LA-UR-12-24953
Approved for public release; distribution is unlimited.

Title: MPAS-Ocean Briefing

Author(s): Jacobsen, Douglas W.
Jones, Philip W.
Ringler, Todd D.
Petersen, Mark R.
Maltrud, Mathew E.

Intended for: SciDAC SUPER Meeting, 2012-09-26/2012-09-27 (Chicago, Illinois, United States)

# MPAS-Ocean Briefing

MPAS-O Development Team:
Doug Jacobsen, Todd Ringler, Mark Petersen,
Phil Jones, Mathew Maltrud

Los Alamos National Laboratory

September 24, 2012

Los Alamos NATIONAL LABORATORY EST.1943

# What is MPAS?

The Model for Prediction Across Scales (MPAS) is described as:

- Framework Development Project
- Developed Between NCAR and LANL

Using the shared MPAS framework, several dynamical cores have been developed.

- Hydrostatic Atmospheric Core
- Land-Ice Core
- Non-Hydrostatic Atmospheric Core
- Oceanic Core
- Shallow Water Core

Los Alamos NATIONAL LABORATORY EST.1943

# What is MPAS?

MPAS provides model developers with tools for the rapid prototyping and development of dynamical cores. It also allows developers to:

- Leverage Improvements to Shared Code
- Utilize Already Developed Operators
- Profile Dynamical Cores Using Builtin Profiling Tools

Los Alamos NATIONAL LABORATORY EST.1943

# Best Practices

While development on MPAS progresses, many developers contribute to code. In order to maintain consistency within the project we have a set of best practices developers follow. These include:

- Revision Control
- Design Documents
- Code Review Process
- Regression Tests
  - Temporal Convergence
  - Performance Profiling
  - Model Exercise

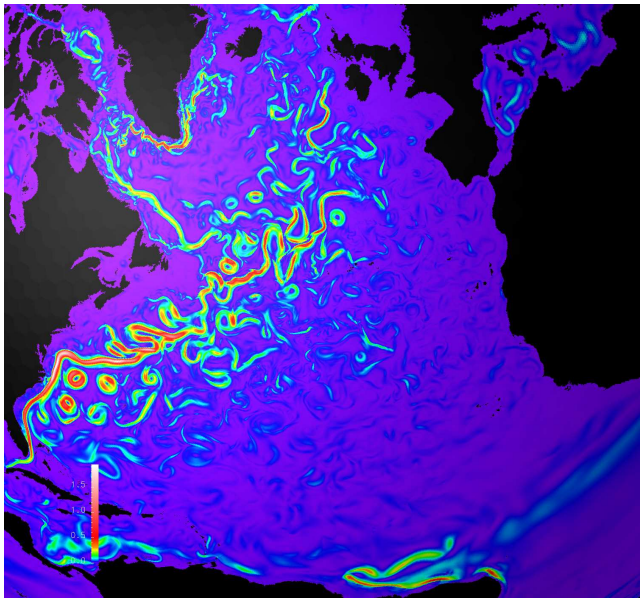
Los Alamos NATIONAL LABORATORY EST.1943

# MPAS-Ocean

While there are several cores developed under the MPAS framework, this talk will discuss shared code as well and specifically the ocean core. MPAS-Ocean is a fully functioning Ocean model. It is capable of simulating using both:

- High Resolution Meshes
- Highly Variable Resolution Meshes

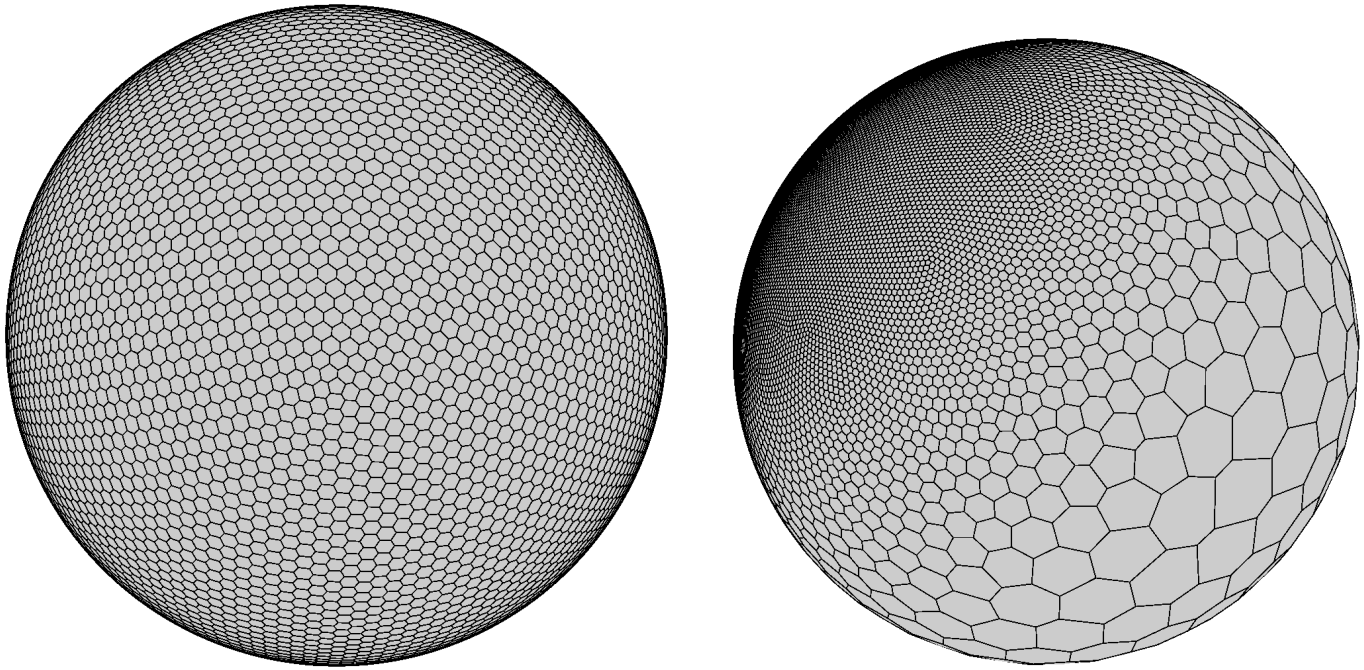
Kinetic Energy ($\frac{m^2}{s^2}$) in a variable resolution simulation. 7.5km in the North Atlantic, and 37.5km elsewhere.



Los Alamos NATIONAL LABORATORY EST.1943

# Unstructured Meshes

The MPAS framework utilizes unstructured meshes. These meshes can be created in a variety of ways but typically we use Spherical Centroidal Voronoi Tessellations (SCVTs).

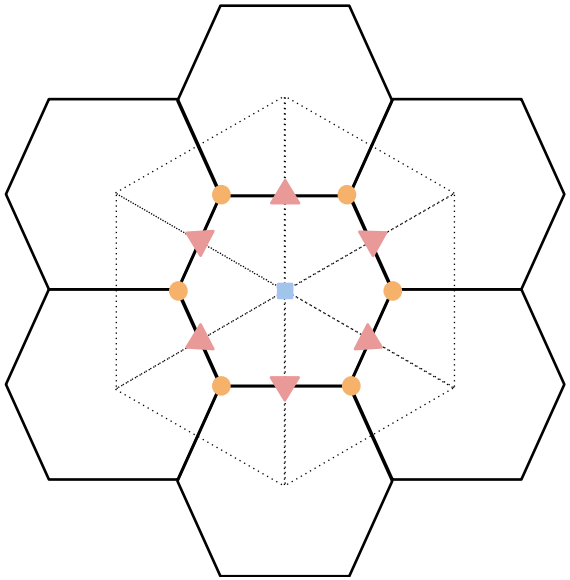


Los Alamos NATIONAL LABORATORY EST.1943

# Unstructured Meshes

The figure below shows the spatial layout of data:

- Primary Mesh (Voronoi, Black Solid Hexagons)
- Dual Mesh (Delaunay, Black Dotted Triangles)
- Scalar Quantities (Blue Squares)
- Vector Quantities (Red Triangles)
- Vorticity Quantities (Orange Circles)



While the data is unstructured horizontally, it is structured vertically. For example:

```
temperature(nVertLevels, nCells)
velocity(nVertLevels, nEdges)
vorticity(nVertLevels, nVertices)
```

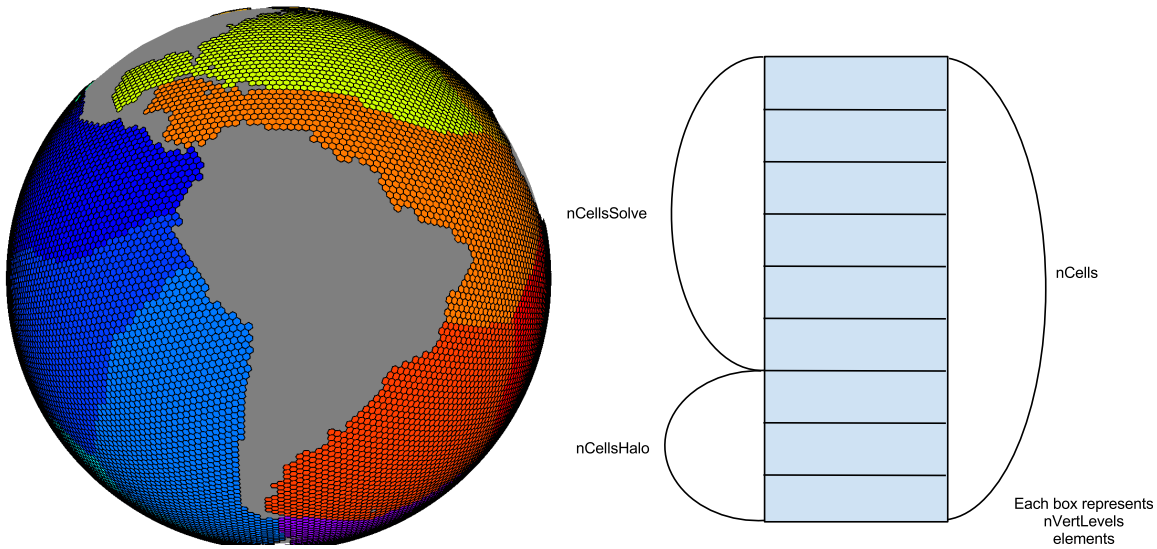
Within a block, typical dimensions are:

- nVertLevels: 40
- nCells: 200-500
- nVertices: 400-1000
- nEdges: 600-1500
- 3-D Fields: 20-100

Los Alamos NATIONAL LABORATORY EST.1943

# Indirect Addressing

Because MPAS uses unstructured meshes, data stored in memory is unstructured as well. Meaning, the next contiguous element in an array might not be a neighbor of the previous element.




Elements are ordered so that owned elements come before halo elements. Ordering can be modified to improve data access patterns.

# Discrete Divergence Operator

Before getting into a more complex example, the following loop shows how divergence is computed at cell centers:

```
!divergence(nVertLevels, nCells)
divergence(:,:) = 0.0
do iEdge = 1, nEdges
  cell1 = cellsOnEdge(1, iEdge)
  cell2 = cellsOnEdge(2, iEdge)

  invAreaCell1 = 1.0 / areaCell(cell1)
  invAreaCell2 = 1.0 / areaCell(cell2)

  do k = 1, maxLevelEdgeBot(iEdge)
    divergence(k, cell1) = divergence(k, cell1) &
                         + dvEdge(iEdge) * u(k, iEdge) * invAreaCell1
    divergence(k, cell2) = divergence(k, cell2) &
                         - dvEdge(iEdge) * u(k, iEdge) * invAreaCell2
  end do
end do
```

# Indirect Addressing cont.

As an example of the unstructured/structured data use, the following loop is an example of how to compute the coriolis force at an edge:

```
do iEdge = 1, nEdgesSolve
  cell1 = cellsOnEdge(1, iEdge)
  cell2 = cellsOnEdge(2, iEdge)

  invLength = 1.0 / dcEdge(iEdge)

  do k = 1, maxLevelEdgeTop(iEdge)
    q = 0.0

    do j = 1, nEdgesOnEdge(iEdge)
      eoe = edgesOnEdge(j, iEdge)
      workpv = 0.5 * (Vor_edge(k, iEdge) + Vor_edge(k, edgeOnEdge)
      q = q + weightsOnEdge(j, iEdge) * u(k, eoe) * workpv * h_edge(k, eoe)
    end do

    tend_contribution = (q - (ke(k, cell2) - ke(k,cell1)) * invLength
    tend(k, iEdge) = tend(k, iEdge) + edgeMask(k, iEdge) * tend_contribution
  end do
end do
```

# Communications

In addition to needing indirect addressing, communication routines were developed as part of the shared framework. These routines handle two different types of communications:

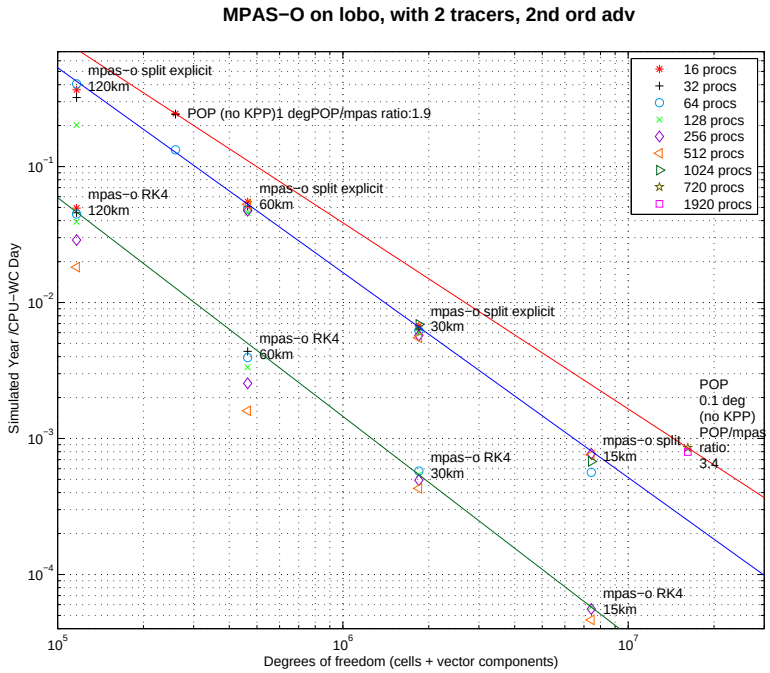
- All-to-All Communications
- Halo Exchange Communications

Both of these routines were written to reduce the amount of time spent waiting for MPI communications to complete.

Los Alamos NATIONAL LABORATORY EST.1943

# Where do we stand?

Some basic performance testing of MPAS-O shows we can currently achieve 2 SYPD using 3000 processors on a quasi-uniform 15km global mesh.




Typical production climate models seek a throughput of 10 SYPD. We are looking for roughly a factor of 4.

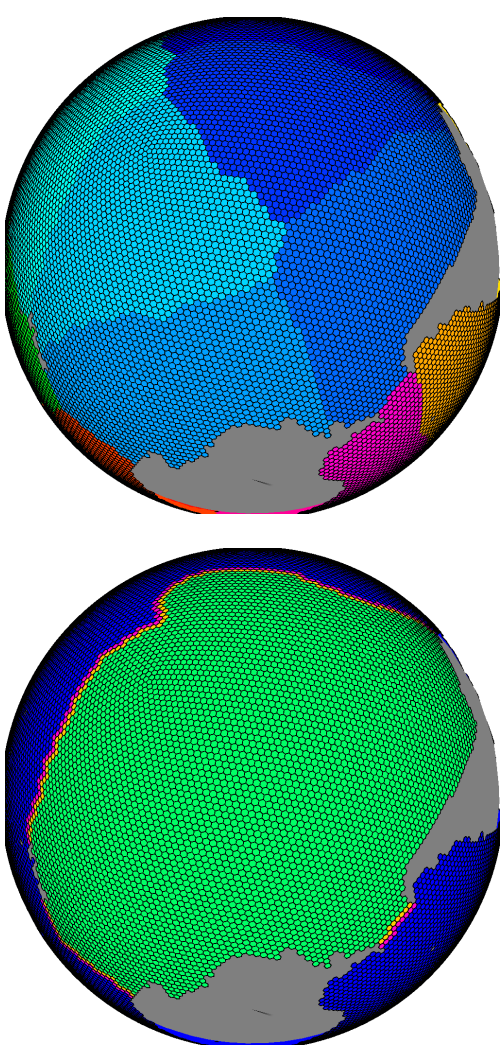
# Performace Additions

Recent additions to the MPAS framework, with a focus on performace improvements, include:

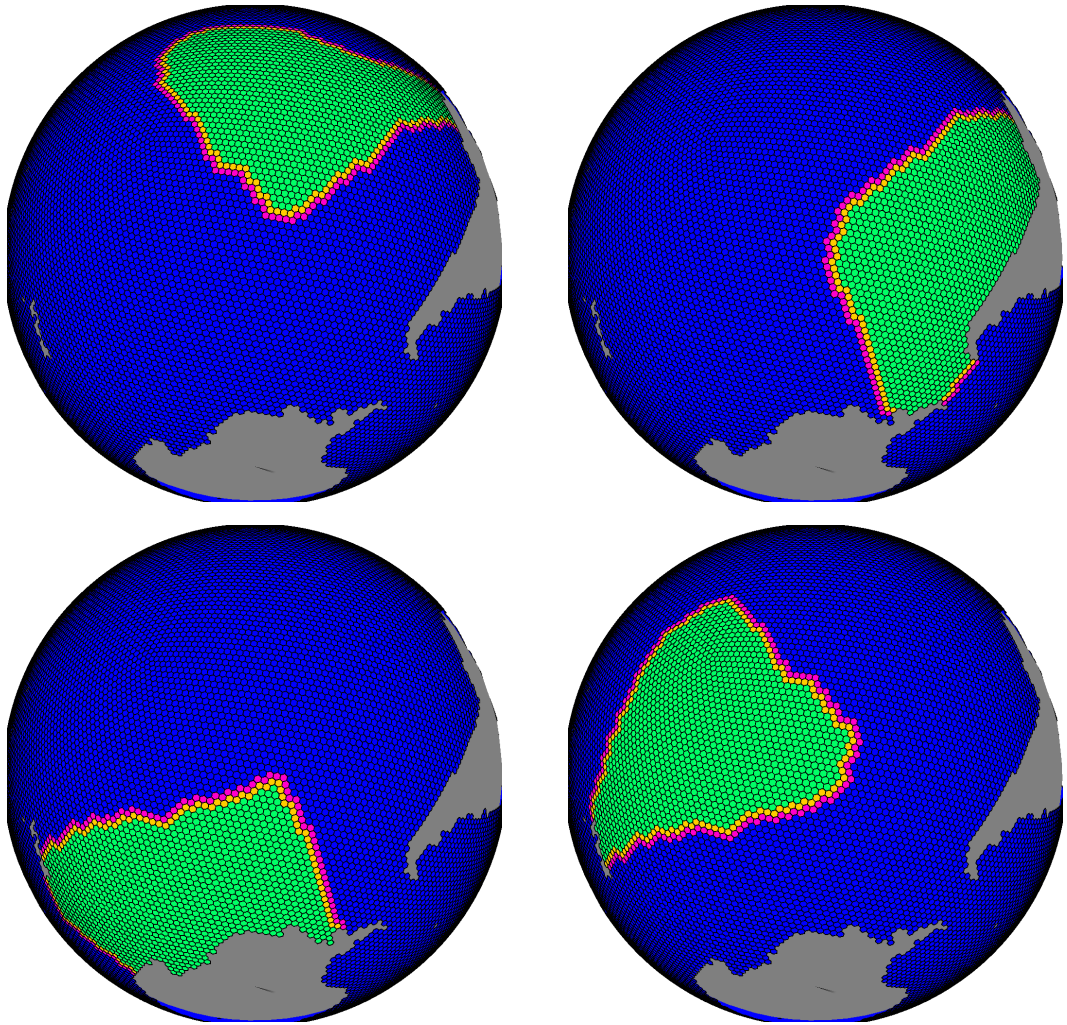
- Parallel I/O using the NCAR Parallel I/O library built on top of pNETCDF
- Multiple computational blocks Per MPI Process to allow heterogeneous computing
- Arbitrary size halos to tune efficiency of MPI send/recvs

Los Alamos NATIONAL LABORATORY EST.1943

# What are multiple computational blocks?



Decomposed Blocks and Owned Cells



Local Blocks

# Working with SUPER

While we have yet to work directly with members of the SUPER team, we currently have in place:

- Built-in Profiling Using Timers
- Timers using MPI_WTime and PAPI

Because of the already existing hooks with PAPI, it will be relatively easy to add hardware counters as another profiling tool.

Los Alamos NATIONAL LABORATORY EST.1943

# Where do we want to be?

For the future, we have goals of implementing multiple levels of parallelism within MPAS, including:

- OpenMP (Needs Exploration)
- GPU Implementations
  - CUDA-Fortran
  - OpenACC

In addition to improving our parallelism, we would like to profile and tune expensive portions of code currently used within MPAS-O.

Our key need is higher performance, as measured by SYPD using a 15-30km global resolution mesh. We also seek to increase our flop efficiency, in getting it closer to peak on a given machine.

Los Alamos NATIONAL LABORATORY EST.1943

# The End

Questions?

Los Alamos NATIONAL LABORATORY EST.1943