Implementation and Evaluation of the 5+1 waves MHD solver in the Kalypsso Hybrid CPU-GPU Platform.

G. Barreau[1,2], B. Bernecker[1,2], P. Kestener[1,3]
1: CEA DAM DIF, F-91297 Arpajon, France
2: Université Paris-Saclay, CEA, LMCE, F-91680, Bruyères-le-Châtel, France
3: Université Paris-Saclay, CEA, LiHPC, F-91297 Arpajon, France

I) INTRODUCTION / CONTEXT

Challenge and know problems:

- **Computational cost – 3D MHD** simulations are computationally **expensive**, particularly due to the propagation and refinement of discontinuities (shocks, contact surfaces) across the domain.
- **Portability & parallelism –** Writting efficient parallel code across multi-architectures (**CPUs,GPUs**) requires significant development overhead, interfering code maintainability and portability.
- **Solver robustness –** A single Riemann solver may fail or lose accuracy in specific flow regimes, limiting the range of physical configurations a code can reliably handle.

Solutions provided with Kalypsso:

- **Kalypsso** (Kokkos Applicative LaYer fot Parallel and Scalable Solvers on Octrees): C++ platform capable of solving ideal MHD equations **[6]**.
- **AMR** (Adaptative Mesh Refinement) **- Kalypsso** is coupled with **p4est [2]**, a software library allowing to manage **AMR on octree** → Dynamically concentrates resolution where it is needed, drastically reducing the total cell count while preserving accuracy near discontinuities.
- **Kokkos** - Kalypsso is coupled with **Kokkos** ecosystem **[3]** allowing a performance portability → greatly simplifies development for HPC applications.
- **Multi-solver architecture** – Several Riemann solvers of different numerical nature are implemented in Kalypsso and can be selected at runtime, improving robustness across flow regimes and easing the integration of future solvers.

II) METHODOLOGY

The ideal compressible MHD system of equations:

- $\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}) = 0,$
- $\frac{\partial \rho \boldsymbol{u}}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}\boldsymbol{u} + (P_{tot})\mathbb{I} - \boldsymbol{B} \otimes \boldsymbol{B}) = 0,$
- $\frac{\partial \rho E}{\partial t} + \nabla \cdot [(\rho E + P_{tot})\boldsymbol{u} - \boldsymbol{B}(\boldsymbol{B} \cdot \boldsymbol{u})] = 0,$
- $\frac{\partial \boldsymbol{B}}{\partial t} + \nabla \cdot (\boldsymbol{u} \otimes \boldsymbol{B} - \boldsymbol{B} \otimes \boldsymbol{u}) = 0$

with $P_{tot} = P_{th} + \frac{|\boldsymbol{B}|^2}{2}$ and $E = \frac{P_{th}}{\rho(\gamma-1)} + \frac{|\boldsymbol{u}|^2}{2} + \frac{|\boldsymbol{B}|^2}{2}$

Solved with

MUSCL-Hancock procedure [5]:

1 - Compute primitive variables $\mathbf{Q}_i^n$ from cell-centered conservative variables $\mathbf{U}_i^n$

2 - Compute limited slopes:

$$\boldsymbol{\delta}\mathbf{Q}_i = \mathrm{sign}(\delta \mathbf{Q}_{C,i}) \min(2|\delta \mathbf{Q}_{U,i}|, 2|\delta \mathbf{Q}_{D,i}|, |\delta \mathbf{Q}_{C,i}|)$$

3 - Advance solution at half time step with:

$$\mathbf{Q}_i^{n+1/2} = \mathbf{Q}_i^n - \frac{\Delta t}{2\Delta x} \mathbf{A}_{\mathbf{Q}} \delta \mathbf{Q}_i$$

4 - Compute half time step predictor of evolution of face-centered scheme:

$$\boldsymbol{Q}_{i,L}^{n+1/2} = \boldsymbol{Q}_i^{n+1/2} - \frac{1}{2}\delta \boldsymbol{Q}_i$$

$$\boldsymbol{Q}_{i,R}^{n+1/2} = \boldsymbol{Q}_i^{n+1/2} + \frac{1}{2}\delta \boldsymbol{Q}_i$$

5 - Compute fluxes by using Riemann solver:

$$\boldsymbol{F}_{x,i-1/2}^{n+1/2} = \boldsymbol{RS}\left(\boldsymbol{Q}_{i,L}^{n+1/2}, \boldsymbol{Q}_{i-1,R}^{n+1/2}\right)$$

6 - Update conservative variables from $\boldsymbol{t_n}$ to $\boldsymbol{t_{n+1}}$:

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

AMR:

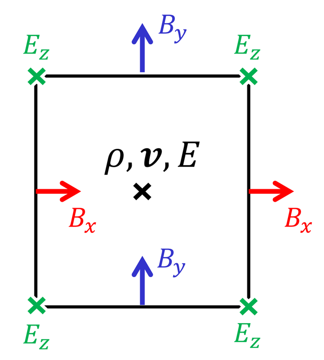
The domain is decomposed into base **blocks of size** $(block_x \times block_y \times block_z)$
Each **refinement level** increase resolution by a factor 2 per spatial direction.
At level l, cells are refined by a **factor $\mathbf{2^l}$** relative to the coarsest level.
For an **uniform 2D mesh** ($l_{min} = l_{max} = l$) → equivalent to a Cartesian resolution of: $(block_x \cdot 2^l) \times (block_y \cdot 2^l)$

III) MULTI-SOLVER ARCHITECTURE

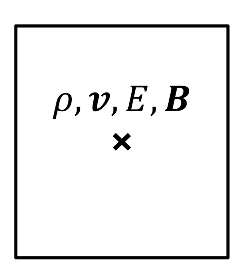
The object-oriented C++ design abstracts the solver interface, so that **adding a new solver**, whether a different Riemann solver (HLL, HLLD, Local Lax-Friedrichs,,,) or a different spatial discretization, **requires minimal changes** to the rest of the codebase.

Two representative solvers are presented in this work:

	HLLD-CT [1]	5+1 waves solver [4]
Grid type	Staggered	Cell-centered
B-field storage	Face-centered	Cell-centered
$\nabla \cdot \mathbf{B} = 0$	Constraint Transport (machine precision)	Not enforced – truncation-level errors, entropy satisfying
Robustness (low β)	May fail (not entropy-satisfying)	Robust via local entropy correction
Status	Community reference	Recent (2024)



Staggered grid



Cell-centered grid

IV) NUMERICAL RESULTS

Classical ideal MHD flow:

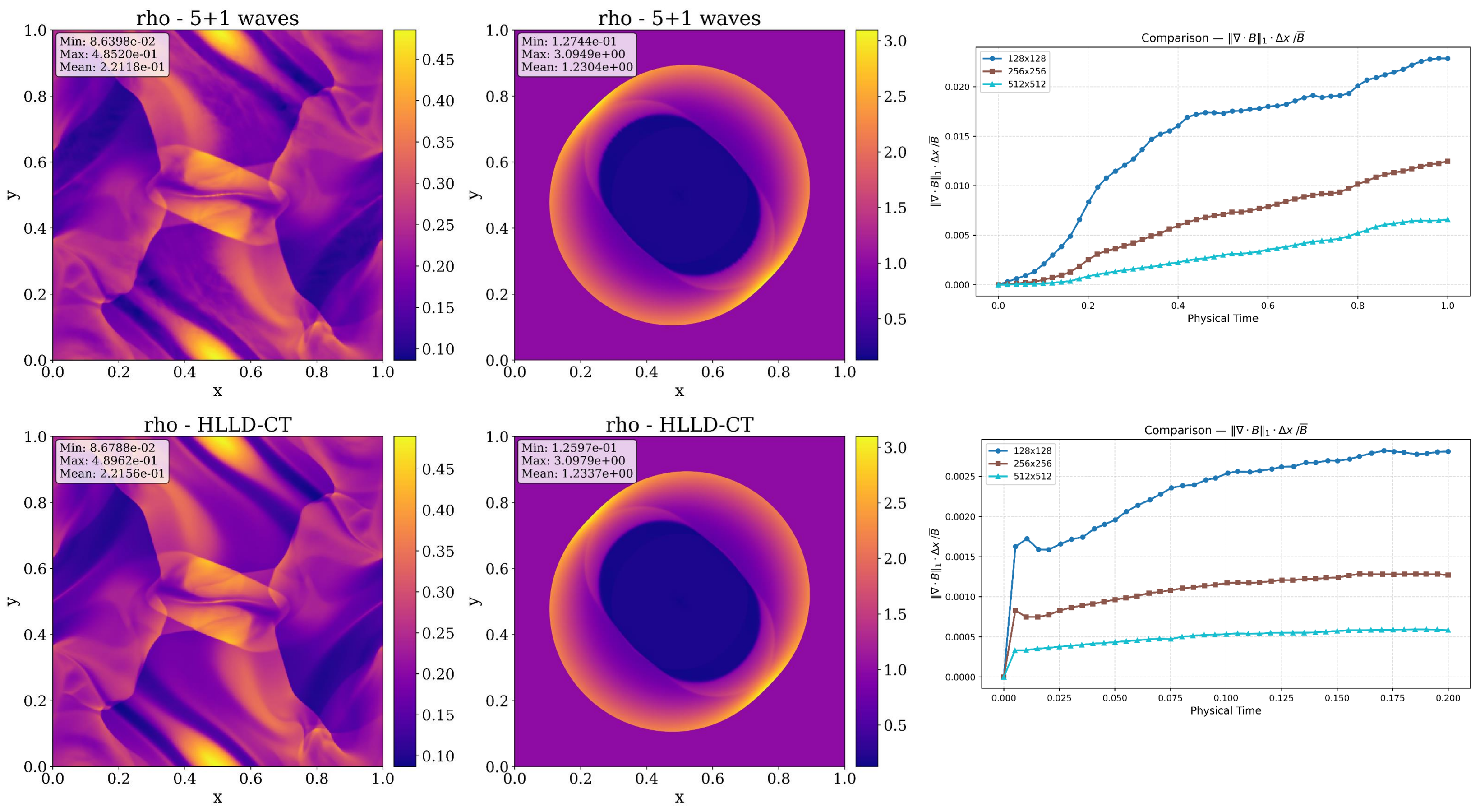


Fig 1. Comparaison between 5+1 waves and HLLD-CT solvers: Left: density contours of Orszag-Tang flow with $lvl_{min} = 4, lvl_{max} = 6$, base blocks: 8×8 cells as AMR parameters (max effective resolution: 512×512). Middle: density contours of MHD blast flow with $lvl_{min} = 3, lvl_{max} = 10$, base blocks: 4×4 as AMR parameters (max effective resolution: 4096×4096). Right: $\nabla \cdot B$ error evolution for Orszag-Tang (top) and MHD blast (bottom) of the 5+1 waves solver.

Block size	l_{min}	l_{max}	GPU Mcell-update/s (time)	CPU Mcell-update/s (time)
4×4	3	10	$60\ (86s) - 107\ (45s)$	$5.1\ (1022s) - 14.8\ (325s)$
8×8	3	9	$108\ (59s) - 163\ (39s)$	$9\ (720s) - 20.6\ (307s)$
16×16	3	8	$149\ (59s) - 206\ (43s)$	$12.1\ (731s) - 23.4\ (378s)$

Tab 1. GPU/CPU comparative performance of a 2D MHD Blast for HLLD-CT (red) and 5+1 waves (blue) solvers on different AMR configurations (equivalent to a max effective resolution of 4096×4096). GPU runs are made on a single Nvidia A100 GPU and CPU runs are made on 32 CPU cores on a single AMD Milan node on the CCRT/Topaze supercalculator.

Low β plasma regimes:

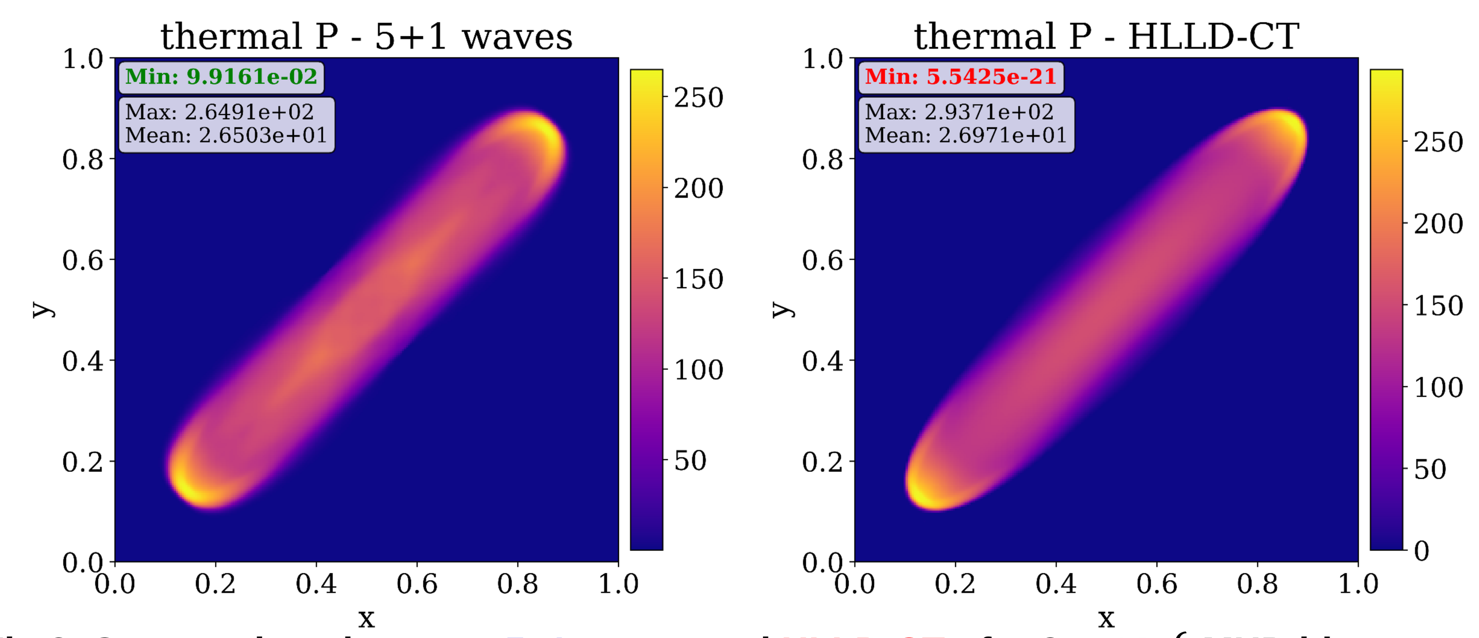


Fig 2. Comparaison between 5+1 waves and HLLD-CT of a $\boldsymbol{\beta} \approx 10^{-6}$ MHD blast. AMR: $lvl_{min} = 3, lvl_{max} = 5$, base blocks: 8×8 (max effective resolution 256×256).

- Initialisation where $\beta = \frac{P_{th}}{B^2/2} \approx 10^{-6} \rightarrow P_{th} \ll P_{mag} = |\mathbf{B}|^2/2$
- HLLD-CT is **instable** in these flow regimes → Using **a threshold** for P_{th} to avoid $P_{th} < 0$.
- 5+1 waves with the entropy-satisfying formulation is **stable in low $\boldsymbol{\beta}$** plasma regimes.

CONCLUSION

- **Kalypsso: a new platform allowing large 2D/3D MHD simulations using AMR (p4est) and performance portability (Kokkos).**
- **Object-oriented design allows easy integration of new solvers, regardless of their numerical nature.**
- **Multiple solvers cover a broad range of physical regimes , users select the most appropriate (low plasma β→ 5+1 waves, strict $\nabla \cdot \mathbf{B} = 0$ → HLLD-CT).**
- **Multiple solvers allows cross validation between solvers.**

[1]: S, Fromang et al, 2006, Astronomy & Astrophysics, 457(2), 371-384.
[2]: C. Burstedde et al 2011, p4est, SIAM Journal on Scientific Computing
[3]: C. R. Trott et al 2022, IEEE Transactions on Parallel and Distributed Systems 33.4
[4]: P. Tremblin et al 2024, Journal of Computational Physics 519
[5]: B. van Leer, 2003, Fluid Dynamics and Co-located Conferences
[6]: P, Kestener, 2025, SSRN Scholarly Paper