

mini-ramses



Romain Teyssier



PRINCETON
UNIVERSITY

mini-ramses: data structure

- Global oct indexing uses a quadruplet `hash_key=(l,i,j,k)`.
- Hash table: `igrid=get_grid(hash_key)`
 - Octs are pulled from memory using a simple hash function, collisions are handled using a linked list (same as a python dictionary)
- Octs are sorted in memory first `per refinement level`, then `per Hilbert key` (using a fast radix sort)
- Octs can be grouped in blocks called `superocts`
- `Perfect load balancing` at every time step and for each level
- Remote information is managed (using MPI) via a `software cache`
 - Use only `10'000 ghost octs` to store (temporarily) remote information.
 - Use many small MPI messages while computing
 - Use `pre-fetch for optimized communications` (with a larger cache size)

mini-ramses: data structure

- Optimized execution using **clean octs**, **dirty octs** and **ghost octs**
- **Clean octs** have **all neighbors** in local memory
- **Dirty octs** have **at least one neighbor** in:
 - Physical boundaries
 - Other MPI domain
 - Coarser refinement level
- **Ghost octs** are created temporarily in the cache memory from:
 - Physical BC rules (reflexive, zero gradient, imposed...)
 - MPI communications
 - Coarse-to-fine interpolation
- Use GPU or OpenMP **asynchronicity** to:
 - Build the cache while updating the clean octs
 - Then update the dirty octs

mini-ramses: initial conditions

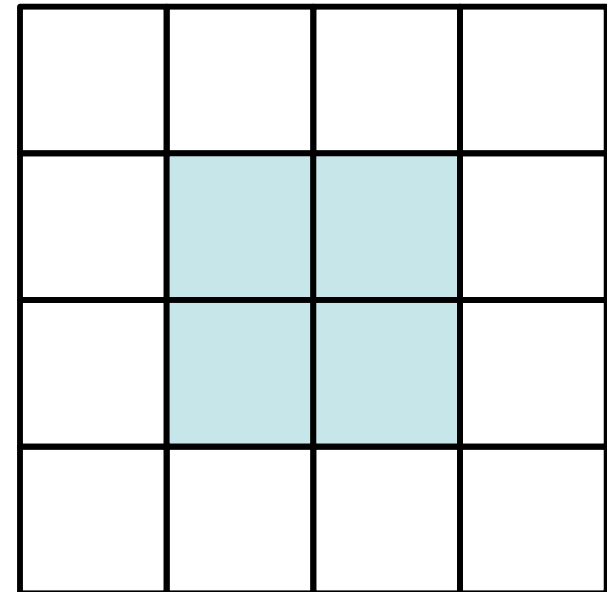
- Various predefined initial conditions
 - Use `condinit.f90` initial condition setup file.
 - Default using regions and namelist block `INIT_PARAMS`
 - Predefined ICs: `INIT=COEUR`, `INSTA`, `DOUBLEMACH`, `OT`, `PONO`, `ABC`, `LOOP...`
 - `PATCH=../patch/init/halo` and `PATCH=../patch/init/merger`
- MUSIC initial conditions using `filetype='grafic'` or `filetype='grafic_zoom'`
- GADGET initial conditions using `filetype='gadget'`
 - Generated using e.g. the DICE code
- RAMSES initial conditions using `filetype='ramses'`
 - Reads RAMSES output files (not restart files)
 - Still need to specify the initial magnetic field separately

mini-ramses: boundary conditions

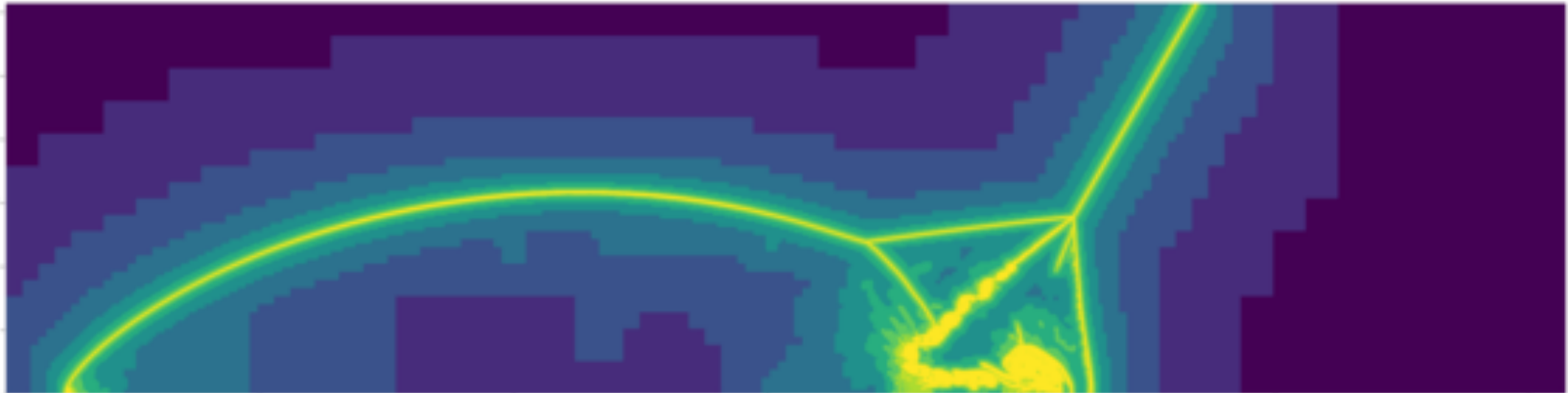
- Various boundary conditions (default is periodic)
 - Arbitrary box geometry using oct Cartesian indexing at `bound_levelmin`:
 - Reflective, zero gradient, imposed...
 - Isolated for gravity solver (FMM, MG)

Example from `sedov2d.nml`

<code>bound_levelmin=3</code>	<code>periodic=.false.,.false.</code>
<code>box_size=1.0</code>	<code>nbound=4</code>
<code>box_xmin=1</code>	<code>bound_dir=1,1,2,2</code>
<code>box_ymin=1</code>	<code>bound_type=1,1,1,1</code>
<code>box_xmax=-1</code>	<code>bound_shift=1,-1,1,-1</code>
<code>box_ymax=-1</code>	<code>bound_xmin= 0,-1, 0, 0</code>
	<code>bound_xmax= 1, 0, 0, 0</code>
	<code>bound_ymin= 1, 1, 0,-1</code>
	<code>bound_ymax=-1,-1, 1, 0</code>



Double Mach reflection (boundary conditions)

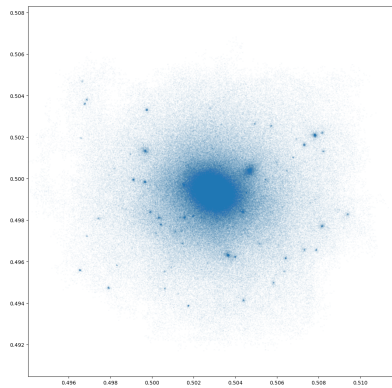


mini-ramses: physics routines implemented

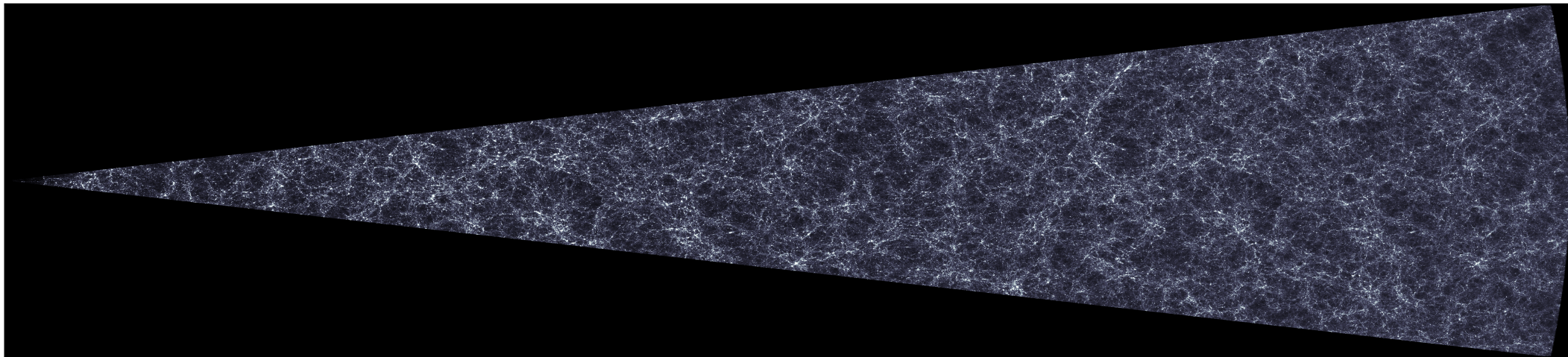
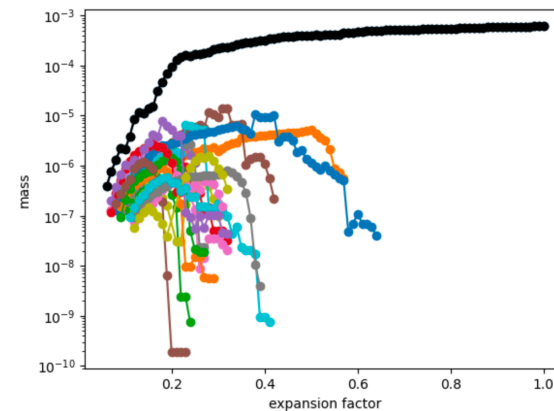
- MUSCL-Hancock hydro solver (minmod, moncen, LLF, HLL, HLLC)
- MHD solver now merged with hydro solver (HLLD, Roe)
- PIC solver (CIC, TSC, PCS)
- CG and MG gravity solver
- Star formation recipe with 3 different models
- Dual energy update using entropy
- Thermal supernova feedback
- Mechanical supernova feedback
- Subgrid turbulence LES model
- Turbulence driving
- Equilibrium cooling, constant UV background with self-shielding
- Halo finder, merger tree (Rayan Ait-Ekioui)
- Light cone (Takanori Codron-Akieda)
- Non-equilibrium cooling with H and He chemistry (Joke Rosdahl)
- RAMSES-RT (Joki Rosdahl)
- Sink particles (Ziyong Wo, Nick Choustikov, William Groger)
- Dust MHD/PIC (Eric Moseley)
- FMM gravity solver (Jun-Young Lee)

mini-ramses: clump finder, merger tree, light cone

- Clump finder
 - New PHEW implementation
 - Central halos and satellite halos
 - Particle unbinding



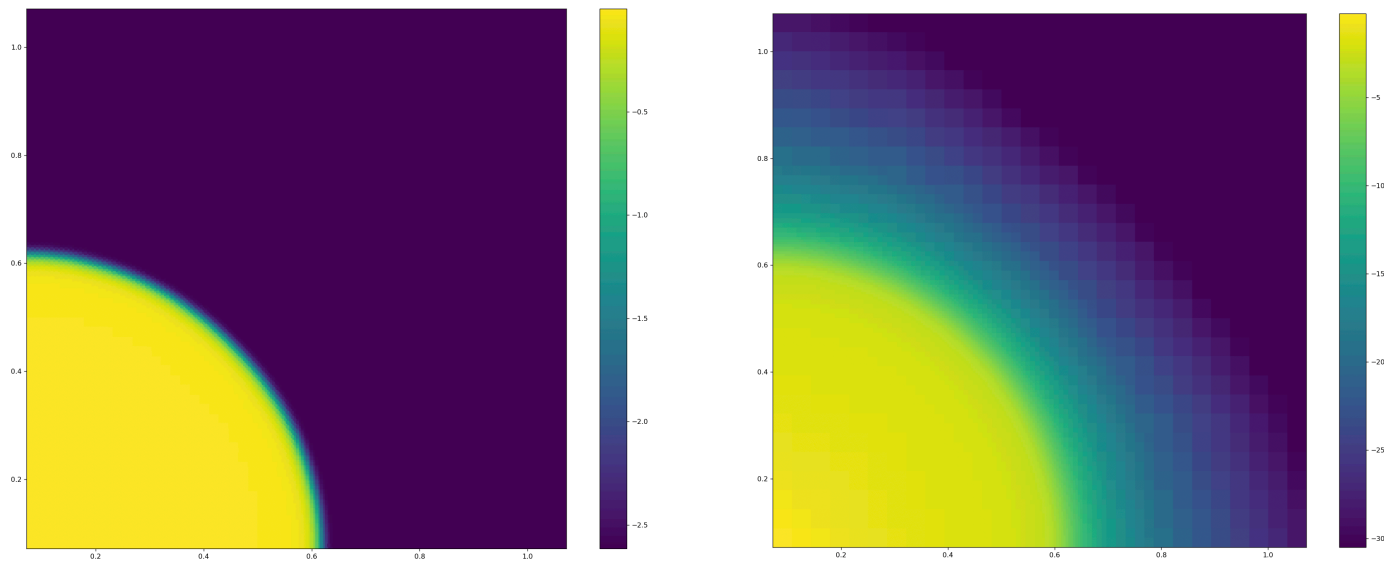
- Merger tree
 - On the fly using tracer particles
 - Collect host clump properties in post processing



- Rayan Ait-Ekioui, Takanori Cordon-Akieda, James Sunseri

mini-ramses: sink particles and M1-RT

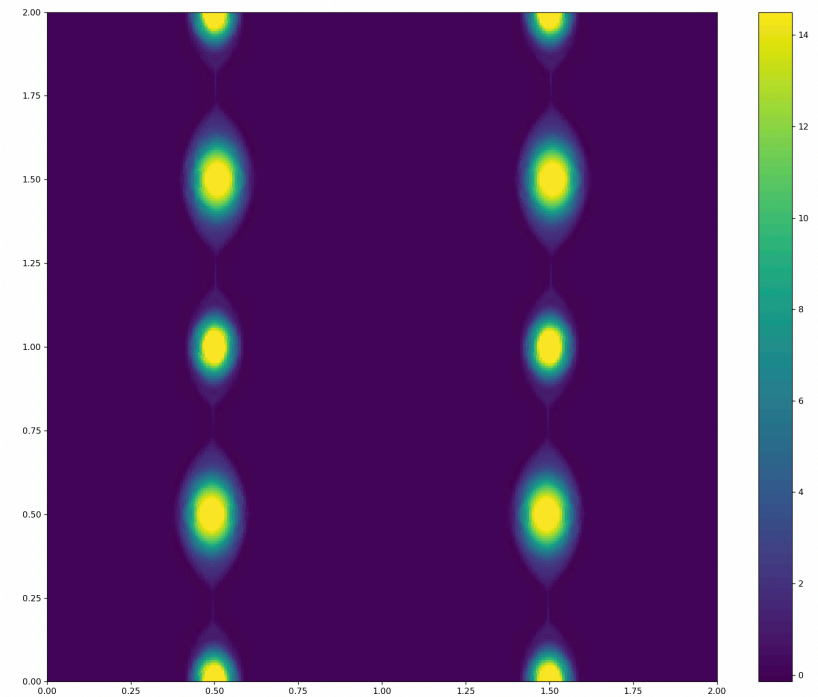
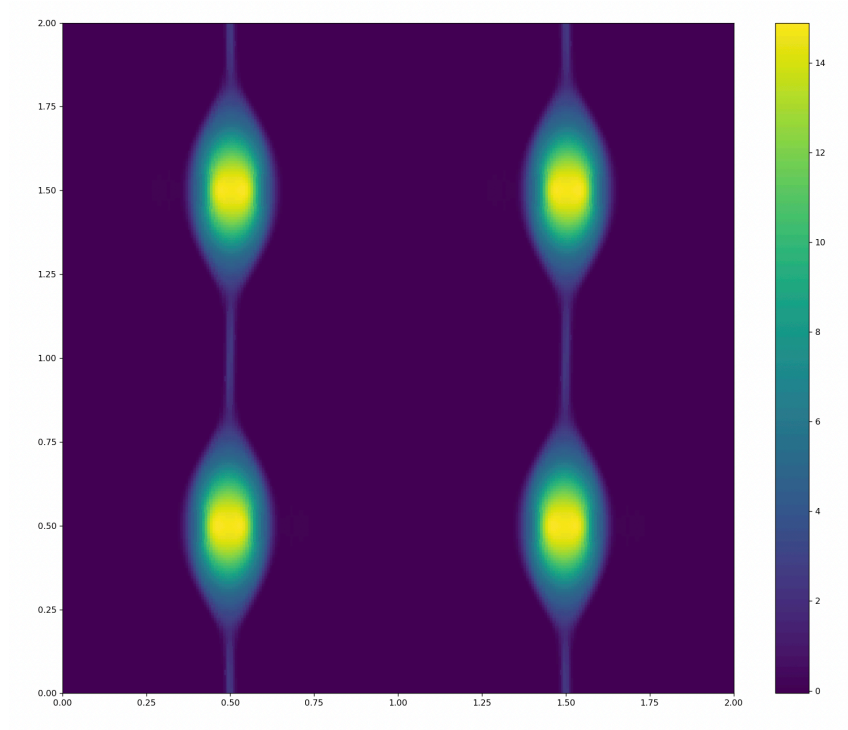
- Sink formation and evolution
 - Seeding using clump finder (Ziyong Wu)
 - Accretion and feedback (Nick Choustikov)
 - Merging (William Groger)
- Radiative transfer with M1 solver (Joki Rosdahl)



mini-ramses: initial conditions

- 1D tests:
 - HYDRO: `advect1d.nml`, `blast1d.nml`, `sedov1d.nml`, `tube1d.nml`
 - MHD: `brio_wu.nml`, `dai_woodward.nml`
- 2D tests:
 - HYDRO: `sedov2d.nml`, `double_mach.nml`, `rt.nml`
 - MHD: `loop.nml`, `ot.nml`, `current_sheet.nml`
 - RT: `stromgren2d.nml`, `beam2d.nml`, `shadow2d.nml`
- 3D tests:
 - N-BODY: `dmo.nml`, `dmo_zoom.nml`
 - HYDRO: `sedov3d.nml`, `coeur.nml`, `cosmo.nml`, `cosmo_zoom.nml`, `halo.nml`, `merger.nml`, `dice.nml`
 - MHD: `pono.nml`, `abc.nml`, `halo_mhd.nml`
 - SINK: `bondi.nml`, `agn_jet.nml`

Current sheet (MHD)



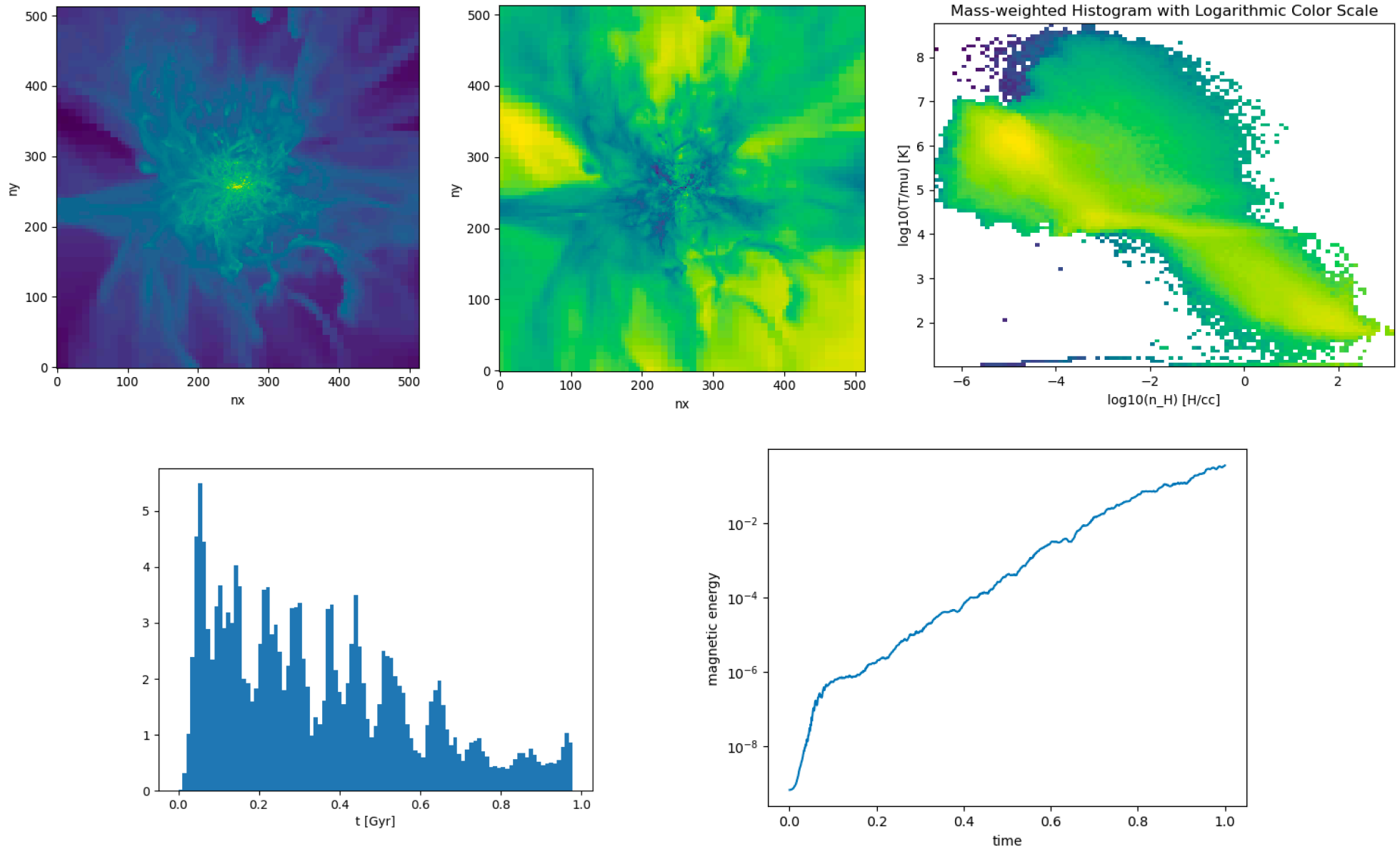
mini-ramses: new features

- New output format
 - `access='stream'` python read using `numpy.fromfile`
 - Backup frequency `bkp_timrs_hrs=2` and `bkp_modulo=3`
 - Restart folder: double precision, face-centered magnetic field (6 vars)
 - You can change the number of cpu at restart
 - Output folder: single precision, cell-centered magnetic field (3 vars)
 - You can specify the number of files per output folder
 - Default `nfile=1` with $1 \leq nfile \leq ncpu$
- Maximum level 64 using `HILBERT=3` (no time penalty).

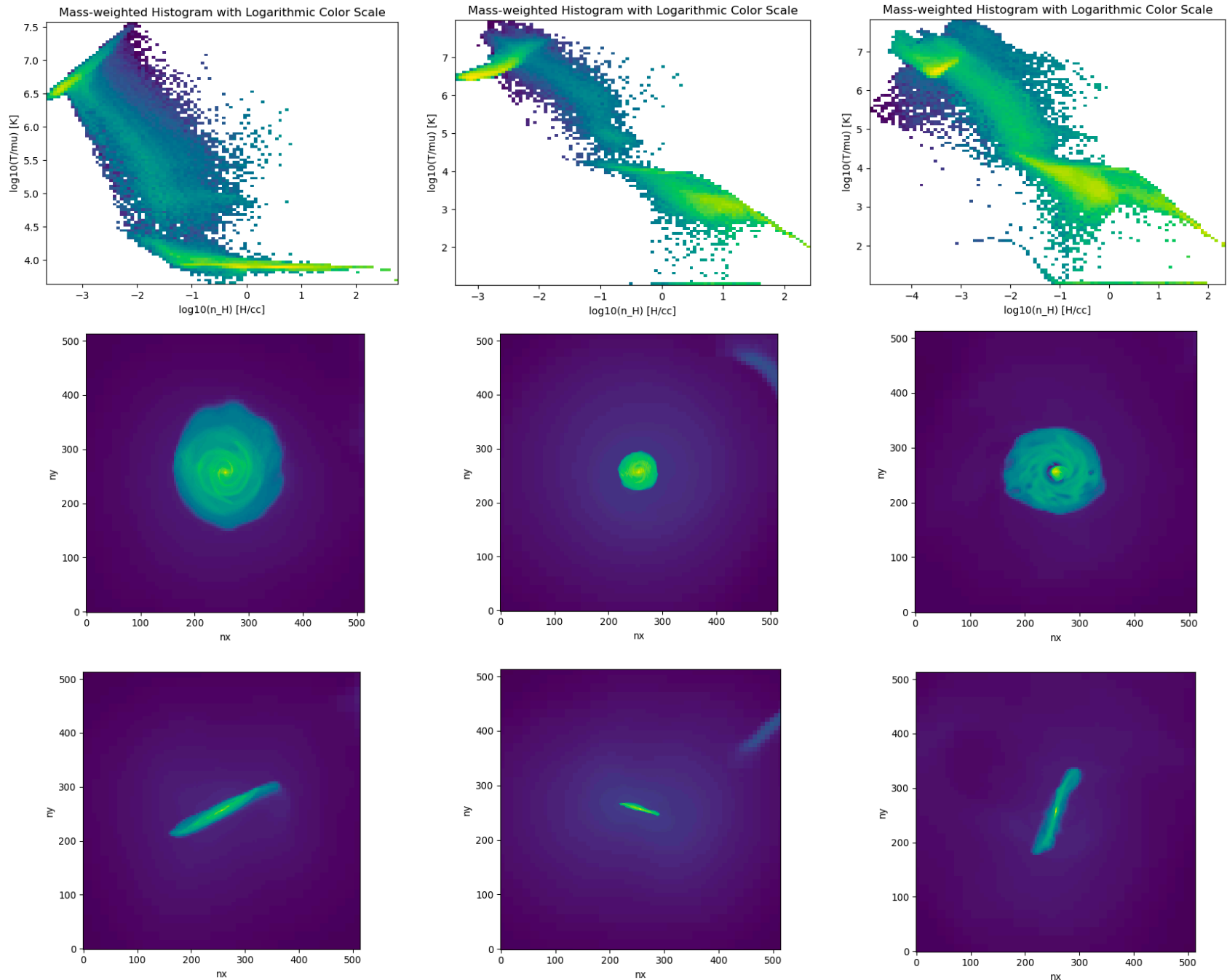
mini-ramses: new features

- Makefile:
 - `make NDIM=3 MPI=1 DEBUG=1 UNITS=MERGER GRAV=1 PATCH=../patch/init/merger NPSCAL=2 MHD=1`
- cmake also available:
 - `cmake .. -DNDIM=3 -DINIT=PONO -DMHD=1`
- `NENER=0` implemented but not tested
- 5 hydro variables for NDIM=1, 2 and 3 and also for MHD
- `NPSCAL=0` for additional passive scalars
- Python package `miniramases.py` (reading and plotting)
- Python utils `amr2img.py`, `map2img.py`, `run2img.py`, `debug.py`, `history.py`, `log.py`, `sfr.py`...
- F90 utils `amr2map.f90` and `part2map.f90`

mini-ramses: galactic dynamo in a cooling halo



mini-ramses: cosmological galaxy formation



mini-ramses: software engineering

- <https://bitbucket.org/rteyssie/mini-ramses>
- GPU hydro kernel prototype developed in Nvidia Fortran by Bob Caddy (Princeton Research Software Engineer). See folder [mini-ramses/gpu](#)
- Princeton Open Hackathon with Nvidia in June 2025 to complete the hydro kernel (MPI). No AMR yet.
- 2 billion cell updates per GPU per second (A100) in single precision
- 250 million cell updates per CPU-node per second (AMD 64 cores)
- Folder [mini-ramses/doc](#)
- I/O options to implement in the future:
 - Frequency of variables at run time? Other derived variables?
 - Particle birth index sorting or particle halo membership sorting?
 - Cell Hilbert index sorting or cell halo membership sorting?