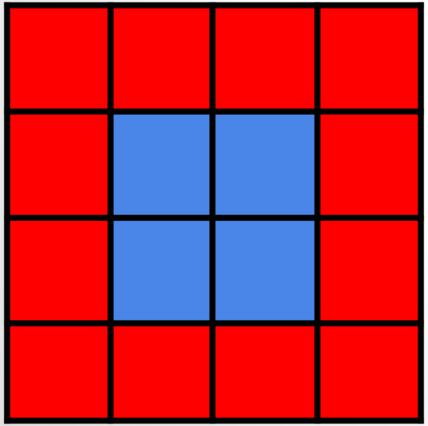
Boundary-Value Problems

Hsi-Yu Schive (薛熙于)
National Taiwan University

Poisson Equation

- $\boxed{\nabla^2 \phi(\boldsymbol{r}) = \rho(\boldsymbol{r})}$
  - ρ: mass density, Φ: gravitational potential, assuming $4\pi G = 1$
- Task: given ρ in V and Φ at ∂V, where V is the computational domain of interest and ∂V is the boundary → solve Φ in V



🟥 Given Φ

🟦 Given ρ, solveΦ

Relaxation Methods

- $L\phi = \rho \rightarrow \boxed{\frac{\partial \phi}{\partial t} = L\phi - \rho}$
    - L: elliptic operator
    - Let the system relax until equilibrium is established

      $\frac{\partial \phi}{\partial t} = 0 \rightarrow L\phi = \rho$ (solution of the original elliptic equation)
- Initial guess of Φ within V must be provided as well
- We only care about the final (relaxed) solution
    - Intermediate (unrelaxed) solution is allowed to have large errors
    - Relaxation can be very time-consuming
    - Key concerns of various numerical schemes:
        - Speed up the relaxation process
        - It’s fine to introduce larger errors in the intermediate results

Relaxation Methods for Poisson Eq.

- $\nabla^2 \phi = \rho \rightarrow \dfrac{\partial \phi}{\partial t} = \nabla^2 \phi - \rho$
  - **2D discrete form:** $\dfrac{\partial \phi}{\partial t} = \dfrac{\partial^2 \phi}{\partial x^2} + \dfrac{\partial^2 \phi}{\partial y^2} - \rho$
  - **FTCS scheme (refer to the lecture note of “Initial-Value Problems”) assuming** $\Delta x = \Delta y = \Delta$**:**

$$\frac{\phi_{i,j}^{n+1} - \phi_{i,j}^{n}}{\Delta t} = \frac{1}{\Delta^2}\left(\phi_{i+1,j}^{n} + \phi_{i-1,j}^{n} + \phi_{i,j+1}^{n} + \phi_{i,j-1}^{n} - 4\phi_{i,j}^{n}\right) - \rho_{i,j}$$

  - **CFL stability:** $\Delta t \le \Delta^2/4 \rightarrow$ **let** $\Delta t = \Delta^2/4$

$$\boxed{\phi_{i,j}^{n+1} = \frac{1}{4}\left(\phi_{i+1,j}^{n} + \phi_{i-1,j}^{n} + \phi_{i,j+1}^{n} + \phi_{i,j-1}^{n} - \Delta^2 \rho_{i,j}\right)}$$

Jacobi’s method

→ **iterate until relaxed (convergence)**

Gauss-Seidel Method

- **Jacobi's method converges very slowly**
  - **Number of iterations** $\propto N^2$ **on a** NxN **grid**
  - **Recall that the characteristic diffusion time scale** $\propto \Delta^2$
- **Gauss-Seidel method**
  - **In place update**

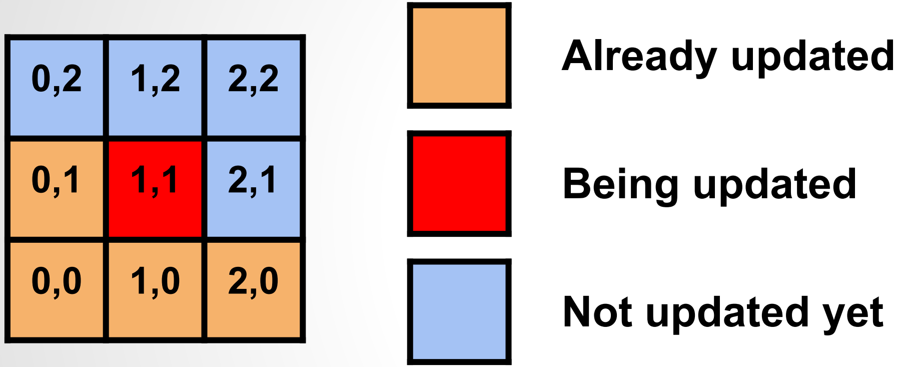
$$\phi_{i,j}^{n+1} = \frac{1}{4}\left(\underline{\phi_{i+1,j}^{n/n+1} + \phi_{i-1,j}^{n/n+1} + \phi_{i,j+1}^{n/n+1} + \phi_{i,j-1}^{n/n+1}} - \Delta^2 \rho_{i,j}\right)$$

use the updated values (n+1) instead of the original values (n) whenever available

  - **Converge faster than the Jacobi's method**
    - **But the number of iterations still scales as** N^2

Gauss-Seidel Method

- **Example of updating procedure**



0,2	1,2	2,2
0,1	1,1	2,1
0,0	1,0	2,0

🟧 **Already updated**

🟥 **Being updated**

🟦 **Not updated yet**

$$\phi_{1,1}^{n+1} = \frac{1}{4}\left(\phi_{2,1}^{n} + \phi_{0,1}^{n+1} + \phi_{1,2}^{n} + \phi_{1,0}^{n+1} - \Delta^2 \rho_{1,1}\right)$$

Successive Overrelaxation Method

- Rewrite the Gauss-Seidel method by defining a *correction* term

$$\begin{aligned}
\phi_{i,j}^{n+1} &= \frac{1}{4}\left(\phi_{i+1,j}^{n/n+1} + \phi_{i-1,j}^{n/n+1} + \phi_{i,j+1}^{n/n+1} + \phi_{i,j-1}^{n/n+1} - \Delta^2 \rho_{i,j}\right) \\
&= \phi_{i,j}^{n} + \frac{1}{4}\underbrace{\left(\phi_{i+1,j}^{n/n+1} + \phi_{i-1,j}^{n/n+1} + \phi_{i,j+1}^{n/n+1} + \phi_{i,j-1}^{n/n+1} - 4\phi_{i,j}^{n} - \Delta^2 \rho_{i,j}\right)}_{\xi_{i,j}\ \text{(residual)}} \\
&\equiv \phi_{i,j}^{n} + \underbrace{\frac{\xi_{i,j}}{4}}_{\text{correction}}
\end{aligned}$$

- SOR method: $\boxed{\phi_{i,j}^{n+1} = \phi_{i,j}^{n} + w\frac{\xi_{i,j}}{4}}$
  - $1 < w < 2$: *overrelaxation* parameter (as if CFL>1)
  - Number of iterations $\propto N$ instead of N^2 on a NxN grid
    - Much better than the Jacobi's and Gauss-Seidel methods

Successive Overrelaxation Method

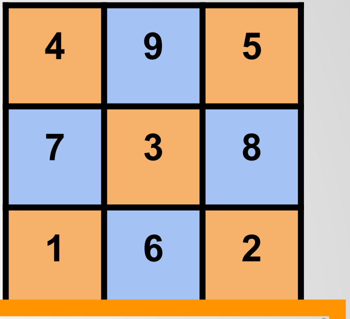
- **Determining an optimum w is non-trivial**
  - **Depend on the elliptic operator**
  - **Depend on the grid size N**
  - **Determine it empirically in advance if the equation to be solved is fixed**

- **How to determine whether the solution has converged?**
  - **Numerical solution will never reach *exact* equilibrium due to floating-point round-off errors**
    - **Single precision: ~10^{-6} – 10^{-8}**
    - **Double precision: ~10^{-15} – 10^{-17}**
  - **Use the norm of the residual $\xi_{i,j}$**

$$\text{Error} \equiv \frac{1}{N^2} \sum_{i,j} \left| \frac{\xi_{i,j}}{\phi_{i,j}} \right|$$

    - **Must express error in a *dimensionless* form**
    - **Terminate the iteration when errores reach the machine precision**

Successive Overrelaxation Method



4	9	5
7	3	8
1	6	2

- Odd-even ordering
  - SOR scheme does not specify the order in which different cells are updated
  - We can divide all cells into odd and even cells

$$\boxed{\underline{\phi^{n+1}_{i,j}} = \underline{\phi^{n}_{i,j}} + \frac{w}{4}\left(\underline{\phi^{n/n+1}_{i+1,j} + \phi^{n/n+1}_{i-1,j} + \phi^{n/n+1}_{i,j+1} + \phi^{n/n+1}_{i,j-1}} - \underline{4\phi^{n}_{i,j}} - \Delta^2 \rho_{i,j}\right)}$$

  - Odd cells depend only on even cells, and vice versa
  - So all odd (even) cells can be updated at once using the old values of the even (odd) cells
  - Enable fine-grained parallelism (e.g., GPU threads)

Multigrid (MG) Method

- SOR method is still not very efficient for large problems
  - Low-k modes (i.e., smooth parts) take long time to relax
  - High-k modes (i.e., non-smooth parts) however relax quickly

- Multigrid method
  - Use coarse grids to solve low-k modes
    - So low-k modes can relax faster (because of fewer cells)
    - But it cannot capture the high-k components
  - Use fine grids to solve high-k modes
    - Correct the high-k solution not captured by the coarse grids
    - Number of iterations can be reduced significantly since low-k errors have been reduced substantially in the coarse-grid solution
  - Communication between coarse and fine grids: restriction and prolongation

Residual and Correction

- **Target discretized linear equation:** $L_h \phi_h = \rho_h$
  - **Subscript** h **denotes the cell size** Δh
  - **Here** ϕ_h **is an <u>exact</u> solution to this discretized eq.**
  - **Let** $\tilde{\phi}_h$ **be an <u>approximate</u> solution to this eq.**

➡ $\begin{cases} \text{residual } \xi_h = L_h \tilde{\phi}_h - \rho_h \\ \text{correction } \phi_h^{corr} = \phi_h - \tilde{\phi}_h \end{cases}$ $\boxed{L_h \phi_h^{corr} = -\xi_h}$

For example, $\tilde{L}_h = -(4/\Delta^2)I$ **when solving the Poisson eq. with the Jacobi's method (*try to derive it!*)**

- **Relaxation method procedure**
  1. **Given** $\tilde{\phi}_h^{old}$**, compute** ξ_h
  2. **Approximate** L_h **with a simpler operator** $\tilde{L}_h$
  3. **Get an approximate correction** $\tilde{\phi}_h^{corr}$ **by solving** $\tilde{L}_h \tilde{\phi}_h^{corr} = -\xi_h$
  4. **Update the approximate solution** $\tilde{\phi}_h^{new} = \tilde{\phi}_h^{old} + \tilde{\phi}_h^{corr}$
  5. **Repeat steps 1–4 by taking** $\tilde{\phi}_h^{old} = \tilde{\phi}_h^{new}$ **until convergence**

Supplement

- **Jacobi's method:** $\tilde{L}_h = -(4/\Delta^2)I$ **?**

$$\tilde{\phi}^{n+1}_{i,j} = \frac{1}{4}\left(\tilde{\phi}^{n}_{i+1,j} + \tilde{\phi}^{n}_{i-1,j} + \tilde{\phi}^{n}_{i,j+1} + \tilde{\phi}^{n}_{i,j-1} - \Delta^2 \rho_{i,j}\right)$$

$$\underbrace{\tilde{\phi}^{n+1}_{i,j} - \tilde{\phi}^{n}_{i,j}}_{\tilde{\phi}^{corr}_{i,j}} = \frac{1}{4}\left(\tilde{\phi}^{n}_{i+1,j} + \tilde{\phi}^{n}_{i-1,j} + \tilde{\phi}^{n}_{i,j+1} + \tilde{\phi}^{n}_{i,j-1} - 4\tilde{\phi}^{n}_{i,j} - \Delta^2 \rho_{i,j}\right)$$

$$= \frac{\Delta^2}{4}\underbrace{\left(L\tilde{\phi}^{n}_{i,j} - \rho_{i,j}\right)}_{\xi_{i,j}\ \textbf{(residual)}}$$

$$\Updownarrow$$

$$\tilde{L}_h \tilde{\phi}^{corr}_{i,j} = -\xi_{i,j}$$

Coarse-Grid Correction

- **MG method adopts** $\tilde{L}_h = L_{2h}$

$$\begin{aligned}
\xi_{2h} &= R\,\xi_h \\
L_{2h}\tilde{\phi}_{2h}^{corr} &= -\xi_{2h} \rightarrow \text{ obtain } \tilde{\phi}_{2h}^{corr} \\
\tilde{\phi}_h^{corr} &= P\,\tilde{\phi}_{2h}^{corr} \\
\tilde{\phi}_h^{new} &= \tilde{\phi}_h^{old} + \tilde{\phi}_h^{corr}
\end{aligned}$$

An exact solution is obtained here (but will be generalized later in “From Two-Grid to Multigrid”)

  - R **: restriction (averaging) operator**
  - P **: prolongation (interpolation) operator**

- **This procedure does NOT resolve the high-k (**$\lambda \sim 2\Delta h$**) components**
  - **Use the original fine grid for that**

Two-Grid Scheme

- Apply a smoothing operator (S) before and after the coarse-grid correction for updating the high-k modes
  - Standard smoothing operator: applying the Gauss-Seidel relaxation with the odd-even ordering for a few (n) iterations

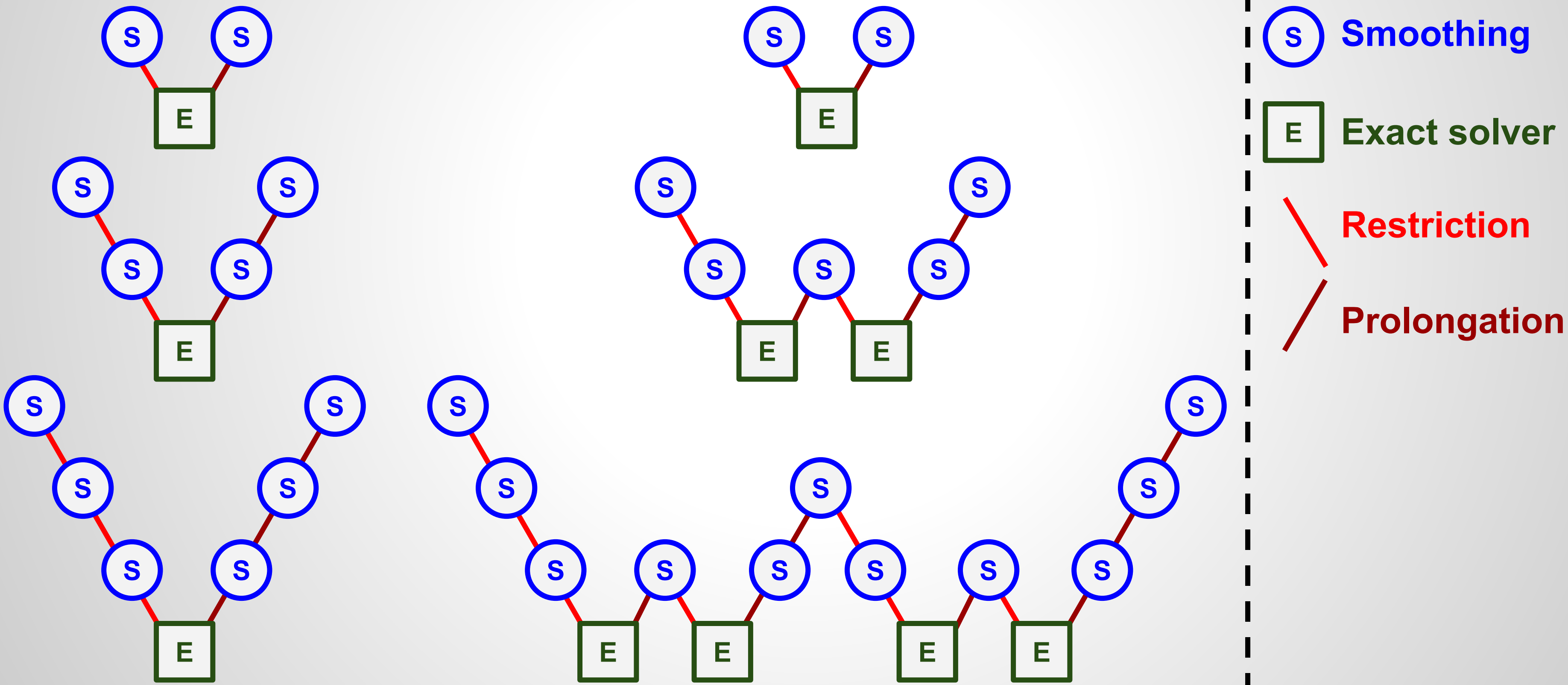
- Procedure summary

1. Provide an initial guess of $\tilde{\phi}_h^{old}$
2. Pre-smoothing: $\tilde{\phi}_h^{old,smooth} = S\tilde{\phi}_h^{old}$
3. Compute the residual $\xi_h = L_h\tilde{\phi}_h^{old,smooth} - \rho_h$
4. Apply the coarse-grid correction on ξ_h to obtain $\tilde{\phi}_h^{new}$
5. Post-smoothing: $\tilde{\phi}_h^{new,smooth} = S\tilde{\phi}_h^{new}$
6. Repeat steps 2–5 by taking $\tilde{\phi}_h^{old} = \tilde{\phi}_h^{new,smooth}$, where $\tilde{\phi}_h^{new,smooth}$ is taken from the previous step 5, until convergence

From Two-Grid to Multigrid

- **When solving $L_{2h}\tilde{\phi}_{2h}^{corr} = -\xi_{2h}$ in the coarse-grid correction, instead of obtaining an exact solution of $\tilde{\phi}_{2h}^{corr}$, we can**
  - **Introduce an even coarser grid (i.e., cell spacing of $4\Delta h$)**
  - **Initialize $\tilde{\phi}_{4h}^{corr}$ as zero**
  - **Apply n number of two-grid iterations to obtain an approximate solution of $\tilde{\phi}_{4h}^{corr}$**
  - **Interpolate (prolongate) $\tilde{\phi}_{4h}^{corr}$ to get an approximate solution of $\tilde{\phi}_{2h}^{corr}$**
- **MG method: apply this procedure recursively**
  - **Stop creating coarser grids when an exact solution can be obtained easily**
  - **Use an exact solver on the coarsest grid**
  - **Use the norm of residual as the convergence check**
    - **Same as the relaxation method**

V-cycle and W-cycle

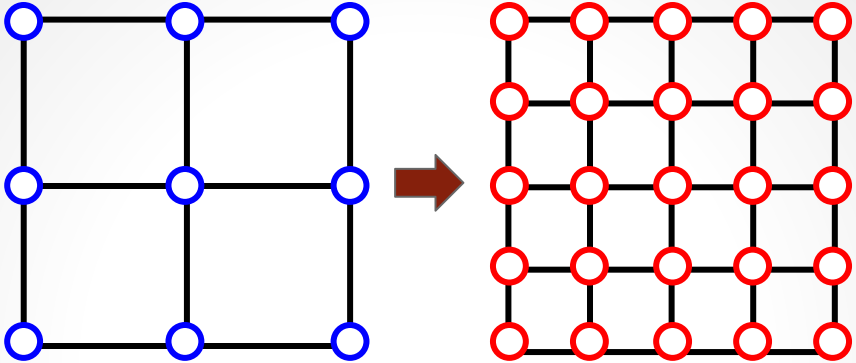


V-cycle (n=1)

W-cycle (n=2)

Prolongation

- Interpolation

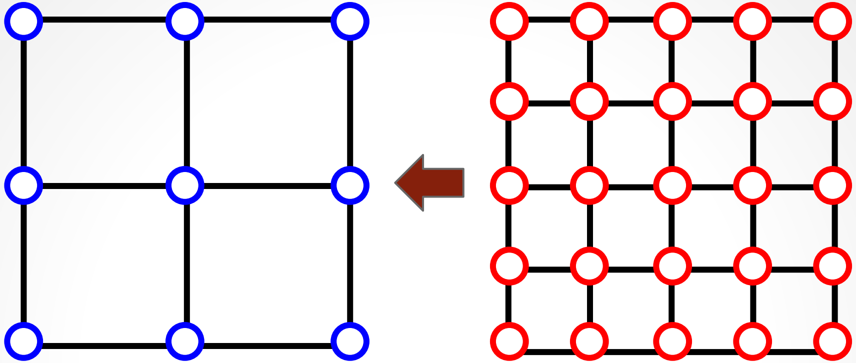


- Be aware that here values are defined at grid intersections
  - In comparison, finite-volume schemes adopt volume-averaged quantities, which are approximately cell-centered
- Bilinear interpolation operator:

$$P = \begin{bmatrix} 1/4 & 1/2 & 1/4 \\ 1/2 & 1 & 1/2 \\ 1/4 & 1/2 & 1/4 \end{bmatrix}$$

Restriction

- **Averaging**



- **Restriction operator in 2D:** $R = \dfrac{1}{4}P^T$
  - **For the bilinear interpolation:**

$$R = \begin{bmatrix} 1/16 & 1/8 & 1/16 \\ 1/8 & 1/4 & 1/8 \\ 1/16 & 1/8 & 1/16 \end{bmatrix}$$

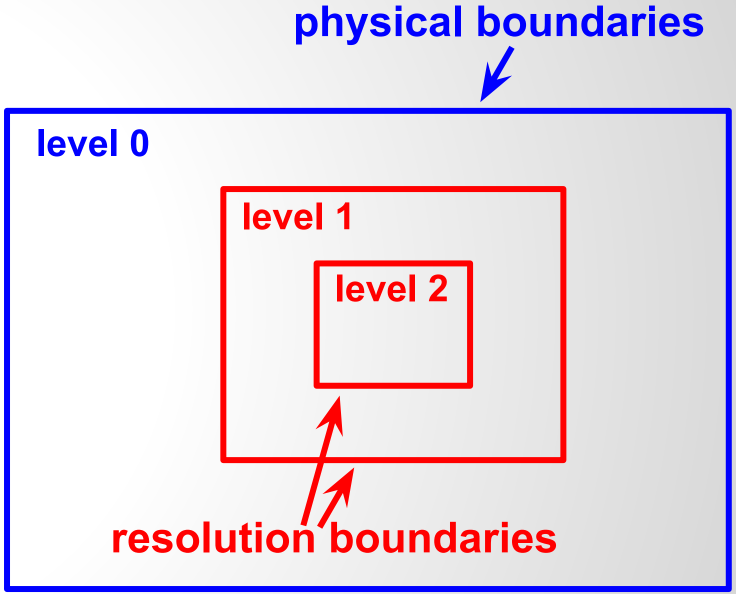
Solving Poisson in AMR

1. Root level 0: invoke Poisson solver with the physical boundary condition (e.g., isolated, periodic)

2. Refinement level 1
   a. Calculate potential on the resolution boundaries by interpolating the coarse-grid potential (also a good initial guess for the fine-grid potential)
   b. Invoke Poisson solver with the Dirichlet boundary condition just obtained

3. Repeat step 2 for all refinement levels

- Caution: ensure mass density on different levels are consistent with each other
  - Restriction operation

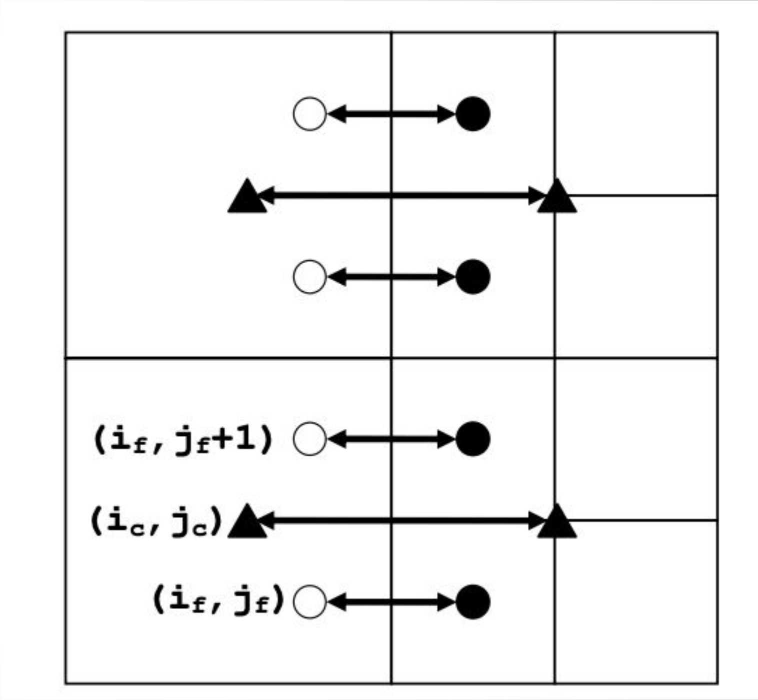


Solving Poisson in AMR: Issue 1

- **Errors on the coarse-fine boundaries**
  - **Due to unsmooth potential across these boundaries**
  - **Discontinuity in normal derivative of potential acts like *pseudo mass sheets* (ζ) on the coarse-fine interfaces**

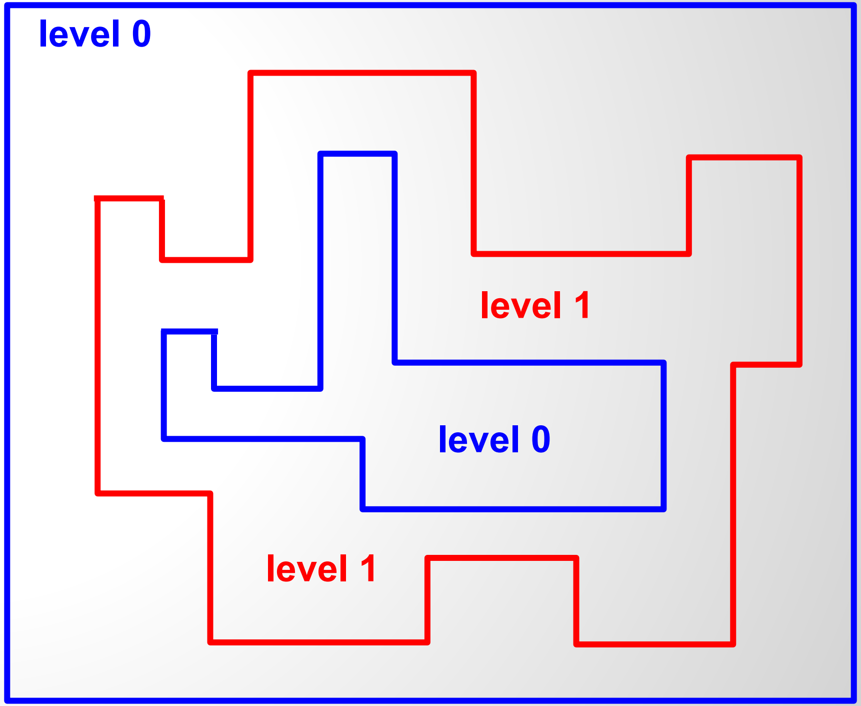
$$\zeta \leftrightarrow \frac{\partial}{\partial n}\left(\left.\frac{\partial \phi}{\partial n}\right|_{\epsilon^+} - \left.\frac{\partial \phi}{\partial n}\right|_{\epsilon^-} \right)$$

  - **Eliminating these errors with high efficiency is challenging. Example references:**
    - ***Ricker, P. M. 2008, ApJS, 176, 293***
    - ***Huang, J., & Greengard, L. 2000, SIAM J. Sci. Comput., 21, 1551***



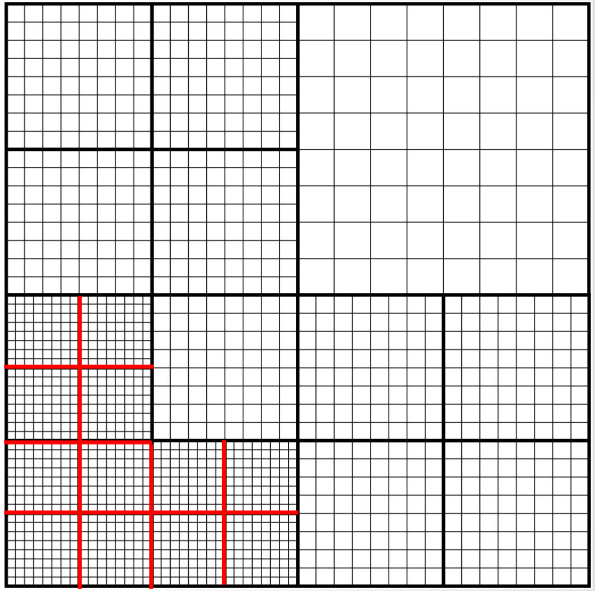
Solving Poisson in AMR: Issue 2

- Resolution boundaries can have very complicate AMR grid geometry
  - Complex data structure
  - Difficult to solve it efficiently
  - Non-trivial parallelization



Solving Poisson in AMR: Issue 3

- **Adjacent patches on the same refinement level in general <u>cannot</u> be computed independently**
  - **Because we cannot know the boundary condition of individual patch in advance**
  - **Interpolation on the boundaries of individual patch will lead to pseudo mass sheets on these interfaces too**
  - **Different from the hyperbolic systems (e.g., hydro, MHD)**
  - **Difficult for efficient parallelization**



Homework

1. Implement the relaxation methods of (i) Jacobi, (ii) Gauss Seidel, and (iii) SOR in 2D. You could refer to the previous example code of solving the diffusion equation.
   a. Demonstrate numerically that they are second-order accurate (by testing a problem with a known analytical solution)
   b. Empirically determine the optimum overrelaxation parameter w in SOR on a 16x16 grid
   c. How do (1) wall-clock time and (2) the number of iterations required to reach convergence scale with the grid size? Test on 16x16, 32x32, and 64x64 grids for all the three methods.

Deadline: April 27 at 11 PM

Reference

1. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*
2. Multigrid examples:
   a. GAMER: https://github.com/gamer-project/gamer/blob/master/src/SelfGravity/CPU_Poisson/CPU_PoissonSolver_MG.cpp
   b. FLASH: http://flash.uchicago.edu/site/flashcode/user_support/flash_ug_devel/node63.html#SECTION05110260000000000000
   c. Enzo: https://github.com/enzo-project/enzo-dev/blob/master/src/enzo/MultigridSolver.C