2021/03/10-12, 22-23
CfCA / NAOJ

Athena++による シミュレーション実習

富田賢吾（東北大学）
岩﨑一成（国立天文台）
高棹真介（大阪大学）
小野智弘（東京工業大学）

この実習について

目標:

- 磁気流体力学シミュレーションの基本を学ぶ
- Athena++を用いて実際に**自分の研究**ができるようになる
- CfCAのXC50を初めとする大型計算機を使えるようになる

今回カバーしないこと:

- Athena++を自分の研究のために(大幅に)改造する
- Linux、プログラミング、並列計算の手法

経験者の方には簡単すぎる内容だと思います。
遠慮なく先に進んだり自分の仕事をして頂いて構いません。

実習にあたって

- 今回の講習会はCfCA史上おそらく最大の講習会です。我々にとってもこの規模の講習会をオンラインで運営することはチャレンジです。
- 講習会をスムーズに運営するには皆さんの協力が不可欠です。
- この講習会には文字通り教授から学部生まで参加しています。レベルはそれぞれ違いますので、各自のペースで進めてください。
- 必要に応じて部分的に参加して頂くことは構いません。ただし、それまでの内容は扱ったものとして講義・実習は進めますので、各自で学習しておくようにしてください。講義資料は全て公開しています。
- 基本編を受けてから応用編への参加希望を変更しても構いません。
- 難しすぎてついていけない・つまらなさすぎて耐えられない・体質に合わない等の場合はご自由に受講を中止して頂いて構いません。

実習にあたって

- 質問：口頭での質問を受け付ける時間も取りますが、原則として質問は随時Slackの各項目のチャンネルに書き込んでください。
- 悪い質問の仕方：「なんか動きません」
- 良い質問の仕方：「～～の課題に取り組んでいて、〇〇の所を試したのですが××というエラーが出ました。これは△△という意味だと思うのですが、どう対処すればいいでしょうか。」
- オンラインですので、困った顔をしていても我々には把握できません。助けを求めてもらわないと助けることができないので、問題が発生した場合にはそれを我々に伝える努力をしてください。
- 自助と互助のお願い：エラーが出た時は脊髄反射的に質問せず、まずはエラーメッセージを読んで理解する努力をしてください。
  また参加者同士でSlackを通じて教え合うことを推奨します。

公開コードの利用

公開コードの意義と必要性

- 現代では研究および計算機の発展によりシミュレーションは複雑化
  個人または少人数で最先端のコードを開発するのは極めて困難
  → コードを共有することでコストを削減し、科学的研究に集中できる
- 宇宙物理学において対象は異なっても多くの物理過程は共通
  → 開発したコード・技術・知見を共有し幅広い分野の発展に貢献
- 多くのユーザーが使うことでコードがテスト・比較・検証される
  → シミュレーションの品質（信頼性・性能・機能）の向上
- 優れたコードは国内外の共同研究ネットワークを構成するキーとなる
  → コードに魅力があればそれを通じて優秀な人材が集まる
  → 新しいアイディアや共同研究の可能性が広がる
- 優れたコードは数値シミュレーション・宇宙物理学の教材となる
  → 若手人材の体系的な育成に貢献する

公開コードの利用について

メリット:
- 開発のコストが下がり研究に注力できる(車輪の再発明の回避)
- 高性能のコードであれば計算時間も節約できる
- 信頼性あるコードなら論文の手法の説明が簡単にできる
- 同じコードを使っている人と知識を共有しやすい

デメリット:
- 簡単すぎる→結果を盲信する、自分で考えない、原理を理解しない
- 拡張が困難、あるいは整合性を保つコストが高い
- バグやバージョン間の不整合等の問題が発生し得る
- 研究の根幹を他人に依存するリスクと不安
- コードを書いてもコード開発能力が高いという評価を得るのが難しい

メリットとデメリットを意識して上手に利用して下さい。

公開コードの利用についてお願い

- Athena++に限らず公開コードには多大なリソースが費やされています。適切な引用と謝辞の記載をお願いします（実際に研究に使用した場合はもちろん、参考にした場合や一部を使用した場合も）。
- 使ったからといって開発者を共著に入れる必要はありません。
- バグや問題点・改善提案があれば速やかに報告して下さい（修正方法がわからなくても構いません）。ただし、人的リソースが限られているので時間がかかる場合もありますが、泣かない。
- 計算の結果に責任を負うのは開発者ではなく研究者自身です。コードを盲信せず、必ず物理的に計算が正しいことを確認しましょう。
- 特定のコードに忠誠を誓う必要はありません。問題に合わせて良い（機能・性能・信頼性）コードを選択してください。
- 上手に利用して効率よく、沢山の良い研究を推進して下さい。

コードの選び方

それぞれの手法・コードに長所・短所があり、全能のものはありません。

- サポートする機能・物理： 目的の計算を実現することができるか？
- 性能・互換性： 目的の計算を妥当な時間内に完了できるか？
  自分が使える計算機で動作するか？
- ドキュメント: 利用するために必要な説明が十分提供されているか？
- 信頼性: コードは安定で、十分にテストされているか（バグが少ないか）？
  長期的にサポート・開発が継続されるか？
- 可読性・拡張性: コードは理解しやすいか、必要な拡張が可能か？
- サポート: 開発者または他のユーザーからサポートは得られるか？特に、研究室内や近しい知り合いからの助けは重要になり得る。

特定のシミュレーションコードに忠誠を誓う必要はありません（誓うべきではありません）。問題に応じて最適な手段を選ぶことが重要です。

Athena++の紹介

Athena++とは

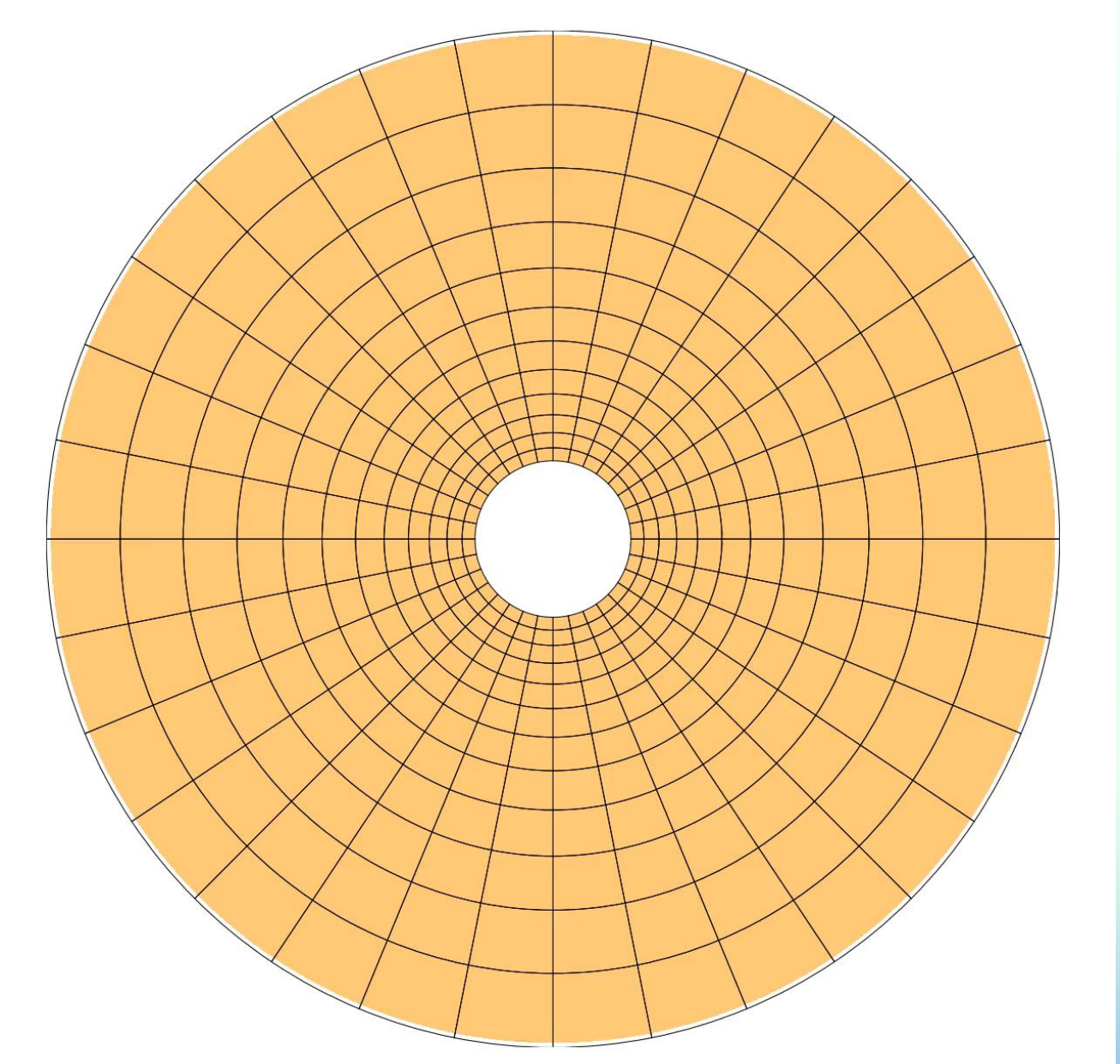
- Princeton高等研究所を中心に開発している公開MHDコード
- 論文: Stone, Tomida, White, and Felker (2020) ApJS
- 柔軟な座標系: **non-uniform spacing, Spherical, Cylindrical**…
- Static / **Adaptive Mesh Refinement (AMR)**（一様時間刻み）
- 実績ある近似リーマン解法＋CT法を用いた安定なMHDソルバ
- 多様な物理過程を含み幅広い宇宙物理学の問題に適用可能
  - 一般相対論、輻射輸送、自己重力、化学反応…
- 高い実行性能と大規模並列化：ベクトル化、動的スケジューリング他
- **Hybrid parallelization**: MPI + OpenMP
- **並列入出力 with MPI/HDF5**, 標準的な解析ソフトをサポート
- 使いやすく、習得しやすく、維持しやすい設計＋ドキュメント
- 厳重な品質管理：Intel CompilerやCray MPIライブラリのバグを発見
- 多様な大型計算機をサポート; Intel, IBM BG/Q, Xeon Phi etc.

Athena++の機能

公開済み	開発済み・未公開	開発中	計画中
MHD w/ PLM & PPM	Non-ideal MHD (Hall)	Particles (star / dust)	Radiation transfer (VTEF+implicit etc.)
Flexible coordinates	Radiation transfer (Direct transfer)	Cosmology	Ambipolar Diffusion (two fluid approx.)
Static and Adaptive Mesh Refinement	Chemical reactions	Post-processing radiation transfer	Heterogeneous parallelization
Special and General Relativity (fixed metric)	Full General Relativity (dynamic metric - GRAthena++ Daszuta et al.)	Hybrid PIC Plasma	
MPI + OpenMP	Self-gravity (Multigrid)	GPU/multi architecture	
Passive scalars		Developer's guide	
Self-Gravity (FFT)			
Shearing-Box			
Orbital Advection			
Resistivity & diffusion			
General EOS			
Website / Tutorial			
Code Paper			

赤：日本のグループが中心的に開発、**青**：日本のグループが開発に協力

Athena++ Logo Demonstration



Athena++

HOME
Download
Documentation
Publications

Athena++ Radiation MHD code

About

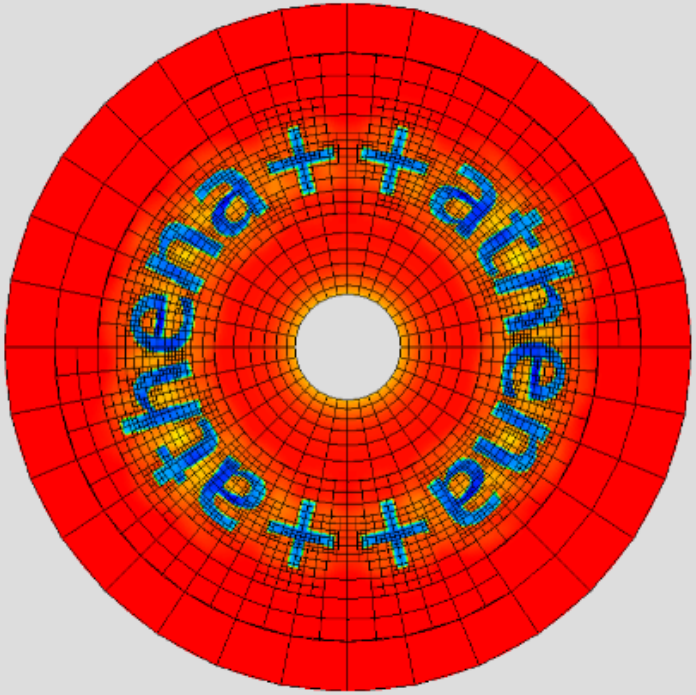
Athena++ is a complete re-write of the Athena astrophysical magnetohydrodynamics (MHD) code into C++. Compared to earlier versions, the Athena++ code has (1) much more flexible coordinate and grid options including adaptive mesh refinement, (2) new physics including general relativity, (3) significantly improved performance and scalability, and (4) improved source code clarity and modularity. Please understand that this is a **BETA** version, and expect frequent updates and improvements.

Core Developers

James M. Stone
Professor
Department of Astrophysical Sciences
Princeton University

Kengo Tomida
Assistant Professor
Department of Earth and Space Science
Osaka University

Christopher White
Department of Astronomy
University of California, Berkeley



http://princetonuniversity.github.io/athena/

Athena++

- HOME
- ダウンロード
- ドキュメント
- チュートリアル

Links

- Athena++ 公式
- 管理者のページ

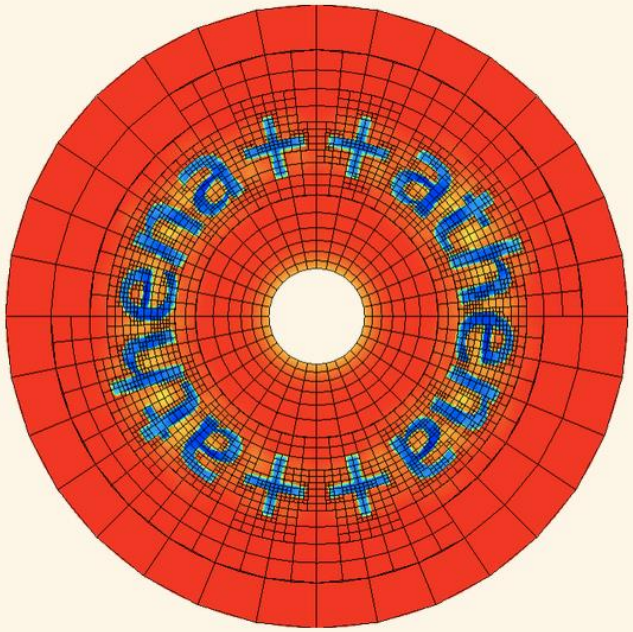
管理者

Kengo Tomida
Associate Professor
Astronomical Institute
Tohoku University

Athena++ 日本語サポートページ

紹介

Athena++はAthena MHDコードをC++言語で完全に再設計した新しい宇宙物理学用（輻射）磁気流体シミュレーションコードです。
近似リーマン解法とConstrained Transport法を組み合わせた磁気流体計算に加え、
(1) Adaptive Mesh Refinement(AMR)を含む柔軟な格子配置
(2) 一般相対性理論を含む多様な物理過程
(3) 現代的な計算機に対応した高い性能とかつ大規模並列性
(4) モジュール化された読みやすいコード 等の特徴があります。
現在公開しているコードはβ版であり、改善のため頻繁に更新される可能性があることを御理解の上ご利用ください。



http://www.astr.tohoku.ac.jp/~tomida/athena/

With Varvakeion Athena at The National Archeological Museum of Athens
(The most accurate reproduction of Athena in Parthenon, dated to 200-250 AD,
The original in Parthenon was 11.5m tall and produced in 5th century BC but lost.)

Athena++のライセンス

Athena++は3条項BSDライセンスで公開されています。これは非常に緩いライセンスで、日本語では以下のような意味になります。

変更の有無を問わず、ソースやバイナリ形式での再配布や利用は、次の条件を満たせば許可される。

1. ソースコードの再配布は、上記の著作権表示、ここに列挙された条件、および下記の免責条項を保持すること。
2. バイナリ形式の再配布は、上記の著作権表示、ここに列挙された条件、および下記の免責条項は、ドキュメントまたは他の資料で配布すること。
3. このソフトウェアのコントリビューター(貢献者)の名前は、特定の書面による事前の許可なしに、このソフトウェアから派生した製品の保証または販売促進のために使用してはいけない。

⇒つまり、コードの著作権表示を保持すればほぼあらゆる形の再利用が認められています。GPLと異なりコードの公開義務もありません。

Athena++の運用ポリシー

Athena++はコードの品質を保証するために、コードの開発やコントリビューションについて比較的保守的なポリシーを取っています。

- コードのコントリビューションは無条件には受け付けない。
- 本体に取り込むコントリビューションはコード全体にとって有益でありかつ責任を持って保守できるものに限る。
- 本体に取り込む機能は可能な限り他の全ての機能との互換性を保つこと。また性能に悪影響を及ぼしてはならない。
- コードは全てAthena++のデザインとルールに従い、他の部分との整合性を保たなければならない。
- 逆に独自拡張（パッチであれコード全体であれ）の配布は自由。

⇒ 開発に関わってみたい方は御相談下さい。

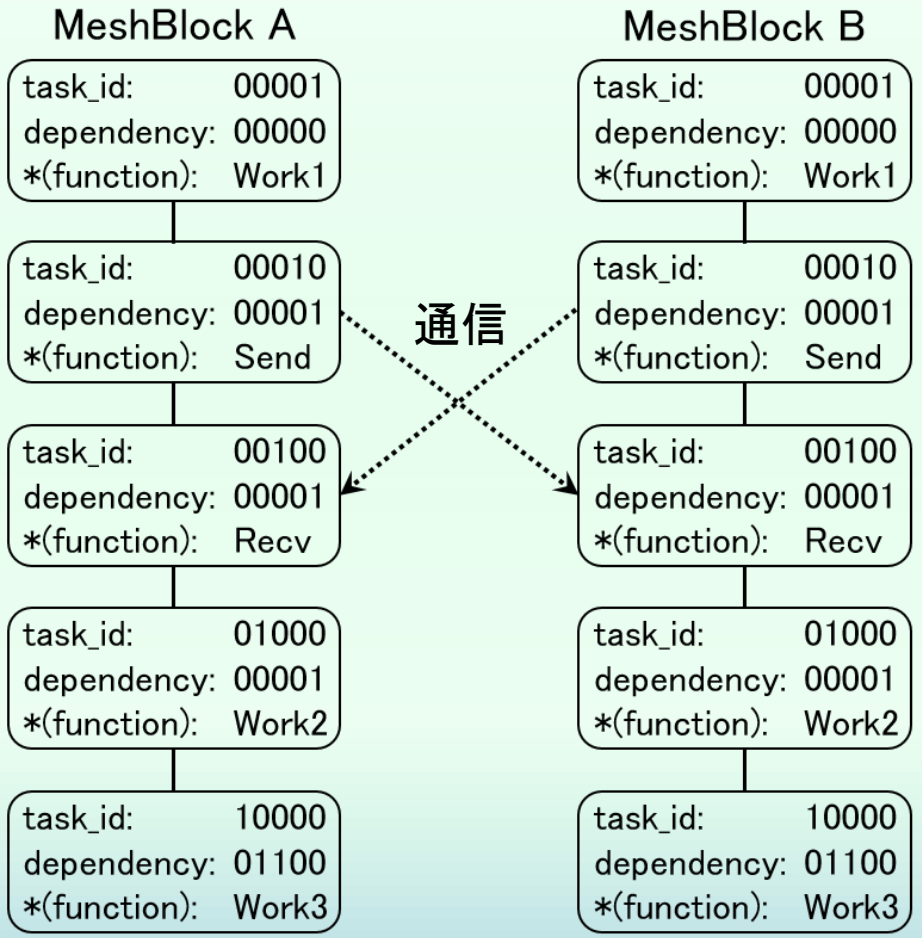
Athena++の特徴：TaskList

従来のシミュレーションコードでは
各計算ステップの実行順序は固定

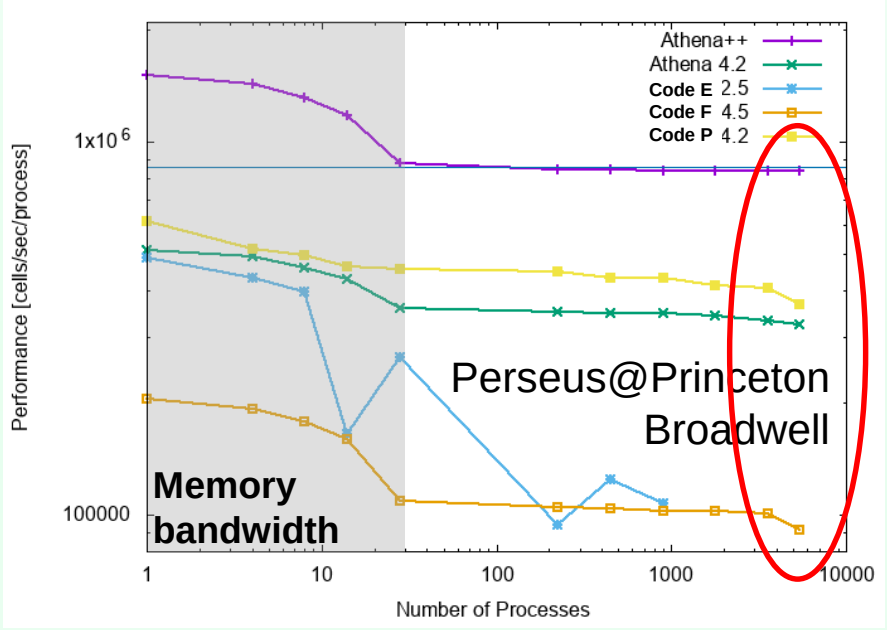
Athena++ではこれを小さなTaskに
分割し、その間の順序と依存関係
を記述するTaskListを導入

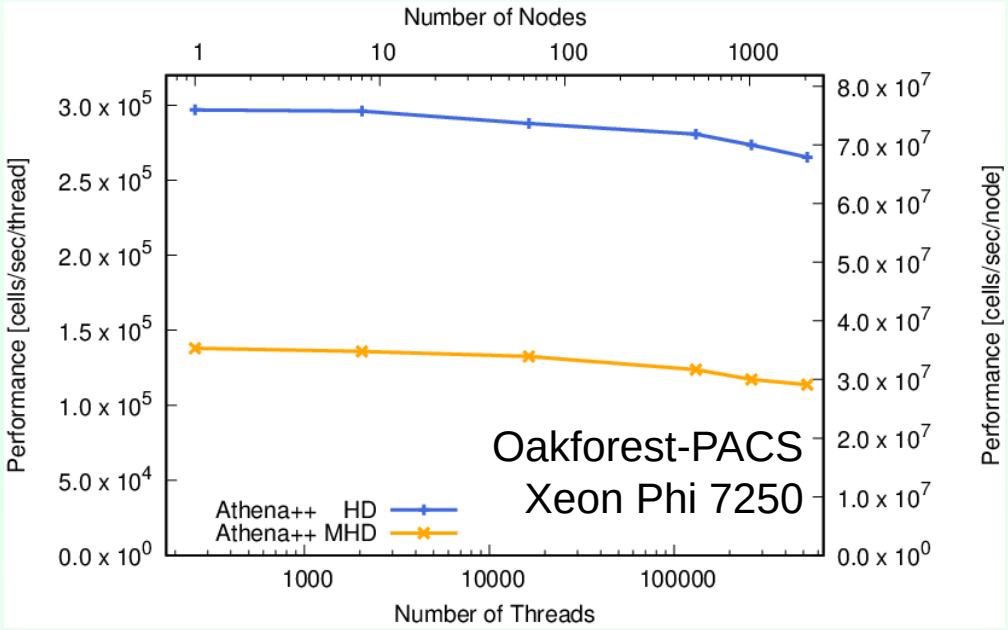
- TaskとTask間の依存関係を分離
→コードのモジュール化を促進
- Taskの実行順序は動的に決定
→通信と計算の同時実行が可能に

⇒開発の効率化・並列性能の向上
⇒領域・時間ごとに異なるTaskList
を適用することも可能



Athena++の性能





64^3 cells / core用いた理想磁気流体力学の弱スケーリングテスト

- ~10,000 並列でも>97%の弱スケーリング性能
- ~50万並列でも~85% という高い性能を発揮
- 公開されている宇宙物理学向けAMR MHDコードの中で最も高速
- 開発費用も非常に安い（FLASHとEnzoは大規模プロジェクト）

Athena++開発ネットワーク

国際協力でコード開発、ユーザーも順調に増加中

初回Developers & Users meeting: Las Vegas, 2019年春

Athena++ Workshop @ UNLV

流体力学の数値計算法の基礎

移流方程式

最もシンプルな双曲型方程式として移流方程式を考える:

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0, \qquad (\mathrm{c} = \text{流速、定数})$$

この方程式の一般解としてダランベールの解$u = f(x - ct)$がある(fは任意関数)。元の微分方程式はuが微分可能であることを要求しているが、ダランベールの解は微分できない関数(例えば階段関数)も許容する。そのような解を弱解と呼ぶ。

この方程式を陽的な(時間微分項以外は全て既知の情報だけで計算する)差分法で解くにはいくつかの方法が考えられる:

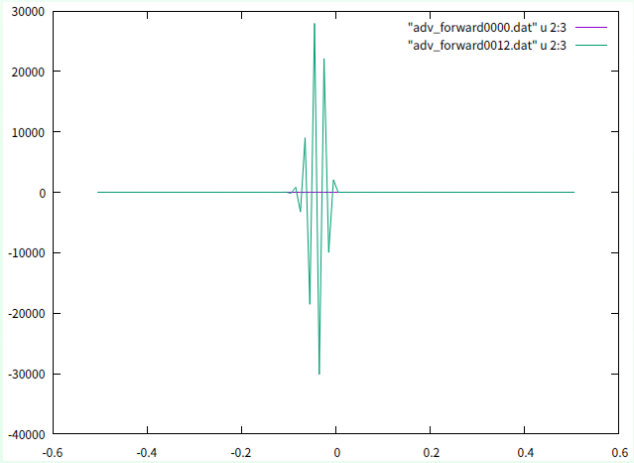
$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + c\frac{u_{i+1}^n - u_i^n}{\Delta x} = 0, \qquad \text{(前進差分)}$$

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + c\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x} = 0, \qquad \text{(中心差分)}$$

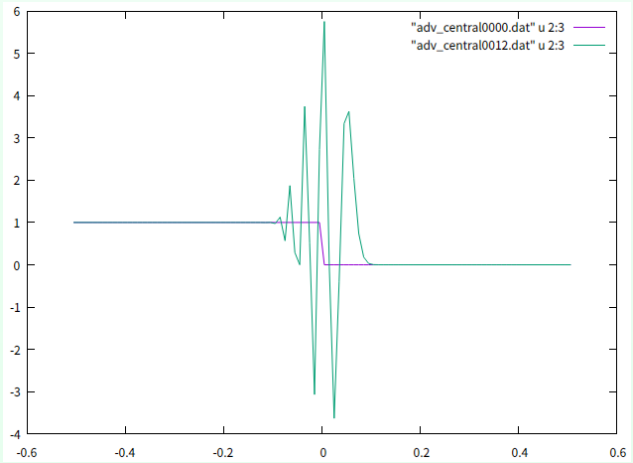
$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + c\frac{u_i^n - u_{i-1}^n}{\Delta x} = 0, \qquad \text{(後退差分)}$$

前進差分と後退差分がΔxについて1次精度なのに対し中心(中央)差分は2次精度。

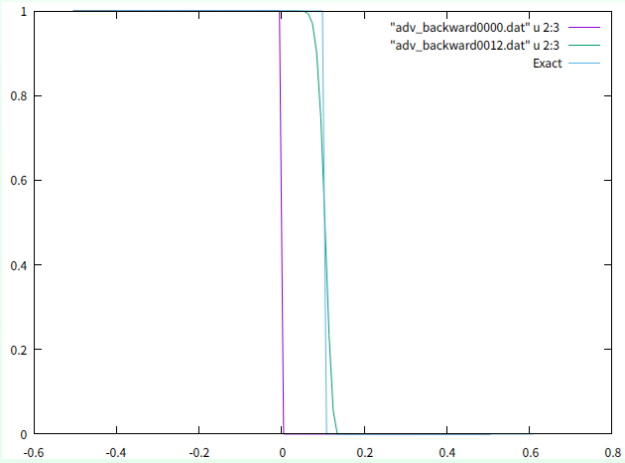
数値実験



前進差分



中心差分



後退差分

（階段関数をc=0.1, Δx=0.01, Δt=0.08, t=1.04まで移流させた時の結果）
Δtをどれだけ小さく取っても前進差分と中心差分は数値不安定（振動）を起こす。
一方、後退差分は急な多少鈍りはするものの正しい移流スピードを再現できる。
⇒ 中心差分(FTCS)は精度が良くても移流方程式には使い物にならない。

何故？ ダランベールの解f(x-ct)からわかるように、この系では情報は左から右にしか伝わらない。前進差分や中心差分では右からの非物理的な情報が入り込む。
情報の流れを考慮した後退差分＝風上差分だけが安定。

安定性とCFL条件

$$u_i^{n+1} = u_i^n - \frac{\nu}{2}(u_{i+1}^n - u_{i-1}^n), \qquad \text{(中心差分)}$$

$$u_i^{n+1} = u_i^n - \nu(u_i^n - u_{i-1}^n), \qquad \text{(後退差分)}$$

$\left(\text{ただし}\nu = \frac{c\Delta t}{\Delta x}\right)$。これをフーリエ成分に分解しvon Neumannの安定性解析を行う:

$$u_i^n = e^{ikx}, \qquad u_i^{n+1} = ge^{ikx}, \qquad (g\text{: growth rate})$$

中心差分について,

$$ge^{ikx} = e^{ikx} - \frac{\nu}{2}\left[e^{ik(x+\Delta x)} - e^{ik(x-\Delta x)}\right],$$

$$g = 1 - i\nu\sin(k\Delta x), \qquad |g| = \sqrt{1 + \nu^2 \sin^2 k\Delta x} > 1,$$

$|g| > 1$ は振幅が指数関数的に増大する、つまり無条件に不安定。一方後退差分は

$$g = \sqrt{1 - 2\nu(1-\nu)[1 - \cos(k\Delta x)]}, \qquad \text{(後退差分)}$$

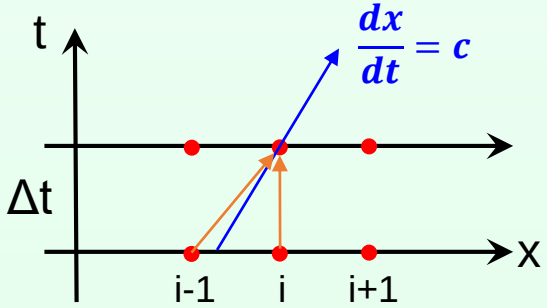
後退差分は$0 < \nu \leqq 1$で安定、つまり時間刻みを$\Delta t \leqq \frac{\Delta x}{|c|}$のようにとればよい。

この条件をCFL（Courant-Friedrichs-Lewy）条件と呼ぶ。

νをCFL数と言い、$\Delta t = \nu\frac{\Delta x}{|c|}$, $(0 < \nu \leqq 1)$のように書くことがある。νの安定な範囲は次元や離散化によっても変わるが、安定な範囲でできるだけ大きく取るのが良い。

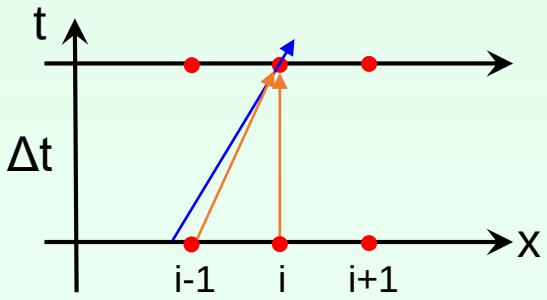
風上･中心差分の安定性

安定性とCFL条件について直観的に考えてみる。



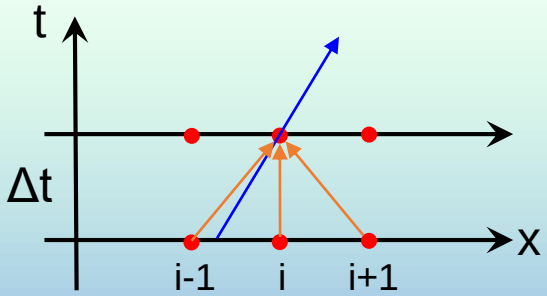
風上差分で$\Delta t \leq \frac{\Delta x}{c}$を満たしている場合

情報は青線に沿って伝わるので、x_{i-1}とx_iでの値を適当な重みで組み合わせれば（鈍るが）正常に動作する。



風上差分で$\Delta t > \frac{\Delta x}{c}$の場合

情報はΔtの間に1セルを飛び越えて伝わってしまうためx_{i-1}とx_iでの値だけから正しい結果を得ることができない。



中心差分の場合

Δtが小さくても、物理的な情報の流れとは逆流した情報を使って計算しているため、正しい結果を得ることができない。

流体力学の保存形

一次元オイラー方程式は以下のようなベクトルを用いて保存形に書ける：

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} = 0 \quad \text{ただし} \quad \mathbf{U} = \begin{bmatrix} \rho \\ \rho v_x \\ \rho v_y \\ \rho v_z \\ E \end{bmatrix} \quad \mathbf{F} = \begin{bmatrix} \rho v_x \\ \rho v_x^2 + P \\ \rho v_x v_y \\ \rho v_x v_z \\ (E+P) v_x \end{bmatrix},$$

この方程式も双曲型の保存形なので、数値的に解くにはやはり風上性を考慮する必要がある。しかし、この方程式は非線形で、また複数の変数が結合しているため、どのように「風上」を判定していいかは自明ではなく、また成分毎に「風上」が異なる。

これを解くにはいくつかの方法がある：

1. Lax-Friedrichs法のように、適当に中心差分で離散化した上で拡散（人工粘性）を入れて安定化させる→動くが、解が鈍るため現代ではあまり使われていない
2. 無条件に安定な陰解法を使う→複雑で計算コストが高いためあまり使われない
3. 式を変形して幾つかの方程式の組に分離する（特性曲線）
4. セルの間での流束を物理的な解に基づいて決める（Riemann Solver）

有限差分法

2次元の場合で考える。この保存形の方程式を離散化する一つ目の方法は、微分を有限間隔の差分で近似する有限差分法である:

$$\frac{\Delta U}{\Delta t}+\frac{\Delta F_x}{\Delta x}+\frac{\Delta F_y}{\Delta y}=0$$

$$\frac{U_{i,j}^{n+1}-U_{i,j}^{n}}{\Delta t}+\frac{F_{x;i+1/2,j}-F_{x;i-1/2,j}}{\Delta x}+\frac{F_{y;i,j+1/2}-F_{y;i,j-1/2}}{\Delta y}=0$$

i-1,j+1	i,j+1	i+1,j+1
i-1,j	i,j	i+1,j
i-1,j-1	i,j-1	i+1,j-1

i, jはそれぞれx, yを離散化したセルの番号。この場合、各格子点は上式を評価するサンプリング点であり、$U_{i,j}^{n}$はその点におけるUの値であると解釈される。

流束Fの具体的な様式が与えられていれば、微分を風上性を考慮した適切な差分で近似することで、Uをどのように時間発展させればよいかが決まる。

有限体積法

もう一つの考え方は、空間を微小なセルに区切り、このセルの体積について積分した方程式を考える。この手法を有限体積法と呼ぶ。

$$\frac{\partial}{\partial t}\int U dV + \oint \boldsymbol{F}\cdot\boldsymbol{n} dS = 0$$

$$\frac{\partial}{\partial t}\langle U\rangle \Delta V + \sum \langle \boldsymbol{F}\rangle \cdot \boldsymbol{n}\Delta S = 0$$

（ここまでは数学的に厳密な式であることに注意。）

$$\frac{U_{i,j}^{n+1} - U_{i,j}^{n}}{\Delta t} + \Delta S_x \frac{F_{x;i+1/2,j} - F_{x;i-1/2,j}}{\Delta V} + \Delta S_y \frac{F_{y;i,j+1/2} - F_{y;i,j-1/2}}{\Delta V} = 0$$

2次元デカルト座標では $\Delta V = \Delta \mathrm{x}\Delta \mathrm{y}, \Delta S_x = \Delta \mathrm{y}, \Delta S_y = \Delta \mathrm{x}$ である。この時、Uはセル内の値の平均値として定義されている。またFもセルの表面積で平均した流束である。この場合も、セル表面での流束Fが決まればUを時間発展させることができる。

代入してみればわかるが、この場合前頁の有限要素法の表式と見かけ上一致する。

有限差分法と有限体積法の違い

例えば3次元球座標で考えると

$$\boldsymbol{\nabla}\cdot\boldsymbol{F}=\frac{1}{r^2}\frac{\partial}{\partial r}(r^2F_r)+\frac{1}{r\sin\theta}\frac{\partial}{\partial\theta}(\sin\theta\,F_\theta)+\frac{1}{r\sin\theta}\frac{\partial}{\partial\varphi}F_\varphi$$

このうち動径(r)方向だけ考えると、有限差分法では

$$\frac{\partial U}{\partial t}+\frac{1}{r^2}\frac{\partial}{\partial r}(r^2F_r)\sim\frac{U_{i,j}^{n+1}-U_{i,j}^{n}}{\Delta t}+\frac{r_{i+1/2}^2F_{r;i+1/2}-r_{i-1/2}^2F_{r;i-1/2}}{r_i^2\Delta r}$$

有限体積法では球座標上の微小体積・面積について積分するので

$$\Delta V=\iiint r^2\sin\theta\,drd\theta d\varphi\,,\qquad \Delta S_r=\iint r^2\sin\theta\,d\theta d\varphi$$

$$\frac{\partial}{\partial t}\int UdV+\oint\boldsymbol{F}\cdot\boldsymbol{n}dS\rightarrow\frac{U_{i,j}^{n+1}-U_{i,j}^{n}}{\Delta t}+\frac{\Delta S_{r;i+1/2}F_{r;i+1/2}-\Delta S_{r;i-1/2}F_{r;i-1/2}}{\Delta V}$$

$$\Rightarrow\frac{U_{i,j}^{n+1}-U_{i,j}^{n}}{\Delta t}+\frac{r_{i+1/2}^2F_{r;i+1/2}-r_{i-1/2}^2F_{r;i-1/2}}{r_{i+1/2}^3-r_{i-1/2}^3}$$

この分母は$r_i^2\Delta r$と一致しない。有限体積法の方が保存則と整合的な離散化である。

補足：Euler法とLagrange法

ここではu(x,t)を「場」の変数ととらえて、空間上の動かない各点でこれを離散化した。このような形式をEuler法と呼ぶ（流体力学に出てくるEuler微分に対応）。

一方、流体力学に出てくるLagrange微分

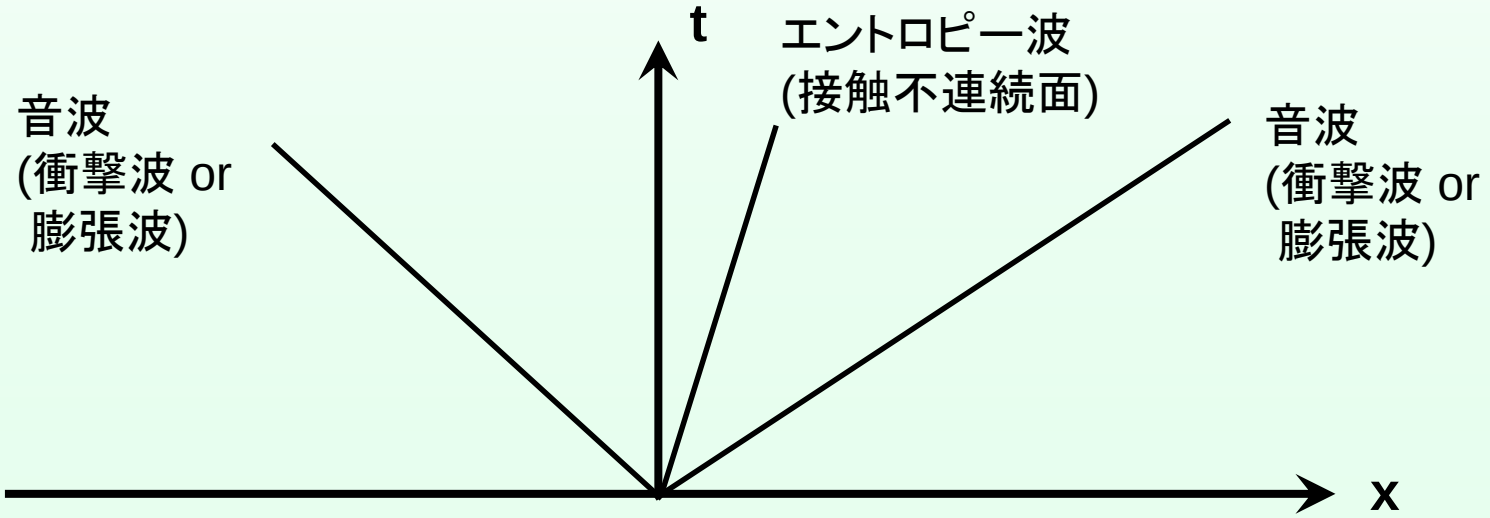
$$\frac{D}{Dt} = \frac{\partial}{\partial t} + \boldsymbol{v} \cdot \boldsymbol{\nabla}$$

に対応して、「流れに乗った」点やセルを用いて離散化する方法をLagrange法という。これを計算機上で表現するには粒子を使う方法（例：SPH = Smoothed Particle Hydrodynamics）や移動・変形する格子を使う方法（Moving Mesh）がある。

Euler法は規則正しい格子を用いることが多く比較的実装しやすい。Lagrange法は時間発展で格子や粒子分布が変化するため複雑になりがちだが、移流に強い（セルを跨ぐことによる拡散がない）等の利点がある。

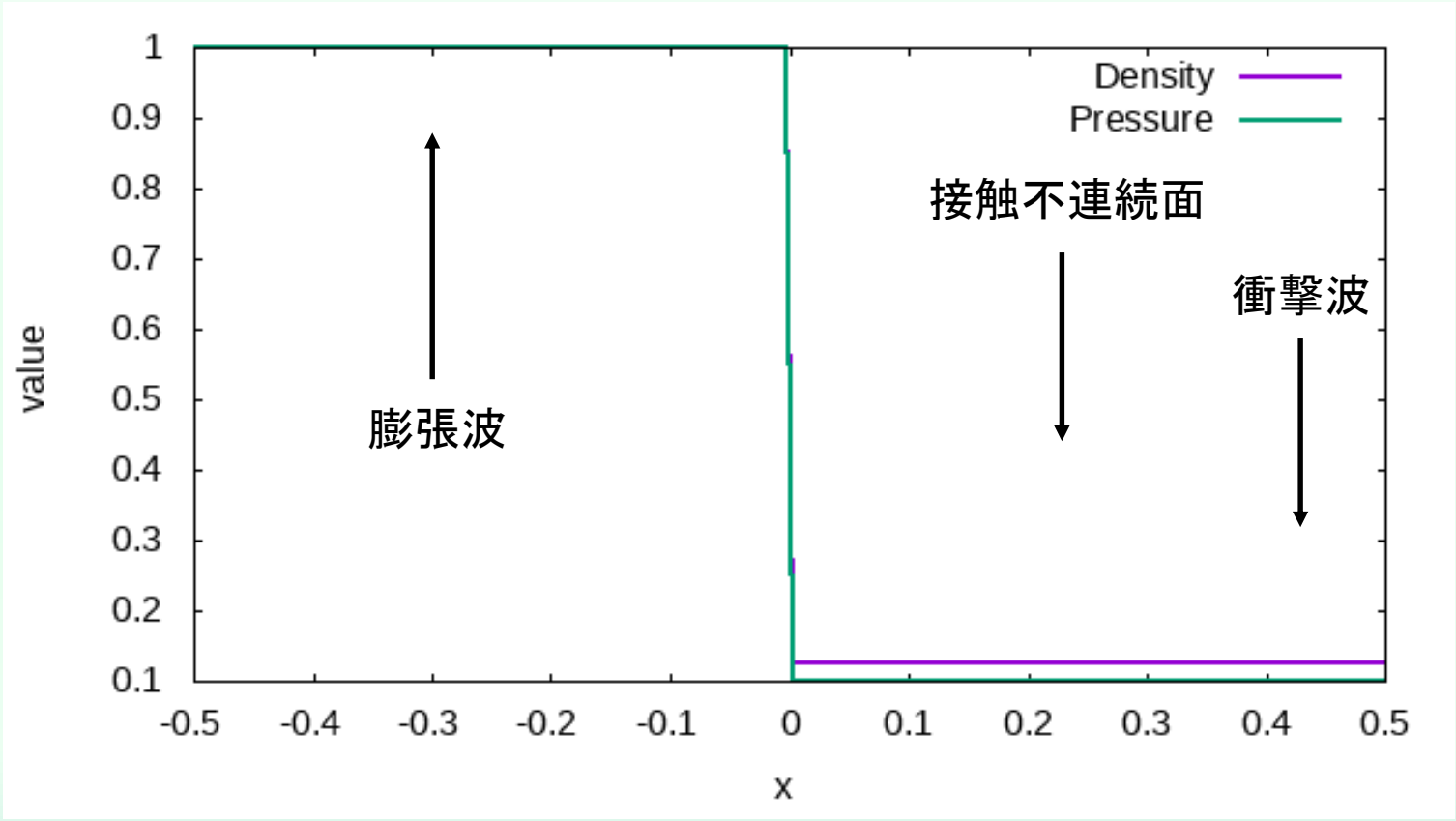
それぞれに得意不得意があるので、問題に応じて適切な手法を選ぶことが重要。Athena++はEuler法のコード。

リーマン問題



シミュレーションにはセル境界での流束Fが必要。
「左右のセルで物理量が異なるときに、その間にどのような流束が流れるか」
このような問題をリーマン（Riemann）問題と呼び、解析解が存在する。
左側のセルが $(\rho, v_x, v_y, v_z, P) = (\rho_l, v_{x,l}, v_{y,l}, v_{z,l}, P_l)$
右側のセルが $(\rho, v_x, v_y, v_z, P) = (\rho_r, v_{x,r}, v_{y,r}, v_{z,r}, P_r)$
の時、どのようなことが起こるだろうか？
オイラー方程式に従う流体の場合、この流れは3つの波で特徴づけられる：
左行きの音波、エントロピー波、右行きの音波

リーマン問題（続き）



リーマン問題（別名衝撃波管問題）の代表的な例: Sod の衝撃波管 ($\gamma = 7/5$)
(ρl, vx,l, vy,l, vz,l, Pl) = (1,0,0,0,1),　(ρr, vx,r, vy,r, vz,r, Pr) = (0.125,0,0,0,0.1)
この解は自己相似的である（形を保ったまま左右に広がる）。
Riemann solverではこの問題の（近似）解を用いて各セルの境界の流束を求める。

Riemann Solver

流体力学については多数の（近似）Riemann Solverが提唱されているが、ここでは代表的なものを挙げる：

- Exact Solver: リーマン問題の解析解を数値的に求める。精度良く解けるが、計算コストが高く実装も複雑になりがちであまり使われていない。
- Roe's Solver: 方程式を線形化し対角化することで固有の波に分解し、それらの重ね合わせで流束を求める。全ての波を解くことができるが、線形化の副作用で特に強い膨張波などで不安定になり易い。
- HLL, HLLE: 右行きの音波と左行きの音波を衝撃波として扱い、その中間状態は一様であると仮定し、Rankine-Hugoniot条件で状態間を繋ぐことで近似的に流束を求める。計算コストが安く安定だが、精度は低い。
- HLLC: 音波に加え、接触不連続面も含めた中間状態を考えることで、全ての波を解くことができる。Roe法と同程度の実用上十分な精度があり、かつ安定。

これらの方法は物理的な解を用いているので、安定化させるのに人工的な拡散を必要としない（必要なだけの人工粘性が自動的に入る、という解釈もできる）。

安定性と高次精度化

数値不安定（振動）を起こさないことを言い換えると、
「ある時刻でxについて単調な領域は次の時刻でも単調でなければならない」
ことになる。あるいは「新たな極大・極小点を生み出さない」とも言い換えられる。

実は、安定性と高次精度化の間には非常に強い制限が存在する。
Godunovの定理：
「u_i^{n+1}がu_i^nの線形結合で表される2次以上のスキームは単調性を維持できない」
あるいは「u_i^{n+1}がu_i^nの線形結合で表される安定なスキームは高々１次精度である」
つまり、単純な線形補間を組み合わせるだけでは高次精度スキームは作れない。
（風上差分が何故安定かというと、風上の判定に非線形な操作が入っているため。）

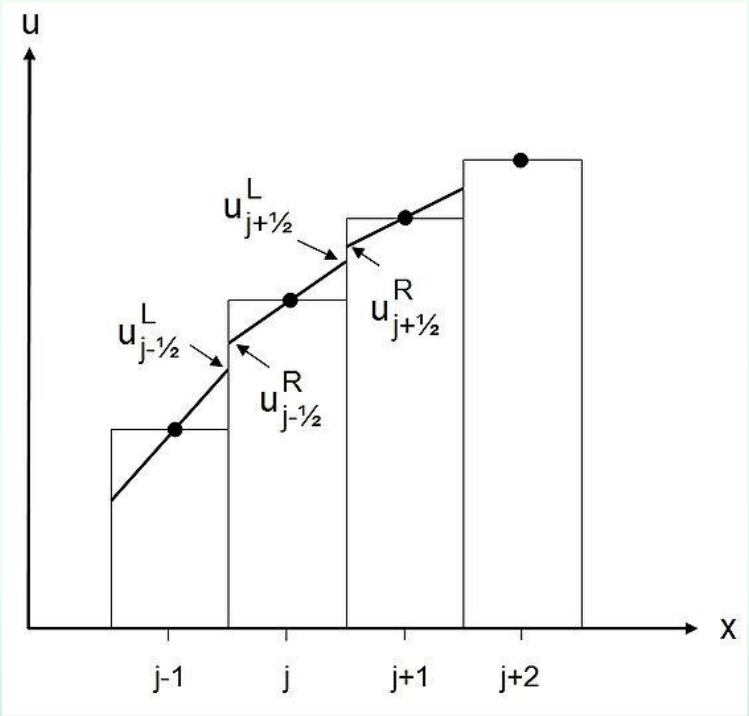
補足：Total Variance Diminishing
詳細には立ち入らないがTotal Varianceという量を次で定義する：

$$\mathrm{TV}(u_i^n) = \sum_i |u_i^n - u_{i-1}^n|$$

この量が増加しない手法、つまり$\mathrm{TV}\left(u_i^{n+1}\right) \leq \mathrm{TV}(u_i^n)$のものをTVDであると言い、
TVDであることと単調性を満たすことが同等であることが示されている。

MUSCL法

Godunovの定理を乗り越えて高次精度化する処方箋:
Monotonic Upstream-centered Scheme for Conservation Laws (van Leer 1979)



(図:Wikipedia)

有限体積法では各セルの離散化された値u_i^nは
そのセル内の値の平均値であった。

この平均値を保つようにセル内部の物理量に分布を考える。例えば1次元デカルト座標において、
各セル内で1次関数的な分布まで考えると:

$$u_i^n(x) = u_i^n + \delta_i(x - x_i)$$

勾配δ_iの決め方は様々だが、中心差分を取ると

$$\delta_i = \frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x}$$

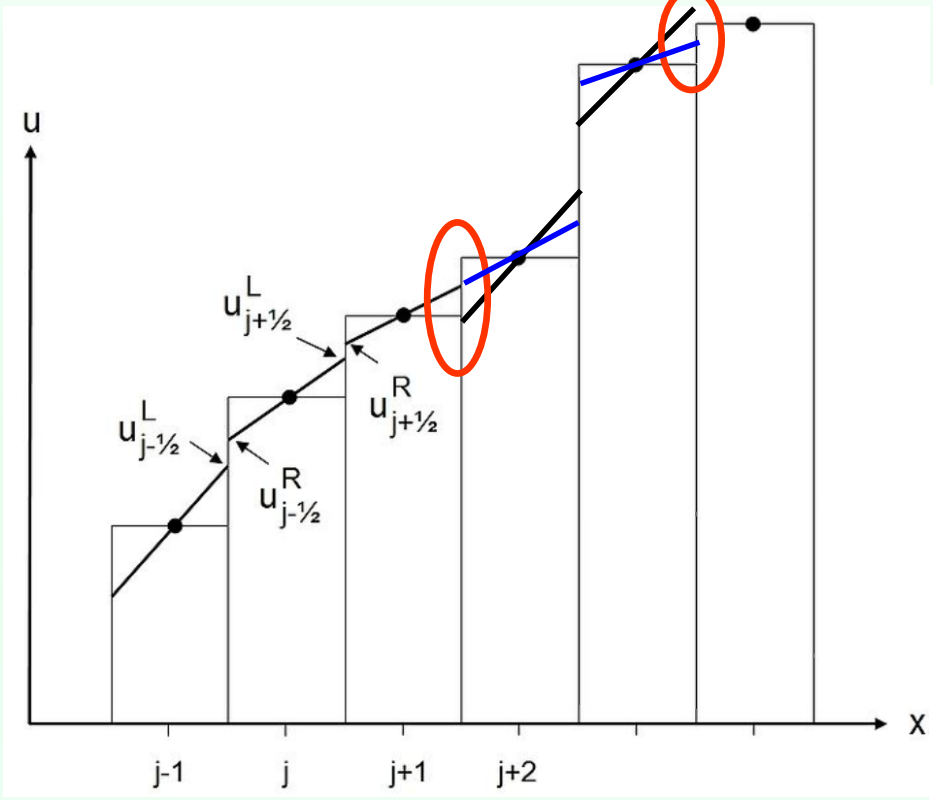
すると、各セル境界で左側と右側の値が得られる:

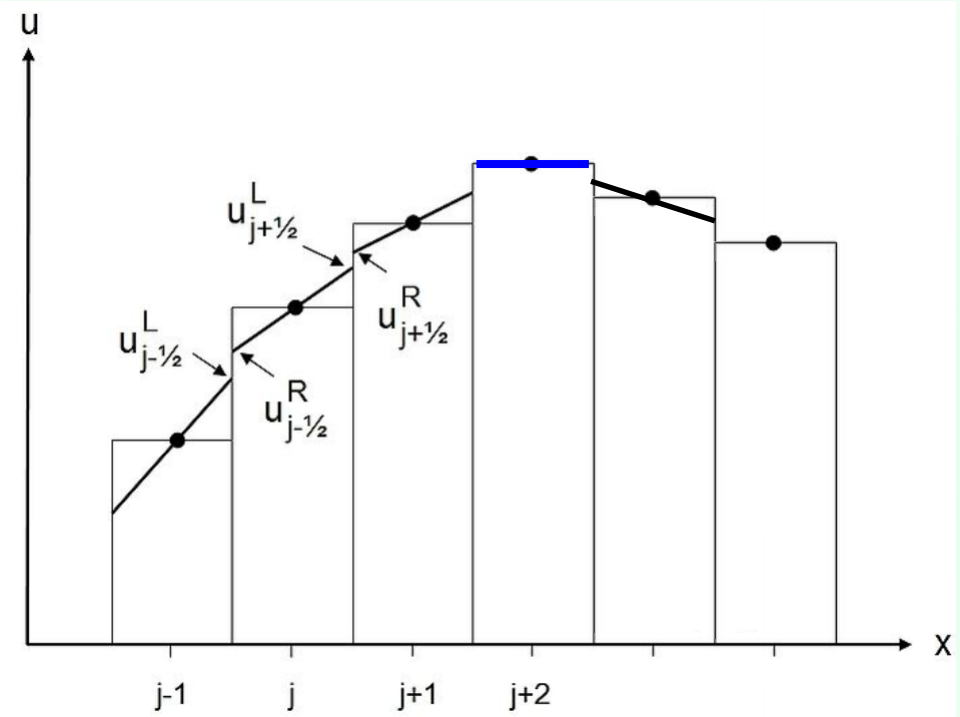
$$u_{i+1/2}^L = u_i^n(x_i + \Delta x/2)$$

$$u_{i+1/2}^R = u_{i+1}^n(x_{i+1} - \Delta x/2)$$

この二つを適当に用いて$F_{i+1/2}^n$を評価する。

Slope (Flux) Limiter





Case1: Discontinuity

Case2: Extremum

中心差分は$u^L_{i+1/2}$と$u^R_{i+1/2}$の間に逆転を引き起こすことがあり、単調性を保てない。

そのため、slope limiterを導入して単調性を保つ勾配を計算する。例えば $\frac{u^n_{i+1}-u^n_i}{\Delta x}$ と $\frac{u^n_i-u^n_{i-1}}{\Delta x}$ を比較し、逆符号なら$\delta_i = 0$、同符号なら絶対値の小さい方を採用するminmod limiterがある。

詳しくはhttps://en.wikipedia.org/wiki/Flux_limiter を参照。

磁気流体力学の数値計算法の基礎

MHDの基礎方程式

$$\frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{v}) = 0$$ mass conservation

$$\frac{\partial \rho \boldsymbol{v}}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{v}\boldsymbol{v} - \boldsymbol{B}\boldsymbol{B} + P_{\mathrm{tot}}) = \boldsymbol{0}$$ equation of motion

$$\frac{\partial E}{\partial t} + \boldsymbol{\nabla} \cdot [(E + P_{\mathrm{tot}})\boldsymbol{v} - \boldsymbol{B}(\boldsymbol{B} \cdot \boldsymbol{v})] = 0$$ energy equation

$$P_{\mathrm{tot}} = P + \frac{B^2}{2}$$ total pressure

$$E = \frac{P}{\gamma - 1} + \frac{1}{2}\rho v^2 + \frac{B^2}{2}$$ total energy

$$\frac{\partial \boldsymbol{B}}{\partial t} - \boldsymbol{\nabla} \times (\boldsymbol{v} \times \boldsymbol{B}) = \boldsymbol{0}$$ induction equation

注：天文学ではCGS-Gauss単位系を使うのが通例ですが、多くのシミュレーションコードでは $\frac{B^2}{4\pi} \rightarrow B^2$ とした計算単位系を用います。

磁場の誘導方程式

$$\frac{\partial \boldsymbol{B}}{\partial t} - \nabla \times (\boldsymbol{v} \times \boldsymbol{B}) = 0$$

この方程式が磁場の発展を決める。どうやって解けばいいか？

1. 式変形して保存形に変形し、有限体積法を適用する:

$$\frac{\partial \boldsymbol{B}}{\partial t} + \nabla \cdot (\boldsymbol{vB} - \boldsymbol{Bv}) = 0$$

この形であればMHD版リーマンソルバを適用することができる。
この場合**B**は各セル内の体積平均と解釈される。

2. 各セル「表面」で**B**を定義し、面積分とStokesの定理を適用する:

$$\frac{\partial}{\partial t} \int \boldsymbol{B} dS - \oint (\boldsymbol{v} \times \boldsymbol{B}) \cdot d\boldsymbol{l} = 0$$

$$\frac{\boldsymbol{B}^{n+1}_{x,k,j,i+1/2} - \boldsymbol{B}^{n}_{x,k,j,i+1/2}}{\Delta t} \Delta S = \sum \boldsymbol{\varepsilon} \cdot \Delta \boldsymbol{l}$$

$\boldsymbol{\varepsilon} = \boldsymbol{v} \times \boldsymbol{B}$: 起電力（Electromotive force, EMF）

$\nabla \cdot \boldsymbol{B} = 0$の条件

$\nabla \cdot \boldsymbol{B} = 0$: 磁場の発散はゼロ（モノポールの禁止）

これは拘束条件であって時間発展の式ではない。現実の世界ではこれは初期条件で成立していればその後も成立する

$$\frac{\partial(\nabla \cdot \boldsymbol{B})}{\partial t} = \nabla \cdot [\nabla \times (\boldsymbol{v} \times \boldsymbol{B})] = 0, \qquad (\leftarrow \ \nabla \cdot [\nabla \times A] = 0)$$

離散化した系ではこの条件は自動的には満たされないが、もし$\nabla \cdot \boldsymbol{B}$が生成されると非物理的な力が働き、計算は破綻（爆発）する。
前頁手法1の有限体積法はこの条件を満たさない。

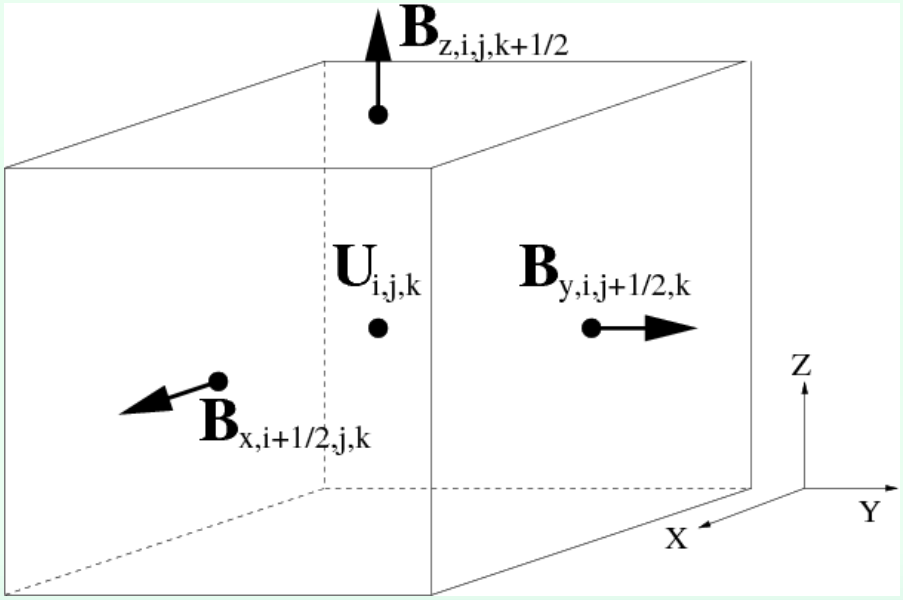
この場合は、非物理的に生成される$\nabla \cdot \boldsymbol{B}$を消去する必要がある:

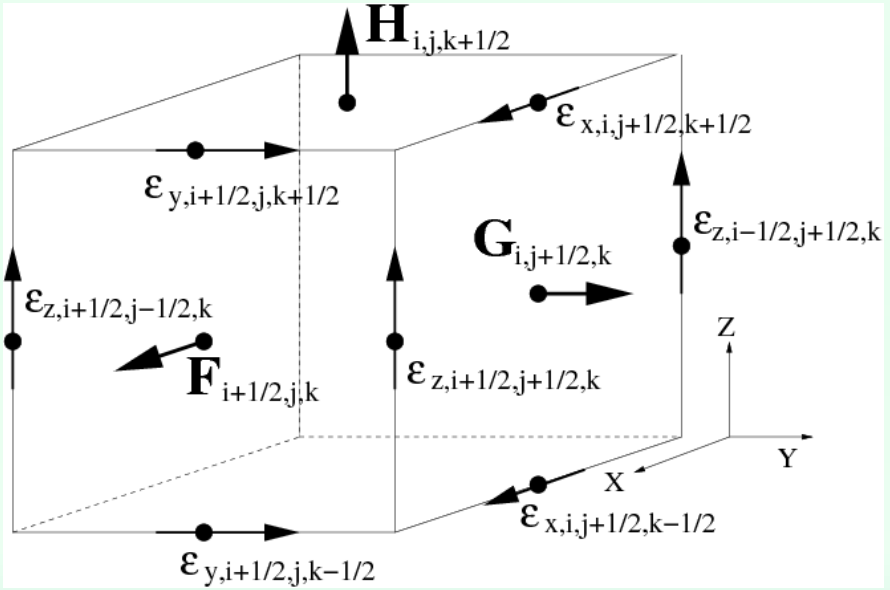
- Projection: $\nabla \cdot \boldsymbol{B}$についてのポアソン方程式を解いて消す: 高コスト
- Parabolic/hyperbolic correction: 一般化したラグランジュの未定乗数法を用い、これを新たな変数として解くことで$\nabla \cdot \boldsymbol{B}$を消去する。実装が容易、安定、高効率だが、正しいかは？ (Dedner et al. 2002)

(Figures: Jim Stone's lecture note)

Constrained Transport

一方、手法2の"有限面積法"は適切に離散化すればこの条件を自動的に満たすことができる→ CT法(Ewans & Hawley 1988):



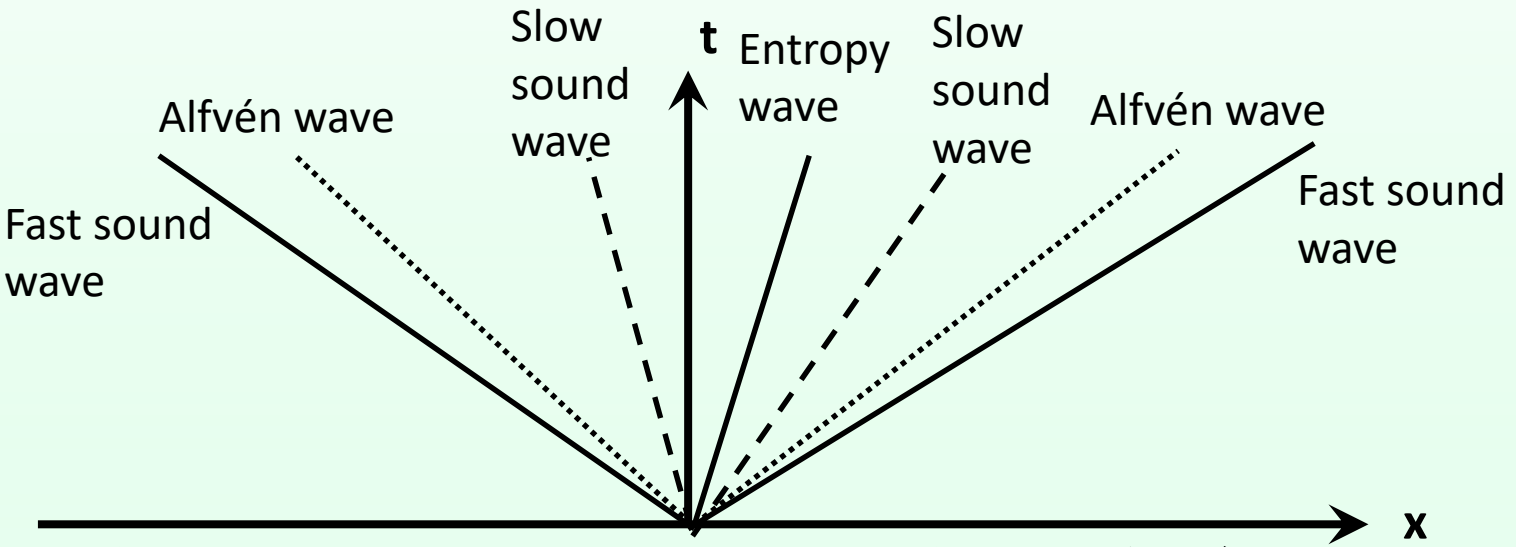


この方法では磁場はセルの表面で、EMFはセルの辺で定義される。

「有限面積法」 $\frac{\boldsymbol{B}_{x,k,j,i+1/2}^{n+1}-\boldsymbol{B}_{x,k,j,i+1/2}^{n}}{\Delta t}\Delta S=\sum \boldsymbol{\varepsilon}\cdot\Delta \boldsymbol{l}$ を使うと、

EMFの計算法によらず、うまく打ち消し合って$\nabla\cdot\boldsymbol{B}=0$が満たされる。

MHD波



MHDでは流体の場合と異なり7種の波（固有モード）がある

Alfvén 速度: $v_A = \frac{B}{\sqrt{4\pi\rho}}$

Fast/slow magnetosonic speeds (θ=**B** と **k**の角度):

$$v_{\pm} = \sqrt{\frac{1}{2}\left[V_A^2 + c_s^2 \pm \sqrt{(V_A^2 + c_s^2)^2 - 4V_A^2 c_s^2 \cos^2\theta}\right]}$$

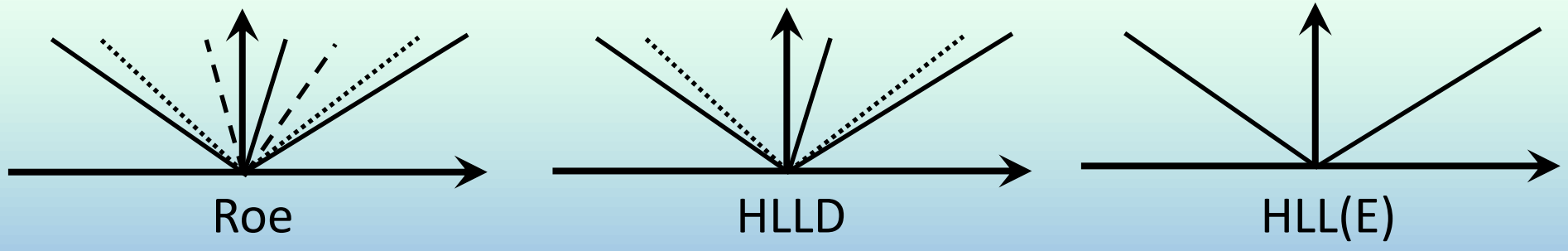
注：MHDではこれらの速度が縮退することがあり、厳密に双曲型ではない

MHDのリーマンソルバ

CT法で誘導方程式を解いても流体を解くのにリーマンソルバが必要

- (Exact solver: 非常に複雑であまり現実的ではない)
- Roe's solver: 線形化した波を解く: あまり安定性が高くない
- HLL/HLLE: 速い磁気音波だけを解く – 安定だが拡散が強い
- (HLLC: 音波とエントロピー波を解く – メリットが少ない)
- HLLD (Miyoshi & Kusano 2005): 速い音波、Alfven波、エントロピー波を解く – 安定かつ高速・高精度で、現代の事実上標準の解法

HLLDを第一選択として、不安定になった時にはHLLEを試すのが良い。



補足 : 保存量とprimitive変数

$$\boldsymbol{U} = \begin{pmatrix} \rho \\ \rho v_x \\ \rho v_y \\ \rho v_z \\ E \end{pmatrix} \leftrightarrow \boldsymbol{W} = \begin{pmatrix} \rho \\ v_x \\ v_y \\ v_z \\ P \end{pmatrix}, \qquad \boldsymbol{U} = \begin{pmatrix} \rho \\ \rho v_x \\ \rho v_y \\ \rho v_z \\ B_x \\ B_y \\ B_z \\ E \end{pmatrix} \leftrightarrow \boldsymbol{W} = \begin{pmatrix} \rho \\ v_x \\ v_y \\ v_z \\ B_x \\ B_y \\ B_z \\ P \end{pmatrix}$$

流体力学を保存形で解く場合でも、解析や操作によってはprimitive変数と呼ばれる非保存量を使う方が便利なことがあり、Athena++をはじめ多くの流体コードではこれらを使い分けている。例えばPLM法の補間に保存量を使うと圧力が負になる可能性があるため使えない。衝撃波管問題ではprimitive変数を出力していた。

この他に、characteristic変数と呼ばれる固有モードに対応する変数の組があり、高次精度化の補間にはこれを使うのが高コストだが精度が出ることが知られている。（Brio-Wuの衝撃波管問題ではこれを使用するように設定してある）

良いシミュレーションとは

ここまで体験してもらったように、シミュレーションコードを使えばそれらしい答えは割と簡単に出る。しかし、結果が出てもそれが正しいという保証はないし、公開コードを使っていても結果が正しいことを証明するのはユーザーの責任である。

では、シミュレーション結果が正しいことはどう証明すれば良いだろうか？

- 少なくとも理想化された状況について解析解を再現することを確認する

→ 解析解がない複雑な問題も多々ある（解がわかっているならそもそもシミュレーションは不要）
→ Kelvin Helmholtz不安定性等の単純な問題なら線形解析で分散関係がわかる

- 分解能を上げた時に同じ結果が得られること（＝収束性）を確認する → 次頁
- 解が満たすべき物理的性質を持っていることを確認する

→ 対称性 - ただしシミュレーションでは数値誤差のため対称性が破れることがある
→ （非自明な）保存量 - 保存形では$\rho, \rho\boldsymbol{v}, E$は自明に保存するのでそれ以外

- 解の振る舞いが少なくとも定性的に物理に基づいて理解できる

→ どの物理過程が支配的か？
→ 既知の現象（不安定性・安定性）との関係？

⇒ Enterキーを押して出てきた結果を垂れ流すのは良いシミュレーションではない。

収束性

前々回までにやった通り、適切なアルゴリズムを使えば、分解能を上げれば離散化した系は元の方程式の正しい近似解に十分近づくことが期待される。

では、「十分近い」ことはどうやって確かめる？

- 数値的に誤差を評価してそれが十分（例えば0.001%以下）小さいことを示す

→ 厳密だが、宇宙物理では衝撃波が多数現れ、不連続面では一次精度しか出ない（←Godunovの定理）ため、この意味での誤差を十分小さくするのは難しい。

- 興味ある現象の物理的なスケールを十分分解する

→ 重力ならジーンズ波長、放射冷却なら冷却長、線形不安定なら不安定波長
→ 乱流等、物理過程によっては無限に小さい構造を生じ得る

- 分解能に対して解の振舞を調べ、その影響を予測する（cf. リチャードソン補外）
- 少なくとも興味ある物理量が分解能に大きく依存しないことを確認する

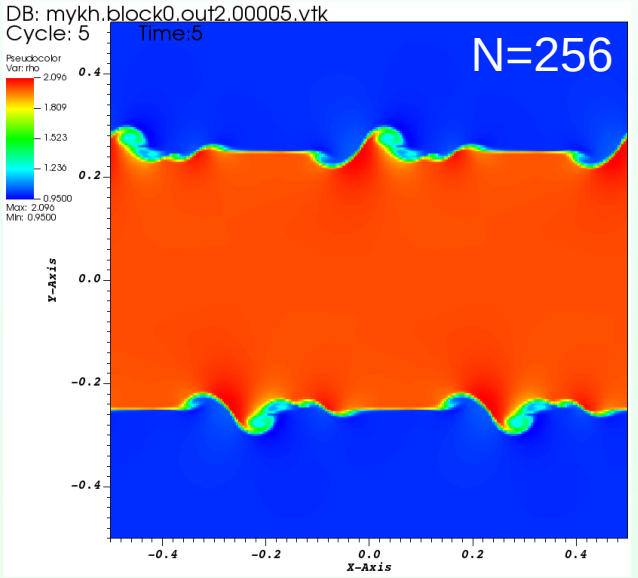
→ 小スケールの乱流は分解できなくとも平均的・統計的性質は収束することがある
→ 小さい構造を全て分解することが系の理解に必要とは限らない

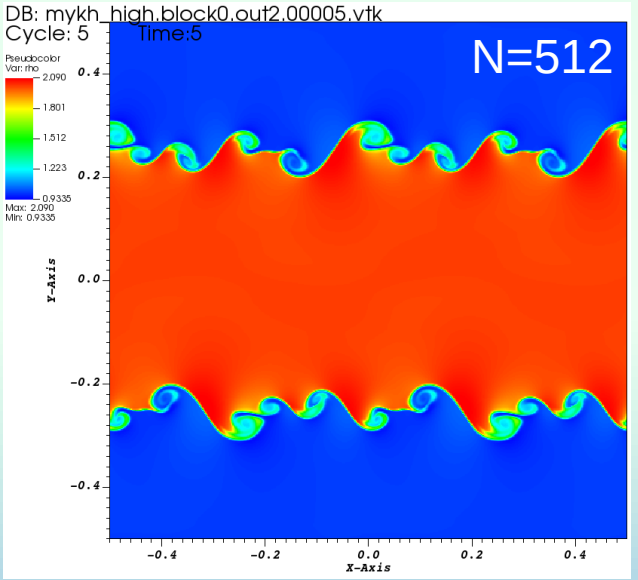
⇒ 良いシミュレーションには対象の深い物理的理解が不可欠。

収束性に関する注意

系によっては分解能を上げるとより細かい構造が発生して、厳密な意味での収束性がどこまで行っても実現しない場合がある。

- 無限小の波長が不安定な場合
  例：境界層の厚みがゼロのKH不安定（→右図）
- 粘性なしの乱流
- 熱伝導なし・温度ゼロまで到達する冷却不安定





このような系で厳密な意味での収束を実現するには何らかの小スケールの構造を抑制する（可能なら物理的な）機構が存在し、その機構によって決まる最小スケールを分解する必要がある。

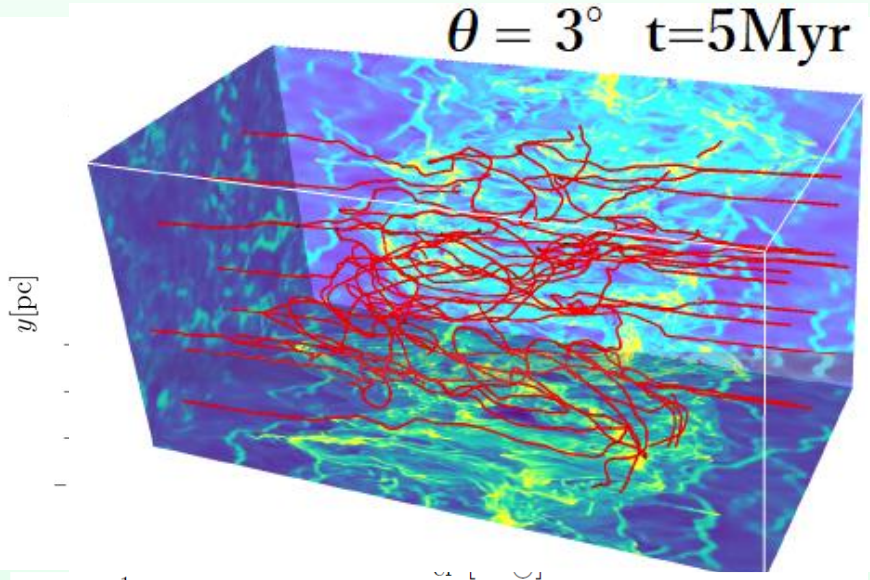
- KH → 有限厚さの境界層では短波長は安定化する
- 乱流 → 粘性を入れて小スケールの渦を抑制する
- 冷却 → 熱伝導を入れて小スケールを安定化する

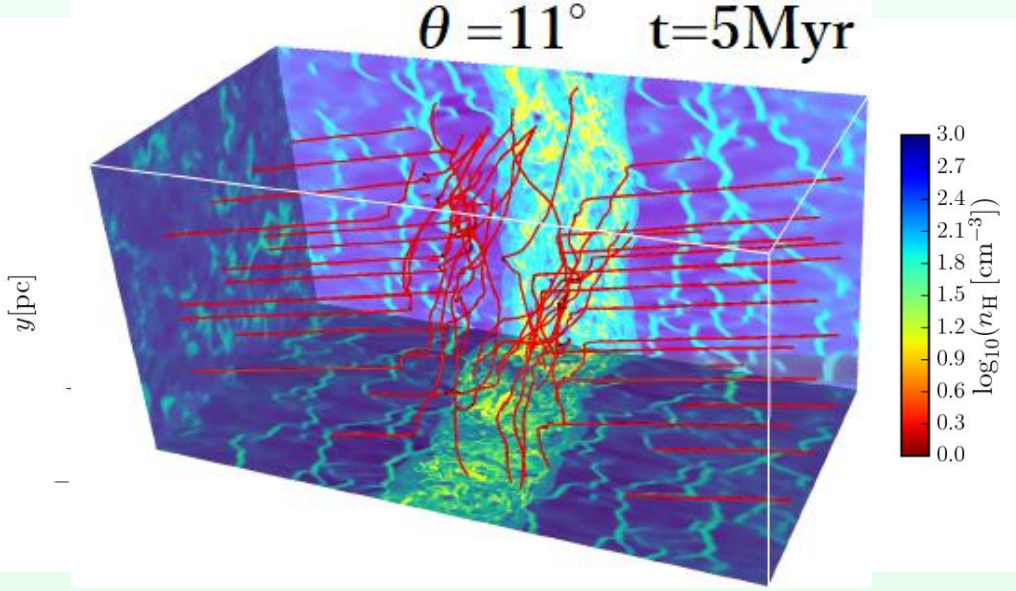
⇒ただし宇宙ではこれらの物理過程が効くスケールが非常に小さいため、「本当に系と同じスケールを分解したシミュレーション」は困難

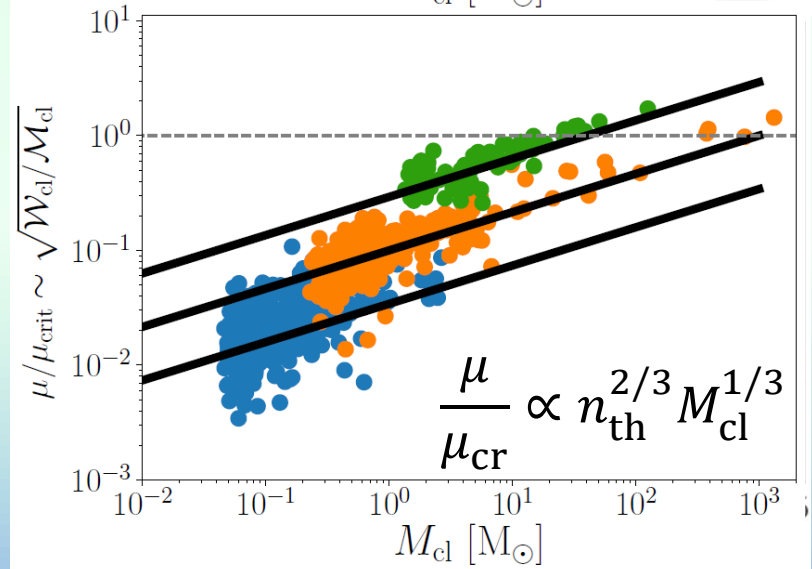
Athena++による
シミュレーション例の紹介

(Iwasaki et al. 2019)

分子雲形成





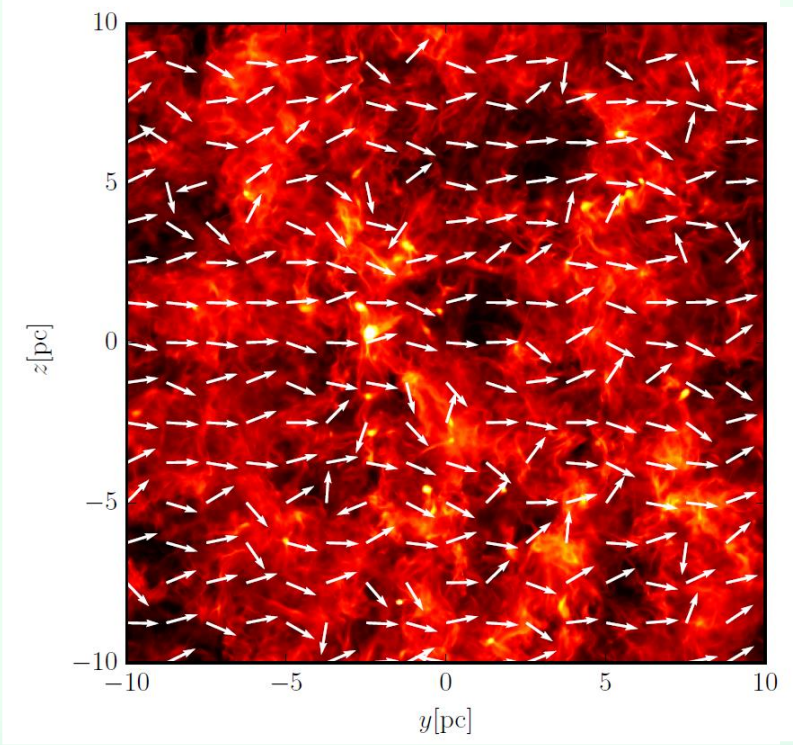


衝突ガス流（超新星爆発・渦状腕・銀河合体などによる星間ガス圧縮のモデル）による分子雲形成過程。
磁場と圧縮流の方向によって分子雲形成効率がどう依存するかを調べた。
← 最近は更に高解像度の計算で、星形成過程の初期条件となる分子雲コアの質量・磁束分布を調べている。

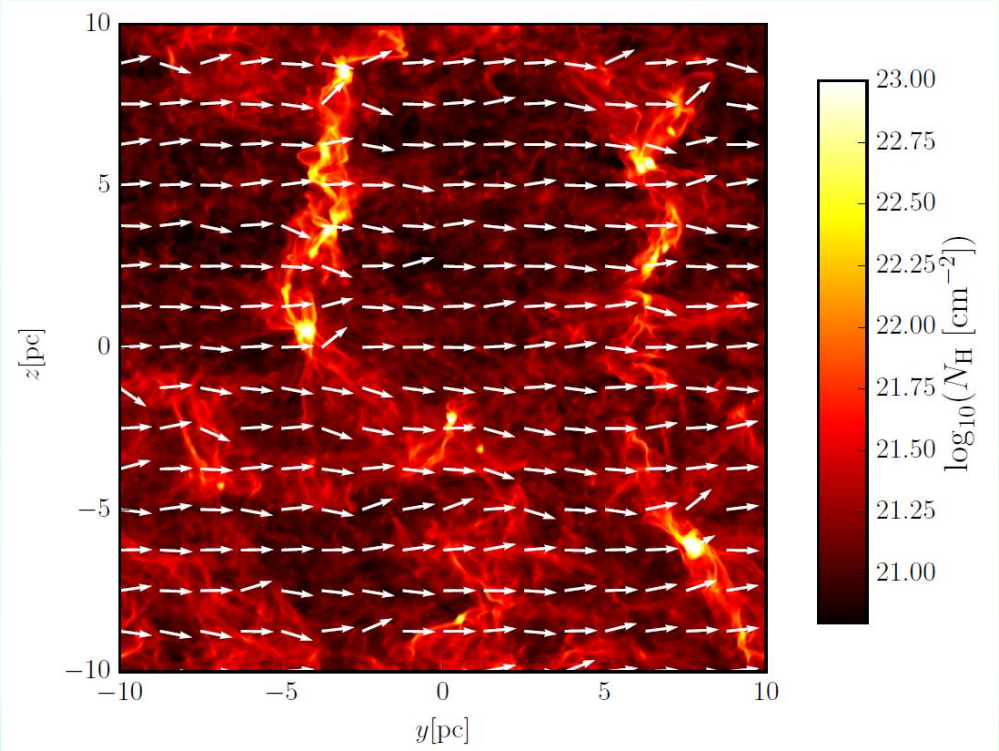
(Iwasaki et al. in prep.)

分子雲コア・フィラメント形成

θ=3°　t=10Myr



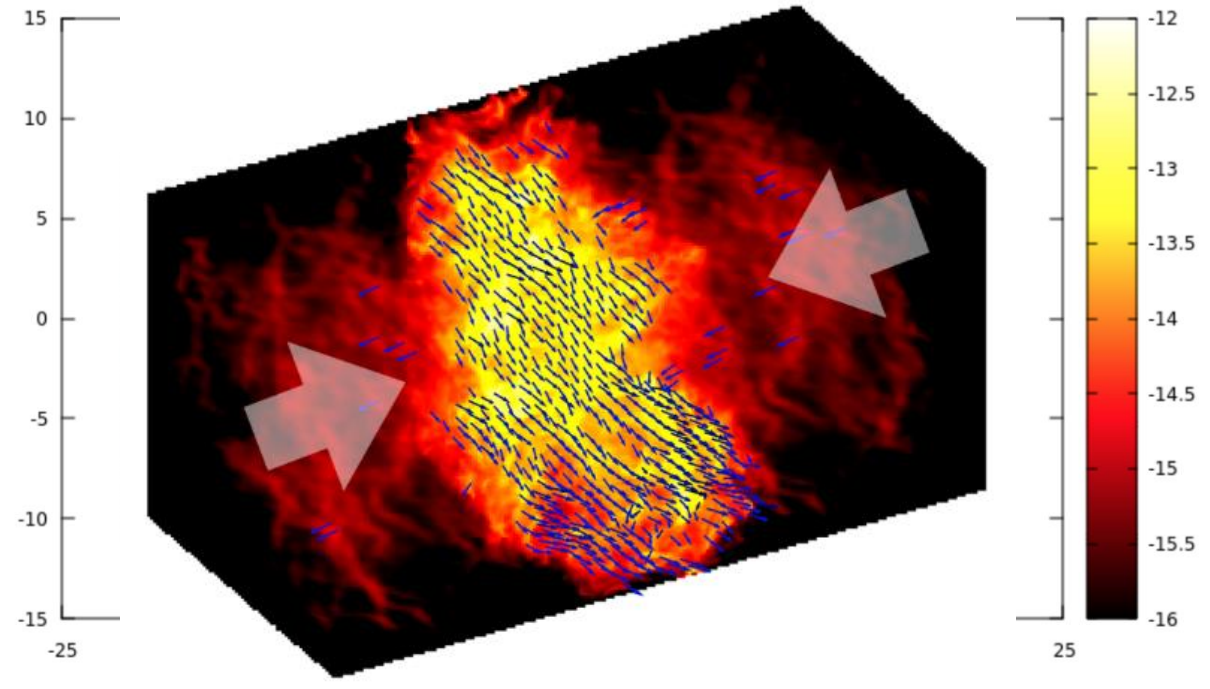
θ=11°　t=7.5Myr

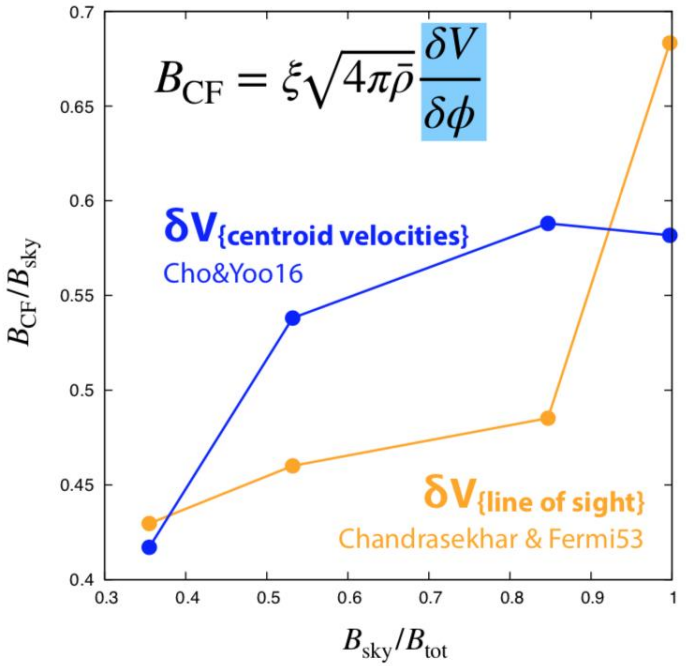


自己重力なし（左）とあり（右）の場合の面密度分布
左: super-Alfvenic乱流がcoherentな構造形成を阻害→コア・クランプ的な構造
右: 事故呪力によって磁力線に直交する方向に伸びた高密度フィラメントが形成
→ 自己重力による分子雲内の構造形成の第一原理的計算

(Tanaka et al. in prep.)

Synthetic Observation

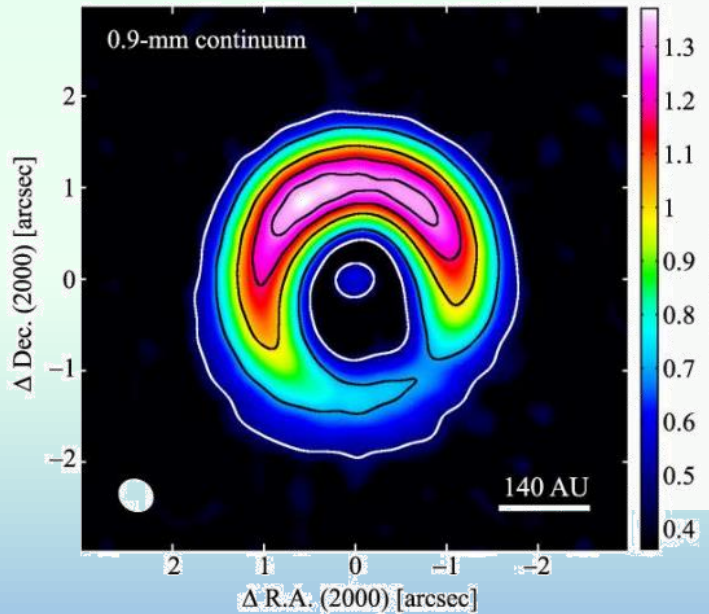


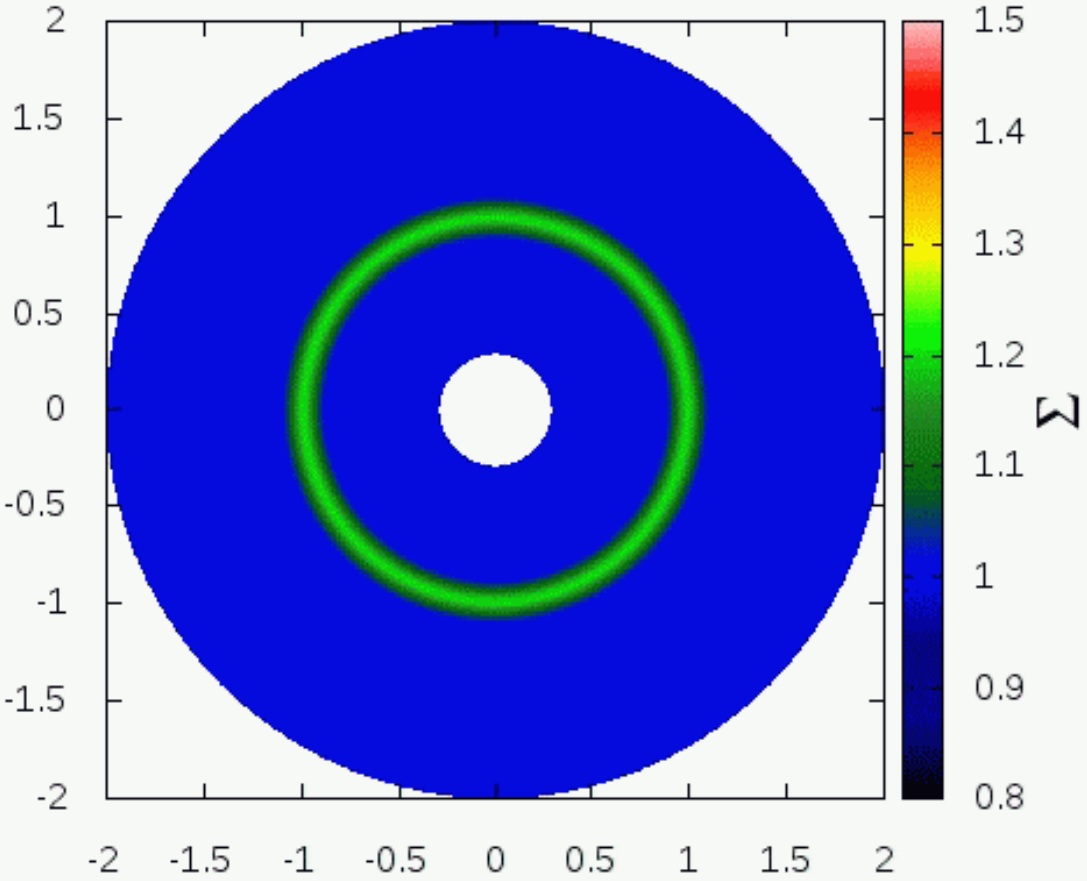


国立天文台ALMA共同科学研究事業
分子雲形成シミュレーションを後処理輻射輸送計算で可視化
直線偏光を含むダスト連続波 & CO分子輝線
PlanckやJCMT-BISTRO による分子雲偏光マップ観測との比較や
Chandrasekhar-Fermi法などの磁場推定法の精度検証に利用可能

ロスビー波不安定性によるΣ渦生成

Ono, Muto, Tomida and Zhu, 2018, ApJ, 864, 70
圧力バンプにおけるシアの非線形発展を高解像度2次元流体シミュレーションで調べた





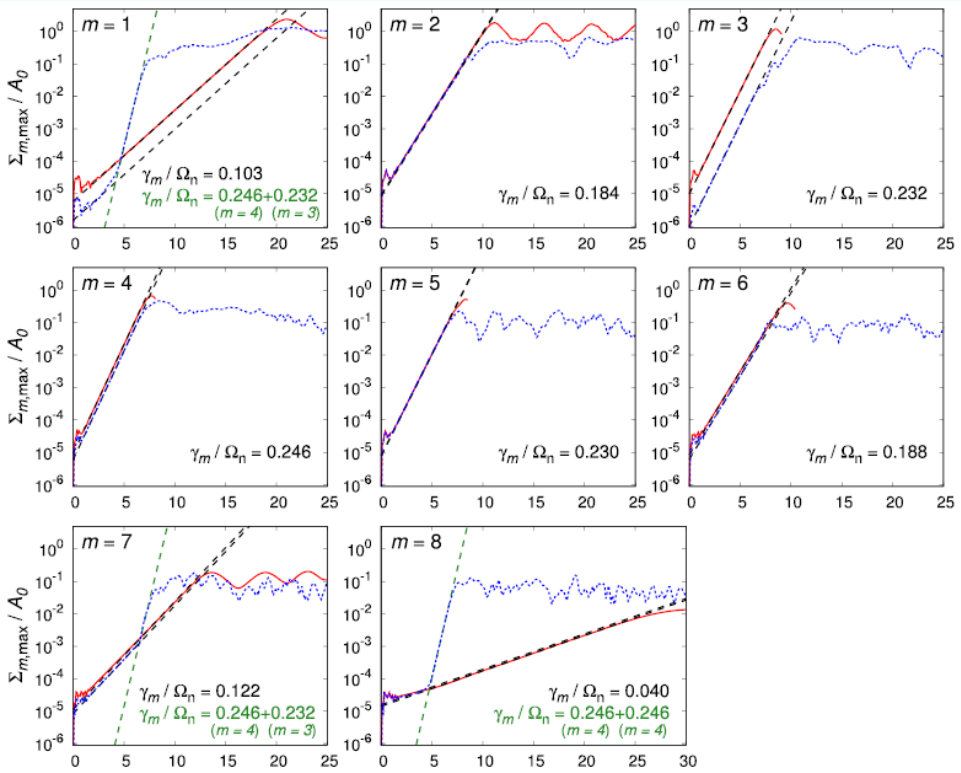
HD142527

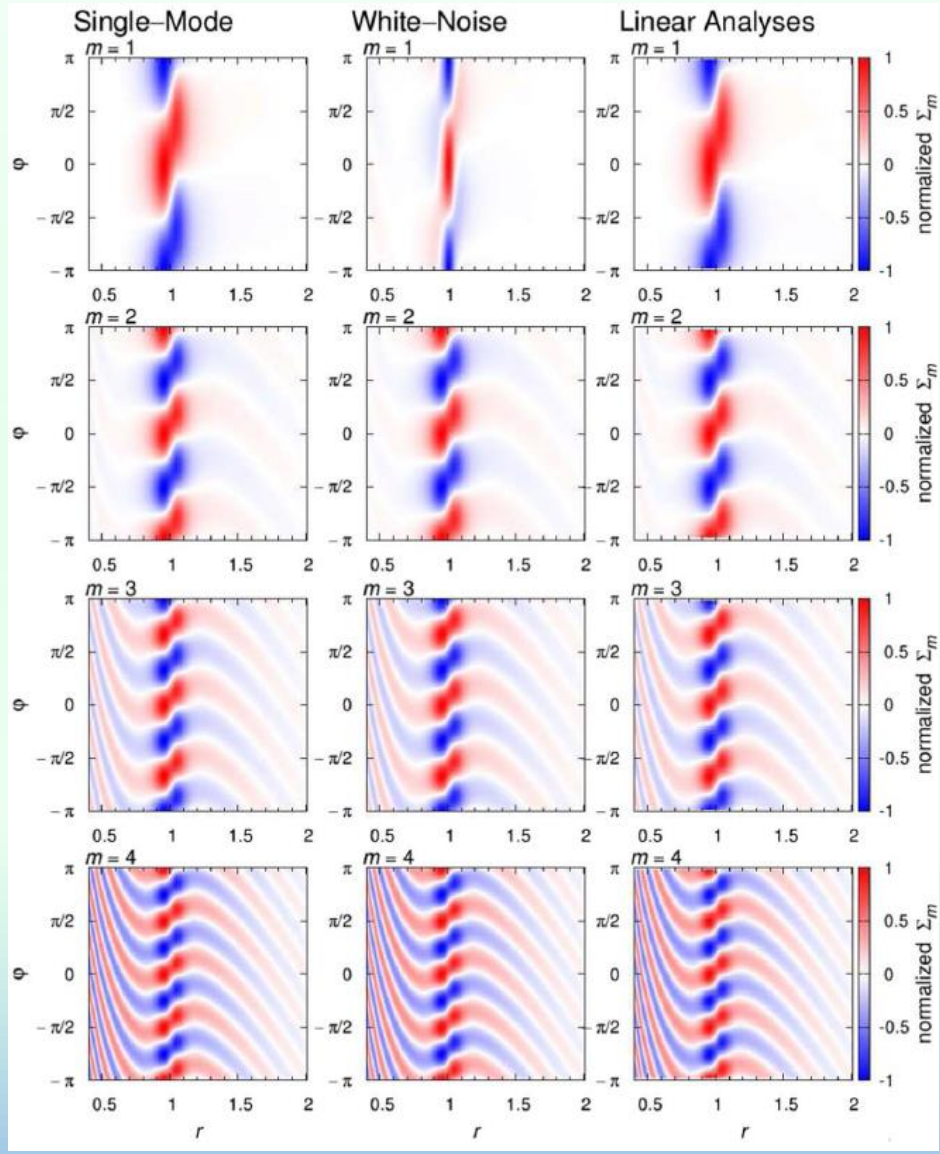
Fukagawa+13

観測に似た三日月構造が形成
ただし観測はダスト、シミュレーションはガスであることに注意

Ono et al., 2018, ApJ, 864, 70

RWIの理論とシミュレーションの比較

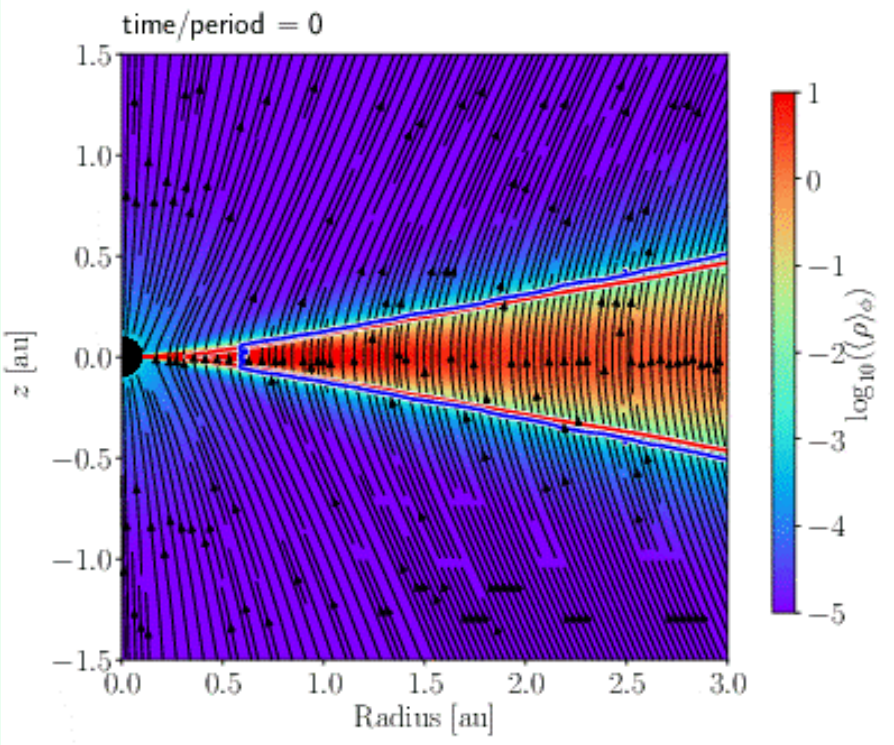


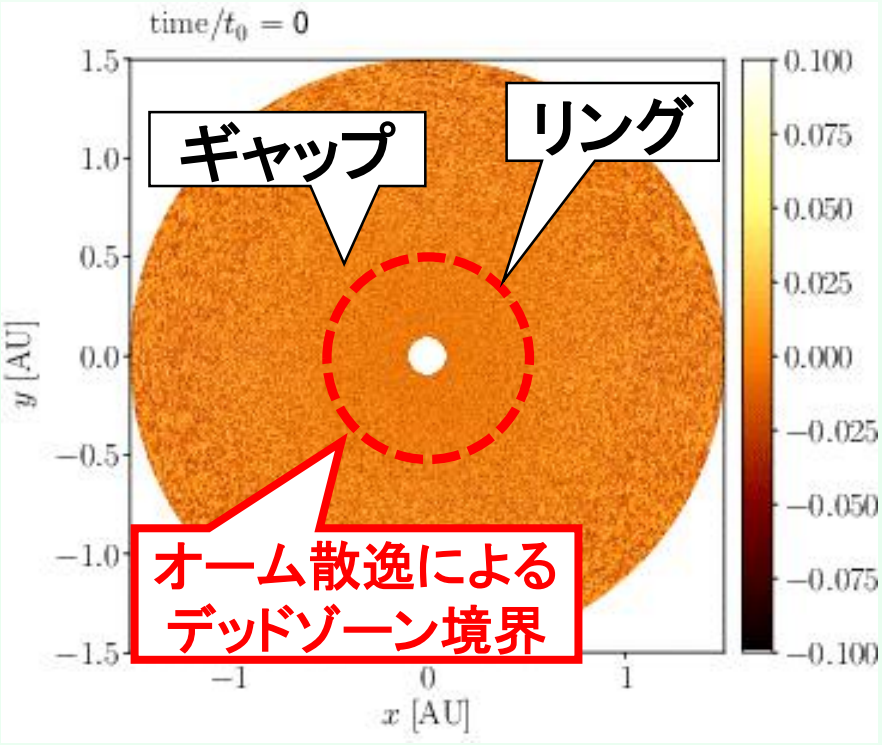


Athena++の計算は線形理論による
モード解析、及び線形～弱非線形理論
による成長率の予測と良く一致
→ 成長はモード結合によって飽和
シミュレーションの非常に良いテスト

(Iwasaki et al. in prep.)

原始惑星系円盤の大局計算

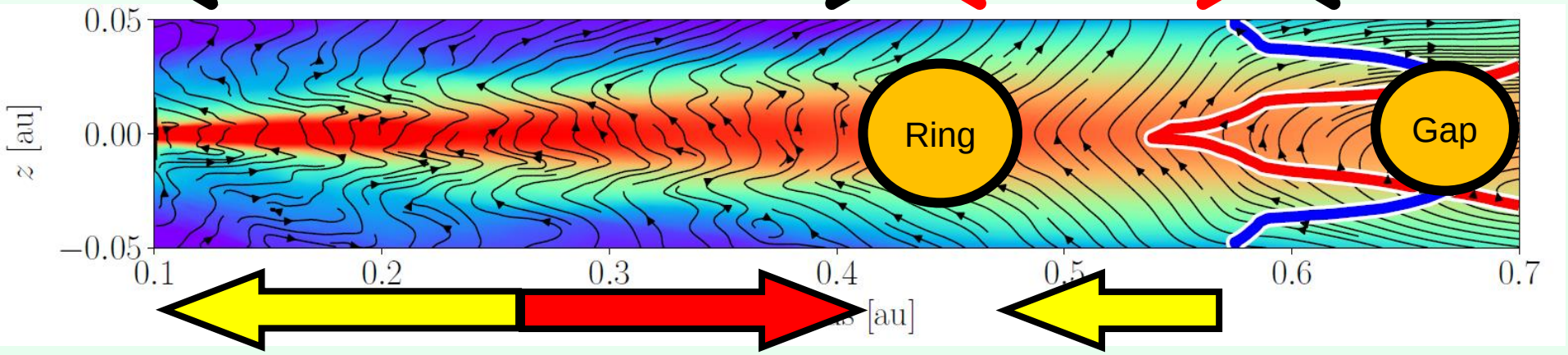




3次元大局的非理想磁気流体(オーム散逸＋両極性拡散)シミュレーション
複雑な構造形成：乱流領域、デッドゾーン、ウィンド、リング・ギャップ・・・
- 非一様・非線形な非理想磁気流体効果によるもの？境界条件？
- ロスビー波不安定が二次的な三日月状構造を作り出す

注：スケールが違うのでHL Tau等のリングと直接比較してはいけない

リング-ギャップ構造の成因



ガス降着　　外側へ移動

大域磁場による角運動量輸送

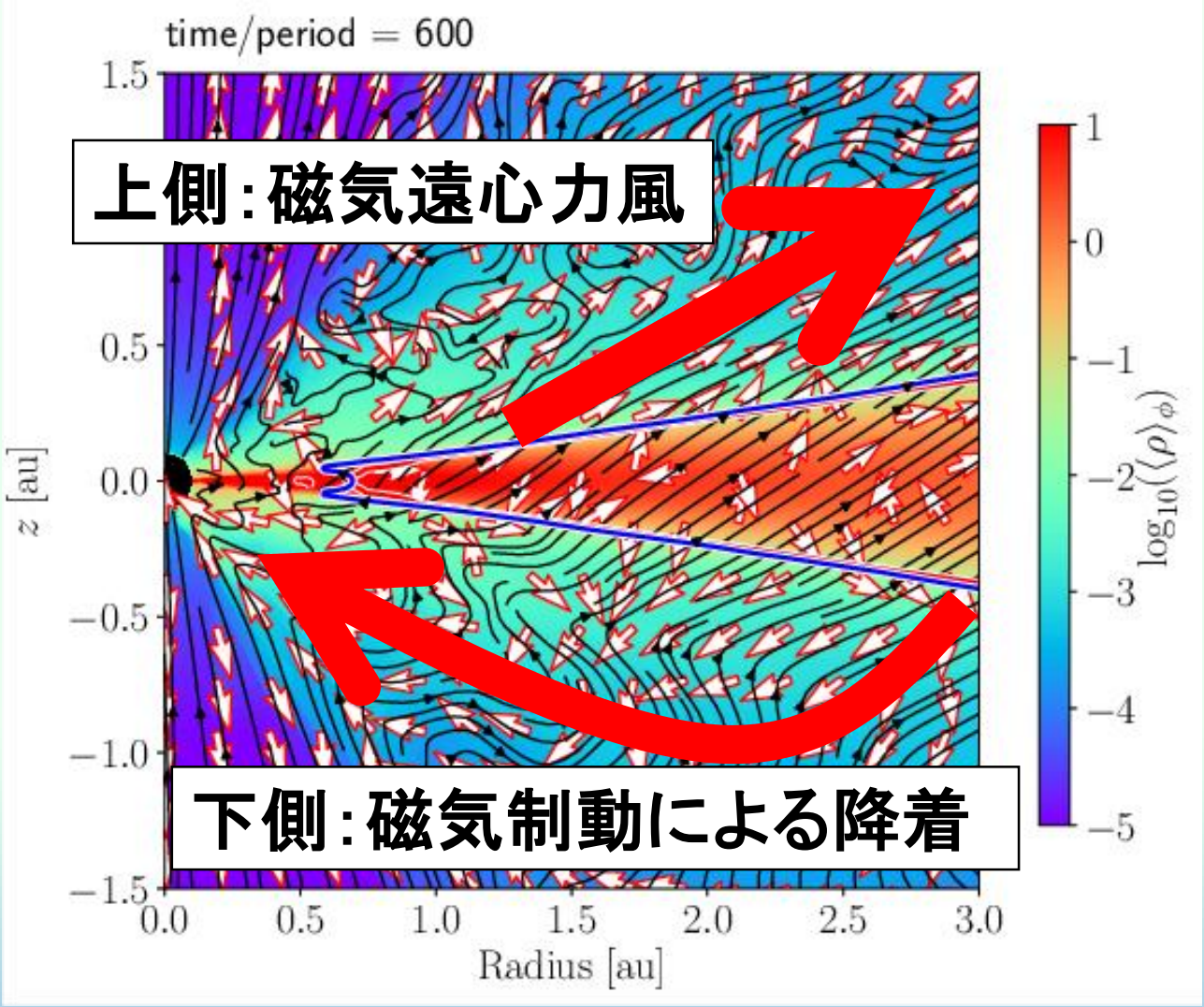
乱流粘性による角運動量輸送

中間的領域では非理想磁気流体効果で乱流は抑制されるが、まだ角運動量輸送は働いている

角運動量輸送機構・輸送効率が領域によって異なることにより

リングーギャップ構造が形成される

長時間進化 - 上下対称性の破れ



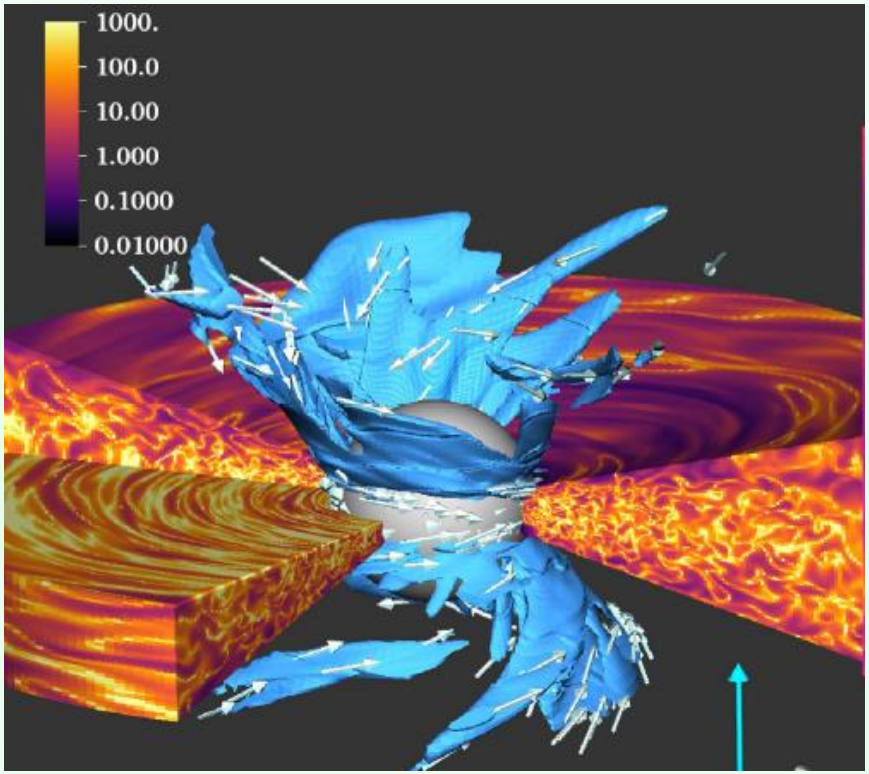
長時間計算の結果、外側のデッドゾーンで顕著な上下対称性の破れが成長

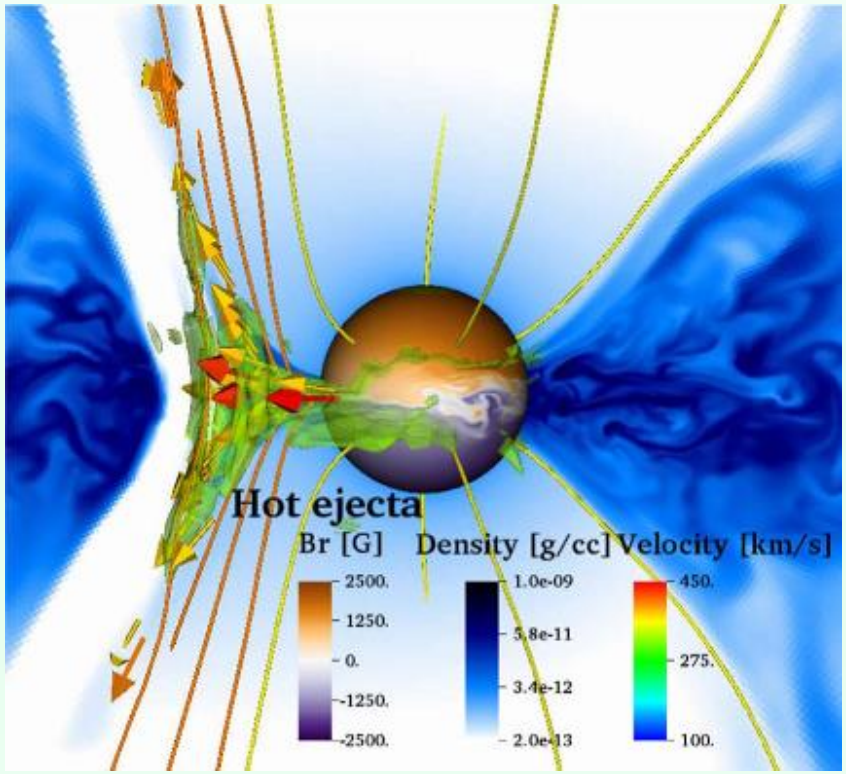
大域磁場による角運動量輸送が支配的な領域では磁場形状に上下非対称性があると、これを増幅する機構があることを発見。

→ 円盤の歪み（warp）の起源？これが円盤進化にどう影響するかは検討中

(Takasao et al. 2018, 2019)

星-円盤相互作用

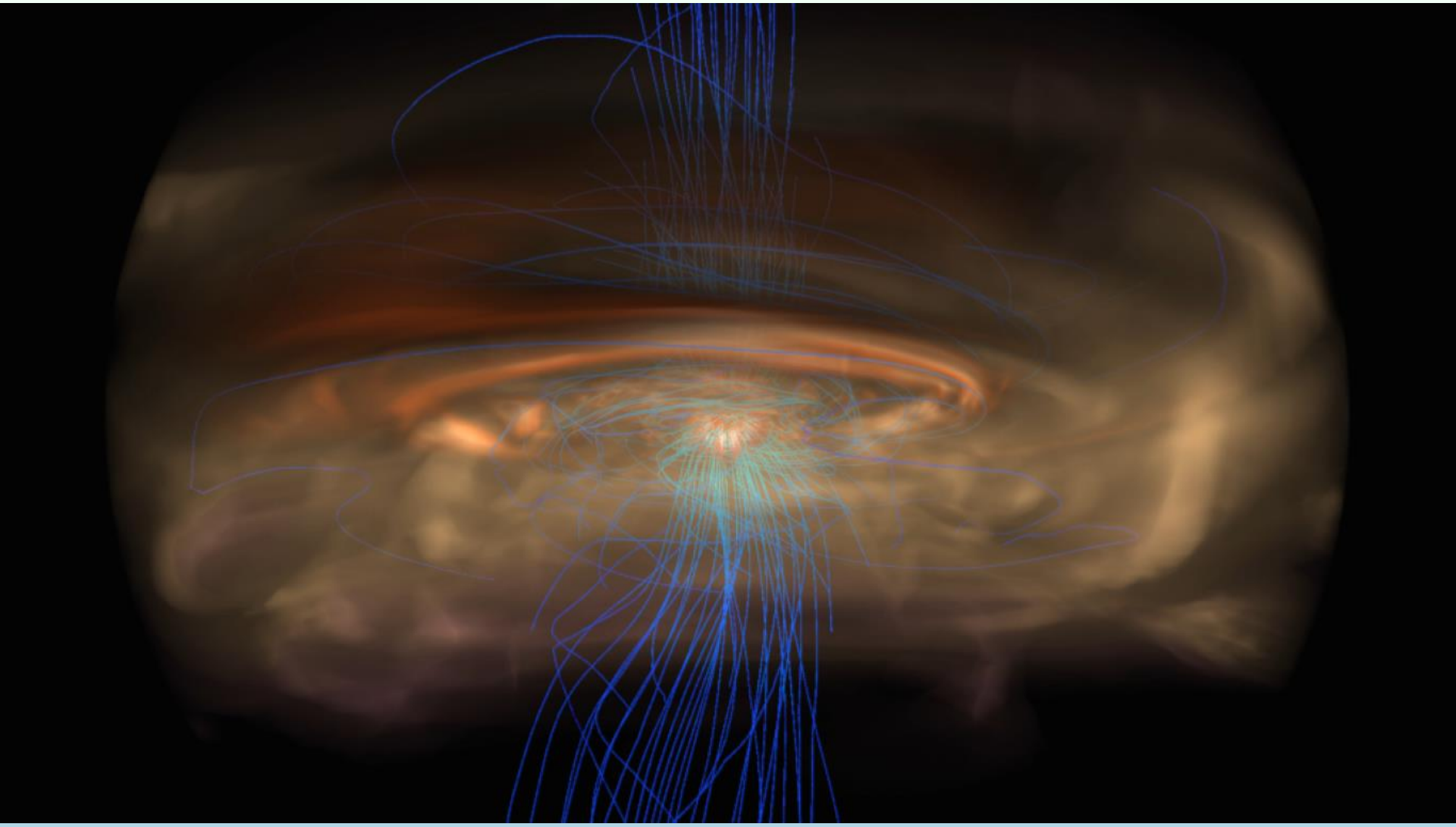




左：磁場の弱い原始星への降着流のシミュレーション。従来高緯度への降着は強い原始星磁場に沿った磁気圏降着と考えられていたが、円盤風が角運動量を失って高緯度に降着する新しい機構を提唱。

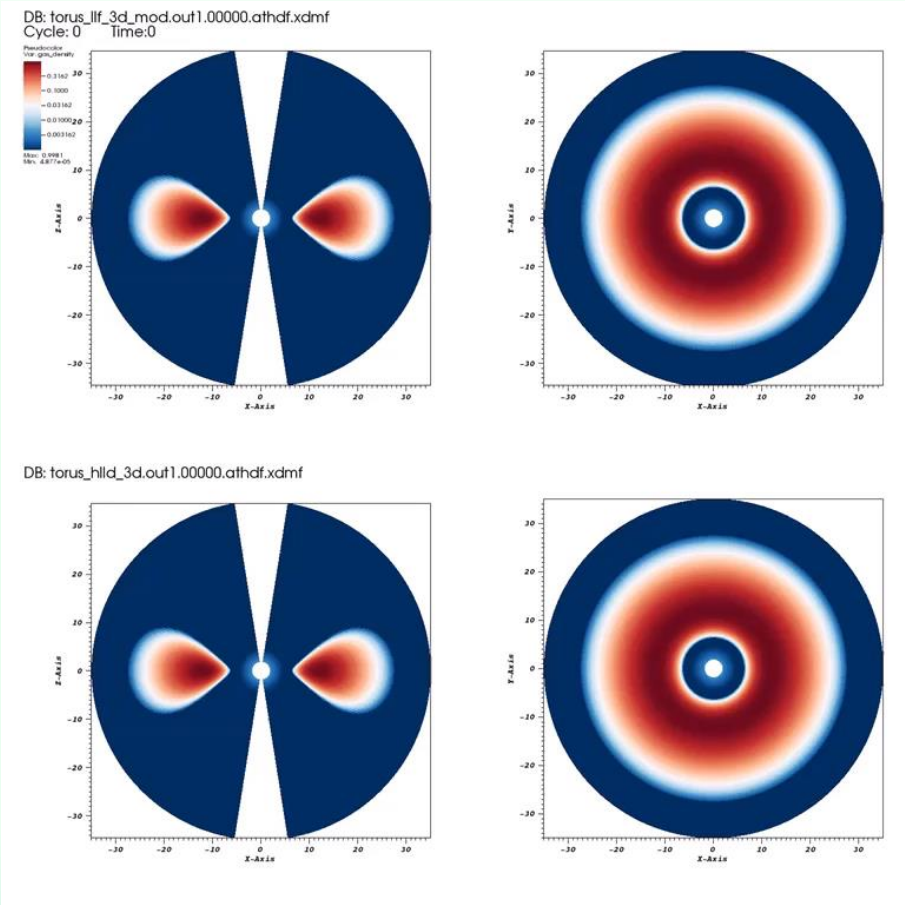
右：円盤内での大局的磁場のリコネクションによって駆動される巨大原始星フレア駆動機構の新しい理論モデル

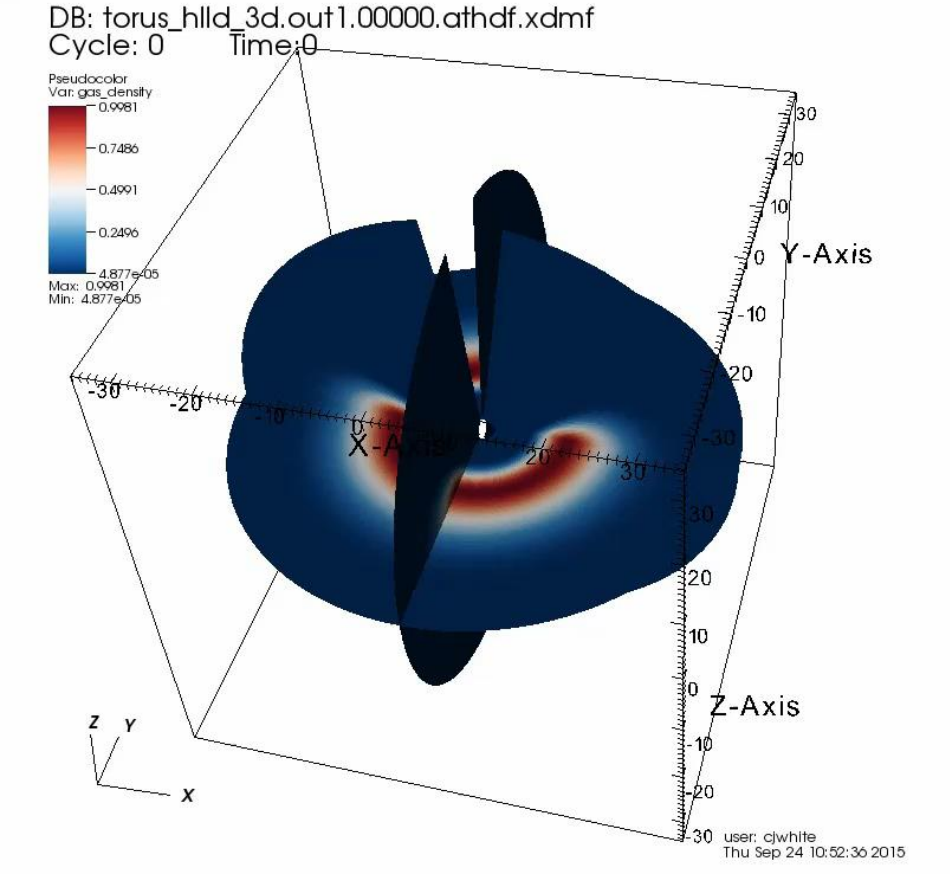
原始星フレア



(Takasao et al. 2019, visualization by T. Takeda @ Vasa Entertainment)

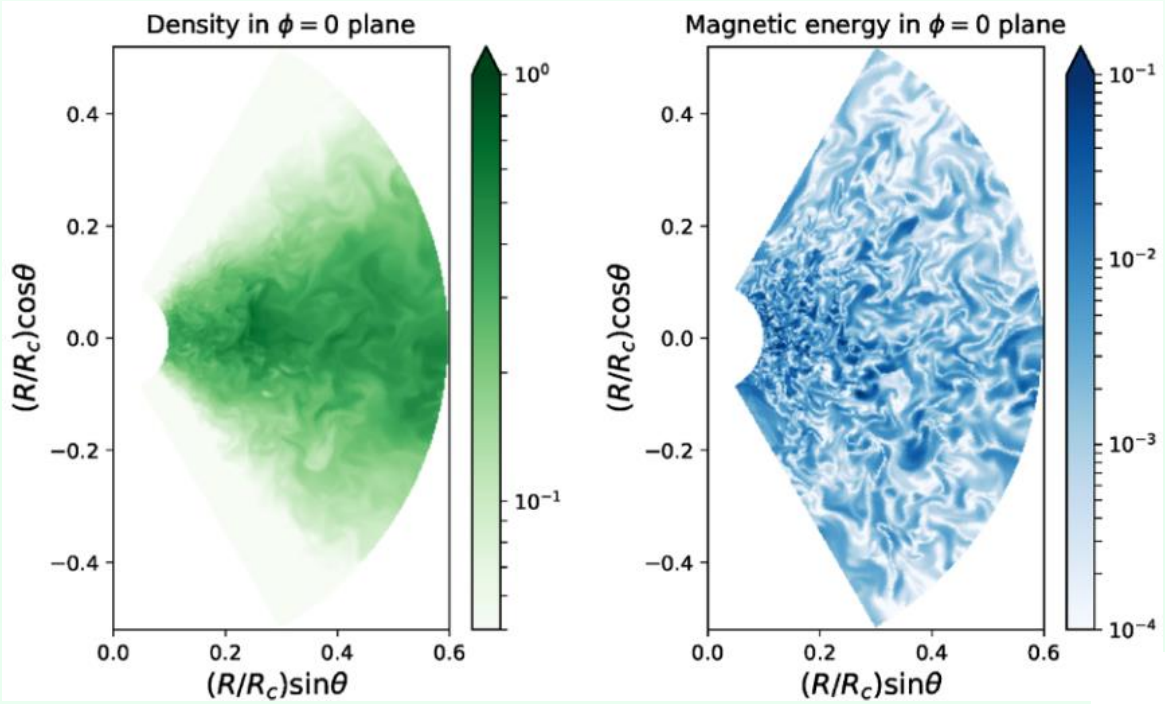
ブラックホール降着円盤

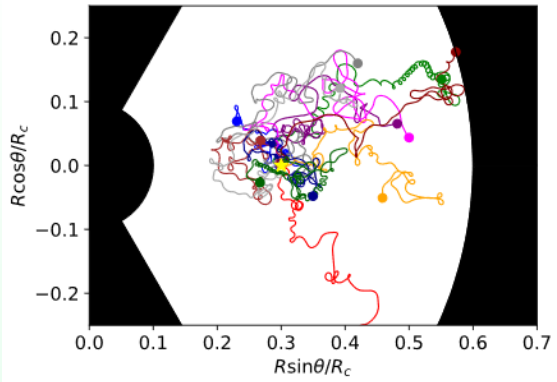


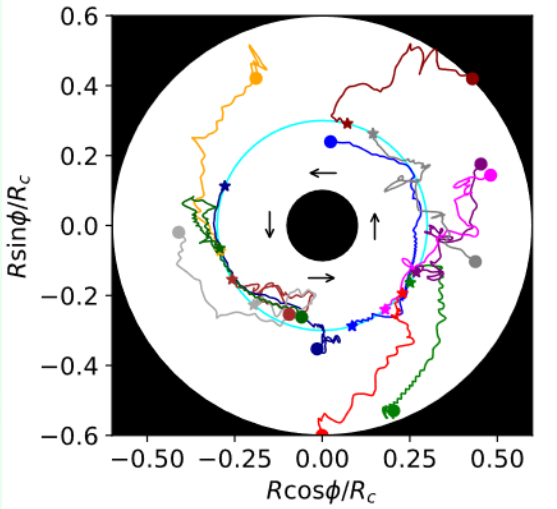


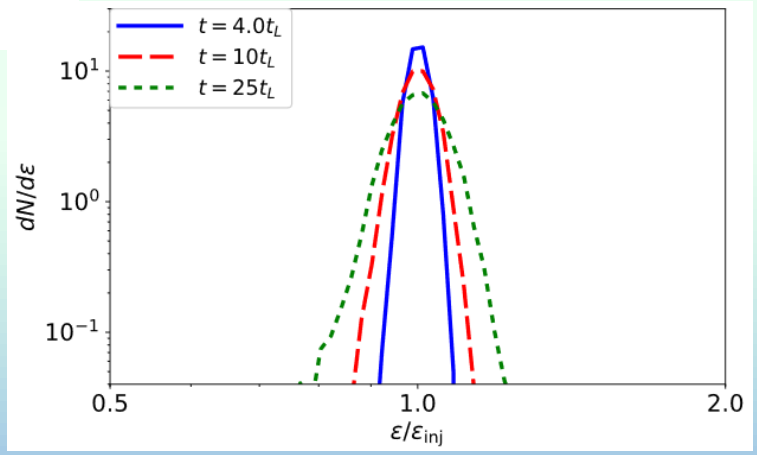
GRMHD (fixed-metric) 計算 (White, Stone & Gammie 2016)
Event Horizon Telescopeのためのシミュレーションコード比較プロジェクトにも参加（Porth et al. 2019, ApJS）

AGNでの粒子加速









(Kimura, Tomida & Murase 2019)
高エネルギー宇宙線の加速機構の一つとして乱流による統計的加速を調べた。
Athena++でMRI乱流円盤を作り、その中で高エネルギー粒子の軌道を追跡→

Mesh Refinement

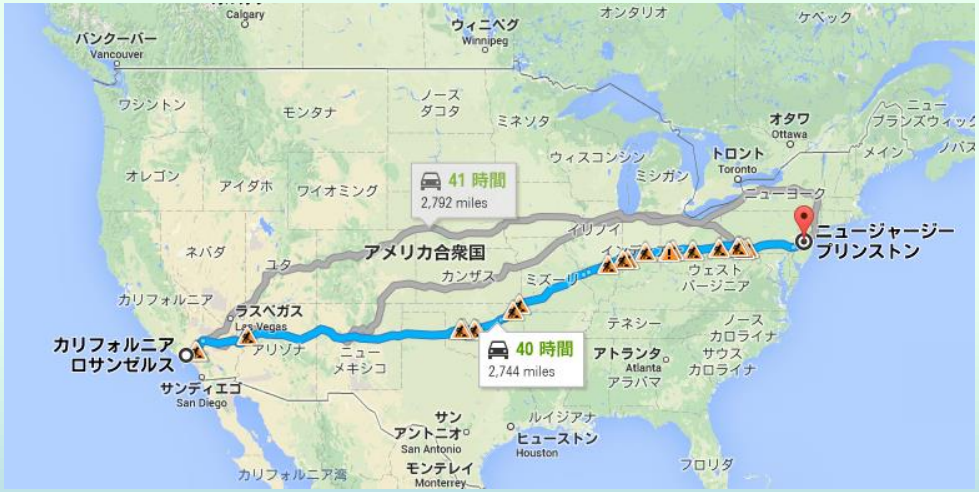
AMRの必要性

太陽半径が1mだとすると:

太陽から地球 1AU → ~ 200m
（オフィスから食堂まで、くらい）

太陽系 (heliopause)
160AU ~ 35km 仙台市くらい

分子雲コア（星形成の初期条件）
0.1pc → ~ 4,500km
アメリカ東海岸から西海岸まで



例えば星形成シミュレーションは
アメリカ全土から個人一人を特定するくらいの分解能が必要

Adaptive Mesh Refinement

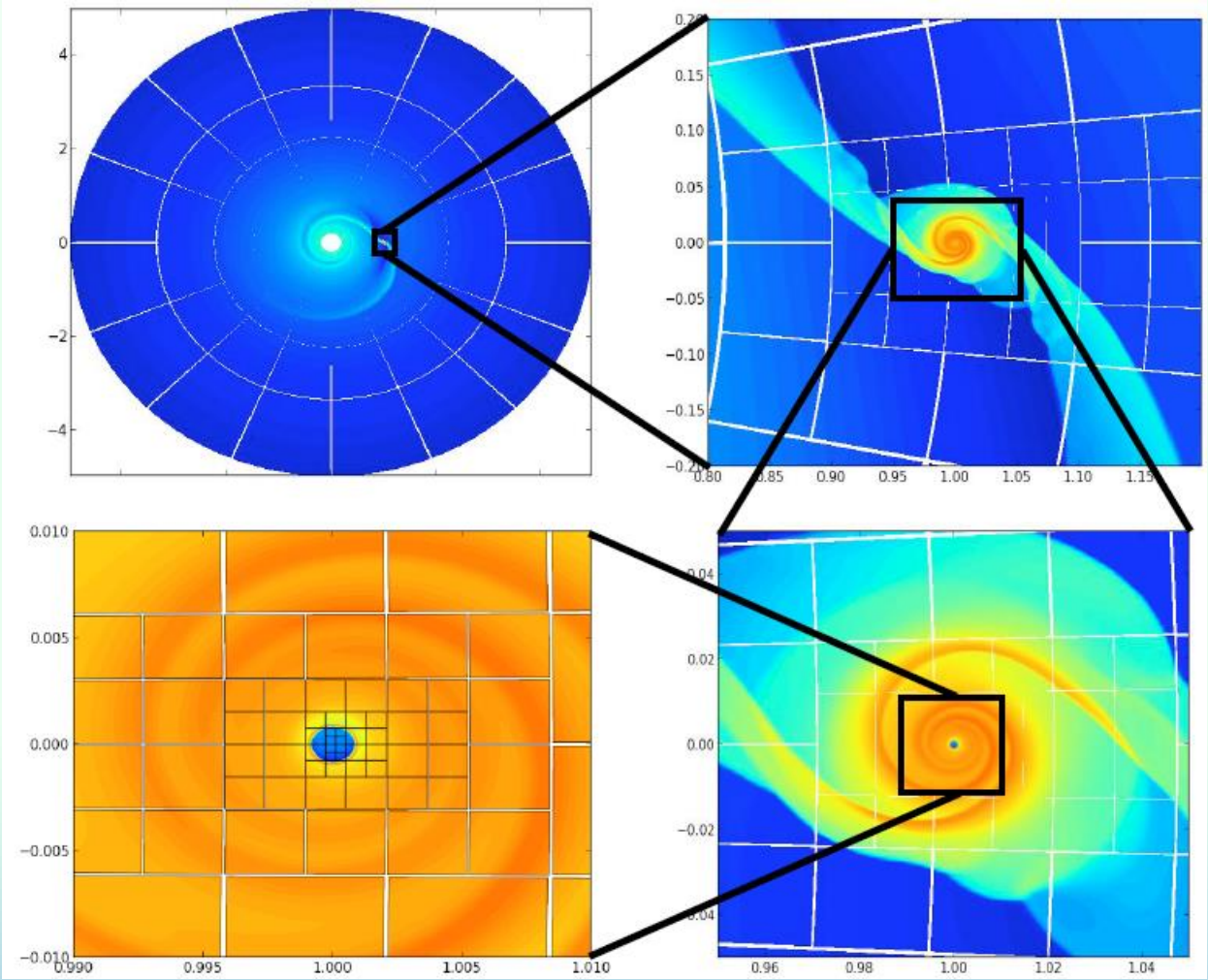
宇宙物理、特に重力が関わる現象を分解するには高解像度が必要

しかし、全領域を高解像度で分解する必要は必ずしもない。

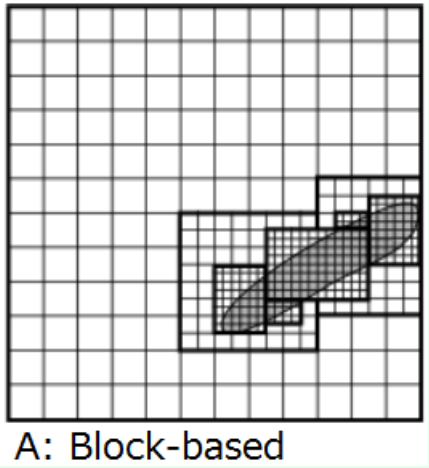
必要な領域のみ高解像度化
→ Adaptive Mesh Refinement

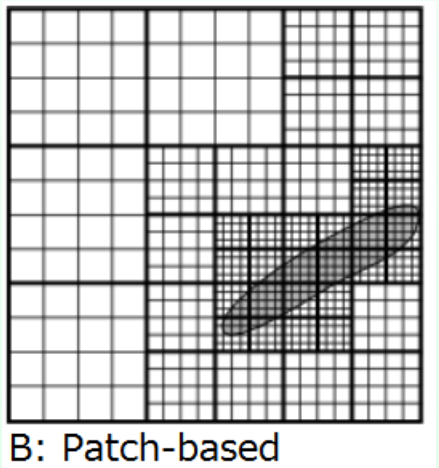
今日の宇宙物理学向けシミュレーションコードでは標準的な機能の一つ

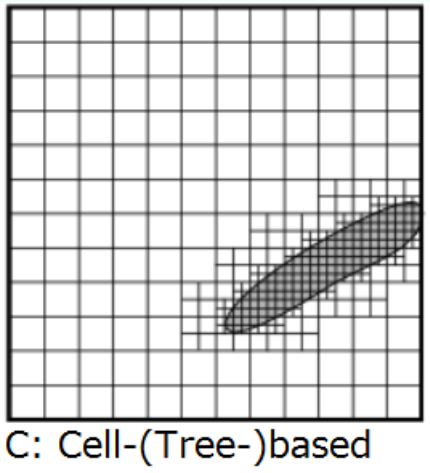
→ A circumplanetary disk embedded in a protoplanetary disk with Athena++.
(by Zhaohuan Zhu @ UNLV)



Athena++の格子設計


A: Block-based

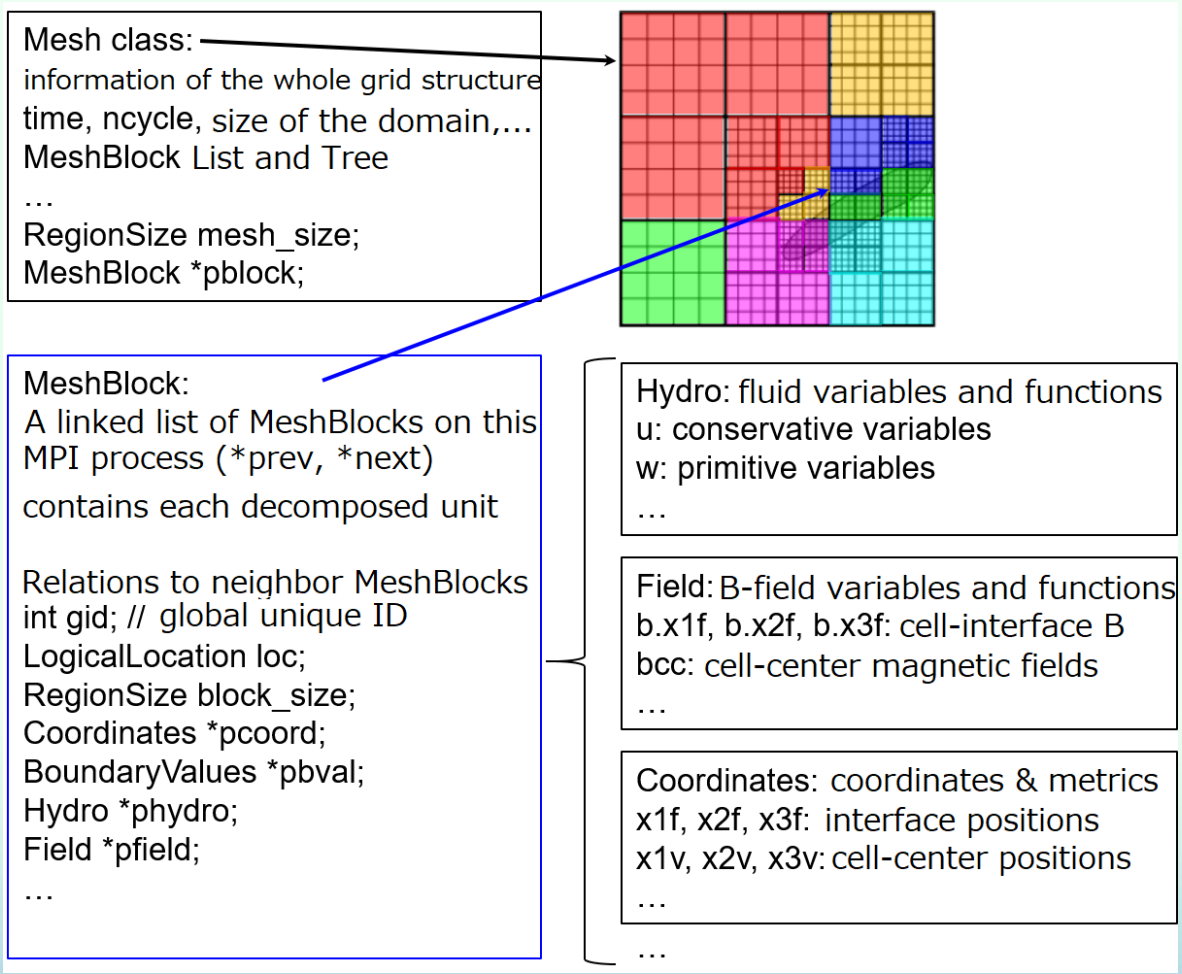

B: Patch-based


C: Cell-(Tree-)based

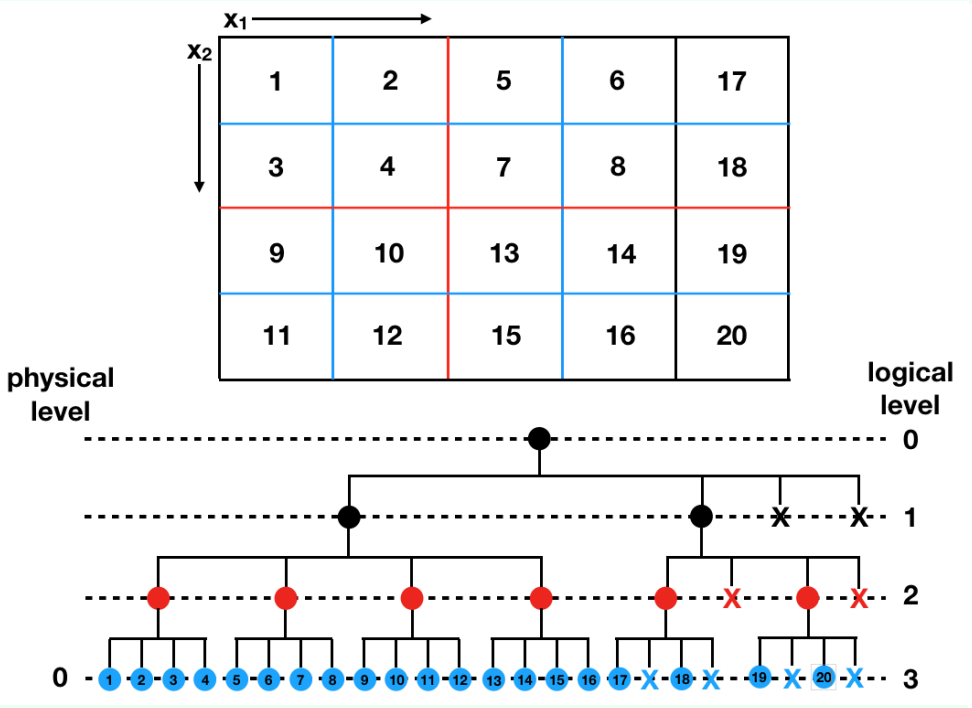
Pros	High efficiency **Uniform within block** **Use of existing scheme**	**Simple relations btw levels** **Uniform within block** **Use of existing scheme** Parallelization by space-filling curve	Highest efficiency Logically beautiful Parallelization by space-filling curve
Cons	Grids are not unique **Non-trivial grid generation** **Complex parallelization**	Lower efficiency (depending on patch size)	**Performance Issue** **Complicated grids** **(non-trivial neighbor cell)** **Hard to write,read,analyze**
Examples	Original: Berger & Colella 1989 Orion, PLUTO(Chombo) CASTRO(Boxlib), Enzo,…	FLASH(PARAMESH) Peano, Nirvana, SFUMATO,… **Athena++**	RAMSES, ART

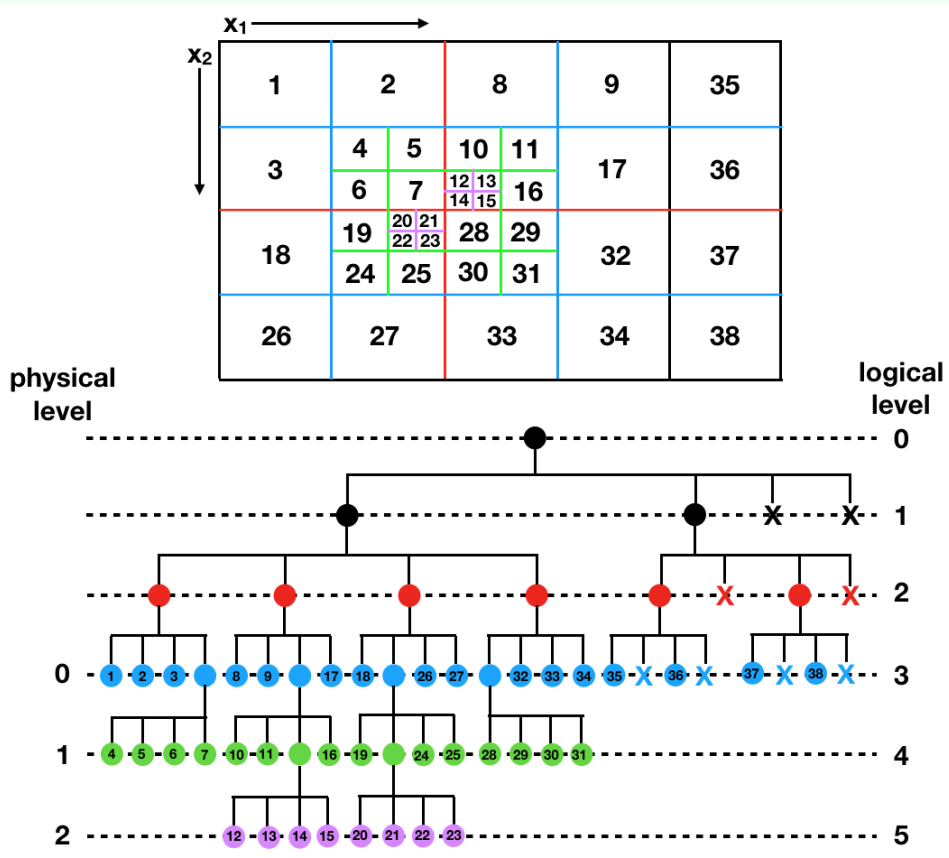
データ構造

- Mesh: 計算領域全体に対応するclass
  計算全体に関わる情報と（そのノードに存在する）MeshBlockを保持する
- MeshBlock:
  並列計算の分割単位
  かつAMRの分割単位
  流体等の物理量を保持
  全て論理的に同じ形（同じセル数）を持つ。



MeshBlockTree構造



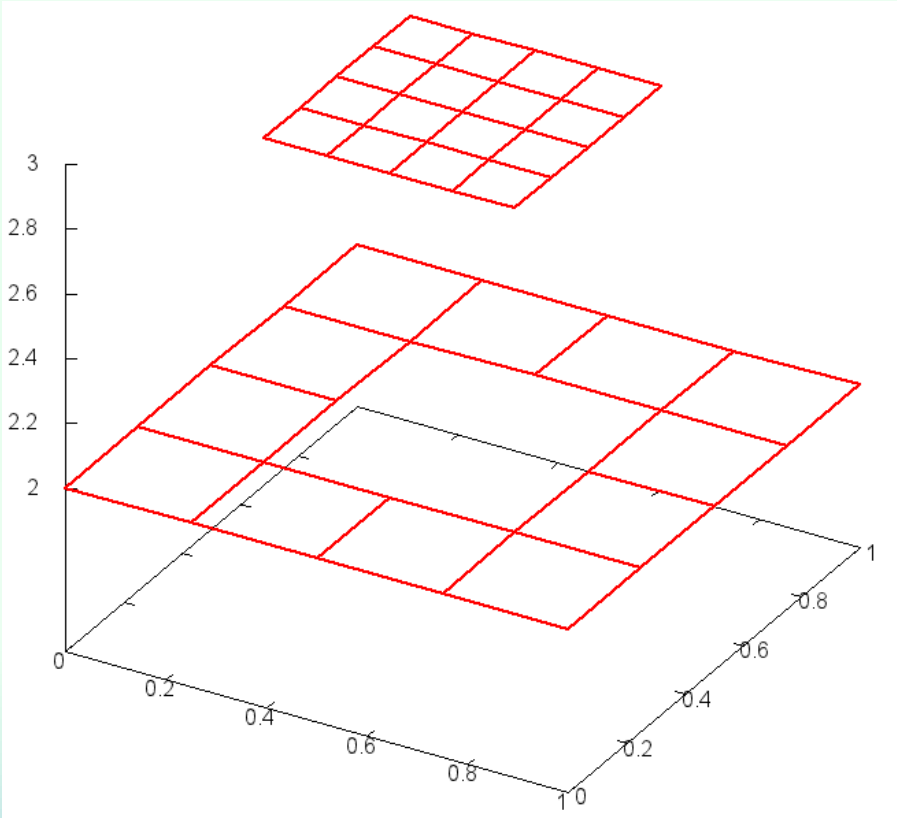


Left: 一様格子, right: AMR

- MeshBlockは8分木（3次元、2次元なら4分木）に格納・番号付け
- Global ID (gid) : Z-orderingで計算される一意なID
- LogicalLocation (loc): ツリー上での論理的位置 (lx1,lx2,lx3,level)

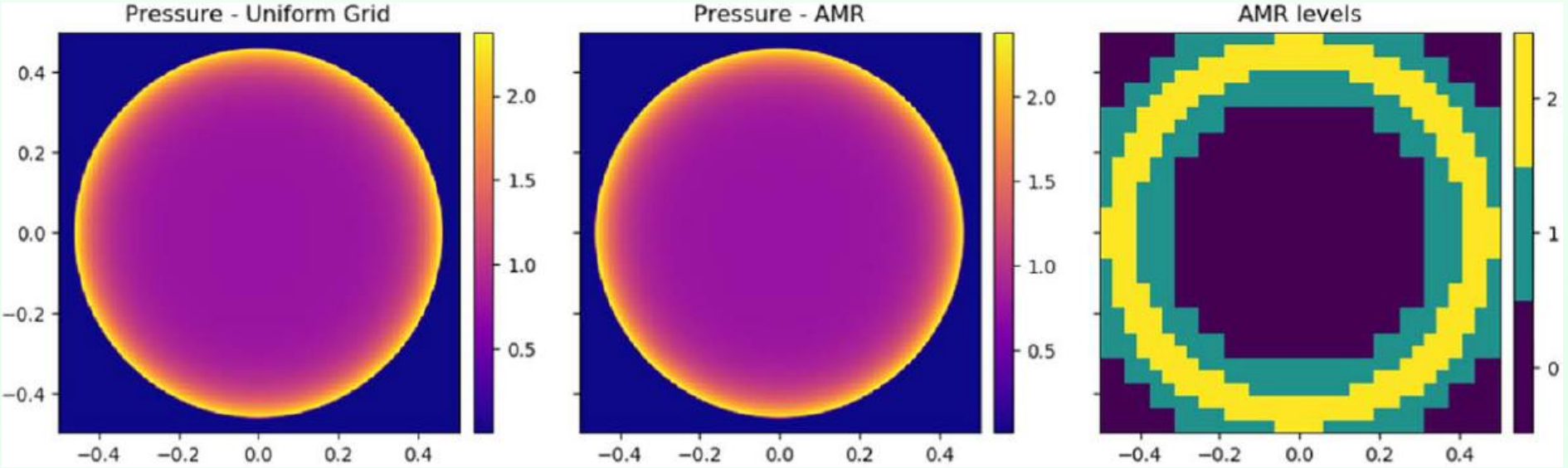
Athena++のAMRの特徴

- Athena++では細かい格子を生成した時、重複する粗い格子は消去
→空間の１点は一つ（だけ）の格子に含まれる。粗い格子には穴がある
- 一様時間刻みのみサポート
計算量は多めだが並列化が簡単かつ性能も出しやすい
実際には非一様時間刻みでのメリットはそれほど多くはない
- デフォルトでは格子数を均等するように並列化するが、これを修正する「コスト」の概念や、自動ロードバランスモードもあり



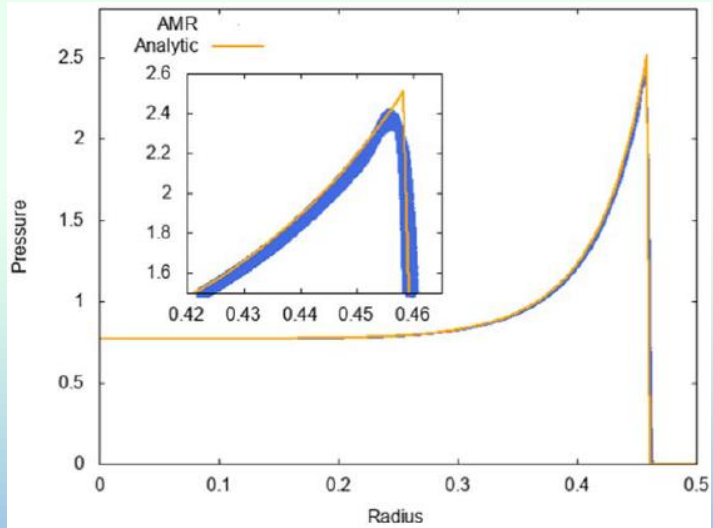
Adaptive Mesh Refinement

(Stone, Tomida, White, Felker, 2020, ApJS)

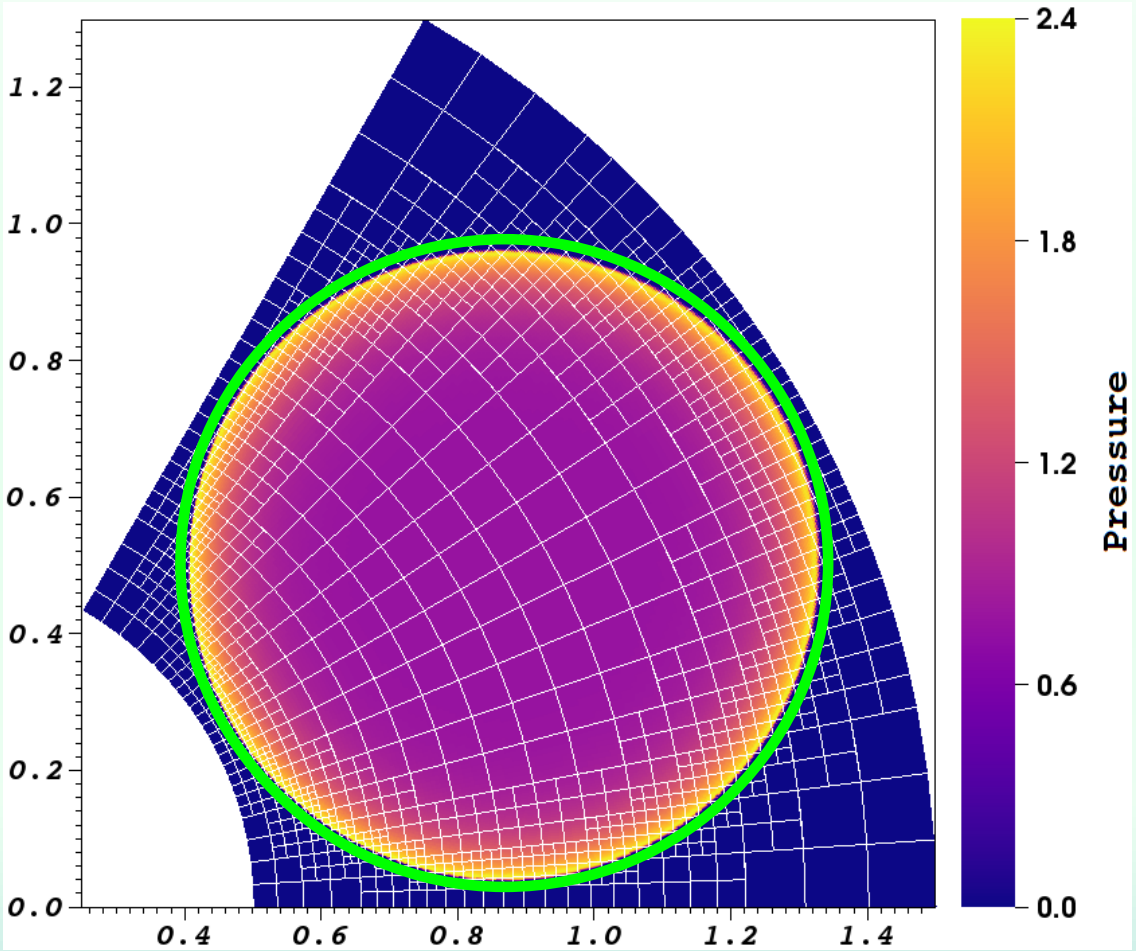


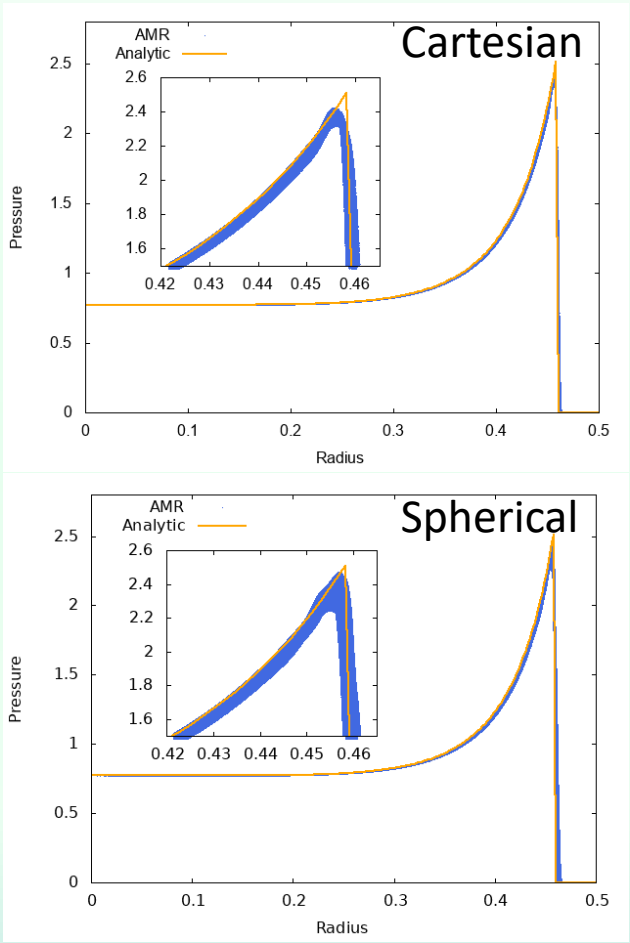
衝撃波面でのみ高分解能にするような条件でAMRを用いた計算例。512^3の解（左）に対してAMR（中）でほぼ同じ解が得られており、また解析解とも良く一致する（右図）。設定や並列数にもよるが、この問題ではAMRを用いると1/6程度の時間で計算できた。

AMRは強力だが、高分解能にする条件を誤ると正しい結果が得られない。また解析もやや面倒になる。



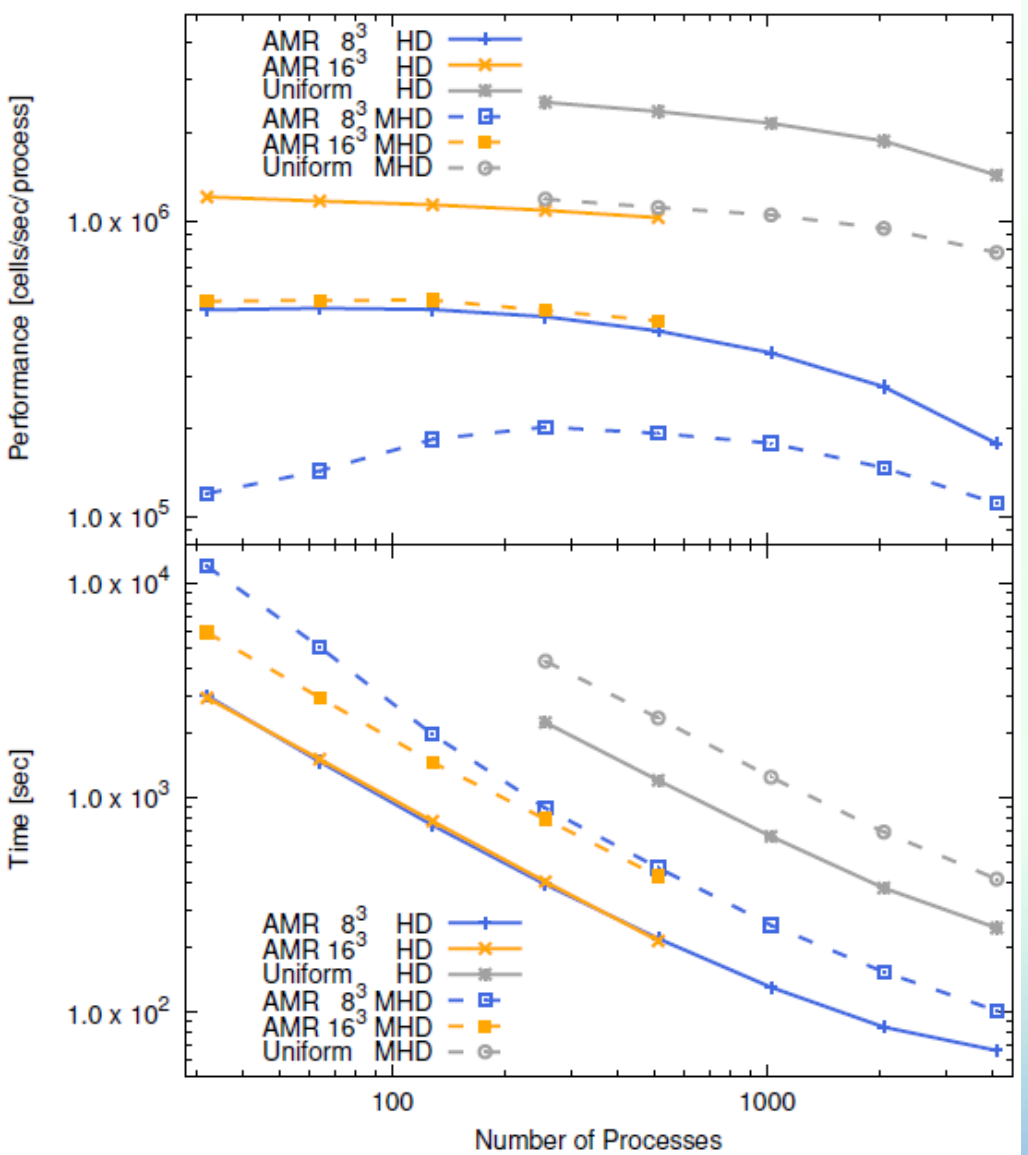
極座標でのAMR





- Athena++は対称性を良く維持できる（目の錯覚に注意）
- 解析解を非常に良く再現できる

AMRの性能



HD/MHD Blast wave tests
Uniform: 512^3
AMR: 3 levels with 8^3 and
16^3 MeshBlocks
2nd-order PLM,
HLLE for HD, HLLD for MHD
Cray XC50 @ NAOJ, Skylake
ストロングスケーリング

（計算時間短縮効果の半分は高温領域の分解能が下がったことで時間刻みが伸びたため）

もっと触ってみたい人へ

Athena++のコードの構造

athena/
  src/ -- source code directory
    bvals/ -- boundary conditions including communications
    coordinates/ -- coordinate definitions
    eos/ -- equation of state
    field/ -- magnetic field integrator
    hydro/ -- hydrodynamics integrators
    mesh/ -- grid generation, refinement
    outputs/ -- I/O functions
    pgen/ -- problem generators
    utils/ -- other utilities
  inputs/ -- sample input parameter files
  tst/ -- regression test scripts
  vis/ -- visualization scripts

どこを読めばいいか

流体ソルバを理解したい→hydro/
磁場ソルバを理解したい→field/
グリッドの構造を理解したい→mesh/
計算の流れを理解したい*→main.cpp, task_list/
並列化を理解したい**→mesh/, bvals/（**: 魔境です）
*: Athena++ではTaskListと呼ばれる、計算を細かいタスクに分割し、その順序とタスク間の依存関係をリストとして保持し、計算を実行する際にはタスクの進行状況に応じて動的に順序を入れ替えます。そのため所謂「メインループ」はTaskList::DoTaskListの以下の部分です。

```
80   while (nmb_left > 0) {↓
81 #pragma omp parallel for reduction(- : nmb_left) num_threads(nthreads) schedule(dynamic,1)↓
82     for (int i=0; i<nmb; ++i) {↓
83       if (DoAllAvailableTasks(pmesh->my_blocks(i), stage, pmesh->my_blocks(i)->tasks)↓
84           == TaskListStatus::complete) {↓
85         nmb_left--;↓
86       }↓
87     }↓
88   }↓
```