# *Ultra-scale Visualization Climate Data Analysis Tools (UV-CDAT)*

Delivering science and technology solutions to national needs in climate

Department of Energy • Office of Science • Biological and Environmental Research

## DOE BER Climate Research

Kitware

NYU·poly

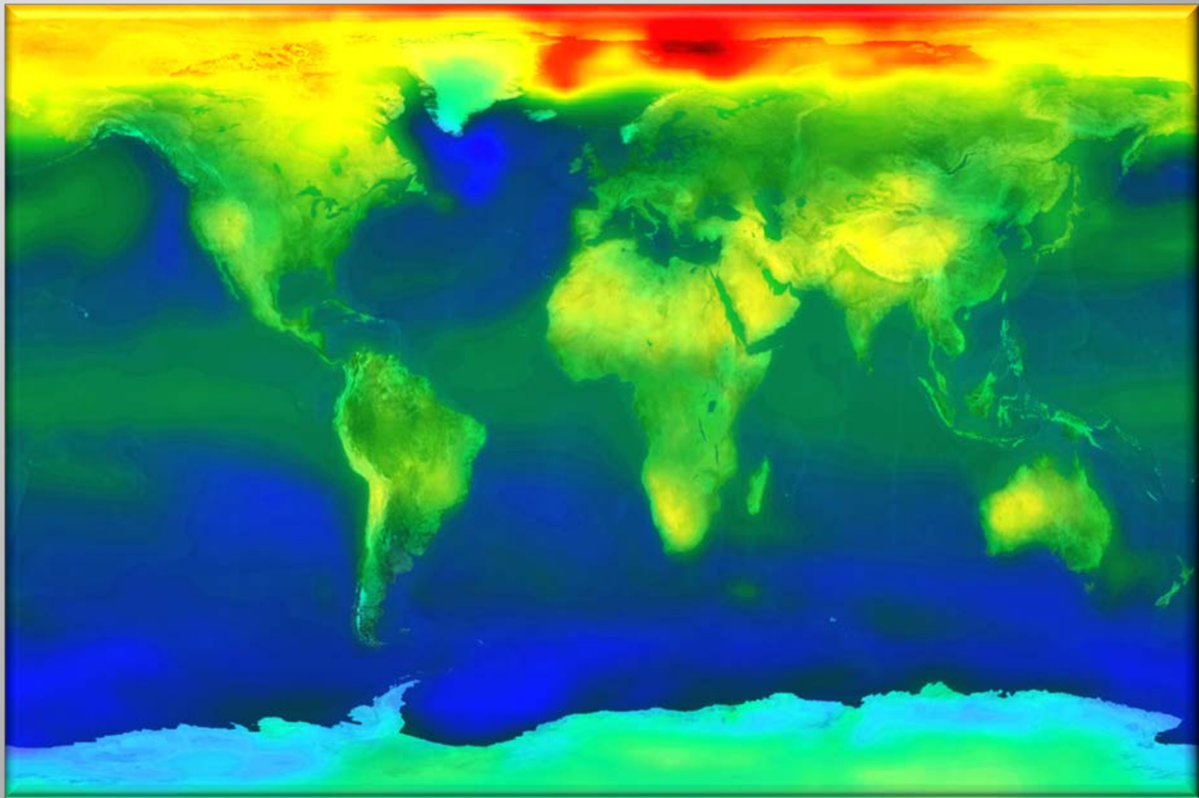
SCI www.sci.utah.edu

Los Alamos NATIONAL LABORATORY EST 1943

NASA

OAK RIDGE National Laboratory

U.S. DEPARTMENT OF ENERGY Office of Science

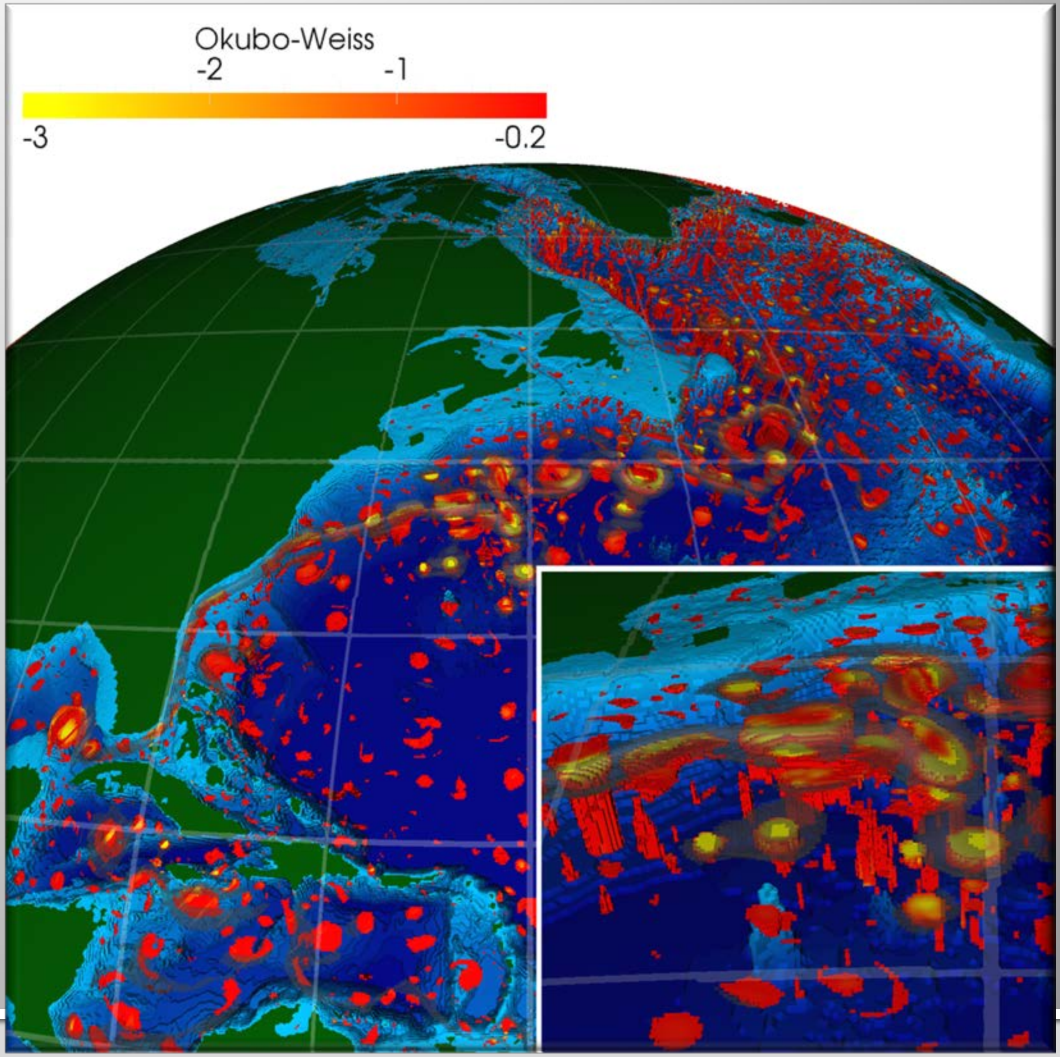


# The Infrastructure Behind the Ultrascale Visualization Climate Data Analysis Tools (UV-CDAT)

LLNL: Dean N. Williams , Charles Doutriaux, and PT Bremer
LANL: James Ahrens, John Patchett, and Sean Williams
ORNL: Galen Shipman, Ross Miller, Chad Steed, and John Harney
Kitware: Berk Geveci, Andy Bauer, Dave Partyka
NASA: Thomas Maxwell
NYU-Poly: Claudio Silva, Emanuele Santos, Huy Vo, and David Koop
University of Utah: Valerio Pascucci

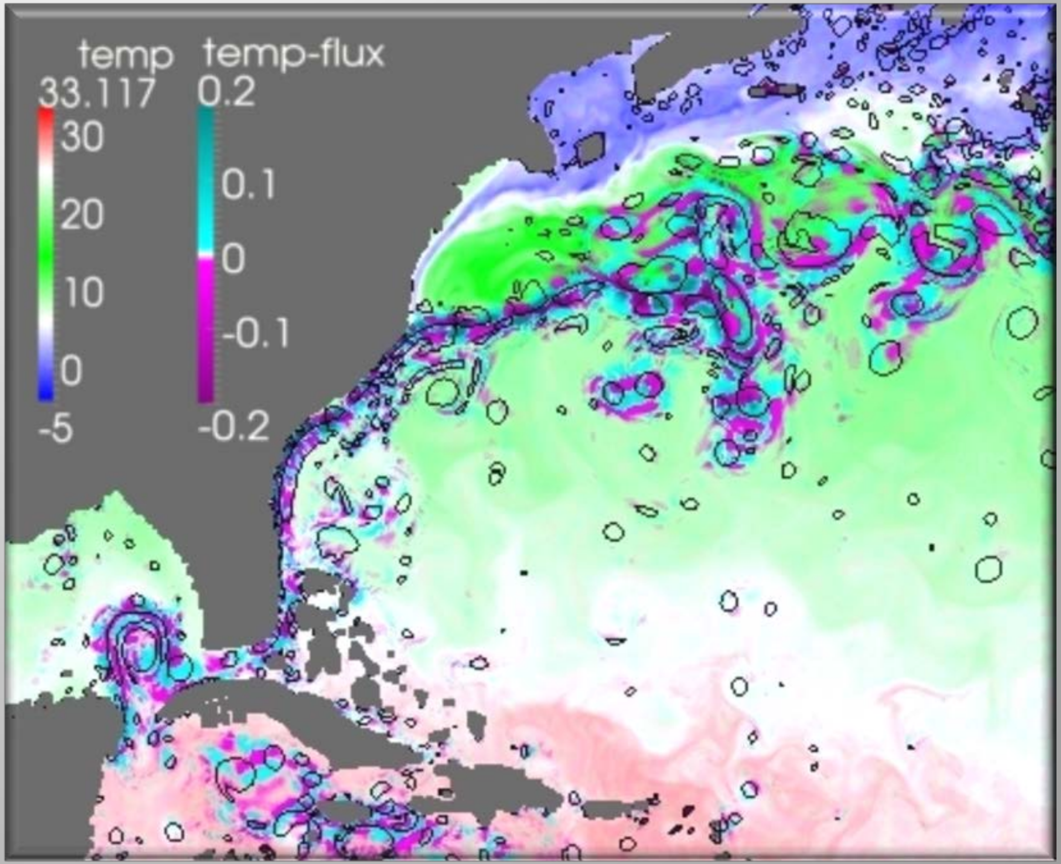
# Goal: Enable Science, e.g. Eddy Studies

- Scientific research in conjunction with Los Alamos National Laboratory ocean modelers
- The work has focused on long-lived, 100 km vortices called mesoscale eddies
- Work has appeared at EuroVis 2011 and will appear at IEEE Vis 2011
- Current work focuses on the role of eddies in regulating temperature and salt concentration in the ocean



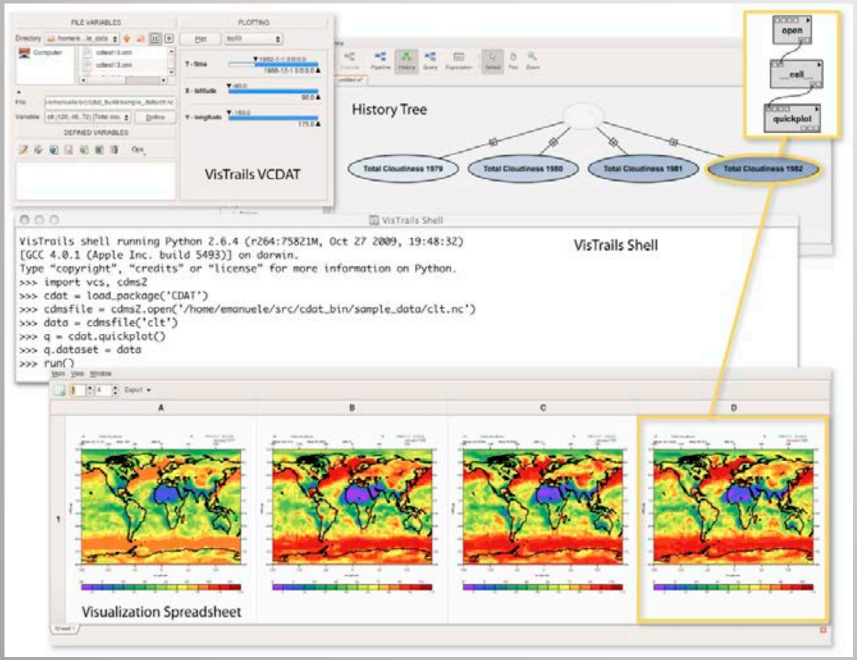
# Eddy-Driven Heat Transport

- Eddies are involved in heat, salt, and nutrient transport, but the process is not well understood
- Using the Okubo-Weiss parameter (contours in black), we compute the temperature flux into (cyan) and out of (magenta) eddies in the Gulf Stream



- Our ongoing analysis indicates complicated "daisy-chained" fluxing between eddies, possibly driving the shape of the Gulf Stream

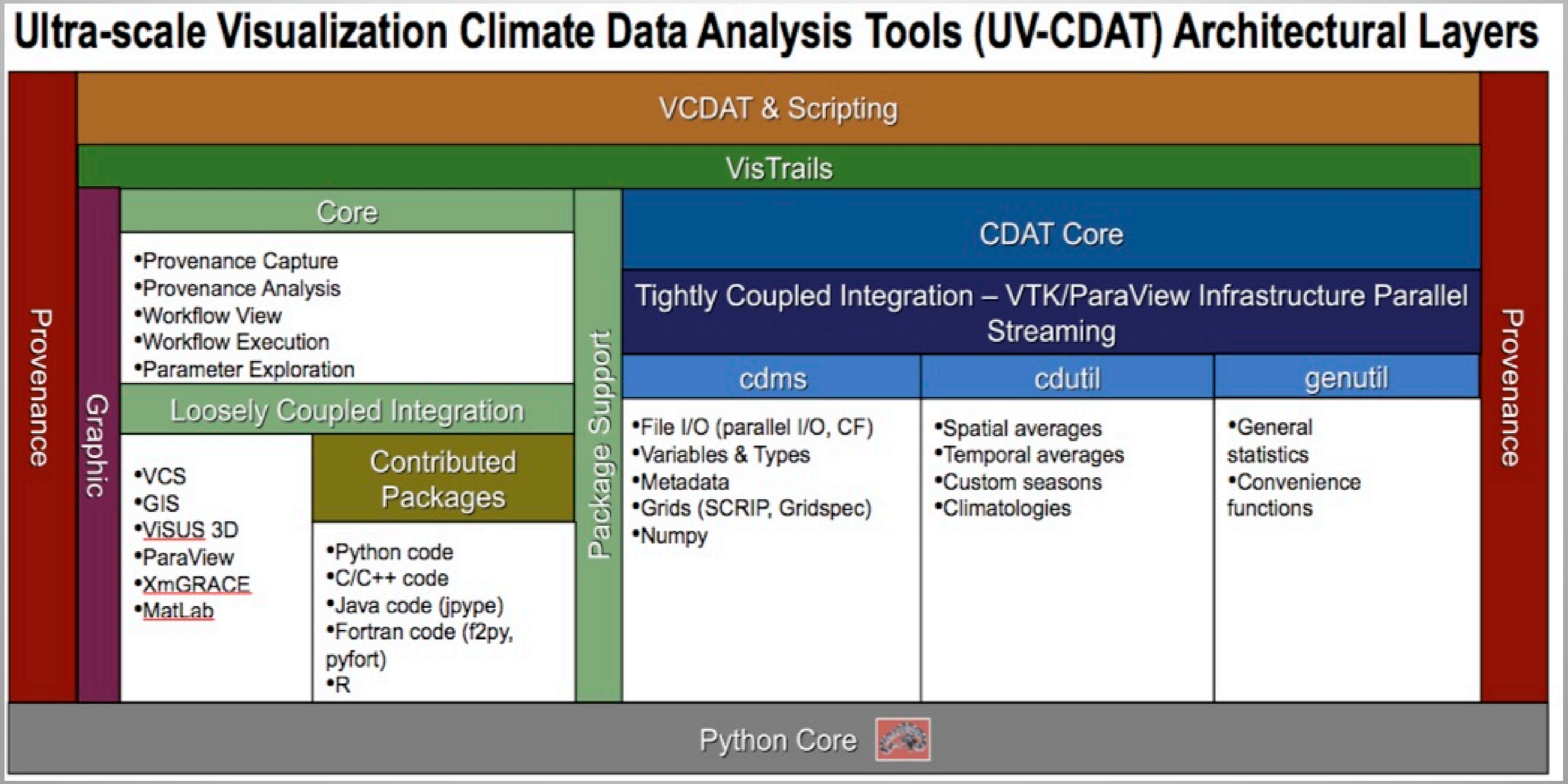
# UV-CDAT Design Requirements



```
VisTrails shell running Python 2.6.4 (r264:75821M, Oct 27 2009, 19:48:32)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin.
Type "copyright", "credits" or "license" for more information on Python.
>>> import vcs, cdms2
>>> cdat = load_package('CDAT')
>>> cdmsfile = cdms2.open('/home/emanuele/src/cdat_bin/sample_data/clt.nc')
>>> data = cdmsfile('clt')
>>> q = cdat.quickplot()
>>> q.dataset = data
>>> run()
```

- Consistent GUI (Qt)
- Multiple OS support
- Python scripting
- Provenance
- Powerful graphics
- Easily extensible
- Loosely and tightly coupled workflows
- Parallelism
- Remote execution

# UV-CDAT Architecture Layers



Ultra-scale Visualization Climate Data Analysis Tools (UV-CDAT) Architectural Layers

VCDAT & Scripting

VisTrails

Provenance

Graphic

Core

- •Provenance Capture
- •Provenance Analysis
- •Workflow View
- •Workflow Execution
- •Parameter Exploration

Loosely Coupled Integration

- •VCS
- •GIS
- •ViSUS 3D
- •ParaView
- •XmGRACE
- •MatLab

Contributed Packages

- •Python code
- •C/C++ code
- •Java code (jpype)
- •Fortran code (f2py, pyfort)
- •R

Package Support

CDAT Core

Tightly Coupled Integration – VTK/ParaView Infrastructure Parallel Streaming

cdms

- •File I/O (parallel I/O, CF)
- •Variables & Types
- •Metadata
- •Grids (SCRIP, Gridspec)
- •Numpy

cdutil

- •Spatial averages
- •Temporal averages
- •Custom seasons
- •Climatologies

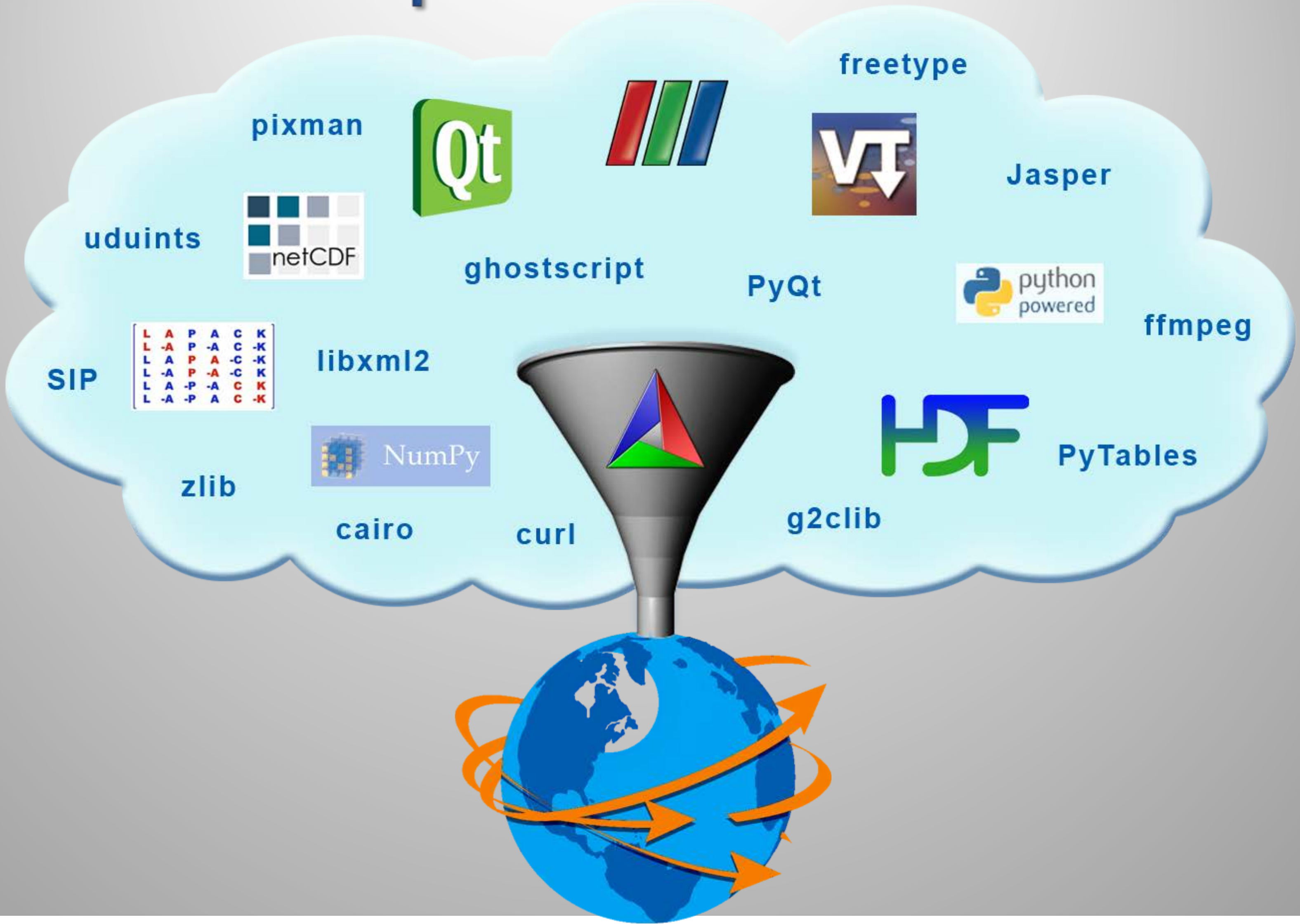
genutil

- •General statistics
- •Convenience functions

Provenance

Python Core

# UV-CDAT Components



freetype

pixman

Jasper

uduints

ghostscript

PyQt

ffmpeg

libxml2

SIP

PyTables

zlib

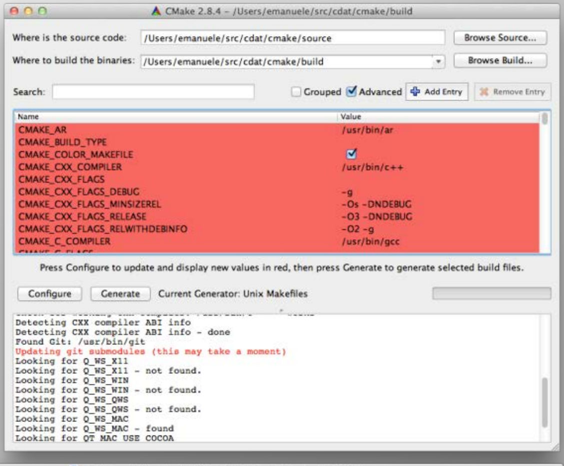
g2clib

cairo

curl

# Using CMake for Building UV-CDAT

- >git clone git://uv-cdat.llnl.gov/uv-cdat.git
- >cmake-gui ../source
- >make –j8
- >./bin/vcdat



CMake 2.8.4 – /Users/emanuele/src/cdat/cmake/build

Where is the source code: /Users/emanuele/src/cdat/cmake/source

Where to build the binaries: /Users/emanuele/src/cdat/cmake/build

Search:

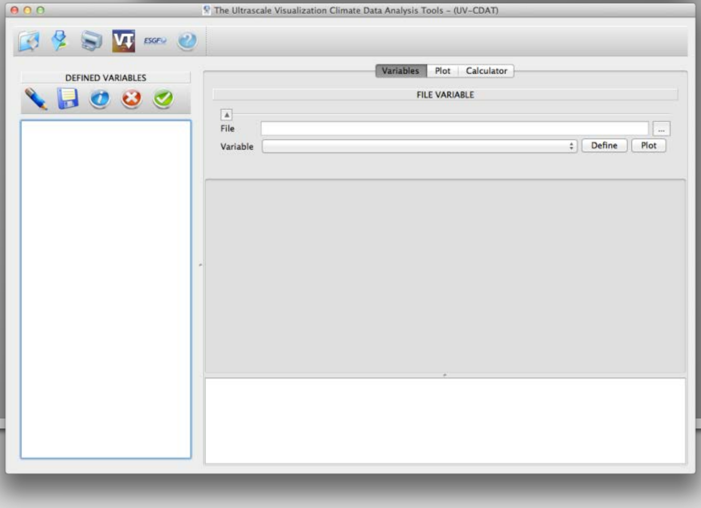
□ Grouped ☑ Advanced

| Name | Value |
|---|---|
| CMAKE_AR | /usr/bin/ar |
| CMAKE_BUILD_TYPE | |
| CMAKE_COLOR_MAKEFILE | ☑ |
| CMAKE_CXX_COMPILER | /usr/bin/c++ |
| CMAKE_CXX_FLAGS | |
| CMAKE_CXX_FLAGS_DEBUG | -g |
| CMAKE_CXX_FLAGS_MINSIZEREL | -Os -DNDEBUG |
| CMAKE_CXX_FLAGS_RELEASE | -O3 -DNDEBUG |
| CMAKE_CXX_FLAGS_RELWITHDEBINFO | -O2 -g |
| CMAKE_C_COMPILER | /usr/bin/gcc |

Press Configure to update and display new values in red, then press Generate to generate selected build files.

Current Generator: Unix Makefiles

```
Detecting CXX compiler ABI info
Detecting CXX compiler ABI info - done
Found Git: /usr/bin/git
Updating git submodules (this may take a moment)
Looking for Q_WS_X11
Looking for Q_WS_X11 - not found.
Looking for Q_WS_WIN
Looking for Q_WS_WIN - not found.
Looking for Q_WS_QWS
Looking for Q_WS_QWS - not found.
Looking for Q_WS_MAC
Looking for Q_WS_MAC - found
Looking for QT_MAC_USE_COCOA
```



The Ultrascale Visualization Climate Data Analysis Tools – (UV-CDAT)

# Using CMake for Building UV-CDAT

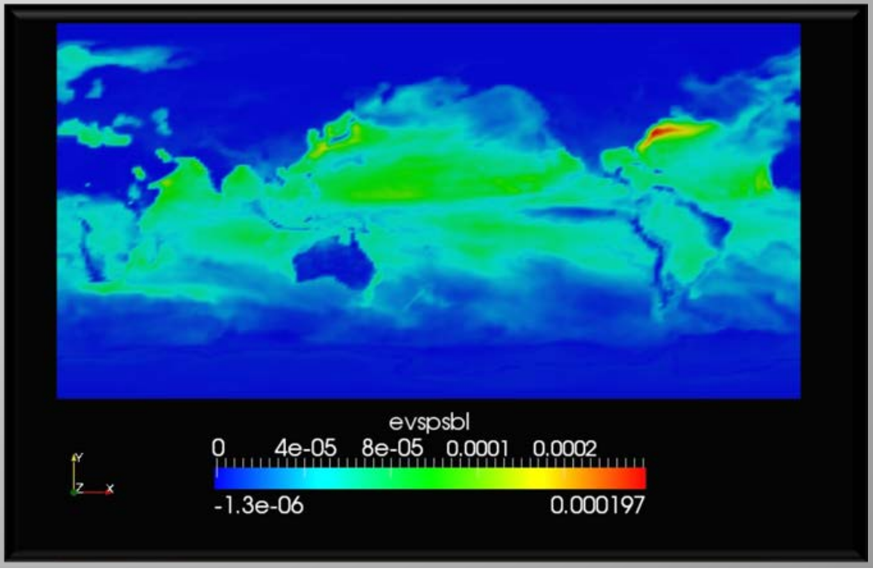
- Provides transparent builds on Linux and Mac
- Simple three step build process
  - Compiles and installs over 40 packages
  - Packages consist of over 7 million lines of C/C++/FORTRAN/Python code
- Successfully transitioned to CMake Build System.
- Improves productivity by removing build system as a hurdle to development.

# Spatio-Temporal Parallelism

- Decompose data on time and space boundaries
  - Align the problem to existing parallel hardware
  - improves overall processing time
- Validated on Jaguarpf with Ocean Data
- Required engineered changes to ParaView
  - Added an MPI Communicator structure
- Outcome – Climate scientists can produce visualizations of time series much more quickly

# Integration into UV-CDAT tool

- Generating new use-cases for Ocean analysis
  - Not everything is easily parallelized (eddies)
  - Implementing many as ParaView filters
- Spatio-Temporal Implementation in ParaView
  - a key component of UV-CDAT

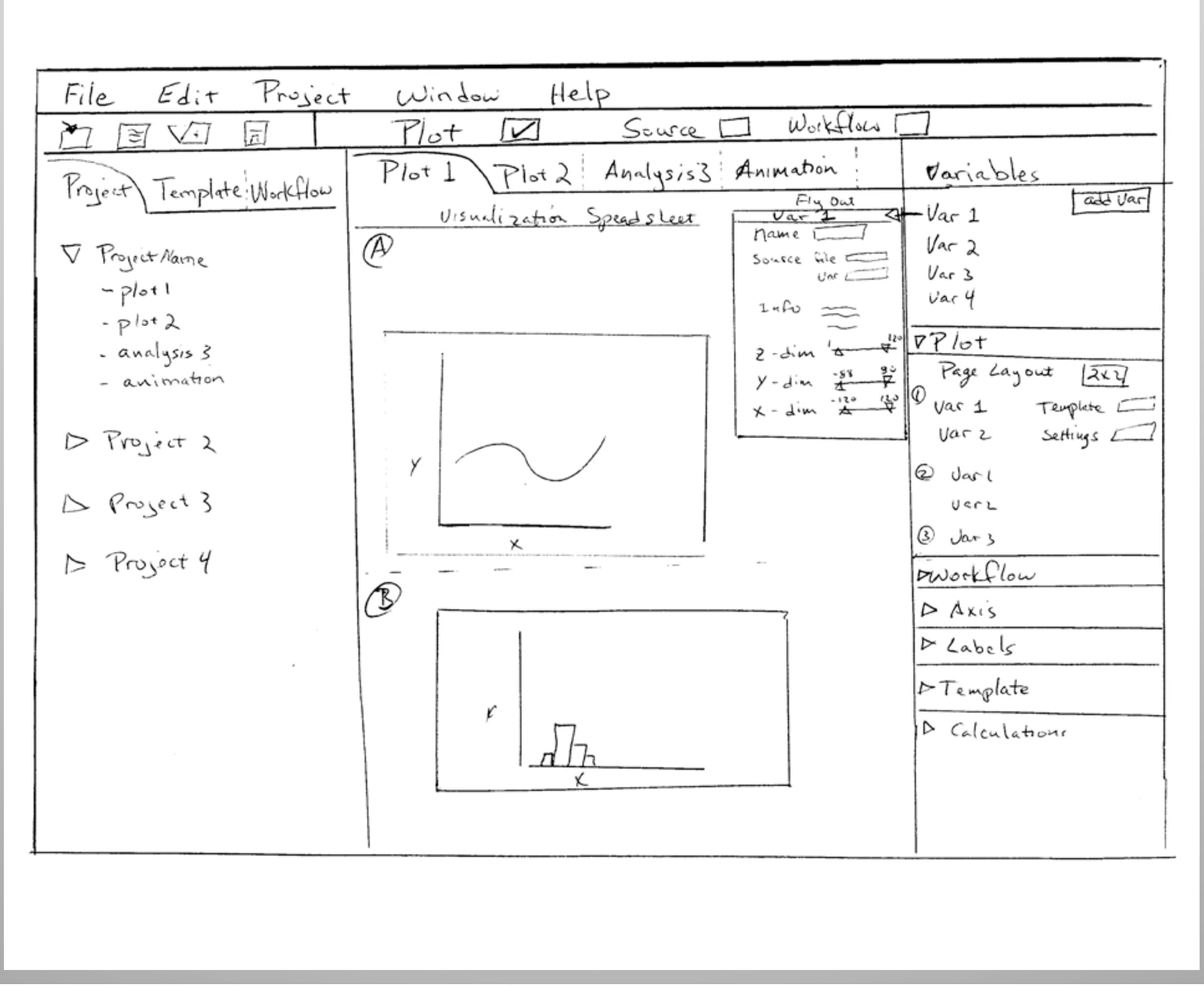


# Temporal Parallelism:

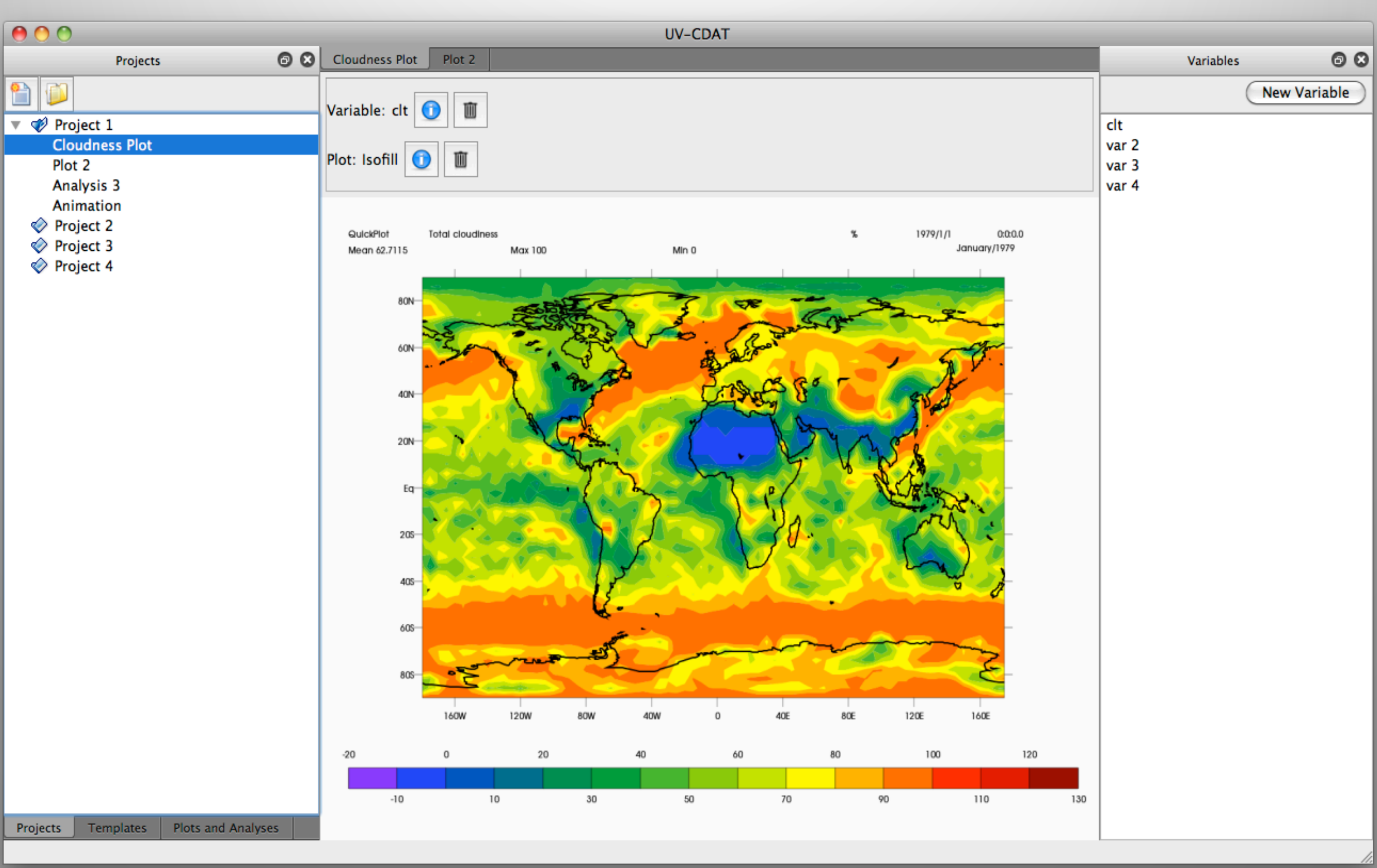
## Challenges to I/O

- Many climate models output a separate file for each time step.  In order to visualize how a particular variable changes over time, each individual file must be opened and processed.
- Added new classes to VTK to execute a single visualization pipeline on multiple files simultaneously
- An 'embarrassingly parallel' task, but it offered the opportunity for enormous speedup.
  - When running on Jaguar, could render dozens of images simultaneously
  - Scalability was  limited by the filesystem performance.

# UV-CDAT Integrated GUI



# Current: Prototyping NEW GUI

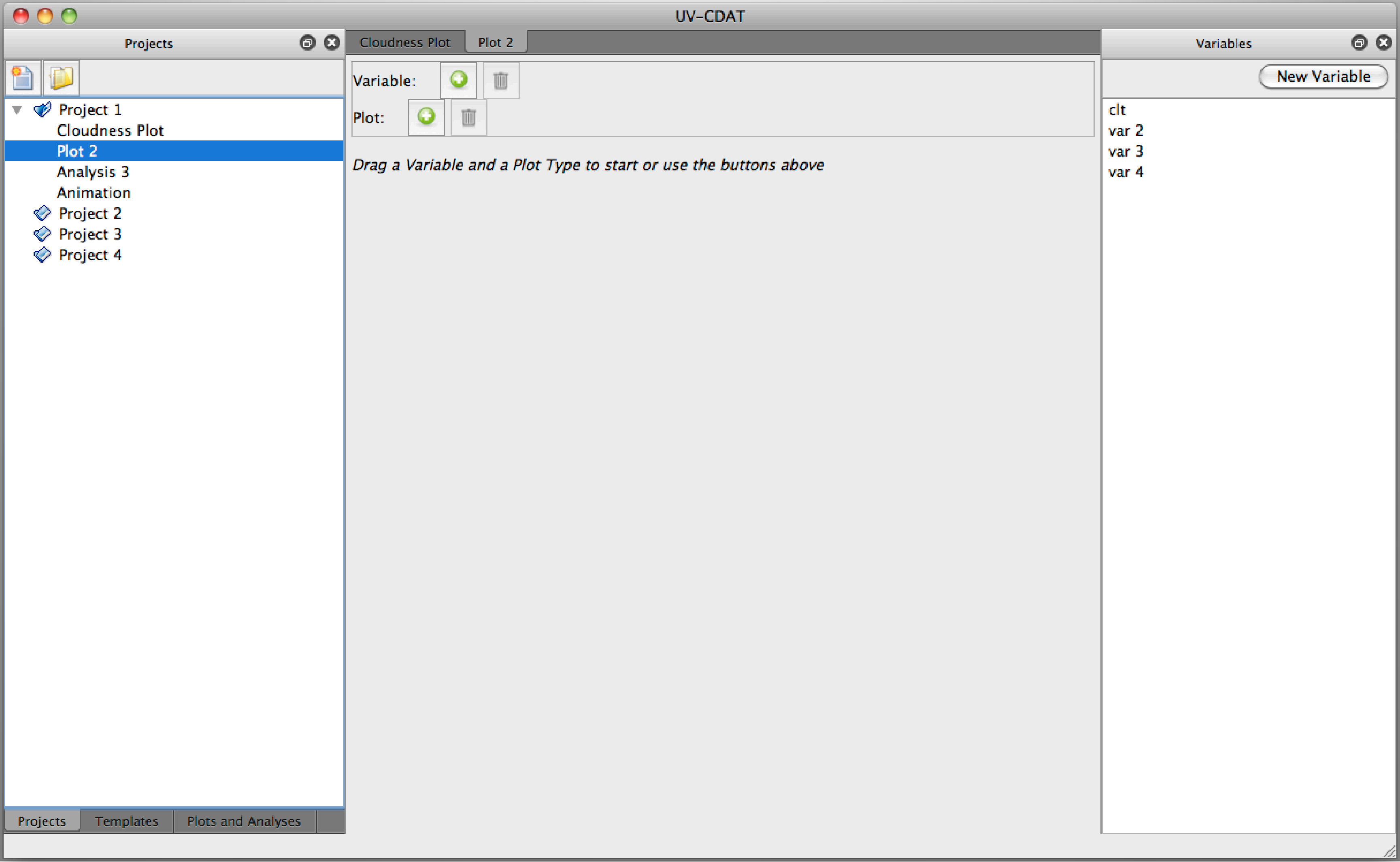


Variable: clt

Plot: Isofill

# Integrated UV-CDAT GUI:

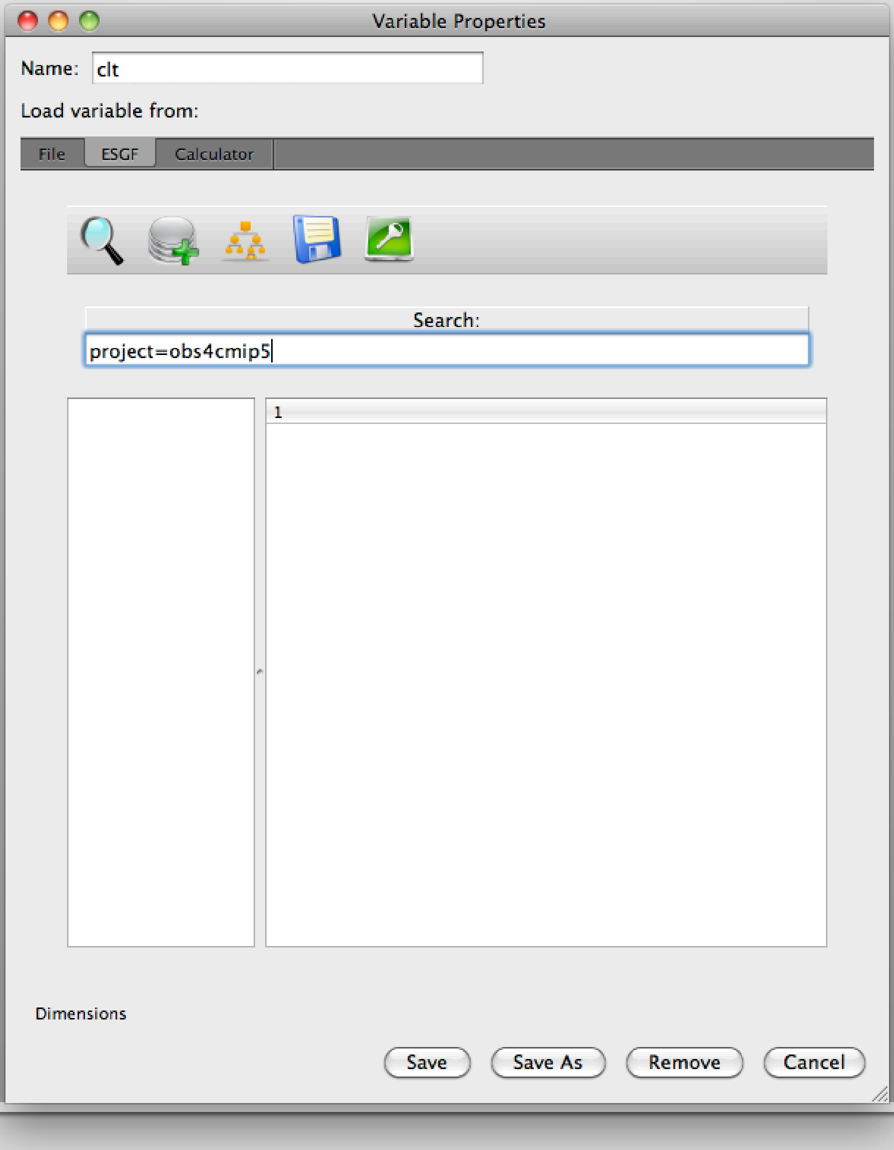
## Project, Plot, and Variables View



*Drag a Variable and a Plot Type to start or use the buttons above*

# Integrated UV-CDAT GUI:

## Earth System Grid Federation (ESGF) Access



Name: clt

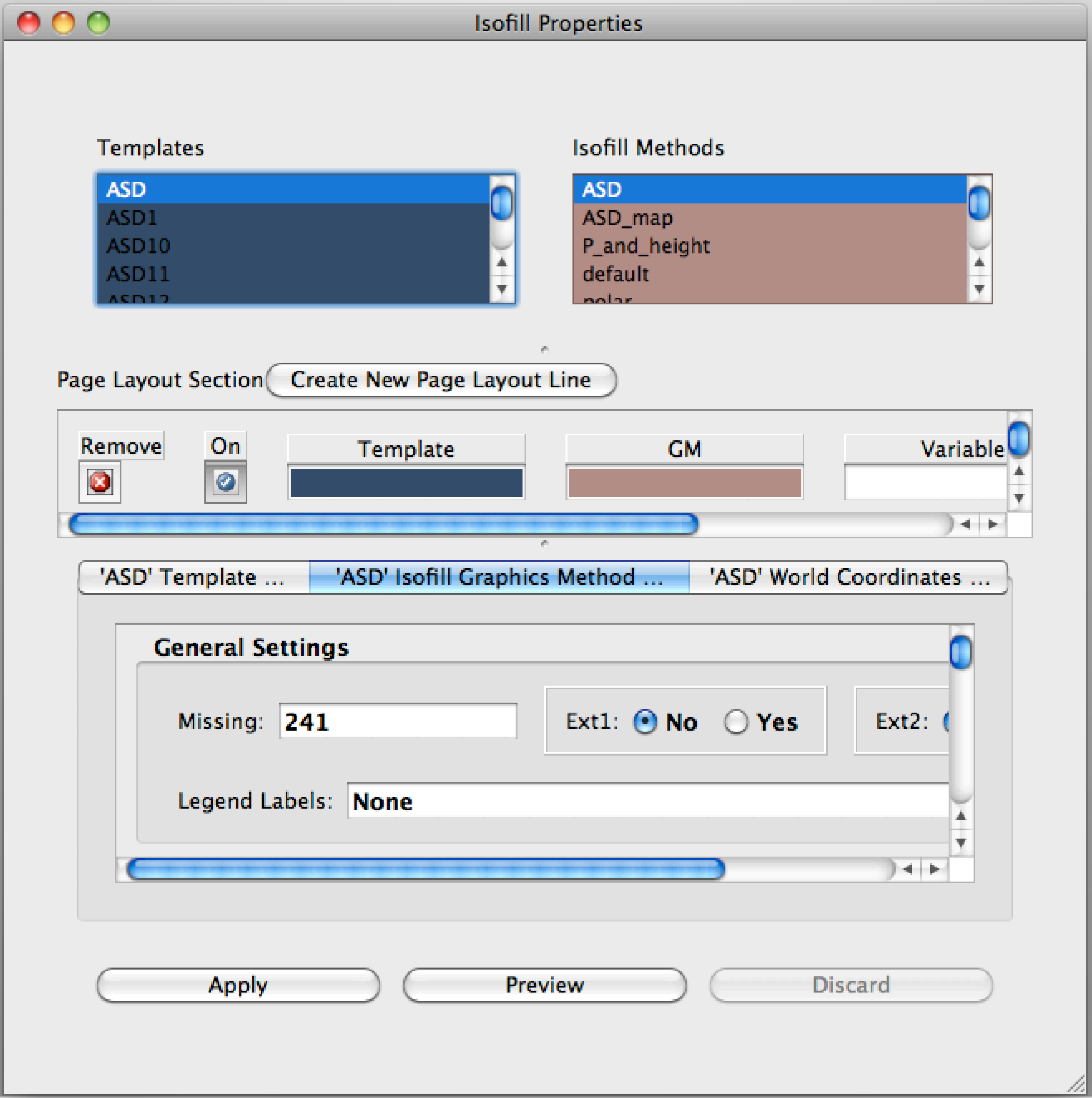
Load variable from:

Search:

project=obs4cmip5

# Integrated UV-CDAT GUI:

## Isofill Properties View



Missing: 241

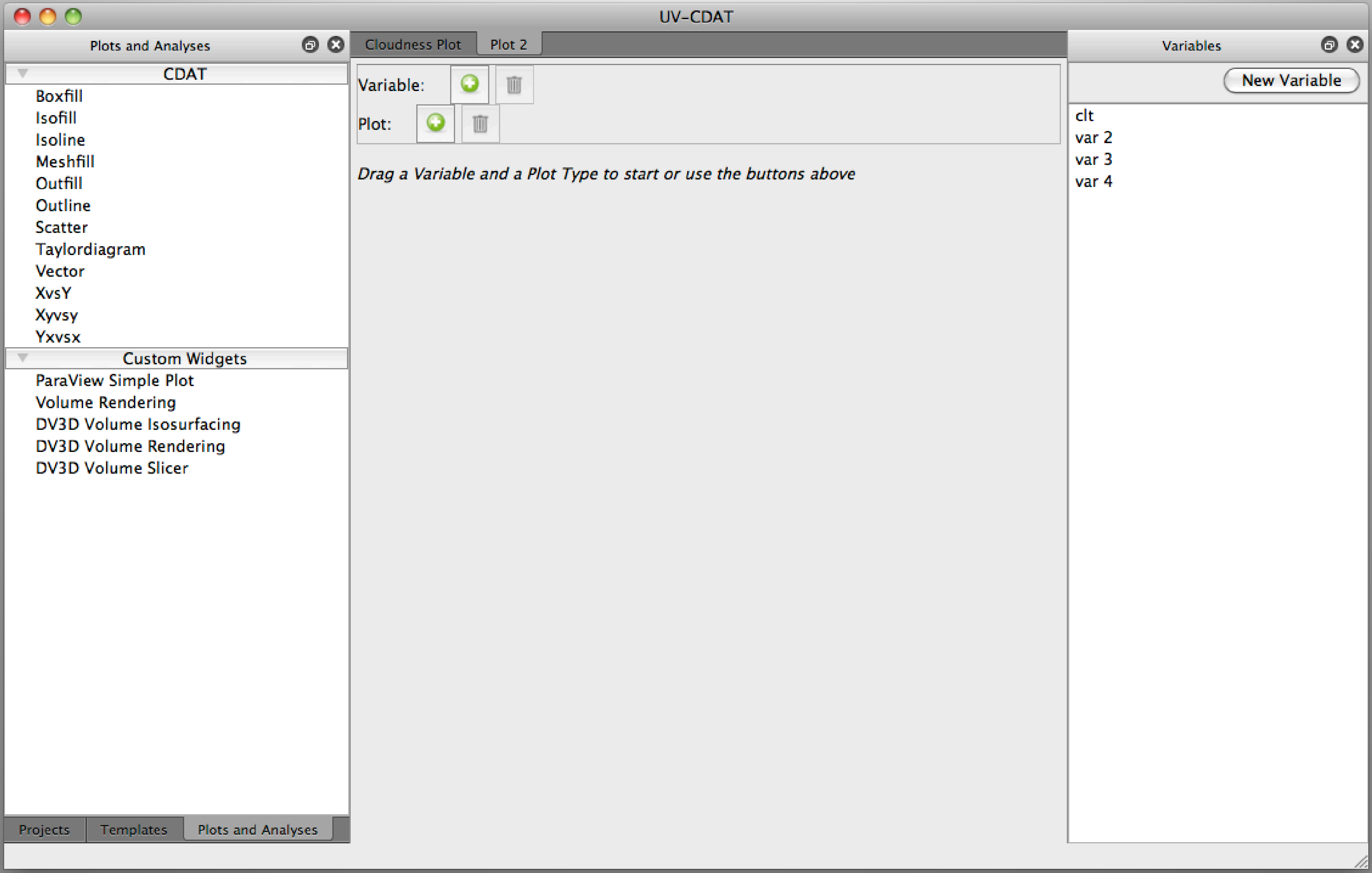
Ext1: ● No ○ Yes

Ext2:

Legend Labels: None

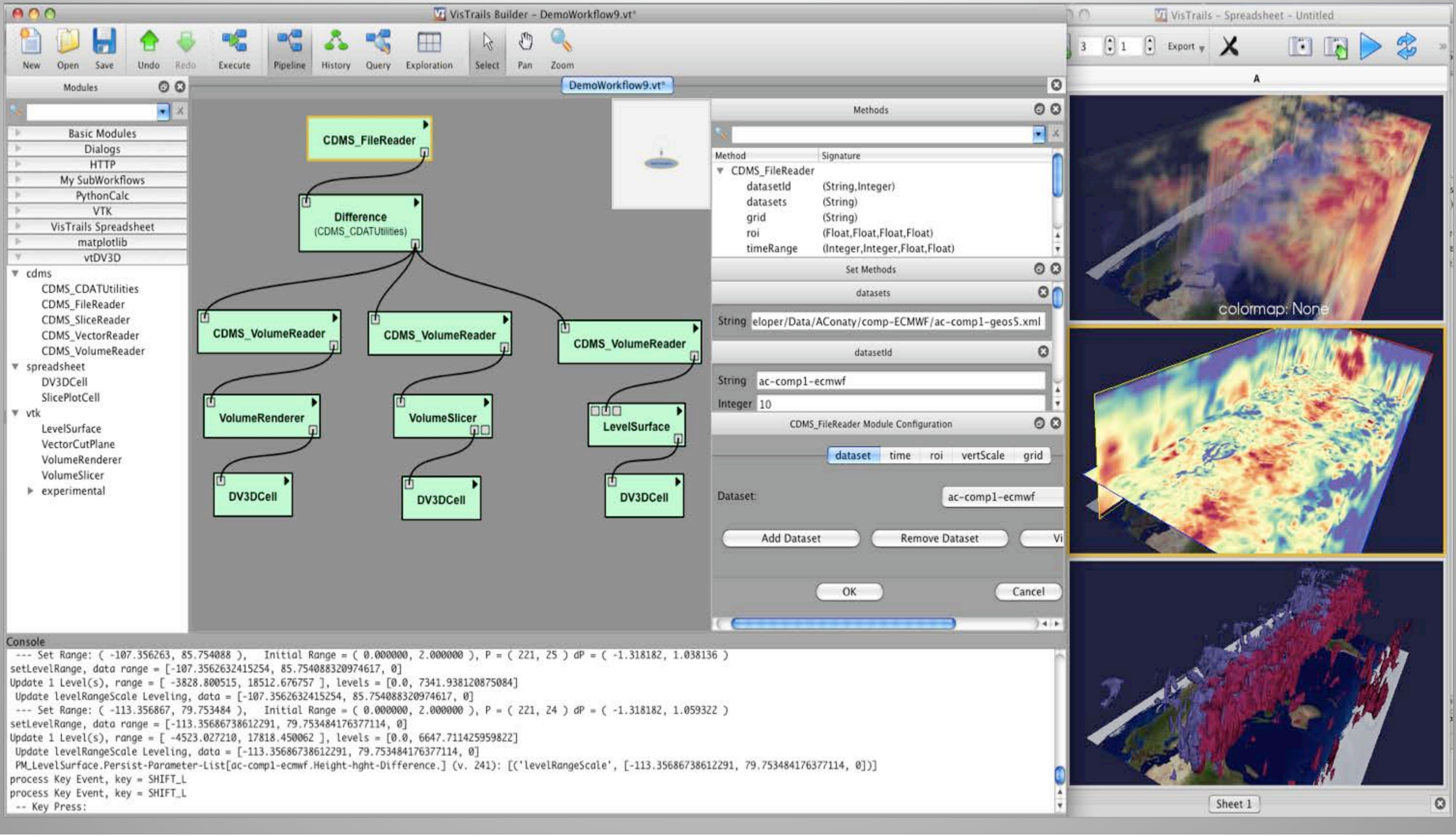
# Integrated UV-CDAT GUI:

## Plot and Analysis View



*Drag a Variable and a Plot Type to start or use the buttons above*

# Extensibility, e.g., vtDV3D



| Method | Signature |
|---|---|
| ▼ CDMS_FileReader | |
| datasetId | (String,Integer) |
| datasets | (String) |
| grid | (String) |
| roi | (Float,Float,Float,Float) |
| timeRange | (Integer,Integer,Float,Float) |

datasets

String: eloper/Data/AConaty/comp-ECMWF/ac-comp1-geos5.xml

datasetId

String: ac-comp1-ecmwf

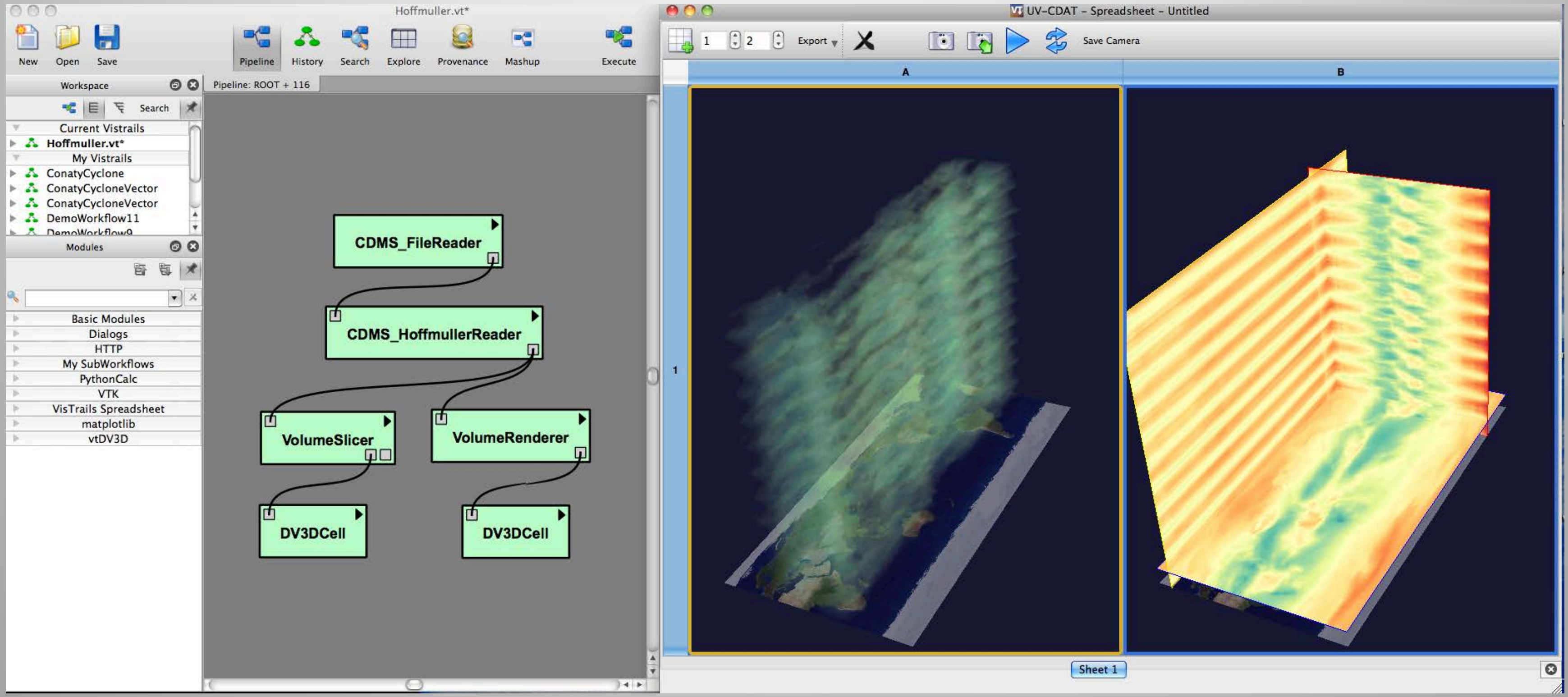
Integer: 10

Dataset: ac-comp1-ecmwf

Console

```
 --- Set Range: ( -107.356263, 85.754088 ),   Initial Range = ( 0.000000, 2.000000 ), P = ( 221, 25 ) dP = ( -1.318182, 1.038136 )
setLevelRange, data range = [-107.3562632415254, 85.754088320974617, 0]
Update 1 Level(s), range = [ -3828.800515, 18512.676757 ], levels = [0.0, 7341.938120875084]
 Update levelRangeScale Leveling, data = [-107.3562632415254, 85.754088320974617, 0]
 --- Set Range: ( -113.356867, 79.753484 ),   Initial Range = ( 0.000000, 2.000000 ), P = ( 221, 24 ) dP = ( -1.318182, 1.059322 )
setLevelRange, data range = [-113.35686738612291, 79.753484176377114, 0]
Update 1 Level(s), range = [ -4523.027210, 17818.450062 ], levels = [0.0, 6647.711425959822]
 Update levelRangeScale Leveling, data = [-113.35686738612291, 79.753484176377114, 0]
 PM_LevelSurface.Persist-Parameter-List[ac-comp1-ecmwf.Height-hght-Difference.] (v. 241): [('levelRangeScale', [-113.35686738612291, 79.753484176377114, 0])]
process Key Event, key = SHIFT_L
process Key Event, key = SHIFT_L
 -- Key Press:
```

# 3D Hoffmuller (lat-long-time) plots



# Stay tuned…

- UV-CDAT (alpha) is very close to being released
- You can start creating your own analysis code right now
- WE WANT TO HEAR FROM YOU!!!