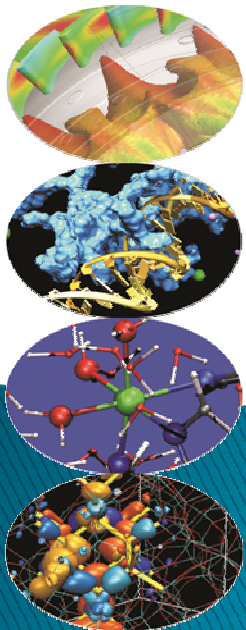
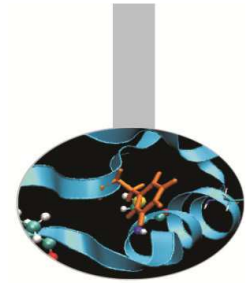


An introduction to Adaptive Mesh Refinement (AMR): Numerical Methods and Tools

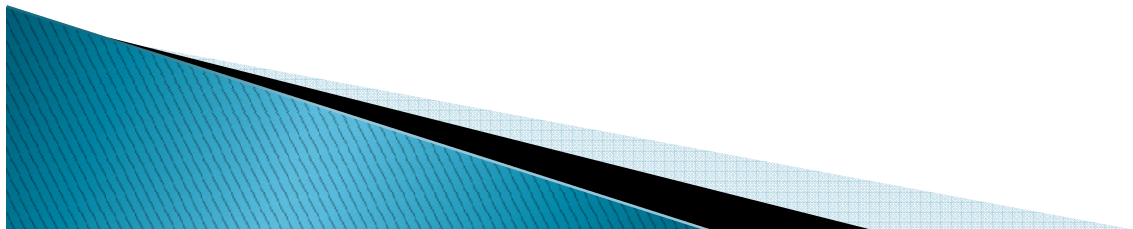


HPC Numerical Libraries
11–13 March 2015
CINECA – Casalecchio di Reno (BO)

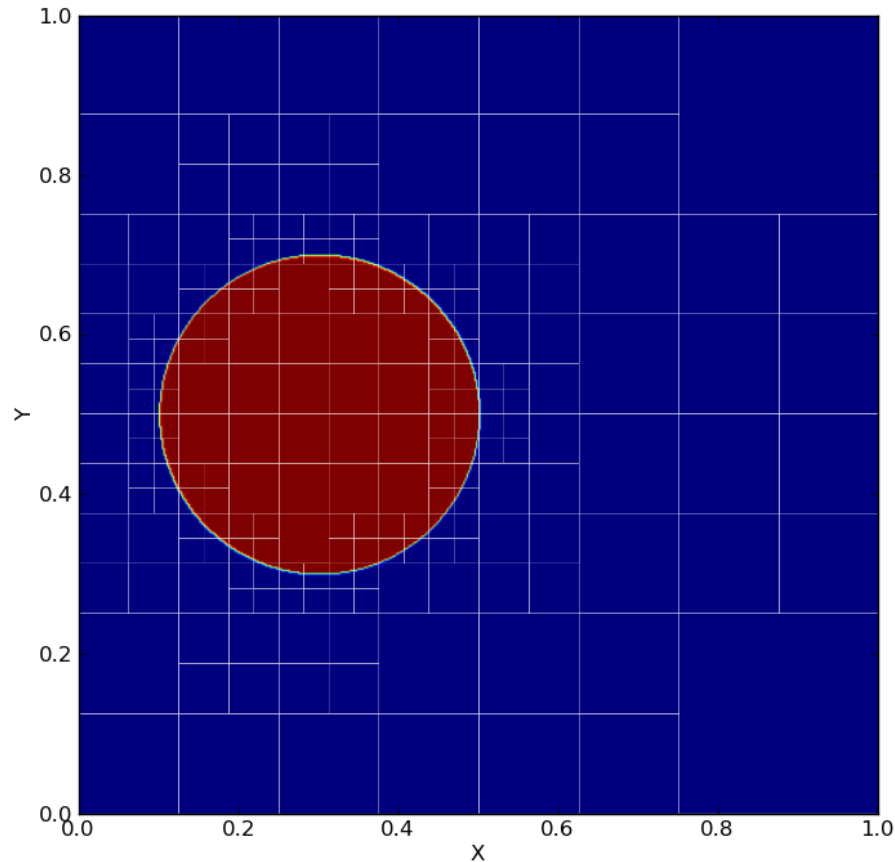
Massimiliano Guarrasi - m.guarrasi@cineca.it
Super Computing Applications and Innovation Department

AMR - Introduction

- Solving Partial Differential Equations (PDEs)
 - PDEs solved using discrete domain
 - Algebraic equations estimate values of unknowns at the mesh points
 - Resolution/Spacing of mesh points determines error
 - Initial Solution and Boundary condition are needed
- Goal of grid adaptivity:
 - tracking features much smaller than overall scale of the problem providing adequate higher spatial and temporal resolution where needed.



AMR - Introduction



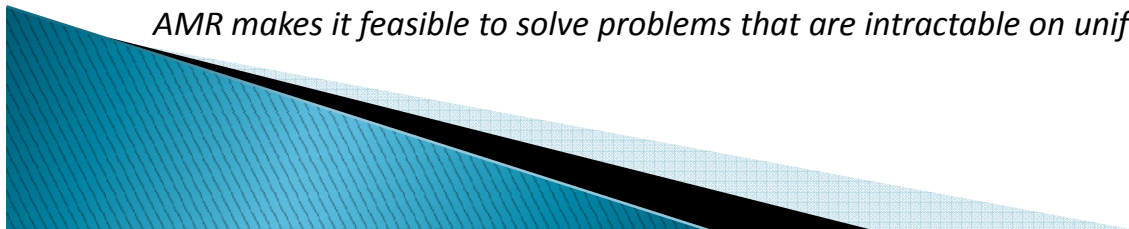
Uniform meshes

- High resolution required for handling difficult regions (discontinuities, steep gradients, shocks, etc.)
- Computationally extremely costly

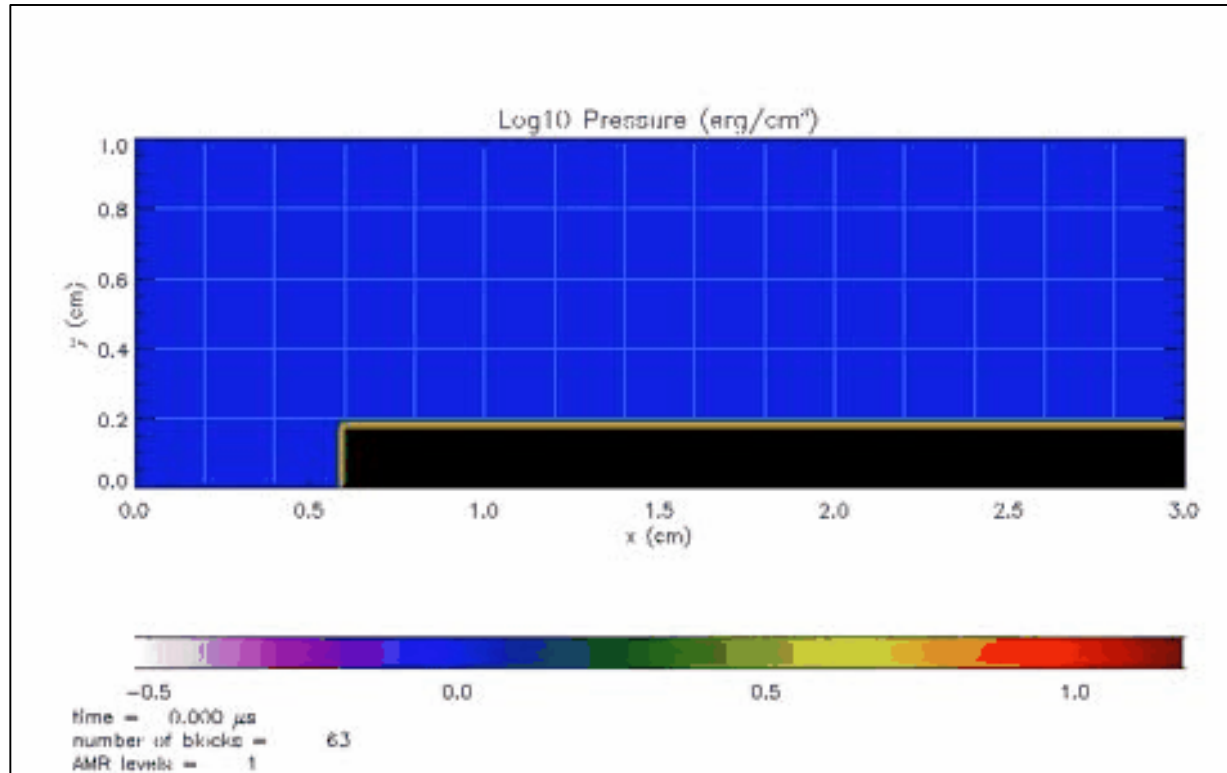
Adaptive Mesh Refinement

- Start with a coarse grid
- Identify regions that need finer resolution
- Superimpose finer sub-grids only on those regions
- Increased computational savings over a static grid approach.
- Increased storage savings over a static grid approach.
- Complete control of grid resolution, compared to the fixed resolution of a static grid approach.

AMR makes it feasible to solve problems that are intractable on uniform grid



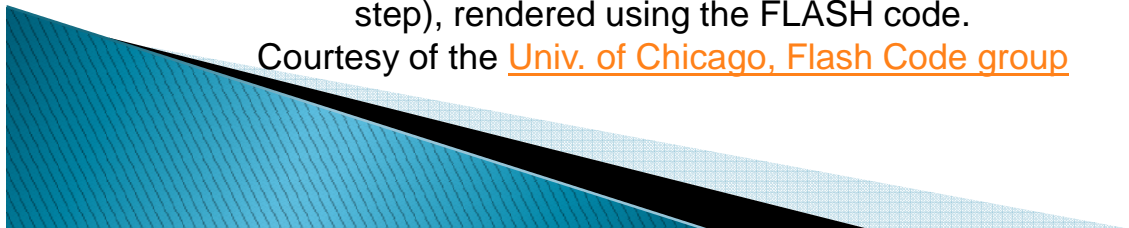
AMR - Applications



- CFD
- Astrophysics
- Climate Modeling
- Turbulence
- Mantle Convection
- Modeling
- Combustion
- Biophysics
- and many more

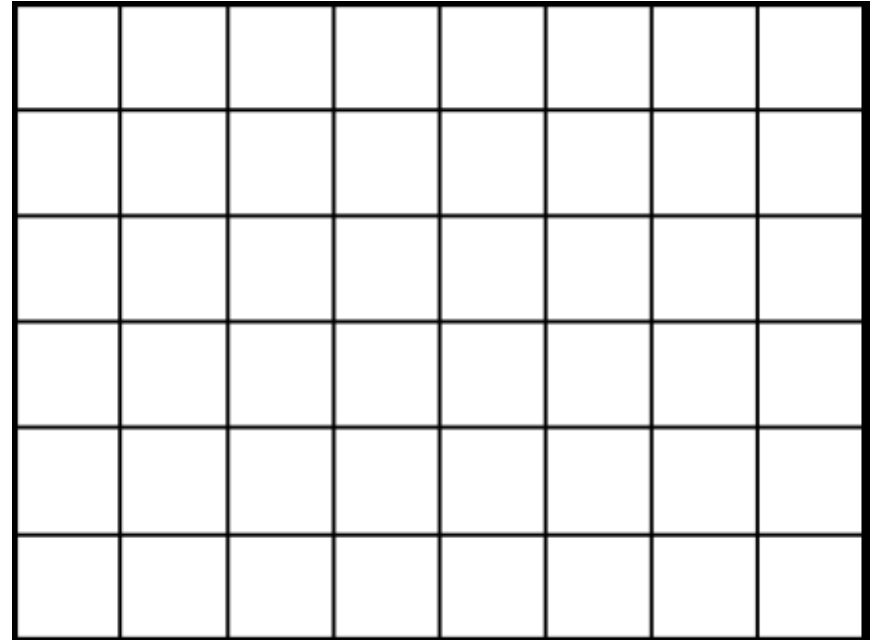
Demo of a Shock wave passing over a step function (wind tunnel with a step), rendered using the FLASH code.

Courtesy of the [Univ. of Chicago, Flash Code group](#)

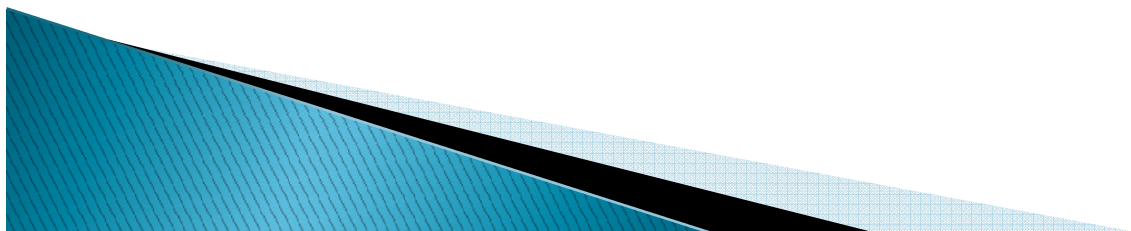


AMR Techniques

 mesh distortion

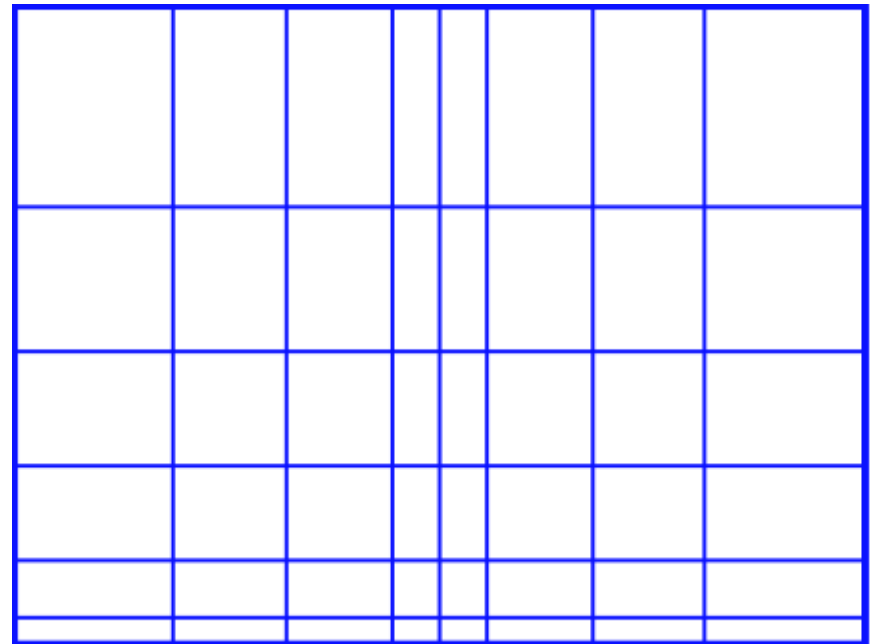


Courtesy of Dr. Andrea Mignone, University of Turin

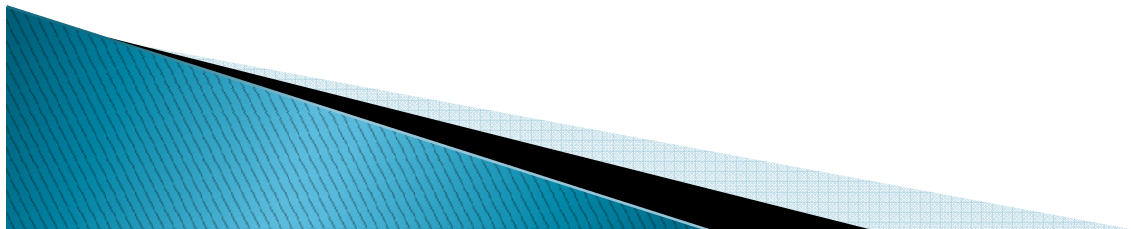


AMR Techniques

 mesh distortion

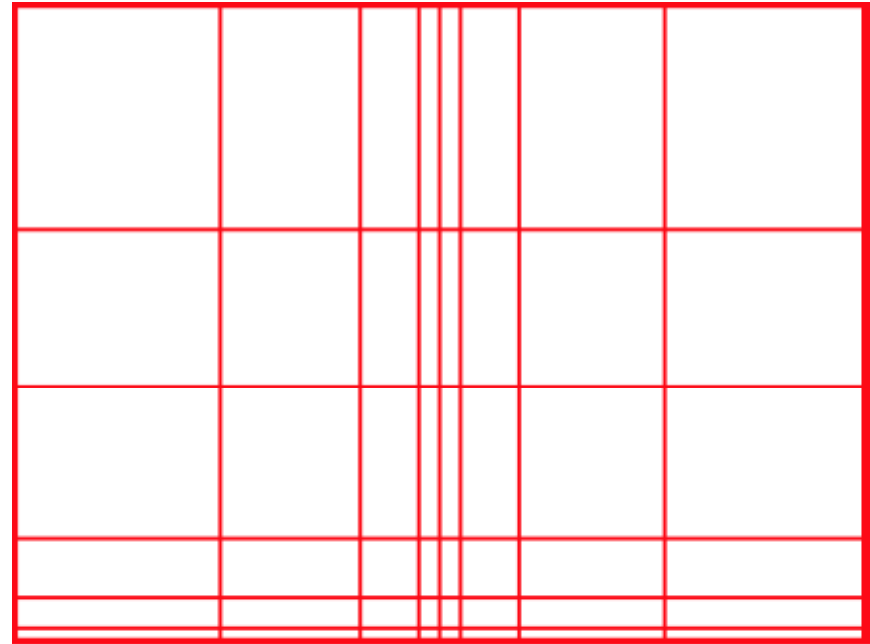


Courtesy of Dr. Andrea Mignone, University of Turin

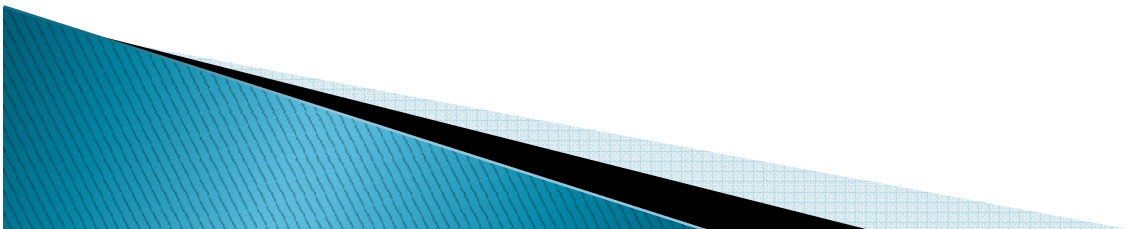


AMR Techniques



 mesh distortion

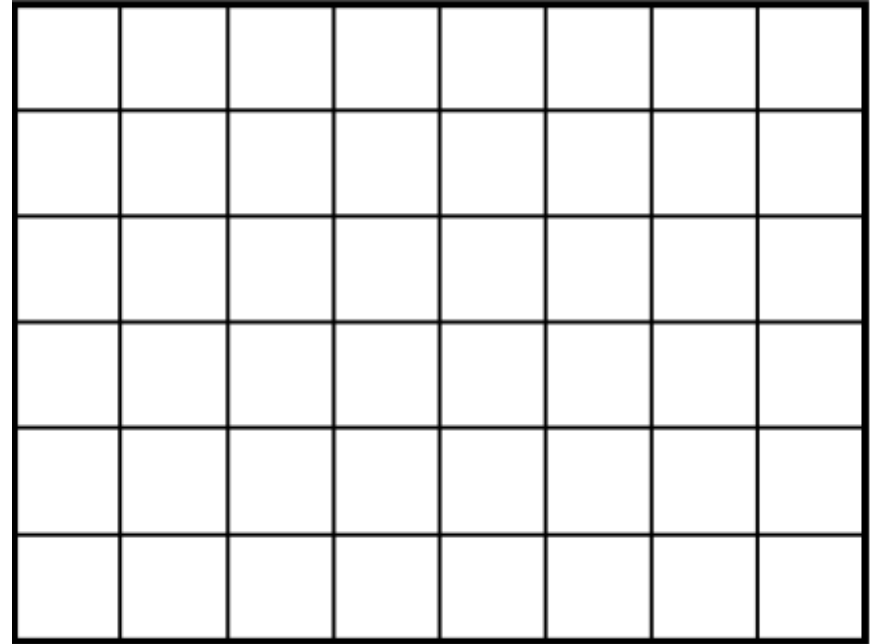


Courtesy of Dr. Andrea Mignone, University of Turin

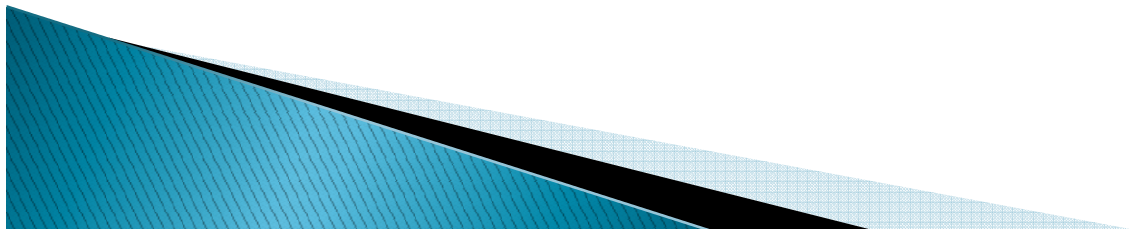


AMR Techniques



-  mesh distortion
-  point-wise structured refinement (tree-based)

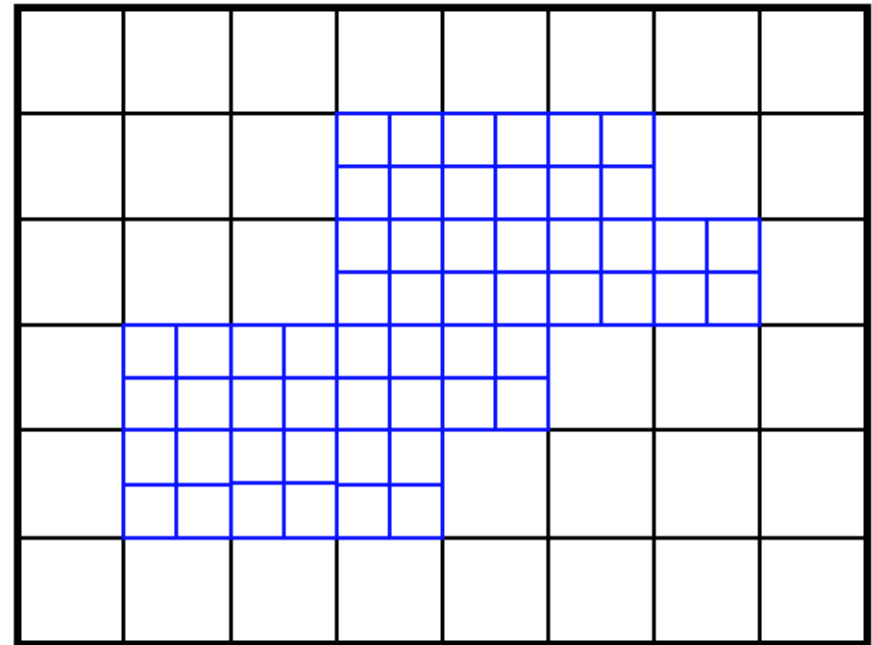


Courtesy of Dr. Andrea Mignone, University of Turin

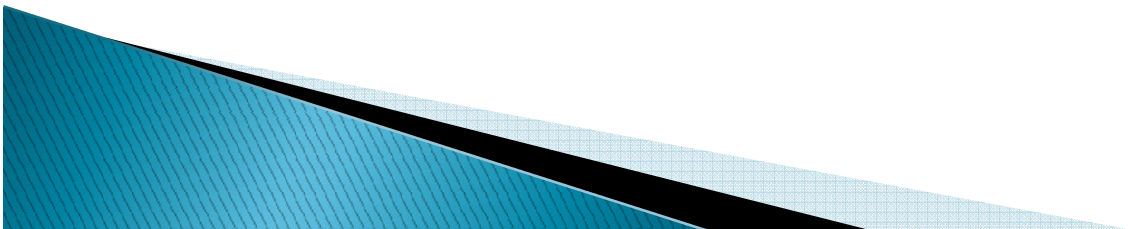


AMR Techniques



-  mesh distortion
-  point-wise structured refinement (tree-based)

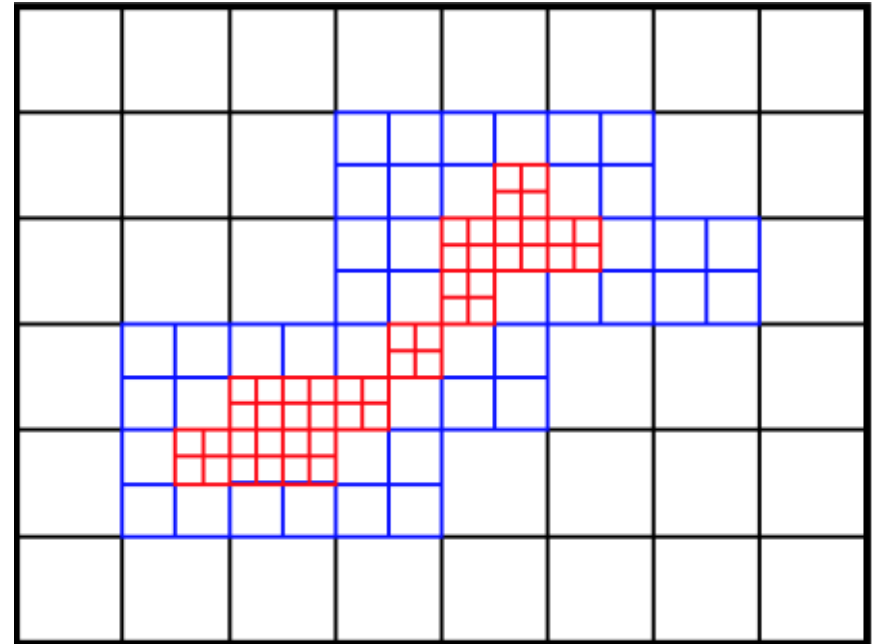


Courtesy of Dr. Andrea Mignone, University of Turin

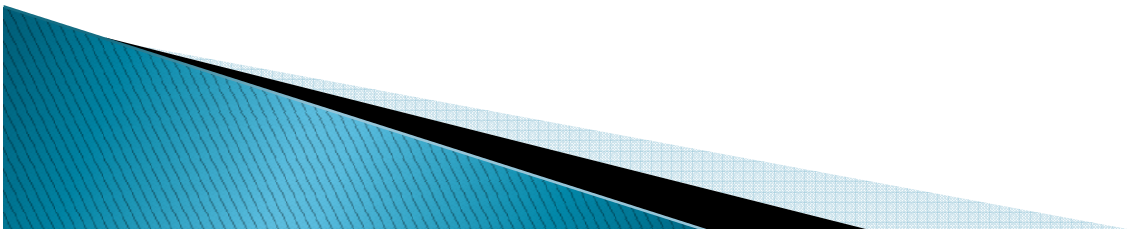


AMR Techniques

-  mesh distortion
-  point-wise structured (tree-based) refinement

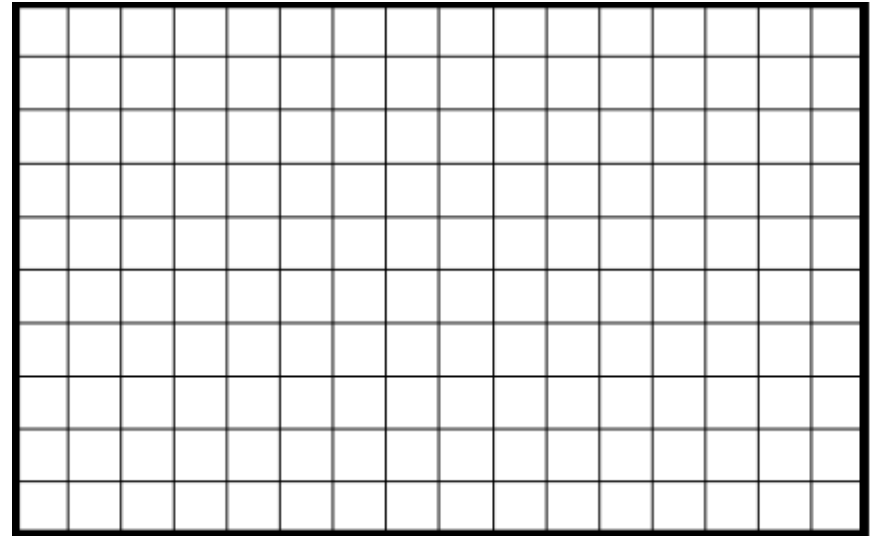


Courtesy of Dr. Andrea Mignone, University of Turin

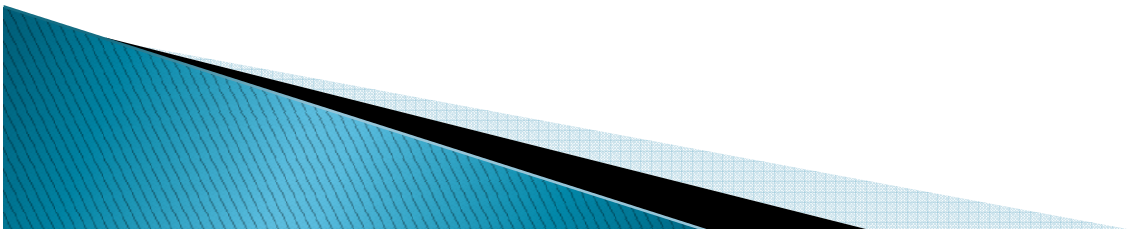


AMR Techniques

- ☐ mesh distortion
- ☐ point-wise structured (tree-based) refinement
- ☐ block structured

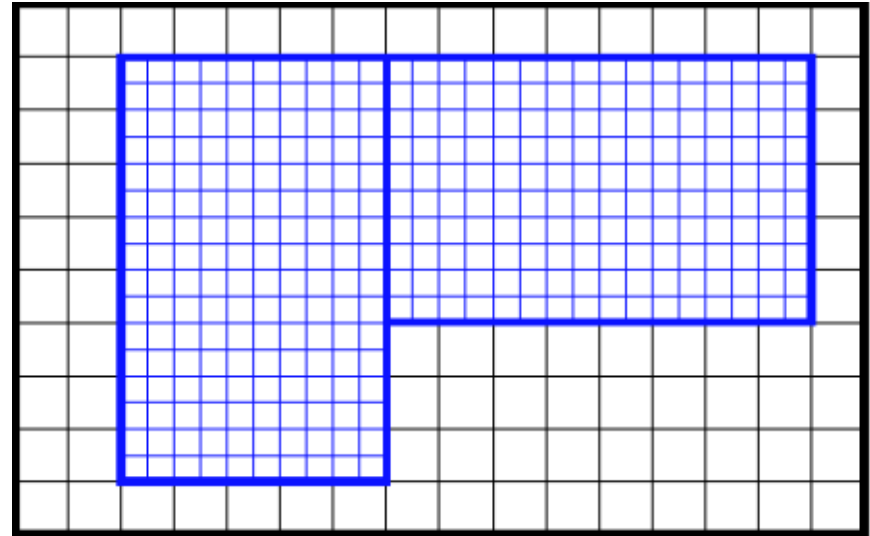


Courtesy of Dr. Andrea Mignone, University of Turin

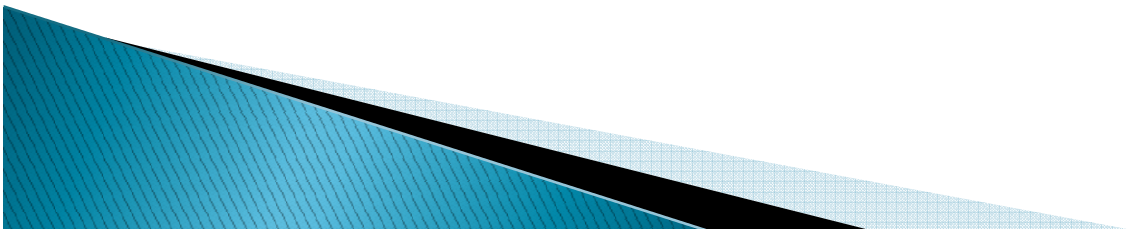


AMR Techniques

- ☐ mesh distortion
- ☐ point-wise structured (tree-based) refinement
- ☒ block structured

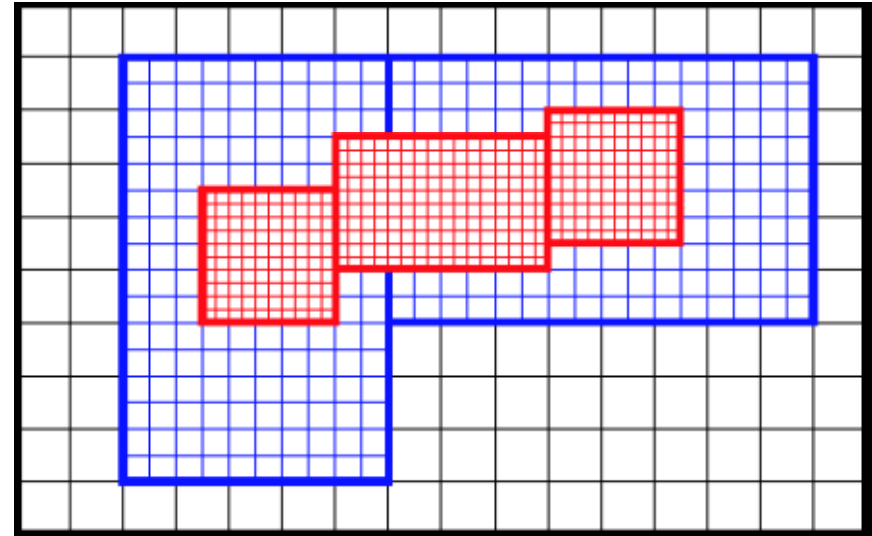


Courtesy of Dr. Andrea Mignone, University of Turin

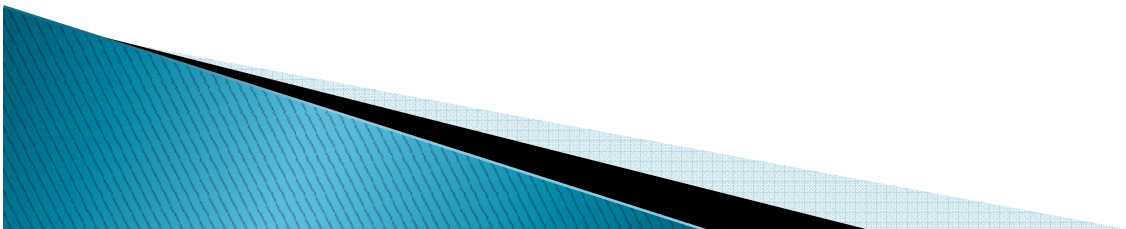


AMR Techniques

- ☐ mesh distortion
- ☐ point-wise structured (tree-based) refinement
- ☐ block structured:

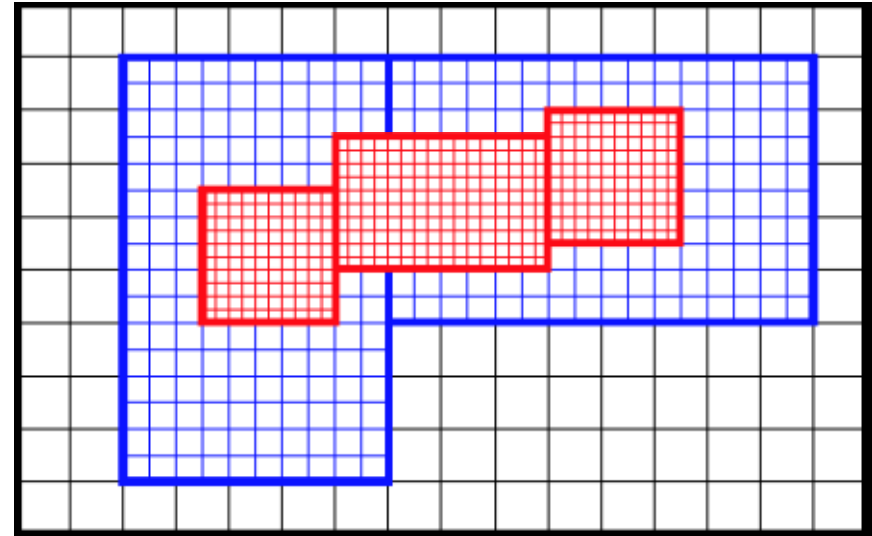


Courtesy of Dr. Andrea Mignone, University of Turin



AMR Techniques

- ☐ mesh distortion
- ☐ point-wise structured (tree-based) refinement
- ☒ block structured:

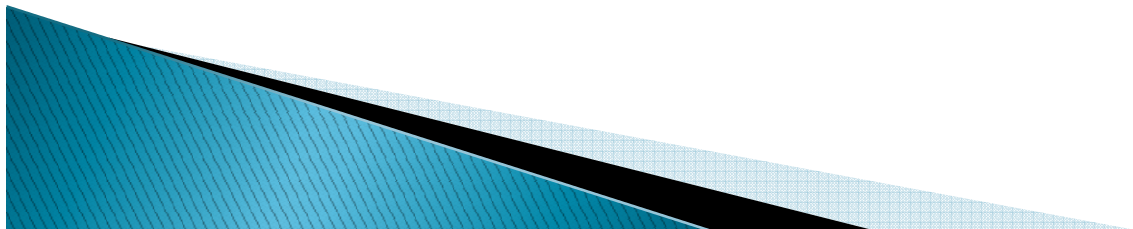


Courtesy of Dr. Andrea Mignone, University of Turin

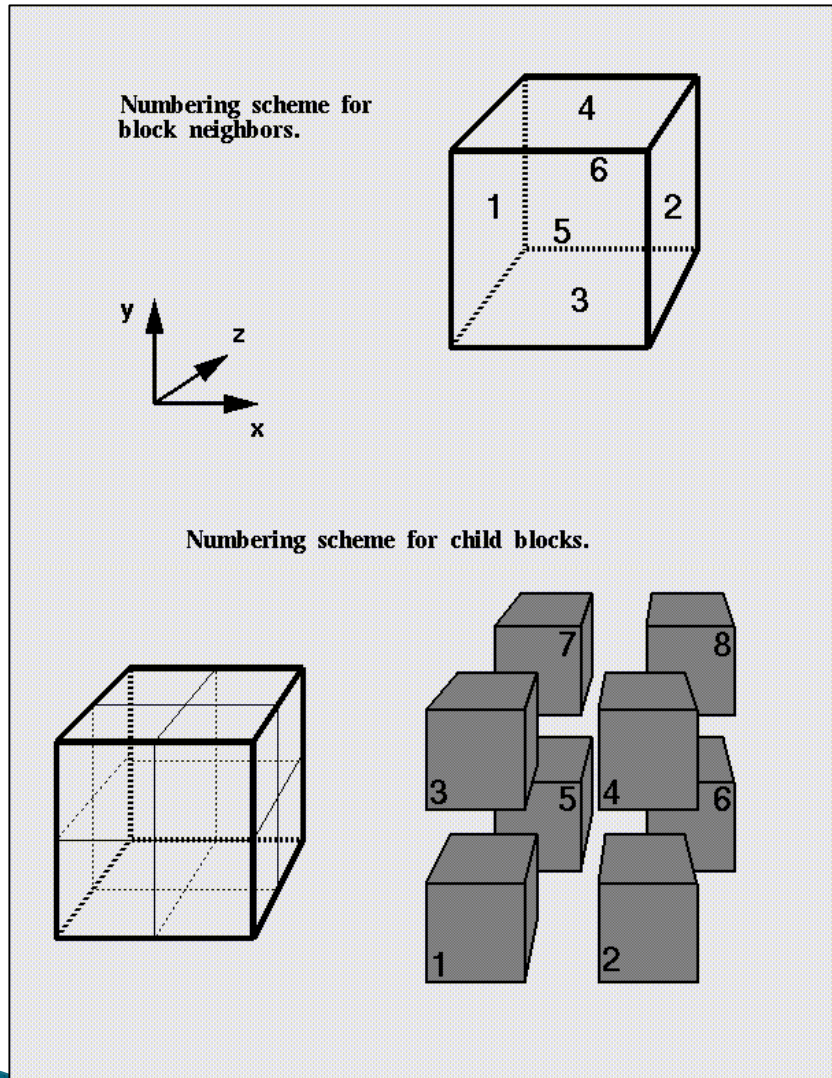
- data blocks are created so that the same stencil can be used for all points and no special treatment is required.
- High level objects that encapsulate the functionality for AMR and its parallelization are independent of the details of the physics algorithms and the problem being solved.
- Simplifies the process of adding/replacing physics modules as long as they adhere to the interface requirements.

Existing Frameworks

- PARAMESH – <http://www.physics.drexel.edu/~olson/paramesh>
- SAMRAI – <https://computation.llnl.gov/casc/SAMRAI/>
- p4est – <http://www.p4est.org/>
- Chombo – <https://commons.lbl.gov/display/chombo/Chombo>
- and many more

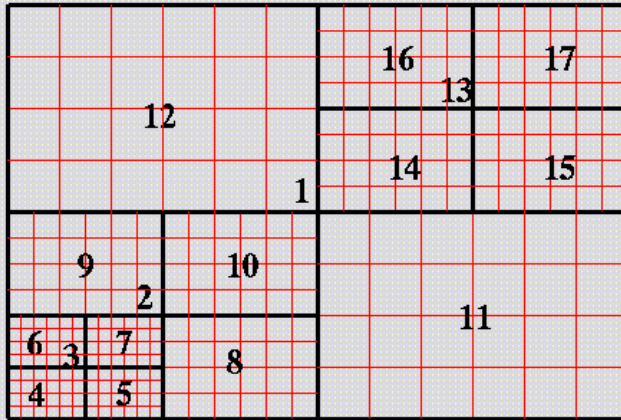


Block Numbering

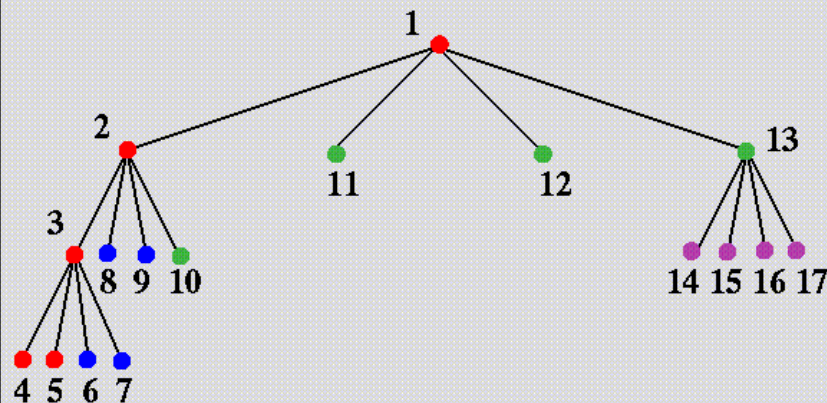


- All the grid blocks are related to one another as the nodes of a tree.
- The starting block is called root block, and the blocks with an higher resolution are called leaf blocks.
- When a leaf block is designated for refinement, it spawns 2 child blocks in 1D, 4 child blocks in 2D or 8 child blocks in 3D, and the original block is called mother (or parent) block.
- These child blocks cover the same physical line, area or volume as their parent but with twice the spatial resolution.
- Usually it is helpful to use a particular numbering algorithm (see next slides).

Typical grid hierarchy



- Each block has a fixed number of grid points
- Each block can be divided into 2^{ndim} sub-blocks
- Blocks are distributed between processes minimizing communications (see next slides)

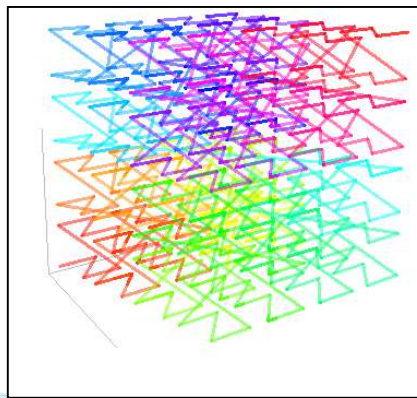
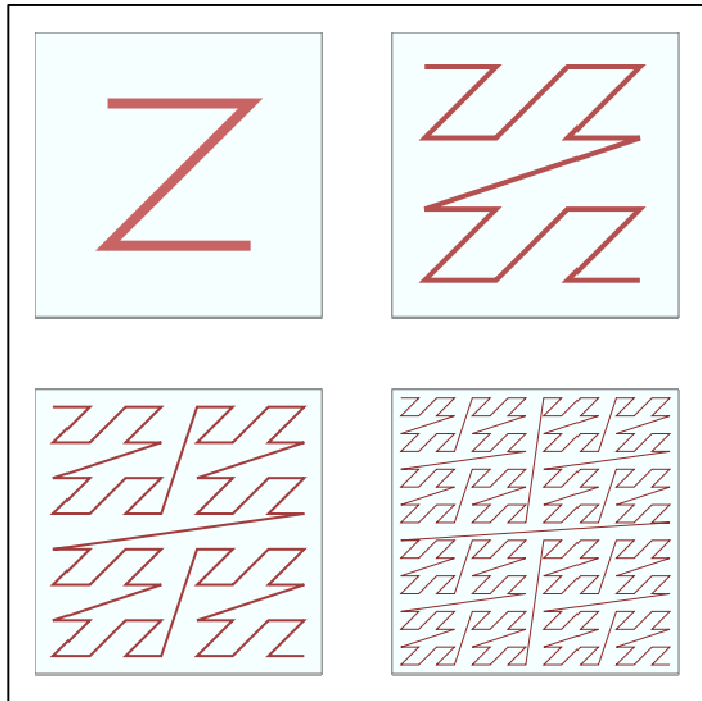


An Example:

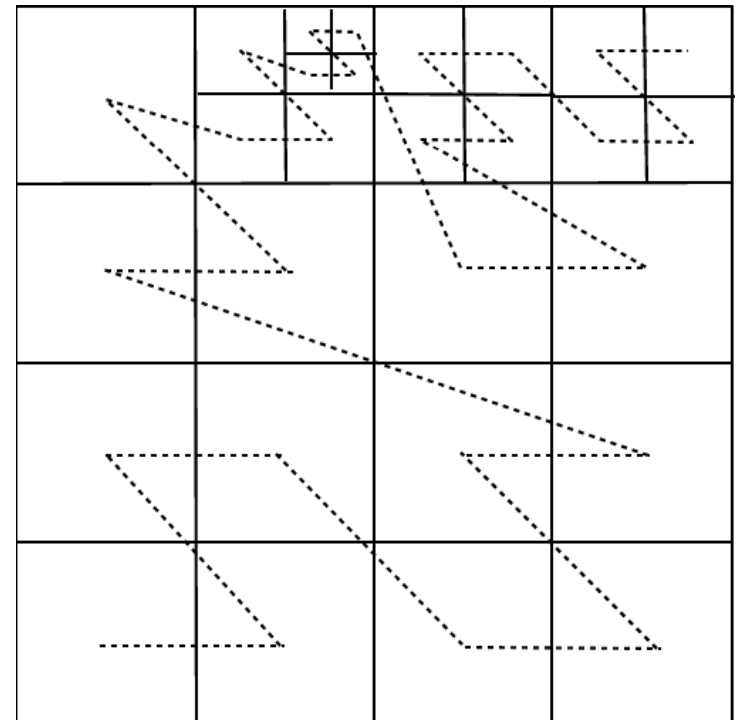
- 6 x 4 grid is created on each block
- The numbers assigned to each block designate the blocks location in the quad-tree
- The numbers assigned to each block designate the blocks location in the quad-tree

From [Paramesh User Guide](#)

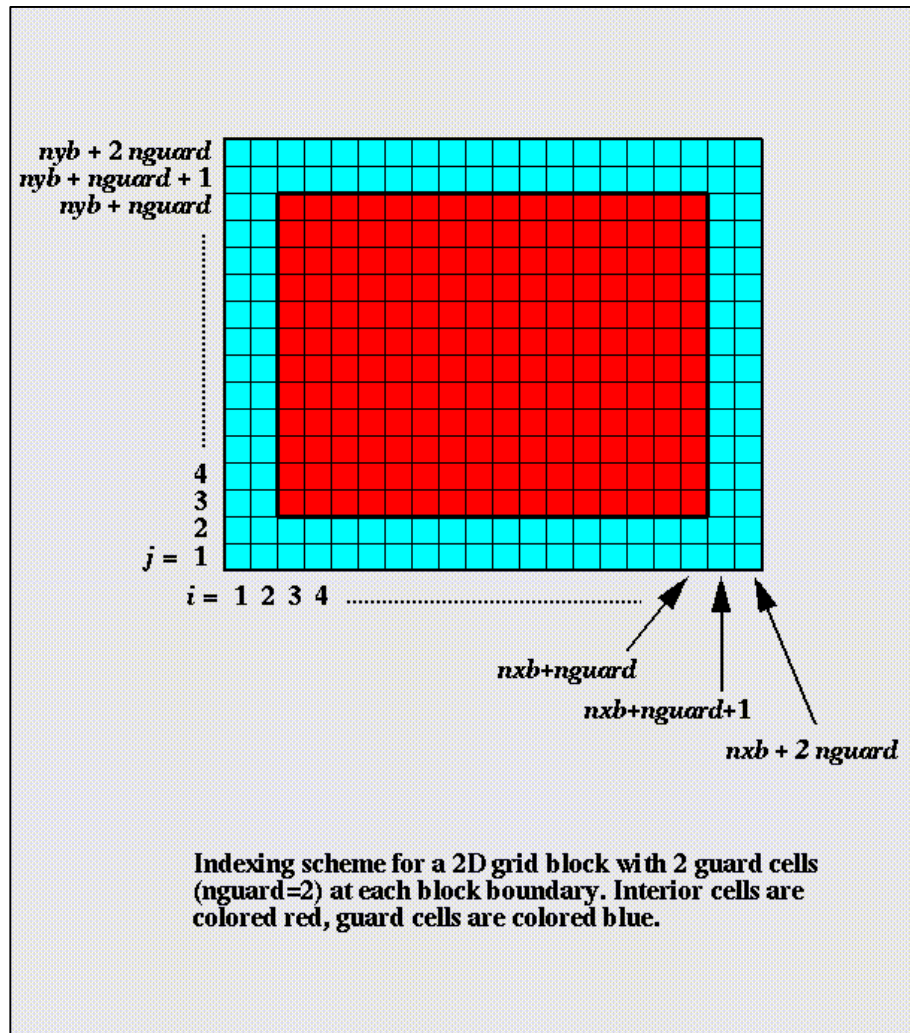
Block ordering



- Usually, the most used block ordering algorithm is Morton (or Z) ordering.
- It is particularly useful in order to:
 - Optimize the usage of cache memory;
 - Optimize ghost cells communications between process (see next slide);



Block Structure



Usually, each block is composed by:

- standard cells
- ghost cells

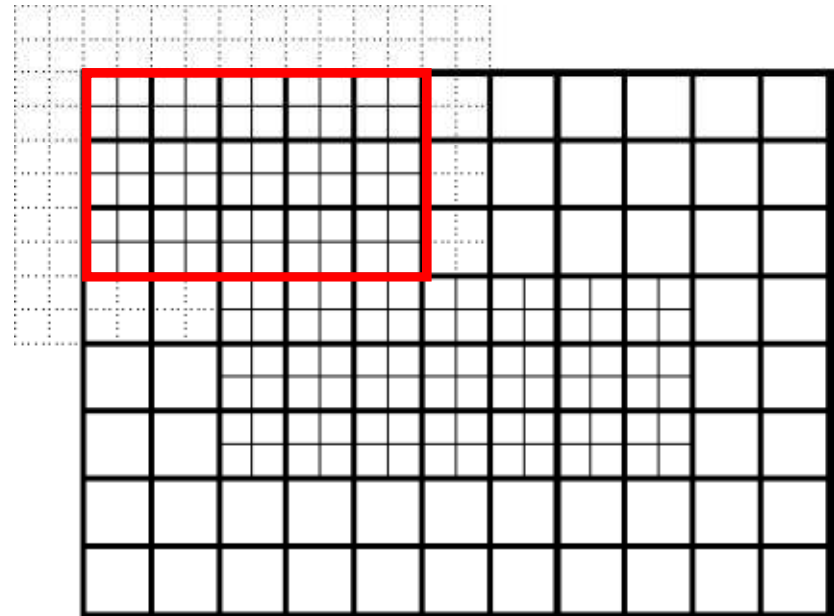
In Fortran, the indexes starts with 1 and ends with $N_{(X \text{ or } Y \text{ or } Z)} + 2 * (\text{number of ghost cells})$

In C, the indexes starts with 0 and ends $N_{(X \text{ or } Y \text{ or } Z)} + 2 * (\text{number of ghost cells}) - 1$

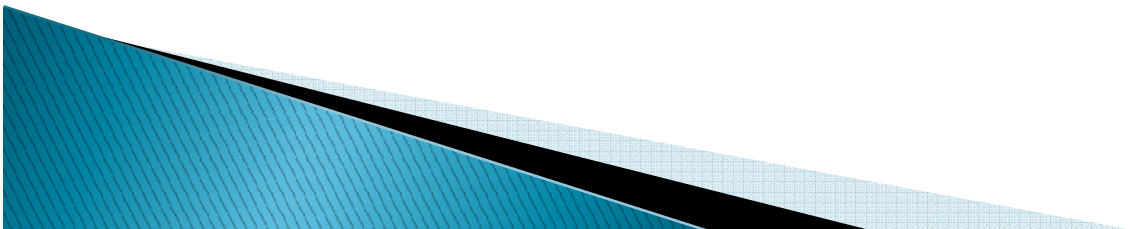
From [Paramesh User Guide](#)

Passing Ghost Cells

- ▶ ghost zones values need to be filled before integration;

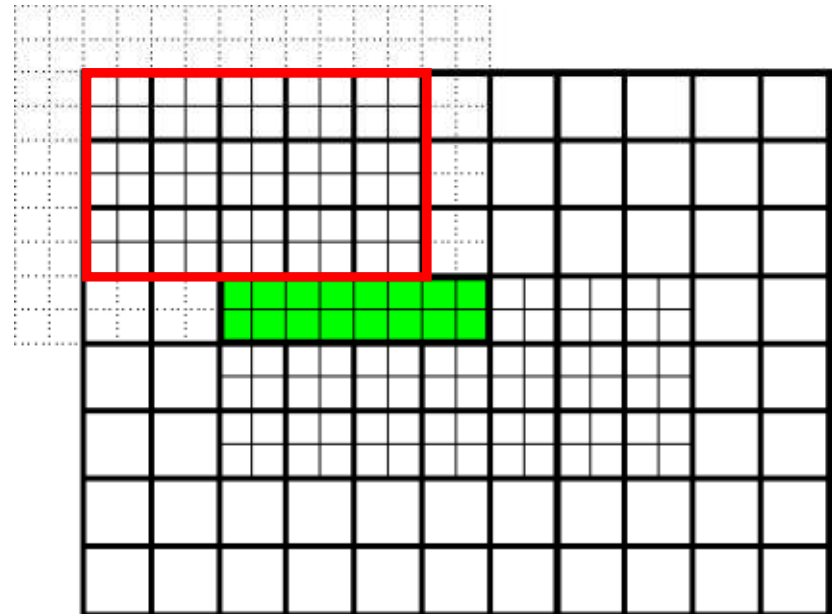


Courtesy of Dr. Andrea Mignone, University of Turin

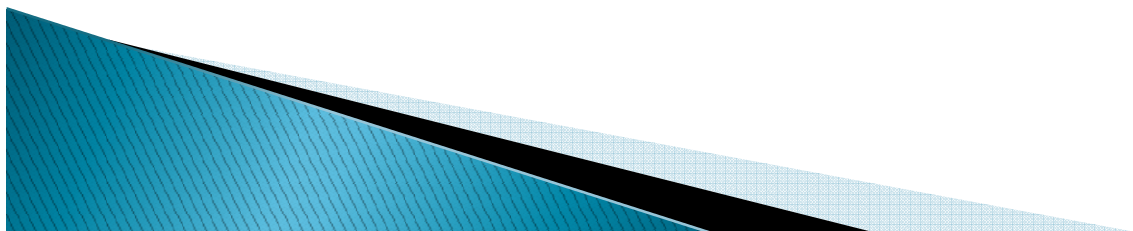


Passing Ghost Cells

- ▶ ghost zones values need to be filled before integration;
- ▶ Patches at the same level are synchronized.

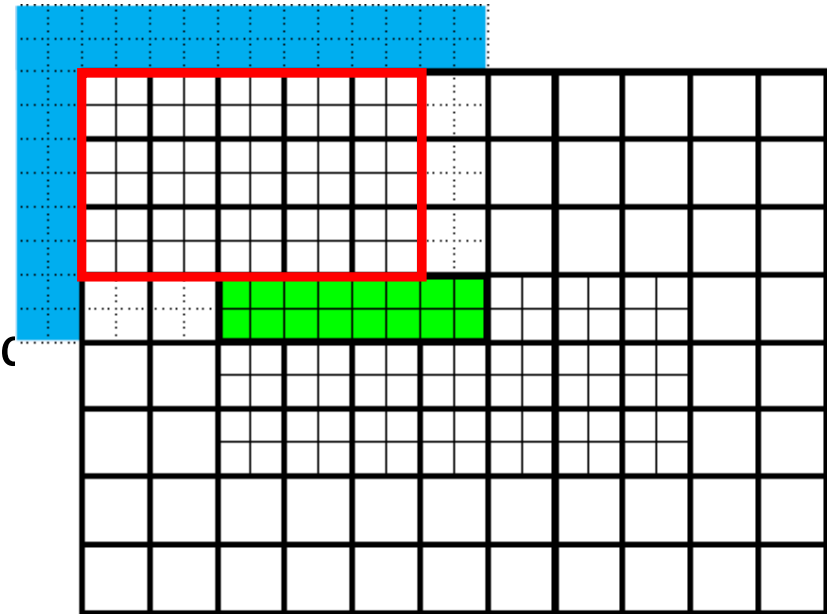


Courtesy of Dr. Andrea Mignone, University of Turin

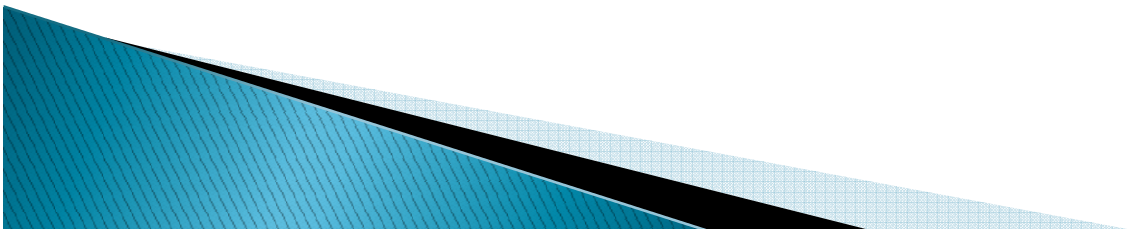


Passing Ghost Cells

- ▶ ghost zones values need to be filled before integration;
- ▶ Patches at the same level are synchronized;
- ▶ Physical boundaries are imposed externally;

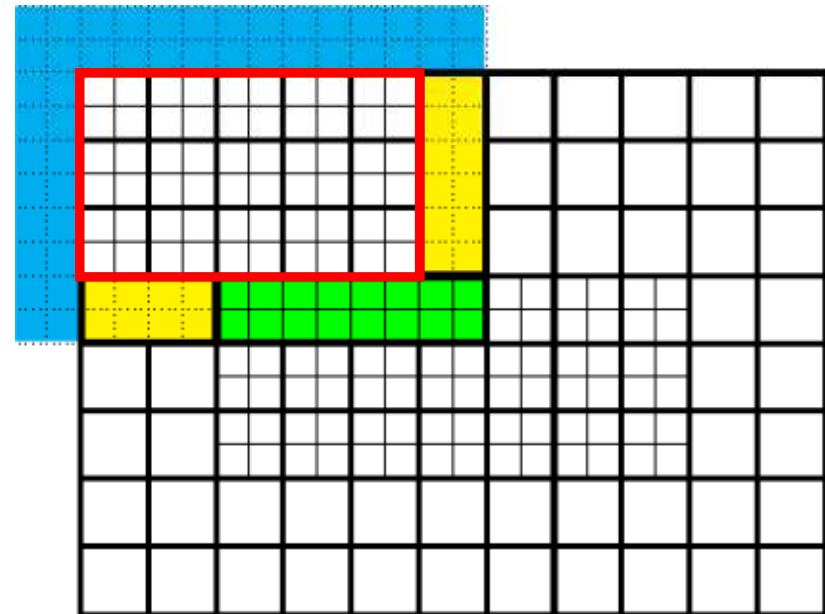


Courtesy of Dr. Andrea Mignone, University of Turin



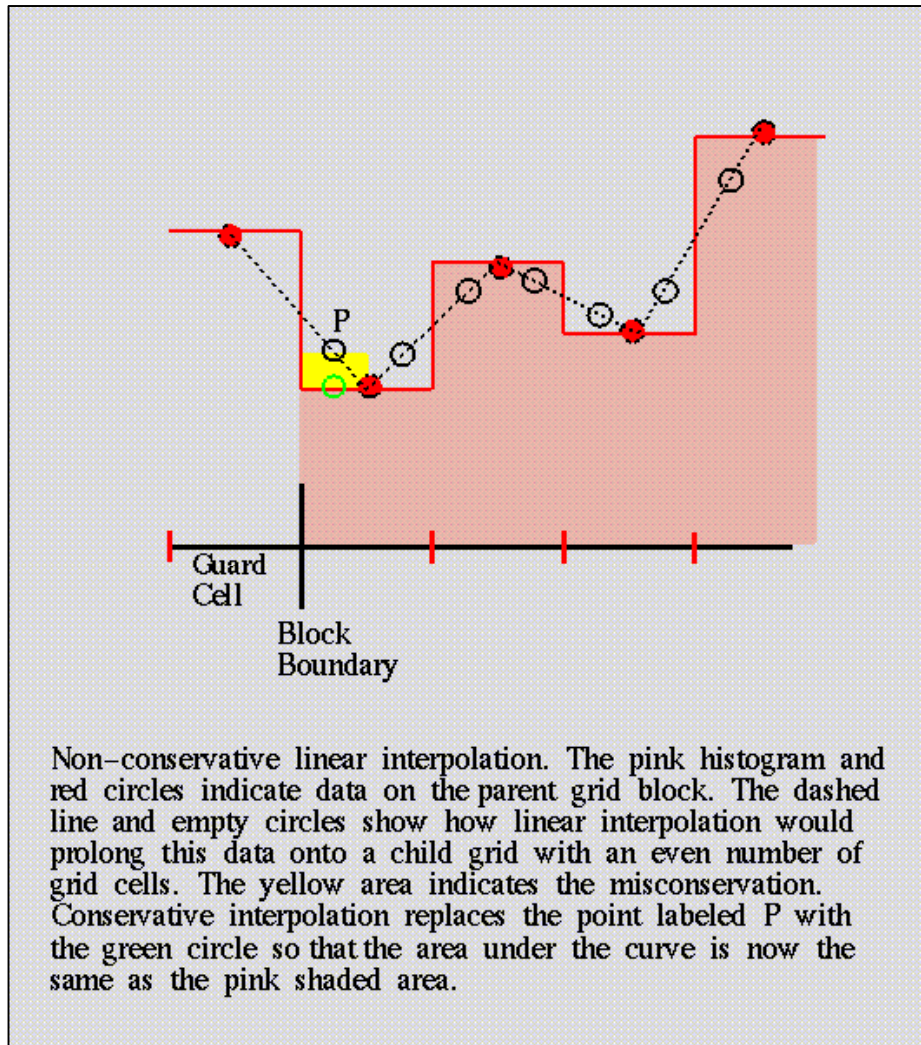
Passing Ghost Cells

- ▶ ghost zones values need to be filled before integration;
- ▶ Patches at the same level are synchronized;
- ▶ Physical boundaries are imposed externally;
- ▶ Fine-Coarse and Coarse-Fine interface need interpolation / averaging
- ▶ Integration proceeds as for the single-grid case



Courtesy of Dr. Andrea Mignone, University of Turin

Ghost cells communications



From [Paramesh User Guide](#)

When we pass the ghost cells to the adjoining blocks, if these blocks have different resolutions we must modify the data.

The most simple (and used) method is the interpolation method:

- If we must pass the ghost cells to a block with higher resolution we can use the linear interpolation to artificially increase the resolution.
- If we must pass the ghost cells to a block with lower resolution we can average the data in order to have the same resolution.

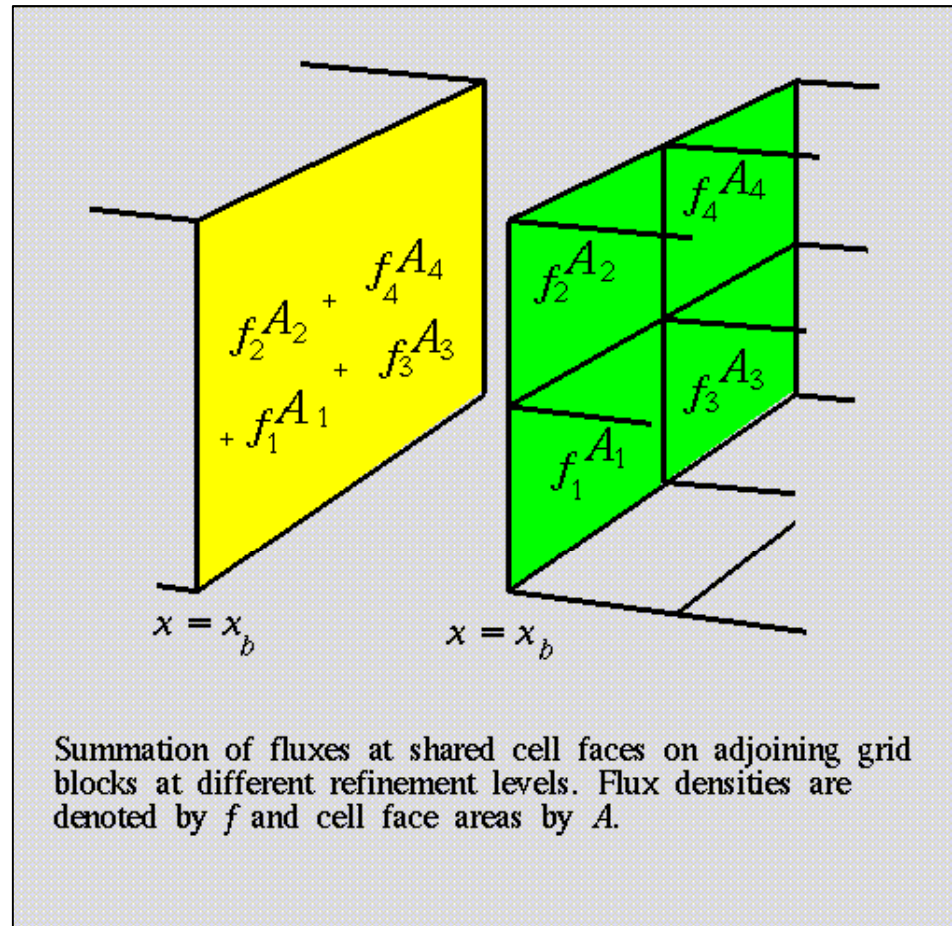
Pros:

- Easy to implement
- It is possible to use many different kind of interpolation (linear, quadratic, and so on) increasing precision

Cons:

- Non-conservative

Passing ghost cells to neighbors blocks



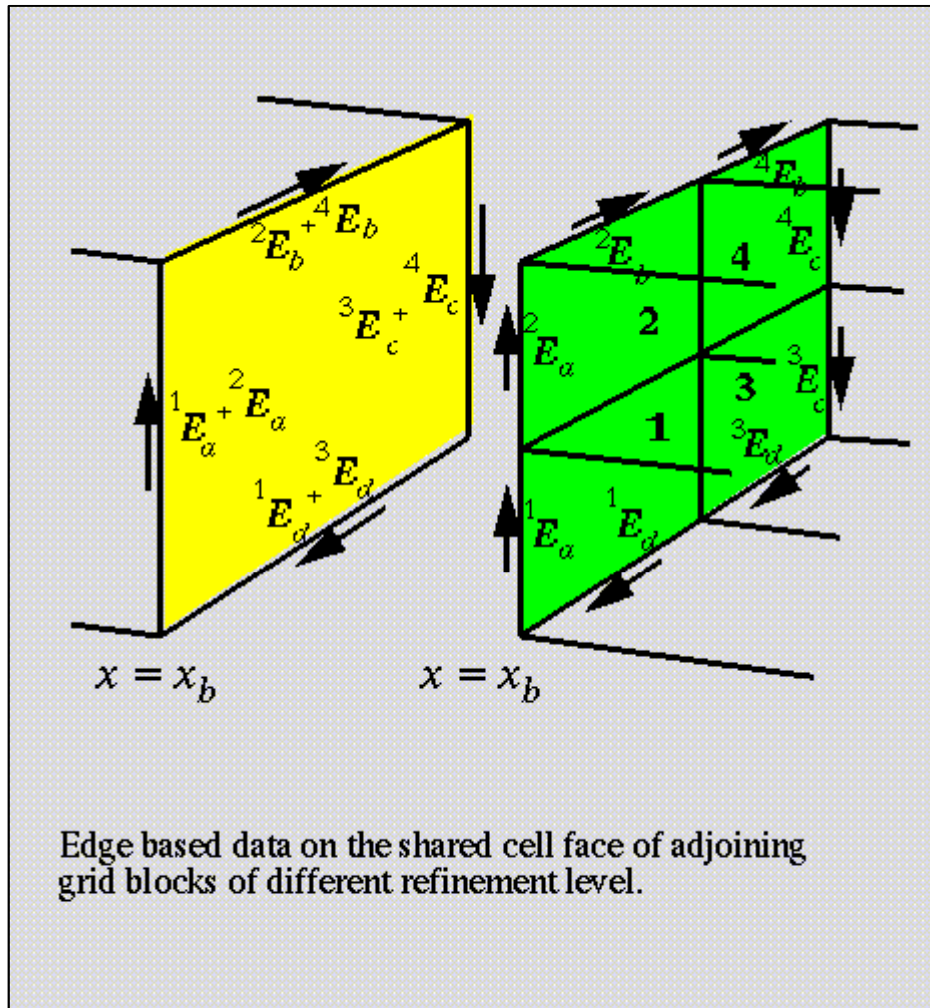
Flux conservation:

It is possible to ensure flux conservation after the interpolation checking the equation:

$$f_1 A_1 + f_2 A_2 + f_3 A_3 + f_4 A_4 = F_{\text{Tot}} A_{\text{Tot}}$$

From [Paramesh User Guide](#)

Passing ghost cells to neighbors blocks

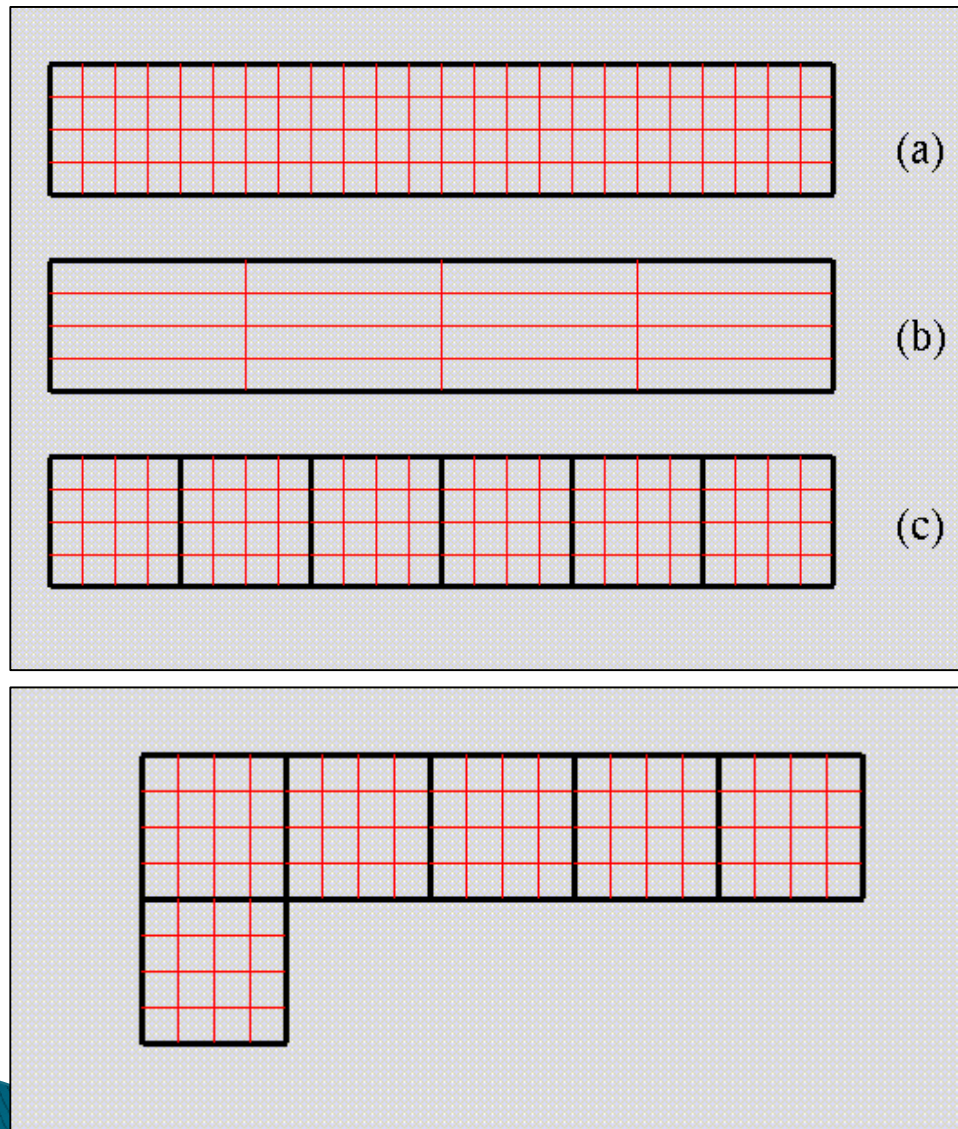


Circulation integral control:
It is possible also to check the value of some physical quantity at the edges of the cells

From [Paramesh User Guide](#)

NOTE: Both these three methods are usable in order to change the resolution of the blocks.

Particular Geometries



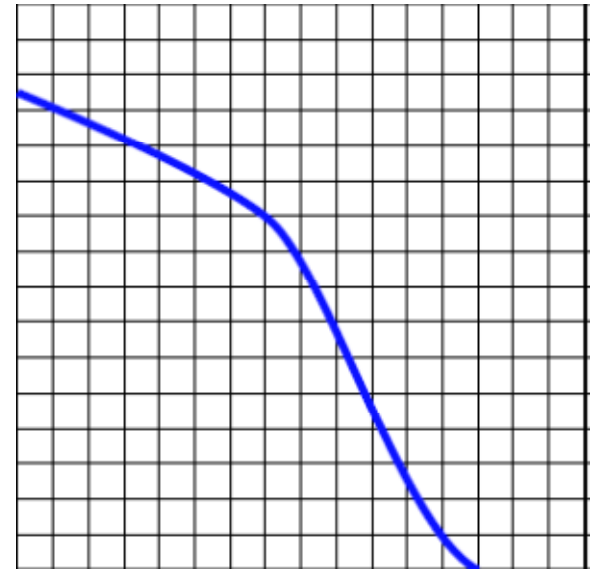
When we have a non symmetric computational domain many different approach can be used. For a rectangular domain:

- We can have different number of points per block on x and y directions ($dx = dy$)
- We can have different number of points on x and y directions ($dx \neq dy$)
- We can use more blocks on the x directions , and 1 block on x direction (same resolution on x and y, and more parallelizable)

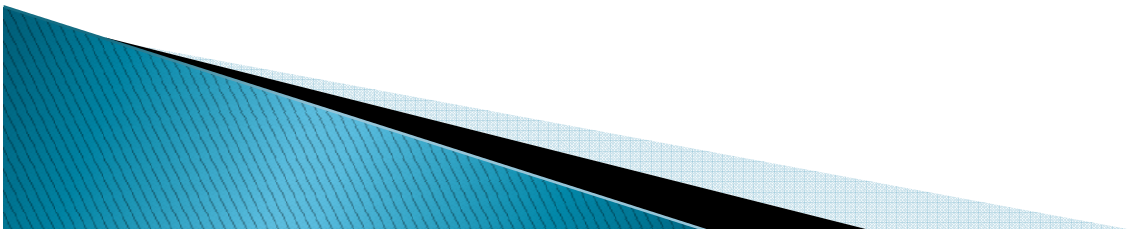
If we have more complicate computational domains, we can always use more blocks in order to fully cover the whole domain.

How to refine

- ▶ fill data, level 0

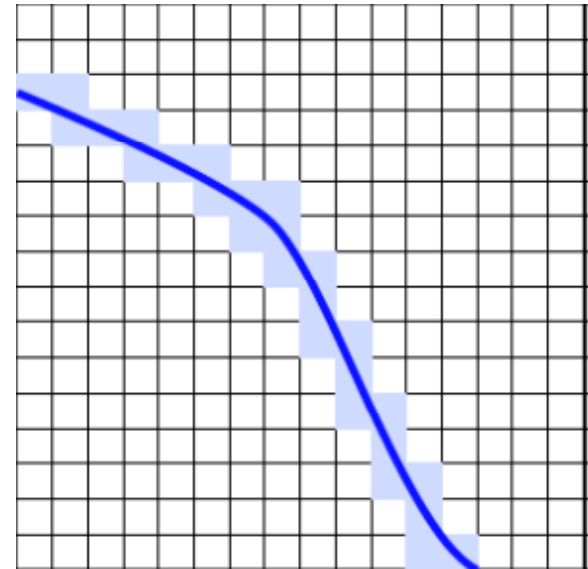


Courtesy of Dr. Andrea
Mignone, University of Turin

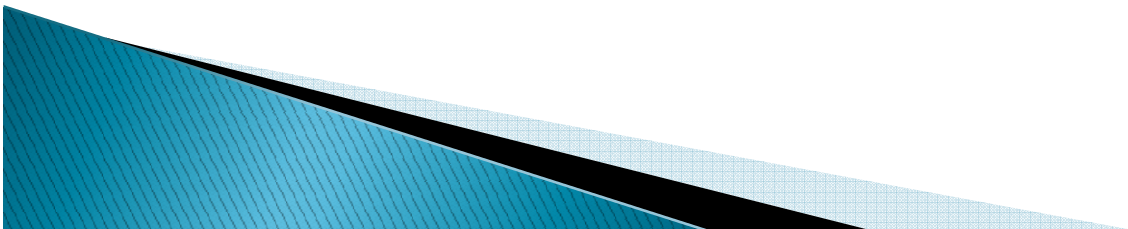


How to refine

- ▶ fill data, level 0
- ▶ find where refinement is needed;

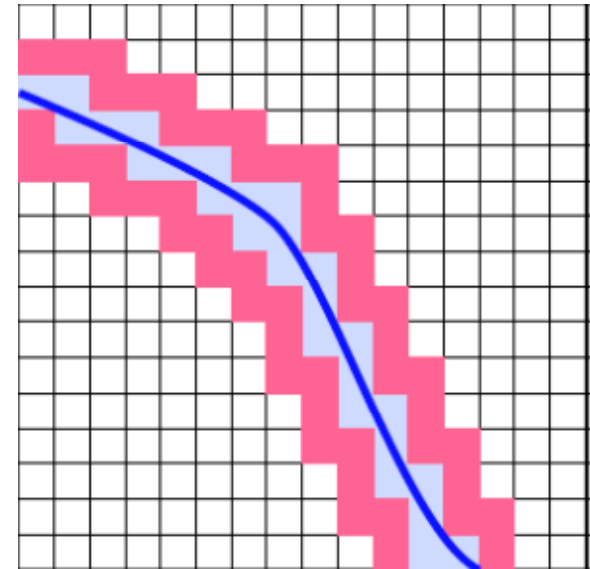


Courtesy of Dr. Andrea
Mignone, University of Turin

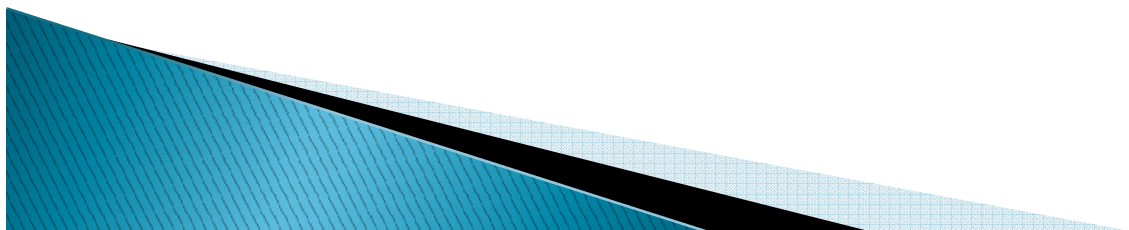


How to refine

- ▶ fill data, level 0
- ▶ find where refinement is needed;
- ▶ group cells into patches according to the “grid efficiency”

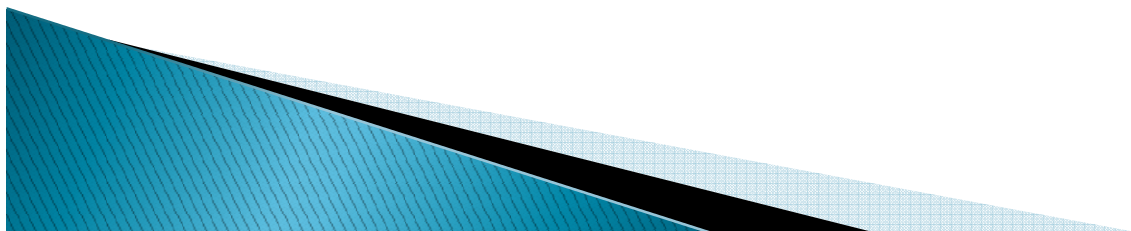
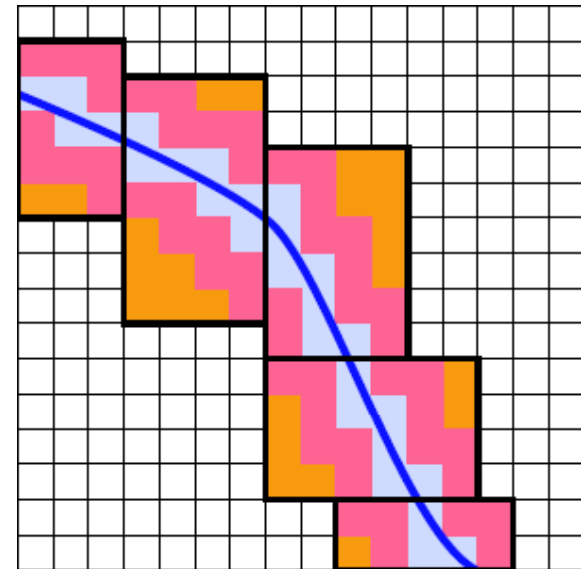


Courtesy of Dr. Andrea Mignone, University of Turin



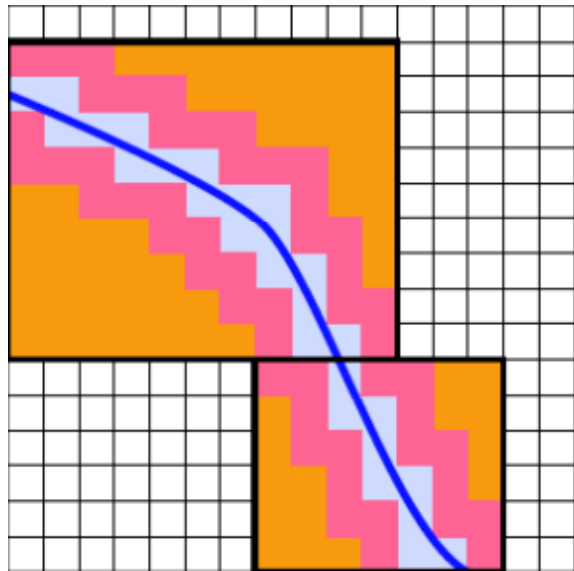
How to refine

- ▶ fill data, level 0
- ▶ find where refinement is needed;
- ▶ group cells into patches according to the “grid efficiency”
- ▶ refine and ensure proper nesting

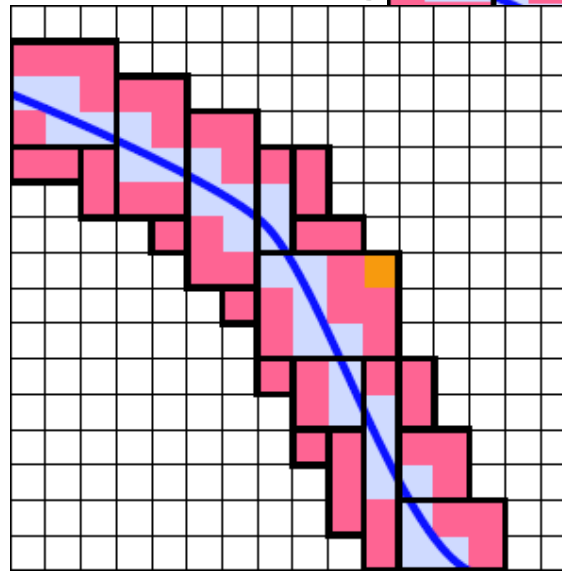


How to refine

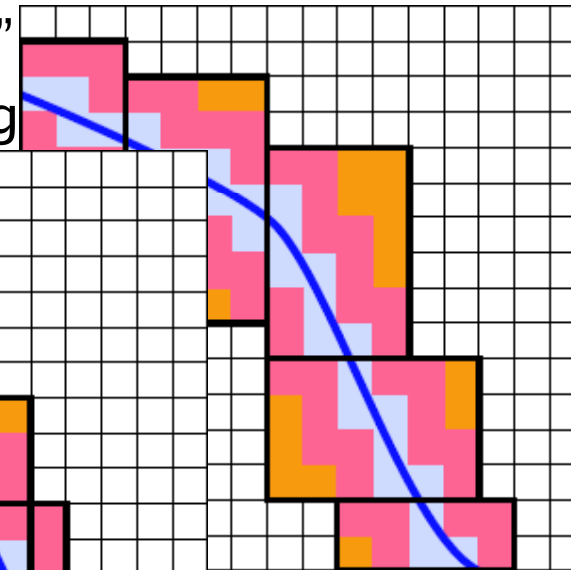
- ▶ fill data, level 0
- ▶ find where refinement is needed;
- ▶ group cells into patches according to the “grid efficiency”
- ▶ refine and ensure proper nesting



efficiency = 0.5

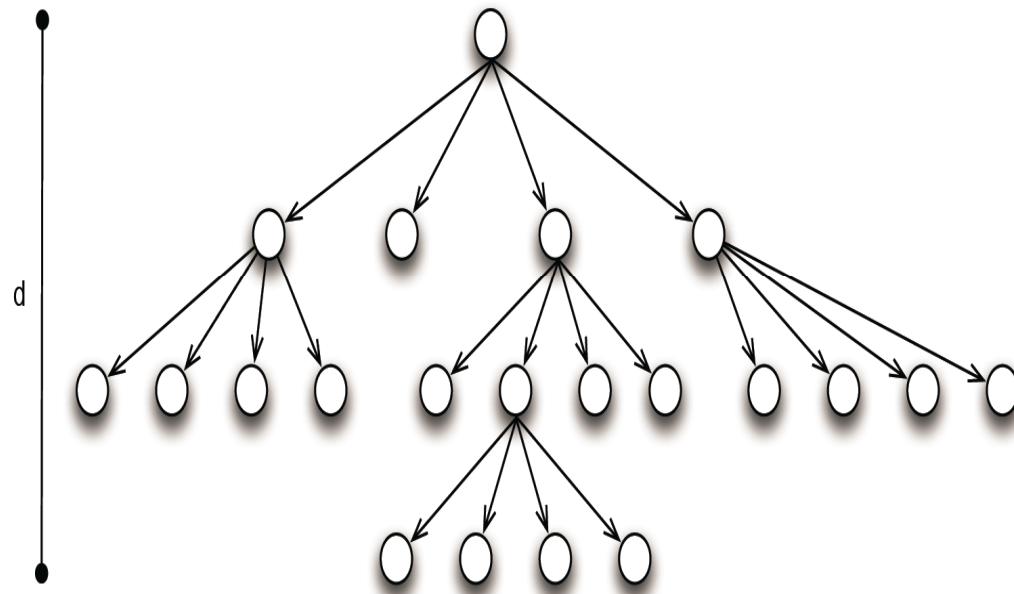


efficiency = 0.9



efficiency = 0.7

Little more background on AMR



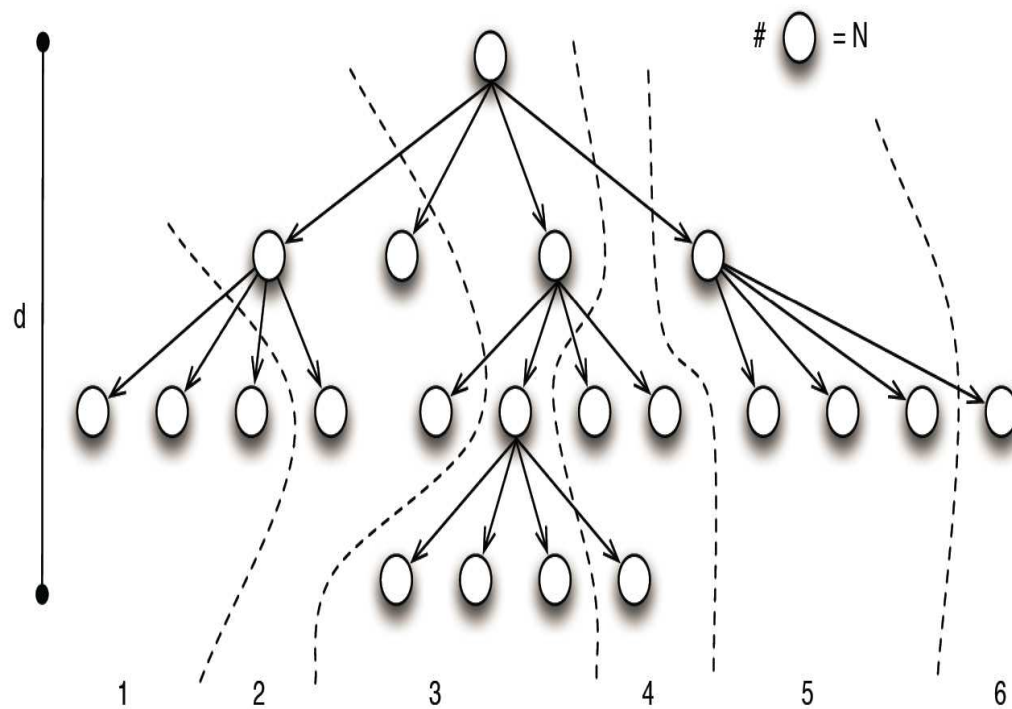
Refinement structure can be represented using a quad-tree (2D)/oct-tree (3D)

An important condition in AMR

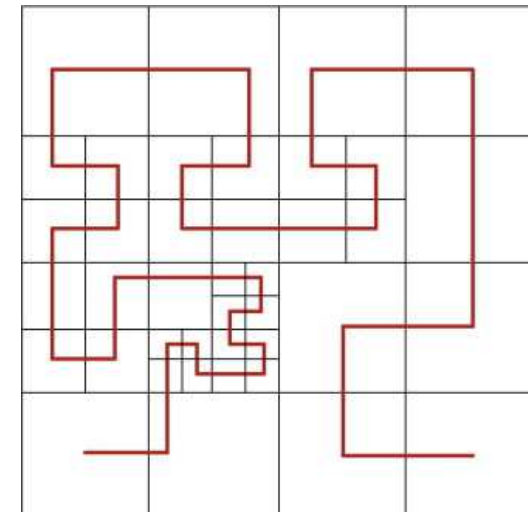
Refinement levels of neighboring blocks differ by ± 1

Note: This is generally true, but Chombo library allow more than 1 refinement level discrepancy.

Traditional Approach - Parallel Implementations



- A set of blocks assigned to a process
- Use space-filling curves for load balancing



Traditional Approach - Disadvantages

- Adaptive mesh restructuring:

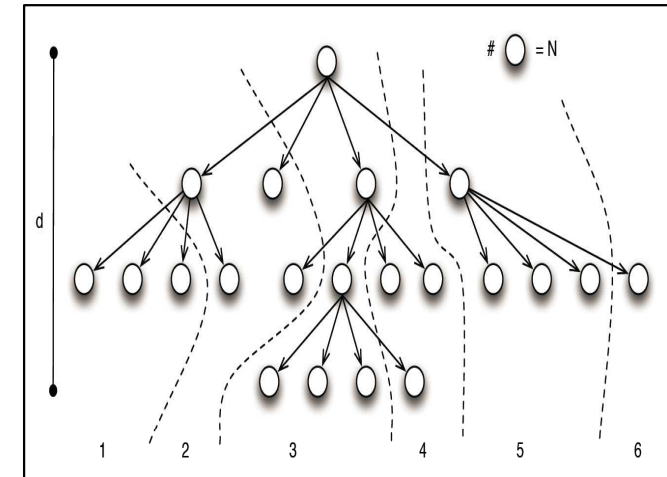
- Tree metadata replicated on each process
 - ✓ Required memory increases with # of cores
 - ✓ Memory can become a problem if we use more than 10^5 cores (and more than 10^6 boxes)
- Level-by-level restructuring
 - Ripple propagation
 - Step needed to propagate restructuring \propto level of refinement (d)

- Load Balancing

- Memory needed \propto Number of blocks used
- Time needed \propto Number of blocks used

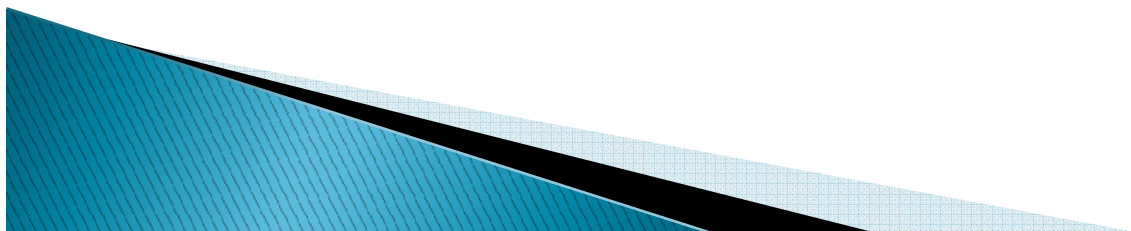
- Currently for 3D problems with less than 10^6 boxes standard AMR library scales up to few tens of thousands of cores

- This is a serious problem considering that next generation supercomputers will require the use of many hundreds of thousands of cores



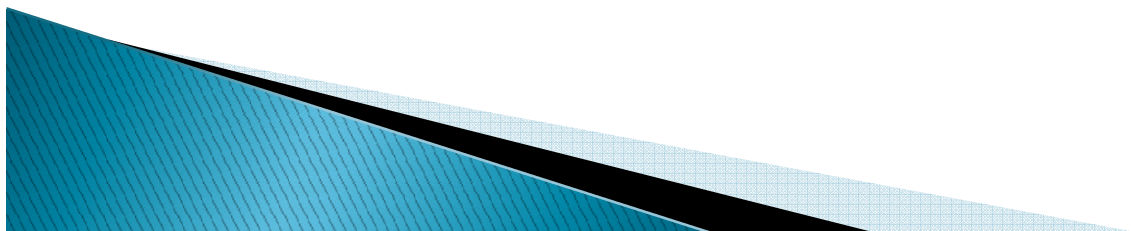
Improving AMR: Possible strategies

1. Compress tree metadata
 - Already implemented in the last versions of CHOMBO, PARAMESH and SAMRAI libraries
2. Rewrite the algorithm for coarse-fine interpolation in order to minimize communications
 - Already implemented in the last versions of CHOMBO, PARAMESH libraries
 - Using these first two methods it is possible to scale up to 2×10^5 cores using 10^7 grid cells
3. Use a distributed memory version for tree metadata
 - Currently Langer et al are working on the implementation of this algorithm on CHARM++



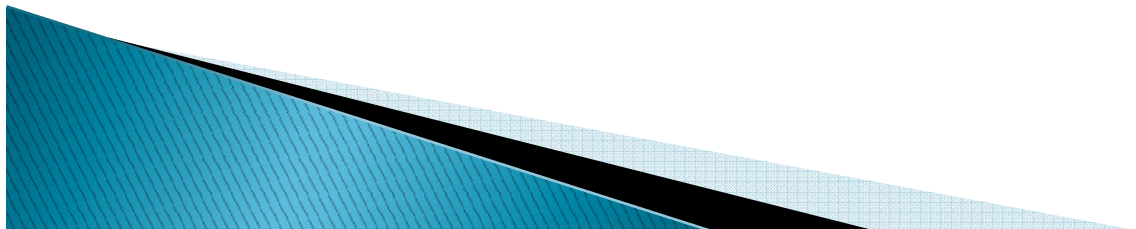
Some additional information about PARAMESH

- Written in Fortran 90
- Easy to implement on a existing code
- Support many geometries (Cartesian, cylindrical, spherical, from 1D to 3D)
- Refinement levels of neighbouring blocks differ by ± 1
- Compatible with hdf5 format
- Some simple routine are already written by the authors of the library in order to save the data and the grid structure into Fortran binary format, and hdf5 format.
- Easy visualization of the results using many external programs (e.g. visit)

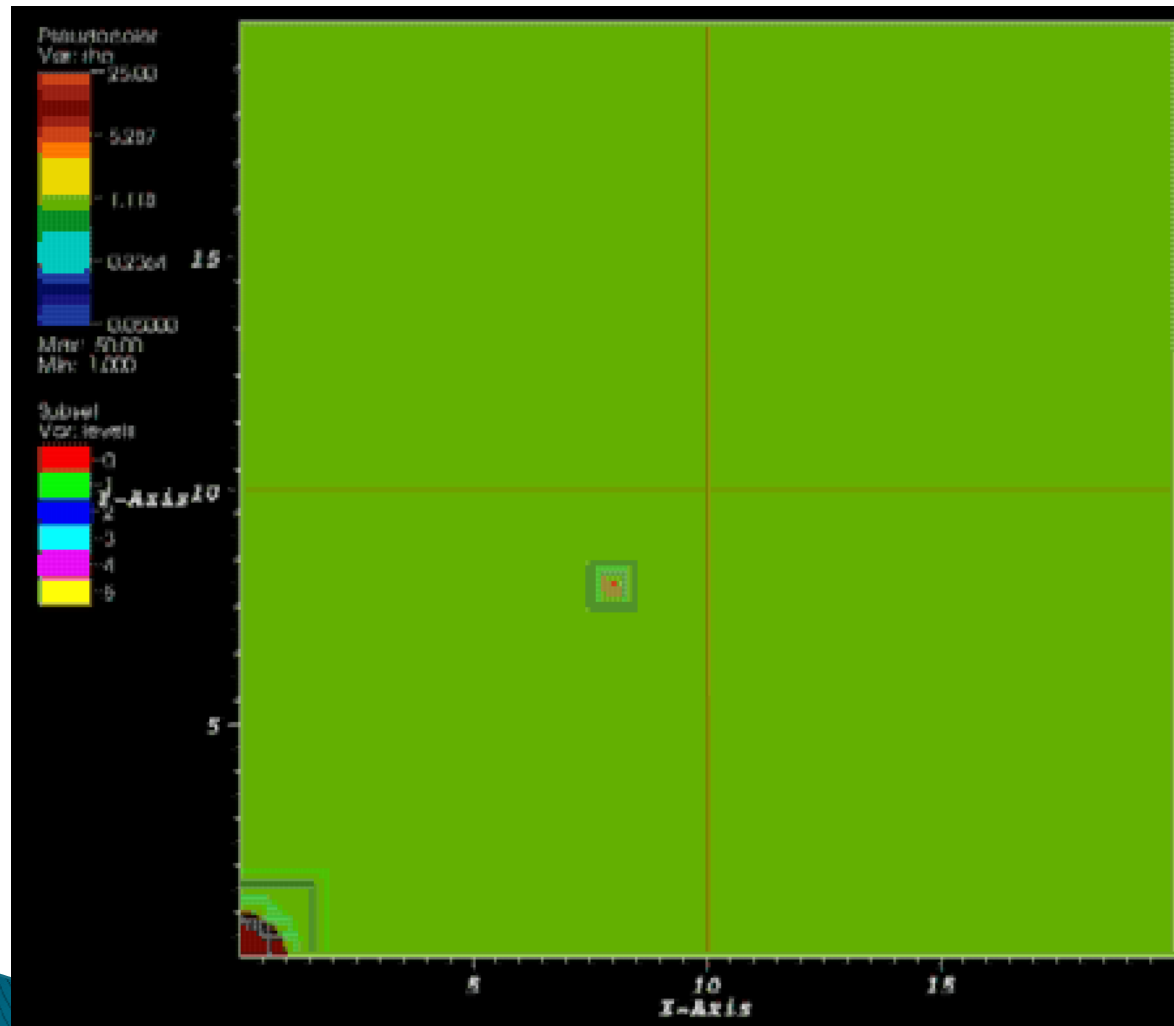


Some additional information about CHOMBO

- Written in C
- Easy to implement on a existing code
- Support many geometries (Cartesian, cylindrical, spherical, from 2D to 3D)
- Compatible with hdf5 format
- Easy visualization of the results using many external programs (e.g. visit)



Example: 2D Blast Wave



Problem: Blast Wave
– Cloud Interaction

Base Grid: 128x128

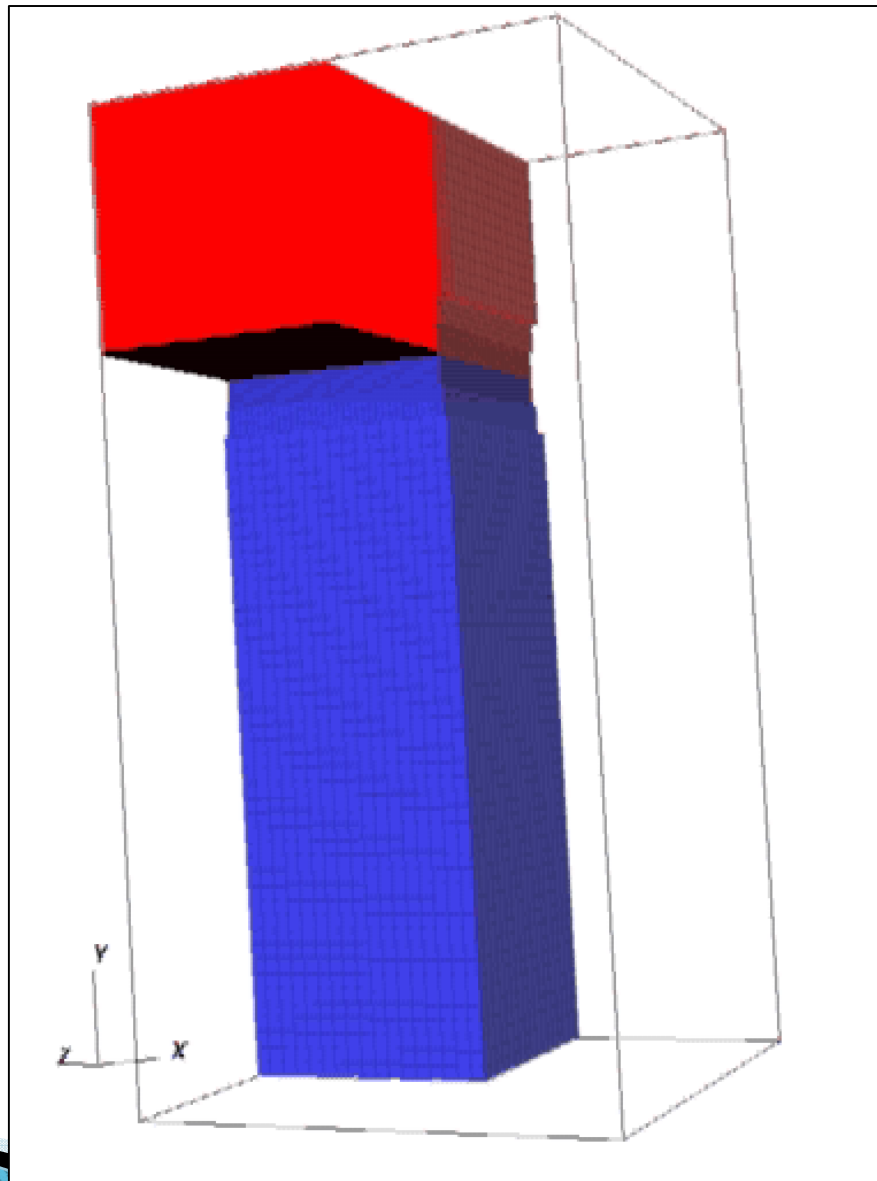
Levels of Refinement:
5 (eq. 4096x4096)

Method: Unsplit PPM

Code: PLUTO +
Chombo Lib

Courtesy of Dr. Andrea Mignone, University of Turin

Example: 3D Rayleigh-Taylor



Problem:

Rayleigh Taylor

Base Grid:

32x64x32

Levels of Refinement:

2 (eq. 128x256x128)

Method:

Unsplit PPM

Code:

PLUTO + Chombo Lib

Courtesy of Dr. Andrea Mignone,
University of Turin

Thank you for attention

The author is grateful to Dr. A. Mignone for the help given during the making of these slides