PDE -- hydrodynamics III & elliptical systems

Lecture 13, Computational Astrophysics (ASTR660)

Hsiang-Yi Karen Yang, NTHU, 12/08/2022

Announcements

- ***HW5 is due TODAY .*** Late submission within one week will receive ***75%*** of the credit
- The final ***oral presentations*** are scheduled on ***12/29*** and ***1/5***. A tentative schedule is available and will be posted on eLearn. Please let me know ASAP if you have any conflicts (you could switch time with someone else if they agree)
- The final ***product*** of the project is due on ***1/12***. Please start early and try to finish the final project on time!

Date	Time	Name	Title
12/29/22	14:20-14:40	吳耕緯	3D N-body with fast multipole method
	14:40-15:00	李佳倫	Effects of AGN Quasar Feedback in Galaxy Clusters
	15:00-15:20	簡嘉成	Improving my magnetohydrodynamic simulation for non-ideal plasma fluid
	15:30-15:50	凌志騰	A gravitational SPH simulation of galaxy collision from scratch
	15:50-16:10	鄭文淇	Simulate impact-driven atmospheric loss using N-body simulation
	16:20-16:40	龔一桓	Ram pressure striping of galaxy cluster: Can a subcluster be assumed as gasless
	16:40-17:00	劉一璠	Simulate the Kelvin-Helmholtz instability by using FLASH code
1/5/23	14:20-14:40	胡英祈	Modeling Dust Polarization in Protoplanetary Disk at (Sub)millimeter Wavelengths
	14:40-15:00	周育如	Multi-gaussian fit, EW, and line ratio for [O I] 5577 and 6300 from DG Tau
	15:00-15:20	考司圖巴	Compare MCMC based fitting with phenomenological models for X-ray Spectra
	15:30-15:50	郭弈翔	Calculate the entropy of a subregion using Python computation
	15:50-16:10	謝明學	simulate the tidal locking phenomenon
	16:20-16:40	張修瑜	Three-body problem in the Moon, Earth, and the Sun.
	16:40-17:00	石郡翰	Using the Mechanism of AIRES to Simulate the Propagation of Cosmic Rays in the universe

Term project (60%)

The grades will include:

- *Midterm oral presentation on the project proposal (20%, 10+2 mins/person):* scientific motivations, literature review, descriptions of methods, feasibility
- ***Final oral presentation on the project results (30%, 15+5 mins/person):*** brief review, efforts (accomplishments/difficulties/solutions), results & discussions, performance (precision, accuracy, speed, comparison with previous works), presentation skills
- ***Final product of the project (10%, due on 1/12/2023):*** submission of code & a 2-page short summary of code and key results/milestones, evaluation based on completeness & complexity of the project, quality of the final product (including user interface, visualization, etc)

Previous lecture...

$$\boldsymbol{q}_i^{n+1} = \boldsymbol{q}_i^n - \frac{\Delta t}{\Delta x}\left[\boldsymbol{F}_{i+1/2}^{n+1/2} - \boldsymbol{F}_{i-1/2}^{n+1/2}\right]$$

- One could write the hydro equations in conservative forms and solve them using the finite-volume method. Key is to approximate the ***time-averaged fluxes at cell edges*** accurately
- Lax-Friedrichs/upwind/Lax-Wendroff methods do not work well for discontinuities
- ***Godunov method*** – a high-resolution, shock-capturing method
  - Step 1: approximate solution using ***piecewise constant method (PCM)***
  - Step 2: solve the ***Riemann problem*** at each cell edge
  - Step 3: obtain the time-averaged fluxes; update the solutions

Previous lecture...

- ***Riemann problem*** -- find the fluid solution between t_n and t_{n+1} given an ***arbitrary 1D discontinuity*** between a left state and a right state
  - Exact Riemann solvers calculate the solutions exactly by solving the eigenvalue problem of hydro equations and finding the characteristics
  - Approximate Riemann solvers are often used to save computational cost
- ***MUSCL-Hancock scheme*** – high-order method, used in real scientific applications
  - Step 1: ***data reconstruction using PLM/PPM***
  - Step 2: evolve the solution for half a timestep
  - Step 3: solve the Riemann problem at cell edges to obtain fluxes
- For data reconstruction using PLM/PPM, ***slope limiters*** and ***TVD methods*** are often used for reducing unphysical oscillations

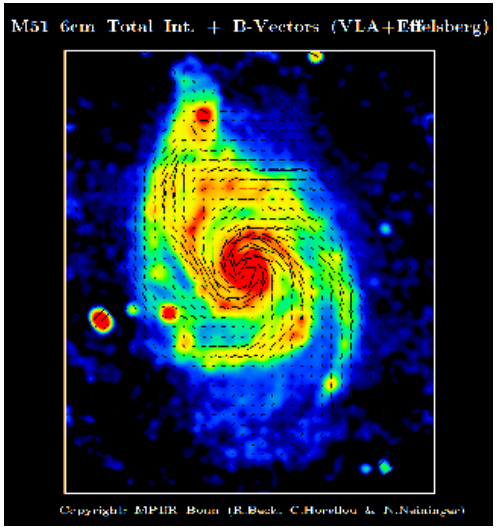
This lecture…

- Continue discussions about hydrodynamics
  - Magnetohydrodynamics (MHD) – equations & properties
  - Numerical methods for solving MHD equations
  - Grid refinement techniques for grid-based codes
- Solving elliptical PDEs – Poisson equation for gravity
- In-class exercise

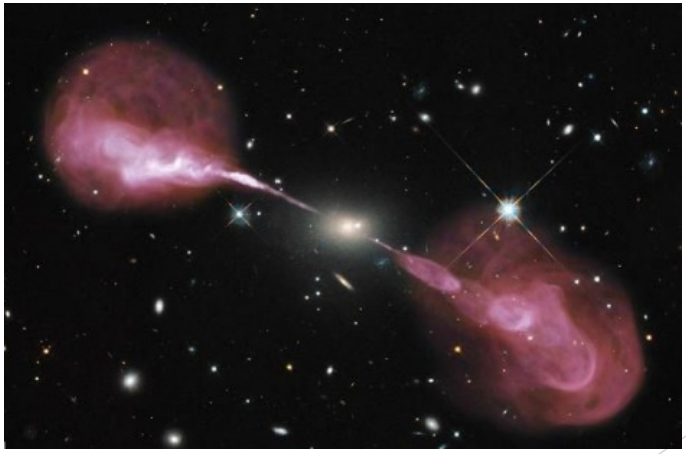
Magnetohydrodynamics (MHD) – equations & properties

Magnetic field in astrophysics

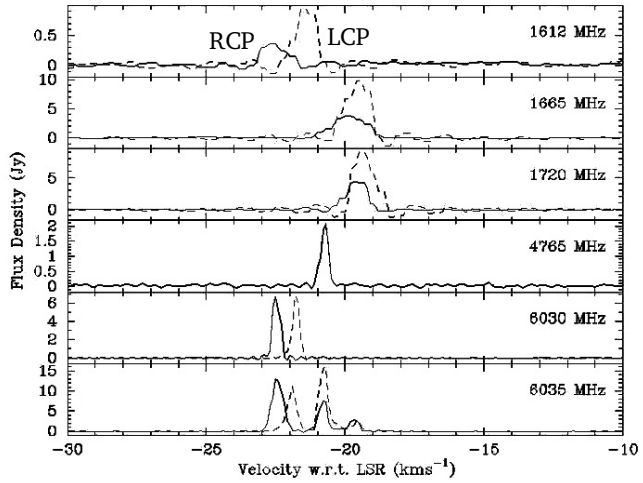
- In astrophysics, because positive and negative charges can rearrange themselves in response to E fields, on large scales we typically have **E** = 0
- By contrast, B field is ubiquitous in the universe
- Observational indications of B fields
  - Synchrotron emission
  - Faraday rotation
  - Zeeman splitting



B field from Faraday rotation in M51 (MPIfR)



Synchrotron emission of Hercules A radio galaxy



Zeeman splitting for different OH masers detected toward G353.41-0.36 (Ellingsen et al. 2002)

Equations for magnetohydrodynamics (MHD)

- Goal: solve coupled Maxwell equations and hydrodynamic equations
- We want a reduced form of Maxwell equations that
  - Does not explicitly include E fields
  - Does not include rapidly varying E fields
- Ohm's law:

$$\mathbf{E} = \frac{1}{\sigma}\mathbf{j}_c - \frac{\mathbf{u}_i}{c}\times\mathbf{B} = \frac{c}{4\pi\sigma}\nabla\times\mathbf{B} - \frac{\mathbf{u}_i}{c}\times\mathbf{B}$$

σ: conductivity, u_i: ion velocity

$$\nabla\cdot\mathbf{E} = 4\pi\rho_c$$

$$\nabla\times\mathbf{E} = -\frac{1}{c}\frac{\partial\mathbf{B}}{\partial t}$$

$$\nabla\cdot\mathbf{B} = 0$$

$$\nabla\times\mathbf{B} = \frac{4\pi}{c}\mathbf{j}_c + \cancel{\frac{1}{c}\frac{\partial\mathbf{E}}{\partial t}}$$

Equations for magnetohydrodynamics (MHD)

- Substitute this into Faraday's law: $\frac{\partial \mathbf{B}}{\partial t} + \nabla \times (\mathbf{B} \times \mathbf{u}_i) = -\nabla \times \left(\frac{c^2}{4\pi\sigma} \nabla \times \mathbf{B} \right)$
- Define the electrical resistivity: $\eta \equiv \frac{c^2}{4\pi\sigma}$
- Then, taking $\boldsymbol{u}_i \approx \boldsymbol{u}$, we have:

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times (\mathbf{B} \times \mathbf{u}) = -\nabla \times \left(\eta \nabla \times \mathbf{B} \right)$$

$$\nabla \cdot \mathbf{B} = 0$$

MHD equations

Apply to non-relativistic, charge neutral, magnetized fluids with electrons and ions described as a single fluid

Continuity equation

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0$$

Momentum equation

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = -\nabla P + \frac{1}{4\pi}(\nabla \times \mathbf{B}) \times \mathbf{B}$$

Total energy equation

$$\frac{\partial \rho E}{\partial t} + \nabla \cdot \left[(\rho E + P)\mathbf{u} \right] = \frac{\eta}{4\pi} |\nabla \times \mathbf{B}|^2$$

heating rate per volume due to Ohmic dissipation

Induction eq. + Ohm's law

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times (\mathbf{B} \times \mathbf{u}) = -\nabla \times \left(\eta \nabla \times \mathbf{B} \right)$$

Divergence-free constraint

$$\nabla \cdot \mathbf{B} = 0$$

Ideal MHD equations

Apply to non-resistive ($\eta \to 0$), infinitely conducting ($\sigma \to \infty$) magnetized fluids

Continuity equation

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0$$

Momentum equation

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = -\nabla P + \frac{1}{4\pi}(\nabla \times \mathbf{B}) \times \mathbf{B}$$

Lorentz force on the matter per unit volume

Total energy equation

$$\frac{\partial \rho E}{\partial t} + \nabla \cdot \left[(\rho E + P)\mathbf{u} \right] = 0$$

Induction eq. + Ohm's law

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times (\mathbf{B} \times \mathbf{u}) = 0$$

Divergence-free constraint

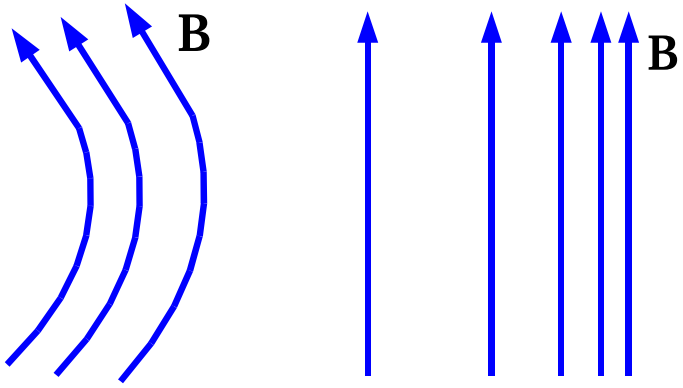
$$\nabla \cdot \mathbf{B} = 0$$

Lorentz force term in MHD equations

- The Lorentz force term can be written as:

$$\mathbf{f}_L = \frac{1}{4\pi}(\nabla \times \mathbf{B}) \times \mathbf{B}$$

$$= \underbrace{\frac{1}{4\pi}(\mathbf{B}\cdot\nabla)\mathbf{B}}_{\text{magnetic tension}} \underbrace{-\frac{1}{8\pi}\nabla(|\mathbf{B}|^2)}_{\text{magnetic pressure}}$$



- Can write the Lorentz force as:

$$\mathbf{f}_L = \nabla \cdot \boldsymbol{T}$$

where the ***Maxwell stress tensor*** is

$$\boldsymbol{T} \equiv \frac{\mathbf{BB}}{4\pi} - \frac{|\mathbf{B}|^2}{8\pi}\boldsymbol{I}$$

Plasma beta and force-free fields

- The ***plasma beta*** parameter expresses the relative importance of thermal pressure and magnetic pressure:

$$\beta \equiv \frac{P}{P_B} = \frac{8\pi P}{B^2}$$

β ≫ 1 (B field unimportant)
β ~ 1 (B field is dynamically important)

- If a system is magnetostatic, i.e., $\rho \frac{D\mathbf{u}}{Dt} = \frac{1}{c}\mathbf{j}_c \times \mathbf{B} - \nabla P = 0$

Lorentz force term

- For a ***low-β magnetostatic*** plasma it is often a good approximation to assume the magnetic field to be ***force-free***:

$$\frac{1}{c}\mathbf{j}_c \times \mathbf{B} = 0$$

Flux-freezing for ideal MHD fluids

- Define the magnetic flux passing through a surface with area dA:

$$\Phi \equiv \int_A \mathbf{B}\cdot\mathbf{n}\,dA$$

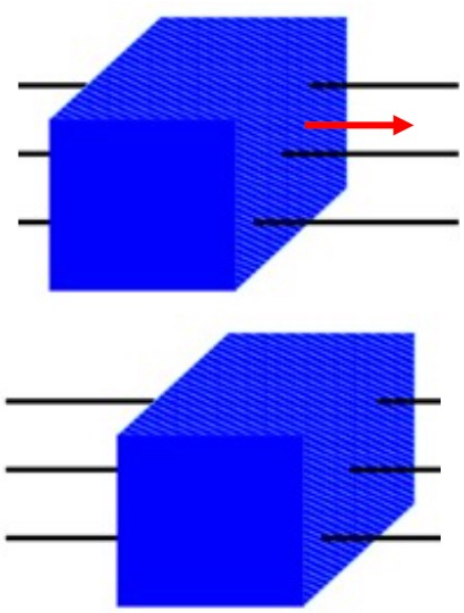
- One can derive that, for ideal MHD fluids:

$$\frac{\partial \mathbf{B}}{\partial t}+\nabla\times(\mathbf{B}\times\mathbf{u})=0 \quad \Rightarrow \quad \frac{D\Phi}{Dt}=0$$

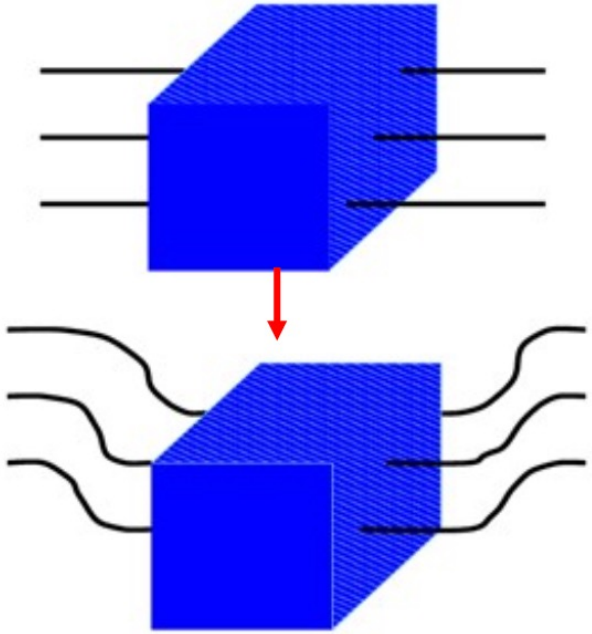
The magnetic flux passing through a surface moving along with the fluid is conserved

Flux-freezing for ideal MHD fluids

Strong fields ($\beta \sim 1$)**:** matter can only move along given field lines (beads on a string)



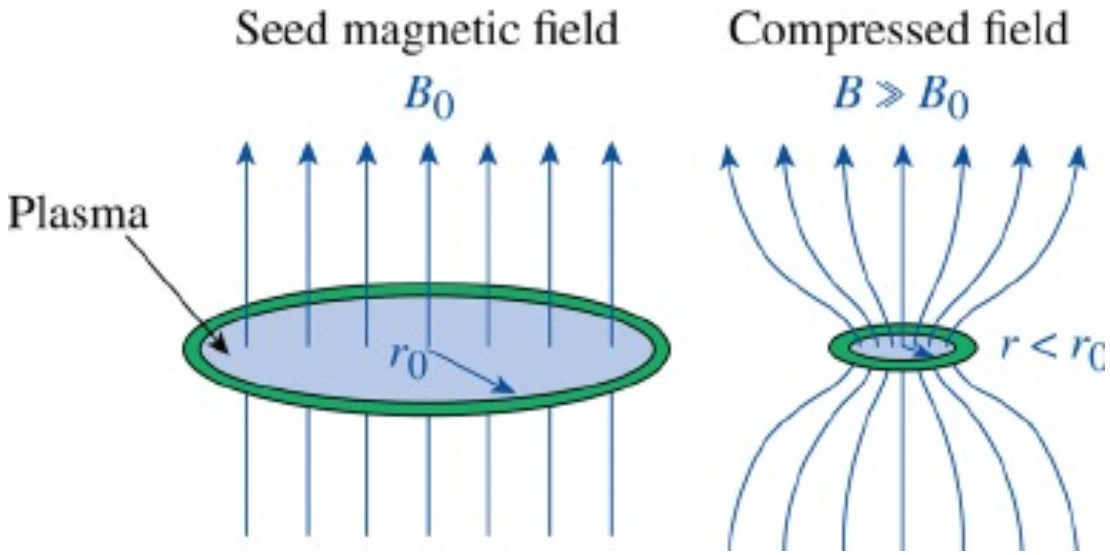
Weak fields ($\beta \gg 1$)**:** field lines are forced to move along with the gas



Credit: C. P. Dullemond

Flux-freezing for ideal MHD fluids

- For environments with ***weak fields*** $(\beta \gg 1)$: field lines are forced to move along with the gas



- This often has important impacts on the formation of astrophysical objects (e.g., formation of stars/compact objects, black hole accretion disks, etc)

Magnetic Reynolds number

- Recall the induction equation: $$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times (\mathbf{B} \times \mathbf{u}) = -\nabla \times \left(\eta \nabla \times \mathbf{B} \right)$$

- We can define the ***magnetic Reynolds number*** to compare the relative importance between the convective and resistive/diffusive terms (*L* is the characteristic length scale for variations in ***B***):

$$Re_M \equiv \frac{L^{-1} u B}{L^{-2} \eta B} = \frac{L u}{\eta}$$

- Example of solar corona:

$L \sim 10^9$ cm $u \sim 10^5$ cm s^{-1} $\eta \sim 10^4$ cm^2 s^{-1} $\Rightarrow$ $\mathrm{Re_M} \sim 10^{10}$

- Thus we can often (but not always) ***ignore η and take the ideal MHD limit***

Characteristics of linearized MHD equations

- Consider adiabatic, linear perturbations about a static, homogenous state:

$$\rho = \rho_0(1+\delta) \quad \mathbf{u} = \mathbf{u} \quad P = P_0(1+\gamma\,\delta) \quad \mathbf{B} = B_0(\hat{\mathbf{n}}+\mathbf{b})$$

- Plugging into the MHD equations:

$$\frac{\partial \delta}{\partial t} + \nabla\cdot\mathbf{u} = 0$$

$$\rho_0 \frac{\partial \mathbf{u}}{\partial t} = -P_0\gamma\nabla\delta + \frac{B_0^2}{4\pi}(\nabla\times\mathbf{b})\times\hat{\mathbf{n}}$$

$$\frac{\partial \mathbf{b}}{\partial t} + \nabla\times(\hat{\mathbf{n}}\times\mathbf{u}) = 0$$

- Assuming solutions of the form $e^{i(\omega t - \mathbf{k}\cdot\mathbf{x})}$ gives:

$$i\,\omega\,\delta - i\,\mathbf{k}\cdot\mathbf{u} = 0$$

$$i\,\omega\,\mathbf{u} = \frac{\gamma\,P_0}{\rho_0} i\,\mathbf{k}\,\delta - \frac{B_0^2}{4\pi\,\rho_0}(i\,\mathbf{k}\times\mathbf{b})\times\hat{\mathbf{n}}$$

$$i\,\omega\,\mathbf{b} - i\,\mathbf{k}\times(\hat{\mathbf{n}}\times\mathbf{u}) = 0$$

Characteristics of linearized MHD equations

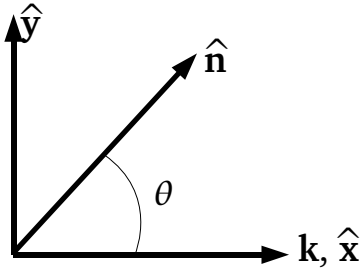
- Define sound speed: $a^2 \equiv \dfrac{\gamma P_0}{\rho_0}$ Alfven speed: $v_A^2 \equiv \dfrac{B_0^2}{4\pi\rho_0}$

- Use eq. (1) and (3) to eliminate δ and $\boldsymbol{b}$:

$$\omega^2 \mathbf{u} + v_A^2[\mathbf{k}\times[\mathbf{k}\times(\hat{\mathbf{n}}\times\mathbf{u})]]\times\hat{\mathbf{n}} - a^2\mathbf{k}\,\mathbf{k}\cdot\mathbf{u} = 0$$

- Take **k** along x-axis and $\hat{\boldsymbol{n}}$ in the xy-plane, and θ the angle between **k** and $\hat{\boldsymbol{n}}$:

$$[\omega^2 - k^2 v_A^2 \cos^2\theta]\mathbf{u} +$$
$$[-k^2(v_A^2 + a^2)u_x + k^2 v_A^2 \cos\theta(u_x\cos\theta + u_y\sin\theta) + k^2 v_A^2\cos^2\theta\, u_x]\hat{\mathbf{x}} +$$
$$[k^2 v_A^2 \sin\theta\cos\theta\, u_x]\hat{\mathbf{y}} = 0$$



Characteristics of linearized MHD equations

Solutions to the previous equation yield three characteristics:

(1) ***Alfven waves*** ($u_x = u_y = 0, u_z \neq 0$):

$$\omega^2 - k^2 v_A^2 \cos^2\theta = 0$$

Transverse waves with speed $v_A \cos\theta$

(2) ***Fast MHD waves*** ($u_x, u_y \neq 0, u_z = 0$; plug in to get matrix equation and require determinant to be zero):

$$\omega^4 - k^2(v_A^2 + a^2)\omega^2 + k^4 v_A^2 a^2 \cos^2\theta = 0$$

Dispersion relation

$$\longrightarrow \omega^2 = \frac{1}{2}\left[(v_A^2 + a^2) \pm \left[(v_A^2 + a^2)^2 - 4 v_A^2 a^2 \cos^2\theta\right]^{1/2}\right] k^2$$

The fast waves correspond to the + sign in the dispersion relation

(3) ***Slow MHD waves***: corresponds to the – sign in the dispersion relation

Numerical methods for solving MHD equations

Key differences between solving HD and MHD equations

- There is an additional equation to solve (the induction equation for **B**)
- There are more characteristics to consider
- The equations are not always hyperbolic
- The solution must maintain the divergence-free character of **B** everywhere

All these factors make numerical MHD much more difficult than pure hydro!!

MHD equations in conservative form

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F_x}}{\partial x} + \frac{\partial \boldsymbol{F_y}}{\partial y} + \frac{\partial \boldsymbol{F_z}}{\partial z} = 0,$$

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ \rho v_x \\ \rho v_y \\ \rho v_z \\ E \\ B_x \\ B_y \\ B_z \end{bmatrix}, \qquad \boldsymbol{F_x} = \begin{bmatrix} \rho v_x \\ \rho v_x^2 + P^* - B_x^2 \\ \rho v_x v_y - B_x B_y \\ \rho v_x v_z - B_x B_z \\ (E + P^*) v_x - B_x (\boldsymbol{B} \cdot \boldsymbol{v}) \\ 0 \\ v_x B_y - v_y B_x \\ v_x B_z - v_z B_x \end{bmatrix}, \text{similarly for } \boldsymbol{F_y}, \boldsymbol{F_z}$$

- Fluid variables can be updated using finite-volume methods similar to pure hydro
- Main problem – how to ensure ***divergence-free constraint*** when updating B fields?

What happens if div(B) /= 0 ?

- Recall that the Lorentz force per unit volume is

$$\mathbf{f}_L = \frac{1}{4\pi}(\mathbf{B}\cdot\nabla)\mathbf{B} - \frac{1}{8\pi}\nabla(|\mathbf{B}|^2)$$

- We wrote it as a conserved flux of a stress tensor:

$$\mathbf{f}_L = \nabla\cdot\boldsymbol{T}, \quad \boldsymbol{T} \equiv \frac{\mathbf{BB}}{4\pi} - \frac{|\mathbf{B}|^2}{8\pi}\boldsymbol{I}$$

- But the above two equations are equivalent only if div(B) = 0 since:

$$\nabla\cdot(\mathbf{BB}) = \mathbf{B}\nabla\cdot\mathbf{B} + (\mathbf{B}\cdot\nabla)\mathbf{B}$$

- If div(B) /= 0, we would effectively introduce a magnetic force component parallel to **B**
- This would also breaks energy conservation

Numerical methods for ensuring div(B) = 0

1) Keep track and evolve the ***vector potential*** **A**, and compute **B** from $\boldsymbol{B} = \nabla \times \boldsymbol{A}$. This would ensure div(B) = 0 as long as our difference approximation to $\nabla \cdot \nabla \times$ always gives zero
2) Apply ***artificial diffusion*** to damp out components of **B** with nonzero divergence
3) For every few timesteps, project out the component of **B** with nonzero divergence – "***divergence cleaning***"
4) Build the divergence constraints into the differencing scheme (e.g., the "***constrained transport***" method)

Divergence cleaning method

- Assume our update method for **B** will produce some nonzero div(B)
- Every few timesteps, we project the numerical solution onto the space of divergence free vector fields
- **Procedure**: an arbitrary vector field **B** and its divergence can be written as:

$$\mathbf{B} = \nabla\phi + \nabla \times \mathbf{A} \qquad \nabla \cdot \mathbf{B} = \nabla^2 \phi$$

- Say our update method yields a field B* with nonzero divergence, we can find the ***undesired component*** $\nabla\phi$ by solving:

$$\nabla^2 \phi^{n+1} = \nabla \cdot \mathbf{B}^*$$

- Finally, we can get the ***divergence cleaned component*** by: $\mathbf{B}^{n+1} = \mathbf{B}^* - \nabla \phi^{n+1}$

Constrained transport (CT) method

- Goal: build the divergence constraints into the differencing scheme
- Using Stokes' theorem:

$$\int_A \frac{\partial \boldsymbol{B}}{\partial t} \cdot d\boldsymbol{A} = \int_A [\boldsymbol{\nabla} \times (\boldsymbol{v} \times \boldsymbol{B})] \cdot d\boldsymbol{A} = \oint_{\partial A} \boldsymbol{v} \times \boldsymbol{B} \cdot d\boldsymbol{l}$$

Define electromotive force (EMF): $\boldsymbol{\varepsilon} = -\boldsymbol{v} \times \boldsymbol{B}$

- Integrate over cell area (e.g., $\Delta y \Delta z$) and time interval ($\Delta t = t_{n+1} - t_n$):

$$B^n_{x,i-1/2,j,k} \equiv \frac{1}{\Delta y \Delta z} \int_{z_{k-1/2}}^{z_{k+1/2}} \int_{y_{j-1/2}}^{y_{j+1/2}} B_x(x_{i-1/2}, y, z, t^n) dy dz$$

$$\varepsilon^{n+1/2}_{y,i-1/2,j,k-1/2} \equiv \frac{1}{\Delta y \Delta t} \int_{t^n}^{t^{n+1}} \int_{y_{j-1/2}}^{y_{j+1/2}} \varepsilon_y(x_{i-1/2}, y, z_{k-1/2}, t) dy dt$$

$$\varepsilon^{n+1/2}_{z,i-1/2,j-1/2,k} \equiv \frac{1}{\Delta z \Delta t} \int_{t^n}^{t^{n+1}} \int_{z_{k-1/2}}^{z_{k+1/2}} \varepsilon_z(x_{i-1/2}, y_{j-1/2}, z, t) dz dt$$

Constrained transport (CT) method

- Write the Stokes' theorem in finite-difference form to update B:

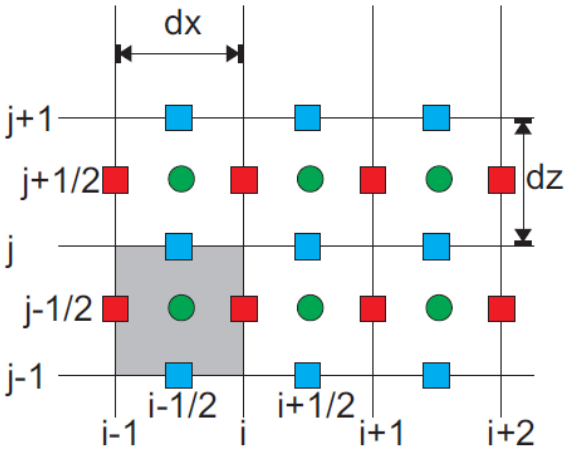
$$B^{n+1}_{x,i-1/2,j,k} = B^{n}_{x,i-1/2,j,k} - \frac{\Delta t}{\Delta y}\left(\varepsilon^{n+1/2}_{z,i-1/2,j+1/2,k} - \varepsilon^{n+1/2}_{z,i-1/2,j-1/2,k}\right) + \frac{\Delta t}{\Delta z}\left(\varepsilon^{n+1/2}_{y,i-1/2,j,k+1/2} - \varepsilon^{n+1/2}_{y,i-1/2,j,k-1/2}\right)$$

area-averaged B field

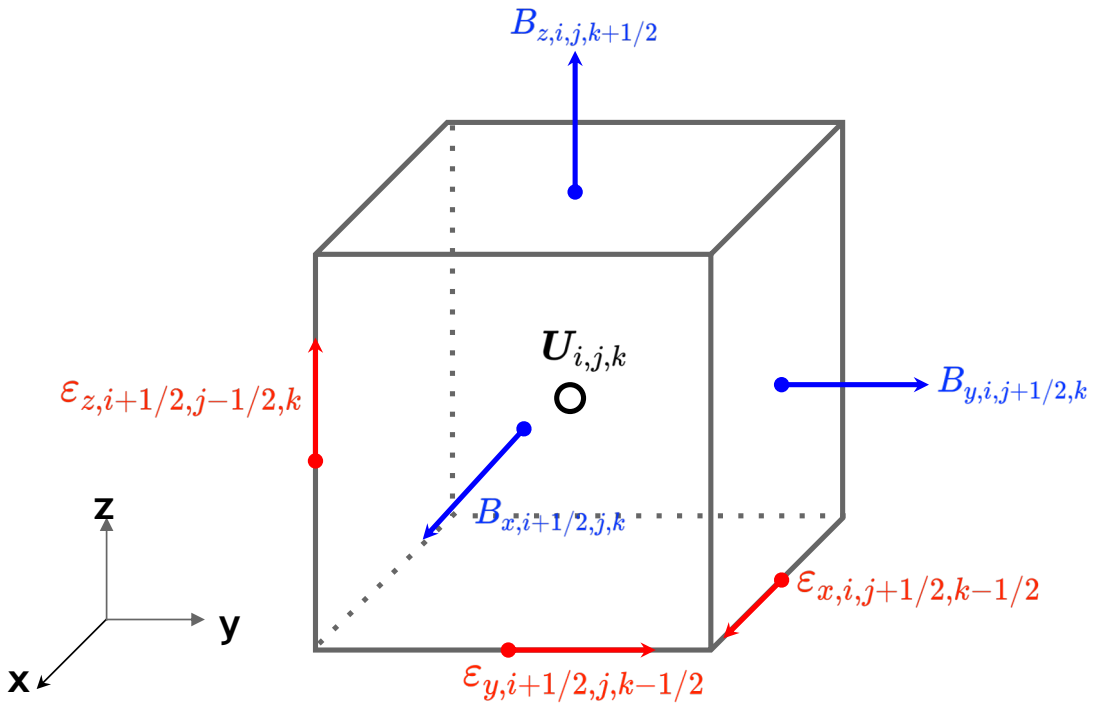
time- and line-averaged EMF

- Similar expressions can be derived for $B^{n+1}_{y,i,j-1/2,k}$ & $B^{n+1}_{z,i,j,k-1/2}$
- This form is again exact (similar to finite-volume scheme); exact way to compute EMF varies
- Area-averaged B fields are located at ***cell faces*** instead of cell centers -> "***staggered (交錯的) mesh***"

Schematic for staggered mesh (Becker & Kaus 2016)



Staggered mesh in the CT method



The CT method ensures div(B) = 0

- Volume integral of the divergence-free constraint:

$$\frac{1}{\Delta x \Delta y \Delta z} \int_{V_{i,j,k}} (\boldsymbol{\nabla} \cdot \boldsymbol{B^n}) dV = 0$$

$$\rightarrow \quad (\boldsymbol{\nabla} \cdot \boldsymbol{B^n})_{i,j,k} = \frac{B^n_{x,i+1/2,j,k} - B^n_{x,i-1/2,j,k}}{\Delta x} + \frac{B^n_{y,i,j+1/2,k} - B^n_{y,i,j-1/2,k}}{\Delta y} + \frac{B^n_{z,i,j,k+1/2} - B^n_{z,i,j,k-1/2}}{\Delta z} = 0$$

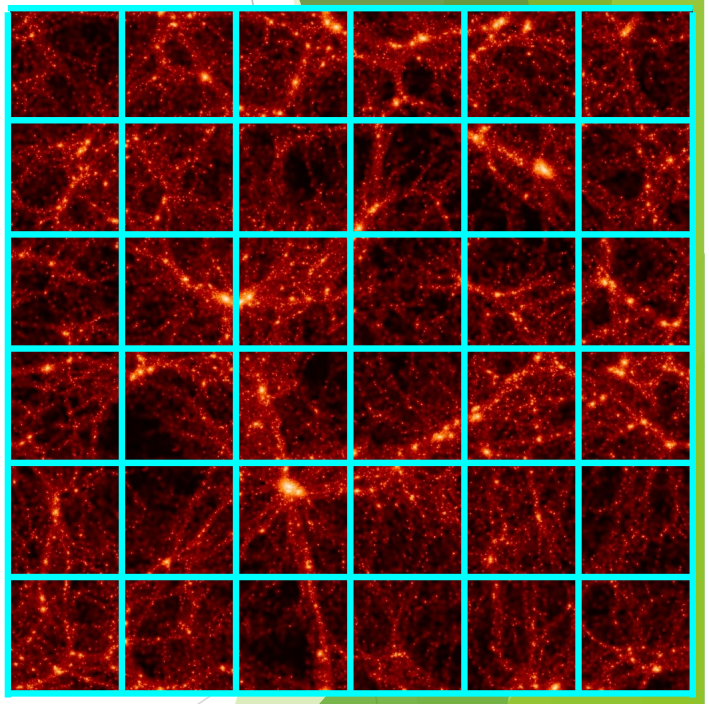
- One can show that CT updates guarantee that $\boldsymbol{\nabla \cdot B^{n+1} = \nabla \cdot B^n}$

- Using the CT method, divergence-free constraint can be preserved to machine precision
- Note that it must be satisfied in the initial condition (e.g., by using vector potential **A**)

Numerical techniques for grid-based codes

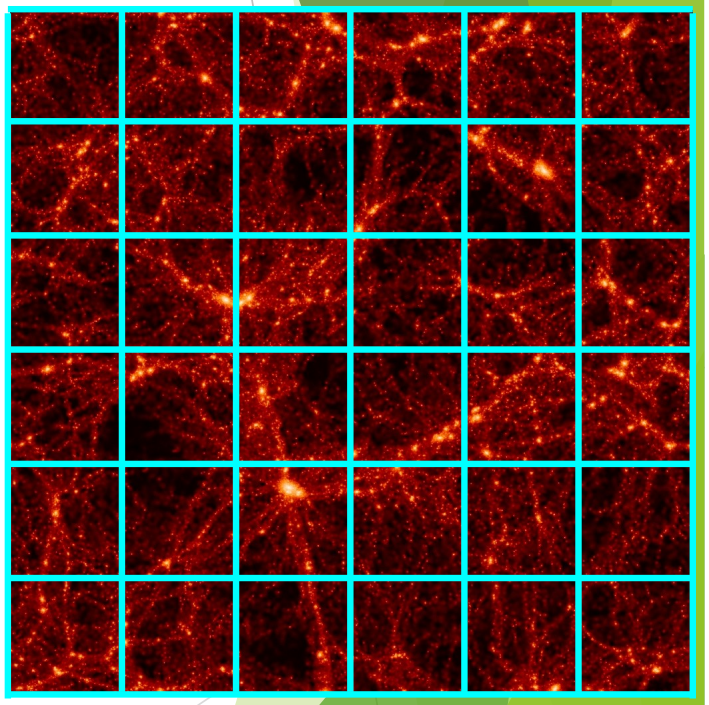
Mesh refinement in grid-based codes

- Ideally we would cover the Universe with a uniform mesh
- Advantages of uniform grid:
  - Constant numerical diffusivity
  - Properties are mathematically simple
  - Easier to parallelize
- Cons – waste of computational resources!
  - Storage $\propto N^3$, cost $\propto N^4$



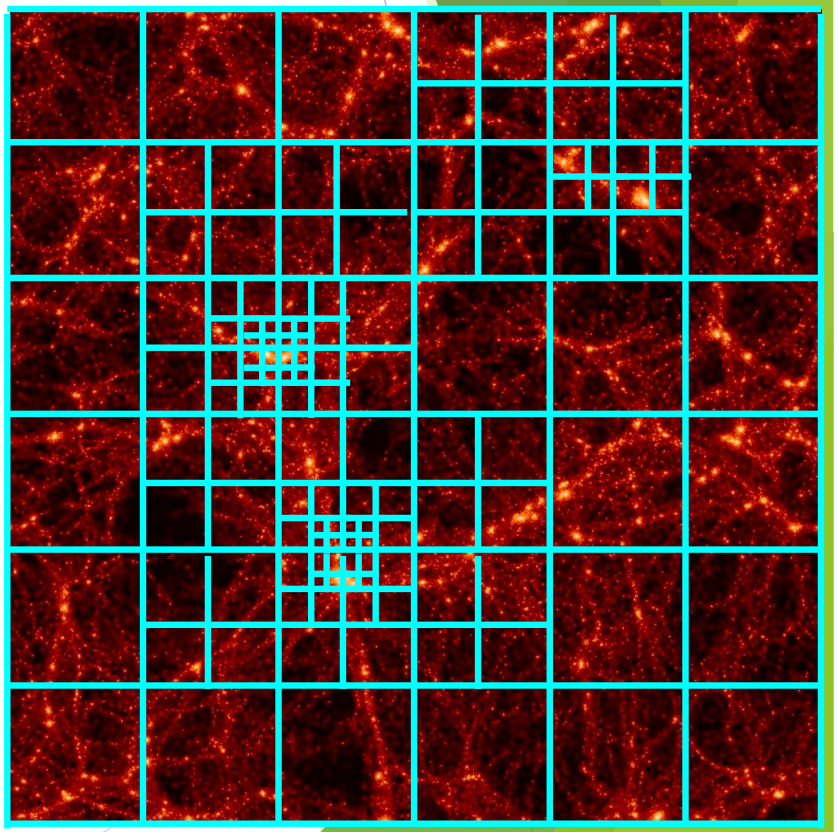
Mesh refinement in grid-based codes

- **Example**: if we want to resolve interstellar distance ($\Delta x \sim 1pc$) in a cosmological volume (L ~ 1Gpc) => $N \sim 10^9$
  - Need 10^{28} bytes per variable
  - A computer requiring 1ns to update one zone would take 3×10^9 years for one timestep!
  - And most of it would be voids….



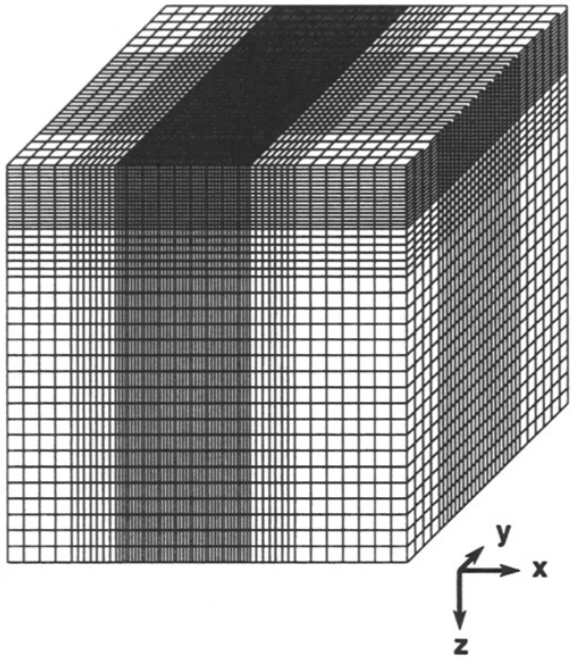
Mesh refinement in grid-based codes

- Need a better way to concentrate resources where interesting things are happening -- more flexible grid refinement techniques
- But many things need to be considered:
  - More sophisticated bookkeeping for storage and parallelization
  - Must derive finite-difference expressions for non-constant zone spacing
  - Operators to transfer info between grids (e.g., convert fluxes across fine-coarse boundaries)
  - A way to determine where to concentrate resources

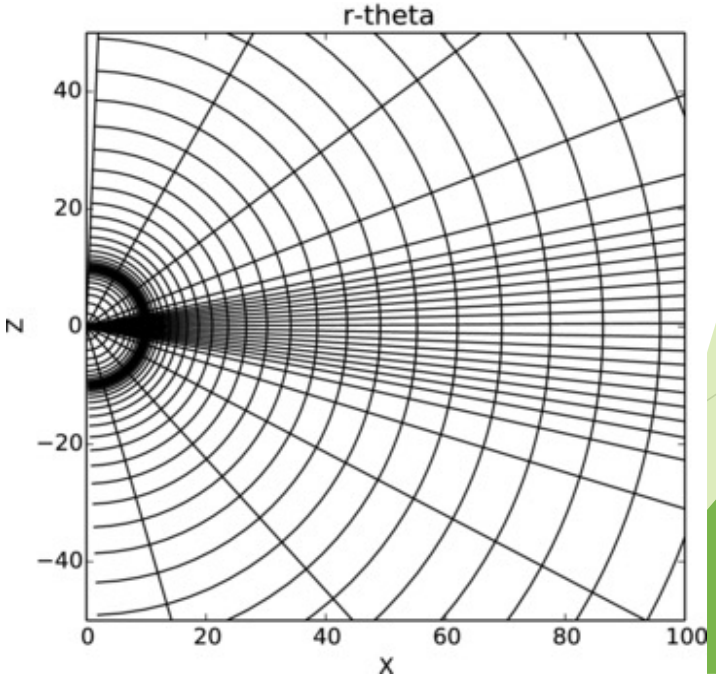


Mesh refinement techniques #1 – non-uniform meshes

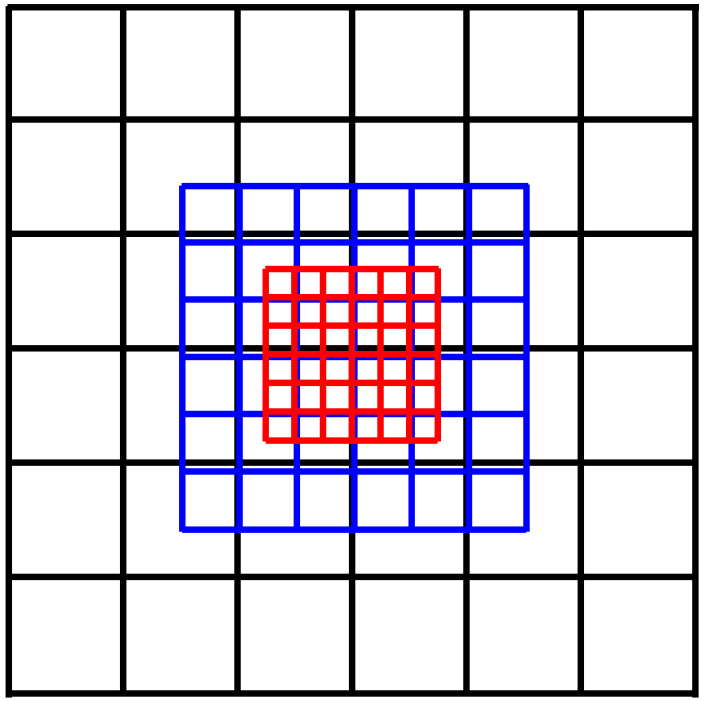
- Cartesian (e.g., galactic disks)



- Spherical (e.g., stellar/black-hole accretion disks)

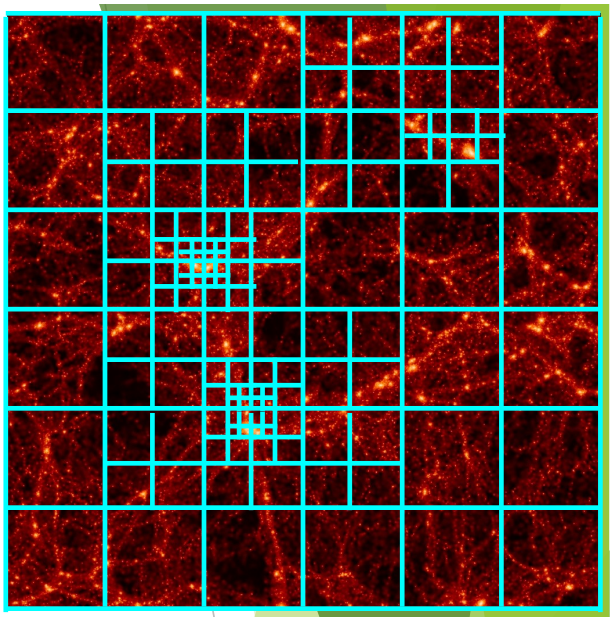


Mesh refinement techniques #2 – nested grids



- Usually works for static meshes – region to be refined doesn't move
- Meshes evolve independently; fine grids take B.C. by interpolation from coarser grids

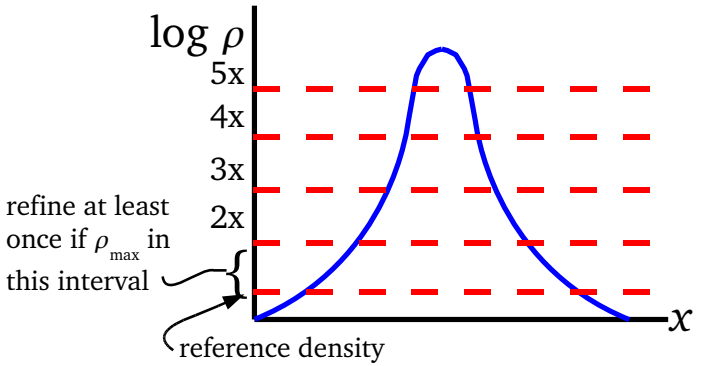
Mesh refinement techniques #3 – adaptive mesh refinement (AMR)



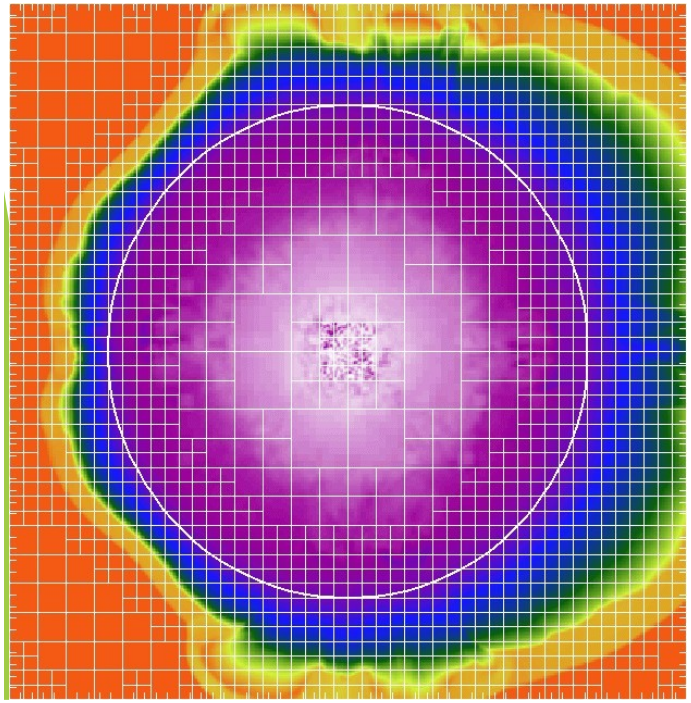
- Similar to nested grids, but refined meshes can be created and destroyed during runtime ***adaptively***
- AMR is widely used for grid-based astrophysical hydro codes (e.g., FLASH, ENZO, PLUTO, RAMSES, Athena++, GAMER)
- CFL criterion ($\Delta t = CFL\frac{\Delta x}{\max(v)}$) is more restrictive on fine meshes
  - Evolve the whole simulation domain using a global timestep constrained by the CFL criterion for fine meshes (e.g., FLASH code)
  - Evolve fine/coarse meshes using different timesteps – ***adaptive time-stepping*** (e.g., ENZO code)

AMR refinement criteria

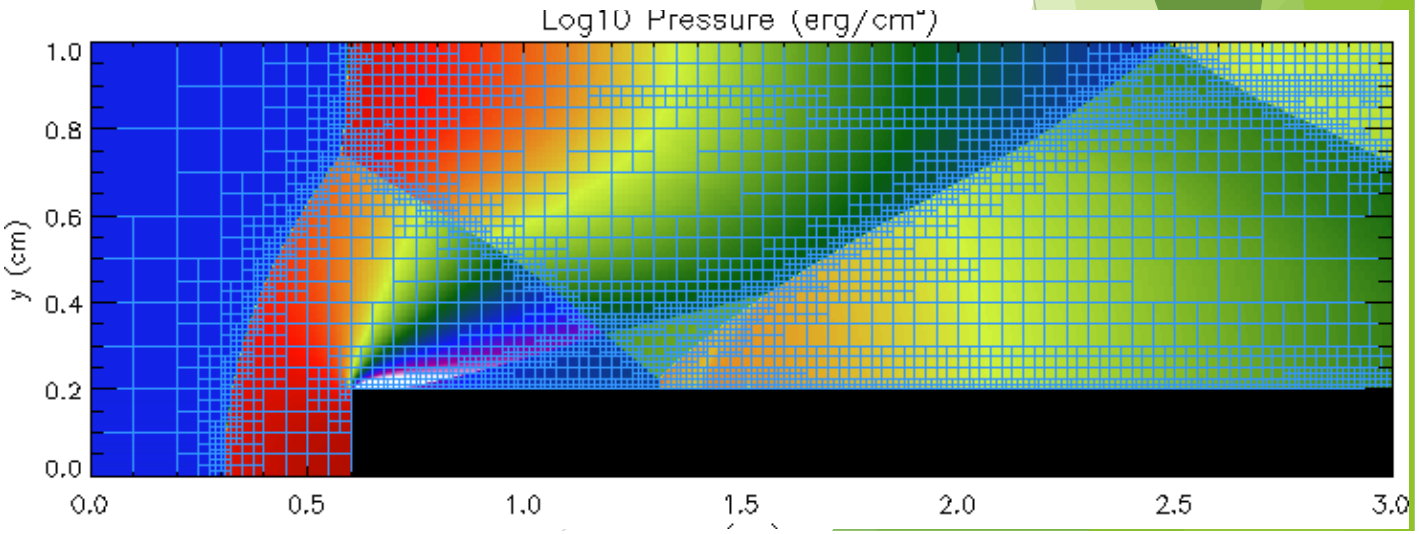
Magnitude of density



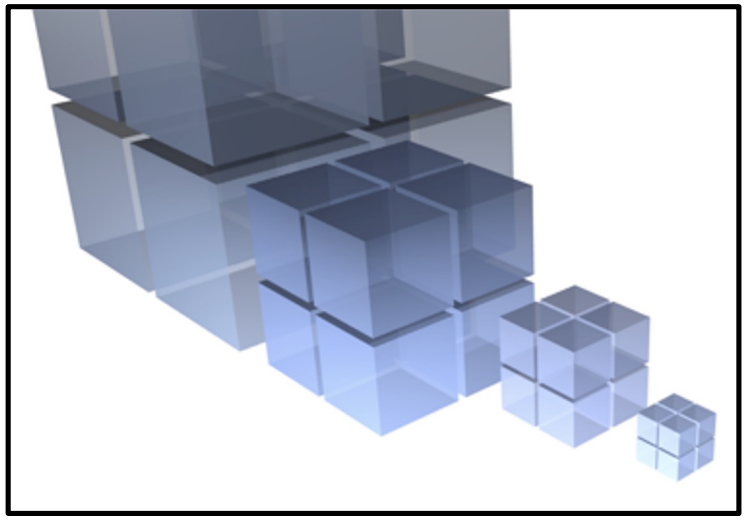
Geometrical (e.g., refine center/edge of object)

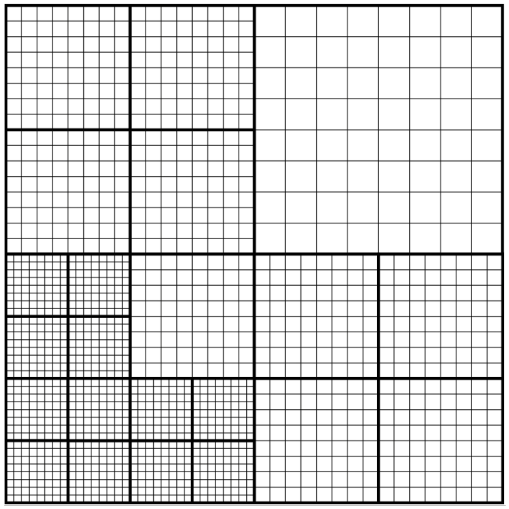


Second derivative of density or temperature (good for capturing shocks)



Example for patch/block-based AMR



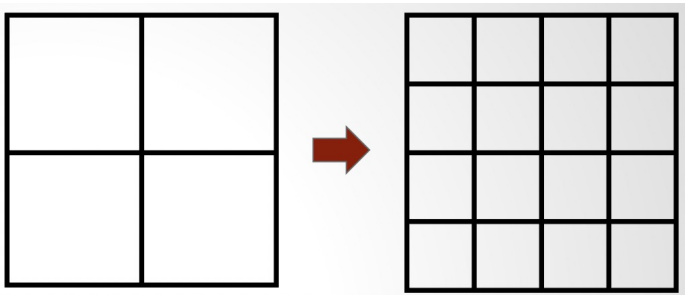


8^3 cells per patch/block; all patches have identical spatial geometry

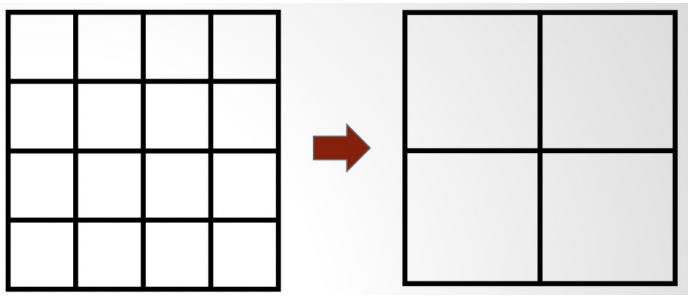
- AMR codes adopting this approach: FLASH, Athena++, GAMER

Dealing with coarse-fine boundary in AMR

- One needs to be careful about ***flux conservation*** and ***divergence-free*** **B** fields at coarse-fine boundaries in AMR codes
- Two types of operations are needed:

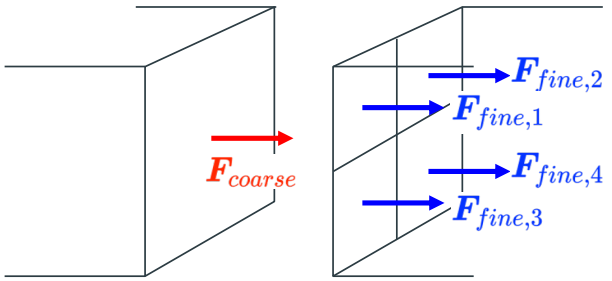


Prolongation (interpolation)



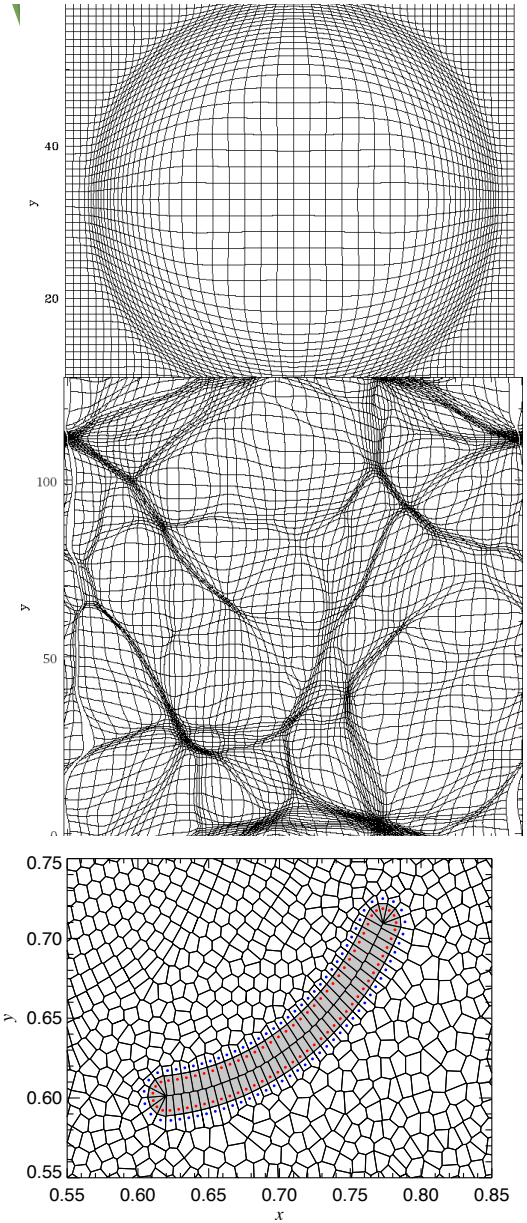
Restriction (averaging)

- In general, $\boldsymbol{F}_{coarse} \neq \frac{1}{4}(\boldsymbol{F}_{fine,1} + \boldsymbol{F}_{fine,2} + \boldsymbol{F}_{fine,3} + \boldsymbol{F}_{fine,4})$
- The flux difference must be corrected (on the coarse grid) to ensure conservation



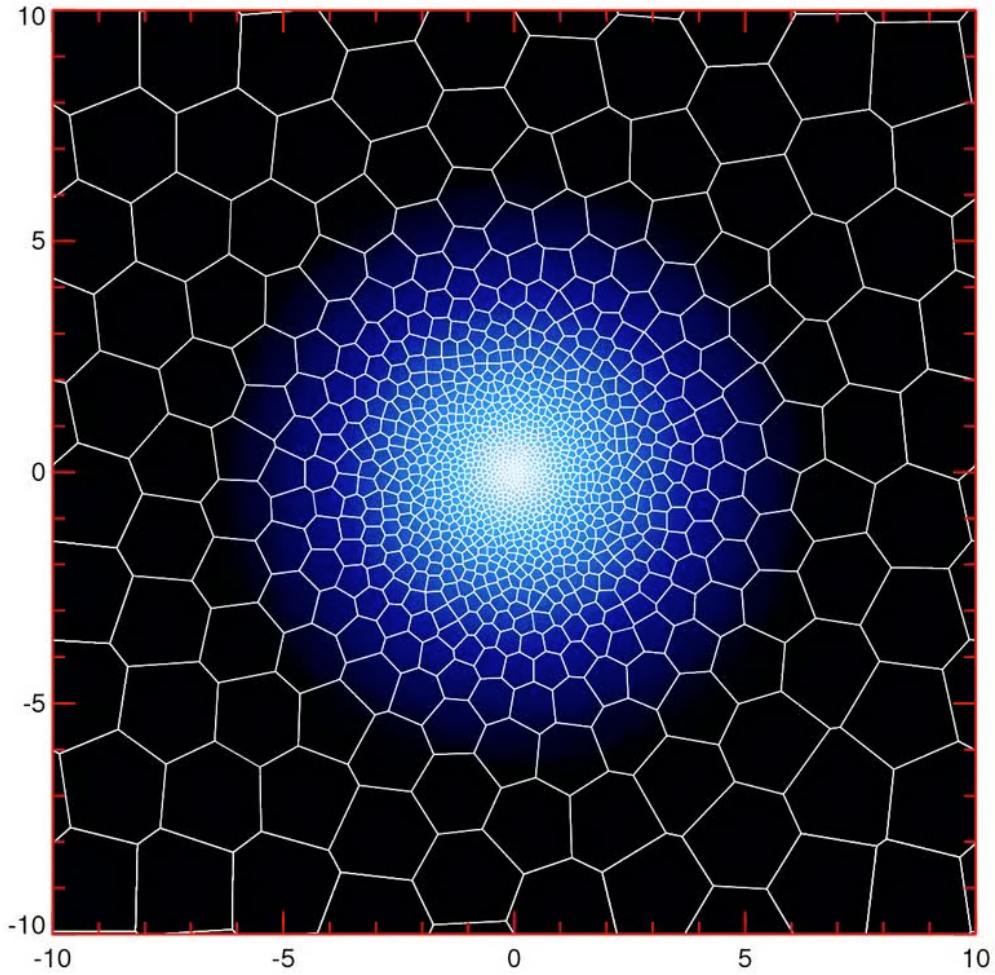
Mesh refinement techniques #4 – Lagrangian/moving meshes

- Mesh is tied to the fluid – stretches when it expands, squeezes when it contracts
- Astrophysical hydro codes that use moving meshes – AREPO, GIZMO
- Advantages:
  - Constant mass resolution – high-density regions are naturally finely refined
  - Sharp resolution of contact discontinuities
  - Easier to use arbitrary/moving B.C.
- Disadvantages:
  - Grid stretching causes numerical dissipation to be different in different dimensions
  - Grid distortion/tangling
  - Harder to implement



Example – AREPO simulations

- AREPO is a ***Lagragian/moving-mesh*** code
- They have cell-generating points that move with the fluid
  - This has the advantage of ***Lagrangian*** codes that can have higher resolutions in high-density regions
- After moving the cell-generating points, they divide the domain at the boundaries of the points and calculate the fluid variables using fluxes at each boundary
  - This has the advantage of ***Eulerian*** codes which has higher accuracy in solving the fluid equations, especially when there are discontinuities



MHD & grid-refinement techniques -- summary

- ***MHD equations*** are used to describe non-relativistic, charge neutral, magnetized fluids
  - MHD equations = HD equations + ***induction equation*** for B + ***divergence-free*** constraint for B
  - B field exerts Lorentz force on the fluid, including magnetic tension and magnetic pressure forces
  - ***Plasma beta ($\beta \equiv \frac{P}{P_B} = \frac{8\pi P}{B^2}$)*** describes whether B field is dynamically important
- ***Ideal*** MHD equations apply to non-resistive, infinitely conducting magnetized fluids
  - Typically true for astrophysical systems (magnetic Reynolds number $Re_M \equiv \frac{Lu}{\eta} \gg 1$)
  - ***Flux-freezing***: magnetic flux passing through a surface moving with the fluid is conserved
  - Three characteristics for linear perturbations: Alfven waves, fast waves, and slow waves

MHD & grid-refinement techniques -- summary

- MHD equations can be solved using finite-volume methods similar to pure hydro
- Numerical methods to ensure div(B) = 0 include
  1) using vector potential **A**
  2) applying artificial diffusion
  3) divergence cleaning
  4) constrained transport (CT) method
- Advanced grid refinement techniques for grid-based codes are developed to save computational costs. Commonly used techniques include:
  1) non-uniform meshes
  2) nested grids
  3) adaptive mesh refinement (AMR)
  4) Lagrangian/moving meshes

PDE – elliptical systems

Recall the 3 types of 2nd-order PDEs

- 2nd-order linear PDEs have the general form:

$$au_{xx} + bu_{xy} + cu_{yy} + du_x + eu_y + fu + g = 0$$

- They are classified into three categories:
  - If $b^2 - 4ac > 0$: ***hyperbolic*** (e.g., wave equation, hydro equations)

$$\frac{\partial^2 \rho}{\partial t^2} - a^2 \nabla^2 \rho = 0$$

  - If $b^2 - 4ac = 0$: ***parabolic*** (e.g., heat equation)

$$\frac{\partial T}{\partial t} = \kappa \nabla^2 T$$

  - If $b^2 - 4ac < 0$: ***elliptic*** (e.g., Laplace equation, Poisson equation)

$$\nabla^2 \phi = 0 \qquad \nabla^2 \phi = 4\pi G \rho$$

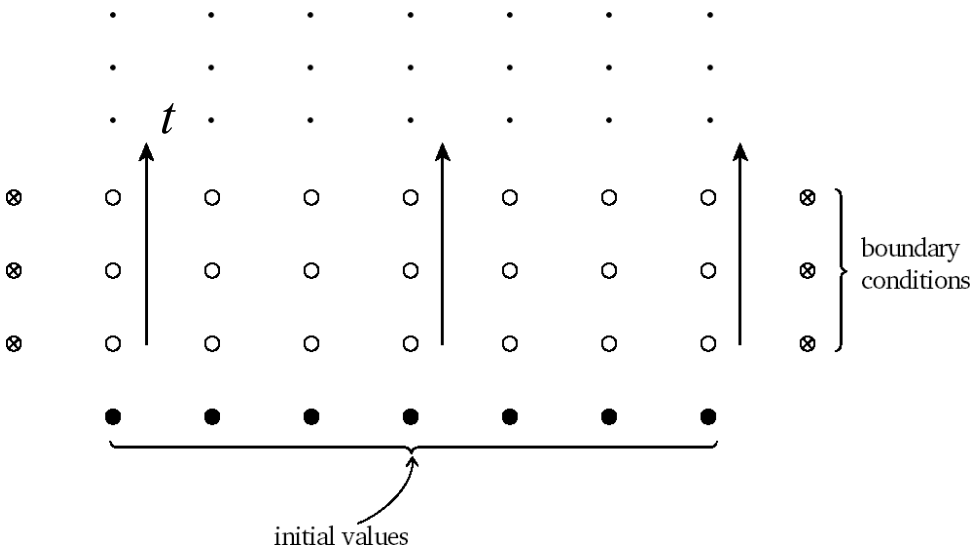
Types of 2nd-order PDEs

Properties of the PDEs are not so simply classified, but roughly speaking:

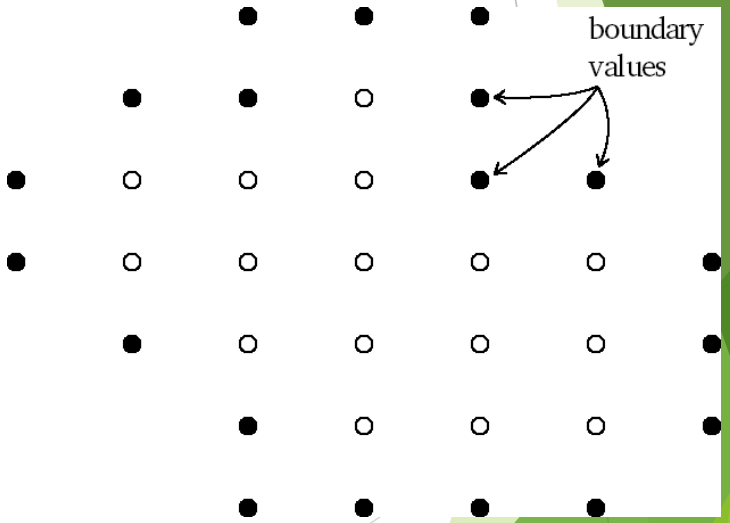
- **Hyperbolic** PDEs: *time-dependent* systems, *conservative* physical processes (e.g., advection, hydro), *not evolving toward steady state*
- **Parabolic** PDEs: *time-dependent* systems, *dissipative/damped* physical processes (e.g., diffusion), *are evolving toward steady state*
- **Elliptic** PDEs: *time-independent* systems (e.g., gravitational potential for a point mass), *already reached steady state*

Types of 2nd-order PDEs

- Time-dependent ***parabolic*** & ***hyperbolic*** equations are typically ***IVPs + BVPs***



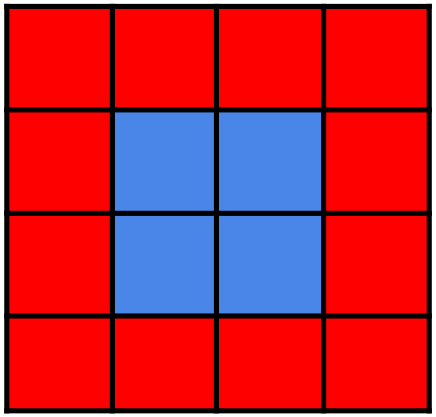
- Time-independent ***elliptic*** equations are typically ***BVPs***



Solving the Poisson equation

$$\nabla^2 \phi(\boldsymbol{r}) = \rho(\boldsymbol{r})$$

- ρ: mass density, ϕ: gravitational potential, assuming $4\pi G = 1$
- This is a BVP: given ρ in domain V and ϕ at the boundary, solve for ϕ in domain V
- This is typically solved ***iteratively***: start from an initial guess for solution, improve it until ***residual*** is as small as desired



Given ϕ

Given ρ, solve ϕ

Relaxation method

- Rewrite the Poisson equation: $L\phi = \rho \rightarrow \boxed{\frac{\partial \phi}{\partial t} = L\phi - \rho}$ *L*: elliptic/Laplacian operator
- **Step 1**: choose initial guess of ϕ
- **Step 2**: let the system ***relax until equilibrium is reached***. That is, when

$$\frac{\partial \phi}{\partial t} = 0 \rightarrow L\phi = \rho$$ *(solution to the original elliptic equation is found)*

- What we care about is only the final (relaxed) solution, so intermediate (unrelaxed) solution is allowed to have large errors
- Relaxation can be very time-consuming ($\propto N^2$) -> key to design an algorithm is to speed up the relaxation process

Relaxation method for Poisson equation

$$\nabla^2\phi = \rho \rightarrow \frac{\partial\phi}{\partial t} = \nabla^2\phi - \rho$$

- We can write down a 2D discrete form with FTCS scheme (see Lecture 10) 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 criterion requires: $\Delta t \leq \Delta^2/4 \rightarrow$ let $\Delta t = \Delta^2/4$
- Then we have the ***Jacobi method***:

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

iterate until relaxation/equilibrium/convergence is reached

Relaxation method #2 – Gauss-Seidel (GS) method

- The Jacobi method converges very slowly (# iterations $\propto N^2$)
- Alternatively, Gauss-Seidel method uses "in-place" updates:

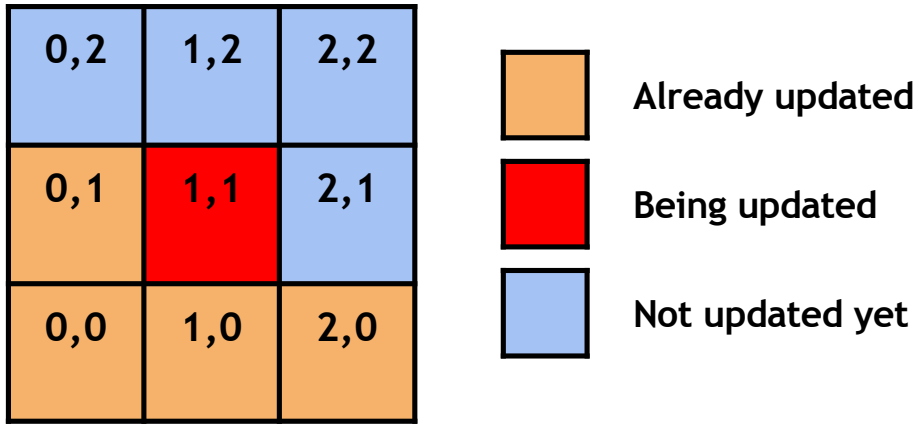
$$\phi_{i,j}^{n+1} = \frac{1}{4}\left(\phi_{i+1,j}^{n/n+1} + \phi_{i-1,j}^{n/n+1} + \phi_{i,j+1}^{n/n+1} + \phi_{i,j-1}^{n/n+1} - \Delta^2 \rho_{i,j}\right)$$

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

- This method converges faster than the Jacobi method
- But # iterations still $\propto N^2$

Relaxation method #2 – Gauss-Seidel (GS) method

- Example of updating procedure



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

Relaxation method #3 – successive over-relaxation (SOR) method

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

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

$\xi_{i,j}$ (residual)

correction

- SOR method: $\phi_{i,j}^{n+1} = \phi_{i,j}^{n} + w\frac{\Delta^2\xi_{i,j}}{4}$

→ $\phi_{i,j}^{n+1} = \phi_{i,j}^{n} + \omega\left(\phi_{i,j,GS}^{n+1} - \phi_{i,j}^{n}\right) = (1-\omega)\phi_{i,j}^{n} + \omega\phi_{i,j,GS}^{n+1}$

which is weighted average of current state and next GS state

Relaxation method #3 – successive over-relaxation (SOR) method

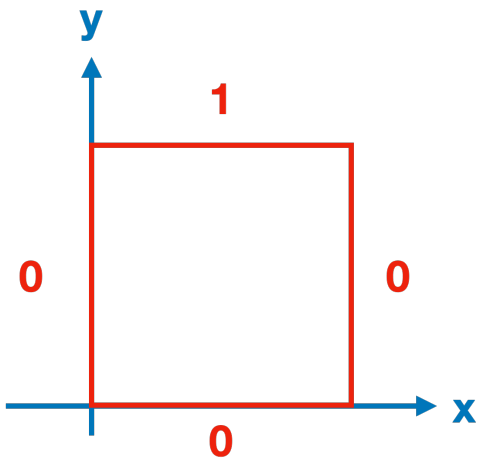
$$\phi_{i,j}^{n+1} = \phi_{i,j}^{n} + \omega\left(\phi_{i,j,GS}^{n+1} - \phi_{i,j}^{n}\right) = (1-\omega)\phi_{i,j}^{n} + \omega\phi_{i,j,GS}^{n+1}$$

- ω is a fixed relaxation parameter chosen to accelerate convergence
  - **$\omega > 1$: over-relaxation** (as if CFL number > 1)
  - $\omega < 1$: under-relaxation
  - $\omega = 1$: Gauss-Seidel method
  - $\omega > 2$ or $\omega < 0$: method diverges
- SOR method converges much faster than Jacobi or GS methods (***# iterations*** $\propto$ ***N*** instead of N^2)
- However, choosing optimal ω is difficult in general

In-class exercises

Exercise – solving the Laplace equation using the Jacobi relaxation methods

- For the Laplace equation, $\nabla^2\phi = \rho = 0 \rightarrow \frac{\partial\phi}{\partial t} = \nabla^2\phi$
- Jacobi method: $\phi_{i,j}^{n+1} = \frac{1}{4}\left(\phi_{i+1,j}^{n} + \phi_{i-1,j}^{n} + \phi_{i,j+1}^{n} + \phi_{i,j-1}^{n} - \Delta^2\rho_{i,j}\right)$
- Let's use this B.C. for ϕ:



$n_x = n_y = 10$
$(x_{min}, x_{max}) = (0, 1)$
$(y_{min}, y_{max}) = (0, 1)$

Exercise – solving the Laplace equation using the Jacobi relaxation methods

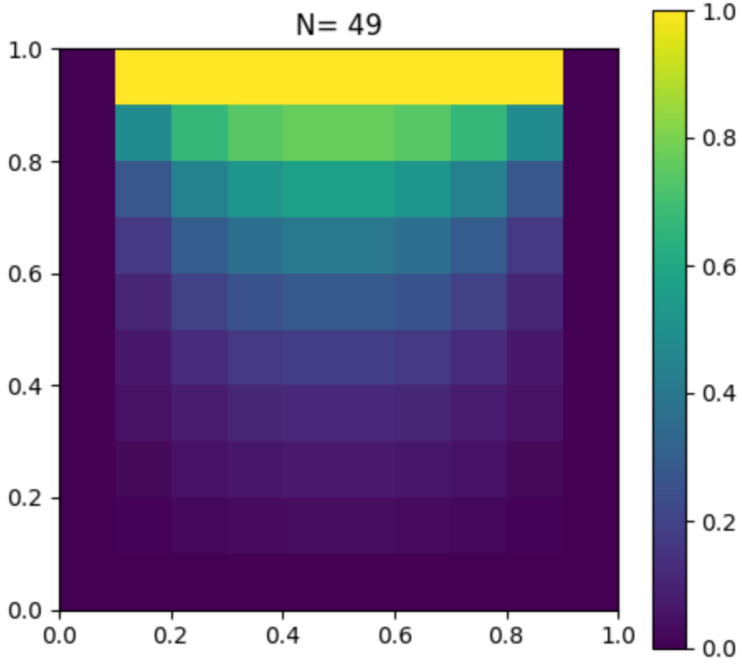
- Download the sample Jupyter Notebook from the following link: https://www.dropbox.com/s/nz2ozh42ddhf1es/laplace_template_L13.ipynb?dl=0
- If you know how to run Jupyter Notebooks from CICA, feel free to do so. Otherwise you could go to the web-based Jupyter Lab: https://jupyter.org/try-jupyter/lab/
- Upload the template to the Jupyter Lab environment. Take a look at the template and understand how it works
- Implement the B.C. in the `set_boundary_conditions` function and the Jacobi method in the `evolve_jacobi` function
- Run the code and visualize the result

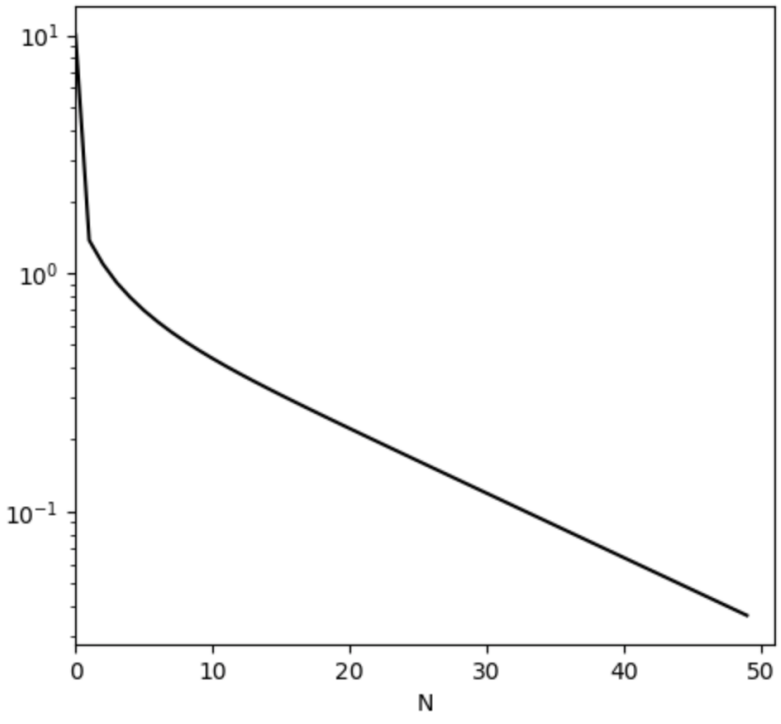
Jacobi method

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

Exercise – solving the Laplace equation using the Jacobi relaxation methods

- Results for the Jacobi method:





Exercise #1b – solving the Laplace equation using the Gauss-Seidel method

- Implement the Gauss-Seidel (GS) method in the evolve_gauss_seidel function
- Run the code and visualize the result. Does it converge faster than the Jacobi method?
- To get the bonus credit, please submit your code and the screenshots for both methods to the TAs by end of today (12/08/2022)

GS method

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

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

References & acknowledgements

- Course materials of Computational Astrophysics from Prof. Kuo-Chuan Pan (NTHU)
- Course materials of Computational Astrophysics from Prof. Hsi-Yu Schive (NTU)
- Course materials of Computational Astrophysics and Cosmology from Prof. Paul Ricker (UIUC)
- “Computational Physics” by Rubin H. Landau, Manuel Jose Paez and Cristian C. Bordeianu
- “Scientific Computing – An Introductory Survey” by Michael T. Heath