoadData( dataFile, ncDims=[['x','y','s']], aspRat )
o2_cont = Contour( o2_coor, ContourBy=['POINTS','o2'], Isosurfaces=[8e-5] )
AddGrid( xlevels=[-180,0], ylevels=[-90,0,90], zlevels=[-5000,0],
     bounds=[-280,80,-90,90,-5500,0], AxisNames=['longitude','latitude','depth'])
Exporting visualizations

- Images: png, bmp, tiff, ppm, jpg, pdf, eps, ps, svg
- Models: pov, vrml, x3d, x3db, webgl
- Movies: ogv, avi
- Data: CVS, Paraview data file
Export images, movies, web, ...

Ex 1: Oceanic oxygen content
Ex 2: Sudden Stratospheric Warming
"Anthropogenic CO2 in the Oceans" by Martin Jucker, cims.nyu.edu/~jucker/media.html
"Antarctic Ozone Hole Animation" by Kane Stone, http://vimeo.com/106151359
Summary and Conclusions

pv_atmos is ...

- a Python module that brings GFD specific functionality to Paraview
- freely available on GitHub or PyPi
- a bunch of scripts, so no installation necessary (but possible)

  With Paraview + pv_atmos you can ...

  - see 3D data in 3D (+ time)
  - "hands-on" understanding
  - extract correlations visually (data extraction/coloring)
  - make complex things easier to understand for non-experts
Get the code:

- ParaView: www.paraview.org, Open Source BSD license
- pv_atmos: github.com/mjucker/pv_atmos, Open Source MIT license

Get help and cite:

- [see above]

Thanks for citing above paper when using pv_atmos!