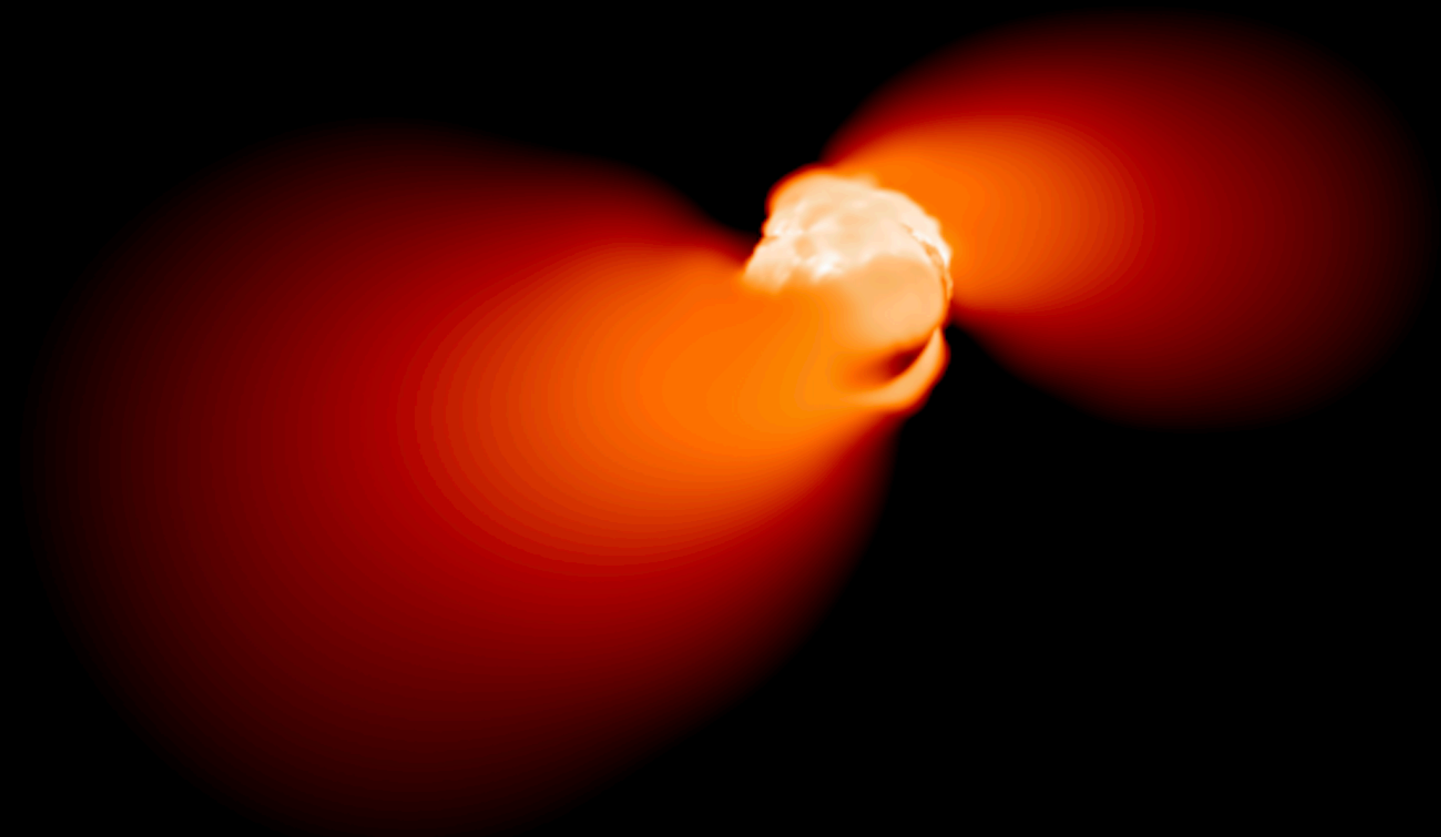
Binary Neutron Star merger simulations with fully General Relativistic Smoothed Particle Hydrodynamics (SPH) code SPHINCS_BSSN



Swapnil Shankar

(with Stephan Rosswog and Peter Diener)

Postdoc, University of Hamburg

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Multi-Messenger Astrophysics in the Dynamic Universe
6 Feb 2026

erc

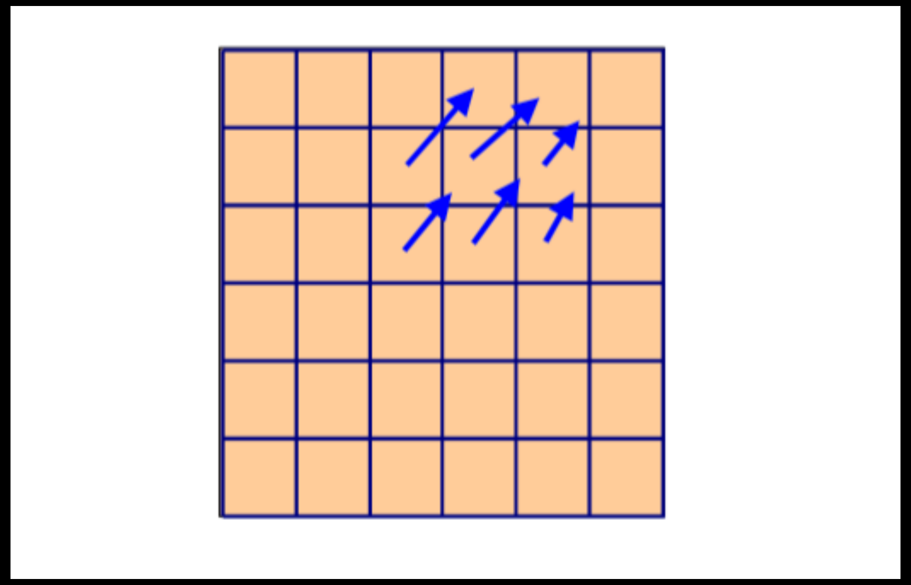
Outline

- Why Lagrangian Hydrodynamics (i.e. using particles, not grids)
- SPHINCS_BSSN code strategy
- Conservative to primitive transformation in SPH with microphysical tabulated EOS
- Simulations

Why Smoothed Particle Hydrodynamics?

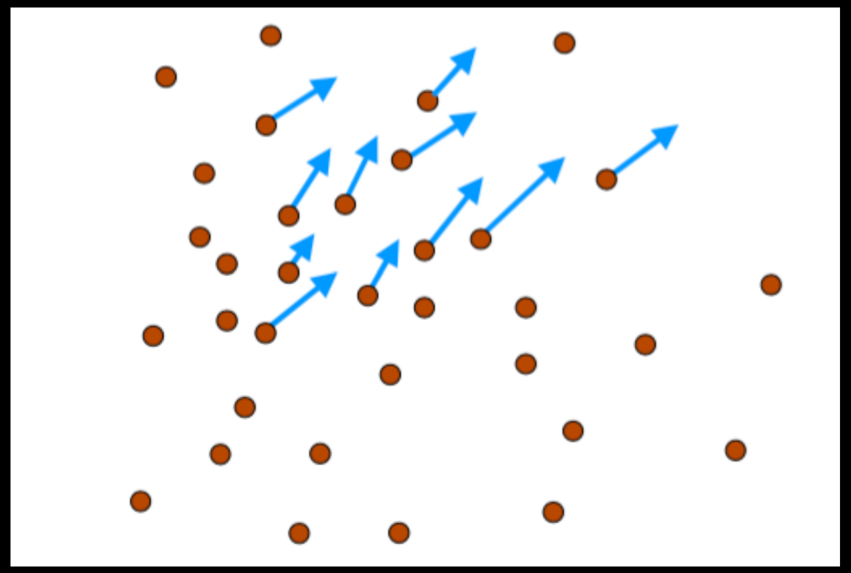
Finite Volume (Magneto)-hydrodynamics

- Eulerian
- Cartesian grid



Smoothed Particle Hydrodynamics (SPH)

- Lagrangian
- Spherical particles



Advantages of SPH

- Exact conservation:
  - mass
  - energy
  - momentum
  - angular momentum

even at finite resolution,

built-in by construction!

- No need for an artificial atmosphere!
- History of all particles available:

No need for artificial tracers!

Basic ideas of Smooth Particle Hydrodynamics

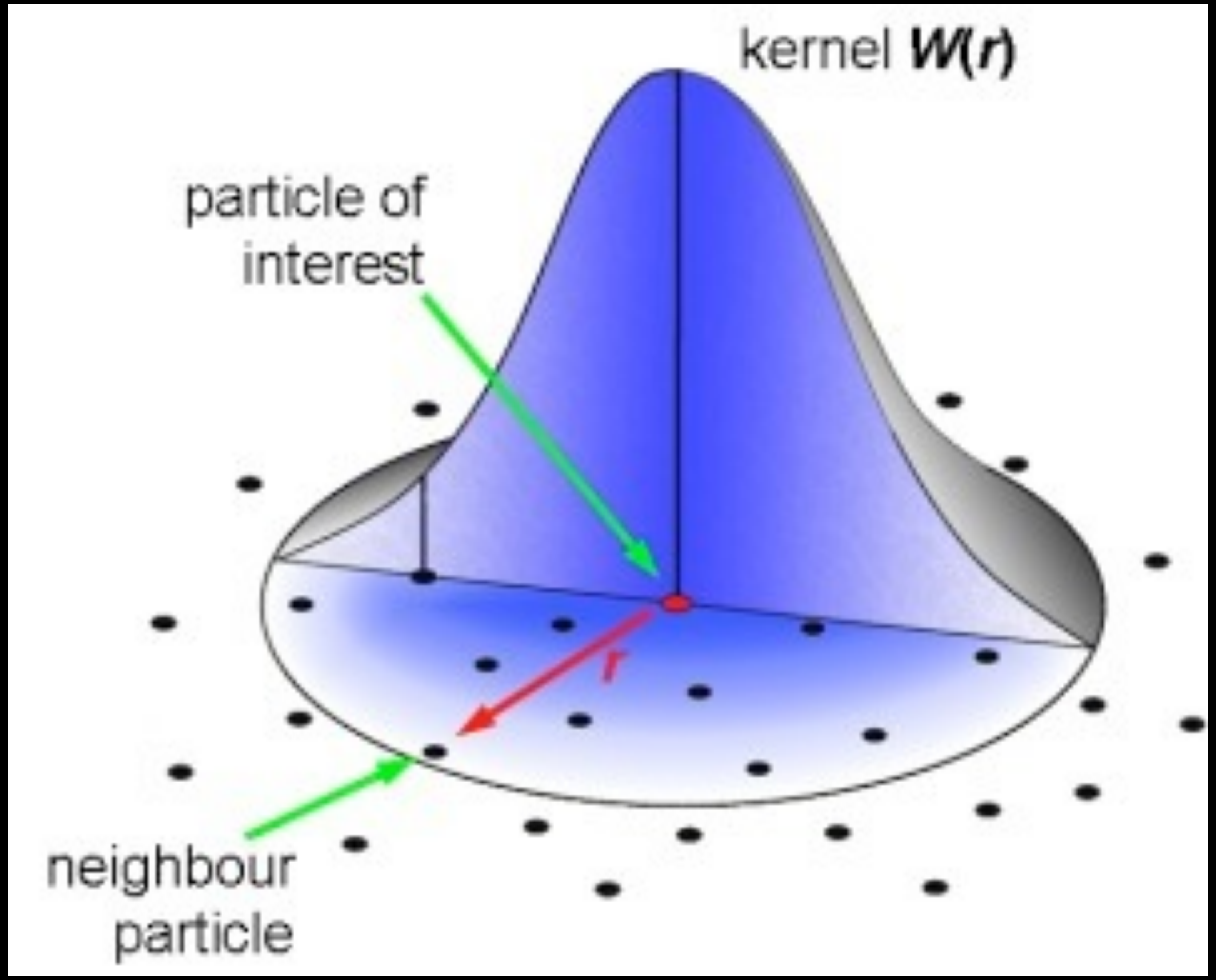
- Replace fluid by finite set of particles

- Particles move with local fluid velocities

- Each particle has an interaction radius $\Longrightarrow$ “smoothing length”

- Each particle carries a smooth “kernel function”: used to recover smooth fields and calculate gradients

- Aim: particles should move in a way so that mass, energy, momentum and angular momentum are conserved *by construction*



SPHINCS_BSSN code strategy

Strategy

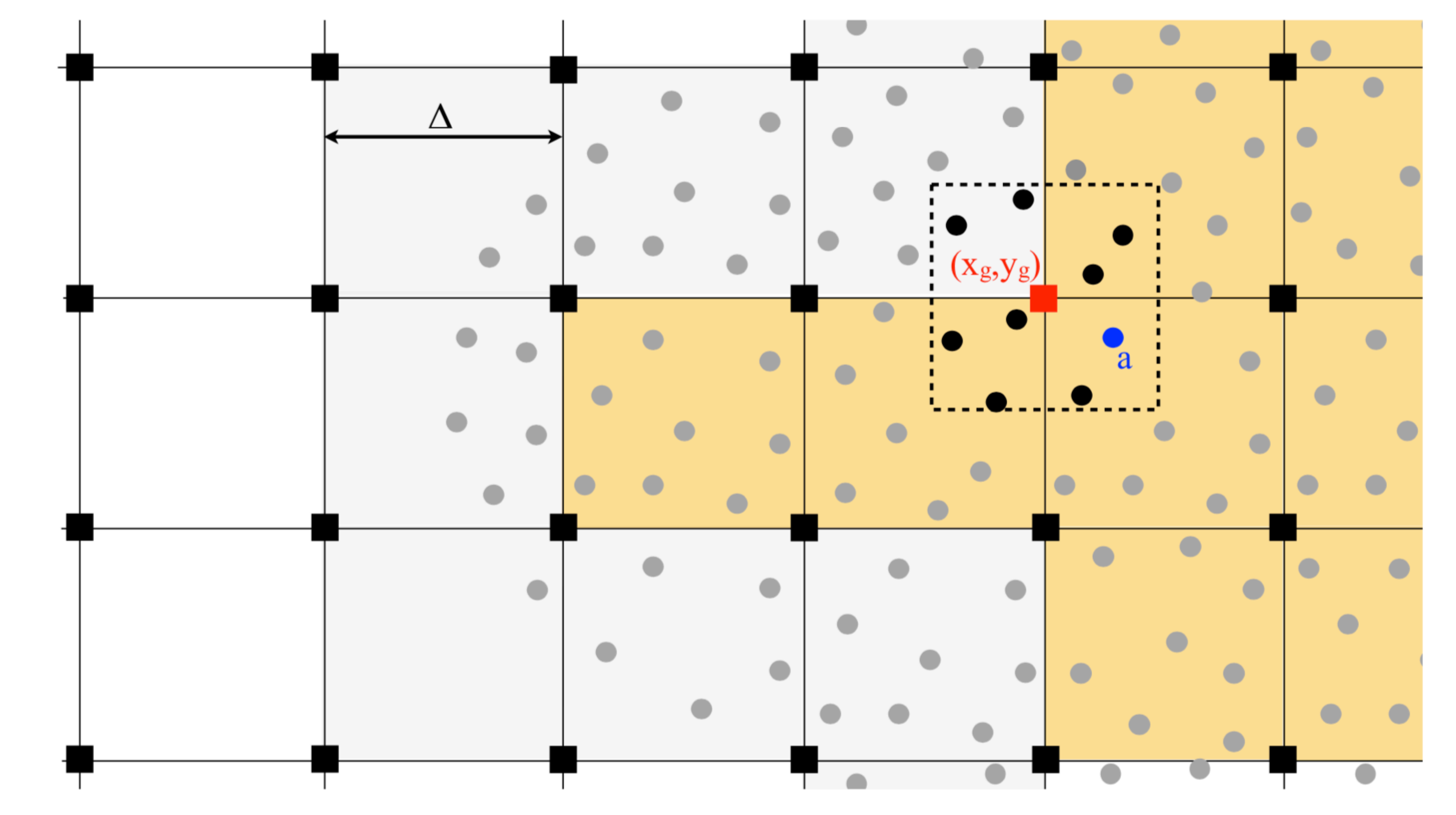
1. Spacetime evolution (mesh-based):

   "Baumgarte-Shapiro-Shibata-Nakamura-Oohara-Kojima" (BSSN-OK) with fixed mesh refinement

2. Matter evolution (particle-based):

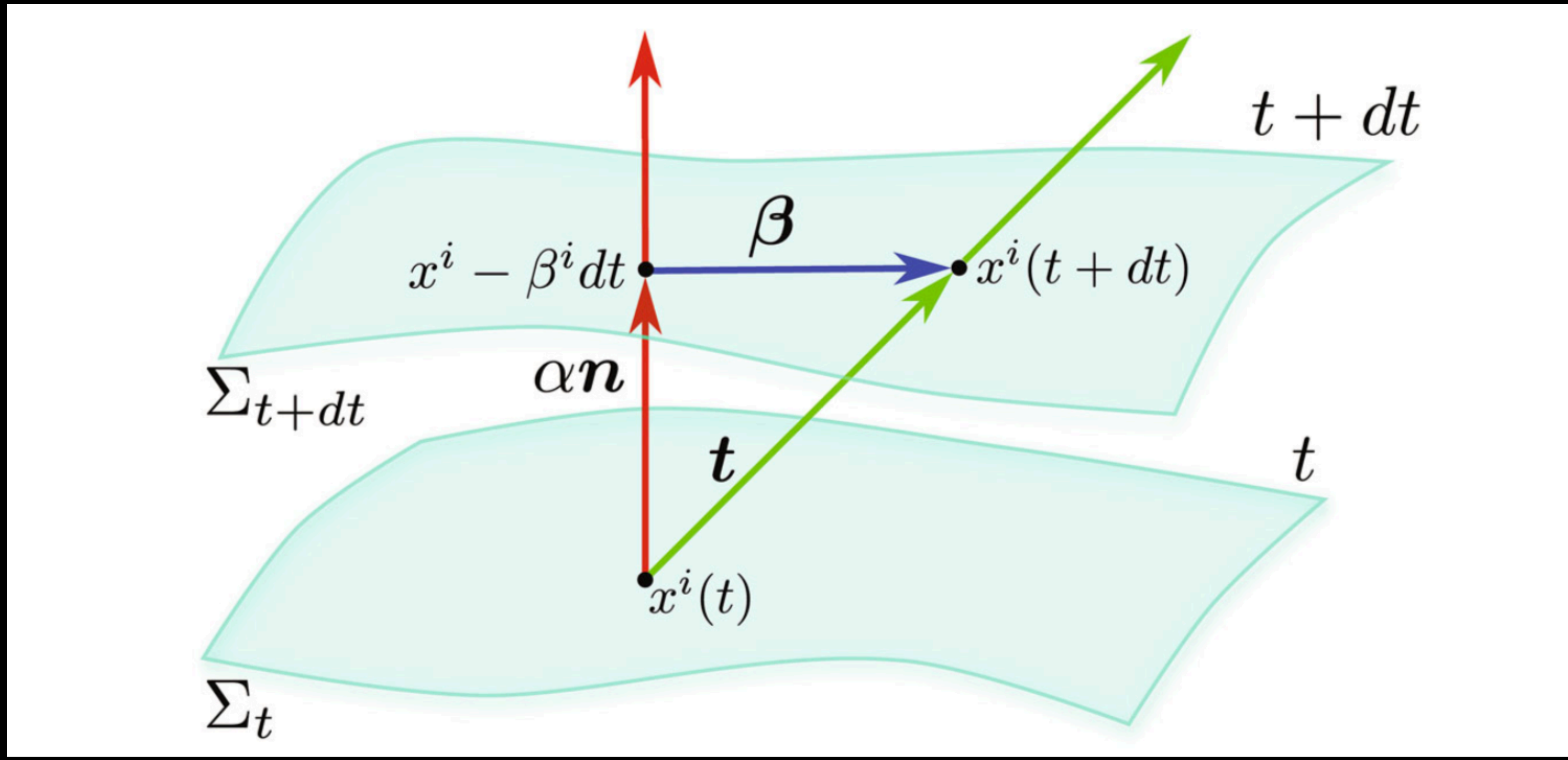
   Freely moving SPH-particles

3. Coupling between the particles and the mesh



1. Spacetime evolution

- Essentially same approach that is taken in Eulerian Numerical Relativity



- "3 + 1 - split": evolve spacelike hypersurfaces forward in time

- Spacetime line element

$$ds^2 = -\underset{\substack{\uparrow \\ \text{“lapse”}}}{\alpha^2} dt^2 + \underset{\substack{\uparrow \\ \text{“spatial metric”}}}{\gamma_{ij}}(dx^i + \beta^i dt)(dx^j + \underset{\substack{\uparrow \\ \text{“shift”}}}{\beta^j} dt)$$

- Similar to magneto-hydrodynamics: i) evolution equations & ii) constraint equations

- Evolution via BSSN-OK equations on a structured mesh

2. General-relativistic SPH

- Derived from Lagrangian

$$L_{\rm GR} = -\int T^{\mu\nu} U_\mu U_\nu \sqrt{-g}\, dV$$

energy-momentum tensor

4-velocities

determinant of metric tensor

- Numerical variables:

canonical momentum per baryon: $$(S_i)_a = \frac{1}{v_a}\frac{\partial L}{\partial v_a^i} = (\Theta \mathscr{E} v_i)_a$$

canonical energy per baryon: $$E = \Sigma_a (\partial L/\partial \vec{v}_a)\,.\,\vec{v}_a - L \implies \hat{e}_a = \left(S_i v^i + \frac{1+u}{\Theta} \right)_a$$

mass density in computing frame: $$\rho_a^* = \Sigma_b m_b W_{ab}(|\vec{r}_a - \vec{r}_b|/h_a)$$

- after a lot of algebra (see Rosswog 2009 for details) one finds:

momentum

$$\frac{d(S_i)_a}{dt} = -\sum_b m_b \left\{ \frac{P_a\, D_i^a}{\rho_a^{*2}} + \frac{P_b\, D_i^b}{\rho_b^{*2}} \right\} + \left(\frac{\sqrt{-g}}{2\rho^*} T^{\mu\nu} \frac{\partial g_{\mu\nu}}{\partial x^i} \right)_a$$

energy

$$\frac{de_a}{dt} = -\sum_b \nu_b \left\{ \frac{P_a v_b^i}{\rho_a^{*2}}\, D_i^a + \frac{P_b v_a^i}{\rho_b^{*2}}\, D_i^b \right\} - \left(\frac{\sqrt{-g}}{2\rho^*} T^{\mu\nu} \partial_t g_{\mu\nu} \right)$$

baryon number

$$\rho_a^* = \sum_b m_b W_{ab}(h_a)$$

- Evolved variables: S_i and $\hat{e}$

- Calculate ρ^* from positions and smoothing lengths.

- BUT: we are not evolving the physical variables we are interested in

$\Longrightarrow$ need to recover "physical variables" (ρ, u, v^i)

from "numerical variables" $(\rho^*, S_i, \hat{e})$

3. Coupling between matter and spacetime

- "mesh needs": energy-momentum tensor $T_{\mu\nu}$ (known at particles)

- "particles need": derivatives of metric/gravitational acceleration terms

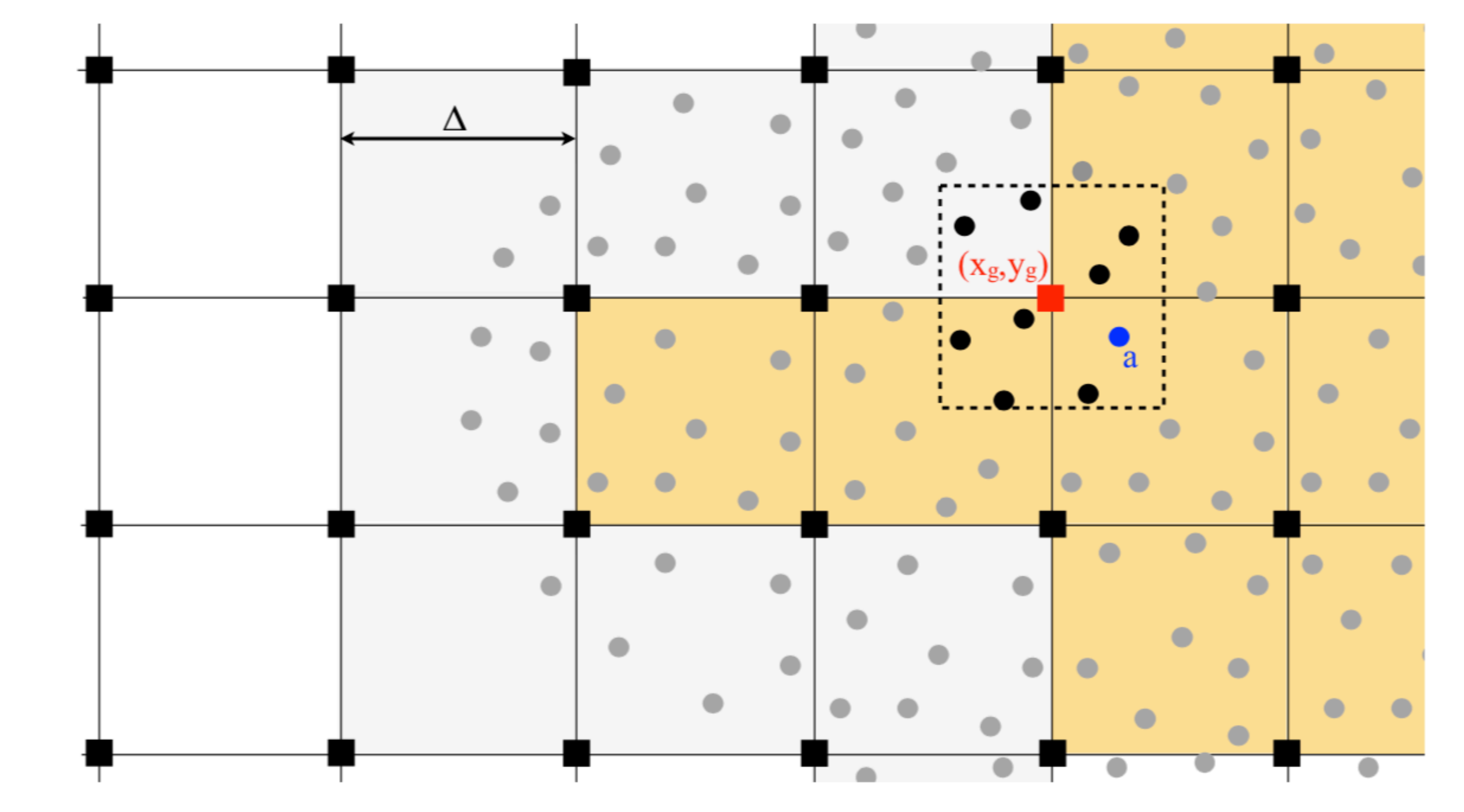
- Particle-Mesh method use in the code:

  (A) mesh → particle

  - the simpler of both steps
  - 5th order Hermite interpolation (Rosswog & Diener 2021)

  (B) particle → mesh

  - lots (!) of experiments
  - approach: "Local Regression Estimate" (LRE) + "Multi-dimensional Optimal Order Detection" (MOOD) (Rosswog, Torsello, Diener 2023)



“Conservative” to “primitive” transformation in SPH with microphysical tabulated EOS

Recovery of primitive variables for finite-T tabulated EoS

- Need to recover primitive variables from evolved variables at each step.

- Primitive variables: (ρ, u, v^i)

- Evolved variables: $(\rho^*, \hat{e}, S_i)$

- Similar to conservative to primitive transformation in Eulerian codes (GRMHD)

- We **develop** new techniques for primitive variable recovery in SPH (using motivations from GRMHD).

$$\rho^* = \sqrt{-g}\,\Theta\,\rho$$

$$S_i = \Theta\,\mathcal{E}\,v_i$$

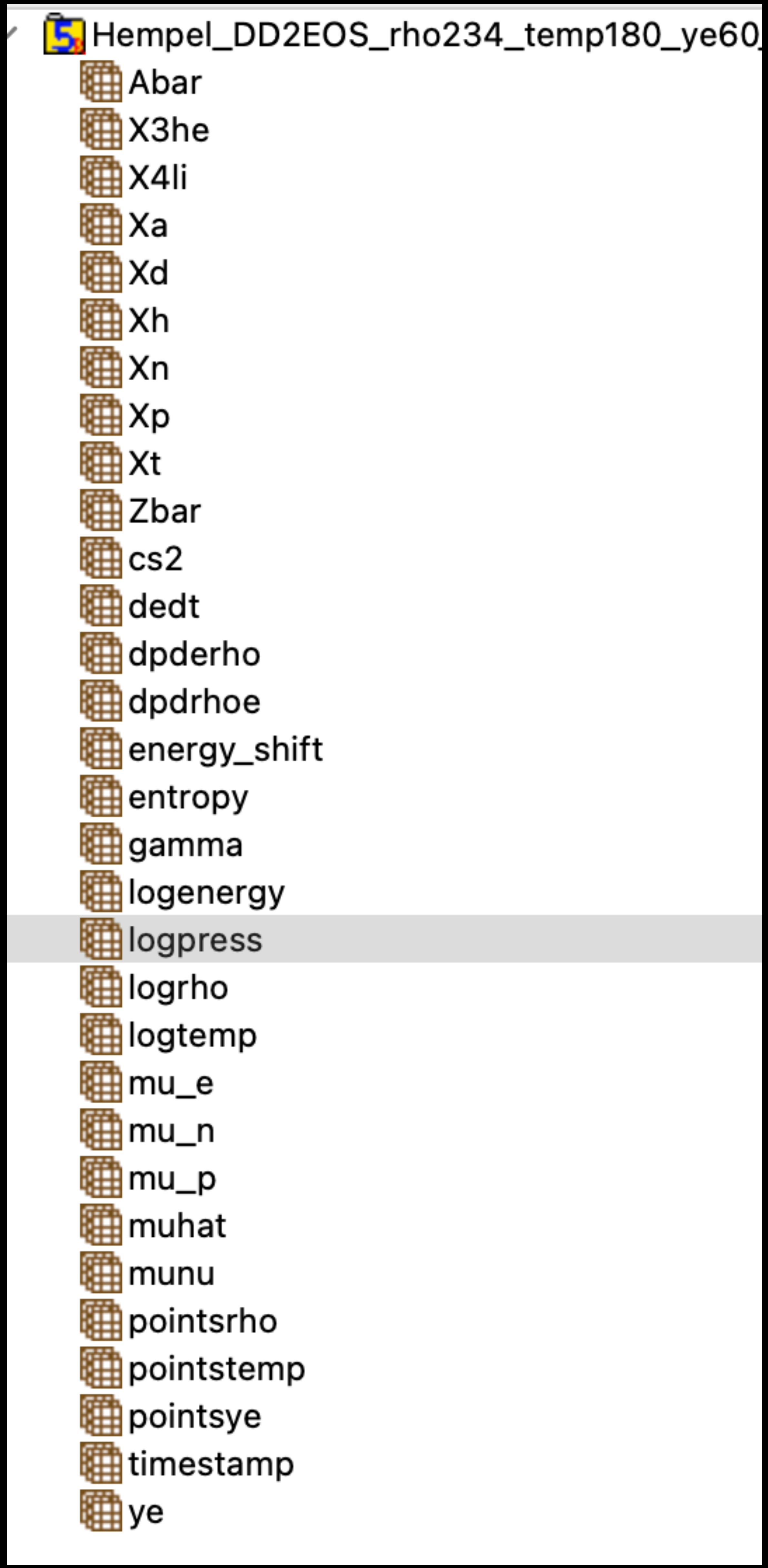
$$\hat{e} = S_i v^i + \frac{1+u}{\Theta}$$

$$\Theta \equiv \frac{1}{\sqrt{-g_{\mu\nu}v^\mu v^\nu}}.$$

$$\mathcal{E} = 1 + u + \frac{P}{\rho}$$

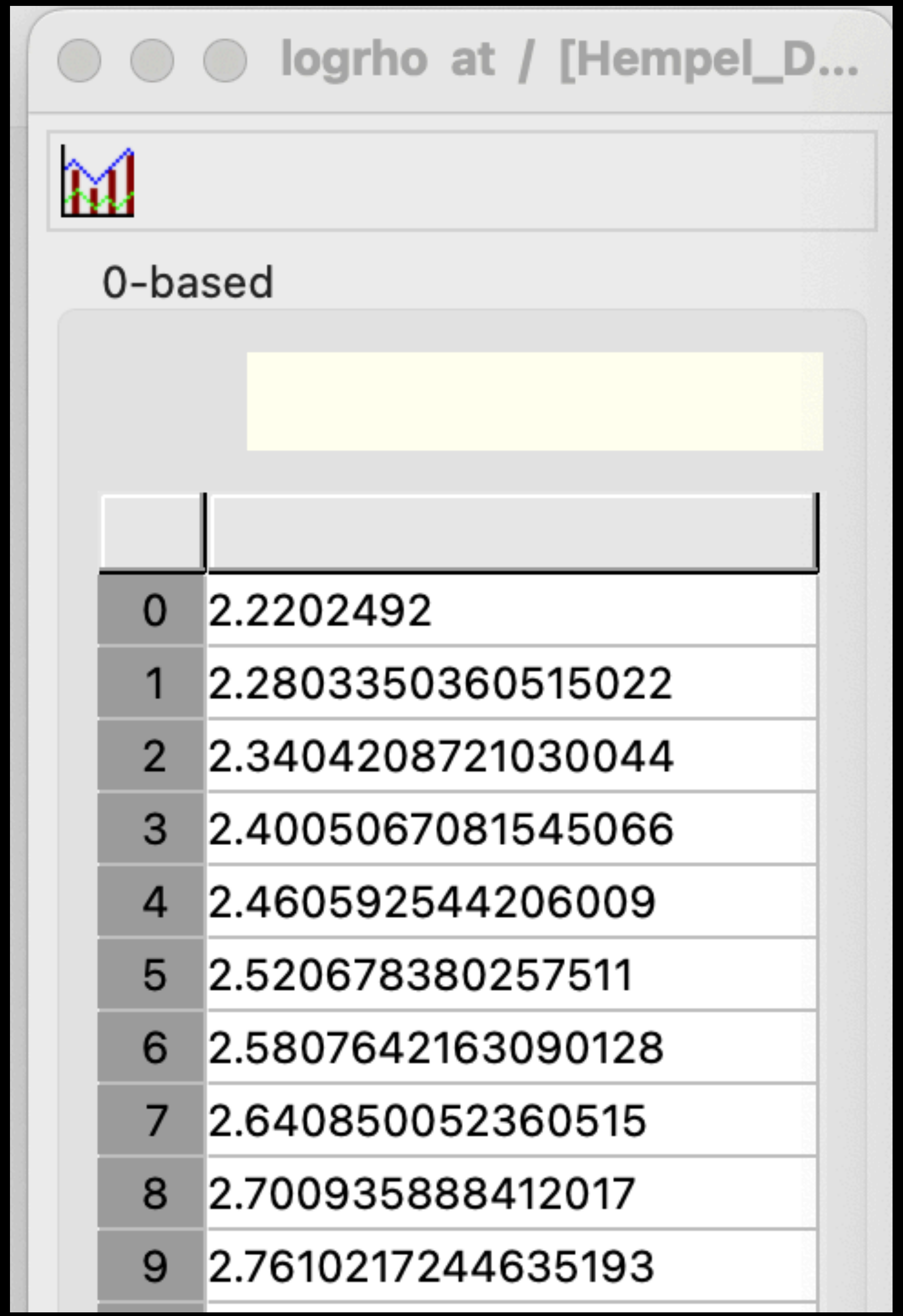
Tabulated EoS

- Adds microphysics to the system.
- Pressure, energy, entropy etc. are provided in a tabulated form as a discrete function of (ρ, T, Ye).
- Need to perform interpolations to calculate these at any given (ρ, T , Ye).
- More expensive and complex than analytic EoSs.
- Examples: stellarcollapse.org, CompOSE etc.



Hempel_DD2EOS_rho234_temp180_ye60

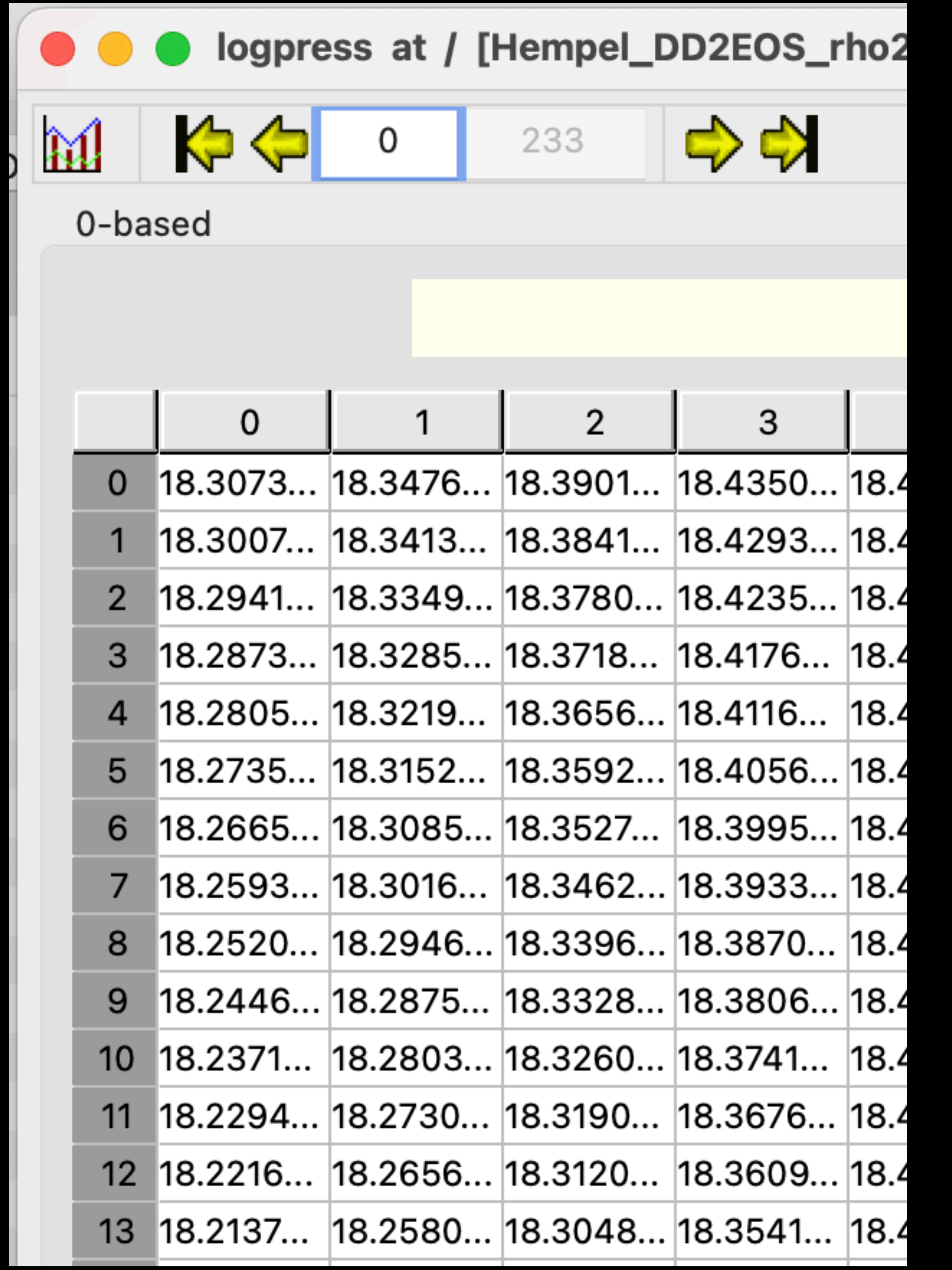
- Abar
- X3he
- X4li
- Xa
- Xd
- Xh
- Xn
- Xp
- Xt
- Zbar
- cs2
- dedt
- dpderho
- dpdrhoe
- energy_shift
- entropy
- gamma
- logenergy
- logpress
- logrho
- logtemp
- mu_e
- mu_n
- mu_p
- muhat
- munu
- pointsrho
- pointstemp
- pointsye
- timestamp
- ye



logrho at / [Hempel_D...

0-based

0	2.2202492
1	2.2803350360515022
2	2.3404208721030044
3	2.4005067081545066
4	2.460592544206009
5	2.520678380257511
6	2.5807642163090128
7	2.640850052360515
8	2.700935888412017
9	2.7610217244635193



logpress at / [Hempel_DD2EOS_rho2

0 233

0-based

	0	1	2	3	
0	18.3073...	18.3476...	18.3901...	18.4350...	18.4
1	18.3007...	18.3413...	18.3841...	18.4293...	18.4
2	18.2941...	18.3349...	18.3780...	18.4235...	18.4
3	18.2873...	18.3285...	18.3718...	18.4176...	18.4
4	18.2805...	18.3219...	18.3656...	18.4116...	18.4
5	18.2735...	18.3152...	18.3592...	18.4056...	18.4
6	18.2665...	18.3085...	18.3527...	18.3995...	18.4
7	18.2593...	18.3016...	18.3462...	18.3933...	18.4
8	18.2520...	18.2946...	18.3396...	18.3870...	18.4
9	18.2446...	18.2875...	18.3328...	18.3806...	18.4
10	18.2371...	18.2803...	18.3260...	18.3741...	18.4
11	18.2294...	18.2730...	18.3190...	18.3676...	18.4
12	18.2216...	18.2656...	18.3120...	18.3609...	18.4
13	18.2137...	18.2580...	18.3048...	18.3541...	18.4

3D Newton-Raphson

- Start with an initial guess x_0
- Iterate until convergence

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \mathbf{f}(\mathbf{x})\mathbf{J}^{-1} = \mathbf{x}_n + \mathbf{dx}_n$$

- Reduce the number of equation from 5 to 3 with the use of certain scalars.

- Use (Θ, ξ, T) as iteration variables.

1D Ridder’s method

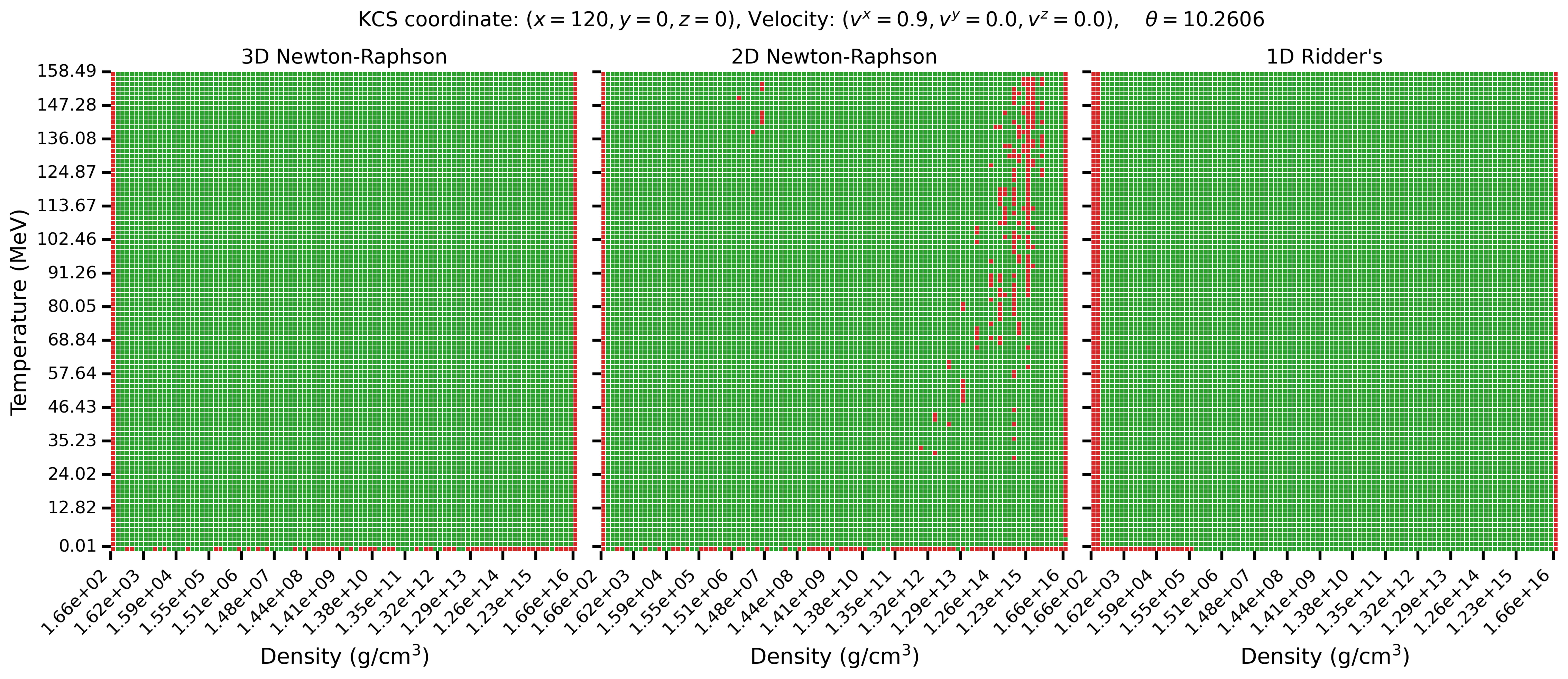
- An iterative method to find the root of a 1D equation
  - Robustness similar to bisection
  - Quadratic convergence similar to Newton’s method
  - Does not require any derivatives

- We iterate over the following equation for pressure P:

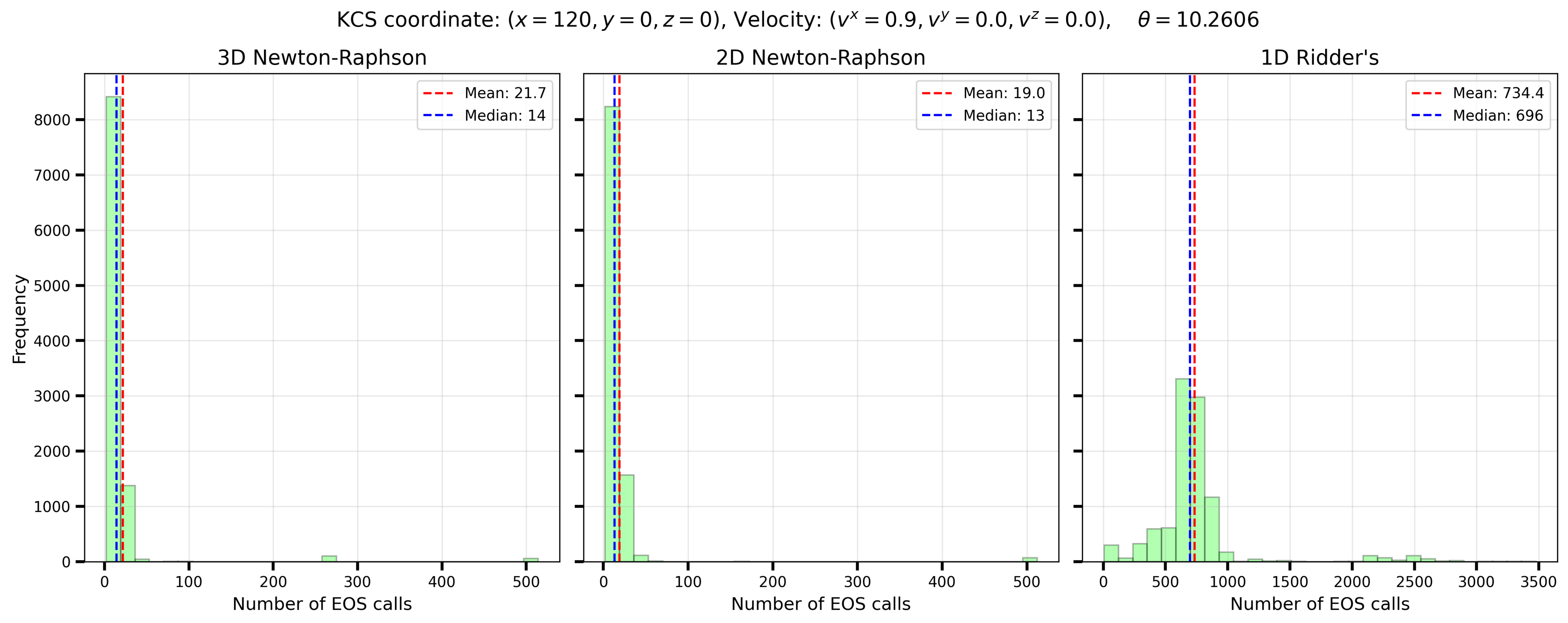
$$f(P) = P - P_{\text{EOS}}(\rho, T, Y_e)$$

- Independent of initial guess
- Very robust in practice

Robustness


KCS coordinate: $(x = 120, y = 0, z = 0)$, Velocity: $(v^x = 0.9, v^y = 0.0, v^z = 0.0)$, $\theta = 10.2606$
3D Newton-Raphson
2D Newton-Raphson
1D Ridder's
Temperature (MeV)
158.49
147.28
136.08
124.87
113.67
102.46
91.26
80.05
68.84
57.64
46.43
35.23
24.02
12.82
0.01
1.66e+02
1.62e+03
1.59e+04
1.55e+05
1.51e+06
1.48e+07
1.44e+08
1.41e+09
1.38e+10
1.35e+11
1.32e+12
1.29e+13
1.26e+14
1.23e+15
1.66e+16
Density (g/cm^3)

Cost


KCS coordinate: $(x = 120, y = 0, z = 0)$, Velocity: $(v^x = 0.9, v^y = 0.0, v^z = 0.0)$, $\theta = 10.2606$

3D Newton-Raphson vs 1D Ridder’s

- Main considerations: robustness and cost
- 3DNR: cheap, less robust (failure rate <1% in BNS production simulations)
- 1D Ridder’s: expensive, very robust
- In production simulations: use 3DNR as primary method, 1D Ridder’s as backup.

Simulations

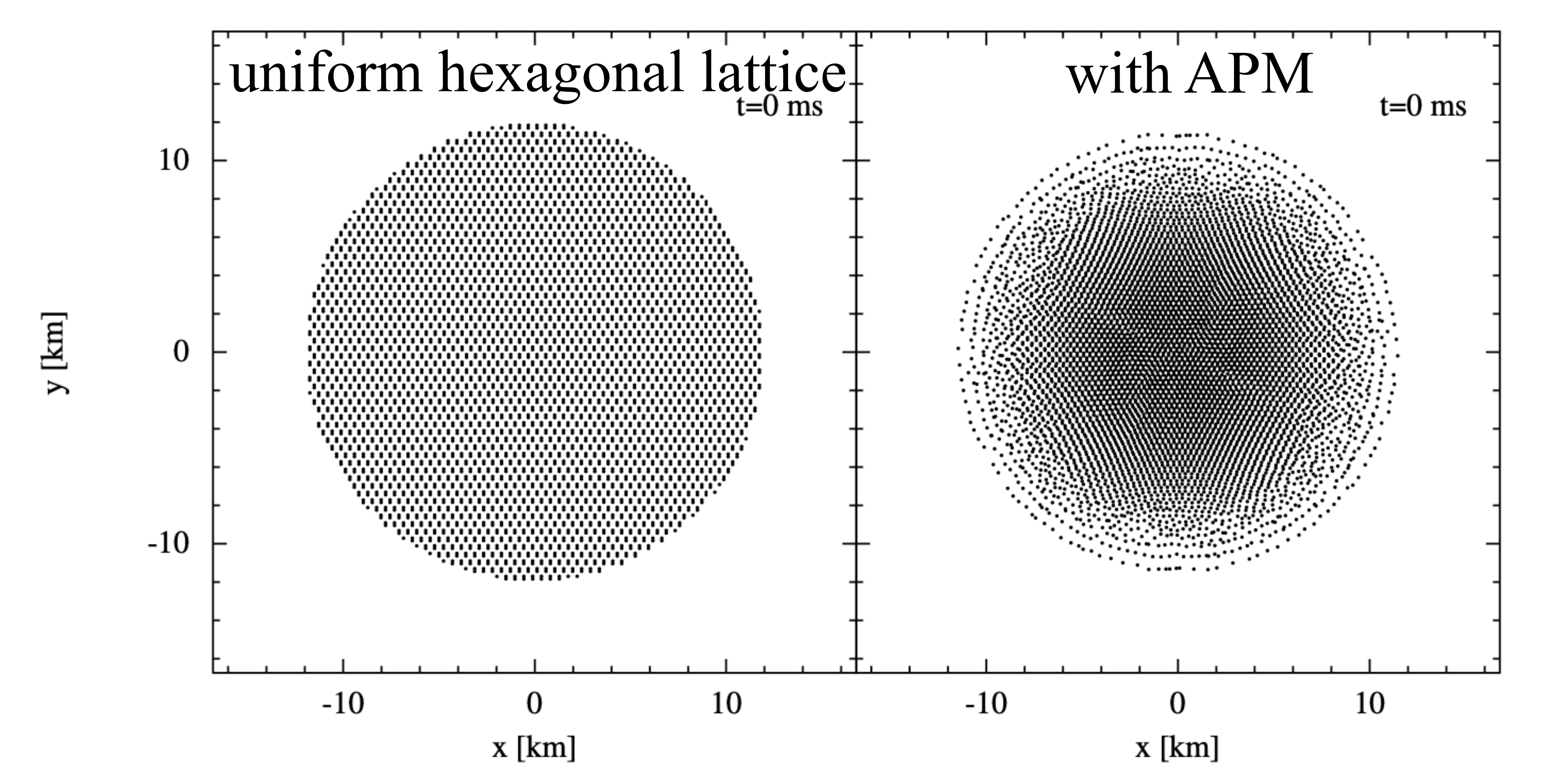
First GR SPH simulations of BNS mergers with finite-T tabulated EoS

- Initial conditions from FUKA-library (with GRHayl for stellarcollapse-format tables)

- Once you have ID, you are not yet done! ⟹ needs to be “translated to particles”

- Done via “Artificial Pressure Method” (Rosswog 2020)

relativistic, single neutron star

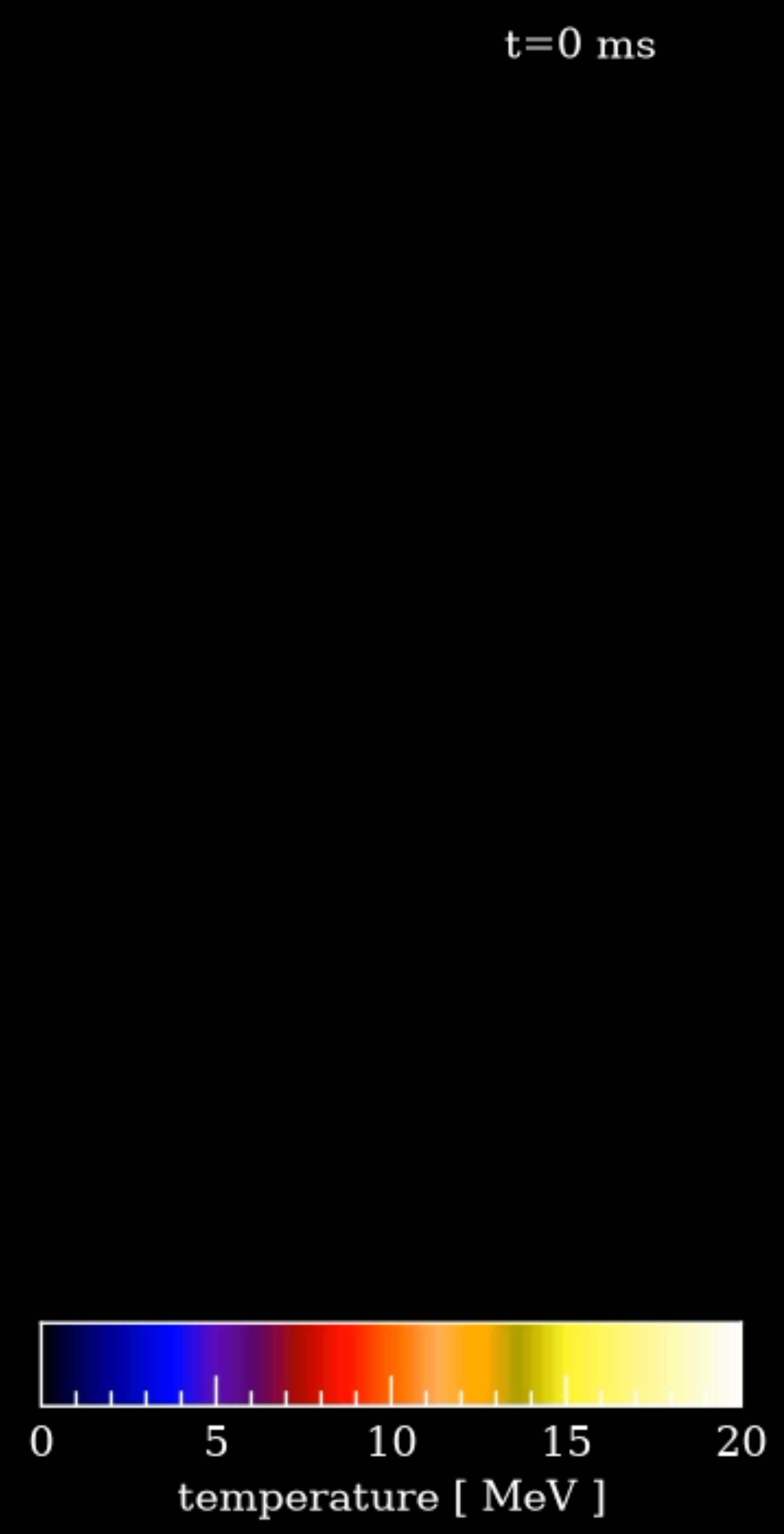


(Rosswog & Diener 2021)

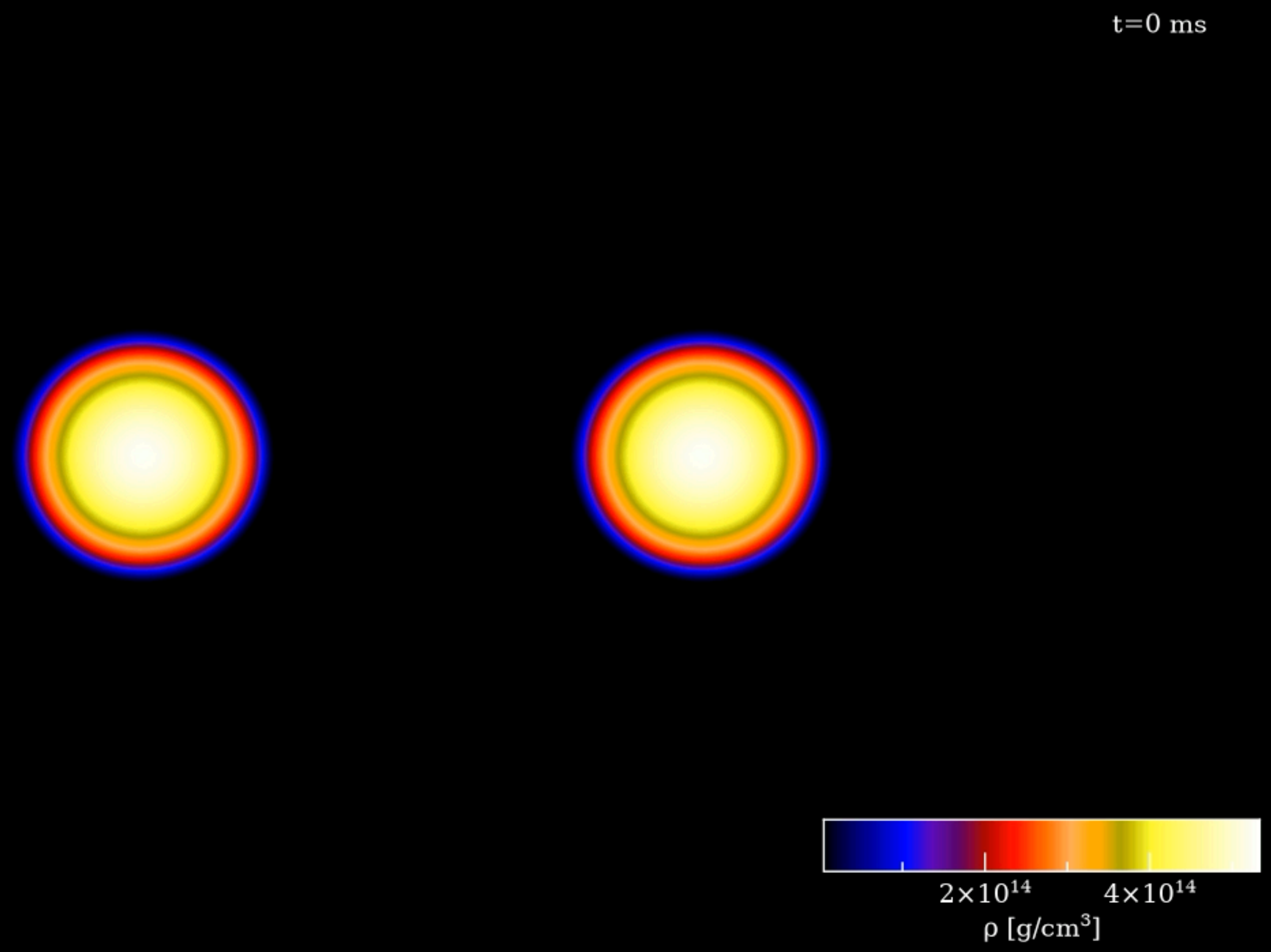
First GR SPH simulations of BNS mergers with finite-T tabulated EoS

- binary: 2 x 1.3 $M_\odot$, irrotational
- spacetime: 7 (fixed) refinement levels
- fluid: 2 million SPH particles
- EOS: DD2 (stellarcollapse.org)

Temperature

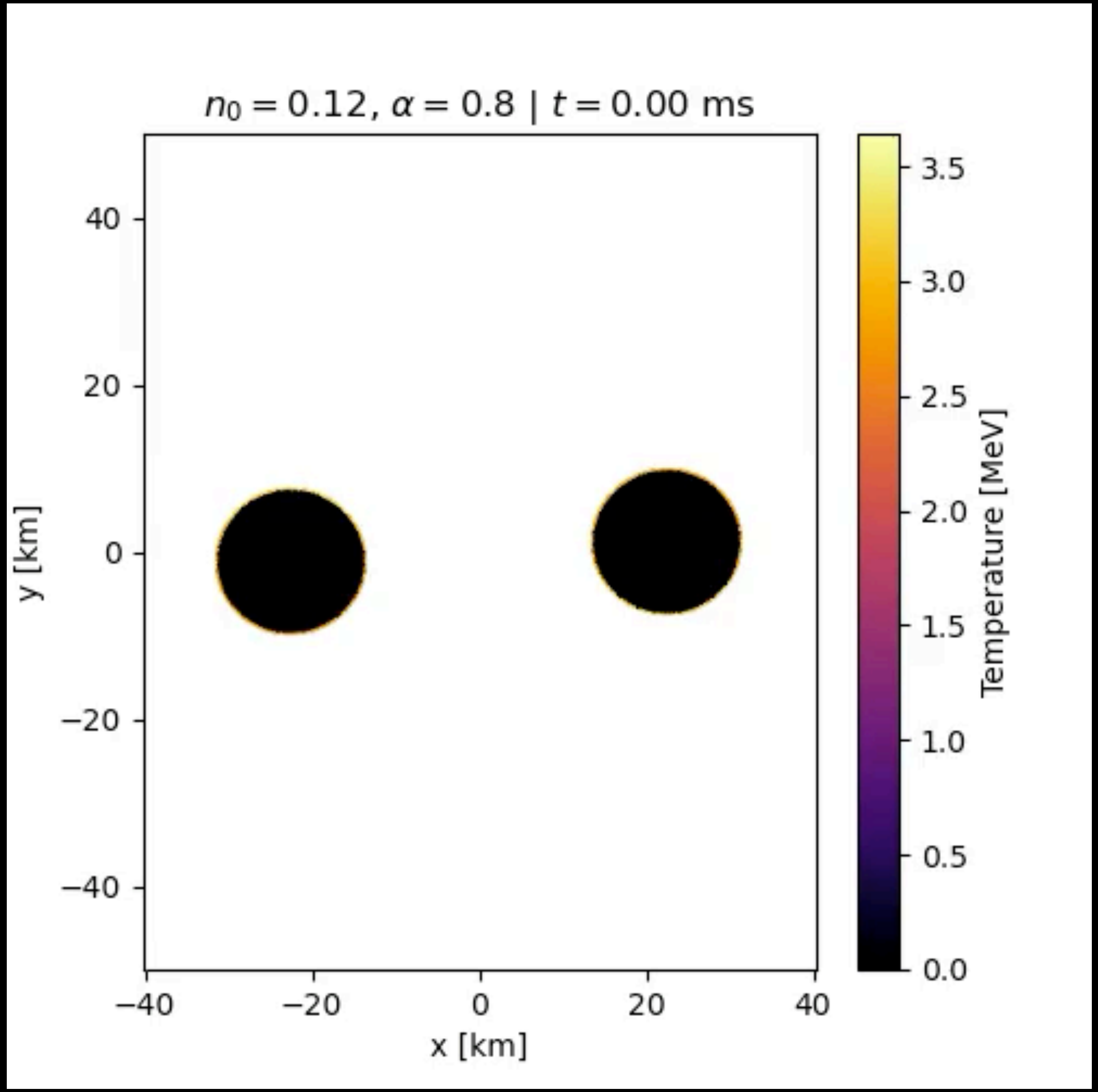


Density



Simulation with cold EOS with a temperature prescription

- binary: 2 x 1.3 $M_\odot$, irrotational
- spacetime: 7 (fixed) refinement levels
- fluid: 2 million SPH particles
- EOS: ENG (Engvik+ PRL 1994), temperature prescription (Raithel+ APJ 2019)



Bhaskas Biswas

arXiv:2601.01402

(Biswas, Rosswog, Diener, Scnabel)

Conclusions

- We have developed new strategies for fast+robust conservative-to-primitive transformation for tabulated EOS in GR-SPH code SPHINCS_BSSN.

- Successfully perform first BNS merger simulations with them.

Thank you