Patch-based AMR algorithms

Donna Calhoun

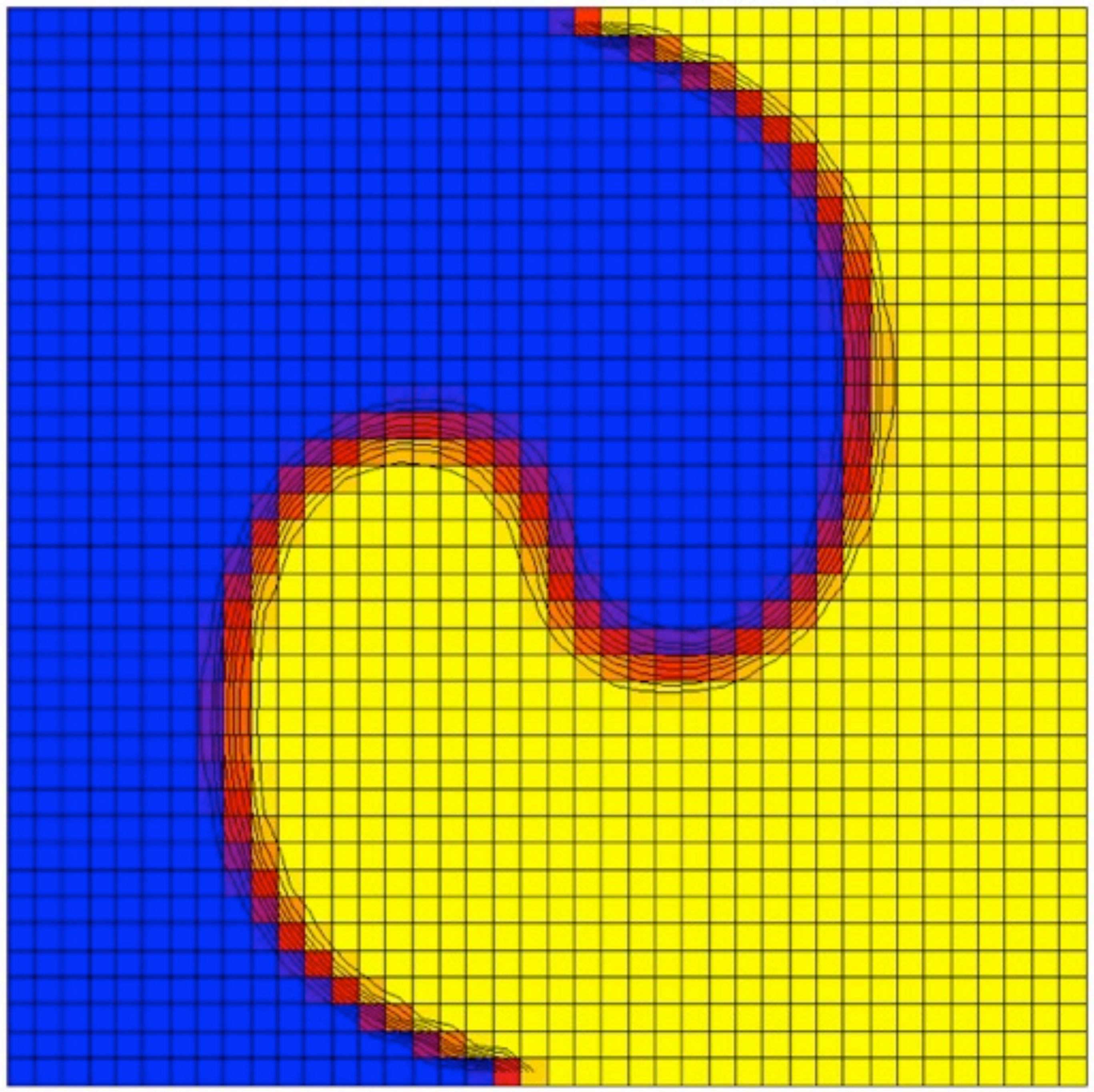
Boise State University

$[\text{HPC}]^3$

Kaust University

February 4-8, 2012

Computations on Cartesian meshes



Why Cartesian meshes?

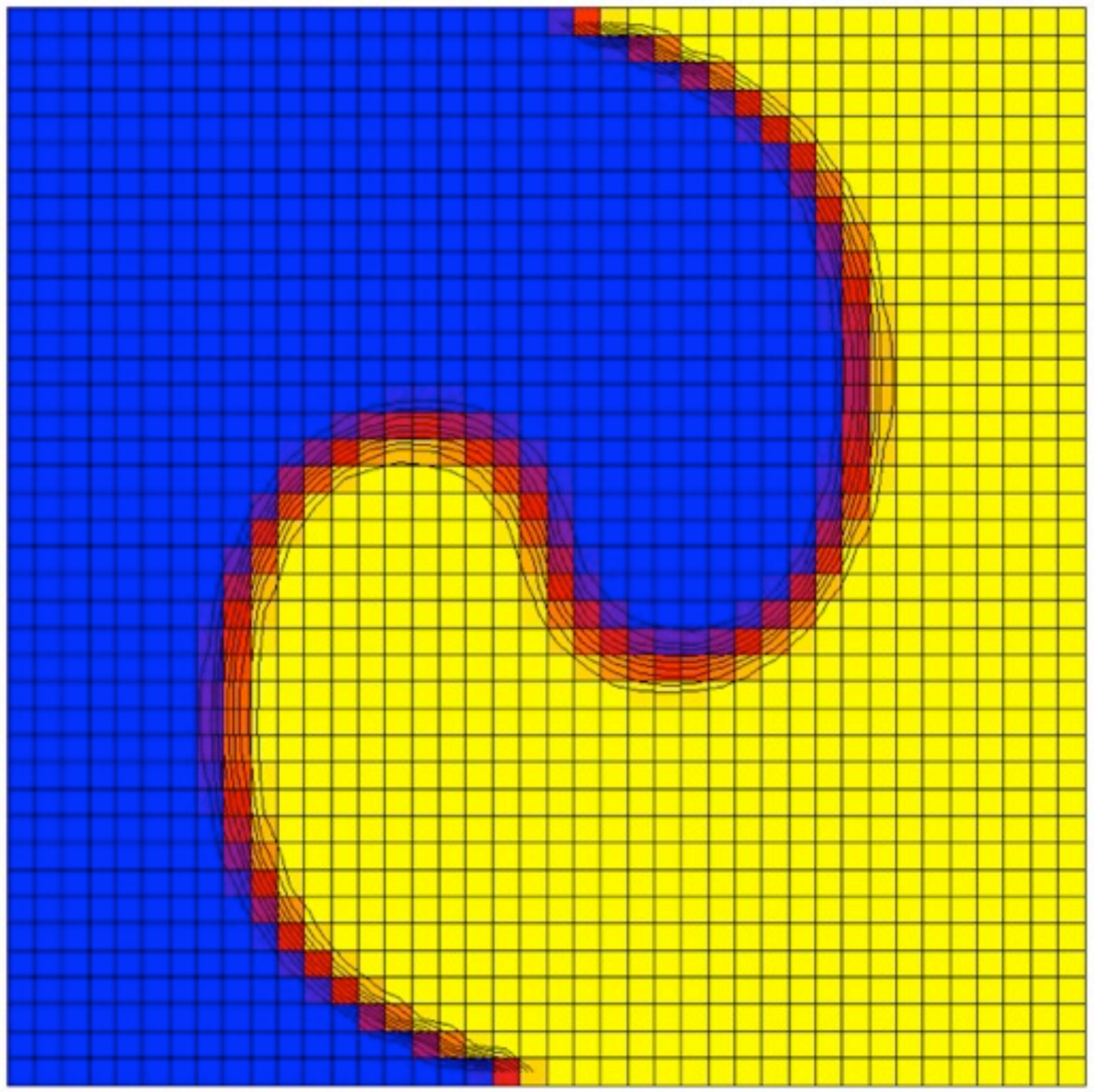
Why use Cartesian meshes instead of unstructured meshes?

- Mesh generation is easier, if not trivial
- Solution is not dependent on the quality of the mesh
- Algorithms are easier to construct on smooth logically Cartesian meshes and the results are more accurate than on unstructured, non-smooth meshes
- Layout of the Cartesian data maps directly to the computer memory layout, improving runtime performance

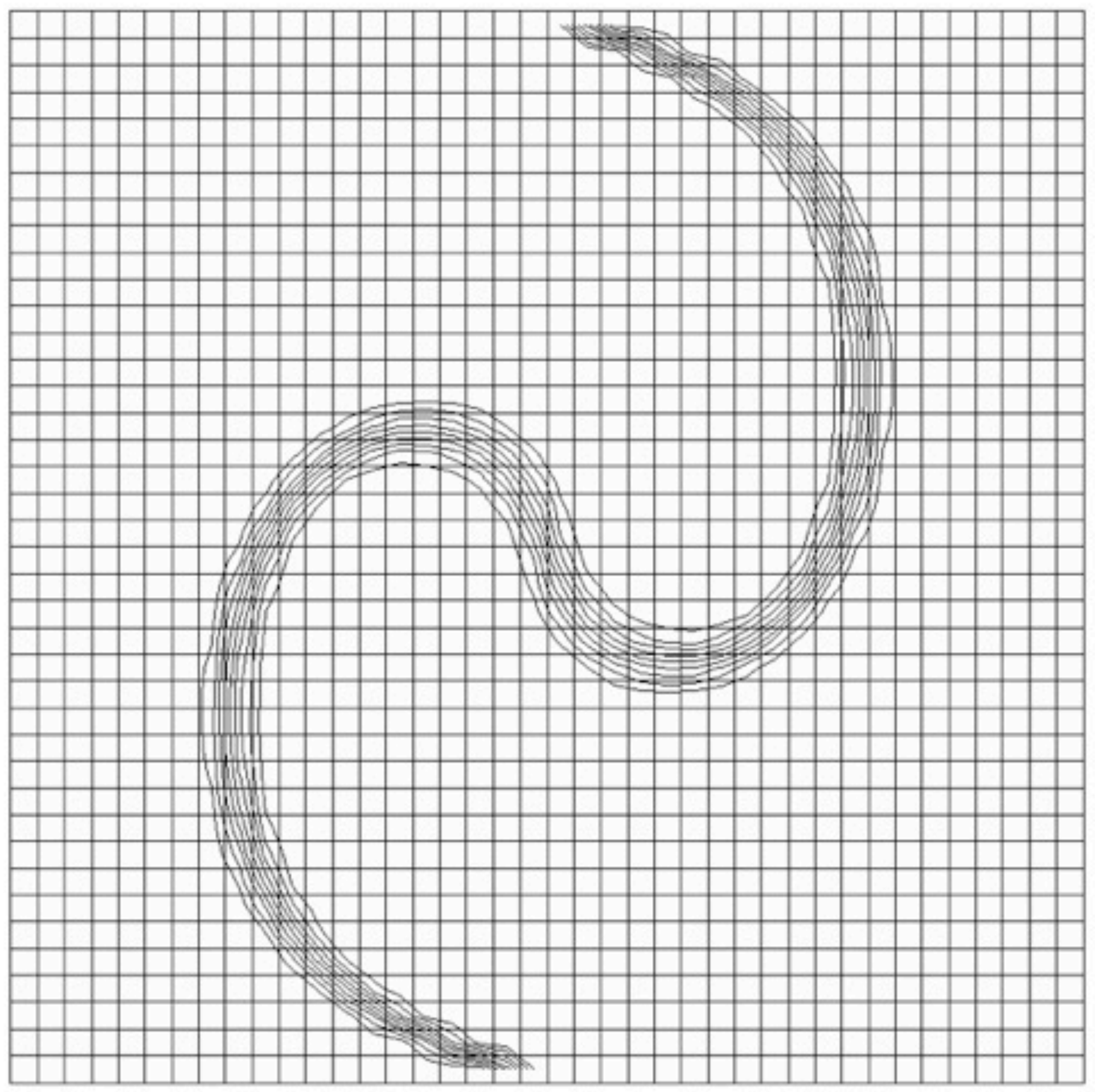
But ...

- Use of computational resources is not efficient.

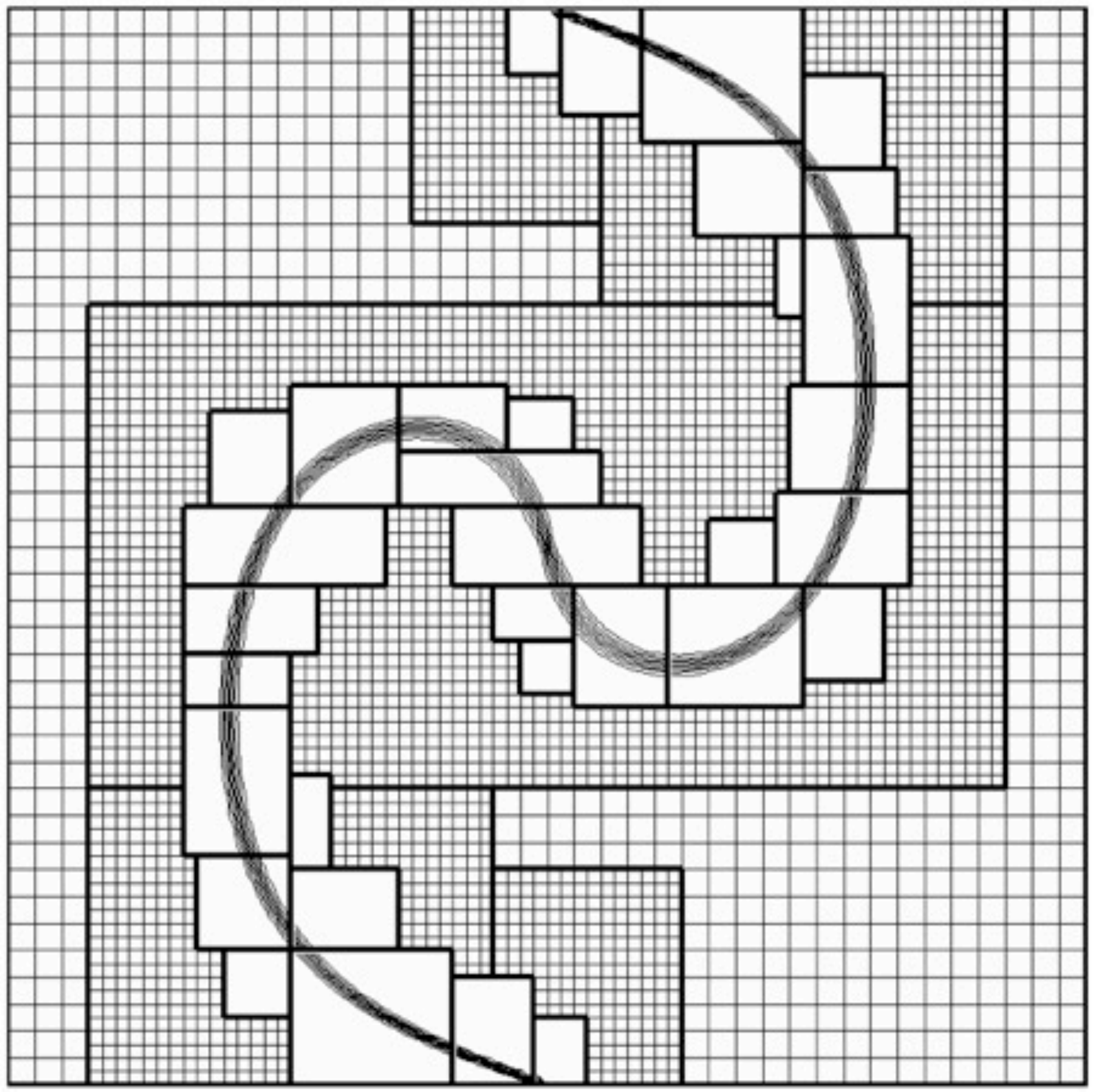
Computations on Cartesian meshes



Solving on Cartesian grids

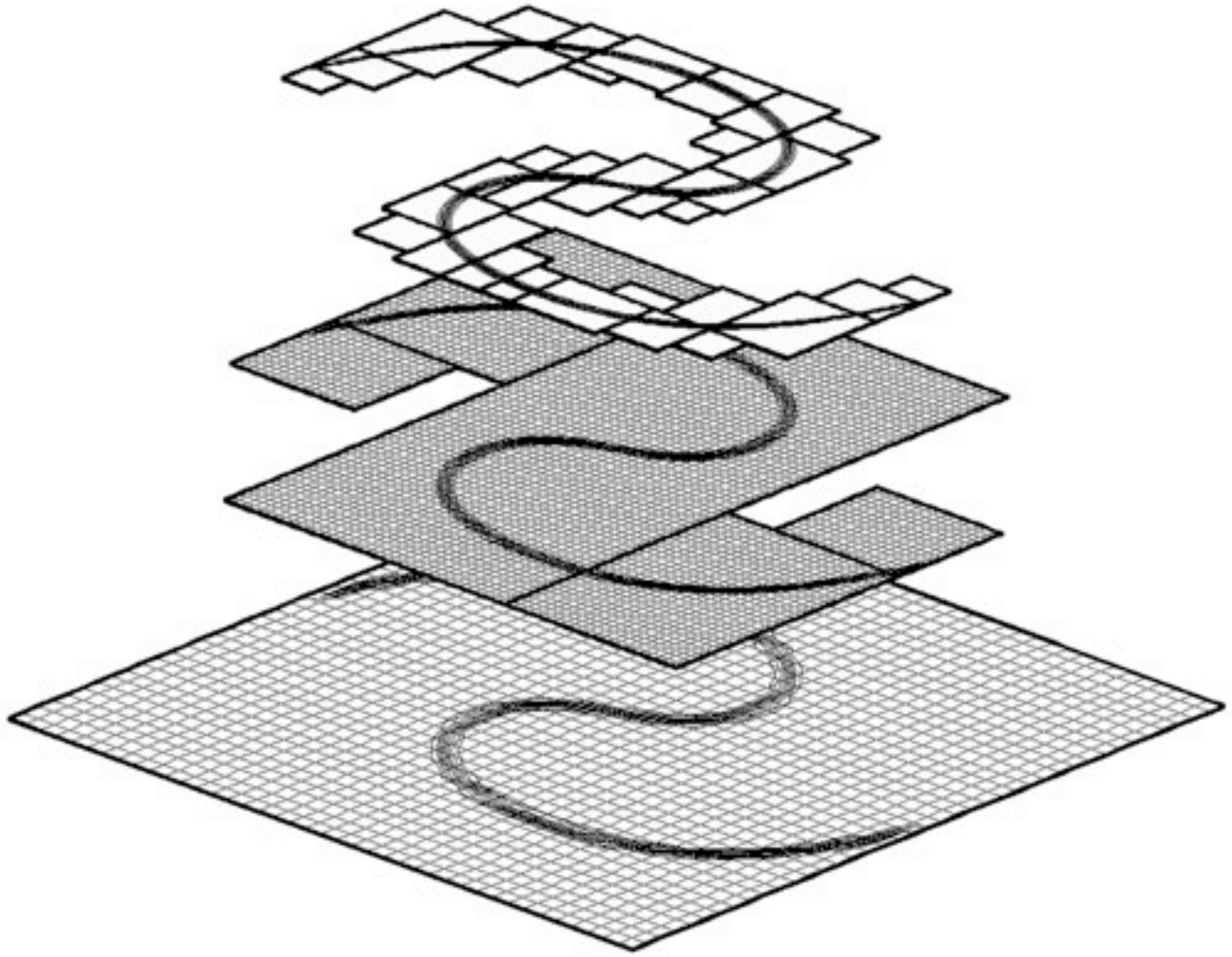


Patch-based AMR (Berger-Oliger)



- Only resolve near the spatial features of interest
- Try to keep advantages of logically rectangular grid

Hierarchical view



- Data is easy to store using (i,j,k) indexing
- Each grid has its own CFL condition (for explicit schemes)
- Existing single grid algorithms can easily be used

Patch-based AMR

Questions :

- Schemes for managing time stepping - how do we manage the solution process on multiple grids?
- How is the solution actually defined?
- How do we ensure smoothness and conservation?
- What might an elliptic solve look like on an AMR hierarchy?

Finite Volume Method

Assume a discretization in *flux form*

Explicit hyperbolic PDEs

$$\frac{Q^{n+1} - Q^n}{\Delta t} + \frac{F^n_{i+1/2} - F^n_{i-1/2}}{\Delta x} = \Psi(Q, \ldots)$$

- Single step algorithm. Lax-Wendroff ideas give us second order
- Fluxes come from solving Riemann problems at cell interfaces.
- Wave limiters may be used to suppress spurious oscillations

Finite volume methods

Elliptic PDEs

$$\frac{F^n_{i+1/2} - F^n_{i-1/2}}{\Delta x} = \Psi$$

Parabolic PDE

$$\frac{Q^{n+1} - Q^n}{\Delta t} + \frac{F^{n+1}_{i+1/2} - F^{n+1}_{i-1/2}}{\Delta x} = \Psi(Q^n, \ldots)$$

where fluxes are given by $F = \pm q_x$.

AMR grid requirements

- Coarse grid and fine grid boundaries are aligned
- Finer meshes are properly nested into coarser ones
- Buffer zone of cells around each mesh
- Assume that we use the same discretization scheme at each level
- Ghost cell values are obtained from coarse grid or neighboring fine grids, if available
- Do not allow grids which overlap multiple levels of refinement

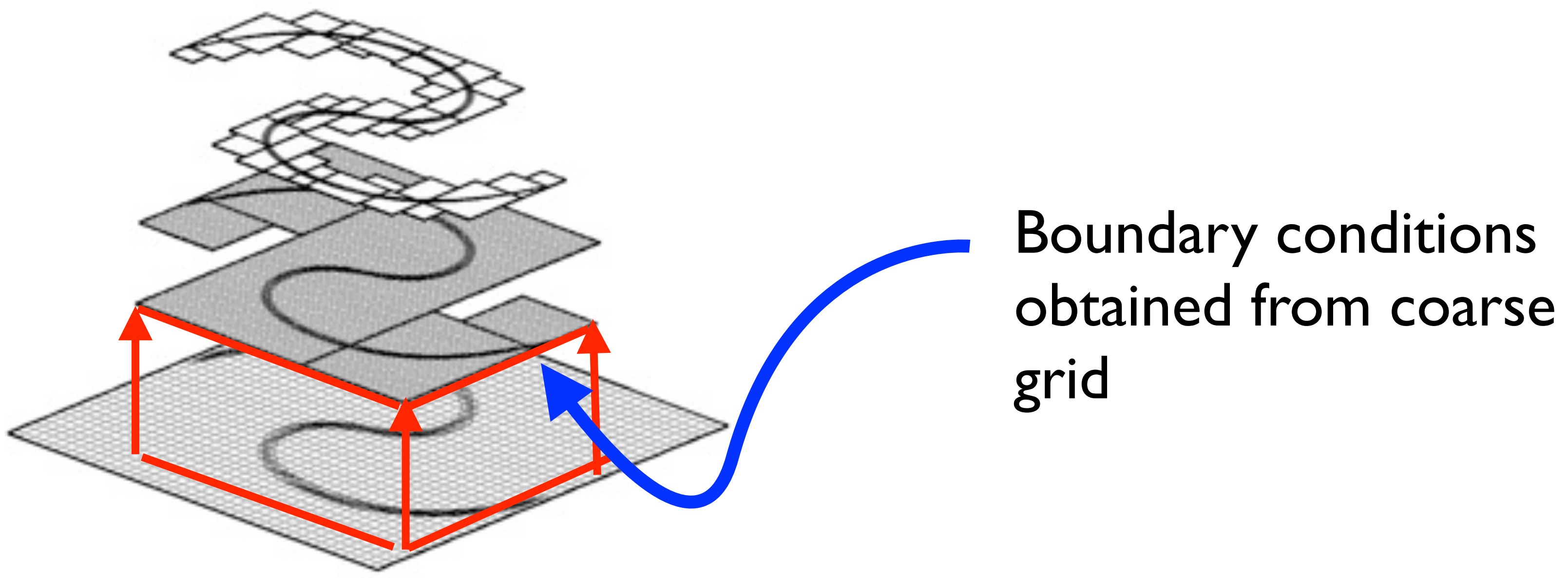
AMR grid algorithms - numerical requirements

- Single grid Cartesian layout should be used whenever possible
- Avoid use of complicated stencils at coarse/fine grid interfaces
- 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

AMR patch-based time-step advance

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

1. Advance at the coarsest level by time step Δt
2. Advance fine grid R time steps, by a time step $\Delta t/R$
3. Use boundary conditions from old and new coarse grid solutions.
4. Average solution from fine grids to coarse grid
5. Regrid to construct new grid hierarchy, if necessary



Example : Allen-Cahn equation

Flow by mean curvature

$$u_t = \nabla^2 u + \frac{1}{D^2}\left(u - u^3\right)$$

Discretize using backward Euler

$$u^{n+1} - \Delta t \nabla^2 u^{n+1} = u^n + \frac{\Delta t}{D^2}\left(u^n - (u^n)^3\right)$$

Solve for fixed number of time steps :

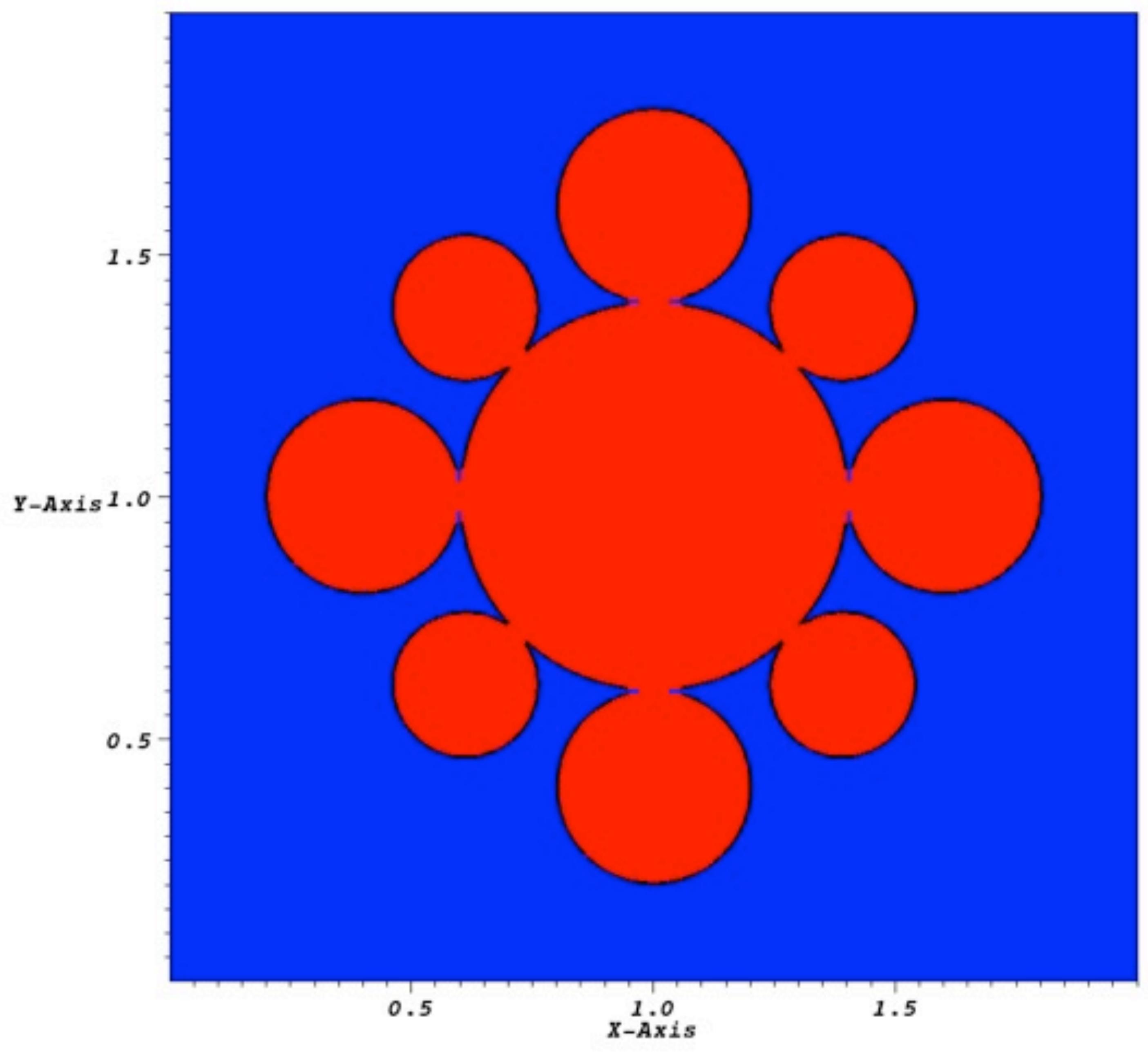
$$L(u^{n+1}) = f(u^n), \qquad n = 1, 2, \ldots, T/\Delta t$$

where

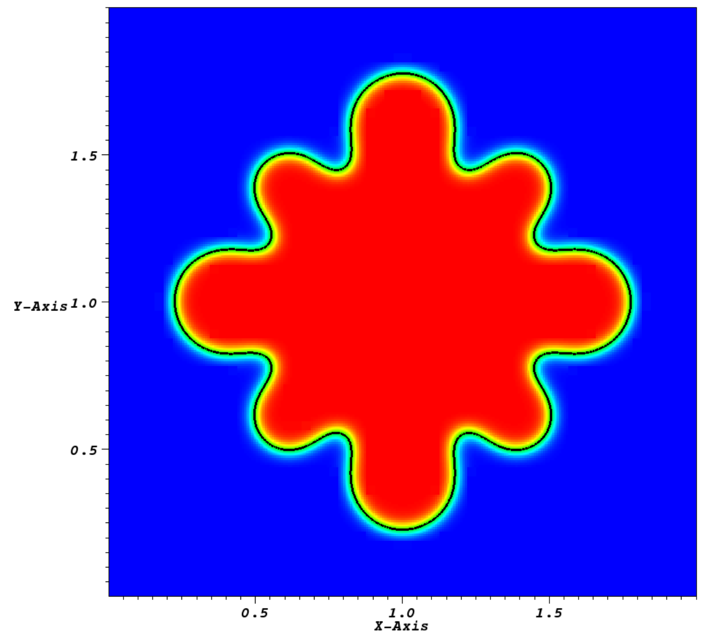
$$L(u) = (\alpha + \beta\nabla^2)\, u$$

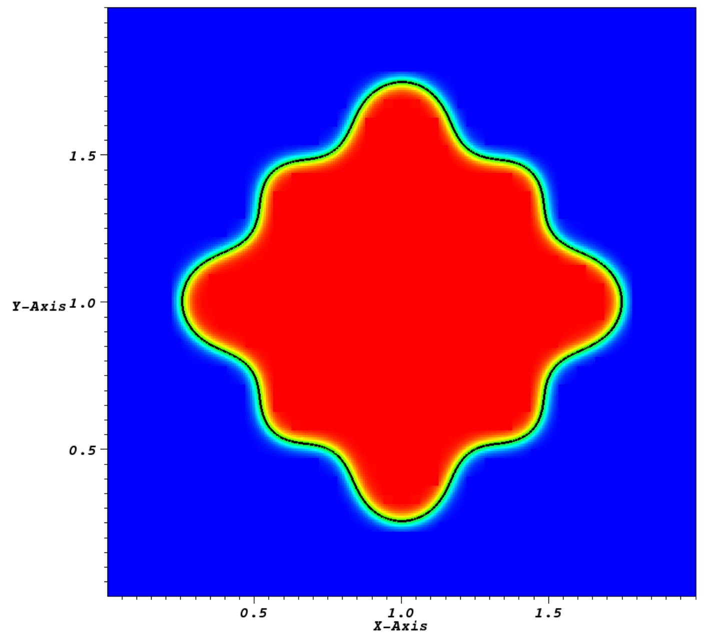
for $\alpha = 1$ and $\beta = -\Delta t$.

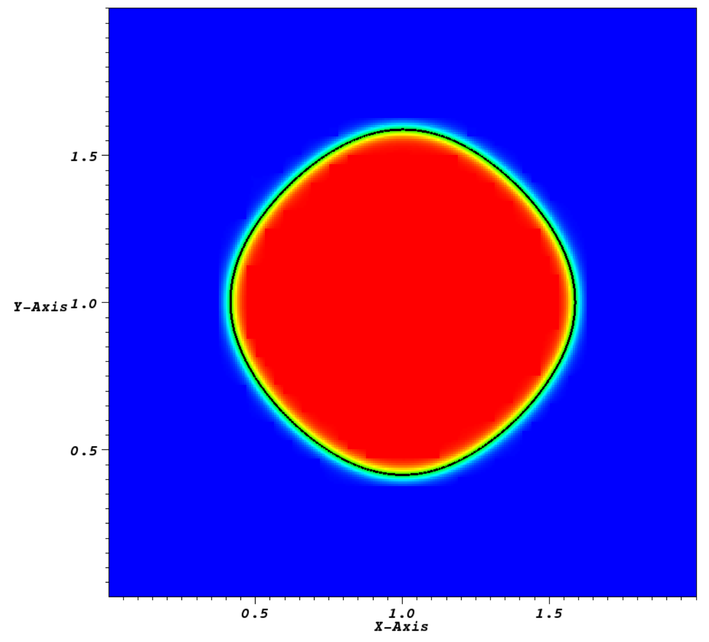
Flow by mean curvature

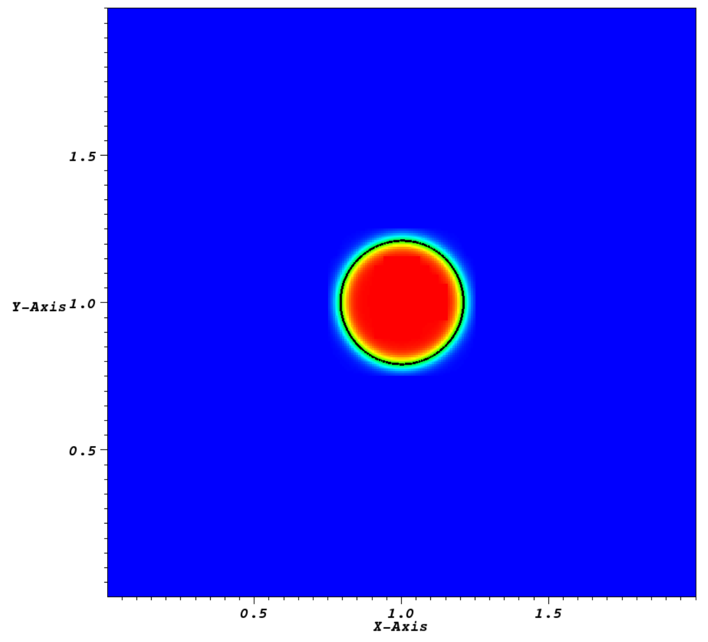


Flow by mean curvature

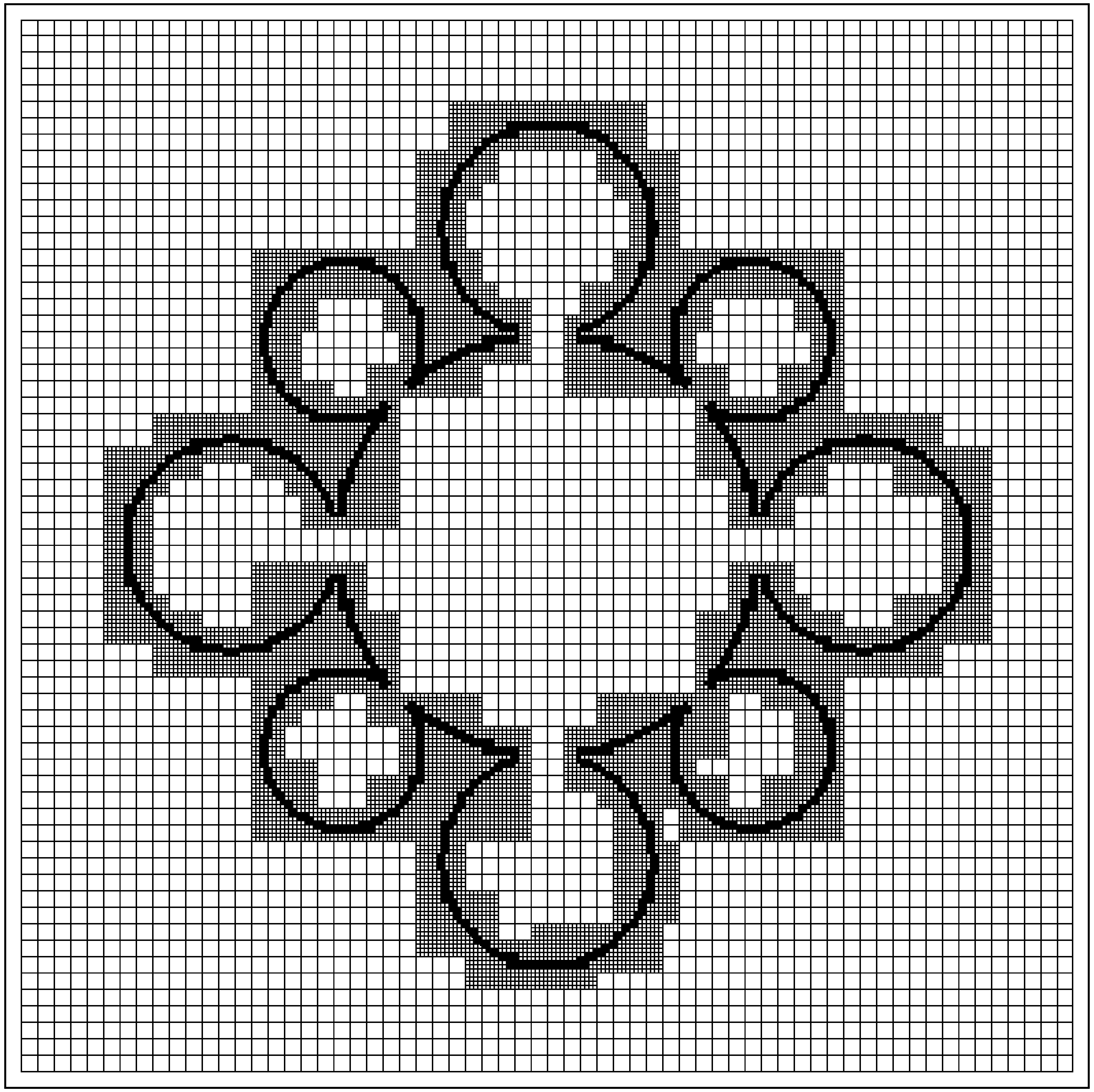




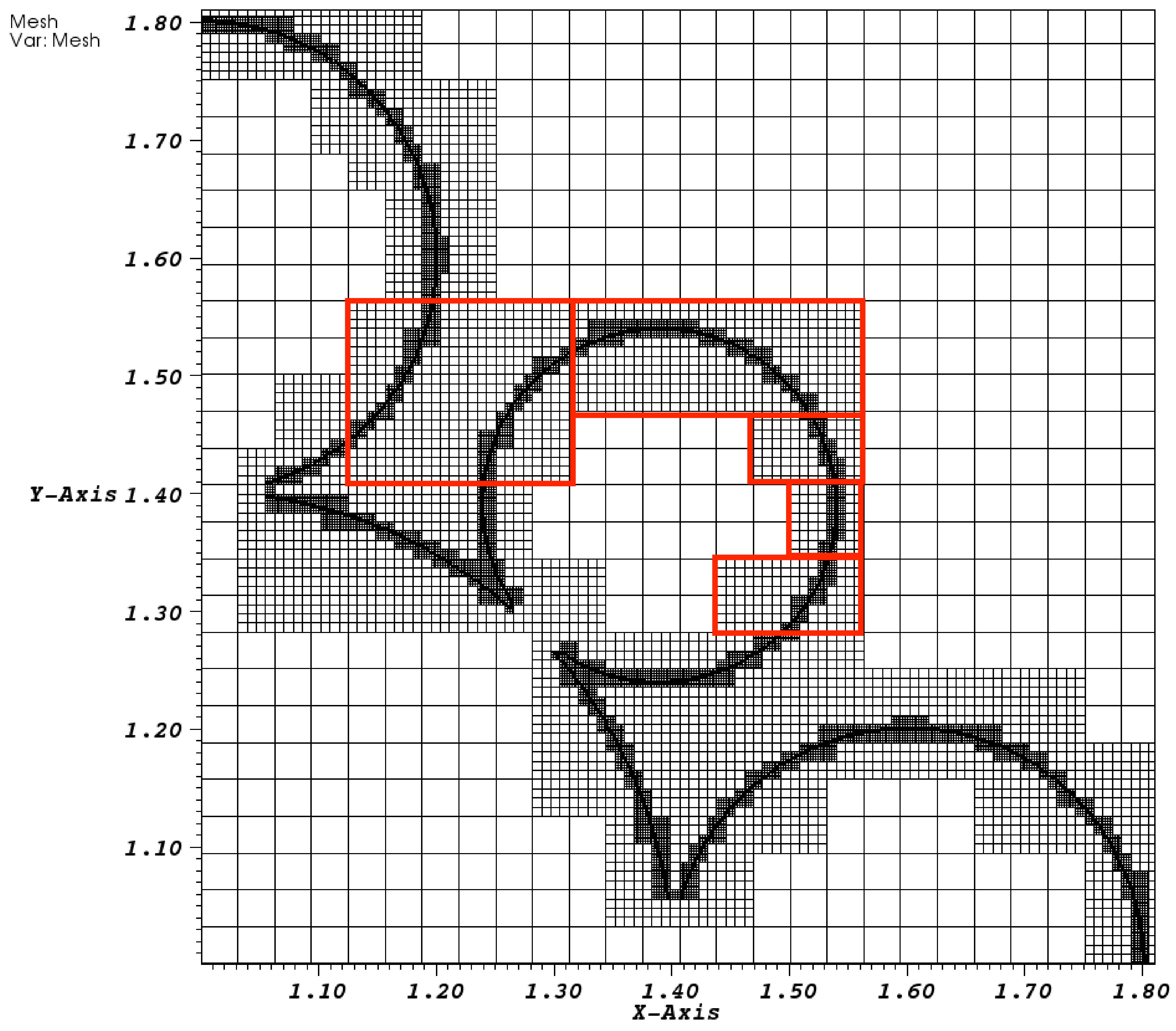




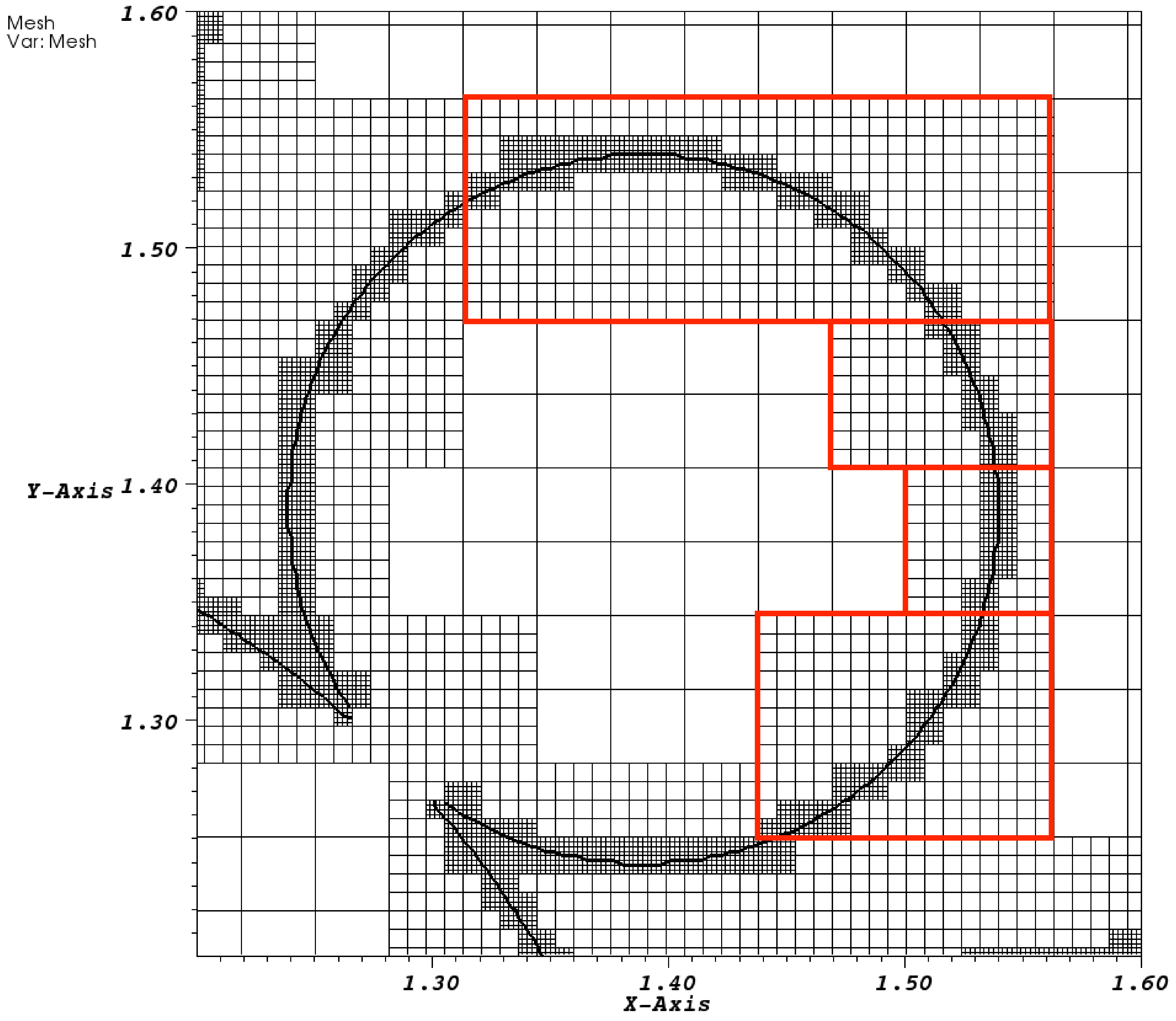
Flow by mean curvature



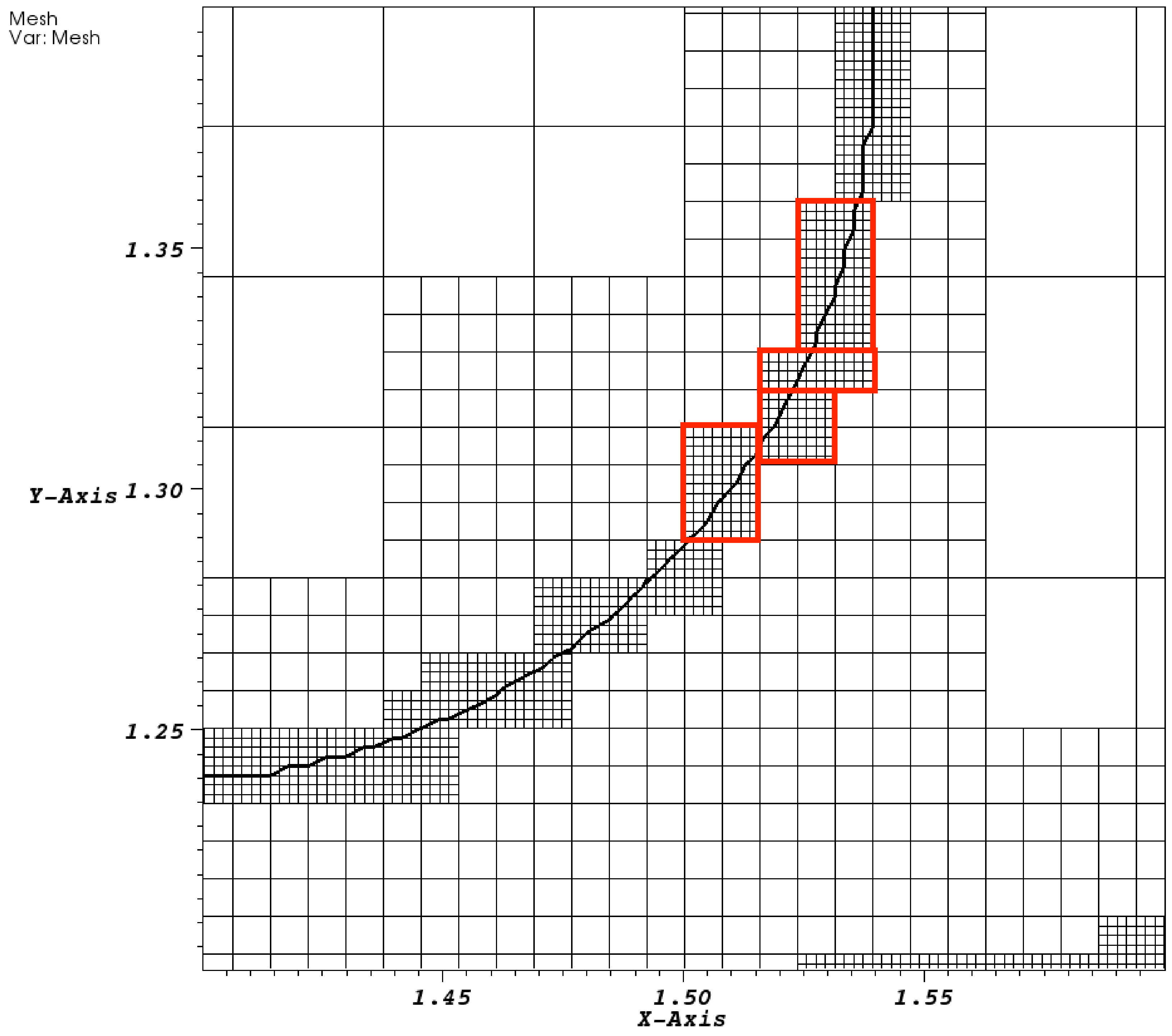
Flow by mean curvature



Flow by mean curvature



Flow by mean curvature



Flow by mean curvature

Single grid :

- Effective mesh resolution : 1024×1024
- Time step $\Delta t = 3.125 \times 10^{-5}$
- Total time steps : 7200, to final time $T = 0.225$.
- Time on CCRT Platine approximately 6.5 hours (1 proc)

With AMR :

- Coarse base grid : 64×64
- 3 levels of refinement with factor 4 refinement
- Subcycling in time: Coarse grid $\Delta t = 5 \times 10^{-4}$
- Time on Platine : 40 minutes on 1 processor, or about 15 minutes on 8 processors.

Factor of 10 speedup

Conservation

We consider a Cartesian mesh $[a_x, b_x]$ with cell centers given by

$$x_i = a_x + (i - 1/2)\, h, \qquad i = 1, 2, \ldots, M$$

with mesh width h. Cell edges given by

$$x_{i-1/2} = a_x + (i-1)\, h$$

We solve for the *average* value over a mesh cell

$$Q_i \approx \frac{1}{h} \int_{x_{i-1/2}}^{x_{i+1/2}} q(x)\, dx$$

For second order schemes, we can make use of the approximation

$$Q_i = q(x_i) + O(h^2)$$

What about conservation?

Hyperbolic case

$$q_t + \nabla \cdot F = 0$$

Integrating

$$\frac{d}{dt}\int_C q(\mathbf{x},t)\,dx = -\int_C \nabla \cdot F\,dA = -\int_{\partial C} F \cdot n\,dS$$

Discrete case

$$\begin{aligned}\sum_{i=1}^{M} Q_i^{n+1} &= \sum_{i=1}^{M} Q_i^n - \frac{\Delta t}{\Delta x}\sum_{i=1}^{M}\left(F_{i+1/2} - F_{i-1/2}\right)\\ &= \sum_{i=1}^{M} Q_i^n - \frac{\Delta t}{\Delta x}\left(F_{M+1/2} - F_{1/2}\right)\end{aligned}$$

What about conservation?

Because our PDE is in conservative form, given by

$$\nabla \cdot F = R$$

so we can apply the divergence theorem to get

$$\int_C \nabla \cdot F \; dV = \int_{\partial C} F \cdot n \; dS = \int_C R \; dV$$

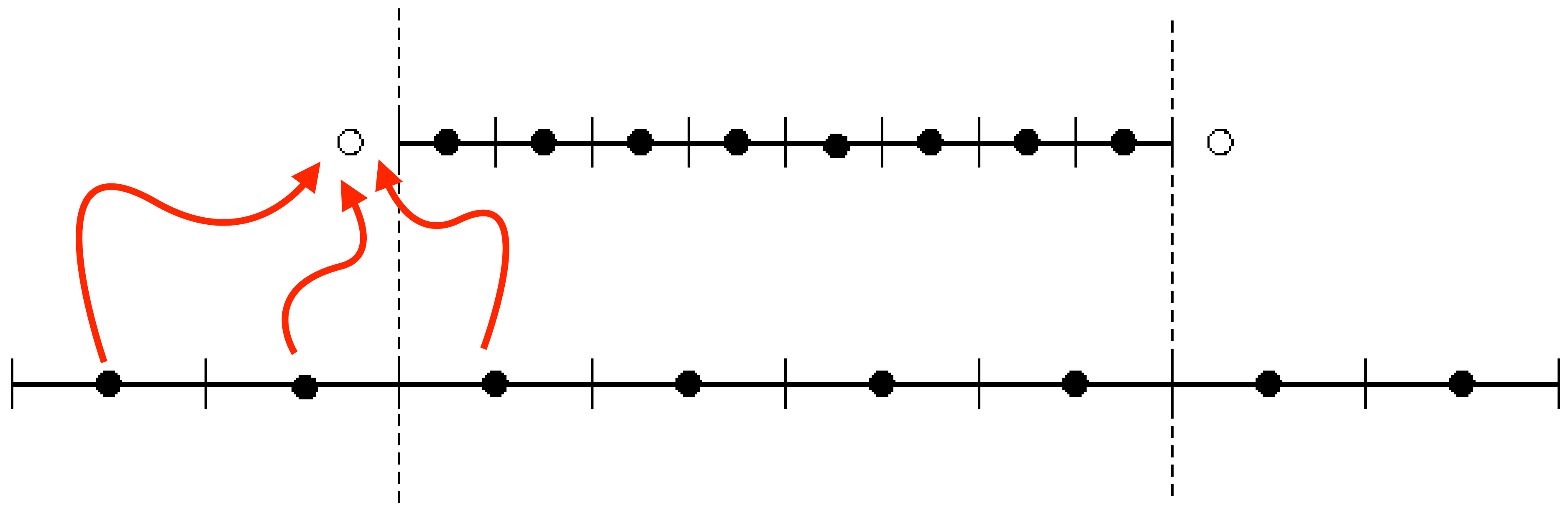
For our 1d elliptic problem, we wish to preserve :

$$\sum_i \left(\frac{F_{i+1/2} - F_{i-1/2}}{h} \right) \; h = \sum_i R_i \; h$$

or

$$F_{M+1/2} - F_{1/2} = h \; \sum_i R_i$$

Conservation at coarse/fine boundaries



On the coarse grid, we have

$$\frac{F_{i+1/2} - F_{i-1/2}}{h_c} = R_i, \qquad i = 1, 2, \ldots, M$$

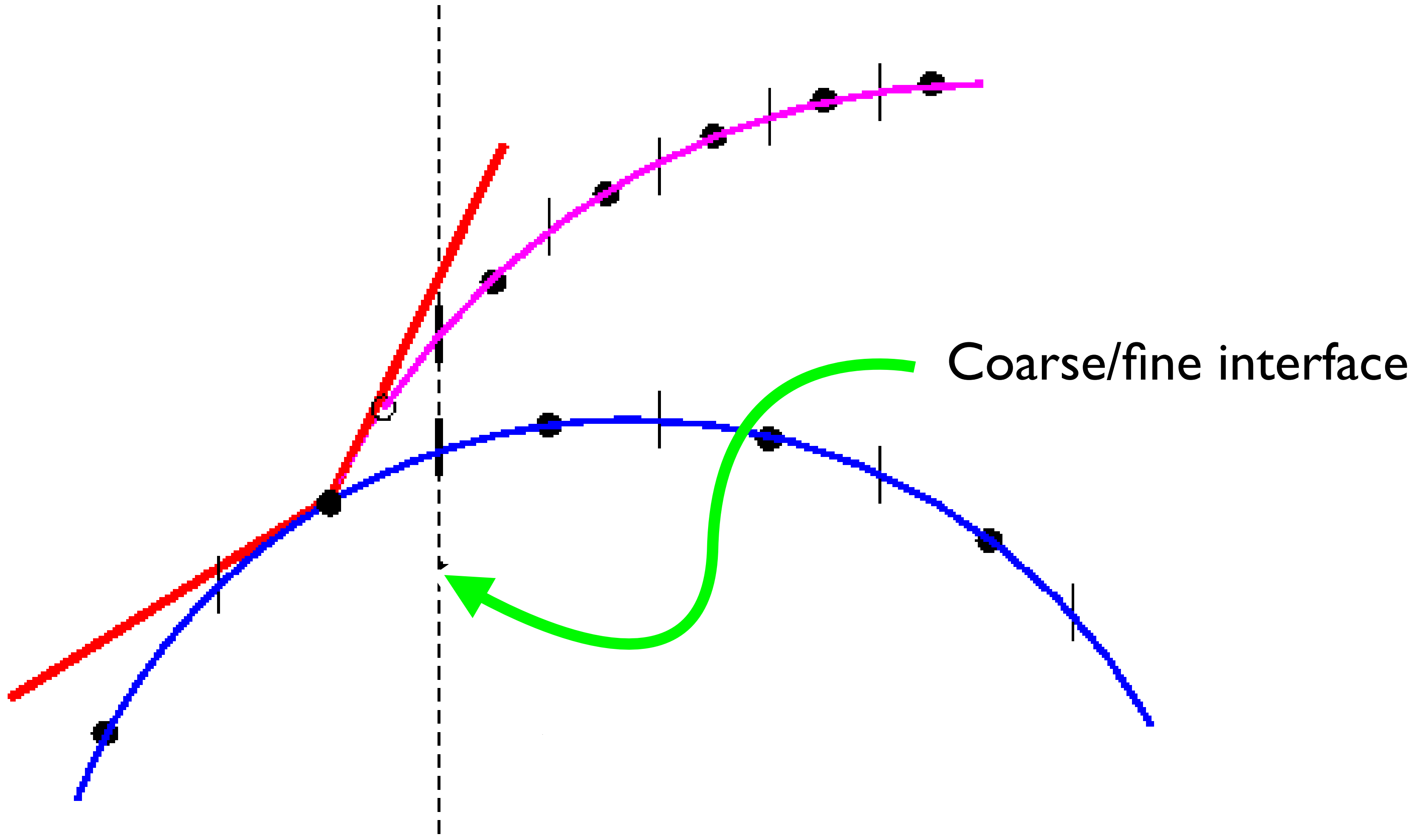
and at the fine interface, we have

$$\frac{f_{j+1/2} - f_{j-1/2}}{h_f} = r_j, \qquad j = 1, 2, \ldots, R_{ref}M/2$$

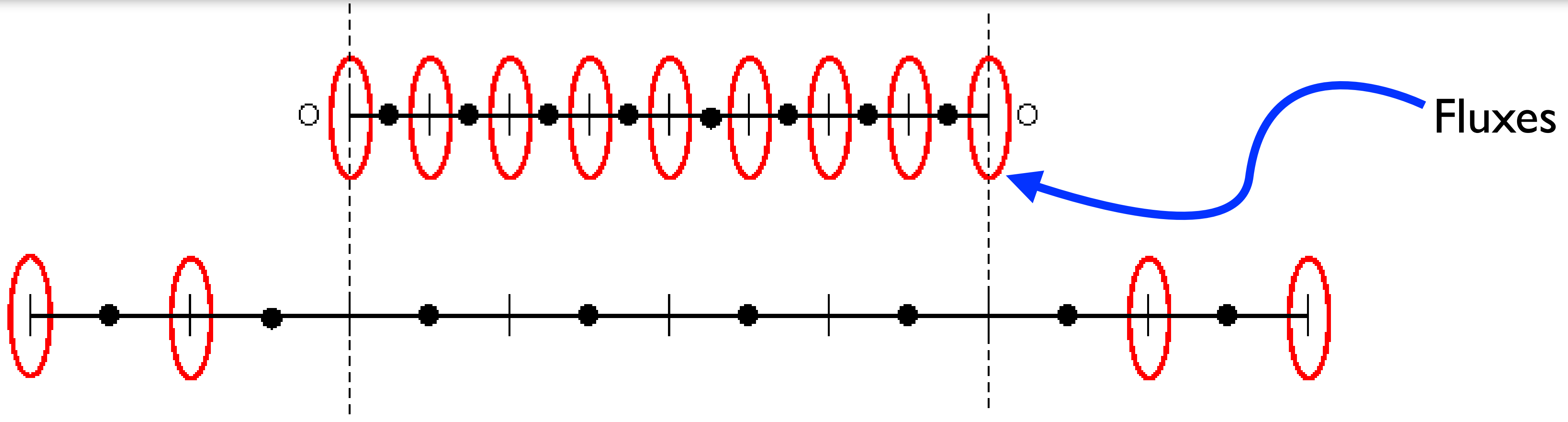
where $h_f = h_c/R_{ref}$.

Conservation

- Not differentiable at the coarse/fine interface
- Not conservative, since two different fluxes are used at the coarse/fine interface



Conservative fix-up - elliptic case



For conservation, we want :

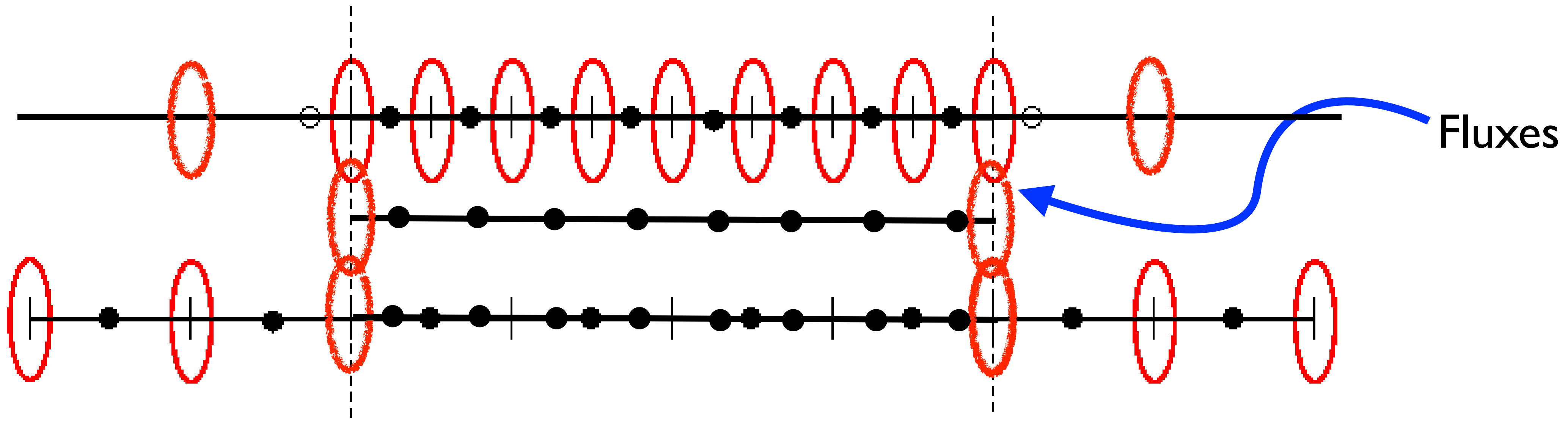
$$\frac{f_{1/2} - F_{i-1/2}}{h_c} = R_i$$

Replace coarse grid flux with fine grid flux.

$$\frac{F_{i+1/2} - F_{i-1/2}}{h_c} + \frac{f_{1/2} - F_{i+1/2}}{h_c} = R_i$$

$$\begin{aligned} \Rightarrow \quad \frac{F_{i+1/2} - F_{i-1/2}}{h_c} &= R_i - \frac{f_{i-1/2} - F_{i+1/2}}{h_c} \\ &= R_i - \delta_i \end{aligned}$$

Conservative fix-up - hyperbolic case



For conservation, we want

$$Q^{n+1} = Q^n - \frac{\Delta t}{\Delta x}\left(f^n_{1/2} - F^n_{i-1/2}\right)$$

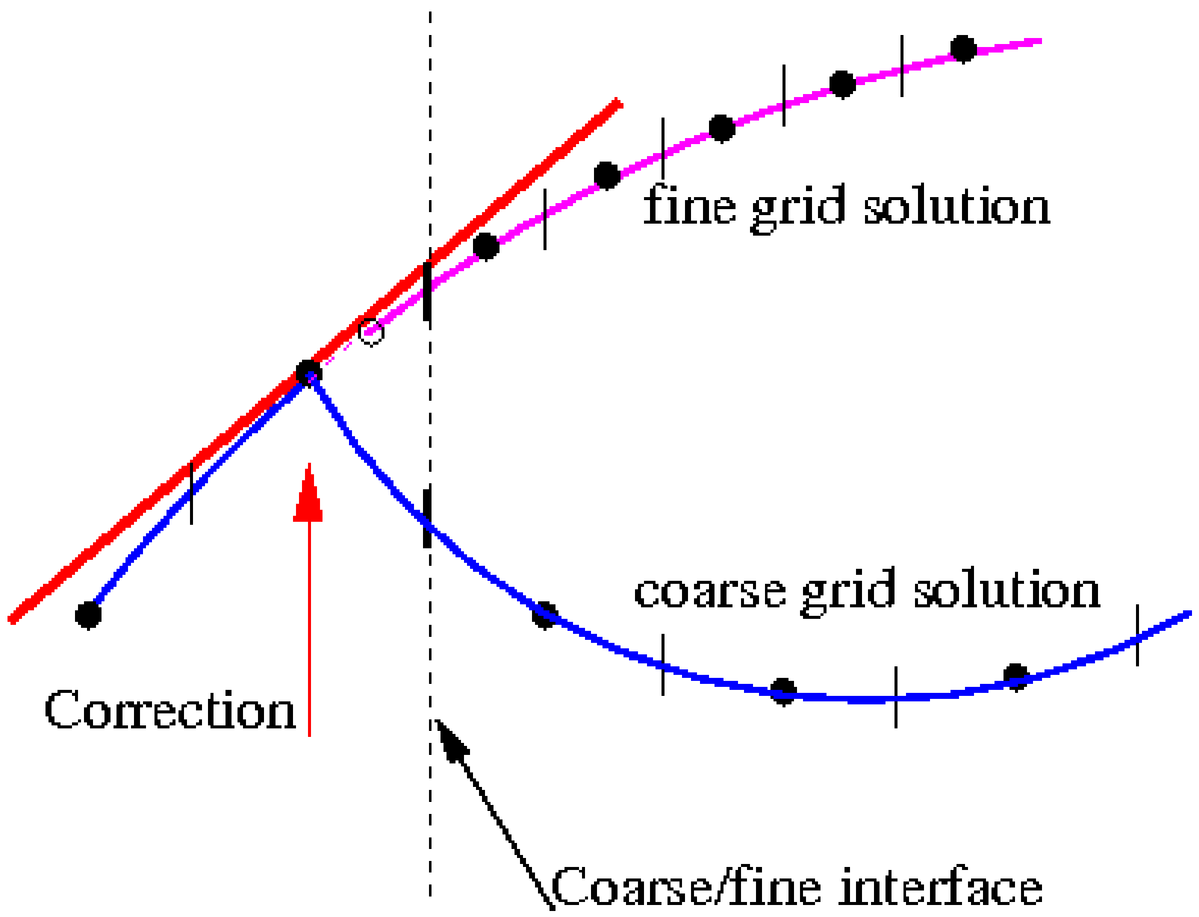
Add correction explicitly

correction term

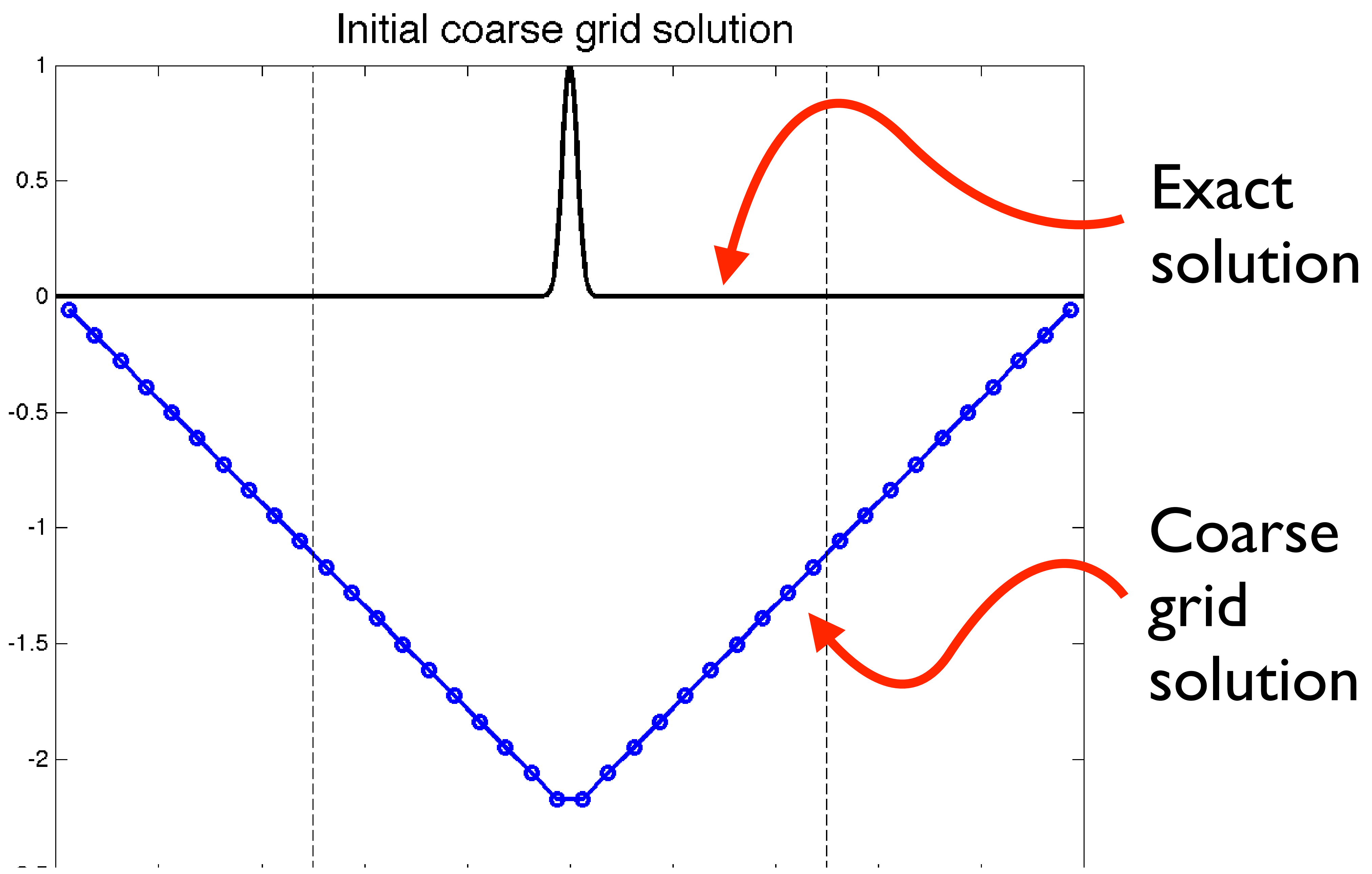
$$\begin{aligned} Q^{n+1} &= Q^n - \frac{\Delta t}{\Delta x}\left(F^n_{i+1/2} - F^n_{i-1/2}\right) - \frac{\Delta t}{\Delta x}\left(f^n_{1/2} - F^n_{i+1/2}\right) \\ &= Q^n - \frac{\Delta t}{\Delta x}\left(F^n_{i+1/2} - F^n_{i-1/2}\right) - \boxed{\Delta t \delta^n_i} \end{aligned}$$

Corrected solution

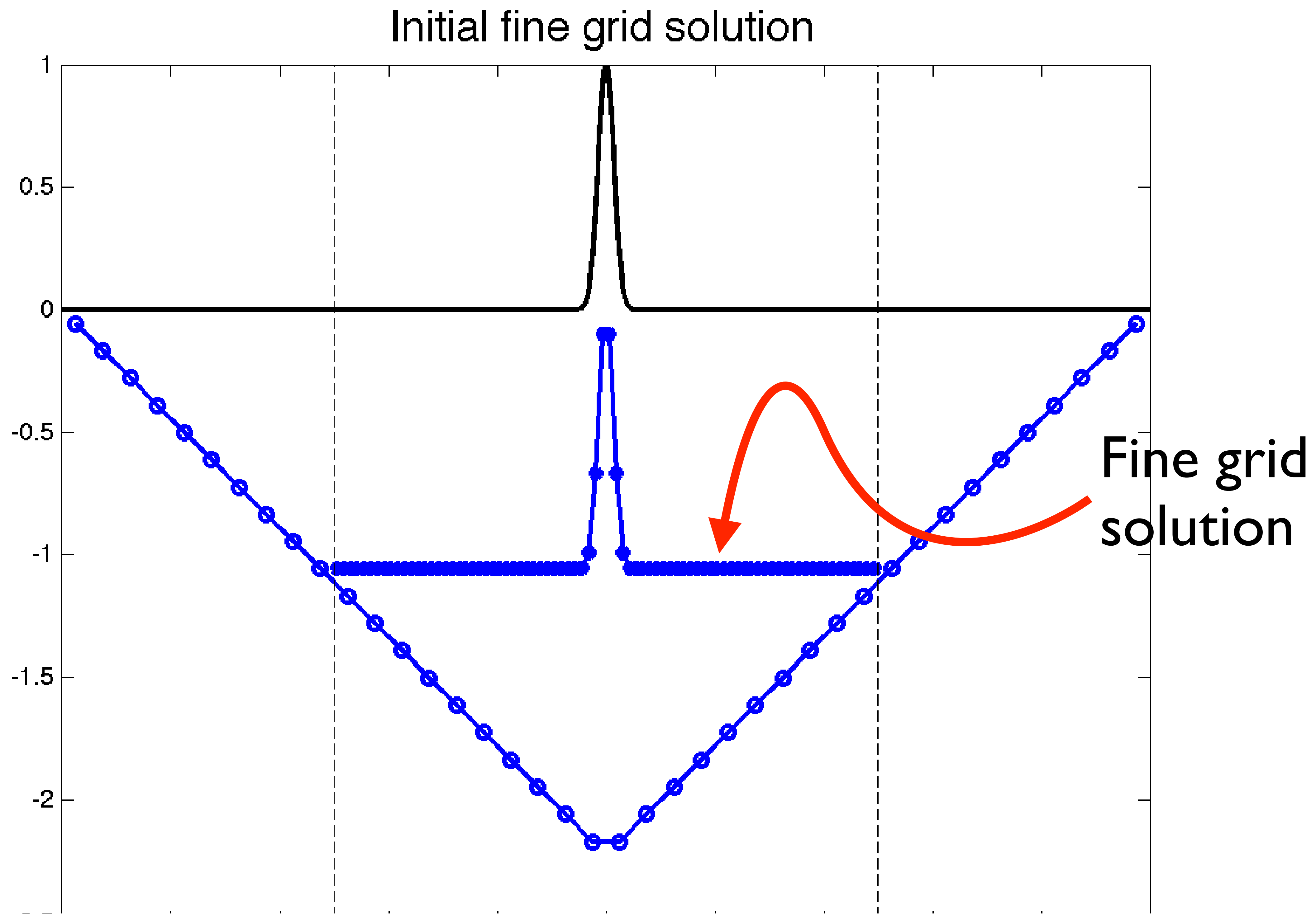
- Composite solution is smooth at coarse-fine interface (although coarse grid solution is not smooth at the coasre/fine interface).
- Final solution is conservative, since a single flux is used at the coarse fine interface.



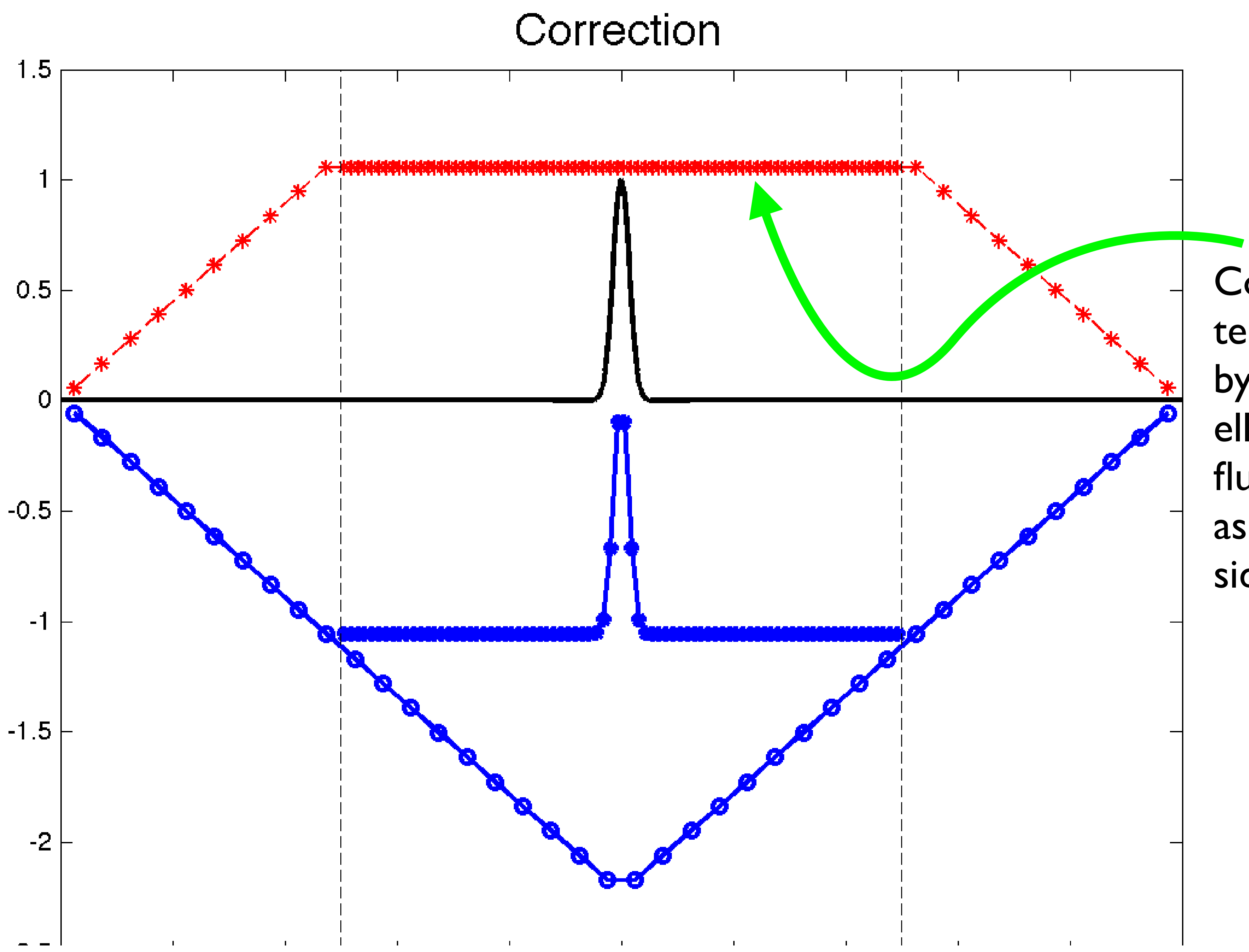
Elliptic solve on a coarse/fine grid



Example

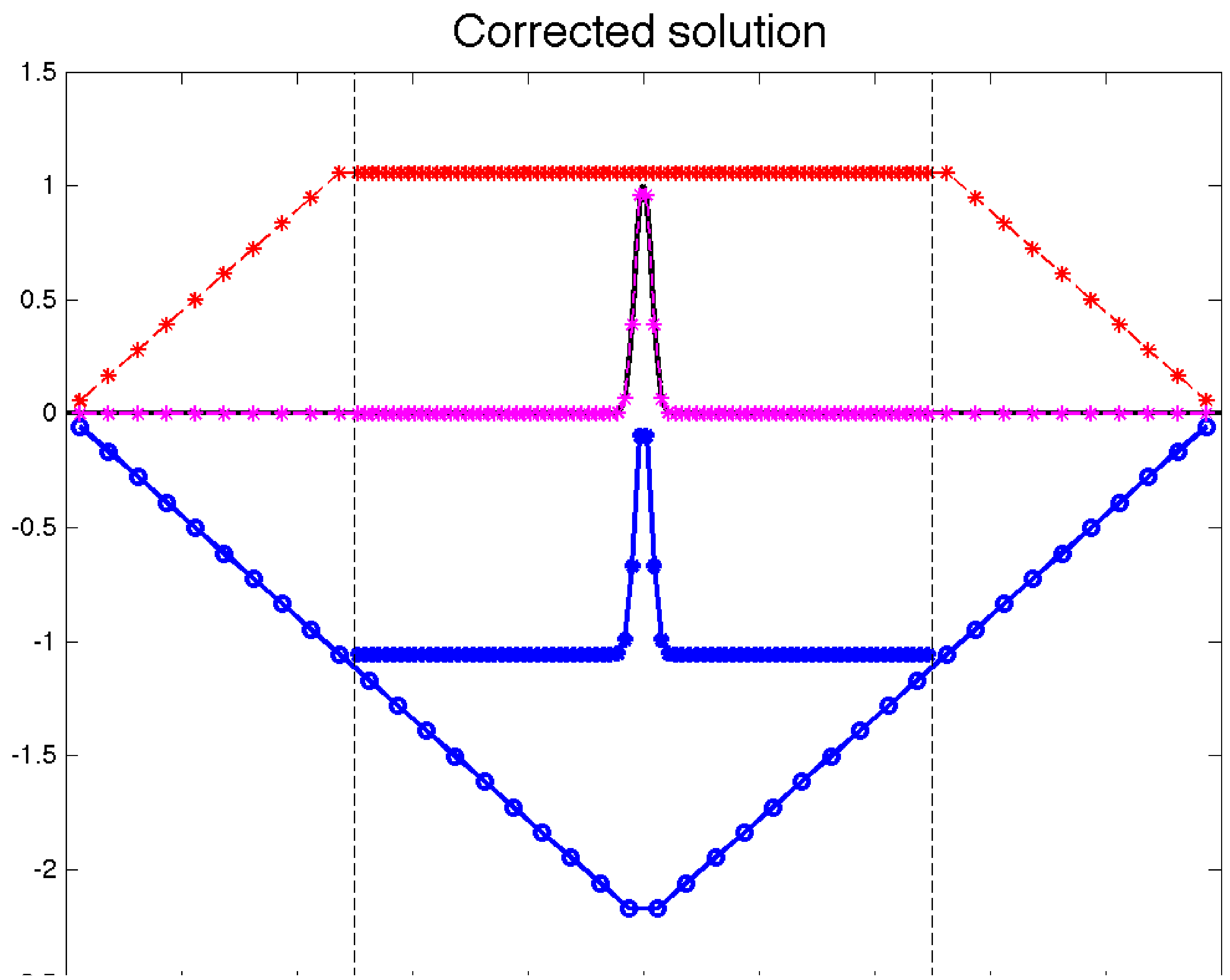


Example

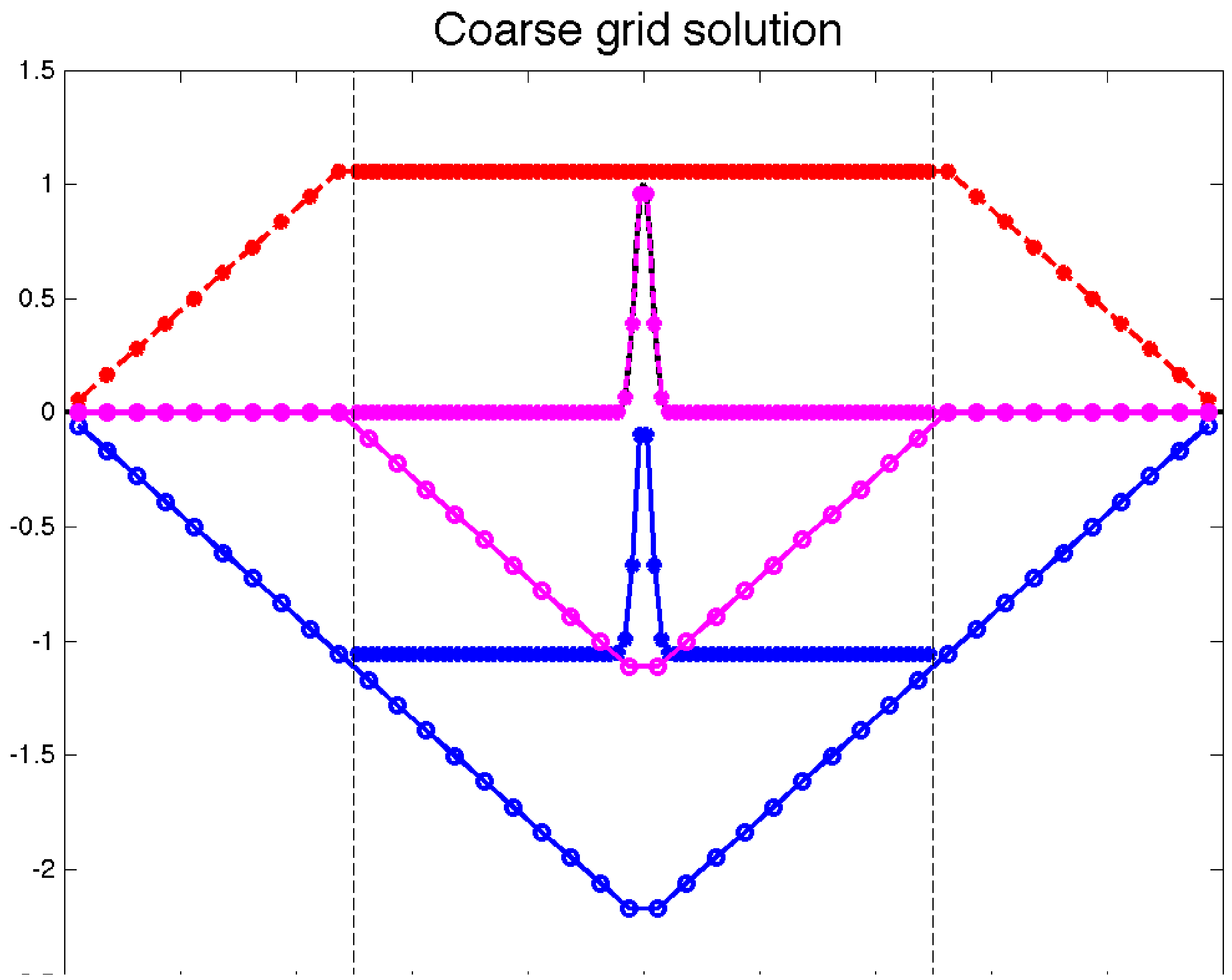


Correction term, obtained by solving elliptic PDE with flux differences as right hand side

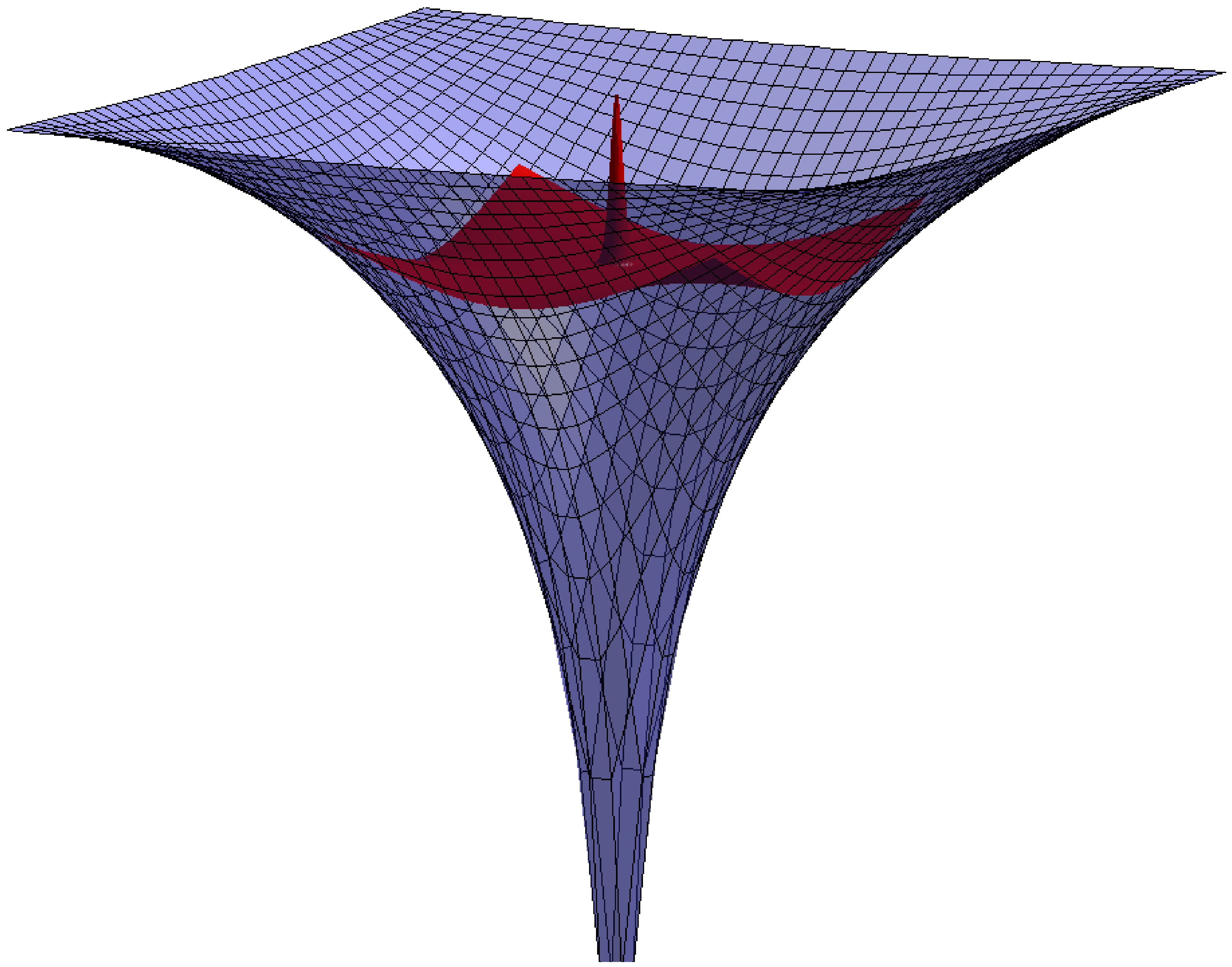
Example1



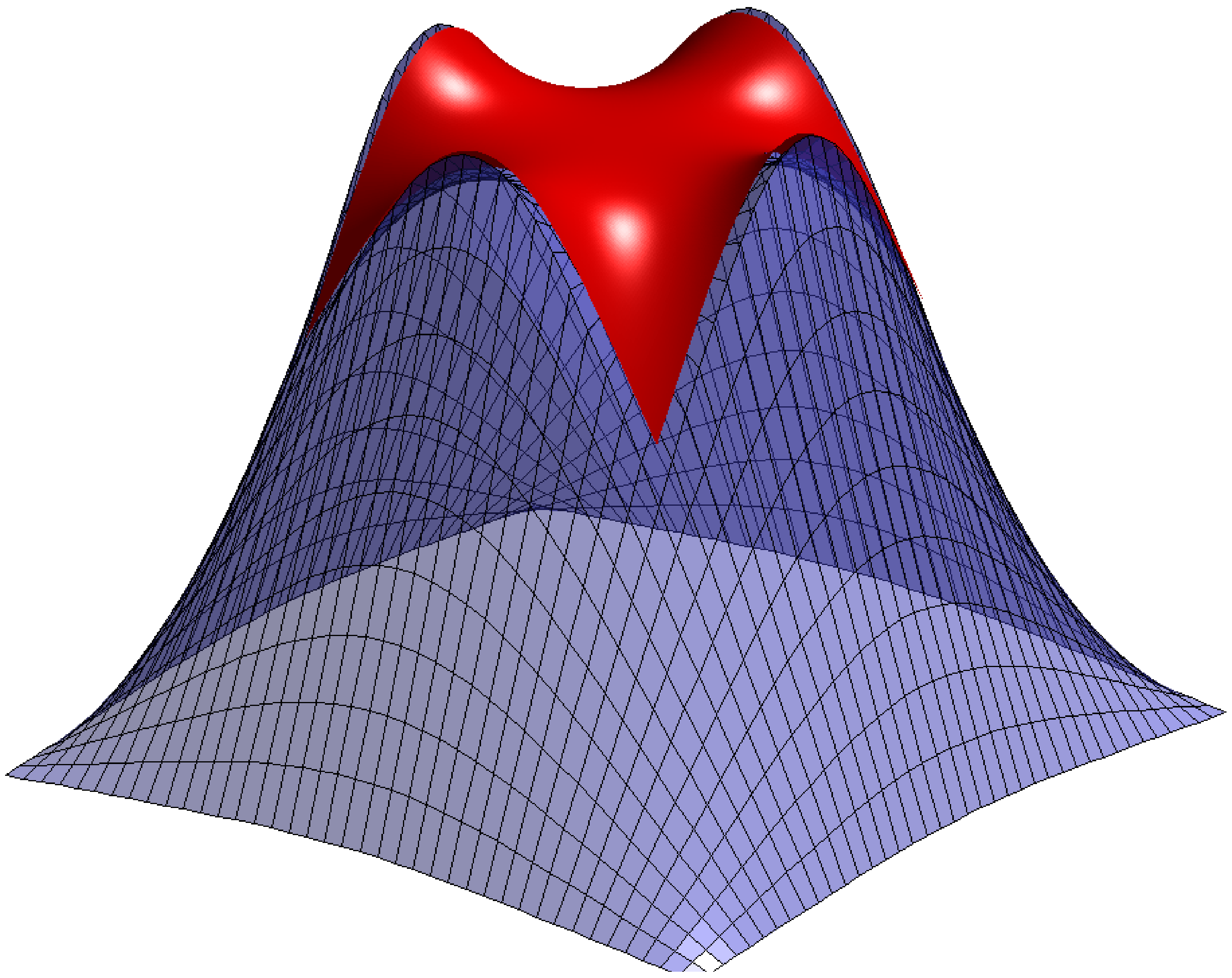
Example



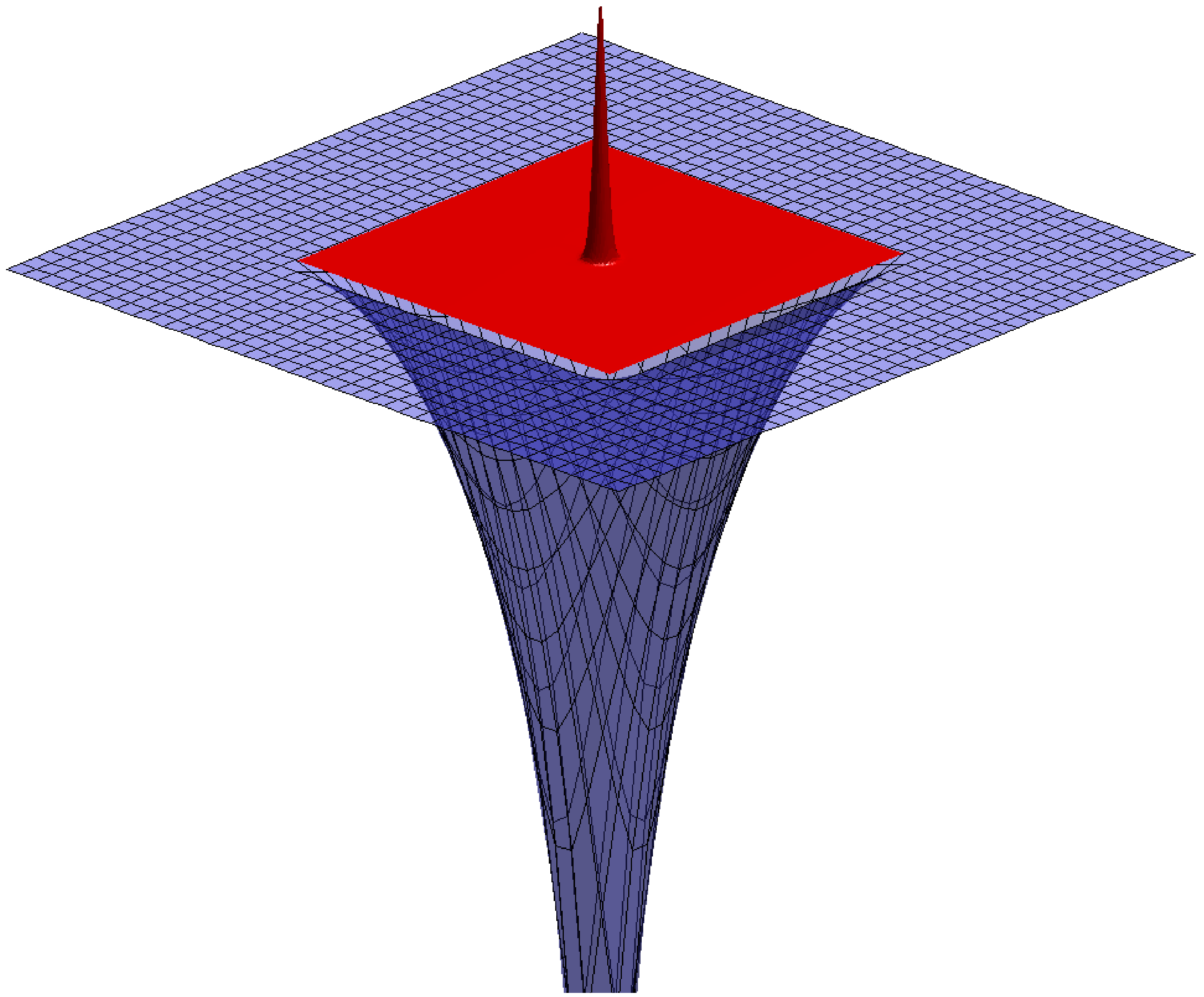
Example - 2d



Example - 2d



Example - 2d



References

For more details on the approach described above, including the multi-grid (Algorithm 2) see :

Ann S. Almgren, John B. Bell, Phillip Colella, Louis H. Howell, and Michael L. Welcome, ”A Conservative Adaptive Projection Method for the Variable Density Incompressible NavierStokes Equations”, J. Comput. Phys. (142), (1998) pp. 1-46.

and

D. Martin, P. Colella, and D. Graves, ”A cell-centered adaptive projection method for the incompressibe Navier-Stokes equations in three dimensions”, J. Comput. Phys. (227), (2008) pp. 1863-1886.

Projection Methods

$$\begin{aligned} \mathbf{u}_t + (\mathbf{u}\cdot \underset{\sim}{\nabla})\mathbf{u} &= -\underset{\sim}{\nabla} p + \mu\nabla^2\mathbf{u} \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}$$

Or in terms of a *projection operator* $\mathbb{P}$:

$$\begin{aligned} \mathbf{u}_t &= \mathbb{P}\left(-(\mathbf{u}\cdot \underset{\sim}{\nabla})\mathbf{u} + \mu\nabla^2\mathbf{u}\right) \\ \underset{\sim}{\nabla}\pi &= (\mathbb{I}-\mathbb{P})\left(-(\mathbf{u}\cdot \underset{\sim}{\nabla})\mathbf{u} + \mu\nabla^2\mathbf{u}\right) \end{aligned}$$

where $\mathbb{P}$ is defined as

$$\begin{aligned} \mathbb{P}\mathbf{u}^* &= \mathbf{u}^* - \underset{\sim}{\nabla}\pi \\ \nabla^2\pi &= \nabla\cdot\mathbf{u}^* \end{aligned}$$

- Efficient elliptic and parabolic solvers are essential
- Need to avoid artifical compression or expansion of the fluid at non-conforming mesh cell interfaces

Existing software for AMR

- AMRClaw (M. J. Berger, and R. J. LeVeque)
- Chombo (ANAG, LBL, P. Collela)
- p4est (Ghattas, Burstedde, ...)
- Peano (Weinzierl, ...)
- PyClaw (work in progress!)

Link to Github PyClaw Wiki for a more complete list