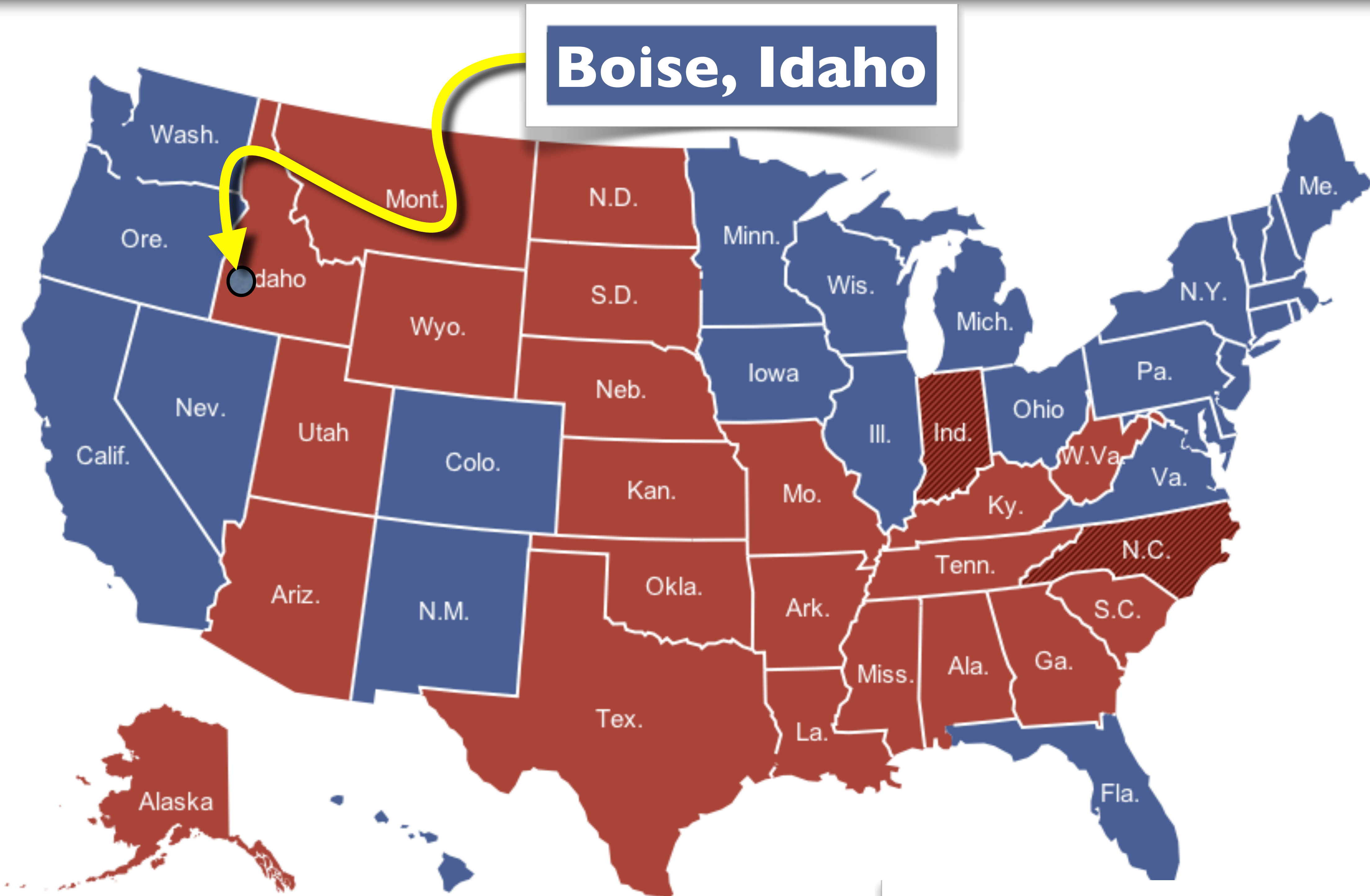
A hybrid adaptive mesh framework for finite volume schemes on a forest of locally refined Cartesian meshes

Donna Calhoun (Boise State University)

Carsten Burstedde (University of Bonn, Germany)

SIAM Geosciences
Padua, Italy
June 17-20, 2013

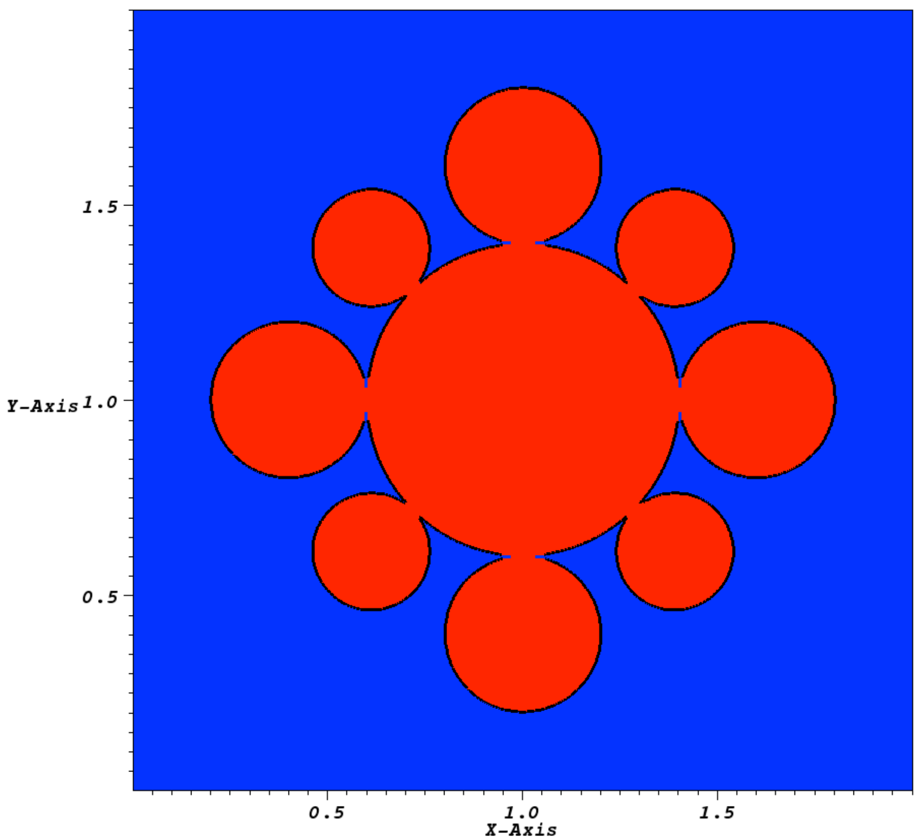
Where is Boise?

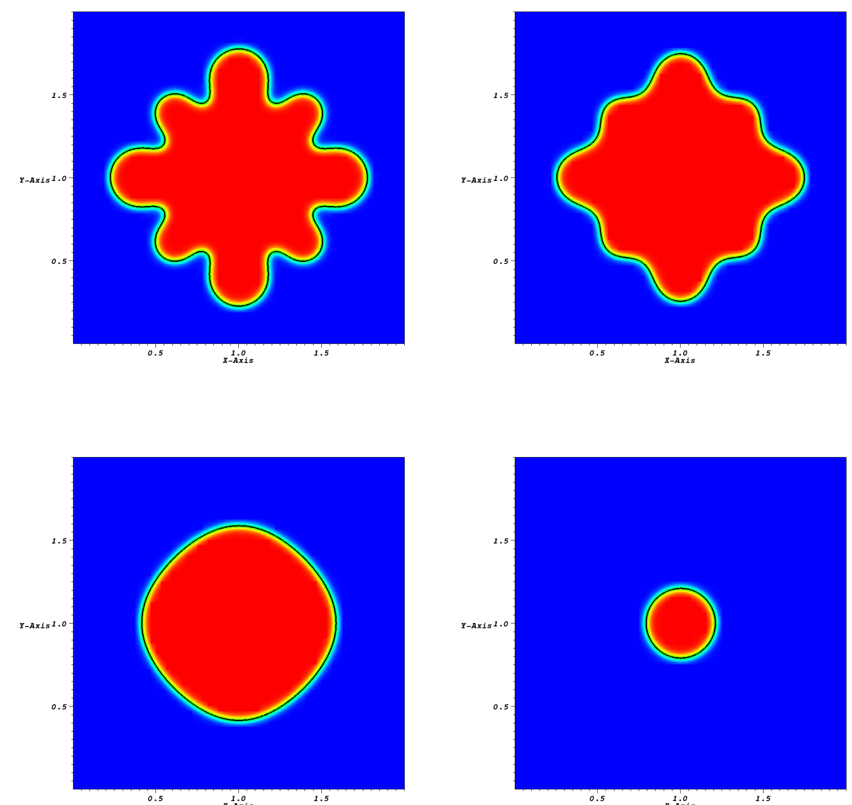


Boise comes from boisé which is French for 'wooded' or 'forested'.

Adaptive Mesh Refinement

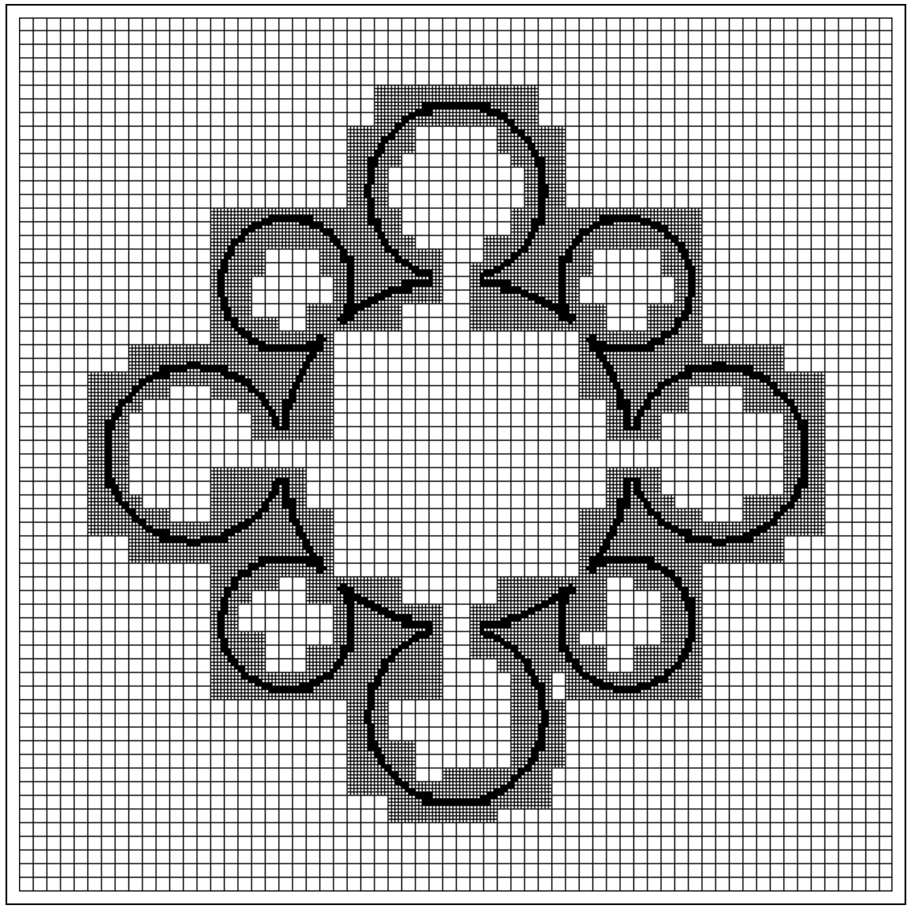
When solving PDEs using mesh based methods, it is generally recognized that many problems could benefit enormously from a multi-resolution grid, or spatial adaptivity.



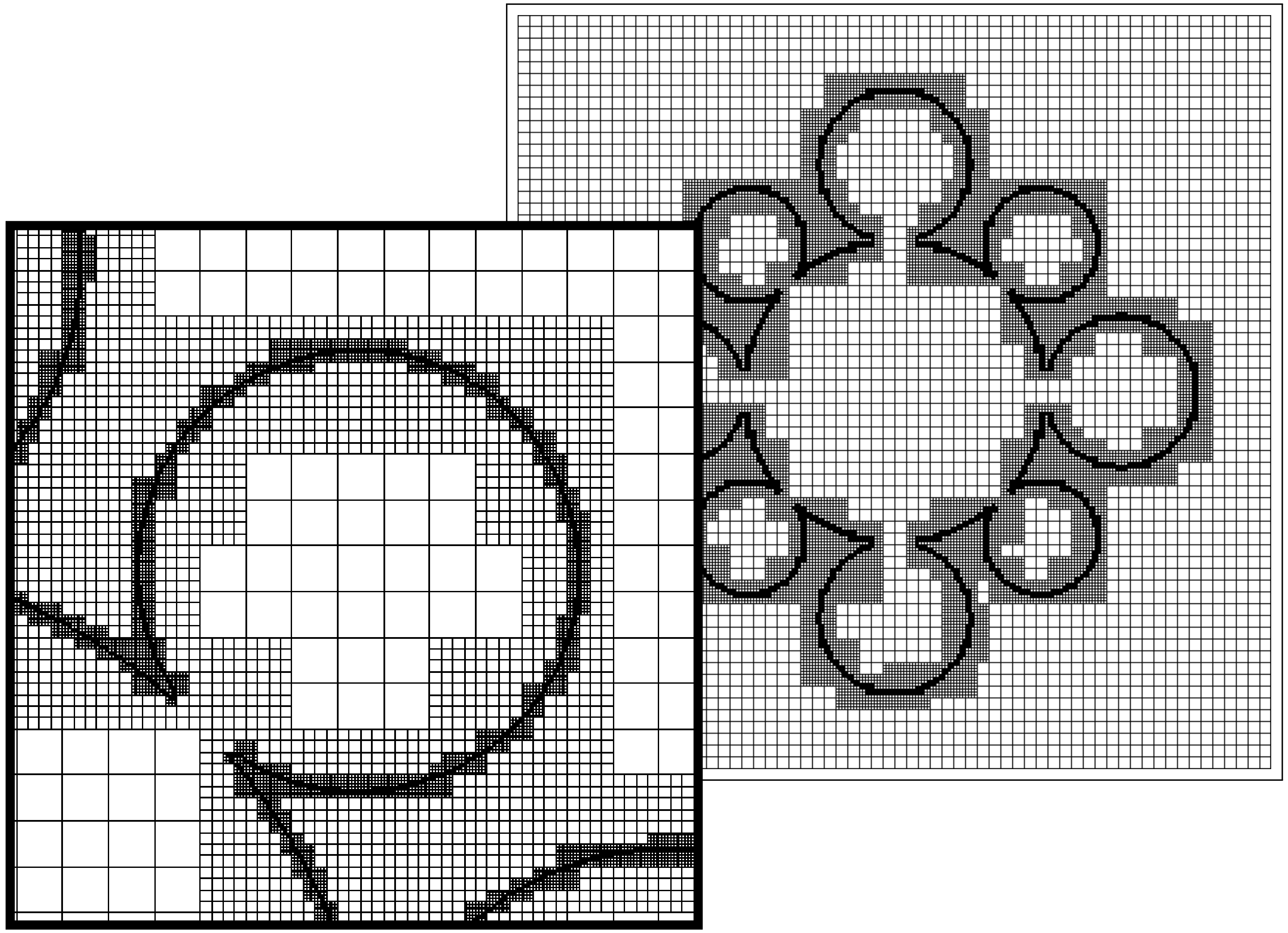


Allen Cahn equation - Flow by mean curvature

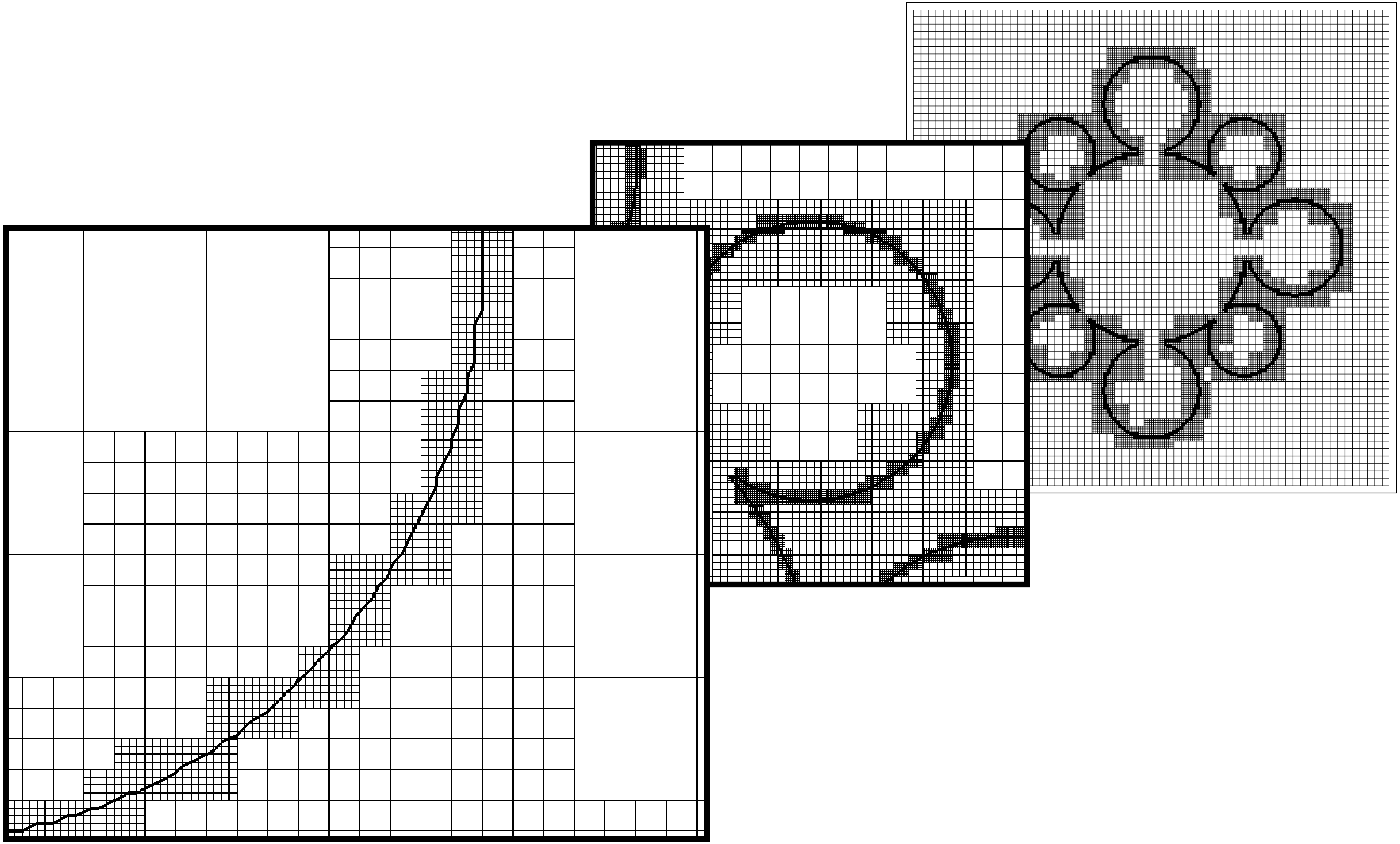
Example : Flow by mean curvature



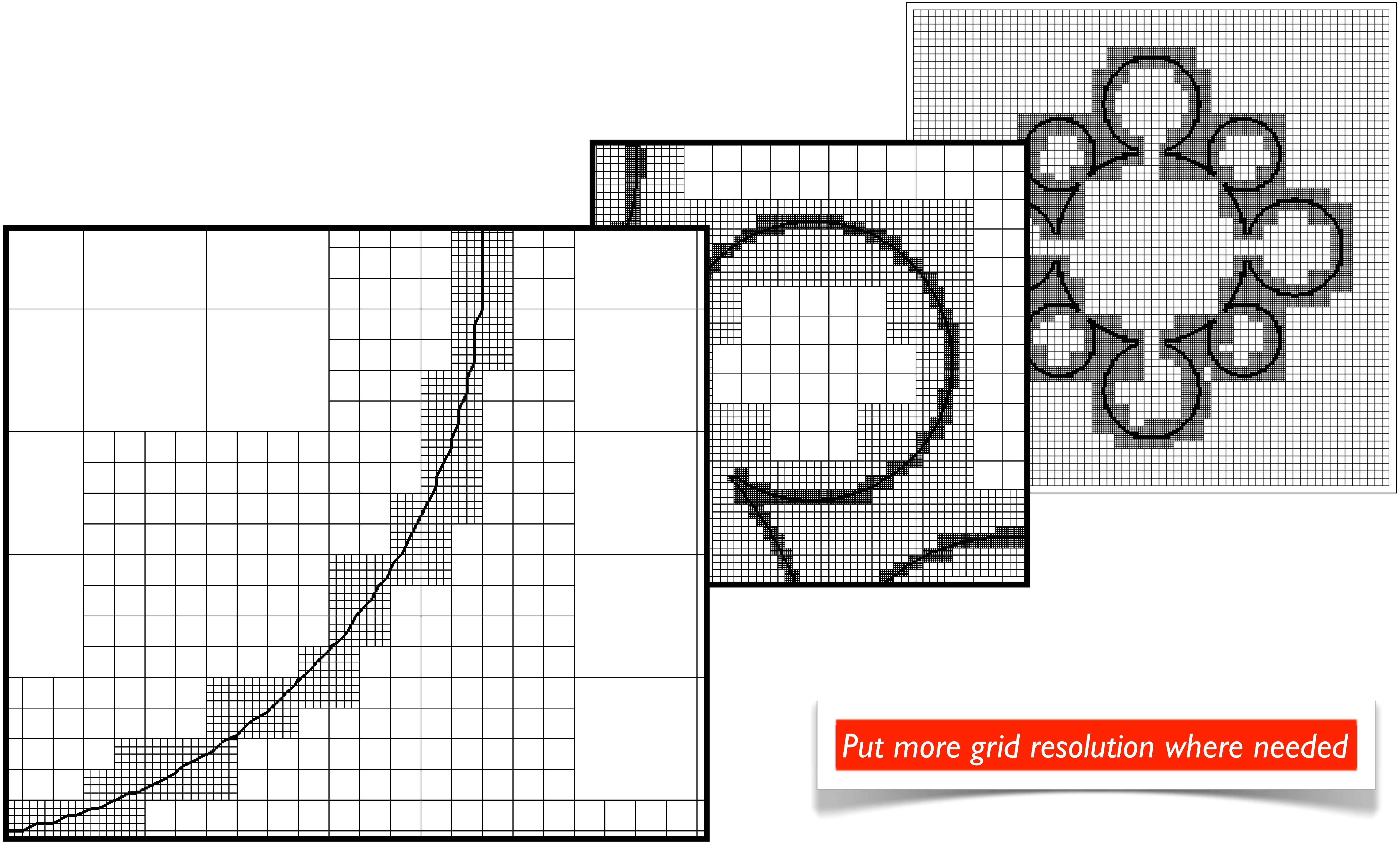
Example : Flow by mean curvature



Example : Flow by mean curvature

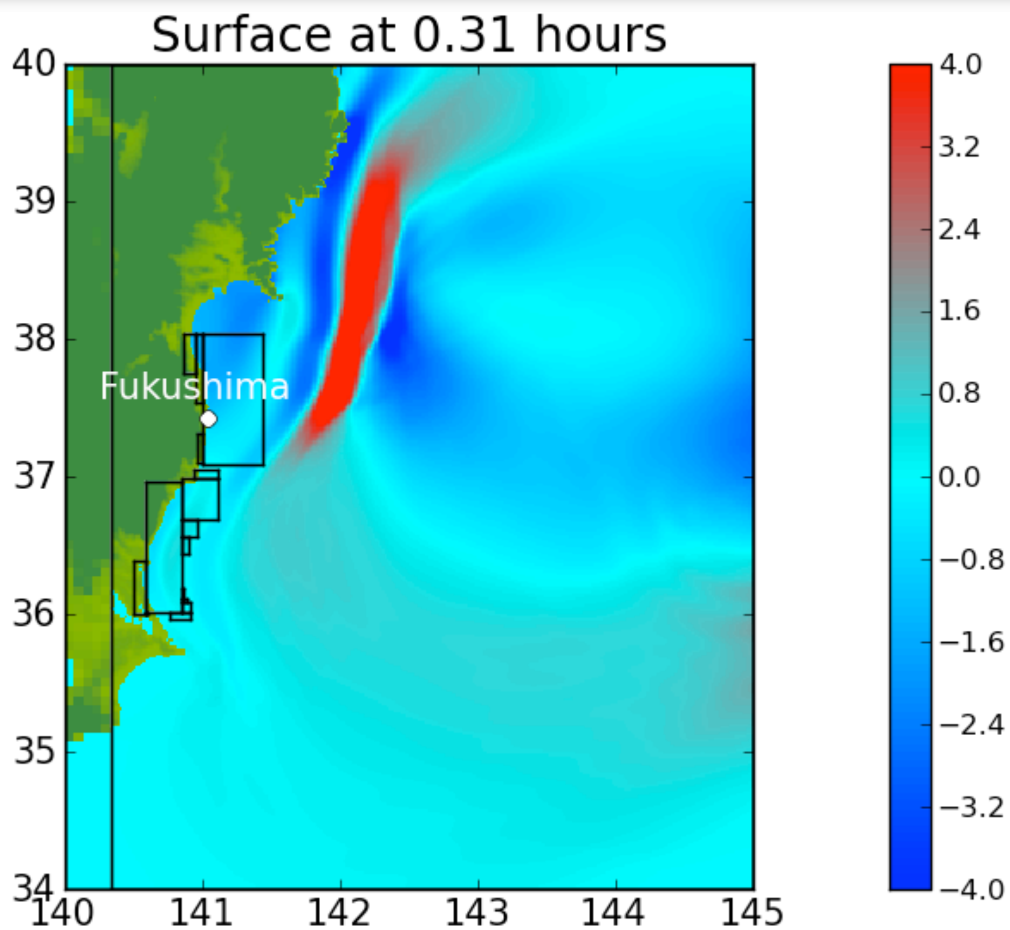


Example : Flow by mean curvature

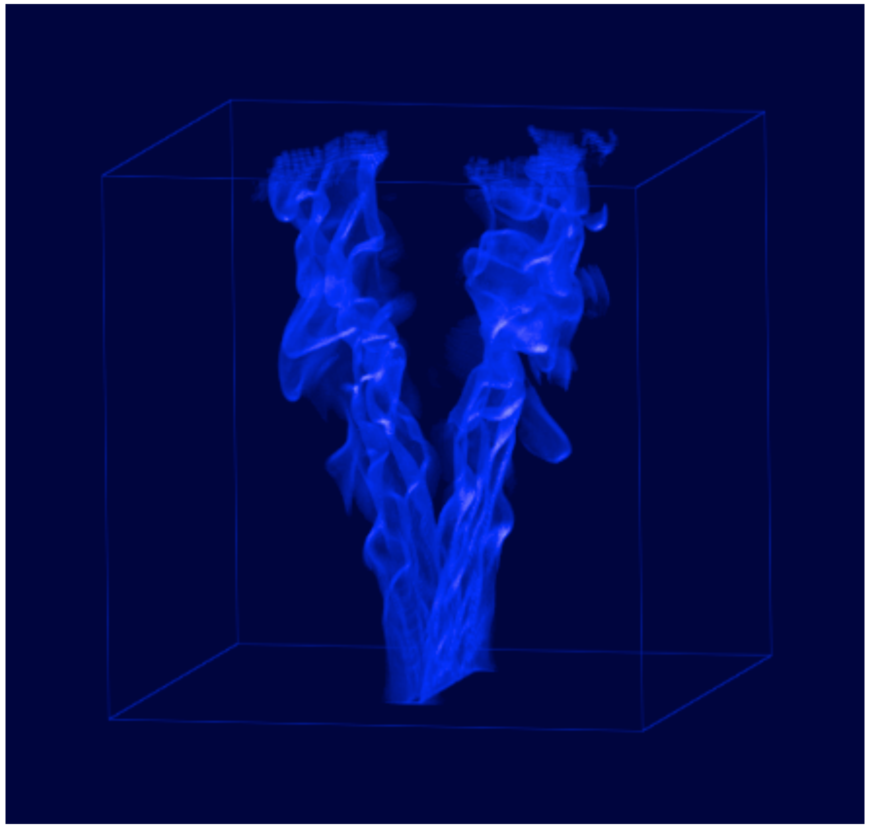


Put more grid resolution where needed

Applications for AMR



Tsunami modeling (R. LeVeque, D. George, M. Berger)



Rod stabilized V-flame (J. B. Bell, Lawrence Berkeley Lab)

- Tracer transport in the atmosphere
- Astrophysics
- Shock capturing for aerodynamic applications
- Regional weather forecasting, hurricanes, ...

Isaac Newton Institute, Fall 2012

"Multi-scale numerics for the ocean and atmosphere"

Isaac Newton Institute, Fall 2012

"Multi-scale numerics for the ocean and atmosphere"

What I learned from the atmospheric science community :

Isaac Newton Institute, Fall 2012

"Multi-scale numerics for the ocean and atmosphere"

What I learned from the atmospheric science community :

- There remains skepticism about how effective adaptive mesh refinement (AMR) can be in weather and climate modeling

Isaac Newton Institute, Fall 2012

"Multi-scale numerics for the ocean and atmosphere"

What I learned from the atmospheric science community :

- There remains skepticism about how effective adaptive mesh refinement (AMR) can be in weather and climate modeling
- Coarse/fine boundaries with abrupt resolution changes are a source of much numerical hand-wringing,

Isaac Newton Institute, Fall 2012

"Multi-scale numerics for the ocean and atmosphere"

What I learned from the atmospheric science community :

- There remains skepticism about how effective adaptive mesh refinement (AMR) can be in weather and climate modeling
- Coarse/fine boundaries with abrupt resolution changes are a source of much numerical hand-wringing,
- Multirate time stepping is not always used,

Isaac Newton Institute, Fall 2012

"Multi-scale numerics for the ocean and atmosphere"

What I learned from the atmospheric science community :

- There remains skepticism about how effective adaptive mesh refinement (AMR) can be in weather and climate modeling
- Coarse/fine boundaries with abrupt resolution changes are a source of much numerical hand-wringing,
- Multirate time stepping is not always used,
- Lack of good refinement criteria dampens enthusiasm for trying out AMR,

Isaac Newton Institute, Fall 2012

"Multi-scale numerics for the ocean and atmosphere"

What I learned from the atmospheric science community :

- There remains skepticism about how effective adaptive mesh refinement (AMR) can be in weather and climate modeling
- Coarse/fine boundaries with abrupt resolution changes are a source of much numerical hand-wringing,
- Multirate time stepping is not always used,
- Lack of good refinement criteria dampens enthusiasm for trying out AMR,
- When AMR *is* used in numerical weather prediction, the goals are often modest : "Do no harm!"

Isaac Newton Institute, Fall 2012

"Multi-scale numerics for the ocean and atmosphere"

What I learned from the atmospheric science community :

- There remains skepticism about how effective adaptive mesh refinement (AMR) can be in weather and climate modeling
- Coarse/fine boundaries with abrupt resolution changes are a source of much numerical hand-wringing,
- Multirate time stepping is not always used,
- Lack of good refinement criteria dampens enthusiasm for trying out AMR,
- When AMR *is* used in numerical weather prediction, the goals are often modest : "Do no harm!"
- Grids are often only static; not dynamically refined.

Work with AMR

Mesh generation and mesh adaptation for large-scale Earth-system modelling, (Phil. Trans. Roy. Soc, 2009) compiled and edited by N. Nikiforakis

- M. Berger, et al (AMRClaw)
- C. Jablonowski et al
- H. Weller (INI organizer)
- C. Gatti-Bono and P. Colella (Chombo)
- R. Klein, N. Nikiforakis et al
- J. Behrens et al
- G. Pau, A. Almgren, J. Bell (LBL, California)
- C. Castro et al
- M. Piggot et al

Finite volume schemes

Finite volume schemes

- Second order finite volume schemes on logically Cartesian meshes,

Finite volume schemes

- Second order finite volume schemes on logically Cartesian meshes,
- PDEs are typically solved in "conservation" form, and so discrete conservation of mass, momentum and so on is assured,

Finite volume schemes

- Second order finite volume schemes on logically Cartesian meshes,
- PDEs are typically solved in "conservation" form, and so discrete conservation of mass, momentum and so on is assured,
- Performance efficiency is obtained by using locally solution adapted nested grids (adaptive mesh refinement, or AMR)

Finite volume schemes

- Second order finite volume schemes on logically Cartesian meshes,
- PDEs are typically solved in "conservation" form, and so discrete conservation of mass, momentum and so on is assured,
- Performance efficiency is obtained by using locally solution adapted nested grids (adaptive mesh refinement, or AMR)
- Geometry handled using grid mappings, or cut-cell methods, in conjunction with multi-block domains.

Finite volume schemes

- Second order finite volume schemes on logically Cartesian meshes,
- PDEs are typically solved in "conservation" form, and so discrete conservation of mass, momentum and so on is assured,
- Performance efficiency is obtained by using locally solution adapted nested grids (adaptive mesh refinement, or AMR)
- Geometry handled using grid mappings, or cut-cell methods, in conjunction with multi-block domains.
- Widespread use in science and engineering,

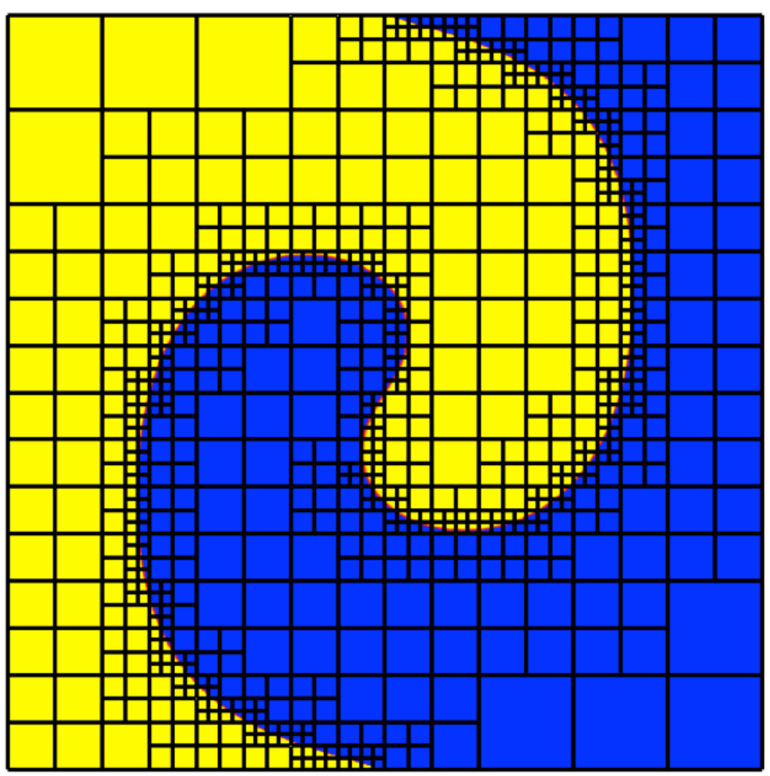
Finite volume schemes

- Second order finite volume schemes on logically Cartesian meshes,
- PDEs are typically solved in "conservation" form, and so discrete conservation of mass, momentum and so on is assured,
- Performance efficiency is obtained by using locally solution adapted nested grids (adaptive mesh refinement, or AMR)
- Geometry handled using grid mappings, or cut-cell methods, in conjunction with multi-block domains.
- Widespread use in science and engineering,
- "Collocated", "P0", "A-Grids", "ijk" grids, non-conforming (for AMR) meshes

Many flavors of adaptivity

- Block-structured AMR (Berger, Oliger, Colella, ...)
- Tree-based adaptivity (Popinet, Tessyier, ...)
- Finite-element adaptivity includes both h-refinement (increase mesh resolution) and p-refinement (increase order of accuracy)

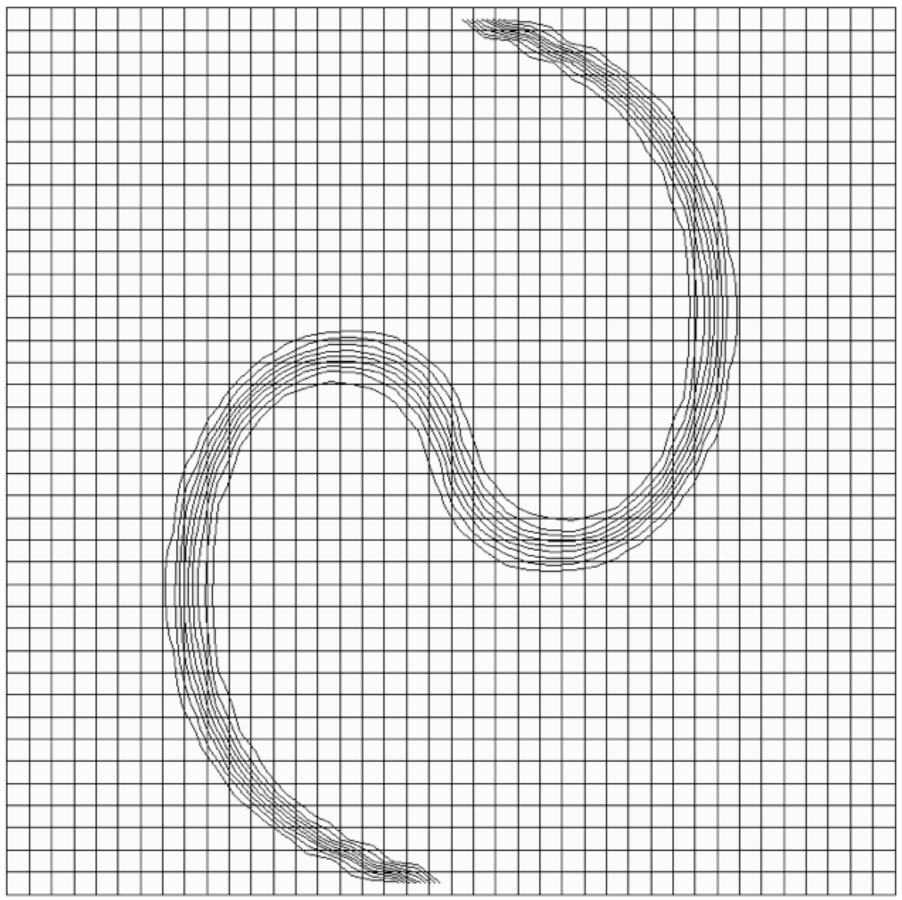
q(1) at time 0.7500



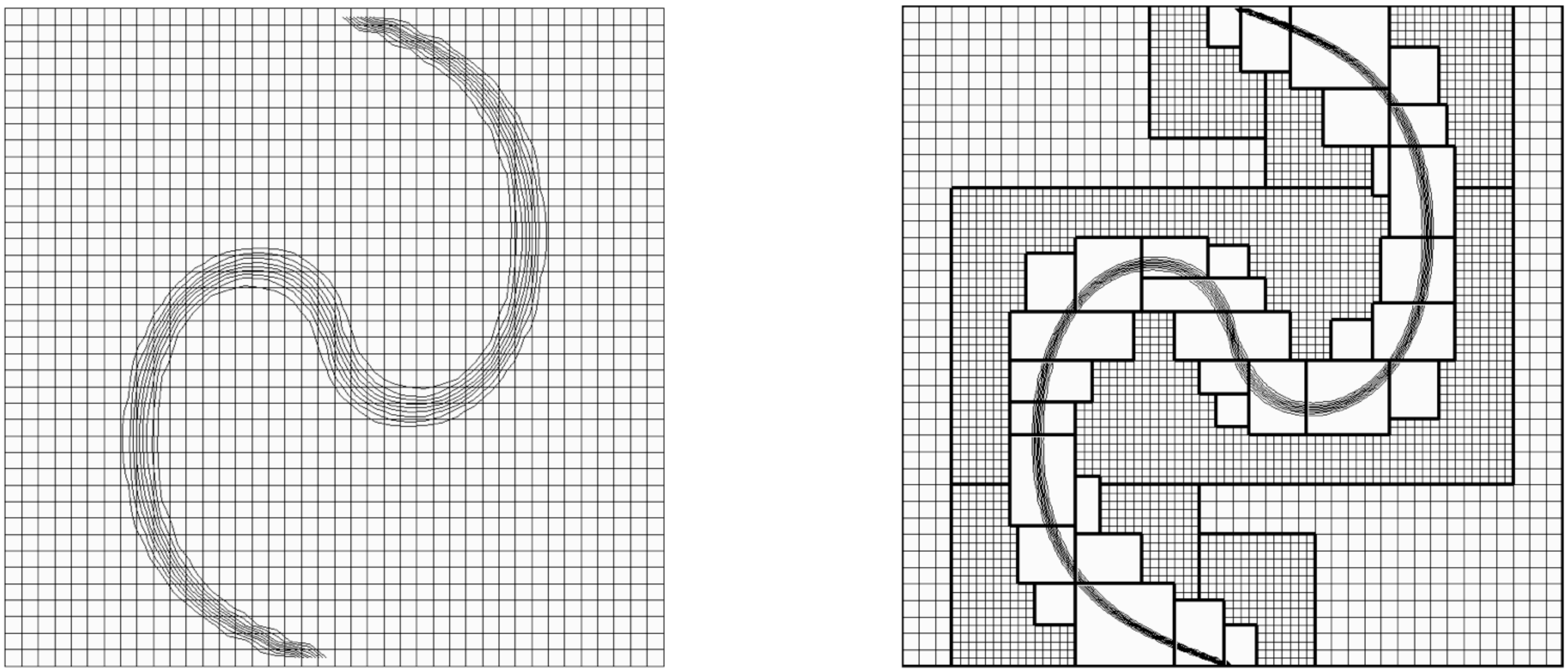
Tree-based adaptivity :

- *Gerris* (S. Popinet, NIWA, NZ),
- *Ramses* (R. Tessyier) and many other codes (including several in astrophysics)

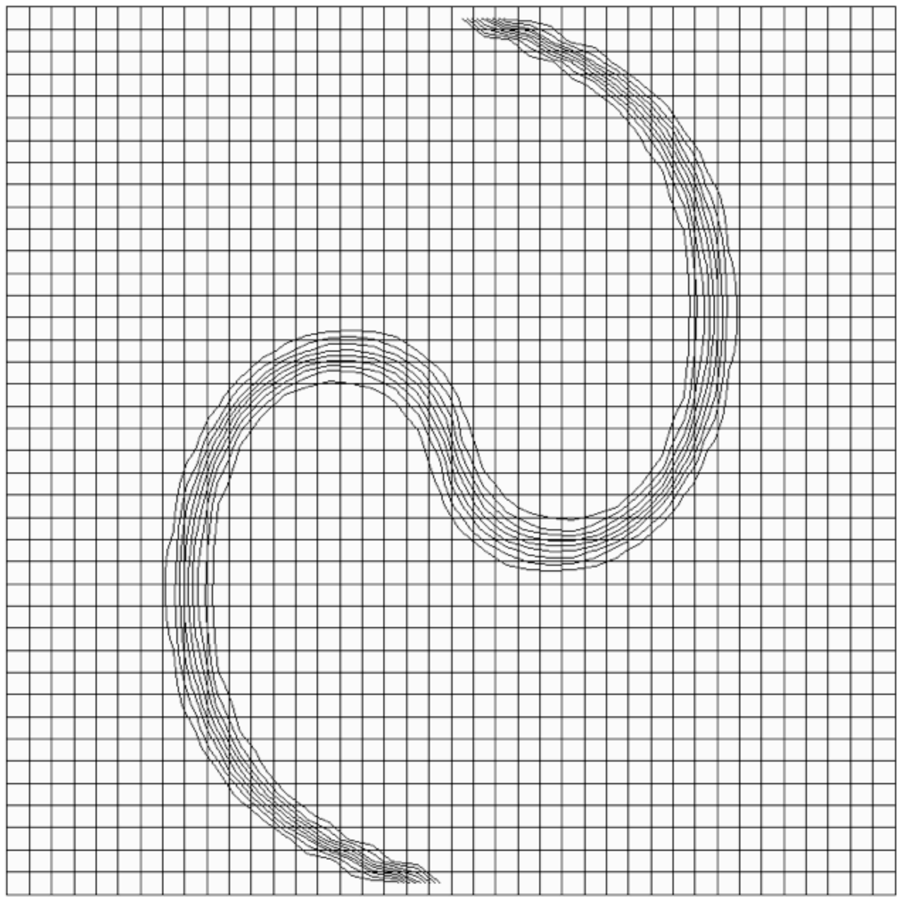
Block structured AMR (Berger, Oliger 1984)

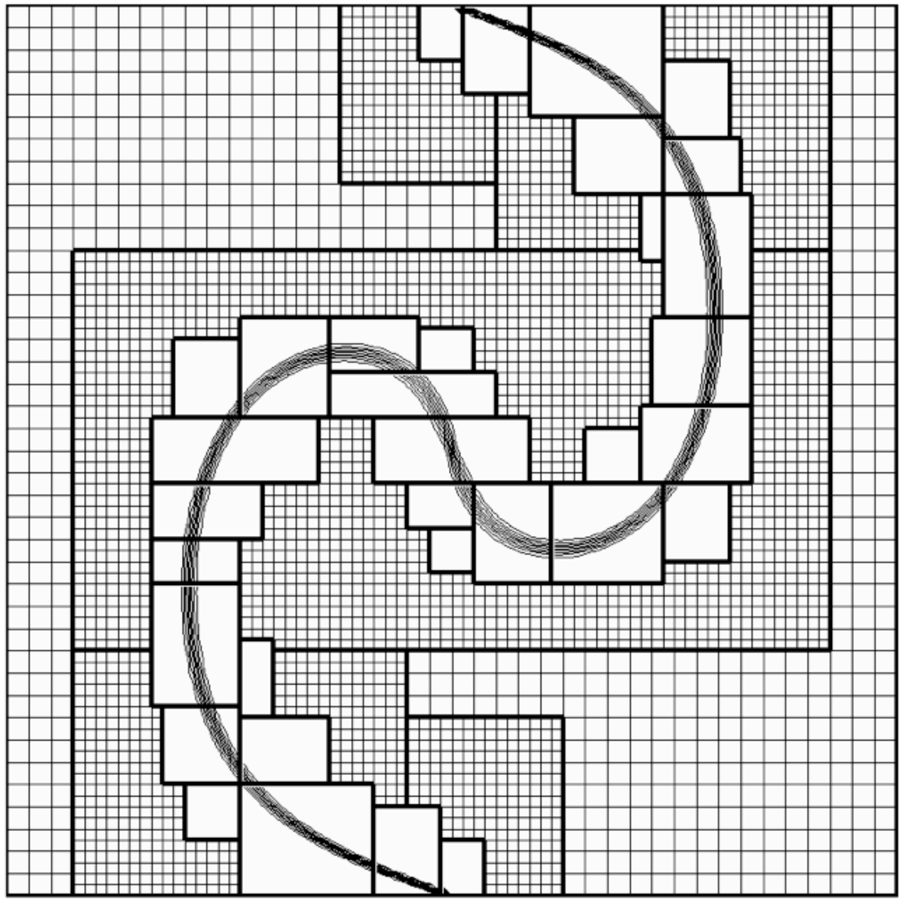


Block structured AMR (Berger, Oliger 1984)



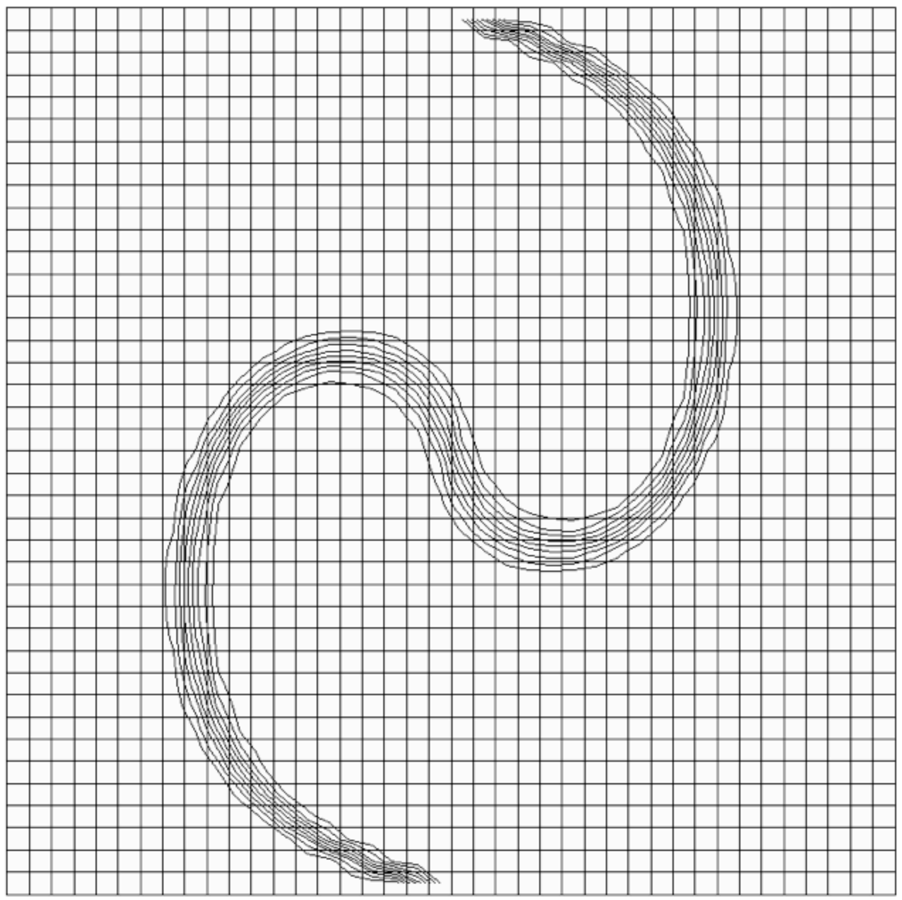
Block structured AMR (Berger, Oliger 1984)

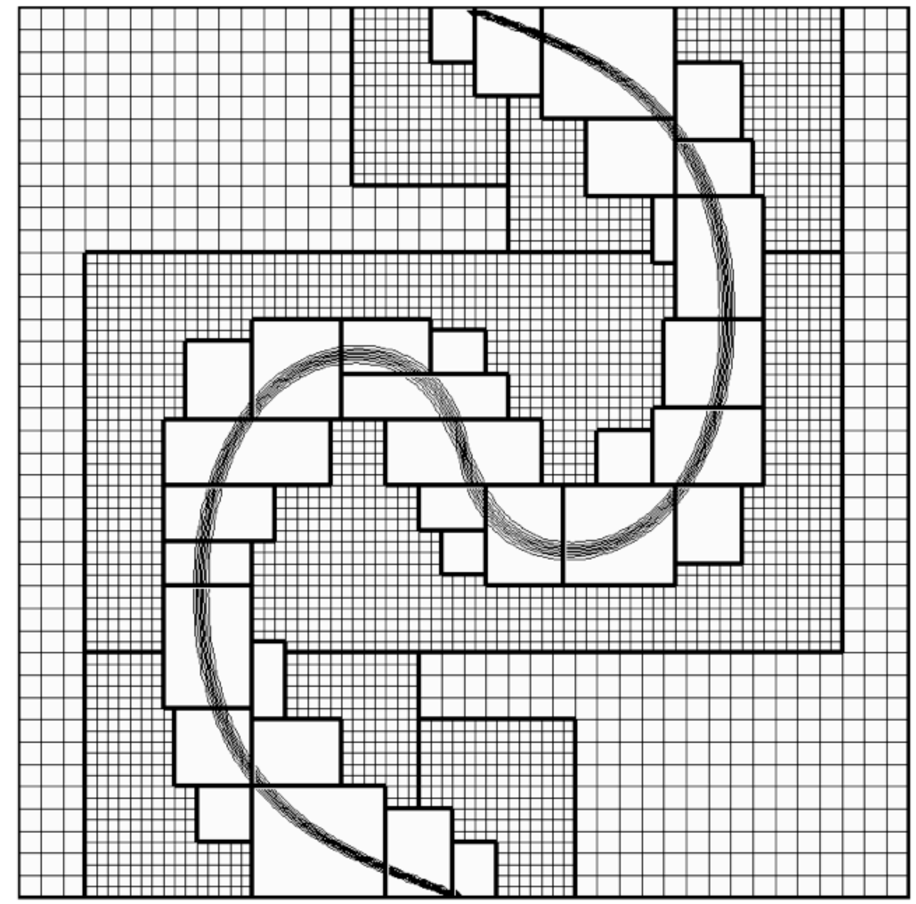




- Originally designed to improve shock capturing methods

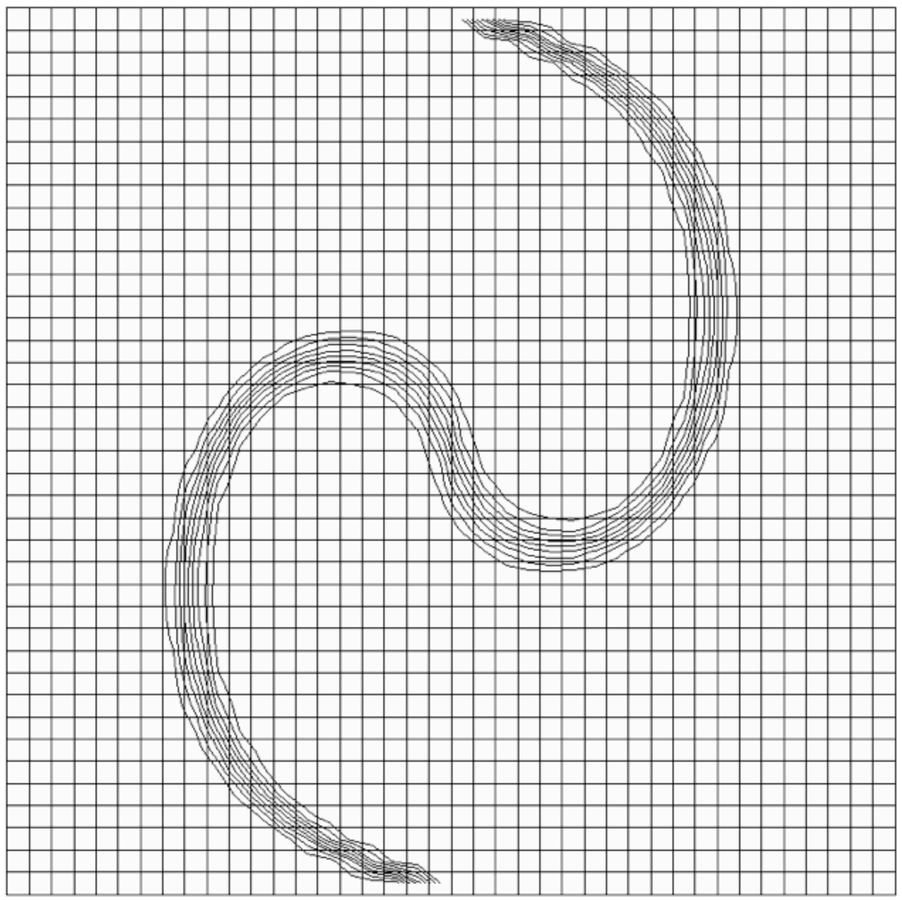
Block structured AMR (Berger, Oliger 1984)

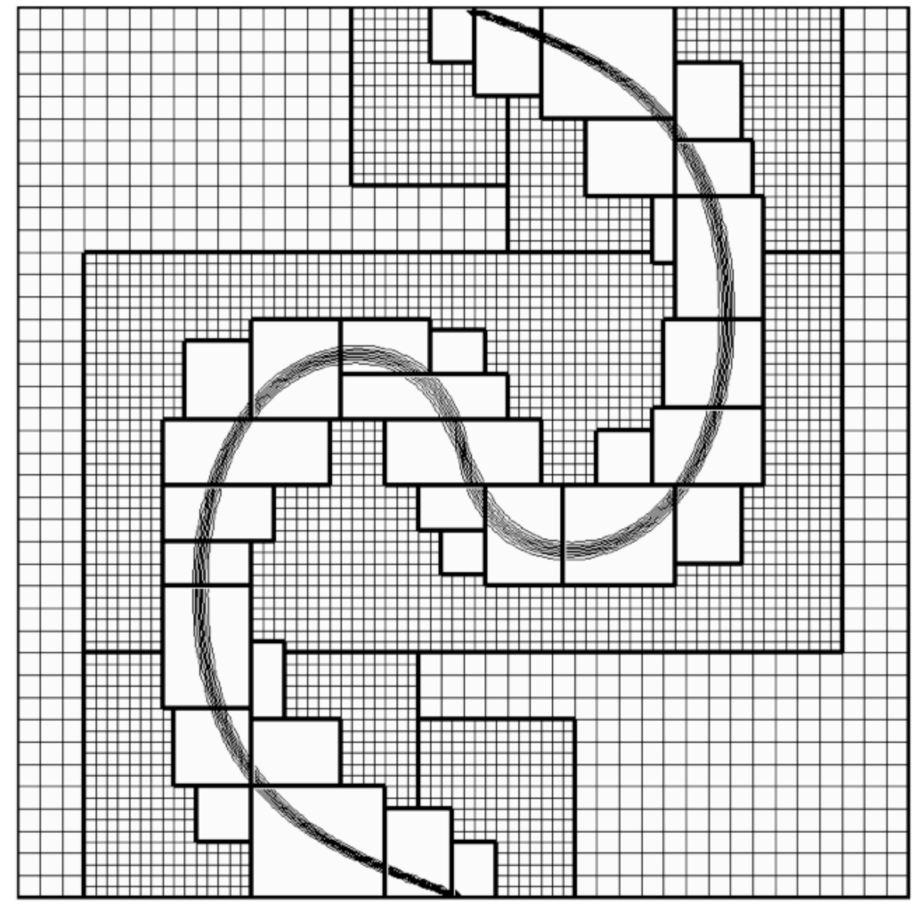




- Originally designed to improve shock capturing methods
- Gained widespread use in many application areas

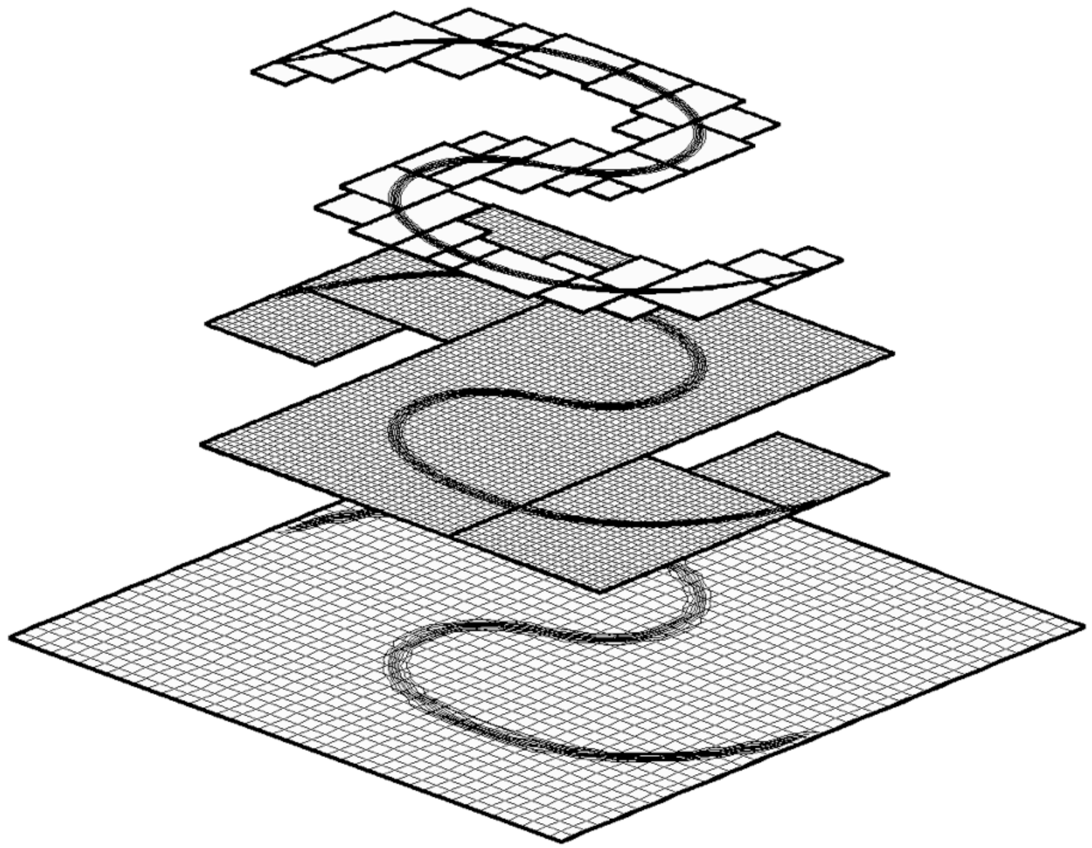
Block structured AMR (Berger, Oliger 1984)



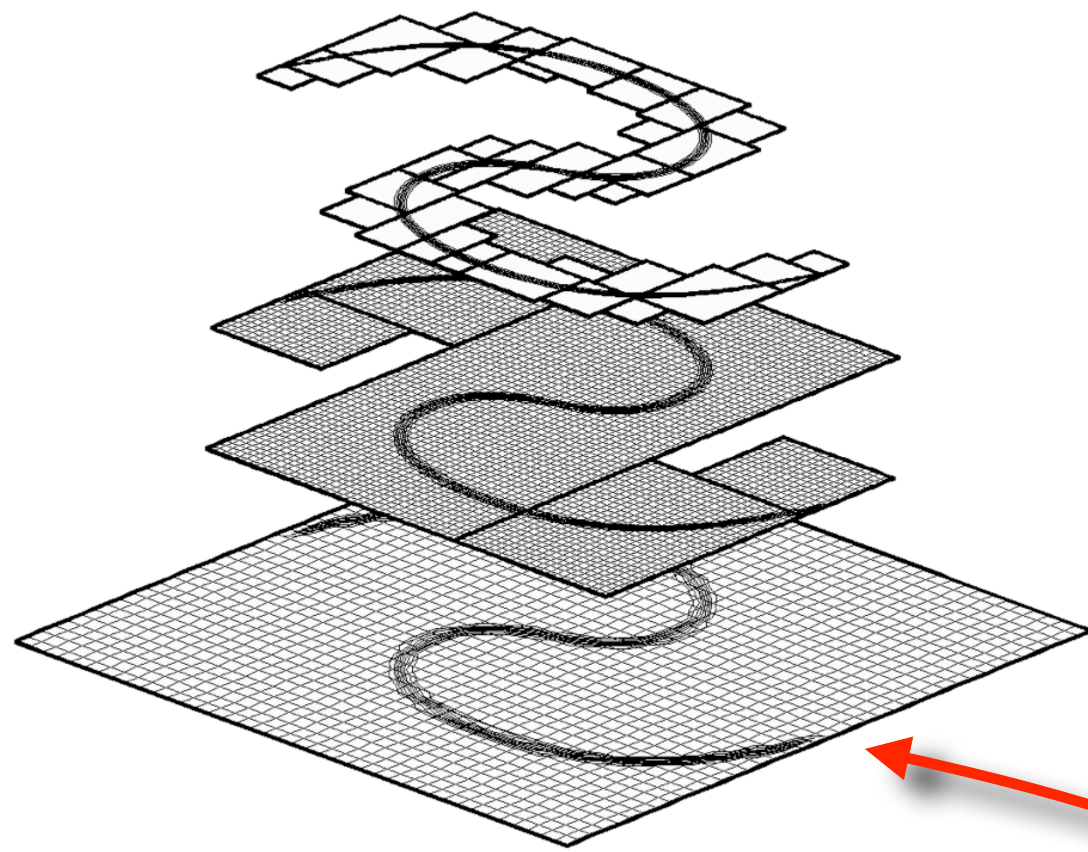


- Originally designed to improve shock capturing methods
- Gained widespread use in many application areas
- Colella, Bell, LeVeque, Almgren, Deiterding, and many others have developed methods and solvers

Block structured AMR (ala Berger and Oliger)

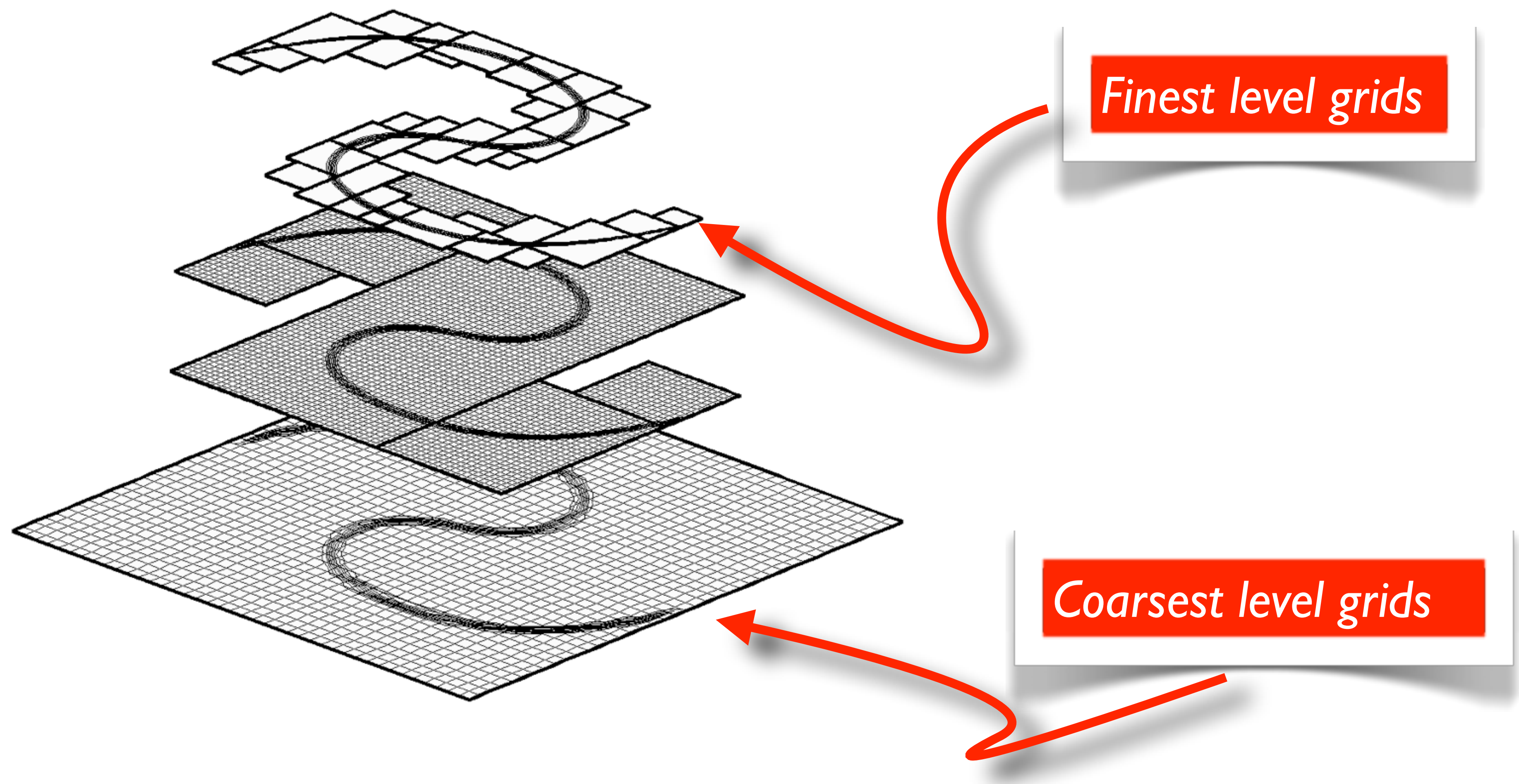


Block structured AMR (ala Berger and Oliger)

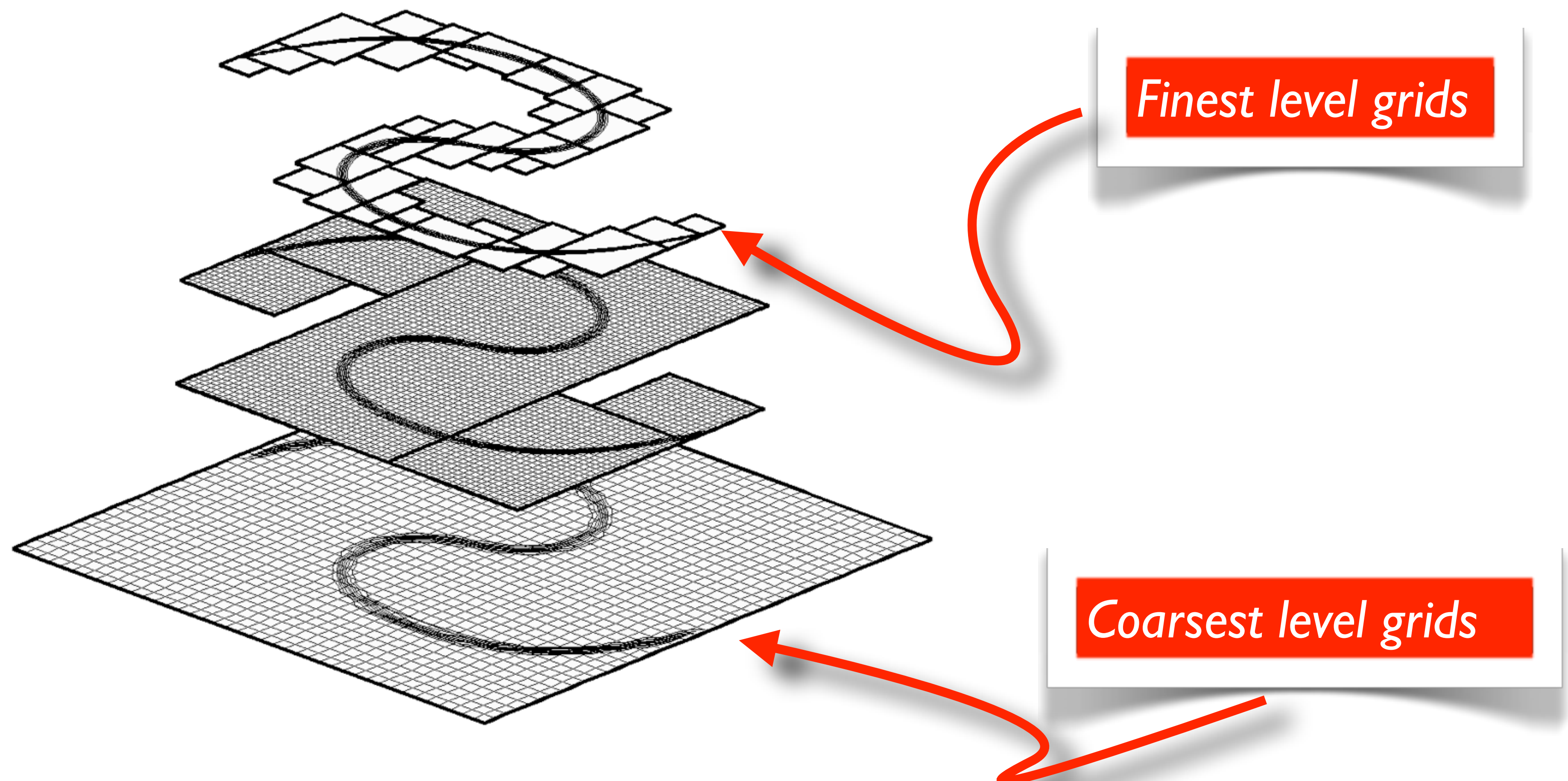


Coarsest level grids

Block structured AMR (ala Berger and Oliger)

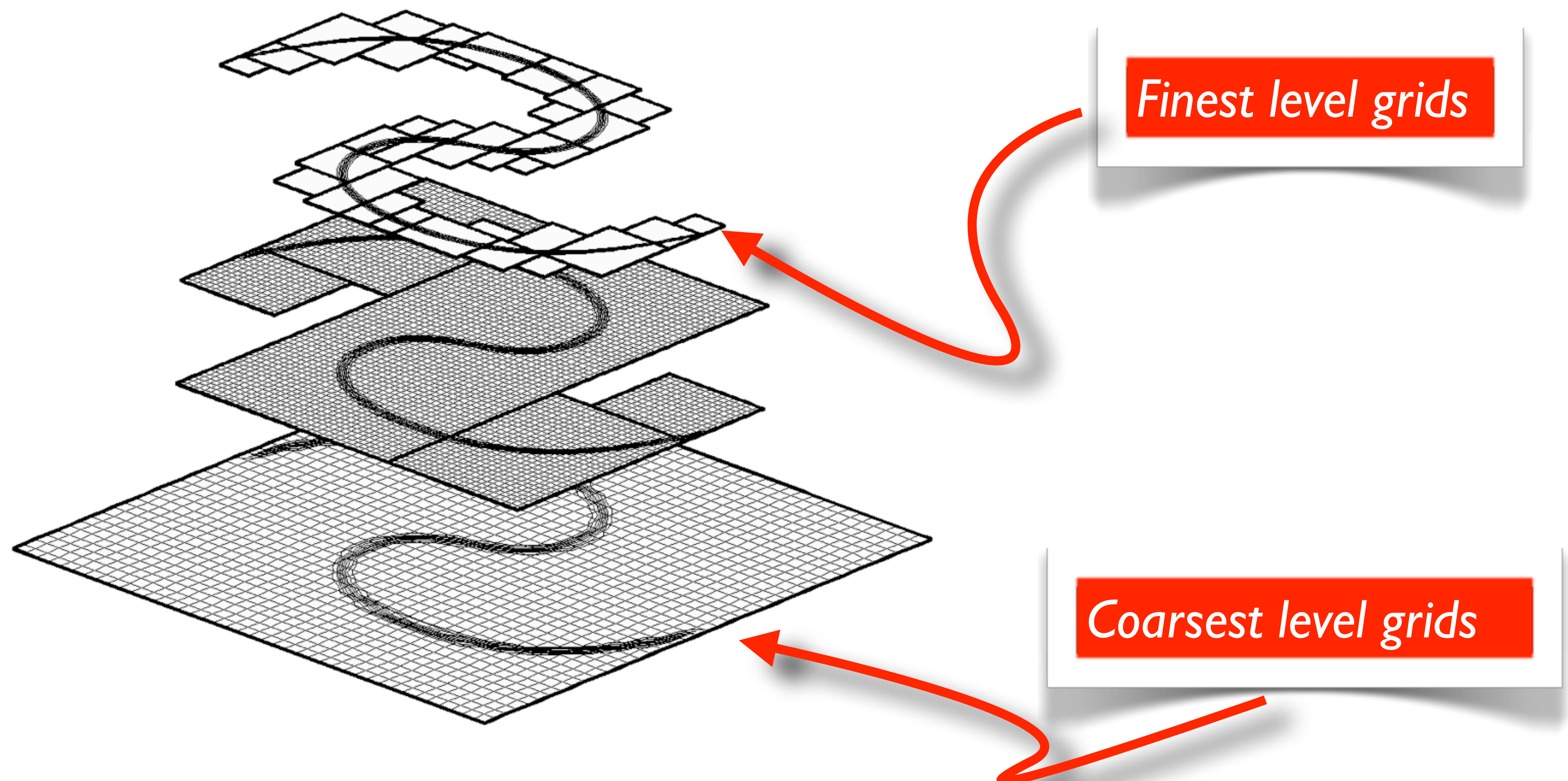


Block structured AMR (ala Berger and Oliger)



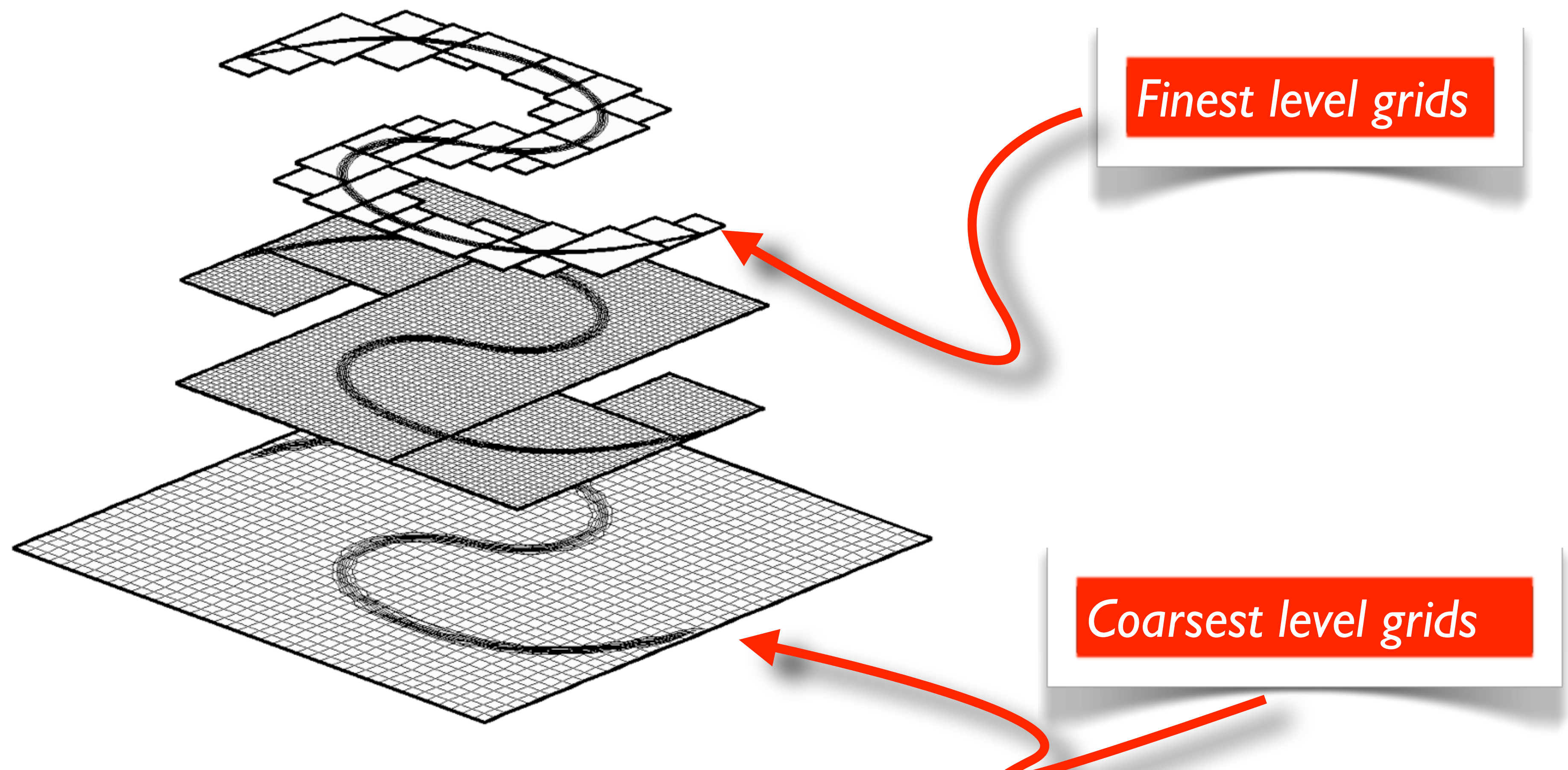
- Data is stored in nested, layered hierarchy of overlapping, logically Cartesian grids,

Block structured AMR (ala Berger and Oliger)



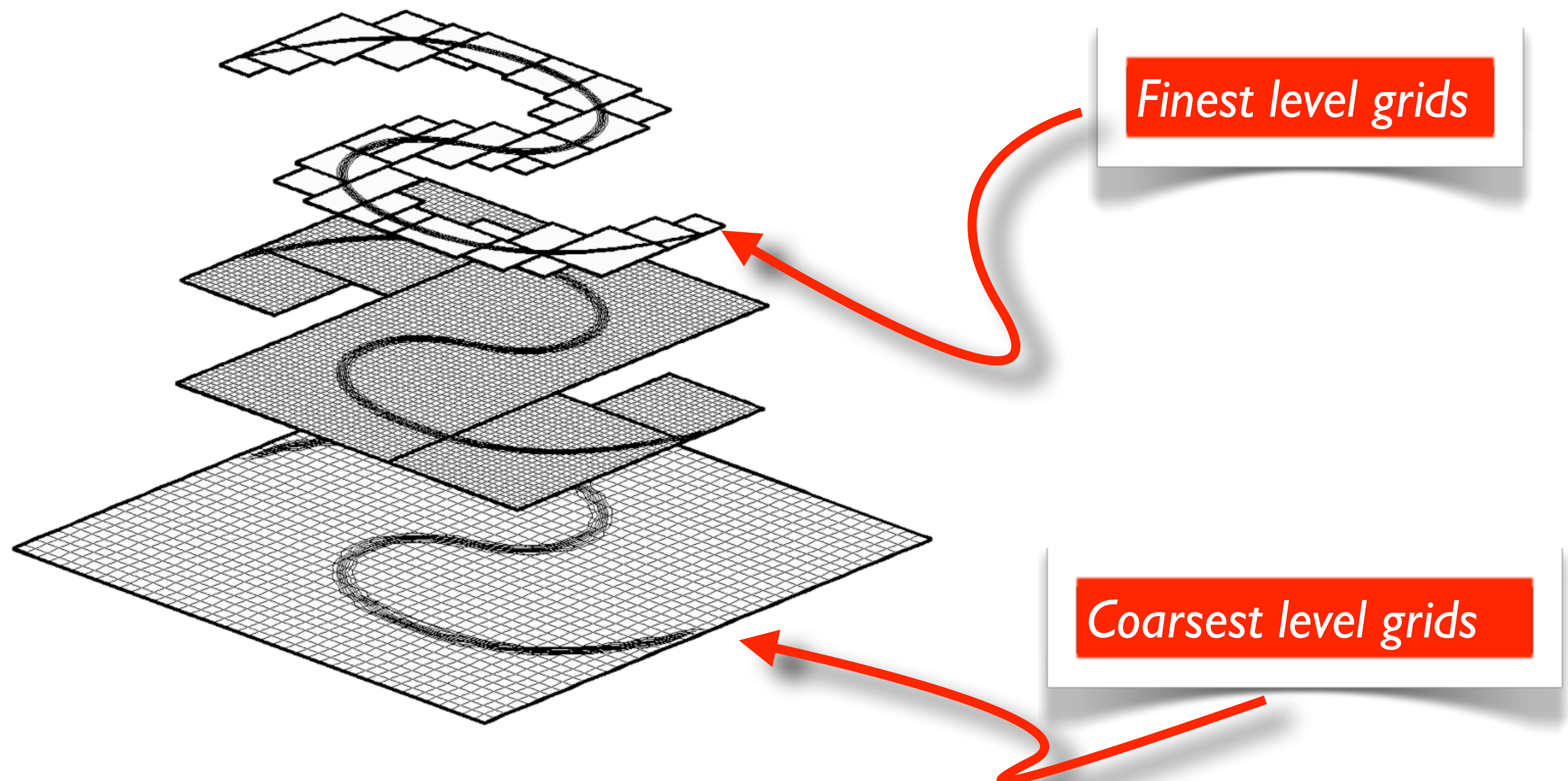
- Data is stored in nested, layered hierarchy of overlapping, logically Cartesian grids,
- Multi-rate time stepping based on mesh size,

Block structured AMR (ala Berger and Oliger)



- Data is stored in nested, layered hierarchy of overlapping, logically Cartesian grids,
- Multi-rate time stepping based on mesh size,
- Grids are dynamically refined and de-refined to adapt to the solution features of interest.

Block structured AMR (ala Berger and Oliger)



- Data is stored in nested, layered hierarchy of overlapping, logically Cartesian grids,
- Multi-rate time stepping based on mesh size,
- Grids are dynamically refined and de-refined to adapt to the solution features of interest.
- Communication between grids is done via ghost cells.

Block structured AMR rules

Block structured AMR rules

- Coarse grid and fine grid boundaries are aligned

Block structured AMR rules

- Coarse grid and fine grid boundaries are aligned
- Finer meshes are properly nested into coarser ones

Block structured AMR rules

- Coarse grid and fine grid boundaries are aligned
- Finer meshes are properly nested into coarser ones
- Ghost cell values are obtained from coarse grid or neighboring fine grids, if available

Block structured AMR rules

- Coarse grid and fine grid boundaries are aligned
- Finer meshes are properly nested into coarser ones
- Ghost cell values are obtained from coarse grid or neighboring fine grids, if available
- Do not allow grids which overlap multiple levels of refinement

Block structured AMR rules

- Coarse grid and fine grid boundaries are aligned
- Finer meshes are properly nested into coarser ones
- Ghost cell values are obtained from coarse grid or neighboring fine grids, if available
- Do not allow grids which overlap multiple levels of refinement
- Averaging fine grid solution to coarse grid; interpolate coarse grid solution to the fine grid.

Block structured AMR rules

- Coarse grid and fine grid boundaries are aligned
- Finer meshes are properly nested into coarser ones
- Ghost cell values are obtained from coarse grid or neighboring fine grids, if available
- Do not allow grids which overlap multiple levels of refinement
- Averaging fine grid solution to coarse grid; interpolate coarse grid solution to the fine grid.
- Buffer cells keep solution features on finest level

Block structured AMR rules

- Coarse grid and fine grid boundaries are aligned
- Finer meshes are properly nested into coarser ones
- Ghost cell values are obtained from coarse grid or neighboring fine grids, if available
- Do not allow grids which overlap multiple levels of refinement
- Averaging fine grid solution to coarse grid; interpolate coarse grid solution to the fine grid.
- Buffer cells keep solution features on finest level
- Grid clustering algorithm balances number of grids and refinement efficiency.

Block structured AMR philosophy

Block structured AMR philosophy

- Take advantage of existing Cartesian grid solvers whenever possible,

Block structured AMR philosophy

- Take advantage of existing Cartesian grid solvers whenever possible,
- Avoid use of complicated stencils at coarse/fine grid interfaces by operating locally on patches whenever possible

Block structured AMR philosophy

- Take advantage of existing Cartesian grid solvers whenever possible,
- Avoid use of complicated stencils at coarse/fine grid interfaces by operating locally on patches whenever possible
- Numerical solution on grid hierarchy should have the same order of accuracy as the single grid algorithm.

Block structured AMR philosophy

- Take advantage of existing Cartesian grid solvers whenever possible,
- Avoid use of complicated stencils at coarse/fine grid interfaces by operating locally on patches whenever possible
- Numerical solution on grid hierarchy should have the same order of accuracy as the single grid algorithm.
- Conservation should be maintained if PDE is in conservative form

Block structured AMR philosophy

- Take advantage of existing Cartesian grid solvers whenever possible,
- Avoid use of complicated stencils at coarse/fine grid interfaces by operating locally on patches whenever possible
- Numerical solution on grid hierarchy should have the same order of accuracy as the single grid algorithm.
- Conservation should be maintained if PDE is in conservative form
- Overhead in managing multiple grid levels should not impact performance significantly

Block structured AMR philosophy

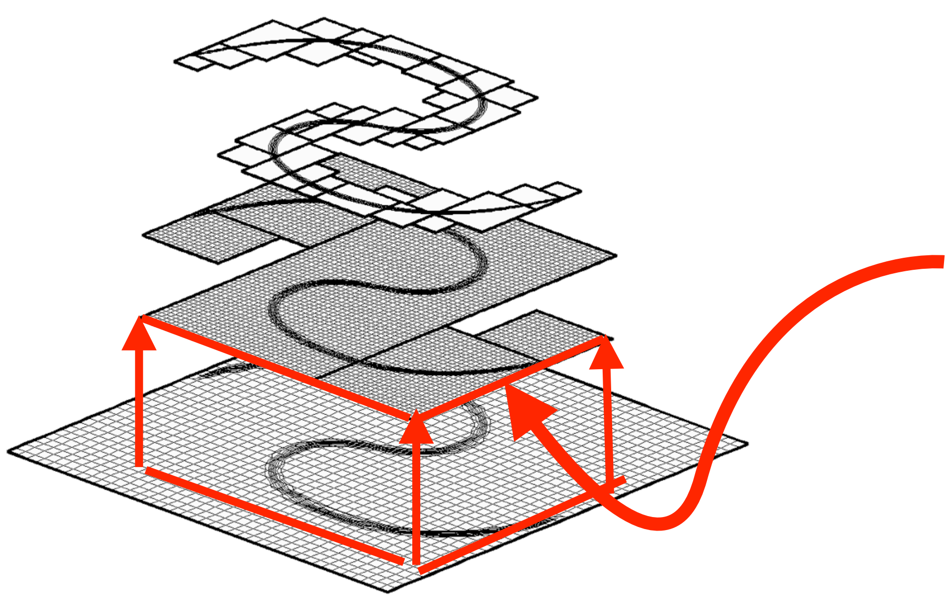
- Take advantage of existing Cartesian grid solvers whenever possible,
- Avoid use of complicated stencils at coarse/fine grid interfaces by operating locally on patches whenever possible
- Numerical solution on grid hierarchy should have the same order of accuracy as the single grid algorithm.
- Conservation should be maintained if PDE is in conservative form
- Overhead in managing multiple grid levels should not impact performance significantly
- Subcycle finer grid time steps (multi-rate time stepping)

Explicit, single step multi-rate time stepping

A single time step advance, assuming a refinement factor of R.

1. Advance at the coarsest level by time step Δt
2. Interpolate coarse grid solution to fine grid ghost cells
3. Advance fine grid R time steps, by a time step $\Delta t/R$
4. Average solution from fine grids to coarse grid,
5. Adjust coarse grid solution to assure flux continuity at the coarse/fine boundaries,
6. Tag cells for refinement and regrid

Grids at the same level exchange ghost cell values directly



Fine grid boundary conditions interpolated in space and time from coarse grid

Software for AMR

How can I add adaptivity to my existing (Cartesian) code?

- It is much harder than it looks (and you don't really just "add" adaptivity)
- And there are several general purpose codes already available which can use your single grid solver.

Block structured AMR codes

- General purpose (freely available) block-structured codes
  - **PARAMESH** (NASA/Goddard)
  - **SAMRAI** (Lawrence Livermore National Lab)
  - **BoxLib** (Lawrence Berkeley Lab)
  - **Chombo** (Lawrence Berkeley Lab)
  - **AMRClaw** (University of Washington/NYU)
- All are large frameworks, with many developers
- Mostly C++ and Fortran libraries (no GUIs) that started life as research codes.

See my website for a list of several more application specific codes

Block structured AMR codes

"PARAMESH is a package of Fortran 90 subroutines designed to provide an application developer with an easy route to extend an existing serial code which uses logically Cartesian structured mesh into a parallel code with adaptive mesh refinement"

SAMRAI - "Object oriented C++ library developed to provide algorithmic and software support to large scale multiphysics problems relevant to the US Department of Energy (DOE)"

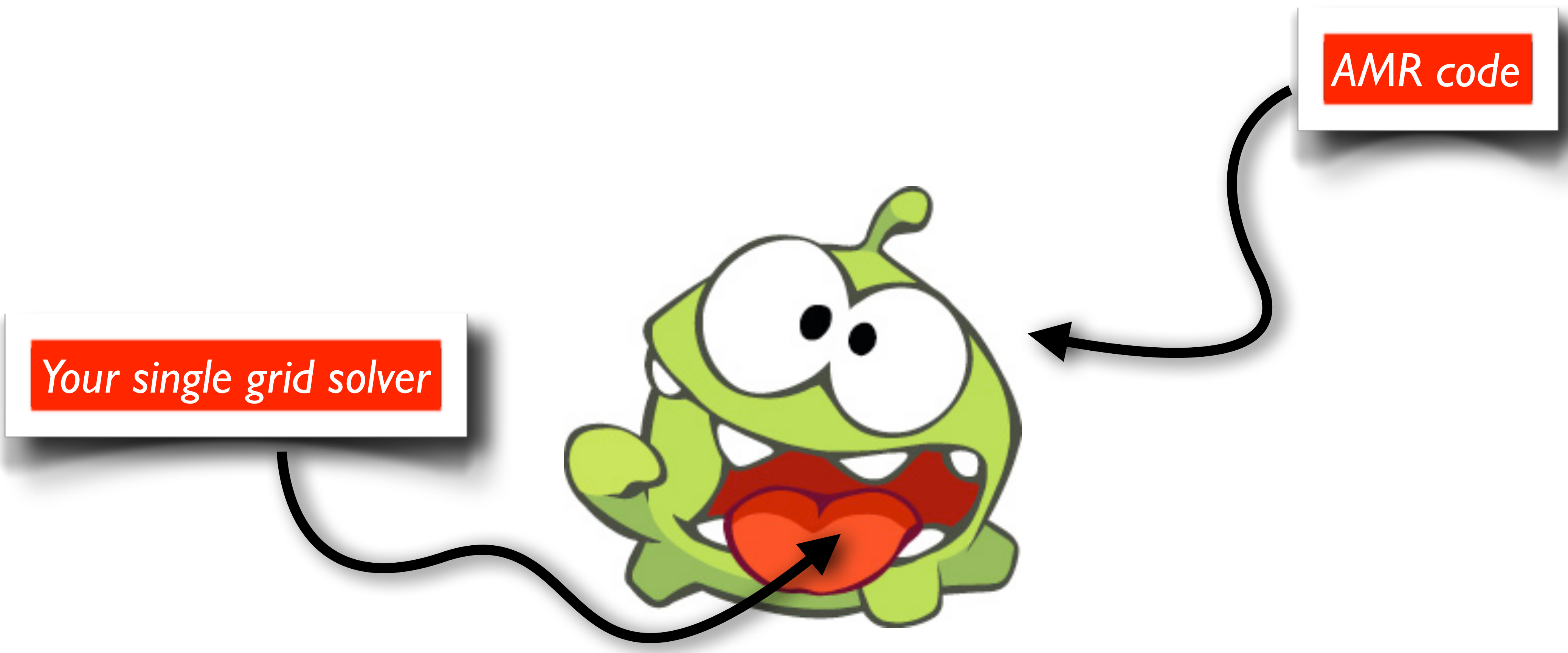
Boxlib - "These libraries provide the software infrastructure for the computational activities (combustion, astrophysics, porous media flow) at the Center for Computational Sciences and Engineering (CCSE) at Lawrence Berkeley Labs"

Using block structured AMR codes

Mental model of how this might work :

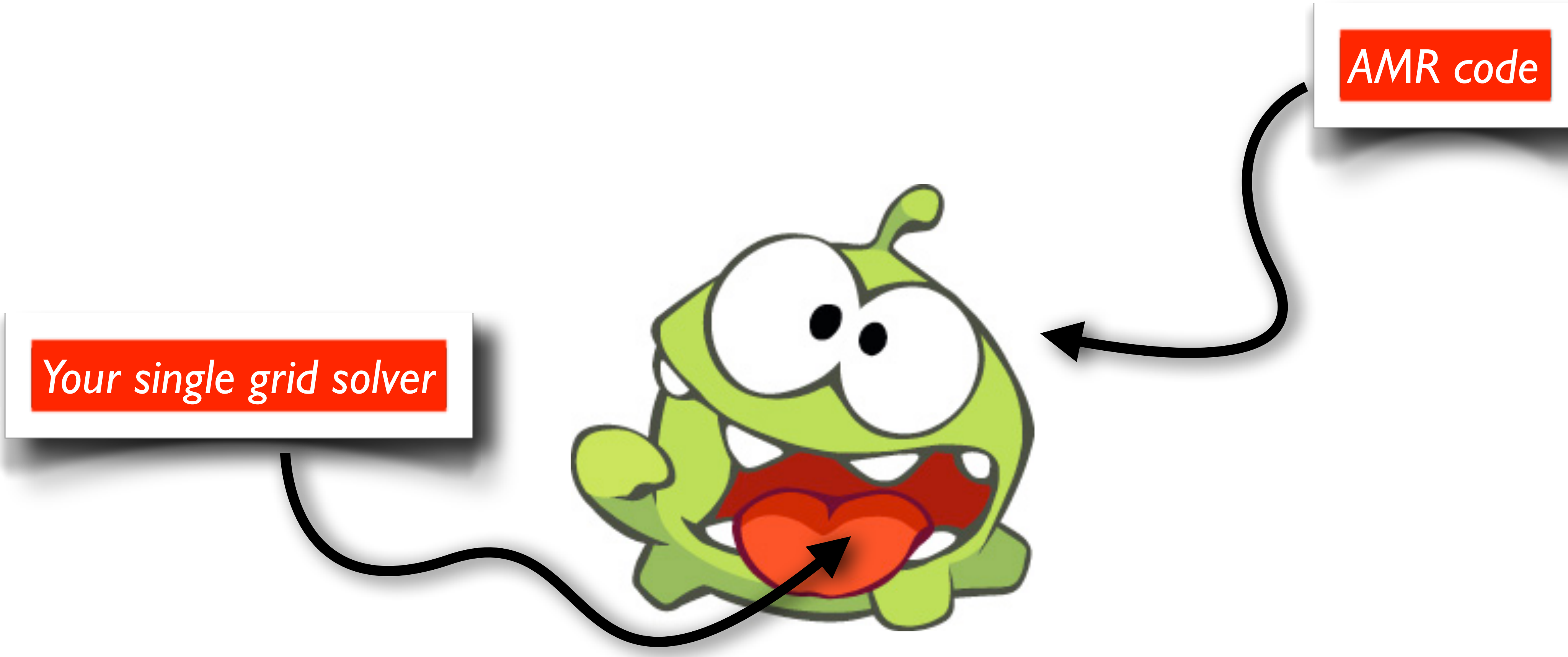
Using block structured AMR codes

Mental model of how this might work :



Using block structured AMR codes

Mental model of how this might work :



** Idea for code name : OmNum*

Block structured AMR codes

The Dream

Block structured AMR codes

The Dream

```
AMR.run(max_time, max_steps);
```

Block structured AMR codes

The Dream

```
AMR.run(max_time, max_steps);
```

Your single grid solver is called from here.

Block structured AMR codes

The Dream

```
AMR.run(max_time, max_steps);
```

Your single grid solver is called from here.

Block structured AMR codes

The Reality

```
Tuple< RefCountedPtr<AMRLevelOpFactory<
LevelData<FArrayBox> > >, SpaceDim> velTGAOpFactoryPtrs;

for (int idir = 0; idir < SpaceDim; idir++)
{velTGAOpFactoryPtrs[idir] =
    RefCountedPtr<AMRLevelOpFactory<LevelData
    <FArrayBox> > >
    ((AMRLevelOpFactory<LevelData<FArrayBox> >*)
    (new AMRPoissonOpFactory())); //.....
```

Block structured AMR codes

The Reality

```
Tuple< RefCountedPtr<AMRLevelOpFactory<
LevelData<FArrayBox> > >, SpaceDim> velTGAOpFactoryPtrs;

for (int idir = 0; idir < SpaceDim; idir++)
{velTGAOpFactoryPtrs[idir] =
    RefCountedPtr<AMRLevelOpFactory<LevelData
    <FArrayBox> > >
    ((AMRLevelOpFactory<LevelData<FArrayBox> >*)
    (new AMRPoissonOpFactory())); //.....
```

Your single grid solver

Block structured AMR codes

The Reality

```
Tuple< RefCountedPtr<AMRLevelOpFactory<
LevelData<FArrayBox> > >, SpaceDim> velTGAOpFactoryPtrs;

for (int idir = 0; idir < SpaceDim; idir++)
{velTGAOpFactoryPtrs[idir] =
    RefCountedPtr<AMRLevelOpFactory<LevelData
    <FArrayBox> > >
    ((AMRLevelOpFactory<LevelData<FArrayBox> >*)
    (new AMRPoissonOpFactory())); //.....
```

Your single grid solver

you

Retro...

```
node(ndjhi,mptrnx) = node(ndjhi,mptr)
node(ndjhi,mptr)   = node(ndjlo,mptr) + nyl - 1
node(ndjlo,mptrnx) = node(ndjhi,mptr) + 1
node(ndihi,mptrnx) = node(ndihi,mptr)
node(ndilo,mptrnx) = node(ndilo,mptr)

rnode(cornxlo,mptrnx)   = cxlo
rnode(cornylo,mptrnx)   = cymid
rnode(cornyhi,mptrnx)   = cyhi
rnode(cornxhi,mptrnx)   = cxhi
node(nestlevel,mptrnx)  = node(nestlevel,mptr)
rnode(timemult,mptrnx)  = rnode(timemult,mptr)
go to 10
```

Retro...

```
node(ndjhi,mptrnx) = node(ndjhi,mptr)
node(ndjhi,mptr)   = node(ndjlo,mptr) + nyl - 1
node(ndjlo,mptrnx) = node(ndjhi,mptr) + 1
node(ndihi,mptrnx) = node(ndihi,mptr)
node(ndilo,mptrnx) = node(ndilo,mptr)

rnode(cornxlo,mptrnx)    = cxlo
rnode(cornylo,mptrnx)    = cymid
rnode(cornyhi,mptrnx)    = cyhi
rnode(cornxhi,mptrnx)    = cxhi
node(nestlevel,mptrnx)   = node(nestlevel,mptr)
rnode(timemult,mptrnx)   = rnode(timemult,mptr)
go to 10
```

Why are AMR codes challenging to develop?

Why are AMR codes challenging to develop?

- Heterogeneous data structures associated with storing hierarchy of grids,

Why are AMR codes challenging to develop?

- Heterogeneous data structures associated with storing hierarchy of grids,
- Dynamically creating and destroying grids :

Why are AMR codes challenging to develop?

- Heterogeneous data structures associated with storing hierarchy of grids,
- Dynamically creating and destroying grids :
  - Need a "factory" paradigm to create new grids and any auxiliary data arrays (material properties, metric terms, bathymetry, etc) that go with each new grid,

Why are AMR codes challenging to develop?

- Heterogeneous data structures associated with storing hierarchy of grids,
- Dynamically creating and destroying grids :
  - Need a "factory" paradigm to create new grids and any auxiliary data arrays (material properties, metric terms, bathymetry, etc) that go with each new grid,
- Periodic boundary conditions and multi-block domains,

Why are AMR codes challenging to develop?

- Heterogeneous data structures associated with storing hierarchy of grids,
- Dynamically creating and destroying grids :
  - Need a "factory" paradigm to create new grids and any auxiliary data arrays (material properties, metric terms, bathymetry, etc) that go with each new grid,
- Periodic boundary conditions and multi-block domains,
- Providing support for multi-rate time stepping,

Why are AMR codes challenging to develop?

- Heterogeneous data structures associated with storing hierarchy of grids,
- Dynamically creating and destroying grids :
  - Need a "factory" paradigm to create new grids and any auxiliary data arrays (material properties, metric terms, bathymetry, etc) that go with each new grid,
- Periodic boundary conditions and multi-block domains,
- Providing support for multi-rate time stepping,
- Storing information at grid boundaries needed to couple the solution on grids

Why are AMR codes challenging to develop?

- Heterogeneous data structures associated with storing hierarchy of grids,
- Dynamically creating and destroying grids :
    - Need a “factory” paradigm to create new grids and any auxiliary data arrays (material properties, metric terms, bathymetry, etc) that go with each new grid,
- Periodic boundary conditions and multi-block domains,
- Providing support for multi-rate time stepping,
- Storing information at grid boundaries needed to couple the solution on grids
- Parallel load balancing for both the solution step and refinement step

Why can AMR codes be difficult to use?

Why can AMR codes be difficult to use?

- Time stepping beyond just single step explicit schemes (IMEX, SSP, RK, ...)

Why can AMR codes be difficult to use?

- Time stepping beyond just single step explicit schemes (IMEX, SSP, RK, ...)
- Understanding how overall time stepping interacts with dynamic grid creation, destruction and management.

Why can AMR codes be difficult to use?

- Time stepping beyond just single step explicit schemes (IMEX, SSP, RK, ...)
- Understanding how overall time stepping interacts with dynamic grid creation, destruction and management.
- Implicit solvers,

Why can AMR codes be difficult to use?

- Time stepping beyond just single step explicit schemes (IMEX, SSP, RK, ...)
- Understanding how overall time stepping interacts with dynamic grid creation, destruction and management.
- Implicit solvers,
- Coupling of multiple solvers (e.g. advection + diffusion + elliptic solvers + source terms),

Why can AMR codes be difficult to use?

- Time stepping beyond just single step explicit schemes (IMEX, SSP, RK, ...)
- Understanding how overall time stepping interacts with dynamic grid creation, destruction and management.
- Implicit solvers,
- Coupling of multiple solvers (e.g. advection + diffusion + elliptic solvers + source terms),
- Multi-physics,

Why can AMR codes be difficult to use?

- Time stepping beyond just single step explicit schemes (IMEX, SSP, RK, ...)
- Understanding how overall time stepping interacts with dynamic grid creation, destruction and management.
- Implicit solvers,
- Coupling of multiple solvers (e.g. advection + diffusion + elliptic solvers + source terms),
- Multi-physics,
- Data visualization

Why can AMR codes be difficult to use?

- Time stepping beyond just single step explicit schemes (IMEX, SSP, RK, ...)
- Understanding how overall time stepping interacts with dynamic grid creation, destruction and management.
- Implicit solvers,
- Coupling of multiple solvers (e.g. advection + diffusion + elliptic solvers + source terms),
- Multi-physics,
- Data visualization
- Computing diagnostics on a nested grid hierarchy,

Why can AMR codes be difficult to use?

- Time stepping beyond just single step explicit schemes (IMEX, SSP, RK, ...)
- Understanding how overall time stepping interacts with dynamic grid creation, destruction and management.
- Implicit solvers,
- Coupling of multiple solvers (e.g. advection + diffusion + elliptic solvers + source terms),
- Multi-physics,
- Data visualization
- Computing diagnostics on a nested grid hierarchy,
- Error estimation, tuning for efficient use of grids, ...

But what if you have ideas about ...

- Multi-stage, multi-step, IMEX, SSP, parallel-in-time, exponential integrators, and other time stepping schemes in an adaptive setting,
- Accuracy of multi-rate schemes for PDEs with mixed elliptic/parabolic/hyperbolic terms.
- Elliptic and parabolic solvers (iterative? direct? Explicit? Fast multipole?)
- Parallelism in the AMR setting?
- Error estimation
- Higher order accuracy
- Complex physics

Should you write yet-another-AMR code?

ForestClaw : A hybrid approach to AMR

ForestClaw : A hybrid approach to AMR

Use an existing parallel quad/octree library to do the grid management

ForestClaw : A hybrid approach to AMR

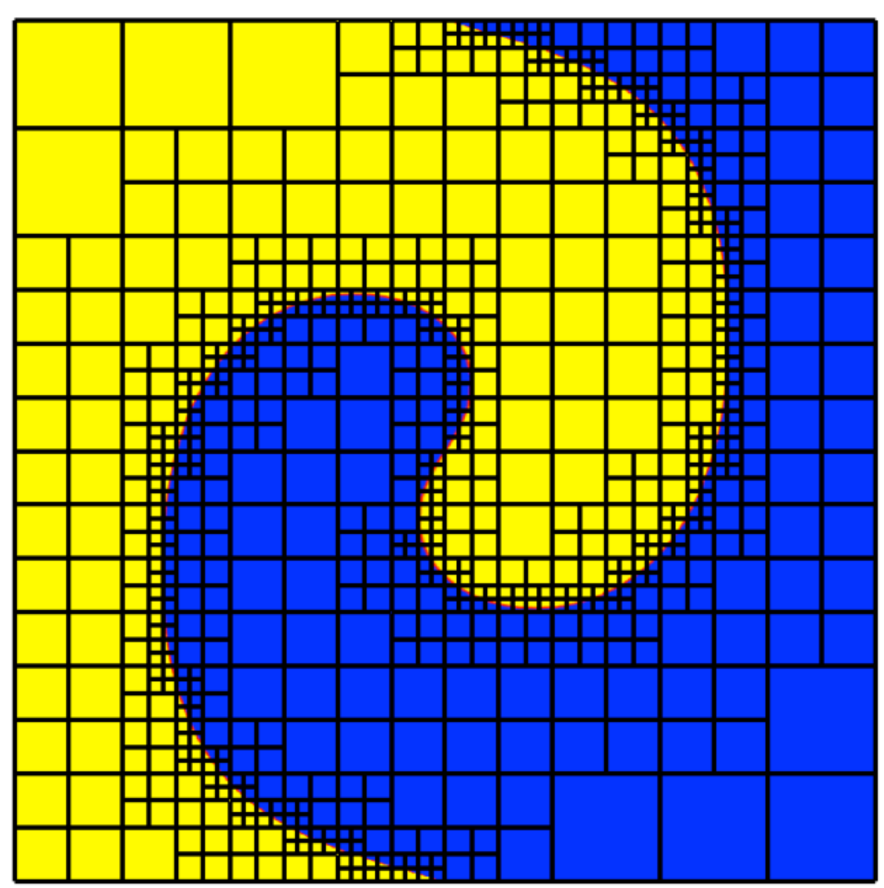
Use an existing parallel quad/octree library to do the grid management

- Store fixed sized non-overlapping grids as leaves in a tree

ForestClaw : A hybrid approach to AMR

Use an existing parallel quad/octree library to do the grid management

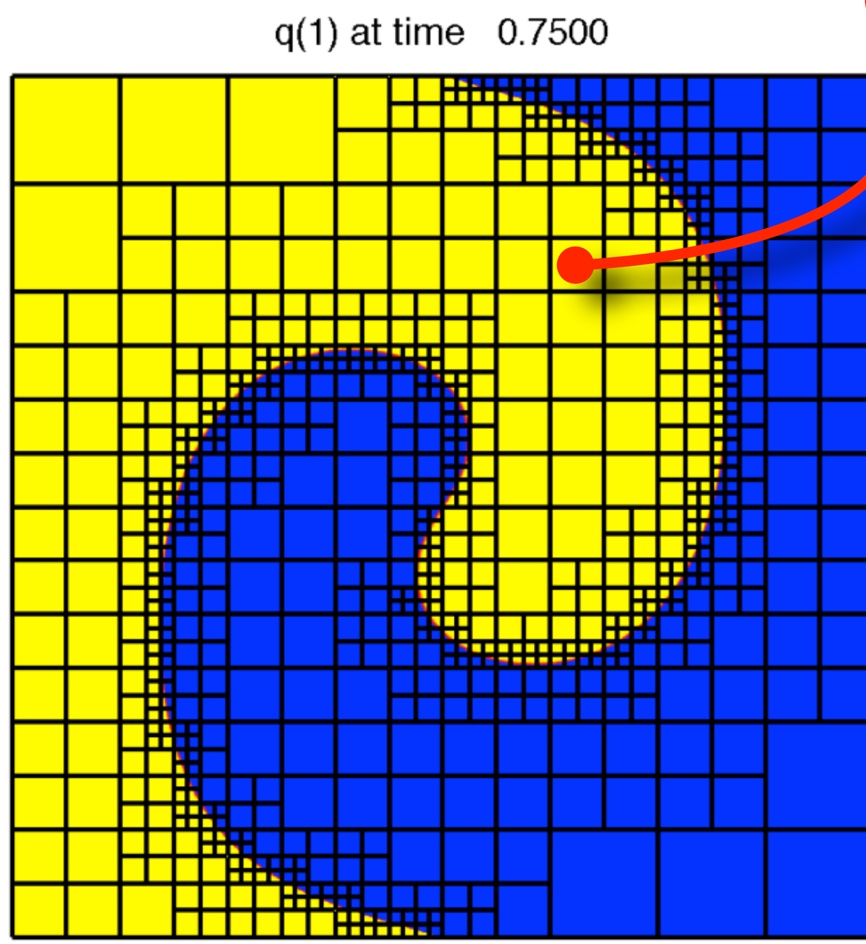
- Store fixed sized non-overlapping grids as leaves in a tree



ForestClaw : A hybrid approach to AMR

Use an existing parallel quad/octree library to do the grid management

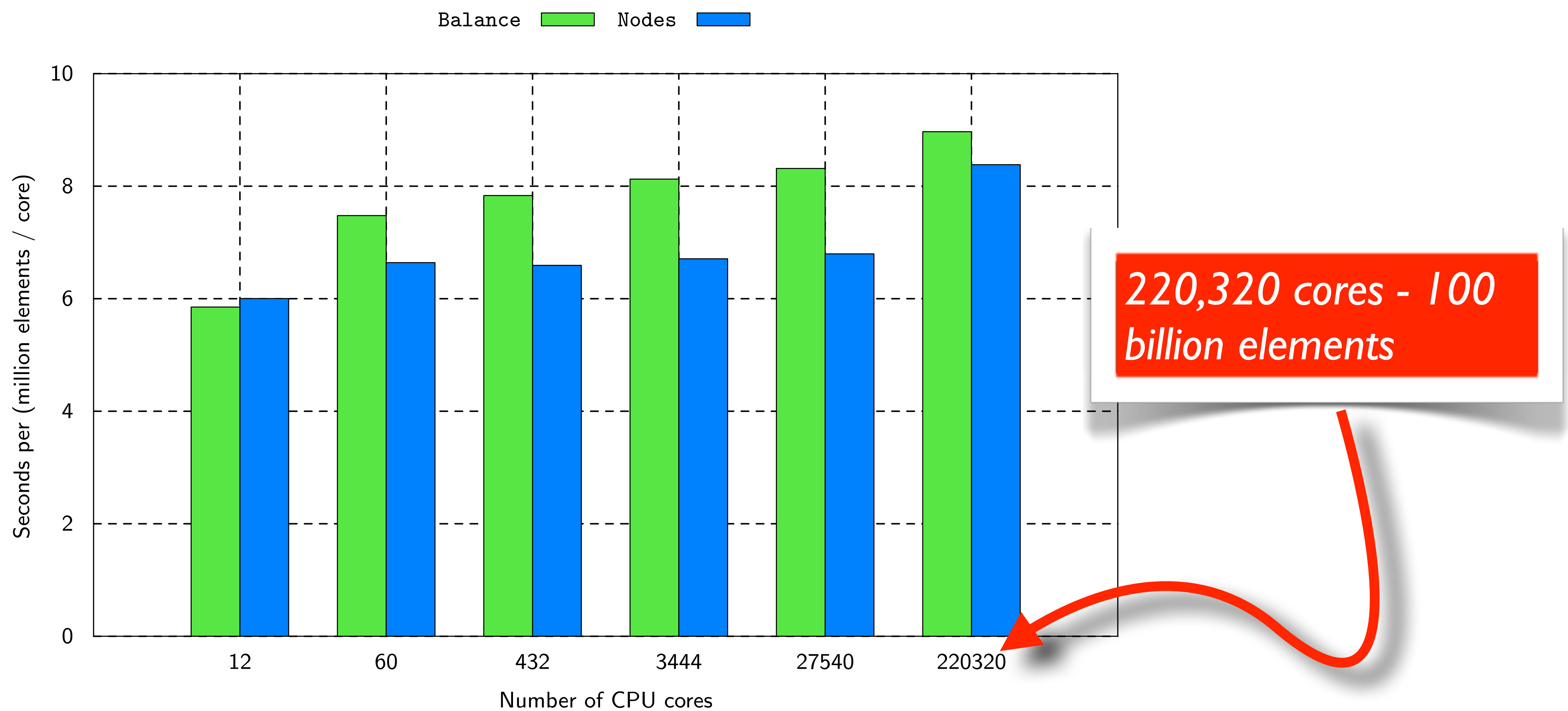
- Store fixed sized non-overlapping grids as leaves in a tree



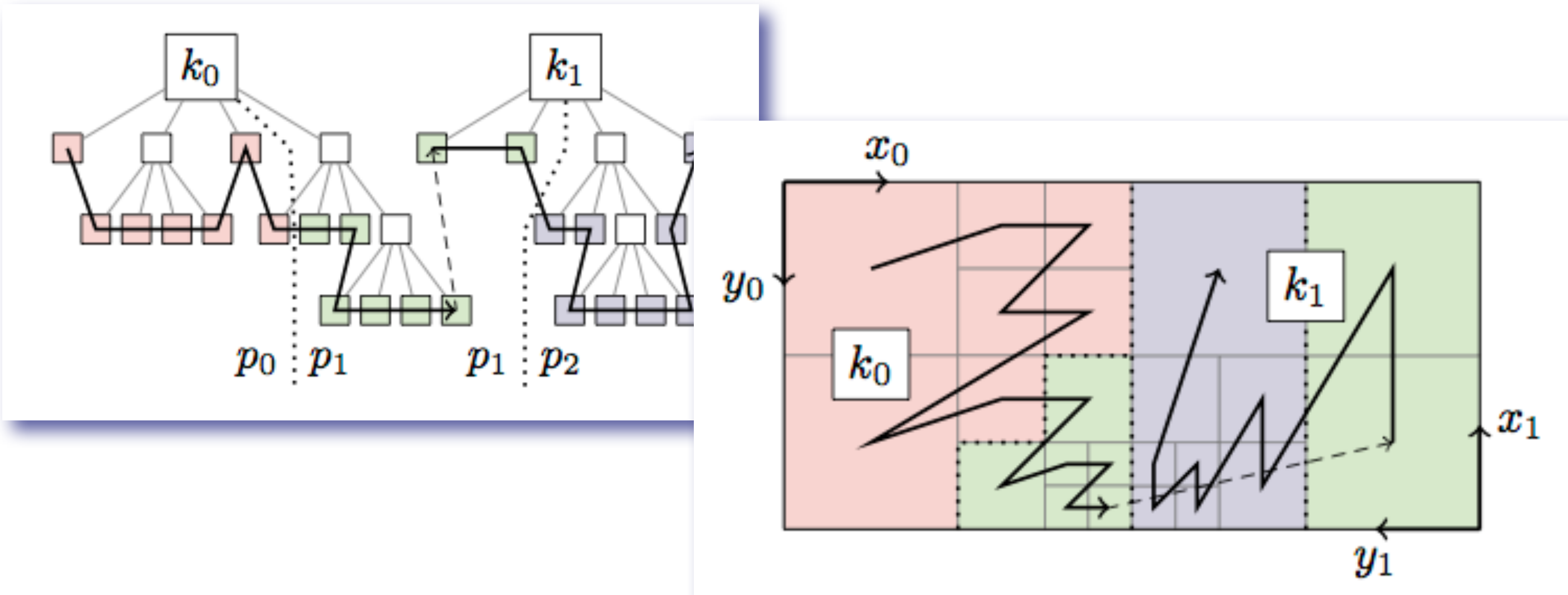
Tree structure based on refining quadrants, but each quadrant is itself a grid

p4est

- Parallel, multi-block code for managing a forest of adaptive quad- or octrees.
- Highly scalable on realistic applications of interest
- Developed by Carsten Burstedde (Univ. of Bonn), with Wilcox, Ghattas and others



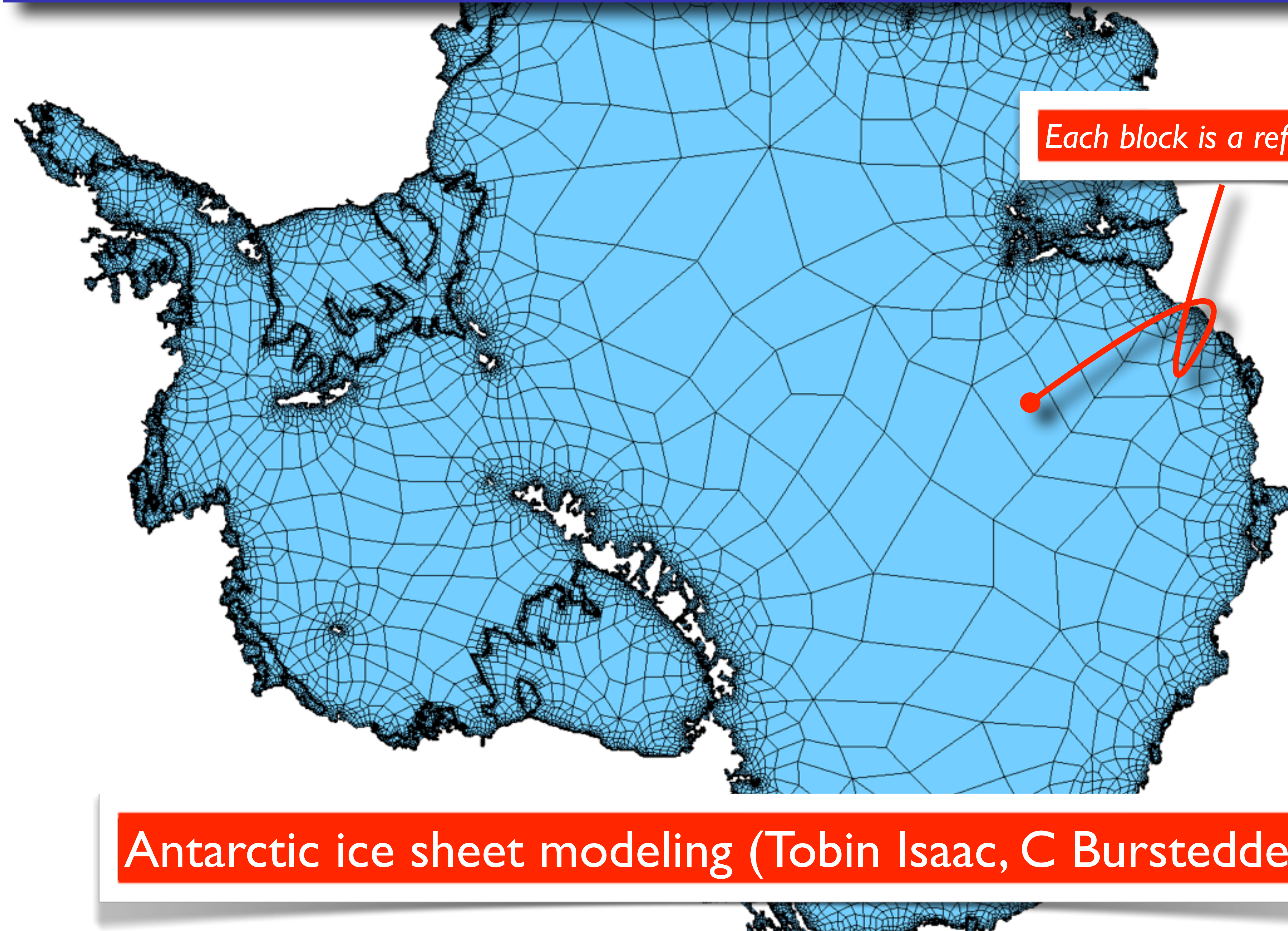
p4est



High scalability is achieved while preserving data locality by using space-filling curves.

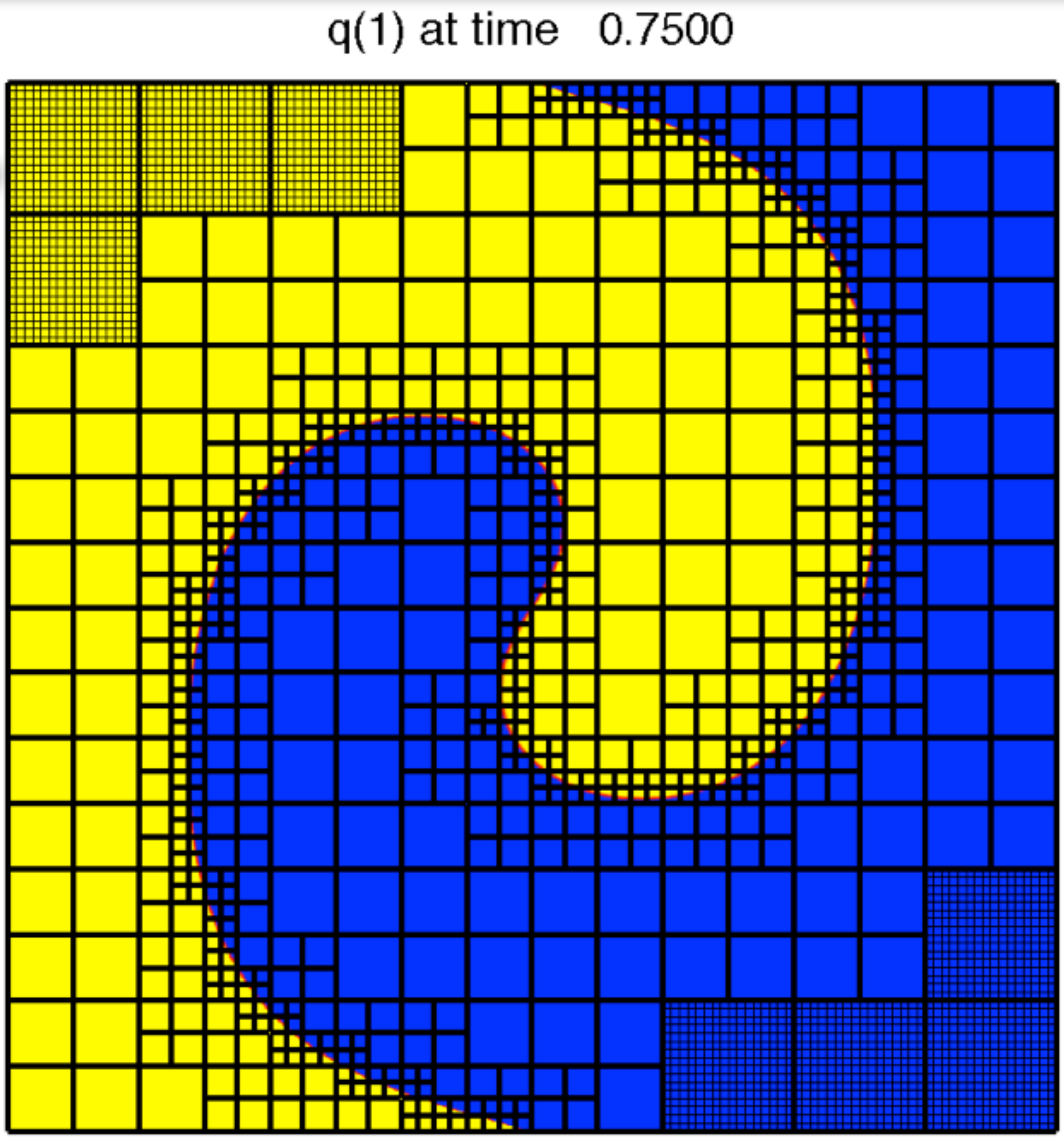
Carsten Burstedde, Lucas C. Wilcox, and Omar Ghattas, "p4est: Scalable Algorithms for Parallel Adaptive Mesh Refinement on Forests of Octrees", SISC (2011)

Multi-block support in p4est



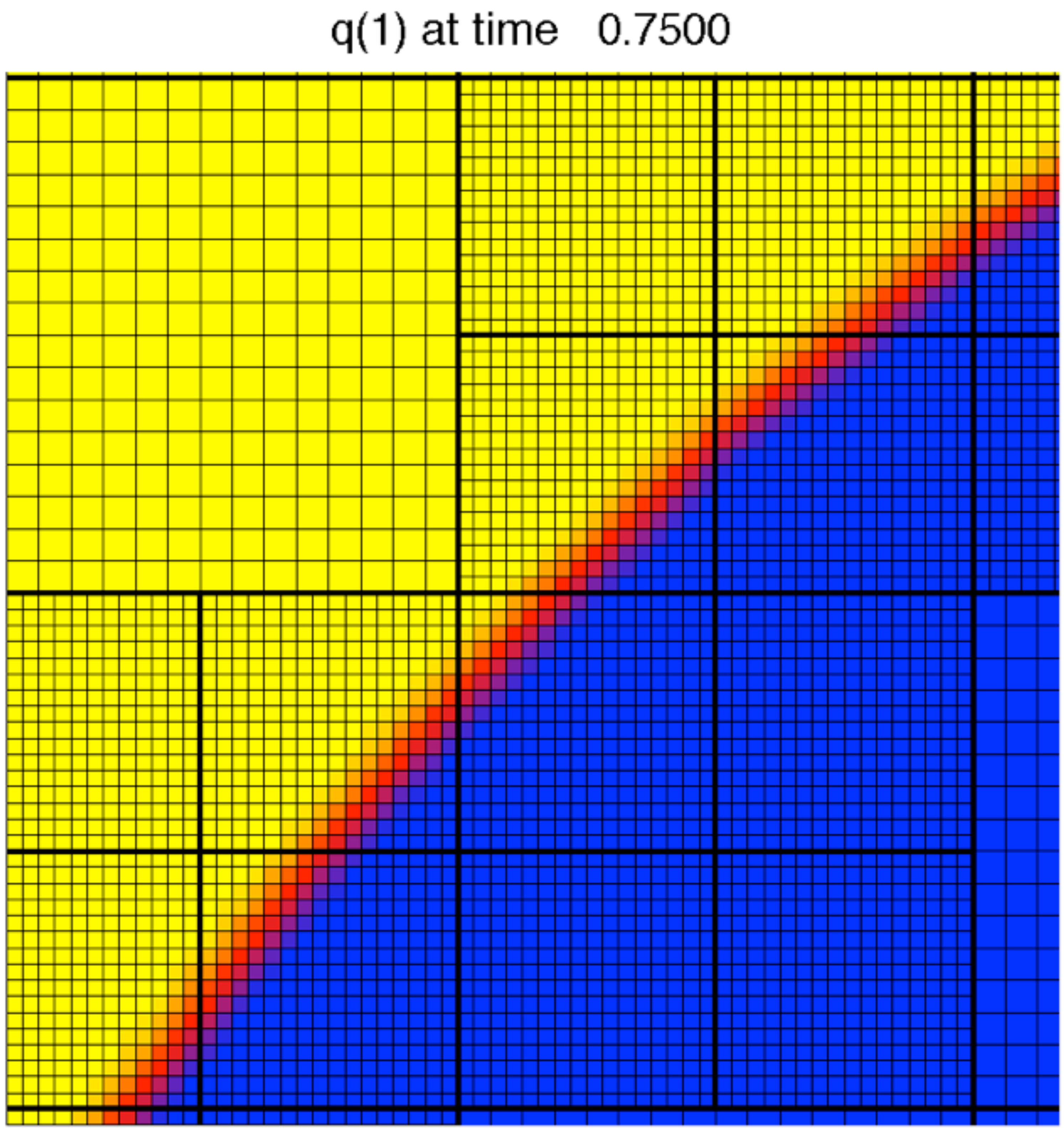
Antarctic ice sheet modeling (Tobin Isaac, C Burstedde)

A hybrid approach to AMR



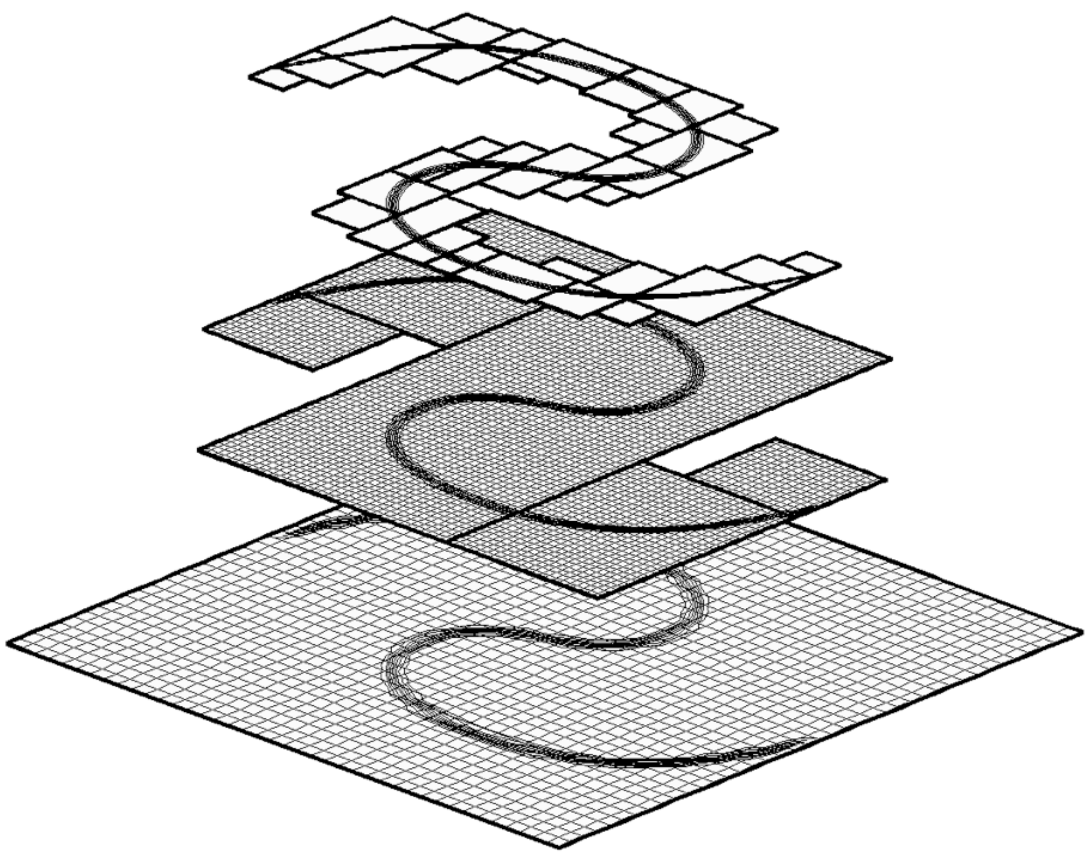
Each leaf is a fixed size grid

A hybrid approach to AMR



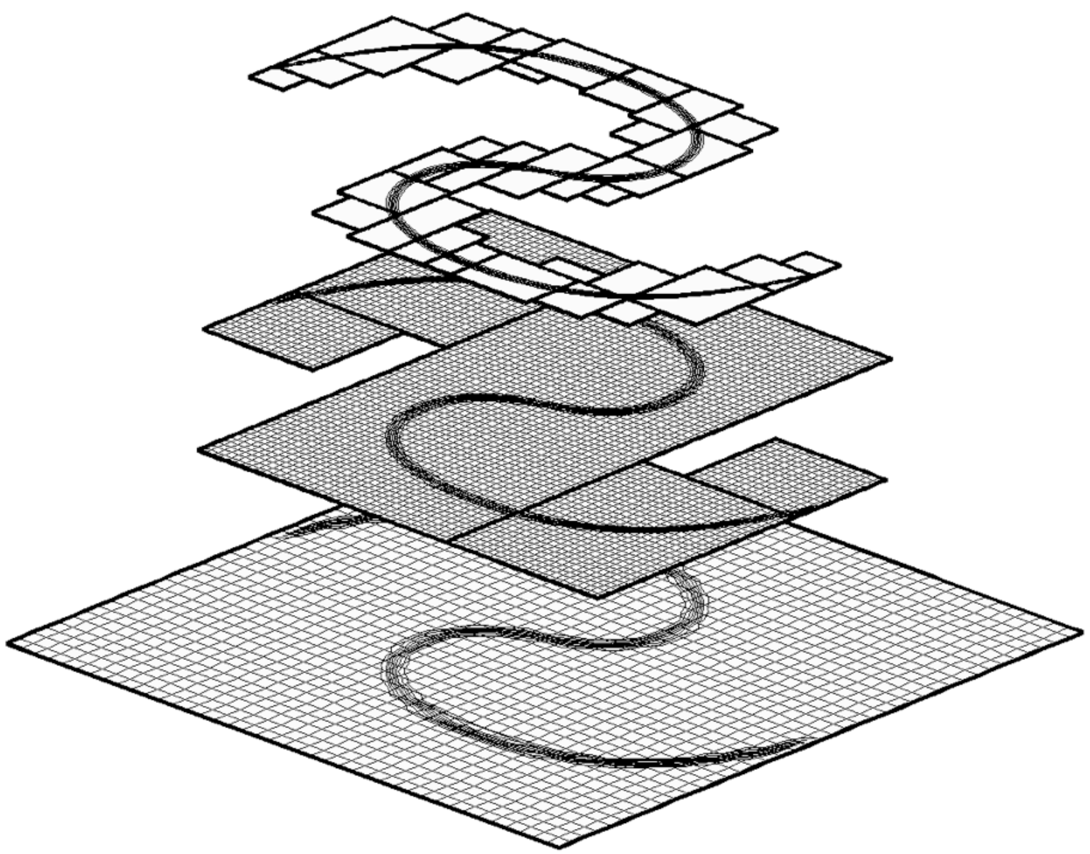
A hybrid approach to AMR

A hybrid approach to AMR

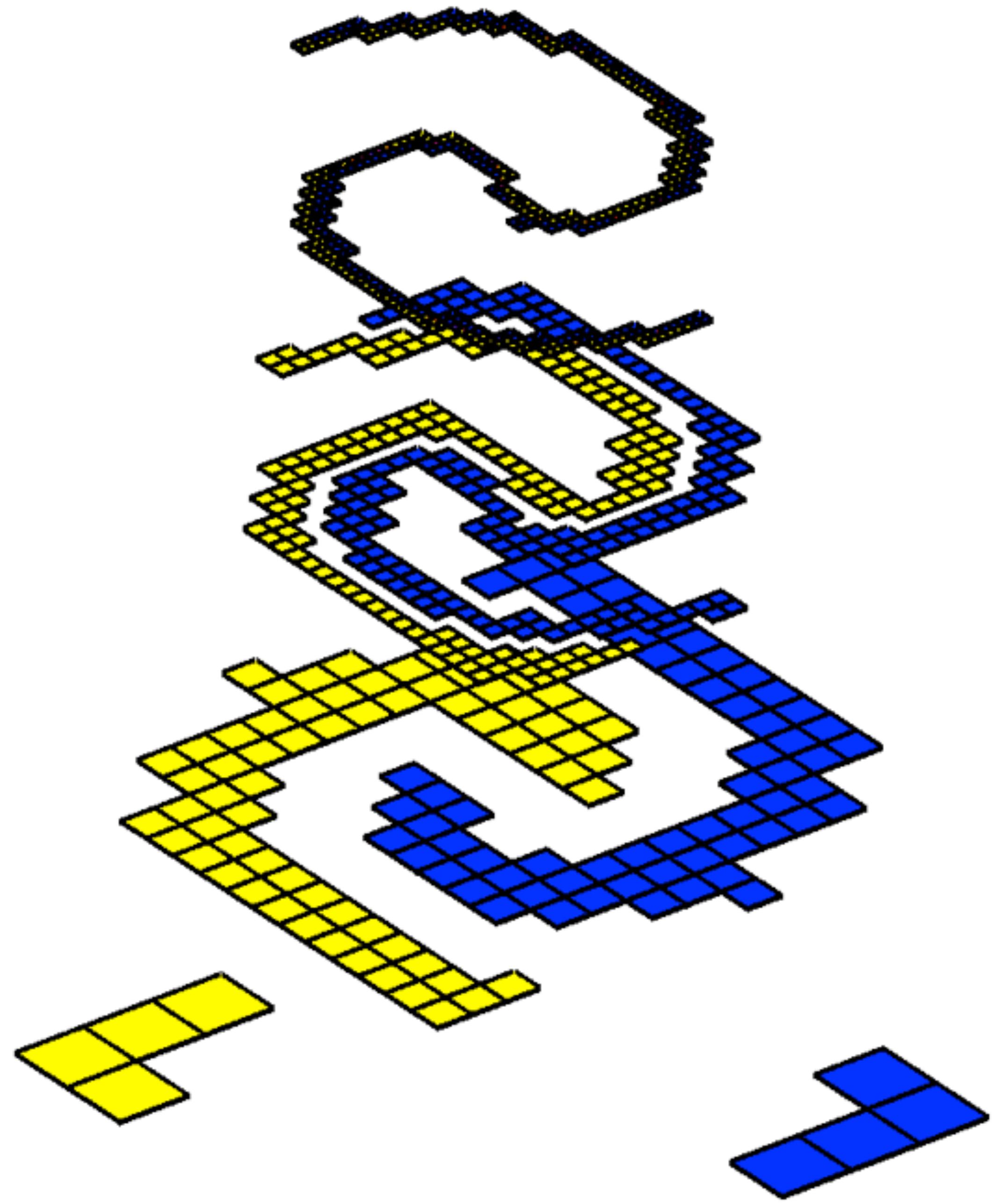


Berger-Oliger approach

A hybrid approach to AMR

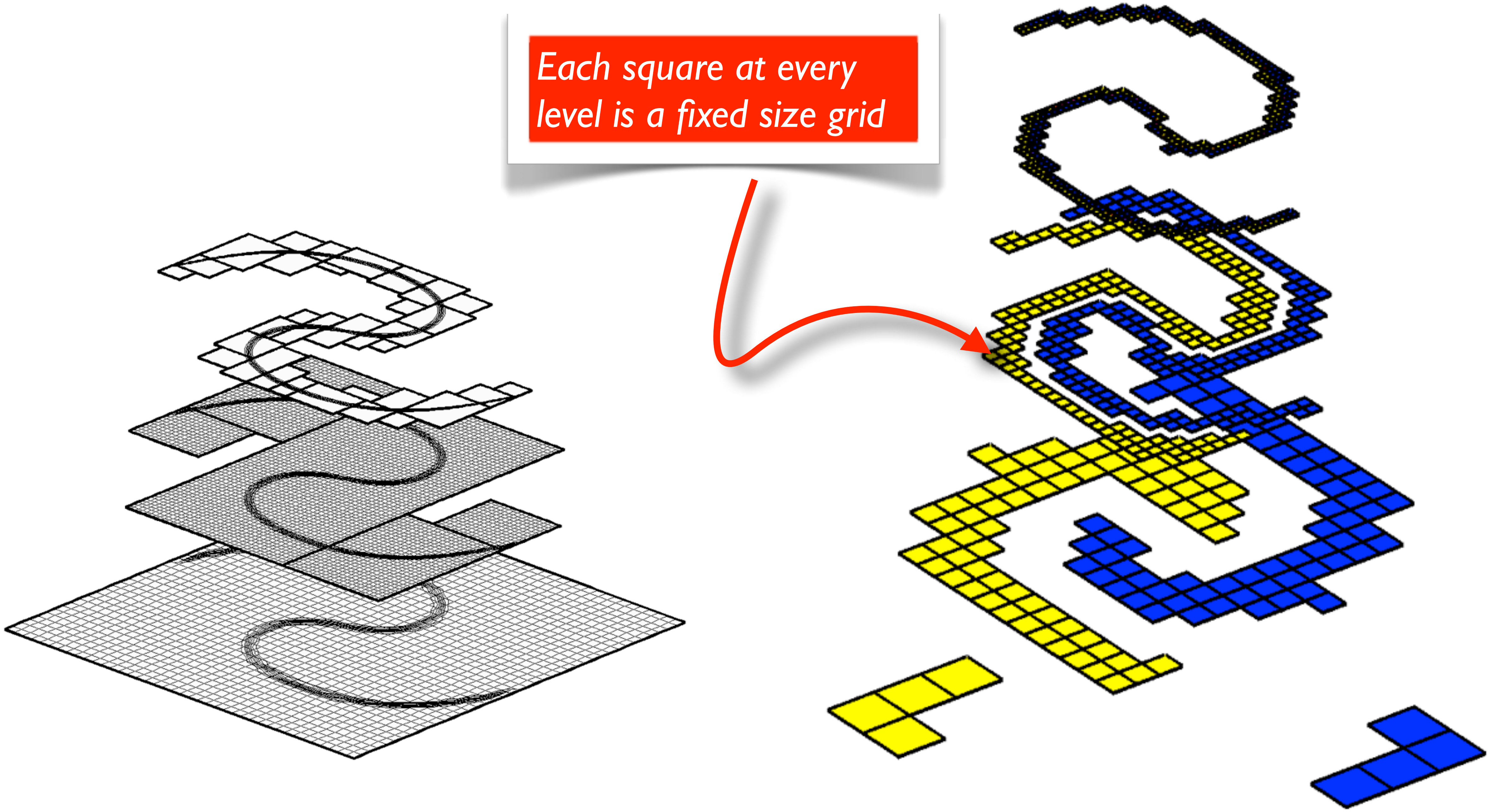


Berger-Oliger approach



ForestClaw

A hybrid approach to AMR



Berger-Oliger approach

ForestClaw

How is this easier for the developer?

How is this easier for the developer?

- Management of "boxes" (leaves) containing grids handled behind the scenes,

How is this easier for the developer?

- Management of "boxes" (leaves) containing grids handled behind the scenes,
- Refinement based on tagging leaves is handled trivially : one grid is refined into four grids.

How is this easier for the developer?

- Management of "boxes" (leaves) containing grids handled behind the scenes,
- Refinement based on tagging leaves is handled trivially : one grid is refined into four grids.
- Interpolation/averaging and ghost cell exchange all done using few prescribed patterns of grid intersections - no need to store indices of locations of fine grid in the coarse grid

How is this easier for the developer?

- Management of "boxes" (leaves) containing grids handled behind the scenes,
- Refinement based on tagging leaves is handled trivially : one grid is refined into four grids.
- Interpolation/averaging and ghost cell exchange all done using few prescribed patterns of grid intersections - no need to store indices of locations of fine grid in the coarse grid
- Parallelization is handled at the tree level, in a separate library,

How is this easier for the developer?

- Management of "boxes" (leaves) containing grids handled behind the scenes,
- Refinement based on tagging leaves is handled trivially : one grid is refined into four grids.
- Interpolation/averaging and ghost cell exchange all done using few prescribed patterns of grid intersections - no need to store indices of locations of fine grid in the coarse grid
- Parallelization is handled at the tree level, in a separate library,
- Multi-block domains are handled automatically

How is this easier for the user?

How is this easier for the user?

- Ghost cell exchanges only occur at the edges/faces of grids, not in the interior of coarse grids,

How is this easier for the user?

- Ghost cell exchanges only occur at the edges/faces of grids, not in the interior of coarse grids,
- Predictable pattern for grid neighbors - one grid at the same level, two at a finer level, or "half" at a coarser level (in 2d),

How is this easier for the user?

- Ghost cell exchanges only occur at the edges/faces of grids, not in the interior of coarse grids,
- Predictable pattern for grid neighbors - one grid at the same level, two at a finer level, or “half” at a coarser level (in 2d),
- Composite view of the grid hierarchy provides a cleaner mental map of what the solution looks like (no overlapping grids)

How is this easier for the user?

- Ghost cell exchanges only occur at the edges/faces of grids, not in the interior of coarse grids,
- Predictable pattern for grid neighbors - one grid at the same level, two at a finer level, or “half” at a coarser level (in 2d),
- Composite view of the grid hierarchy provides a cleaner mental map of what the solution looks like (no overlapping grids)
- Generic multi-block interface allows user to construct complicated domains

How is this easier for the user?

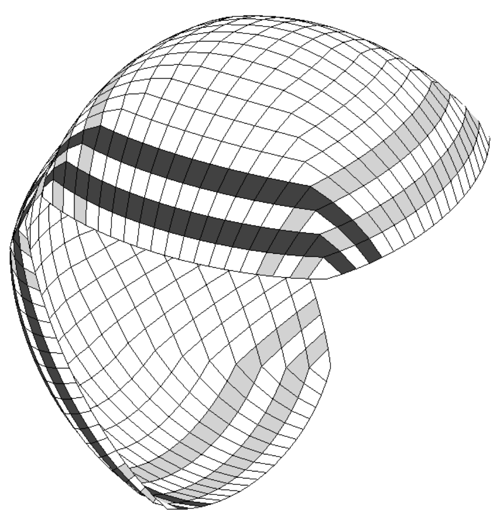
- Ghost cell exchanges only occur at the edges/faces of grids, not in the interior of coarse grids,
- Predictable pattern for grid neighbors - one grid at the same level, two at a finer level, or "half" at a coarser level (in 2d),
- Composite view of the grid hierarchy provides a cleaner mental map of what the solution looks like (no overlapping grids)
- Generic multi-block interface allows user to construct complicated domains
- Has the aesthetic appeal of a quad/octree refinement with the superior performance of the Berger-Oliger approach to AMR.

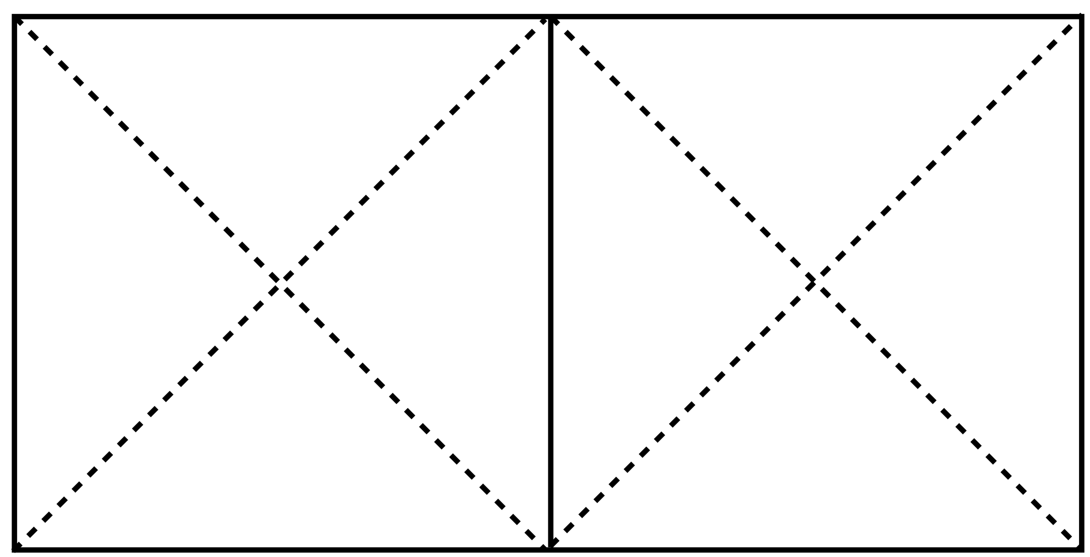
Other AMR approaches using quad/octrees

- "Building Cubes Method" (Sasaki, Akahito, Yamazaki, ...)
- Parallel adaptive methods for weather prediction C. Jablonowski, Oehmke, Stout and others
- NIRVANA (U. Ziegler)
- Racoon II (J. Dreher)
- PARAMESH (NASA,/Drexel, MacNeice, Olson)
- Block-structured AMR codes (Chombo, Boxlib, AMRClaw, SAMRAI, AMROC, ...) could probably be run with fixed size grids and prescribed refinement regions

A two-patch sphere grid?

- Our sphere grid is like the cubed-sphere grid, but with two patches
- We will refer to the grid resolution by the number of grid cells on a patch edge, which is approximately 90 degrees.

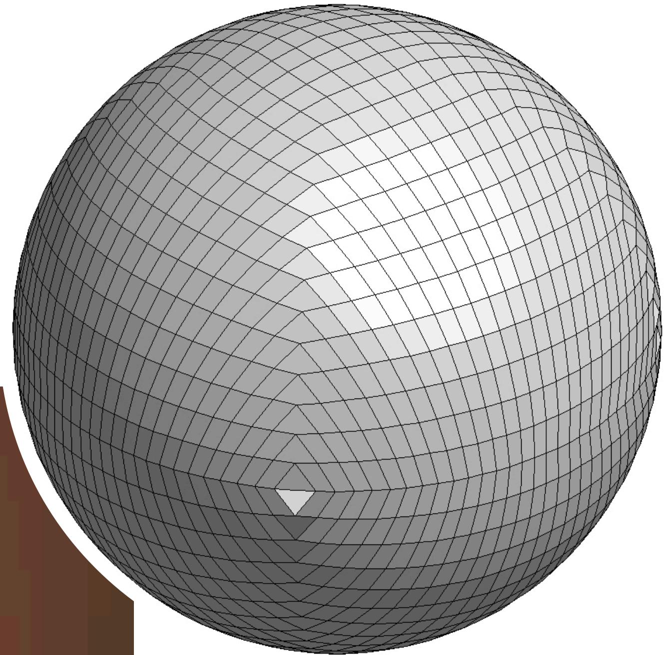




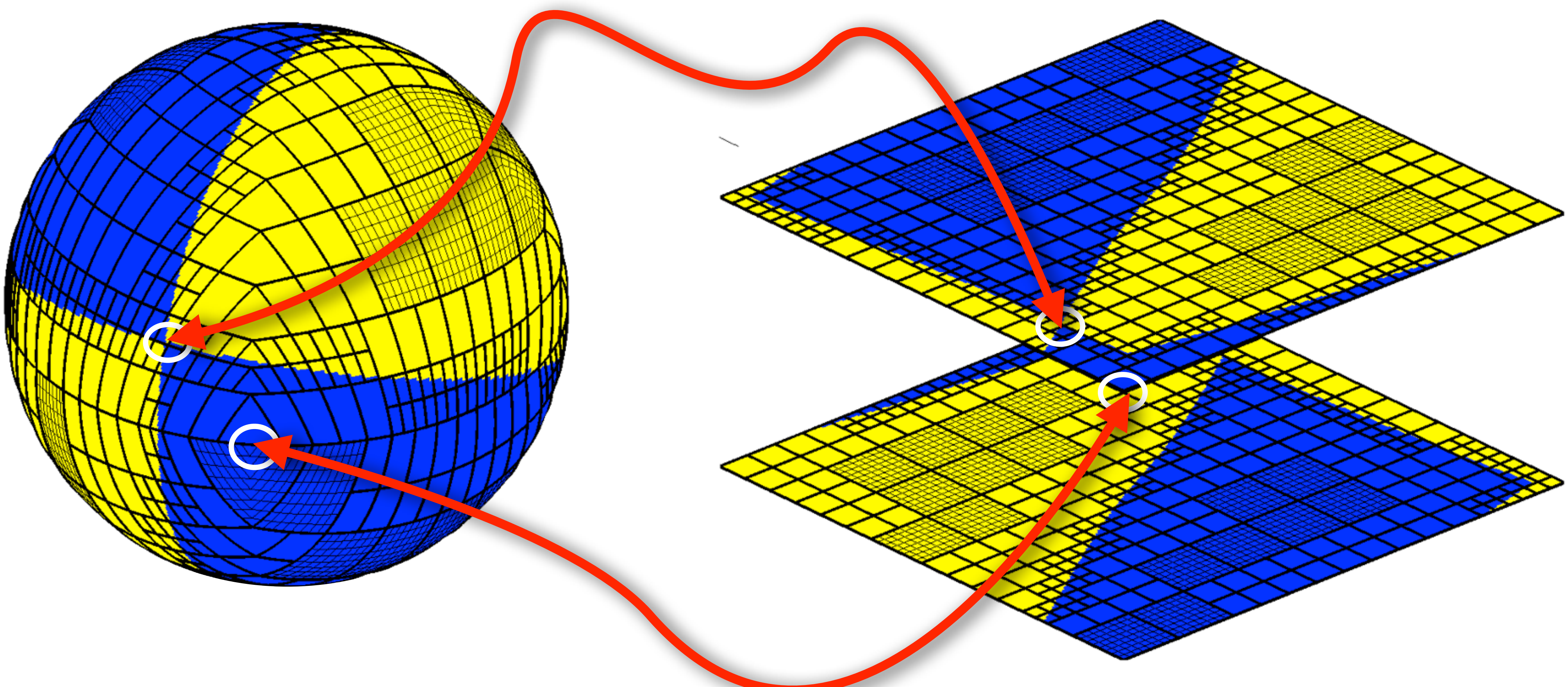
90 degrees =
N mesh cells

Dashed and solid lines are discontinuities in the mapping

A Pillow grid?



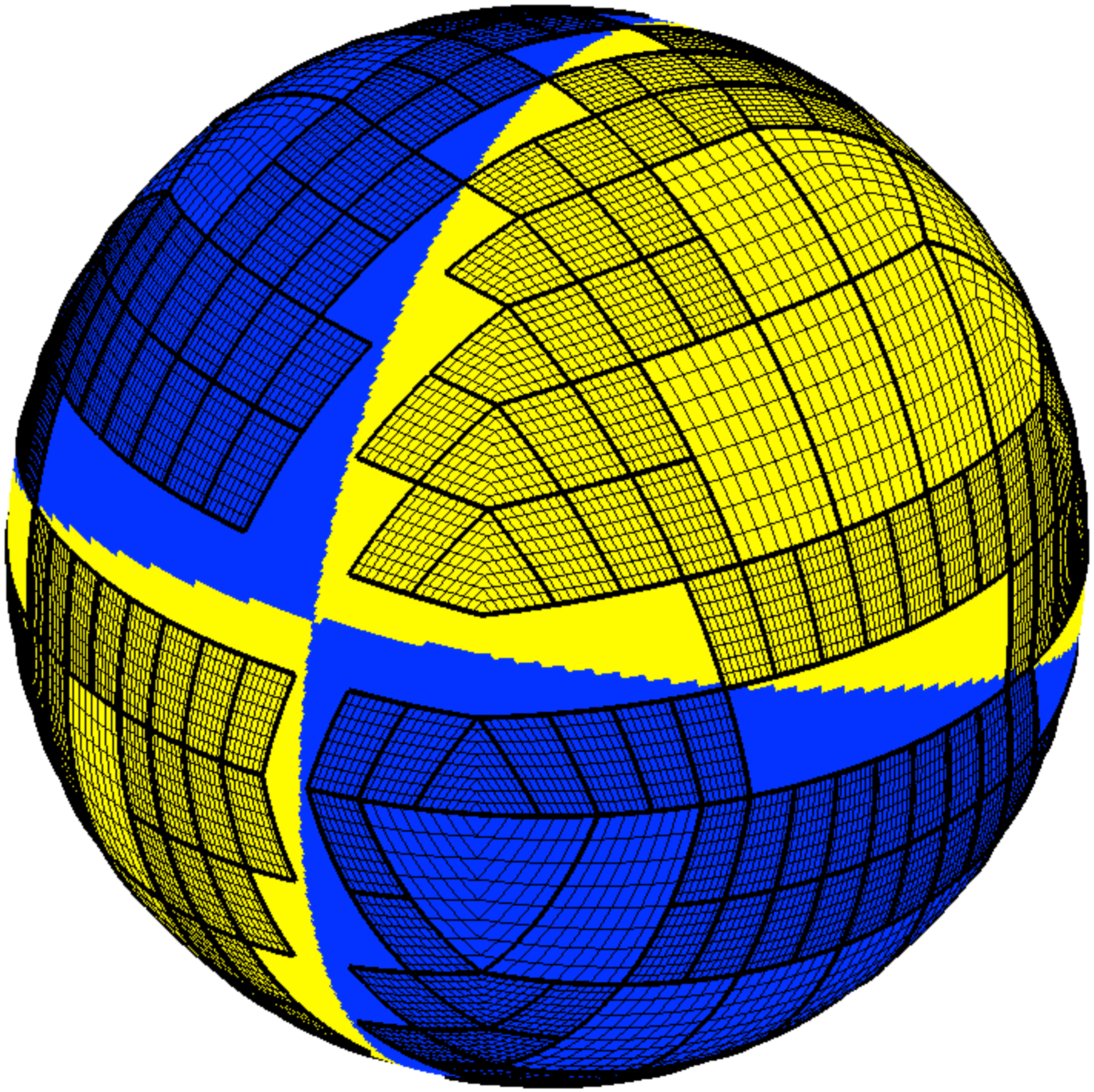
Scalar transport on the sphere



Scalar advection using finite volume wave propagation algorithms (ClawPACK, R. J. LeVeque)

Scalar advection

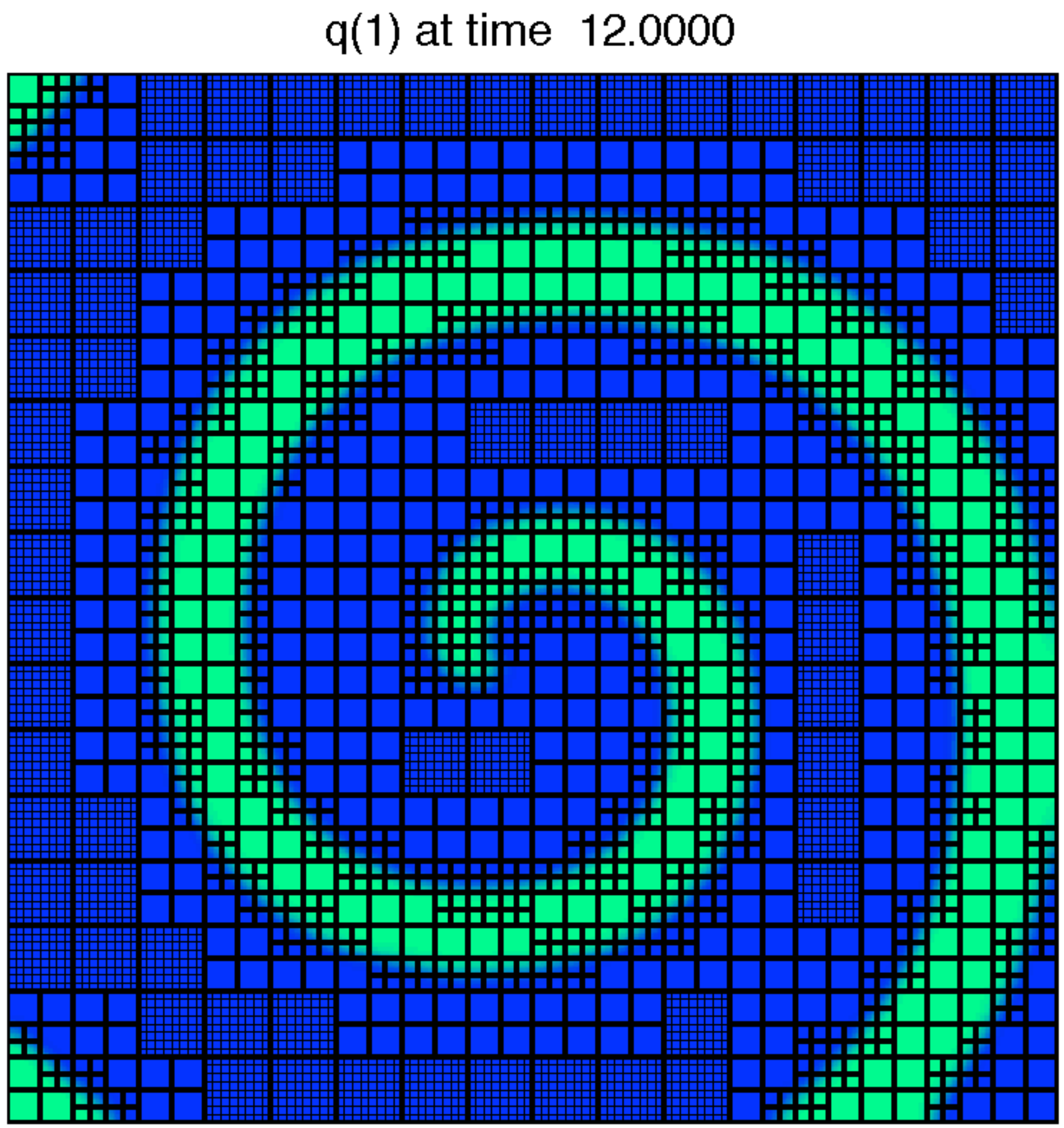
Scalar advection



Spiral waves (Barkely model)

Reaction-diffusion using an explicit Runge-Kutta Chebychev (RKC) time stepping

Spiral waves (Barkely model)



Reaction-diffusion using an explicit Runge-Kutta Chebychev (RKC) time stepping

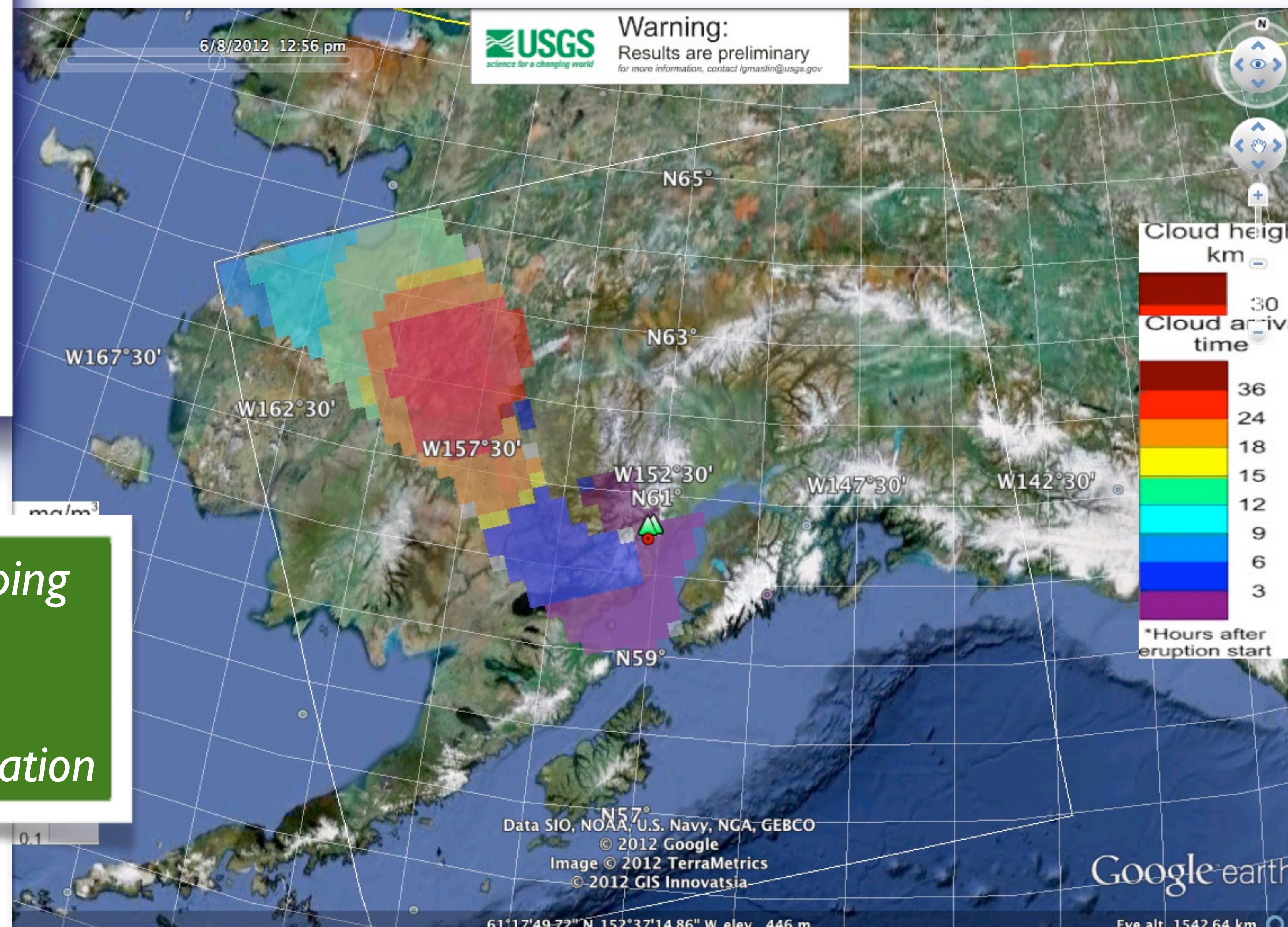
Near future challenges

- Include anisotropic refinement for atmospheric applications by putting 3d grids into 2d quadtrees
- Parallel ghost cell exchanges (p4est is already fully parallelized; parallel exchanges between ghost cells need to be worked out)
- General handling of grid orientations in multi-block setting,
- New topologies, i.e. the cubed-sphere (already available in p4est)

see *http://www.forestclaw.org*

Ash cloud modeling

Ash3d



- *Split horizontal, vertical time stepping*
- *Fully conservative,*
- *Eulerian, finite volume*
- *Algorithms based on wave propagation*

Ash3d :A finite-volume, conservative numerical model for ash transport and tephra deposition, Schwaiger, Denlinger, Mastin, JGR (2012)