ISSN 2959-6157

Comparing Performance in the Codes AREPO, GIZMO and GAMER-2

Wuhan Xu[1, *]

[1]Kangchiao International School East China Status, Suzhou, China
*Corresponding author: 14335@kcisg.com

Abstract:

Modern astrophysical simulation is the main aspect of human getting insight from the cosmos. Numerical simulations became irreplaceable tools for generating various phases of the universe, the formation of galaxies, and other astrophysical phenomena. As the advances of computational power, validity and diversity of astrophysical modeling have been growing. In addition to the development of the processor performance, the diversity of the software community is also rising therefore, research between various simulation codes is worthwhile. It allowed researchers to know the nature of different simulation techniques linked to their basic algorithms. This study summarizes the basics of common simulation methods used by researchers including N-body Simulations, Smoothed Particle Hydrodynamics, Monte Carlo Methods, Finite Difference and Finite Element Methods. Furthermore, this research analyzes the basics of AREPO, GIZMO, and GAMER-2 codes, and compared the performance of these codes. Last, according to the analysis, this study discusses and demonstrates the advantages and limitations of these codes. These results shed light on guiding further exploration of cosmology simulations.

Keywords: Cosmology simulations; particle algorithms; AREPO; GIZMO; GAMER-2.

1. Introduction

In astrophysics, theoretical anticipation is largely driven by numerical simulations of astrophysical processes. It is crucial for advancing human understanding of the universe by modeling different phases of the universe, which are challenging to observe by human timescales. The foundations of astrophysical modeling were laid with the work of scientists like Isaac Newton, who formulated the laws of motion and universal gravitation. These principles allowed for the prediction of planetary movements, initiating the field of celestial mechanics. In the early 20th century, models of stellar structure and evolution were developed. Arthur Eddington's work on the internal structure of stars was pivotal. He proposed that stars are in hydrostatic equilibrium, balancing gravitational collapse with pressure from nuclear fusion. The discovery of nuclear reactions as the energy source of stars was significant. Hans Bethe's 1939 paper on energy production in stars marked a key advancement in understanding stellar processes [1].

The advent of computers revolutionized astrophysical modeling. Numerical simulations became essential tools for solving complex equations governing star formation, galaxy dynamics, and cosmology. The development of the Big Bang theory and cosmic microwave background studies led to advanced cosmological models. Recent projects like IllustrisTNG and THESAN have provided insights into how galaxies evolve over time, accurately reproducing observed galaxy properties such as morphology, size, and color distribution [2, 3]. These models incorporate complex processes like star formation, the process of reionization, and black holes. Simulations of the universe's large-scale structure have successfully replicated the cosmic web's filamentary structure observed in galaxies. Simulations have predicted gravitational wave signatures from black hole mergers, which have been confirmed by LIGO and Virgo observations [4]. These models help understand the masses and spins involved in such events.

The purpose of this essay is to compare the performance of different astrophysical codes which is significant since it helps researchers deepen the understanding of different aspects and applicability of various astrophysical codes. Thus, researchers can ensure accuracy and reliability in simulations. Each code has its strengths and weaknesses, influenced by their numerical algorithms and performance optimization. By comparing them, researchers can validate results across different approaches, identify the most suitable tools for specific simulation goals. Additionally, such comparisons can highlight differences in computational efficiency, which is important for understanding different model including their time step, parallelization and particle solver etc. This study will mention some basic introduction to astrophysical simulation and some classical simulation methods, introduce the principle of simulation methods, the implementation of numerical schemes and typical results, compare different simulations, introduce the most advanced simulation results and some application scenarios, and summarize the limitations of all current simulations and the future prospects.

2. Key Simulation Methods

N-body simulations are fundamental for studying gravitational interactions among multiple bodies. They are used extensively to understand the dynamical evolution of star clusters, galaxies, and dark matter distributions. The primary challenge is the computational cost, which increases with the number of bodies. Recent advancements, such as the use of tree algorithms and parallel computing, have improved their efficiency. The gravitational force calculation remains central to these simulations.ad Because gravity is a long-range force, the calculation of the gravity for each particular particle in the simulation is dependent upon the masses and positions of all other particles. N^2 is the lowest efficiency case that assesses the gravity pairwise for every particle. The most advanced N-body simulations are performed on clusters with thousands of compute nodes and can involve up to a trillion particles. Running the naïve pairwise technique on simulations this large would simply not be possible. Rather, N-body codes use a spatial acceleration of some kind to translate the gravity computation into an N log N calculation [5].

Smoothed Particle Hydrodynamics (SPH) is a powerful and versatile tool for simulating fluid dynamics, especially applicable to problems in which turbulent and complex flows (e.g., in a star forming cloud or in supernova explosions) are more prevalent. SPH has been modified and improved upon with respect to the classical approach, so it is currently adequate for many applications such as accretion disks and mergers of stars.

Monte Carlo simulations are employed to tackle problems with high degrees of randomness or complexity, such as radiative transfer and particle interactions in astrophysical plasmas. These methods provide statistical sampling solutions to problems where direct calculation is impossible.

Finite Difference and Finite Element Methods are used to find solutions to differential equations that describe physical processes, such as heat conduction in stars or the movement of fluid in accretion discs. The differential equations are discretised in space on a grid, and then the equations are evolved in time. More recent simulation techniques include adaptive mesh refinement, which will increase the resolution in the regions of interest without significant extra computational cost.

3. Descriptions of the Code

3.1 GIZMO

GIZMO is a largely paralleled astrophysical simulation code that is based on meshless Lagrangian Godunov-type approach [6-8]. The approach described by Rocha et al. is based on the DM macro-particles' rate of scattering in phase space is applied by GIZMO in its implementation of elastic self-interactions [8]. The probability of an interaction is calculated as:

$$P_{ij} = (\sigma / m) m_i v_{ij} g_{ij}^{\delta t} \tag{1}$$

where m_i is the mass of the macroparticle, v_{ij} is the relative velocity difference between the two particles, and g_{ij} is the number density factor that accounts for smoothing kernel overlap between the two macroparticles [9]. The code includes support for both ideal and non-ideal MHD, allowing for the simulation of complex magnetic interactions in astrophysical fluids. It incorporates features like Ohmic resistivity, ambipolar diffusion, and the Hall effect. GIZMO facilitates the simulation of star formation processes and black hole dynamics, including on-the-fly formation and seeding based on user-defined criteria. It also models feedback mechanisms that influence surrounding gas dynamics. The code includes pre-built libraries for various cooling processes, e.g., photo-ionization and molecular cooling, enabling realistic thermal dynamics in simulations [10].

GIZMO employs a hybrid MPI+OpenMP parallelization strategy, enabling it to efficiently scale on massively parallel systems, handling problem sizes with billions of resolution elements. It is designed as a multi-physics code which include various modules that support hydrodynamics, magneto-hydrodynamics (MHD), star formation, black hole dynamics, and more. This modularity allows for tailored simulations that can address specific astrophysical questions. The code supports multiple numerical methods for fluid dynamics, including Lagrangian mesh-free finite-volume methods, SPH, and fixed-grid approaches. This flexibility enables users to choose the most appropriate method for their simulation's characteristics, such as the need for high resolution in certain regions or the handling of complex geometries. In addition to its diversity, GIZMO employs hierarchical adaptive time-stepping, which allows it to efficiently manage simulations with varying dynamical timescales [10].

3.2 GAMER-2

Gamer-2 is an improved version of Gamer-1 that comes with additional features like bitwise reproducibility, memory pool, multiple hydrodynamic schemes, and radiative processes with the grackle library. Significant advancements in scalability, accuracy, stability, and performance are also included. Gamer-2 has involved numerical algorithms comprising particle integration, the AMR structure, gravity and hydrodynamic solvers, and other characteristics. Gamer-2 employs several performance optimization techniques, such as GPU implementation, hybrid parallelization, and memory management [11].

3.3 AREPO

AREPO is a massively paralleled astrophysical simulation code that uses a finite-volume approach on a moving mesh approach for gravitational interactions [12]. Arepo discretizes the 3D space into cells to simplify calculations. The cells are constructed by Voronoi tessellation, forming a mesh. One considers the fluxes between the cells and the average variables of each cell [11, 13]. Voronoi tessellation uses mesh-generating points to create cells. The area closest to a mesh-generating point relative to all other points is called its cell. An ensemble of tetrahedra is formed by the connections between neighboring mesh-generating points (also known as Delaunay triangulation). Its unique characteristic is that each tetrahedron's circumsphere is devoid of any additional mesh-generating points. To construct a mesh, as shown in follows:

l Begin with four points or a tetrahedron's corners.

- Insert into the tetrahedron a mesh-generating point.
- Make connections between the tetrahedron's points and corners.
- As seen in Figure 1, divide the tetrahedron into four smaller ones.
- Continue for every tetrahedron.

However, the resulting tetrahedra may violate the property of Delaunay triangulation. To restore it, one iteratively applies edge-flipping to affected tetrahedra before inserting the next point. Every mesh-generating point in the simulation has the ability to move, and the overall motion of gas (or another material) in its cell will have an impact on its acceleration and velocity. Every time step, the motion is updated. Whereas, the motion of the gas and the mesh is separate. Additionally, a cell's position, size, and volume can all fluctuate. Nevertheless, each cell must maintain its mass and size within a predetermined range, which can be done in a few different ways, including regularization, (de-)refinement, and velocity corrections. In practical simulations, the algorithm uses a hybrid of particle-mesh and oct-tree, or a particle-particle particle-mesh, for diverse types of interaction. The technique employs a mix of particle-mesh and oct-tree, or a particle-particle particle-mesh, for various forms of interaction in real-world simulations. A sketch is shown in Fig. 1 [13].

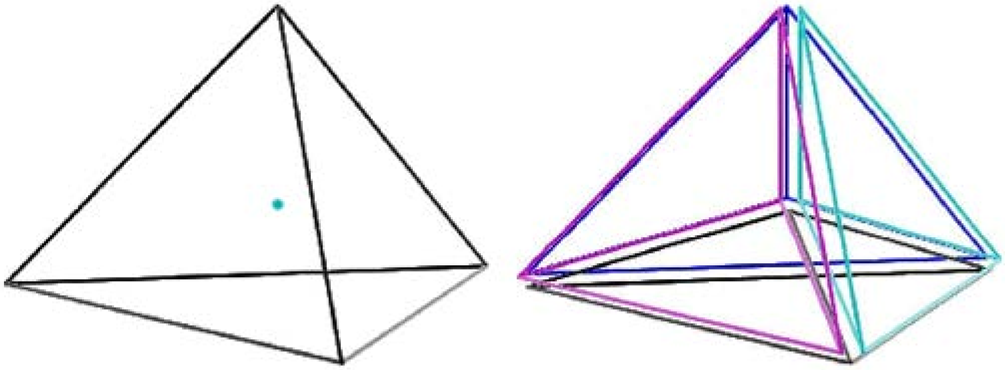

Fig. 1 Divides the tetrahedron into four smaller ones [13].

A "distorted" mesh may aggravate the constraints on time-step and limit the spatial resolution. Therefore, mesh regularization is used to solve this problem, having zones of similar gas characteristics represented with cells of comparable size [13, 14]. To achieve this, one uses a modified Lloyd algorithm: At each time step, for a certain cell, a velocity addition is imposed on the mesh-generating point, bringing the point closer to the geometric center. However, there is a problem. If one applies this correction to a star-forming gas, the systematic differences among masses of cells cause the mesh to move away from the densest regions (e.g. central part of a galaxy) but leave the gas there. Hence, the mass per cell in these regions will surge, decreasing the mass resolution. This betrays the expectations. There are two solutions, i.e., to lessen the intensity of the regularization settings; or to increase the regularization condition's tolerance for abrupt changes. When there are significant density gradients present, this prevents unintentional mesh correction from occurring. Additionally, the refinement and de-refinement of cells is another procedure to reserve an expected resolution. If a cell's mass is too great, one splits it (refinement), and if two cells' masses are too tiny, one converge (de-refinement).

One defines two state vectors:

$$U = \begin{pmatrix} \rho \\ \rho v \\ \rho e \end{pmatrix}, F = \begin{pmatrix} \rho v \\ \rho v v^T + P \\ \rho e v + P v \end{pmatrix} \quad (2)$$

where ρ, *v*, e are density, velocity and energy density of the particles at a point [13]. For the free transport case, where source minus sink equals 0, the continuity equation can be rewritten as:

$$\frac{\partial U}{\partial t} + \nabla \cdot F = 0 \quad (3)$$

One uses the finite-volume approach, where the cell averages of the state vectors for these cells characterize the state of the fluid. For cell *i*, one integrate its state vector U:

$$Q_i = \begin{pmatrix} m_i \\ p_i \\ E_i \end{pmatrix} = \int_{V_i} ?UdV \quad (4)$$

One can acknowledge the change in this quantity for two causes, i.e., time (rate) and position (gradient). The former is calculated using Gauss' theorem to convert the volume integral into a surface integral, where w is the velocity of every point of the cell boundary. If at any point w = o, it is a Eulerian scheme; while if w = v, it is a Lagrangian scheme, where the surface would be moving along with the local flow. In fact, w is dependent on the velocities of the adjacent cells across the interface. To calculate it, one employs the Harten-Lax-van Leer (HLL) framework and find the eigenvalue. Notably, all the calculations here are in the frame that is moving with the interface (as shown in Fig. 2) [13].

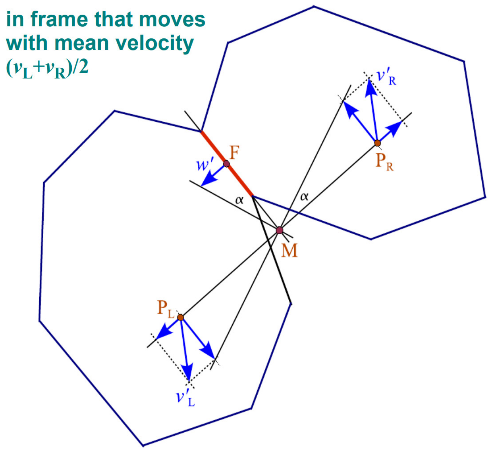

Fig. 2 Demonstration of moving mesh [13].

4. Comparison and state-of-art simulation results

Both code results for this cross-section at 2, 5, and 10 Gyr are displayed in Fig. 3. The high resolution in black and the fiducial resolution simulations in green is shown in this figure. The GIZMO simulations are represented by solid lines, whereas the AREPO simulations are represented by dotted lines. One can infer from the figure that improved agreement between the simulation results is achieved as the simulations' resolution is increased. In another words, there is greater agreement between the density profiles obtained from both code at the higher resolution compared to the fiducial resolution [9]. Other results are shown in Fig. 4 and Fig. 5.

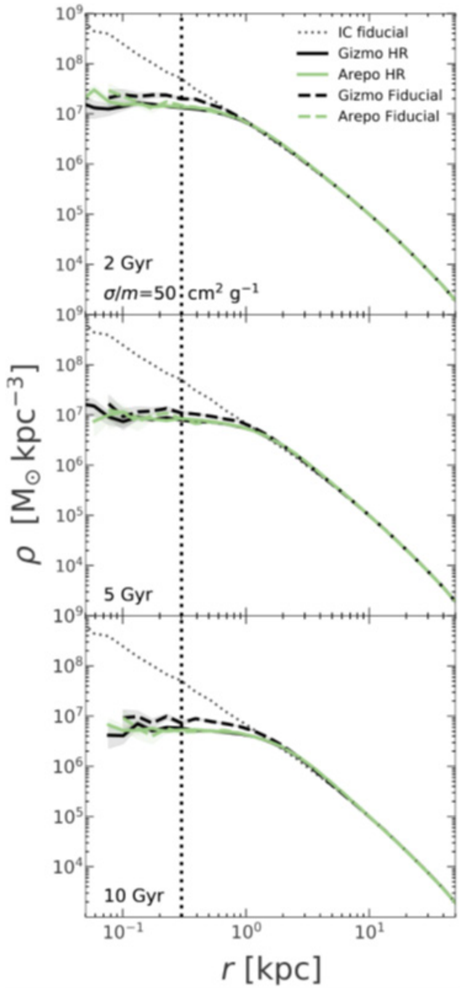

Fig. 3 Halo density [9].

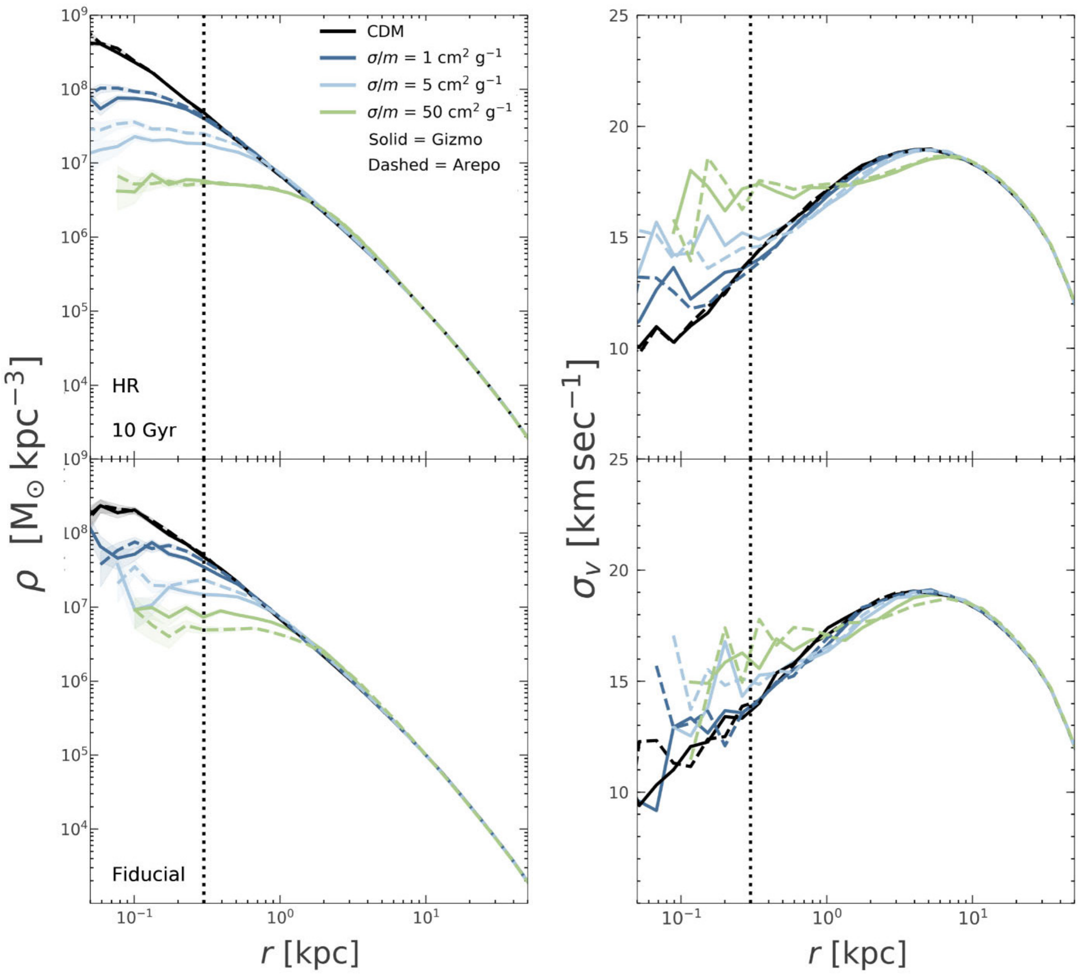

Fig. 4 Evolving simulated by both code [9].

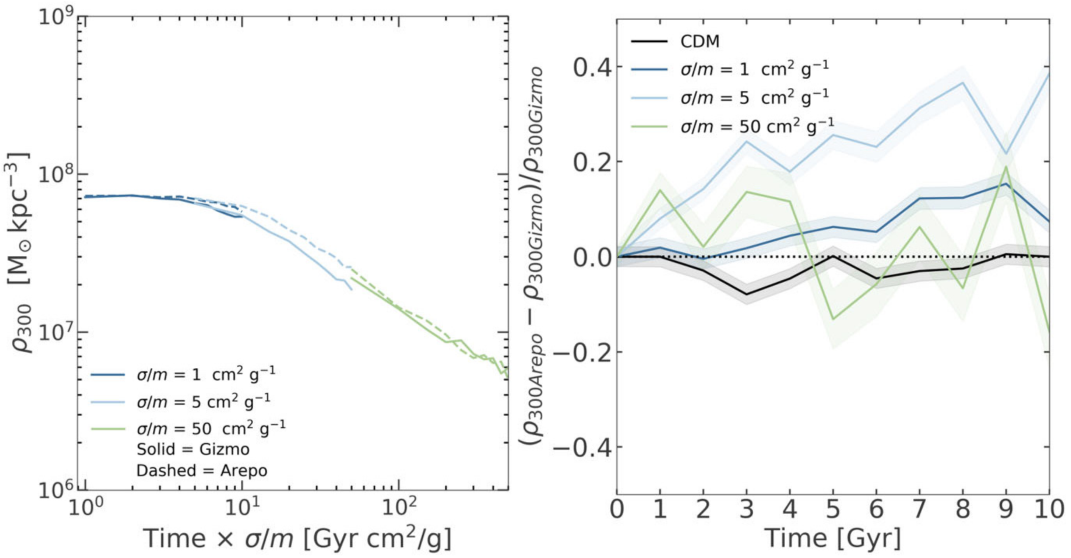

Fig. 5 Density simulation (300pc) [9].

The haloes generated using AREPO have greater densities than those calculated with GIZMO for cross-sections of σ/m = 1.5 cm^{2}g^{-1}. After approximately 4-5 billion years, the haloes that developed with GIZMO and had a cross-section of 50 cm cm^{2}g^{-1} were found to be denser than those that evolved with AREPO. The inversion is seen in both the higher resolution simulations and the fiducial resolution simulation [15, 16]. Furthermore, the variations in the quantity of DM self-interactions for each time step are established at an early stage of the simulation and remain constant for a 10-generation run. For example, over the course of the 10 Gyr simulation, the GIZMO 1 cm cm^2g^{-1} run consistently displays around 10,000 DM self-interactions per Gyr, and the AREPO 1 cm cm^2g^{-1} run consistently displays approximately 7000 DM self-interactions every Gyr. Similar to this, the corresponding AREPO run has about 30 000 DM self-interactions per Gyr, while the GIZMO 5 cm cm^2g^{-1} run regularly has about 40 000 [10]. The calcualtion efficiencies are illustrtaed in Fig. 6 [11]. Some state-of-art results are shown in Fig. 7 [17], Fig. 8 [10] and Fig. 9 [12].

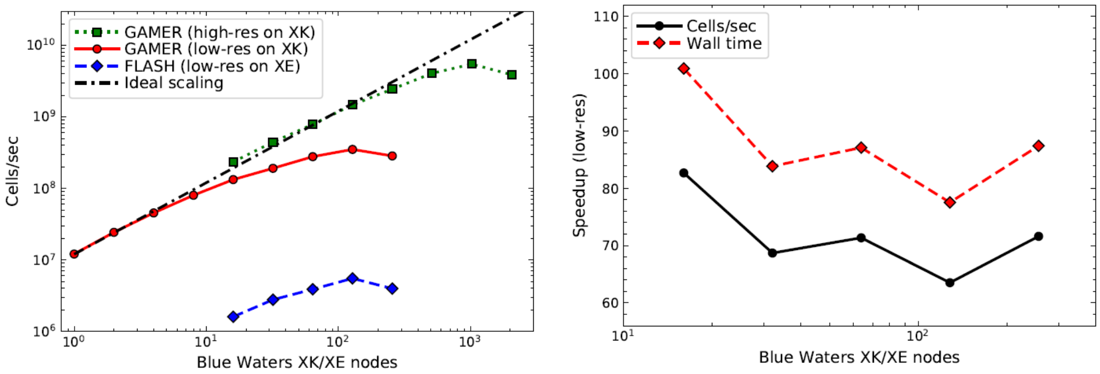

Fig. 6 The efficiency of two codes that run on GPU and CPU(similar scale) generating galaxy cluster merger [11].

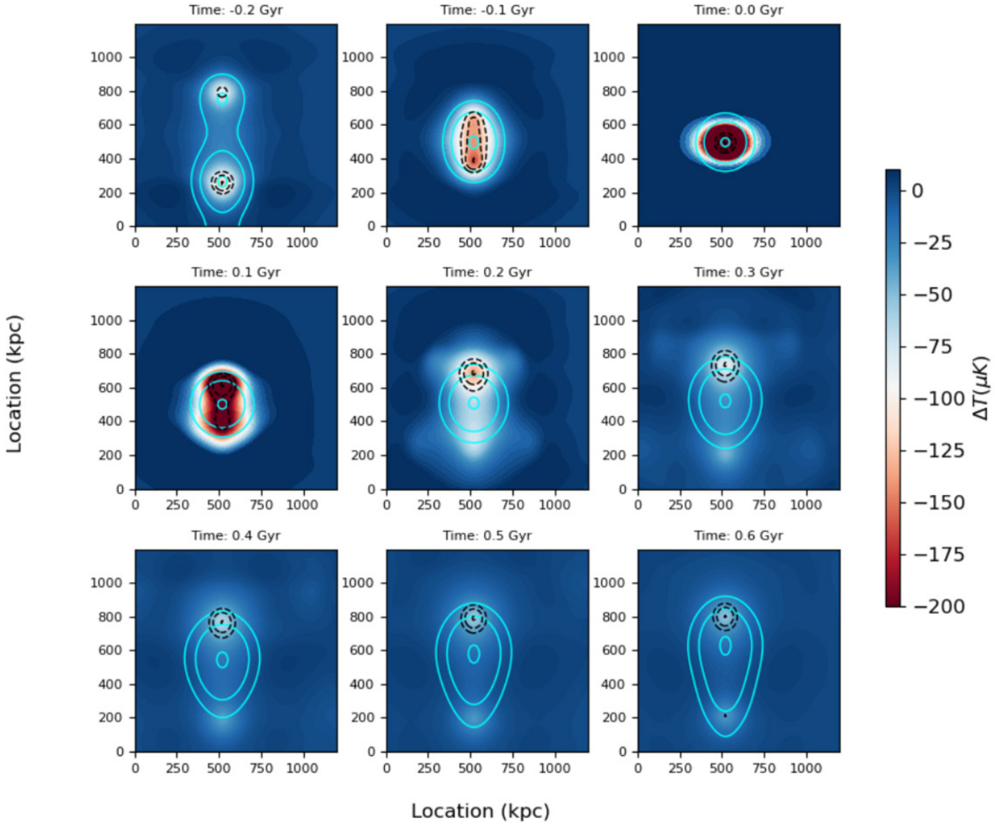

Fig. 7 Evolution of the SZ signal generated by GAMER-2 [17].

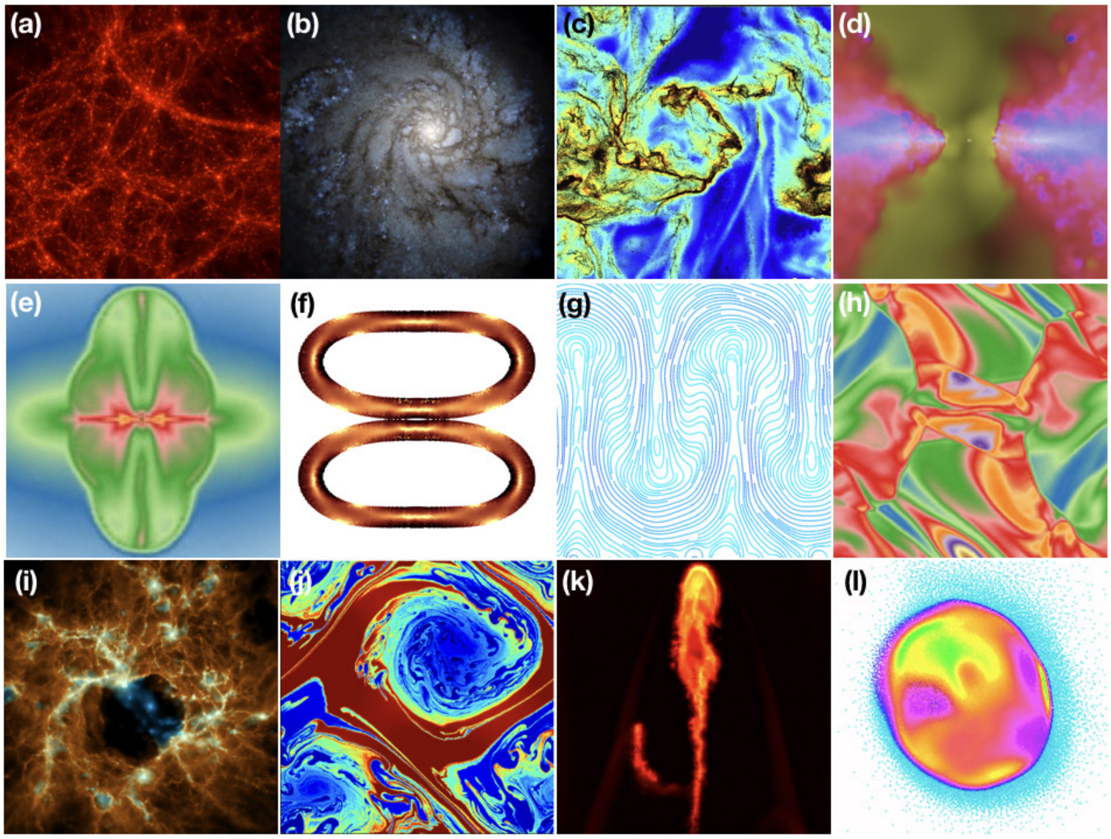

Fig. 8 Examples of GIZMO-run simulations: (a) Cosmology: Large-scale simulation of dark matter and baryonic cosmology (b) Formation of galaxies (c) Dust dynamics (aero) enigmatic black holes Stellar proto-disks (f) Solid-body/elastic dynamics (g) Physics of plasmas (h) Hydraulic magneto dynamics (i) Formation of stars (j) Dynamics of fluids (k) Fluids in multiple phases in the ISM/CGM/IGM (l) Simulations of impact and many materials [10].

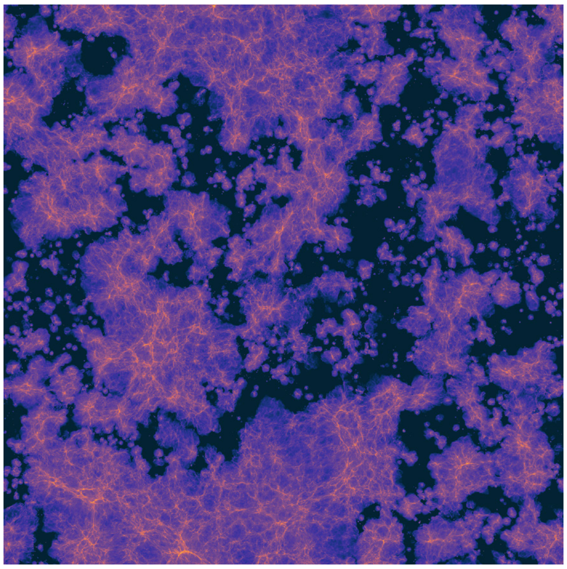

Fig. 9 Recombination rate density generated by thesan-1 using AREPO at *z* = 6.6 occupy 2.5% of the box volume [12].

5. Limitations

Recent research highlights several significant issues with astrophysical codes. The overheating problem alters the multi-phase gas structure of the ISM by changing the highly-ionized, hot (> 10,000 K) gas in Hii regions into partially-ionized, warm (~8000 K) gas. Additionally, insufficient spatial resolution smooths out the shock patterns. These mistakes in the gas phase structure might cause issues when estimating line luminosities [15, 16]. When the three resulting channels are nonlinearly connected, it becomes difficult to accurately model the low-density bubble evacuated by ionization feedback and the swept-up shell. This leads to challenges in replicating their strengthened or weakened impacts on the final momentum output of supernova explosions. Consequently, it creates uncertainties in the amount of gas expelled from a galaxy and the total energy or momentum contributed by star feedback.

6. Conclusion

Currently, the software community of astrophysical simulation has grown more and more vigorous. In conclusion, this study demonstrates the basics of astrophysical simulation, provide several examples of common simulation methods, and introduce three advanced codes: AREPO, GAMER-2, and GIZMO. In addition, the comparison between these codes has been mentioned and this study will list several simulation results from recent projects that using the code mentioned in the paper. It can be anticipated that there will be more and more simulation projects being set up by small-scale research groups due to the variety of communities. Thus, the research comparing different codes is significant to help researchers to know the nature of different simulation techniques and take advantage from them.

References

[1] Bethe H A. Energy Production in Stars. Physical Review, 1939, 55(1): 103-103.

[2] Nelson D, Pillepich A, Springel V, et al. First results from the IllustrisTNG simulations: the galaxy colour bimodality. Monthly Notices of the Royal Astronomical Society, 2017, 475(1): 624-647.

[3] Kannan R, Enrico G, Smith A M, et al. Introducing the thesan project: radiation-magnetohydrodynamic simulations of the epoch of reionization. 2021, 511(3): 4005-4030.

[4] Abbott R, Abbott T D, Abraham S, et al. GWTC-2: Compact Binary Coalescences Observed by LIGO and Virgo during the First Half of the Third Observing Run. Physical Review X, 2021, 11(2).

[5] Goldbaum N J. Extracting Insights from Astrophysics Simulations. Astronomical Data Analysis Software and Systems XXVII, 2017, 522: 225.

[6] Hopkins P F. A new class of accurate, mesh-free hydrodynamic simulation methods. Monthly Notices of the Royal Astronomical Society, 2015, 450(1): 53-110.

[7] Hopkins P F, Wetzel A, Keres D, et al. FIRE-2 simulations: physics versus numerics in galaxy formation. Monthly Notices of the Royal Astronomical Society, 2018, 480(1): 800-863.

[8] Rocha M, Annika, Bullock J S, et al. Cosmological simulations with self-interacting dark matter I. Constant-density cores and substructure. Monthly Notices of the Royal Astronomical Society, 2013, 430(1): 81-104.

[9] Meskhidze H, Mercado F J, Sameie O, et al. Comparing implementations of self-interacting dark matter in the gizmo and arepo codes. Monthly Notices of the Royal Astronomical Society, 2022, 513(2): 2600-2608.

[10] Hopkins P F. A new public release of the GIZMO code. arXiv preprint arXiv:1712.01294, 2017.

[11] Schive H Y, ZuHone J A, Goldbaum N J, et al. GAMER-2: a GPU-accelerated adaptive mesh refinement code–accuracy, performance, and scalability. Monthly Notices of the Royal Astronomical Society, 2018, 481(4): 4815-4840.

[12] Weinberger R, Springel V, Pakmor R. The Arepo public code release. The Astrophysical Journal Supplement Series, 2020, 248(2): 32.

[13] Springel V. E pur si muove: Galilean-invariant cosmological hydrodynamical simulations on a moving mesh. Monthly Notices of the Royal Astronomical Society, 2010, 401: 791-851.

[14] Deng Y, Li H, Kannan R, et al. Simulating ionization feedback from young massive stars: impact of numerical resolution. Monthly Notices of the Royal Astronomical Society, 2023, 527(1): 478-500.

[15] Smith A, Kannan R, Tacchella S, et al. The physics of Lyman-α escape from disc-like galaxies. Monthly Notices of the Royal Astronomical Society, 2022, 517(1): 1-27.

[16] TACCHELLA S, Smith A, Kannan R, et al. Hα emission in local galaxies: star formation, time variability, and the diffuse ionized gas. Monthly Notices of the Royal Astronomical Society, 2022, 513(2): 2904-2929.

[17] Felix S, Gogoi A, Shavelle K, et al. Decoding the Early Universe: Exploring a Merger Scenario for the High-Redshift Cluster JKCS041 using Numerical Models. arXiv preprint arXiv:2402.13334, 2024.