Argonne NATIONAL LABORATORY

ParVis and MCT update

Robert Jacob

CESM Workshop Software Engineering Working Group.

June 20, 2013

Breckenridge, CO

U.S. DEPARTMENT OF ENERGY | Office of Science

NCAR NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

Sandia National Laboratories

Pacific Northwest NATIONAL LABORATORY

UCDAVIS UNIVERSITY OF CALIFORNIA

U.S. DEPARTMENT OF ENERGY

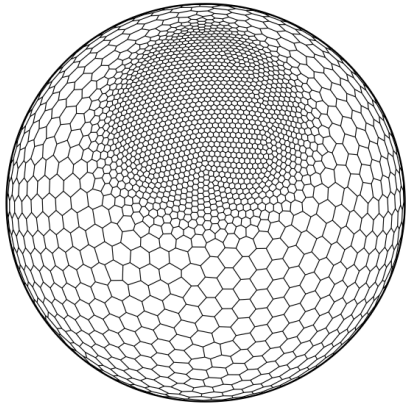
ParVis Team

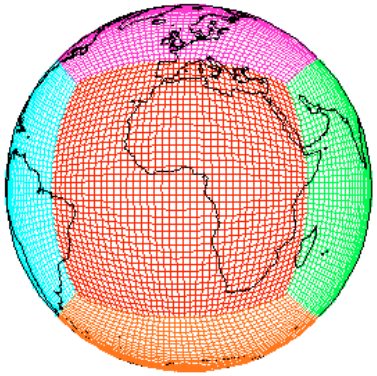
- At Argonne:
  - Rob Jacob, Xiabing Xu, Jayesh Krishna, Sheri Mickelson, Tim Tautges, Mike Wilde, Rob Ross, Rob Latham, Jay Larson, Mark Hereld, Ian Foster
- At Sandia:
  - Pavel Bochen, Kara Peterson, Dennis Ridzal, Mark Taylor
- At PNNL
  - Karen Schuchardt, Jeff Daly, Jian Yin
- At NCAR
  - Don Middleton, Mary Haley, Dave Brown, Rick Brownrigg, Dennis Shea, Wei Huang, Mariana Vertenstein
- At UC-Davis
  - Kwan-Lu Ma, Jinrong Xie

Supported by the Regional and Global Climate Modeling Program
of the Office of Biological and Environmental Research
of the U.S. Department of Energy's Office of Science

Motivation

- DOE/NSF Community Atmosphere Model (CAM) at 0.125 degrees
  - Single 3D variable: 616 MB
  - Single 2D variable: 25 MB
  - Single history file: 24 GB
  - 1 year of monthly output: 288 GB
  - 100 years of monthly: 28.8 TB
    - CMIP3: 35TB

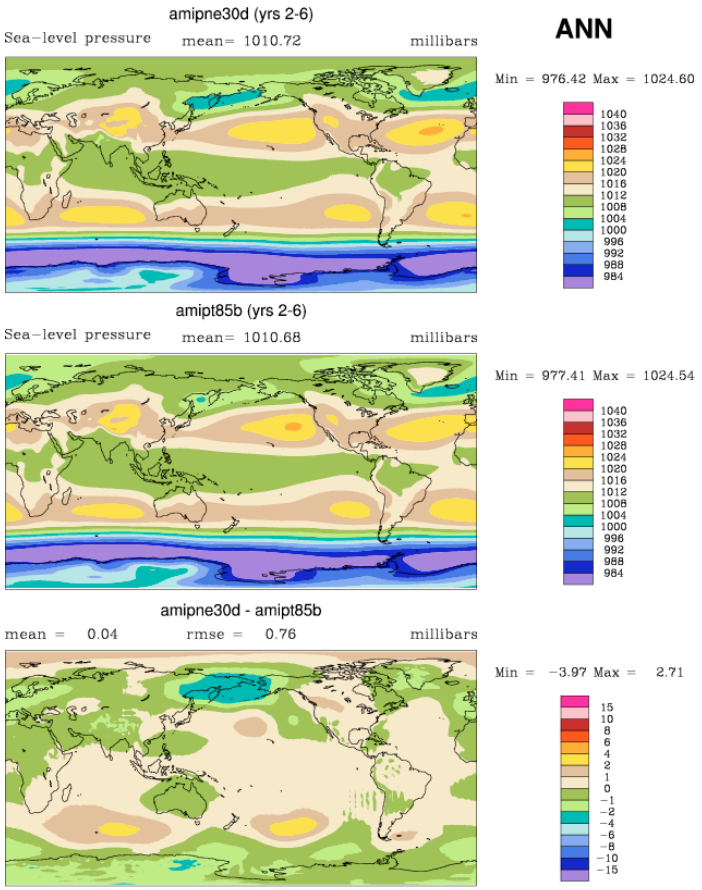


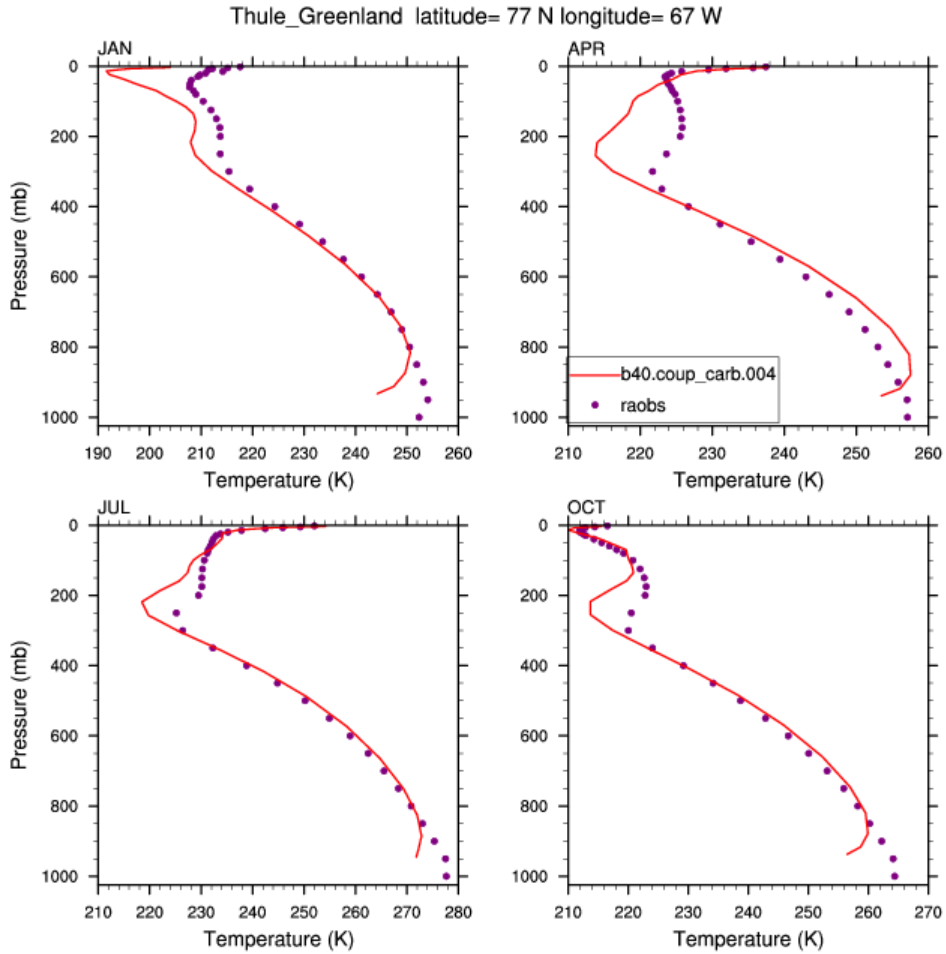


Output data getting larger

Grids no longer rectangular

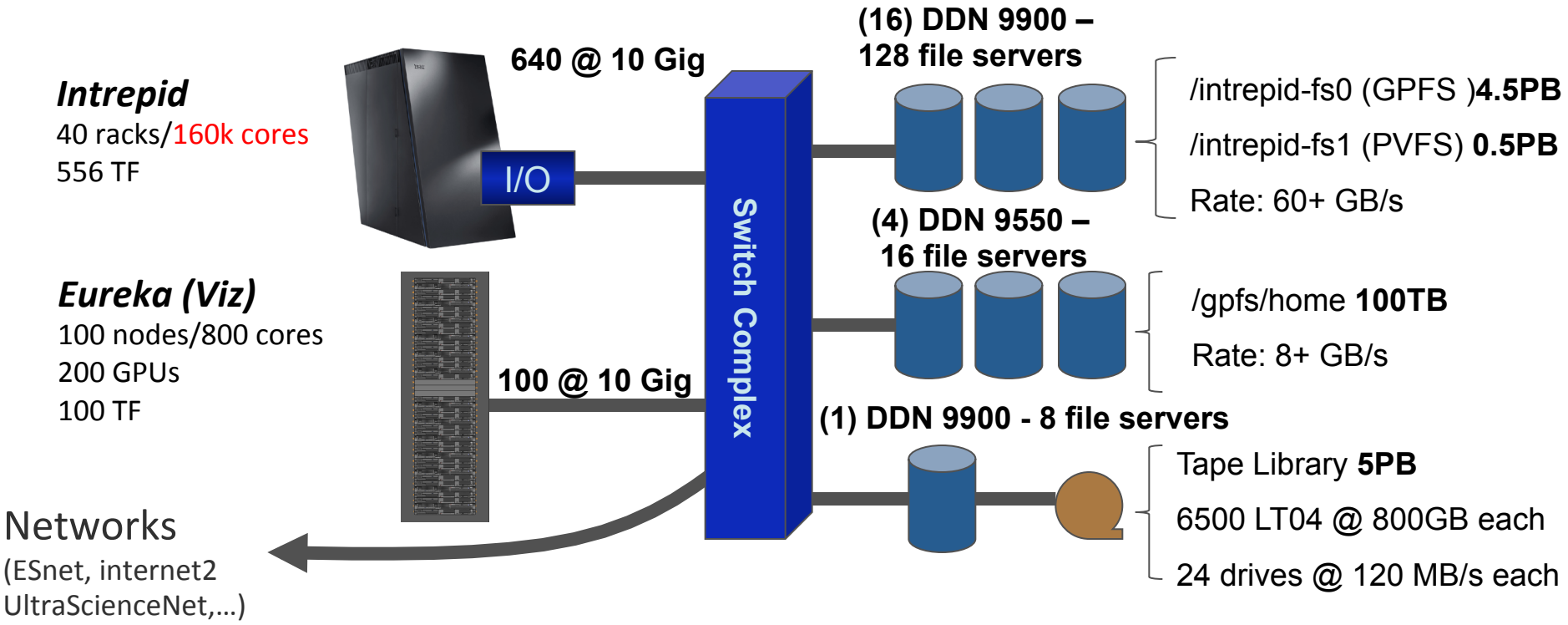
Insight about climate comes mostly from computationally undemanding (to plot) 2D and 1D figures.





Why? The atmosphere and ocean have a small aspect ratio; 10,000 km vs. 10 km.

Lots of processing power is available to analyze large Earth science data sets



Argonne Leadership Computing Facility

Existing Data Analysis and Visualization (DAV) tools have not kept up with growth in data sizes and grid types.

- NCAR Command Language (NCL)
- Climate Data Analysis Tools (CDAT)
- Grid Analysis and Display System (GrADS)
- Ferret

} No or limited parallelism. Assume lat-lon grids

parVis

(Parallel Analysis Tools and New Visualization Techniques for Ultra-Large Climate Data Sets)

- Speed up data analysis and visualization through data- and task-parallelism mostly in the post-processing phase.
- AND natively support multiple grids
- AND reconstruct the discretization used in the models.

Three main products from ParVis

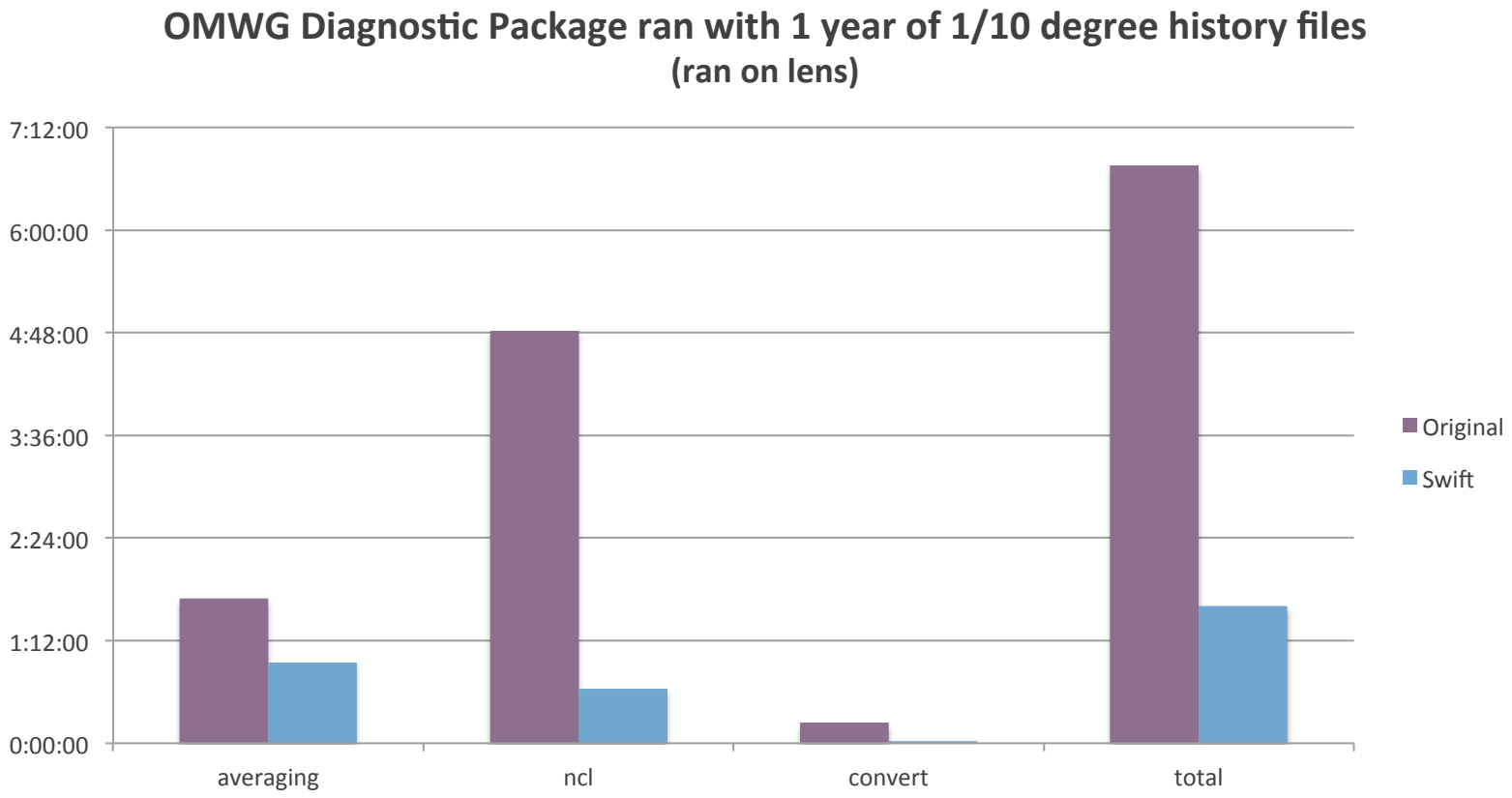
- ParGAL – A new DOE library for performing climate analysis functions with data parallelism
- ParNCL – An implementation of NCL using ParGAL
- Task-parallel diagnostic scripts with Swift

Task-parallel versions of standard diagnostic scripts using *Swift*

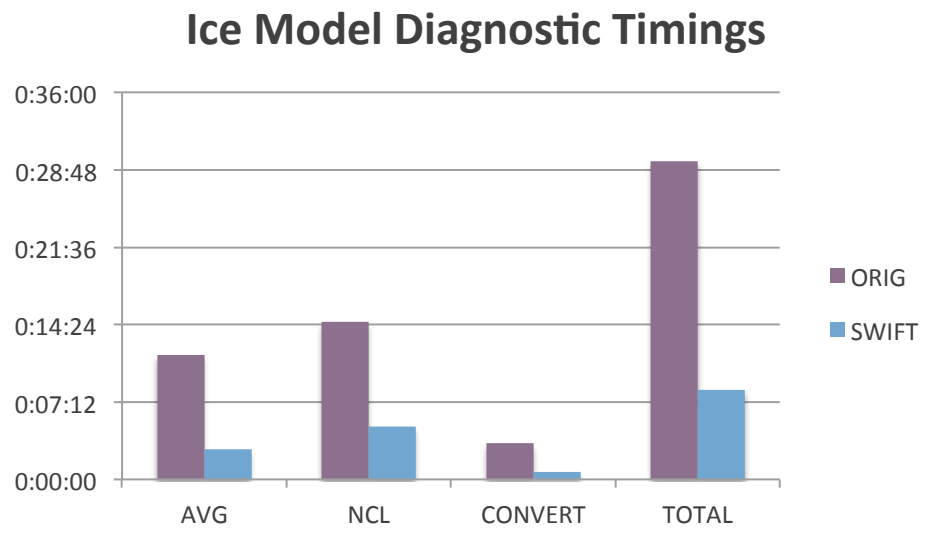
- **Not created by ParVis but we're using it.**
- **Swift is a parallel scripting system for Grids and clusters**
  - for loosely-coupled applications - application and utility programs linked by exchanging files
- **Swift is easy to write**: simple high-level C-like functional language
  - *Small Swift scripts can do large-scale work*
- **Swift is easy to run**: a Java application. Just need a Java interpreter installed.
- **Swift is fast**: Karajan provides Swift a powerful, efficient, scalable and flexible execution engine.
  - *Scaling close to 1M tasks – .5M in live science work, and growing*
- Swift **usage** is growing:
  - *applications in neuroscience, proteomics, molecular dynamics, biochemistry, economics, statistics, and more.*

Last Year: Swift versions of AMWG and OMWG

- Both released to community
- Tutorial for swift-based scripts given



Swift versions of land model and sea-ice model diagnostics now available!



Comparison of 2 20-year 2 degree data sets

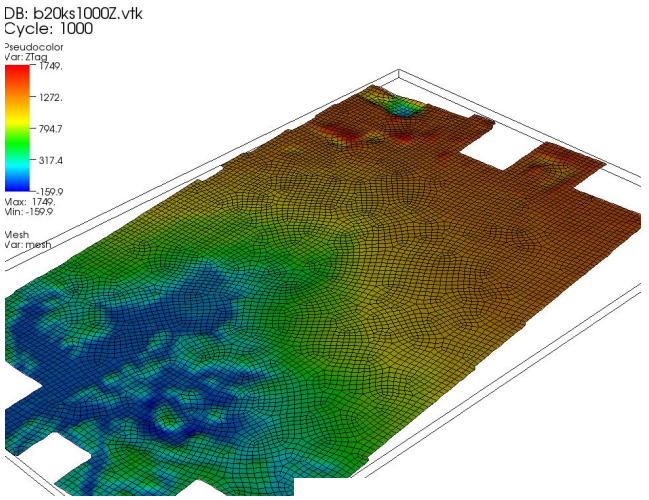
ParGAL - Parallel Gridded Analysis Library

- The main product from ParVis.
  - Data parallel C++ Library
  - Parallel versions of typical climate analysis functions.
  - Reads and computes on structured and unstructured numerical grids

- Built upon existing tools
  - MOAB
  - Intrepid
  - PnetCDF
  - MPI

Mesh-Oriented datABase (MOAB)

- MOAB is a library for representing structured, unstructured, and polyhedral meshes, and field data on those meshes
- Uses array-based storage, for memory efficiency



Jakobshavn ice bed (in VisIt/MOAB)

Intrepid ***INteroperable Tools for Rapid dEveloPment of compatIble Discretizations***

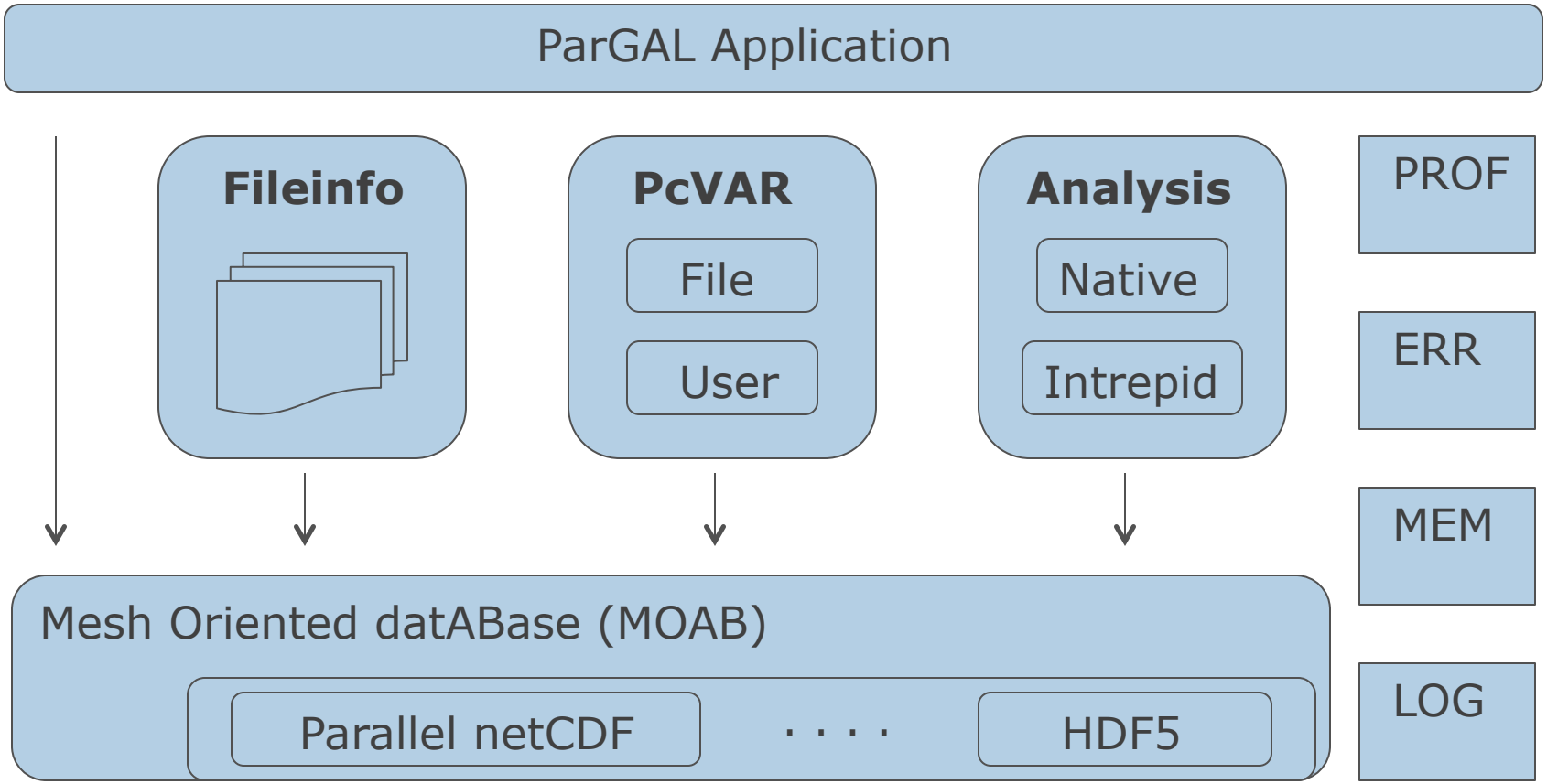
A Trilinos package for compatible discretizations: a suite of stateless tools for

- Cell topology, geometry and integration
- Discrete spaces, operators and functionals on cell worksets
- Up to order 10 *H(grad)*, *H(curl)* and *H(div)* FE bases on Quad, Triangle, Tetrahedron, Hexahedron, and Wedge cell topologies

PNetCDF: NetCDF output with MPI-IO

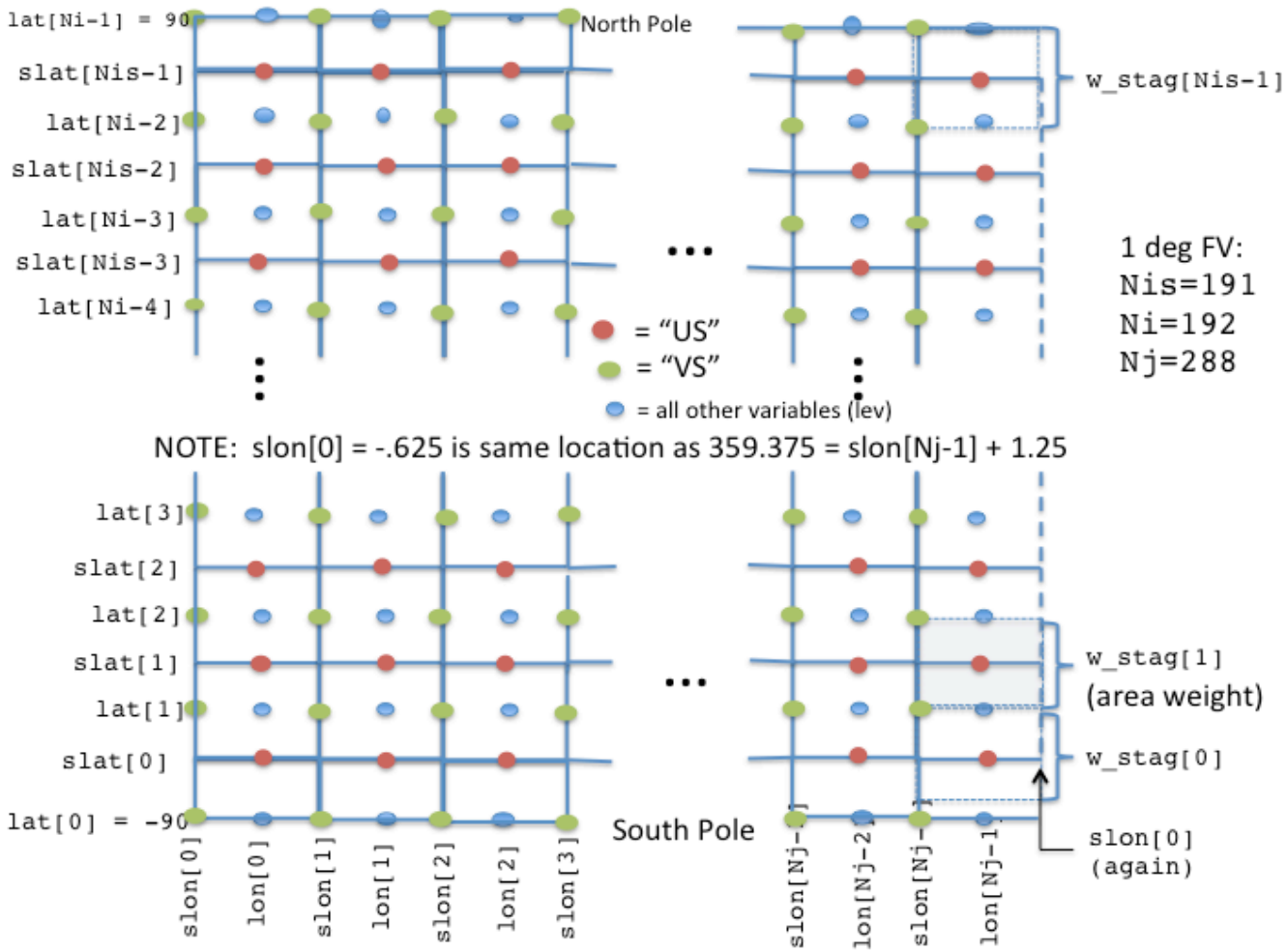
- Based on NetCDF
- Final output is indistinguishable from serial NetCDF file
- Noncontiguous I/O in memory using MPI datatypes
- Noncontiguous I/O in file using sub-arrays
- Collective I/O

ParGAL Architecture



ParGAL represents discretizations as they are in the model. Algorithms are aware of grid location of data.

CAM's Finite Volume Grid



Note: Community should decide on grid metadata standards ASAP

Consequences of lack of metadata. Which grid?

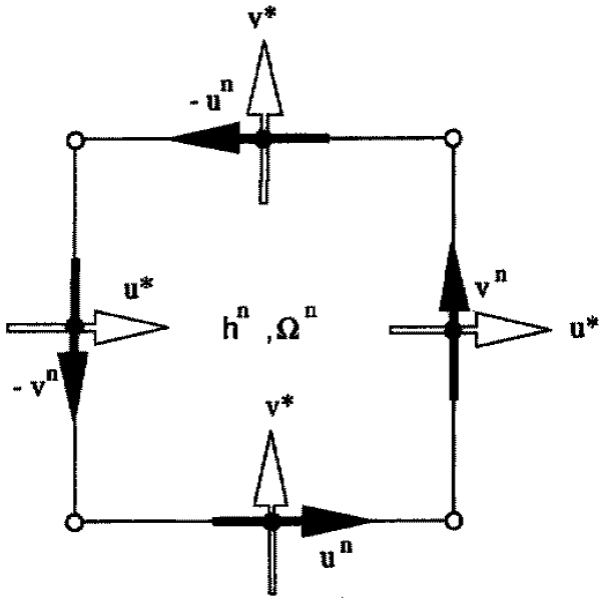
- CAM-FV, CAM-SE, and CAM-EUL all have in their history file….

```
// global attributes:
                :Conventions = "CF-1.0" ;
                :source = "CAM" ;
```

```
if dimension names "lon","lat","slat" and "slon" all exist then
        Its the CAM-FV grid
else if global attribute "np" exists then
        Its the CAM-SE (HOMME) grid
else if dimension names "lon" and "lat" exist then
        Its the CAM-EUL grid
else
        Unknown CAM grid, error.
```

Consequences of lack of metadata. Where are the variables located?

Suppose you know its from CAM-FV…



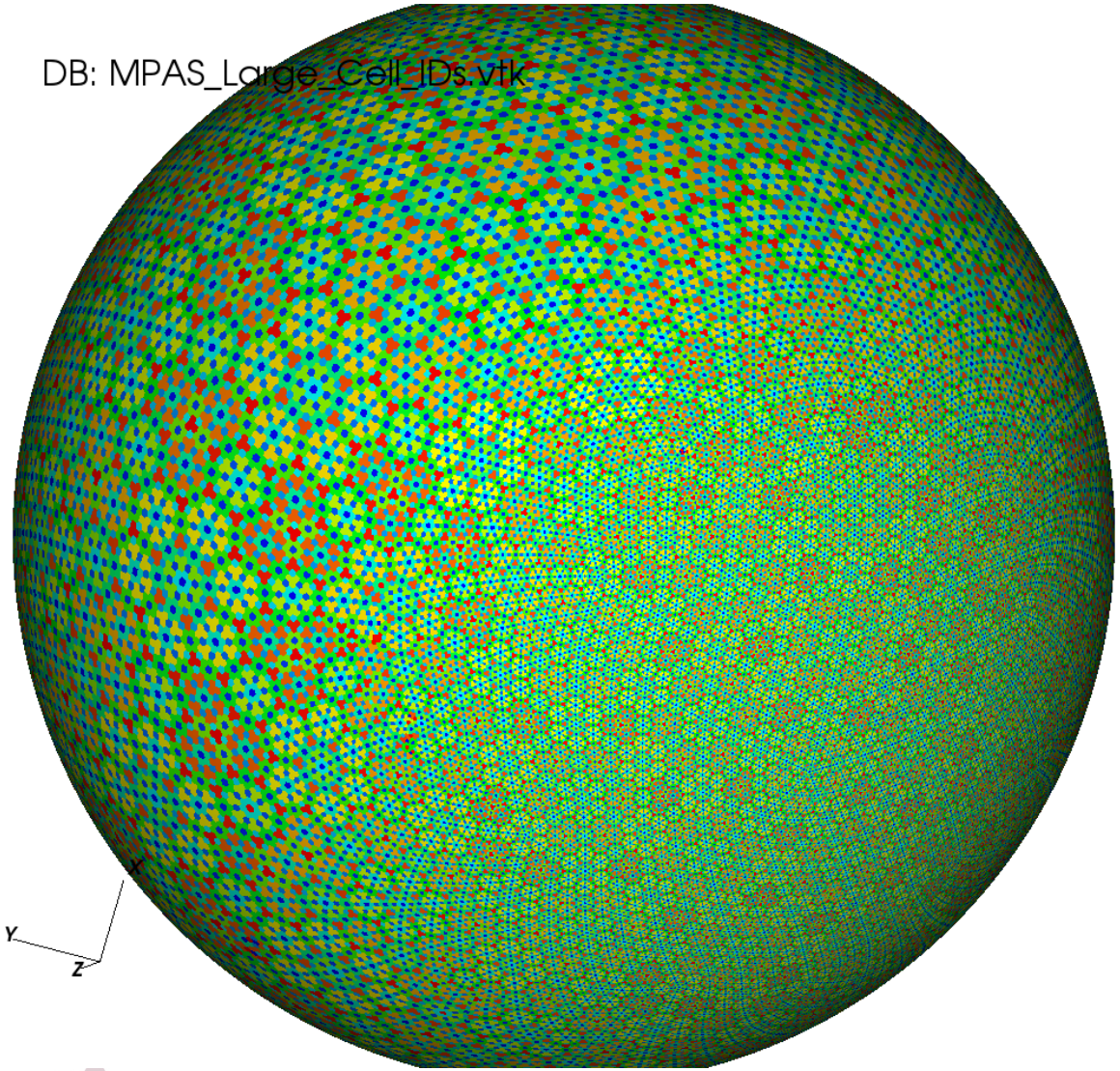

```
variables:
        float T(time, lev, lat, lon) ;
                T:units = "K" ;
                T:long_name = "Temperature" ;
```

```
if variable's dimensions contain at least ("lat" AND "lon")
   it should be on a cell center
 if variable's dimensions contain at least ("slat" AND "lon")
   it should be on a north/south edge
 if variable's dimensions contain at least  ("lat" AND "slon")
    it should be on an east/west edge
```

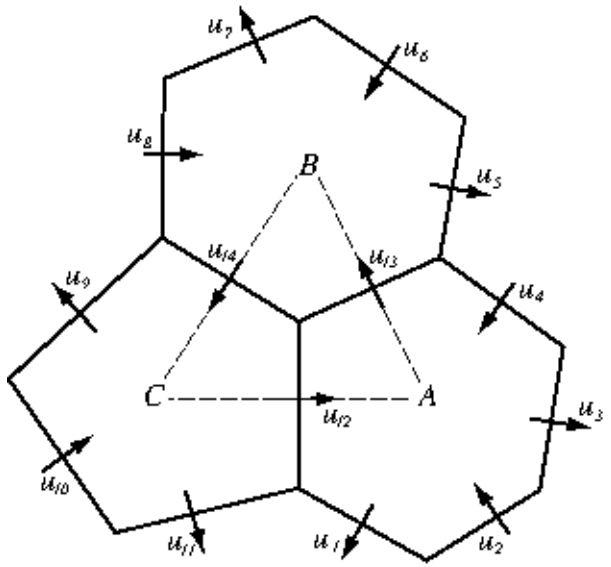
ParGAL grid/data read-and-setup capability

- Last year:
  - CAM-FV
  - CAM-EUL
- Now:
  - CAM-SE
  - MPAS (still beta…)

Once we've read in the NetCDF and filled in the missing metadata, MOAB can output in other formats (e.g. vtk)



Refined MPAS grid



ParGAL method for calculating vorticity (data parallel; using Intrepid and MOAB).

- The finite element method is used to solve the following weak equations for streamfunction and velocity potential using Intrepid

$$\int \nabla\psi \cdot \nabla\varphi \, d\Omega = \int \mathbf{v} \cdot (\mathbf{k} \times \nabla\varphi) d\Omega$$

$$\int \nabla\chi \cdot \nabla\varphi \, d\Omega = \int \mathbf{v} \cdot \nabla\varphi d\Omega$$

- Periodic boundary conditions along the latitudinal boundary and Neumann boundary conditions at the poles are used

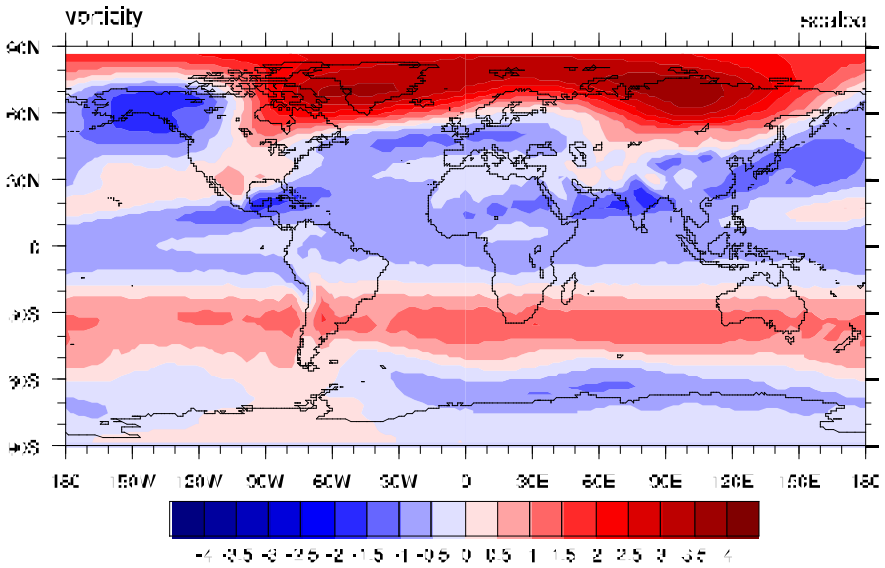
$$\int_\Gamma \left(\frac{\partial\chi}{\partial n} - \mathbf{v} \cdot \mathbf{n} \right) d\Gamma = 0 \qquad \int_\Gamma \left(\frac{\partial\psi}{\partial n} - \mathbf{v} \cdot \mathbf{t} \right) d\Gamma = 0$$

- The weak equations hold on arbitrary subdomains thereby enabling calculations from **regional** velocity data (e.g. WRF grids)

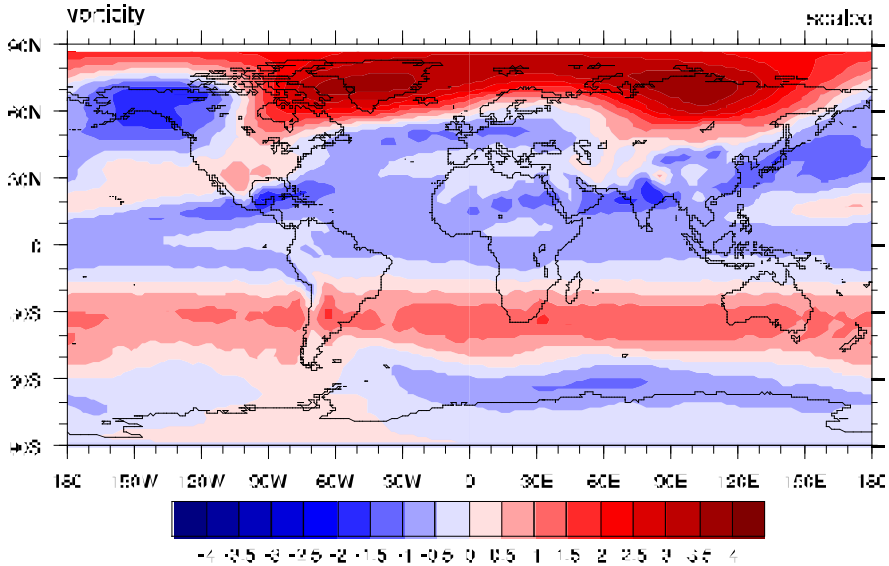
- Intrepid can support solution of these equations on triangles and quads and eventually on polygons.

Calculating Vorticity

Intrepid



NCL (uv2vrG_Wrap)



- Calculated locally on each element
- Easily parallelizable
- Global data not required

- Uses spherical harmonics
- Requires global data

$$vorticity = \frac{1}{r\cos\phi}\frac{\partial v}{\partial \lambda} - \frac{1}{r}\frac{\partial u}{\partial \phi} + \frac{u}{r}\tan\phi$$

ParGAL spatial derivative algorithm status

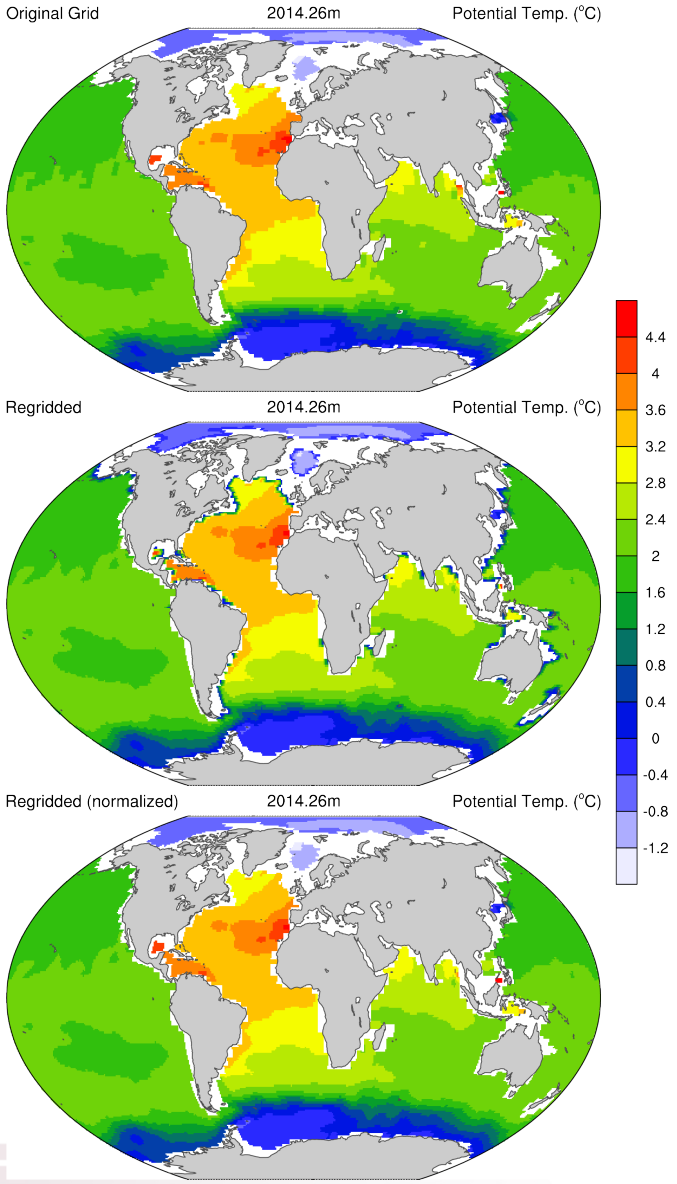
NCL Function	Description	Status
uv2dv(F,f,G,g)	Divergence from velocity field	In ParGAL for CAM-FV, CAM-Eul and CAM-SE grids
uv2vr(F,f,G,g)	Vorticity from velocity field	In ParGAL for CAM-FV, CAM-Eul and CAM-SE grids
uv2vrdv(F,f,G,g)	Vorticity and divergence from velocity field	In ParGAL for CAM-FV, CAM-Eul and CAM-SE grids
uv2sfvp(F,f,G,f)	Streamfunction and velocity potential from velocity field	Implementation developed for nodal quantities, structured grids
sfvp2uv(f,g)	Wind components from streamfunction and velocity potential	Implementation developed for nodal quantities, structured grids

NCAR Command Language (NCL)

A scripting language tailored for the analysis and visualization of geoscientific data

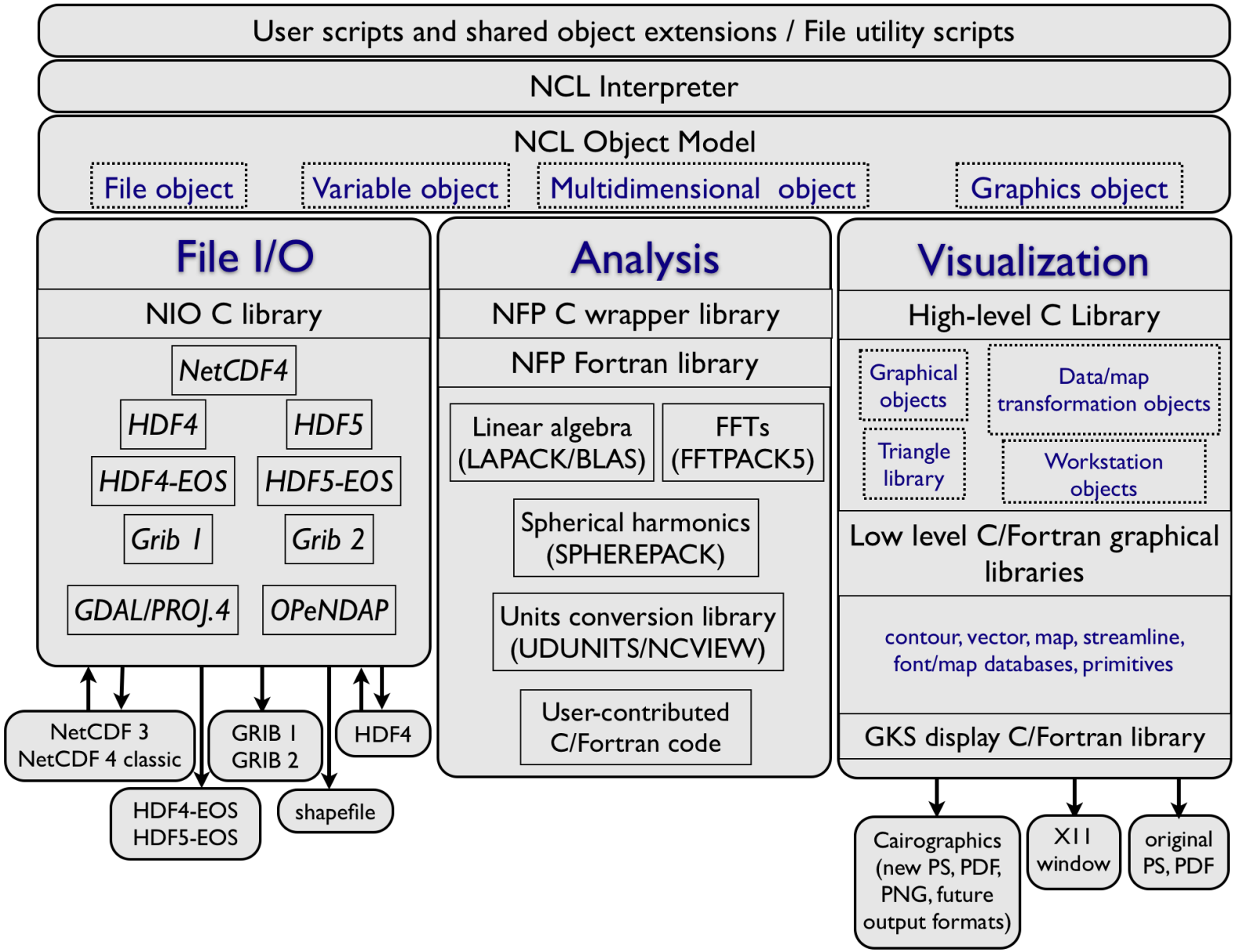
1. Simple, robust file input and output
2. Hundreds of analysis (computational) functions
3. Visualizations (2D) are publication quality and highly customizable

- **Community-based tool**
- Widely used by CESM developers/ users
- UNIX binaries & source available, free
- Extensive website, **regular workshops**

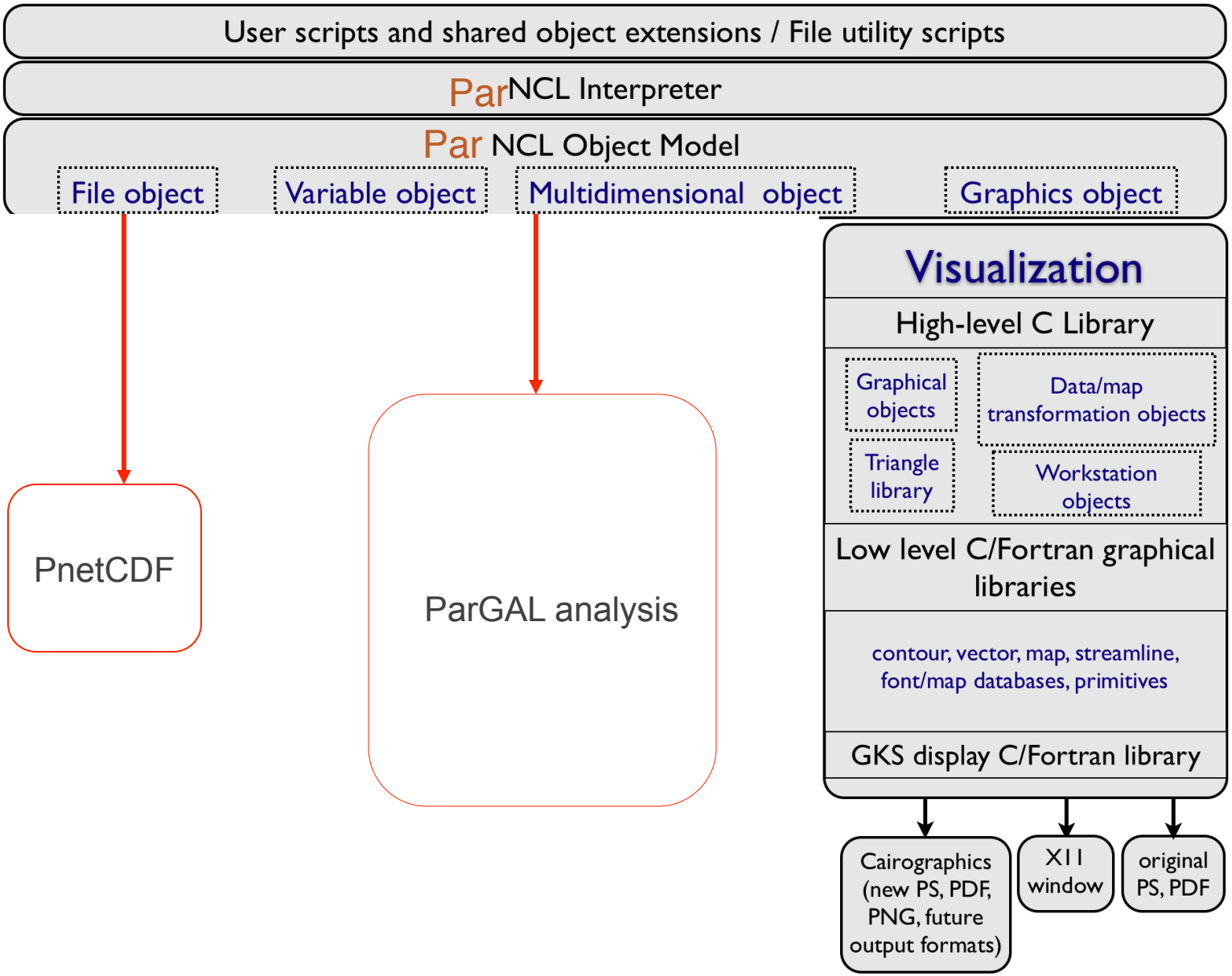


http://www.ncl.ucar.edu/

NCL architecture



ParNCL architecture



ParNCL function status

Also: algebraic operations and subsetting on distributed multi-dimensional arrays.

NCL function name	Climate Data Grids Supported		
	CAM Eulerian Spectral grid	CAM Finite Volume grid	CAM HOMME grid
addfile	YES	YES	YES
addfiles	YES	YES	YES
dim_avg_n	YES	YES	YES
dim_max_n	YES	YES	YES
dim_min_n	YES	YES	YES
dimsizes	YES	YES	YES
max	YES	YES	YES
min	YES	YES	YES
print	YES	YES	YES
printVarSummary	YES	YES	YES
systemfunc	YES	YES	YES
uv2vrA	YES	YES	YES
uv2dvA	YES	YES	YES
abs	YES	YES	YES
acos	YES	YES	YES
asin	YES	YES	YES
atan	YES	YES	YES
atan2	YES	YES	YES
cos	YES	YES	YES
exp	YES	YES	YES
fabs	YES	YES	YES
floor	YES	YES	YES
log	YES	YES	YES
log10	YES	YES	YES
sin	YES	YES	YES
sinh	YES	YES	YES
ceil	YES	YES	YES
Visualization functions	YES	YES	YES

ParNCL: Looks serial, executes in parallel!

An NCL script executed by ParNCL

```
f = addfiles(diri+fili, "r") ;   open file

tt = f[:]->T(0,{500},{-30:30},:) ;  read a section of data.


wks  = gsn_open_wks("ps","parvis_t")  ; open a PS file

plot = gsn_csm_contour_map(wks, tt(:,:),False)
```

Read with parallel I/O

a parallel data object

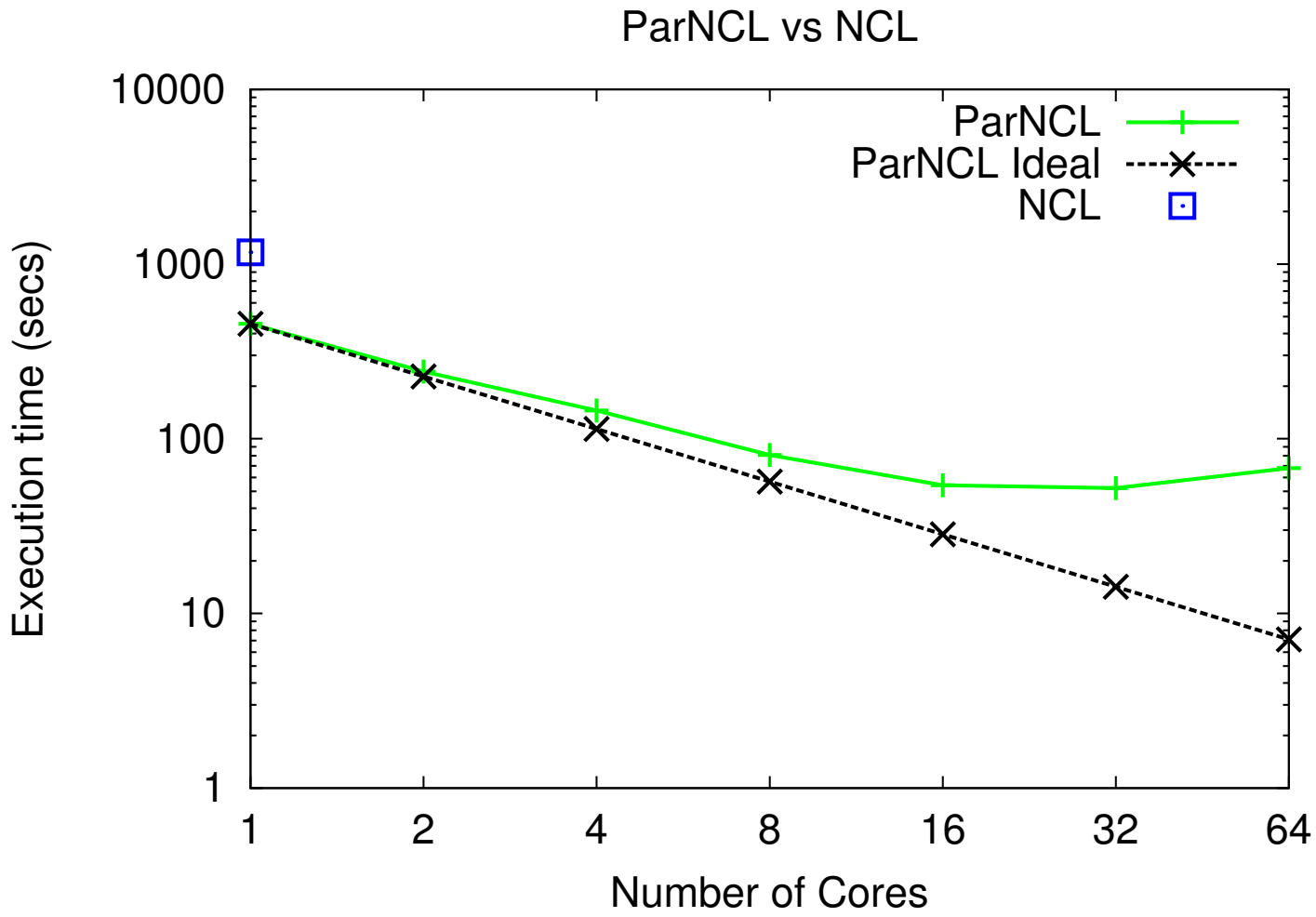
In the ParNCL interpreter, tt is gathered to one node and passed to normal NCL graphics routines for plotting.

Using ParNCL requires a parallel environment:

Now: `> ncl myscript.ncl`

With ParNCL: `> mpirun -np 32 parcnl myscript.ncl`

Calculating Vorticity: ParNCL vs NCL



Total Execution time for reading 4 timesteps of data and calculating the vorticity field on each level of a 768x1152x26 grid (FV 0.25) vs. number of cores.

Additional ParVis area: Compression

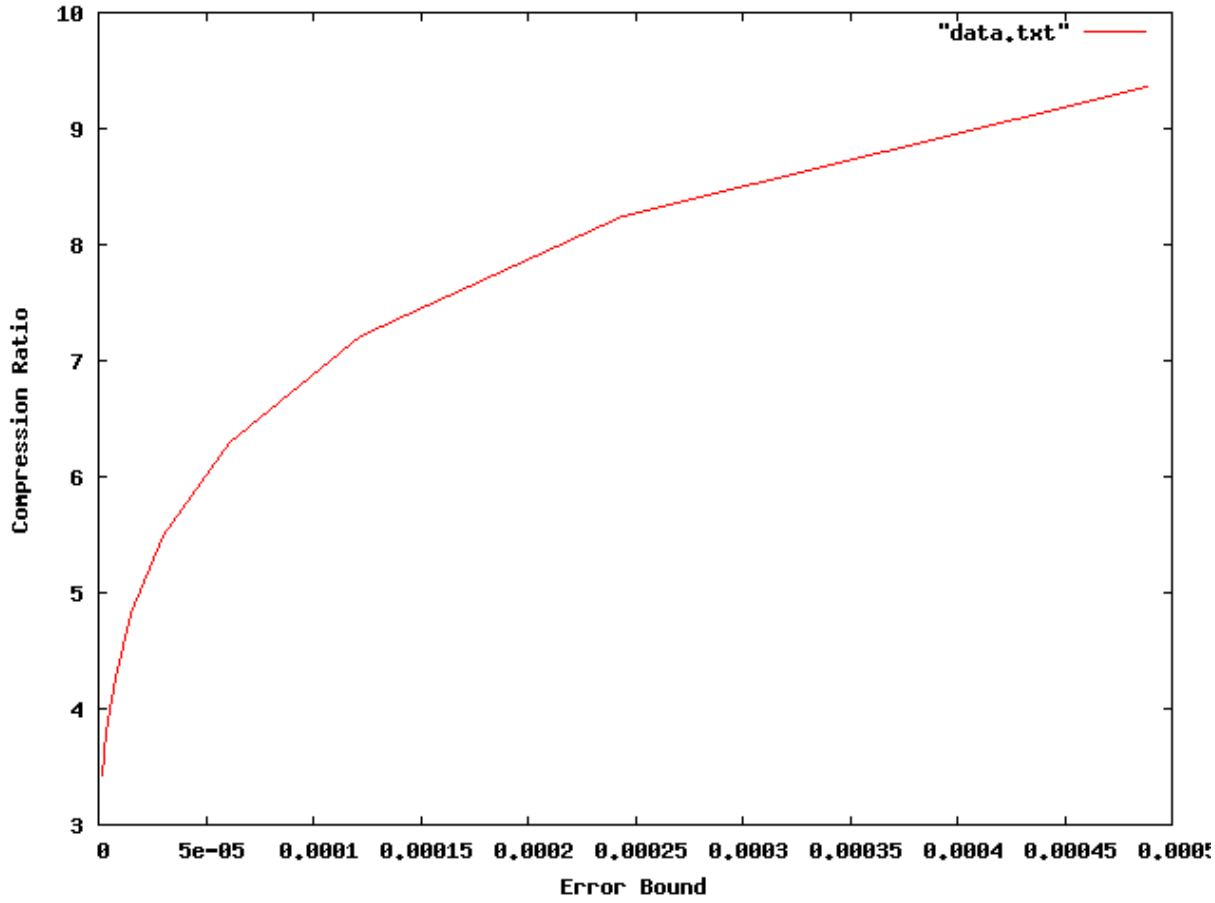
- Completely random data can not be compressed without information loss but many climate output fields are smooth, not random.

- Lossless compression can reduce volume of the climate data without information loss
  - Reduce storage, memory, and network requirements to store, process, and transfer the data
  - Compression can potentially speedup analysis and visualization applications
    - Light weight and Integrate well with the applications

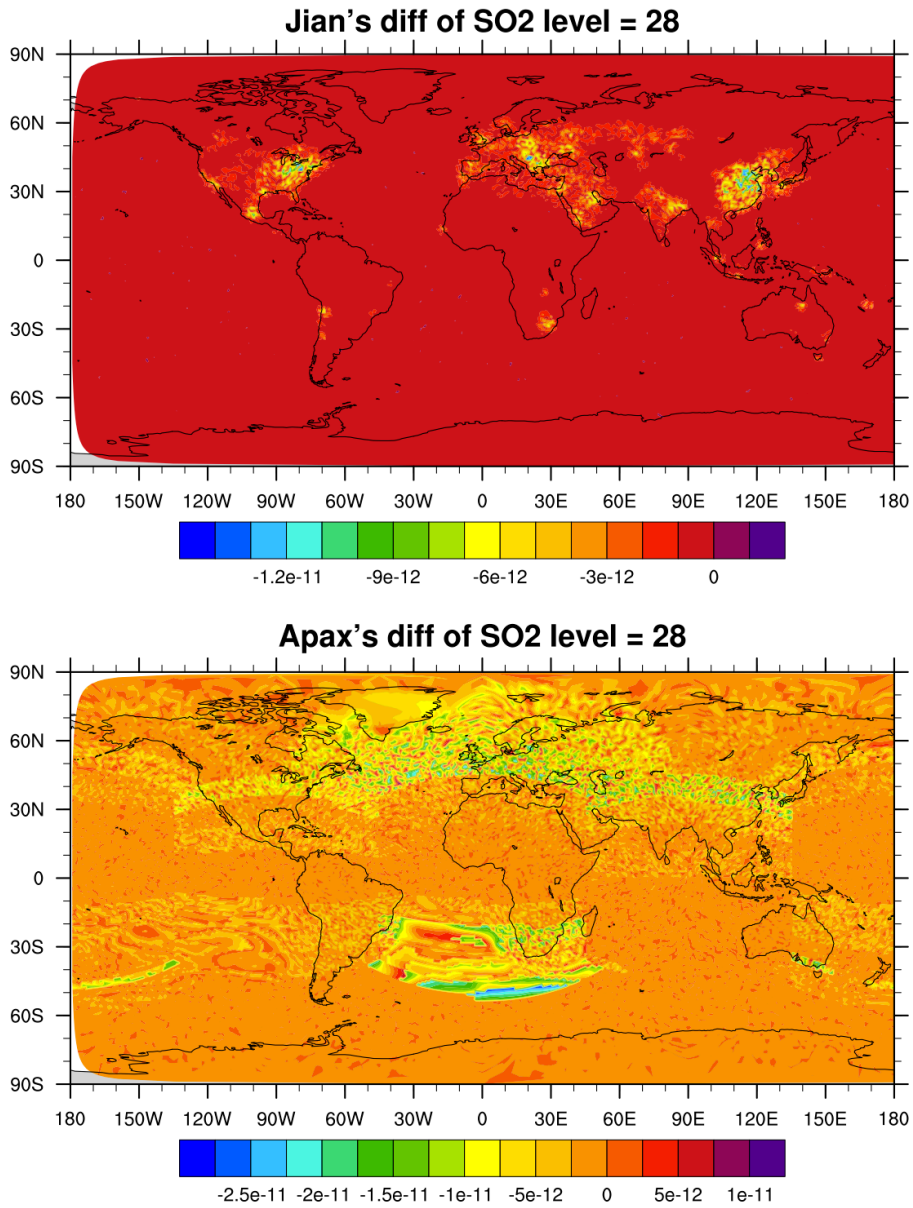
- Lossy compression can achieve higher compression ratio
  - May be appropriate for some applications.

- Need for compression is here now.

Lossy Compression results



- Error for each value is bounded
- Preliminary results show that we can achieve a compression ratio around 10 when the error bound is 0.1%
- Further improvement is possible with improvement in the second part of our two-stage compression



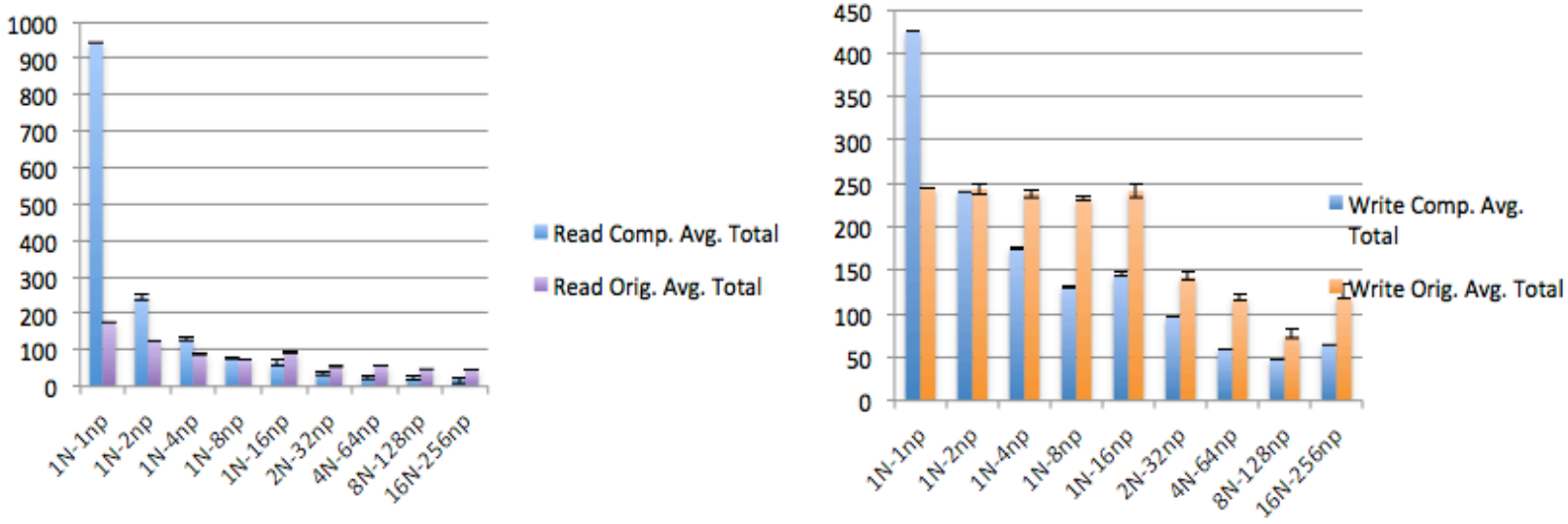
Difference between compressed and uncompressed data (concentration of Sulfate at lowest atmospheric model level) for ParVis compression scheme (top) and Apax scheme (bottom)

Incorporating compression: PNetCDF

- Fetch compressed data through MPI-IO
- Advantages
  - Reduce disk overhead
  - Reduce communication overhead
  - Can be added to existing applications (PIO, ParGAL)
- Disadvantage
  - Challenging when PnetCDF accesses and data compression are not aligned
  - Pipelining is difficult

PNetCDF with compression test case

- 28 km data (27 layers) temperature data from the GCRM.
- Test on linux cluster with fat memory nodes.



What you can do…

- Let us know:
  - Where bottlenecks are in your analysis workflow. What algorithms take too long or need too much memory?
  - What kind of post processing analysis would you like to do but can't?

- Attend our session at 2013 Fall AGU.
  - IN006: Big Data in the Geosciences: New Analytics Methods and Parallel Algorithms
  - Conveners: Jitendra Kumar, Robert Jacob, Forrest Hoffman, Don Middleton

- Check the website: trac.mcs.anl.gov/projects/parvis
  - Subscribe to ParVis announcement mailing list: parvis-ann
  - ParNCL beta2 available now!
  - ParNCL 1.0 release in early July!

MCT Update

MCT Philosophy: Model coupling vs. "the coupler"

- MCT is not a coupler
- Instead, MCT provides datatypes/methods you add to models (and a separate coupler or driver if you want one) to make the models "couple-able".
- Must also follow a few programming standards (suggested but not required for using MCT)
  - Separate "initialization" and "run" methods.
  - Do not use MPI_COMM_WORLD everywhere.
  - Avoid global data types

Model Coupling Toolkit: History

- Pre-History:
  - 1996 parallel coupler for Fast Ocean Atmosphere Model (Jacob, UW-Madison)
  - 1998 Physical-space Statistical Analysis System (Larson, NASA)

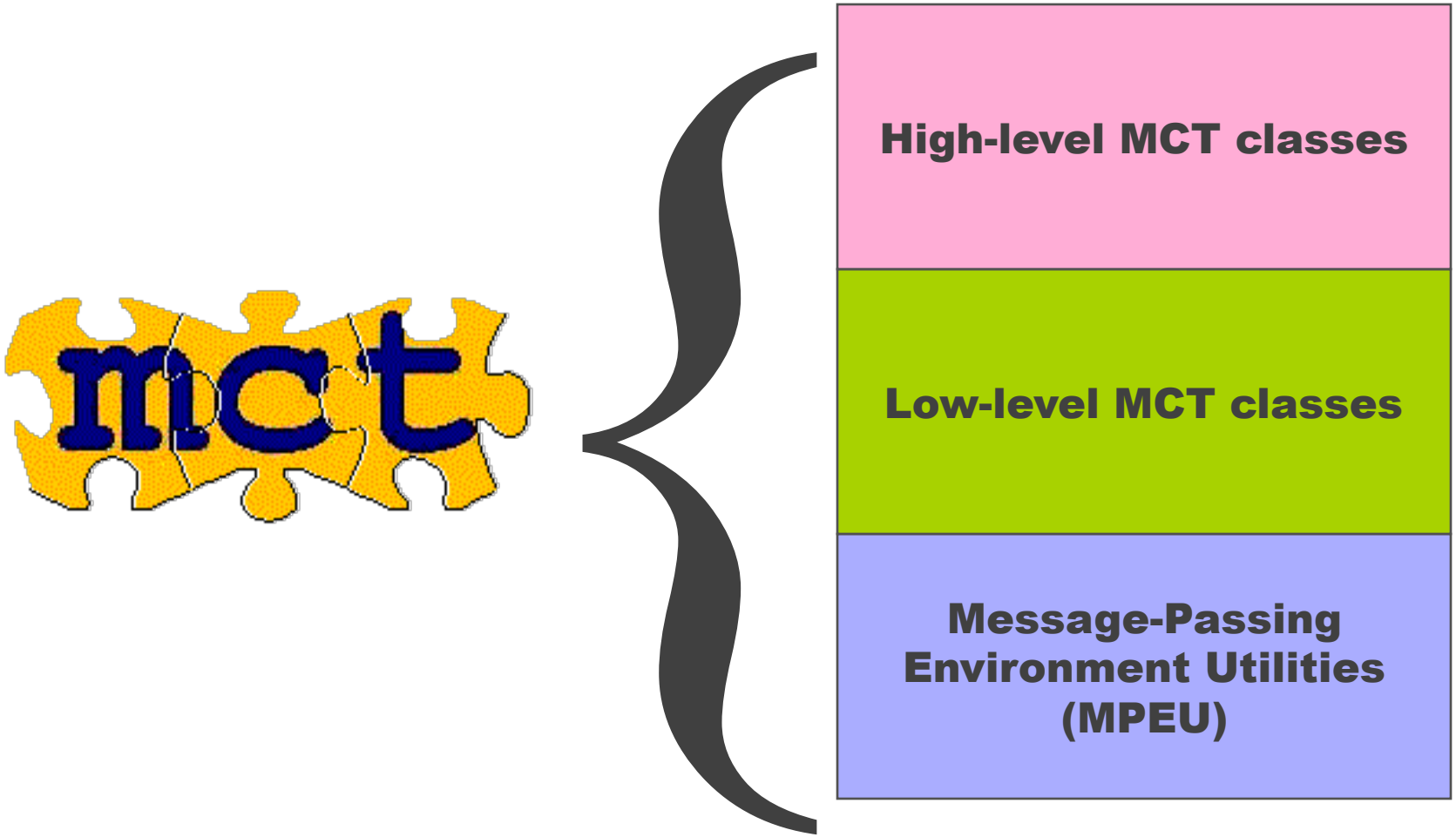
- U.S. Department of Energy ACPI *Avante-Garde* project (2000-2001)
  - First work on parallelizing CCSM coupler (cpl5->cpl6)
  - MCT 1.0

- DOE Scientific Discovery through Advanced Computing (SciDAC) (2001-2011)
  - MCT 2.0 – MCT 2.7.3
  - CCSM3, CCSM4, CESM1.

- DOE Climate Science for a Sustainable Energy Future (CSSEF) (2011 – 2015)
  - MCT 2.8
  - Next generation MCT

All DOE support is from the climate modeling program in the Office of Biological and Environmental Research (BER) in the Office of Science.

MCT Architecture



MCT Recent history

- 01/06/2010: MCT 2.7.0 released in CCSM4
  - Limted used of OpenMP
- 02/28/2010: MCT 2.7.1 released in CESM1
- 11/30/2010: MCT 2.7.2 released in CESM1.0.3

(CW2010 in Toulouse, France. December, 2010)

- 01/25/2011: MCT 2.7.3 add debugging option to configure

(2011: Some divergence between Argonne and NCAR MCT repositories)

- 02/07/2012: MCT 2.7.4 update autoconf build to latest version (Jim Edwards)

MCT More Recent history

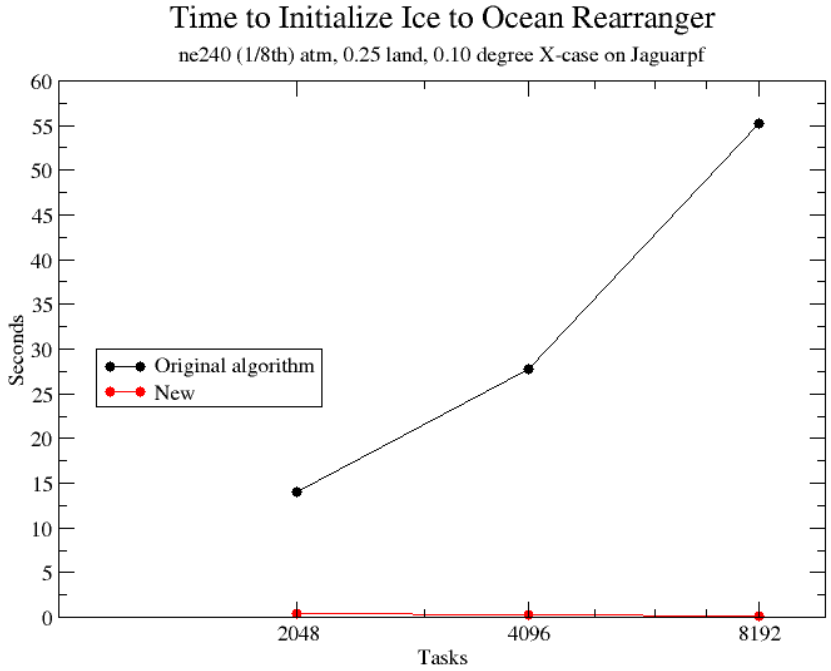
- MCT 2.8.0 - Released April 30, 2012 (first standalone release since 2.6)!
  - Merged differences in Argonne and NCAR MCT repos.
  - New datatype in AttributeVector to speed up copies (thanks to Bill Sacks, NCAR)
  - ANL and NCAR repos in sync!

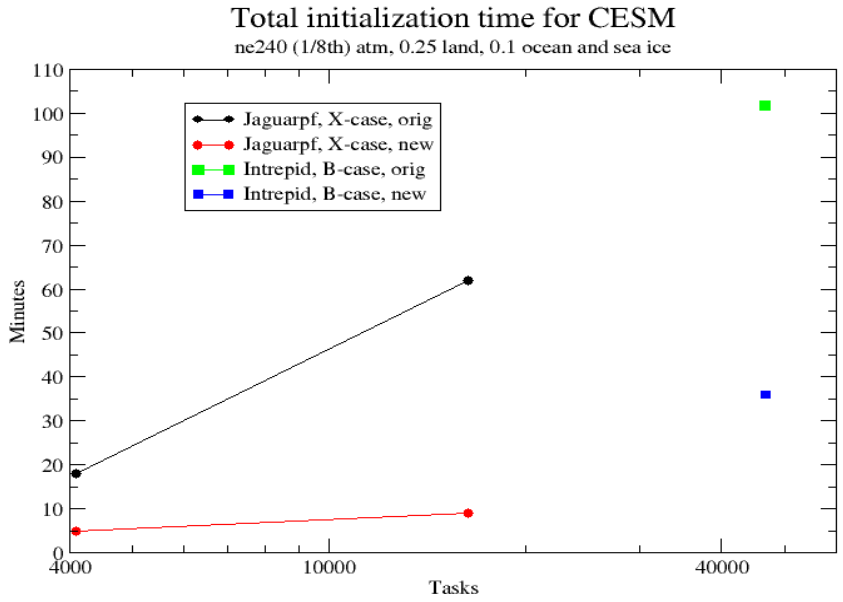
- 07/12/12 - MCT 2.8.1 Convert Argonne repository to git
  - Repository now world readable. Copy on github.com
  - Full 10+ year history MCT development converted.
  - Github provides SVN interface allowing CESM to pull directly. Eliminate duplicate repo at NCAR.

```
git clone http:git.mcs.anl.gov/MCT.git
```

Continuing to improve cpl7/MCT performance at scale.

- Slow initialization time for high-resolution, high-processor count cases.
  - 1/8th degree runs on Intrepid were taking over 1.5 hours to initialize.
  - Traced to initialization of MCT's Rearranger (equal to 2 Routers) which moves data between overlapping decompositions. Thanks Tony!





MCT More Recent history

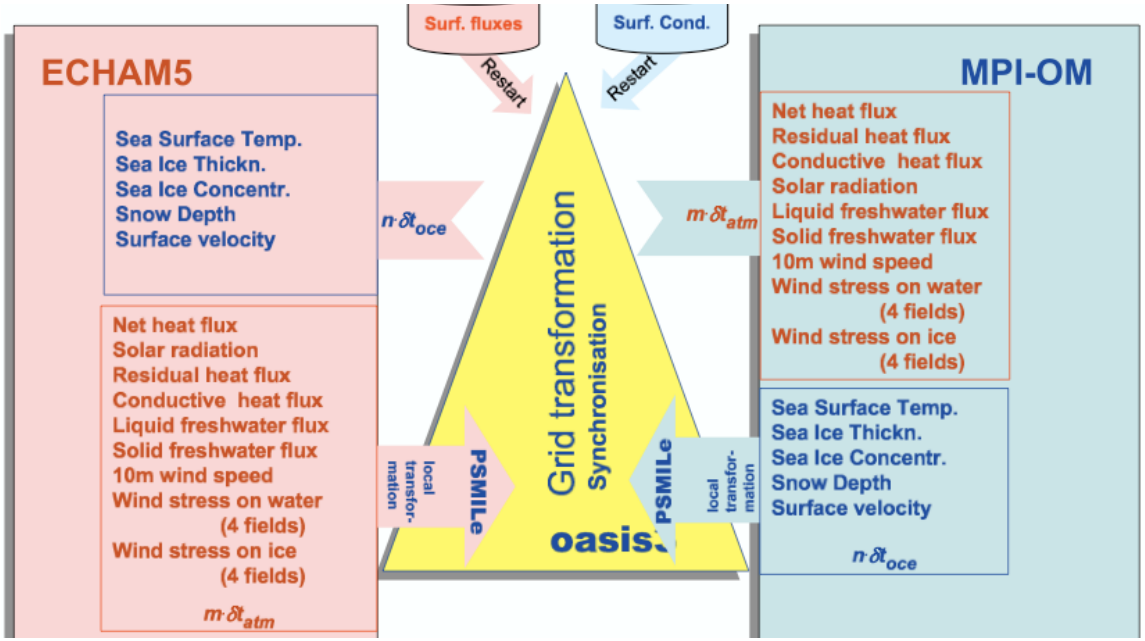
- 09/12/12 - MCT 2.8.2
  - Includes fix for slow Router init.
  - Released in CESM 1.1
  - Not released separately.

- 12/19/12 – MCT 2.8.3 Current Standalone version
  - Public release
  - All of above changes plus some minor compiler fixes

- CESM1.2 includes some small modifications to MCT for some compilers
  - To be merged in to next public release

New user! OASIS3-MCT

First release: August, 2012
OASIS3-MCT2: June, 2013



CSSEF research demands on coupling

- Dynamical Adaptive Atmospheric Dynamics
  - Grid points are created and destroyed on a coupler processor
  - Changing cell sizes for **just one** grid within coupler will require online calculation of new interpolation weights
    - Which requires more information about **both grids** then currently in coupler.
- Development of MPAS-Ocean
  - Need to retain information about unstructured grids for interpolation weight calculation.
- Resiliency and Scaling
  - Dynamic load balancing and resilient computing means points could move from processor to processor.
  - Millions of threads and small per-core memory means need more parallelism and optimize for low-memory

Solution:
Re-Implement MCT data model with MOAB

- MOAB = Mesh Oriented dAtaBase
  - A database for mesh (structured and unstructured) and field data associated with mesh
  - *Tuned for memory efficiency first*, speed a close second
  - Serial, parallel look very similar, parallel data constructs imbedded in MOAB interface
  - http://trac.mcs.anl.gov/projects/ITAPS/wiki/MOAB
  - Developed under DOE SciDAC program
  - Includes parallel I/O and visualization capabilities.
  - Included in nuclear engineering exascale co-design center.

AttrVect (iMesh)

```
type AttrVect
      type(List) :: iList
      type(List) :: rList
      iBase_TagHandle,dimension(:),pointer :: itagh
      iBase_TagHandle,dimension(:),pointer :: rtagh
      iBase_EntityHandle,dimension(:),pointer :: enths
end type AttrVect
```

- Built on top of iMesh interface to MOAB
  - INTEGER/REAL attribute lists retained
  - Natural equivalence between “attribute” and “tag”
  - Attributes now stored contiguously and referenced by a handle iBase_TagHandle (implemented as an integer)
  - Mesh entities referenced by iBase_EntityHandles
  - Progress has been slow

Near term plans for MCT and MCT-MOAB

- Build test program for MCT Router initialization. -DONE!
- Build similar program for MCT-MOAB “Router” initialization.
- Test with high-resolution, high-core-count (100K) cases on Intrepid.
- Compare performance for initialization and runtime.

Additional MCT development

- New Features to aid in debugging Router times
  - GSMap and MCTWorld print().
    - Print contents to ascii file for later reading
  - Router init internal timers
    - Invoked with optional string argument to Router init.
  - RouterTest.F90 - test program which reads in output GSMap and MCTWorld info and builds a Router.
    - Will build on same number of procs and same decomposition as original model.
  - On branch but not yet released
- Also planned: Conversion from F90 to F95 (and later F2003)

MCT: To be continued…

www.mcs.anl.gov/mct