RAMSES Namelist Parameter Reference

cuRAMSES-kjhan – February 2026

Juhan Kim

Based on RAMSES by Romain Teyssier

This document provides a complete reference for every namelist parameter accepted by RAMSES and the cuRAMSES-kjhan extensions. Parameters are grouped by their Fortran namelist block (`&RUN_PARAMS`, `&AMR_PARAMS`, etc.). Each entry specifies the parameter name, Fortran type, default value, and a detailed description including valid ranges and interactions with other parameters.

The namelist file uses standard Fortran namelist syntax. Each block begins with `&BLOCK_NAME` and ends with a single `/`.

Contents

1 &RUN_PARAMS – Global Run Control

This mandatory block controls the physics modules to activate, restart behaviour, domain decomposition strategy, and general simulation parameters.

cosmo logical default: `.false.`

Enable cosmological mode. When `.true.`, RAMSES uses comoving (super-comoving) coordinates with the expansion factor $a(t)$ as the time variable. The box length is interpreted in h^{-1} Mpc. Friedmann equations are integrated internally.

Enabling this flag also activates expansion-factor–based output scheduling (see `aout` in Section 3). Cosmological initial conditions (GRAFIC2 format) must be provided via `initfile`.

pic logical default: `.false.`

Enable the Particle-In-Cell (PIC) method for collisionless N-body dynamics (dark matter, stars). Particles are deposited onto the AMR grid using cloud-in-cell (CIC) interpolation, and forces are interpolated back to particle positions.

Required for any simulation containing dark matter particles. Usually combined with `poisson=.true.`

poisson logical default: `.false.`

Enable the self-gravity Poisson solver. RAMSES uses an adaptive multigrid (MG) method on the AMR hierarchy with V-cycles and red–black Gauss–Seidel smoothing. Convergence is controlled by `epsilon` in `&POISSON_PARAMS`.

Must be `.true.` whenever `pic=.true.` or whenever gas self-gravity is desired.

hydro logical default: `.false.`

Enable the hydrodynamics (or MHD) solver. RAMSES employs a second-order MUSCL–Hancock scheme with approximate Riemann solvers (see `scheme`, `riemann` in Section 6).

Set to `.true.` for any simulation involving baryonic gas.

nrestart integer default: 0

Restart from checkpoint (output snapshot) number `nrestart`.

- `nrestart=0` – fresh start from initial conditions.
- `nrestart=`N – load `output_`N`/` and resume.

The number of MPI processes must match the run that produced the checkpoint. RAMSES reads all AMR, hydro, particle, and gravity data from the snapshot directory.

`nremap` integer default: 5

Load-balancing frequency: perform domain decomposition every `nremap` coarse time steps. Recommended value: **5** (balances redistribution overhead against growing load imbalance). Set to `0` to disable load balancing entirely.

Note: Benchmarks (200 M particles, 12 ranks, 10 steps) show that `nremap=5` reduces total runtime by 18% compared to `nremap=1`, with load-balance overhead at 6.3% of wall time. Larger values (e.g. 10) save overhead but allow imbalance to grow.

`nsubcycle` integer array default: `1,1,2`

Time sub-cycling factors per AMR level. The i-th entry gives the number of fine time steps per coarse step at level $\texttt{levelmin} + i - 1$. Typical usage:

- `1` for coarse levels (no sub-cycling).
- `2` for fine levels (halve the time step at each finer level).

The array has up to `levelmax-levelmin+1` entries. Any unspecified trailing entries default to `1`.

Example:

```
nsubcycle = 1, 1, 1, 2, 2, 2, 2
```

`ncontrol` integer default: 1

Print control output (energy diagnostics, timing) every `ncontrol` coarse time steps to standard output.

`nstepmax` integer default: 1000000

Maximum number of coarse time steps. The simulation stops when `nstep` reaches this value, even if the final output time has not been reached. Useful for short test runs.

`ordering` character default: `'hilbert'`

Domain decomposition ordering strategy.

`'hilbert'` Hilbert space-filling curve. Standard choice for moderate core counts ($\lesssim 1000$).

`'ksection'` K-section tree-based decomposition. Provides $O(k)$ message scaling (where k is the branching factor) for large core counts. Enables hierarchical MPI exchanges and memory-based load balancing (see `memory_balance`).

When `ordering='ksection'`, the communication pattern in ghost zone exchanges, multigrid solvers, and `build_comm` all switch to ksection tree routing automatically.

`memory_balance` logical default: `.false.`

Enable memory-based load balancing. When `.true.`, the bisection histogram weights each cell by its memory footprint (grid metadata + attached particles) instead of uniform cell count.

Requires `ordering='ksection'`.

The cell cost function is:

$$C_{\text{cell}} = \frac{\texttt{mem\_weight\_grid}}{\texttt{twotondim}} + n_{\text{part}} \times \frac{\texttt{mem\_weight\_part}}{\texttt{twotondim}}$$

where n_{part} is the number of particles attached to the parent grid. The weight parameters `mem_weight_grid` (default 270) and `mem_weight_part` (default 12) are set in the same namelist block.

Note: All histogram variables (`bisec_hist`, `bisec_cpu_load`, `cell_cost`) use 64-bit integers (`integer(i8b)`) and `MPI_INTEGER8` to avoid overflow at high particle counts.

`sink` logical default: `.false.`

Enable sink particle formation and evolution. Sink particles represent compact objects (e.g. black holes, protostars) that accrete gas from their surroundings. When active, cells exceeding a density threshold at `levelmax` can spawn sink particles.

See also `sink_AGN`, `bondi`, `Mseed` in Section 8.

`sinkprops` logical default: `.false.`

Output detailed sink particle properties (mass, position, velocity, accretion rate, spin) to dedicated files at each snapshot.

`lightcone` logical default: `.false.`

Enable lightcone output mode. When `.true.`, particles and/or cells crossing the observer's past lightcone are written to special output files during the simulation. See Section 9 for additional parameters.

`verbose` logical default: `.false.`

Enable verbose output during initialization and evolution. Prints additional diagnostics (grid counts, memory usage, load balance statistics) to standard output at each coarse step.

`jobcontrolfile` character(128) default: '' `(empty)`

Path to a runtime job control file. When set to a non-empty string, rank 0 reads this file at every coarse step to check for user-requested actions. The action is broadcast to all ranks via `MPI_BCAST`.

File format. Each line contains two integers separated by whitespace:

```
step_number  action_code
```

- `step_number = 0` — match immediately (every step).
- `step_number =` N — match when `nstep_coarse` $= N$.
- `action_code = 0` — do nothing.
- `action_code = 1` — write an extra snapshot and continue.
- `action_code = -1` — write an extra snapshot and stop gracefully (sets `nstepmax= nstep_coarse`).

If multiple lines match, `-1` (stop) takes priority over `1` (output). The file is *not* deleted after reading; lines with `step_number = 0` re-trigger every step until the file is removed or modified by the user.

Example. To request a graceful stop at the next coarse step:

```
echo "0 -1" > jobcontrol.txt
```

To schedule extra outputs at specific steps:

```
100 1
200 1
500 -1
```

Note: Output numbering (`ifout`) continues incrementally; extra outputs from job control use the next sequential number automatically.

1.1 GPU Acceleration

The following parameters control GPU acceleration. They require compilation with `make USE_CUDA=1`; when compiled without CUDA support, any `gpu_*` flags set to `.true.` are silently reset to `.false.` with a warning.

All three flags default to `.false.` (CPU-only mode). They can be combined independently to select the optimal configuration for the available hardware.

`gpu_hydro` logical default: `.false.`

Enable hybrid CPU/GPU hydrodynamics solver. When `.true.`, the Godunov solver dispatches large grid batches to the GPU while small batches are processed by CPU threads. The full mesh (`uold`, `f`, `son`) is uploaded to GPU memory once per `godunov_fine` call ($\sim$19 GB for a typical 512^3 run with 11 hydro variables).

Requires one MPI rank per GPU.

Note: On GPUs with limited PCIe bandwidth (e.g. NVIDIA A40 via PCIe Gen4), the mesh upload and gather/scatter overhead can exceed the GPU compute benefit, resulting in a net slowdown. Benchmark before enabling in production. The code automatically falls back to CPU-only if GPU memory is insufficient.

`gpu_poisson` logical default: `.false.`

Enable GPU-accelerated Poisson multigrid (MG) V-cycle for AMR levels. Gauss–Seidel red–black smoothing and residual computation run on the GPU; halo exchanges use pinned-memory asynchronous transfers.

The GPU MG solver uploads ∼6.4 GB of MG arrays (phi, f, flag2, nbor, igrid) plus ∼320 MB for GPU restrict/interpolation buffers.

Note: Like `gpu_hydro`, this feature is PCIe-bandwidth limited. Benchmarks on A40 (PCIe Gen4) show GPU MG is slower than CPU MG. On GPUs with high-bandwidth host–device links (e.g. Grace Hopper), GPU MG may provide a benefit.

`gpu_fft` logical default: `.false.`

Enable cuFFT direct Poisson solve for fully uniform AMR levels. When a level is fully covered (all $N = n_x \times n_y \times n_z \times 8^{\ell-1}$ grids exist), the Poisson equation is solved via 3D FFT instead of multigrid V-cycles. This bypasses iterative convergence entirely.

The solver flow is:

1. Gather local cell RHS values into a global 3D array via `MPI_ALLREDUCE`.
2. Upload to GPU and execute forward R2C FFT (cuFFT).
3. Apply the spectral Green's function $G(\mathbf{k}) = \Delta x^2/(N^3 L_k)$ where $L_k = 2\cos k_x + 2\cos k_y + 2\cos k_z - 6$.
4. Inverse C2R FFT and download result to host.
5. Scatter ϕ back to RAMSES cells.

GPU memory usage: ∼3 GB per rank for a 512^3 grid.

Performance: For a 512^3 base grid (4 MPI ranks, A40), the cuFFT direct solve reduces the base-level Poisson time from 242 s (MG V-cycle) to ∼20 s (**12× speedup**).

Note: This flag is independent of `gpu_poisson`. When both are `.false.`, all Poisson solving uses CPU multigrid. When only `gpu_fft=.true.`, the base level uses cuFFT on GPU while AMR levels use CPU multigrid — this is often the optimal configuration on PCIe-limited GPUs.

Warning: The current implementation uses `MPI_ALLREDUCE` to gather the global RHS array, which becomes a bottleneck for $N > 256^3$ (∼1 GB of data). For 256^3 and below, the overhead is negligible.

Examples:

```
! CPU-only (default, safest)
gpu_hydro   = .false.
gpu_poisson = .false.
gpu_fft     = .false.

! cuFFT base level only (recommended for PCIe GPUs)
gpu_fft     = .true.

! Full GPU acceleration (for NVLink/CXL GPUs only)
gpu_hydro   = .true.
gpu_poisson = .true.
gpu_fft     = .true.
```

2 `&AMR_PARAMS` – Adaptive Mesh Refinement

This mandatory block controls the AMR grid hierarchy, memory allocation sizes, and the simulation box geometry.

`levelmin` integer required

Minimum (base) AMR level. The base grid has $2^{\texttt{levelmin}}$ cells per dimension.

Example: `levelmin=7` produces a 128^3 base grid. `levelmin=9` produces a 512^3 base grid.

This level is fully covered – every cell at `levelmin` exists on exactly one MPI process.

`levelmax` integer required

Maximum AMR level. Determines the finest attainable resolution:

$$\Delta x_{\min} = \frac{L_{\text{box}}}{2^{\texttt{levelmax}}}$$

For cosmological zoom-in simulations, this controls the physical resolution at $z = 0$. The number of refinement levels beyond `levelmin` is `levelmax - levelmin`.

Refinement criteria (`m_refine`, `ivar_refine`) determine which cells actually refine up to this level.

`nexpand` integer array default: 1

Number of buffer (guard) cell layers per level to ensure smooth transitions between refinement levels. The i-th entry applies to level $\texttt{levelmin} + i - 1$. Typical value: `1` for all levels.

Larger values produce wider buffer zones around refined patches, improving solution quality at the cost of more cells.

`ngridtot` integer(i8b) required

Total number of AMR grids (octs) allocated across all MPI processes. Each process receives $\texttt{ngridmax} = \texttt{ngridtot}/\texttt{ncpu}$ grids. Each grid (oct) contains $2^{\texttt{ndim}}$ cells (8 cells in 3D).

Warning: RAMSES allocates full arrays at startup based on `ngridmax`. The virtual memory footprint is approximately $\texttt{ngridmax} \times 20$ bytes $\times\, \texttt{nvar}$. This must not exceed the system's `CommitLimit` (typically RAM $\times$ `overcommit_ratio`/100).

Rule of thumb: For N processes on a node with M GB of RAM,

$$\texttt{ngridtot} < \frac{M \times 0.5}{20 \times \texttt{nvar}} \times N$$

nparttot integer(i8b) required

Total particle allocation across all MPI processes. Each process gets **npartmax** = **nparttot/ncpu**. Should be at least 2× the total number of DM + star particles expected during the simulation (to accommodate load imbalance and new star particle creation).

Example:

```
! 100M DM particles, allow for stars
nparttot = 300000000
```

boxlen real(dp) default: 1.0

Box length in code units. For cosmological runs, the box size is typically read from the IC header (in h^{-1} Mpc) and this parameter is overridden. For non-cosmological (idealised) setups, `boxlen` defines the physical domain extent.

3 &OUTPUT_PARAMS – Snapshot Output

Controls when and how simulation snapshots are written to disk.

noutput integer default: 1

Number of output snapshots requested. The corresponding times or expansion factors must be listed in `tout` (non-cosmological) or `aout` (cosmological).

aout real(dp) array default: --

Scale factors at which to write output snapshots (cosmological mode only, i.e. when `cosmo=.true.`). The array must contain `noutput` entries, in ascending order.

Example:

```
noutput = 4
aout    = 0.1, 0.2, 0.5, 1.0
! Outputs at z = 9, 4, 1, 0
```

tout real(dp) array default: --

Output times in code units (non-cosmological mode). The array must contain `noutput` entries, in ascending order.

foutput integer default: 1000000

Write an output snapshot every `foutput` coarse time steps, regardless of the `aout`/`tout` schedule. Useful for periodic checkpointing in long runs. Set to a very large number to effectively disable.

outformat character default: 'original'

Output file format for snapshots.

'original'	Standard RAMSES per-CPU binary format. Each MPI process writes separate files (amr_NNNNN.outNNNNN, hydro_NNNNN.outNNNNN, etc.).
'hdf5'	Single HDF5 file per snapshot (data_NNNNN.h5). Uses MPI parallel I/O for collective writes. The HDF5 file stores all AMR, hydro, gravity, particle, and sink data in a hierarchical group structure. **Requires compilation with** make HDF5=1.

Note: The standard auxiliary files (info_NNNNN.txt, header_NNNNN.txt, compilation.txt, makefile.txt, namelist.txt) are always written regardless of outformat.

informat character default: 'original'

Input (restart) file format.

'original'	Read from standard per-CPU binary files. The number of MPI processes must match the run that produced the checkpoint.
'hdf5'	Read from the single HDF5 file (data_NNNNN.h5). Currently requires the same number of MPI processes as the original run. **Requires compilation with** make HDF5=1.

Note: informat and outformat can be set independently, allowing cross-format conversion (e.g. restart from binary and output to HDF5, or vice versa).

4 &INIT_PARAMS – Initial Conditions

Specifies the format and location of initial condition files.

filetype character default: 'grafic'

Initial condition file format.

'grafic'	GRAFIC2 binary format (Bertschinger 2001). Each level's IC directory contains binary files for density perturbations, velocities, and (optionally) particle displacements.
'ascii'	Text-based initial conditions (for simple test problems).

initfile character array default: --

Paths to IC directories, one per AMR level. initfile(1) corresponds to levelmin, initfile(2) to levelmin + 1, and so on.

Each directory must contain the following binary files:

- ic_deltab – baryon density perturbation field
- ic_velbx, ic_velby, ic_velbz – baryon velocity fields

- `ic_velcx`, `ic_velcy`, `ic_velcz` – dark matter (CDM) velocity fields
- `ic_poscx`, `ic_poscy`, `ic_poscz` – dark matter displacement fields (optional, for multi-level zoom-in)
- `ic_tempb` – baryon temperature perturbation (optional)
- `ic_pvar_00001`, ... – passive scalar fields (optional, for zoom-in refinement tagging; see `ivar_refine`)
- `ic_refmap` – refinement map (optional)

Example:

```
initfile = '/data/IC/level_07'
         , '/data/IC/level_08'
         , '/data/IC/level_09'
```

5 `&REFINE_PARAMS` – Refinement Criteria

Controls which cells are refined in the AMR hierarchy. These parameters are **critical for zoom-in simulations**, where background regions must remain coarse while the zoom region refines to high resolution.

`m_refine` real(dp) array default: -1

Quasi-Lagrangian mass threshold per level. The i-th entry applies to level $\texttt{levelmin}+i-1$. A cell is flagged for refinement when the effective mass indicator $\phi \geq \texttt{m\_refine}(i)$.

Typical value: **8.0** for all levels (refine when the equivalent of ≥ 8 particles occupies a cell). Provide one entry for each level from `levelmin` to `levelmax`.

Interacts with `ivar_refine` and `mass_cut_refine` to determine which particles contribute to the density used for the refinement decision.

Example:

```
! 6 levels of refinement (levelmin=7, levelmax=13)
m_refine = 8., 8., 8., 8., 8., 8., 8.
```

`ivar_refine` integer default: -1

Variable index controlling the refinement criterion in `poisson_refine`. This parameter fundamentally changes how refinement regions are selected:

`ivar_refine = 0`:
Use `cpu_map2` for refinement control. During initialization, `cpu_map2` is set by `init_refmap` from `ic_refmap` (if present); during evolution, it is updated by `rho_fine` based on the local density field. This is the standard quasi-Lagrangian approach.

Warning: In zoom-in simulations, this can cause uncontrolled AMR expansion into background regions if `cpu_map2` is not properly restricted by `mass_cut_refine`.

`ivar_refine > 0` (e.g. 11):
During initialization, use passive scalar criterion: `uold(cell, ivar_refine) / uold(cell, 1)` $>$ `var_cut_refine`.

Recommended for zoom-in: set `ivar_refine=NVAR` (the last hydro variable), and create `ic_pvar_NNNNN` files with value 1.0 inside the zoom region and 0.0 in the background. After initialization, `cpu_map2` (set by `rho_fine` with `mass_cut_refine` filtering) takes over.

`ivar_refine < 0` (default):
Pure density-based refinement at both initialization and runtime. A cell is refined when `uold(cell, 1)` $\geq$ `m_refine` $\times\, m_{\rm sph}/V_{\rm cell}$.

`var_cut_refine`

real(dp) default: -1

Threshold for passive-scalar-based refinement when `ivar_refine` > 0. A cell is refined only if

$$\frac{\texttt{uold}(\text{cell},\ \texttt{ivar\_refine})}{\texttt{uold}(\text{cell},\ 1)} > \texttt{var\_cut\_refine}$$

Typical value: **0.01** for zoom geometry tagging (the passive scalar is 1.0 inside the zoom region, 0.0 outside).

`mass_cut_refine`

real(dp) default: -1

Particle mass threshold for quasi-Lagrangian refinement. In `rho_fine`, dark matter particles with mass $\geq$ `mass_cut_refine` are *excluded* from the density computation that drives cell refinement. This prevents heavy (coarse-level) background particles from triggering spurious refinement.

Set this to the DM particle mass at the finest IC level. Reference values for a $100\,h^{-1}$ Mpc box ($\Omega_m = 0.3$, $h = 0.68$):

IC finest level	mass_cut_refine
8	1.19209e-07
9	1.49012e-08
10	1.86265e-09
11	2.32831e-10
12	2.91038e-11
13	3.63798e-12

Note: This parameter interacts with `ivar_refine` and `m_refine`. All three should be set consistently for zoom-in simulations.

`interpol_var`

integer default: 0

Interpolation variable type used when prolongating (interpolating) data from coarse to fine grids.

- `0` Conservative variables (ρ, ρv, E).
- `1` Primitive variables (ρ, v, P). **Recommended** for cosmological simulations to avoid interpolation artefacts in low-density regions.

`interpol_type` integer default: 1

Interpolation slope limiter for prolongation.

`0` MinMod limiter – more diffusive, more robust.

`1` MonCen (monotonised central) limiter – less diffusive. **Recommended**.

`sink_refine` logical default: `.false.`

Force maximum refinement around sink particles. When `.true.`, a contribution equal to `m_refine` is added to the refinement indicator ϕ for every cell containing a sink particle, ensuring refinement up to `levelmax`.

`jeans_ncells` real(dp) default: -1

Jeans refinement criterion. If > 0, cells are refined to resolve the local Jeans length by at least this many cells:

$$\Delta x < \frac{\lambda_J}{\texttt{jeans\_ncells}}$$

Enabling this also activates a polytropic equation-of-state floor to prevent artificial fragmentation (Truelove criterion). Typical value: **4** (minimum of 4 cells per Jeans length).

6 `&HYDRO_PARAMS` – Hydrodynamics Solver

Controls the gas dynamics solver configuration.

`gamma` real(dp) default: $5/3$

Adiabatic index γ of the ideal gas equation of state, $P = (\gamma - 1)\rho e$. Standard value: $5/3$ for a monatomic ideal gas. Use $7/5$ for diatomic gas or $4/3$ for radiation-dominated flow.

`courant_factor` real(dp) default: `0.8`

Courant–Friedrichs–Lewy (CFL) number for time step control. The time step at each level is $\Delta t = \texttt{courant\_factor} \times \Delta x / v_{\max}$. Typical: **0.8**. Lower values increase stability at the cost of more time steps.

`scheme` character default: `'muscl'`

Hydrodynamics integration scheme.

`'muscl'` MUSCL–Hancock (Monotonic Upstream-centred Scheme for Conservation Laws), second-order in space and time. This is the only production scheme in RAMSES.

`slope_type`　integer　default: 1

Slope limiter for MUSCL piecewise-linear reconstruction.

`1`	MinMod – most robust, more diffusive.
`2`	MonCen – monotonised central, less diffusive. **Recommended** for production runs.
`3`	Unlimited – no limiting (unstable; testing only).

`riemann`　character　default: `'llf'`

Approximate Riemann solver for inter-cell flux computation.

`'llf'`	Local Lax–Friedrichs (Rusanov). Most diffusive but unconditionally stable. Good default.
`'hll'`	Harten–Lax–van Leer. Two-wave solver.
`'hllc'`	HLL with Contact restoration. Three-wave solver, most accurate for contact discontinuities. **Recommended for cosmological simulations**.
`'exact'`	Exact Riemann solver (expensive; primarily for validation).

`pressure_fix`　logical　default: `.false.`

Enable pressure floor to prevent negative pressures in strong shocks or highly supersonic flows. When the internal energy becomes negative, RAMSES falls back to a pressure estimate from the total energy.

Recommended: `.true.` for cosmological simulations. See also `beta_fix`.

`beta_fix`　real(dp)　default: `0.0`

Pressure fix parameter. Controls the magnitude of the pressure floor: $P_{\text{floor}} =$ `beta_fix` $\times \rho v^2/2$. Typical value: **0.5** when `pressure_fix=.true.`

`isothermal`　logical　default: `.false.`

Isothermal mode. When `.true.`, the energy equation is not solved and the gas temperature remains constant. Reduces the number of hydro variables by one.

7 &POISSON_PARAMS – Gravity Solver

Controls the multigrid Poisson solver for self-gravity.

`epsilon`　real(dp)　default: 10^{-4}

Multigrid convergence criterion. The V-cycle iteration at each level stops when the residual norm satisfies $\|r\|/\|r_0\| <$ `epsilon`. Typical value for cosmological runs: 10^{-5} to 10^{-4}. Tighter values improve force accuracy but increase iteration count.

`gravity_type` integer default: 0

Gravity model selection.

`0`	Self-gravity (solve Poisson equation on the AMR grid).
`>0`	Analytical gravitational potential (e.g. for test problems with known solutions). The integer value selects the specific analytical profile.

`cg_levelmin` integer default: 999

Minimum level at which the conjugate gradient (CG) fallback solver activates. When the multigrid solver stalls at high AMR levels, CG provides guaranteed convergence. Set to `levelmax` for best convergence behaviour.

Typical: `cg_levelmin = levelmax`. The default (999) means CG is effectively disabled unless `levelmax` is absurdly large.

`cic_levelmax` integer default: 0

Maximum level for cloud-in-cell (CIC) particle mass deposition.

- `0` – deposit particles at all levels (standard).
- $N > 0$ – deposit particles only up to level N; finer levels inherit the coarse density by prolongation.

Rarely modified.

8 &PHYSICS_PARAMS – Sub-grid Physics

Controls cooling, star formation, stellar/AGN feedback, and cosmological parameters. This block is optional; omit it entirely for adiabatic (non-radiative) simulations.

8.1 Cooling and UV Background

`cooling` logical default: `.false.`

Enable radiative cooling with a metal-dependent cooling function. When `.true.`, RAMSES integrates the cooling/heating rate at each time step using tabulated cooling curves. Requires `hydro=.true.`

`haardt_madau` logical default: `.false.`

Enable the Haardt & Madau (2012) ultraviolet background model for cosmic reionization. Provides a redshift-dependent photo-heating and photo-ionization rate. Used together with `cooling`.

`z_reion` real(dp) default: `8.5`

Reionization redshift. Hydrogen reionization heating is applied instantaneously at this redshift. Typical range: 6–10, depending on the reionization model.

`z_ave` real(dp) default: `0.0`

Initial mean metallicity of the gas in solar units ($Z_\odot$). Applied uniformly at initialization. Use `0.0` for primordial composition.

`delayed_cooling` logical default: `.false.`

Delay radiative cooling in supernova-heated gas to prevent overcooling. When a cell receives SN energy, cooling is suppressed for a duration related to the Sedov–Taylor phase. Improves the effectiveness of stellar feedback in regulating star formation.

`tol` real(dp) default: 10^{-3}

Tolerance for the implicit cooling solver. The Newton–Raphson iteration converges when the relative temperature change $|\Delta T/T| <$ `tol`.

8.2 Star Formation

`n_star` real(dp) default: `0.1`

Star formation hydrogen number density threshold in H cm^{-3}. Only gas denser than this value is eligible for star formation. Typical range: 0.1–10.

`eps_star` real(dp) default: `0.0`

Star formation efficiency per free-fall time $\epsilon_{\rm ff}$. The star formation rate density is $\dot\rho_\star = \epsilon_{\rm ff}\,\rho_{\rm gas}/t_{\rm ff}$. Typical value: **0.01–0.02** (1–2% per free-fall time). Set to `0.0` to disable star formation entirely.

`del_star` real(dp) default: 200

Overdensity threshold for star formation (in units of the cosmic mean density). Gas must exceed $\delta >$ `del_star` in addition to the density threshold `n_star`.

`m_star` real(dp) default: -1

Minimum stellar particle mass in code units. When a star-forming cell would produce a particle below this mass, the event is stochastically deferred to the next time step.

- < 0: use the cell gas mass (no minimum).
- > 0: explicit minimum mass.

T2_star real(dp) default: 0

ISM polytropic equation-of-state temperature floor in Kelvin. Gas above the star-formation density threshold `n_star` follows a polytropic relation:

$$T = T_{2,\star}\left(\frac{n}{n_\star}\right)^{\gamma_\star - 1}$$

where $\gamma_\star$ is `g_star`. This prevents artificial fragmentation below the resolution limit (Jeans mass floor).

g_star real(dp) default: 1.6

Polytropic index $\gamma_\star$ for the ISM equation of state (see `T2_star`). Typical value: **1.6** (stiff polytrope) or **5/3** (adiabatic floor).

8.3 Stellar Feedback

f_w real(dp) default: 0

Mass loading factor for supernova-driven winds. The wind mass flux is $\dot{M}_w =$ `f_w` $\times \dot{M}_\star$. Set to `0` to disable winds. Typical range: 1–5.

f_ek real(dp) default: 1.0

Kinetic energy fraction of supernova feedback. Controls the partition between kinetic (`f_ek`) and thermal ($1 -$ `f_ek`) energy injection. `f_ek=1` is purely kinetic feedback; `f_ek=0` is purely thermal.

eps_sn1 real(dp) default: 0

Type Ia supernova energy per event in units of 10^{51} erg. Set to `0` to disable Type Ia SN feedback.

eps_sn2 real(dp) default: 0

Type II supernova energy per event in units of 10^{51} erg. Set to `0` to disable Type II SN feedback.

yieldtablefilename character default: --

Path to the chemical yield table file for metal enrichment calculations. Required when metal-dependent cooling or chemical evolution tracking is enabled.

8.4 Cosmological Parameters

omega_b real(dp) default: --

Baryon density parameter Ω_b. Overrides the value read from the IC file header. Must be consistent with the initial conditions and other cosmological parameters (Ω_m, H_0, etc. are read from the GRAFIC2 header).

8.5 AGN and Sink Particle Parameters

Mseed real(dp) default: --

Seed black hole mass in solar masses ($M_\odot$). When a sink particle forms, it is initialised with this mass. Typical range: 10^4–10^6 $M_\odot$ for cosmological simulations.

sink_AGN logical default: `.false.`

Enable AGN feedback from sink particles. When `.true.`, sink particles inject thermal and/or kinetic energy into their surroundings based on their accretion rate. Requires `sink=.true.`

bondi logical default: `.false.`

Enable Bondi–Hoyle–Lyttleton accretion for sink particles. The accretion rate is computed from the local gas density, sound speed, and relative velocity:

$$\dot{M} = \frac{4\pi G^2 M_{\rm BH}^2 \rho}{(c_s^2 + v_{\rm rel}^2)^{3/2}}$$

Can be boosted by `boost_acc`.

drag logical default: `.false.`

Enable dynamical friction on sink particles. Applies a drag force opposing the sink's motion relative to the background gas. Strength can be amplified by `boost_drag`.

rAGN real(dp) default: --

AGN feedback energy injection radius in units of the cell size at `levelmax`. Feedback energy is distributed over a sphere of this radius centred on the sink particle.

X_floor real(dp) default: --

Hydrogen mass fraction floor. Prevents the hydrogen fraction from dropping below this value due to numerical artefacts. Typical: `0.76`.

eAGN_K real(dp) default: --

AGN kinetic feedback efficiency ϵ_K. Fraction of the accreted rest-mass energy deposited as kinetic energy: $\dot{E}_K = \epsilon_K \dot{M} c^2$.

eAGN_T real(dp) default: --

AGN thermal feedback efficiency ϵ_T. Fraction of the accreted rest-mass energy deposited as thermal energy: $\dot{E}_T = \epsilon_T \dot{M} c^2$.

TAGN real(dp) default: --

AGN heating temperature in Kelvin. The AGN thermal energy is deposited by raising gas temperature toward this value within the feedback region `rAGN`.

r_gal real(dp) default: --

Galaxy definition radius for AGN feedback, in code units. Used to compute the local galaxy properties (stellar mass, gas mass) around a sink particle for AGN mode switching.

T2maxAGN real(dp) default: --

Maximum AGN heating temperature in Kelvin. Caps the temperature increase from a single AGN feedback event to prevent unphysically hot gas.

boost_acc real(dp) default: --

Bondi accretion boost factor. Multiplies the Bondi–Hoyle accretion rate by this factor to compensate for unresolved gas structure near the black hole. Typical range: 1–100. Requires `bondi=.true.`

boost_drag real(dp) default: --

Dynamical friction drag boost factor. Multiplies the drag force by this factor. Requires `drag=.true.`

vrel_merge logical default: --

Use relative velocity criterion for sink particle merging. When `.true.`, two sinks merge only if their relative velocity is below the local escape velocity, in addition to the spatial proximity criterion `rmerge`.

rmerge real(dp) default: --

Sink merging radius in units of the cell size at `levelmax`. Two sink particles closer than this distance are candidates for merging (subject to additional criteria if `vrel_merge=.true.`).

spin_bh logical default: --

Track black hole spin evolution. When `.true.`, the code evolves the dimensionless spin parameter $a_\star$ of each sink particle based on the angular momentum of accreted gas.

mad_jet logical default: --

Enable the magnetically arrested disk (MAD) jet model. When `.true.`, AGN kinetic feedback is launched as a collimated bipolar jet aligned with the black hole spin axis. Requires `spin_bh=.true.`

9 &LIGHTCONE_PARAMS – Lightcone Output

Parameters for lightcone output mode (activated when `lightcone=.true.`). In this mode, particles and/or cells that cross the observer's past lightcone during each time step are written to special output files, enabling the construction of mock galaxy surveys and weak-lensing maps without storing full snapshots.

Configuration parameters include the observer position, opening angle, and selection criteria. Consult the RAMSES lightcone documentation for the full parameter list, which varies by application.

10 &SPHERICAL_REGION_PARAMS

spherical_region logical default: `.false.`

Enable a spherical refinement region. When `.true.`, AMR refinement is restricted to a spherical sub-volume of the simulation box. This is useful for re-simulations of specific halos where a cubic zoom region is not optimal. Additional parameters define the centre and radius of the sphere.

11 &COSMO_PARAMS – Cosmological Parameters

Optional block for overriding cosmological parameters read from the IC file header and for specifying CPL dark energy equation-of-state parameters. When this block is absent, default values are used ($w_0 = -1$, $w_a = 0$, which recovers the standard ΛCDM model).

omega_b real(dp) default: `0.0`

Baryon density parameter Ω_b. Overrides the value set by `&PHYSICS_PARAMS` or the IC header.

omega_m real(dp) default: `1.0`

Total matter density parameter Ω_m. Read from the IC header by default; this namelist entry overrides it.

`omega_l` real(dp) default: `0.0`

Dark energy density parameter $\Omega_{\rm de}$ (equivalent to Ω_Λ for ΛCDM). Read from the IC header by default; this namelist entry overrides it.

`h0` real(dp) default: `1.0`

Hubble constant H_0 in $\mathrm{km\,s^{-1}\,Mpc^{-1}}$. Read from the IC header by default; this namelist entry overrides it.

`w0` real(dp) default: -1

Present-day dark energy equation-of-state parameter w_0 in the CPL (Chevallier–Polarski–Linder) parametrization:

$$w(a) = w_0 + w_a\,(1-a)$$

where a is the scale factor.

Setting $w_0 = -1$ and $w_a = 0$ (the defaults) exactly recovers the cosmological constant Λ (i.e. ΛCDM), producing **bit-identical** results to the code before the CPL extension.

The dark energy density evolves as

$$\rho_{\rm de}(a) = \rho_{\rm de,0}\; a^{-3(1+w_0+w_a)}\; \exp[-3\,w_a\,(1-a)]$$

This modifies the Friedmann equation, Hubble rate $H(a)$, linear growth factor $D_1(a)$, and growth rate $f(a)$.

Note: w_0 and w_a are *not* stored in the GRAFIC2 IC header (which has a fixed 44-byte layout). They must be specified in this namelist block. MUSIC generates CPL transfer functions internally but does not write w_0, w_a to its GRAFIC2 output.

`wa` real(dp) default: 0

Time-variation parameter w_a of the CPL dark energy equation of state. See `w0` for the full parametrization. Typical observational constraints give $w_a \in [-1, 1]$.

`cs2_de` real(dp) default: 0

Dark energy sound speed squared c_s^2 (in units of c^2). This parameter is declared for future use in dark energy perturbation theory (clustering dark energy) but is **not yet used** in the current code. Set to 0 for the standard smooth dark energy assumption.

12 Complete Example: Cosmological Zoom-In

The following namelist illustrates a production cosmological zoom-in simulation with dark matter particles, baryonic gas, self-gravity, cooling, star formation, and AGN feedback.

Listing 1: Cosmological zoom-in namelist

```
&RUN_PARAMS
cosmo    = .true.
pic      = .true.
poisson  = .true.
hydro    = .true.
sink     = .true.
nrestart = 0
nremap   = 5
nsubcycle = 1, 1, 1, 2, 2, 2, 2
nstepmax = 10000000
ordering = 'ksection'
memory_balance = .true.
jobcontrolfile = 'jobcontrol.txt'
/

&AMR_PARAMS
levelmin  = 7
levelmax  = 18
nexpand   = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1
ngridtot  = 400000000
nparttot  = 600000000
/

&OUTPUT_PARAMS
noutput = 10
aout = 0.05, 0.1, 0.15, 0.2, 0.3, 0.4, 0.5, 0.7, 0.85, 1.0
foutput = 500
/

&INIT_PARAMS
filetype = 'grafic'
initfile = '/data/IC/level_07'
         , '/data/IC/level_08'
         , '/data/IC/level_09'
         , '/data/IC/level_10'
         , '/data/IC/level_11'
         , '/data/IC/level_12'
         , '/data/IC/level_13'
/

&REFINE_PARAMS
m_refine = 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8., 8.
ivar_refine      = 11
var_cut_refine   = 0.01
mass_cut_refine = 3.63798e-12
interpol_var     = 1
interpol_type    = 1
/

&HYDRO_PARAMS
gamma          = 1.6666667
```

```
courant_factor = 0.8
scheme         = 'muscl'
slope_type     = 2
riemann        = 'hllc'
pressure_fix   = .true.
beta_fix       = 0.5
/

&POISSON_PARAMS
epsilon      = 1.0e-5
gravity_type = 0
cg_levelmin  = 18
/

&PHYSICS_PARAMS
cooling        = .true.
haardt_madau   = .true.
z_reion        = 8.5
n_star         = 0.1
eps_star       = 0.02
T2_star        = 1.0e4
g_star         = 1.6
del_star       = 200.0
f_ek           = 1.0
sink_AGN       = .true.
bondi          = .true.
Mseed          = 1.0e5
/
```

13 Parameter Cross-Reference Index

Table 1 lists parameters that commonly interact and should be set consistently.

Table 1: Cross-reference of interacting parameters.

Parameter	**Related Parameters**	**Notes**
`cosmo`	`aout`, `omega_b`	Cosmological mode requires scale-factor outputs
`pic`	`poisson`, `nparttot`	Particles need gravity and memory allocation
`ordering`	`memory_balance`	Memory balancing requires `ksection`
`levelmax`	`m_refine`, `cg_levelmin`	Set `cg_levelmin = levelmax`
`ivar_refine`	`var_cut_refine`, `mass_cut_refine`, `m_refine`	All must be consistent for zoom-in
`mass_cut_refine`	`ivar_refine`	Set to finest-level DM particle mass
`pressure_fix`	`beta_fix`	`beta_fix` only effective when fix is on
`T2_star`	`g_star`, `n_star`	Polytropic EOS parameters
`sink`	`sink_AGN`, `bondi`, `Mseed`	AGN feedback requires sink particles

Parameter	Related Parameters	Notes
sink_AGN	eAGN_K, eAGN_T, TAGN, rAGN	AGN feedback parameters
bondi	boost_acc	Boost factor for unresolved accretion
drag	boost_drag	Drag boost factor
spin_bh	mad_jet	MAD jet requires spin tracking
cooling	haardt_madau, z_reion	UV background for reionization heating
ngridtot	nparttot	Both determine per-process memory usage