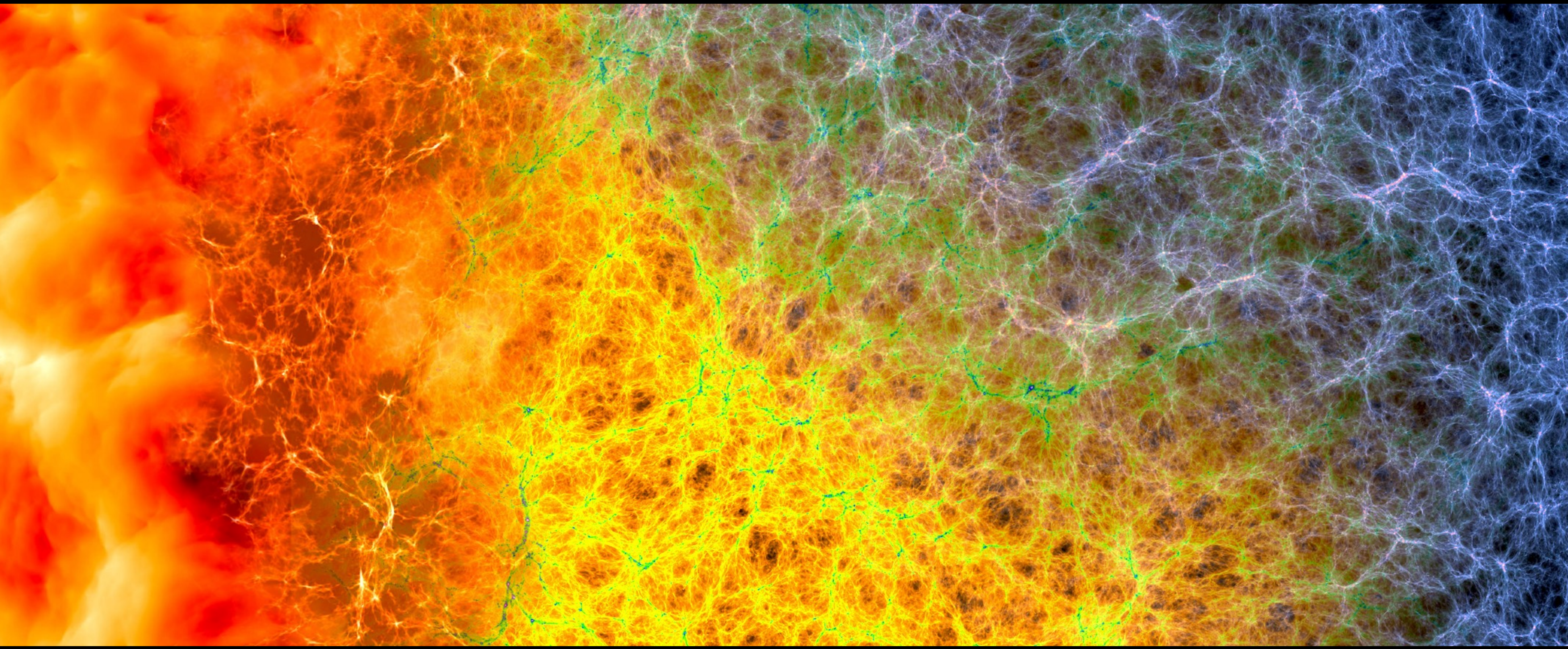
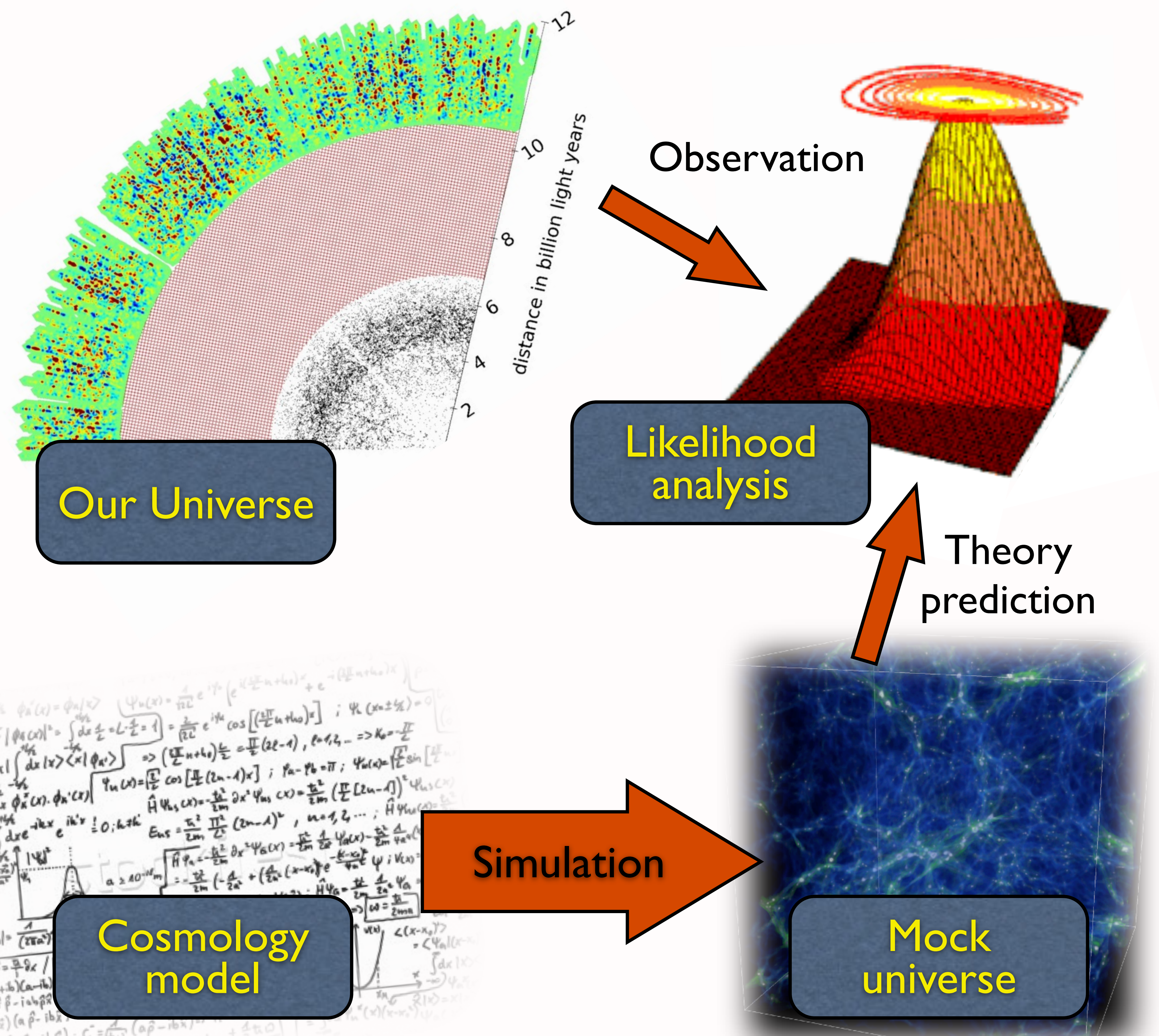
Nyx: cosmological hydrodynamical simulations at the Exascale



Zarija Lukić, zarija@lbl.gov
(Computational Cosmology Center)

The role of simulations in cosmology



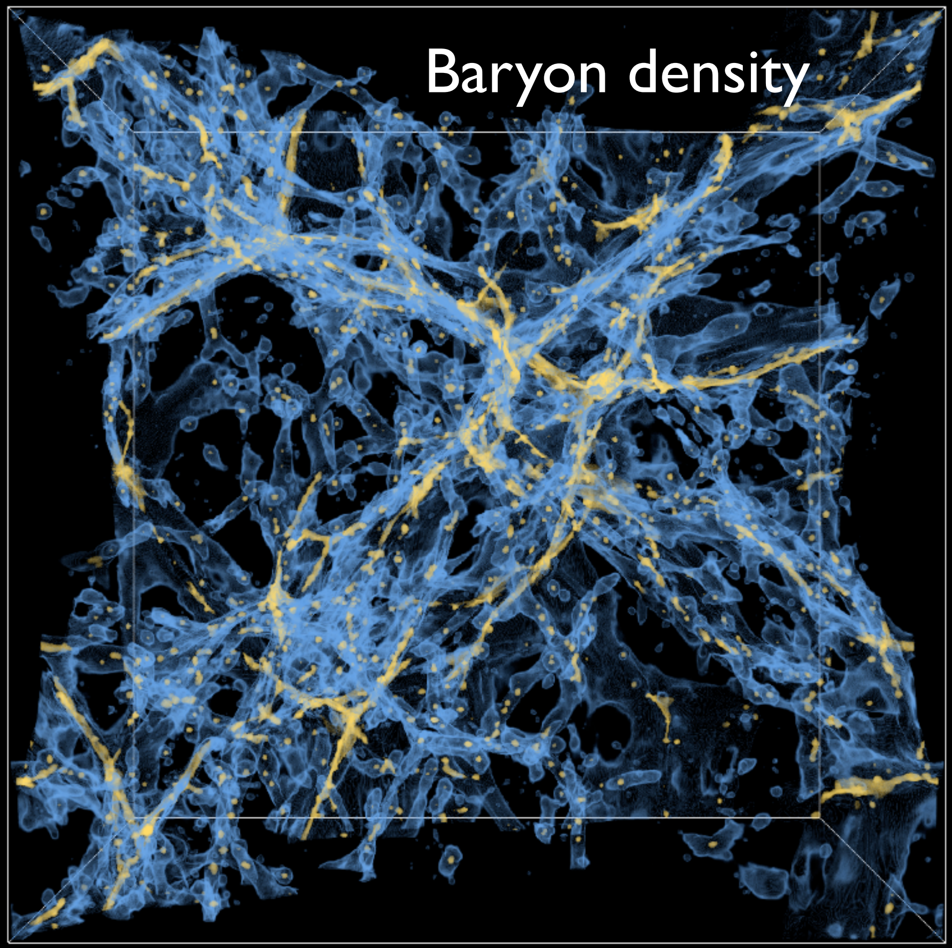
Inferring model parameters from experimental data is a grand challenge in many sciences, including cosmology. Approaching this inverse problem requires high fidelity numerical simulations, which are computationally very expensive.

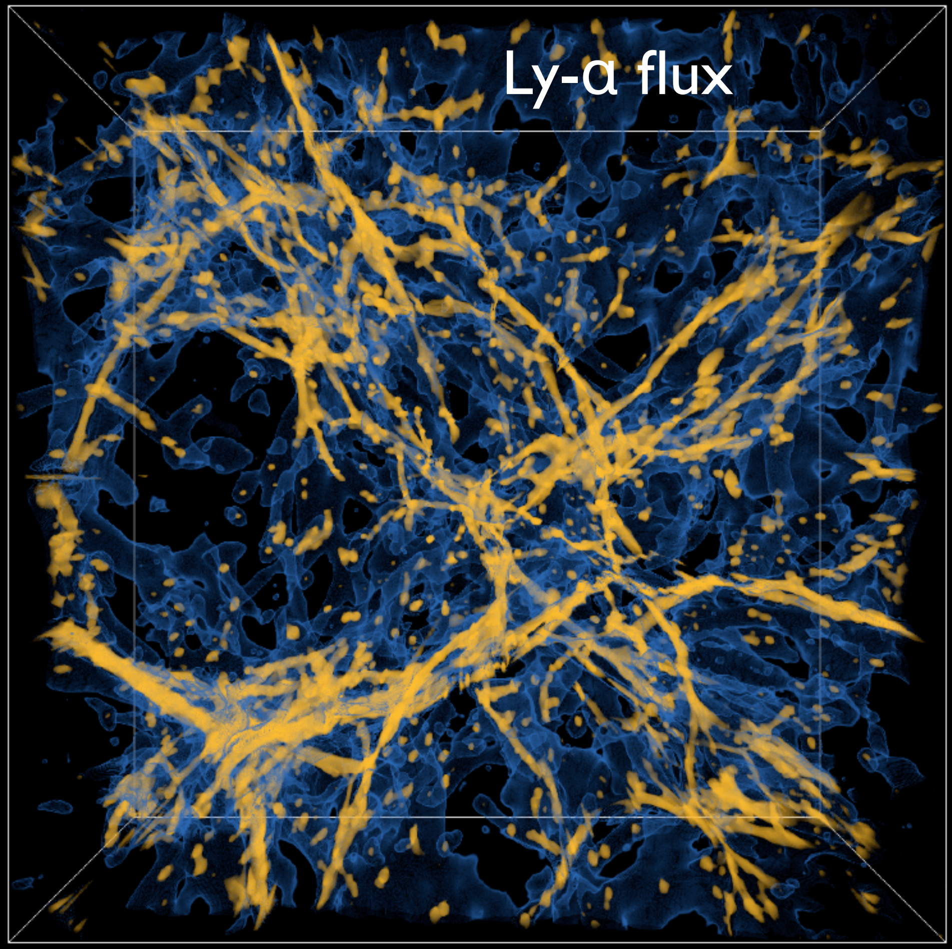
Building realistic mock skies for surveys: (1) Validation of analysis methods, algorithms, and statistical techniques; (2) Incorporate systematic effect and study their impact on observational data; (3) Forecast a survey's ability to constrain cosmological parameters, or decide on optimal strategy.

Critical component for extracting science from future sky surveys like DESI, LSST, WFIRST, and Euclid.

Nyx

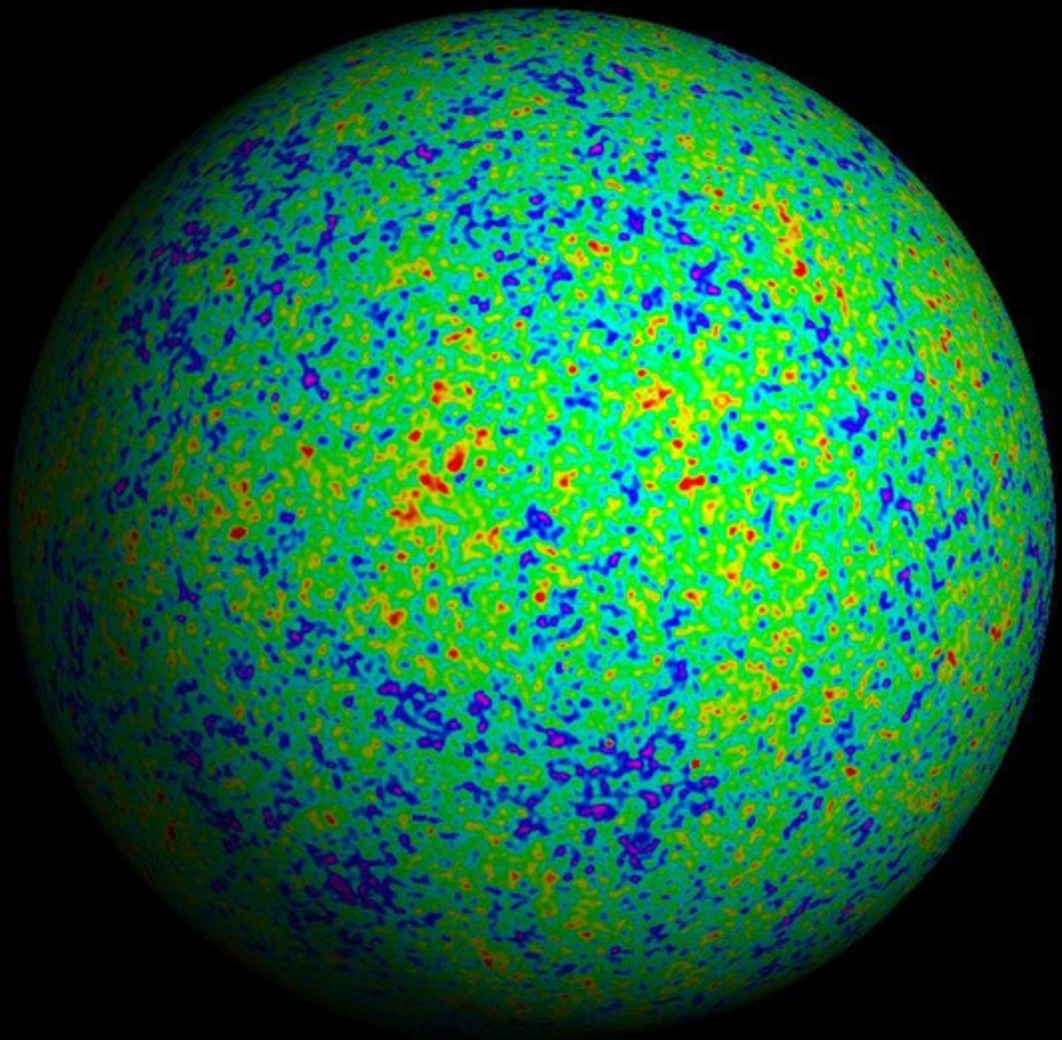
- 3-D Cartesian grid, finite volume representation
- Evolve dark matter as collisionless Lagrangian fluid (N-body particles)
- Evolve baryons as ideal gas on a grid using unsplit Godunov-type methodology
- Adaptive mesh refinement (AMR) to extend dynamic range
- Uses AMReX mesh framework developed at LBL
- Code paper: Almgren et al. (2013)
- Publicly available: https://github.com/AMReX-Astro/Nyx





Seeds of structure

- Inflation results in a universe close to homogeneous.
- Perturbations are produced with (close to) scale invariant power spectrum and Gaussian distribution of amplitudes.



- Linear theory works well while density perturbations are small:

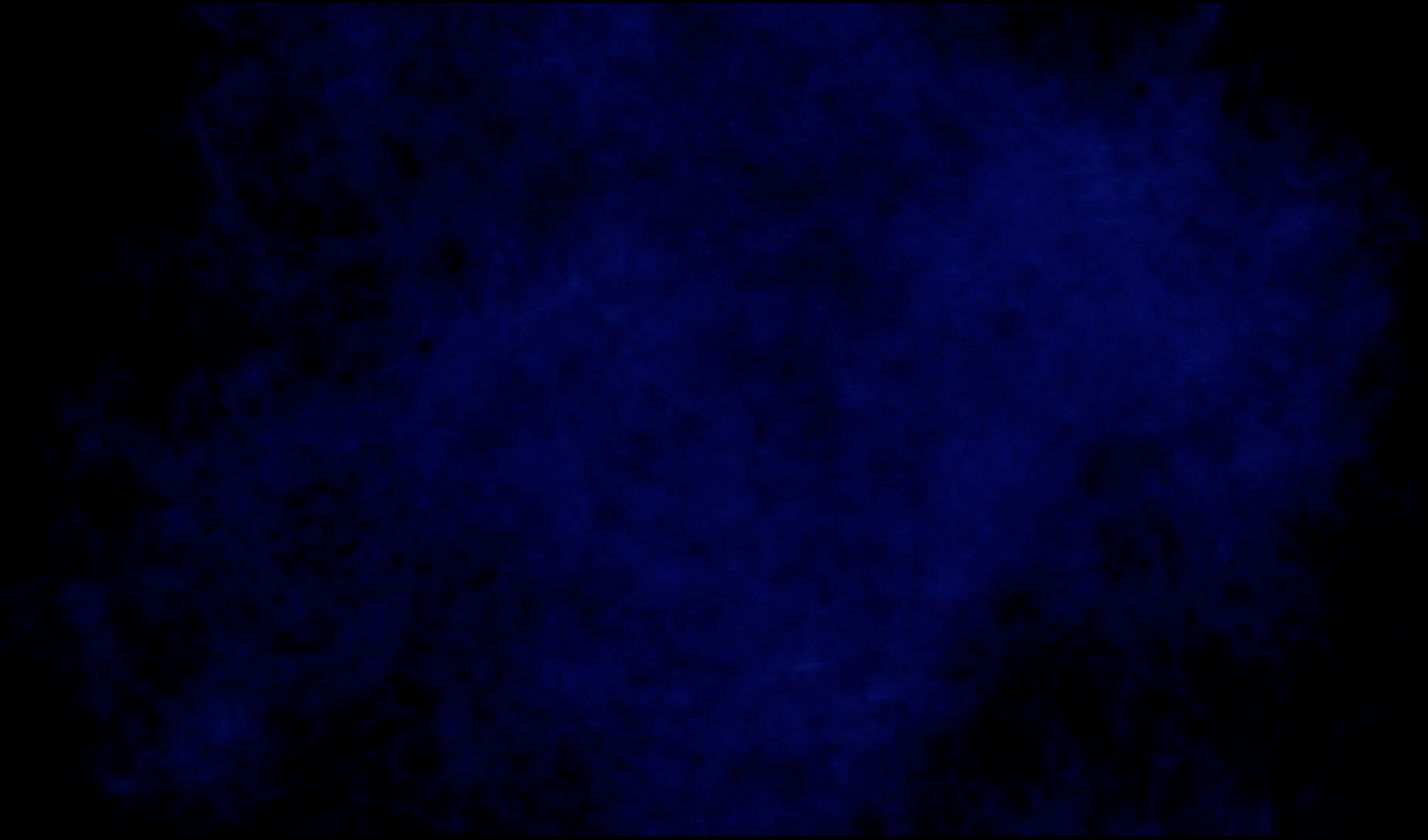
$$\delta_\lambda = \frac{\rho_\lambda - \rho_0}{\rho_0} \ll 1$$

$$\ddot{\delta}_k = \left[4\pi G\rho_0 - \frac{c_s^2 k^2}{a^2}\right]\delta_k - 2H\dot{\delta}_k$$

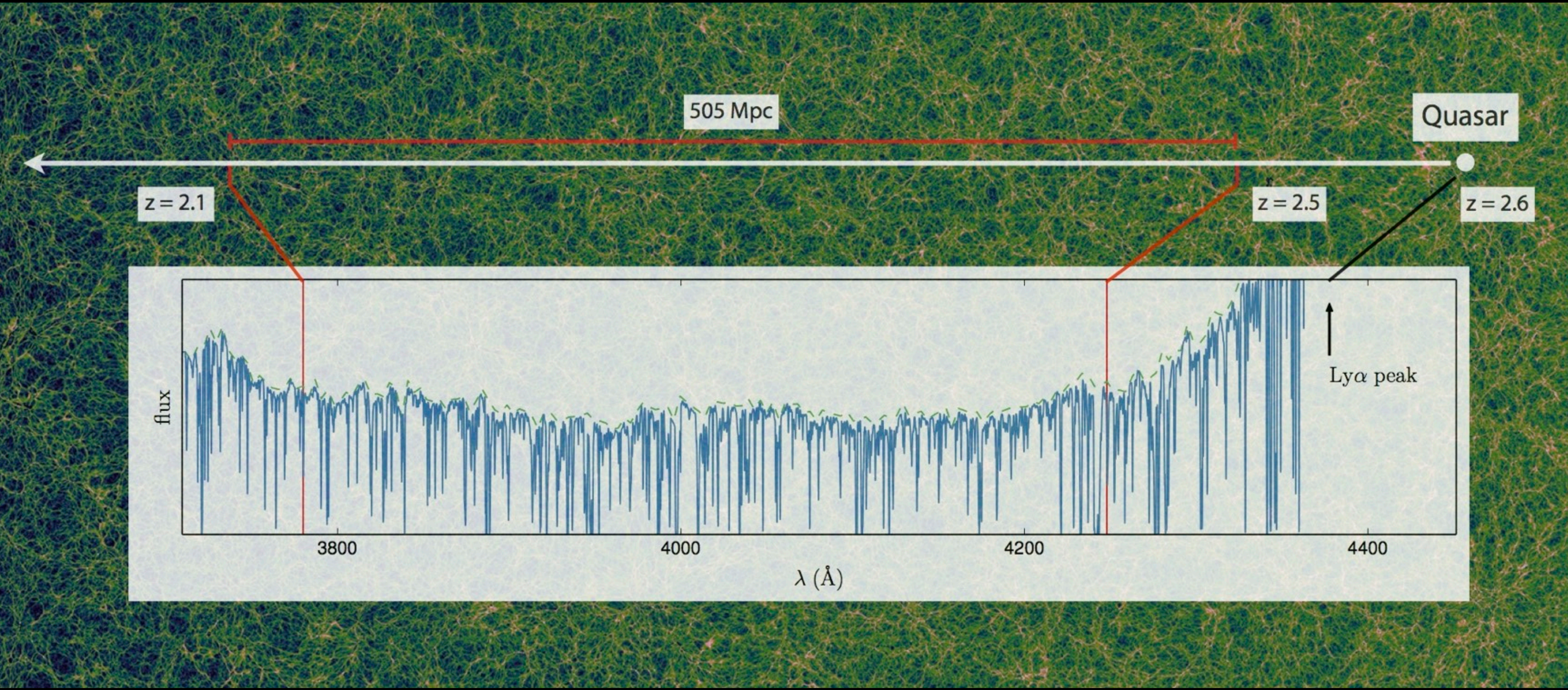
$$\Rightarrow \delta_k(t) = d_\pm \delta_k(t_0)$$

- The only accurate way to evolve density field in the nonlinear regime is via numerical simulations.

Nyx simulation: evolution of baryon density

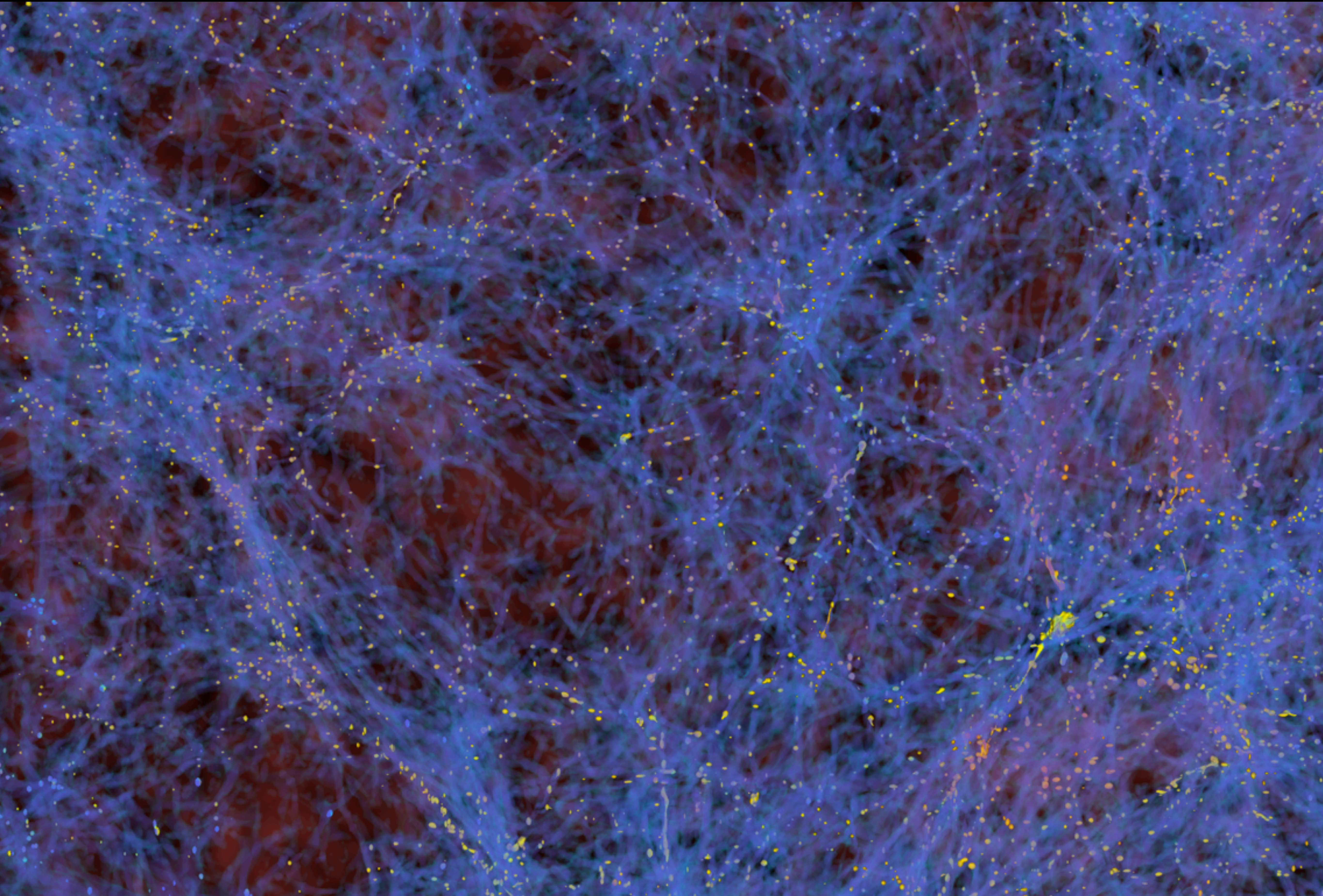


Lya “skewer”



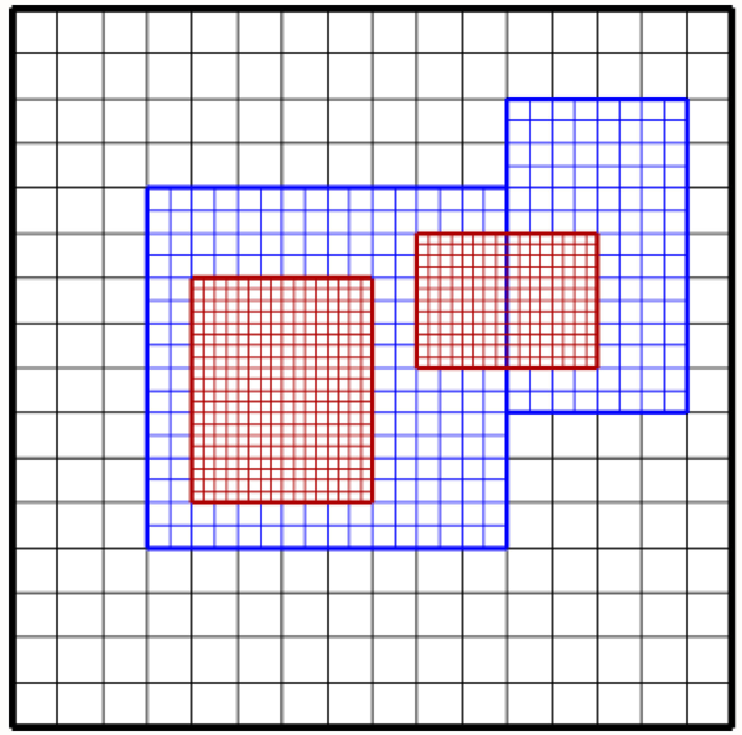
505 Mpc
Quasar
z = 2.1
z = 2.5
z = 2.6
flux
Lyα peak
3800
4000
4200
4400
λ (Å)

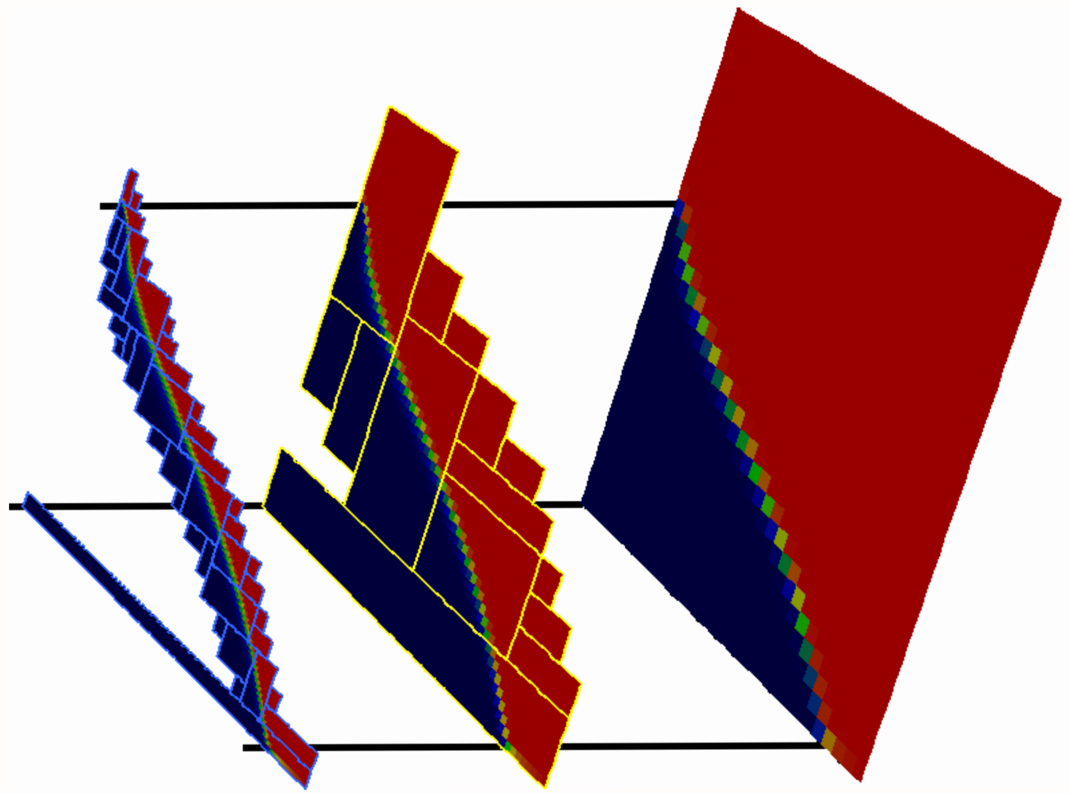
4096^3 hydro simulation (~100 Mpc/h) Blue: F~0; Red: F~1



Mesh-based (AMR) code

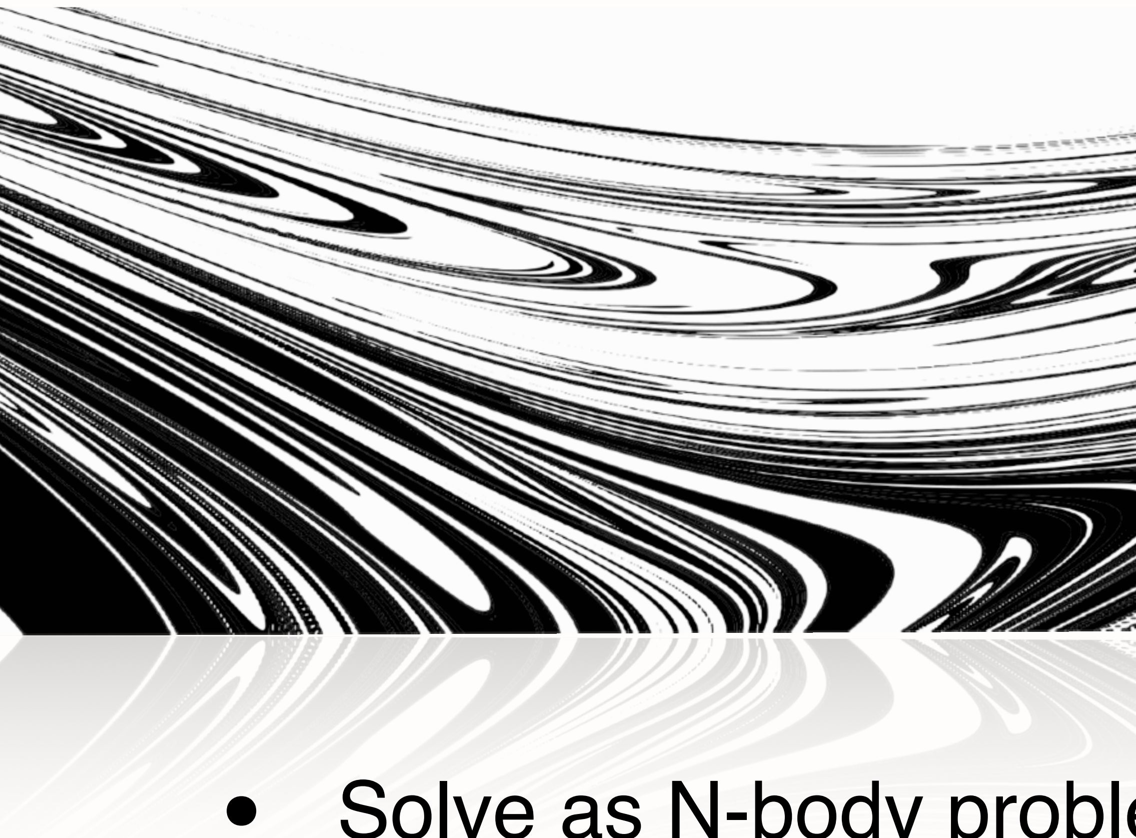
- AMReX framework
- Works as:
  1. Tag cells for refinement on a desired criteria
  2. Group cells into optimal rectangular grids
  3. Chunk grids & distribute them to processes
- Refinement factor 2 or 4
- No strict parent-child relation between patches





Dark matter

- Collisionless fluid evolving under self-gravity



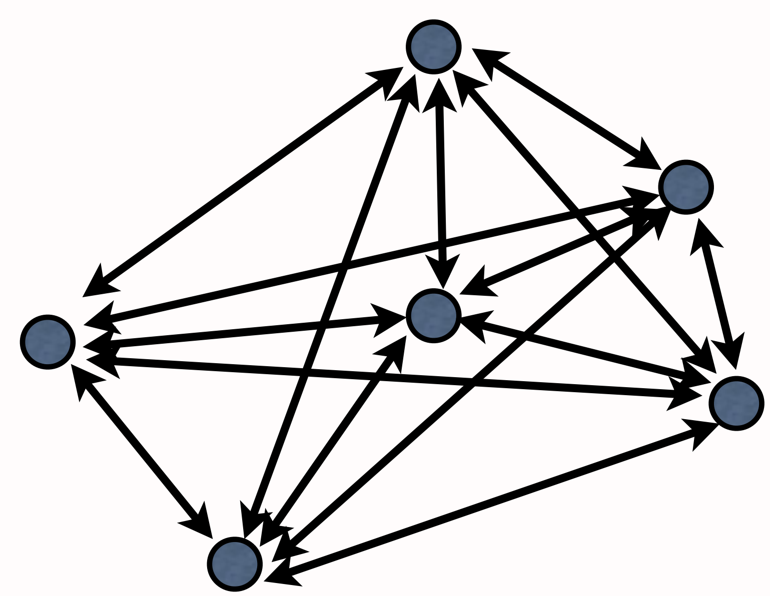
$$\frac{\partial f}{\partial t} + \frac{1}{ma^2}\mathbf{p}\cdot\nabla f - m\nabla\phi\cdot\frac{\partial f}{\partial \mathbf{p}} = 0$$

$$\nabla^2\phi = \frac{4\pi G}{a}(\rho_{tot} - \rho_0)$$

- Solve as N-body problem

$$\frac{d\mathbf{x}_i}{dt} = \frac{1}{a}\mathbf{u}_i$$

$$\frac{d(a\mathbf{u}_i)}{dt} = \mathbf{g}_i$$

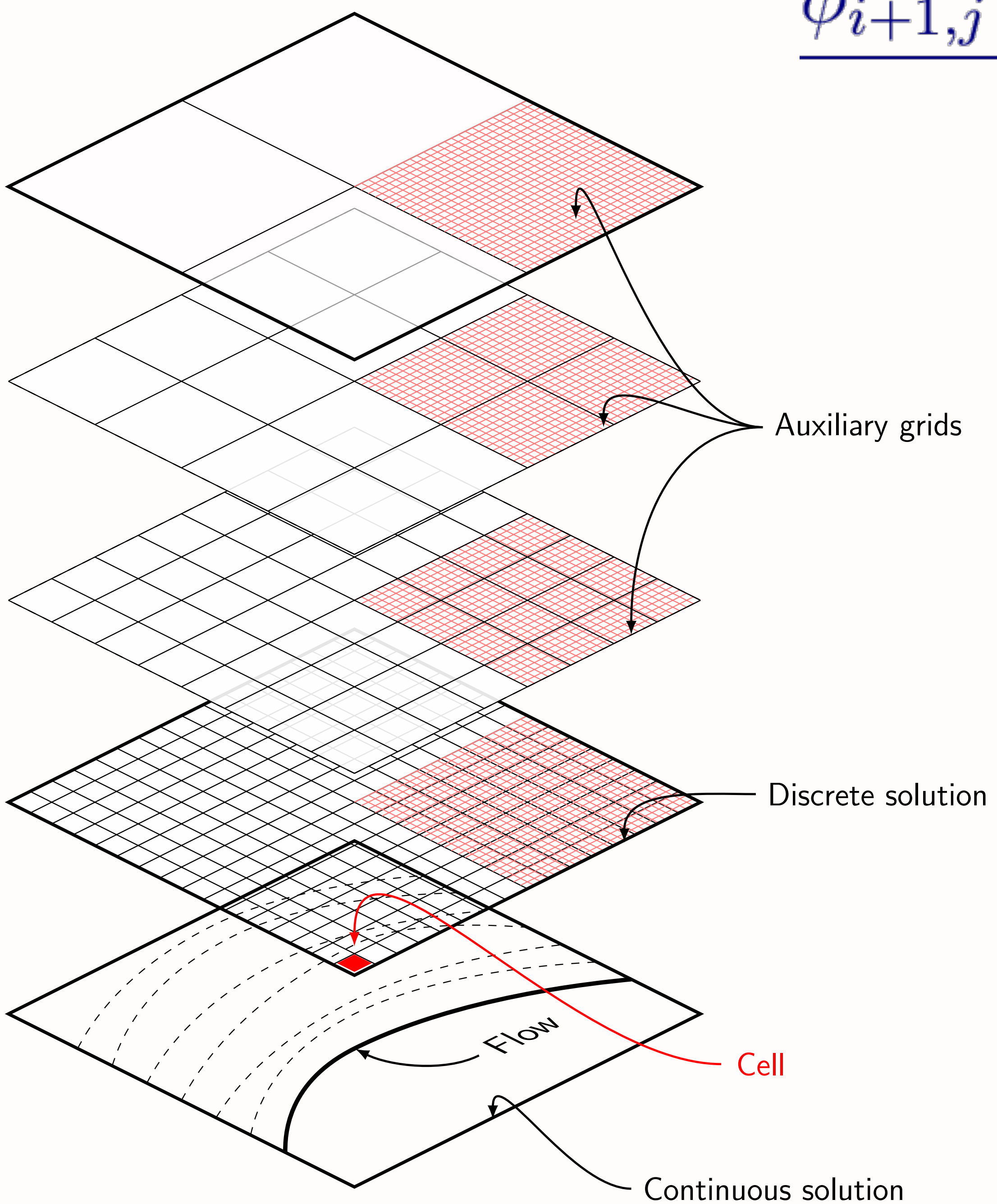


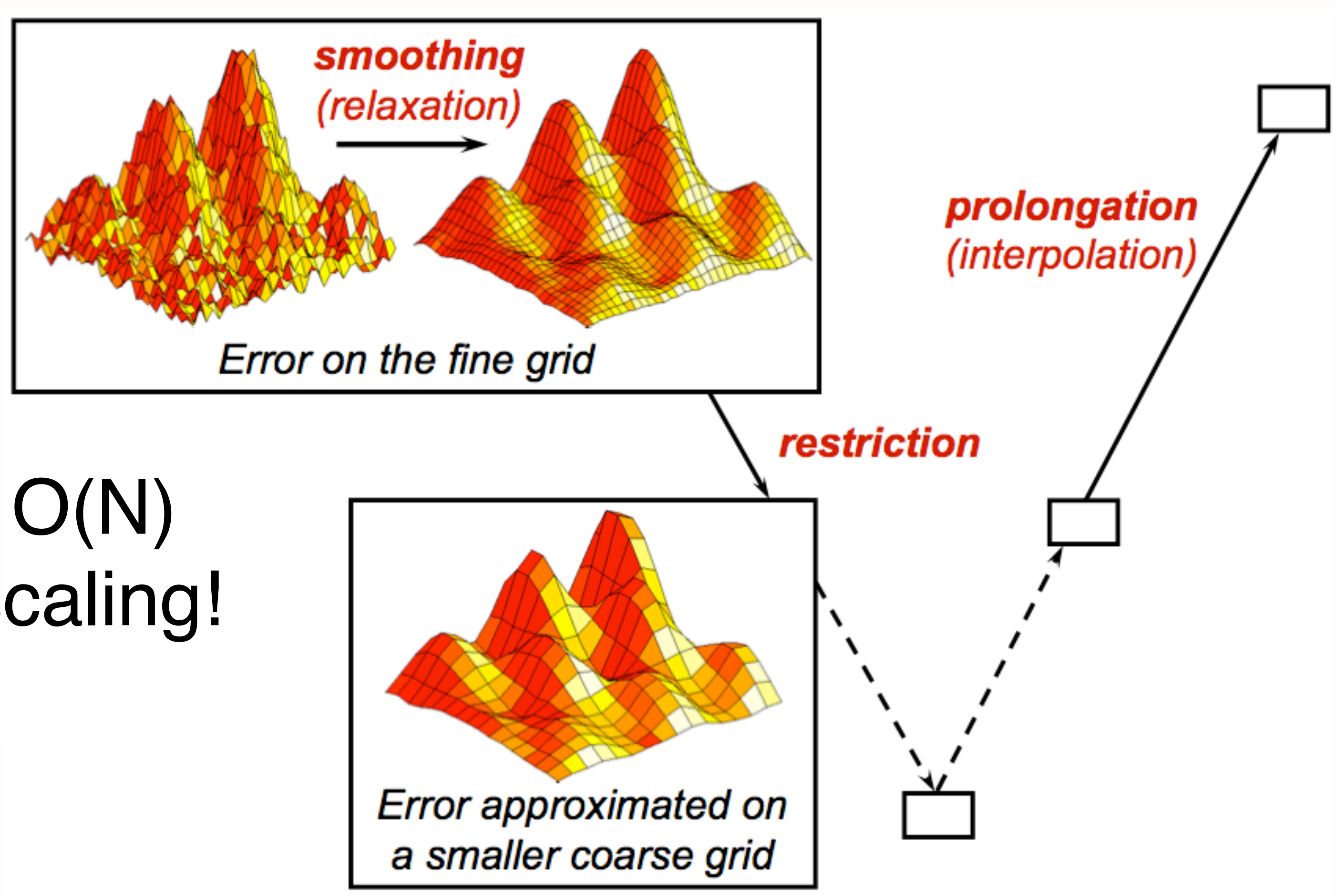
O(N^2)
scaling

Gravity solver

- Deposit mass on a grid, and solve linear system:

$$\frac{\phi_{i+1,j} + \phi_{i,j+1} - 4\phi_{i,j} + \phi_{i-1,j} + \phi_{i,j-1}}{\Delta x^2} = \rho_{i,j}$$



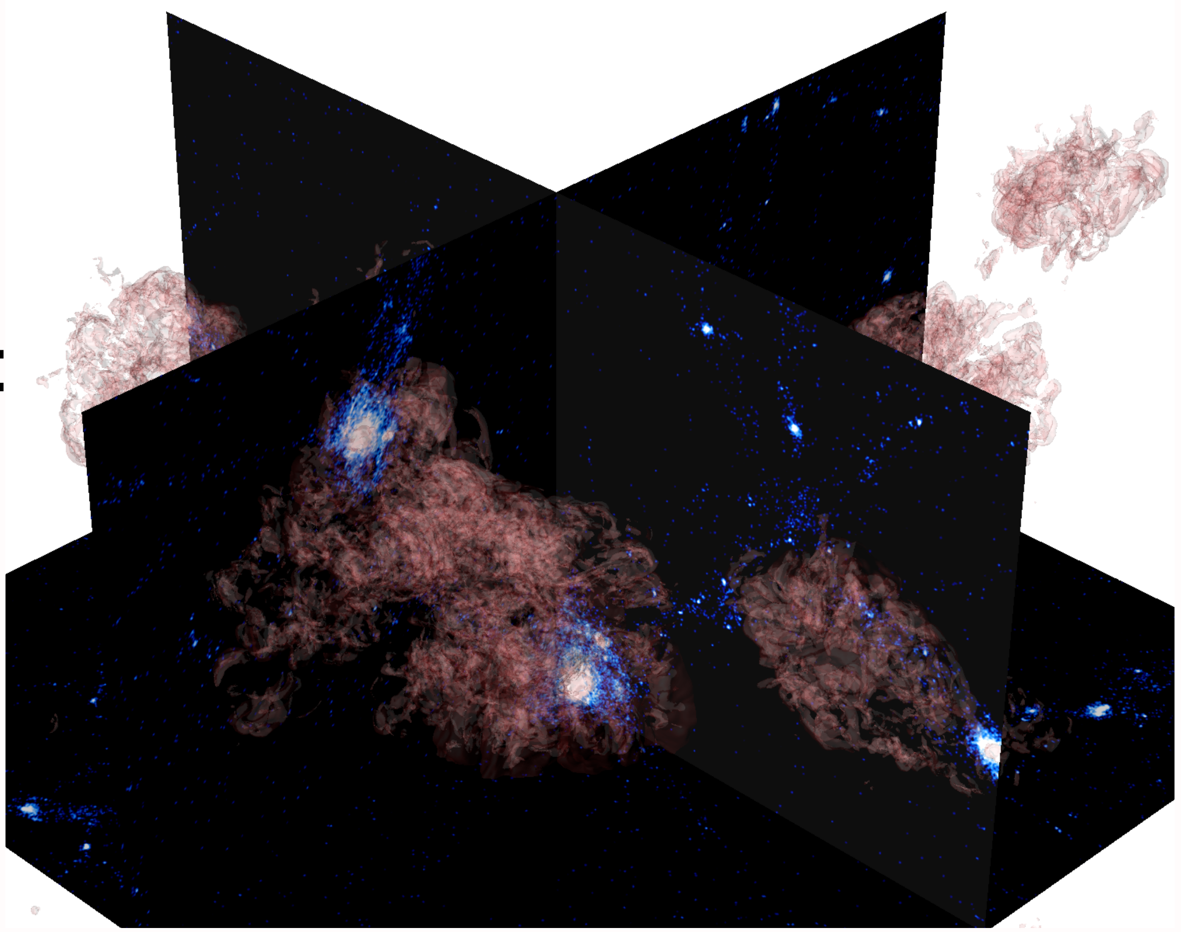


O(N) scaling!

Excellent approach for non-linear variants of Poisson equation

Baryons

- Baryons are modeled as inviscid ideal fluid
- Solve Euler equations of gasdynamics:



$$\frac{\partial \rho_b}{\partial t} = -\frac{1}{a}\nabla \cdot (\rho_b \mathbf{u})$$

$$\frac{\partial (a\rho_b \mathbf{u})}{\partial t} = -\nabla \cdot (\rho_b \mathbf{u}\mathbf{u}) - \nabla p + \rho_b \mathbf{g}$$

$$\frac{\partial (a^2\rho_b E)}{\partial t} = -a\nabla \cdot (\rho_b \mathbf{u} E + p\mathbf{u}) + a\rho_b \mathbf{u}\cdot\mathbf{g} + a\dot{a}\left((2-3(\gamma-1))\rho_b e\right) + a\Lambda_{HC}$$

with gamma-law ideal gas equation of state

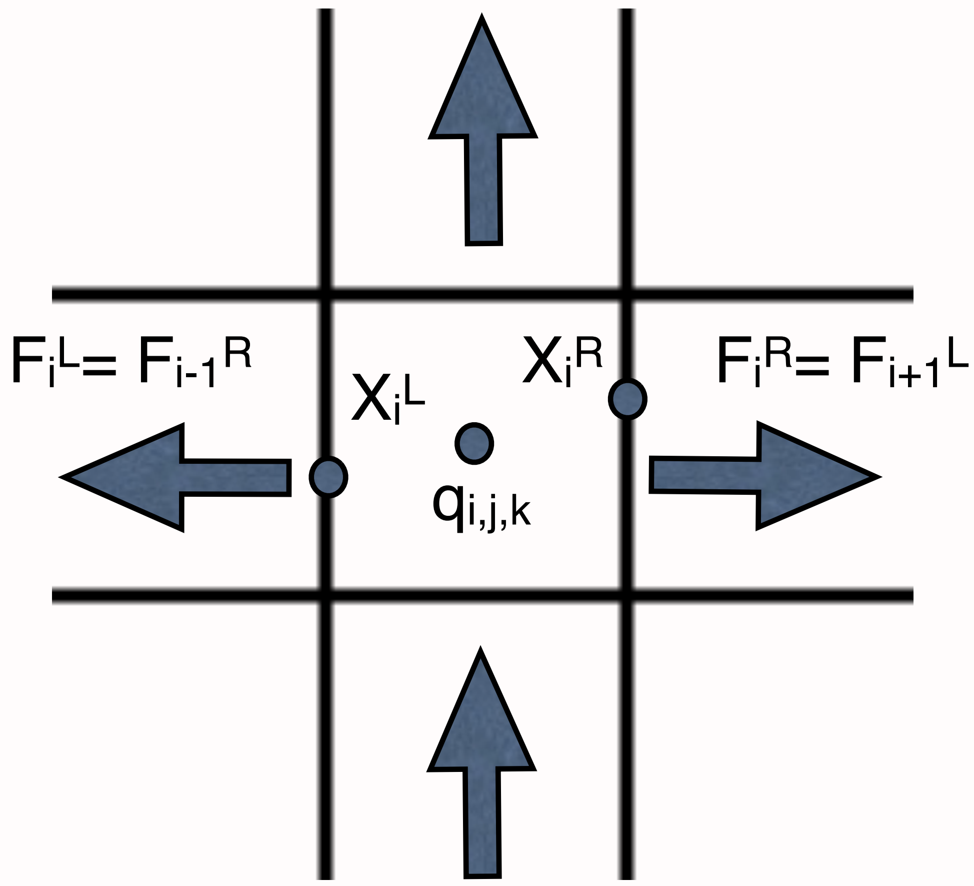
$$p = (\gamma - 1)\rho e$$

Finite volume method

- Calculate "face" values of primitive variables from cell averages using high-order interpolation
- Reconstruct profile of each variable within the cell
- Predict average values on edges over the time step using characteristic extrapolation.
- Compute fluxes by solving exact Riemann problem
- Use these fluxes to update solution to the next timestep

Conservation law:

$$\frac{\partial \mathbf{q}}{\partial t} = -\nabla \cdot \mathbf{F}(\mathbf{q}, t)$$



Atomic composition of the gas

- 2 primordial elements: H and He
- 6 ionic species: H_0, H_+, He_0, He_+, He_{++}, e^-

$$\frac{\mathrm{d}n_{\mathrm{H}_0}}{\mathrm{d}t} = \alpha_{\mathrm{H}_+}(T) n_{\mathrm{H}_+} n_{\mathrm{e}} - \Gamma_{\mathrm{eH}_0}(T) n_{\mathrm{e}} n_{\mathrm{H}_0} - \Gamma_{\gamma\mathrm{H}_0} n_{\mathrm{H}_0}$$

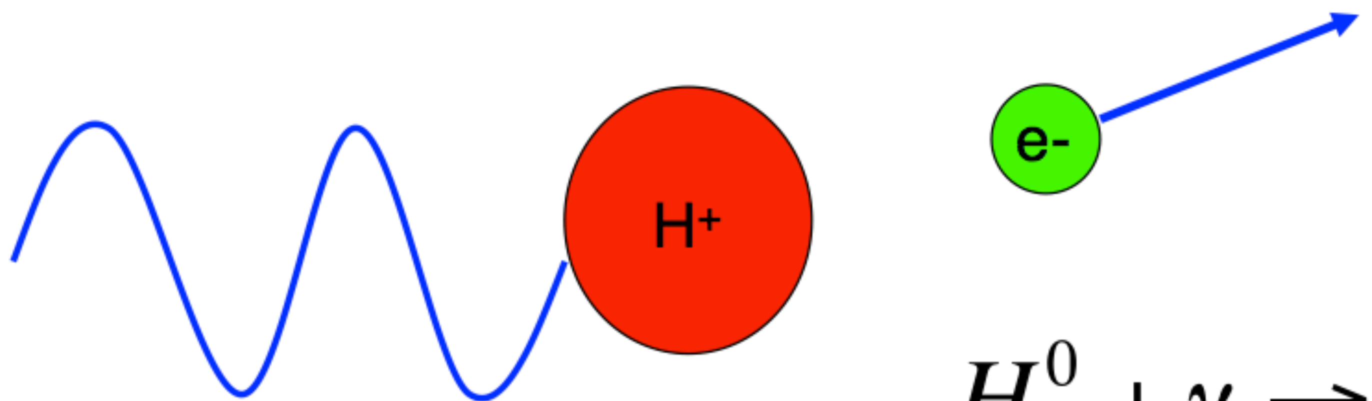
- Timescale on which species evolve:

$$t \sim \left| n \left(\frac{\mathrm{d}n}{\mathrm{d}t} \right)^{-1} \right| \sim \left| n_{\mathrm{e}} (\alpha_{\mathrm{H}_+}(T) - \Gamma_{\mathrm{eH}_0}(T)) - \Gamma_{\gamma\mathrm{H}_0} \right|^{-1}$$

For $z \sim 2,\ J(\nu) \sim \mathrm{few} \times 10^{-22} \mathrm{erg\,s^{-1}\,cm^{-2}\,sr^{-1}\,Hz^{-1}}$

t ~ few x 10^4 years

Photo-heating of the intergalactic gas



E > 13.6 eV photon

$$H^0 + \gamma \rightarrow H^+ + e^-$$

- Mean excess energy of ionizing photon for $J_\nu \propto \nu^{-\beta}$ (Abel & Haehnelt 1999):

$$\langle E \rangle = \frac{h\nu_i}{\beta + 2}$$

- Low density IGM in ionization equilibrium (Miralda-Escudé & Rees 1994):

$$\frac{dT}{dt} = \frac{2}{3k_\mathrm{B}} \langle E \rangle \alpha(T) n - 2HT$$

Modeling the reionization

- **Approach:**
    1. Linearly evolve initial conditions and find halo abundance at different times
    2. Assuming photon emissivity $\eta(M)$, find z_{reion} for each computational cell
    3. Evolve gas in Nyx from initial conditions, and for each cell turn UV background at its own z_{reion}
    4. Inject ΔT amount of heat in one timestep at z_{reion}

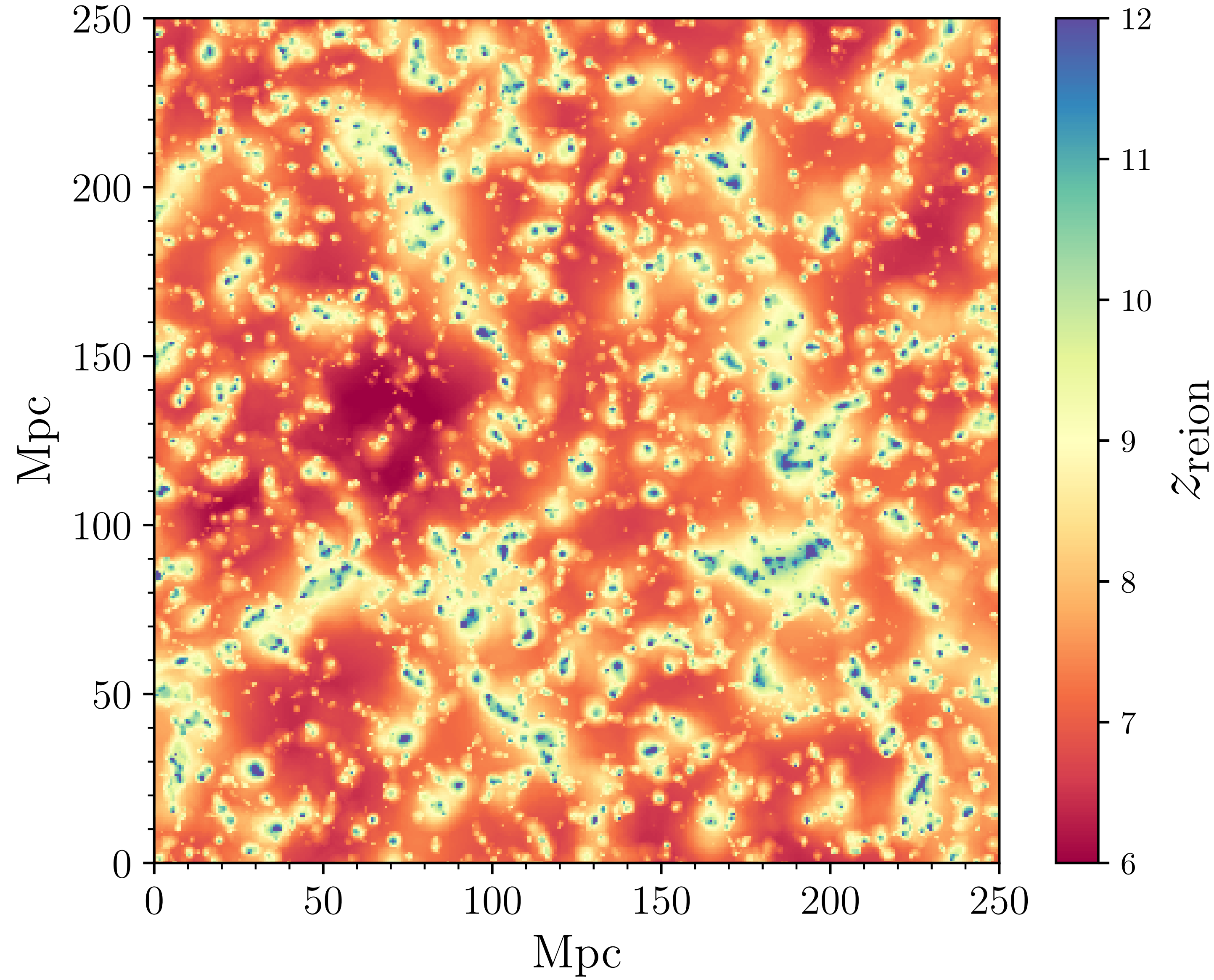
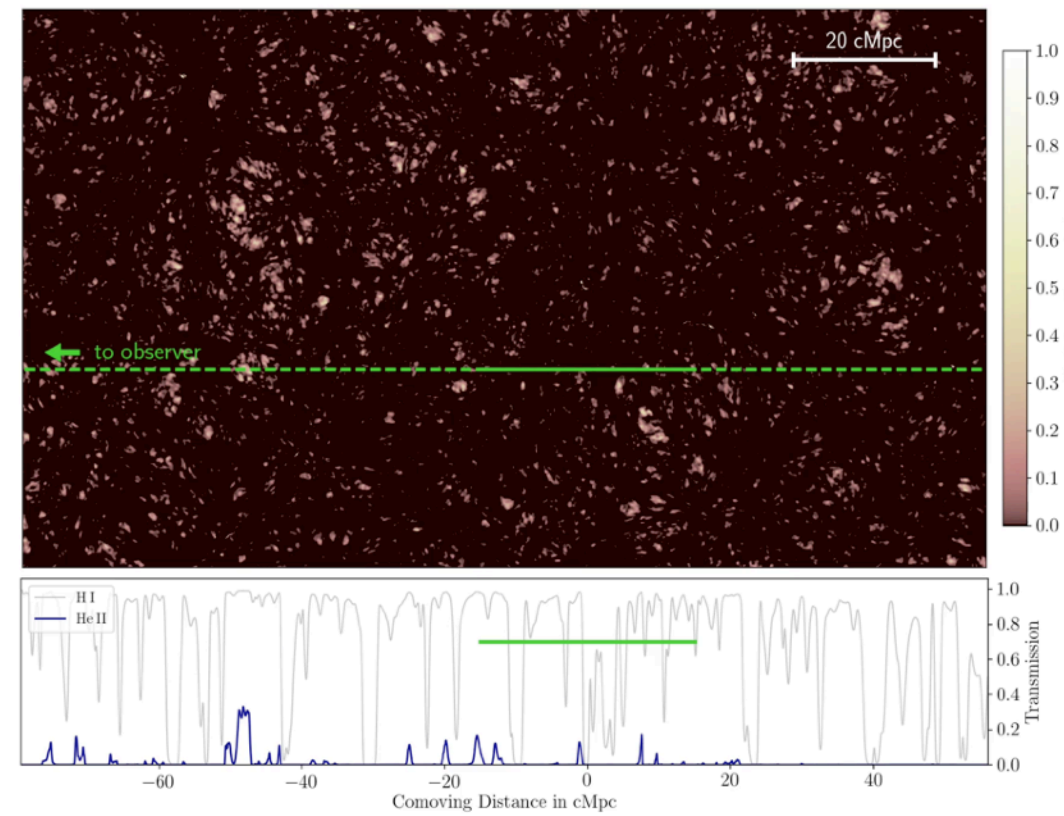


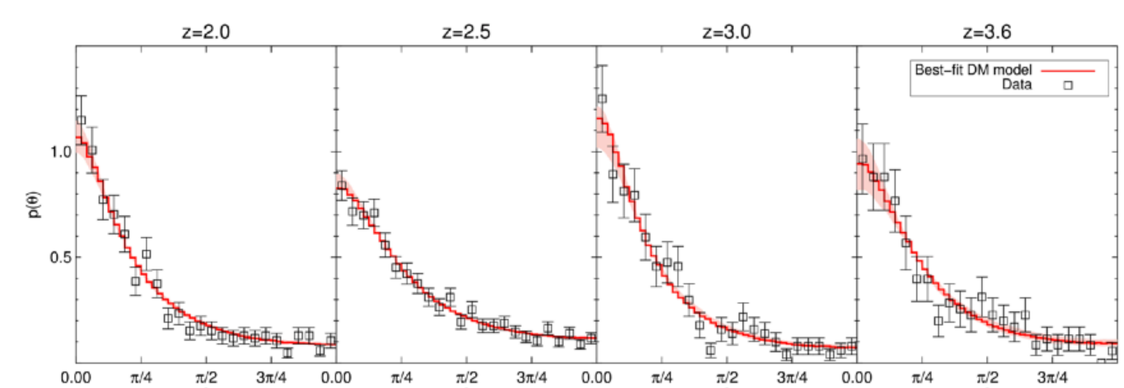
Fig: Redshifts of reionization in one slice of 8192^3 Nyx simulation done on Cori-KNL. Blue parts have ionized earlier in time, red parts later.

Selected Nyx-based results

- Tomographic reconstruction of high-z cosmic structure (Stark et al. 2015, KG Lee++)
- Detection of voids (Krolewski et al. 2018)
- Constraints on the reionization history (Davies et al. 2018)
- Measurements of density-temperature relation (Hiss et al. 2018)
- Measurement of small-scale fluctuations in the baryonic gas (Rorai et al. 2017)
- Inferring post-reionization ionizing background (Davies et al. 2018)
- Constraining quasar lifetimes (Schmidt et al. 2018)
- Inferring thermal evolution of the IGM gas at high (Walther et al. 2019, Wolfson et al. 2021) and low redshifts (Khaire et al. 2019, Hu et al. 2022)
- Establishing simulation requirements (Walther et al. 2021, Chabanier et al. 2023)
- Running simulations for DESI-Lya cosmological inference (ongoing)



Schmidt et al. Astrophysical Journal, 861, 122 (2018)

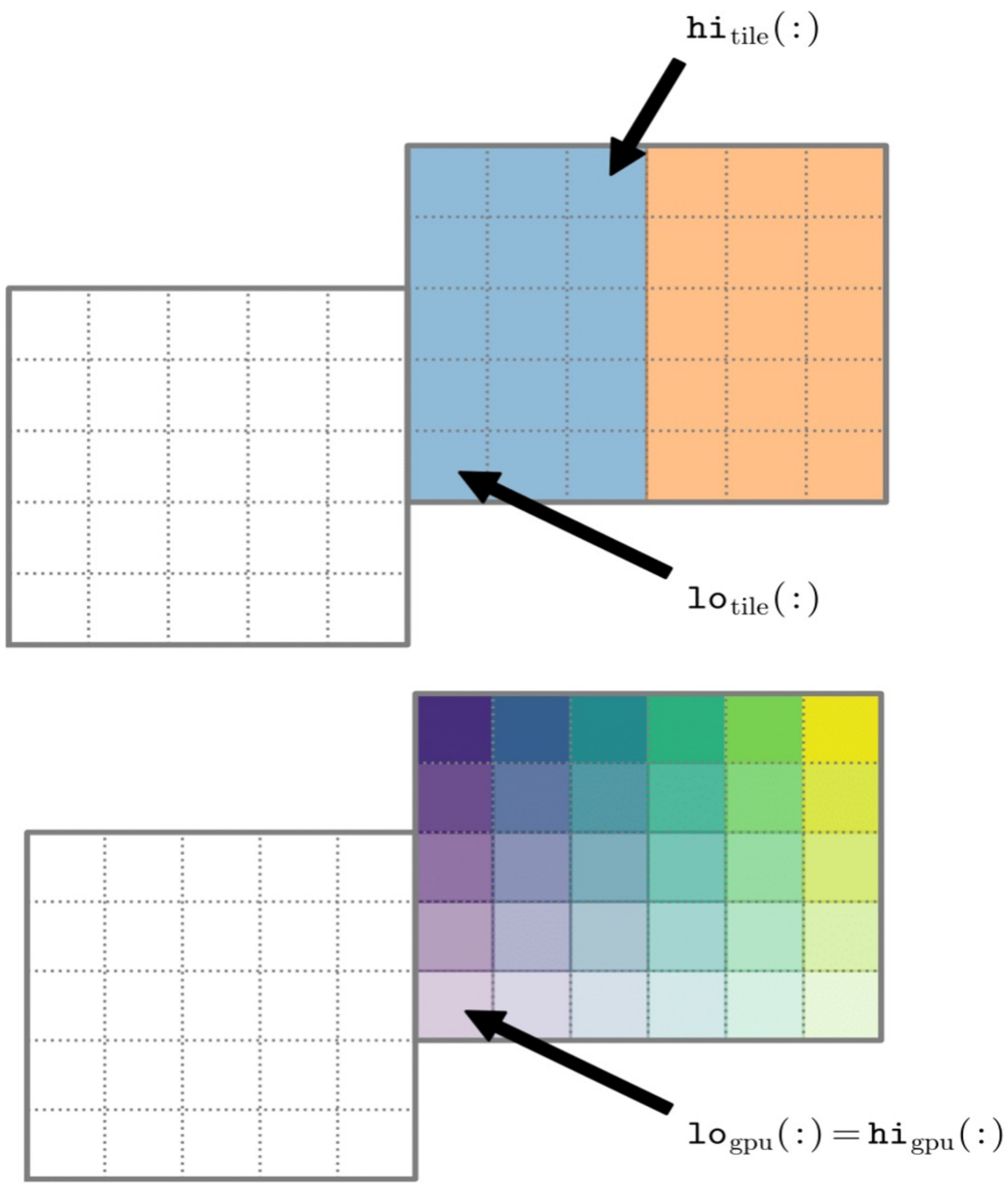


Rorai et al. Science, 356, 418 (2017)

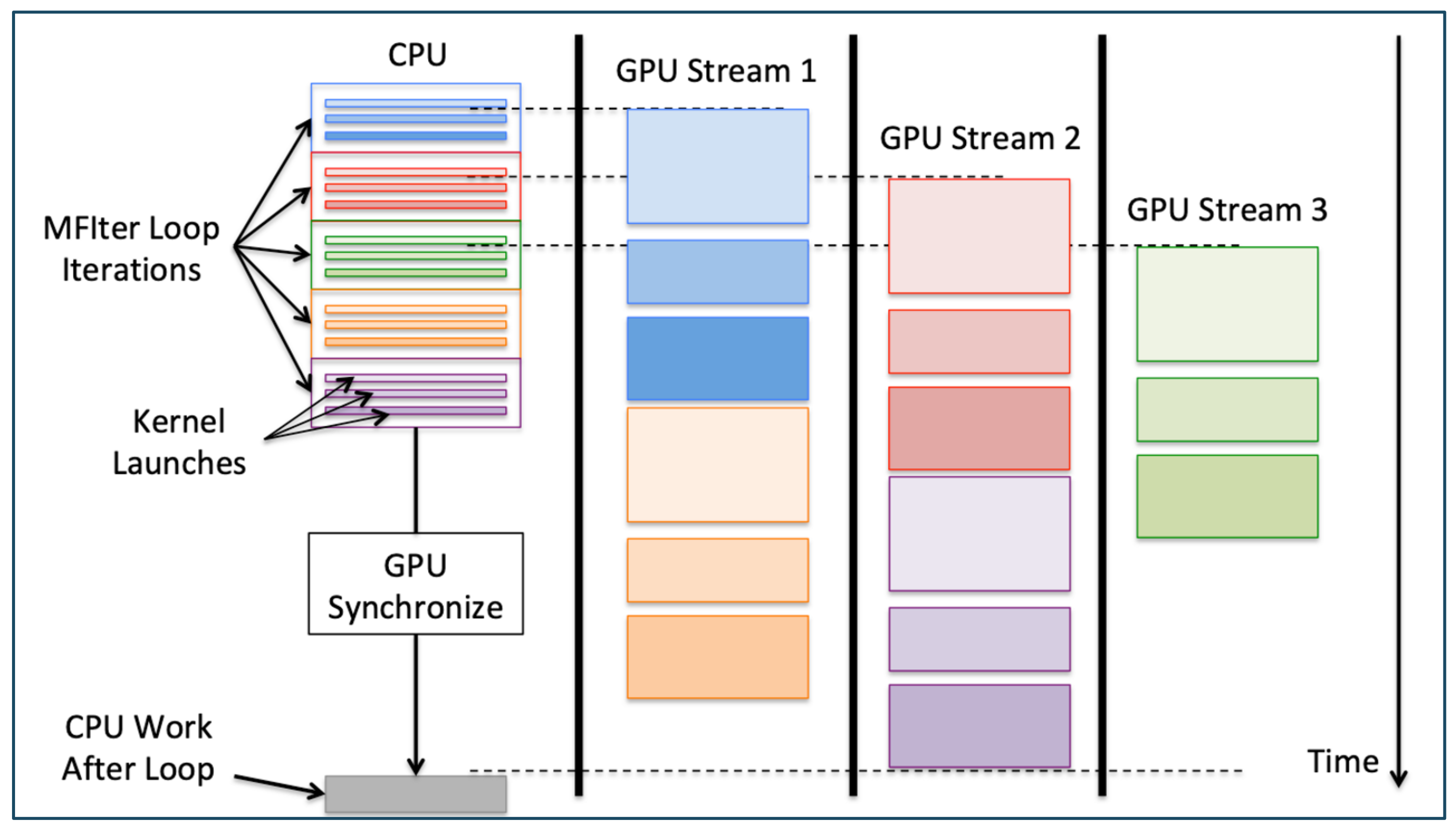
Nyx/AMReX GPU strategy

- **Big picture:**
  - C++ transitioned away from Fortran interfaces
  - Cuda/HIP/DPC++ — better control and performance
  - Compatibility with other software frameworks: OpenMP, OpenACC, Kokkos launches, mdspan
  - Interfaces with other open-source libraries: Sundials, HDF5, compression, etc.
  - Asynchronous I/O
- **APIs:**
  - Designed to be consistent and easy-to-read
  - Mesh and particle data iterators
  - Kernel launching
  - Reductions
  - Parallel communication
- **Under the hood:**
  - Portable and performant code that the user doesn't need to directly interact
  - Loop constructs
  - Memory arenas and memory allocation calls
  - Temporary object handlers
- **Ongoing improvements:**
  - New GPU load balancing approach — dual grid for particles
  - Nyx-RT!
  - Further optimizations for Intel GPUs
  - Fine-level performance improvements
  - Architecture-dependent code improvements

Decomposition on GPUs



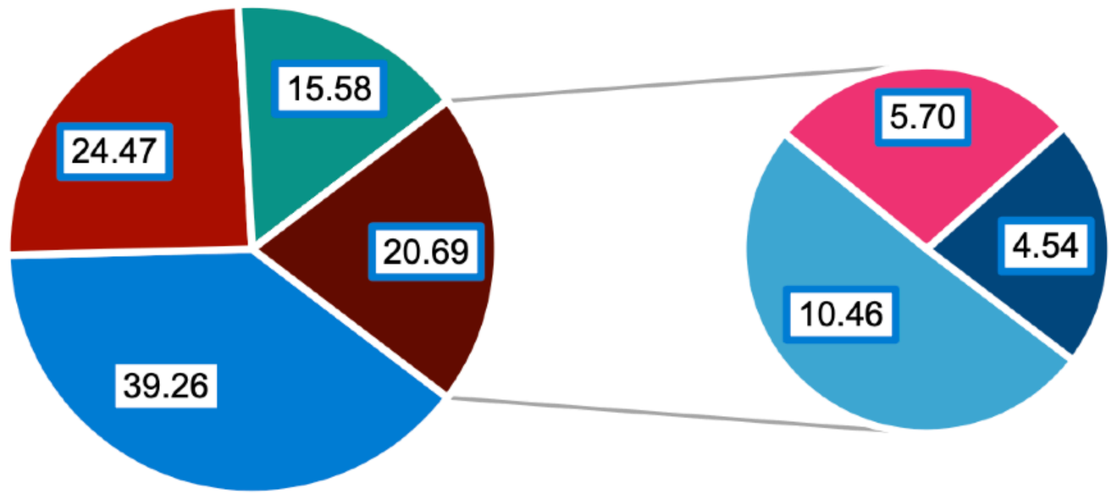
- **OpenMP tiled box:** each thread gets a tile of cells or a ParticleTile
- **CUDA threaded box:**
  - Each thread gets a cell launched a box at a time
  - Each thread gets a particle, launched a box at a time
- Kernels are also launched asynchronously on separate **CUDA streams**, to run as many boxes simultaneously as possible.



Performance profiling

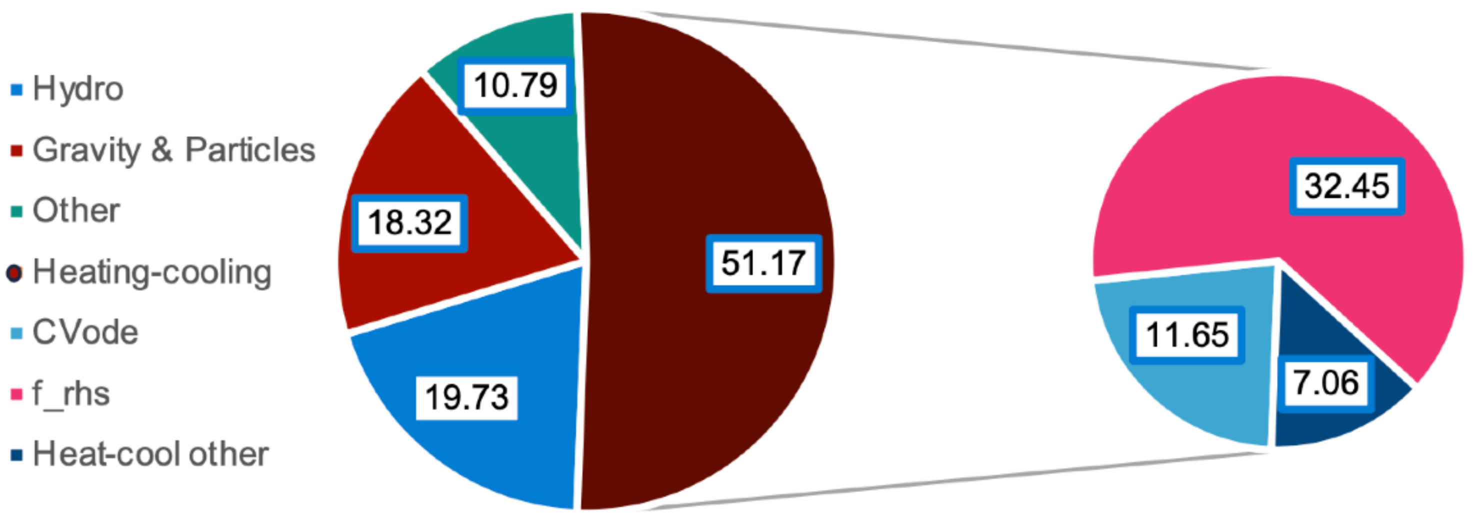
`From` early simulation time (z=200)

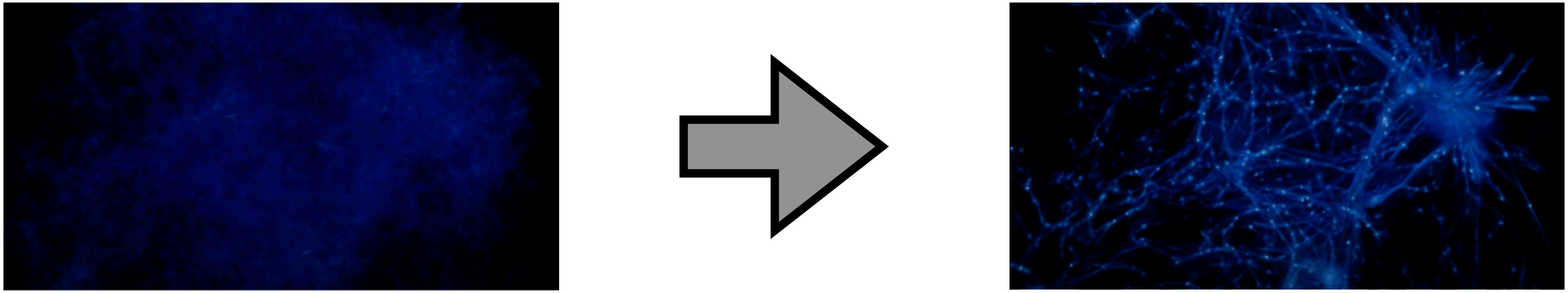
Heating-cooling ≅21% of timestep time
Hydro ≅39% of timestep time
Average 3.83286 seconds per step
17508822.13 cells per second per rank



`From` later simulation time (z=7)

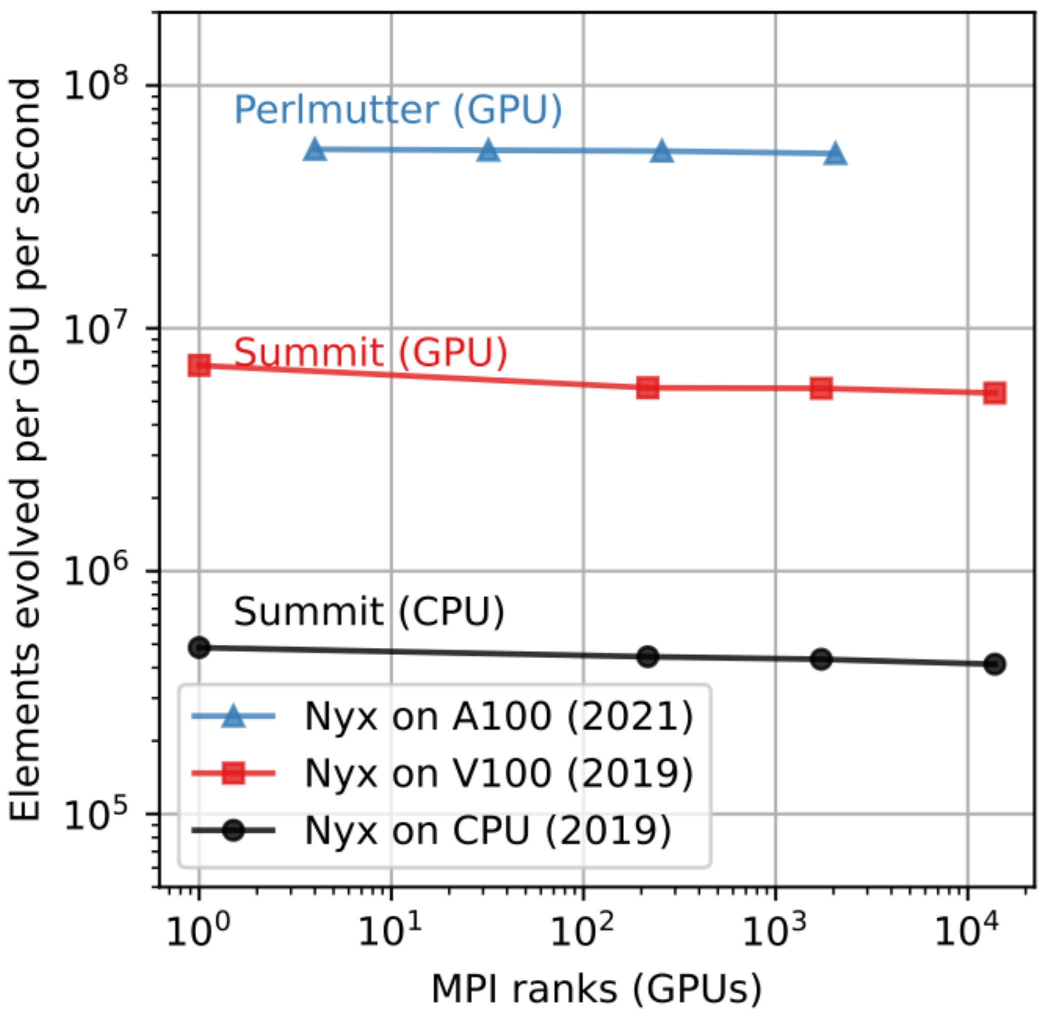
Heating-cooling ≅51% of timestep time
Hydro ≅20% of timestep time
Average 7.82608 seconds per step
8575029.13 cells per second per rank





Exascale performance

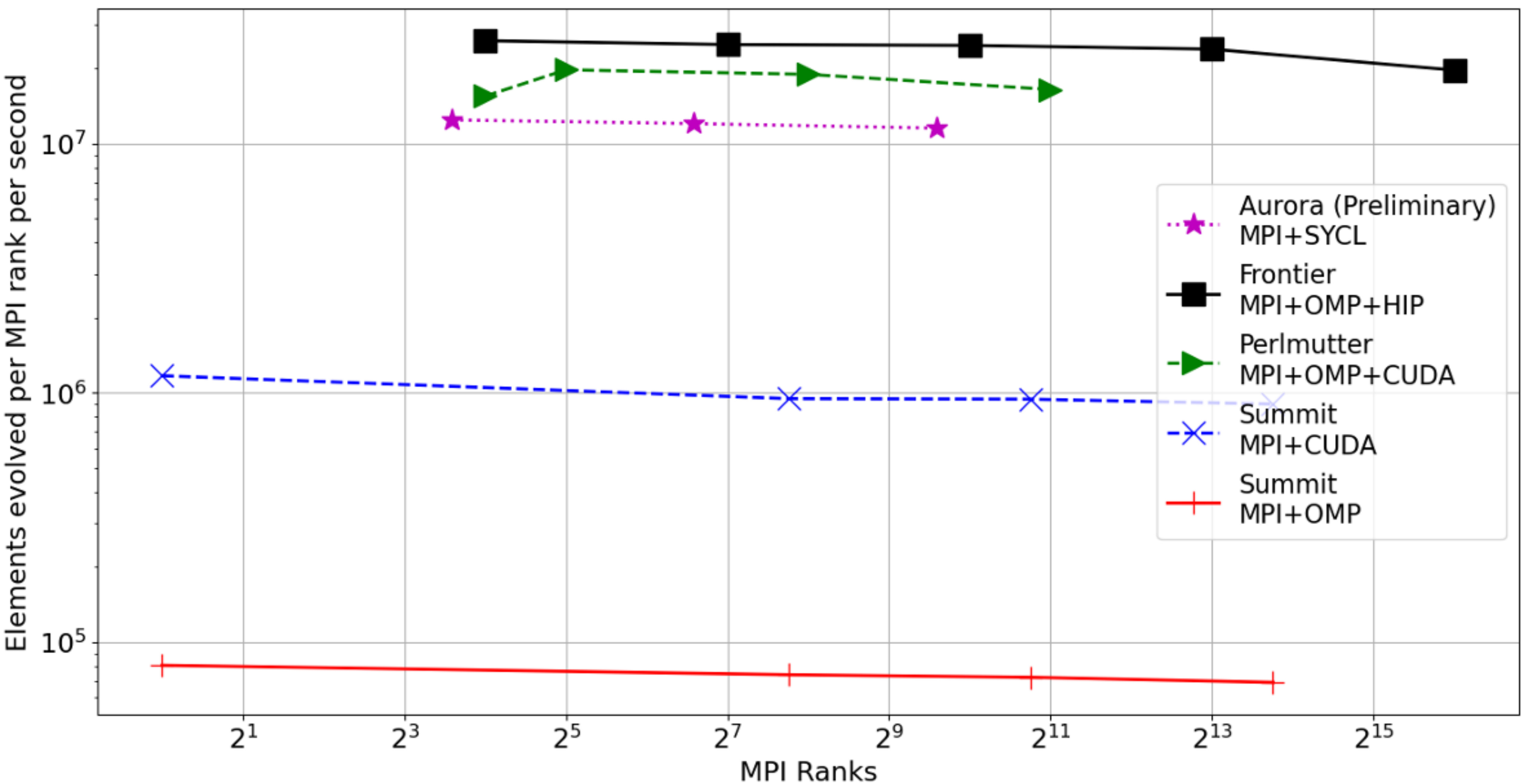
- **What:**
  - We have enabled running Nyx hydrodynamical simulations at-scale on all existing architectures (NVIDIA/Cuda, AMD/HIP, Intel/DPC++)
- **Why:**
  - High fidelity hydrodynamical simulations with high dynamical range are essential for establishing "the truth" in different (astro)physics scenarios. This is needed to extract science from sky surveys, including DESI and VRO/LSST
- **Details:**
  - Ported the solvers from F90 to C++
  - We use MPI to distribute grids across nodes, and OpenMP and/or Cuda/HIP/DPC++ to spread the work over GPU "cores"
  - Full set of physics needed to model intergalactic medium is included, covering also different models of inhomogeneous reionization
  - The code is publicly available on GitHub: https://github.com/AMReX-Astro/Nyx



Weak-scaling of Nyx on different architectures. We are able to fully utilize the power of modern GPUs and maintain parallel scaling going to the full machine size.

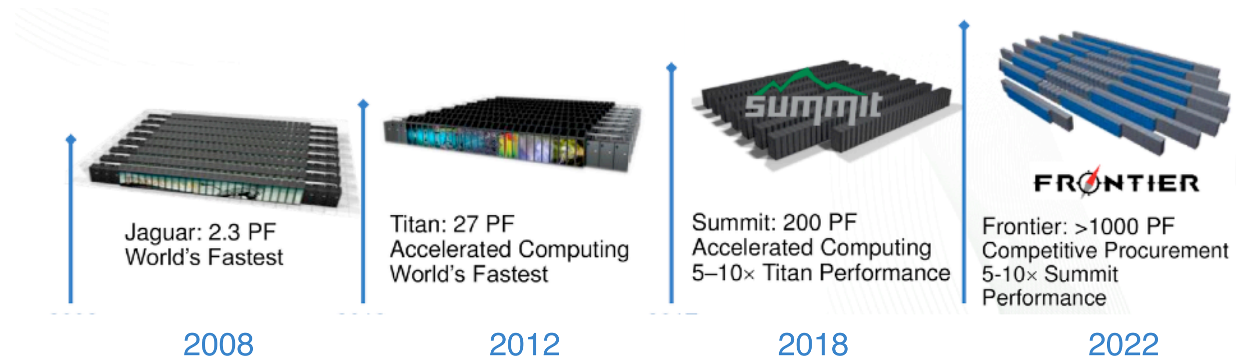
EXASCALE COMPUTING PROJECT

Weak scaling



First exascale system

- **Frontier at OLCF:**
  - We have access under the INCITE program
  - 1.1 Exaflop with 75TB/s read and 35TB/s write internal flash storage system, and a 700PB Lustre filesystem
  - 9,408 nodes; each node has 3rd Gen AMD EPYC CPU with 4 AMD MI250X GPUs
  - Factor of ~20 times improvement over the Perlmutter GPU system
- **For Lya simulations:**
  - We could in principle run an $16{,}384^3$ hydro simulation!
  - Nyx is one of a few codes which can efficiently utilize the full system



The problem (OLCF machines)

System	Jaguar (2009)	Titan (2012)	Summit (2017)	Frontier (2021)
Peak	2.3 PF	27 PF	200 PF	2,000 PF
# nodes	18,688	18,688	4,608	9,408
Node	1 AMD CPU	1 AMD Opteron CPU 1 NVIDIA Kepler GPU	2 IBM POWER9™ CPUs 6 NVIDIA Volta GPUs	1 AMD Trento CPU 4 AMD MI250X GPUs
Memory	0.3 PB DDR2	0.6 PB DDR3 + 0.1 PB GDDR	2.4 PB DDR4 + 0.4 HBM + 7.4 PB On-node storage	4.6 PB DDR4 + 4.6 PB HBM2e + 36 PB On-node storage with 66 TB/s Read 62 TB/s Write
On-node interconnect	NA	PCI Gen2 No coherence across the node	NVIDIA NVLINK Coherent memory across the node	AMD Infinity Fabric Coherent memory across the node
System Interconnect	Cray SeaStar 2.0 GB/s	Cray Gemini network 6.4 GB/s	Mellanox Dual-port EDR IB 25 GB/s	Four-port Slingshot network 100 GB/s
Topology	3D Torus	3D Torus	Non-blocking Fat Tree	Dragonfly
Storage	15 PB, 0.2 TB/s Lustre	32 PB, 1 TB/s, Lustre	250 PB, 2.5 TB/s, IBM Spectrum Scale with GPFS	695 PB HDD+11 PB Flash Performance Tier, 9.4 TB/s and 10 PB Metadata Flash. Lustre
Power	7 MW	9 MW	13 MW	29 MW

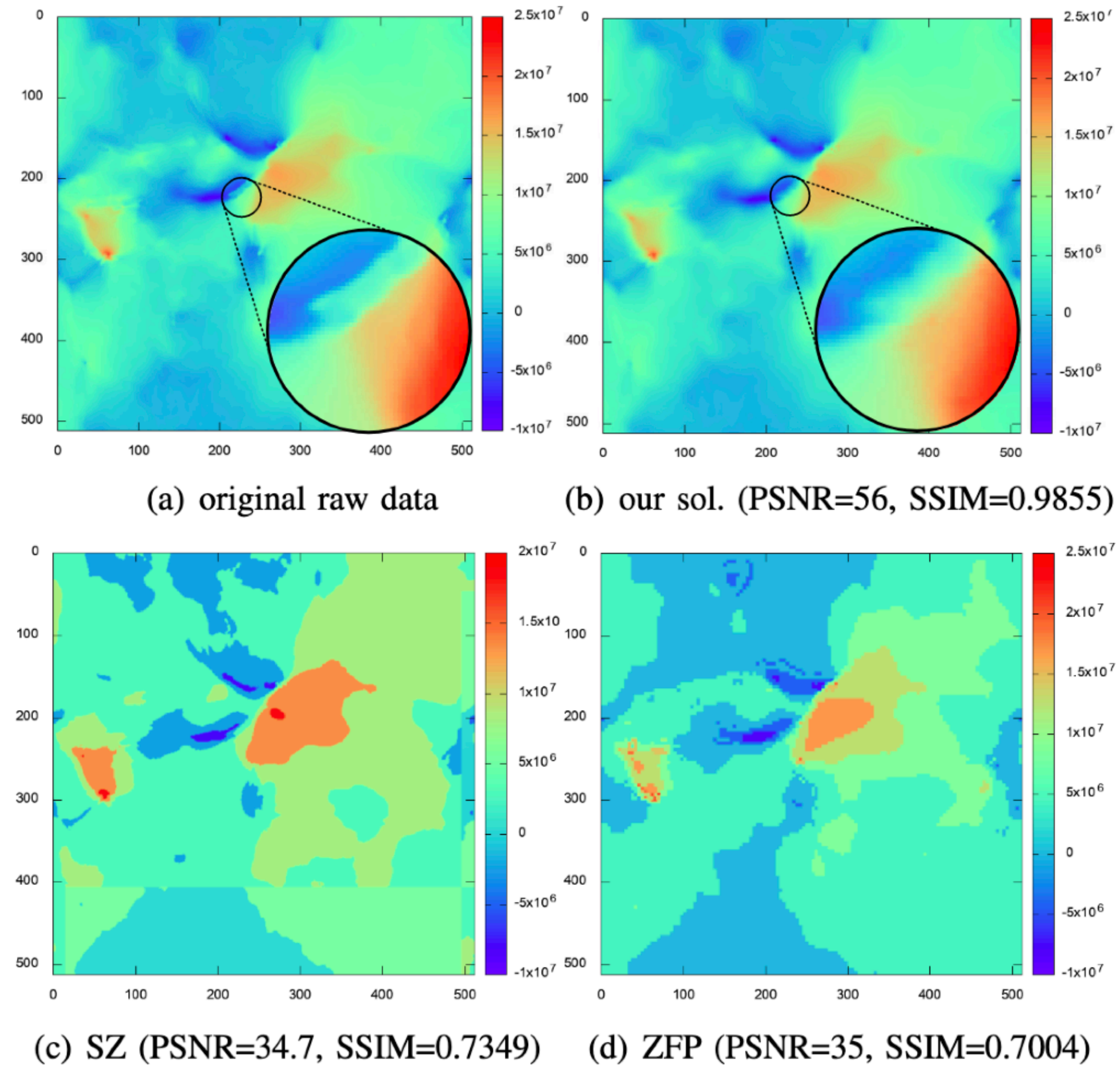
Two ways to tackle the data problem

- Produce summary statistics needed for science results in situ or in transit as opposed to simulation development being decoupled from simulation production and decoupled from science analysis

- Lossy compression of data (lossless too, but that tends to have marginal storage saving)

Adaptive error-bound lossy compression

- Strict user-set point-wise error control is usually desired
- SZ: fit/predict the successive data points with the best fit selection of curve fitting models. The data that can be predicted precisely will be replaced by the code of the corresponding curve-fitting model (Di & Cappello 2016)
- Split the whole dataset into multiple non-overlapped blocks, and select the best-fit prediction method based on their data features
- Effective for the lossy compression with relatively large error bounds
- Liang et al. (2018) demonstrated how to select adaptively and efficiently the best-fit prediction method based on the data features across blocks during the compression



(a) original raw data
(b) our sol. (PSNR=56, SSIM=0.9855)
(c) SZ (PSNR=34.7, SSIM=0.7349)
(d) ZFP (PSNR=35, SSIM=0.7004)

Nyx velocity field, compression ratio 156:1
(Liang et al. 2018)

Henson (master of puppets)

Efficient in situ/in transit:

— Henson is a system for loading multiple codes into the same address space and coordinating their execution, including iterative execution.

Details:

— Position-independent executables: individual codes are loaded as shared objects into the same memory space using dlopen/dlsym facilities, allowing multiple codes to execute on the same node with direct access to each others' memory

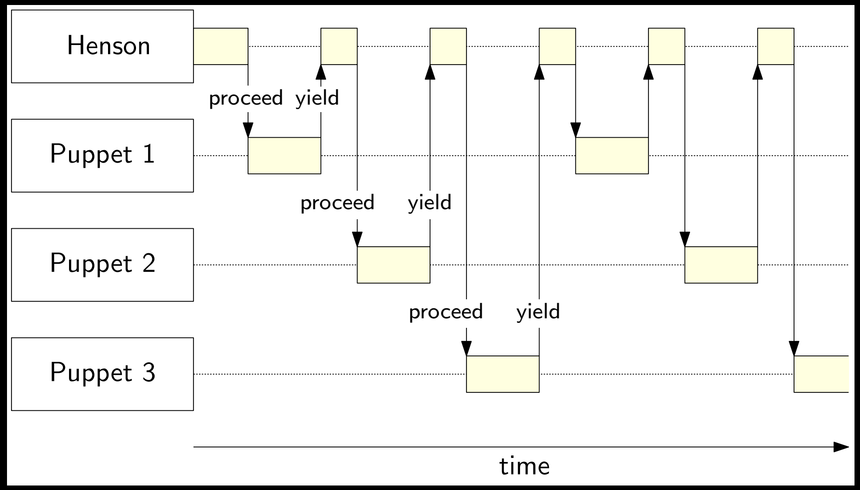
— Coroutines: routines which preserve their context after the exit. You can re-enter where you left last time. Each separate piece of code becomes a coroutine; main coroutine is Henson and it manages others

— High-level ChaiScript is used to manipulate executables

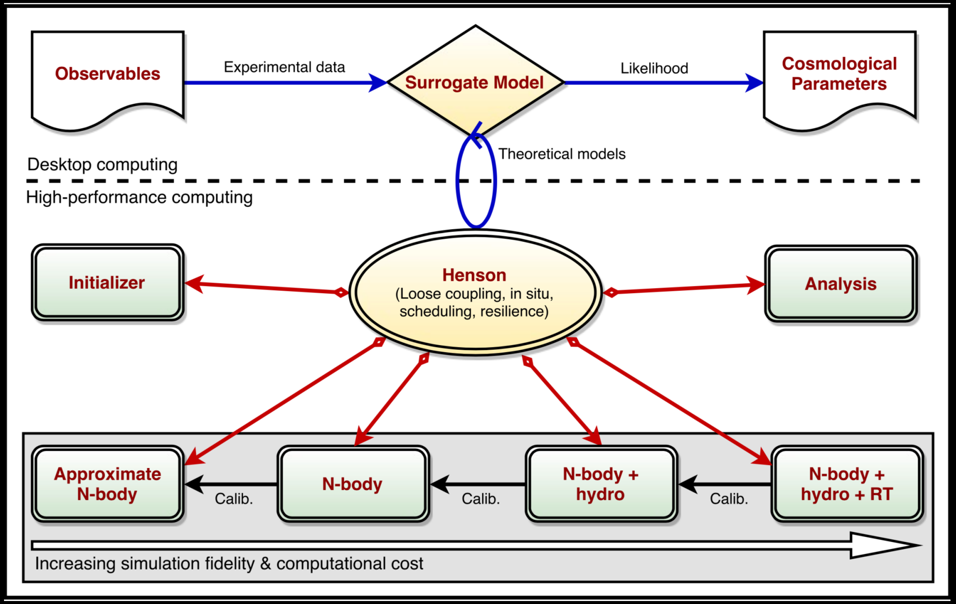
— MPI is managed (and hidden via PMPI) to allow logical execution groups to easily communicate with each other

— Dynamic scheduler enables advanced workflows (e.g., coordinating execution of multiple models driven by a surrogate).

Morozov & Lukić 2016

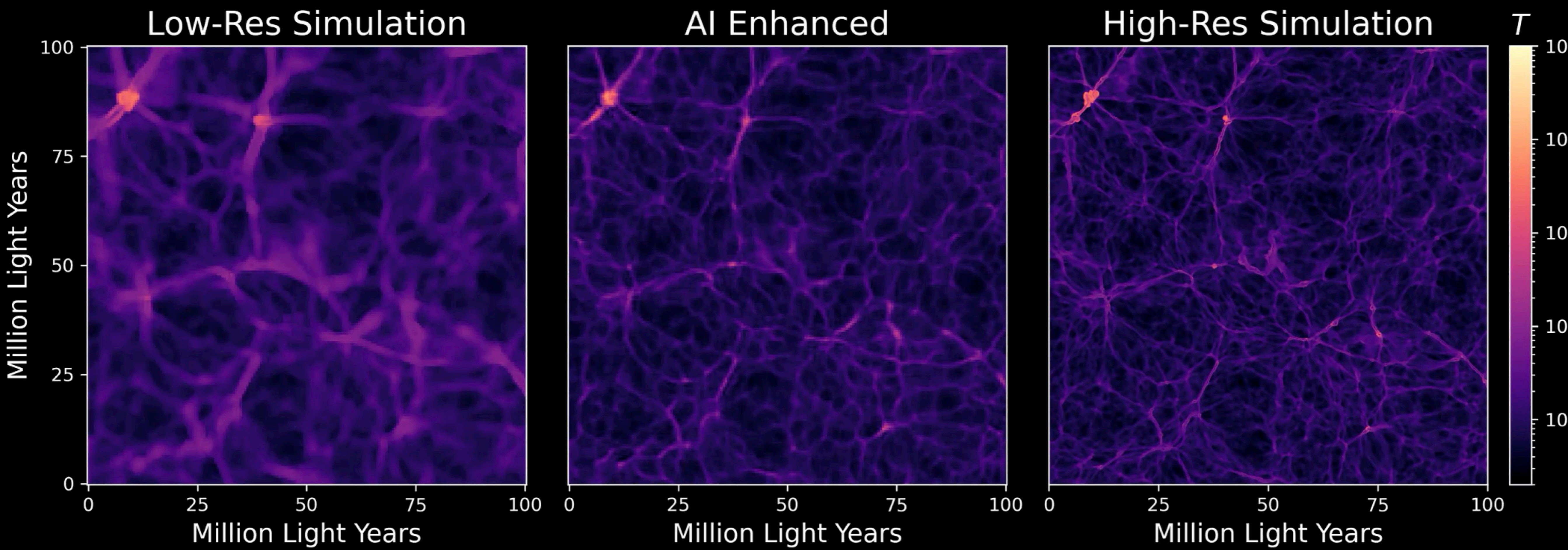


Henson coordinates control flow via cooperative multi-tasking between codes



Dynamic scheduler allows exploration of simulation parameters, driven by the inference process.

Combining simulations and generative AI



Cooper Jacobus, Peter Harrington, and Zarija Lukic:
"Reconstructing Lyα Fields from Low-resolution Hydrodynamical Simulations with Deep Learning", **The Astrophysical Journal** 958, 21 (2023)

Nyx

BERKELEY LAB
Lawrence Berkeley National Laboratory

Nyx regression tests

- The plotfiles generated in each night's test are compared with the benchmark plotfiles

Benchmark Updated	Comparison Failed	Compilation Failed	Crashed	Passed	Performance Drop

date	AMR-density	DR_restart	DoubleRarefaction	DrivenTurbulence	DrivenTurbulence-OMP	LyA	LyA-OMP-no-sundials-tiling	LyA-OMP-tiling	LyA-adiabatic	LyA_Neutrinos	MiniSB	MiniSB-ppm	MiniSB-ppm-ref	MiniSB-ref	Part-2line	Part-2line-nbody	Part-2line-nomesh	Part-2line_restart	Part-2line_restart-nbody	Part-2line_restart-nomesh	Part-load	Part-mass.nosub	Part-mass.nosub-nbody	Part-mass.nosub-nomesh	Part-mass.sub	Part-mass.sub-nbody	Part-mass.sub-nomesh	Sedov	Sedov-ppm	Sod	Sod-ppm	StrongShockTube	StrongShockTube-ppm
plots																																	
2024-06-24 *	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)
2024-06-21 *	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:]	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)
2024-06-20 *	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)
2024-06-19 *	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)
2024-06-18 *	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:]	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)
2024-06-17 *	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)
2024-06-14 *	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)
2024-06-13 *	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)
2024-06-12 *	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)
2024-06-11 *	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)	:)

Summary

- Nyx is a multi-physics cosmological code, targeting intergalactic medium studies. Efficiency and scalability on largest HPC machines are code's essential characteristics since day one.
- We have enabled Nyx on all HPC architectures under the Exascale Computing Project. We are achieving great efficiency and scalability on newest GPU architectures
- RT implementation is currently in the testing phase; this will enable more accurate modeling of the cosmic reionization
- Going forward, storage is going to be a #1 problem for HPC cosmological simulations. One should consider the full scientific workflow, not just the efficiency of simulation code
- Do implement regression tests!