適合格子細分化法（AMR法）による磁気流体シミュレーション

松本倫明（法政大学）

アウトライン

- 導入
- AMRの概要
- 最近の動向
- 定説と実際
- 実装と解法

資料をアップロードしておきます :

http://redmagic.i.hosei.ac.jp/~matsu/tmp/12chiba/

導入

星形成のシナリオ

分子雲→分子雲コア→ファーストコア→原始星

極端に大きなダイナミックレンジ

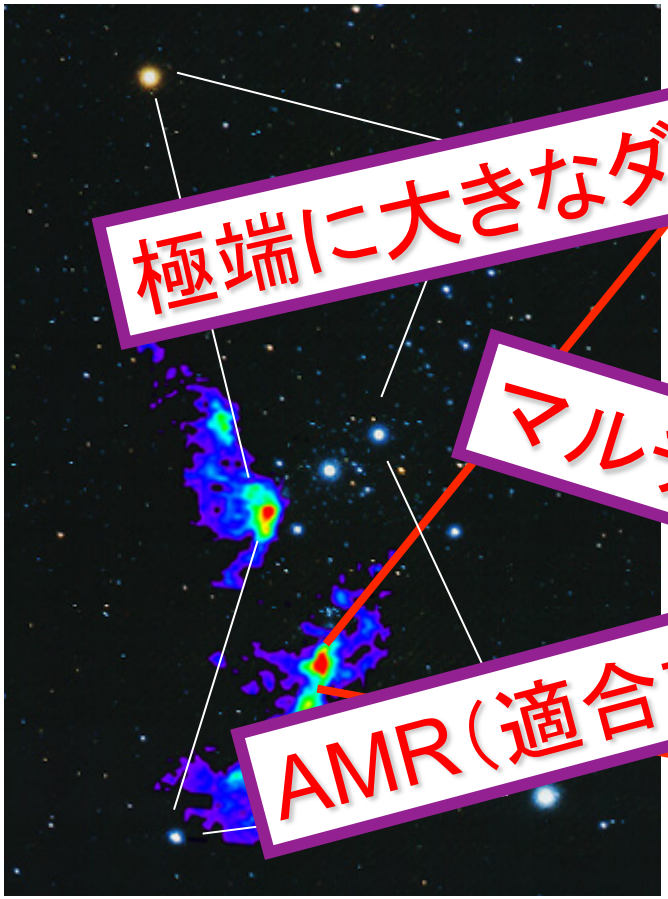
マルチスケール

シミュレーション

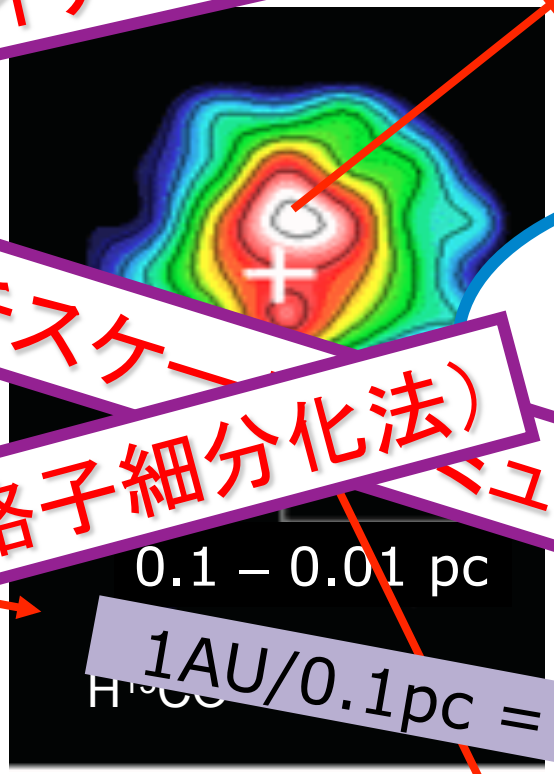
AMR（適合格子細分化法）

1AU/0.1pc = 5×10^{-5}

−5



オリオン分子雲
（optical＋radio）
Sakamoto et al. (1994)

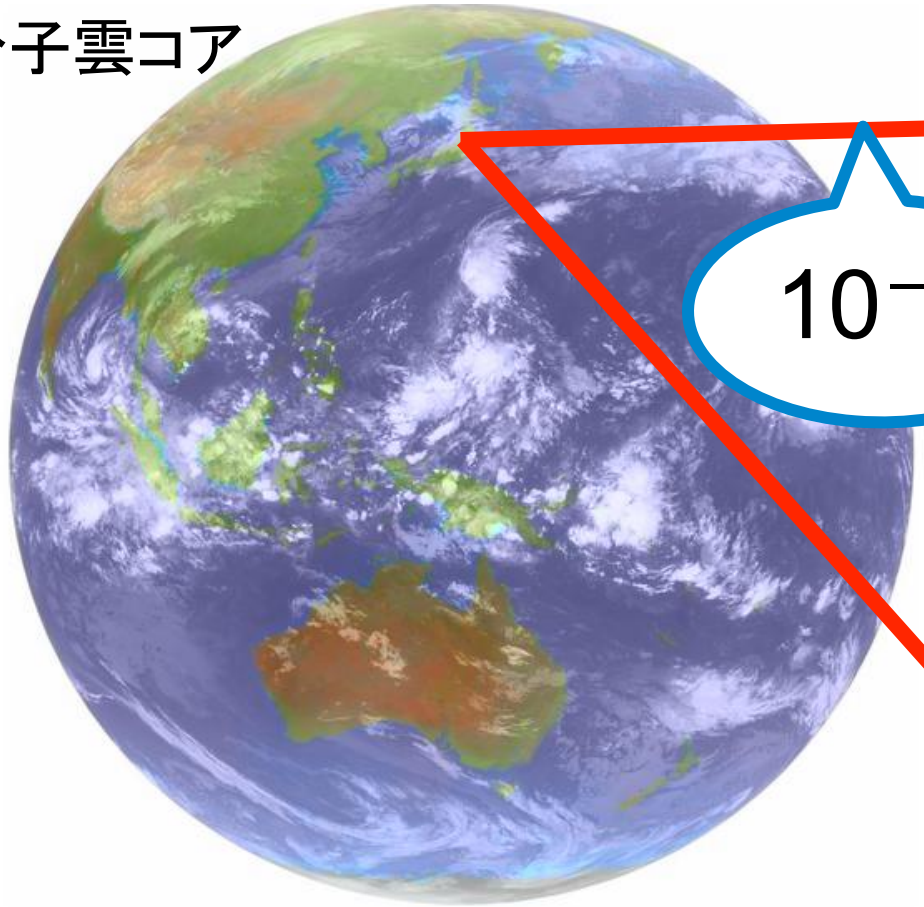


Onishi et al. (1999)

Gueth & Guilloteau (1999)

分子雲コア→ファーストコア→原始星

分子雲コア



10^{-5}

ファーストコア

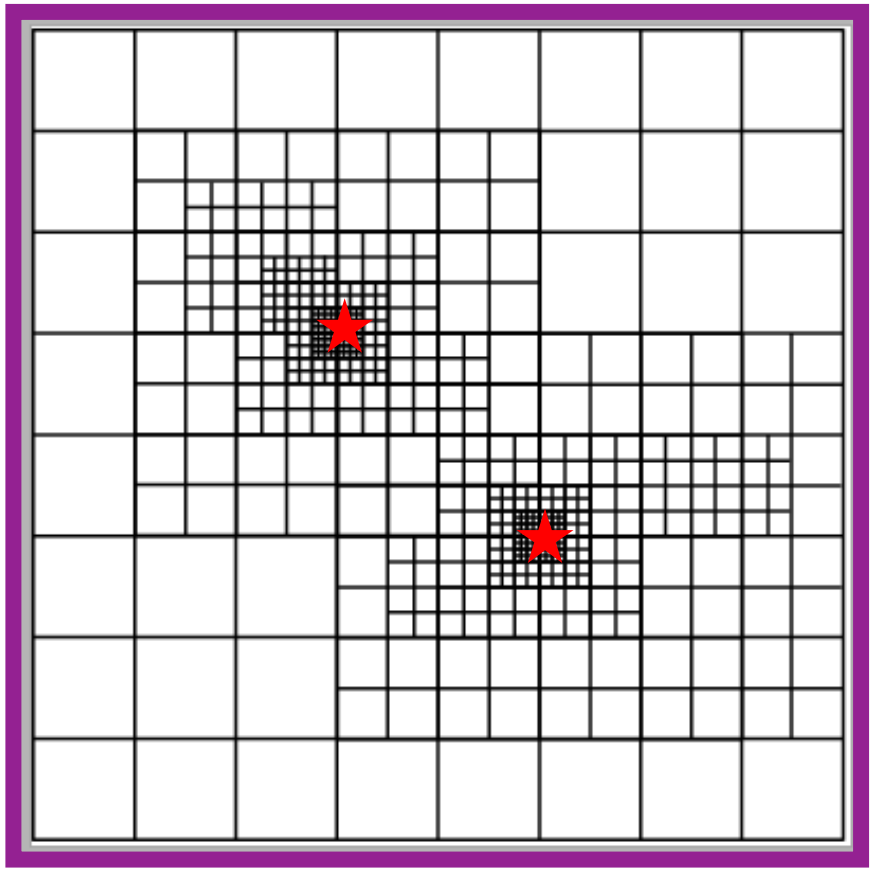
10^{-2}

セカンドコア
（原始星）

シンク粒子（サブグリッドモデル）

AMRの模式図（例）

高解像度が必要な領域を高解像度に



連星系の形成

星の運動とともに
格子を動的に更新する。

AMRとは

- 適合格子細分化法(Adaptive Mesh Refinement; AMR)
- 解適合格子(Solution Adaptive Mesh/Grid)

- 骨子
  - 高解像度が必要な領域を高解像度に。
  - それ以外を低解像度に。
  - 高解像度の領域を動的に変更する。
  - 最大解像度に比して格子点数を節約する。
  - 計算コストの割に高解像なシミュレーションが可能。

AMRを使う目的

- 自己重力による構造形成:
  - 形成される天体を分解したい。
  - 空間スケール∝密度$^{-1/2}$
- 惑星間空間のプラズマ:
  - カレントシートを分解したい。
- 超新星爆発:
  - 燃焼波(デトネーション・デフラグレーション)を分解したい。
- 乱流:
  - パワースペクトルを長いレンジで書きたい。
- つまり、ダイナミックレンジが欲しい。
- 格子点数の割りに高解像度が欲しい。

AMRの概要

JOURNAL OF COMPUTATIONAL PHYSICS 82, 64–84 (1989)

Local Adaptive Mesh Refinement for Shock Hydrodynamics

M. J. BERGER

Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, 10012 New York

AND

P. COLELLA

Lawrence Livermore Laboratory, Livermore, 94550 California

Received September 1987; revised May 19

The aim of this work is the development of an automatic, adaptive mesh refinement strategy for solving hyperbolic conservation laws in two dimensions. There are two main difficulties in doing this. The first problem is the presence of discontinuities, the stability and the effect on them of discontinuities in the mesh. The second problem is how to optimize the algorithm to minimize memory and CPU overhead. This is an important consideration and will continue to be important as more sophisticated algorithms that use data structures other than arrays are developed for use on vector and parallel computers.

全てのAMRはこの論文起源

AMRの分類

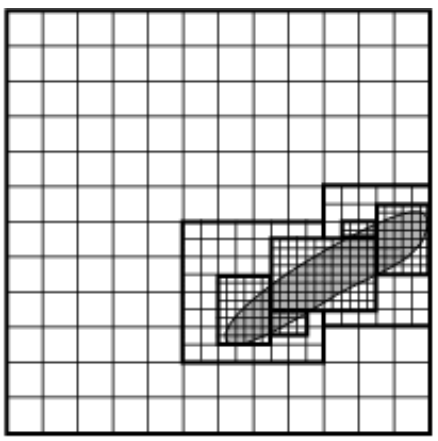
A) パッチ型ブロック構造格子
- パッチ指向
- 最初のAMR
- Berger & Oliger 1984,
- Berger & Colella 1989

B) 八分木型ブロック構造格子
- 八分木構造

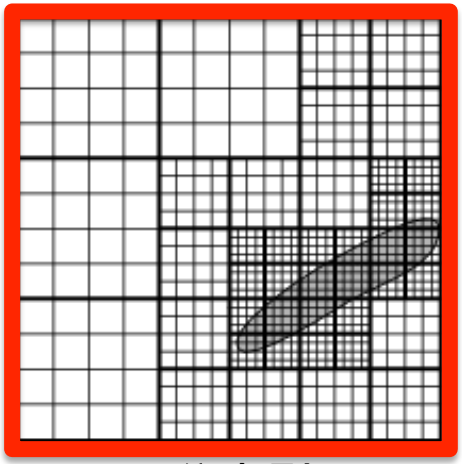
C) セル分割型格子
- 八分木構造

D) 三角形非構造格子
- 天文学ではあまり用いられていない。
- 機体に沿った境界条件に有利。

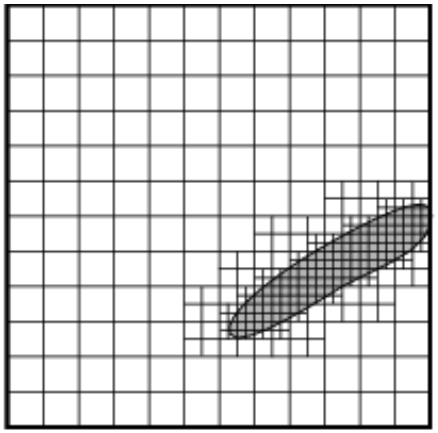
Level = 0 ～ 2



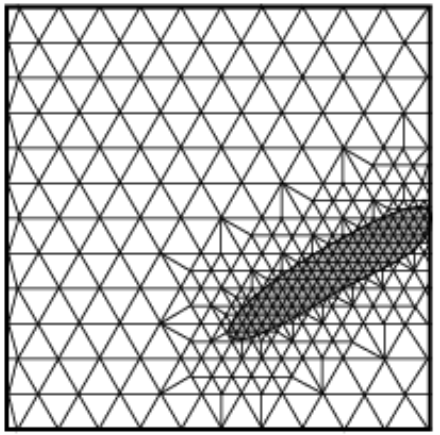
(A) パッチ型
ブロック構造格子



(B) 八分木型
ブロック構造格子



(C) セル分割型格子



(D) 三角形非構造格子

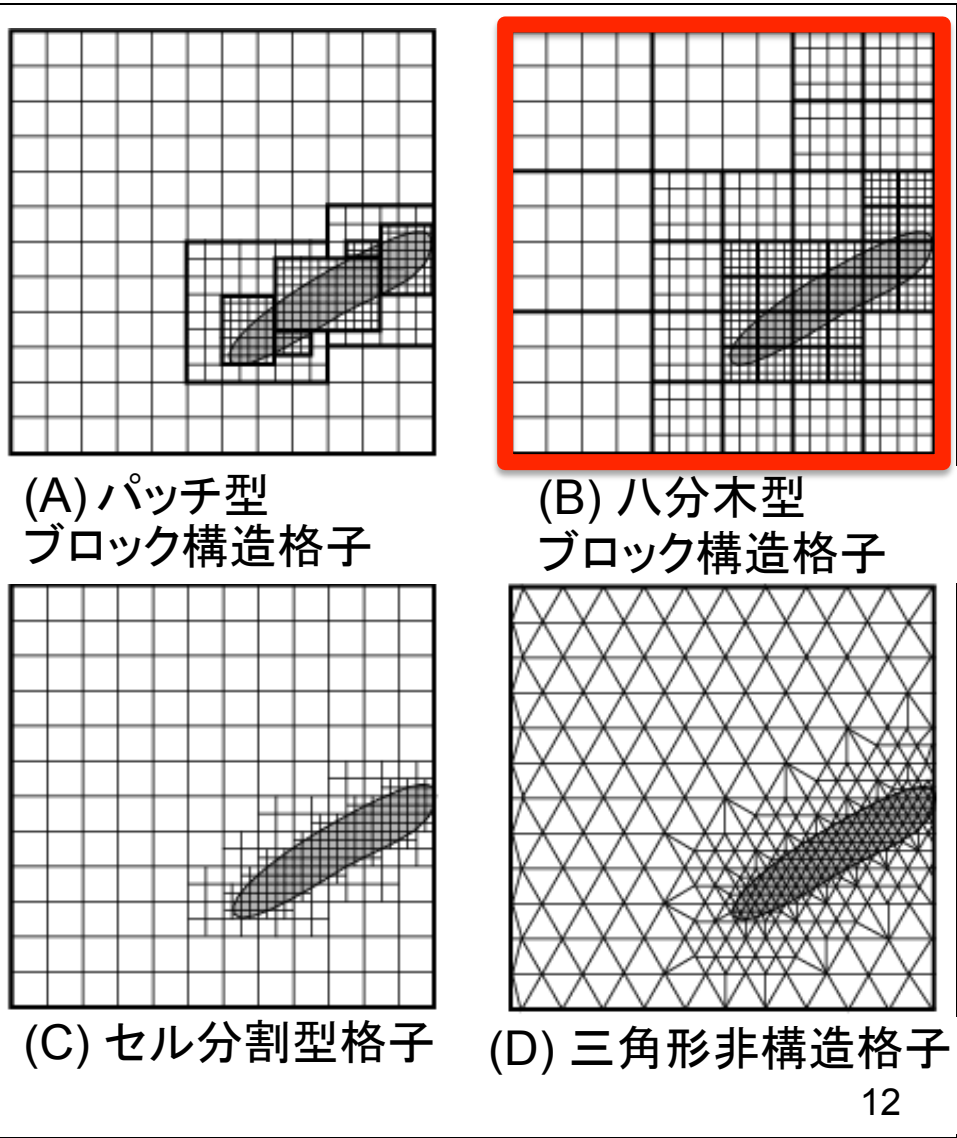
AMRの分類

A) パッチ型ブロック構造格子
- セル数は少なめ。
- ブロック配置のアルゴリズムが複雑。
- 袖のセルが少なく、効率が良い。
- メモリを動的に使う。

B) 八分木型ブロック構造格子
- セル数多め。
- ブロック配置のアルゴリズムが単純。
- 袖のセルが多くい。
- メモリを静的に使う。
- キャッシュの有効活用。

C) セル分割型格子
- セル数は必要最低限。
- オーバーヘッドが大きいか？
- 一様格子ソルバの流用不可。

Level = 0 ～ 2



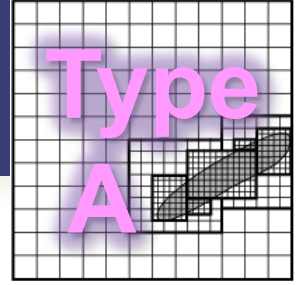
(A) パッチ型
ブロック構造格子

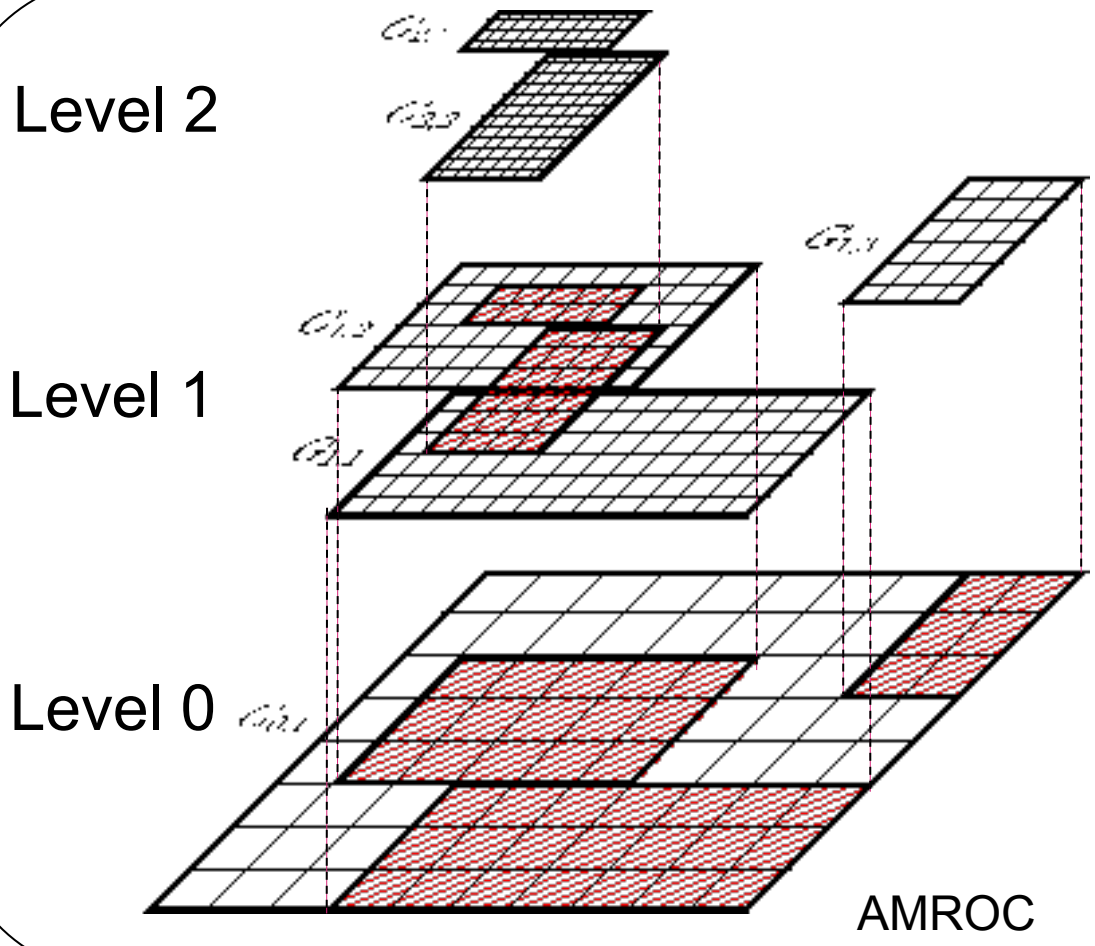
(B) 八分木型
ブロック構造格子

(C) セル分割型格子

(D) 三角形非構造格子

パッチ型ブロック構造格子



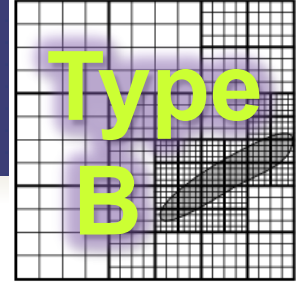


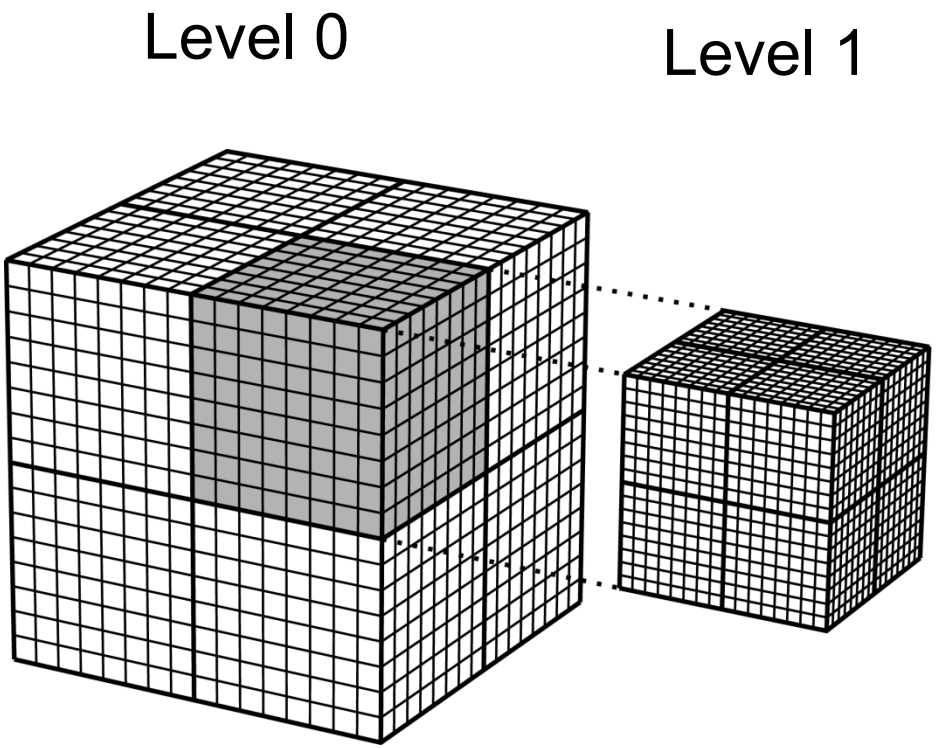
ブロックの大きさは様々（可変長）。

同じレベルのブロックが重なってもOK。

細分化比
= 2, 4, 8…

八分木型ブロック構造格子





SFUMATO

ブロックの大きさは固定（固定長）。

八分木によるデータ管理。

同じレベルのブロックは重ならない。

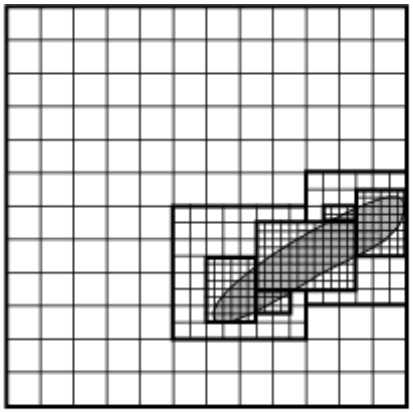
細分比 = 2

天体物理学における主なAMR（古いかも？）

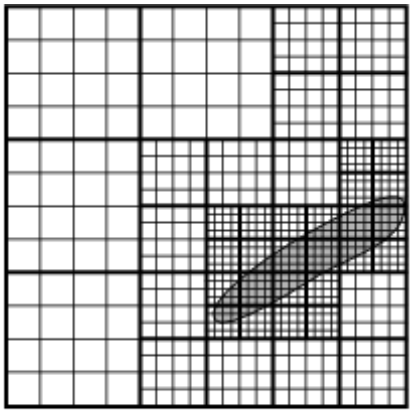
ほかにも沢山。国内にも下記以外に2コード存在。

現在では、MHD と 自己重力は多くのAMRに実装されている。

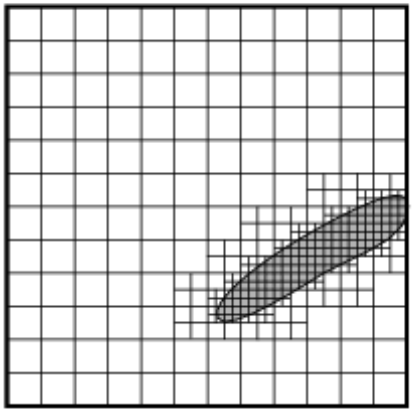
Code name	Author(s)	Main targets	Grid type
ORION	R. Klein	Star formation	A
Enzo	M. Norman	Cosmology	A
FLASH	ASC/U-Chicago	Any	B
BATS-R-US	K. G. Powell	Space weather	B
NIRVANA	U. Ziegler	Any	B
RIEMANN	D. Balsara	ISM	A
RAMSES	R. Teyssier	Cosmology	C
CASTRO	A. S. Almgren CCSE.LBL	Supernovae	A
PLUTO	A. Mignone	Any	A
Athena	J. Stone	Any	Coming seen
SFUMATO	T. Matsumoto	Star formation	B



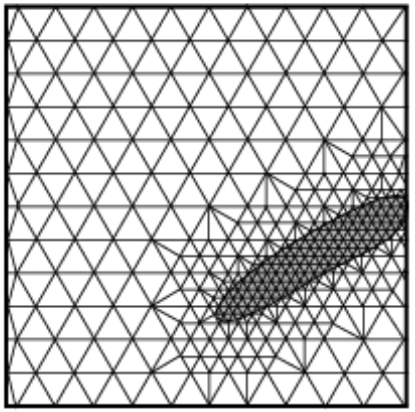
(A) パッチ型ブロック構造格子



(B) 八分木型ブロック構造格子

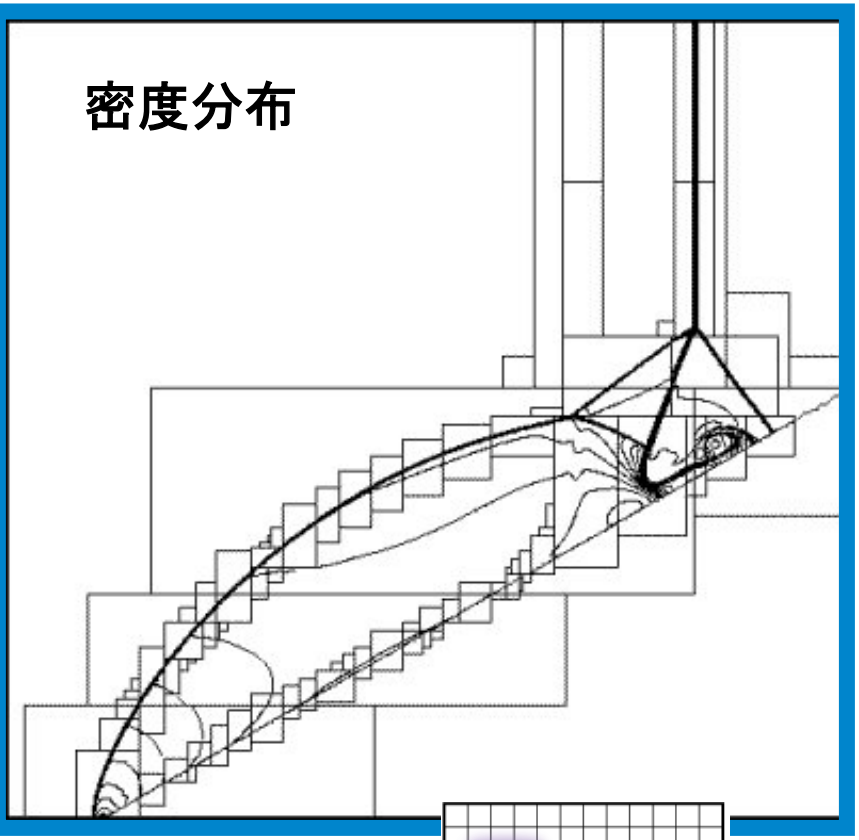


(C) セル分割型格子

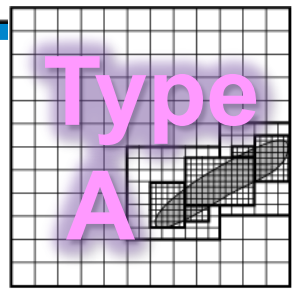


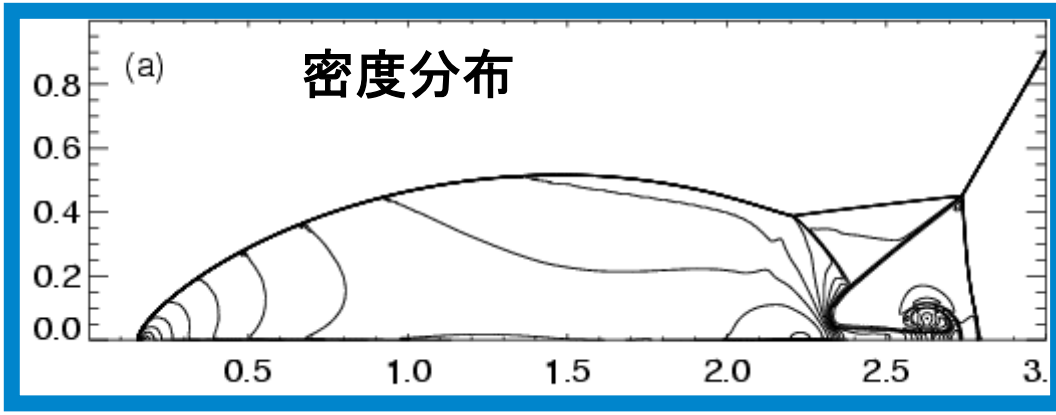
(D) 三角形非構造格子

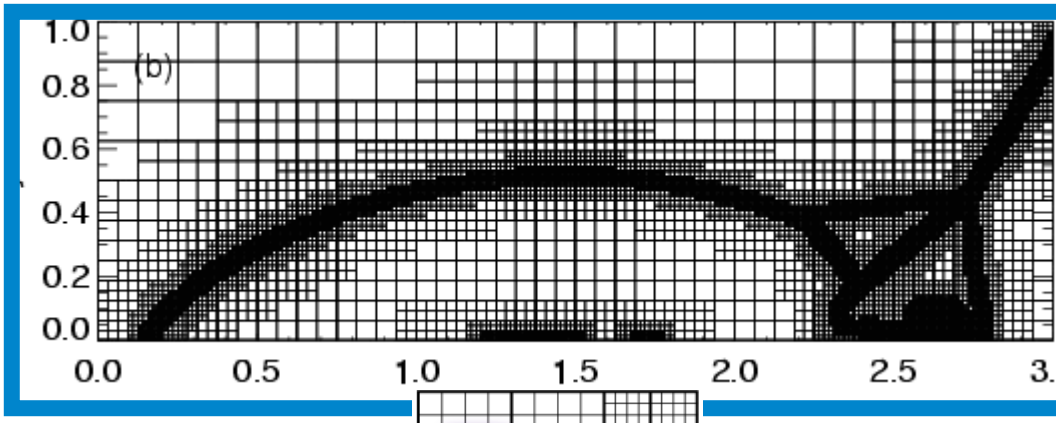
二重マッハ反射問題みるタイプ別の解

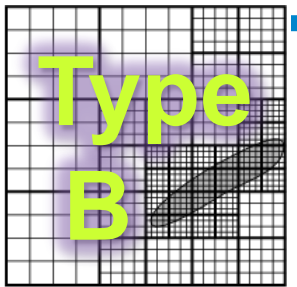


ORION

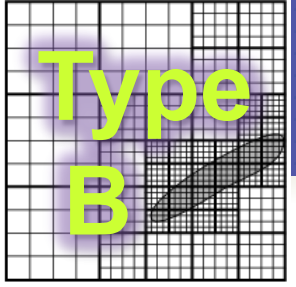




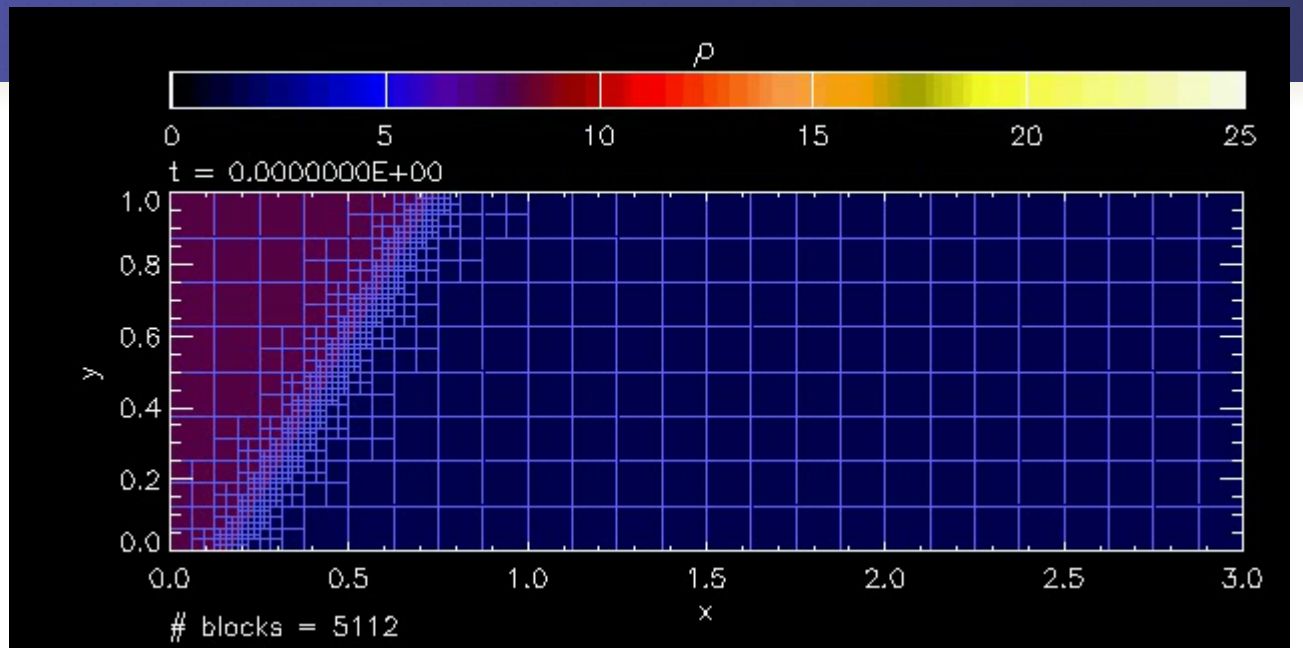




SFUMATO

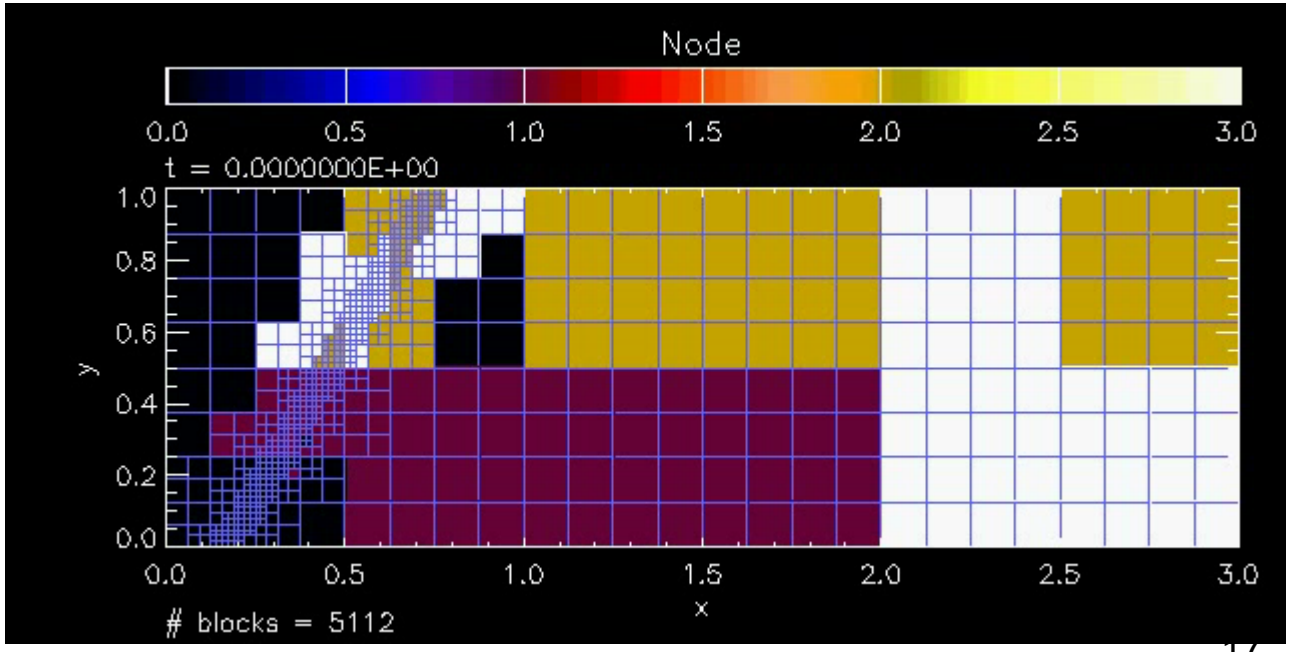


密度分布



ノード(コア)分布
0, 1, 2, 3

SFUMATO

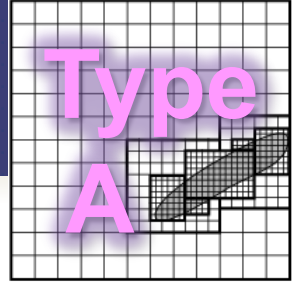


最近の動向

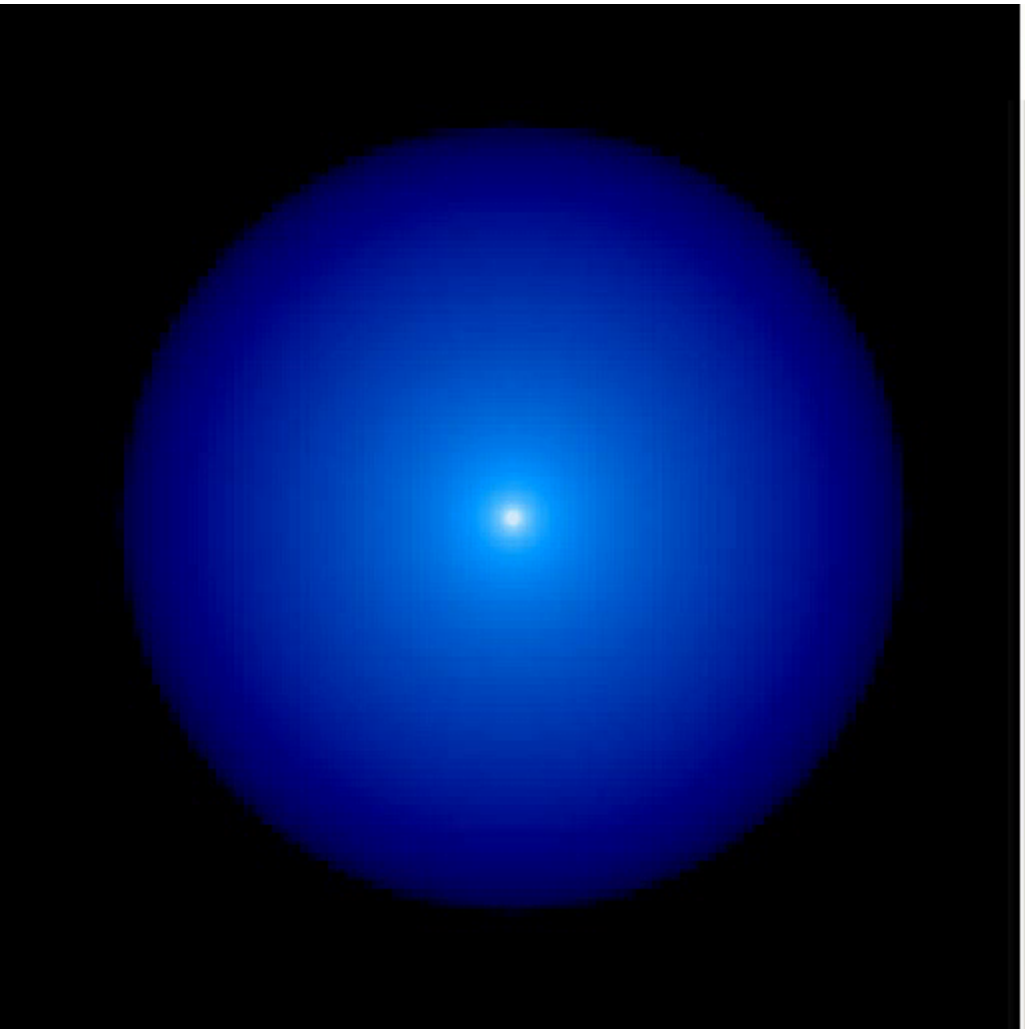
AMRが当たり前の時代になった。

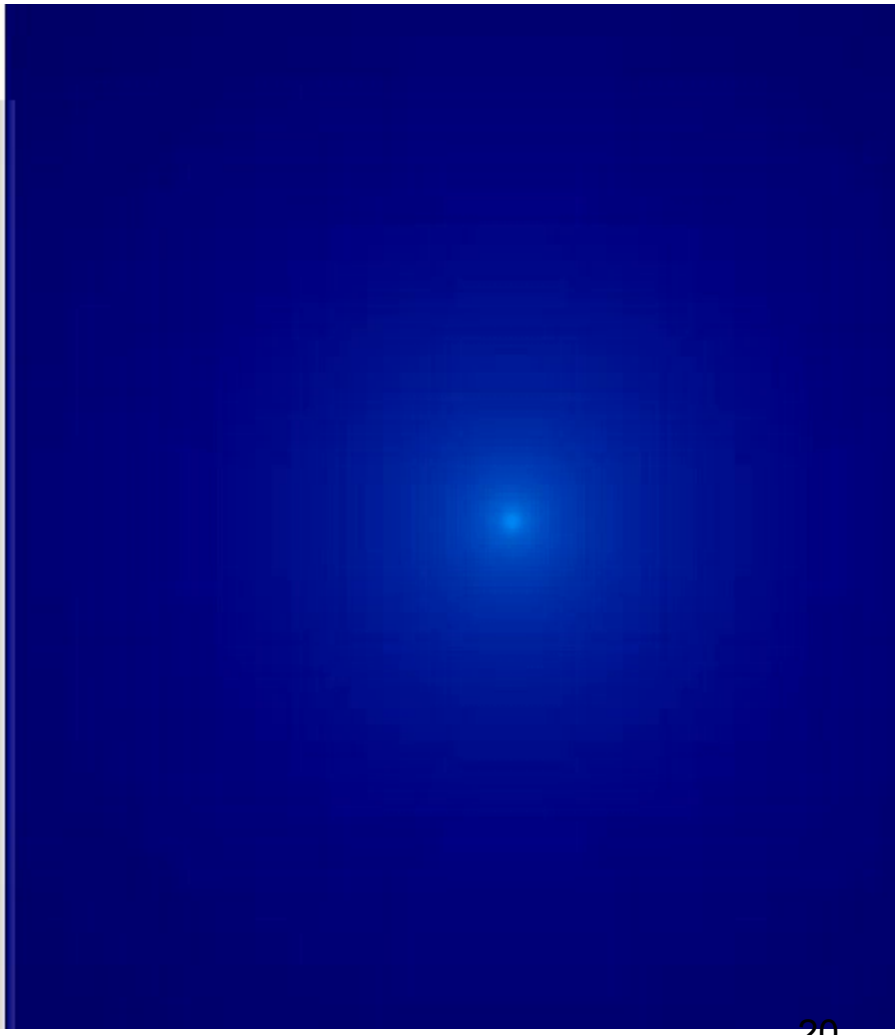
- AMRは特別ではなくなった。
  - 自己重力系のメッシュ法では必須
- 独自性の担保:
  - 新しい物理を導入して勝負
  - 新しいアイディアのモデルで勝負
- AMRの計算例を紹介する。

乱流コアでの多重星形成
ORION: 流体+自己重力+輻射

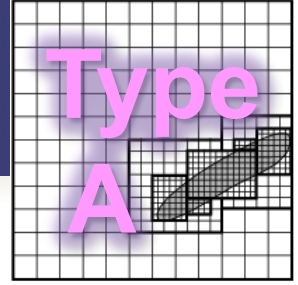


Krumholz, Klein, & McKee (2007)

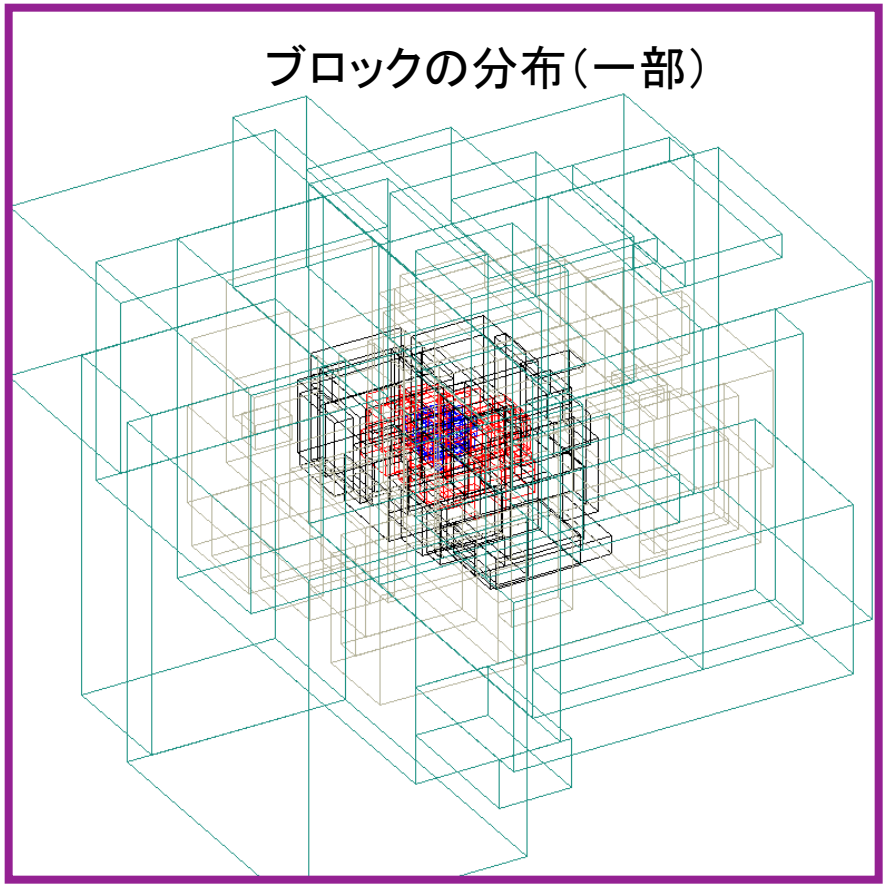


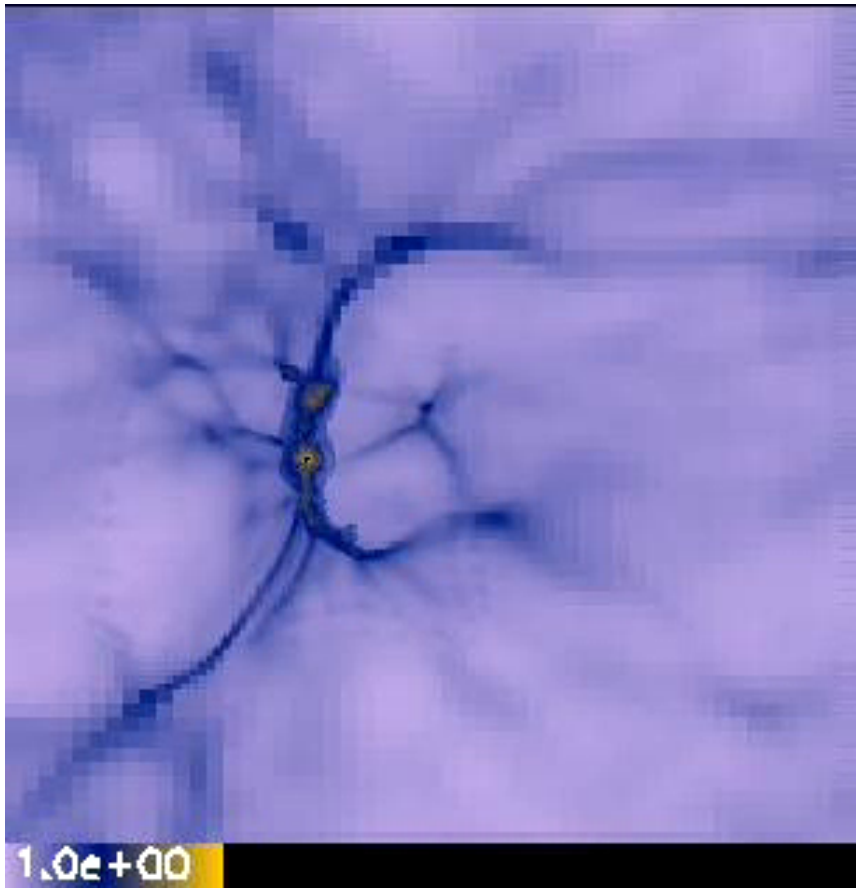


First star formation by Enzo
Abel et al. (2003)



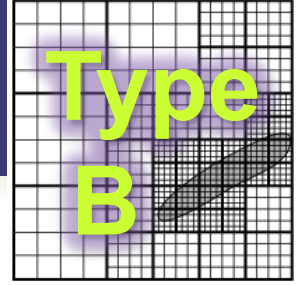
Block-structured grid (patch-oriented type)



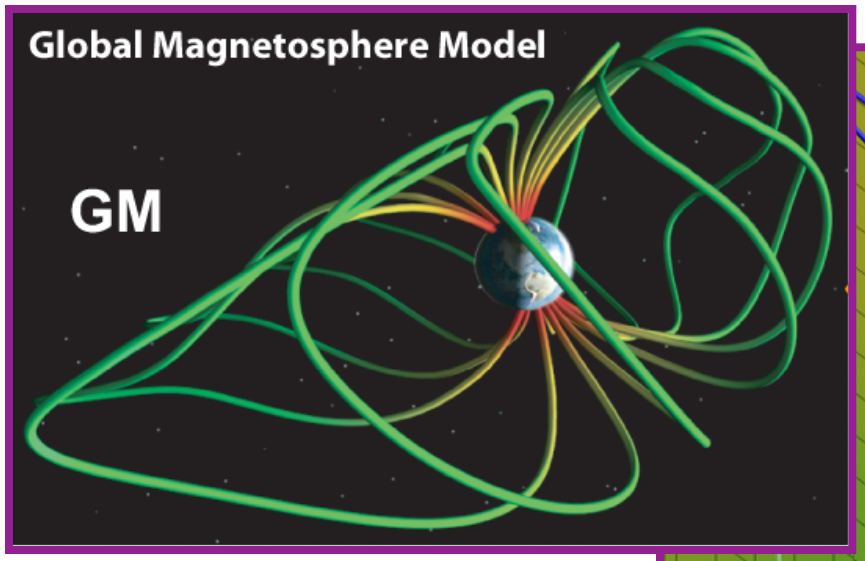


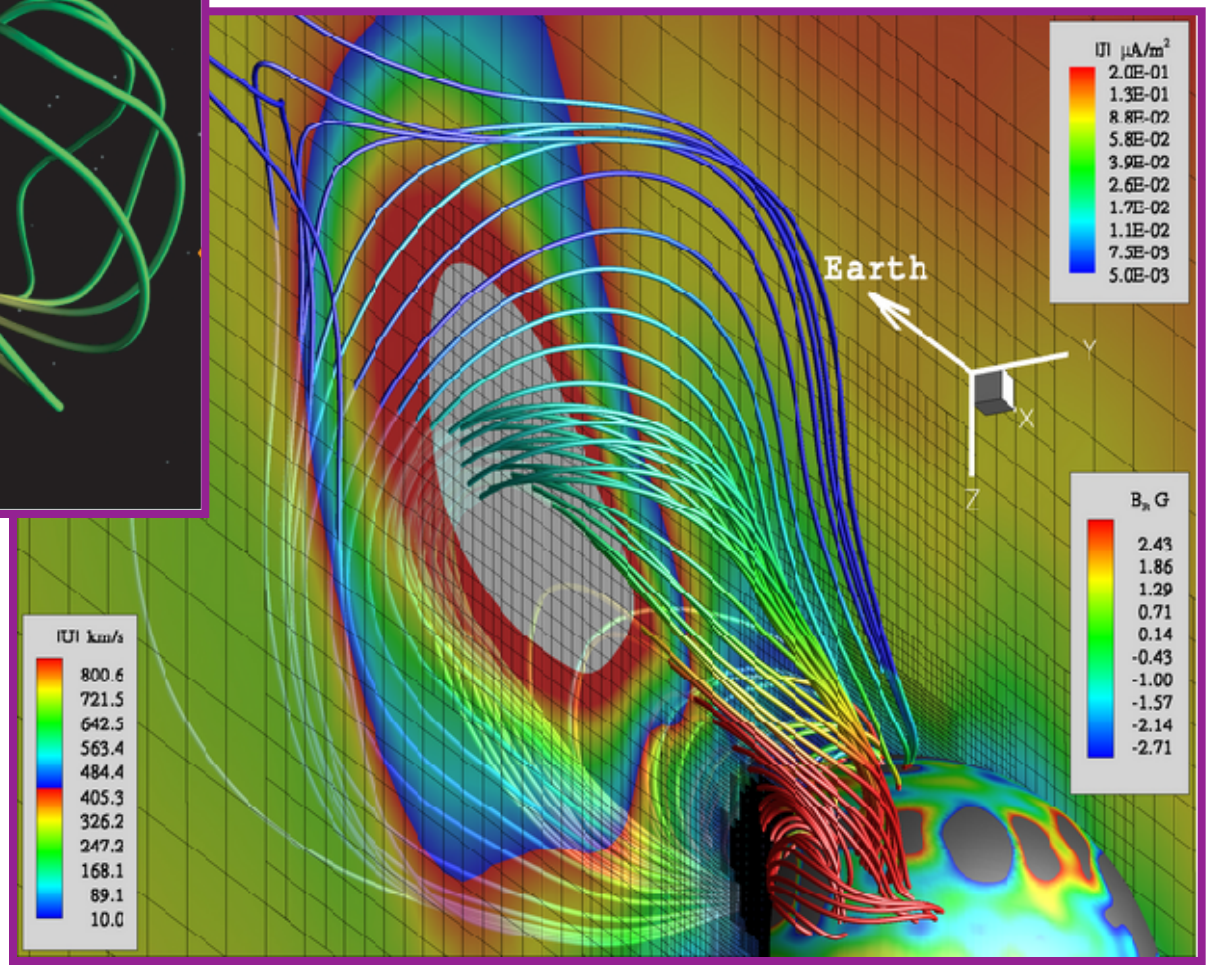
6 kpc ⇒ 100 AU (1,2000倍)

BATS-R-US (K. G. Powell)
Space Weather

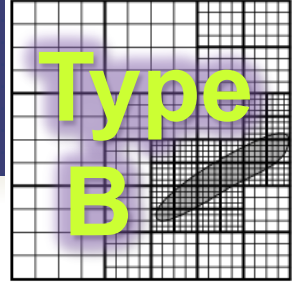


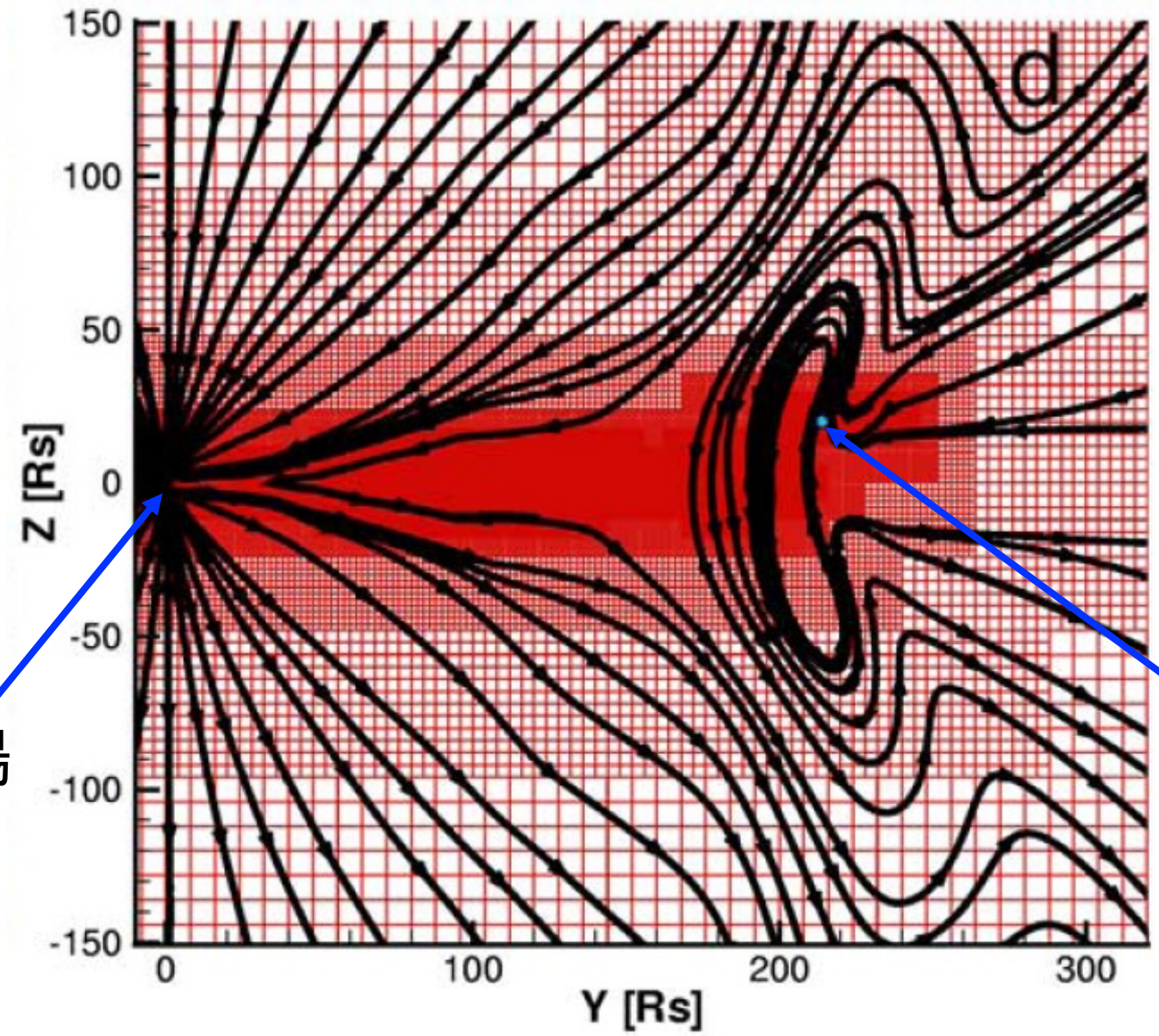
Block-structured grid (oct-tree type)





Coronal Mass Ejection by BATS-R-US Manchester IV et al. (2004)

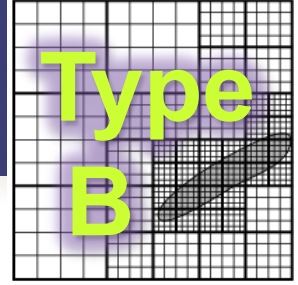




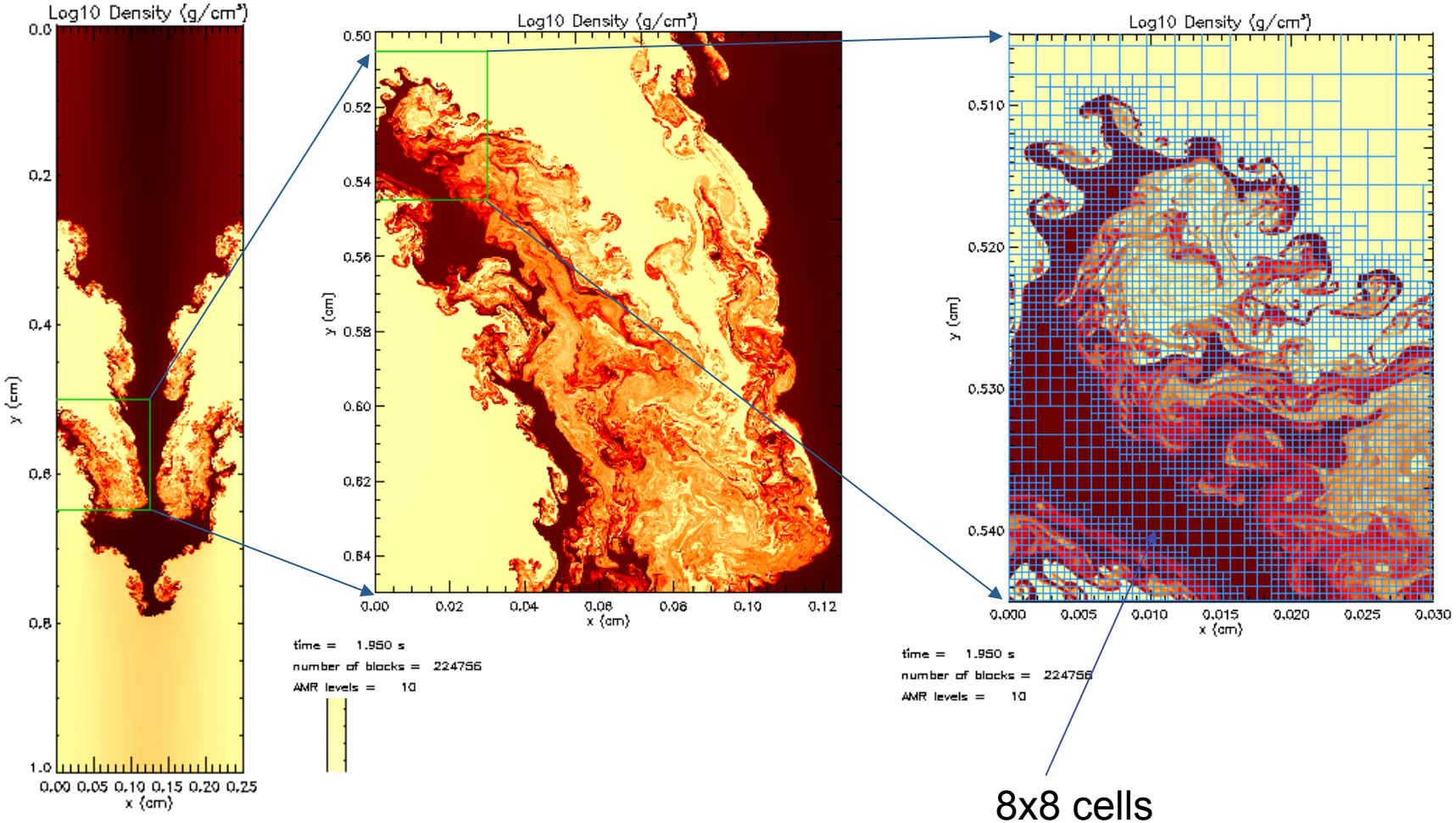
太陽

地球

FLASH (ASC, U-Chicago)
Rayleigh-Taylor Instability

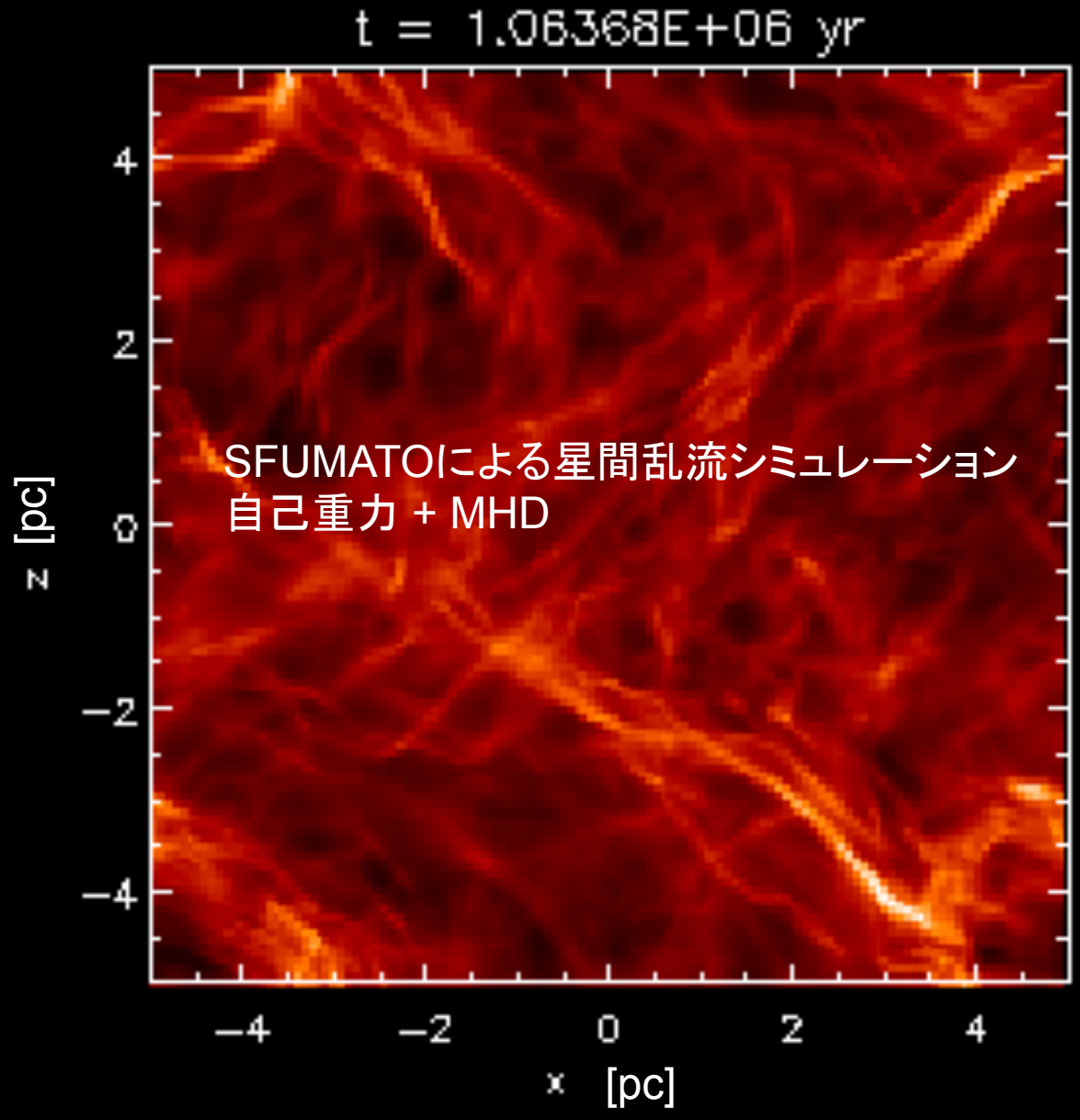


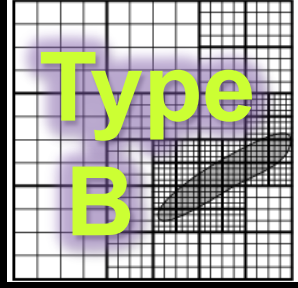
Block-structured grid (oct-tree type)



8x8 cells

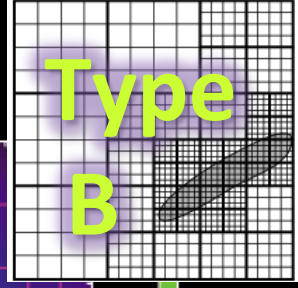
CygOB7　磁場(0.6 micro Gauss)
v=1.9km/s, (10pc)^3, 2.E2/cc, 1.4E4M_sun, マッハ数=10

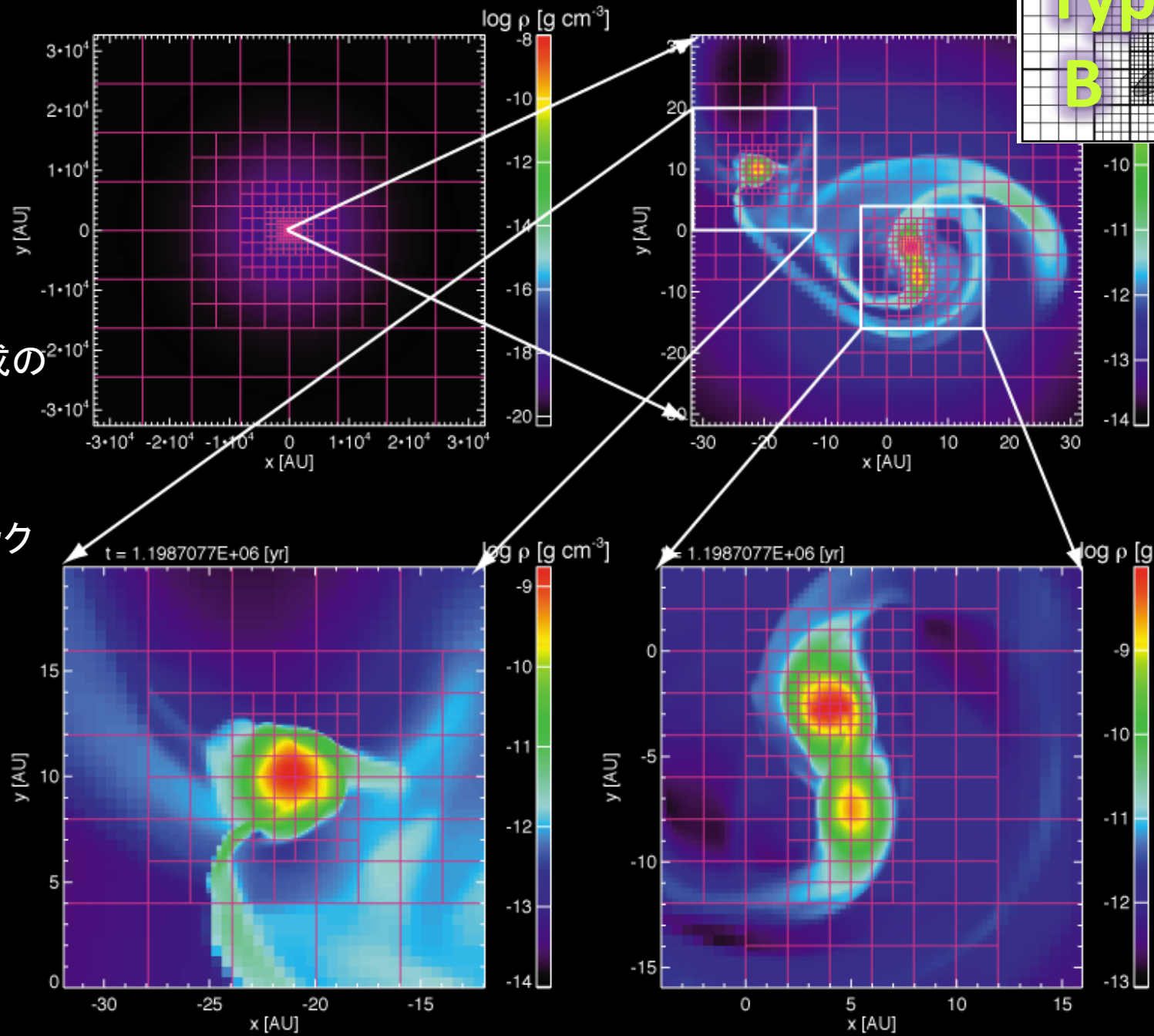




多重星系形成の計算例
SFUMATO

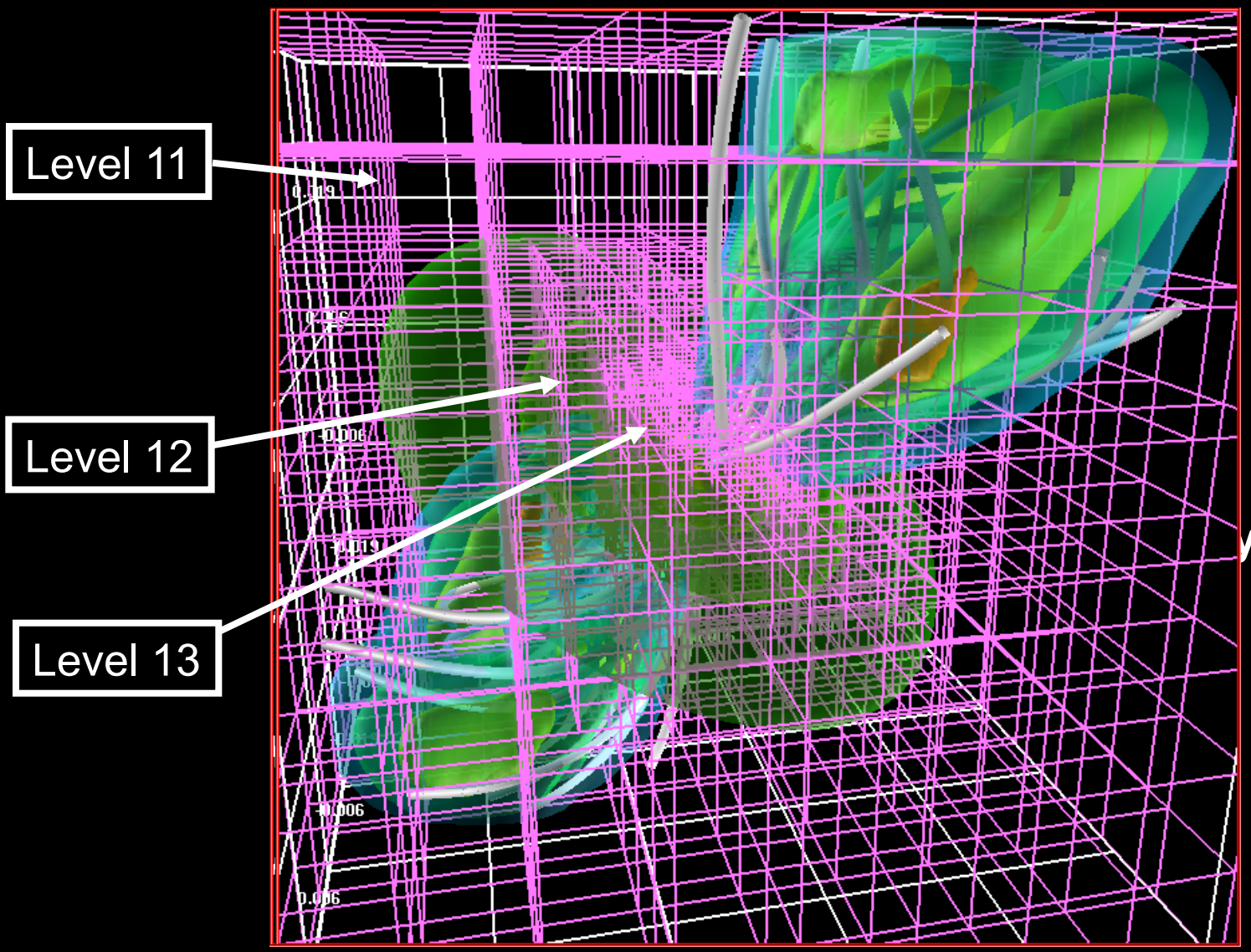
セル数/ブロック
$=8^3$
最大レベル
= 14

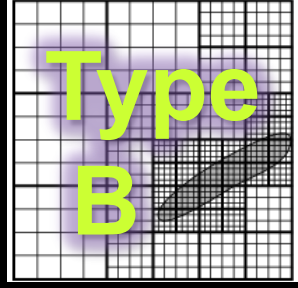




分子雲コア⇒原始星（ファーストコア）
自己重力 + MHD

Matsumoto (2007)

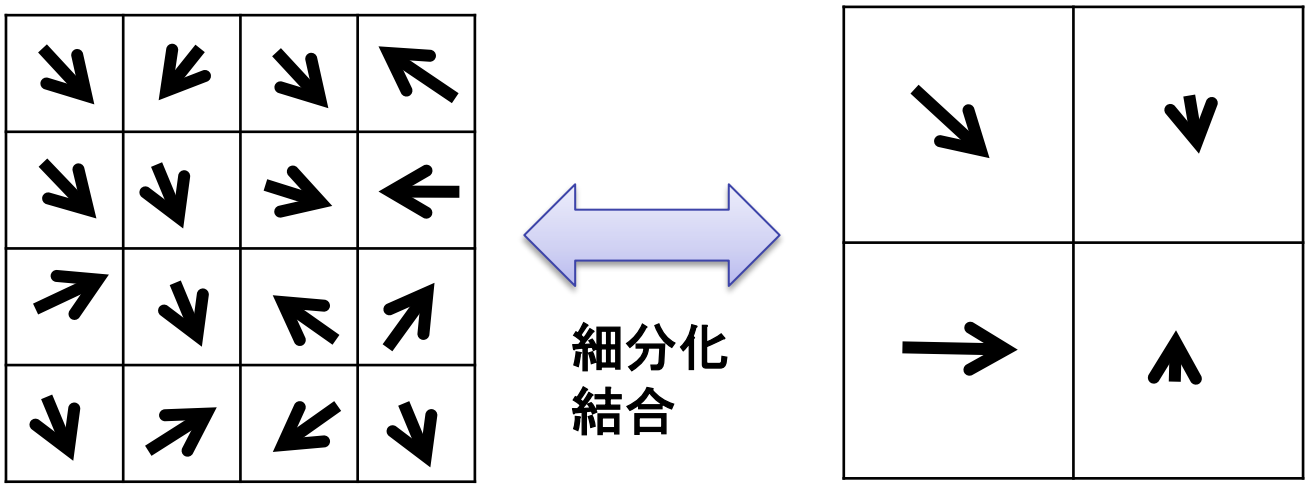




定説
と
実際

AMRは乱流が不得意であるという「定説」

質量・運動量・エネルギーが保存するように細分化と結合



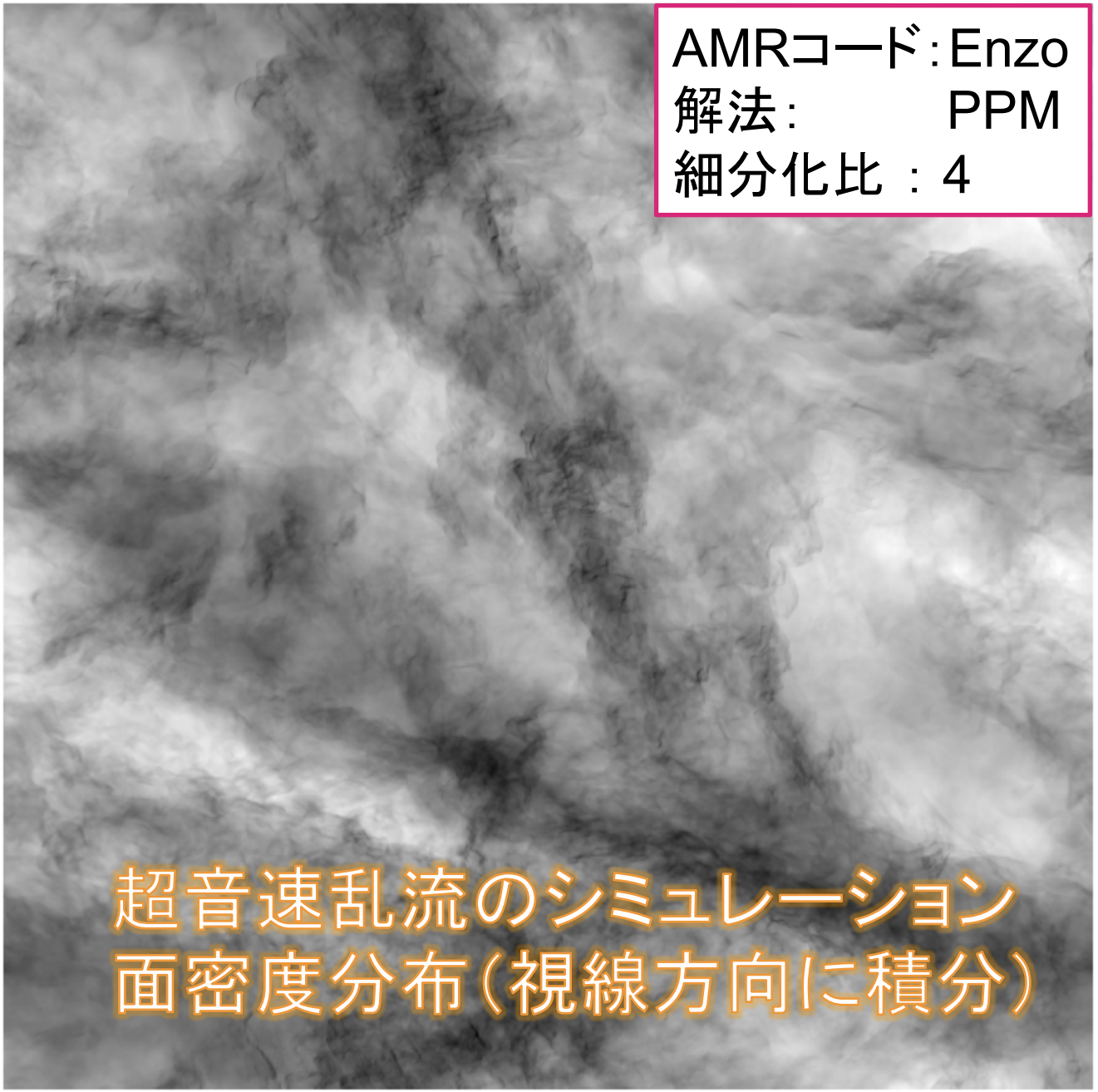
$\rho\vec{v}$ は保存するが、$|\rho\vec{v}|^2$ は保存しない。

全エネルギー (E_K+E_{th}) は保存する。

運動エネルギー(E_K)が熱エネルギー(E_{th})になる。

人為的な加熱に相当する。

AMRコード：Enzo
解法：　　PPM
細分化比 ：4



超音速乱流のシミュレーション
面密度分布（視線方向に積分）

分解能
1024^3
相当

Type A

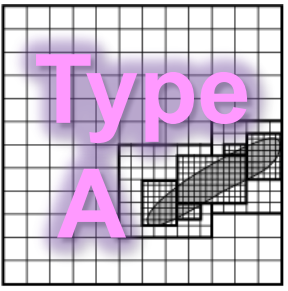
Kritsuk et al. 2006

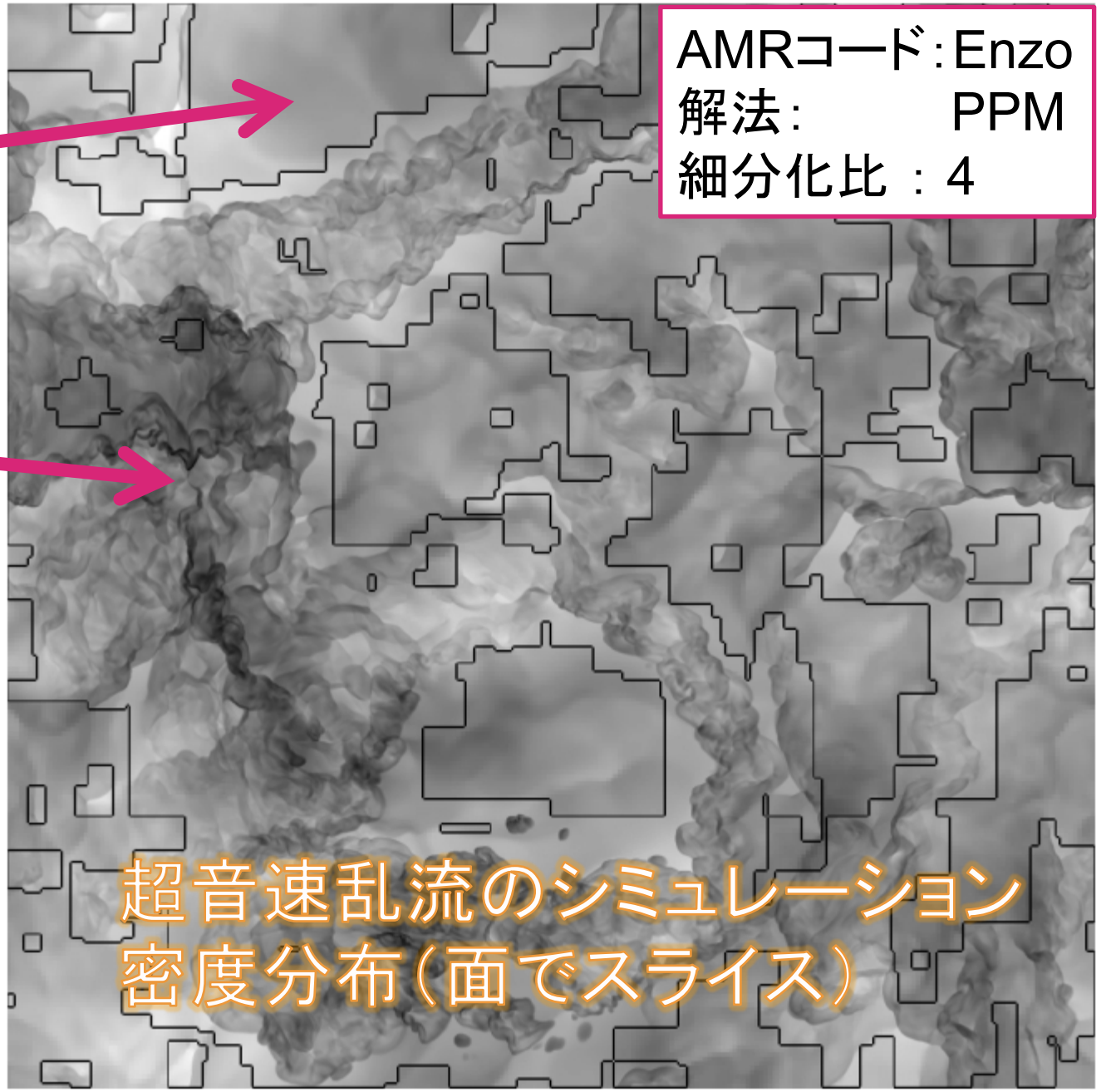
level 0
256^3

Level
1
1024^3

65%

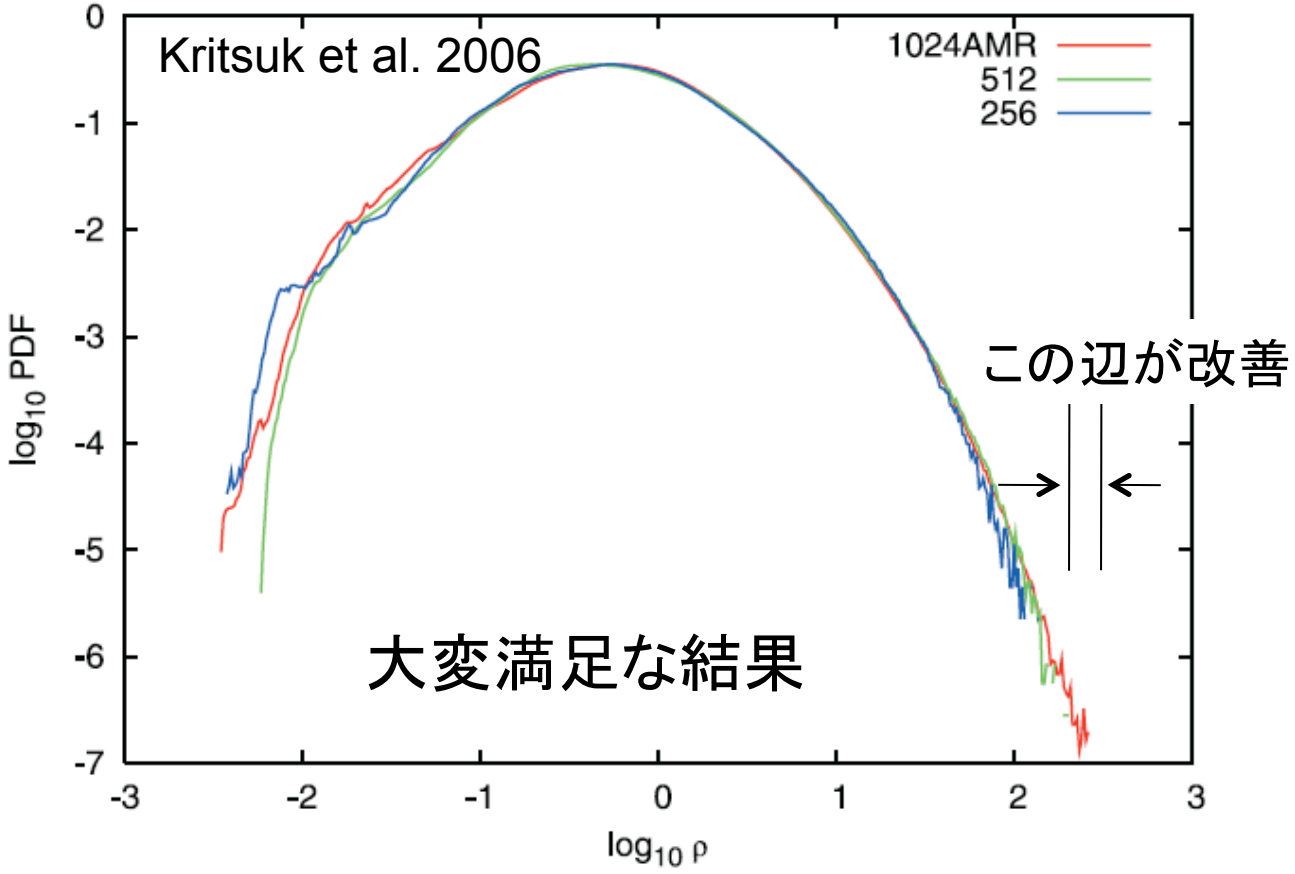
超音速乱流ではOK
亜音速乱流は微妙

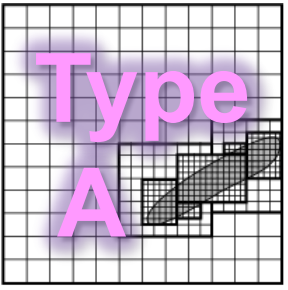




Kritsuk et al. 2006

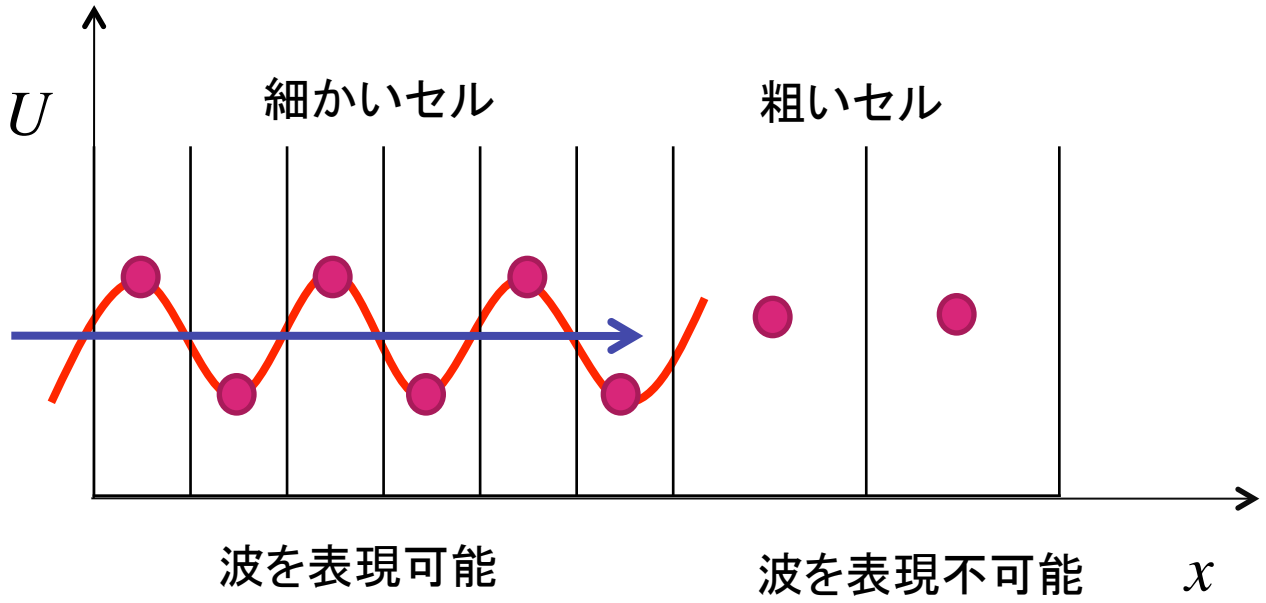
密度のPDF (確率分布関数)
AMRと一様格子の比較





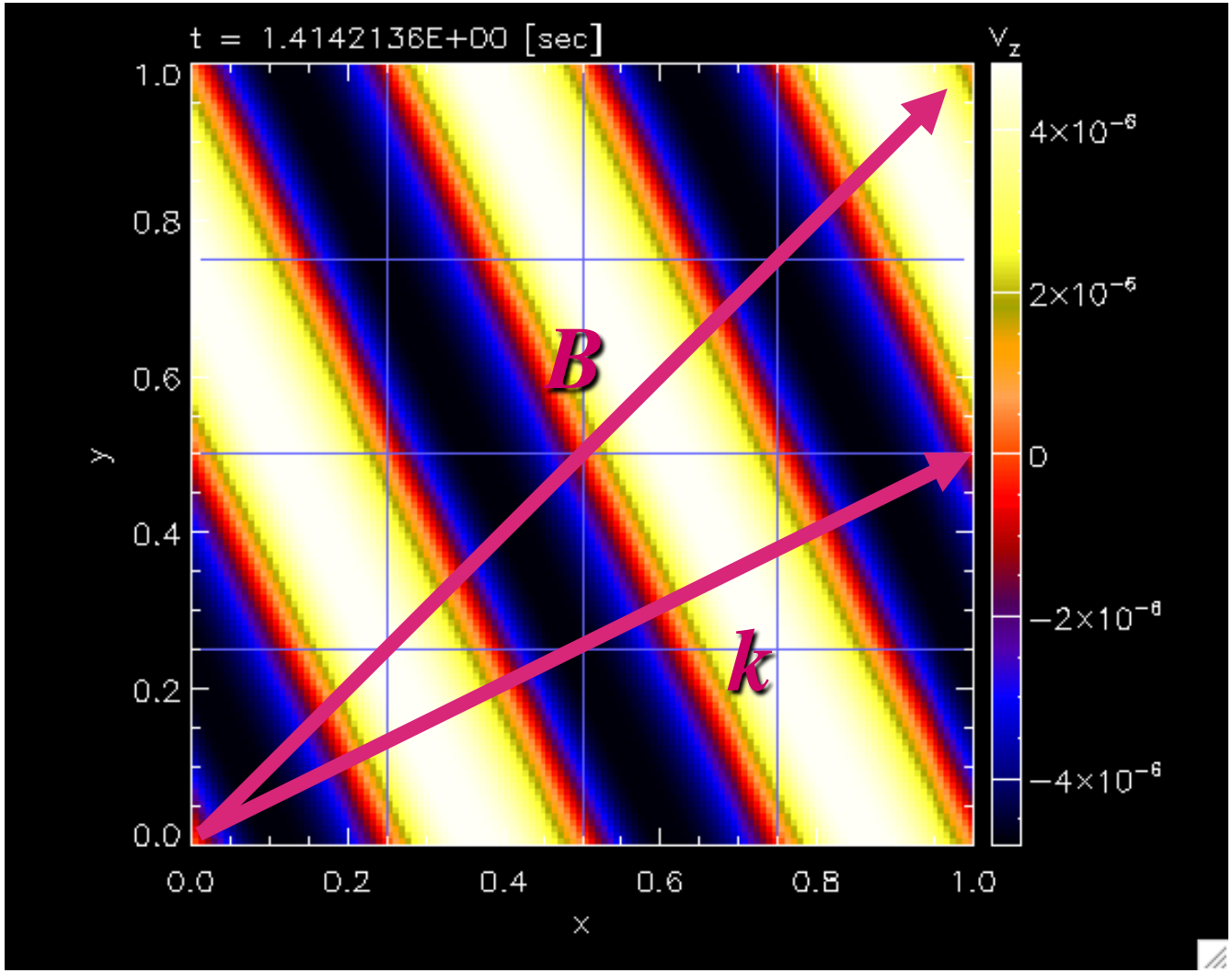
超音速乱流ではAMRは有効（衝撃波の補足）
亜音速乱流では不向き（渦の補足）

AMRをちゃんと作れば「波は反射しない」という「誤解」



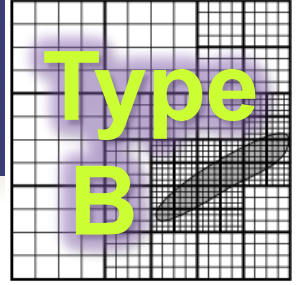
保存形式で解いているけど波のエネルギーはどこへ？
→　反射するしかない

Convergence test for MHD (1/2)
Alfven wave

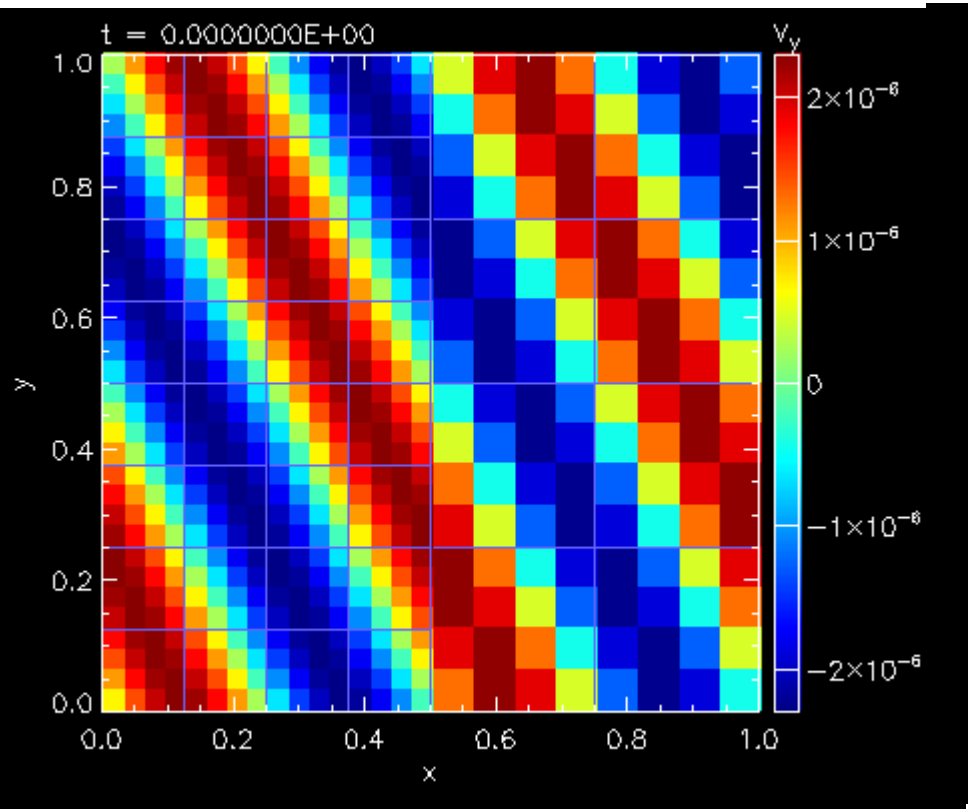


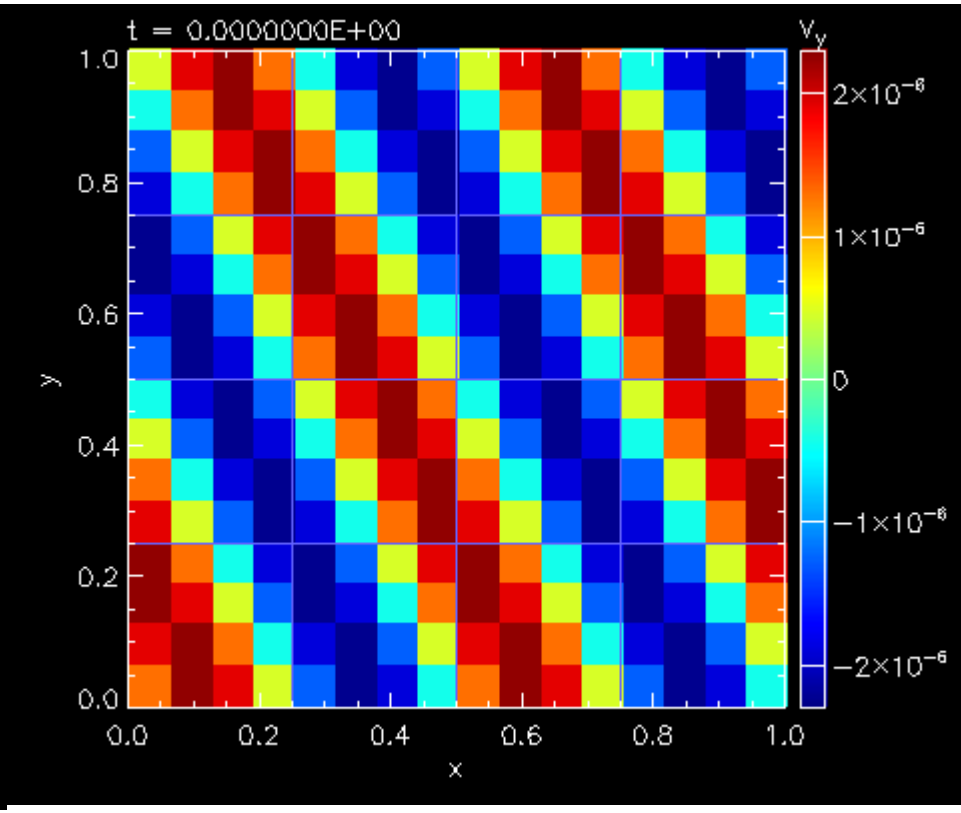
紙面に垂直な
vとBが振動する
sin波

しかし、波は境界で反射する。
Fast wave の反射実験。



SFUMATO

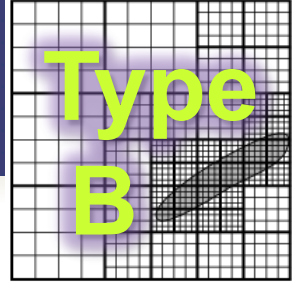




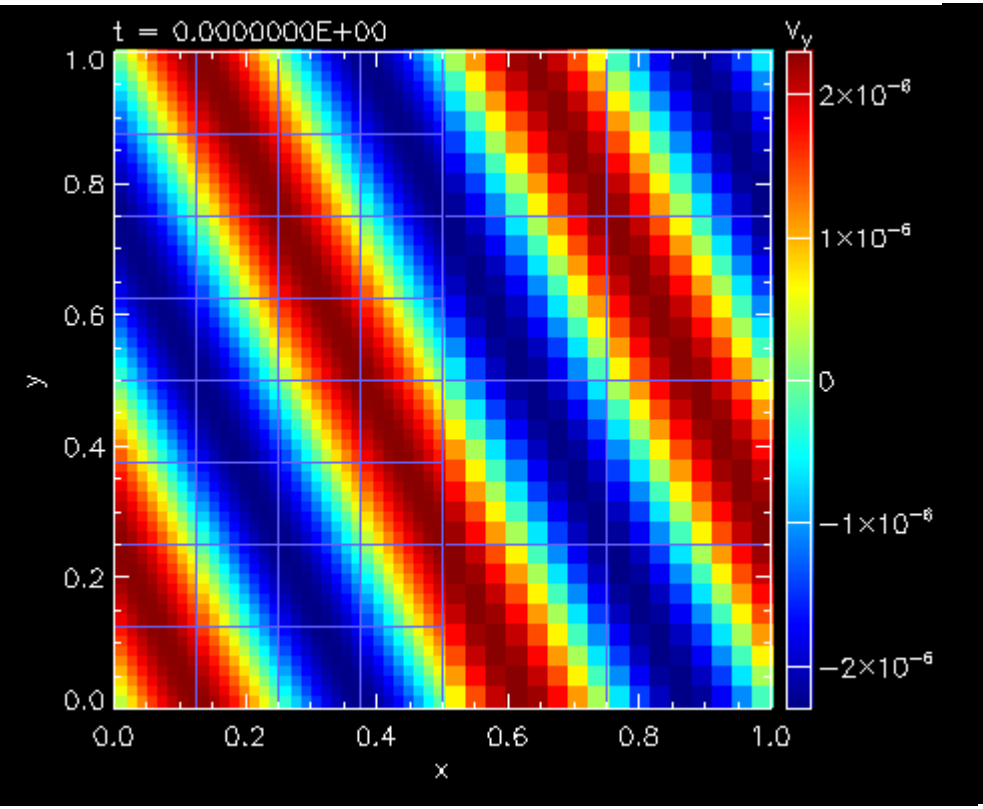
AMR
h = 1/32, 1/16

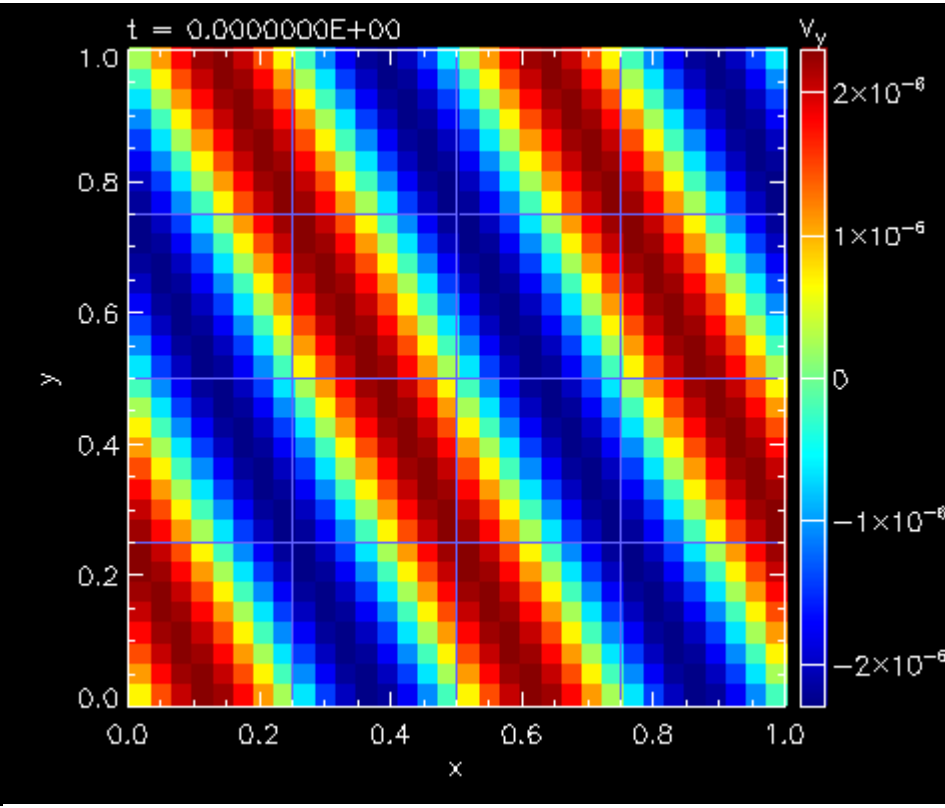
一様格子
h = 1/16

波は境界で反射する。
AMRでは減衰が少ない。



SFUMATO

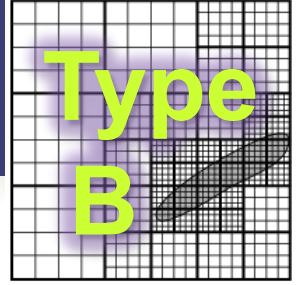




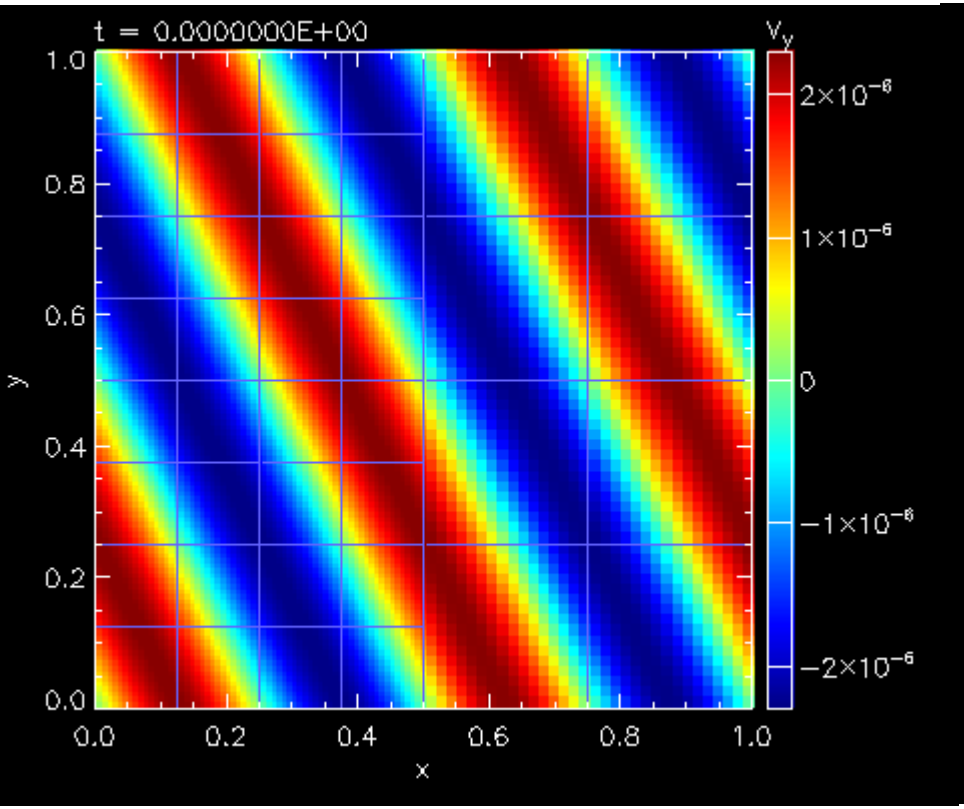
AMR
h = 1/64, 1/32

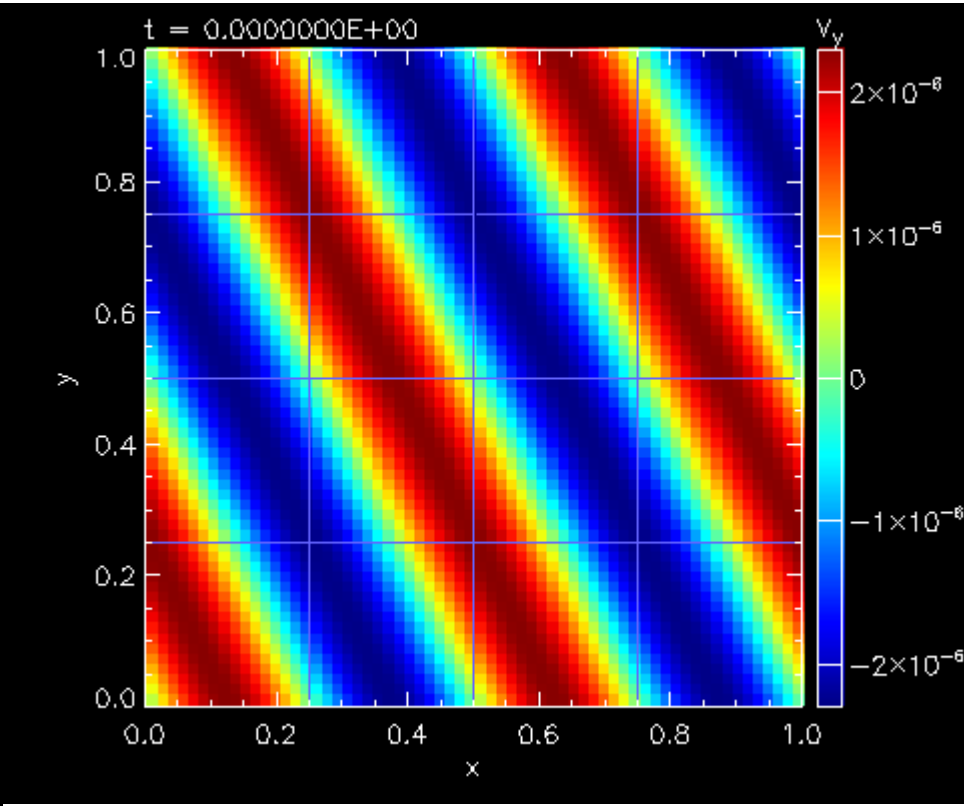
一様格子
h = 1/32

格子が細かくなると反射は少なくなる。



SFUMATO

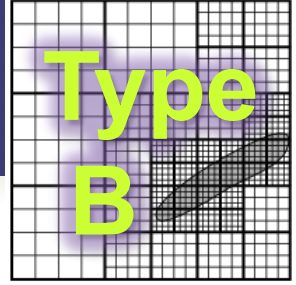




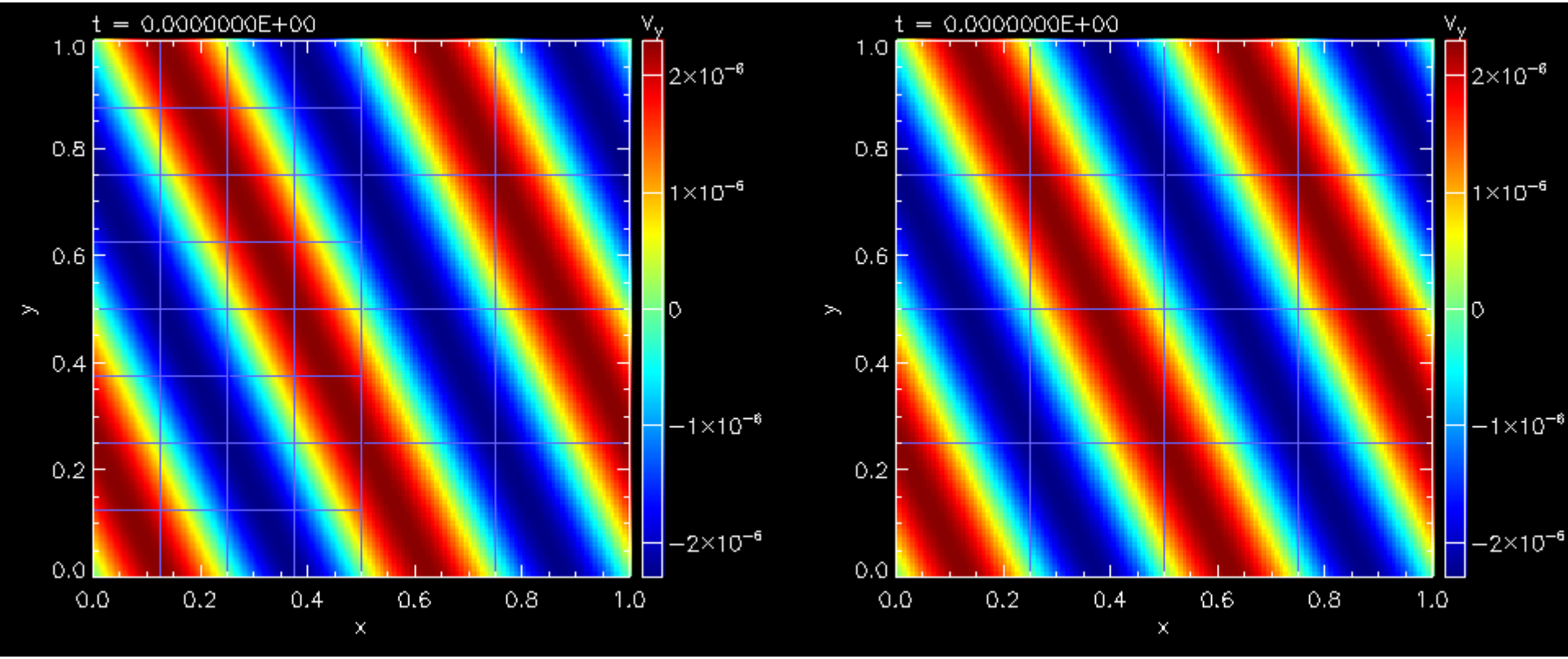
AMR
h = 1/128, 1/64

一様格子
h = 1/64

格子が細かくなると、
ほとんど反射は目視できない。



SFUMATO



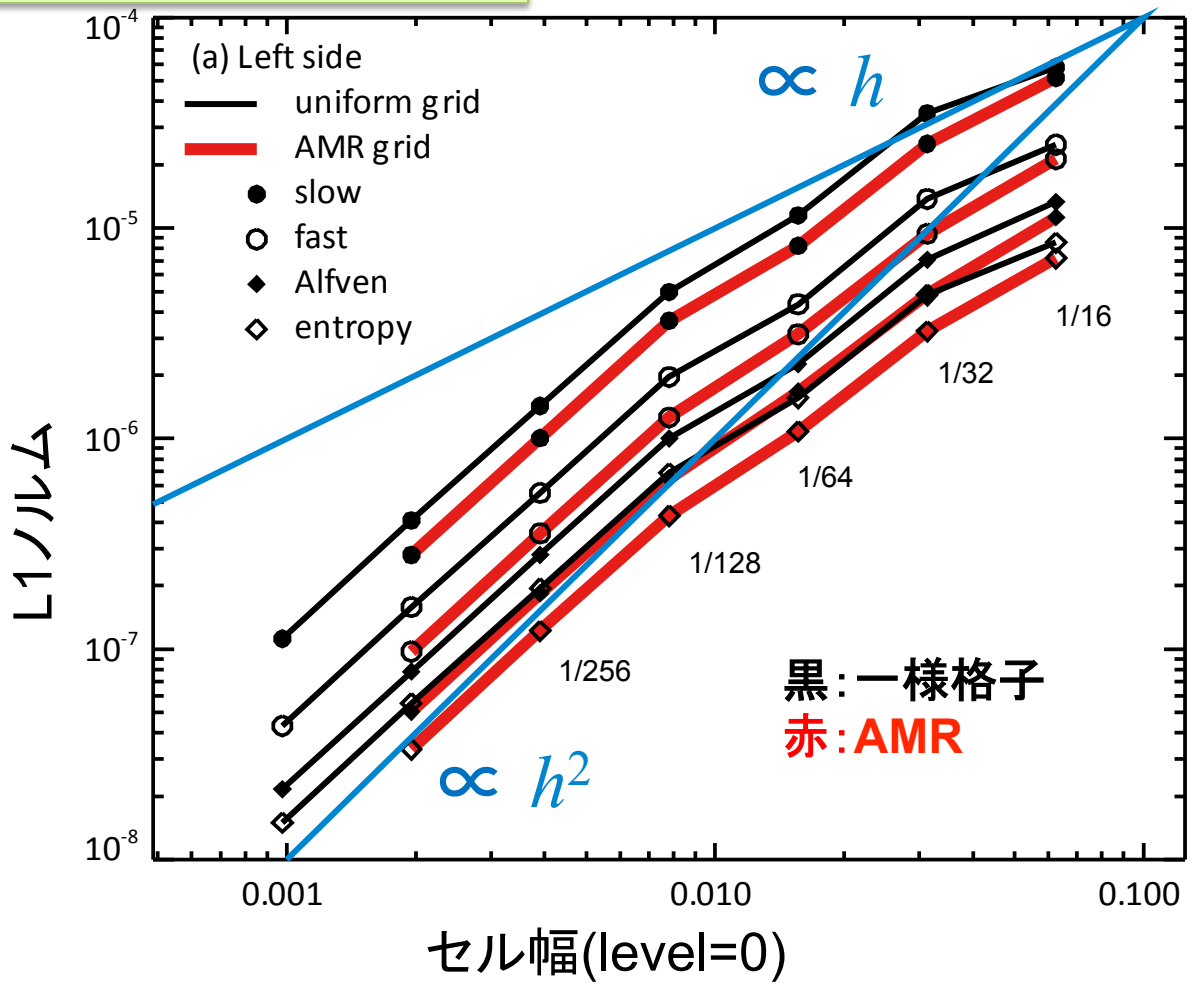
AMR
h = 1/256, 1/128

一様格子
h = 1/128

反射してもスキームの精度を達成する。

2次精度 @ h << 1
1次精度 @ h～1

SFUMATO



実装

AMRコードを作るか作らないか

作る場合

- 利点： 新しい物理の導入が容易。
- 欠点： 膨大なテスト計算の必要性。
  開発時間の必要性
- 開発期間： 流体（MHD）を「書く」だけなら数ヶ月間。
  - 流体部分はラク。松本は流体部分を2週間で集中的に書いた。
  - 自己重力部分の開発は苦労した。
  - 可視化にも苦労した。
- 効率よく開発するには
  - 流体などの既存のソルバを流用する。
  - はじめから3次元版を作る。
  - はじめから並列版を作る。
  - 既存のAMRライブラリ（paramesh, chombo）を利用するのも一考かと。

AMRコードを作るか作らないか

作らない場合

- 既存の公開コードを利用する。
  現在の主流になりつつある。
  シミュレーションコードの複雑化が原因。

- 利点： 手軽に利用できる。開発の不要（テスト計算は必要）。

- 欠点： 改造の困難さ。新しい物理を導入する場合に苦労する。

- アイディア勝負の場合（隙間産業）は、作らない方が良いと思う。

- 開発者と共同研究する。
  - ブレイクスルーの仕事では、開発者が共同研究者であることが多い。

AMRコードとその他の自社製コードの比較

Q. どのくらい大変か？

A. 一様格子の10倍以上大変

コード種別	3dhd	Nested grid	SFUMATO AMR
行数	3,524 （自動生成後6,994)	14,037	47,476
言語	Fortran77 + Perl コード自動生成	Fortran77 + 自作 cpp	Fortran90
並列化	VPP Fortran 指示行の挿入	自動並列 指示行の挿入	MPI並列
バージョン	-	2.12	3.8
実装機能	流体, 自己重力	HD, MHD, 自己重力, 磁気拡散、(sink cell, FLD)	HD, MHD, 自己重力, sink 粒子, 拡散方程式, オーム散逸

解法

AMR格子における解法

本日の話題提供

- ブロック構造格子
  - ブロック自体は普通の一様格子。
  - 粗細(親子)ブロックの関係(時間的・空間的)を工夫する。それだけ！
- 流体・MHD (双曲型偏微分方程式)

$$\frac{\partial U}{\partial t}+\frac{\partial F}{\partial x}+\frac{\partial G}{\partial y}+\frac{\partial H}{\partial z}=S$$

  - 基本コンセプトは Berger & Colella (1989) に尽きる。
  - TVD, Roe法, HLLD法, predictor-corrector 法 (時間空間2次精度)
  - MHDでは、∇・B　の処方箋に複数の流儀がある。一長一短。
- 自己重力(楕円型偏微分方程式)

$$\nabla\Phi = 4\pi G\rho$$

  - AMRと相性が良いのはマルチグリッド法。
  - お勧め参考書「MULTIGRID」 Trottenberg et al. (2001)
- 拡散方程式(放物型偏微分方程式)

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

  - 楕円型偏微分方程式と同じ

双曲型方程式の解法： 2種類の時間発展

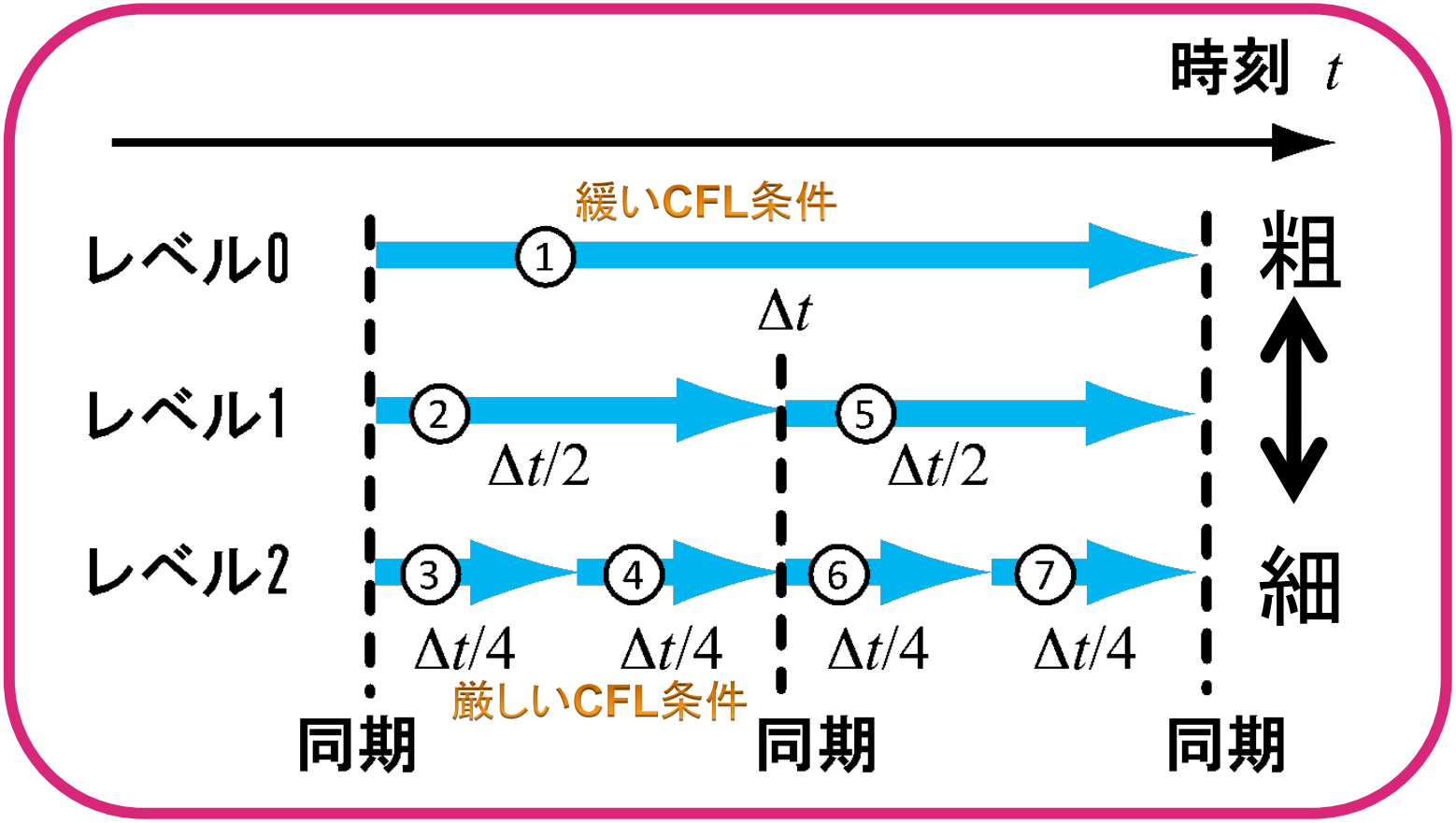
- Adaptive time step
  - 空間だけでなく、時間も細分化する。
  - 親のタイムステップ≧子のタイムステップ
  - C.f., Berger & Colella (1989)
  - 自己重力なしの場合

CFL条件

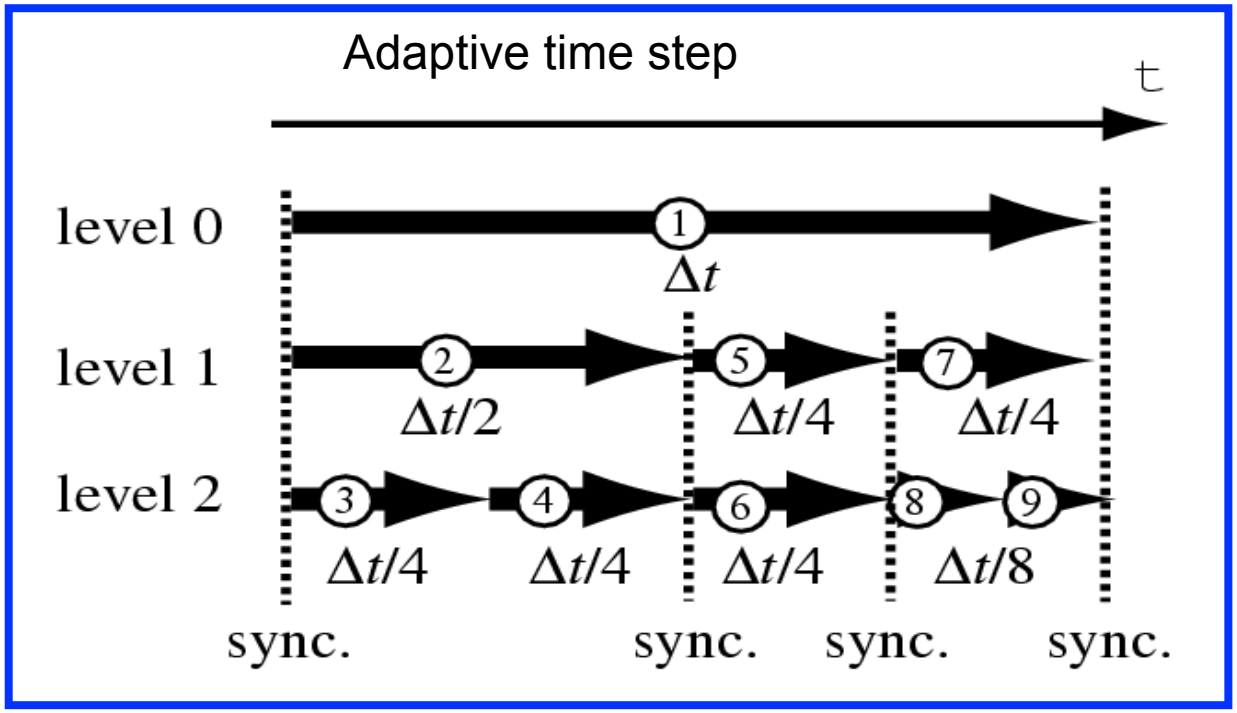
$$\Delta t < \frac{\Delta x}{v}$$

- Synchronous time step
  - 全てのグリッドレベルが同じタイムステップを持つ。
  - 自己重力系、輻射流体の場合 （長距離相互作用）

適合時間ステップ(adaptive time step)
時間も適合細分化



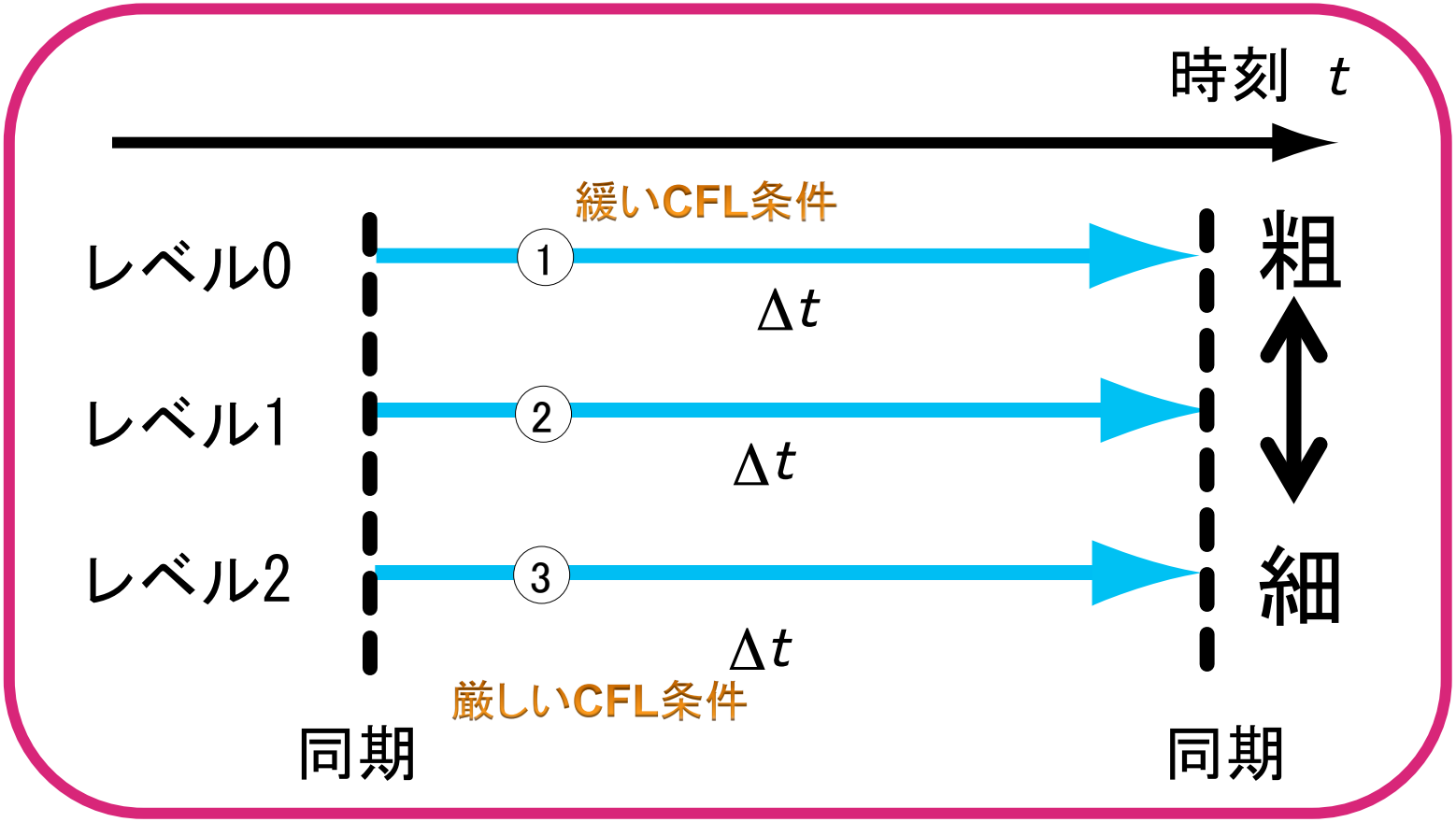
適合時間ステップ(adaptive time step) オプション



子グリッドの時間刻み　$\Delta t_{子} = \Delta t_{親}/2^{n}$　$(n=0, 1, 2, \ldots)$

子グリッドがCFL条件を破るのを防ぐ
子グリッドが律儀に親のCFL条件を守る制限を解除する。

同期時間ステップ(synchronous time step)
時間を適合細分化しない



Adaptive vs synchronous time steps

Adaptive time step

もっとも細かいグリッドレベルが1ステップ進むためのコスト
親のグリッドレベルは1/2ステップ
祖父のグリッドレベルは1/4ステップ

$$1+\frac{1}{2}+\frac{1}{4}+\cdots+\frac{1}{2^{l\max}}=\sum_{l=0}^{l\max}\frac{1}{2^{l}}\approx 2 \qquad (l_{\max} >> 1)$$

トータルで2倍コストがかかる

Synchronous time step

もっとも細かいグリッドレベルが1ステップ進むためのコスト
親のグリッドレベルも1ステップ
祖父のグリッドレベルの1ステップ

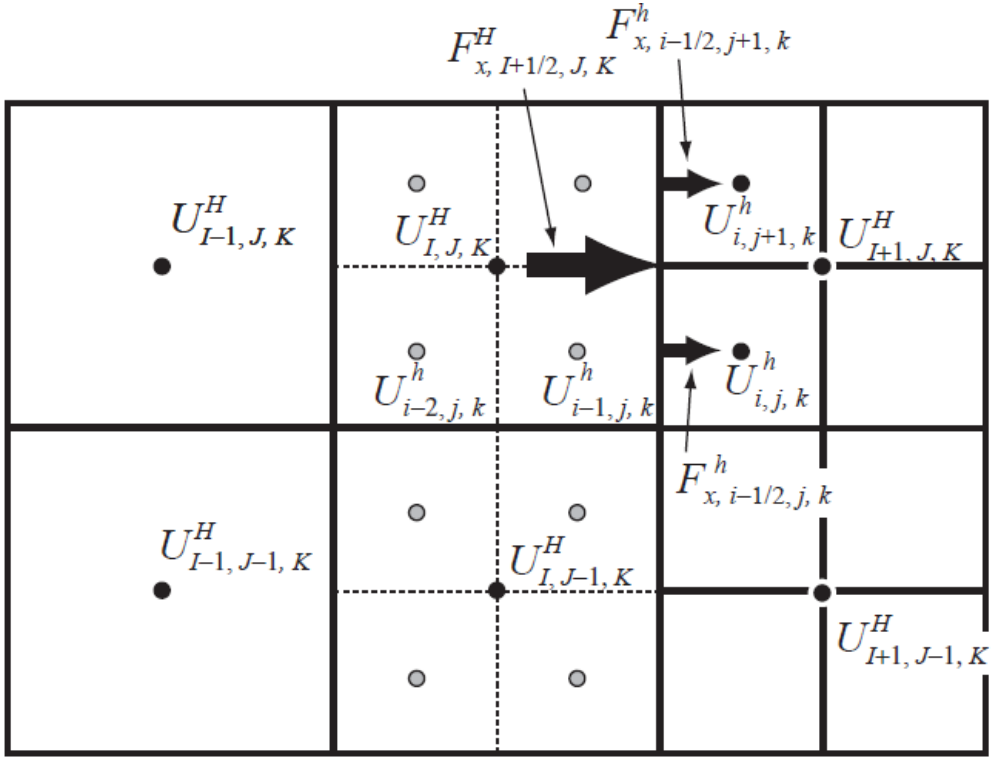
$$1+1+1+\cdots+1=\sum_{l=0}^{l\max}1=l_{\max}+1\approx l_{\max} \qquad (l_{\max} >> 1)$$

トータルで段数倍コストがかかる

特別な理由がない限り、adaptive time step を採用するべき。
特別な理由の例：自己重力、輻射

双曲型方程式の解法
細粗ブロックの境界

- 粗いブロックと接する**細かいブロック**
  - 粗いブロックを時間的に空間的に補間し、細かいブロックの境界条件にセット。
- 細かいブロックと接する**粗いブロック**
  - 細かいブロックの数値流速を使って、時間発展。

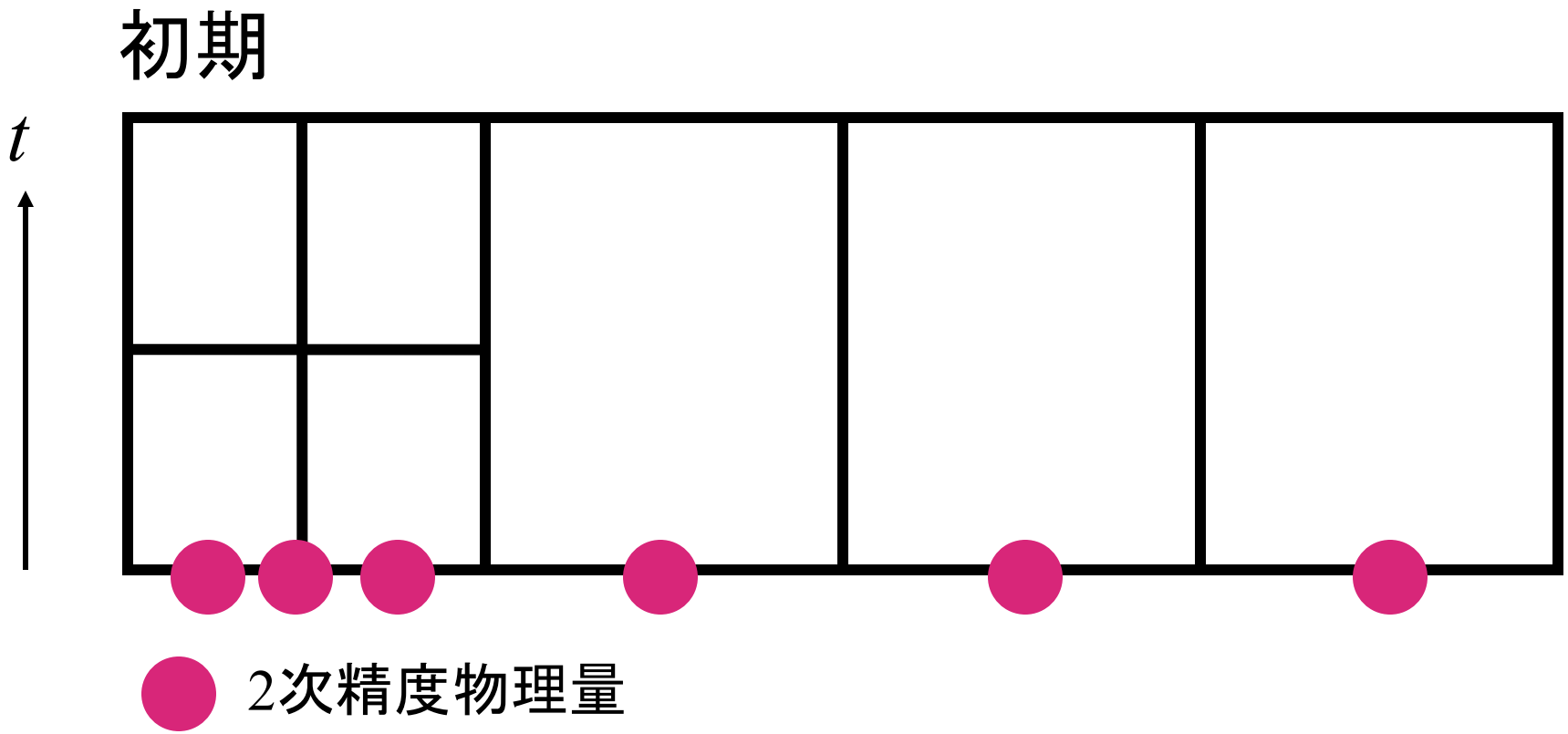


時間推進の方法

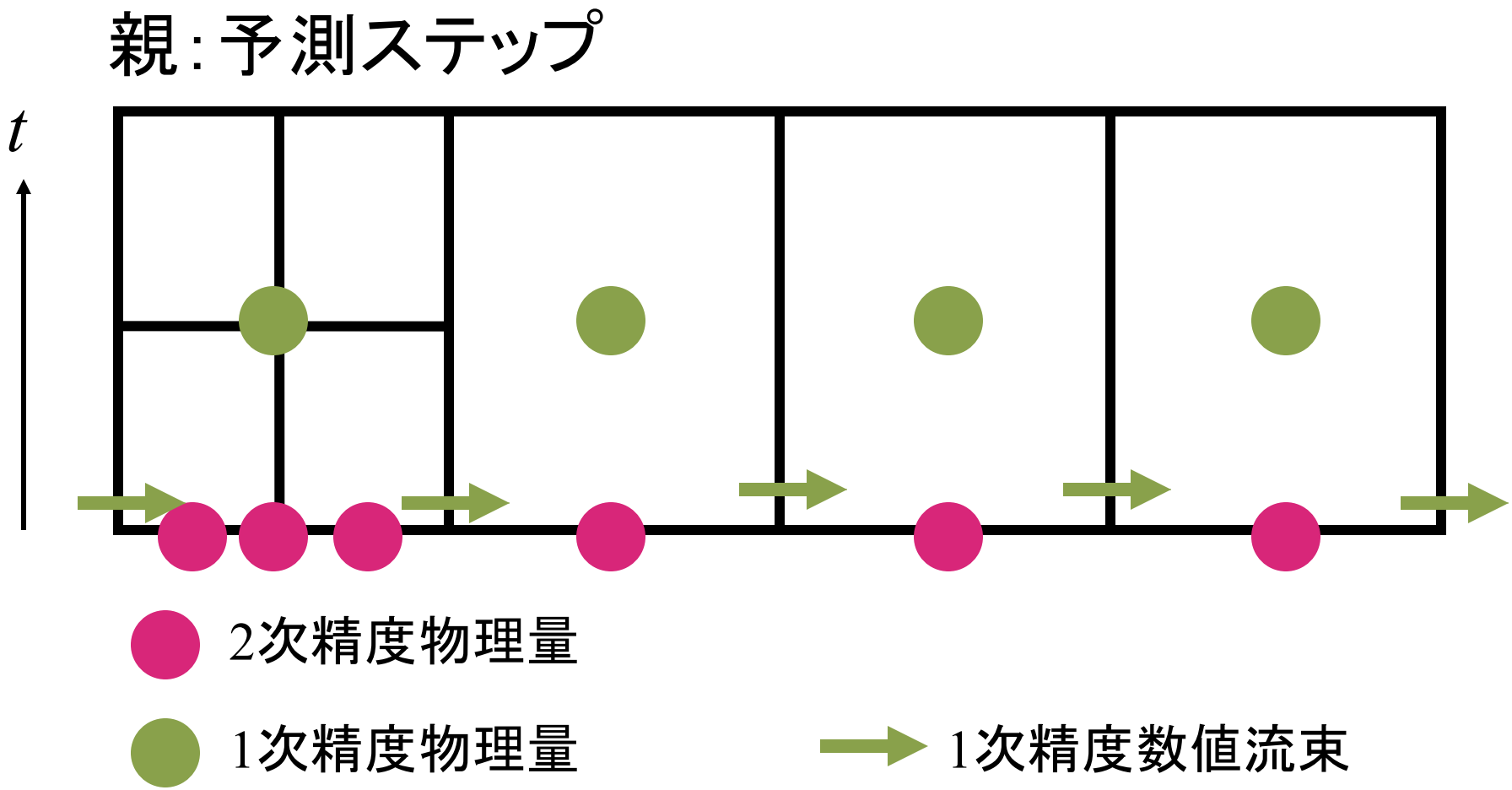
1. 時間を進めるべきレベルを探す。
   a. もっとも時間が進んでいないレベルのうち、もっとも粗いレベル。
2. 時間刻み幅dtを求める。
   a. レベル0はクーラン条件から。
   b. それ以外は親のdtの半分
3. 境界条件の設定
   a. 兄弟ブロックからコピー。
   b. 親ブロックから時間・空間的に内挿。
   c. ユーザの境界条件。
4. 通常の方法で時間推進する。
   a. たとえば、predictor-corrector 法など。
5. 親レベルと同期した場合：
   a. 自分の解を親セルに代入する。
   b. 隣接する親セルの値を修正する（数値流束保存）。
6. 同期している全てのレベルを用いて細分化をする。

l_{sync} ～ $l_{\mathrm{max}} - 1$ を細分化して、
$l_{\mathrm{sync}} + 1$ ～ l_{max} を生成する。
l_{sync} = 同期している最粗レベル

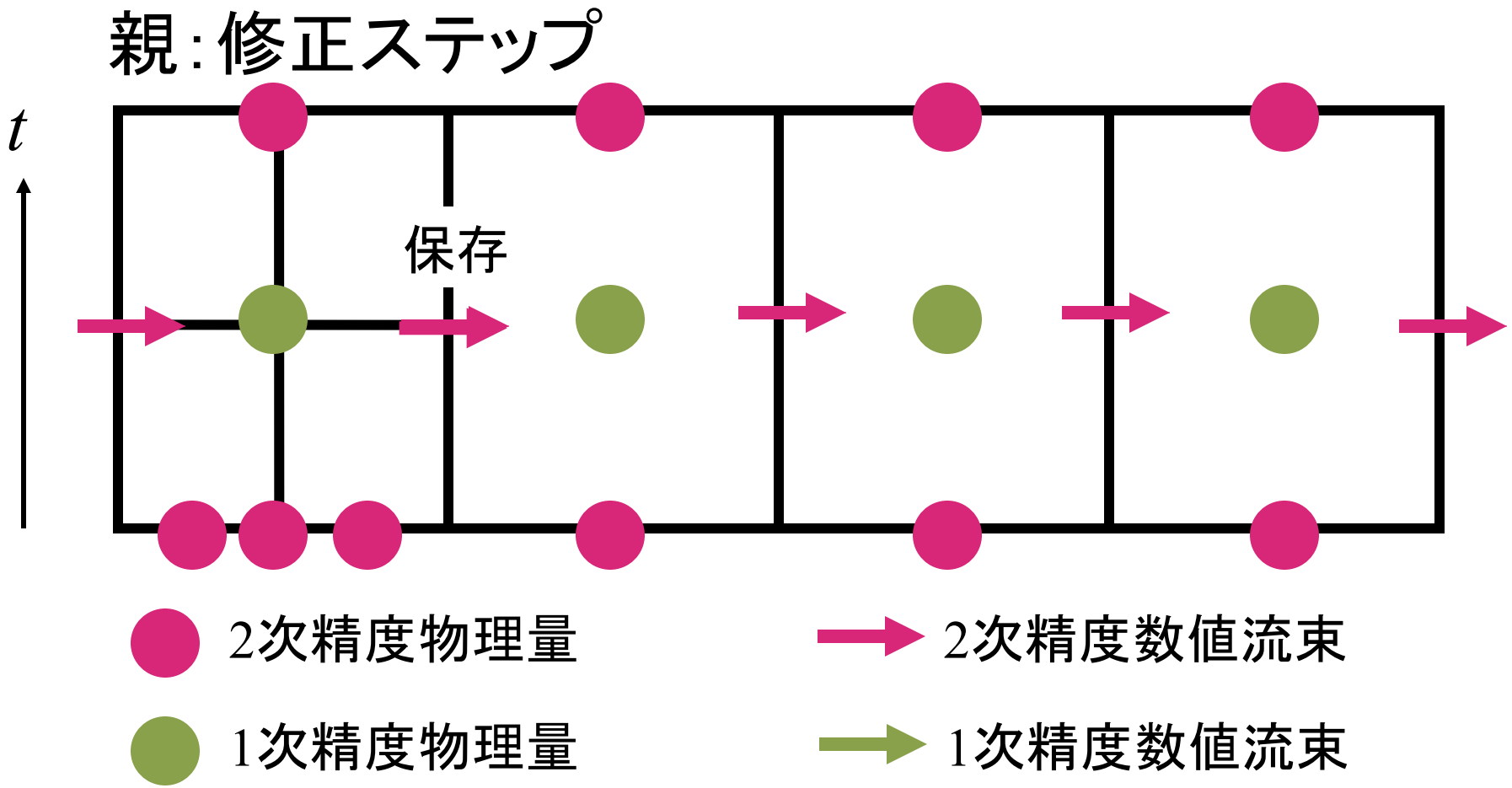
コードの詳細： 流体部分
数値流束補正・マルチタイムステップ



コードの詳細： 流体部分
数値流束補正・マルチタイムステップ

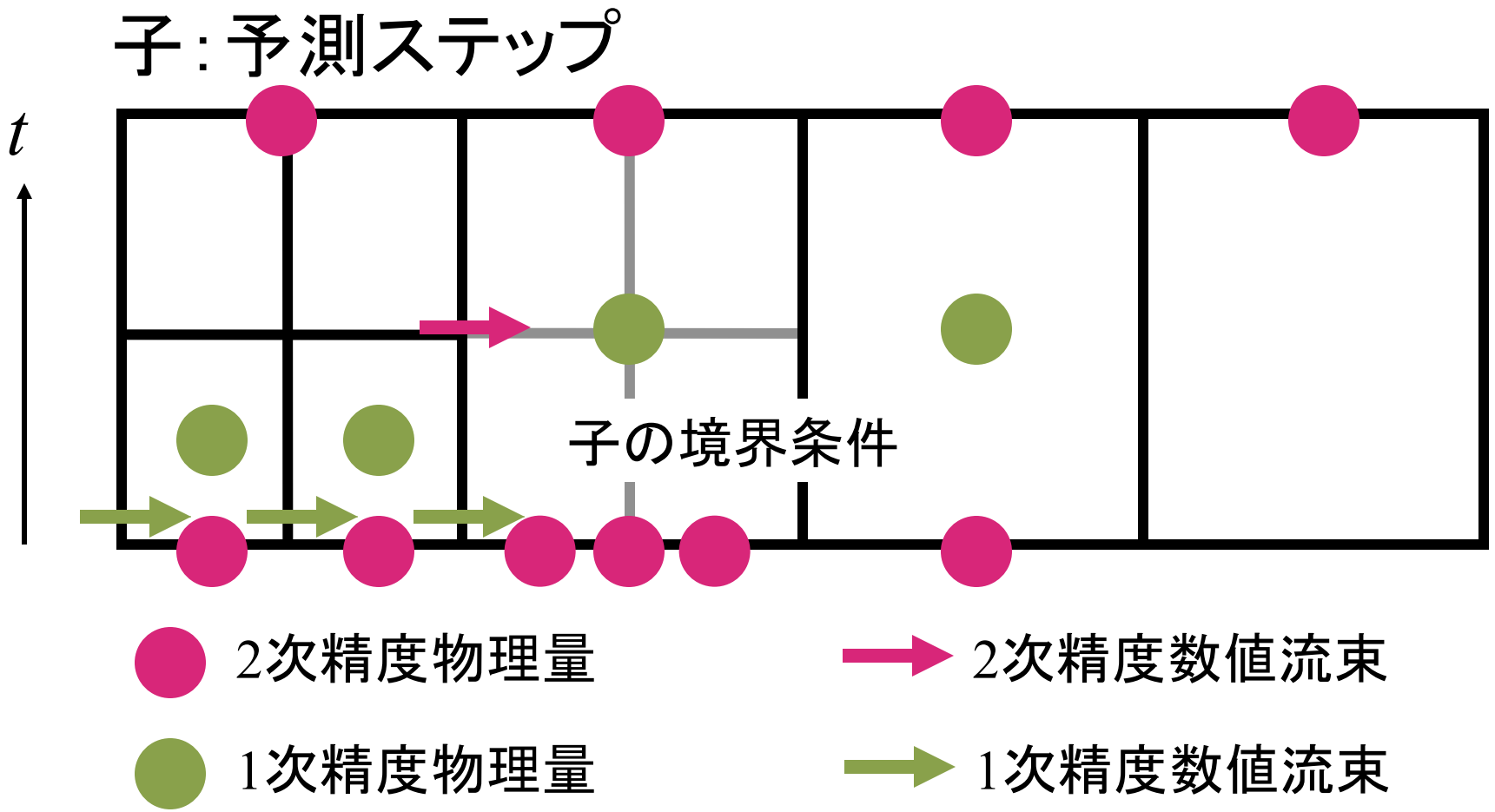


コードの詳細： 流体部分
数値流束補正・マルチタイムステップ

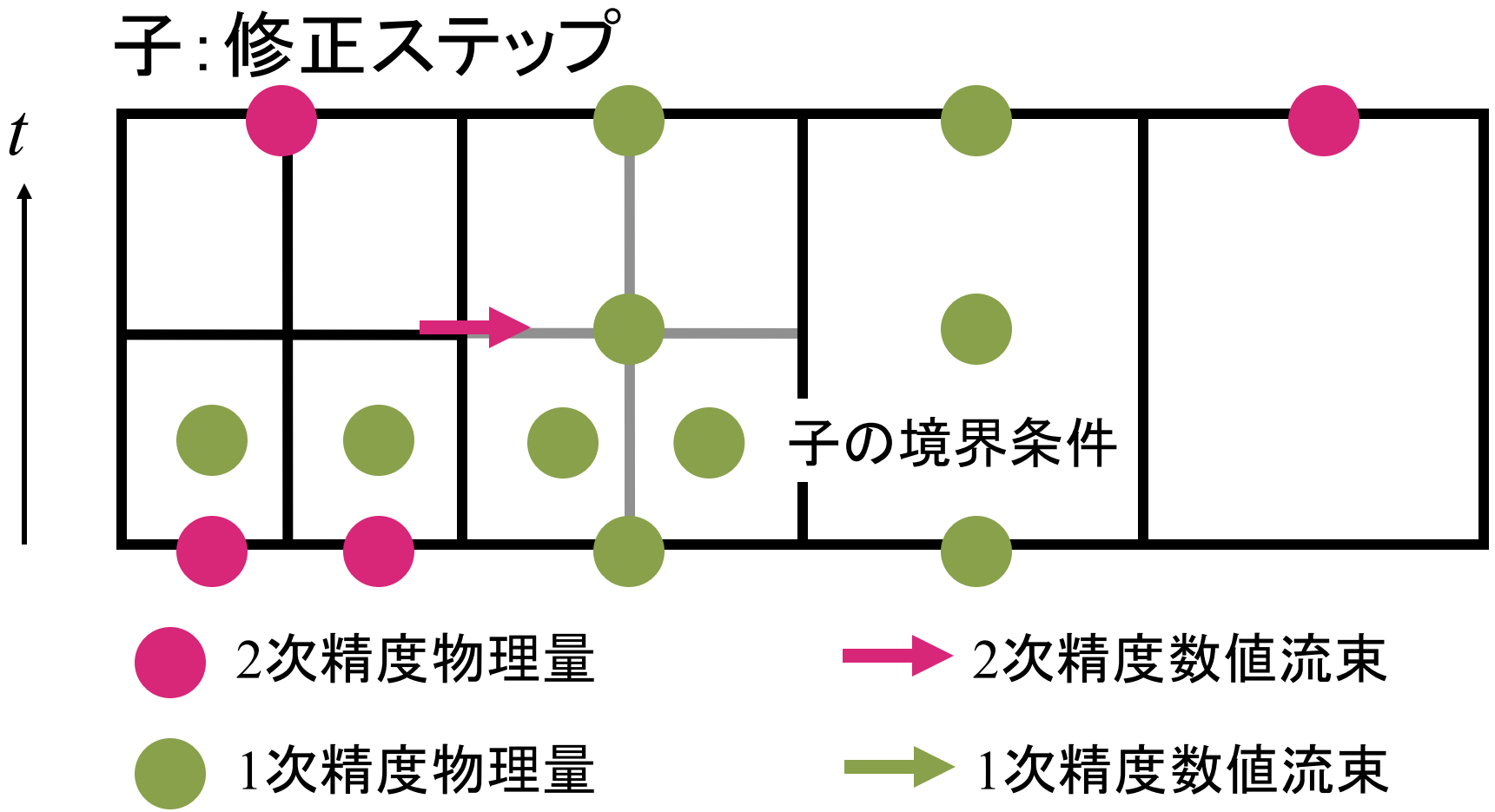


click

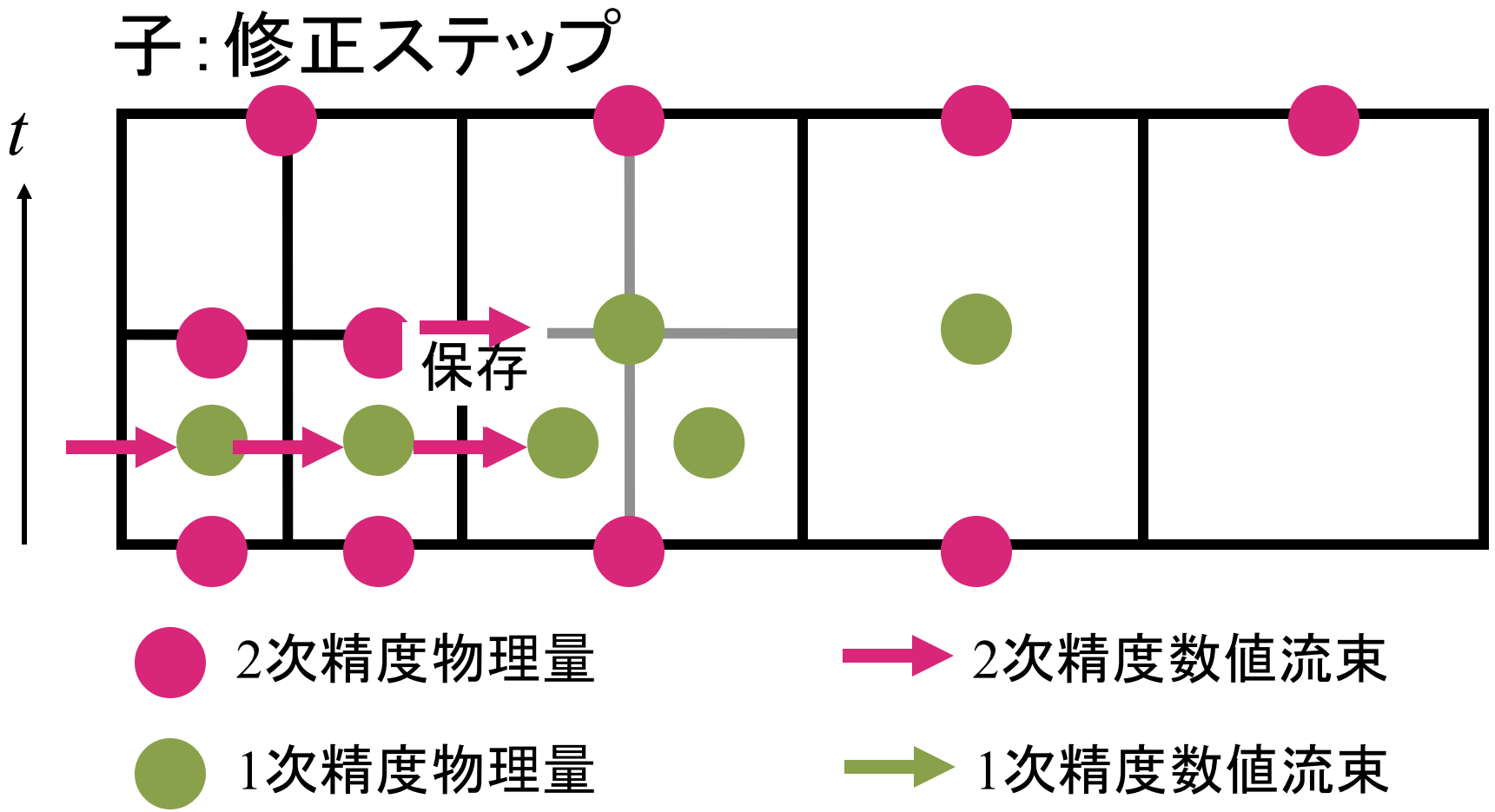
コードの詳細： 流体部分
数値流束補正・マルチタイムステップ



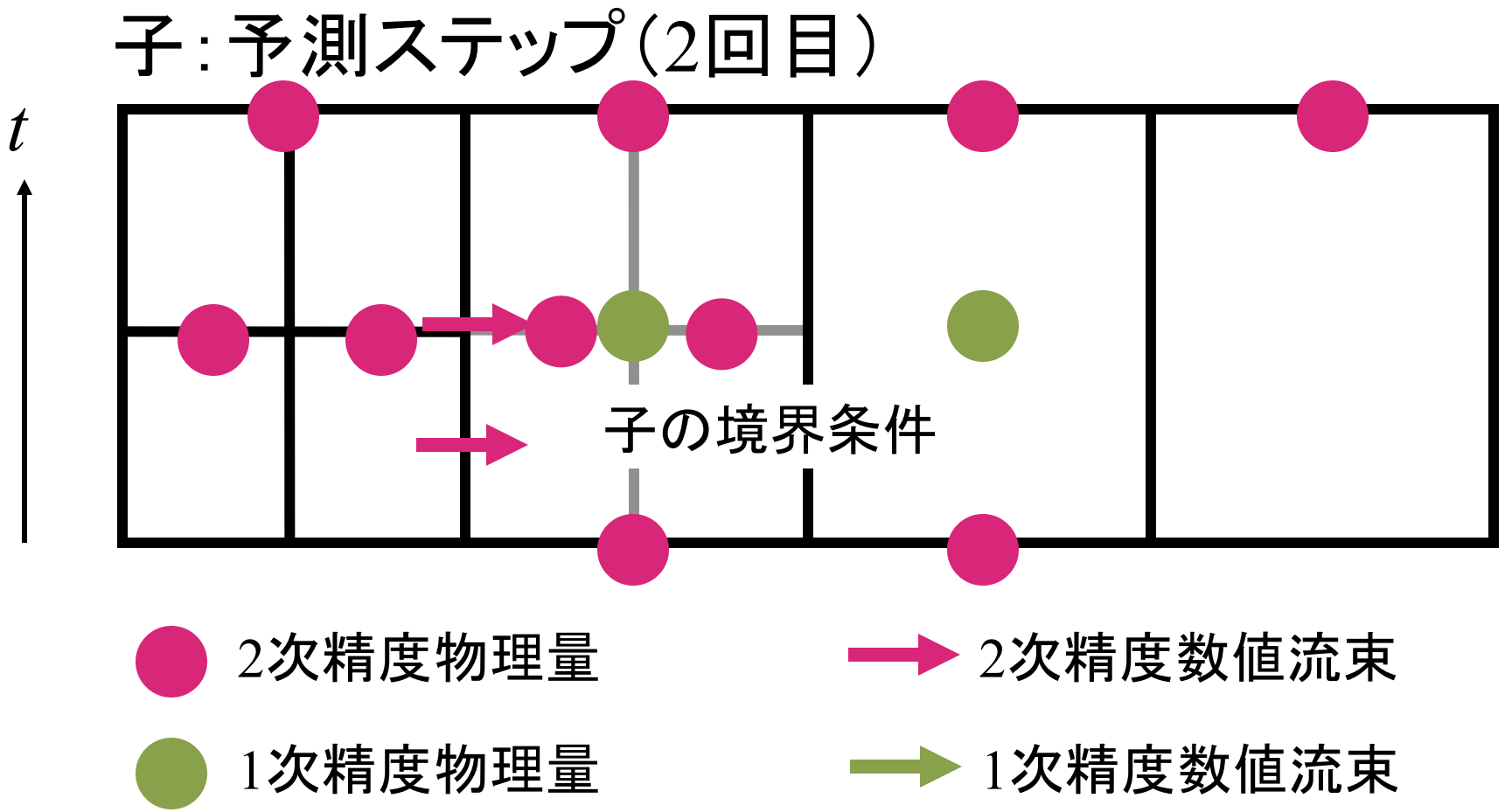
コードの詳細： 流体部分
数値流束補正・マルチタイムステップ



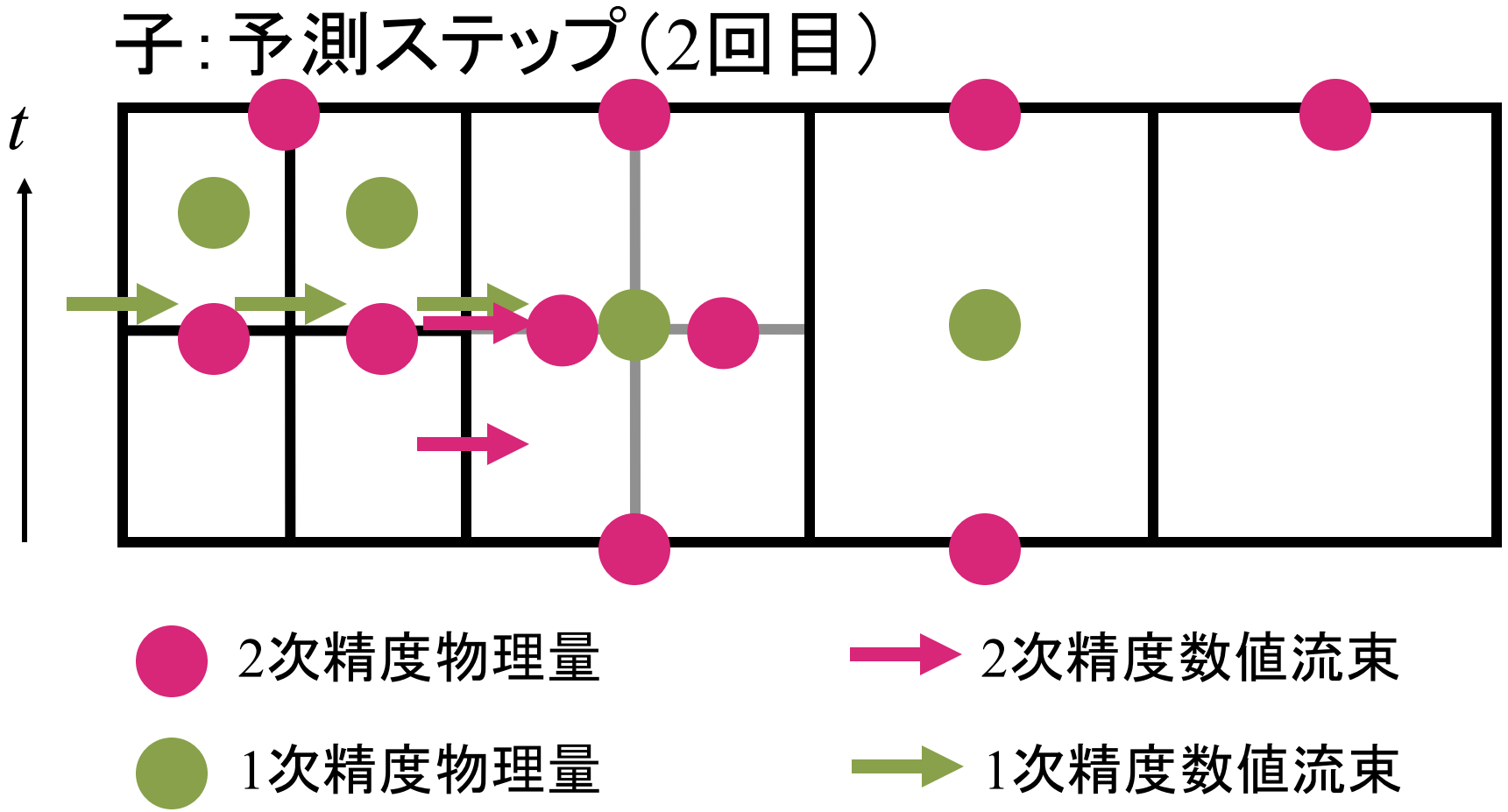
コードの詳細： 流体部分
数値流束補正・マルチタイムステップ



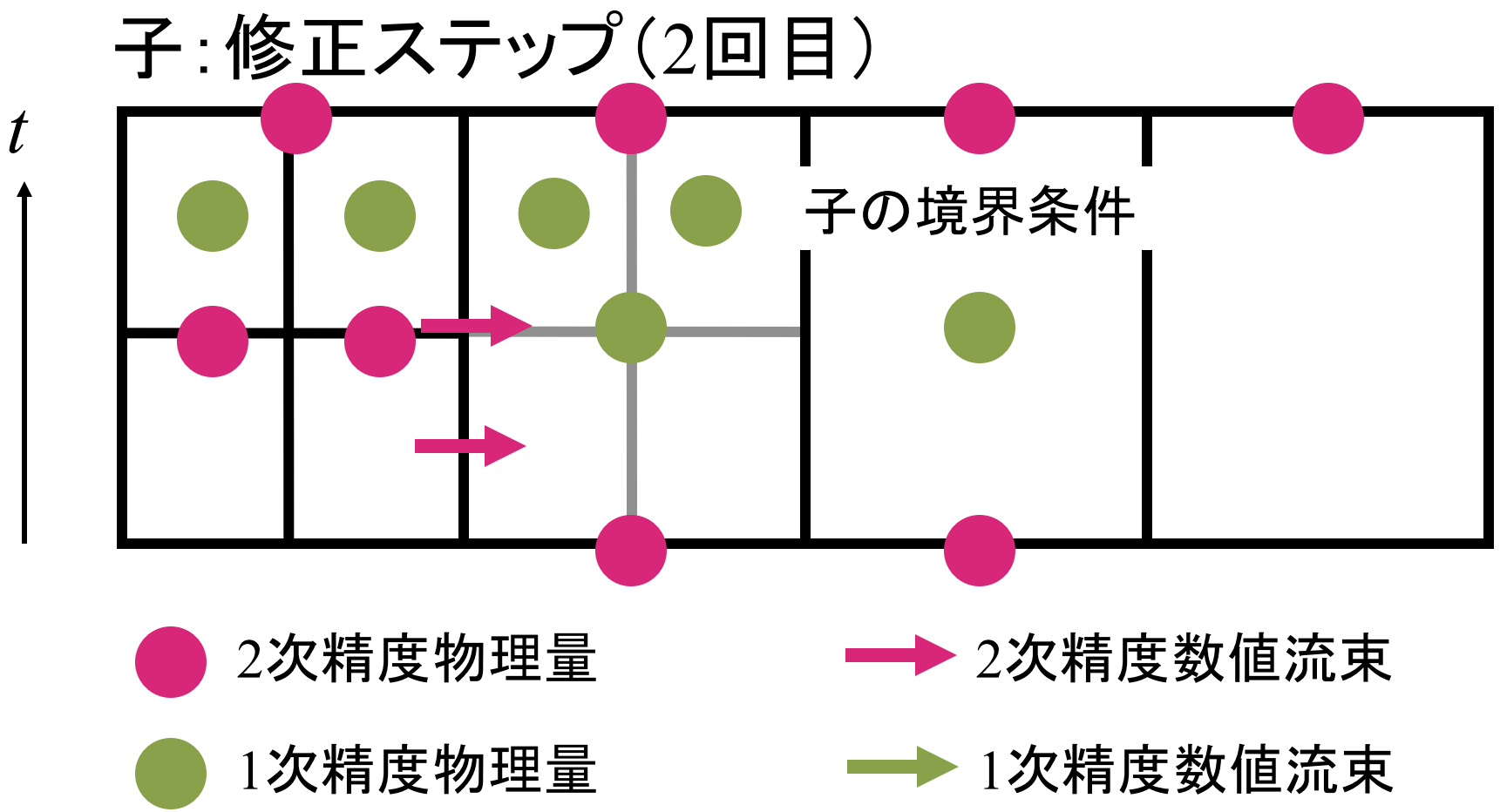
コードの詳細： 流体部分
数値流束補正・マルチタイムステップ



コードの詳細： 流体部分
数値流束補正・マルチタイムステップ

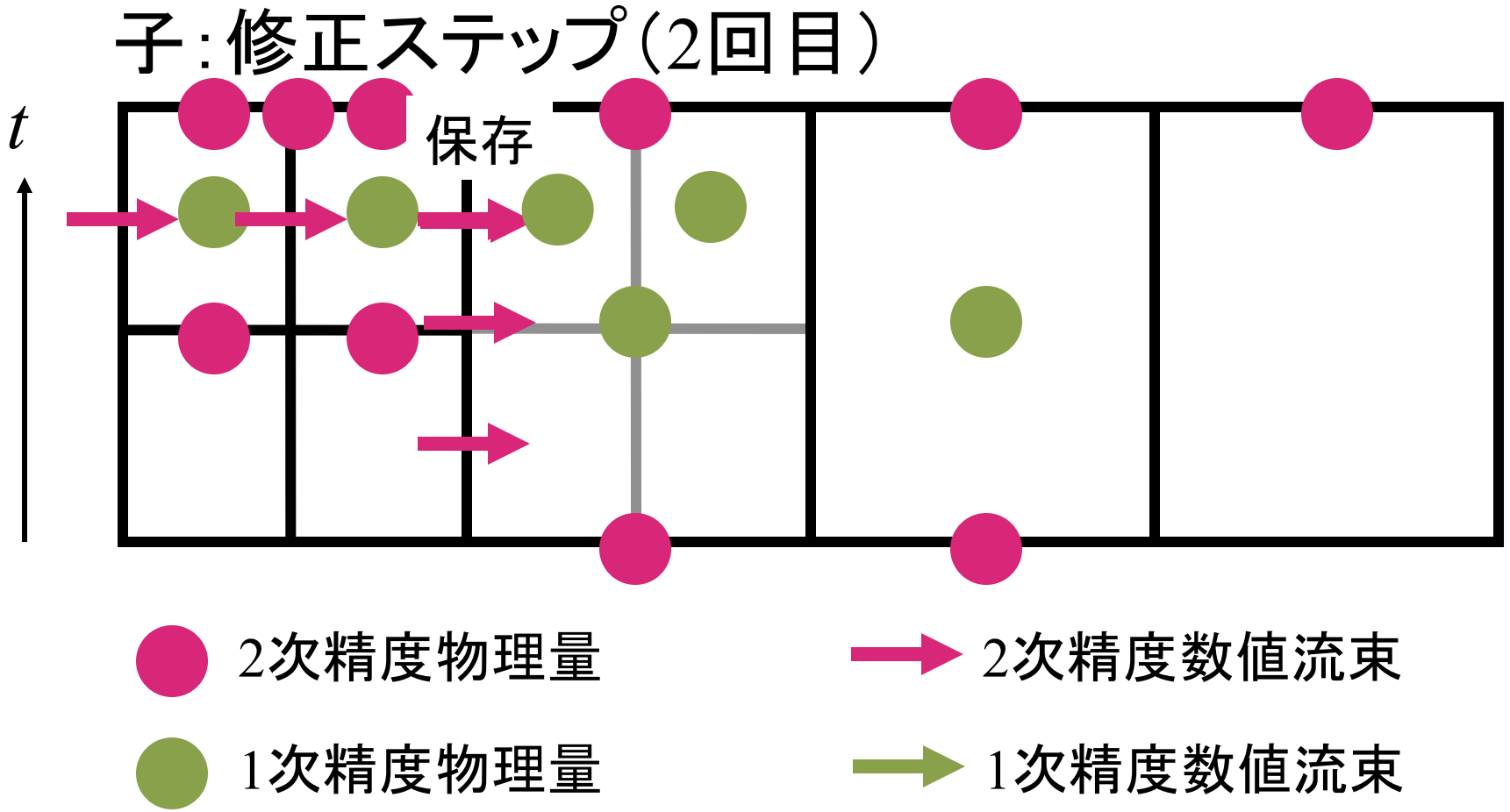


コードの詳細： 流体部分
数値流束補正・マルチタイムステップ



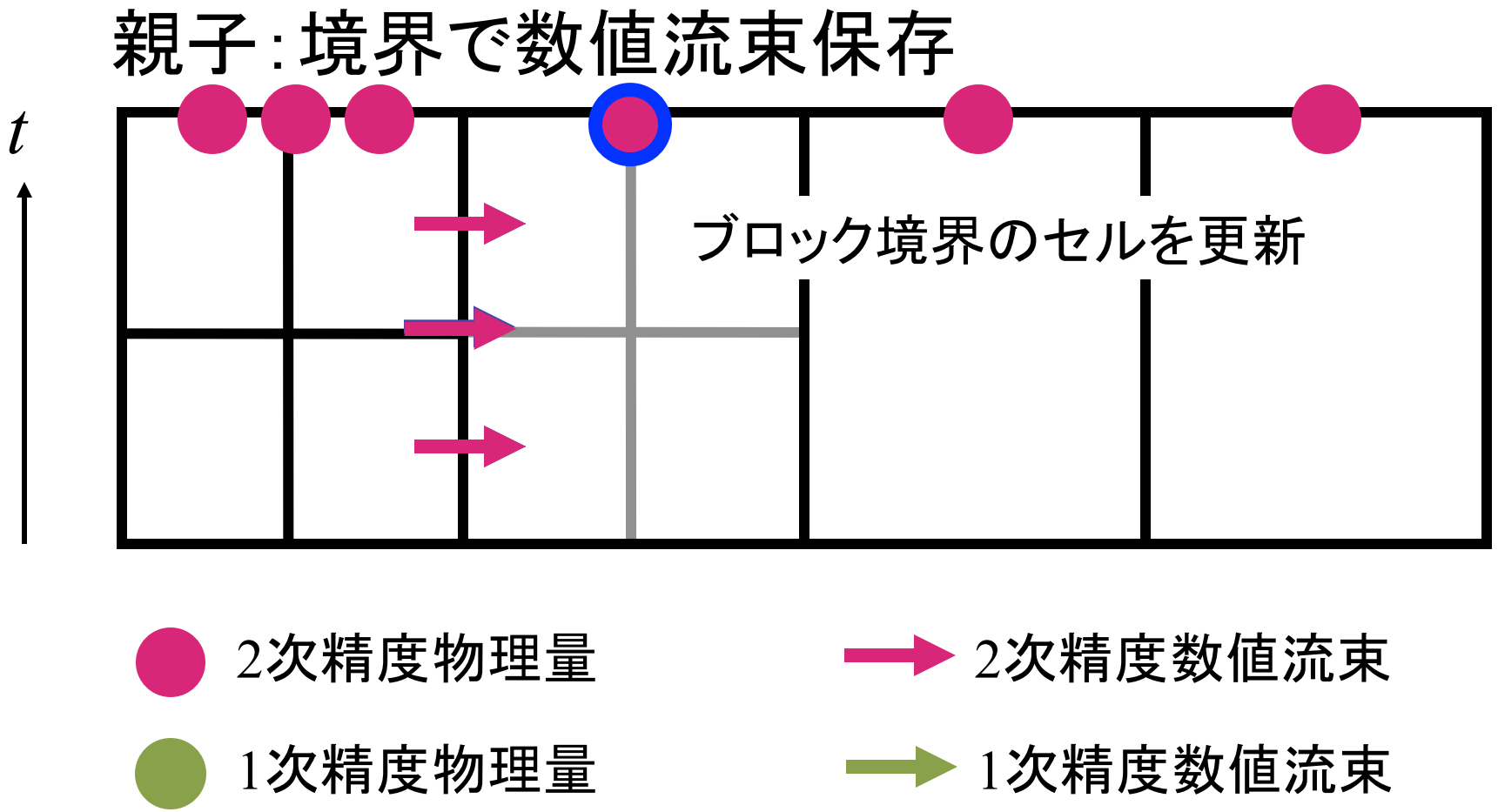
click

コードの詳細： 流体部分
数値流束補正・マルチタイムステップ

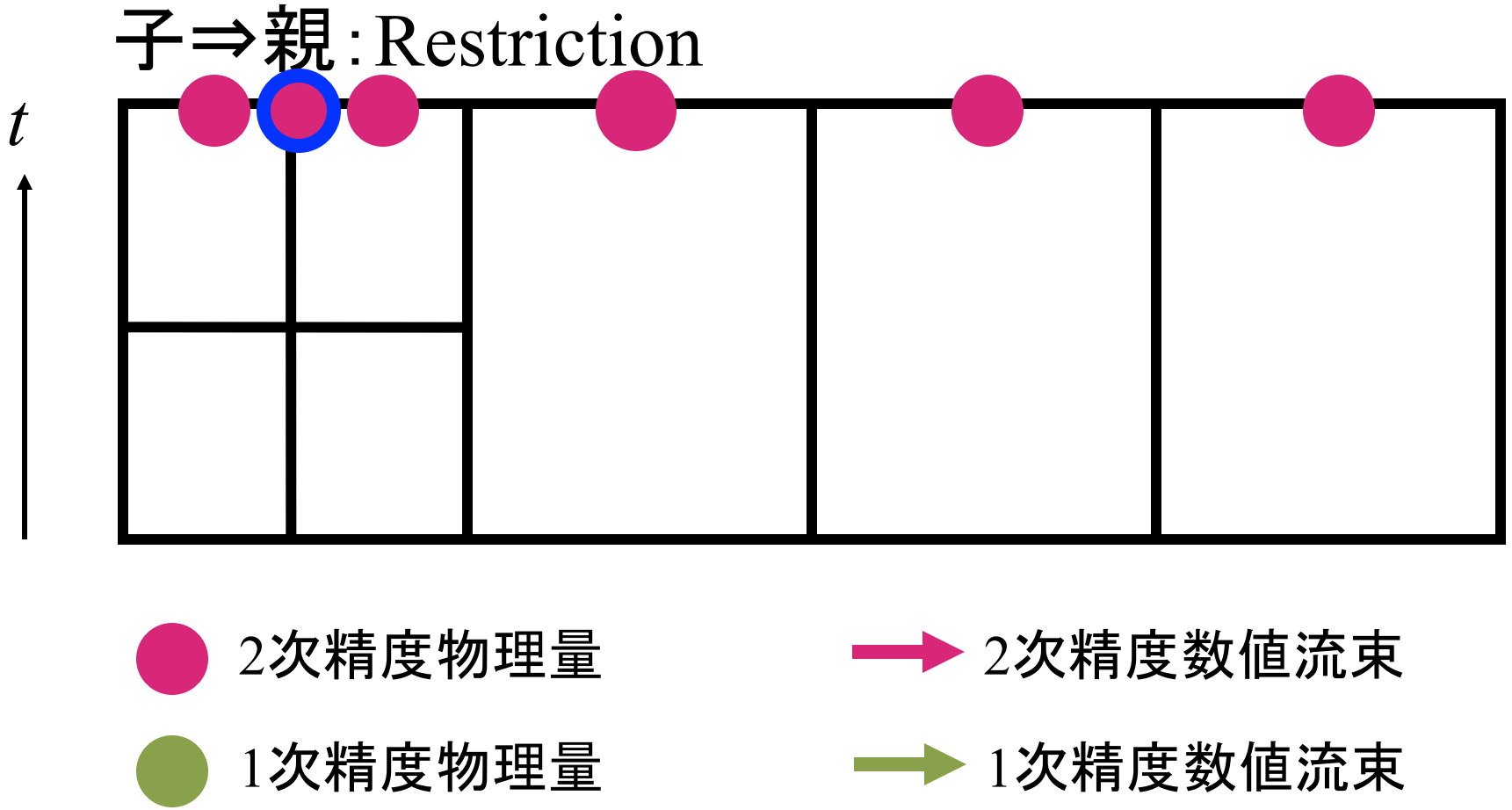


click

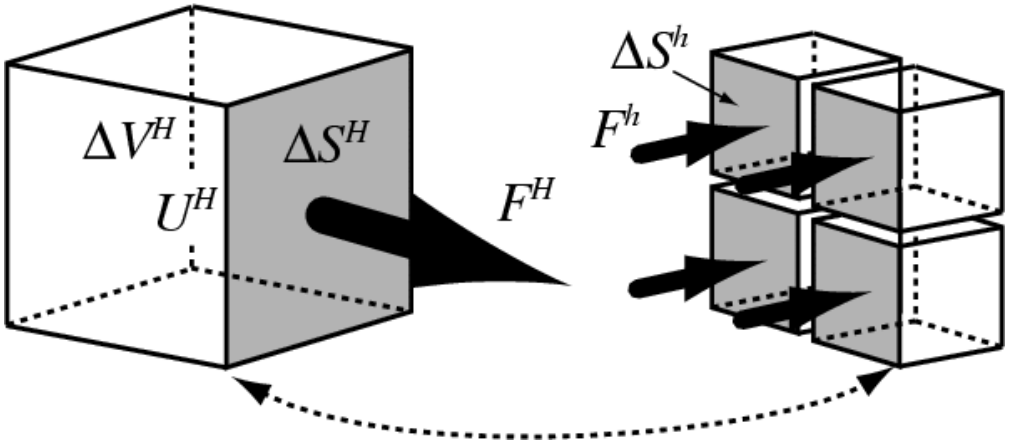
コードの詳細： 流体部分
数値流束補正・マルチタイムステップ



コードの詳細： 流体部分
数値流束補正・マルチタイムステップ



細粗境界での数値流束の保存



数値流束保存 $F^H \Delta S^H \Delta t^H = \sum_{\text{sub-cycle}} \sum_{\text{surface}} F^h \Delta S^h \Delta t^h$ が成立するように、

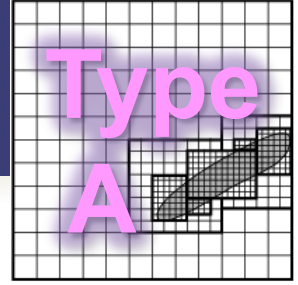
細かいブロックと隣接した粗いセルを補正する

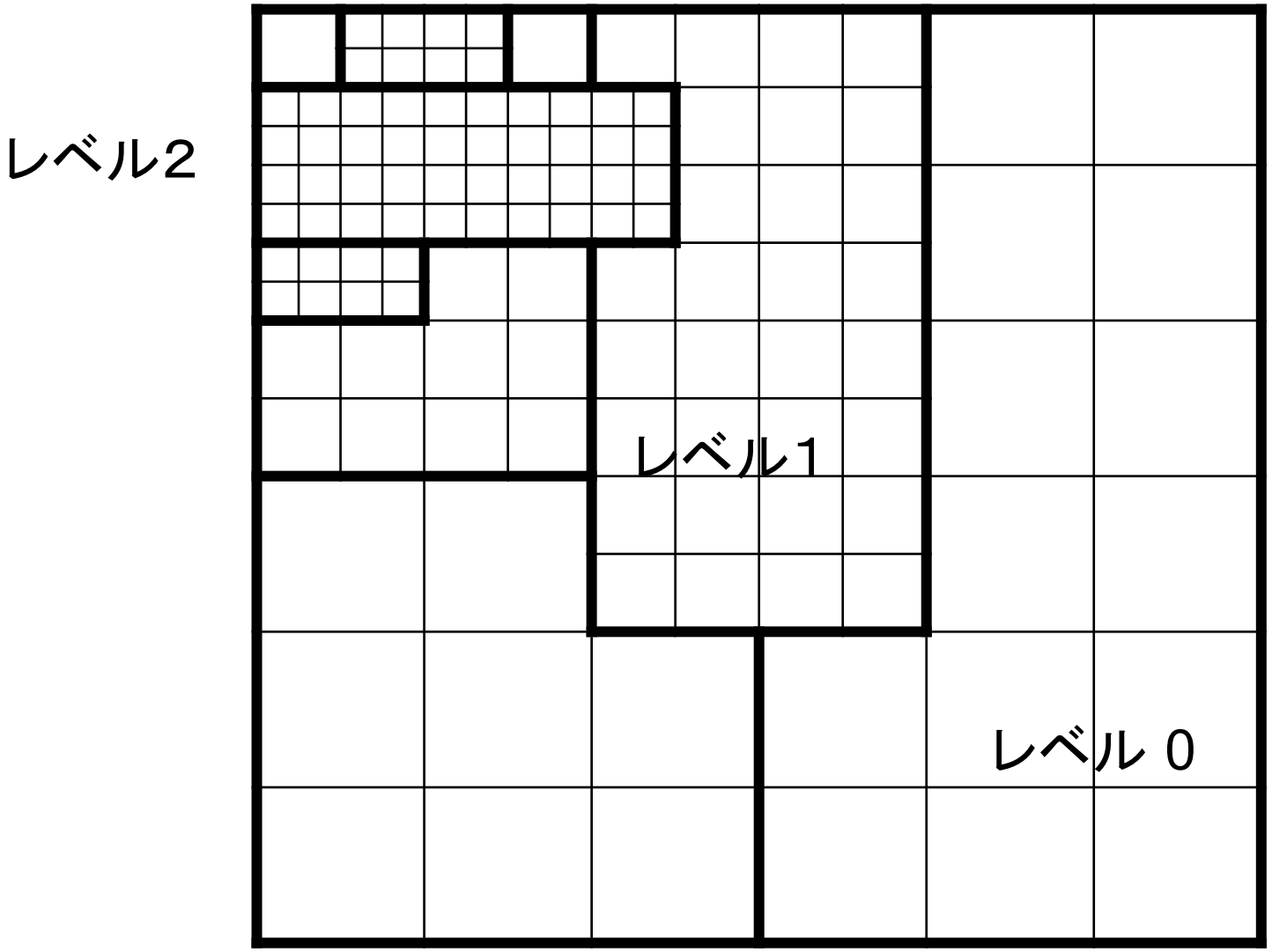
$$U^{H,\text{new}} = U^H - \frac{1}{\Delta V^H} \left(\sum_{\text{sub-cycle}} \sum_{\text{surface}} F^h \Delta S^h \Delta t^h - F^H \Delta S^H \Delta t^H \right)$$

細分化のアルゴリズム

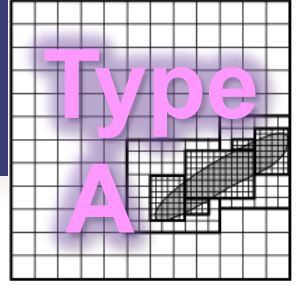
1. 細分化するセルの探査
   a. レベル l に属するセルを順に探査し、細分化条件を満たすセルに印をつける。
   b. 印がついたセルの周囲のセルにも印をつける。
   c. レベル $l+2$ にブロックと重なるセルにも印をつける。
2. ブロック配置の決定
   a. 印がついたセルを含むようにレベル $l+1$ のブロックを作る
3. ブロックの生成
   a. 当該セルがすでに細分化されていれば、その値をコピーする。
   b. 新たに細分化する場合には、レベル l の値を内挿する。
4. 古いブロックの破棄
5. 手順1～4を同期しているレベルで行う。
   a. 粗いレベルから細かいレベルの順に繰り返す。

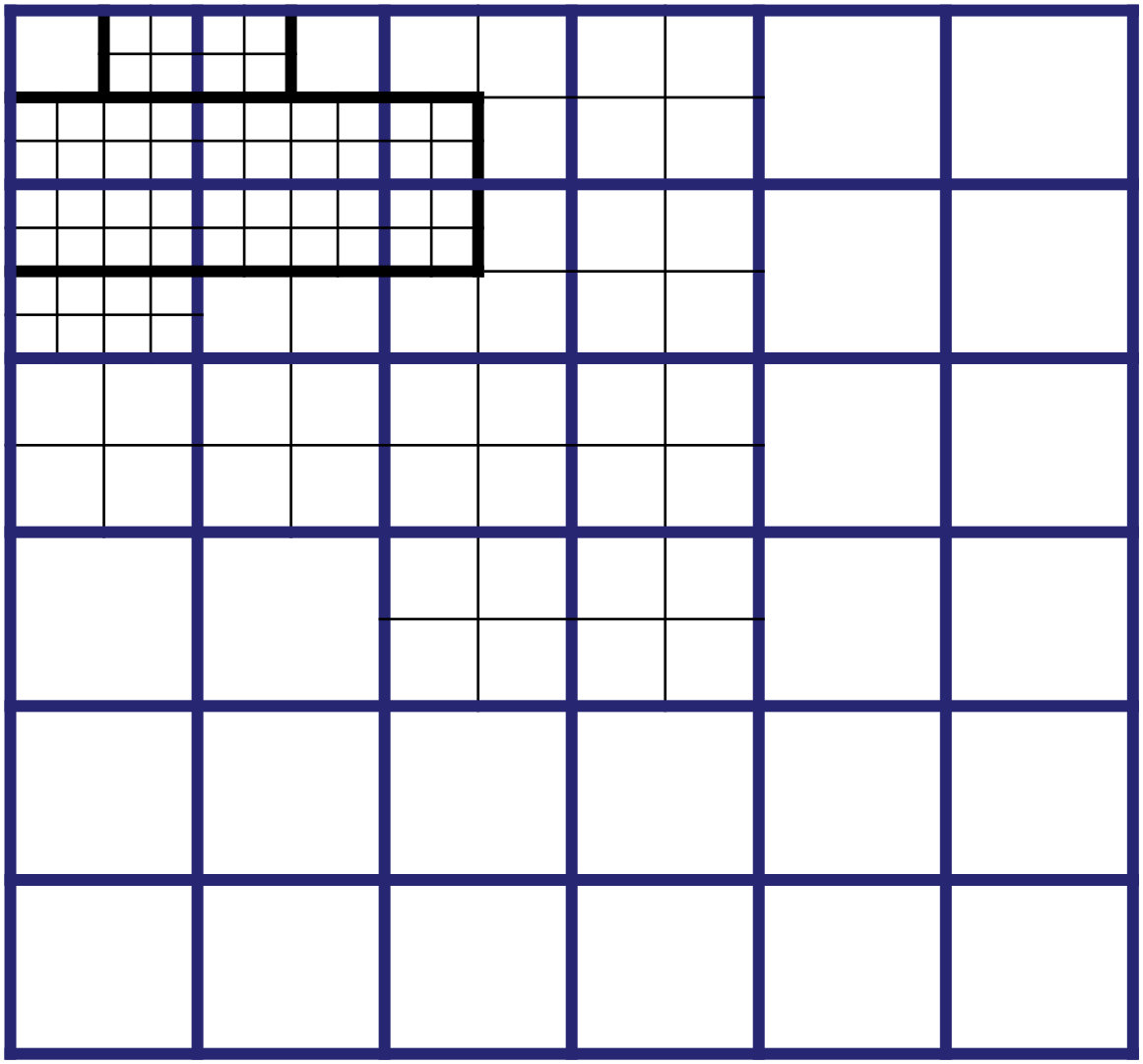
細分化のアルゴリズム
レベル0を細分化してレベル1を生成



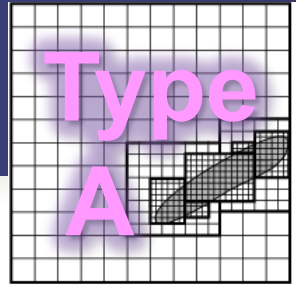


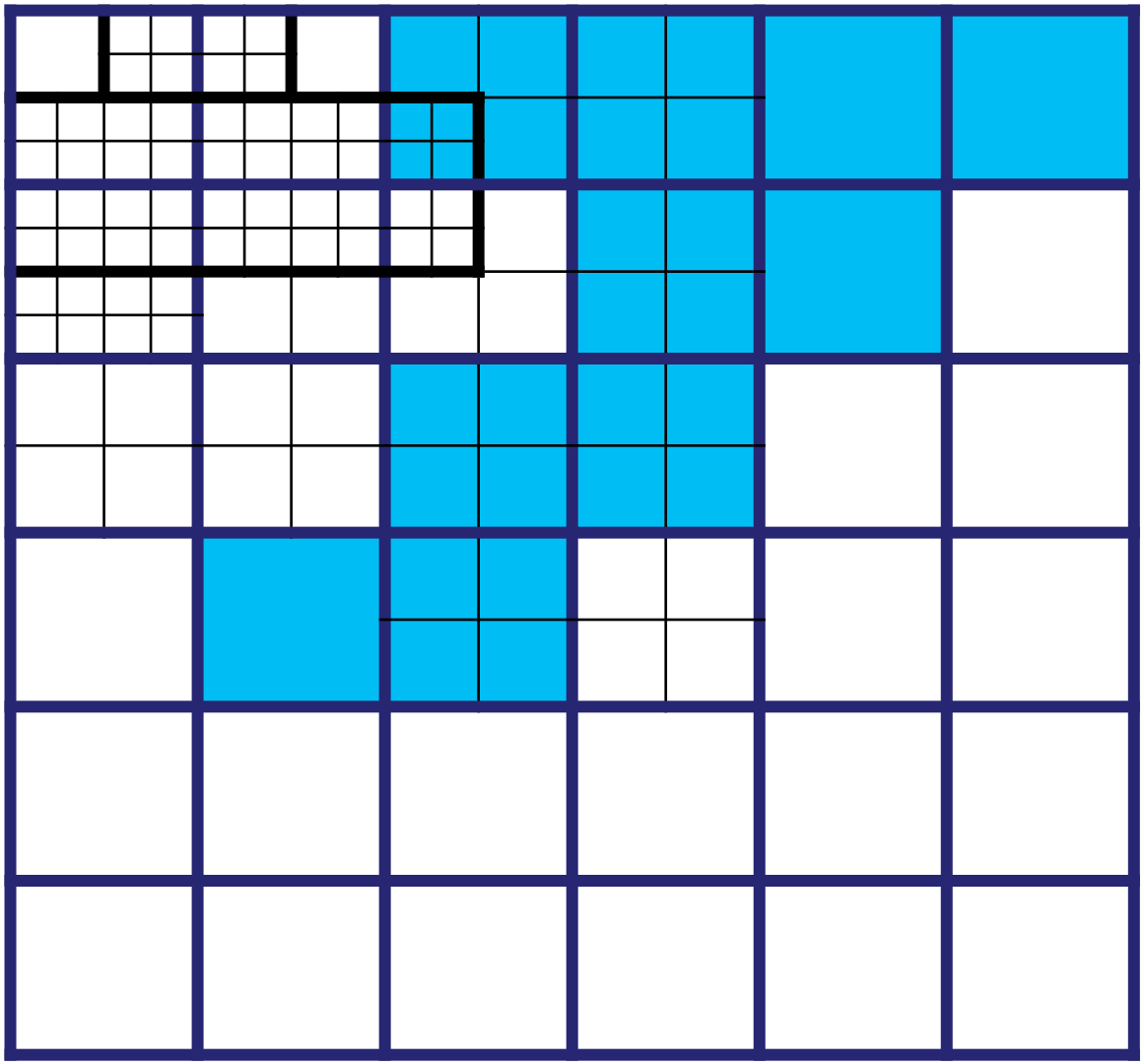
1. レベル0のセルに注目する。



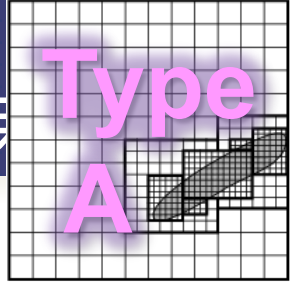


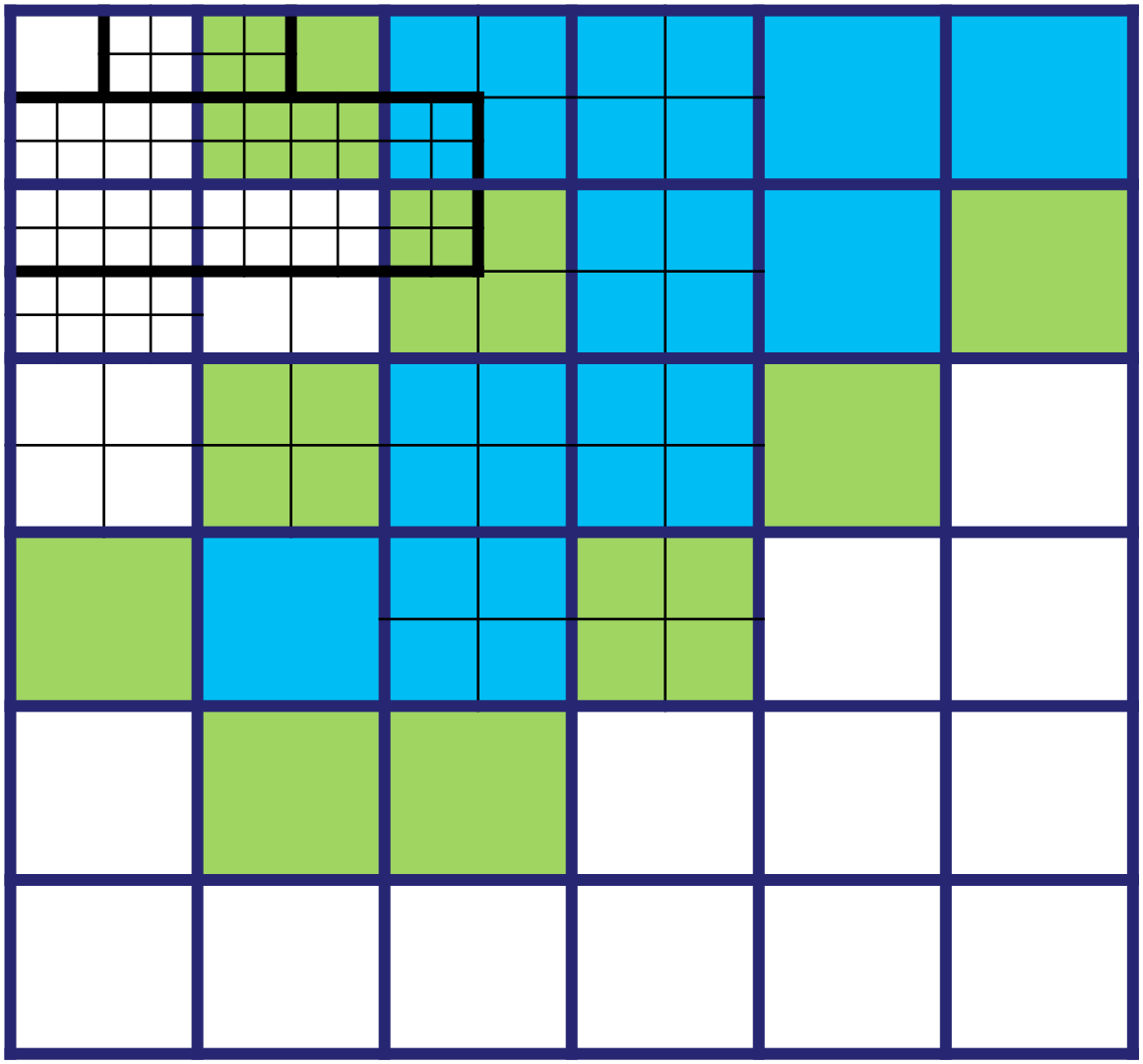
1-a. レベル0の細分化。
レベル0のセルに対して細分化条件を評価する。



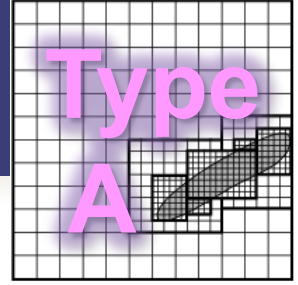


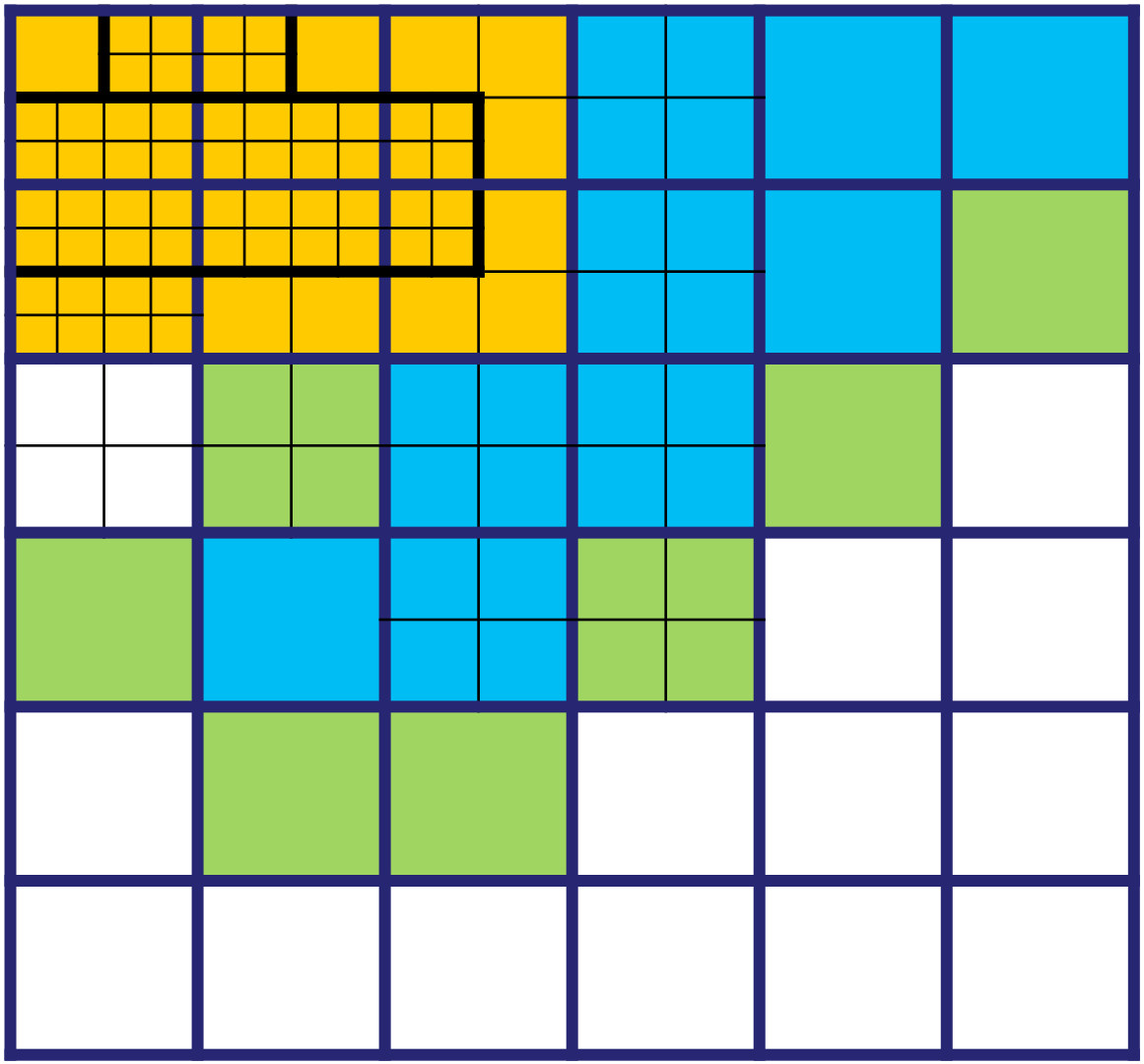
1-b. レベル0の細分化。
周囲の1～2セルにも印をつける。マージンの確保



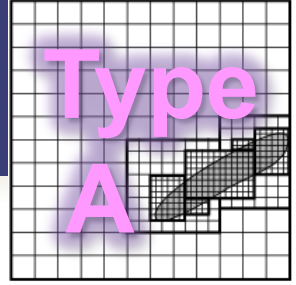


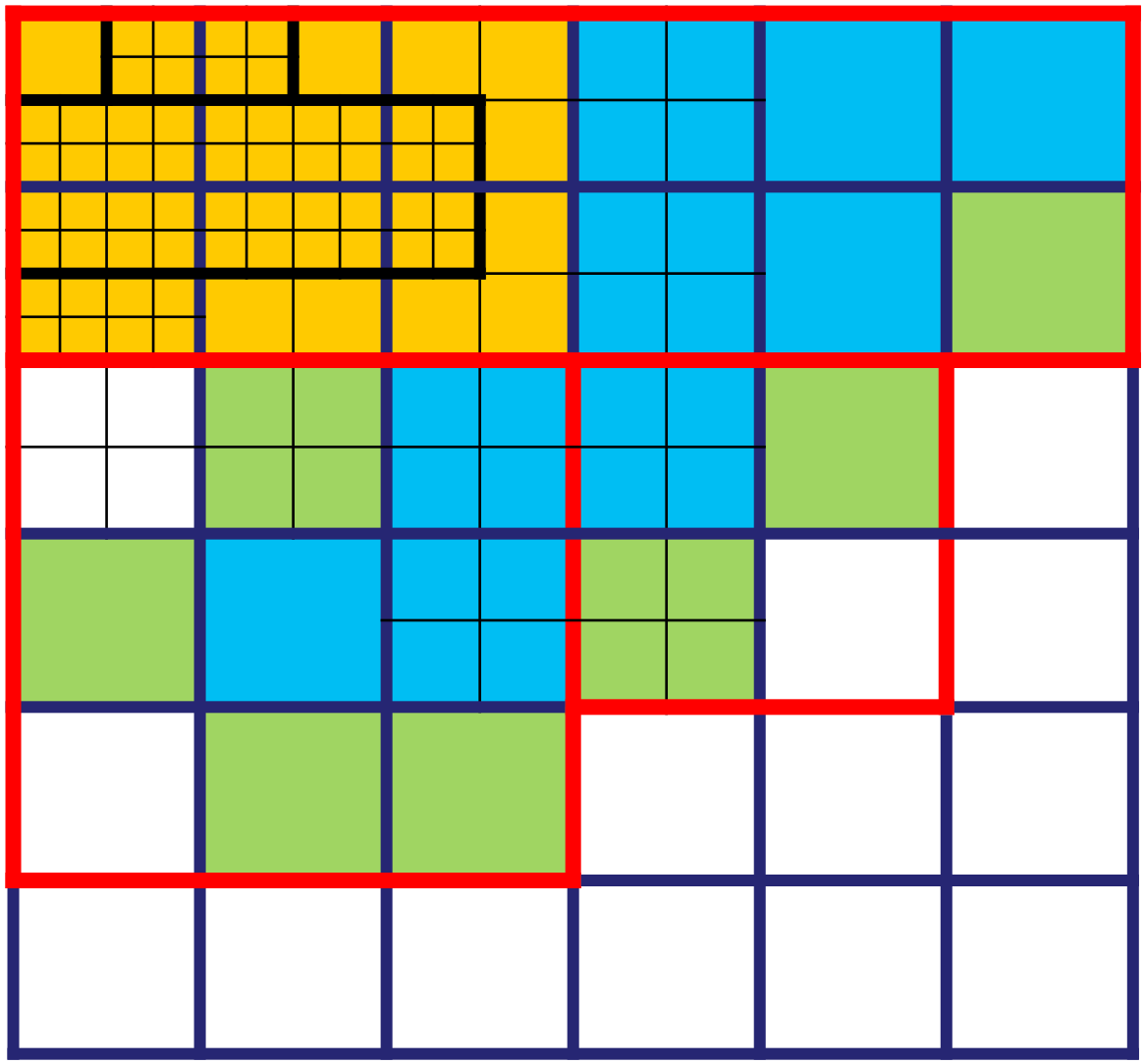
1-c. レベル0の細分化。
レベル2と重なるセルにも印をつける。



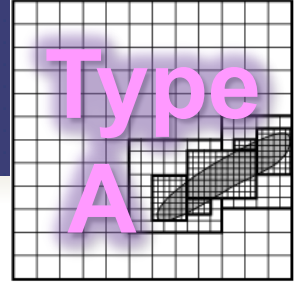


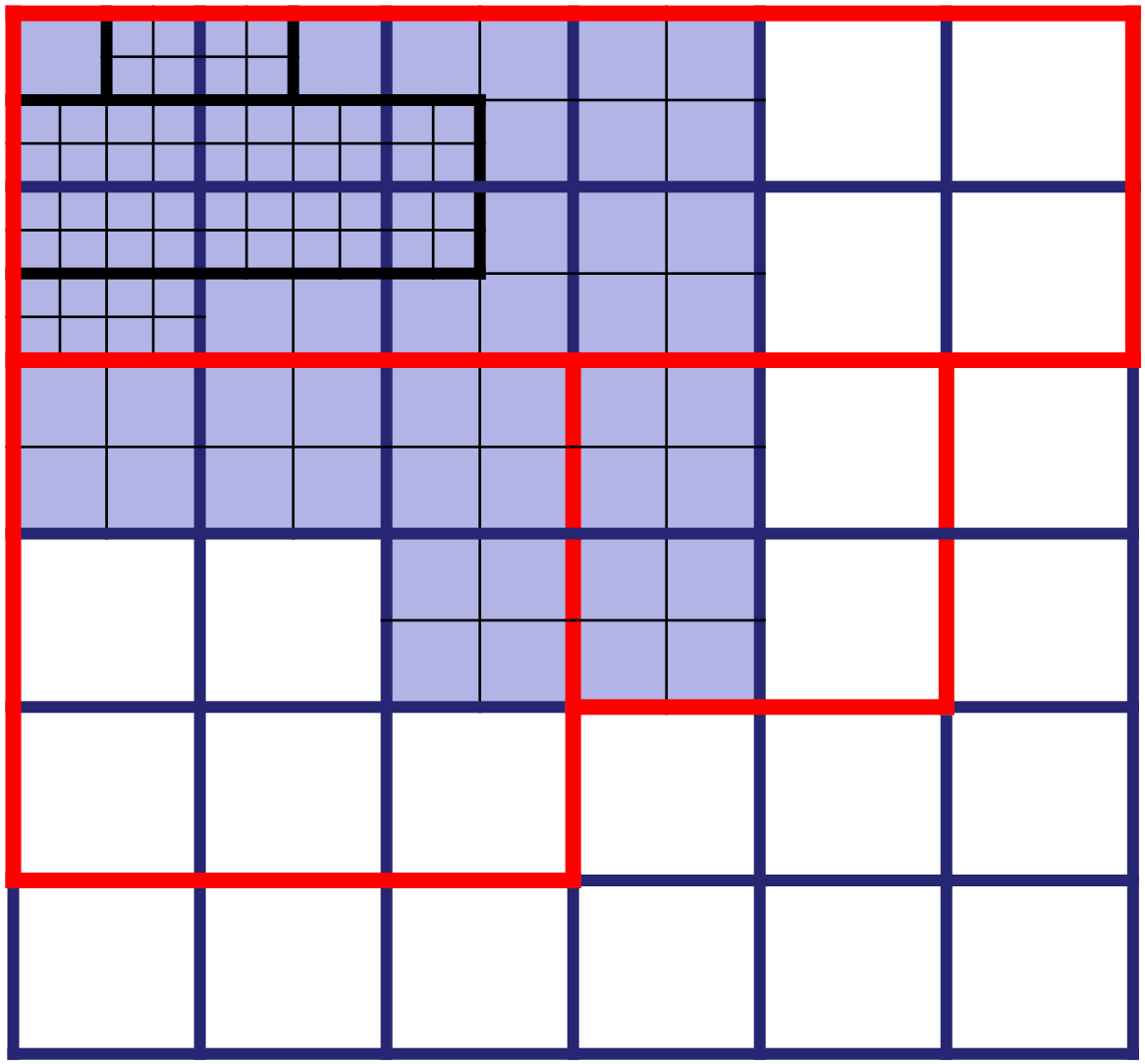
2. レベル1のブロック配置を決定。
ここはいろいろな流儀がある。



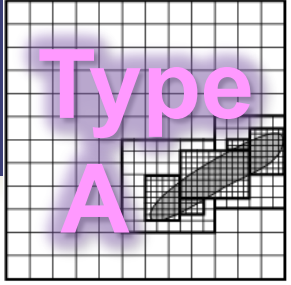


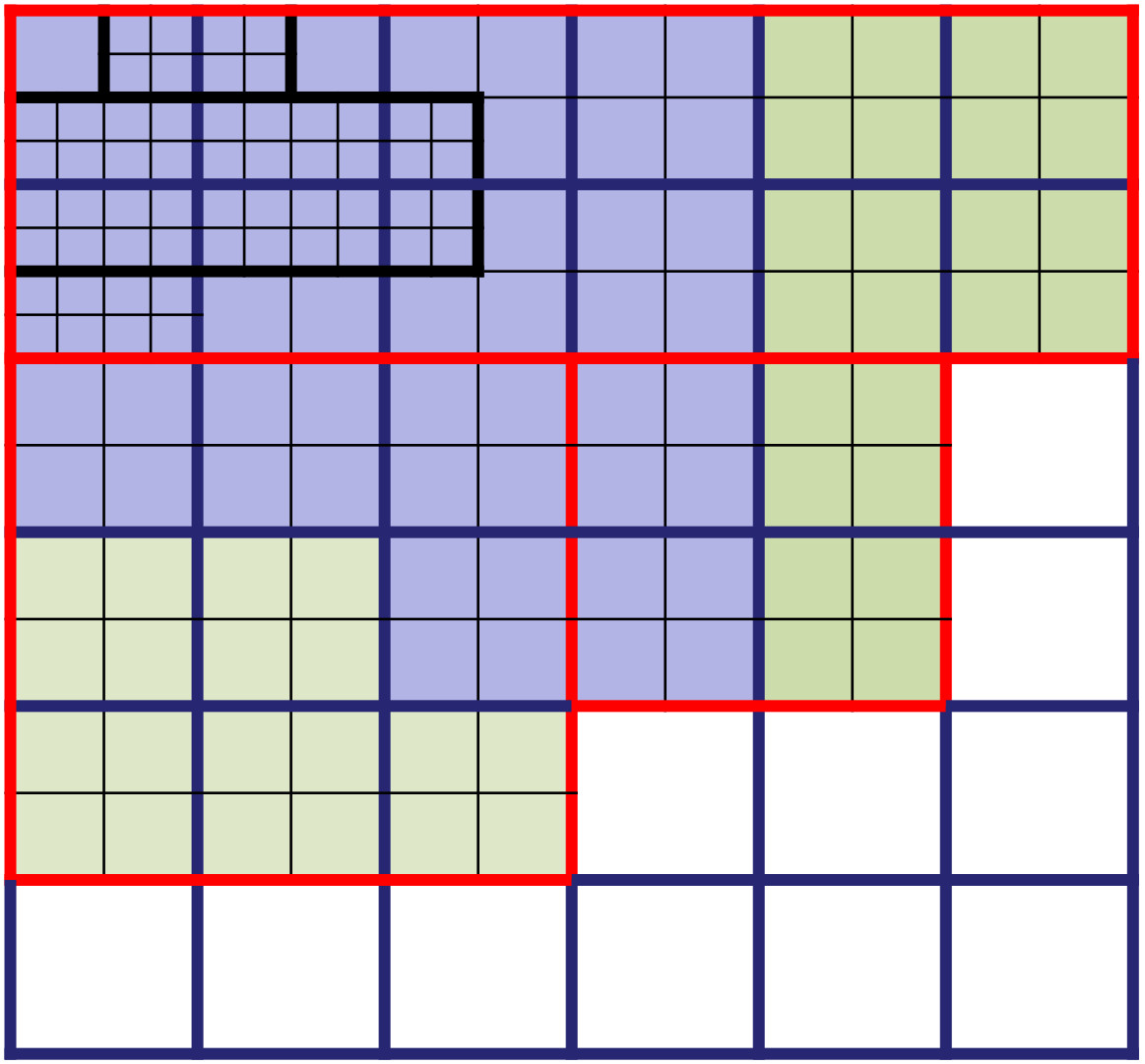
3-a. ブロックの生成。
既存のセルの値をコピー。



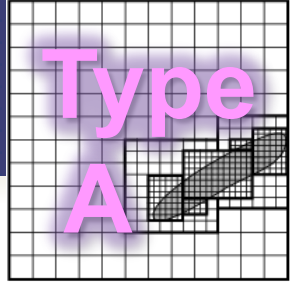


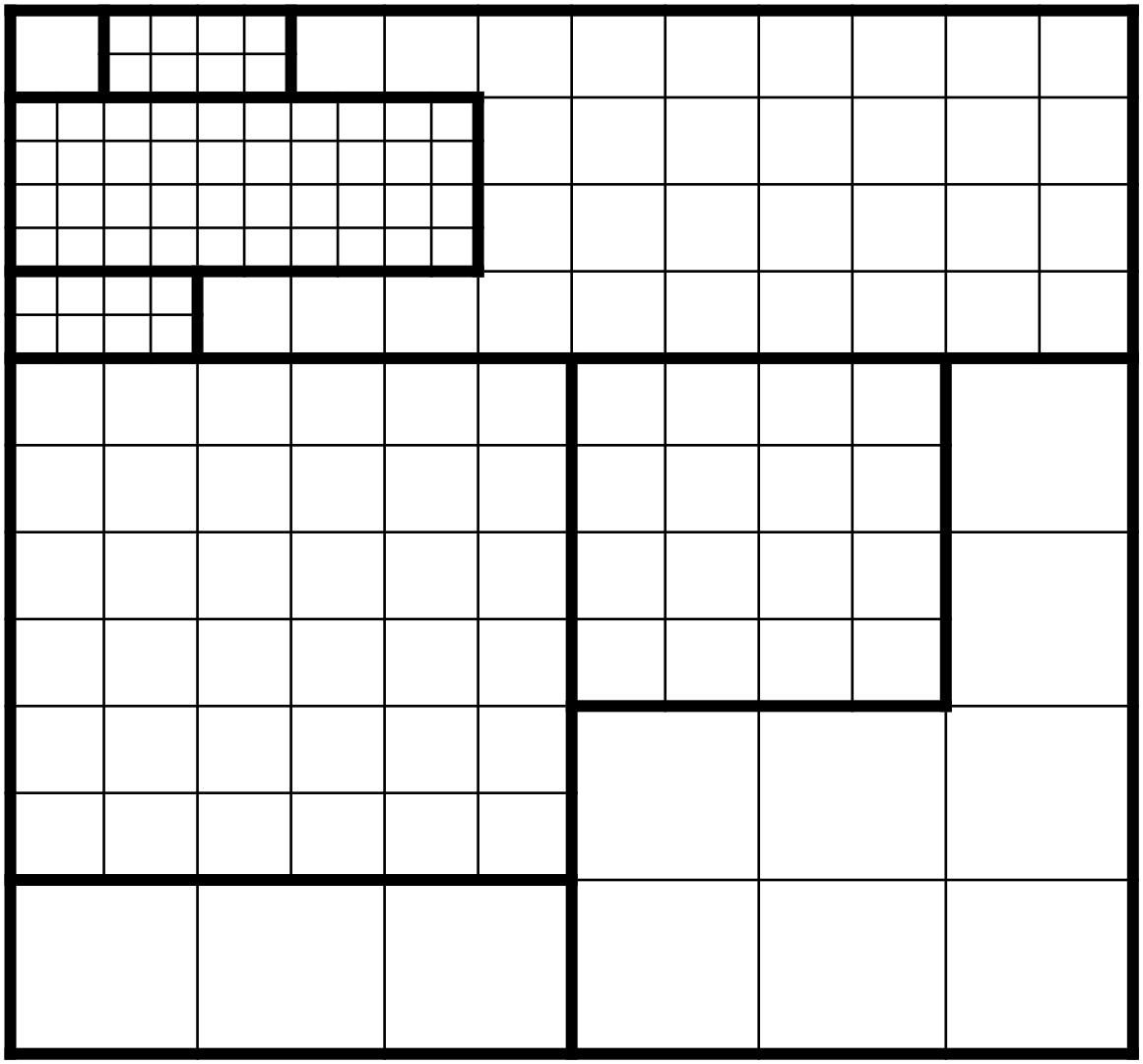
3-b. ブロックの生成。
レベル0から内挿。





4. 古いブロックを破棄して完成。





まとめ

- AMRは宇宙物理学で有効な計算手法
- AMRは「あたりまえ」の時代に
- コードを自作するか、借用するかの選択
- SFUMATOコード
  - 双曲型： HD・MHDの陽解法
  - 楕円型： 自己重力
  - 放物型： 拡散の陰解法

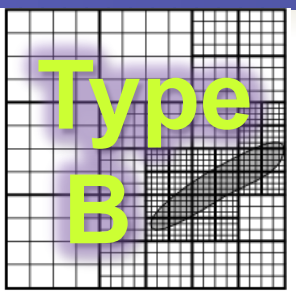
おまけ

並列化の方法

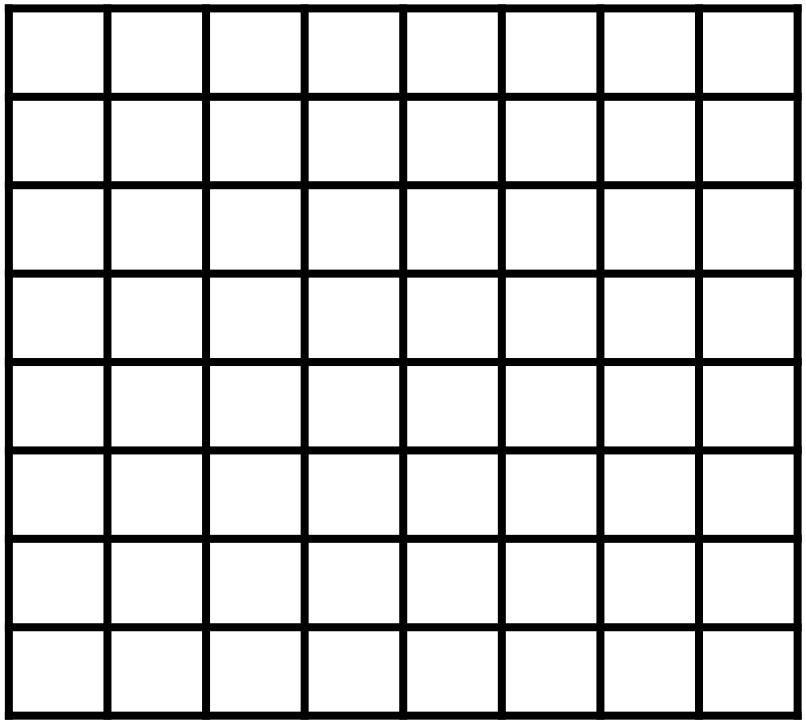
- Block structured gridの場合
  - ブロックをノードに割りつける。
  - 通信量を少なくするために
    - 近所のブロックを同じノードに割り付ける。
    - 近所のブロックを近所のノードに割りつける。

- それ以外の場合
  - よく知りません。

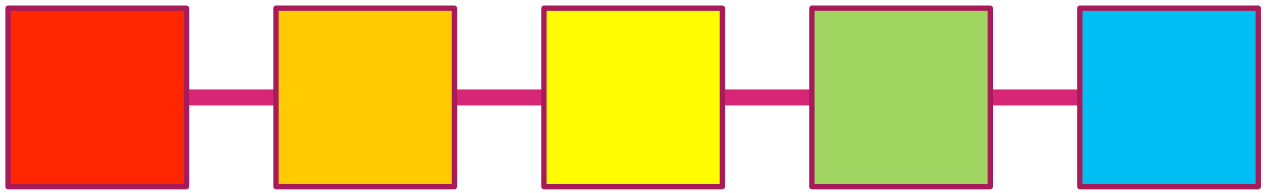
並列化： 良くない方法



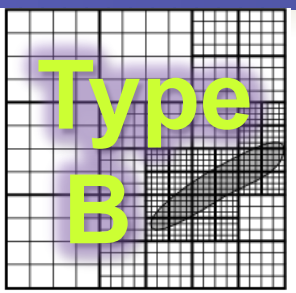
8^2ブロック
= 64 ブロック
=13 ブロック/ノード×5ノード
-1ブロック



ノード



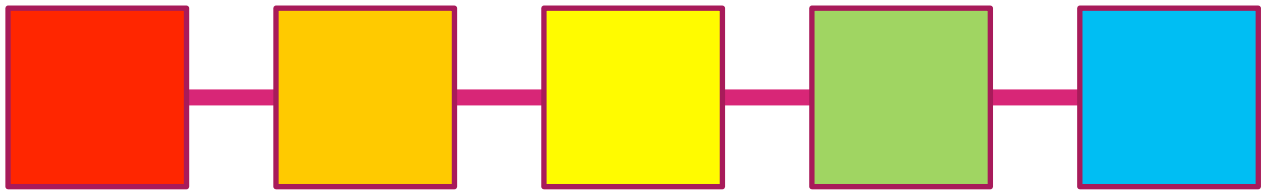
並列化： 良くない方法



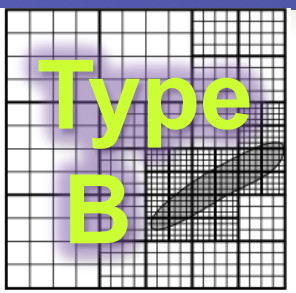
8^2ブロック
= 64 ブロック
=13 ブロック/ノード×5ノード
-1ブロック

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

ノード



並列化： 良くない方法



8^2ブロック
= 64 ブロック
=13 ブロック/ノード×5ノード
-1ブロック

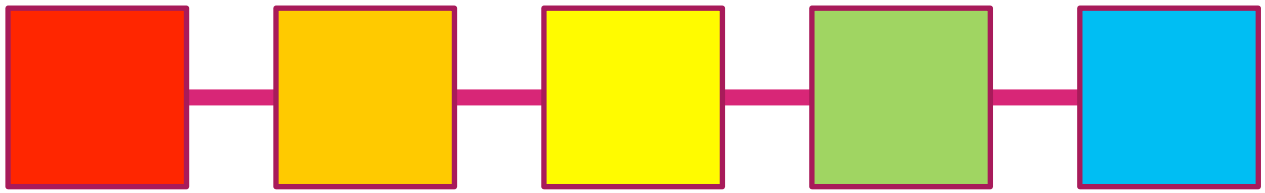
同じノードに割り付けられるブロックが細長く分布。

通信量は多い

ブロック数・ノード数が多くなると、不利になる。
泣き別れのブロックなど。

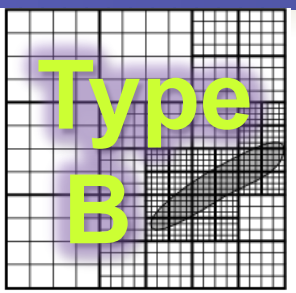
1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

ノード

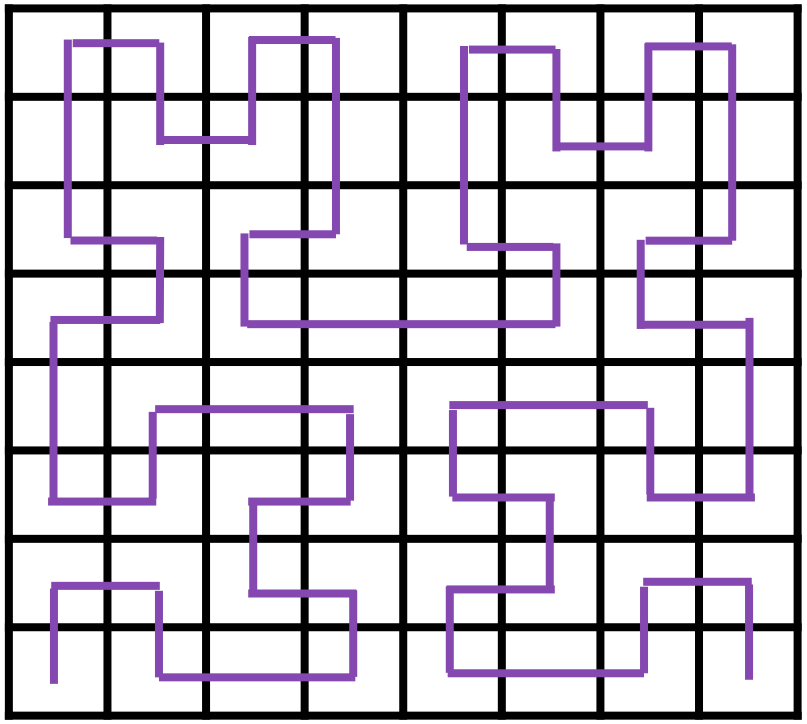


並列化： 良い方法

8^2ブロック
= 64 ブロック
=13 ブロック/ノード×5ノード
-1ブロック

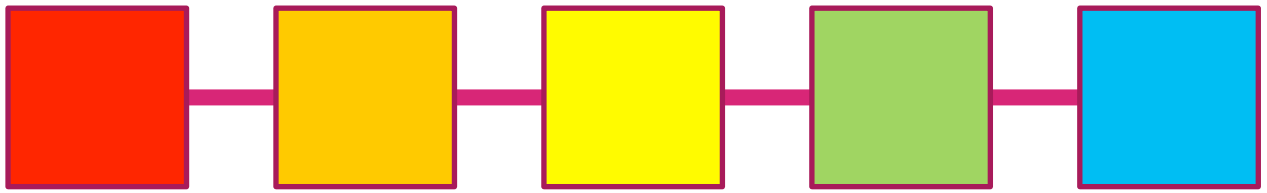


Peano-Hillbert
空間充填曲線
によるオーダリング

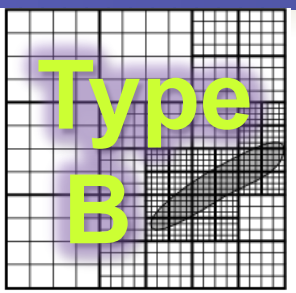


なるべく近くを通る
一筆書き

ノード



並列化： 良い方法



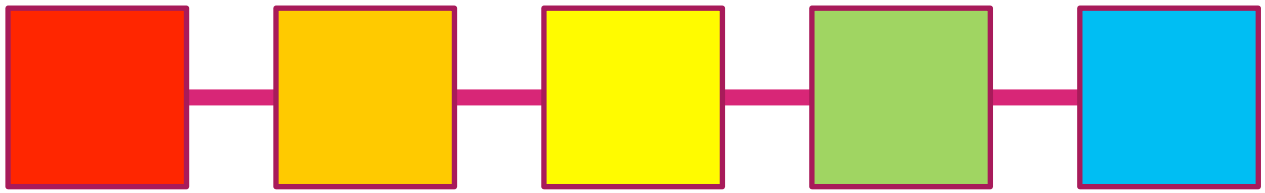
8^2ブロック
= 64 ブロック
=13 ブロック/ノード×5ノード
-1ブロック

同じノードに割り付けられるブロックがコンパクトに分布。

通信量は少ない。

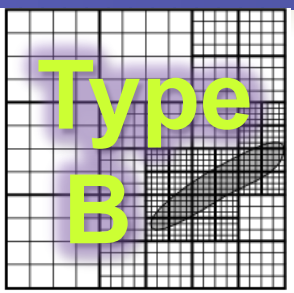
22	23	26	27	38	39	42	43
21	24	25	28	37	40	41	44
20	19	30	29	36	35	46	45
17	18	31	32	33	34	47	48
16	13	12	11	54	53	52	49
15	14	9	10	55	56	51	50
2	3	8	7	58	57	62	63
1	4	5	6	59	60	61	64

ノード

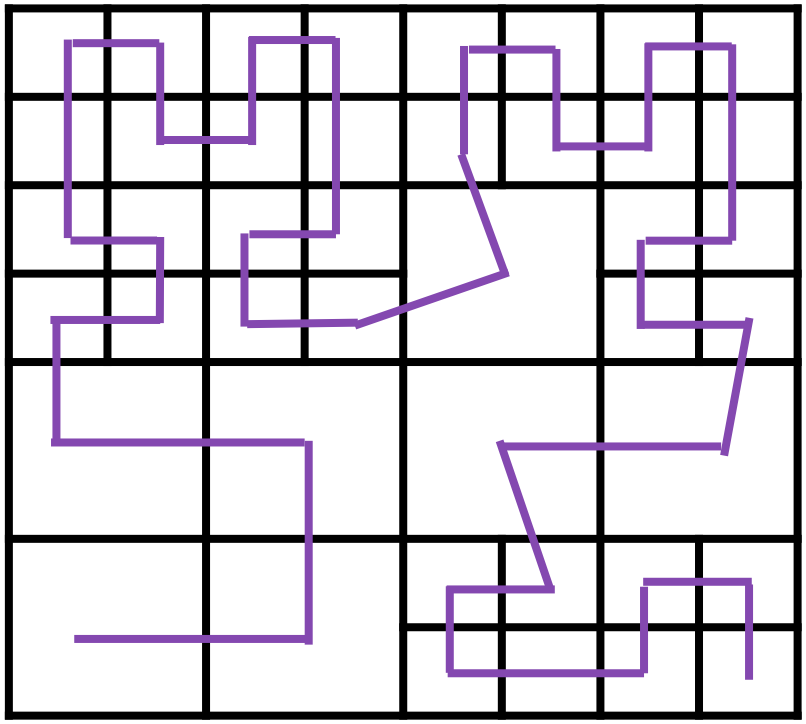


並列化： グリッドレベル横断

= 43 ブロック
= 8 ブロック/ノード×5ノード
+3ブロック

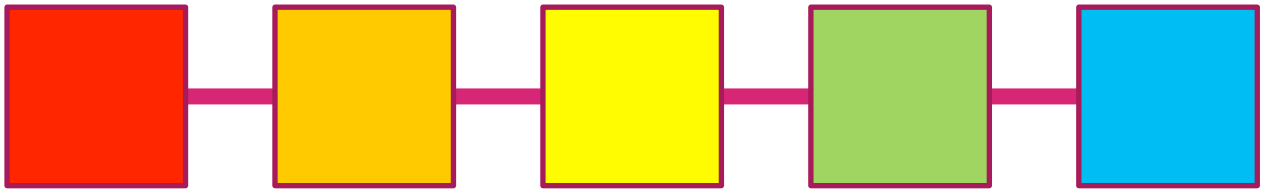


Peano-Hillbert
空間充填曲線
によるオーダリング



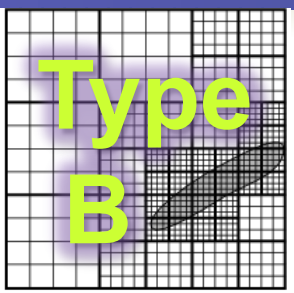
なるべく近くを通る
一筆書き

ノード



並列化： グリッドレベル横断

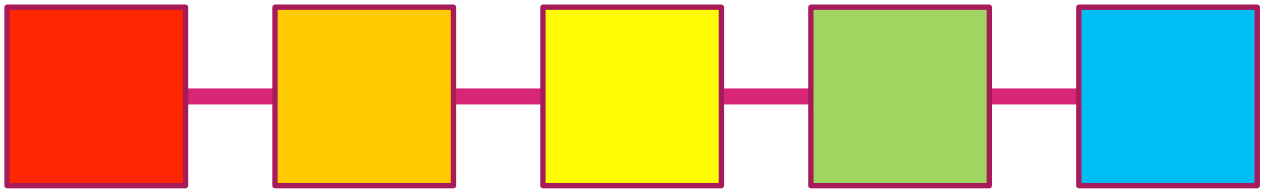
= 43 ブロック
= 8 ブロック/ノード×5ノード
+3ブロック



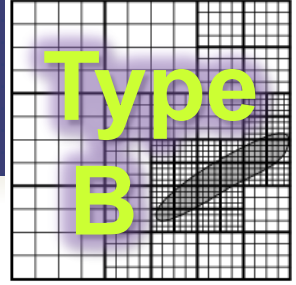
Peano-Hillbert
空間充填曲線
によるオーダリング

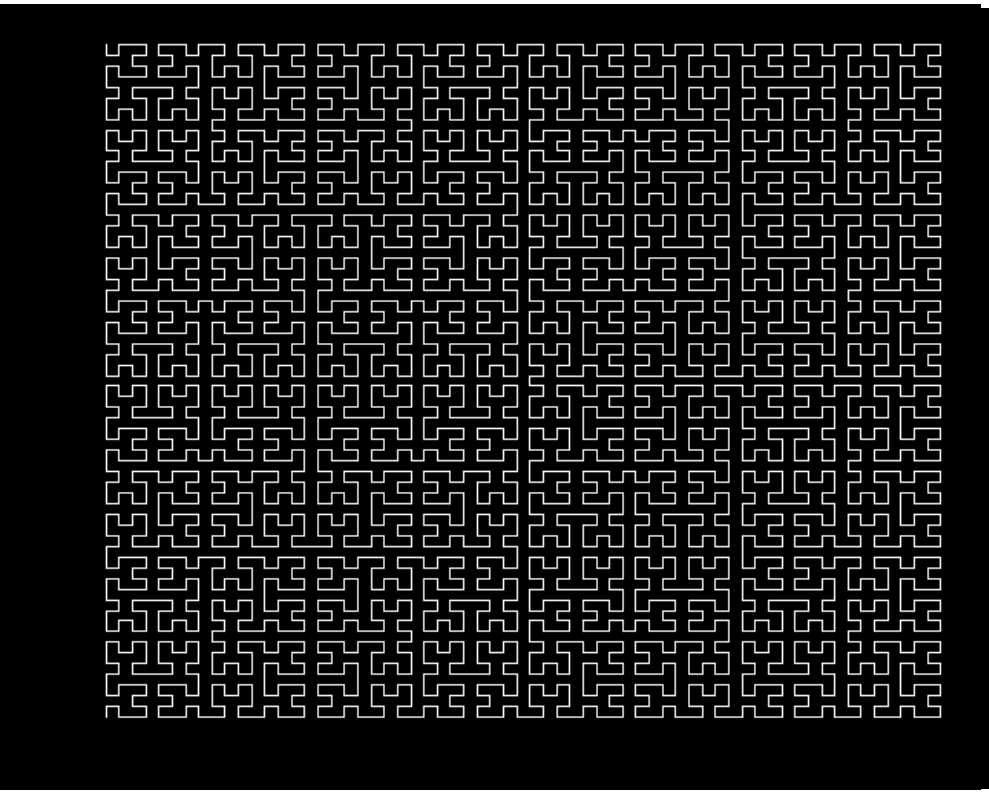
なるべく近くを通る
一筆書き

ノード

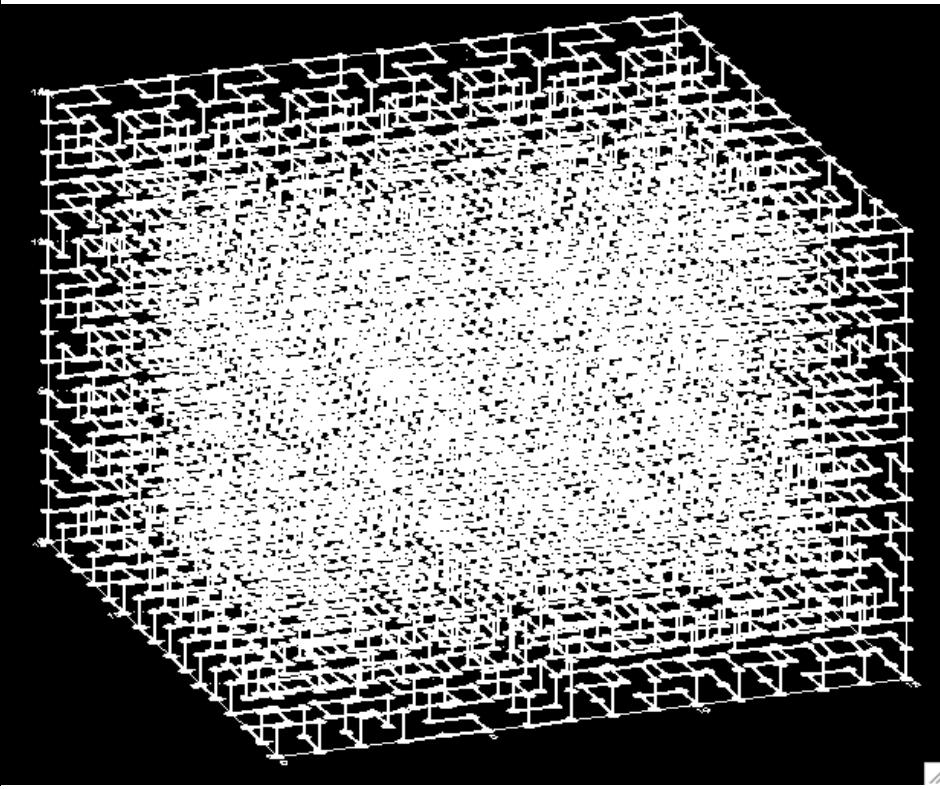


3次元でも同様





2次元のPeano-Hillbert 空間充填曲線

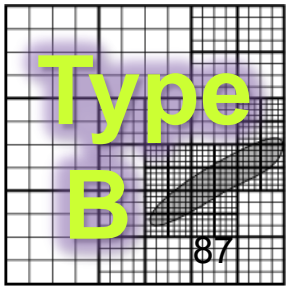


3次元のPeano-Hillbert 空間充填曲線

ブロックのオーダリングまとめ

- Peano-Hillbert空間充填曲線によるオーダリングは、面倒そうですが、意外に簡単な工夫です。
- グリッドレベルごとに行う。
  - Adaptive time step
- グリッドレベルを横断して行う。
  - Synchronous time step

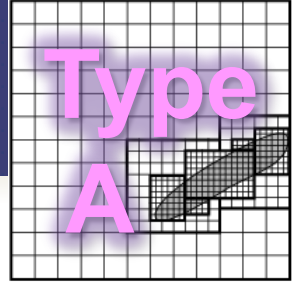
僕は使い分けてないけど。。。



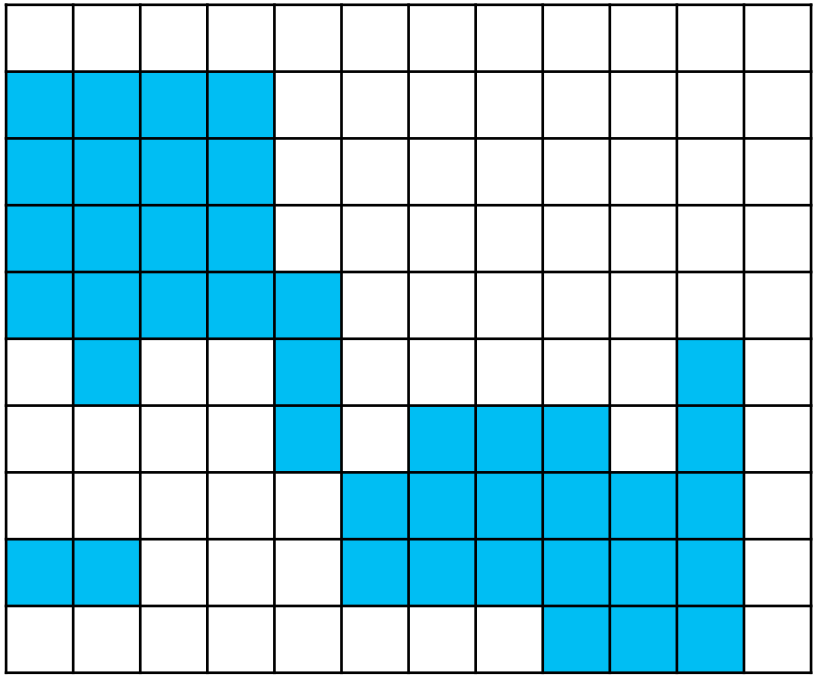
細分化条件: どこを細分化するべきか？

- 何を見たいかに依存する。
- 打切り誤差が閾値を超える部分を細分化 (Berger & Colella 1989)
- 物理量の2階微分が閾値を超える部分を細分化 (Flash など)
- 星形成分野の業界標準
  - Jeans 条件: Jeans 波長を4メッシュ以上で分解する (Truelove et al.)

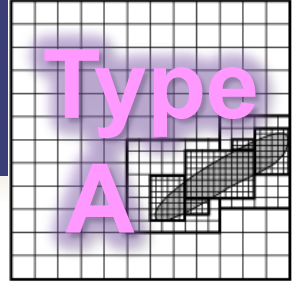
ブロックの決定方法



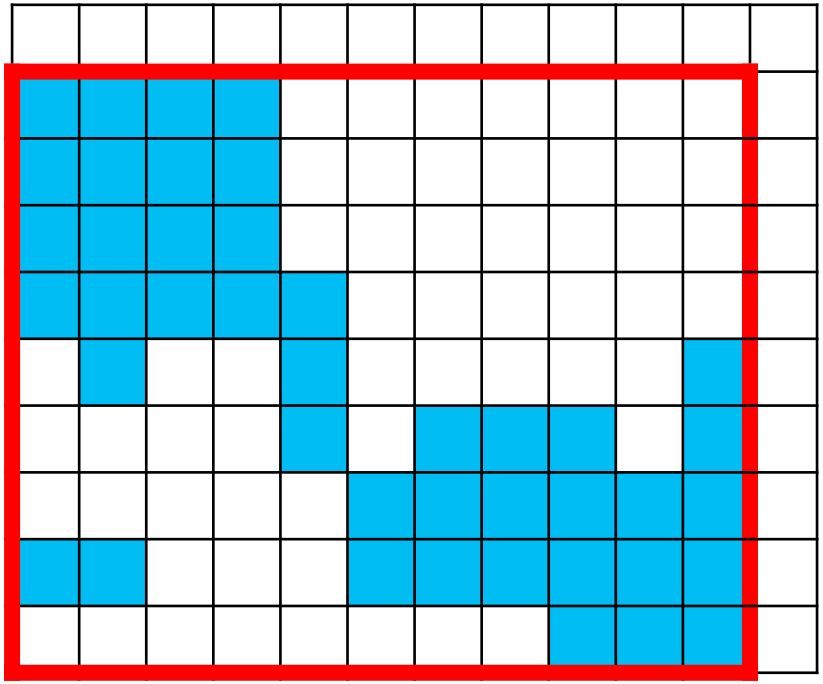
パッチ指向AMRには様々な流儀が存在する。
たとえば、



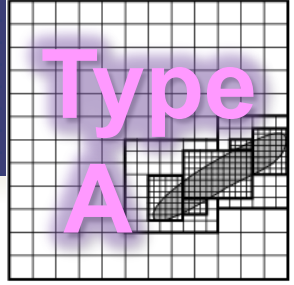
ブロックの決定方法



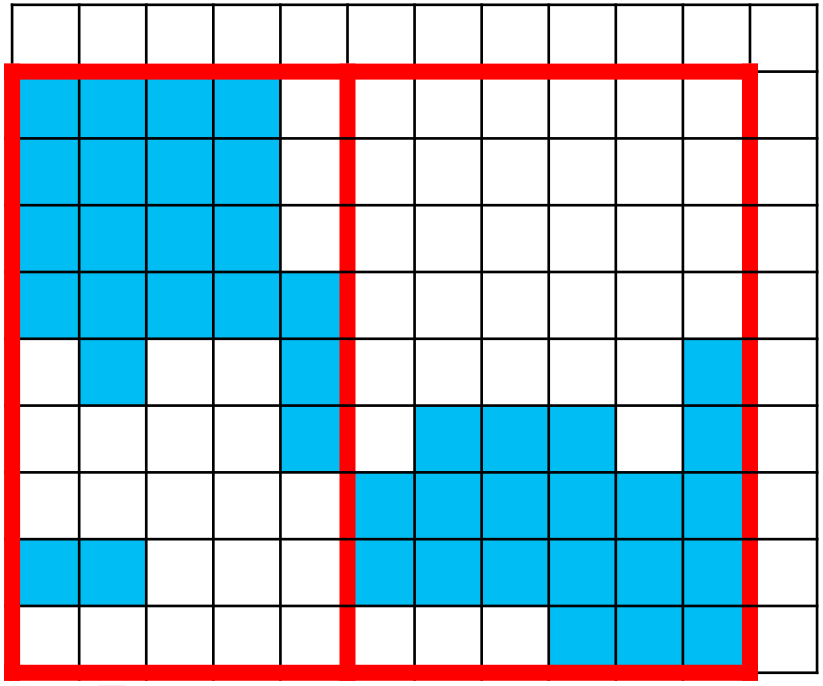
印がついたセルをすべて覆うブロックを考える



ブロックの決定方法



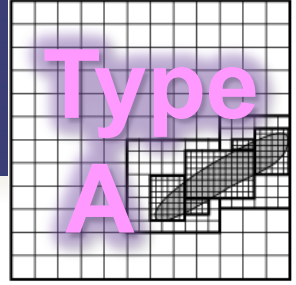
セル数のヒストグラムを書き、最小値部分でブロックを分割する。



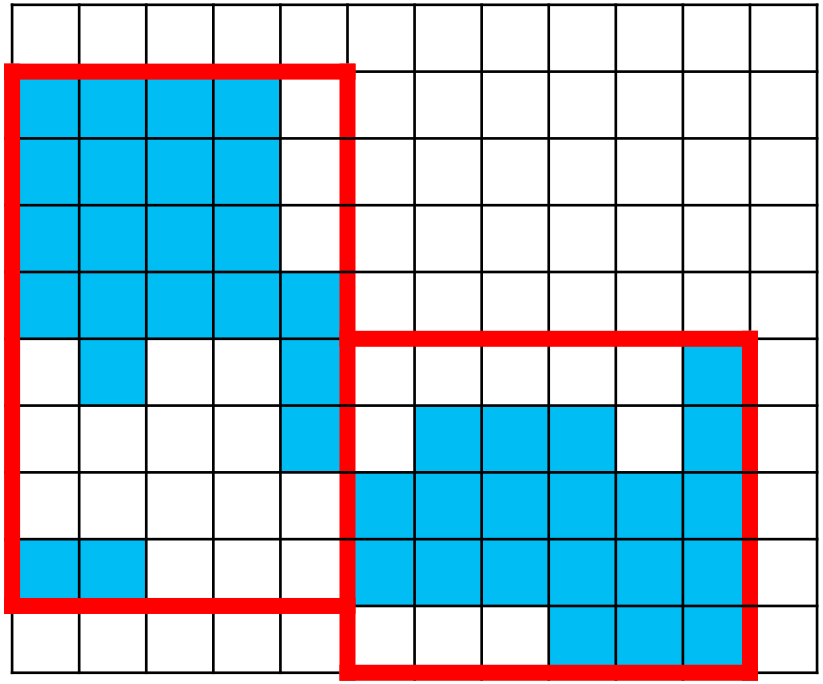
ヒストグラム

最小

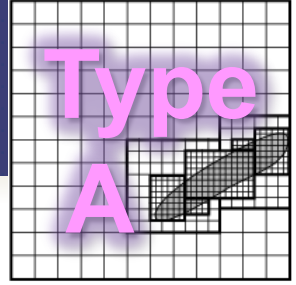
ブロックの決定方法



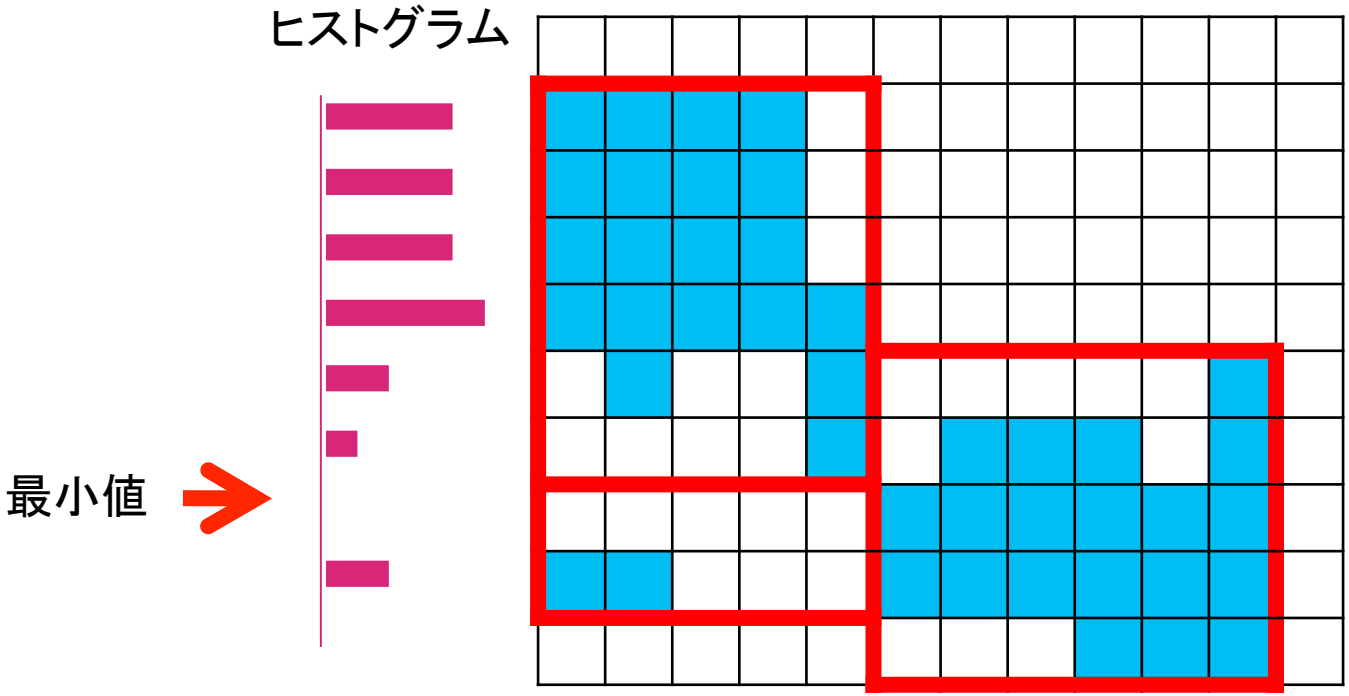
余分な部分を取り除く



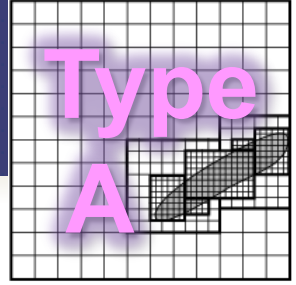
ブロックの決定方法



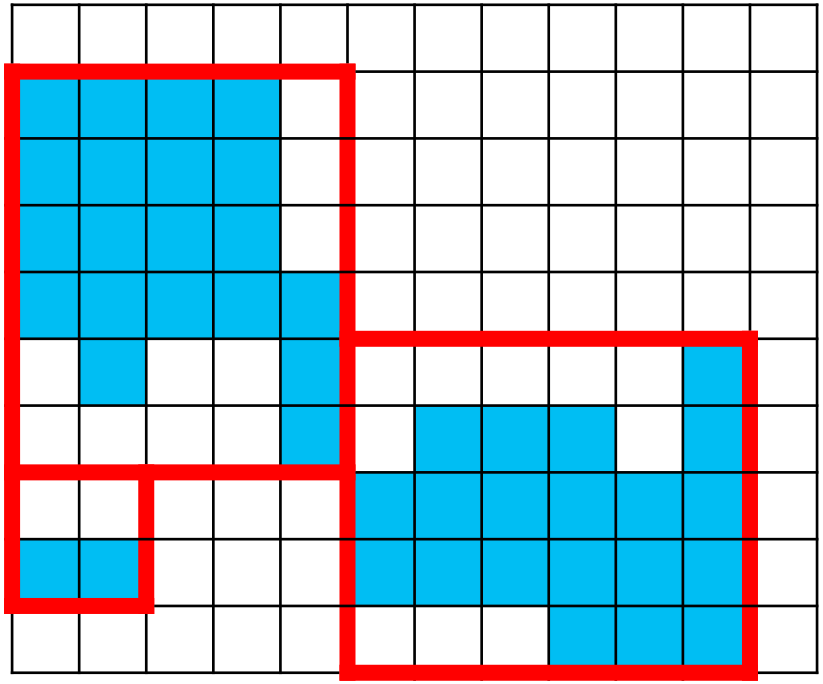
別の方向からセル数のヒストグラムを書き、分割する。



ブロックの決定方法



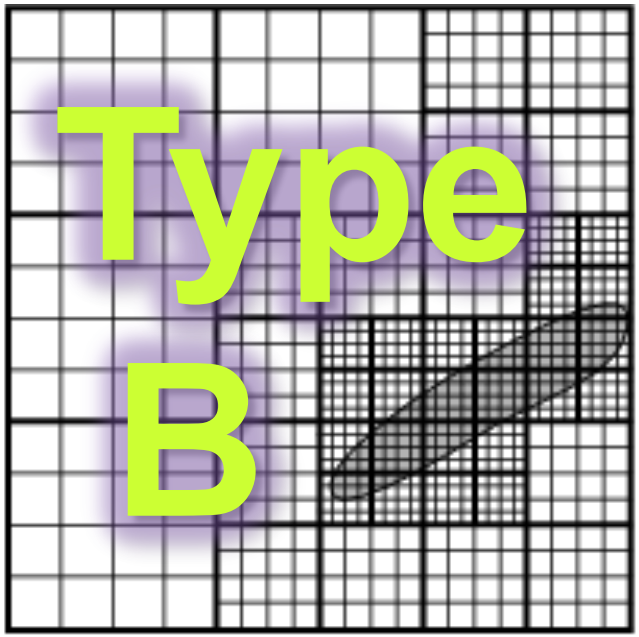
余白を取り除く。



こんなことを繰り返してゆく。

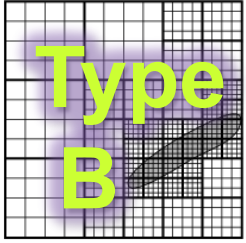
この操作を繰り返すほど、ブロックは小さくなる。
ブロックの最小サイズを決めておこう。

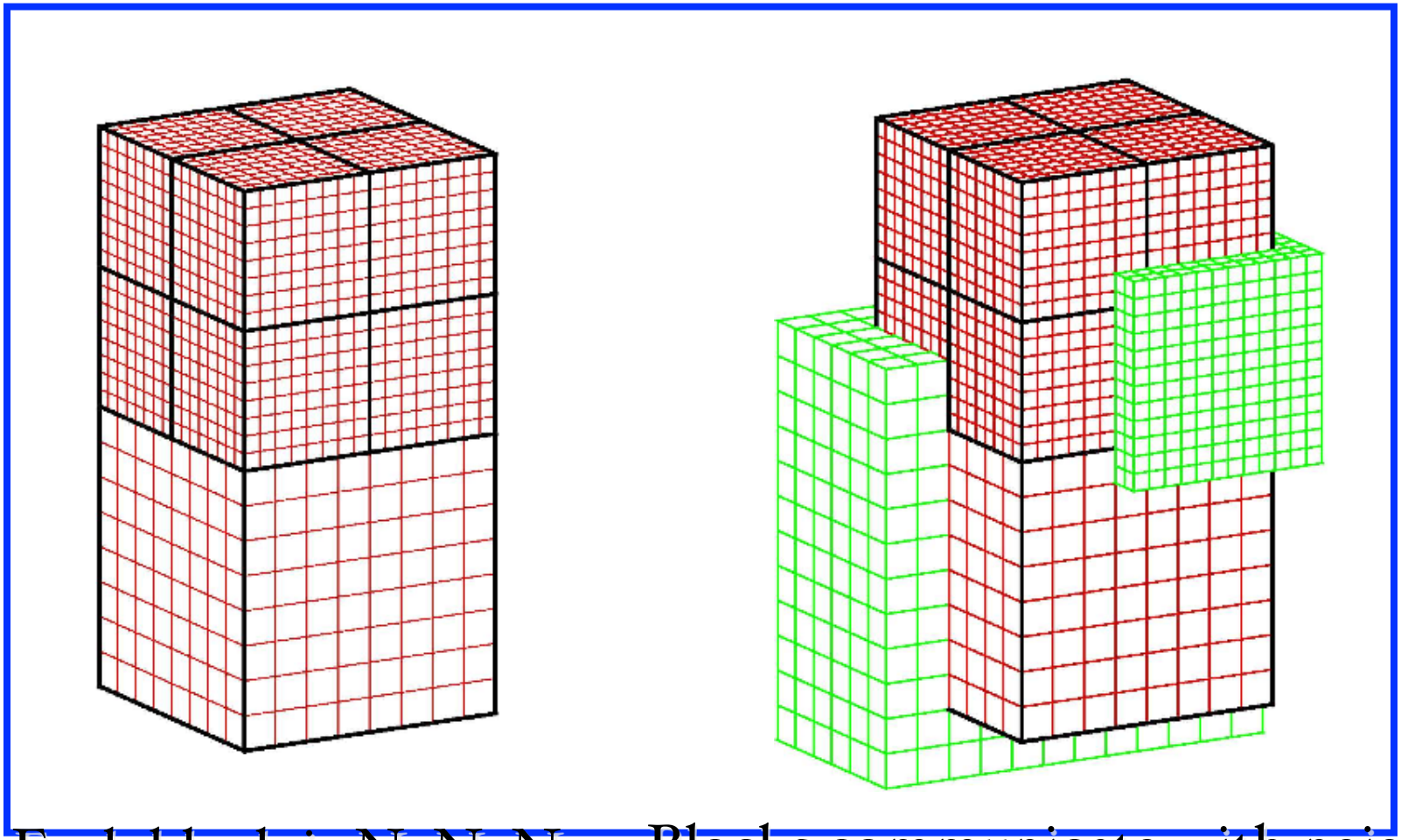
とっても面倒です



を採用しよう！

Block Structure (BATS-R-US)
Self-similar oct tree type



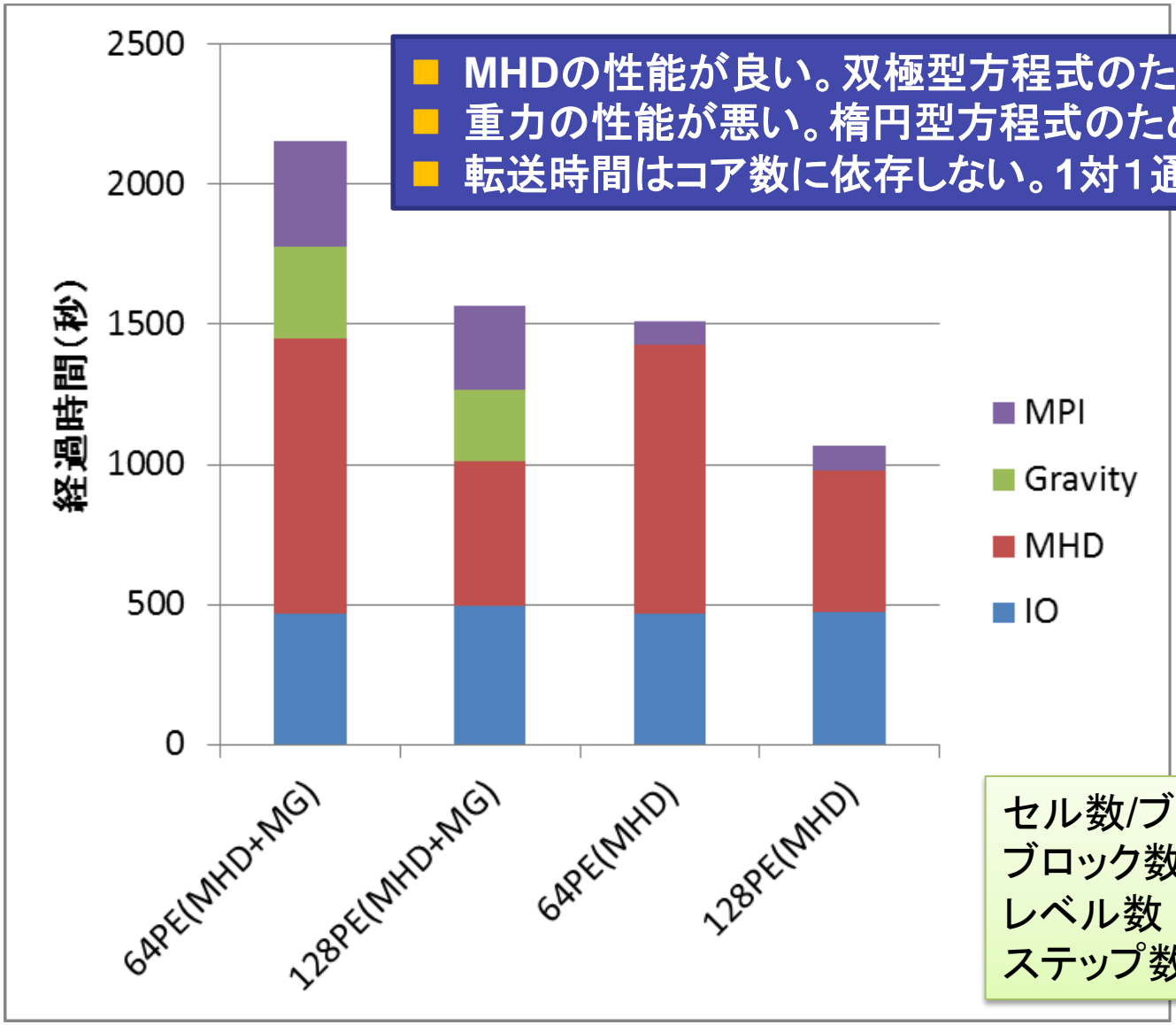


Each block is NxNxN

Blocks communicate with neighbors through “ghost” cells

性能評価

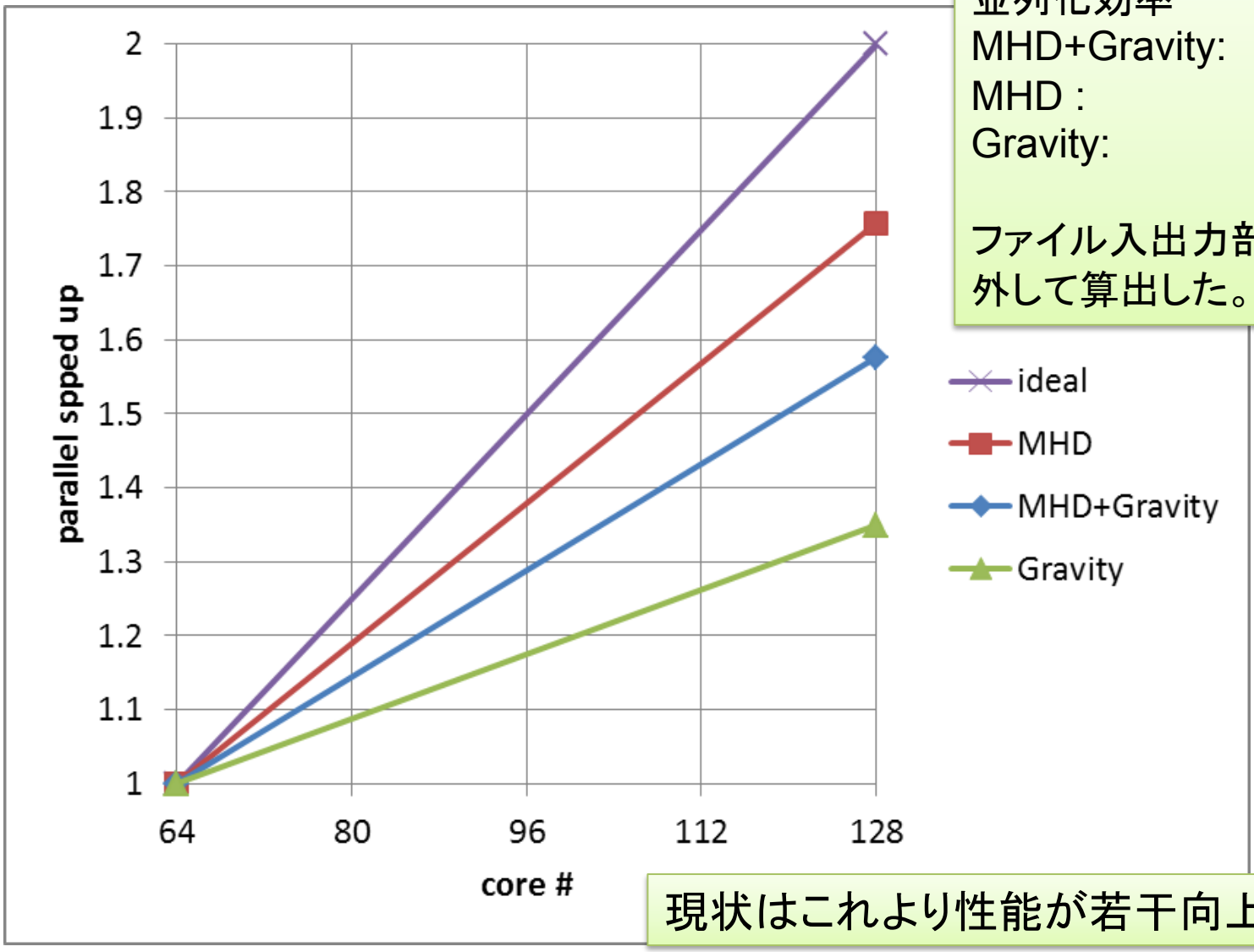
計算時間　64/128コア



- MHDの性能が良い。双極型方程式のため。
- 重力の性能が悪い。楕円型方程式のため。
- 転送時間はコア数に依存しない。1対1通信のため。

セル数/ブロック = 16^3
ブロック数/レベル = 16^3
レベル数 = 4
ステップ数 = 32

Parallel speed up, parallel efficiency



並列化効率
MHD+Gravity: 79 %
MHD : 88 %
Gravity: 67 %

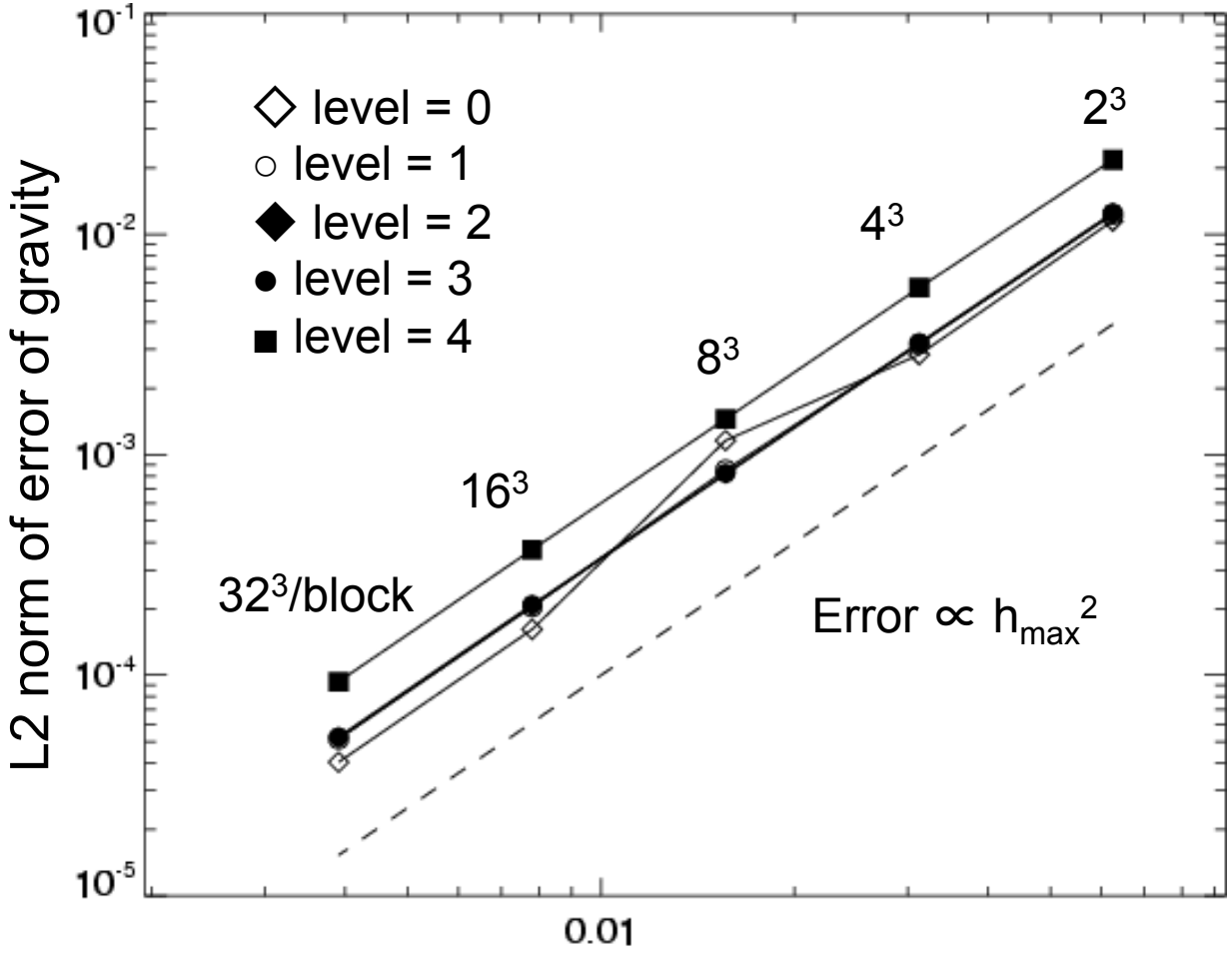
ファイル入出力部分を除外して算出した。

現状はこれより性能が若干向上している

精度評価

自己重力(Multigrid法)の精度

空間2次精度



- Source: binary stars
- Maximum level = 4
- Distribution of blocks is fixed.
- Number of cells inside a block is changed.

Matsumoto 07

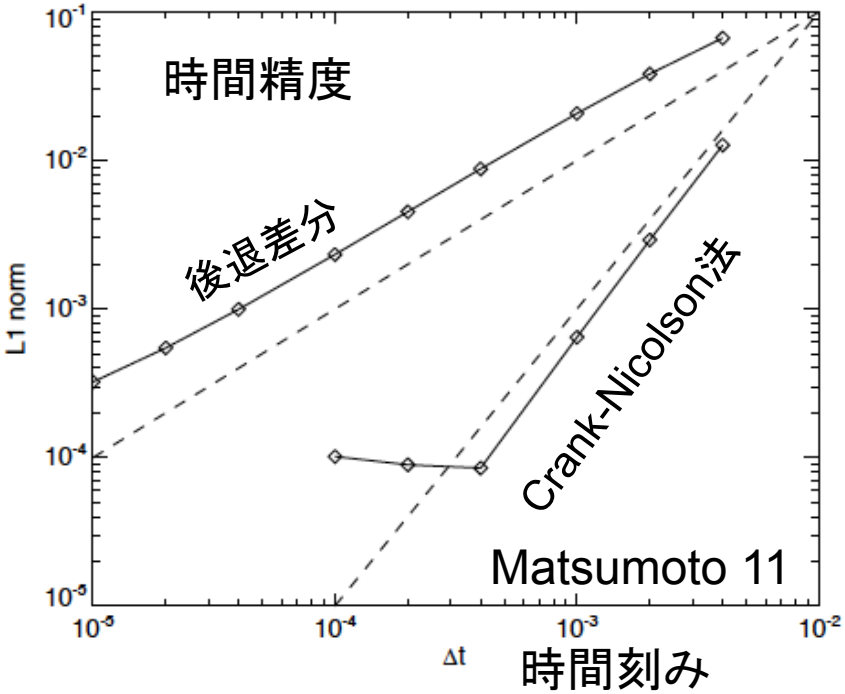
オーム散逸（Multigrid陰解法）の精度

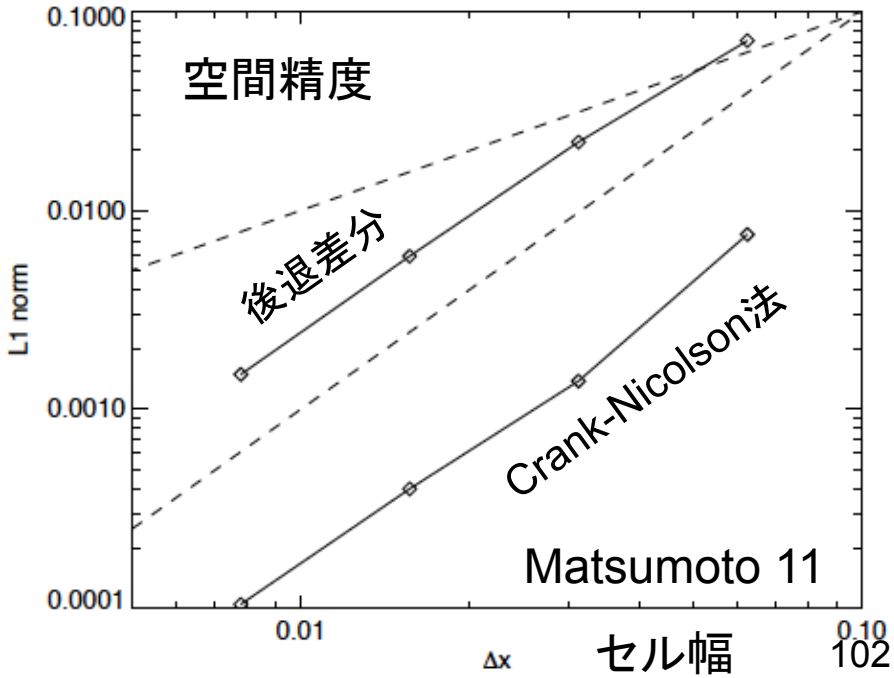
磁場の誘導方程式

$$\frac{\partial \mathbf{B}}{\partial t} = \nabla \times (\mathbf{v} \times \mathbf{B}) - \nabla \times (\eta \nabla \times \mathbf{B})$$

陽解法
時間空間2次精度

陰解法
時間1次／2次精度
空間2次精度

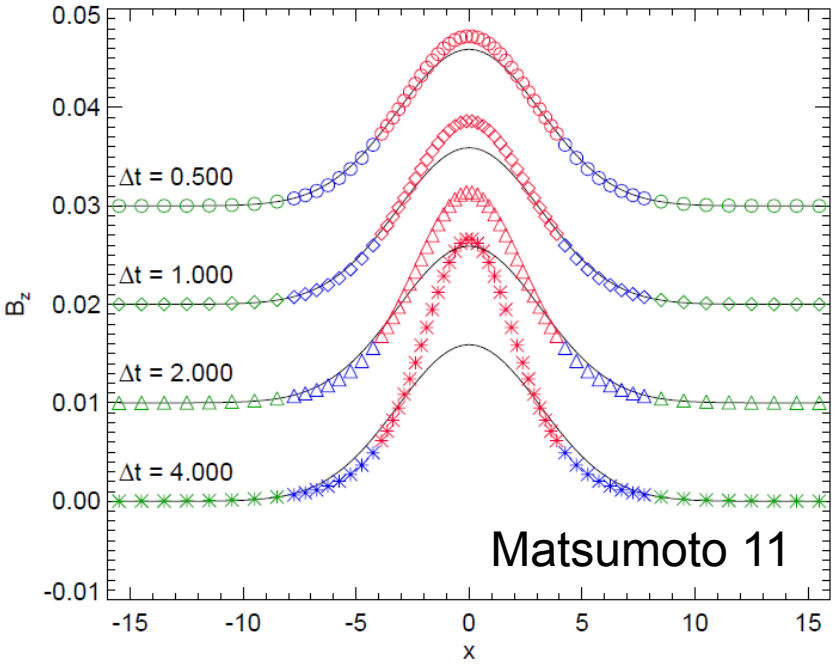




注意： 2次精度が良いとは限らない

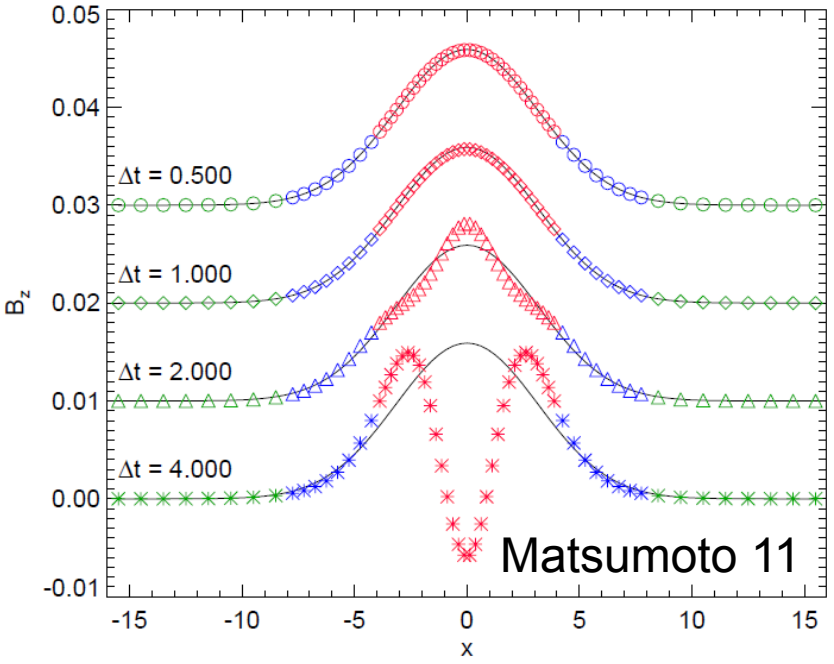
Gaussian が拡散する単純な問題。いろいろな時間刻み（Δt）の解。

後退差分（時間1次精度）



単調な解。
精度は悪い。

Crank-Nicolson法（時間2次精度）



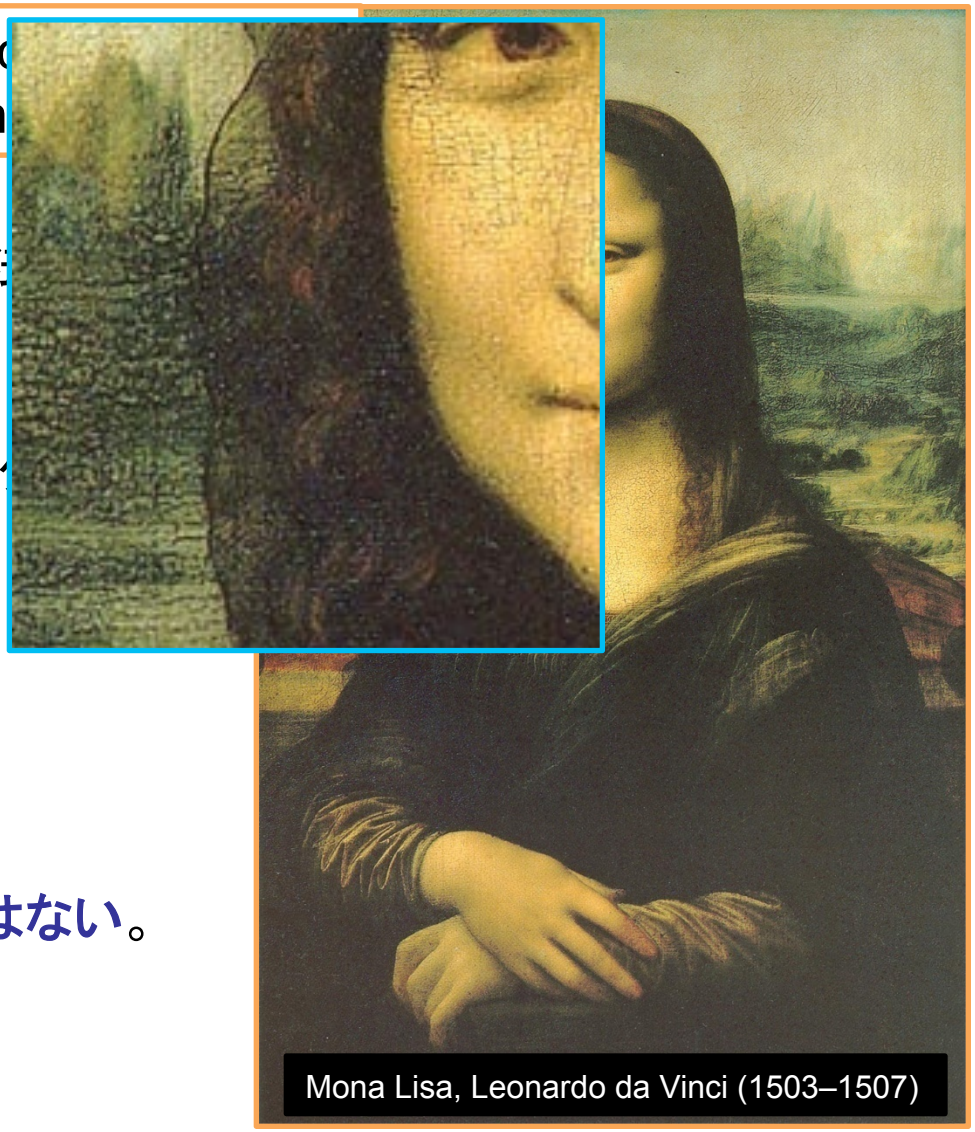
振動する解。
精度は良い。

※ 振動しても安定な解。振動は成長しない。

我々のAMRコード SFUMATOの紹介

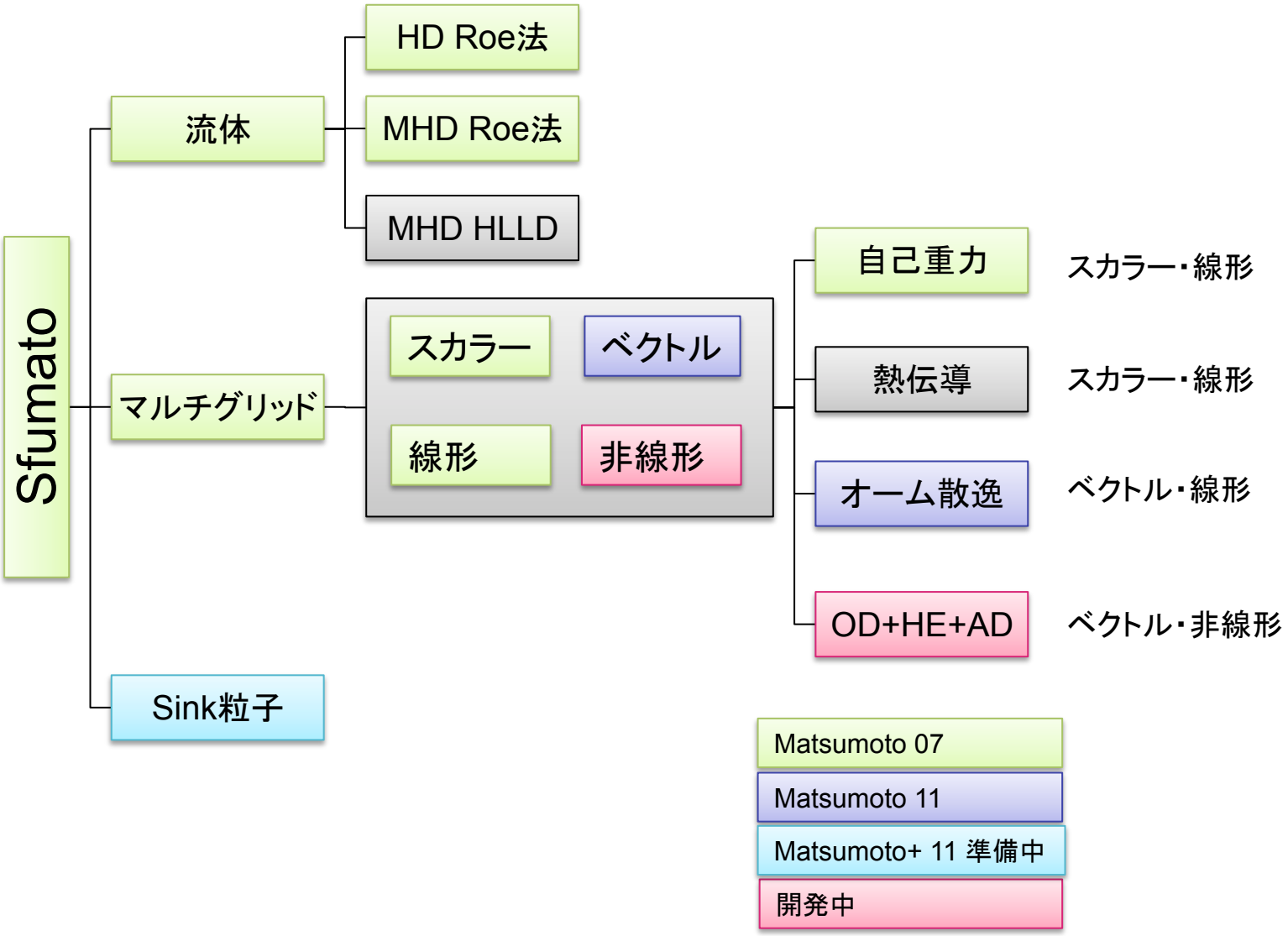
Type B

Self-gravitational **F**luid-dynamic
Mesh **A**daptive **T**echnique with

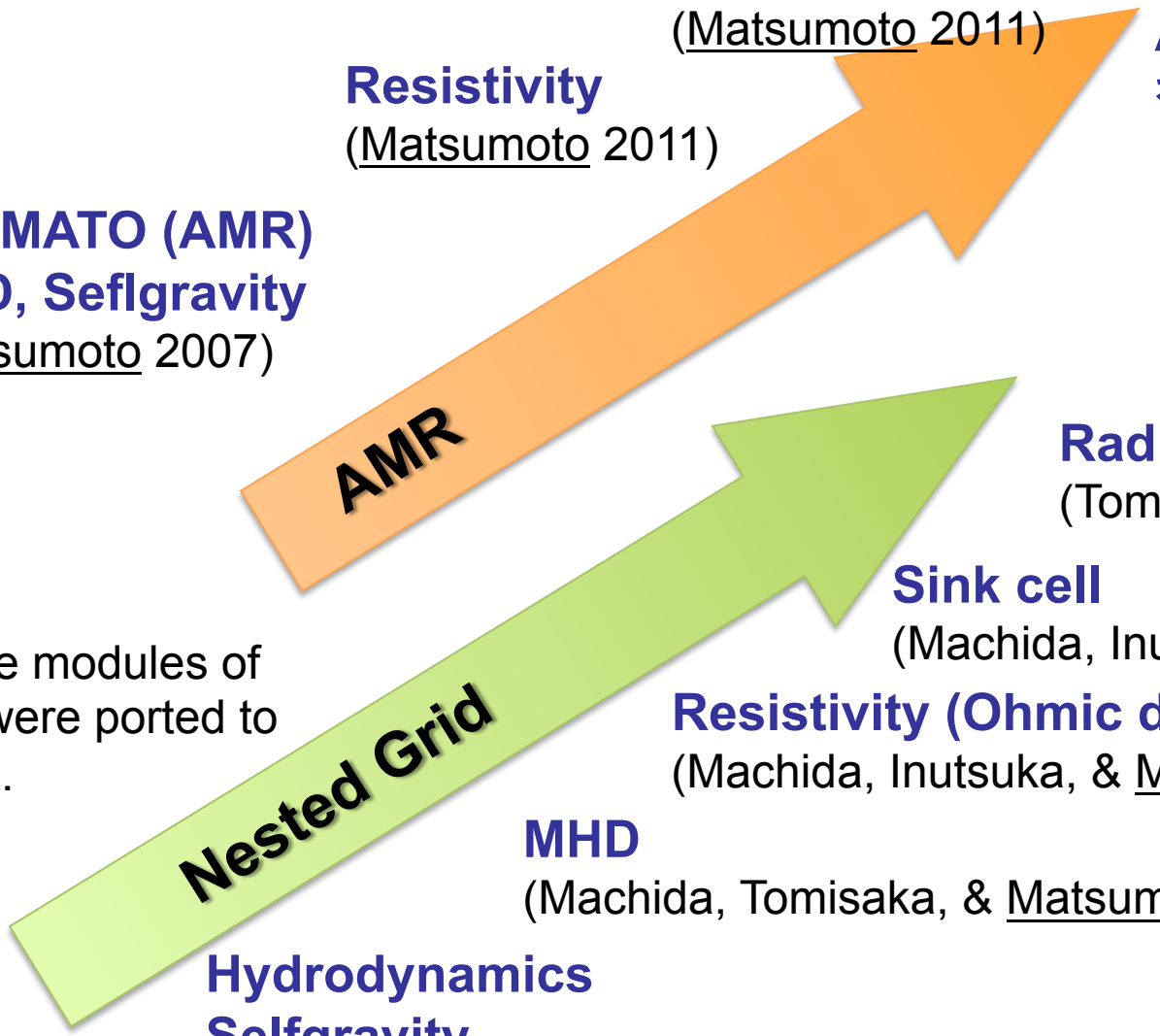


- *Sfumato* は本来、絵画の技法
  オナルド・ダ・ビンチ (1452–1
  によって完成された。
- その後、ルネサンス－バロック
  多くの画家に用いられた。
- モチーフの輪郭をぼかし、
  空気を表現。
- 我々のAMRコードも
  ガス(空気)を表現。
- Matsumotoのアナグラムで**はない**。

AMRコードSfumatoの構成



Adaptiveコードの開発



Sink particle
(Matsumoto 2011)

Ambipolar Diffusion
を開発中

Resistivity
(Matsumoto 2011)

SFUMATO (AMR)
MHD, Seflgravity
(Matsumoto 2007)

Lagrange particle
for chemistry
(Furuya et al. 2011)

Radiation
(Tomida et al. 2010)

Sink cell
(Machida, Inutsuka, & Matsumoto 2008

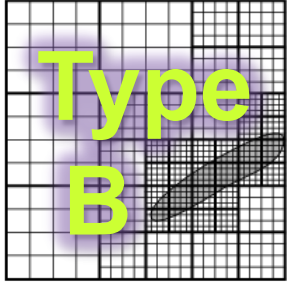
Some modules of
NG were ported to
AMR.

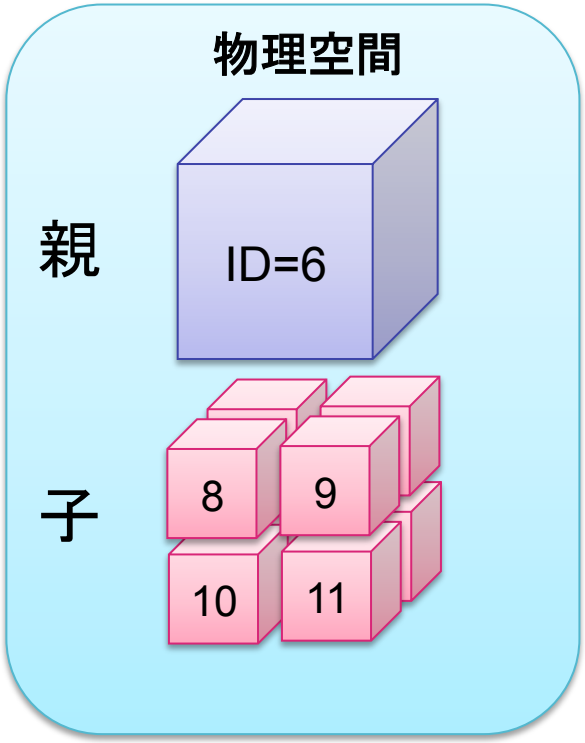
Resistivity (Ohmic dissipation)
(Machida, Inutsuka, & Matsumoto 2006)

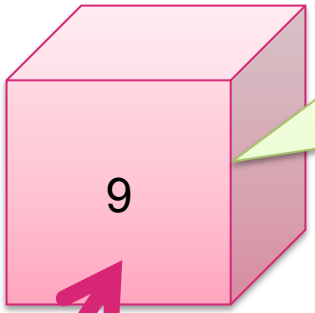
MHD
(Machida, Tomisaka, & Matsumoto 2004)

Hydrodynamics
Selfgravity
(Matsumoto & Hanawa 2003)

データ構造：ブロックの八分木構造



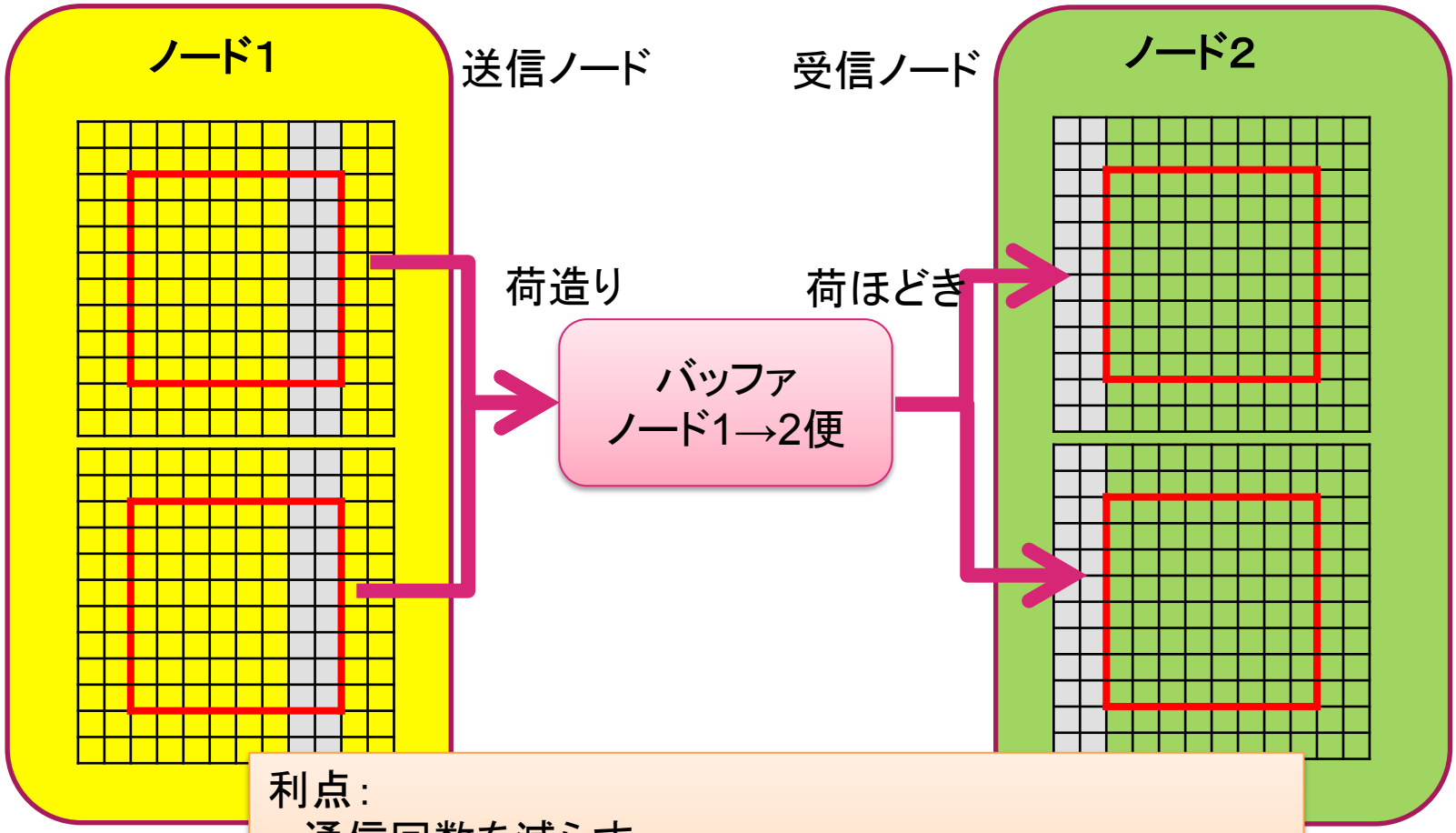




親のID×1
子供のID×8
隣のID×6
グリッドレベル
ブロックの位置(i,j,k)

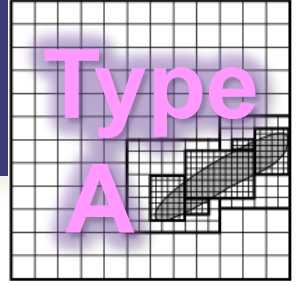
セル Nx×Ny×Nz 個

袖の転送：ノード間をまとめて転送する



利点：
- 通信回数を減らす。
  性能は通信量ではなく通信回数で決まる。
- MPI通信タグの枯渇を回避する。
  ブロックが個別に通信するとタグ数が上限を超える。

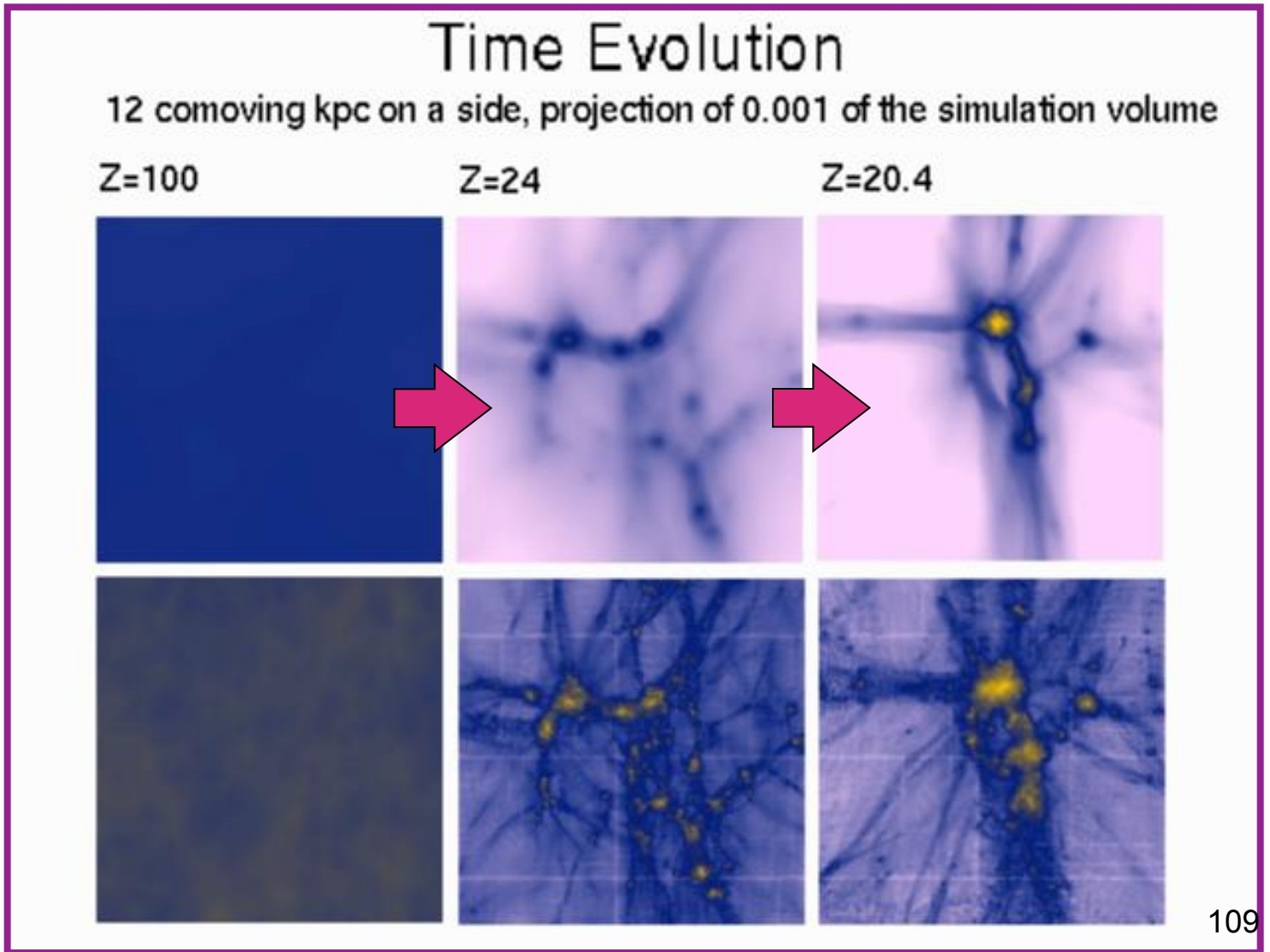
Enzo (M. Norman)
First star formation (Abel et al. 2003)



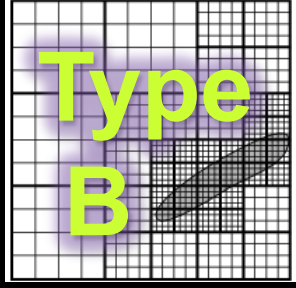
Block-structured grid (patch-oriented type)

Gas

Dark matter



タイプIa超新星爆発（FLASH）
色：反応率と密度



White Dwarf Deflagration

Resolution: 6 km

Initial Bubble Radius: 18 km

Ignition Offset: 42 km

Variable 1: Density [1.5e+07 - 2.0e+07]

Variable 2: Reaction Progress [0.0 - 1.0]

∇･B の処方箋 （AMRに限らない話）

- 非物理的な力 $(\nabla \cdot \boldsymbol{B})\,\boldsymbol{B}$ が計算を破壊する。
- 処方箋
  - Projection method
    - 後処理として、Poisson 方程式を解き、div B に寄与する磁場成分（スカラー場）を差し引く。
    - Poisson 方程式を解くので、重い。
  - Constrained Transport (CT) method
    - Staggered grid で、磁場をセル境界で定義する。
    - 丸め誤差の範囲で div B = 0が保証される。
    - AMRでは複雑。
  - 8-wave formulation
    - 8成分目として、div B の流れを解く。移流速度はガスの速度と同じ。
    - 簡単な実装だが、ソース項が必要になる。div B が溜まる。
  - **Hyperbolic divergence cleaning**
    - div B を等方的に移流させる。同時に、流れを減衰させる。
    - 簡単な実装だが、ソース項が必要になる。

Projection method

$$q = \nabla \cdot B$$ 磁荷を求める

$$\nabla^2 \Psi = q$$ Poisson 方程式を解く

$$B^{new} = B - \nabla \Psi$$ 磁荷が発生する磁場成分を差し引く

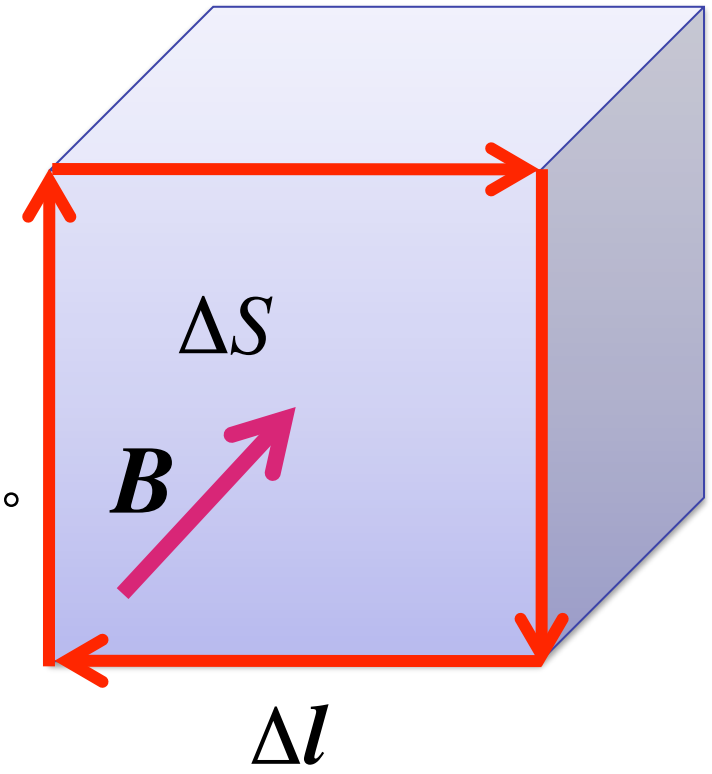
これをMHDが解き終わった後に施す。

解の性質は良いが、Poisson ソルバが重いのが弱点。

Constraint Transport (CT) method

$$\frac{\partial B}{\partial t} = \nabla \times (v \times B)$$ 誘導方程式

磁場をセルの面で定義する。



ストークスの定理を利用して

$$\frac{\Delta B}{\Delta t} = \frac{1}{\Delta S} \oint v \times B \cdot dl$$

この方程式にしたがって、磁場を時間積分する。
起電力v×B を数値流束から流用する。
丸め誤差の範囲で∇・Bが保存する。

磁場: セルの面で定義
数値流束: セルの面で定義
起電力 v×B: セルの辺で定義
風上性を考慮した内挿が必要。最近良い方法が見つかったらしい。

8 waves formulation

$$\partial_t \rho + \nabla \cdot (\rho \mathbf{u}) = 0,$$

$$\partial_t (\rho \mathbf{u}) + \nabla \cdot \left[\rho \mathbf{u}\mathbf{u}^T + \left(p + \frac{1}{2}\mathbf{B}^2 \right) \mathcal{I} - \mathbf{B}\mathbf{B}^T \right] = -(\nabla \cdot \mathbf{B})\mathbf{B},$$

$$\partial_t \mathbf{B} + \nabla \cdot (\mathbf{u}\mathbf{B}^T - \mathbf{B}\mathbf{u}^T) = -(\nabla \cdot \mathbf{B})\mathbf{u}, \qquad (36)$$

$$\partial_t e + \nabla \cdot \left[\left(e + p + \frac{1}{2}\mathbf{B}^2 \right) \mathbf{u} - \mathbf{B}(\mathbf{u} \cdot \mathbf{B}) \right] = -(\nabla \cdot \mathbf{B})\mathbf{u} \cdot \mathbf{B}.$$

磁荷∇・Bによる力を打ち消すように、
ソース項を付加する。

$$\frac{\partial B_x}{\partial t} = -(\nabla \cdot B)u_x$$

$$= -u_x \frac{\partial B_x}{\partial x} - \cdots$$

第8の波が現れる。
位相速度は u_x
磁荷∇・Bをuで運ぶ波。

よどみ点や衝撃波で∇・Bが溜まってしまう。

Hyperbolic divergence cleaning

実装が簡単で性質が良いので流行

$$\partial_t \rho + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{24a}$$

$$\partial_t (\rho \mathbf{u}) + \nabla \cdot \left[\rho \mathbf{u}\mathbf{u}^T + \left(p + \frac{1}{2}\mathbf{B}^2 \right) \mathcal{I} - \mathbf{B}\mathbf{B}^T \right] = -(\nabla \cdot \mathbf{B})\mathbf{B}, \tag{24b}$$

$$\partial_t \mathbf{B} + \nabla \cdot (\mathbf{u}\mathbf{B}^T - \mathbf{B}\mathbf{u}^T + \psi \mathcal{I}) = 0, \tag{24c}$$

$$\partial_t e + \nabla \cdot \left[\left(e + p + \frac{1}{2}\mathbf{B}^2 \right) \mathbf{u} - \mathbf{B}(\mathbf{u} \cdot \mathbf{B}) \right] = -\mathbf{B} \cdot (\nabla \psi), \tag{24d}$$

赤：mixed GLM formulation による項
橙：EGLM formulation による項

$$\partial_t \psi + c_h^2 \nabla \cdot \mathbf{B} = -\frac{c_h^2}{c_p^2} \psi. \tag{24e}$$

c_h と c_p はフリーパラメータ

固有値は9個

$$\lambda_1 = -c_h, \quad \lambda_2 = u_x - c_f, \quad \lambda_3 = u_x - c_a, \quad \lambda_4 = u_x - c_s, \quad \lambda_5 = u_x,$$
$$\lambda_6 = u_x + c_s, \quad \lambda_7 = u_x + c_a, \quad \lambda_8 = u_x + c_f, \quad \lambda_9 = c_h.$$

詳細は Dedner, Kemm, Kroner, Munz, Schnitzer, Wesenberg, 2002, JCP, 175, 645