ParVis and MCT update

Robert Jacob

CESM Workshop Software Engineering Working Group.

June 18, 2012

Breckenridge, CO

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, 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 Earth System Modeling Program of the Office of Biological and Environmental Research of the U.S. Department of Energy's Office of Science

parVis

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

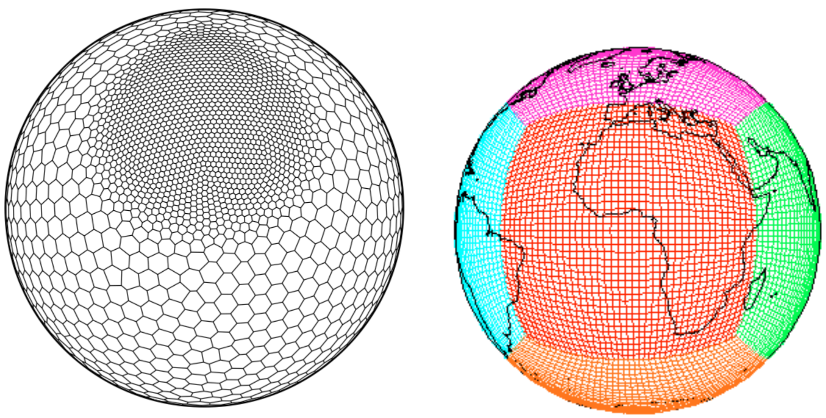
Motivation:

Ability to gain insight from current and future climate data sets $\neq$ Capability of current tools

Motivation

- CAM-SE 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

Output data getting larger



Grids no longer rectangular

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

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

} No parallelism

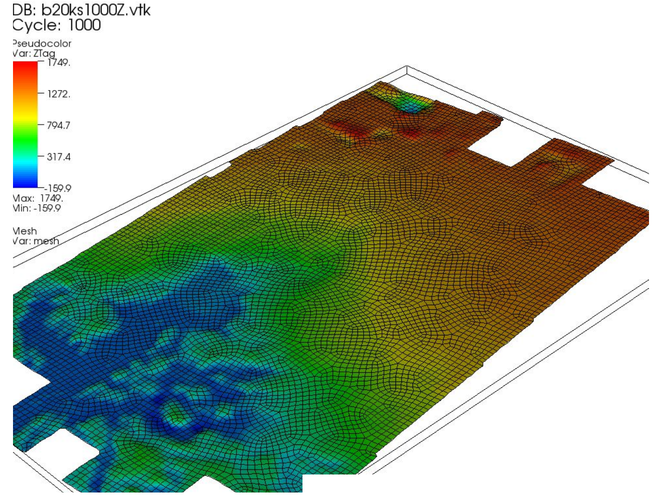
ParVis will speed up data analysis and visualization through data- and task-parallelism AND natively support multiple grids AND reconstruct the discretization used in the models.

Approach

- Use existing tools to speed-up development.
- As much as possible, preserve well-established workflows for analyzing climate data, just speed them up.
- There is a problem *right now* so provide both immediate and long-term help
- Assemble a multi-disciplinary and multi-institutional team to carry out the work.

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 - Parallel Gridded Analysis Library

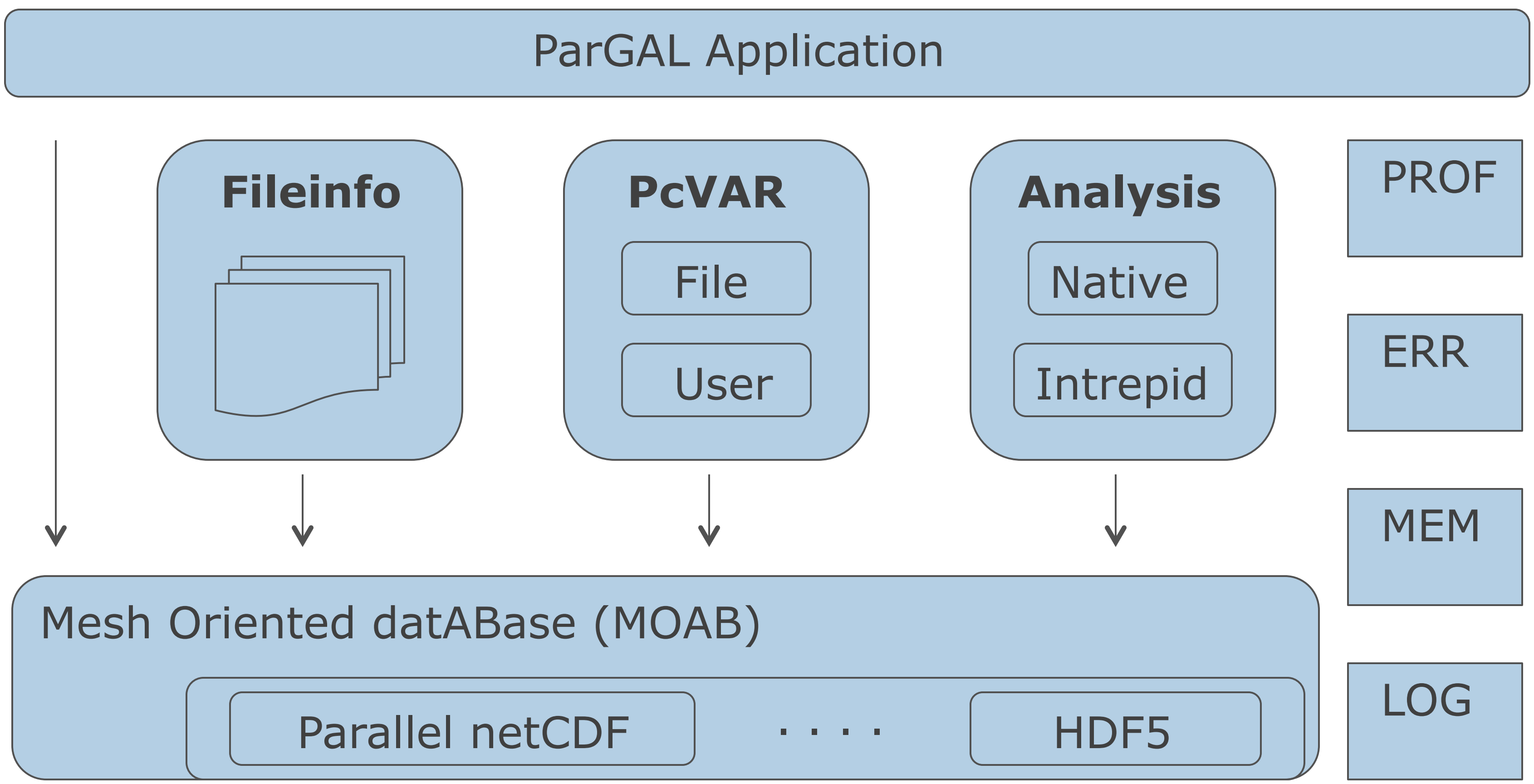
- The main product from ParVis.
  - Data parallel C++ Library
  - Typical climate analysis functionality (such as found in NCL)
  - Structured and unstructured numerical grids

- Built upon existing tools
  - MOAB
  - Intrepid
    - MOAB and Intrepid have already solved the hard problem of how to represent and operate on structured and unstructured grids distributed over processors.
  - PnetCDF
  - MPI

- Will provide data-parallel core to perform typical climate post-processing currently.

- **Will be able to handle unstructured and semi-structured grids in all operations by building on MOAB and Intrepid. Will support parallel I/O by using PnetCDF.**

ParGAL Architecture

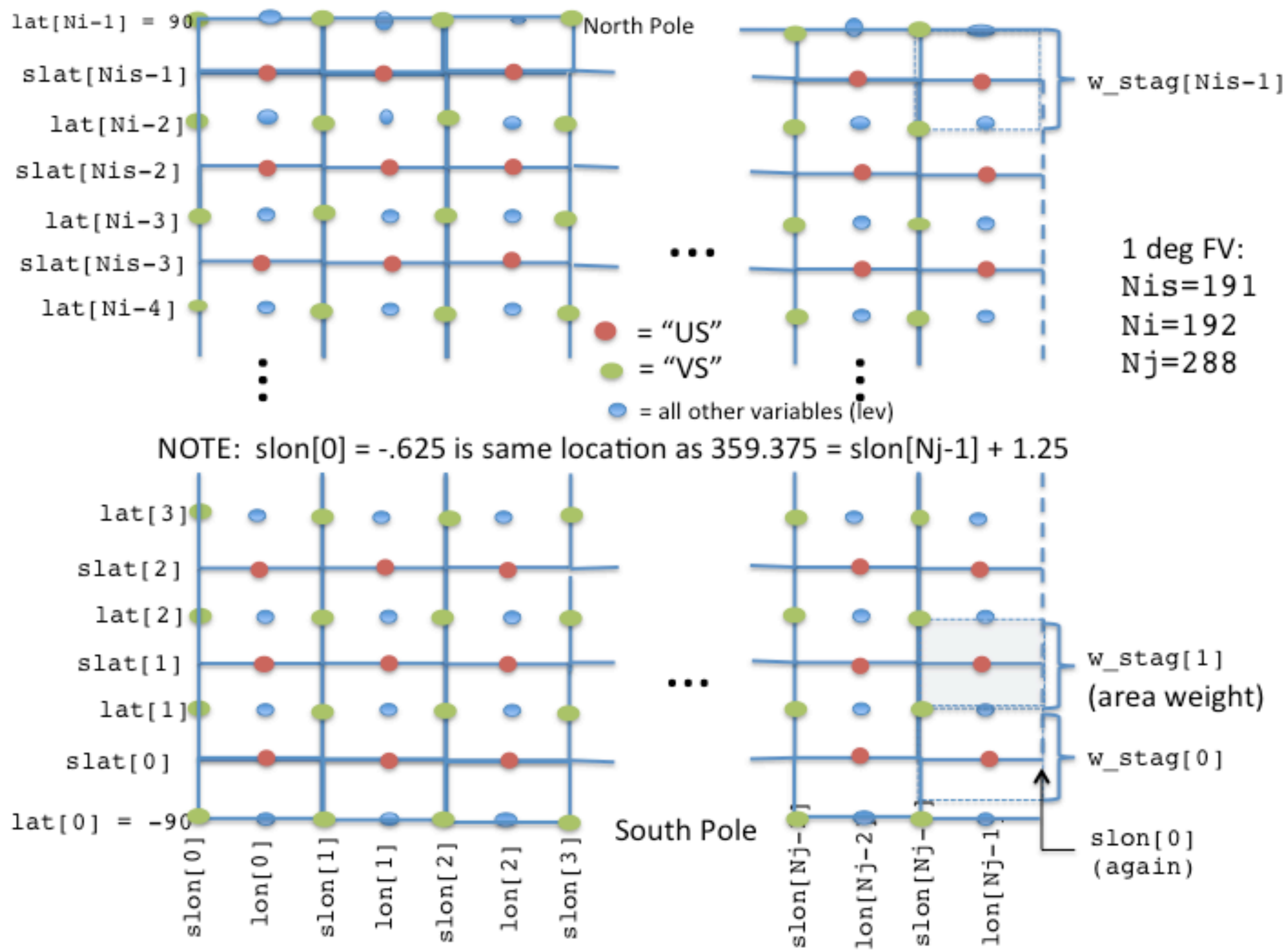


ParGAL Architecture - contd

- Fileinfo
  - Abstraction of multiple files
- PcVAR
  - File Variables
  - User Variables
  - Read/write data through **MOAB**
- Analysis
  - Native: dim_avg_n, max, min (already implemented)
  - **Intrepid**
- MOAB
  - Parallel IO/Storage
- Support Functions
  - MEM, ERR, LOG, PROF

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

Development with Intrepid (a component of ParGAL)

- Divergence and vorticity
  - developed parallel versions using Epetra package from Trilinos
  - will update current ParGAL implementation with new tag_reduce functionality from MOAB
- Streamfunction and velocity potential
  - implemented with Intrepid for global velocity fields
  - investigating approach for limited domains
  - will incorporate into ParGAL
- Irrotational and non-divergent velocity components
  - implementation underway

Calculating Streamfunction and Velocity Potential with Intrepid.

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

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

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

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

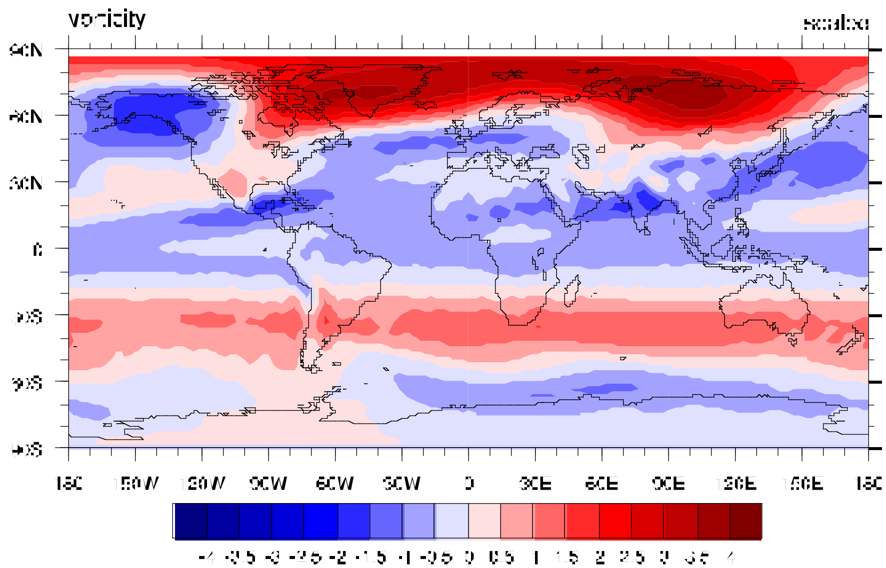
$$\int_\Gamma \left(\frac{\partial\chi}{\partial n} - \mathbf{v} \cdot \mathbf{n} \right) \mathrm{d}\Gamma = 0 \qquad \int_\Gamma \left(\frac{\partial\psi}{\partial n} - \mathbf{v} \cdot \mathbf{t} \right) \mathrm{d}\Gamma = 0$$

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

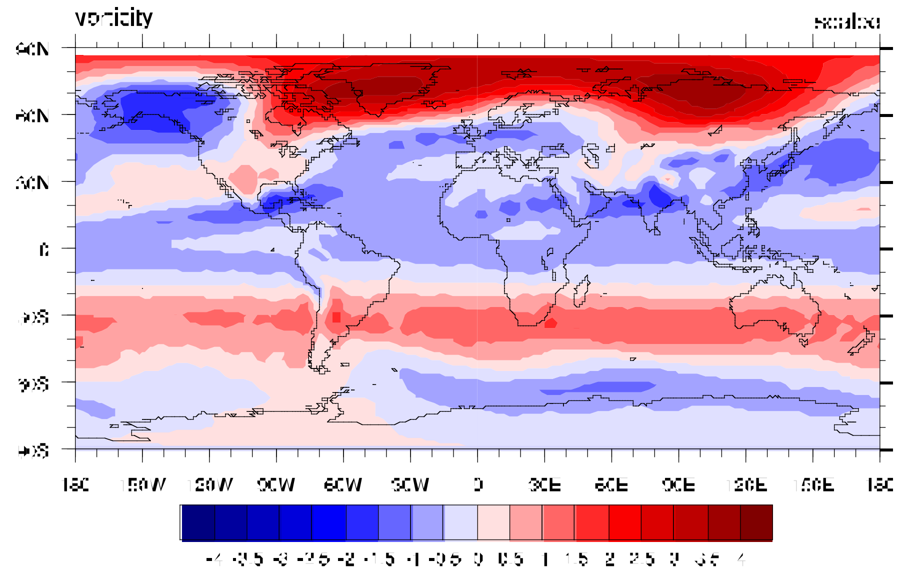
- Intrepid can support solution of these equations on triangles and quads and eventually on polygons.

Calculating Vorticity with Intrepid

Intrepid



NCL (uv2vrG_Wrap)



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

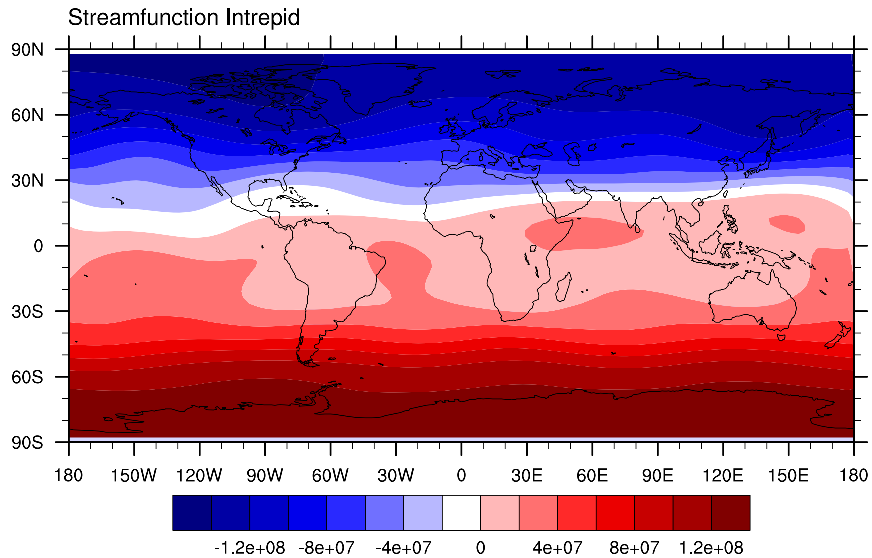
- Uses spherical harmonics
- Requires global data

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

Calculating Streamfunction with Intrepid

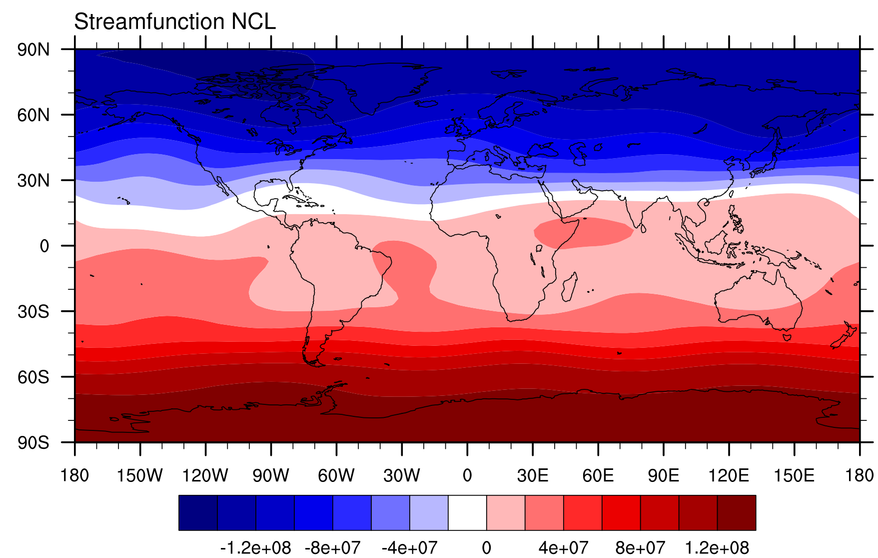
Intrepid

finite element method



NCL (uv2sfvpG)

spherical harmonics

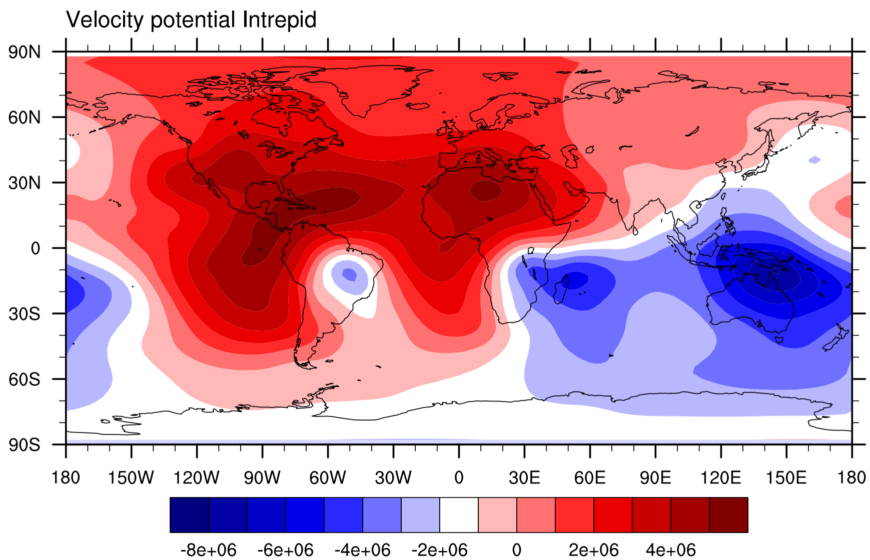


$$\nabla^2 \psi = \nabla \times \mathbf{v}$$

Calculating Velocity Potential with Intrepid

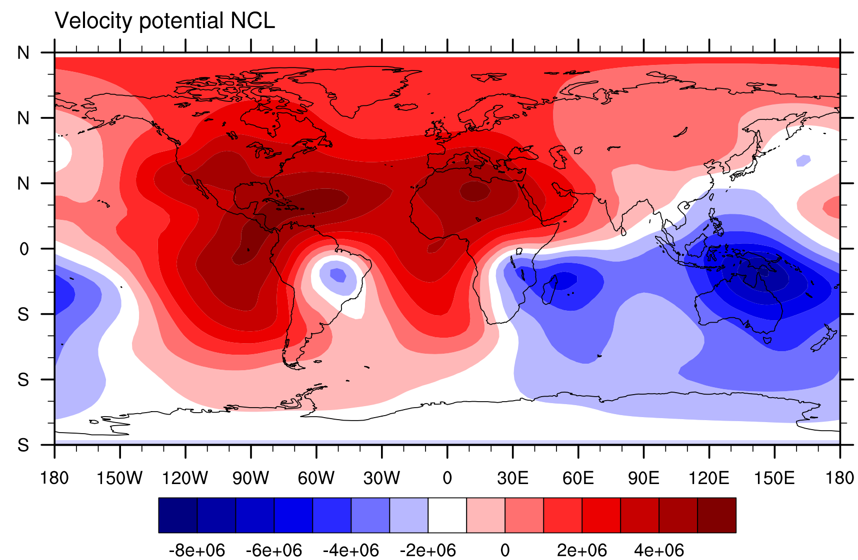
Intrepid

finite element method



NCL (uv2sfvpG)

spherical harmonics



$$\nabla^2 \chi = \nabla \cdot \mathbf{v}$$

ParGAL Development

- Integrated Intrepid-based algorithms into ParGAL
  - divergence, vorticity
- Updated algorithms to use MOAB partition method (SQIJ)
  - Gather (internal to ParGAL), NCL functions dim_avg_n, max, min
- Implemented new NCL algorithms
  - dim_max_n
  - dim_min_n
  - dim_median_n
- Miscellaneous items
  - Added tests of different dimensions (time, lev, lat, lon) for all *_n algorithms
  - Added more test configurations to nightly build/test system (Buildbot)
  - Added support within MOAB and ParGAL to read global and variable attributes
  - Updated installation documentation and README

ParGAL Function Table

NCL function Group	NCL Functions	ParCAL Function
File IO	addfile, addfiles	fileinfo, pcvar
Spherical Harmonic Routines	dv2uv* (4 funcs)	divergence
Meteorology	uv2dv_cfd	divergence
Spherical Harmonic Routines	uv2dv* (4 funcs)	divergence
Meteorology	uv2vr_cfd	vorticity
Spherical Harmonic Routines	uv2vr* (4 funcs)	vorticity
Spherical Harmonic Routines	uv2vrdv* (4 funcs)	vorticity, divergence
General Applied Math	dim_avg, dim_avg_n	dim_avg_n
General Applied Math	dim_max, dim_max_n	dim_max_n
General Applied Math	dim_min, dim_min_n	dim_min_n
General Applied Math	dim_median, dim_median_n	dim_median_n
General Applied Math	max	max
General Applied Math	min	min
Variable Manipulators	delete	pcvar
		gather

NCAR Command Language (NCL)

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

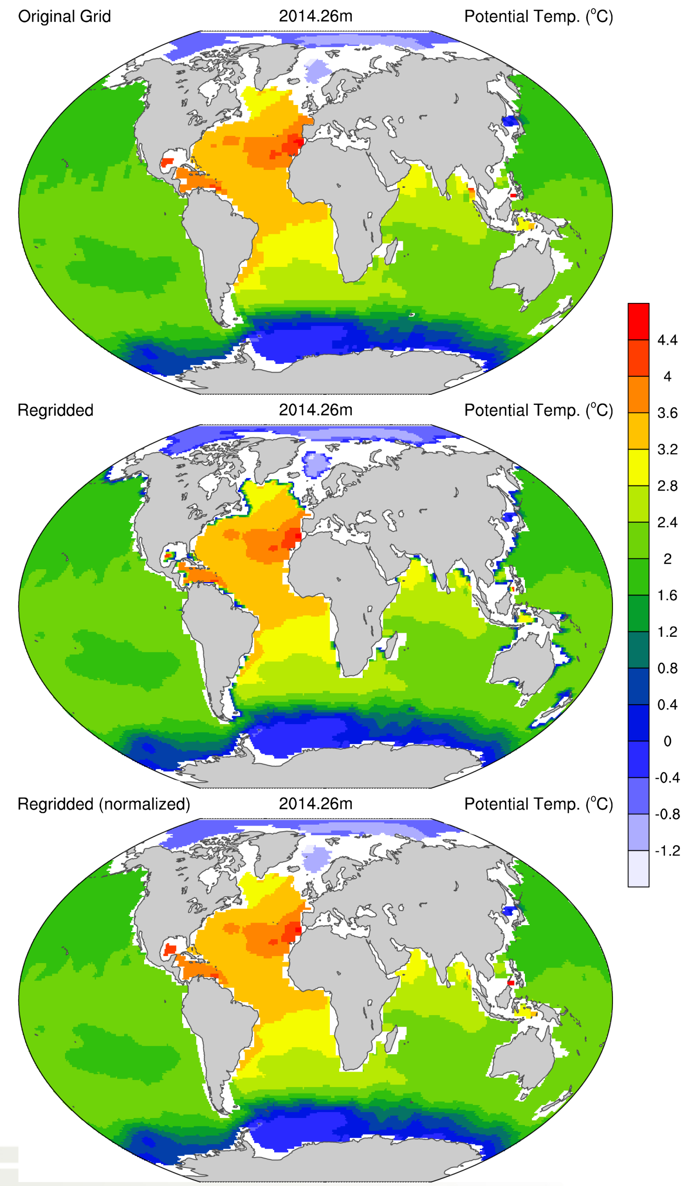
1. Simple, robust file input and output
2. Hundreds of analysis (computational) functions
3. Visualizations (2D) are publication quality and highly customizable

– **Community-based tool**

– Widely used by CESM developers/ users

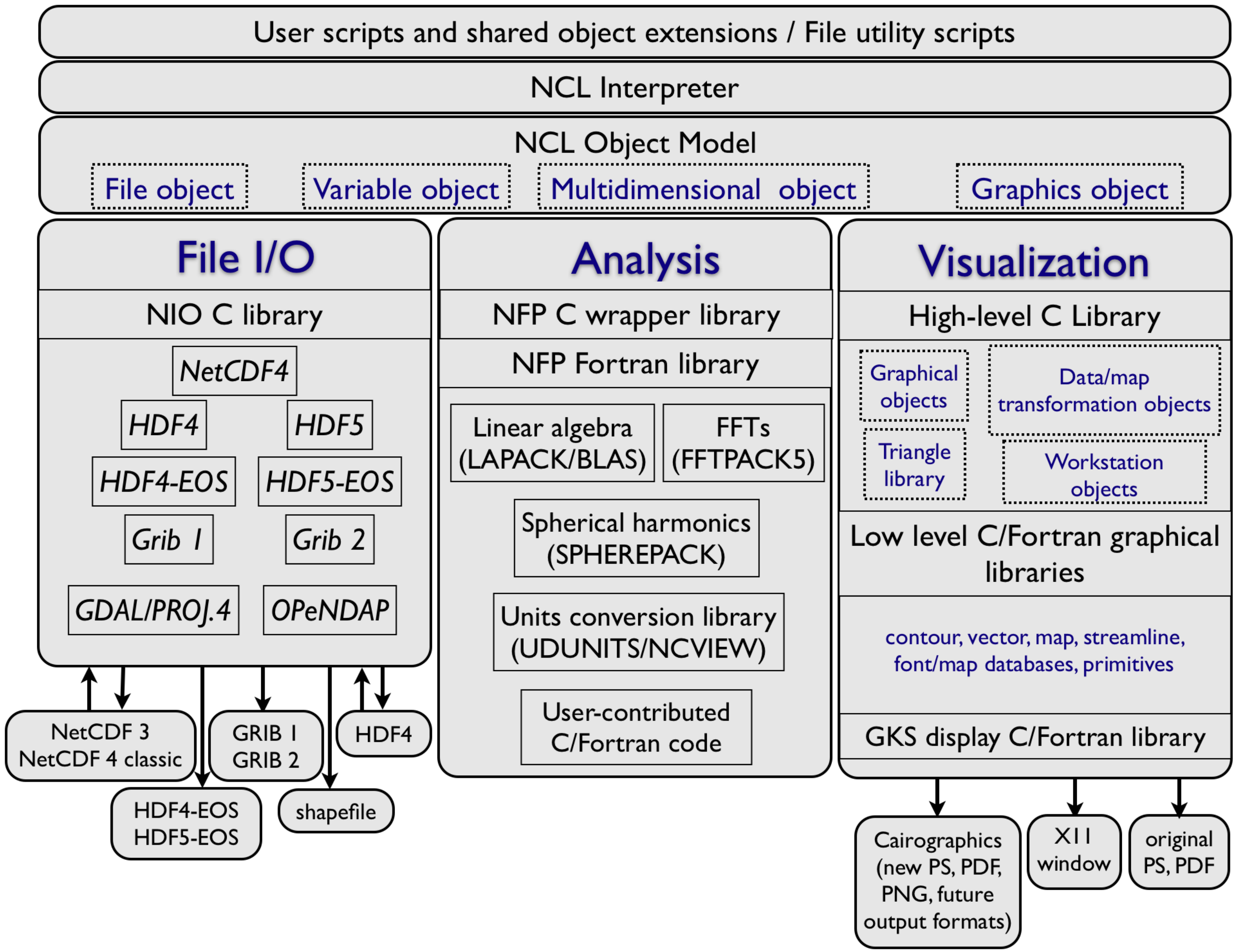
– UNIX binaries & source available, free

– Extensive website, **regular workshops**

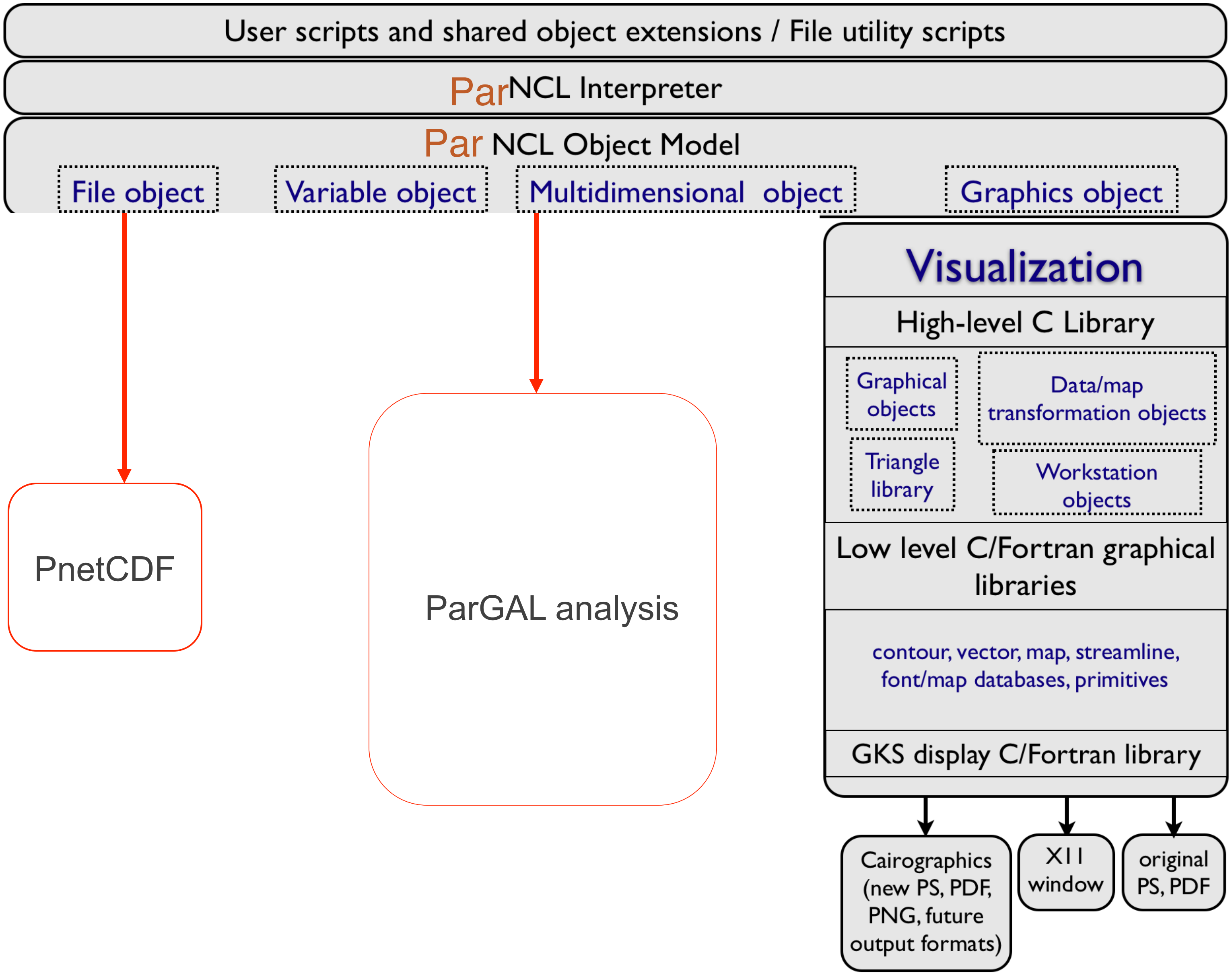


http://www.ncl.ucar.edu/

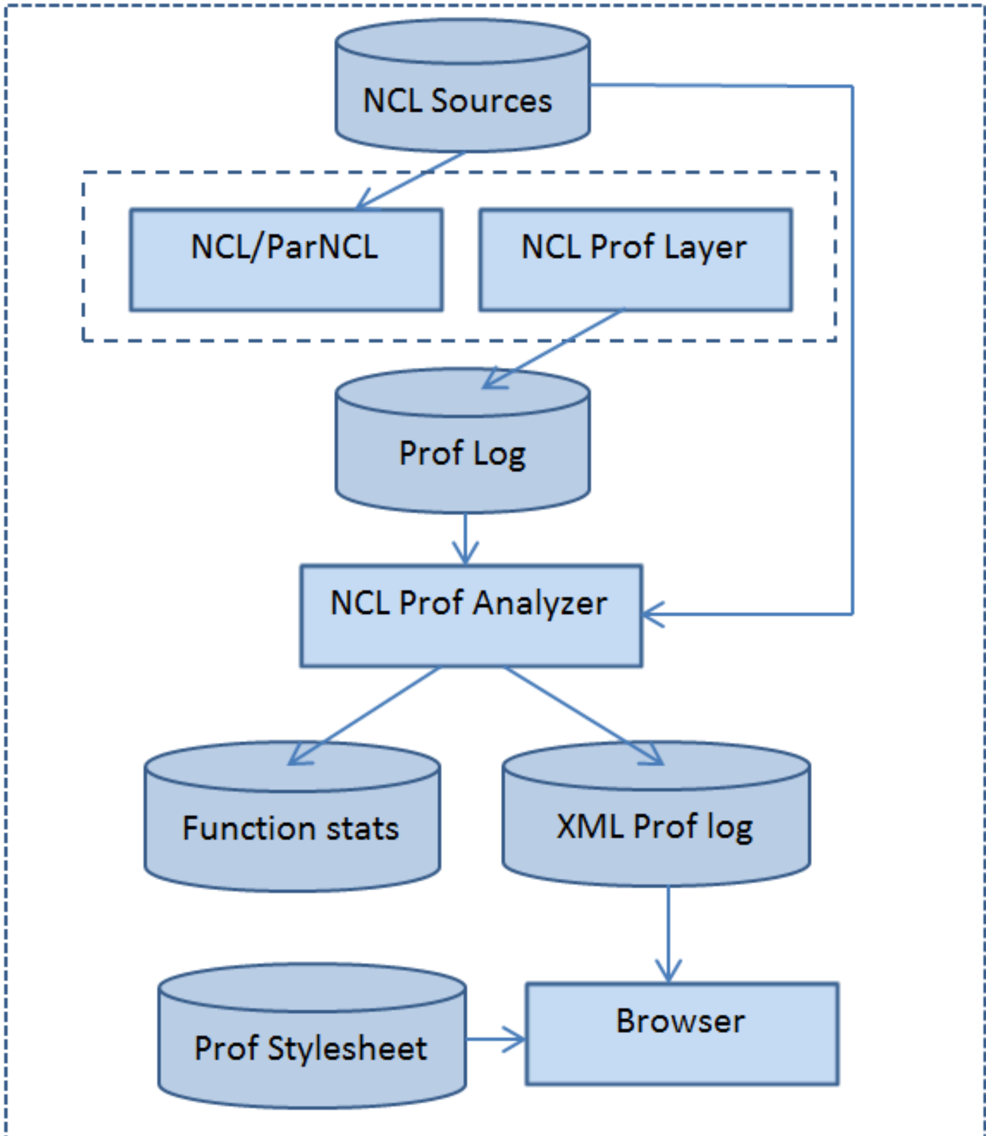
NCL architecture



ParNCL architecture



ParNCL Profiling Layer



- Manual modification of NCL script requires a lot of knowledge
  - Code semantics
  - NCL internals
- The profiling layer profiles scripts automatically
  - Records profile events at runtime
  - Useful for diagnostic pkgs
- Event analyzer script creates XML file containing usage statistics
  - NCL Script lines color coded based on time taken
  - Browser can be used to view XML

ParNCL update

- ParNCL supports addfiles(), NCL's multi-format file reader
  - Time slices of the variable read from file works
  - Need to test all corner cases
  - Only reads NetCDF for now (using PNetCDF).
- ParNCL supports calculating vorticity
  - Based on the current implementation in ParCAL
  - Specifically, duplicating function of uv2vrG_Wrap()
  - Adding support for other variations should be trivial from ParNCL perspective
- Simple math operations on distributed multidimensional variables work
  - abs, cos, asin, atan*, cos, exp, fabs, floor, log
  - Need to test more cases

ParNCL update cont'd…

- Addition, subtraction of distributed multidimensional data works
- Scaling a distributed multidimensional array by a scalar works
- Build changes
  - Can build serial and parallel versions separately
  - Can disable profiling layer at build time and runtime

ParVis is providing immediate help with task-parallel versions of diagnostic scripts using *Swift*

- **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.*

swift

Progress in Task-parallel diagnostics: Swift and AMWG

- **Swift-based AMWG diagnostics released to community!**
  - Officially part of version 5.3 of AMWG released in Feb, 2012.
  - Used daily at NCAR
  - Installed on Lens, the DAV cluster at OLCF.
  - Initial tests of using Pagoda (parallel) in place of NCO (serial)

- Swift package recent developments
  - Working on parallel invocation of multiple MPI and OpenMP applications (for using Pagoda in AMWG)
  - Solved problems with scheduling on Eureka (the DAV cluster at ALCF. Improved scheduling on BG/P and Cray.

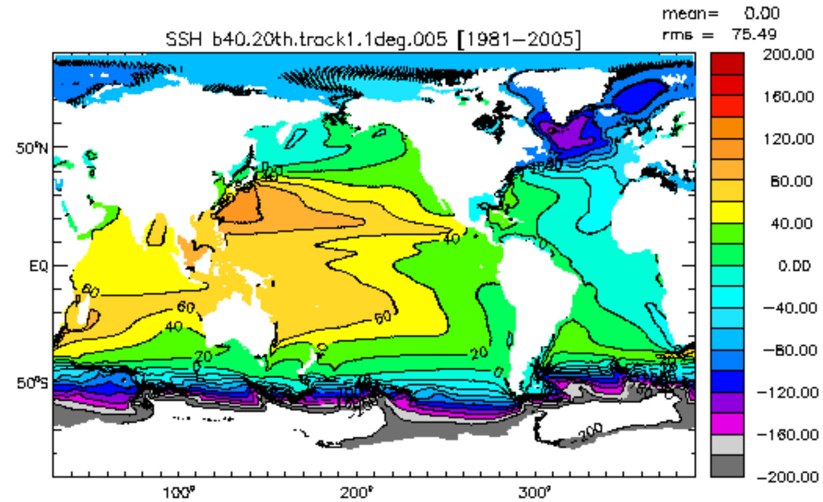
Swift version of OMWG diagnostics

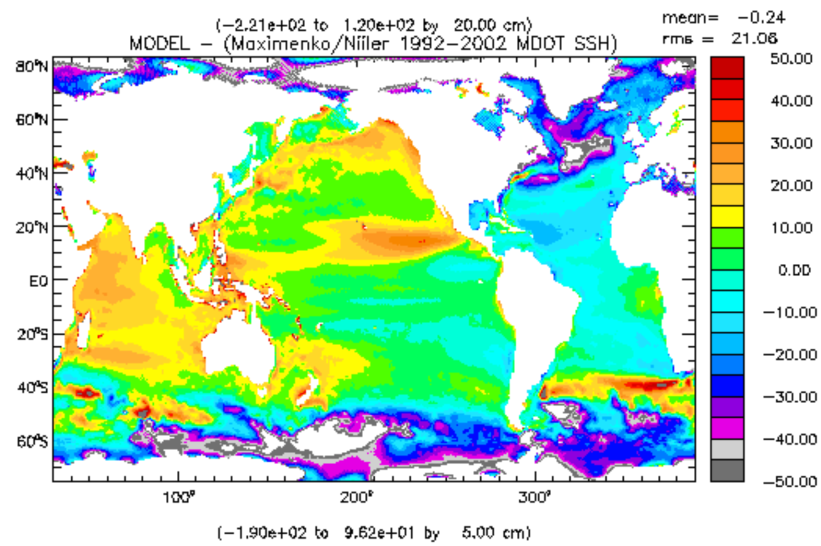
- OMWG diagnostics use non-free software.
- ParVis seeks to use/create only Free and Open Source Software.
- While building Swift version, convert to OMWG diags to all-NCL
  - 87 scripts converted from IDL to NCL
  - All three top-level OMWG control scripts modified to run NCL-based scripts exclusively
  - Graphics appear very similar with identical color table and level spacing to IDL graphics
  - Hope you saw Dave Brown's talk at OMWG at 9:30am today!
- Swift version ready for release.

Tutorial on both AMWG and OMWG swift-based scripts this evening at 6pm Aspen/Blue Spruce room.

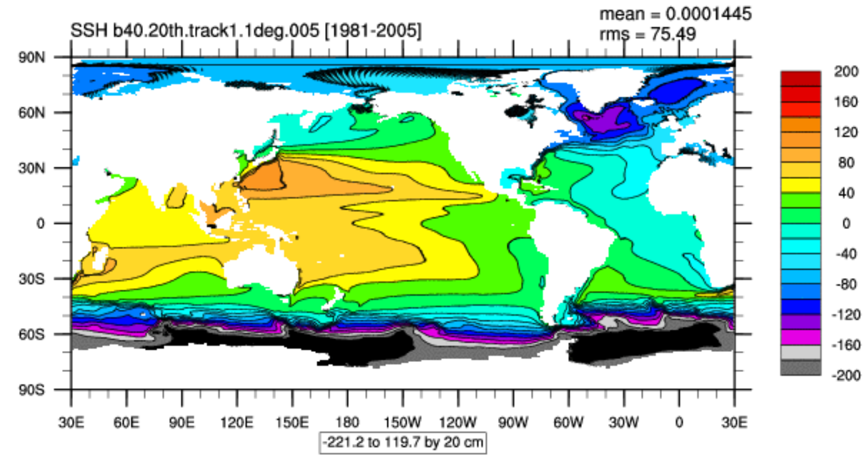
OMWG diagnostics: Sea Surface Height

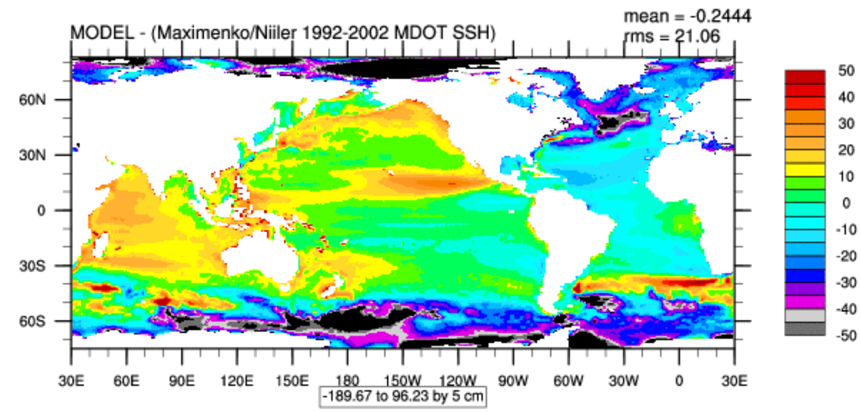
Original

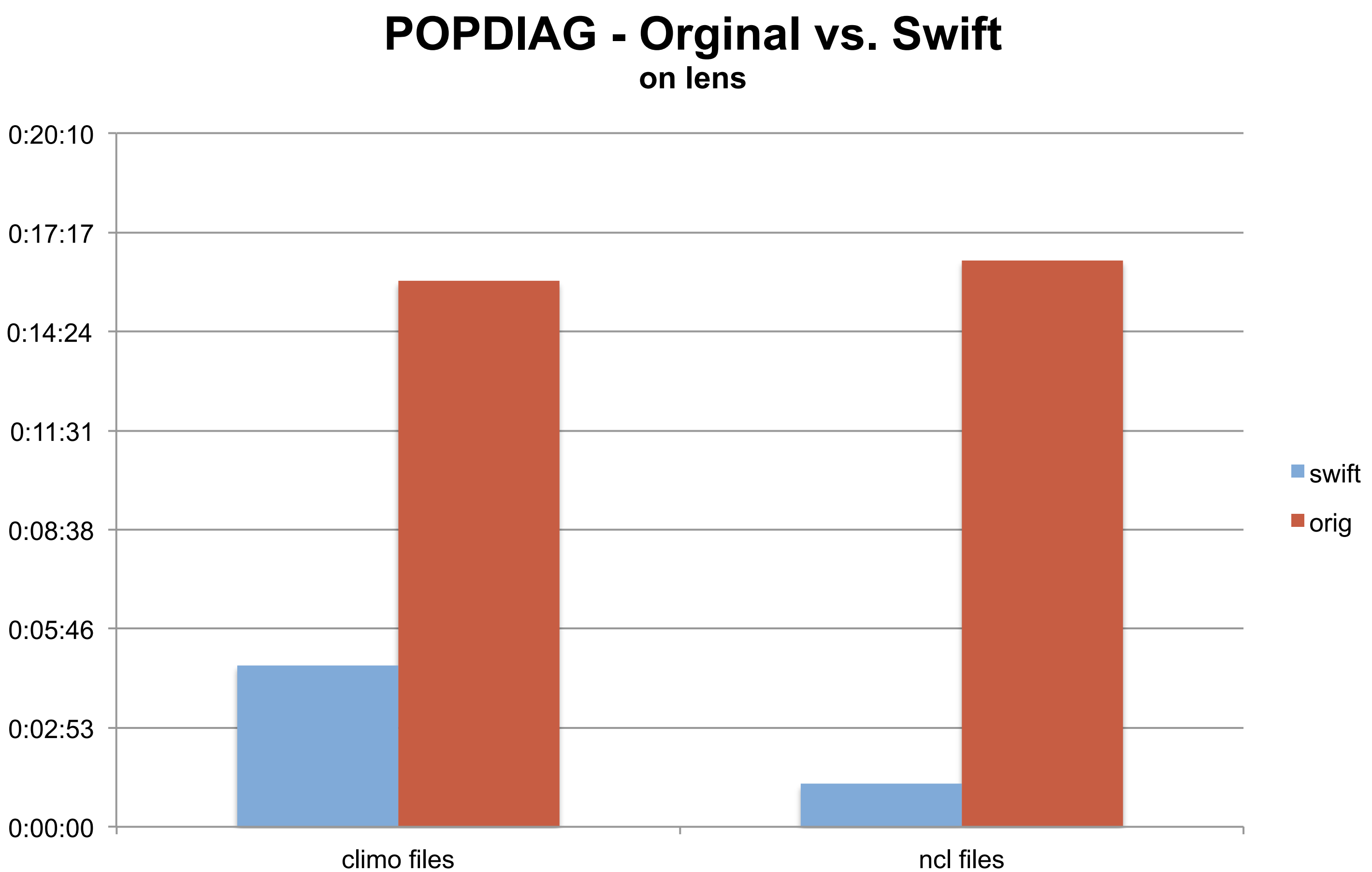




NCL







4 nodes – 8 tasks per node maximum – lens at ORNL
climo: averaging and other pre-processing; HPSS access and image conversion not included;10-year comparison

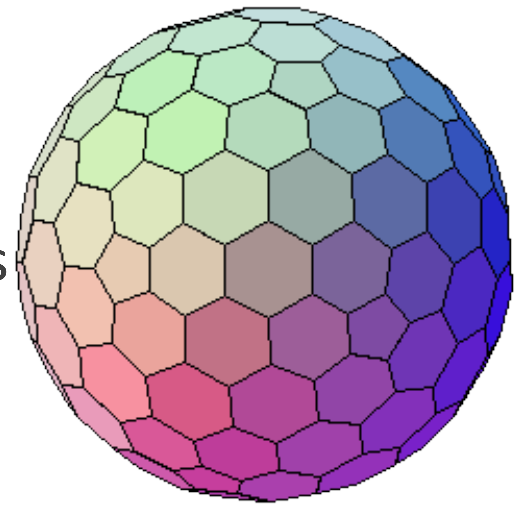
Cloud Computing paradigms: MapReduce

- Working "Streaming Hadoop" prototype kernels for averaging
  - Testbed kernel for probability density function (PDF) estimation implemented; applied to problem of time-evolving PDF f(X,t) estimation
    - Publication: "Visualizing climate variability with time-evolving probability density functions, detecting it with information theory," appeared in Workshop on Data Mining in Earth System Science, ICCS 2012
- FutureGrid (cloud computing resource) project granted, received allocation to do scalability tests and performance studies
- Paper "Mapping Climate Data Analysis onto the Map Reduce Programming Pattern" in preparation.

Using GPUs: Interactive Visualization of Large Geodesic Grid Data

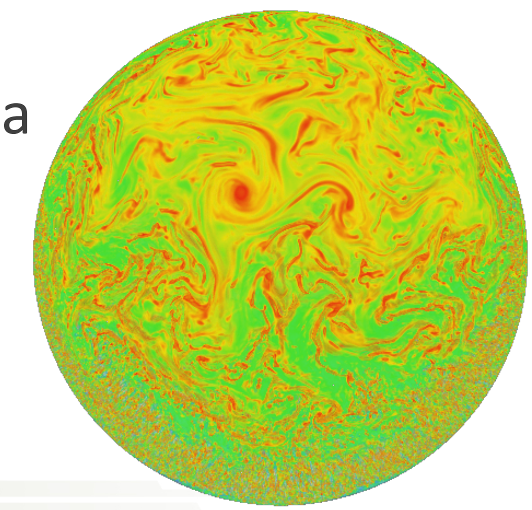
Kwan-Liu Ma, UC Davis

Existing 3D visualization solutions:



- Require a pre-partitioning of each hexagonal cell into multiple tetrahedral cells.
- Do not take advantage of latest GPU features
- Do not offer high-quality rendering

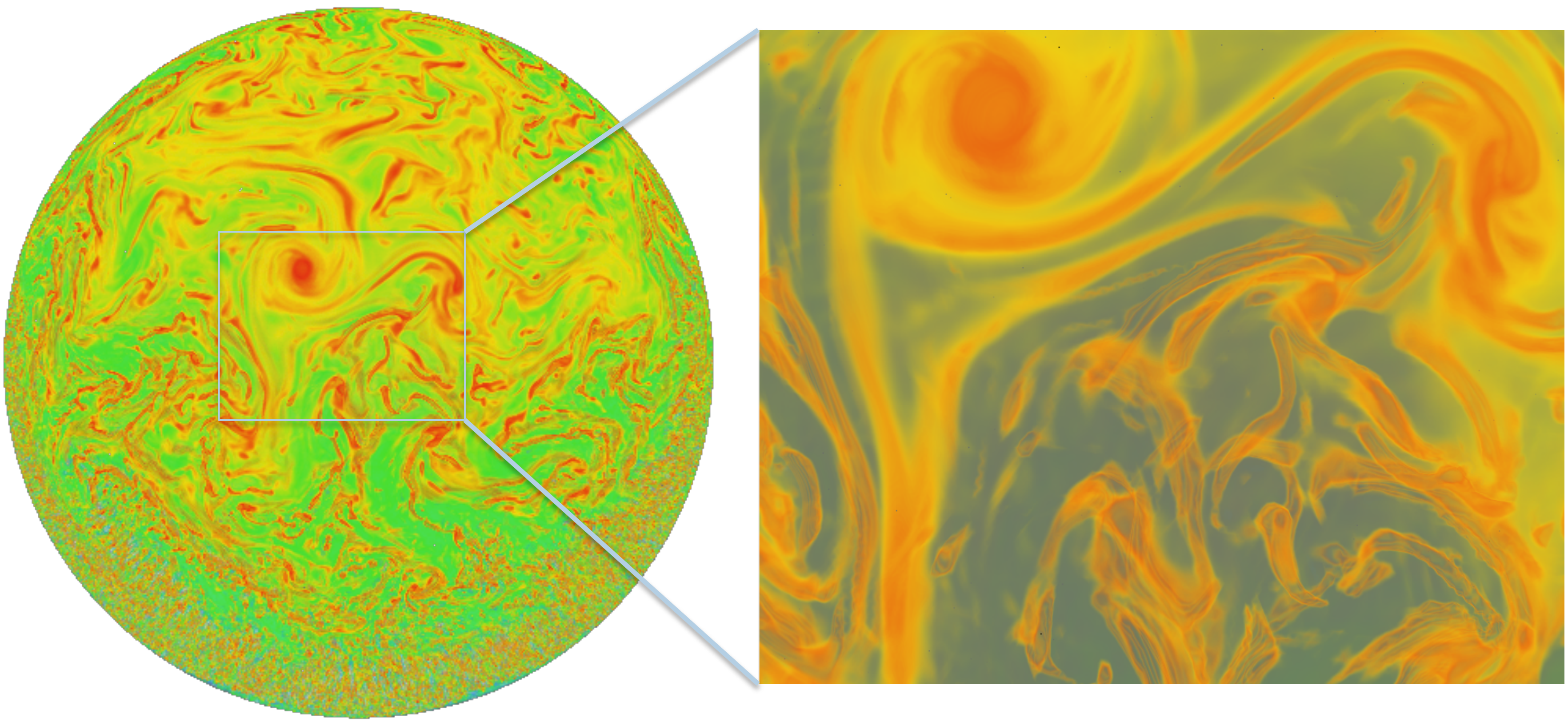
The UC Davis team seeks to provide:



- Advanced visualization of hexagonal grid data
- High quality 3D rendering
- GPU acceleration and parallelization to support Interactive interrogation

Interactive Visualization of Large Geodesic Grid Data

Kwan-Liu Ma, UC Davis



Data: CSU GCRM

Compression can help with growing data sizes

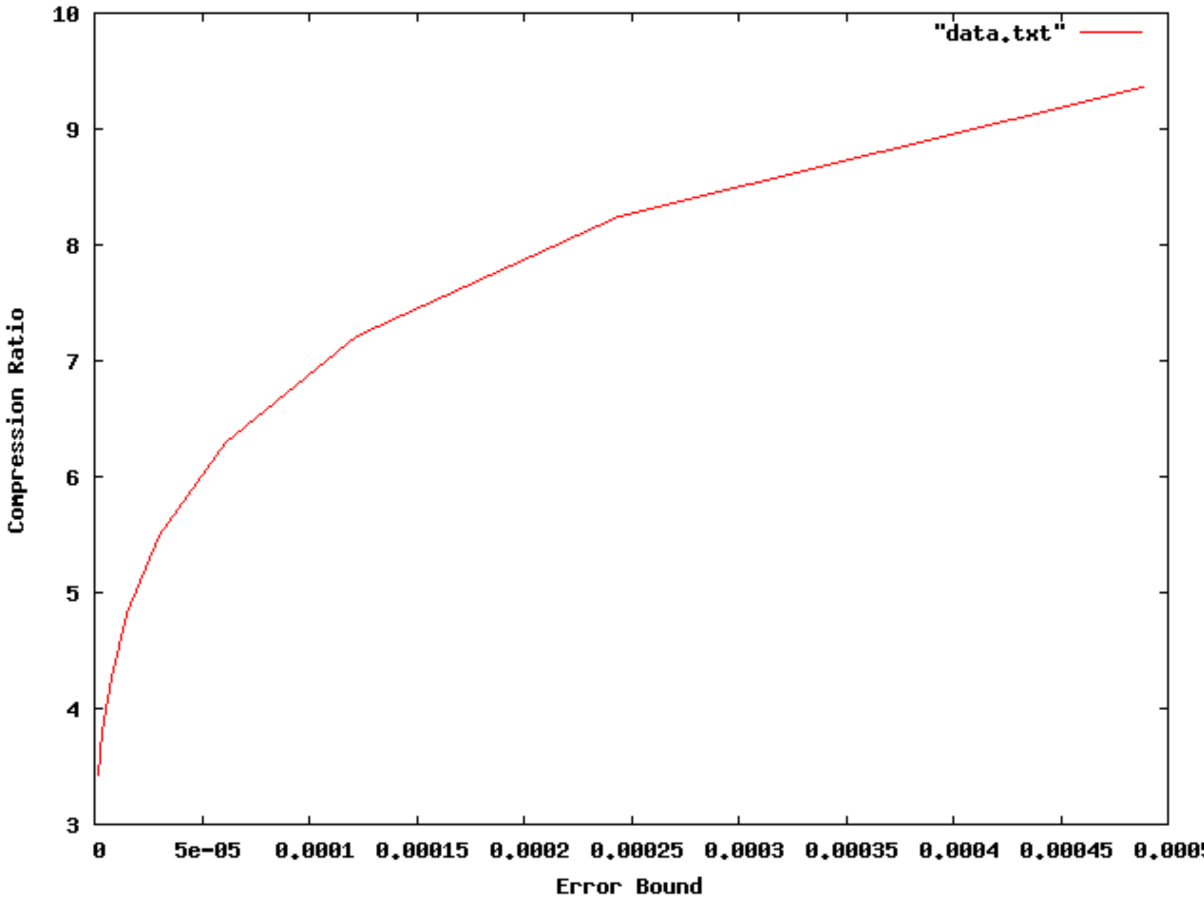
- Completely random data can not be compressed without information loss but many climate output fields are smooth, not random.

- 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: Long run of 0.10 POP at NCAR has to output less data because can't fit on disk.

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

Experimenting with incorporating compression in to PNetCDF

- Fetch compressed data through MPI-IO
- Advantages
  - Reduce disk overhead
  - Reduce communication overhead
- Disadvantage
  - Challenging when PnetCDF accesses and data compression are not aligned
  - Pipelining is difficult

- Implemented a proof of concept prototype and performed some preliminary measurements
  - Read a 2.7 gb netcdf file with uncomopressed data, 39.454 seconds, with compressed data, 27.429 second

What you can do…

- Let us know:
  - Where bottlenecks are in your analysis workflow. What NCL commands take too long or need too much memory?
  - What kind of post processing analysis would you like to do but can't?
  - When do you have to interpolate to some other grid as part of your analysis?

- Attend our session at 2012 Fall AGU.
  - IN008: Challenges in Analysis and Visualization of Large Earth Science Data Sets.
  - Conveners: Robert Jacob, Dean Williams and Wes Bethel

- Check the website: trac.mcs.anl.gov/projects/parvis
  - Subscribe to ParVis announcement mailing list: parvis-ann
  - Watch for beta release of ParNCL/ParGAL at end of August!

MCT Update

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

- MCT-based CPL7 coupler used for all CCSM4/CESM1 CMIP5 integrations!

- MCT development is part of the multi-lab Climate Science for a Sustainable Energy Future (CSSEF) project.

MCT More Recent history

- 2011: Some divergence between ANL and NCAR repositories

- MCT 2.8.0 - Released April 30, 2012 (standalone version)!
  - Will be included in next release of CESM
  - Build system upgraded (thanks to Jim Edwards, NCAR)
  - New datatype in AttributeVector to speed up copies (thanks to Bill Sacks, NCAR)
  - ANL and NCAR repos in sync.

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.

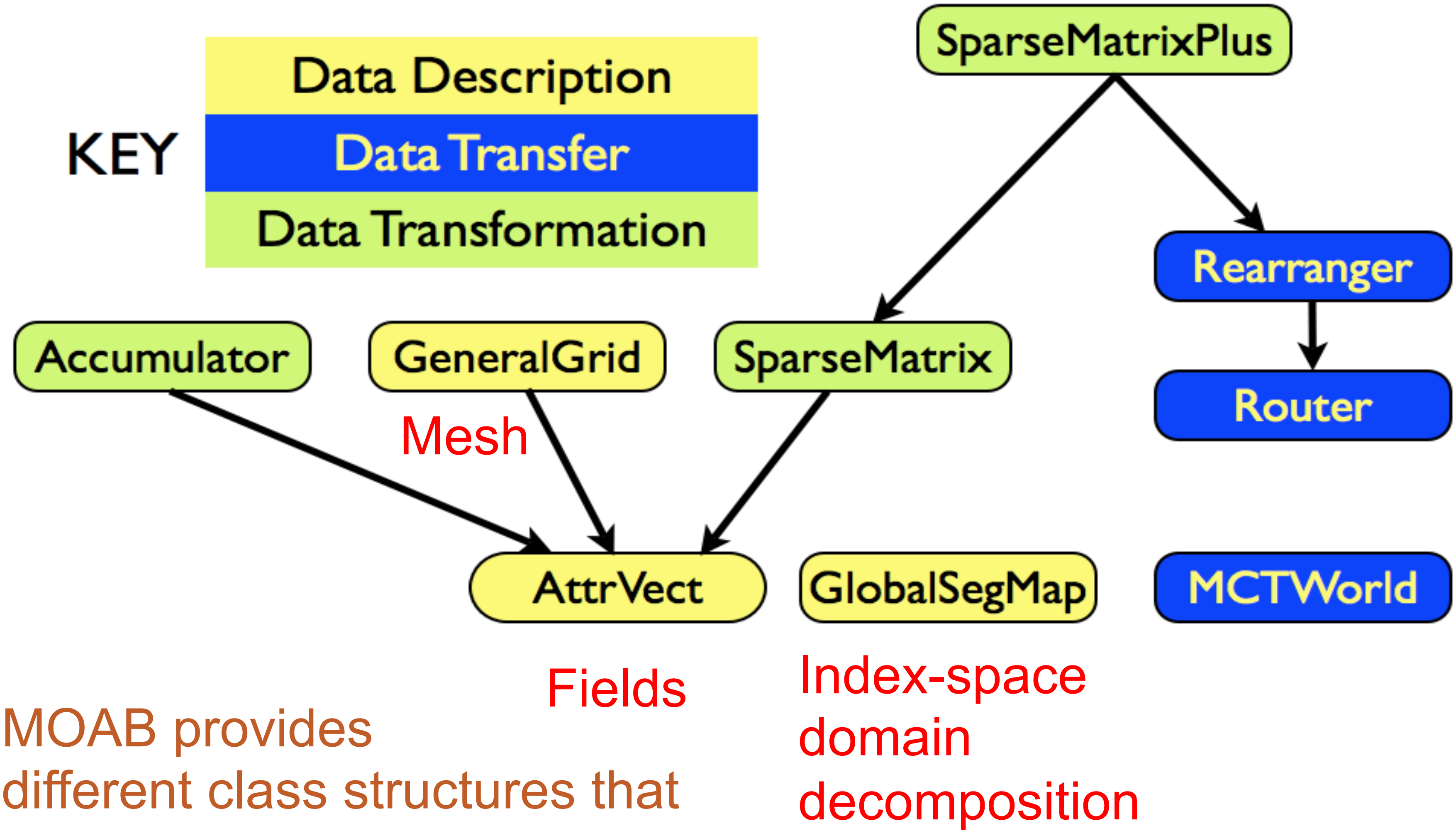
- Development of MPAS-Ocean
  - Need to retain information about unstructured grids for interpolation weight calculation.

- 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.

Review: MCT Classes



MOAB provides different class structures that define mesh, fields, and domain decomposition

AttrVect (Legacy MCT)

```
type AttrVect
      type(List) :: iList
      type(List) :: rList
      integer,dimension(:,:),pointer :: iAttr
      real(FP) ,dimension(:,:),pointer :: rAttr
end type AttrVect
```

- Stores pointwise collections of REAL (INTEGER) fields, or *attributes*, indexible by string tags in iList (iList)
- Key methods:
  - Create/destroy: init(), clean()
  - Query: length - lsize(), # REAL/INTEGER attributes - nIAttr()/ nRAttr(), names of attributes
  - Manipulate: copy(), zero(), append attributes, Import/Export indivudual attributes, sorting,, cross-indexing of attributes

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

iMesh-AttrVect test program

! Initialize MCT (Default 3-D--but empty--iMesh instance created:

```
call MCTWorld_init(1, MPI_COMM_WORLD, comm1, 1)
```

! Initialize MCT AttrVect:

```
call AttrVect_init(av1, rList='field1:field2', &
                   lsize=avsize)
```

! Query embedded iMesh instance to determine dimensionality:

```
call iMesh_getGeometricDimension(%VAL(ThisMCTWorld%mesh), &
                                 geom_dim, ier)
```

! iMesh query function on the new Av tag handle

```
call iMesh_getTagName(%VAL(ThisMCTWorld%mesh), &
                      %VAL(av1%rtagh(1)) , &
                      tagname, ier, %VAL(10))
```

Other AttrVect methods from previous slide also available as-is

Near term plans for MCT and MCT-MOAB

- Build test program for MCT Router initialization.
- 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.

Questions?