Supporting Global Cloud Resolving Model Simulations at NERSC

Prabhat
Lawrence Berkeley National Lab

DOECGF
April 21, 2009

Outline

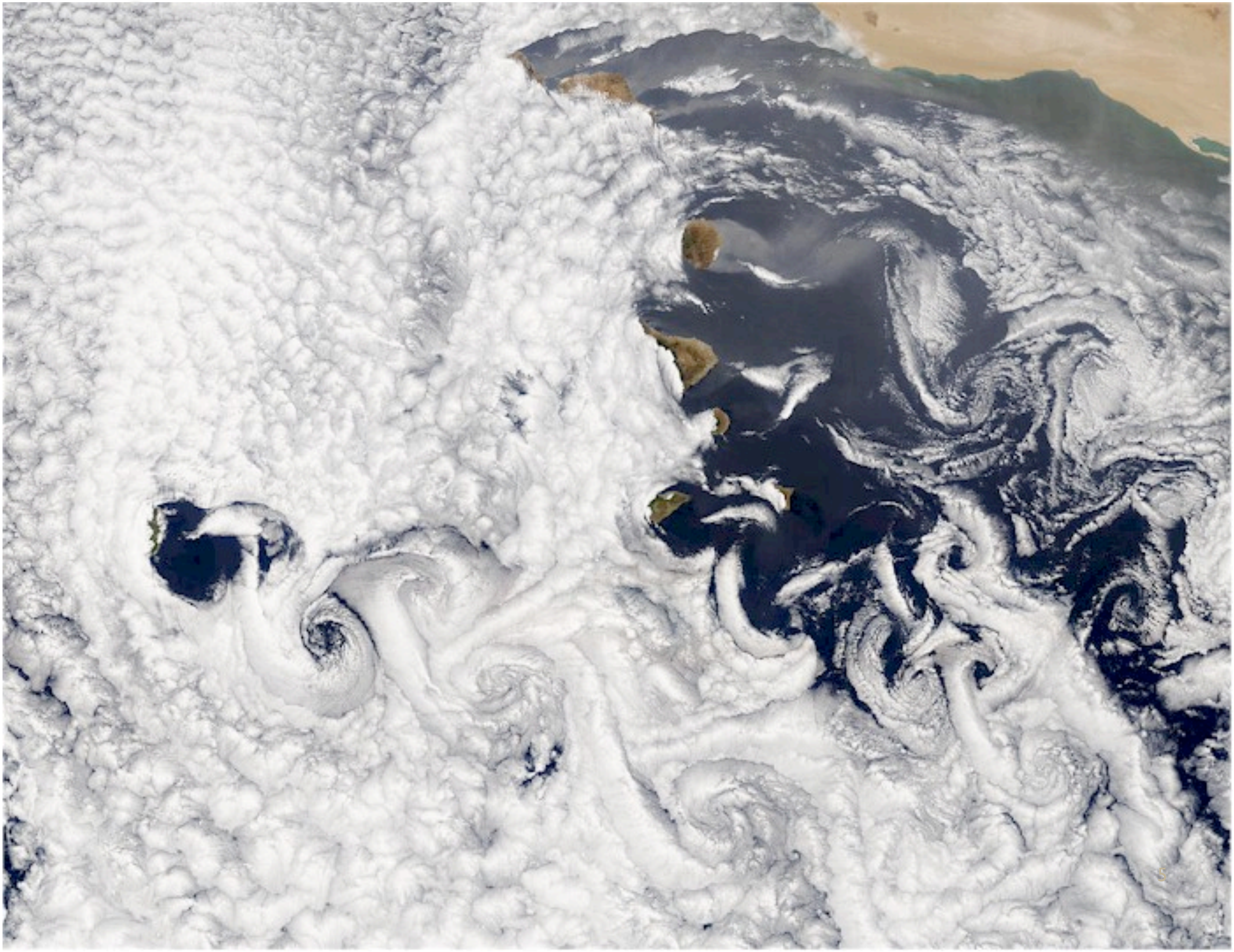
- Science
  - Global Cloud Resolving Models
  - Geodesic Grids
- Our Work
  - Efficient Parallel I/O
  - Data Model
  - Visualization

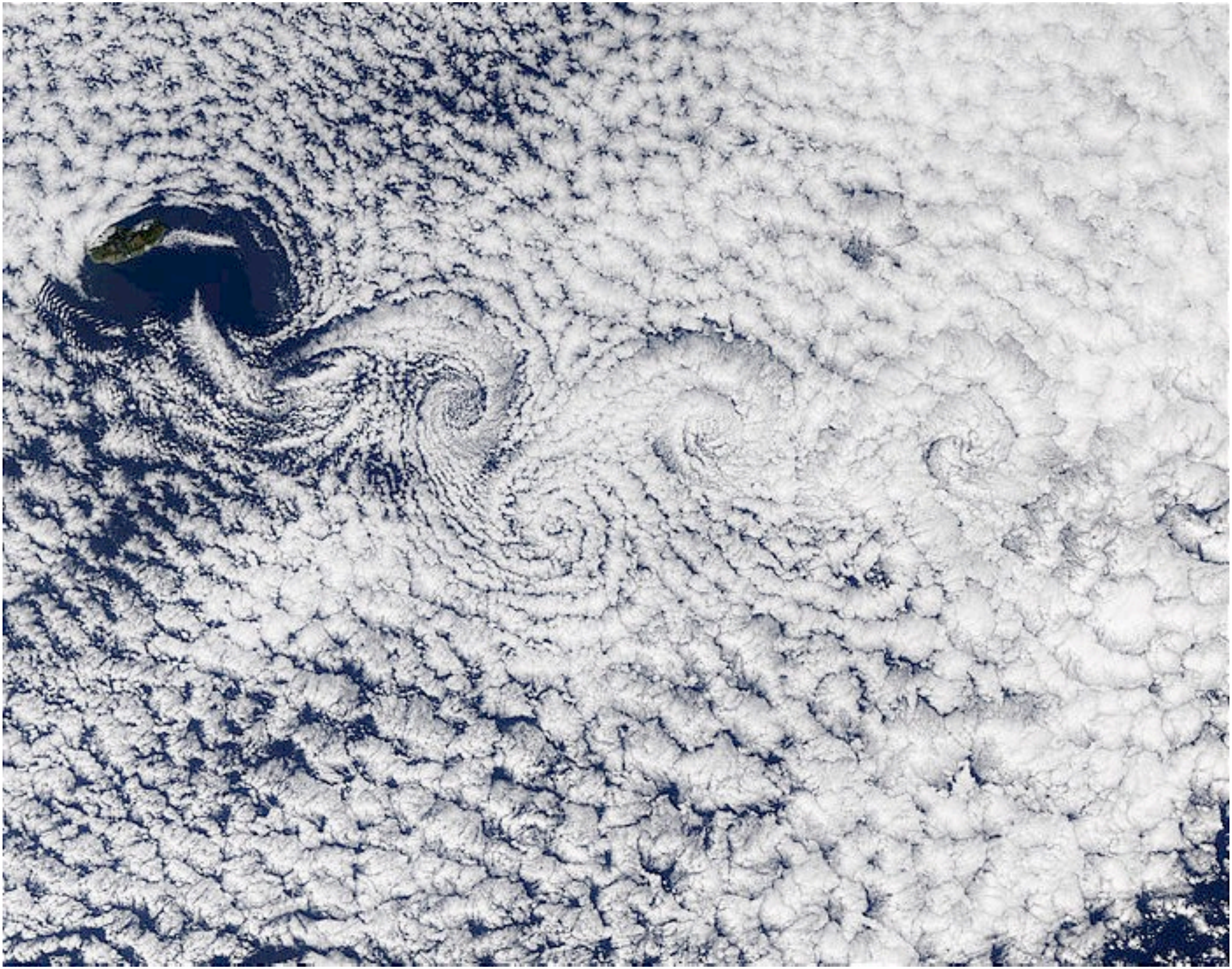
Outline

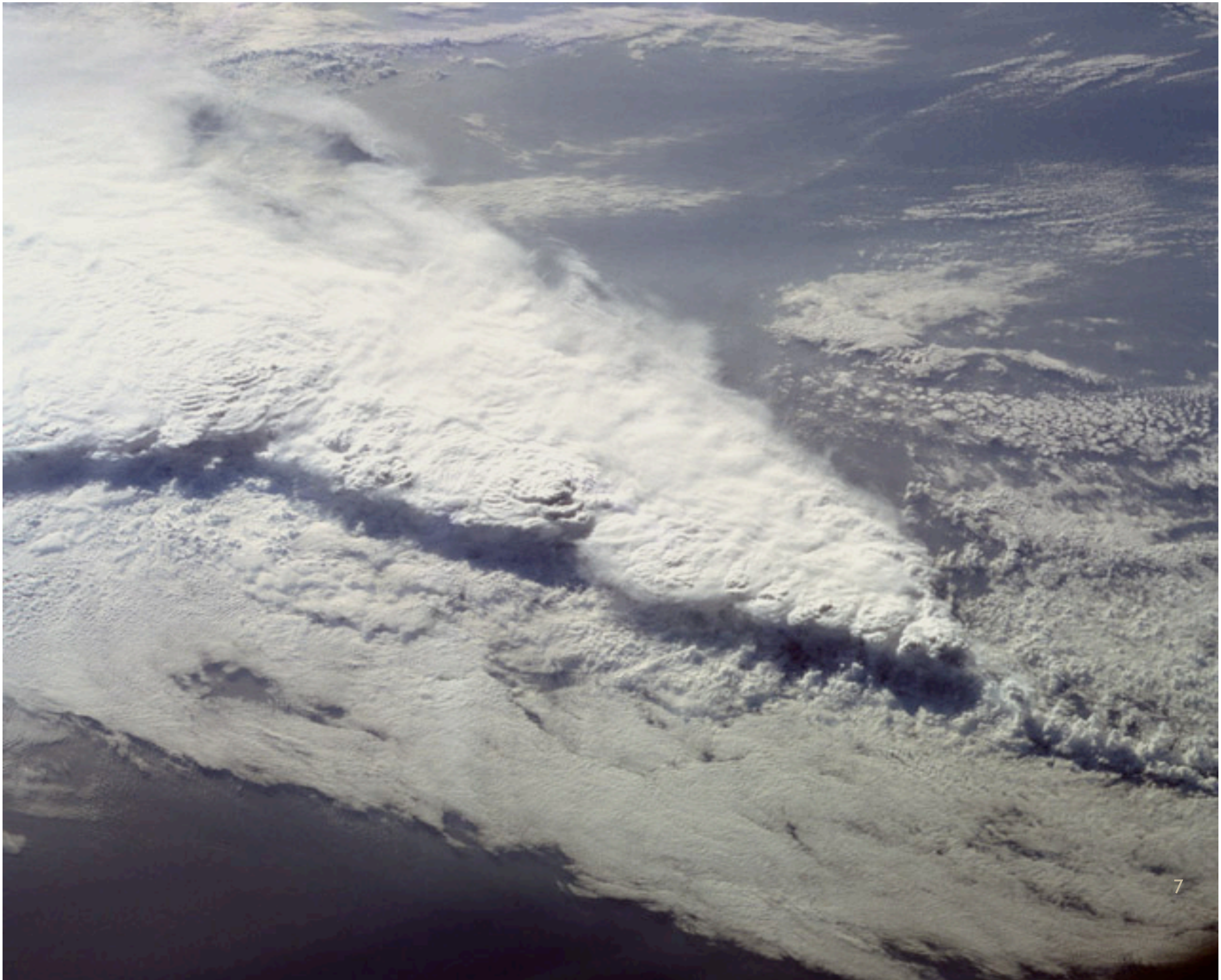
- Science
  - Global Cloud Resolving Models
  - Geodesic Grids
- Our Work
  - Efficient Parallel I/O
  - Data Model
  - Visualization

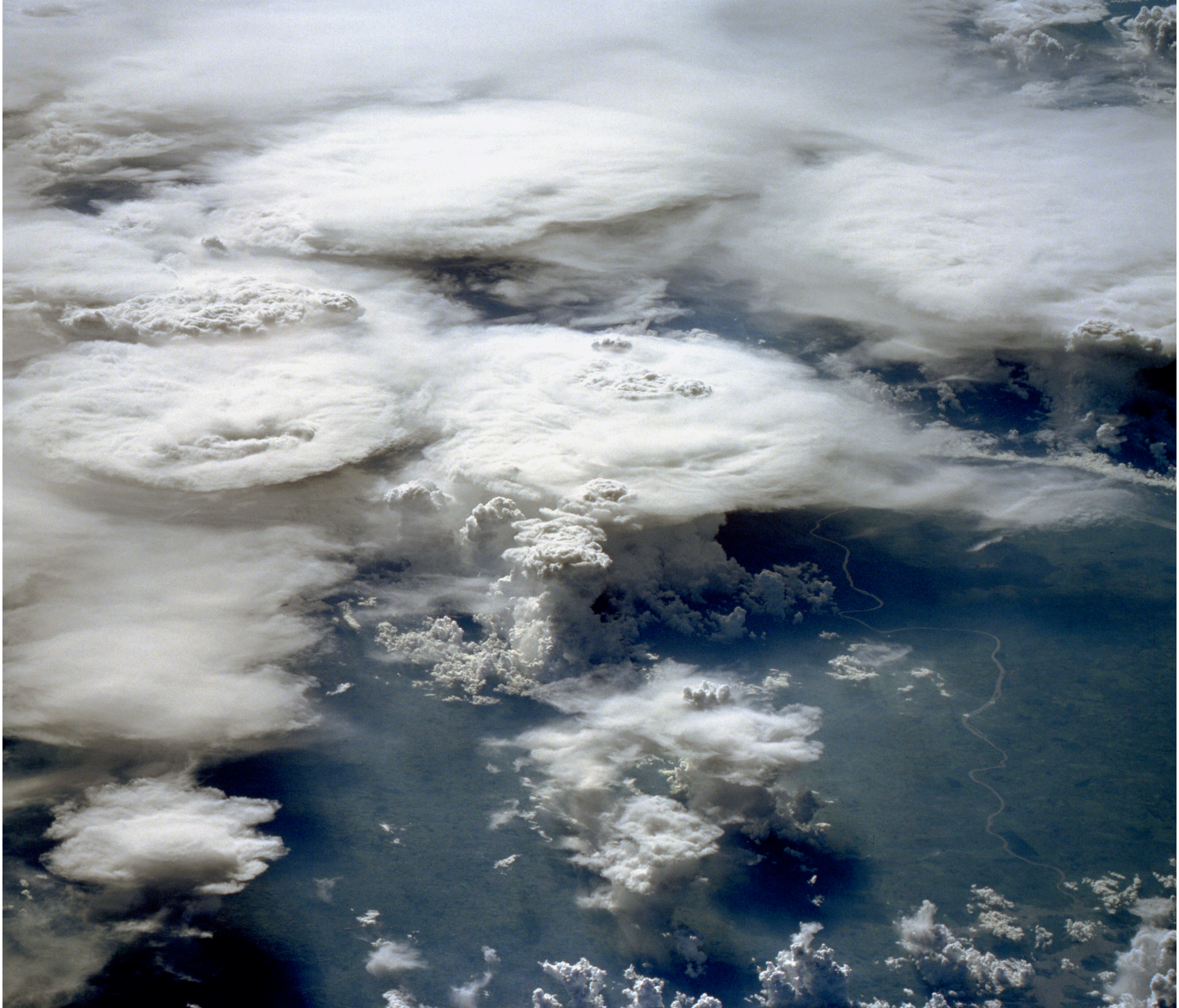
SciDAC/INCITE Projects

- Design and Testing of a Global Cloud Resolving Model (GCRM)
  - *Dave Randall, CSU*
- Community Access to Global Cloud Resolving Model Data and Analyses
  - *Karen Schuchardt, PNNL*





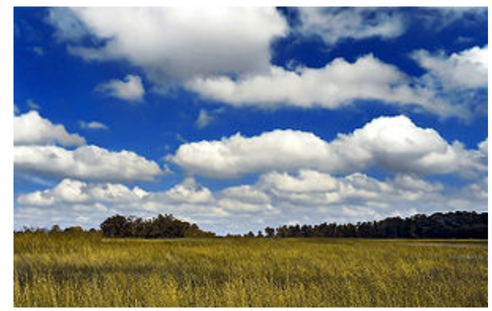




Cloud Resolving Models

- Shown to agree with radar observations
- Existing models use cumulus and stratus cloud parameterizations
- Cirrus clouds known to strongly influence weather patterns
    - Can only be resolved with fine grid resolution (< 4km)

cirrus



cumulus

stratus

Global Cloud Resolving Models

- Computationally expensive to extend a cloud-resolving model to a global model
  - Now possible on high-end systems like Franklin and Jaguar
- GCRM model will be verified using satellite, radar, and in-situ observations

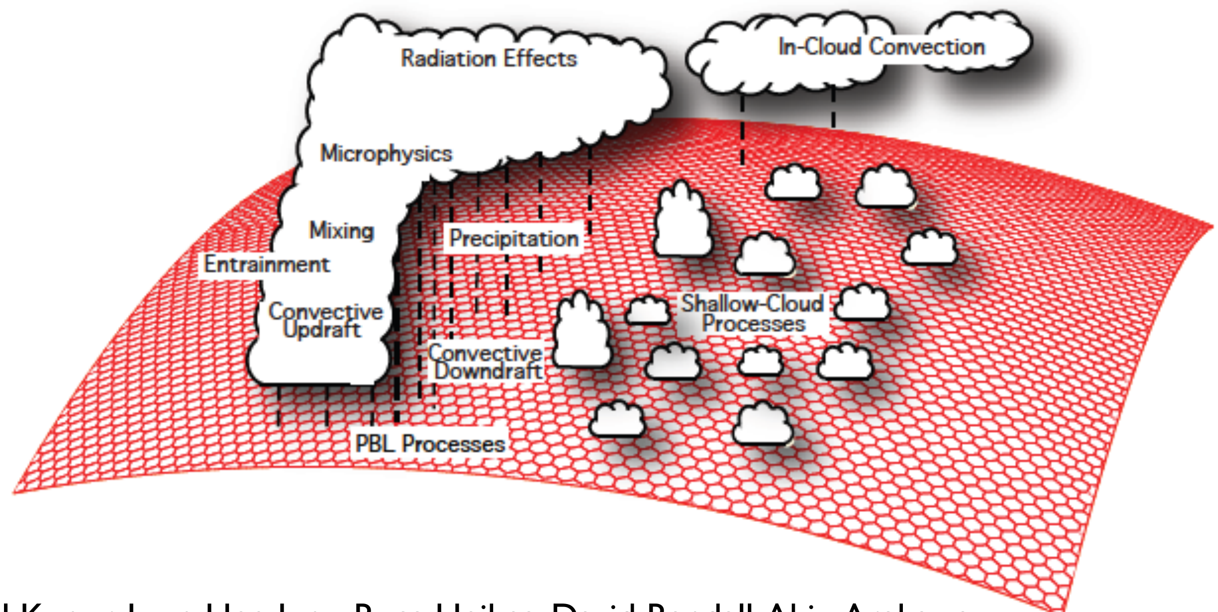
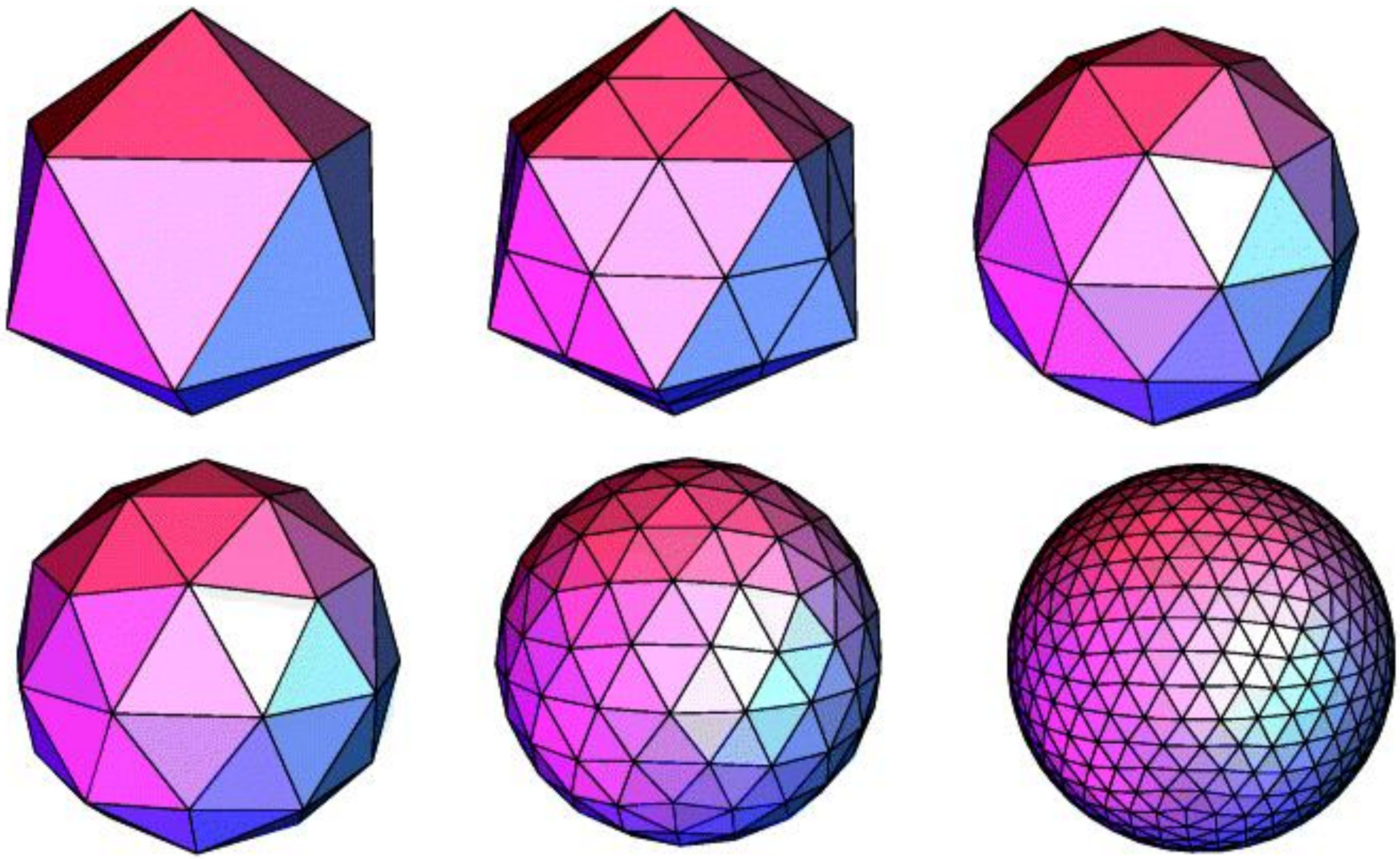


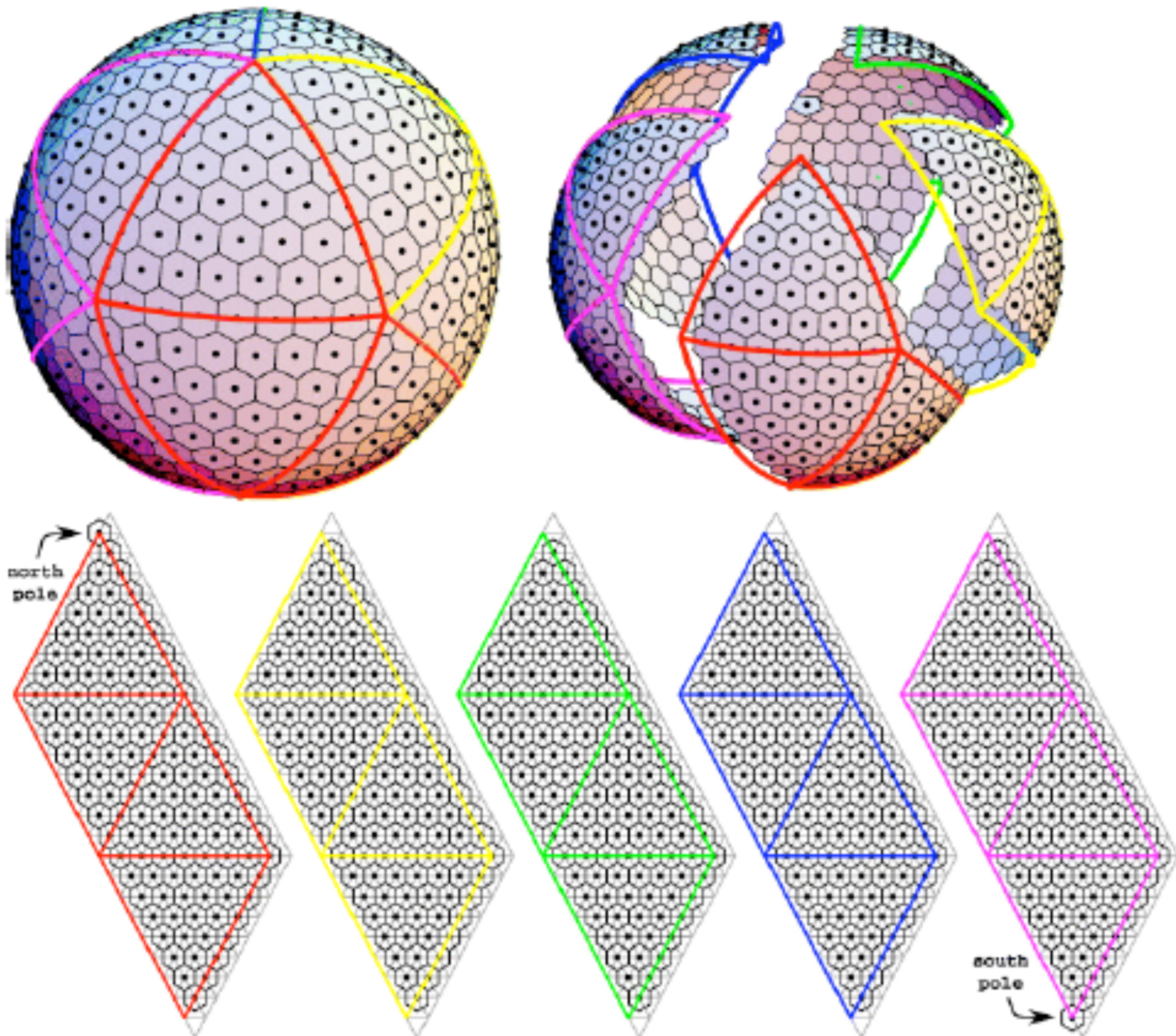
Figure from Celal Konor, Joon Hee Jung, Ross Heikes, David Randall, Akio Arakawa

Geodesic Grid



- Recursive subdivision of an icosahedron

Geodesic Grid



Geodesic Grid Resolution

R	Cell width (Km)	N
10	8	10,485,762
11	4	41,943,042
12	2	167,772,162
13	1	671,088,642

- $N = 10 * 2^{2R} + 2$
- ~17B cells for 25 levels at R=13

GCRM Runs

- Run at 4km resolution (R=11) or better
- Large Concurrency
  - 30K+ cores
  - Cray XT4 (Franklin/Jaguar)

GCRM Data

- ~20 variables
- 16-48 GB per variable
- Dump snapshots every simulated hour
- Overall
  - ~0.3-1 TB/snapshot
  - ~1-10 PB of simulation data for one simulated year

GCRM I/O requirements

- GCRM code
  - dump_buffer(“variable”, *array);
- Write to shared file
- I/O should not add more than 10% overhead
- *Need sustained shared write performance of 2+ GB/s*

PNNL's I/O API

- Implementation in pNetCDF (NetCDF-3)
- 2-phase I/O
  - Subset of compute nodes do aggregation and writes
  - Data shuffling (morton-ordering)
- Observed Collective Write Performance
  ~100-500 MB/s vs. 12 GB/s max

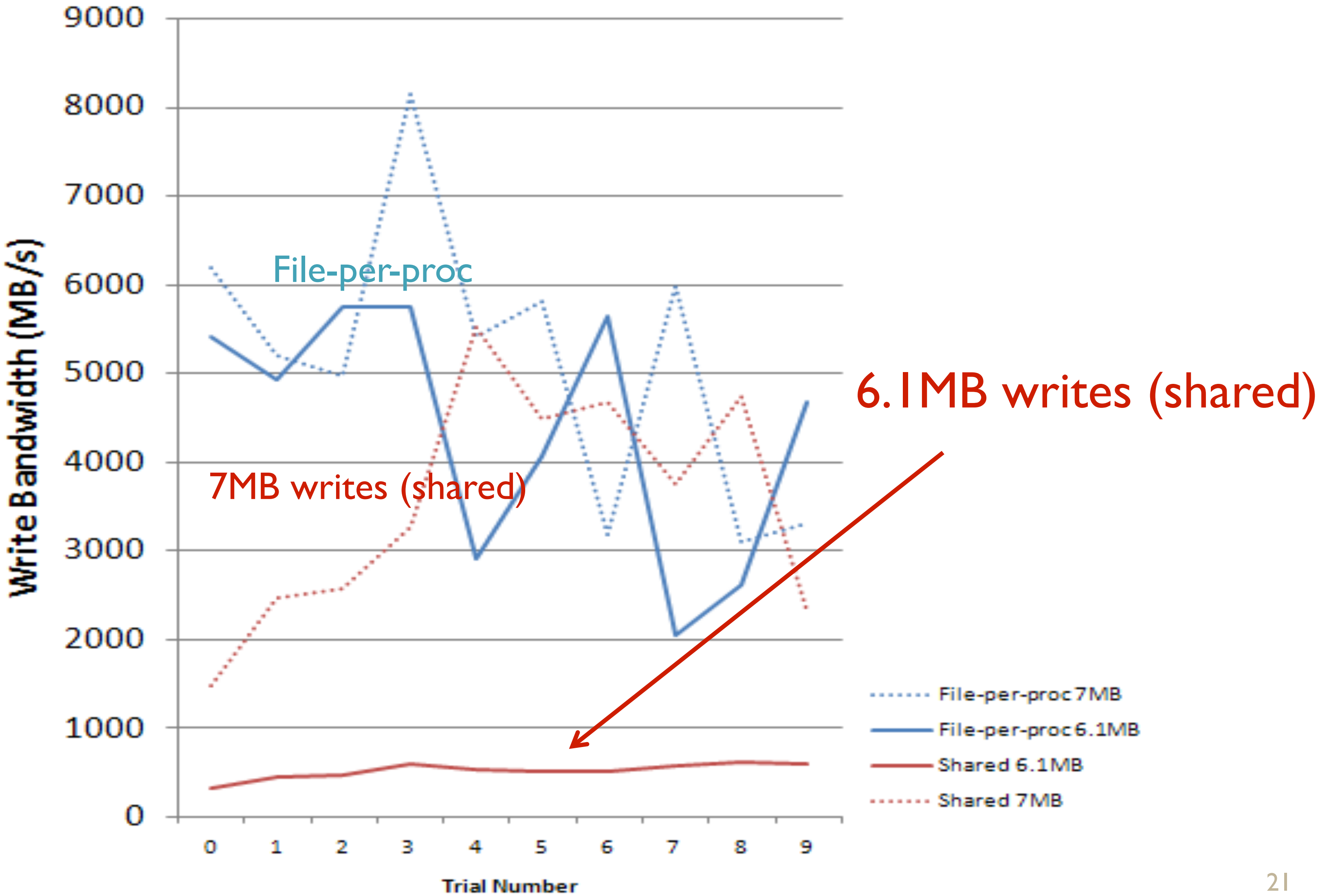
Outline

- Science
  - Global Cloud Resolving Models
  - Geodesic Grids
- Our Work
  - Efficient Parallel I/O
  - Data Model
  - Visualization

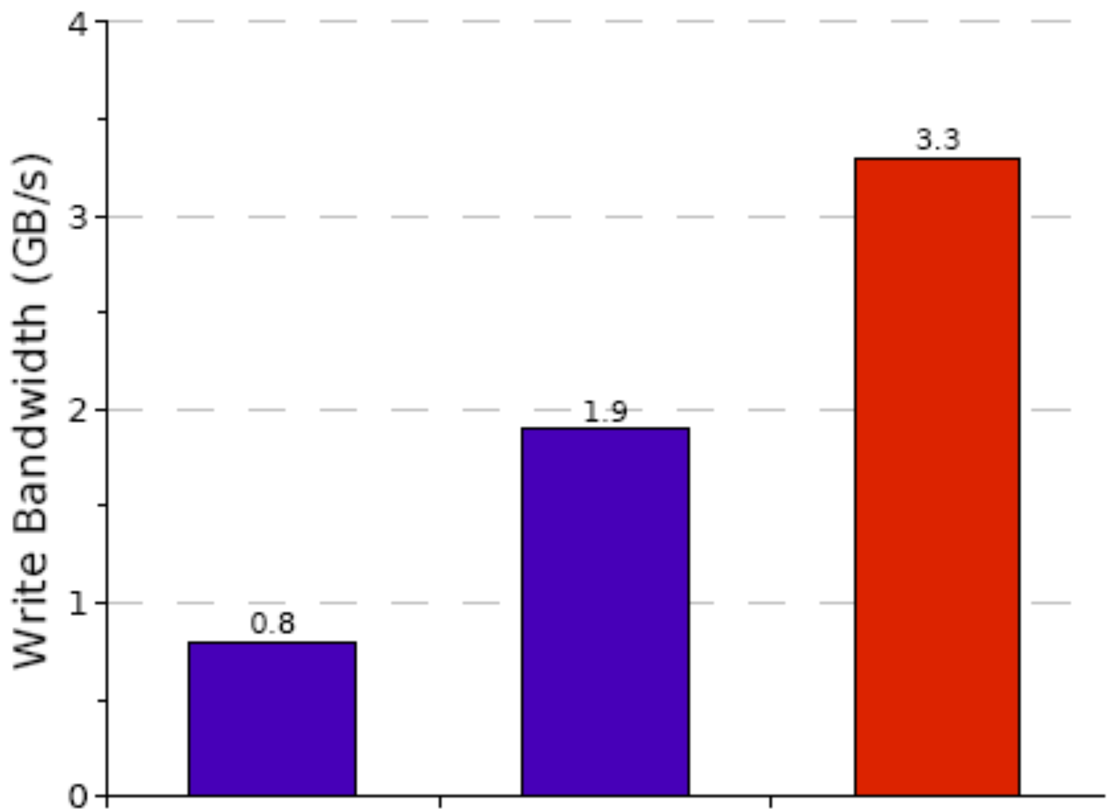
Troubleshooting Tools

- Used IOR to generate I/O patterns
- Used IPM to characterize system issues
  - Impossible to know what POSIX calls are made by the system stack
    - User App
    - pNetCDF/HDF5
    - MPI-IO
    - POSIX

GCRM IO pattern



Tuning the IO pattern



Baseline from PNNL

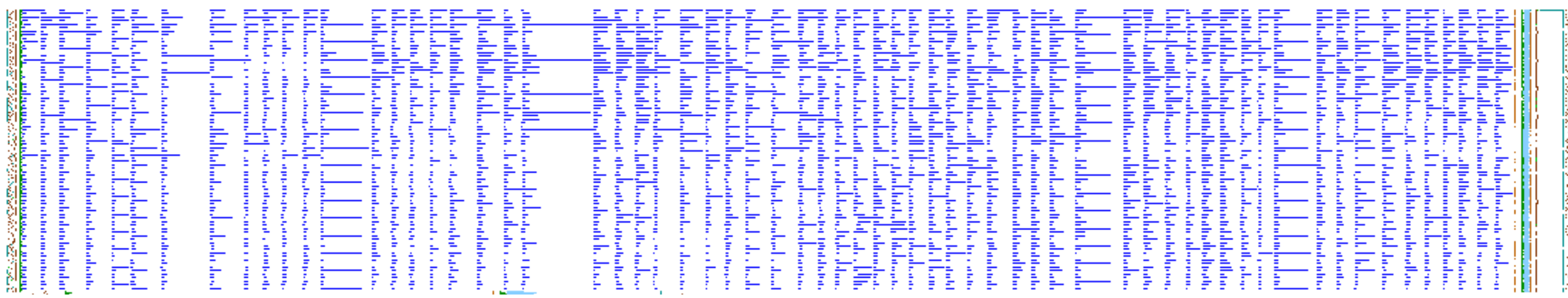
MPI-IO Calls reconfigured

Proof-of-concept IO pattern using MPI-IO in independent mode and 7MB transfers

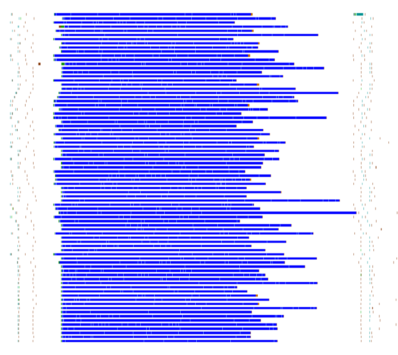
MPI-IO Performance Issues

Synchronous vs. Asynchronous Write Calls for Same IO Pattern

Cray's MPI-IO Implementation (1294 MB/s) ~ MPI-IO VFD collective mode



IOR POSIX Shared File (6535 MB/s) ~ MPI-POSIX VFD



Test Parameters	
Nodes/stripes:	80
Aggregate data:	40GB
Stripe width:	8MB
Write size:	8MB
Writes per node:	64

Key
Open
Read
Write
Seek
Close

Performance issues

- Poor performance when IO patterns do not align to lustre stripes
- 'Read-Modify-Write' semantics pulls stripe data all the way to the client
- Small (~10-100K) buffer writes
  - User level aggregation not working
- MPI-IO implementation issues
  - Conservative, MPI Barriers
  - 2-phase IO doesn't work
- Highly variable latency on a per-OST basis

Ongoing Developments

- NERSC/HDF5 collaboration
  - Make HDF5 lustre-aware
  - Add hooks to HDF5 tunable parameters
  - Pad/align chunks to stripe boundaries
- New Cray MPI-IO version
  - Improve 2-phase I/O implementation
  - Avoid user space solutions
- Hardware upgrades
  - 2x OSTs
  - Split /scratch filesystem in half
    - separate MDS
    - reduce contention

Outline

- Science
  - Global Cloud Resolving Models
  - Geodesic Grids
- Our Work
  - Efficient Parallel I/O
  - Data Model
  - Visualization

Data Model

- Metadata conventions for geodesic mesh
- All types of mesh variables/layout
  - cell/corner/edge centered
- Minimize duplicated data
  - unique vertices, corners, edges
- Ease-of-import for other vis/analysis apps
- Subsetting
- Current implementation in NetCDF-3
  - Future plans for NetCDF-4

```
dimensions:
	time = UNLIMITED ; // (1 currently)
	cells = 10485762 ;
	cellcorners = 6 ;
	corners = 20971520 ;
	celledges = 6 ;
	edges = 31457280 ;
	cellneighbors = 6 ;
	layers = 25 ;
	interfaces = 26 ;

variables:
	float grid_center_lat(cells) ;
		grid_center_lat:long_name = "Latitude of cell center" ;
		grid_center_lat:units = "radians" ;
	float grid_center_lon(cells) ;
		grid_center_lon:long_name = "Longitude of cell center" ;
		grid_center_lon:units = "radians" ;
	float grid_corner_lat(corners) ;
		grid_corner_lat2:long_name = "Latitude of unique cell corners" ;
		grid_corner_lat2:units = "radians" ;
	float grid_corner_lon(corners) ;
		grid_corner_lon2:long_name = "Longitude of unique cell corners" ;
		grid_corner_lon2:units = "radians" ;
```

```
int cell_neighbors(cells, cellneighbors) ;
        cell_neighbors:long_name = "List of neighbors to this cell" ;
        cell_neighbors:units = "unitless" ;
int cell_corners(cells, cellcorners) ;
        cell_corners:long_name = "Indices of cell corners" ;
int cell_edges(cells, celledges) ;
        cell_edges:long_name = "Indices of cell edges" ;

float pressure(time, cells, layers) ;
        pressure:long_name = "Pressure" ;
        pressure:units = "Pa" ;
        pressure:coordinates = "grid_center_lat grid_center_lon" ;
float geopotential(time, cells, interfaces) ;
        geopotential:long_name = "Geo Potential" ;
        geopotential:units = "m**2/sec**2" ;
        geopotential:coordinates = "grid_center_lat grid_center_lon" ;
float u(time, corners, layers) ;
        u:long_name = "U wind component at cell corners" ;
        u:units = "m/sec" ;
float v(time, corners, layers) ;
        v:long_name = "V wind component at cell corners" ;
        v:units = "m/sec" ;
float wind(time, edges, layers) ;
        wind:long_name = "Wind component at faces" ;
        wind:units = "m/sec" ;
```

Outline

- Science
  - Global Cloud Resolving Models
  - Geodesic Grids
- Our Work
  - Efficient Parallel I/O
  - Data Model
  - Visualization

Vis Requirements

- Handle large data
  - Existing tools don’t scale
- Rich vis/analysis feature set
- Remote vis capabilities
  - Keep data at NERSC
- Deploy on workstations/laptops, OS

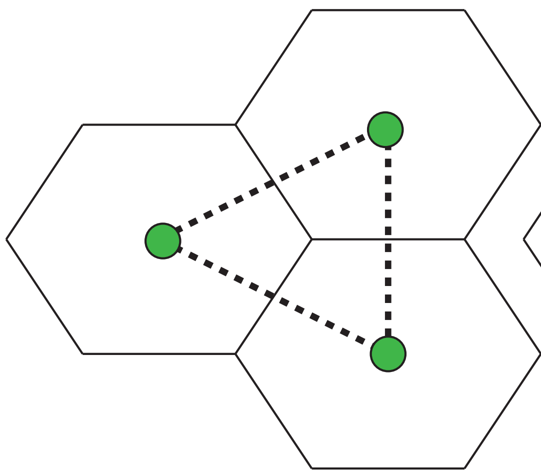
Visualization

- VisIt plugin directly imports Geodesic grid
  - Serial version
- Fully supports GCRM data model
  - All mesh types and variables are supported
  - Different tessellations/meshes are created

http://vis.lbl.gov/~prabhat/Incite19/

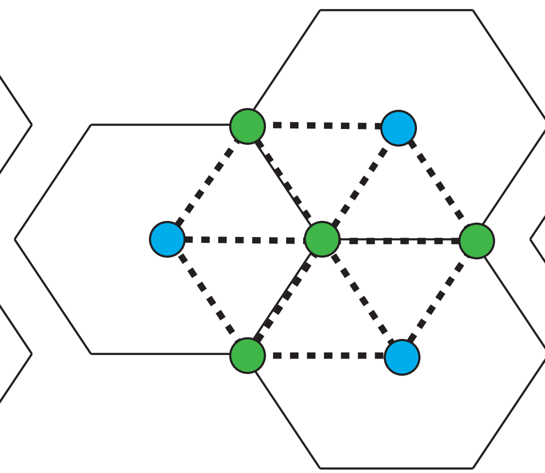
2D/3D Grid Tessellation

Variable defined at
cell centers (face)



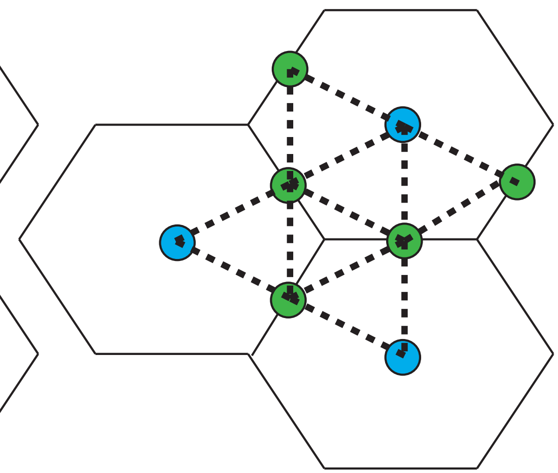
Straightforward. No interpolations.

Variable defined at
cell corners (vertex)

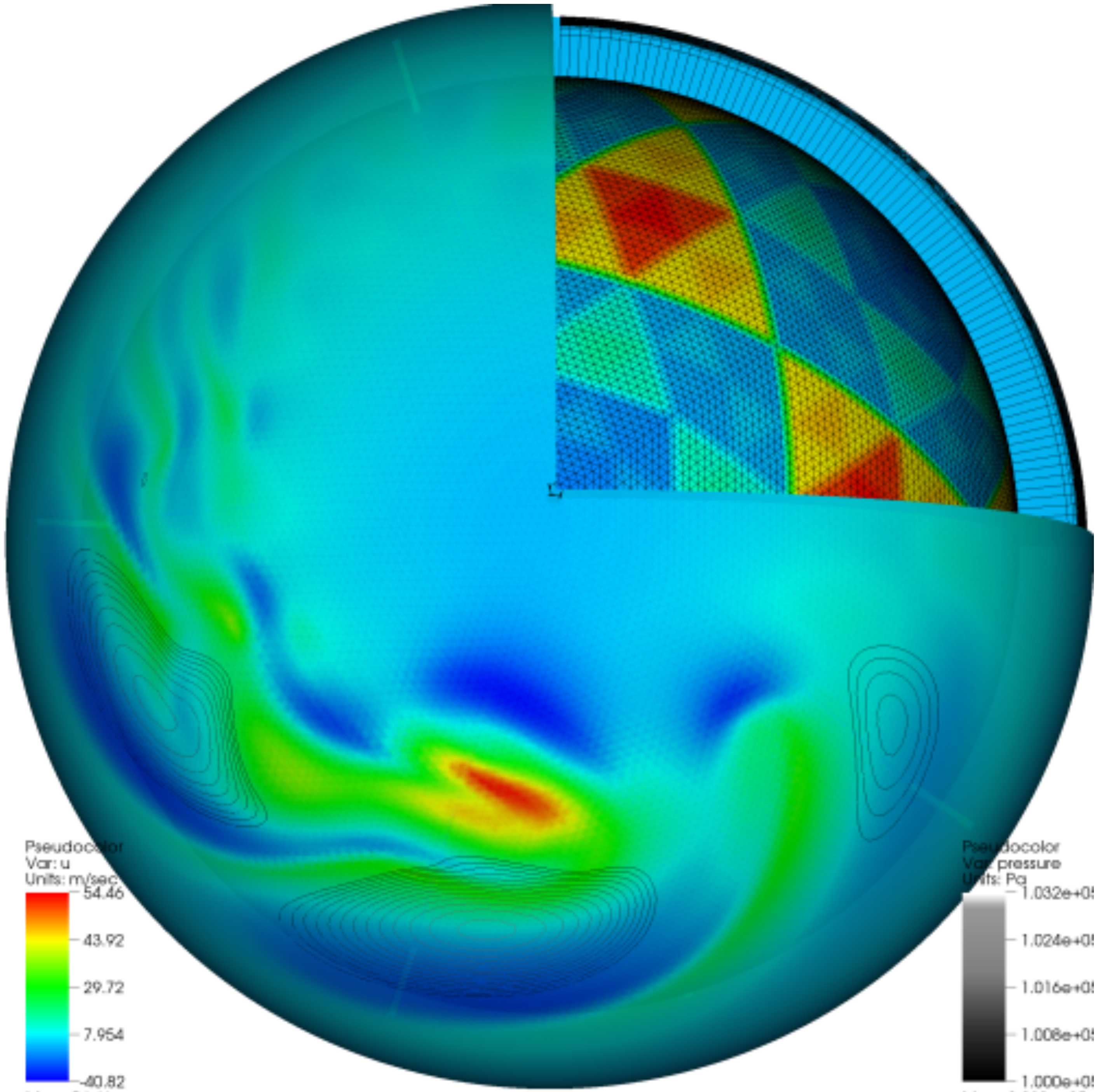


Requires interpolation
to cell centers (blue points)
using information from
surounding corners (green
points)

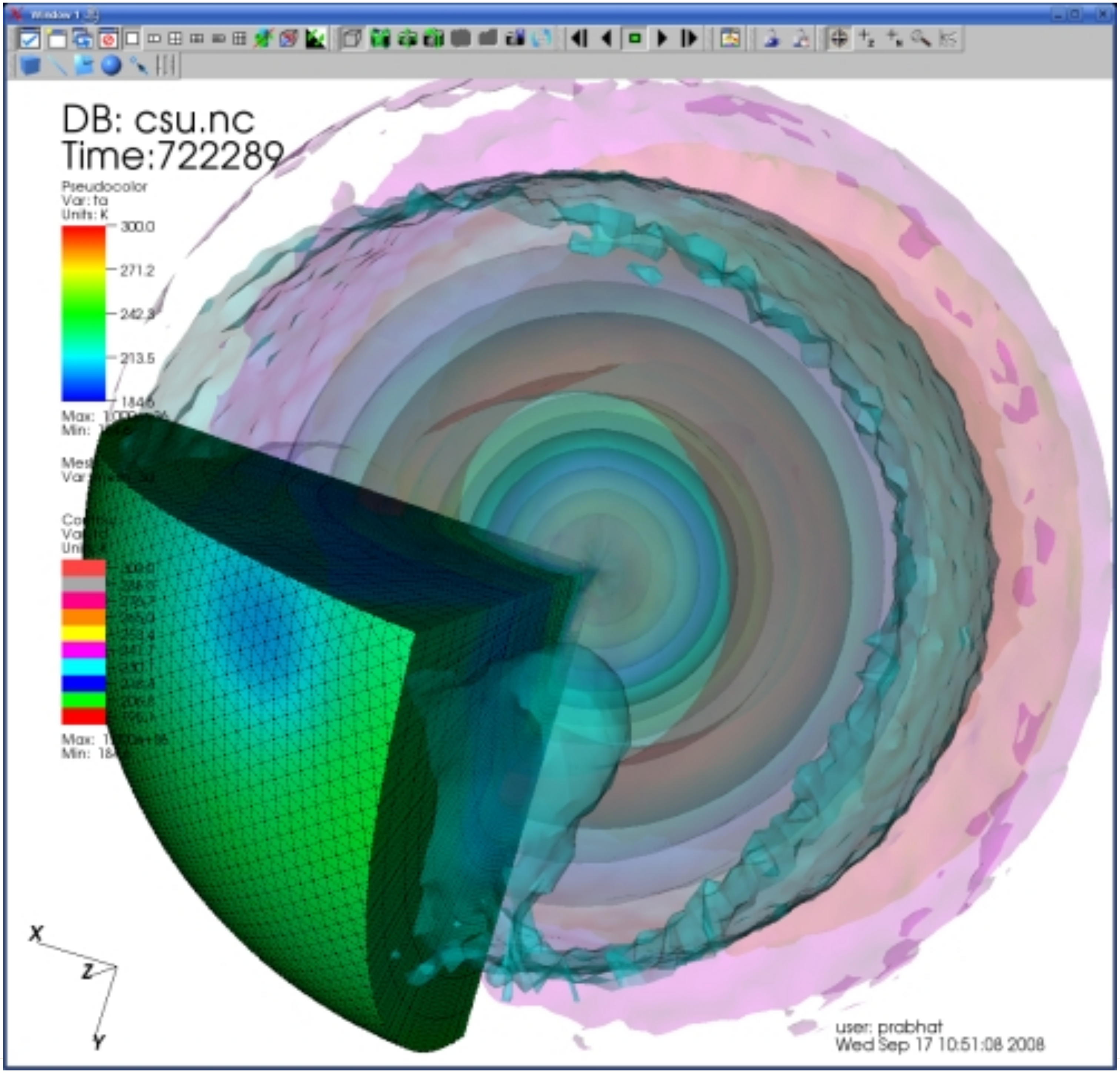
Variable defined at
cell edges (edge)

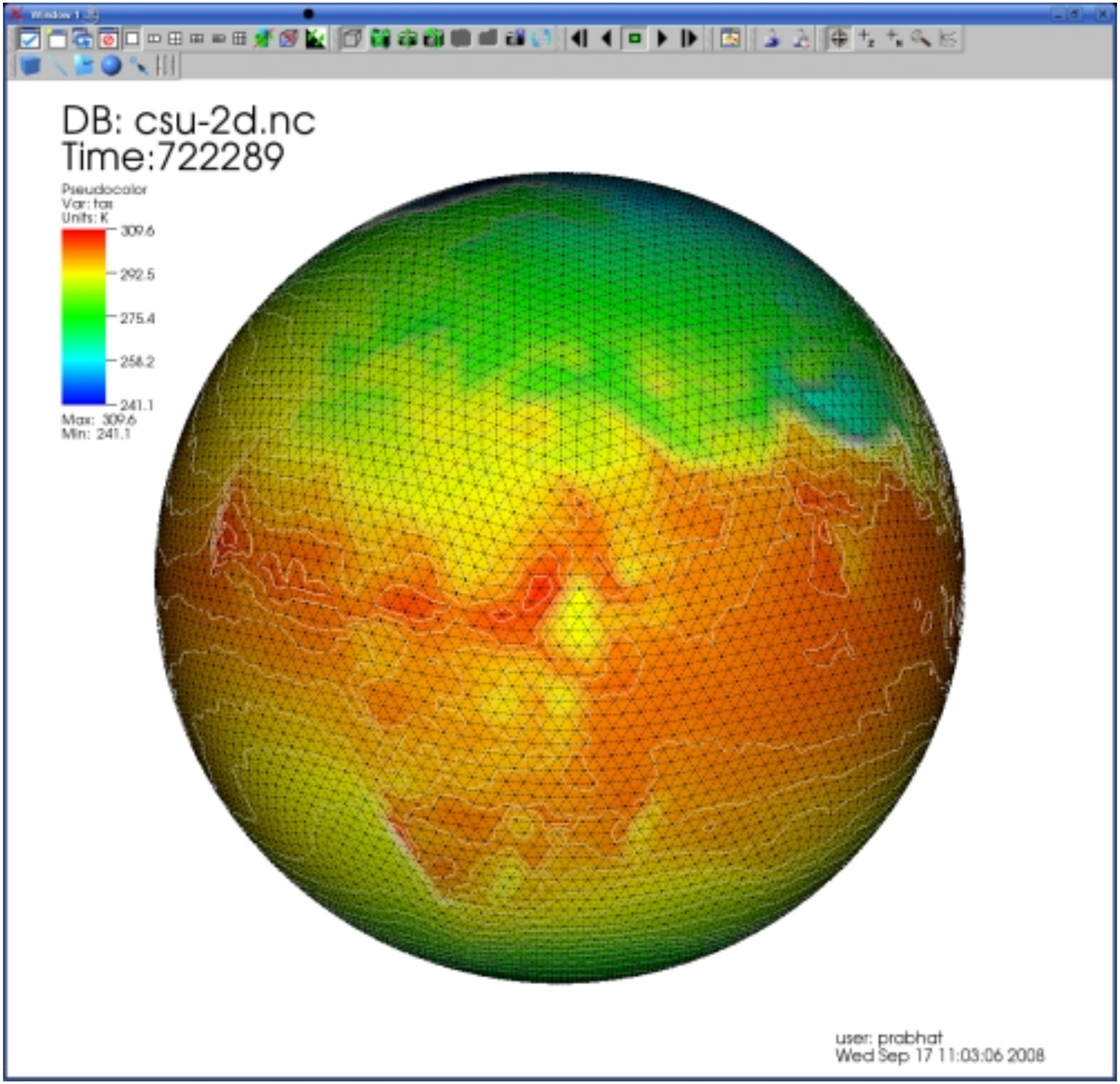


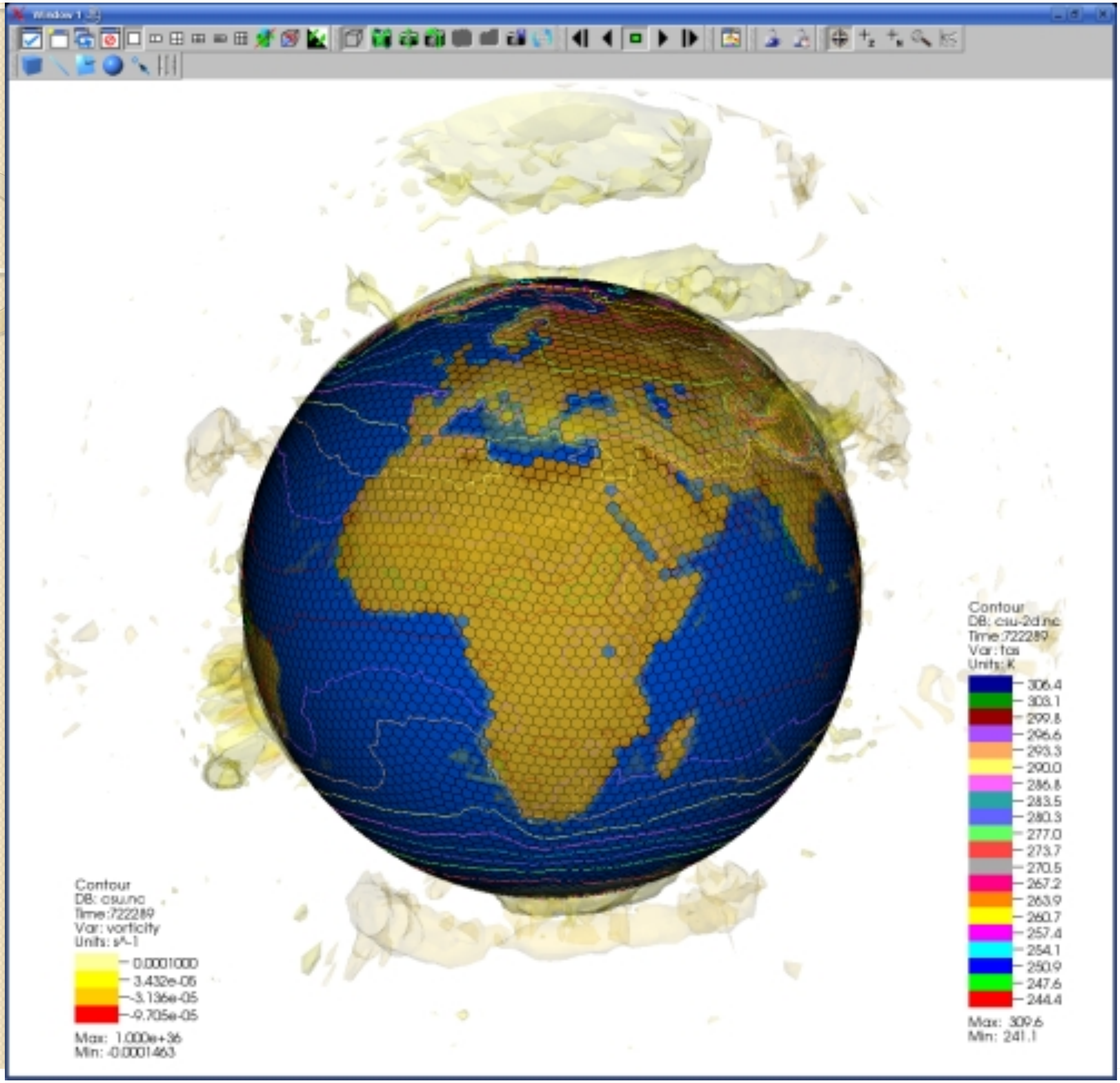
Requires interpolation
to cell centers (blue points)
using information from
surounding edges(green
points)



Pseudocolor
Var: u
Units: m/sec
54.46
43.92
29.72
7.954
-40.82
Pseudocolor
Var: pressure
Units: Pa
1.032e+05
1.024e+05
1.016e+05
1.008e+05
1.000e+05


DB: csu.nc
Time:722289
Pseudocolor
Var: ta
Units: K
300.0
271.2
242.3
213.5
user: prabhat
Wed Sep 17 10:51:08 2008


DB: csu-2d.nc
Time:722289
Pseudocolor
Var: tas
Units: K
309.6
292.5
275.4
258.2
241.1
Max: 309.6
Min: 241.1
user: prabhat
Wed Sep 17 11:03:06 2008


Window 1
Contour
DB: csu.nc
Time:722289
Var: vorticity
Units: s^-1
0.0001000
3.432e-06
-3.136e-05
-9.705e-05
Max: 1.000e+36
Min: -0.0001463
Contour
DB: csu-2d.nc
Time:722289
Var: tas
Units: K
306.4
303.1
299.8
296.6
293.3
290.0
286.8
283.5
280.3
277.0
273.7
270.5
267.2
263.9
260.7
257.4
254.1
250.9
247.6
244.4
Max: 309.6
Min: 241.1

Summary

- GCRM Collective I/O
  - Acceptable level (~2+GB/s)
  - Improvements in I/O hardware at NERSC
  - MPI-IO improvements from Cray
- Data Model
  - Complete
- Visualization
  - Serial plugin implemented
  - Parallel version forthcoming

Acknowledgments

- LBNL/NERSC
  - Mark Howison, Janet Jacobsen, Gunther Weber, Wes Bethel
  - John Shalf, Tony Drummond, Katie Antypas, Andrew Uselton, Shane Canon
  - Michael Wehner
- PNNL
  - Karen Schuchardt, Bruce Palmer, Annette Koontz
- Colorado State Univ
  - Ross Heikes, Dave Randall
- LLNL
  - Hank Childs

Thanks!

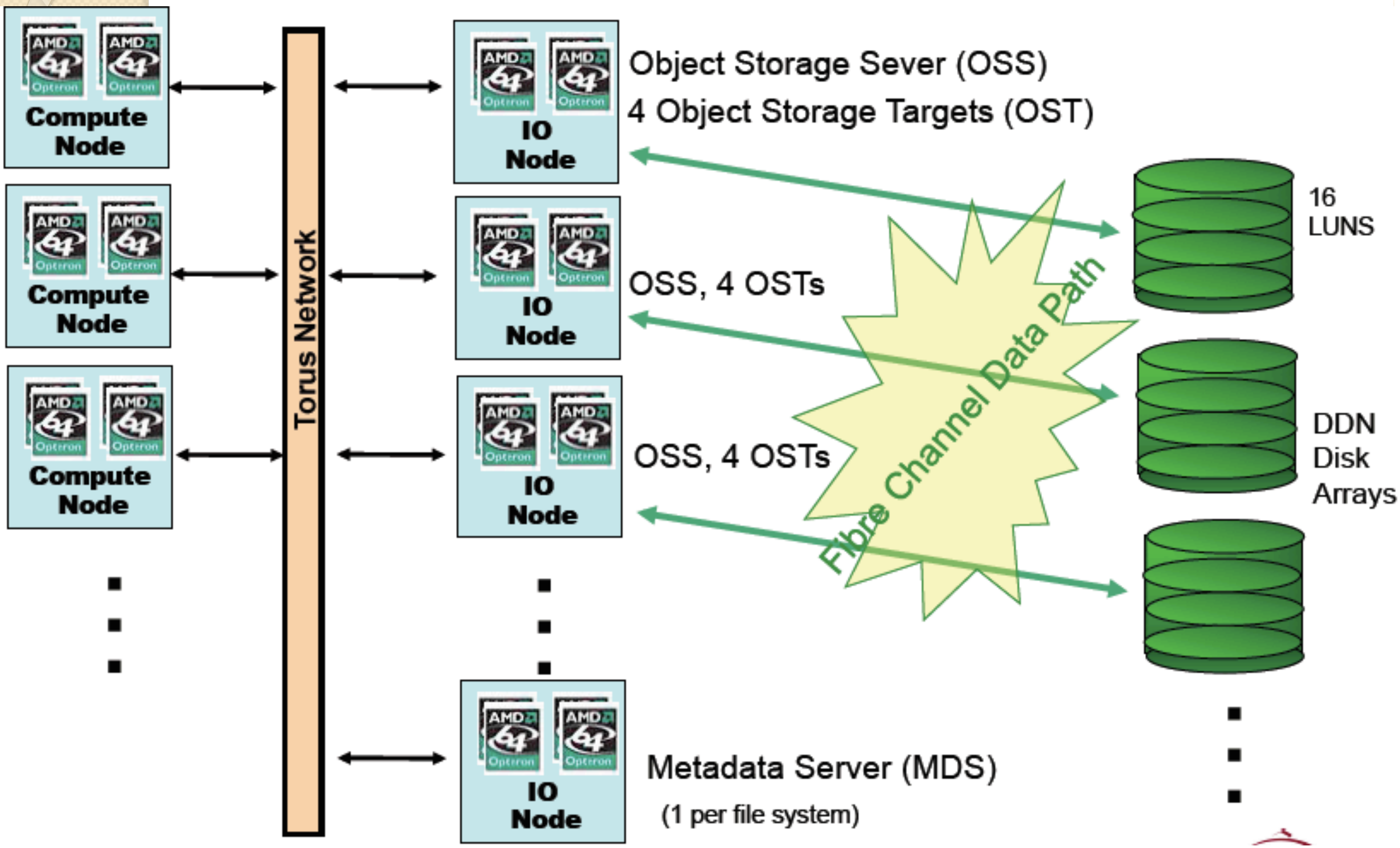
Additional Slides

Franklin's I/O hardware

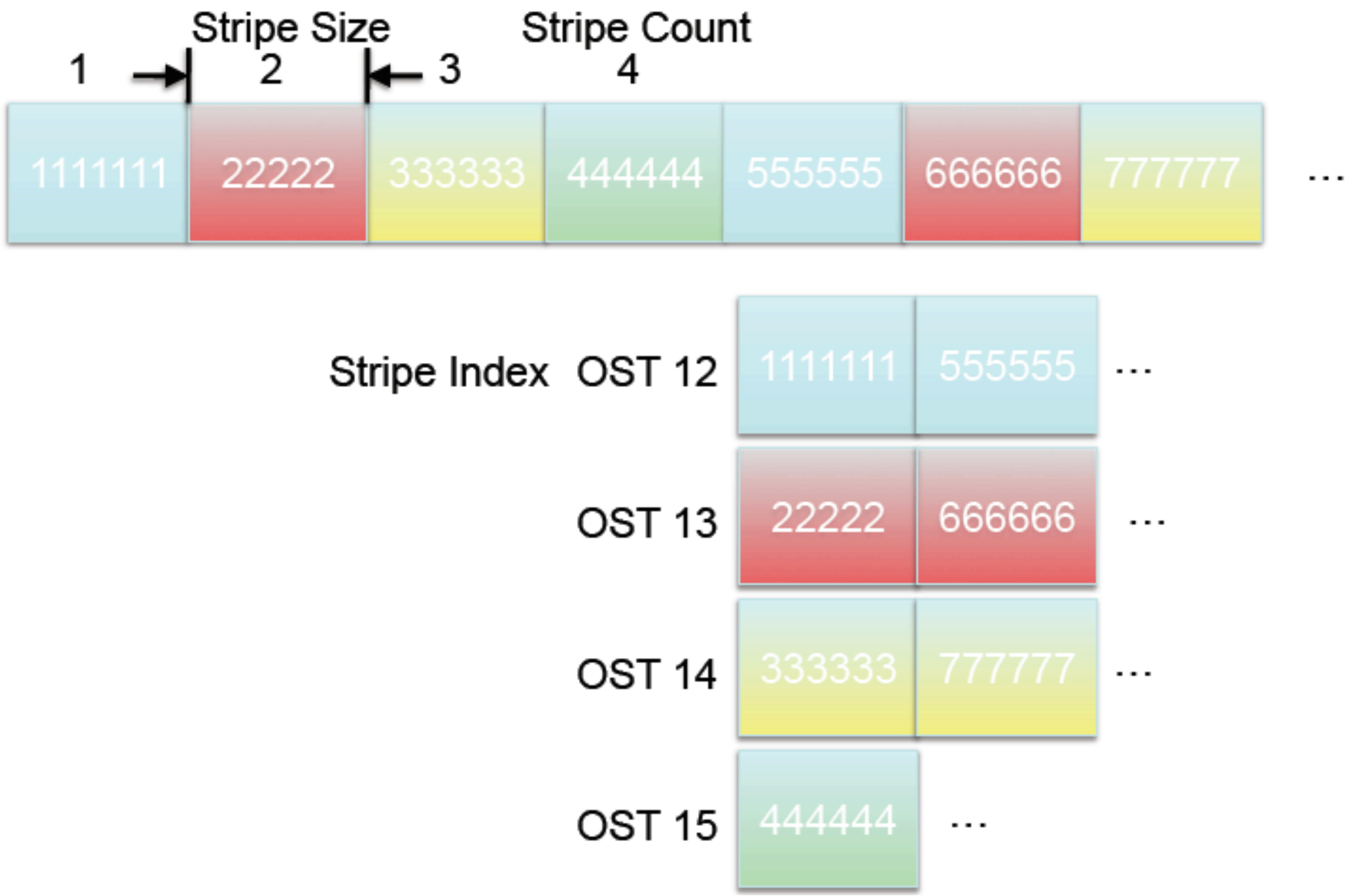
- ~10K nodes, ~40K cores
- SeaStar Network
- 10 OSSs
- 40 OSTs
- Fibre Channel path to DDN Disk Arrays

- 12GB/s peak I/O rate (*Idealized conditions*)

Franklin’s I/O hardware



Lustre Striping



Performance issues

- 2-phase IO offers another solution:
  - Aggregate array on writer nodes
  - Writer node treats data as flat 1D array, which is split into 1MB segments