# Lecture 2
# Mesh-Based N-body Simulations

Baojiu Li (ICC, Durham)

Mexican Numerical Simulations School (03-06 Oct 2016)

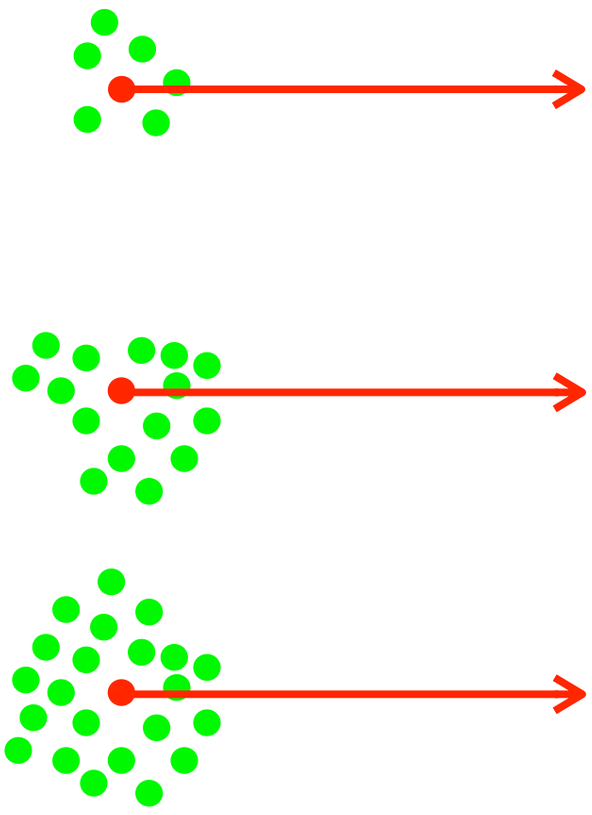
# Challenge for Simulating such Models

- We have seen from Lecture 1 that non-standard models often produce a new force (often called the `fifth' force as it does not belong to any of the four known types of fundamental forces).
- This force can affect the motion of particles, in some cases even the geodesics of photons and massless neutrinos. This will affect the formation of cosmic structures and therefore observations.
- So what is the main difference between such a new force and the standard gravitational force that makes solving it more difficult?

# Chameleon Mechanism & Models

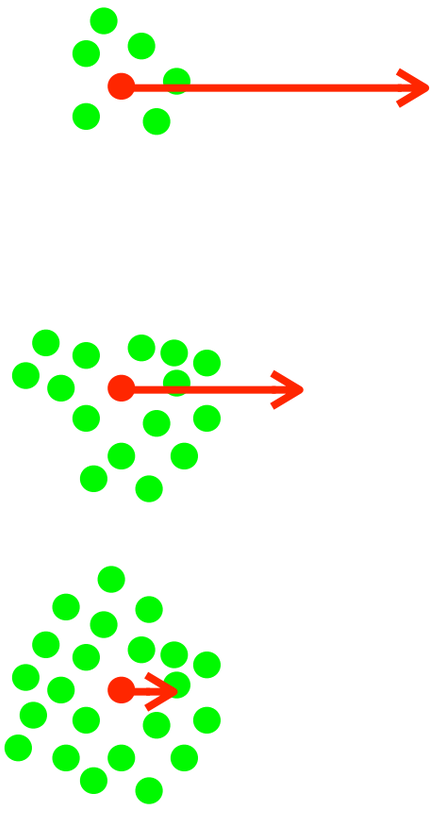
GR force (1 particle)

many particles



the new force (1 particle)

many particles



Yukawa force:
larger m, shorter range

$$\varphi(r) \quad \propto \quad \frac{1}{r}\exp(-mr)$$

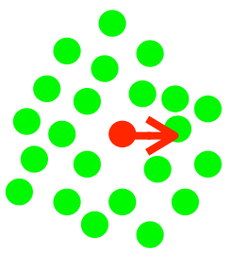
# Chameleon Mechanism & Models

GR force (1 particle)

the new force (1 particle)

Standard gravitational force is environment independent and has an known analytical force law - simple summation is possible

The new force is (usually) environment dependent: it is impossible to calculate the new force produced by a particle without knowing how many particles are surrounding



larger m, shorter range

$$\varphi(r) \quad \propto \quad \frac{1}{r}\exp(-mr)$$

# Challenge for Simulating such Models

- Very complicated, nonlinear, field equations, must be solved in order to accurately calculate this new force.

$$
\begin{aligned}
\nabla^2\varphi + & \left(\nabla^2\varphi\right)^2 & = & \; C(\varphi)\rho_{\rm m} \\
& \nabla^i\nabla^j\varphi\nabla_i\nabla_j\varphi & & \\
& \left(\nabla^2\varphi\right)^3 & & \\
& \nabla^i\nabla^j\varphi\nabla_j\nabla^k\varphi\nabla_k\nabla_i\varphi & & \\
& \nabla^2\varphi\nabla^i\nabla^j\varphi\nabla_i\nabla_j\varphi & & \\
& |\nabla\varphi|^2 & & \\
& \cdots & &
\end{aligned}
$$

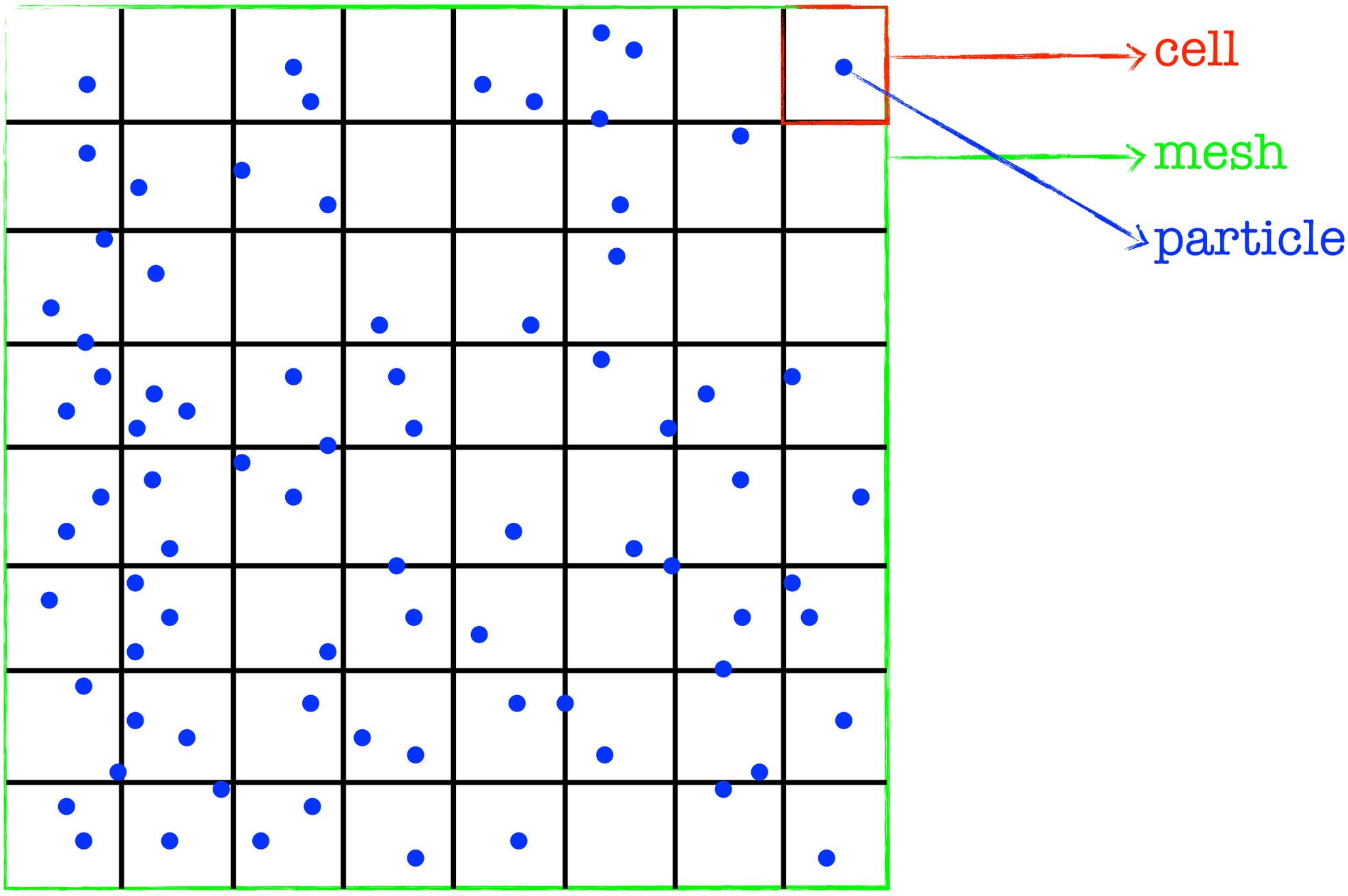
# This and the Next Lectures

- A quick recap of the essentials of the particle mesh N-body simulation technique
- A simplified description of the relaxation method commonly used to solve elliptical partial differential equations with given boundary conditions, and the adaptive mesh refinement (AMR) technique

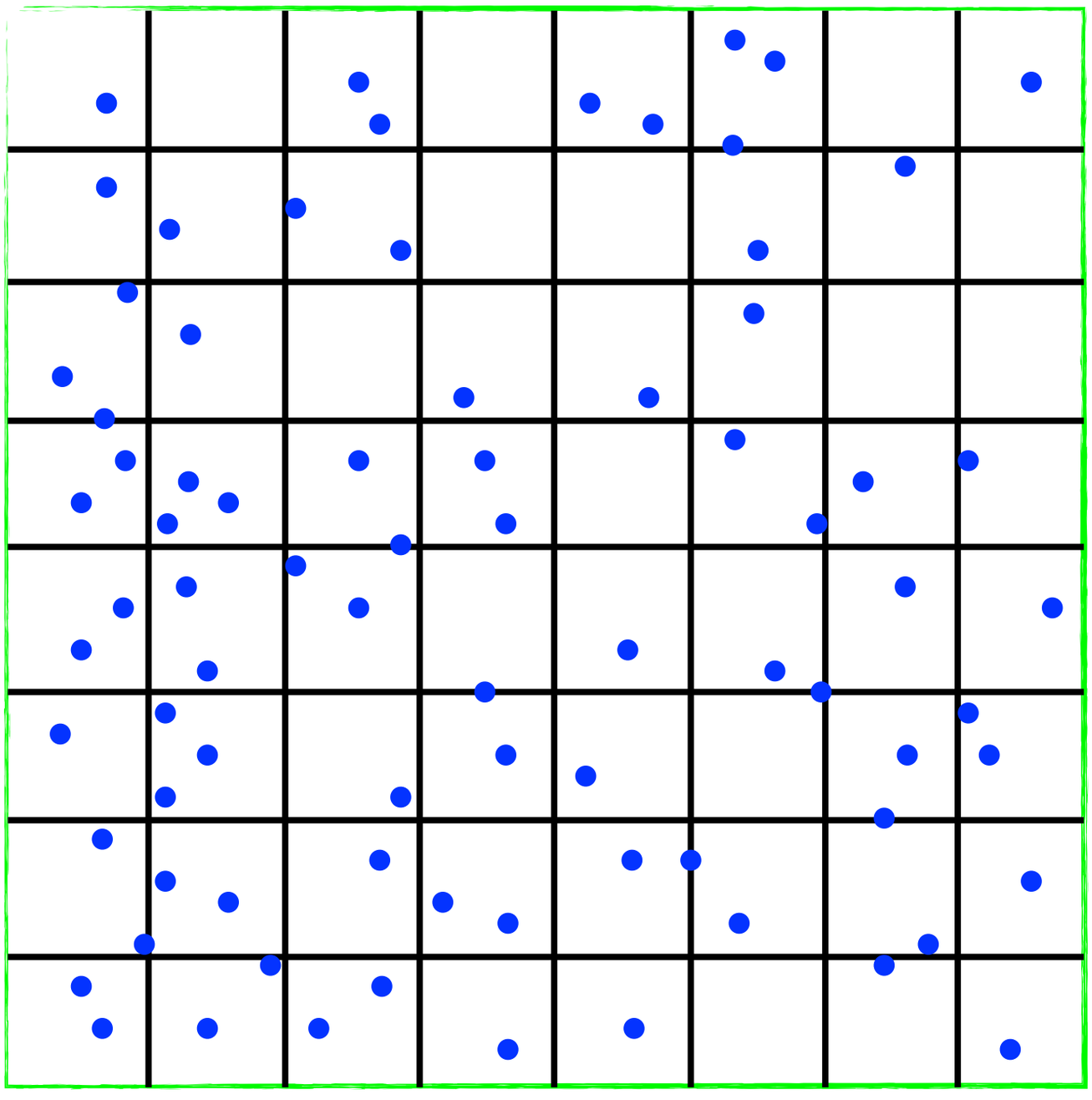
# Mesh-based N-body Simulation Techniques

- Particle based algorithms (covered in pre-school lectures) are suitable to calculate the standard gravitational force, for which an analytical force law ($1/r^2$) exists and doesn't depend on environments.
- The code can sum up forces produced by all particles. There are efficient algorithms for doing this.
- For the new force in non-standard models, mesh-based algorithms are more appropriate, because with them the highly nonlinear differential equations can be solved on meshes, and the new force can be evaluated on the mesh, bypassing any need for direct summation.

# Particle-Mesh (PM) Algorithms

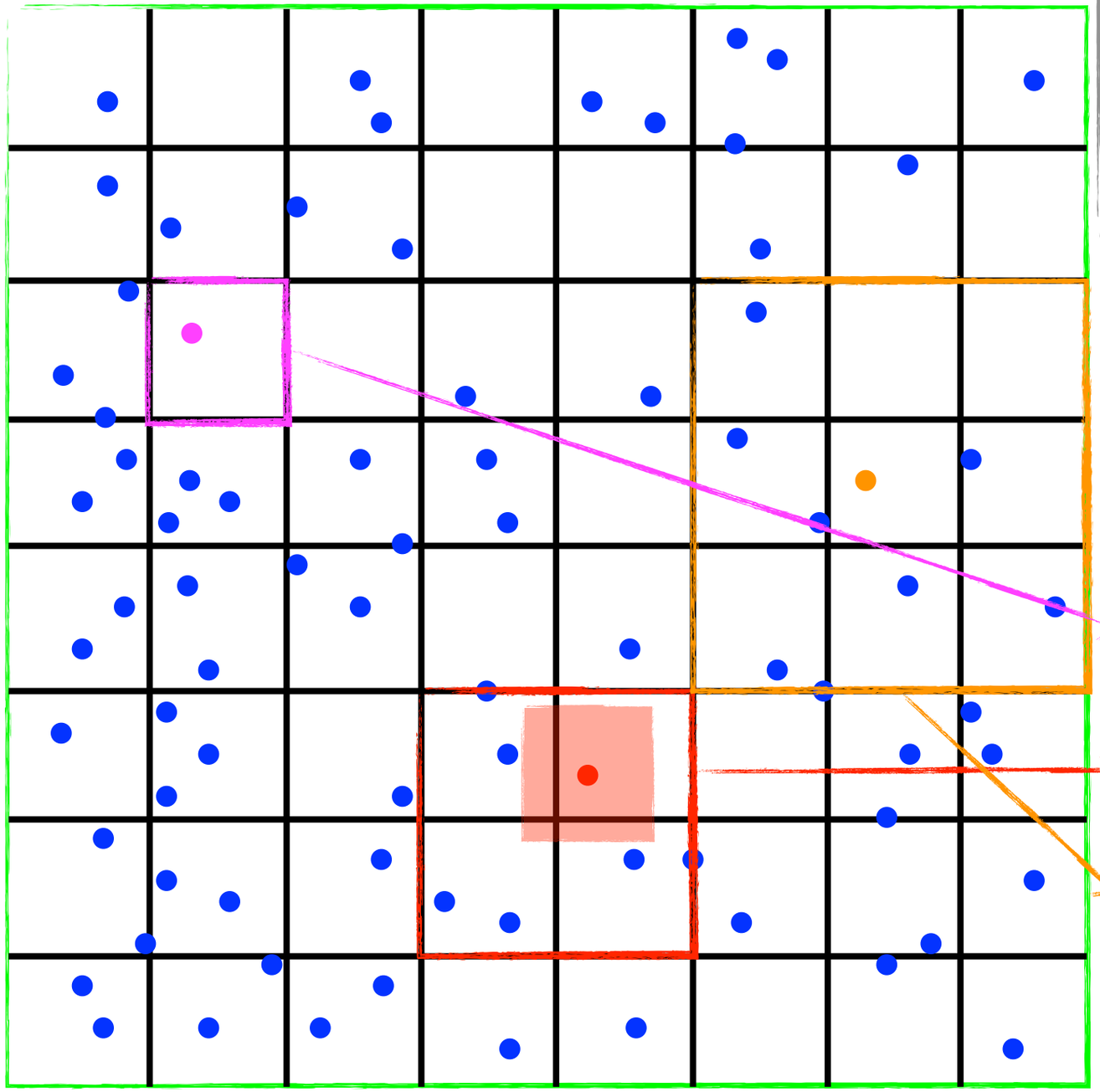


# Particle-Mesh (PM) Algorithms



particle IC

# Particle-Mesh (PM) Algorithms



particle IC

↓

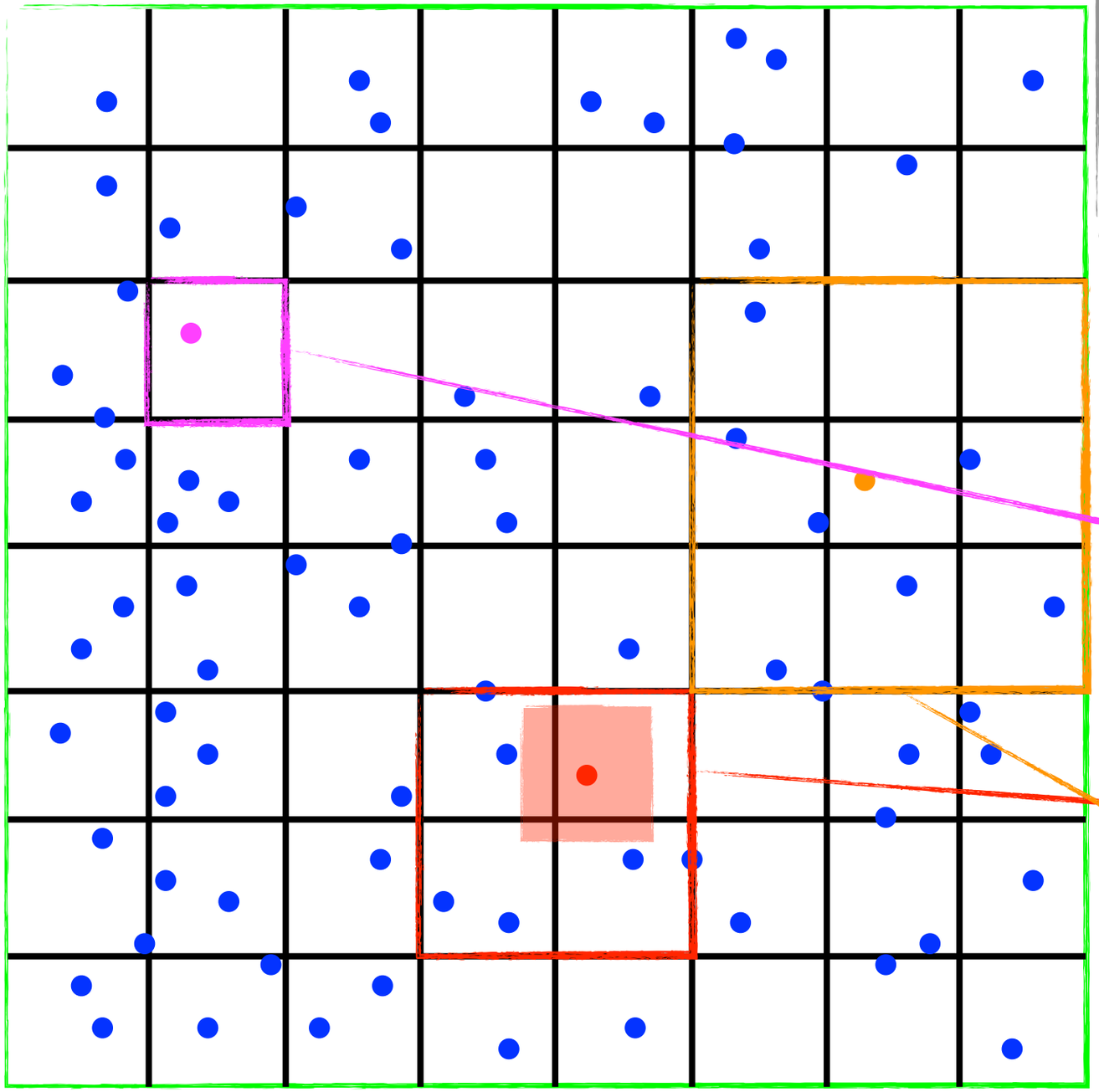
density assignment ($\varrho_{ijk}$)

NGP: Nearest Grid Point

**CIC: Cloud-in-Cell**

TSC: Triangle Shaped Cloud

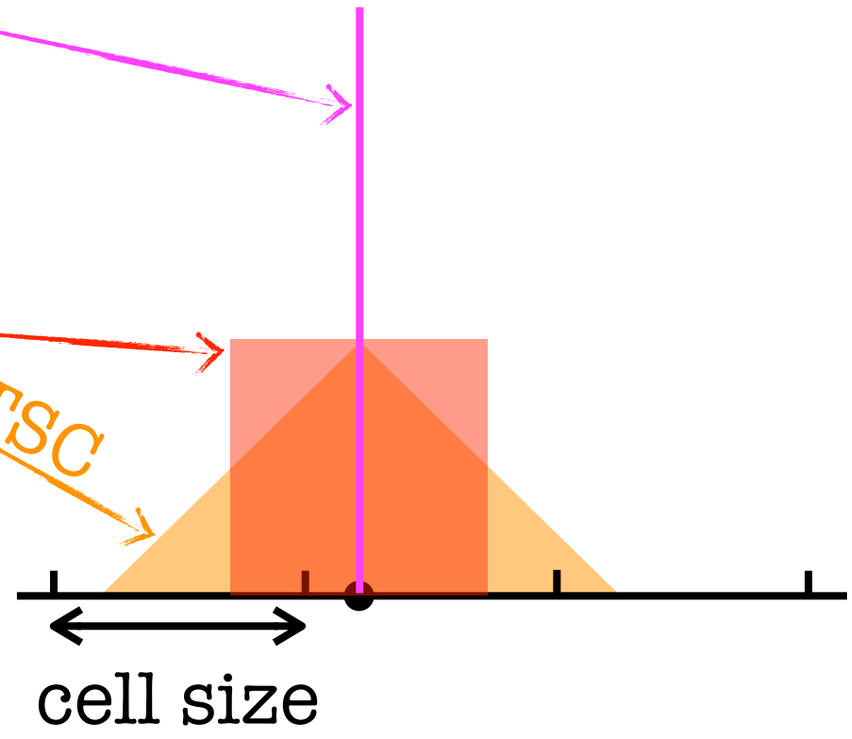
# Particle-Mesh (PM) Algorithms



particle IC

↓

density assignment ($\varrho_{ijk}$)



# Particle-Mesh (PM) Algorithms

particle IC

↓

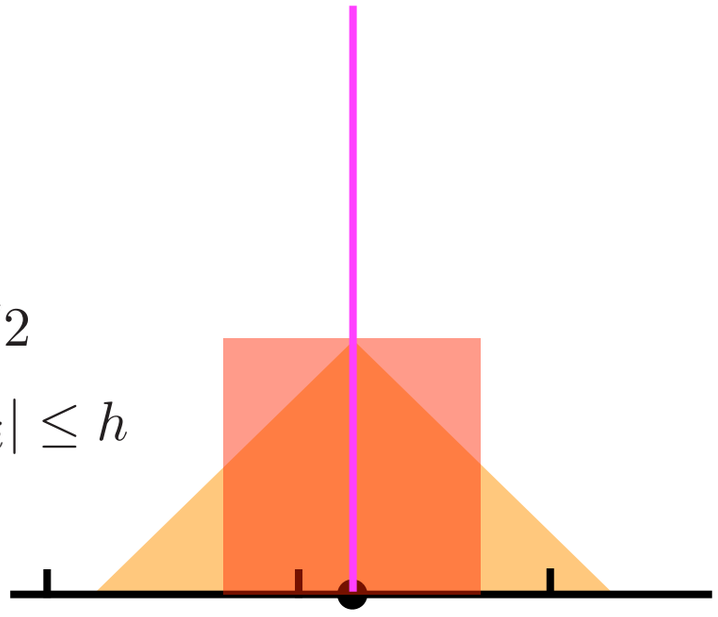
density assignment ($\varrho_{ijk}$)

Let $\Delta x_1, \Delta x_2, \Delta x_3$ be the distance of the **cell centre** to the **particle** along $x, y$ and $z$ direction respectively. Let $h$ be the cell size. Then $W(\Delta x_1, \Delta x_2, \Delta x_3) \equiv W_1(\Delta x_1) W_2(\Delta x_2) W_3(\Delta x_3)$ is the fraction of the considered particle's mass that is assigned to the considered cell, where $W_i(\Delta x_i)$ $(i = 1, 2, 3)$ is defined as

$$\text{NGP} : W_i(\Delta x_i) = \begin{cases} 1 & \text{if } |\Delta x_i| \leq h/2 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{CIC} : W_i(\Delta x_i) = \begin{cases} 1 - \frac{\Delta x_i}{h} & \text{if } |\Delta x_i| \leq h \\ 0 & \text{otherwise} \end{cases}$$

$$\text{TSC} : W_i(\Delta x_i) = \begin{cases} \frac{3}{4} - \frac{\Delta x_i^2}{h^2} & \text{if } |\Delta x_i| \leq h/2 \\ \frac{1}{2}\left(\frac{1}{2} - \frac{\Delta x_i}{h}\right)^2 & \text{if } h/2 < |\Delta x_i| \leq h \\ 0 & \text{otherwise} \end{cases}$$



# Particle-Mesh (PM) Algorithms

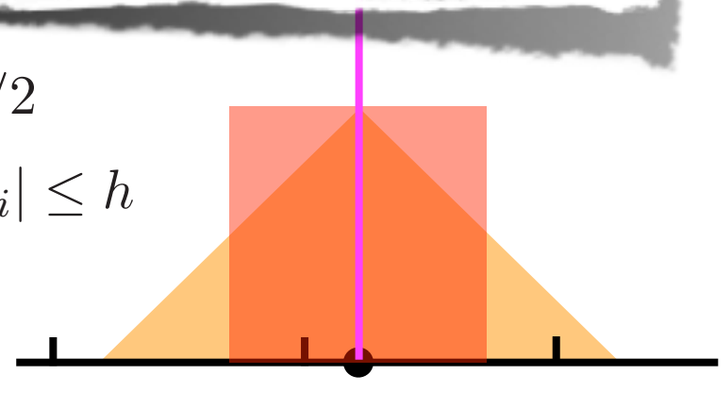
particle IC

Let $\Delta$ … Let
$h$ be … red
parti

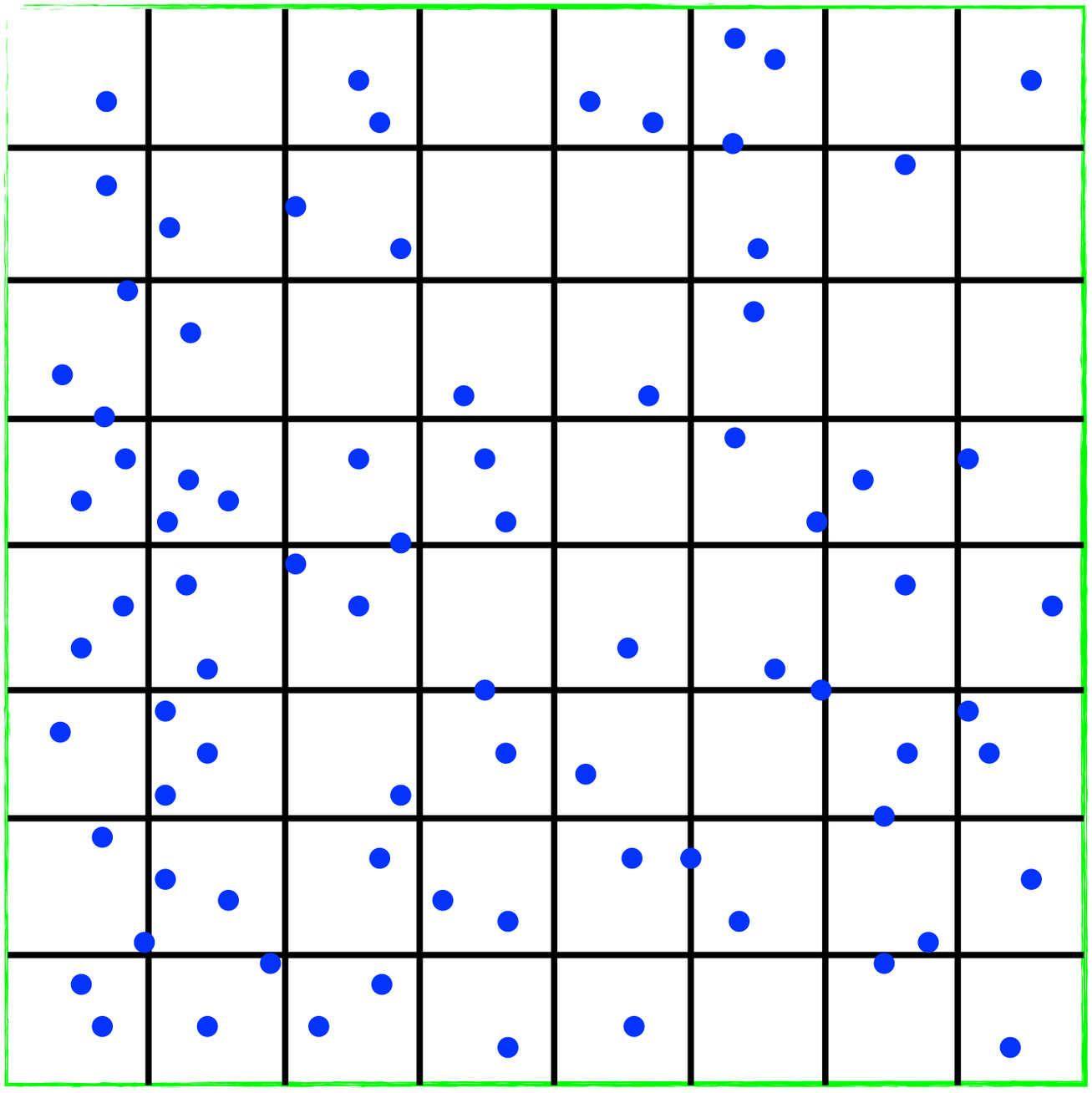
Lower order schemes (e.g., NGP): noisier density field

Higher order schemes (e.g., TSC, PCS): smoother density field allowing higher order Poisson solvers, but potential loss of small scale resolution.

$$\mathrm{TSC}: W_i(\Delta x_i) = \begin{cases} \frac{3}{4} - \frac{\Delta x_i^2}{h^2} & \text{if } |\Delta x_i| \leq h/2 \\ \frac{1}{2}\left(\frac{1}{2} - \frac{\Delta x_i}{h}\right)^2 & \text{if } h/2 < |\Delta x_i| \leq h \\ 0 & \text{otherwise} \end{cases}$$



# Particle-Mesh (PM) Algorithms



particle IC

↓

density assignment ($\varrho_{ijk}$)

↓

potential calculation ($\Phi_{ijk}$)

Fast Fourier Transform:
fast and efficient for linear differential equation

Relaxation:
works for general nonlinear differential equations

# Particle-Mesh (PM) Algorithms

$$\nabla^2\Phi = \frac{\partial^2\Phi}{\partial x^2} + \frac{\partial^2\Phi}{\partial y^2} + \frac{\partial^2\Phi}{\partial z^2} = 4\pi G\delta\rho$$

↓ internal unit!

$$\tilde{\nabla}^2\tilde{\Phi} = \frac{3}{2}a\Omega_m(\tilde{\rho}-1)$$

particle IC

↓

density assignment ($\varrho_{ijk}$)

↓

potential calculation ($\Phi_{ijk}$)

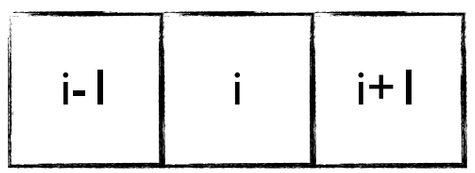
$$\frac{\partial\tilde{\Phi}}{\partial x} \rightarrow \frac{1}{2h}\left(\tilde{\Phi}_{i+1,j,k} - \tilde{\Phi}_{i-1,j,k}\right) \text{ or } \frac{1}{h}\left(\tilde{\Phi}_{i+1,j,k} - \tilde{\Phi}_{i,j,k}\right) \text{ or } \frac{1}{h}\left(\tilde{\Phi}_{i,j,k} - \tilde{\Phi}_{i-1,j,k}\right)$$

$$\frac{\partial^2\tilde{\Phi}}{\partial x^2} \rightarrow \frac{1}{h}\left[\left(\frac{\partial\tilde{\Phi}}{\partial x}\right)_{i+1,j,k} - \left(\frac{\partial\tilde{\Phi}}{\partial x}\right)_{i,j,k}\right]$$

$$\rightarrow \frac{1}{h}\left[\frac{1}{h}\left(\tilde{\Phi}_{i+1,j,k} - \tilde{\Phi}_{i,j,k}\right) - \frac{1}{h}\left(\tilde{\Phi}_{i,j,k} - \tilde{\Phi}_{i-1,j,k}\right)\right]$$

$$= \frac{1}{h^2}\left(\tilde{\Phi}_{i+1,j,k} - 2\tilde{\Phi}_{i,j,k} + \tilde{\Phi}_{i-1,j,k}\right)$$

discretisation example



# Particle-Mesh (PM) Algorithms

$$\nabla^2\Phi = \frac{\partial^2\Phi}{\partial x^2} + \frac{\partial^2\Phi}{\partial y^2} + \frac{\partial^2\Phi}{\partial z^2} = 4\pi G\delta\rho$$

↓ internal unit!

$$\tilde{\nabla}^2\tilde{\Phi} = \frac{3}{2}a\Omega_m(\tilde{\rho} - 1)$$

↓ discretisation

$$\frac{1}{h^2}\left[\tilde{\Phi}_{i+1,j,k} + \tilde{\Phi}_{i-1,j,k} + \tilde{\Phi}_{i,j+1,k} + \tilde{\Phi}_{i,j-1,k} + \tilde{\Phi}_{i,j,k+1} + \tilde{\Phi}_{i,j,k-1} - 6\tilde{\Phi}_{i,j,k}\right] = \frac{3}{2}a\Omega_m\left(\tilde{\rho}_{i,j,k} - 1\right)$$

↓ Fourier transform

$$\tilde{\Phi}(\mathbf{k}) = -\frac{3}{8}a\Omega_m\left[\sin^2\frac{k_x}{2} + \sin^2\frac{k_y}{2} + \sin^2\frac{k_z}{2}\right]^{-1}\tilde{\delta}_{i,j,k}(\mathbf{k});\quad \mathbf{k} = (k_x, k_y, k_z) = \frac{2\pi}{N_c}(l, m, n)$$

↓ inverse Fourier transform

$$\tilde{\Phi}_{i,j,k}$$

particle IC

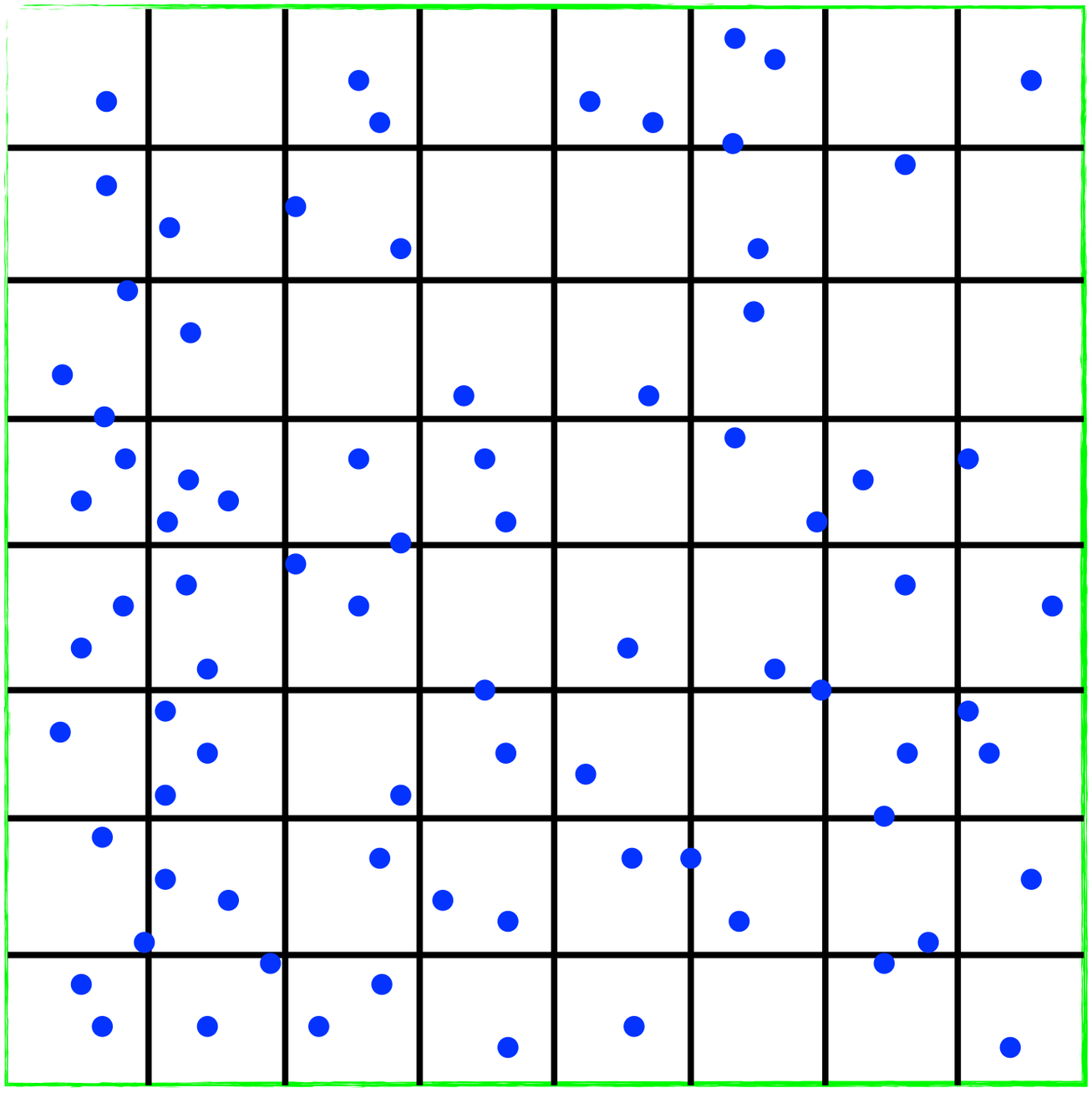
↓

density assignment ($\varrho_{ijk}$)

↓

potential calculation ($\Phi_{ijk}$)

# Particle-Mesh (PM) Algorithms



particle IC

↓

density assignment ($\varrho_{ijk}$)

↓

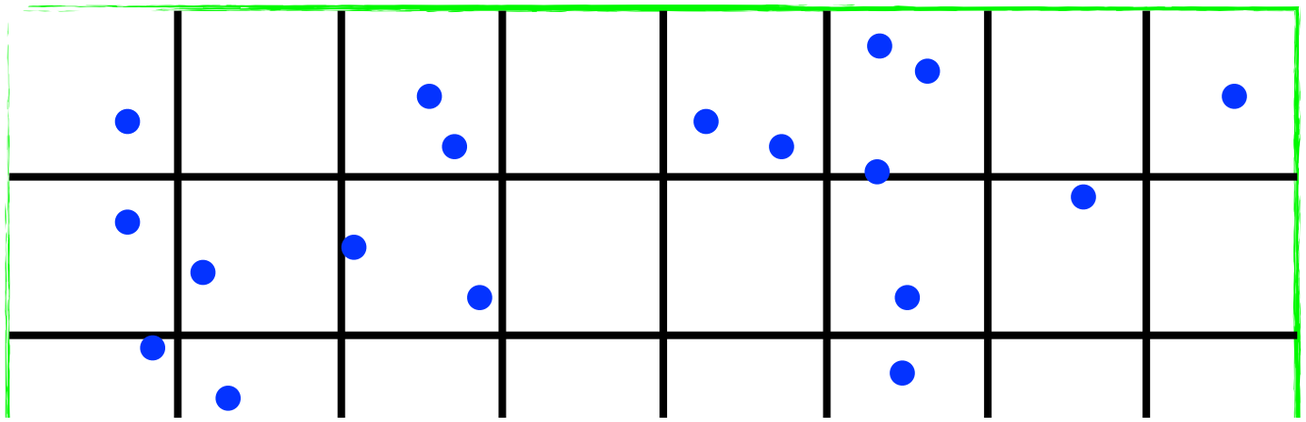
potential calculation ($\Phi_{ijk}$)

↓

force calculation

$$\frac{\partial \tilde{\Phi}}{\partial x} \rightarrow \frac{1}{2h}\left(\tilde{\Phi}_{i+1,j,k} - \tilde{\Phi}_{i-1,j,k}\right)$$

# Particle-Mesh (PM) Algorithms



Note: the equation on the right only gives the forces at cell centres, and what we need is the force at particle position. To get the latter one has to interpolate from the particle centre. The interpolation algorithm must be the same as what is used for density assignment (NGP, CIC, TSC, ...): otherwise the simulation won't conserve momentum!

particle IC

↓

density assignment ($\varrho_{ijk}$)

↓

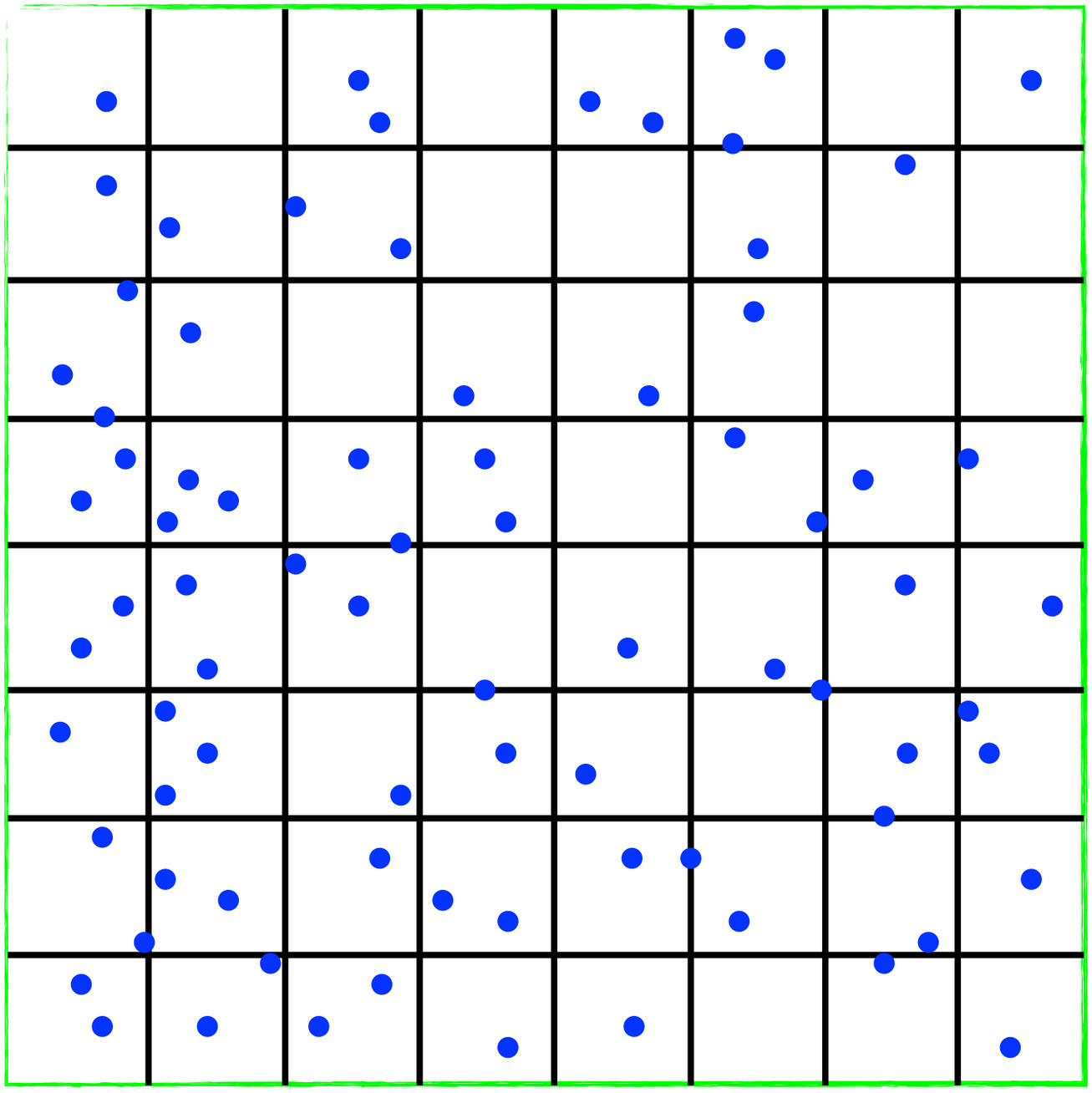
potential calculation ($\Phi_{ijk}$)

↓

force calculation

$$\frac{\partial\tilde{\Phi}}{\partial x} \rightarrow \frac{1}{2h}\left(\tilde{\Phi}_{i+1,j,k} - \tilde{\Phi}_{i-1,j,k}\right)$$

# Particle-Mesh (PM) Algorithms



particle IC

↓

density assignment ($\varrho_{ijk}$)

↓

potential calculation ($\Phi_{ijk}$)

↓

force calculation

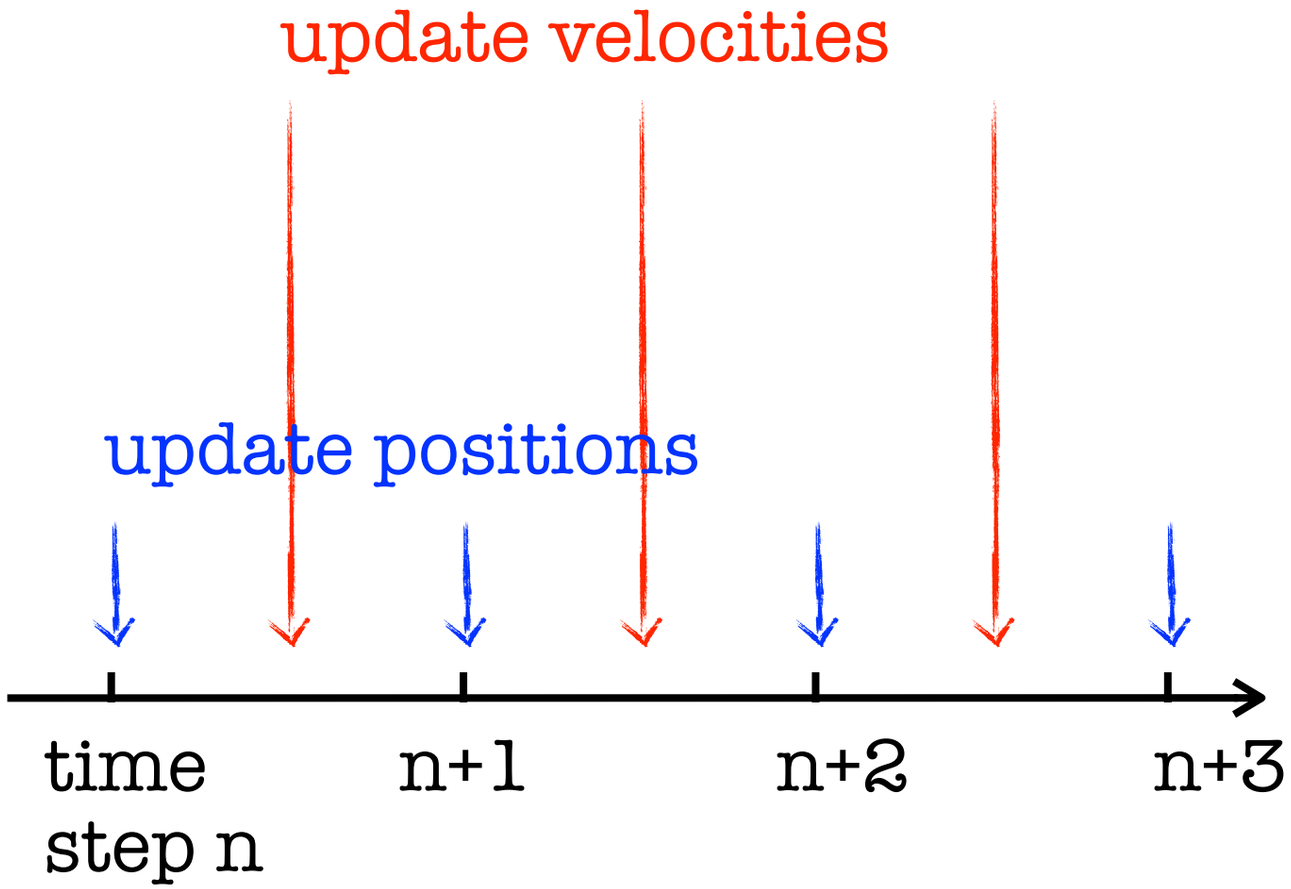
↓

particle movement

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \frac{\mathbf{p}}{a^2}$$

$$\frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = -\frac{1}{a}\nabla_{\mathbf{x}}\Phi$$

# Particle-Mesh (PM) Algorithms




particle IC

↓

density assignment ($\varrho_{ijk}$)

↓

potential calculation ($\Phi_{ijk}$)

↓

force calculation

↓

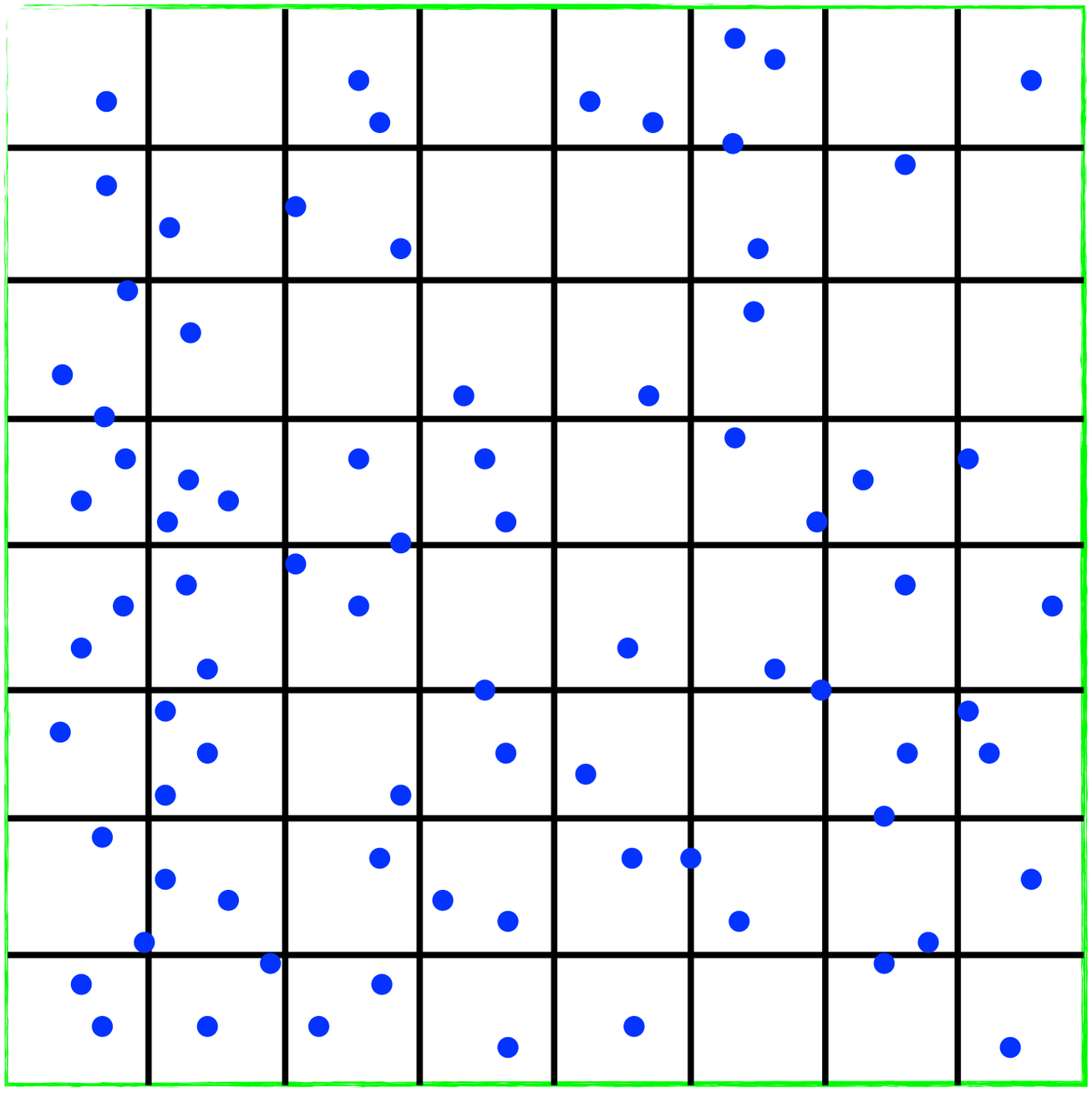
particle movement

leapfrog scheme: shifted integrations for position and velocity, for 2nd order accuracy

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \frac{\mathbf{p}}{a^2}$$

$$\frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = -\frac{1}{a}\nabla_{\mathbf{x}}\Phi$$

# Particle-Mesh (PM) Algorithms



particle IC

↓

density assignment ($\varrho_{ijk}$)

↓

potential calculation ($\Phi_{ijk}$)

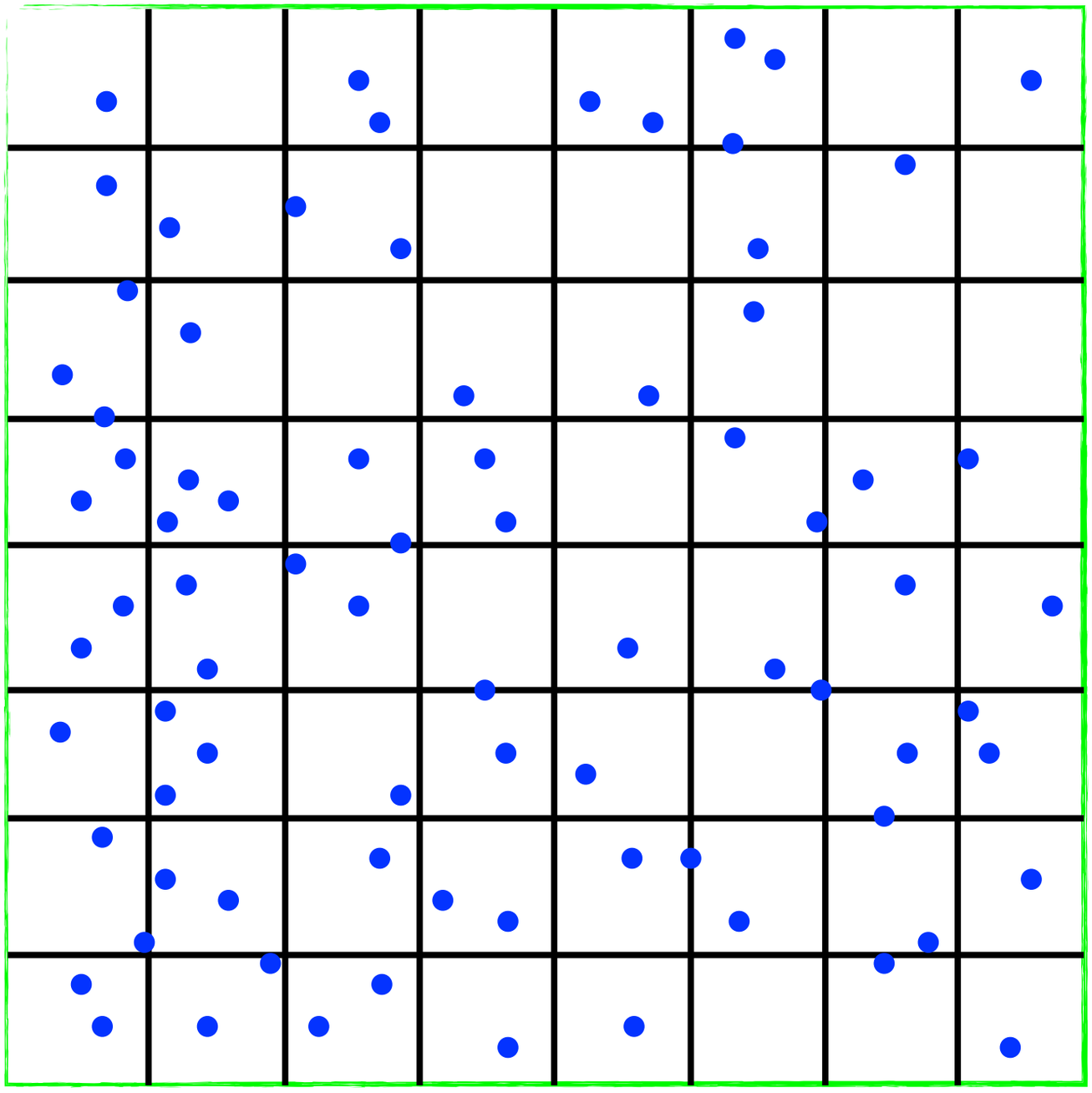
↓

force calculation

↓

particle movement

# Particle-Mesh (PM) Algorithms



particle IC

↓

density assignment ($\varrho_{ijk}$)

↓

potential calculation ($\Phi_{ijk}$)

↓

force calculation

↓

particle movement

↓

final result

# Why the PM Algorithm is not Enough

- PM method usually employs FFT to solve the gravitational potential
- With standard libraries such as FFTW, this has become very efficient
- However, at mesh cell size, force calculation can be artificially anisotropic because of the geometry (cubic) of the cell
- Also, with a uniform mesh resolution, if the resolution is high enough for high density regions, it is a waste of computing resources in low density regions which have fewer particles; if the resolution is just enough in low density regions, it will be too low for high density regions.
- There are ways to overcome these problems.

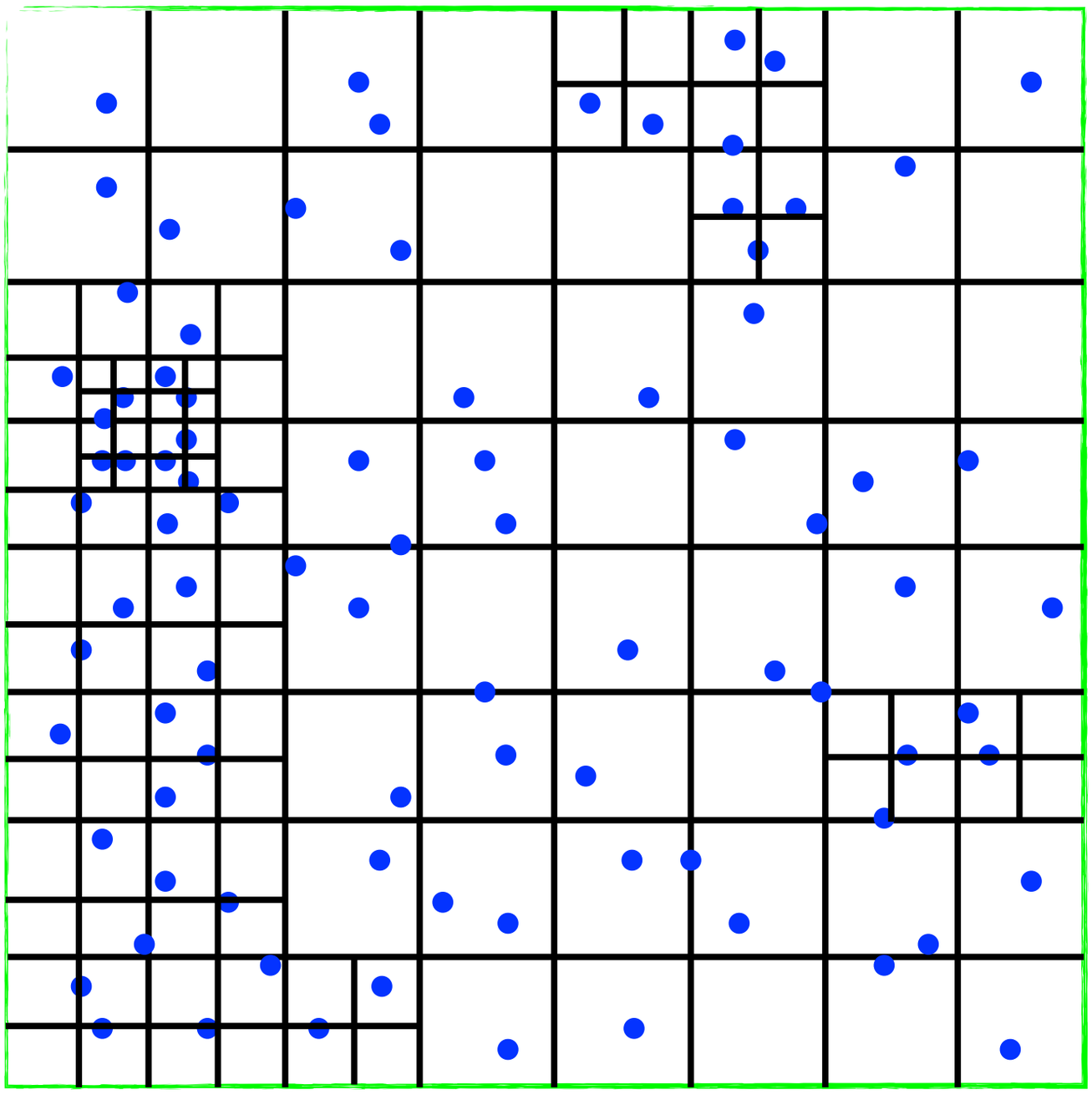
# Particle-Particle-Particle-Mesh (P3M) Method

- Use the PM method to calculate long range force
- Supplement this with a direct summation of short-range forces at cell size scales
- It is clear that this method suffers from the same problem of any direct summation method, namely it cannot be applied to our nonstandard models, because there is no universal force law to sum!

# Adaptive Mesh Refinement (AMR) Method

- Use a uniform mesh to cover the whole simulation domain
- But in higher density regions, place finer meshes (usually with half of the original cell size) to get higher resolution
- In even higher density regions, place an even finder mesh, and so on
- The trigger of a higher (or finer) refinement level is often (though not always, depending on the specific problem) chosen as the density value in a cell: if the density exceeds a preset refinement threshold, then that cell is split into 8 son cells, and so on.

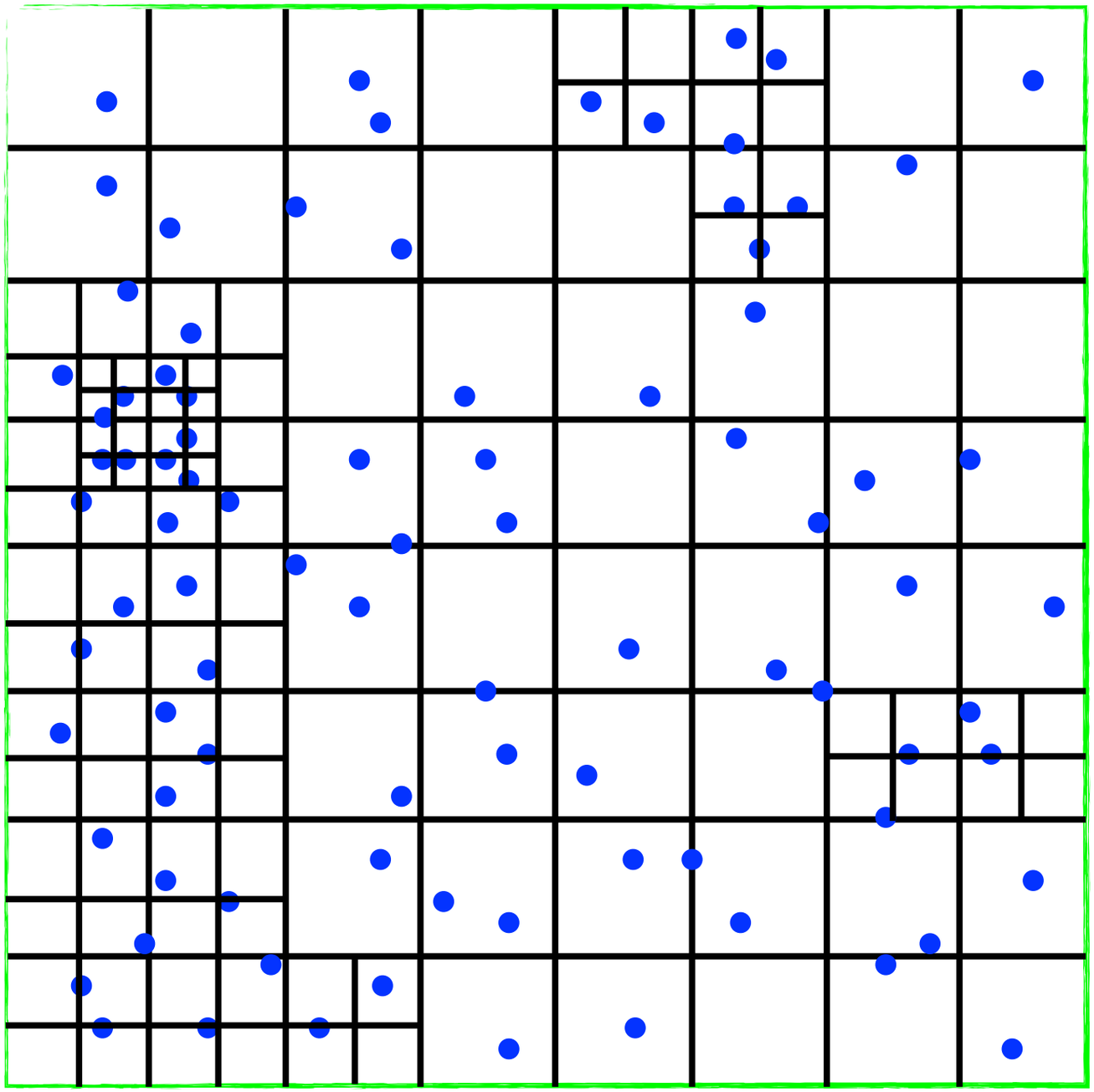
# Adaptive Mesh Refinement (AMR) Method



Pro: only use high mesh resolution where needed, so more efficient

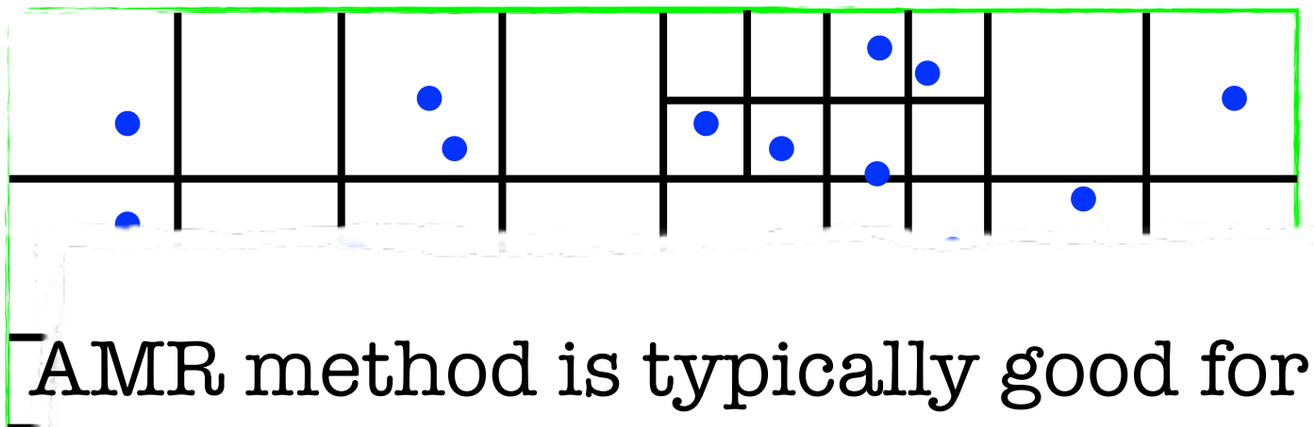
Con: more complicated. The refined meshes can have arbitrary shapes mimicking the geometry of the high-density regions, so FFT no longer applies, and some sort of relaxation method is needed to solve the gravitational potential

# Adaptive Mesh Refinement (AMR) Method



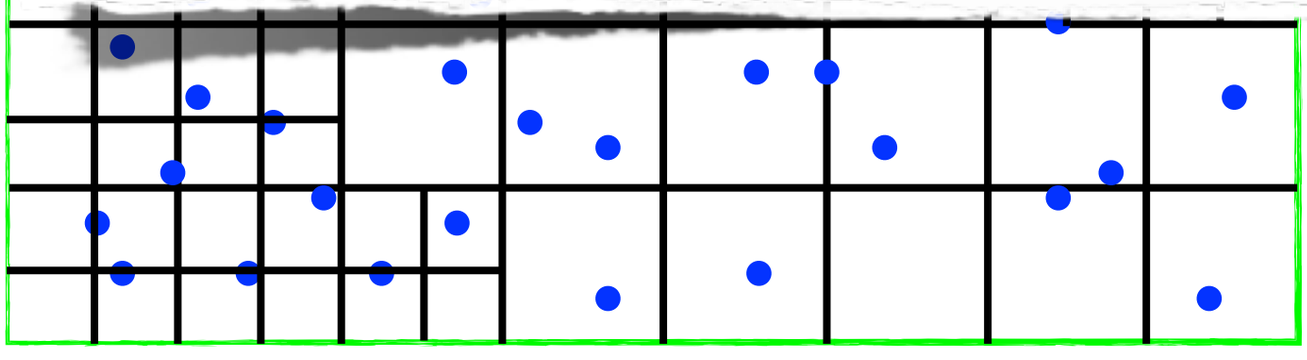
However, this is not a problem for nonstandard models, because there the equations that need to be solved are usually highly nonlinear, for which the FFT method does not apply (at least does not apply easily), and so relaxation is needed anyway!

# Adaptive Mesh Refinement (AMR) Method



However, this is not a problem for nonstandard models, because there the

AMR method is typically good for calculating the density field (e.g., matter power spectrum). However, before refined mesh is created the force resolution is low, and this is not good for reproducing low-mass haloes. You have to make it very easy to trigger mesh refinement to get good statistics of low mass haloes - this can make the code very slow!

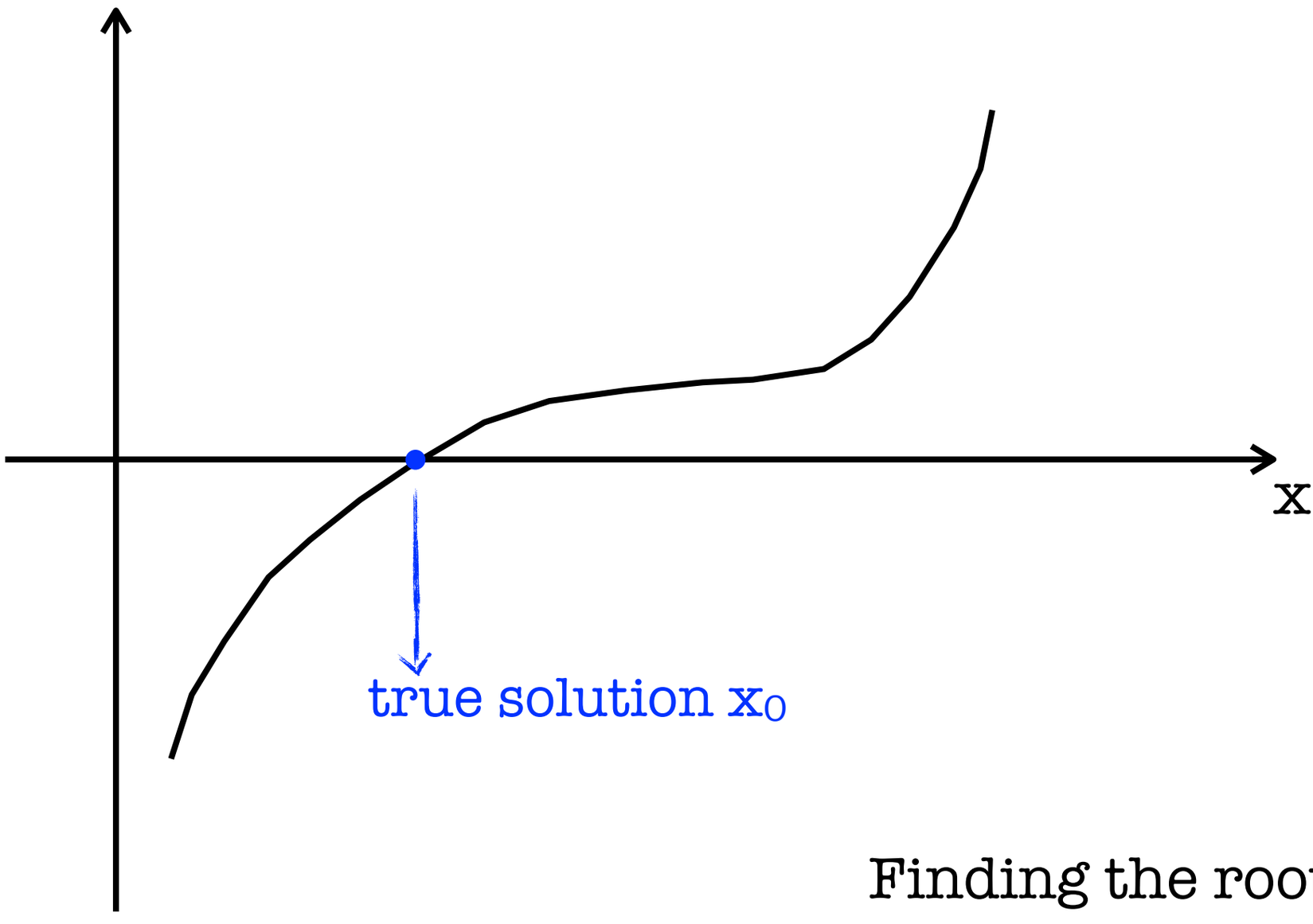


So, what is relaxation?

# Relaxation Method

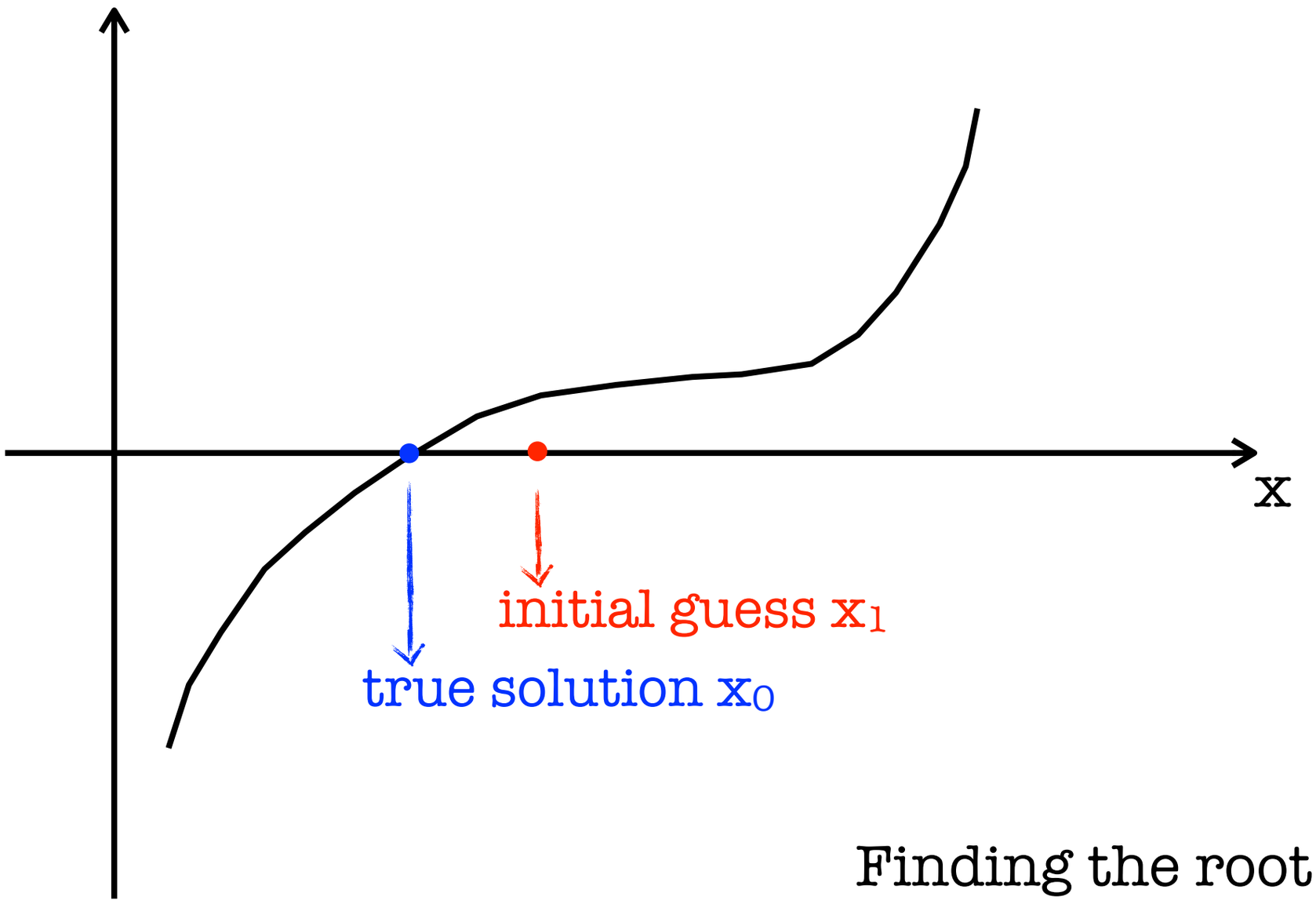
- An iterative method in which one starts with some guess of the solution to an equation, and improves this guess repeatedly until getting close enough to the true solution

# Analogy to the Newton Method



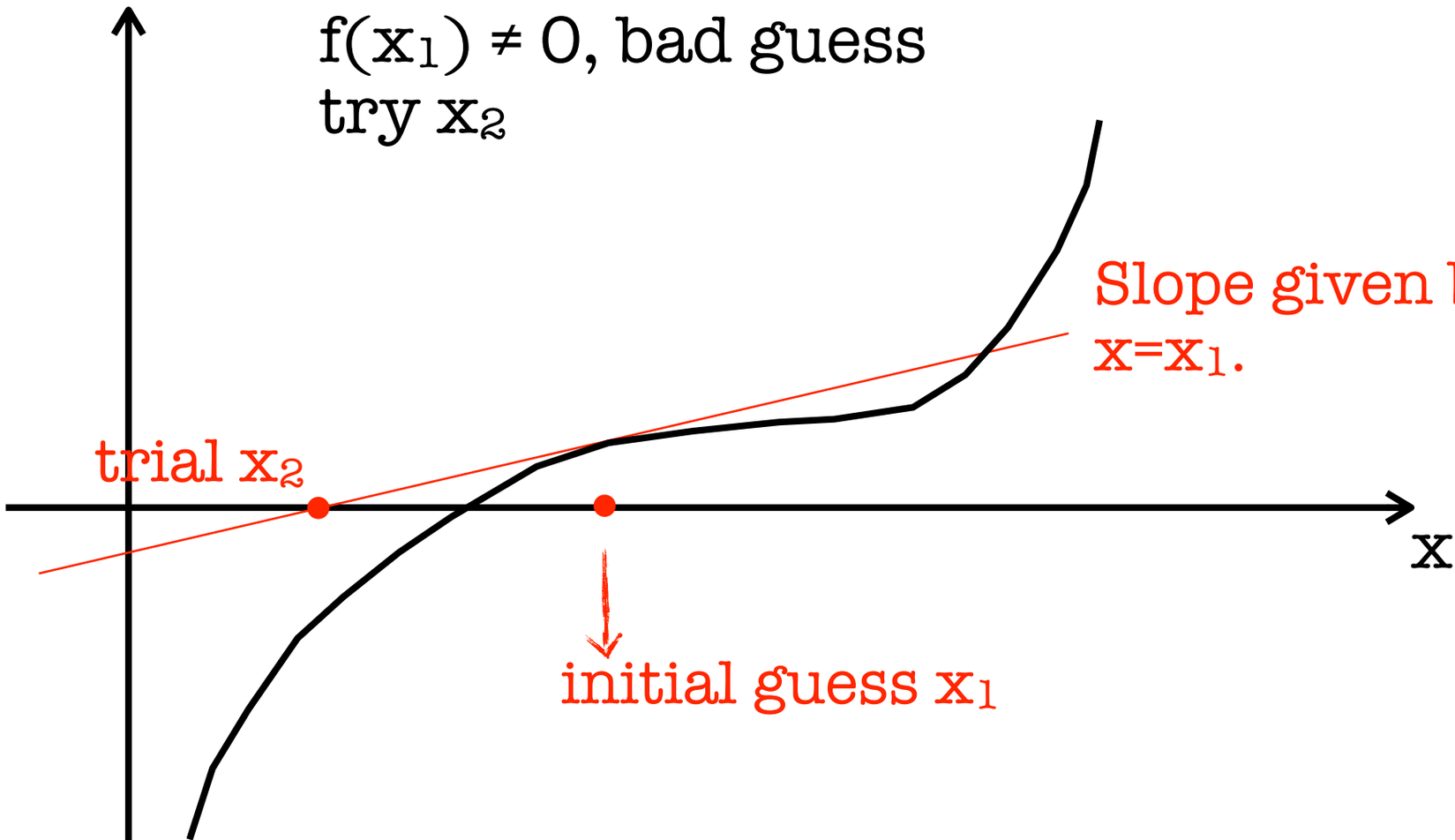

Finding the root of f(x)=0

# Analogy to the Newton Method




Finding the root of f(x)=0

# Analogy to the Newton Method

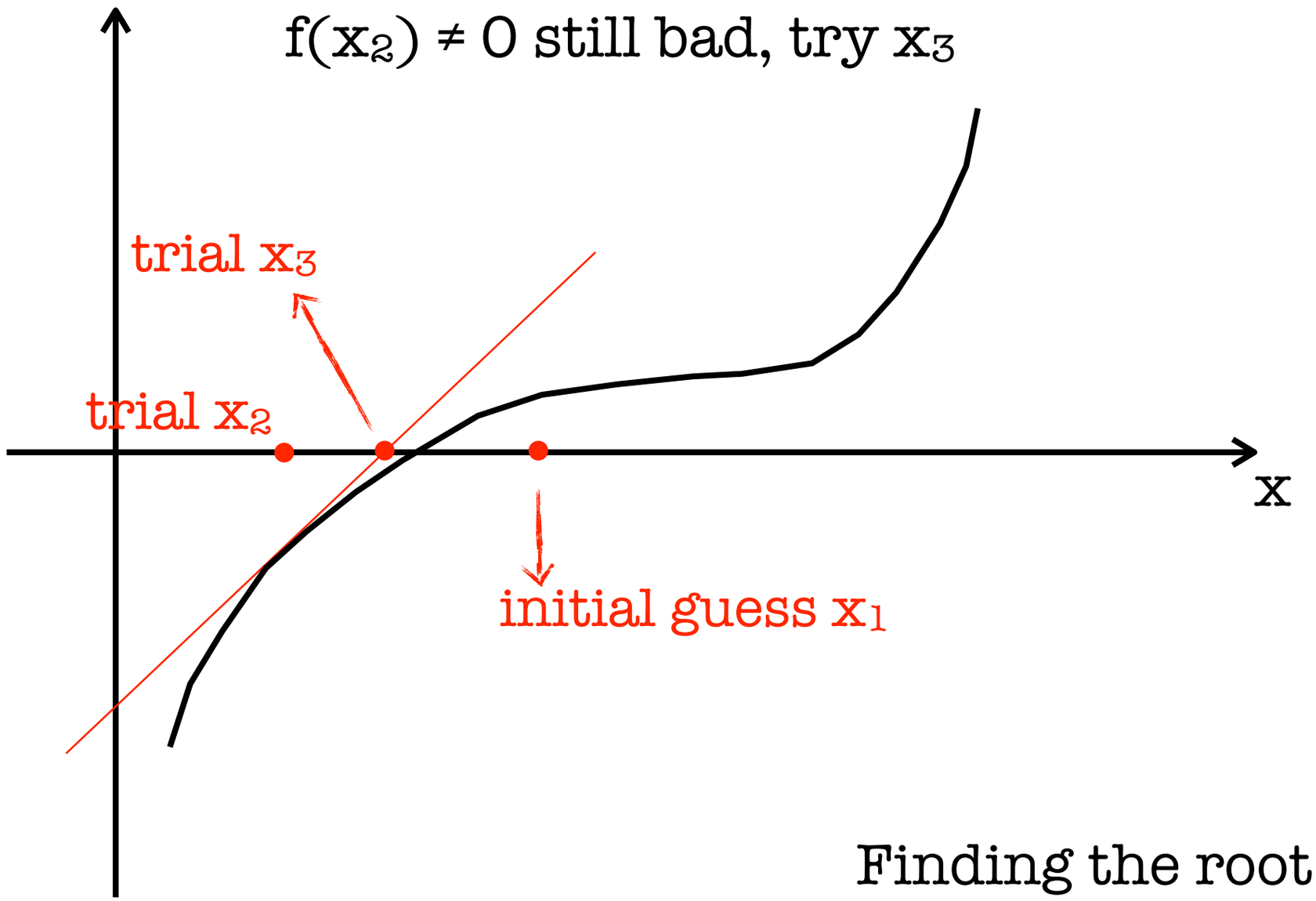



Finding the root of f(x)=0

$x_{new} = x_{old} - f(x_{old})/[df(x_{old})/dx]$

$x_2 = x_1 - f(x_1)/[df(x_1)/dx]$

# Analogy to the Newton Method

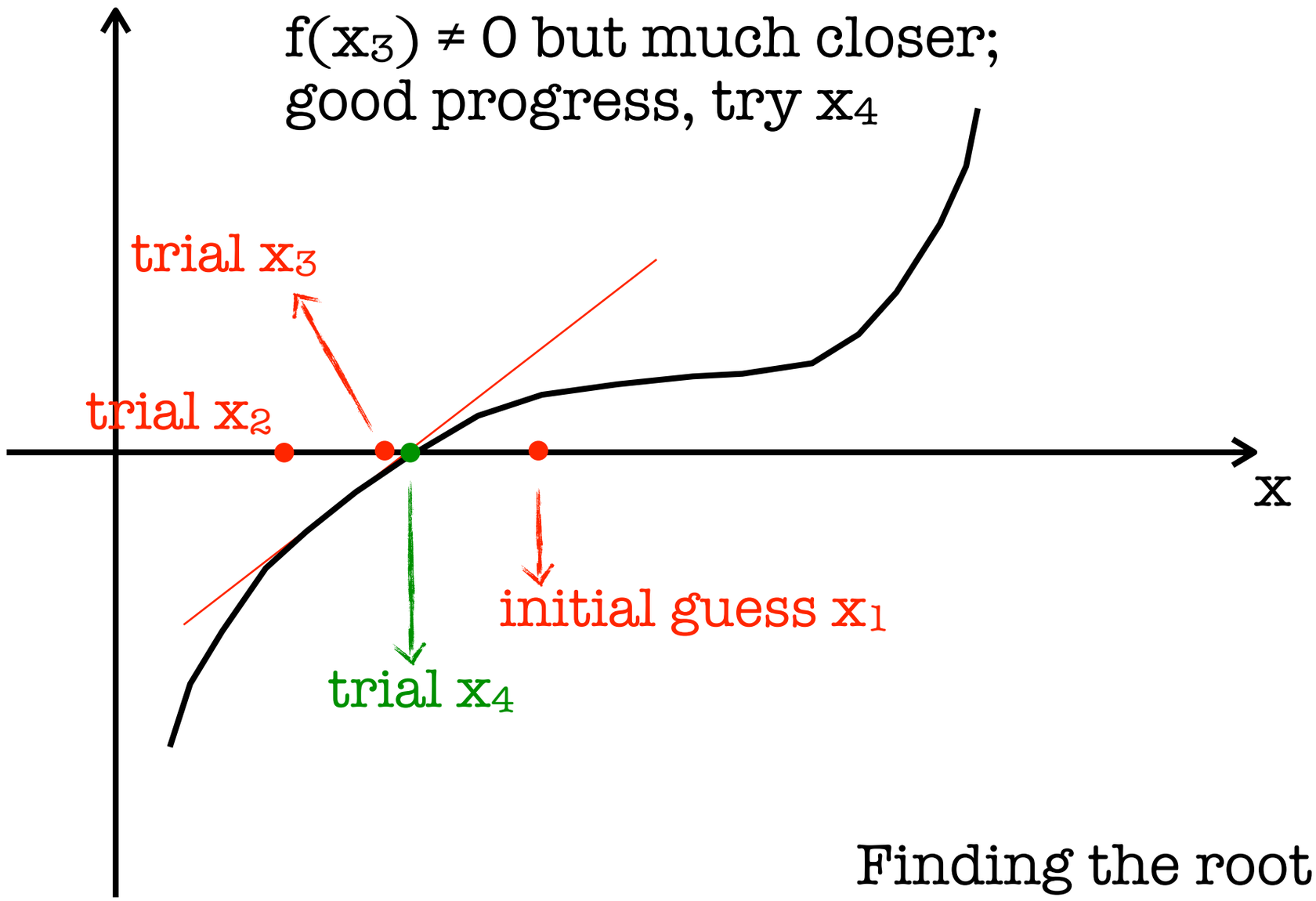



Finding the root of f(x)=0

$x_{new} = x_{old} - f(x_{old})/[df(x_{old})/dx]$

$x_3 = x_2 - f(x_2)/[df(x_2)/dx]$

# Analogy to the Newton Method

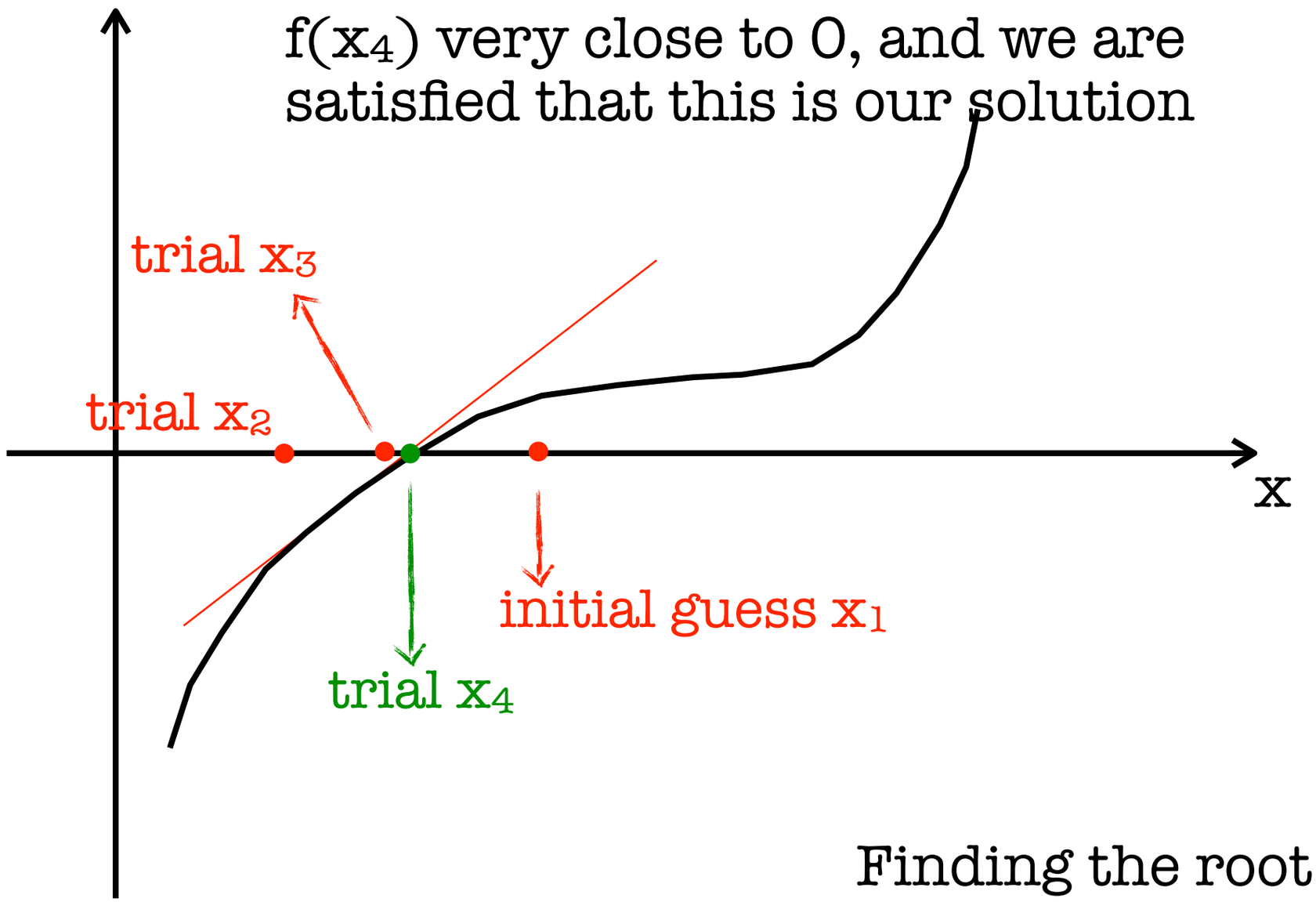



Finding the root of f(x)=0

$x_{new} = x_{old} - f(x_{old})/[df(x_{old})/dx]$

$x_4 = x_3 - f(x_3)/[df(x_3)/dx]$

# Analogy to the Newton Method

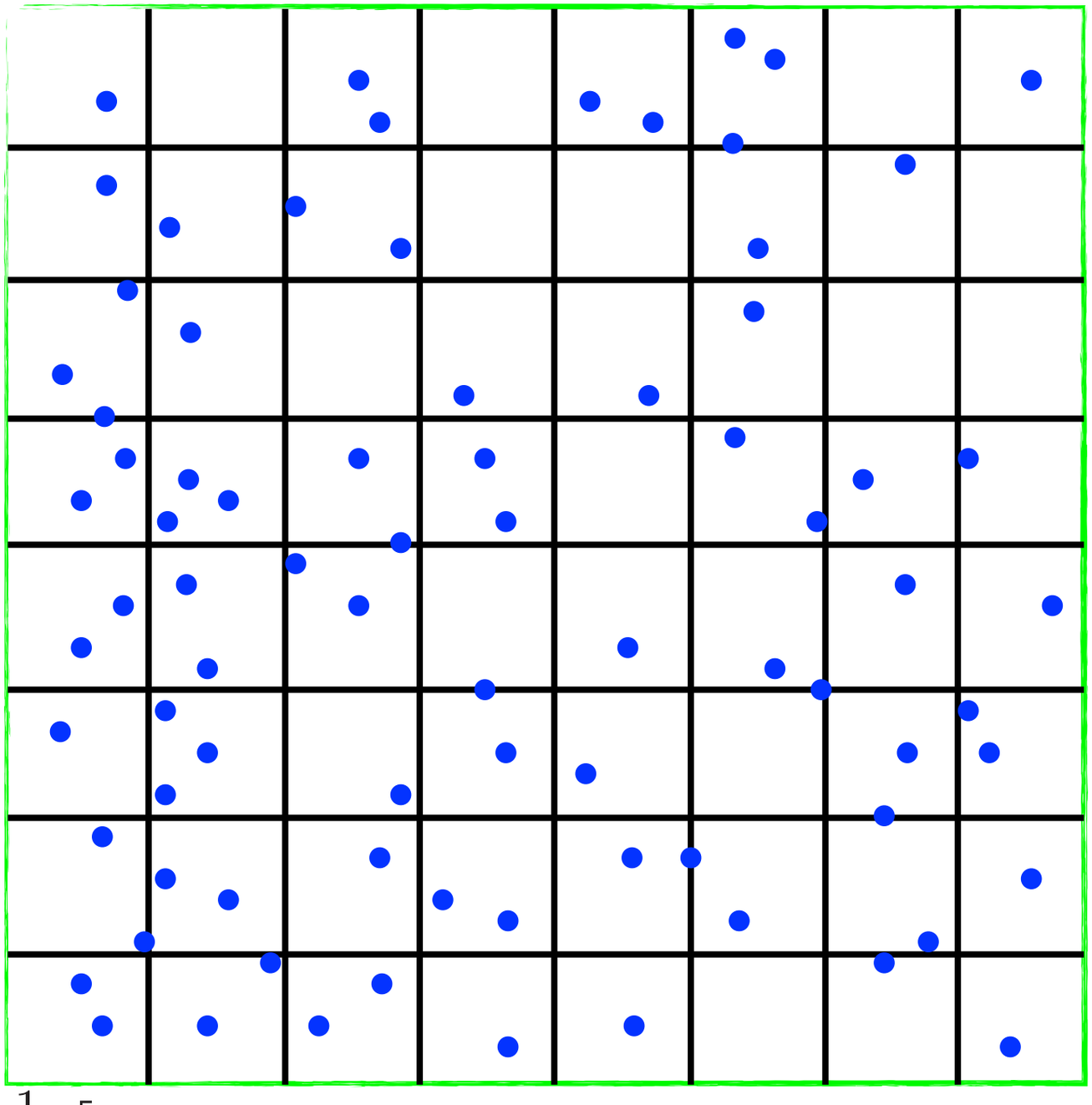



Finding the root of f(x)=0

$x_{new} = x_{old} - f(x_{old})/[df(x_{old})/dx]$

$x_4 = x_3 - f(x_3)/[df(x_3)/dx]$

# Relaxation on a Mesh



We have a value $\varrho_{ijk}$ at (the centre of) each cell, and want to find the solution $\Phi_{ijk}$ to the (algebraic rather than differential!) equation below. The difference from the simple example above is we must find $N^3$ values for $\Phi_{ijk}$ (one at each cell centre)

$$F\left(\tilde{\Phi}_{i,j,k}\right) \equiv f\left(\tilde{\Phi}_{i,j,k}\right) - \frac{3}{2}a\Omega_m\tilde{\rho}_{i,j,k} = 0$$

$$\frac{1}{h^2}\left[\tilde{\Phi}_{i+1,j,k} + \tilde{\Phi}_{i-1,j,k} + \tilde{\Phi}_{i,j+1,k} + \tilde{\Phi}_{i,j-1,k} + \tilde{\Phi}_{i,j,k+1} + \tilde{\Phi}_{i,j,k-1} - 6\tilde{\Phi}_{i,j,k}\right] = \frac{3}{2}a\Omega_m\left(\tilde{\rho}_{i,j,k} - 1\right)$$

# Relaxation on a Mesh

Initial guess of $\Phi_{ijk}$ in each cell

$$\tilde{\Phi}^{\rm new}_{i,j,k} = \tilde{\Phi}^{\rm old}_{i,j,k} - \frac{F\left(\tilde{\Phi}^{\rm old}_{i,j,k}\right)}{\partial F\left(\tilde{\Phi}^{\rm old}_{i,j,k}\right)/\partial\tilde{\Phi}_{i,j,k}}$$

N

Is $F(\Phi_{ijk})$ enough close to 0?

$x_{new} = x_{old} - f(x_{old})/[df(x_{old})/dx]$

Y

End of solution

$$F\left(\tilde{\Phi}_{i,j,k}\right) \equiv f\left(\tilde{\Phi}_{i,j,k}\right) - \frac{3}{2}a\Omega_m\tilde{\rho}_{i,j,k} = 0$$

$$\frac{1}{h^2}\left[\tilde{\Phi}_{i+1,j,k} + \tilde{\Phi}_{i-1,j,k} + \tilde{\Phi}_{i,j+1,k} + \tilde{\Phi}_{i,j-1,k} + \tilde{\Phi}_{i,j,k+1} + \tilde{\Phi}_{i,j,k-1} - 6\tilde{\Phi}_{i,j,k}\right] = \frac{3}{2}a\Omega_m\left(\tilde{\rho}_{i,j,k} - 1\right)$$

# Relaxation on a Mesh

Initial guess of $\Phi_{ijk}$ in each cell

$$\tilde{\Phi}^{\rm new}_{i,j,k} = \tilde{\Phi}^{\rm old}_{i,j,k} - \frac{F\left(\tilde{\Phi}^{\rm old}_{i,j,k}\right)}{\partial F\left(\tilde{\Phi}^{\rm old}_{i,j,k}\right)/\partial\tilde{\Phi}_{i,j,k}}$$

N

Is $F(\Phi_{ijk})$ enough close to 0?

Y

End of solution

F can be linear AND nonlinear!

$$F\left(\tilde{\Phi}_{i,j,k}\right) \equiv f\left(\tilde{\Phi}_{i,j,k}\right) - \frac{3}{2}a\Omega_m\tilde{\rho}_{i,j,k} = 0$$

F is linear in this special case

$$\frac{1}{h^2}\left[\tilde{\Phi}_{i+1,j,k} + \tilde{\Phi}_{i-1,j,k} + \tilde{\Phi}_{i,j+1,k} + \tilde{\Phi}_{i,j-1,k} + \tilde{\Phi}_{i,j,k+1} + \tilde{\Phi}_{i,j,k-1} - 6\tilde{\Phi}_{i,j,k}\right] = \frac{3}{2}a\Omega_m\left(\tilde{\rho}_{i,j,k} - 1\right)$$

# Convergence Criteria

In the Newton method analogy above, convergence can be deemed to be achieved if $f(x_{trial})<\varepsilon$ with $\varepsilon$ being a pre-chosen small number. This is simple because there is only one value of f(x).

$F(\Phi_{ijk})$ takes different values in difference cells of the mesh, so what does convergence mean here?

Convergence can be defined in various ways similar to the Newton analogy.

$$F\left(\tilde{\Phi}_{i,j,k}\right) \equiv f\left(\tilde{\Phi}_{i,j,k}\right) - \frac{3}{2}a\Omega_m\tilde{\rho}_{i,j,k} = 0$$

Is $F(\Phi_{ijk})$ enough close to 0?

residual

$$\left[\frac{1}{N}\sum_{i,j,k}F^2(\tilde{\Phi}_{i,j,k})\right]^{1/2} \leq \epsilon$$

Convergence criterion 1:
residual smaller than some pre-set threshold

# Convergence Criteria

But this is not the only criterion of convergence. In some cases, it is incredibly hard to et the residual smaller than some threshold, so the above condition will perhaps never be satisfied.

However, note that we are solving a discrete version of a continuous equation, and the discretisation itself causes error, which is called the truncation error. This is something we cannot control and cannot beat!

$$F\left(\tilde{\Phi}_{i,j,k}\right) \equiv f\left(\tilde{\Phi}_{i,j,k}\right) - \frac{3}{2} a \Omega_m \tilde{\rho}_{i,j,k} = 0$$

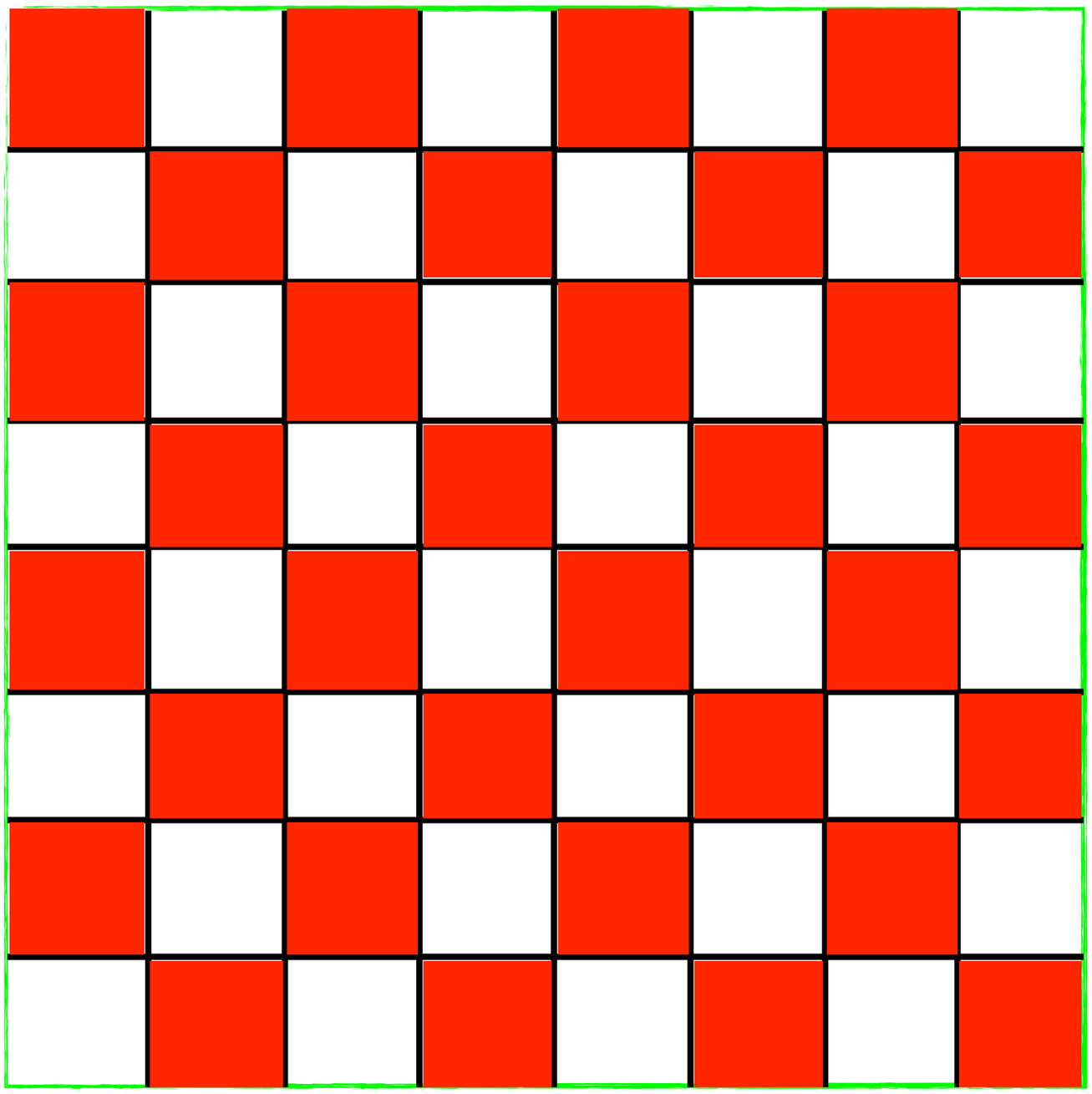
Is $F(\Phi_{ijk})$ enough close to 0?

residual

$$\left[\frac{1}{N}\sum_{i,j,k} F^2(\tilde{\Phi}_{i,j,k})\right]^{1/2} \leq \text{trunc. err.}/3$$

Convergence criterion 2:
residual smaller than the truncation error

# Order to Update Cells



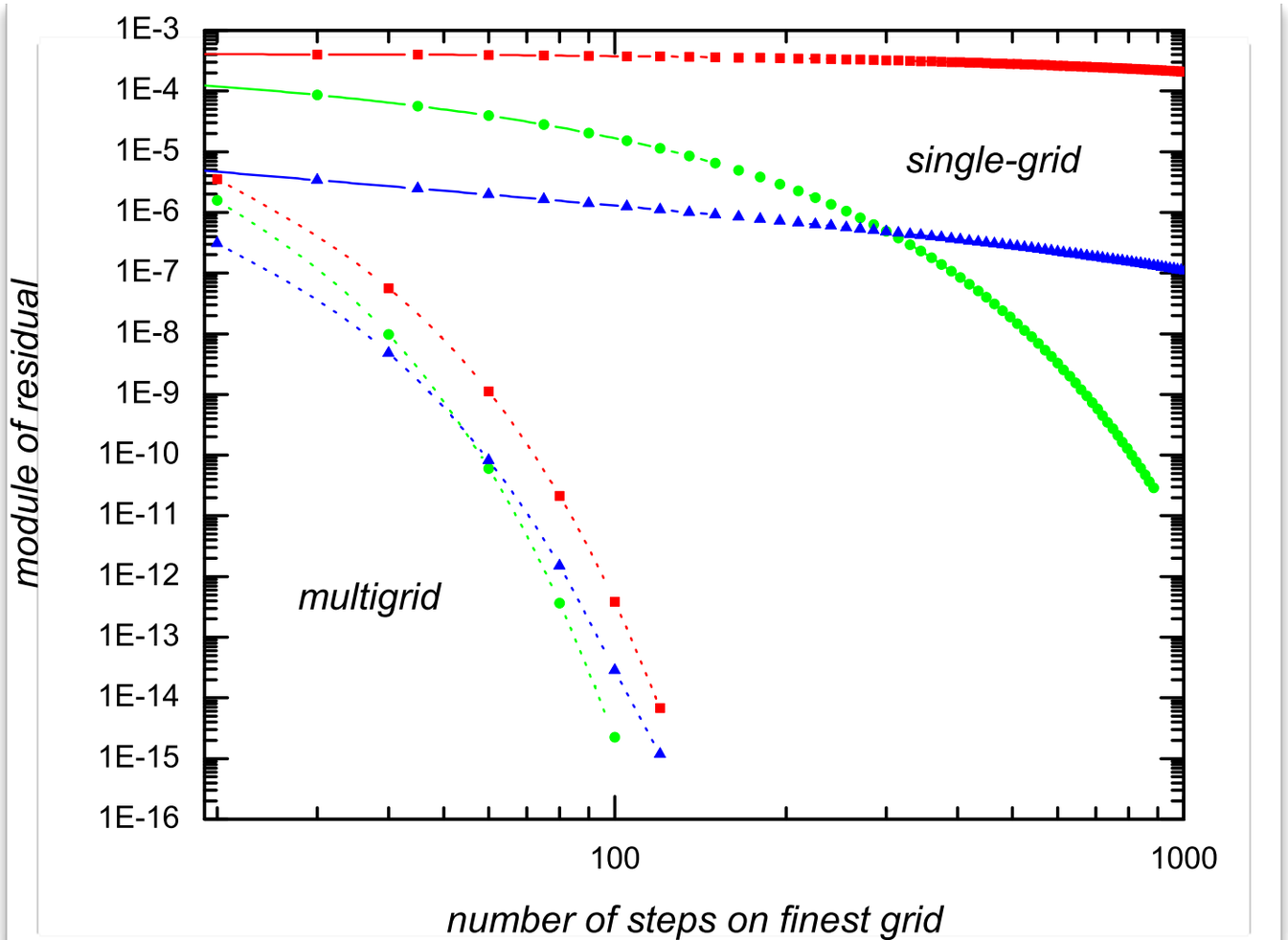
$$\tilde{\Phi}^{\text{new}}_{i,j,k} = \tilde{\Phi}^{\text{old}}_{i,j,k} - \frac{F\left(\tilde{\Phi}^{\text{old}}_{i,j,k}\right)}{\partial F\left(\tilde{\Phi}^{\text{old}}_{i,j,k}\right) / \partial \tilde{\Phi}_{i,j,k}}$$

Update cells with one colour (e.g., red) in the first round (called a relaxation sweep)

Update cells with another colour (while in the second round)

If convergence is not achieved, simply repeat these relaxation sweeps again and again, until convergence.

# The Multigrid Technique

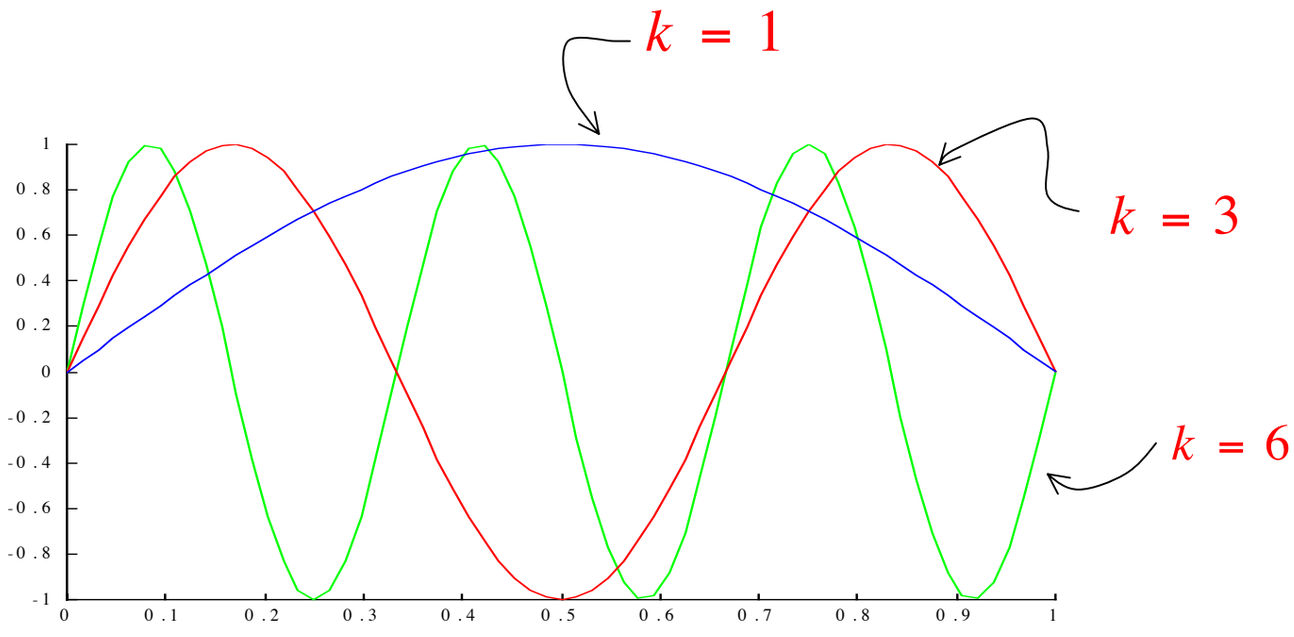



$$\tilde{\Phi}^{\text{new}}_{i,j,k} = \tilde{\Phi}^{\text{old}}_{i,j,k} - \frac{F\left(\tilde{\Phi}^{\text{old}}_{i,j,k}\right)}{\partial F\left(\tilde{\Phi}^{\text{old}}_{i,j,k}\right)/\partial\tilde{\Phi}_{i,j,k}}$$

The residual can be reduced fairly quickly in the first few relaxation sweeps, but then it starts to change more slowly.

# The Multigrid Technique

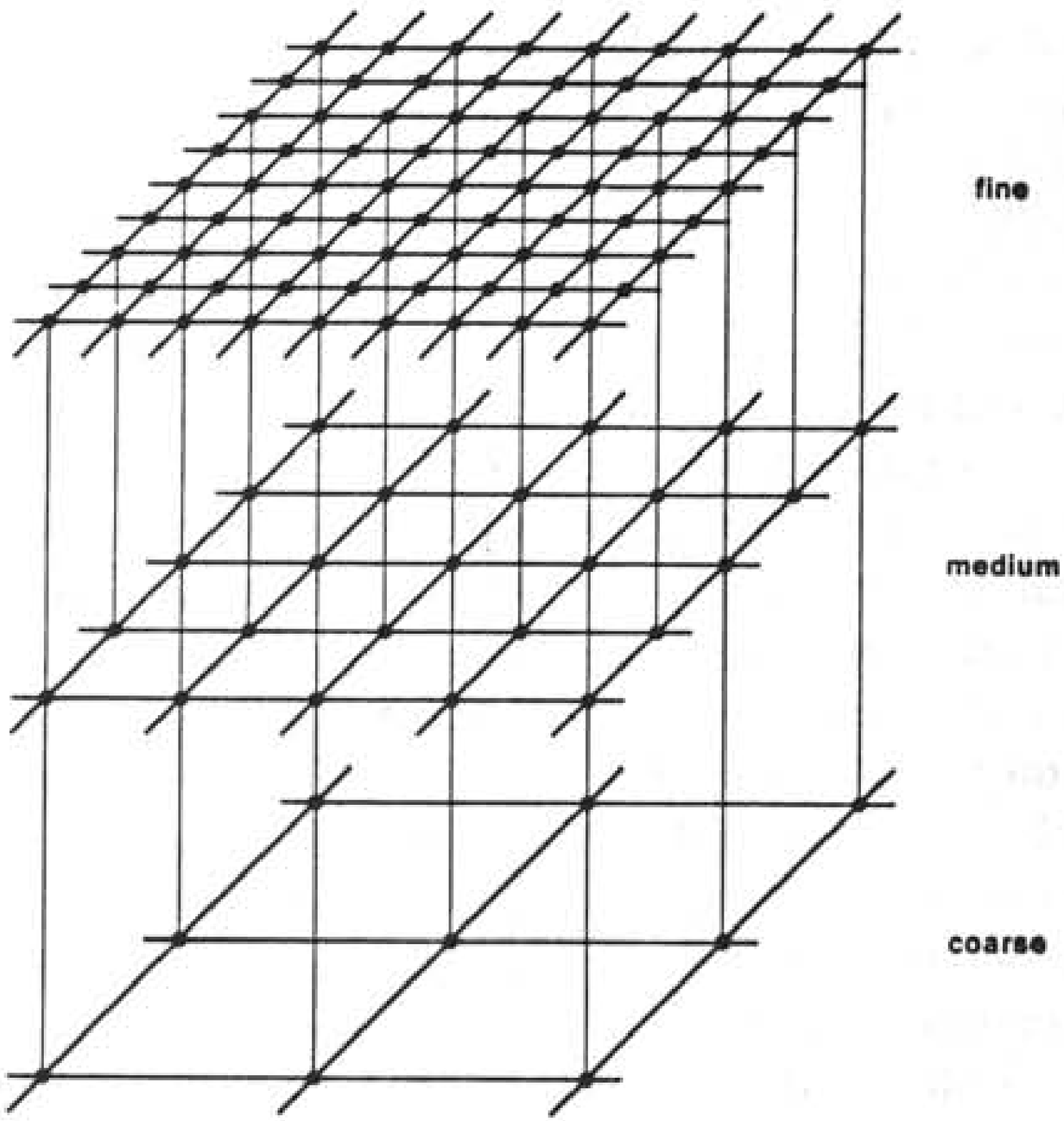
$$\tilde{\Phi}^{\text{new}}_{i,j,k} = \tilde{\Phi}^{\text{old}}_{i,j,k} - \frac{F\left(\tilde{\Phi}^{\text{old}}_{i,j,k}\right)}{\partial F\left(\tilde{\Phi}^{\text{old}}_{i,j,k}\right)/\partial\tilde{\Phi}_{i,j,k}}$$




The residual can be reduced fairly quickly in the first few relaxation sweeps, but then it starts to change more slowly.

This is because relaxation is good at reducing the short wavelength modes of the error, but is very inefficient in reducing modes of the error much larger than the cell size

# The Multigrid Technique

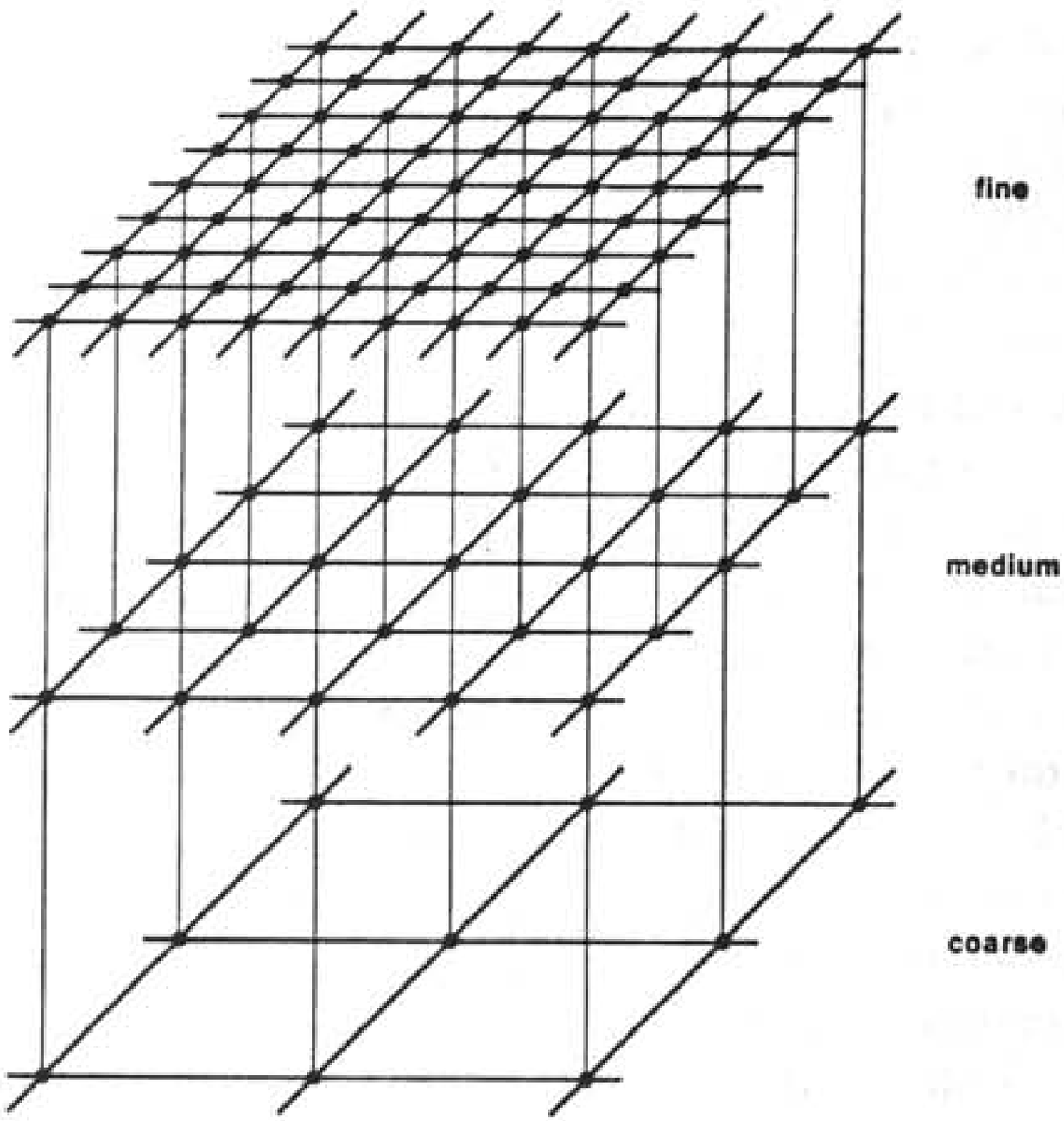



picture taken from Terzopolous 1986

1. Create coarser meshes of the original mesh

Note: AMR creates finer meshes to improve resolution; Multigrid creates coarser meshes to speed up convergence. These coarser meshes are not used to compute gravity force and move particles

# The Multigrid Technique




picture taken from Terzopolous 1986

1. Create coarser meshes of the original mesh

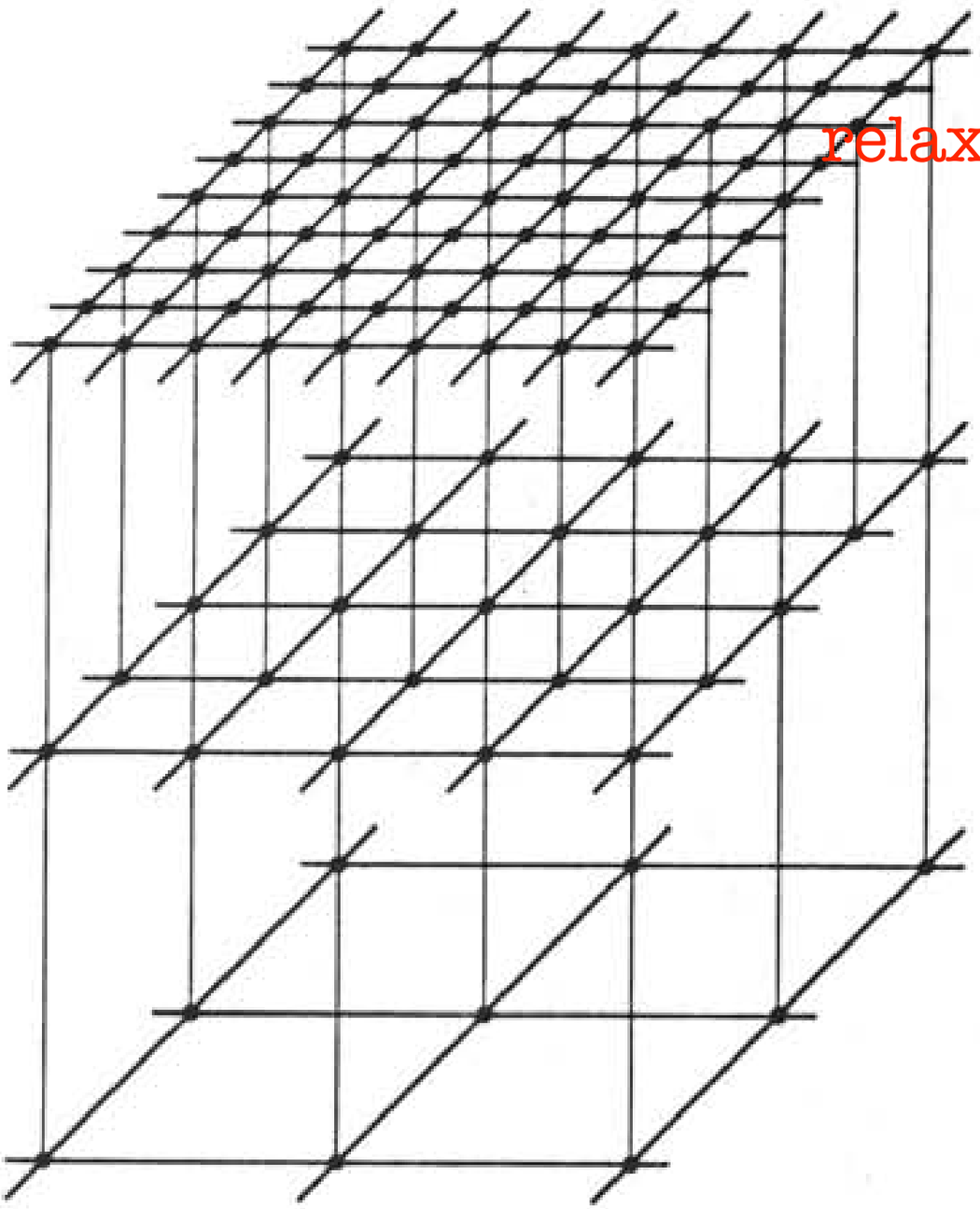
2. do sweeps on the fine level, to reduce the short-wavelength mode of the error

3. go to the medium level and do relaxation there, to reduce the medium-wavelength error mode

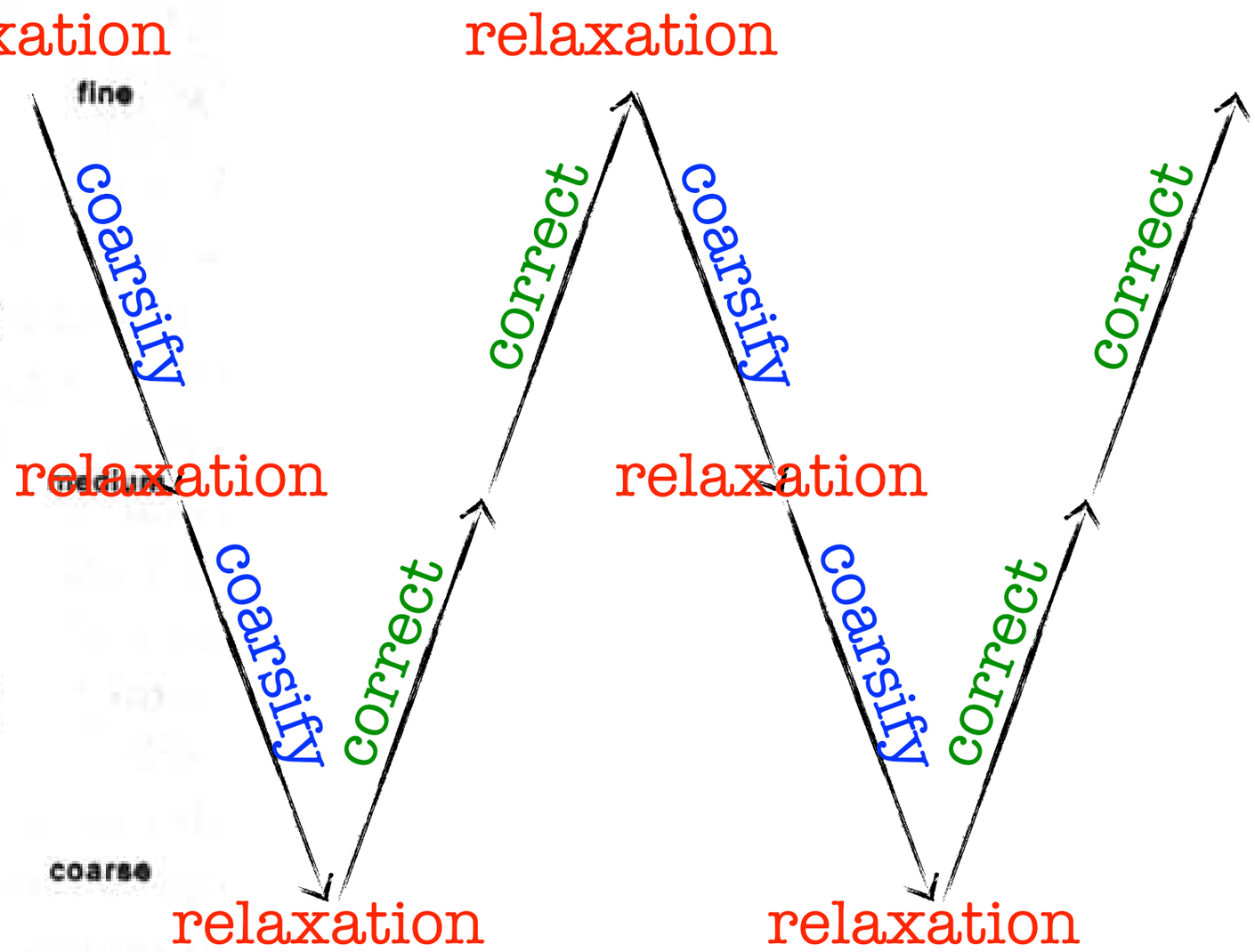
4. go to the coarse level and do relaxation there, to reduce the long-wavelength error mode

5. go back to the fine level, and use coarser level information to update/correct solution there

# The Multigrid Technique

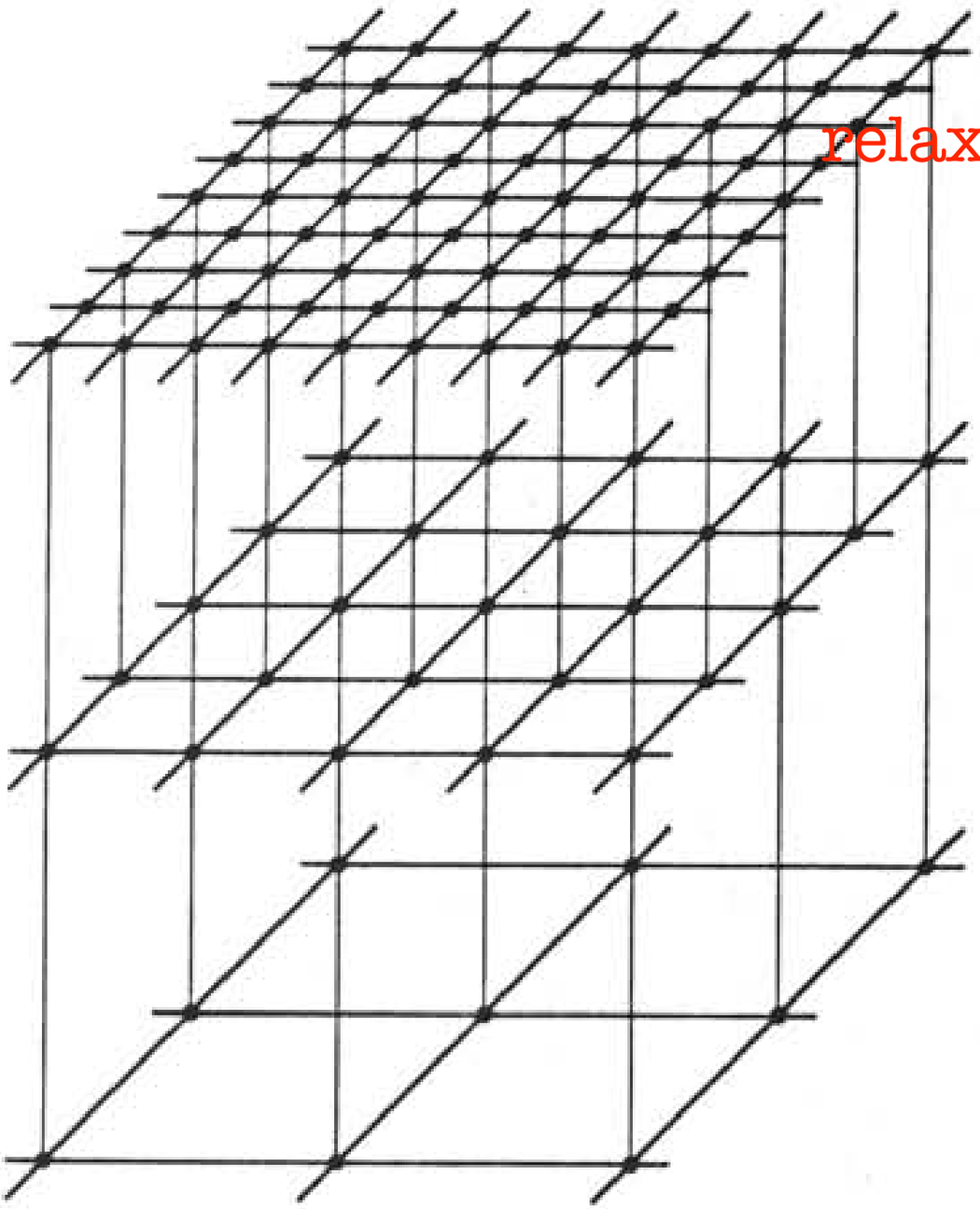


picture taken from Terzopolous 1986

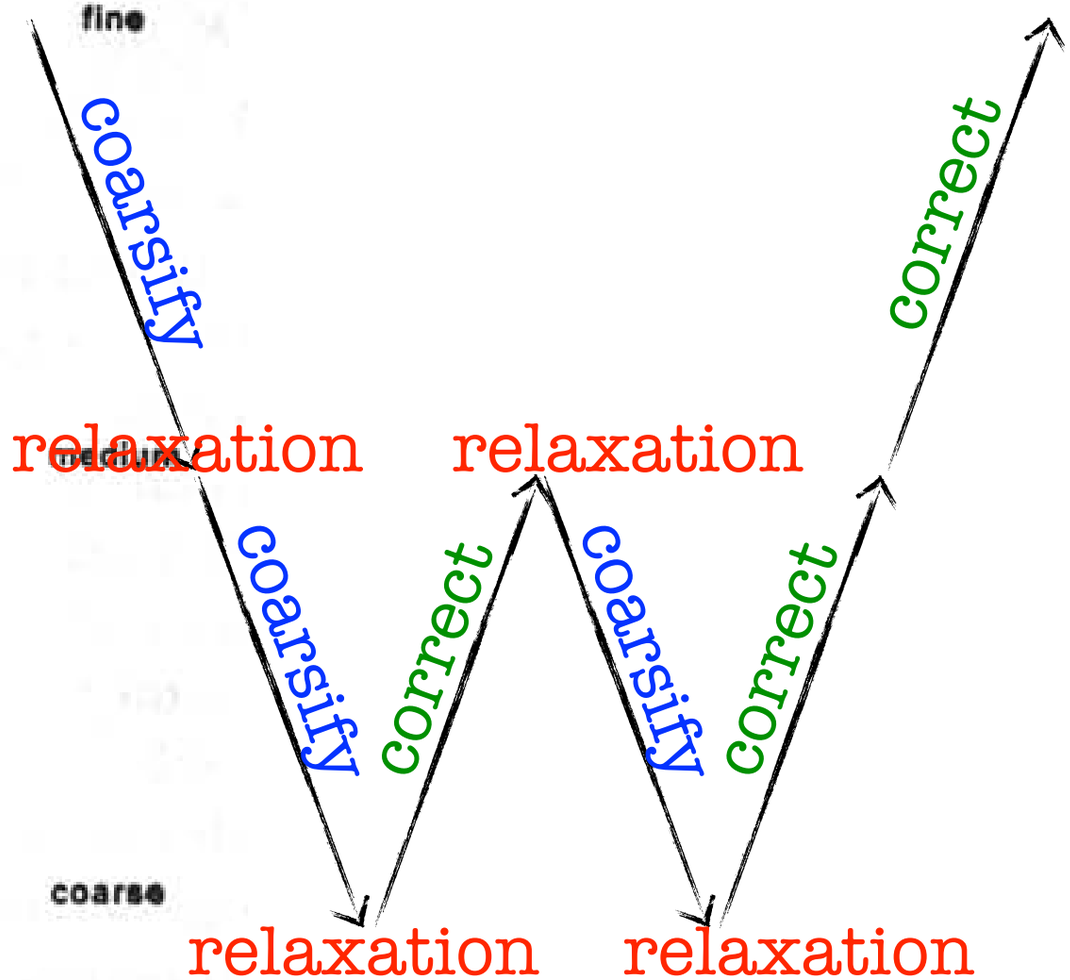



V-cycles

# The Multigrid Technique

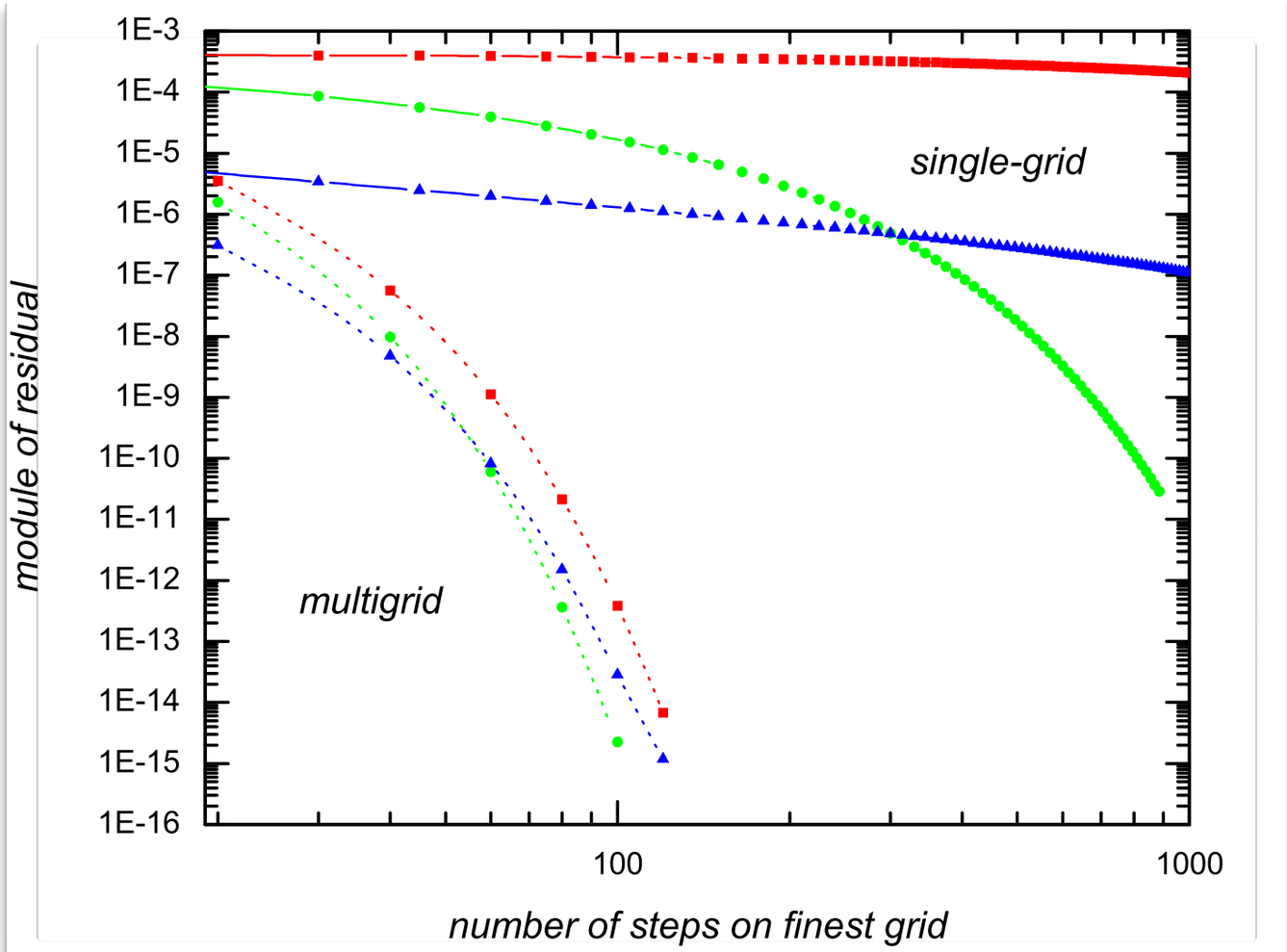


picture taken from Terzopolous 1986




W-cycles

# The Multigrid Technique




Residual decreases much more quickly with multigrid (see left figure), so that the convergence property of the solver is greatly improved.

# Boundary Conditions

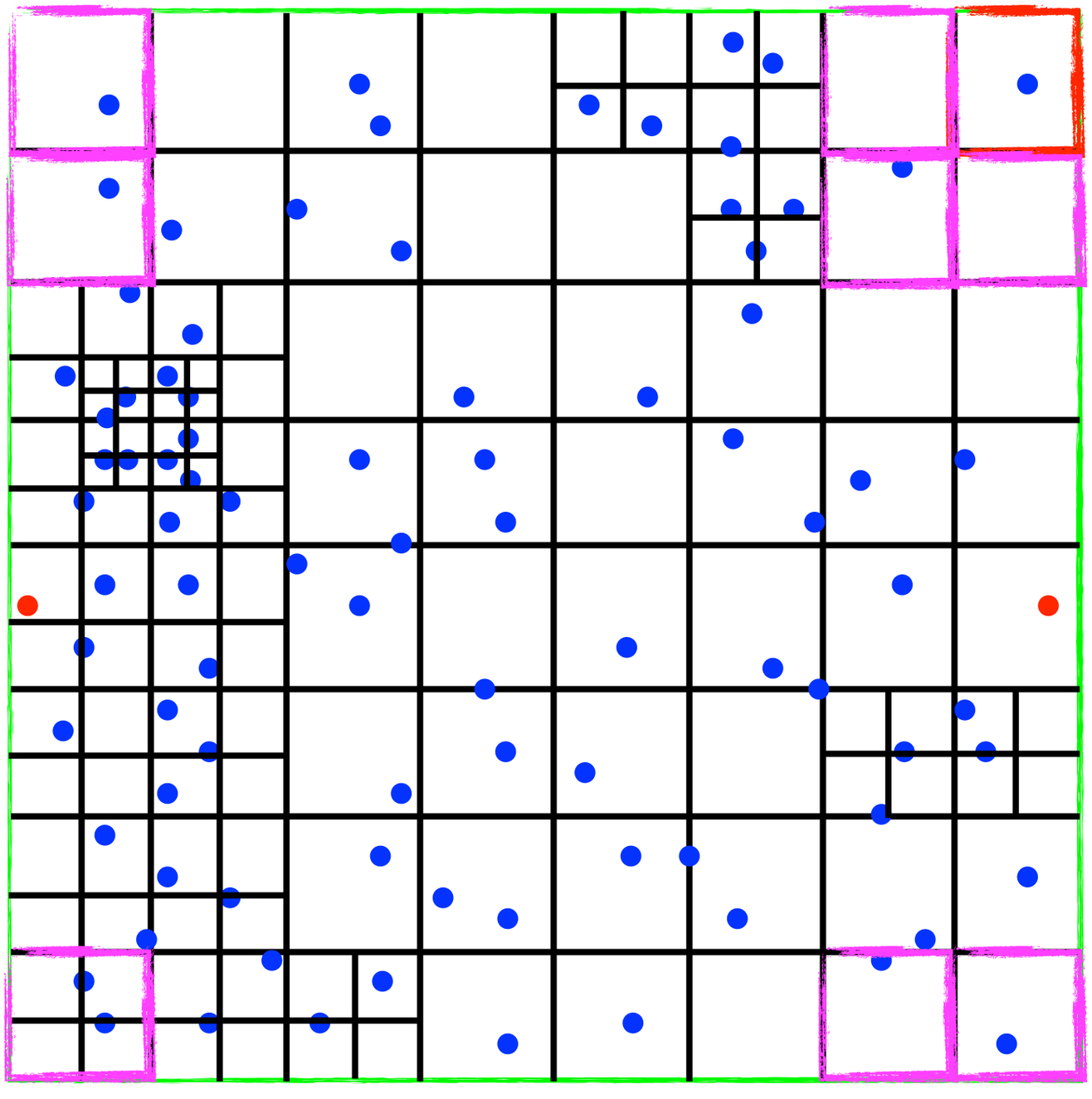
The solution to a differential equation is only unique if the boundary conditions are given. Here we have two sorts of boundary conditions (BCs):

$$F\left(\tilde{\Phi}_{i,j,k}\right) \equiv f\left(\tilde{\Phi}_{i,j,k}\right) - \frac{3}{2} a \Omega_m \tilde{\rho}_{i,j,k} = 0$$

1. BC at the edge of the simulation box (or the whole simulation mesh)

2. BC at the boundary of refinements for AMR simulations

# Boundary Condition for the Box

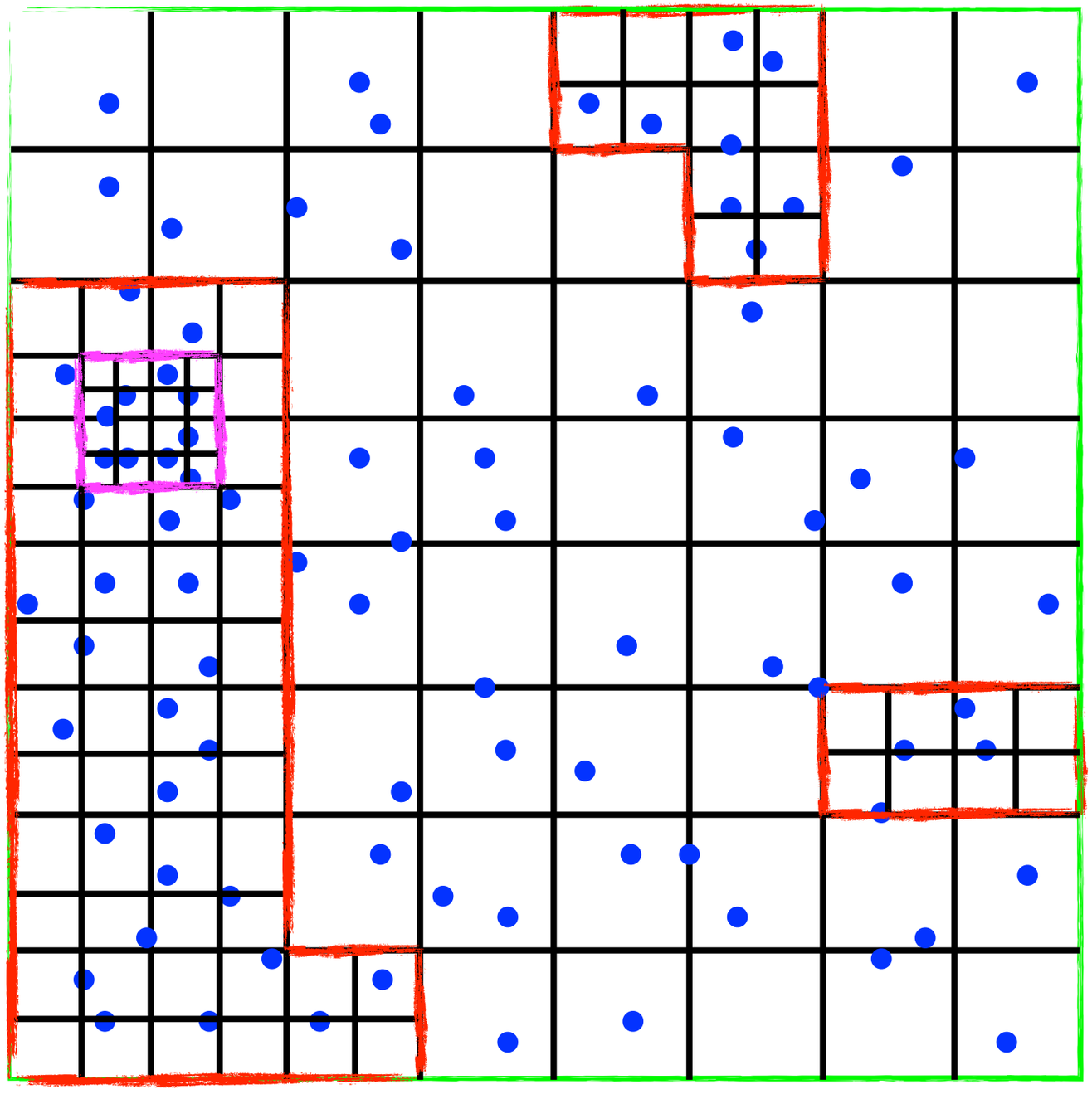


In most cosmological simulations, **periodic** BCs are used for cubic boxes, such that if a particle moves out of the box from one side, it simultaneously reenters on the other side.

The two red particles are very close to each other though they seem to be far away.

The purple cells are neighbours of the red cell.

# Boundary Condition for refinements

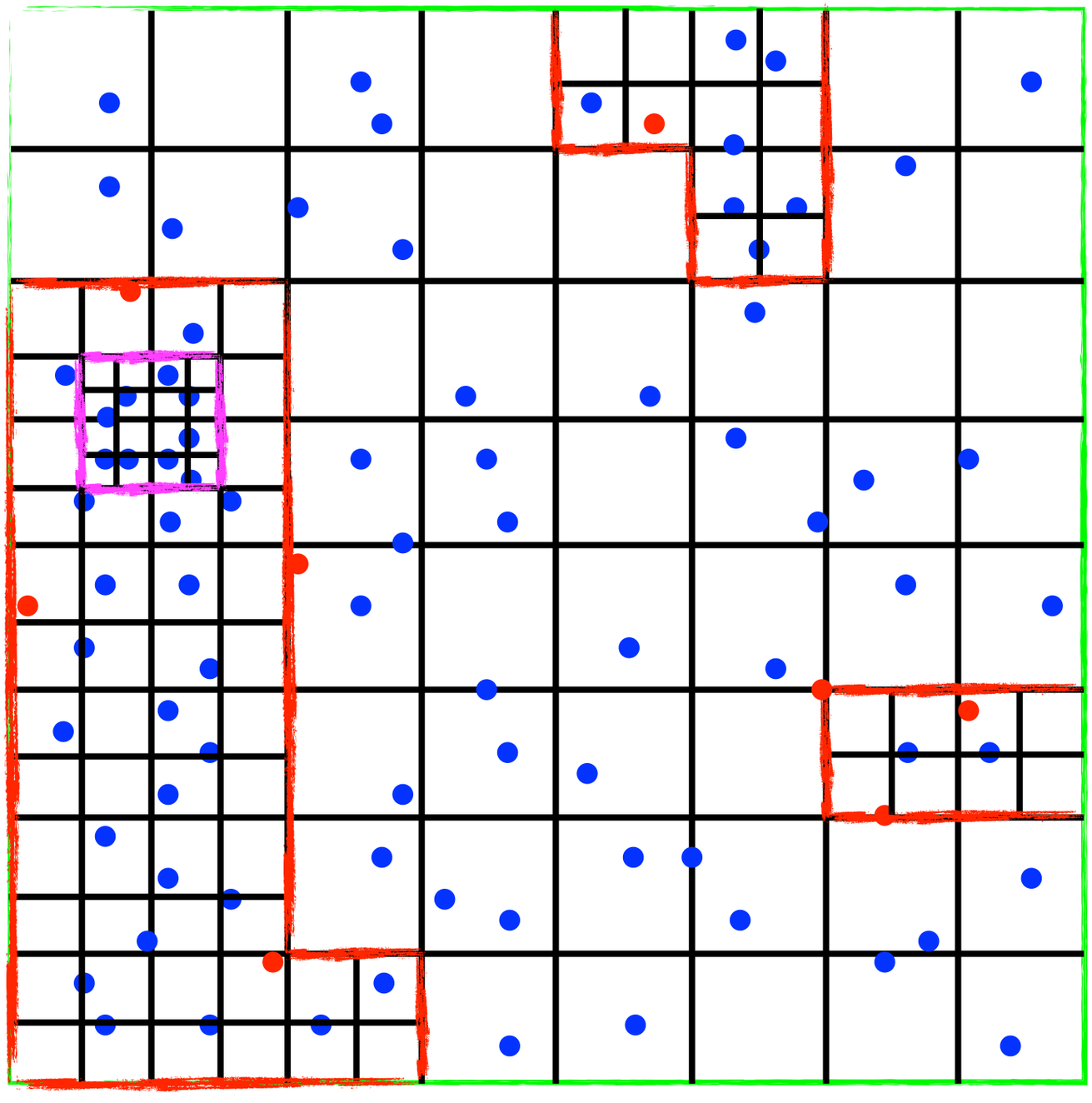


The refinement boundaries can have irregular shapes

**Fixed** BC is used at these boundaries, which means that the values of $\Phi_{ijk}$ in boundary cells are fixed during the relaxation iterations, and new trial values are only given to inner, or non-boundary cells.

Fixed value of $\Phi_{ijk}$ in a boundary cell is obtained by interpolating from the coarser cells. Note that coarse levels are always solved before finer levels.

# A Summary of AMR Algorithm



particle IC

↓

density assignment ($\varrho_{ijk}$)

1. CIC density in coarse level cells

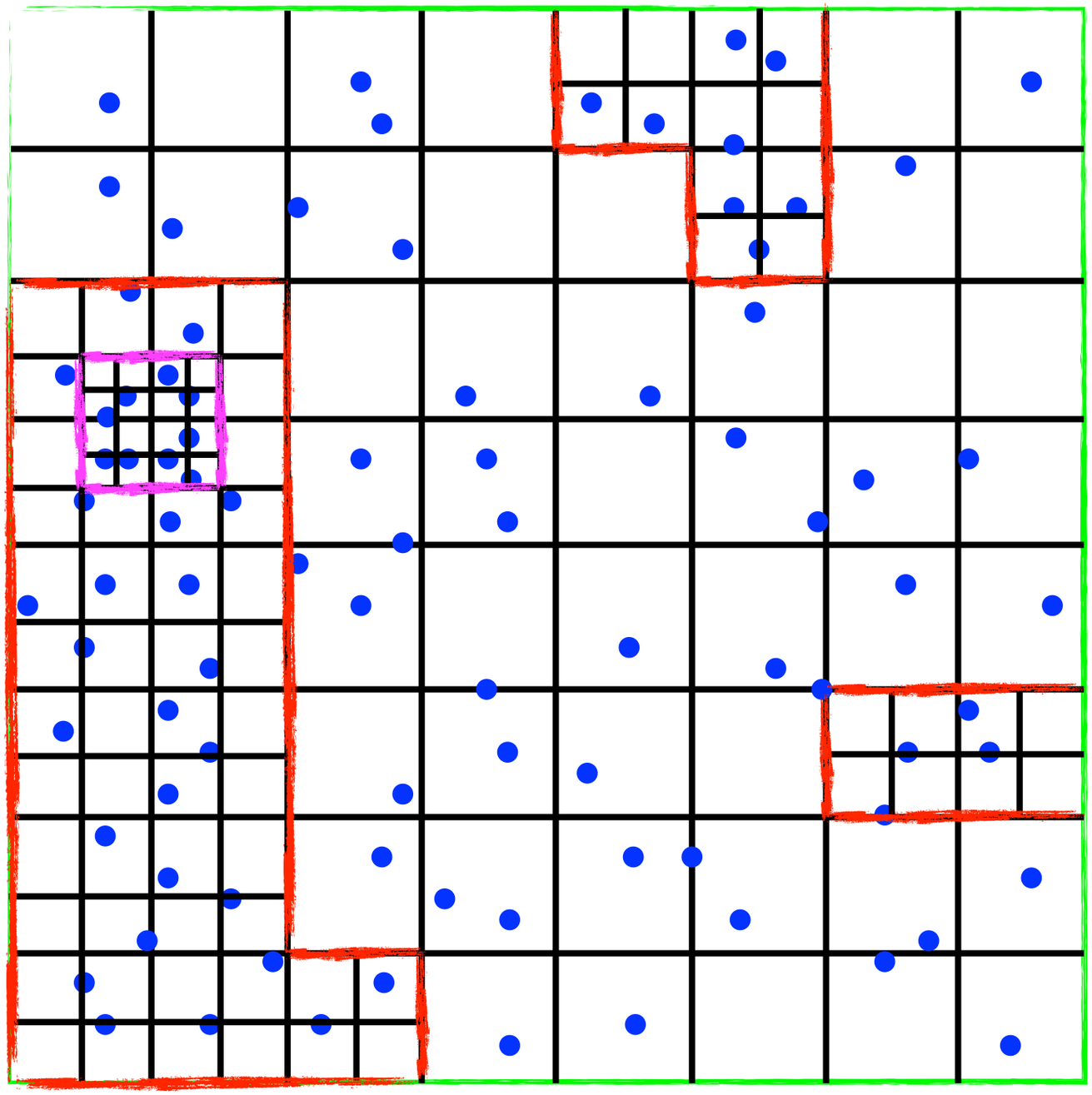
2. refine cells whose density exceeds refinement criterion

3. CIC density in fine level cells

4. repeat until no cells need to be refined (reaching the finest level)

5. be careful: boundary particles (red) contribute to both levels

# A Summary of AMR Algorithm



particle IC

↓

density assignment ($\varrho_{ijk}$)

↓

relaxation to solve($\Phi_{ijk}$)

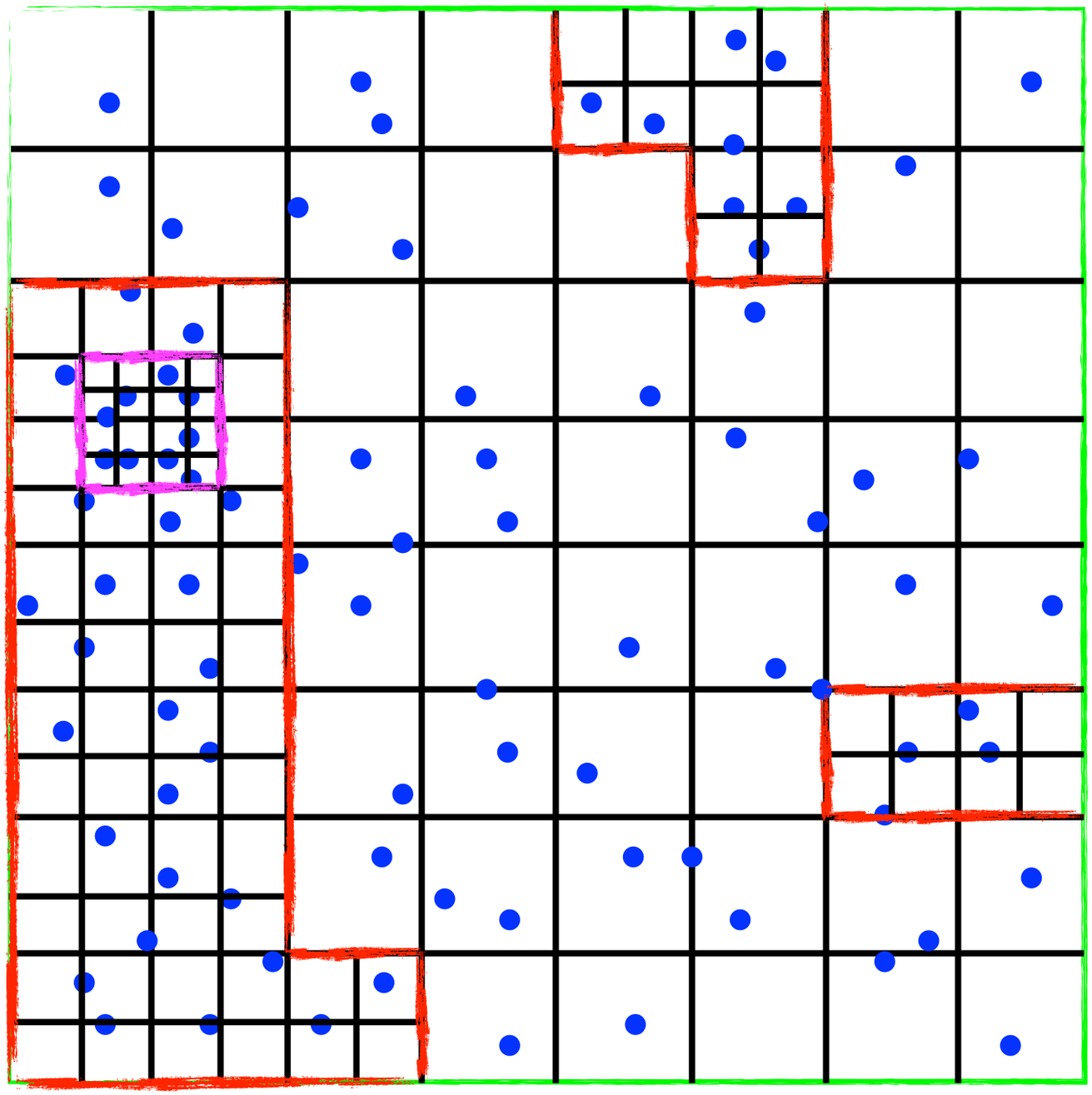
1. solve $\Phi_{ijk}$ on coarse level first

2. use coarse cell values of $\Phi_{ijk}$ to fix the BC on the finer level

3. solve $\Phi_{ijk}$ on the finer level

4. repeat until reaching finest level

# A Summary of AMR Algorithm



particle IC

↓

density assignment ($\varrho_{ijk}$)

↓

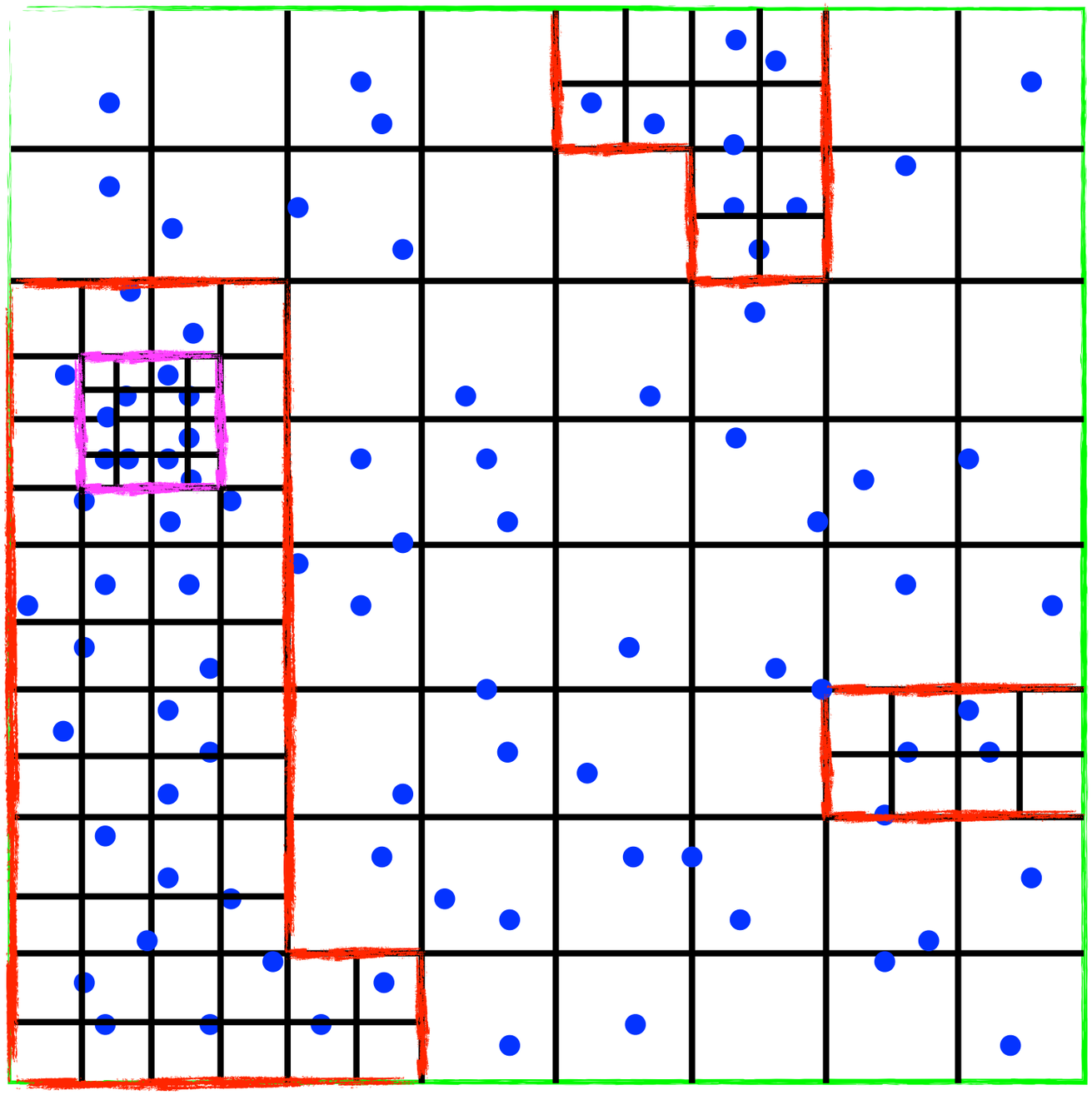
relaxation to solve($\Phi_{ijk}$)

↓

force calculation

Need to do this for all refinement levels. Take care at refinement boundaries, because cells from the coarse level may need to be used.

# A Summary of AMR Algorithm



particle IC

↓

density assignment ($\varrho_{ijk}$)

↓

relaxation to solve($\Phi_{ijk}$)

↓

force calculation

↓

particle movement

Leapfrog time stepping scheme can be used on all levels, but on refinements time steps are generally halved - subcycles.

# A Summary of AMR Algorithm