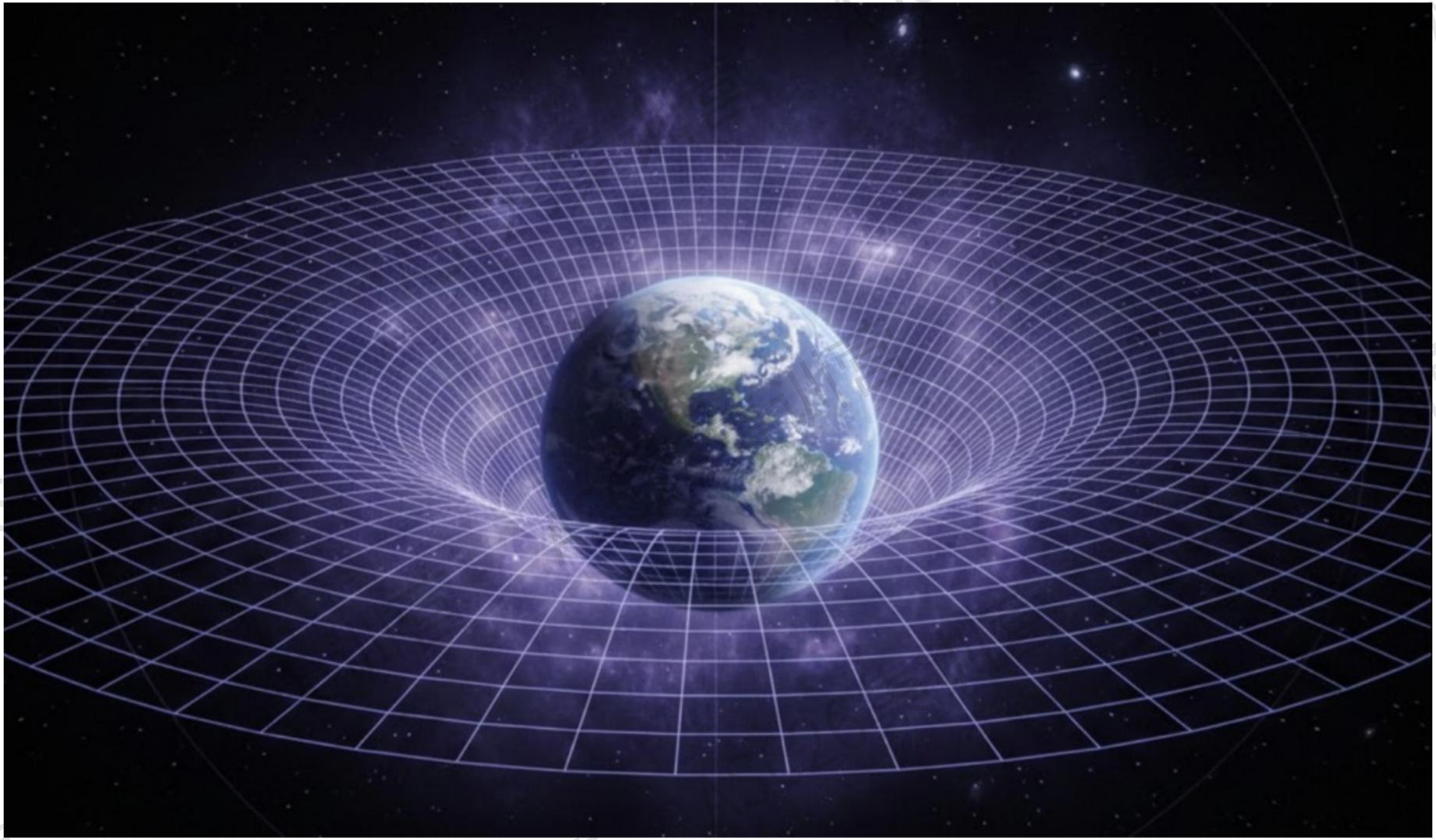
Solving Einstein's Equations on a Computer: A Brief Introduction to Numerical Relativity and the AMSS-NCKU Optimization Project

January 28, 2026

本 ppt 针对计算机专业同学快速理解使用，可能存在不严谨之处，建议可参考数值相对论相关专业教科书

One hour $\sim$ seven years



An observation

Imagine I throw a stone from here with some initial speed; it will follow a trajectory and land somewhere.

A comparison

Neglecting air resistance, if I throw a larger stone, a feather, an apple, ... with the same initial speed, they follow the same trajectory and land at the same place.

Matter tells spacetime how to curve;
spacetime tells matter how to move

$$G_{\mu\nu} + \Lambda g_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}.$$

$$G_{\mu\nu} + \Lambda g_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}.$$

Left-hand side: geometry (curvature of spacetime)

- $G_{\mu\nu}$: the Einstein tensor, built from the metric $g_{\mu\nu}$ and its derivatives, encoding spacetime curvature.

Right-hand side: matter (energy and momentum)

- $T_{\mu\nu}$: the stress-energy tensor, describing energy density, momentum density/flux, and stresses (pressure, shear) of matter/fields.
- G: Newton's gravitational constant; c: the speed of light in vacuum.

Write $T_{\mu\nu}$ as a matrix

$$T_{\mu\nu} = \begin{pmatrix} \rho & S_x & S_y & S_z \\ S_x & S_{xx} & S_{xy} & S_{xz} \\ S_y & S_{yx} & S_{yy} & S_{yz} \\ S_z & S_{zx} & S_{zy} & S_{zz} \end{pmatrix}, \qquad T_{\mu\nu} = T_{\nu\mu}.$$

What are these S's?

- $\rho = T_{00}$: energy density (in an appropriate local rest frame).
- $S_i = T_{0i}$: momentum density / energy flux ($i = x, y, z$).
- $S_{ij} = T_{ij}$: stress-tensor components (pressure and shear).

Vacuum equation

In a vacuum region (no matter or non-gravitational fields), one typically sets

$$T_{\mu\nu} = 0, \qquad \Rightarrow \qquad G_{\mu\nu} = 0.$$

$$G_{\mu\nu} + \Lambda g_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}.$$

Einstein tensor

$$G_{\mu\nu} = R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R, \qquad R = g^{\alpha\beta} R_{\alpha\beta}.$$

$R_{\mu\nu}$: Ricci tensor

Obtained by contracting the Riemann tensor:

$$R_{\mu\nu} = R^{\alpha}{}_{\mu\alpha\nu}.$$

$g_{\mu\nu}$: metric tensor

Used to compute the line element (distance/time interval) and to raise/lower indices:

$$\mathrm{d}s^2 = g_{\mu\nu} \mathrm{d}x^{\mu} \mathrm{d}x^{\nu}, \qquad g^{\mu\alpha} g_{\alpha\nu} = \delta^{\mu}{}_{\nu}.$$

Einstein summation convention

Rule (implicit summation)

- If an index appears **twice** in the same term (typically once up and once down), it is implicitly summed over.
- Indices that appear only once are **free indices**; free indices must match on both sides of an equation.

Example

$$A_i B^i \equiv \sum_{i=1}^{3} A_i B^i, \qquad g_{\mu\nu}\, \mathrm{d}x^\mu \mathrm{d}x^\nu \equiv \sum_{\mu=0}^{3}\sum_{\nu=0}^{3} g_{\mu\nu}\, \mathrm{d}x^\mu \mathrm{d}x^\nu .$$

Start with Euclidean space: what is $\mathrm{d}s^2$?

In 3D Euclidean space, with Cartesian coordinates (x, y, z), the length of an infinitesimal displacement $(\mathrm{d}x, \mathrm{d}y, \mathrm{d}z)$ satisfies

$$\mathrm{d}s^2 = \mathrm{d}x^2 + \mathrm{d}y^2 + \mathrm{d}z^2.$$

This means $\mathrm{d}s^2$ measures the squared distance between nearby points, with equal weighting in all directions.

Generalize to spacetime: the metric as a rule for measuring distance/time

Extend coordinates to spacetime x^μ (with $x^0 = ct$); in general one defines the interval by

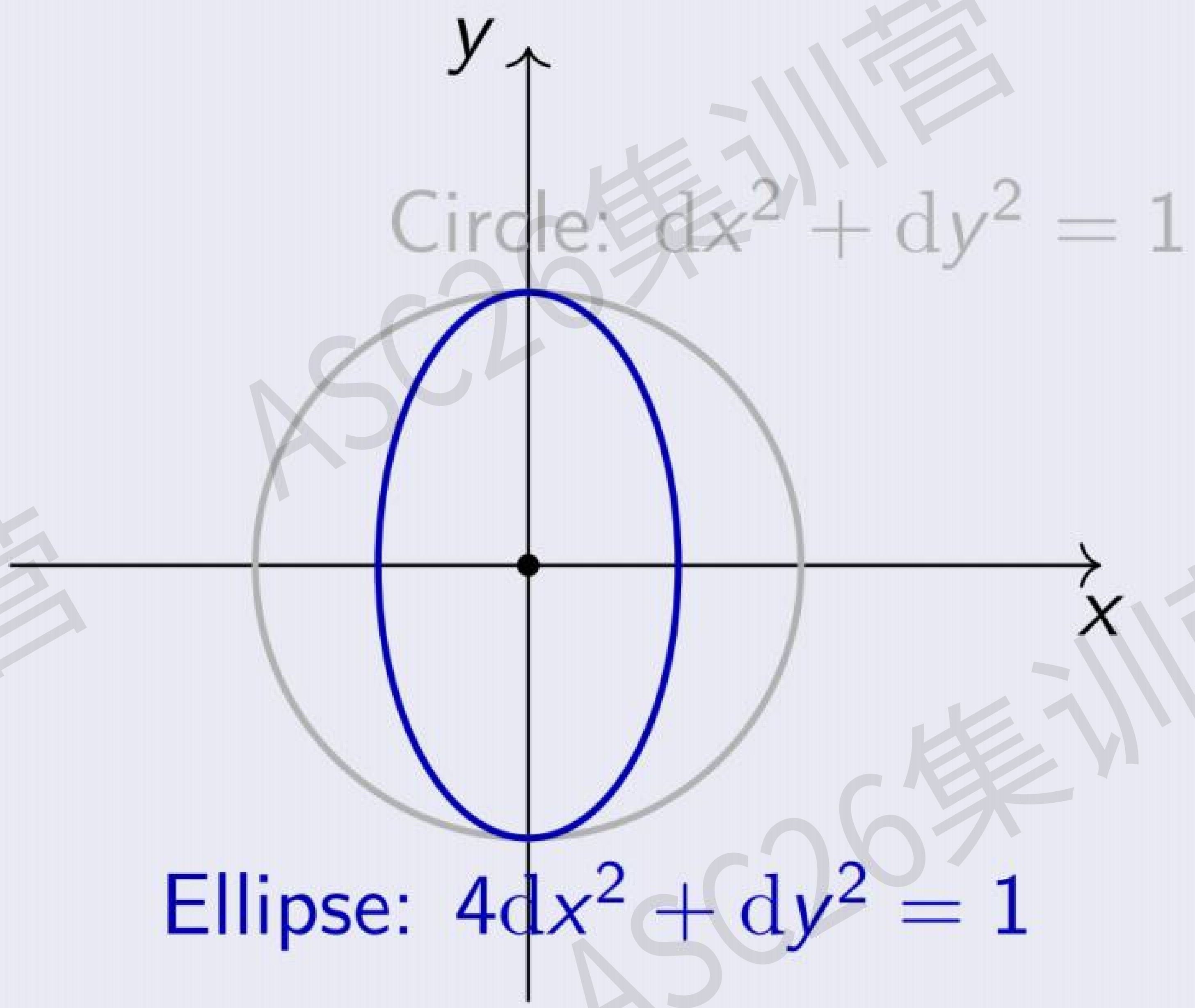
$$\mathrm{d}s^2 = g_{\mu\nu}(x)\,\mathrm{d}x^\mu \mathrm{d}x^\nu$$

In flat spacetime $g_{\mu\nu} = \eta_{\mu\nu} = \mathrm{diag}(-1, 1, 1, 1)$, so

$$\mathrm{d}s^2 = -c^2\mathrm{d}t^2 + \mathrm{d}x^2 + \mathrm{d}y^2 + \mathrm{d}z^2.$$

2D intuition: a metric defines lengths

- Gray circle: Euclidean metric $\mathrm{d}s^2 = \mathrm{d}x^2 + \mathrm{d}y^2$. The set of displacements $(\mathrm{d}x, \mathrm{d}y)$ with $\mathrm{d}s^2 = 1$ forms a circle.
- Blue ellipse: with $\mathrm{d}s^2 = 4\mathrm{d}x^2 + \mathrm{d}y^2$, the x-direction is weighted more, and the $\mathrm{d}s^2 = 1$ set becomes an ellipse.



Contravariant/covariant: upper vs lower indices

- **Contravariant** components: upper indices, e.g. a vector v^{μ}.
- **Covariant** components: lower indices, e.g. v_{μ}.
- Two representations of the same geometric object; the metric relates them.

Another key role of the metric: raising/lowering indices

$$v_{\mu} = g_{\mu\nu}\, v^{\nu}, \qquad v^{\mu} = g^{\mu\nu} v_{\nu}, \qquad g^{\mu\alpha} g_{\alpha\nu} = \delta^{\mu}{}_{\nu}.$$

A simple example (flat spacetime)

If $g_{\mu\nu} = \eta_{\mu\nu} = \mathrm{diag}(-1, 1, 1, 1)$, then

$$\text{expanding } v^{\nu} = g^{\nu\mu} v_{\mu} \text{ (Einstein summation):} \qquad v^{0} = g^{00} v_0 + g^{01} v_1 + g^{02} v_2 + g^{03} v_3.$$

$$\text{with } g^{\nu\mu} = \eta^{\nu\mu} = \mathrm{diag}(-1, 1, 1, 1) \text{ this reduces to} \qquad v^{0} = g^{00} v_0 = -v_0, \quad v^{i} = g^{ii} v_i = v_i \ (i = 1, 2, 3).$$

What is a covector?

- A covector ω acts on a vector v and outputs a number (a linear functional):

$$\omega(v) = \omega_\mu v^\mu.$$

- Linearity means: $\omega(av + bw) = a\,\omega(v) + b\,\omega(w)$.
- Under coordinate changes, ω_μ transforms oppositely to v^μ, so $\omega_\mu v^\mu$ is an invariant scalar.

A common example

The gradient (differential) $\mathrm{d}f$ of a function $f(x)$ is a covector: for any displacement v^μ,

$$\mathrm{d}f(v) = v^\mu \partial_\mu f,$$

which is the rate of change along the direction v.

$$G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}.$$

$$\boxed{R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R = 0}$$

$$R = g^{\mu\nu} R_{\mu\nu}, \qquad R_{\mu\nu} = R^{\alpha}{}_{\mu\alpha\nu}.$$

Riemann curvature tensor $R^{\rho}{}_{\sigma\mu\nu}$

Constructed from the metric (equivalently from the connection $\Gamma^{\rho}{}_{\mu\nu}$), it characterizes curvature:

$$R^{\rho}{}_{\sigma\mu\nu} = \partial_{\mu}\Gamma^{\rho}{}_{\nu\sigma} - \partial_{\nu}\Gamma^{\rho}{}_{\mu\sigma} + \Gamma^{\rho}{}_{\mu\lambda}\Gamma^{\lambda}{}_{\nu\sigma} - \Gamma^{\rho}{}_{\nu\lambda}\Gamma^{\lambda}{}_{\mu\sigma}.$$

Levi–Civita connection (most commonly used in GR)

Uniquely determined by the metric (torsion-free + metric-compatible):

$$\Gamma^{\rho}{}_{\mu\nu} = \frac{1}{2}g^{\rho\sigma}\left(\partial_{\mu}g_{\nu\sigma} + \partial_{\nu}g_{\mu\sigma} - \partial_{\sigma}g_{\mu\nu}\right).$$

Now you know the law that governs gravity.
In principle, you can compute any gravitational phenomenon.

For example: black holes

An observation

Imagine I throw a stone from here with some initial speed; it will follow a trajectory and land somewhere.

A comparison

Neglecting air resistance, if I throw a larger stone, a feather, an apple, ... with the same initial speed, they follow the same trajectory and land at the same place.

Increase the initial speed

If I throw the same object faster, it will land farther away; if it is fast enough, it may even **escape Earth's gravity**.

Light behaves the same way

If we shine a flashlight beam and nothing blocks it, the light keeps traveling—i.e. it can **escape** Earth.

If gravity is strong enough

If an object's gravity is so strong that even light cannot escape, then it is a **black hole**.

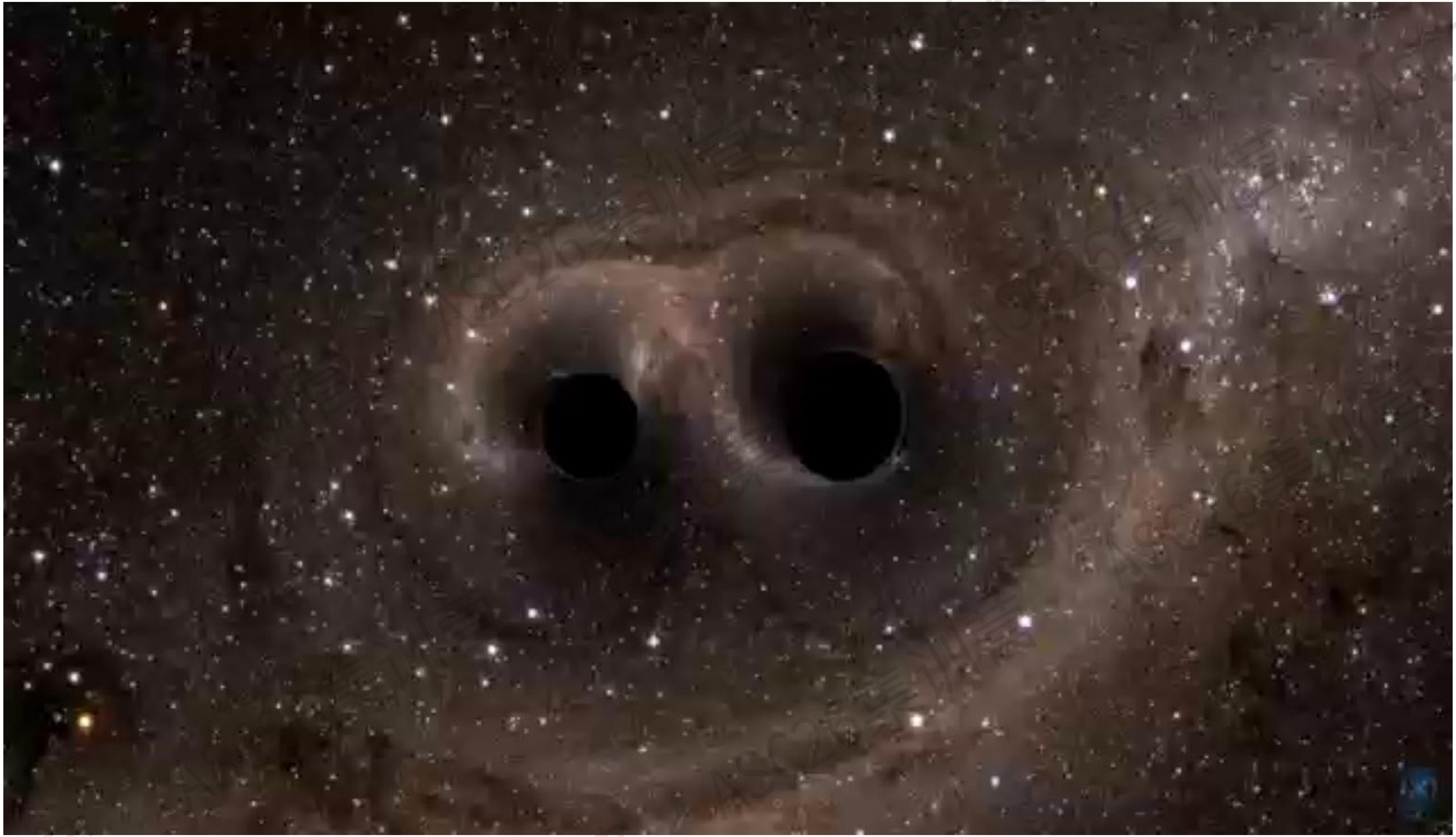
What we call gravity is, in essence, the **curvature of spacetime**, governed by the equation we just wrote down.

GW150914: the first direct observation of a binary black-hole merger

- On Sep 14, 2015, the two LIGO detectors (Hanford and Livingston) recorded an extremely short signal almost simultaneously.
- The signal came from the inspiral and merger of two **stellar-mass black holes**: masses of order tens of solar masses, at a distance of order a billion light-years.
- During the merger, energy equivalent to **a few solar masses** was radiated away as gravitational waves (in a very short time).

GW150914 parameters (rough numbers)

Component masses	$m_1 \approx 36\,M_\odot, \quad m_2 \approx 29\,M_\odot$
Final BH mass	$M_f \approx 62\,M_\odot$
Radiated energy	$\Delta M \approx 3\,M_\odot$ (energy ΔMc^2)
Distance	$D_L \approx 410\,\mathrm{Mpc} \approx 1.3$ billion light-years



If playback fails in your PDF viewer: open GW150914.mp4.

How do we "discover" a binary black-hole merger?

1. **Measure tiny length changes with laser interferometers**: as a GW passes, the two orthogonal arms stretch/squeeze in opposite ways, changing the interference fringes (strain $h \sim 10^{-21}$).
2. **Coincidence across sites**: the same event appears with nearly the same waveform in two widely separated detectors, with a millisecond-scale arrival-time difference (set by the propagation direction).
3. **Matched filtering with theoretical waveforms**: correlate the data against a large bank of relativistic "template waveforms" to extract the signal and estimate parameters.
4. **Statistical significance**: estimate the false-alarm rate from background noise and confirm it is extremely unlikely to be a noise fluctuation.

In one sentence

A binary black-hole merger is not discovered by "seeing light", but by **hearing gravitational waves**.

What are gravitational waves?

What they are

Gravitational waves are tiny **ripples in spacetime geometry**: when a gravitational field changes rapidly, the disturbance propagates outward as a wave.

How they are produced

The most typical sources are **asymmetric accelerated motion of massive bodies**, e.g. inspiral and merger of binaries (especially BBH and BNS). As energy is carried away, the orbit shrinks and the system eventually merges, producing the strongest GW signal.

How they propagate and what they do

GWs propagate at the **speed of light** and pass through matter with negligible absorption or scattering. Their effect is an extremely tiny **relative stretching/squeezing of freely falling test masses**: when one direction is stretched, the perpendicular direction is compressed (and vice versa), which is exactly what an interferometer measures.

Gravitational waves in the linearized theory (flat background)

- Small perturbation around flat spacetime: $g_{\mu\nu} = \eta_{\mu\nu} + h_{\mu\nu}$, with $|h_{\mu\nu}| \ll 1$.
- Trace-reversed perturbation: $\bar{h}_{\mu\nu} = h_{\mu\nu} - \frac{1}{2}\eta_{\mu\nu} h$, where $h = \eta^{\alpha\beta} h_{\alpha\beta}$.
- Gauge condition: $\partial^\mu \bar{h}_{\mu\nu} = 0$.

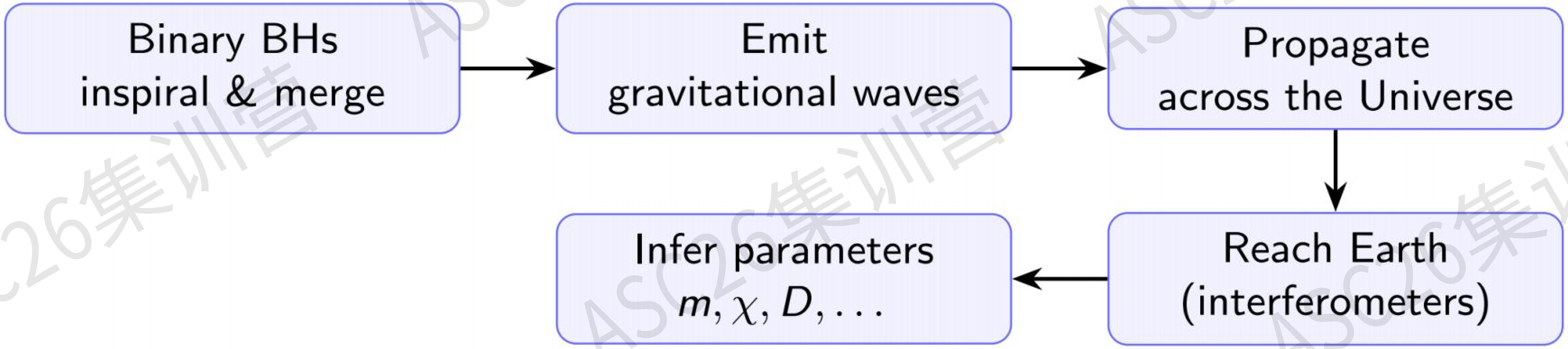
Under the linearized approximation and the gauge condition above, Einstein's equations become

$$\Box \bar{h}_{\mu\nu} = -\frac{16\pi G}{c^4} T_{\mu\nu}, \qquad \Box = -\frac{1}{c^2}\frac{\partial^2}{\partial t^2} + \nabla^2.$$

In vacuum ($T_{\mu\nu} = 0$) we get the wave equation:

$$\Box \bar{h}_{\mu\nu} = 0.$$

Everything seems clear...



But

This requires solving Einstein's equations:

$$\boxed{R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R = 0}$$

But this equation is hard to solve analytically

- **Strongly nonlinear**: the metric appears everywhere and couples to itself; superposition does not hold.
- **A coupled PDE system**: spacetime dynamics must satisfy evolution equations *and* constraint equations.
- **Coordinate freedom (gauge)**: the same geometry can be represented by different coordinates; choosing a good gauge is part of the problem.
- **Closed-form solutions exist only with high symmetry**: e.g. static spherical symmetry (Schwarzschild), stationary axisymmetry (Kerr), homogeneous and isotropic cosmology (FLRW).

Numerical simulation is extremely challenging

A historical comparison

- General relativity was proposed in 1915; but only around 2005 did stable BBH merger simulations break through (before 2005 it was long viewed as "unsolved").
- Kip Thorne (2000) said: *"GW detection will be earlier than numerical simulation of black hole collisions."*

Numerical simulation is extremely challenging

Before 2005: why was stable evolution hard (especially BBH mergers)?

- **Formulation and constraints**: the early ADM system easily excites unstable modes numerically; constraint violations grow rapidly and then destroy the evolution.
- **Gauge (lapse/shift) was not mature**: singularity-avoiding slicing often led to lapse collapse and slice stretching; poor shift choices produced severe coordinate pathologies.
- **Singularities and moving BHs**: excision requires robust horizon tracking and inner-boundary treatment; early puncture methods were often "fixed punctures" and had trouble evolving through merger.
- **Outer boundary conditions**: reflections and constraint injection from a finite boundary contaminate the strong-field region and trigger instabilities.
- **Multiscale dynamics and limited computing power**.

Typical symptom back then: very short evolutions

- From Hahn–Lindquist (1964) through the 1990s–2000s, 3D BBH simulations often lasted only tens of time units: Anninos et al. (1995), Brugmann (1999) ~ 35 t.u., Brandt et al. (2000) ~ 50 t.u., etc.

Numerical simulation is extremely challenging

Around 2005: what changed?

- **Better-posed evolution systems**: generalized harmonic (GH) and BSSN, together with constraint damping and improved discretizations, strongly suppressed unstable growth.
- **Gauge breakthroughs**: e.g. $1 + \log$ lapse and Γ-driver shift (moving puncture) mitigated coordinate stretching, enabling stable motion and mergers.
- **More practical singularity treatment**: excision in the GH framework became more robust; moving puncture avoids an explicit inner boundary.
- **Algorithms and compute caught up**: AMR, parallel frameworks, and larger compute budgets enabled long, high-resolution evolutions.

Numerical simulation is extremely challenging

Key milestones in numerical relativity (stability problem / BBH)

Early: short evolutions

- 1964: Hahn–Lindquist (one of the earliest BBH attempts)
- 1995: Anninos et al. (PRD 52, 2059)
- 1999: Brugmann (IJMP D 8, 85), $\sim$ 35 t.u.
- 2000: Brandt et al. (PRL 85, 5496), $\sim$ 50 t.u.
- 2001: Baker et al. (PRL 87, 121103), $\sim$ 100 t.u.
- 2004: Brugmann et al. (PRL 92, 211101), $\sim$ 150 t.u.

Breakthrough: stable mergers begin

- 2005: Pretorius (PRL 95, 121101), stably!!
- 2006: Campanelli et al. (PRL 96, 111101); Baker et al. (PRL 96, 111102)
- 2007: Penn State (CQG 24, S33); AEI (PRL 99, 041102)
- 2007–2008: Jena/Brugmann (PRD 76, 104015; PRD 77, 024027)
- 2008: Tokyo (PRD 78, 064054)
- 2008: Our group (PRD 78, 124011)

Numerical simulation is extremely challenging

Milestone results

- Pretorius (2005): GH + constraint damping + excision + AMR; first stable BBH evolution through inspiral–merger–ringdown.
- 2005–2006: "moving puncture" BSSN (Campanelli et al.; Baker et al.) made BBH merger simulations a reproducible standard tool.

Just 10 years later, in 2015, we observed gravitational waves from a binary black-hole merger.

Numerical Relativity: From Setup to Observation

Overall goal: from "physical setup" to "observable predictions"

- Solve Einstein's equations on a computer to obtain a spacetime evolution and ultimately produce observables (e.g. GW waveforms).

Physical setup system (BBH/BNS/NS–BH), matter model, boundary/symmetry assumptions

↓

Initial data (elliptic constraints) solve constraint equations $\Rightarrow$ consistent initial data

↓

Time evolution (hyperbolic system) choose formulation/gauge; evolve stably to obtain a discrete spacetime

↓

Observables and comparison extract ψ_4, h, etc. $\Rightarrow$ waveforms/parameter estimation; compare with detector data

Problem setup: realistic vs solvable

From physics to solvable: reality vs tractable

- The problem must be realistic enough (BBH, NS–BH, ...) while also numerically tractable (boundaries, matter models, approximations/assumptions, ...).
- More realism ⇒ more complex equations and larger scale separations ⇒ higher computational cost and more sources of error.

Common trade-offs (examples)

- Use a simpler matter model/EOS, or ignore magnetic fields, neutrinos, radiative feedback, etc. as a first step.
- Use more idealized initial conditions/symmetry assumptions to gain controllability and resolution, then add more physics gradually.

In practice, numerical relativity balances “realistic” and “tractable”.

Constraints and evolution: elliptic + hyperbolic

Equation structure: elliptic + hyperbolic

Elliptic part (constraints)

- Construct constraint-satisfying initial data.
- Typically a globally coupled boundary-value problem; handle infinity/outer boundaries and multi-BH topology.

Hyperbolic part (evolution)

- Evolve in time from the initial data to obtain a discrete spacetime evolution.
- Wave-equation-like: emphasize strong hyperbolicity, stability, and constraint control.

Key difficulty 1: Formalism / Gauge

- General relativity has coordinate freedom: the same physical spacetime can be represented by different coordinates.
- Different formulations/gauges affect hyperbolicity, constraint growth, and numerical stability; a bad choice often "crashes immediately".

Waveform extraction and engineering

Key difficulty 2: Gauge & finite-radius extraction

- Ideally, GWs are defined at future null infinity $\mathcal{I}^+$, but numerically we can only extract at finite radius.
- Extrapolation or CCE (Cauchy–Characteristic Extraction) is used to reduce waveform contamination from gauge, outer boundaries, and finite-radius effects.

Engineering challenges: numerical methods and coding

- Stability (CFL), discretization and dissipation, AMR, boundary conditions, parallelism, and performance tuning decide whether it runs at all and how accurate it is.
- Ultimately: discrete spacetime solution $\rightarrow$ extract observables $(h, \psi_4, \dots)$ $\rightarrow$ draw physical conclusions and compare with observations.

Formalism problem

Start from Einstein's equations

$$G_{ab} := R_{ab} - \frac{1}{2} R g_{ab} = 8\pi T_{ab}.$$

But first rewrite it into a form suitable for numerical evolution

- $\begin{cases} \text{coordinate expansion: ADM, BSSN, GHG, } \ldots \\ \text{tetrad expansion: Ashtekar, Friedrich–Nagy, } \ldots \end{cases}$
- $\begin{cases} 3+1 \text{ form: ADM, BSSN, NOR, } \ldots \\ \text{four dimensional form: GHG, Friedrich–Nagy, } \ldots \end{cases}$
- **constraint correction:** ADM $\rightarrow$ BSSN $\rightarrow$ Z4c $\rightarrow \cdots$

Variables: coordinate vs tetrad

Coordinate expansion (metric variables)

- Evolve $g_{\mu\nu}$ directly (with lapse/shift, harmonic gauge, etc.).
- Examples: ADM, BSSN, Z4c, GHG, ...
- Pros: relatively straightforward to implement; mature ecosystem (AMR and waveform-extraction toolchains).

Tetrad expansion (tetrads / spin connection)

- Describe geometry using tetrads such as $e^a{}_\mu$ (or equivalent variables).
- Examples: Ashtekar variables, the Friedrich–Nagy system, ...
- Highlight: some constructions yield symmetric hyperbolic systems and more controlled boundary treatments; but with more variables and more complex gauge freedom.

Choosing a formulation: hyperbolicity and constraint control

$3+1$ form vs 4D form

$3+1$ (Cauchy)

- Decompose spacetime into spatial slices evolving in time: constraints + evolution equations.
- Typical: ADM, BSSN, NOR, Z4c, CCZ4, ...

4D (harmonic / generalized harmonic)

- Write the principal part as a 4D wave-equation-like system, emphasizing well-posedness and boundary conditions.
- Typical: GHG, Friedrich–Nagy, ...

Why do we need constraint correction?

- In the continuum, constraints should remain satisfied if they are satisfied initially; discretization errors can trigger and amplify constraint violations.
- By rewriting variables/equations and adding constraint damping/propagation (e.g. BSSN, Z4c/CCZ4, GH + damping), "constraint growth" becomes a controllable mode.

3+1 ADM decomposition (Arnowitt–Deser–Misner)

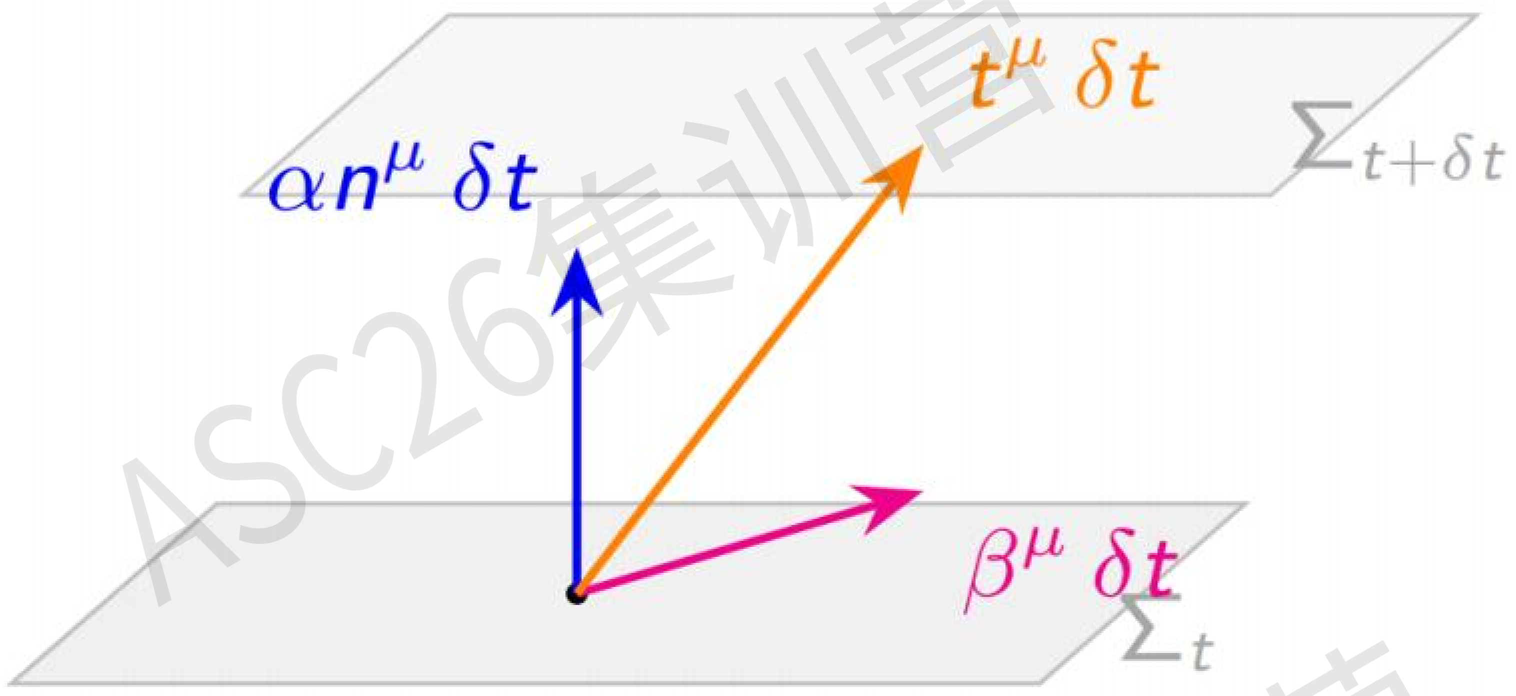
Foliation and metric split

- Choose a time function t and foliate spacetime: ${}^4M \simeq \mathbb{R} \times \Sigma_t$.
- The metric can be written as

$$ds^2 = -\alpha^2 dt^2 + \gamma_{ij}\,(dx^i + \beta^i dt)(dx^j + \beta^j dt),$$

  where α is the lapse, β^i the shift, and γ_{ij} the spatial metric on Σ_t.
- Introduce the extrinsic curvature $K_{ij}(t, \mathbf{x})$ to describe how slices embed in spacetime; it measures the rate of change along the normal:

$$K_{ij} := -\frac{1}{2}\mathcal{L}_n\gamma_{ij} = -\frac{1}{2\alpha}\,(\partial_t - \mathcal{L}_\beta)\gamma_{ij}.$$



$$t^\mu = \alpha n^\mu + \beta^\mu$$

Lie derivative (how to compute it)

For the shift β^i, the Lie derivative of the spatial metric (coordinate components) is:

ADM: 12 evolution equations and 4 constraints (vacuum)

Constraints

$$\mathcal{H} := R + K^2 - K_{ij}K^{ij} \approx 0, \qquad \mathcal{M}^i := \nabla_j(K^{ij} - \gamma^{ij}K) \approx 0.$$

Evolution

$$(\partial_t - \mathcal{L}_\beta)\gamma_{ij} = -2\alpha K_{ij},$$
$$(\partial_t - \mathcal{L}_\beta)K_{ij} = -\nabla_i\nabla_j\alpha + \alpha\big(R_{ij} + KK_{ij} - 2K_{im}K^m{}_j\big).$$

$4 + 6 + 6 = 16$ variables to be solved $\qquad 12 + 4 = 16$ equations to be solved

Coupled elliptic–hyperbolic equations, although complicated but seems consistently solvable

Constraints must be controlled: numerical errors excite constraint violations, and stability depends strongly on the formulation/gauge.

Constrained system

The 16 equations are not independent

- In the continuum theory, if constraints are satisfied, the evolution "preserves" them (the constraint-propagation system closes; related to the Bianchi identities).
- Therefore, one can equivalently choose 12 of the 16 equations as evolution equations; the remaining 4 are constraints (for initial data, boundaries, and error monitoring).

Why do we typically choose 12 evolution equations?

- For simplicity and efficiency: time evolution uses a purely hyperbolic evolution system.
- Constraints are mainly used to construct initial data and to monitor/control errors.

Constrained system

12 independent equations $<$ 16 variables: why is it still consistent?

- This is not an inconsistency: general relativity has 4 coordinate degrees of freedom (gauge freedom).
- In other words, 4 variables should be free (set by gauge conditions) for the system to be self-consistent.

Which variables are gauge?

- In the $3+1$ split, the gauge variables are precisely the lapse α and shift β^i (geometrically: how slices advance).
- In principle α, β^i can be specified freely; in practice we use gauge conditions (e.g. $1+\log$, Γ-driver, harmonic gauge) to close the system and improve stability.

Constrained system

Key numerical difficulty

- Continuum level: 12 variables $\leftrightarrow$ 12 evolution equations; if constraints are zero initially they should remain zero.
- After discretization, constraints are not guaranteed to be preserved (truncation error, outer-boundary reflections, discretization inconsistencies can generate violations).
- It then looks like 12 variables must satisfy $12 + 4$ conditions (over-determined), so we must explicitly control constraints.

Can we build a constraint-preserving scheme?

- Goal: keep constraint violations from growing (or damp them), and avoid injecting constraint errors at the outer boundary.
- Common strategies: better formulations (BSSN, Z4c/CCZ4, GH + damping), constraint damping/adjustments, constraint-preserving boundary conditions, and sometimes constraint projection.

Boundary treatment

- Real physical system: no boundary (not possible for numerics)
- Compactify — energy piles up
- Artificial boundary (how to set boundary conditions)
- Radiative boundary condition
  [Shibata and Nakamura PRD '95]

Fortunately, it is STABLE!

but it introduces extra error!

Outer boundary conditions: why are they tricky?

Numerics live on a finite domain

- The physical system is open: GWs propagate to infinity, and ideal observables are defined at infinity ($\mathcal{I}^+$).
- A Cauchy evolution must be truncated at finite radius $r = R$, introducing an artificial outer boundary; boundary treatment directly affects stability and accuracy.

An ideal outer boundary should do all of the following

- **Absorb outgoing waves**: minimize reflections (reflections contaminate the strong-field region and can trigger instabilities).
- **Preserve constraints**: do not inject constraint-violating modes (otherwise constraint errors grow).
- **Be gauge-compatible**: gauge waves/coordinate effects also reach the boundary; avoid pathologies caused by "coordinate reflections".

Radiative boundary condition (Sommerfeld type)

Idea: treat variables as approximately spherical outgoing waves

$$u(t, r) \approx u_0 + \frac{f(t - r/c)}{r} \qquad \Rightarrow \qquad (\partial_t + c\,\partial_r)\big(r(u - u_0)\big) \approx 0$$

Pros and limitations

- Pros: easy to implement; often provides a numerically stable outer boundary (especially in the far zone / weak-field regime).
- Limitations: not perfectly absorbing—non-spherical/low-frequency components reflect; near-zone strong-field and gauge coupling introduce systematic errors.
- Practical strategy: place the boundary far away (with AMR) + use constraint-preserving boundary conditions (CPBC) or CCE/extrapolation to reduce waveform error.

Treatment of physical singularities

- Kinds of quantities diverge (∞) when we approach physical singularity. The region near singularity must be ruled out from numerical computation.
- Fill black holes with special data (puncture)
  Fill what, how to fill?
- Cut directly (excision)
  How to treat the inner boundary?

Puncture method

Core idea: isolate the singular part analytically

- In initial data construction, treat a BH as an additional asymptotically flat end (wormhole/puncture). One often writes

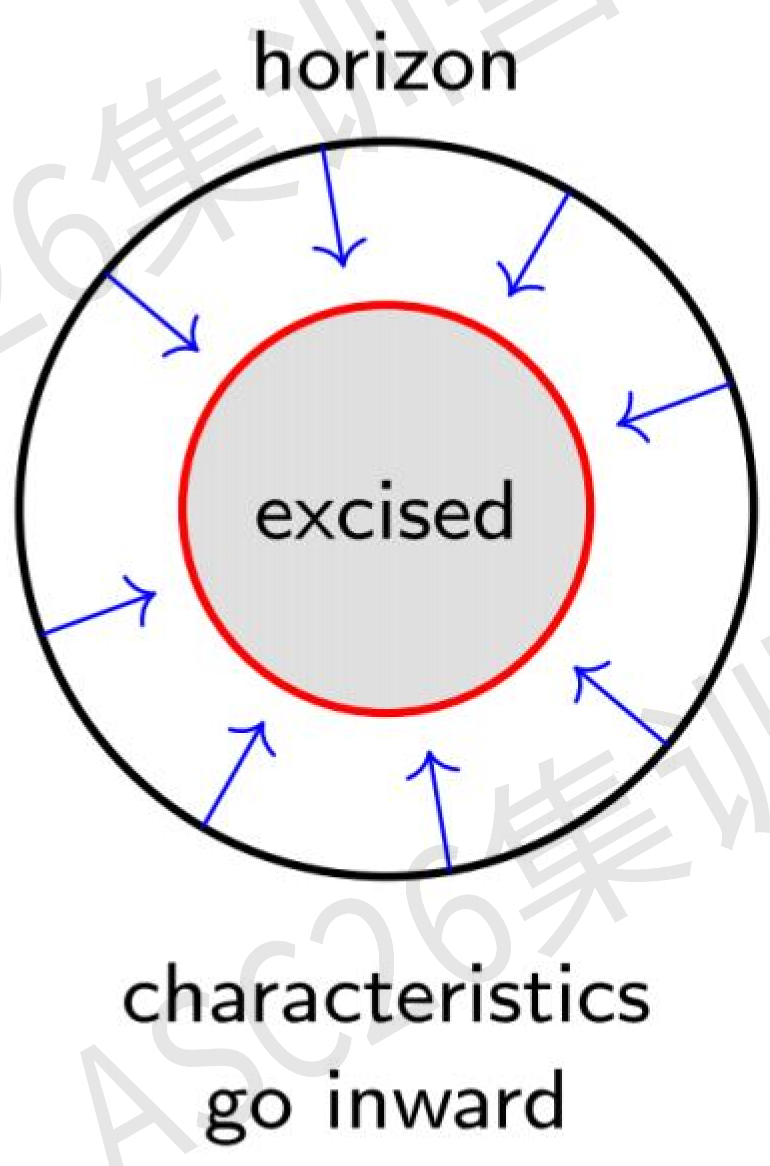
$$\psi = \psi_{\text{sing}} + u, \qquad \psi_{\text{sing}} = 1 + \sum_a \frac{m_a}{2r_a},$$

  where u is the regular part solved numerically.
- With suitable gauge (e.g. $1 + \log$ lapse $+$ Γ-driver shift), the slice avoids the singularity: the lapse collapses near the puncture and the geometry approaches a trumpet.
- Pros: no inner boundary; relatively simple; well-suited to BBH (moving puncture, widely used since 2006).

Excision method

Core idea: excise the singular region from the grid

- Choose an inner boundary $r = r_{\text{exc}}$ inside the (apparent) horizon and remove interior grid points (excision).
- If r_{exc} lies inside the apparent horizon (AH), no physical information can propagate out of the BH: the inner boundary is pure outflow, so in principle no incoming boundary conditions are needed.
- Key challenges: track moving BHs/horizons; keep the excision surface inside the AH; maintain numerical stability of gauge/constraint modes at the inner boundary.
- Typical use: common in generalized harmonic (GH) formulations (e.g. Pretorius 2005).



Evolution PDE system for Einstein's equations (BSSN + gauge)

$$\partial_t \phi = -\frac{1}{6}\left(\alpha K - \partial_i \beta^i\right) + \beta^i \partial_i \phi,$$

$$\partial_t K = -D^i D_i \alpha + \alpha\left(\tilde{A}_{ij}\tilde{A}^{ij} + \frac{1}{3}K^2\right) + \beta^i \partial_i K,$$

$$\partial_t \tilde{\gamma}_{ij} = -2\alpha \tilde{A}_{ij} + 2\tilde{\gamma}_{k(i}\partial_{j)}\beta^k - \frac{2}{3}\tilde{\gamma}_{ij}\partial_k \beta^k + \beta^k \partial_k \tilde{\gamma}_{ij},$$

$$\partial_t \tilde{A}_{ij} = -e^{-4\phi}\left(D_i D_j \alpha - \alpha R_{ij}\right)^{TF} + \alpha\left(K\tilde{A}_{ij} - 2\tilde{A}_{ik}\tilde{A}^k{}_j\right) + 2\tilde{A}_{k(i}\partial_{j)}\beta^k - \frac{2}{3}\tilde{A}_{ij}\partial_k \beta^k + \beta^k \partial_k \tilde{A}_{ij},$$

$$\partial_t \tilde{\Gamma}^i = -2\tilde{A}^{ij}\partial_j \alpha + 2\alpha\left(\tilde{\Gamma}^i{}_{jk}\tilde{A}^{jk} - \frac{2}{3}\tilde{\gamma}^{ij}\partial_j K + 6\tilde{A}^{ij}\partial_j \phi\right) + \tilde{\gamma}^{jk}\partial_{jk}\beta^i + \frac{1}{3}\tilde{\gamma}^{ij}\partial_{jk}\beta^k + \beta^j \partial_j \tilde{\Gamma}^i - \tilde{\Gamma}^j \partial_j \beta^i + \frac{2}{3}\tilde{\Gamma}^i \partial_j \beta^j$$

$$\partial_t \alpha = -2\alpha K + \beta^i \partial_i \alpha,$$

$$\partial_t \beta^i = \frac{3}{4}B^i + \beta^j \partial_j \beta^i,$$

$$\partial_t B^i = \partial_t \tilde{\Gamma}^i - \beta^j \partial_j \tilde{\Gamma}^i - \eta B^i + \beta^j \partial_j B^i.$$

Why so expensive? (tens of thousands of FLOPs per grid point per step)

Where does the cost come from?

- Many tensor variables must be updated at every grid point (e.g. $\tilde{\gamma}_{ij}, \tilde{A}_{ij}, \tilde{\Gamma}^i, \phi, K$ and α, β^i, B^i).
- The RHS contains many spatial derivatives: first order (∂_i), second order ($D_i D_j$), and the Christoffel symbols/derivatives needed to build R_{ij}.
- Discretization typically uses finite differences or spectral methods: each derivative implies a stencil/transform, plus AMR, boundary conditions, and parallel communication.

Engineering takeaway

- To evolve "stably + accurately + for a long time", you must work on both formulation/gauge and numerical algorithms/implementation (dissipation, AMR, parallelism).

Parallelized mesh refinement

- **Several scales involved**
  - ✓ black hole (1) $\Rightarrow \Delta x \sim 0.01$
  - ✓ separation of black holes (10)
  - ✓ wave length of gravitational wave (50)
  - ✓ asymptotic region (1000–10000)
- **Computationally expensive on every grid point**
  (less grid points, much more levels)

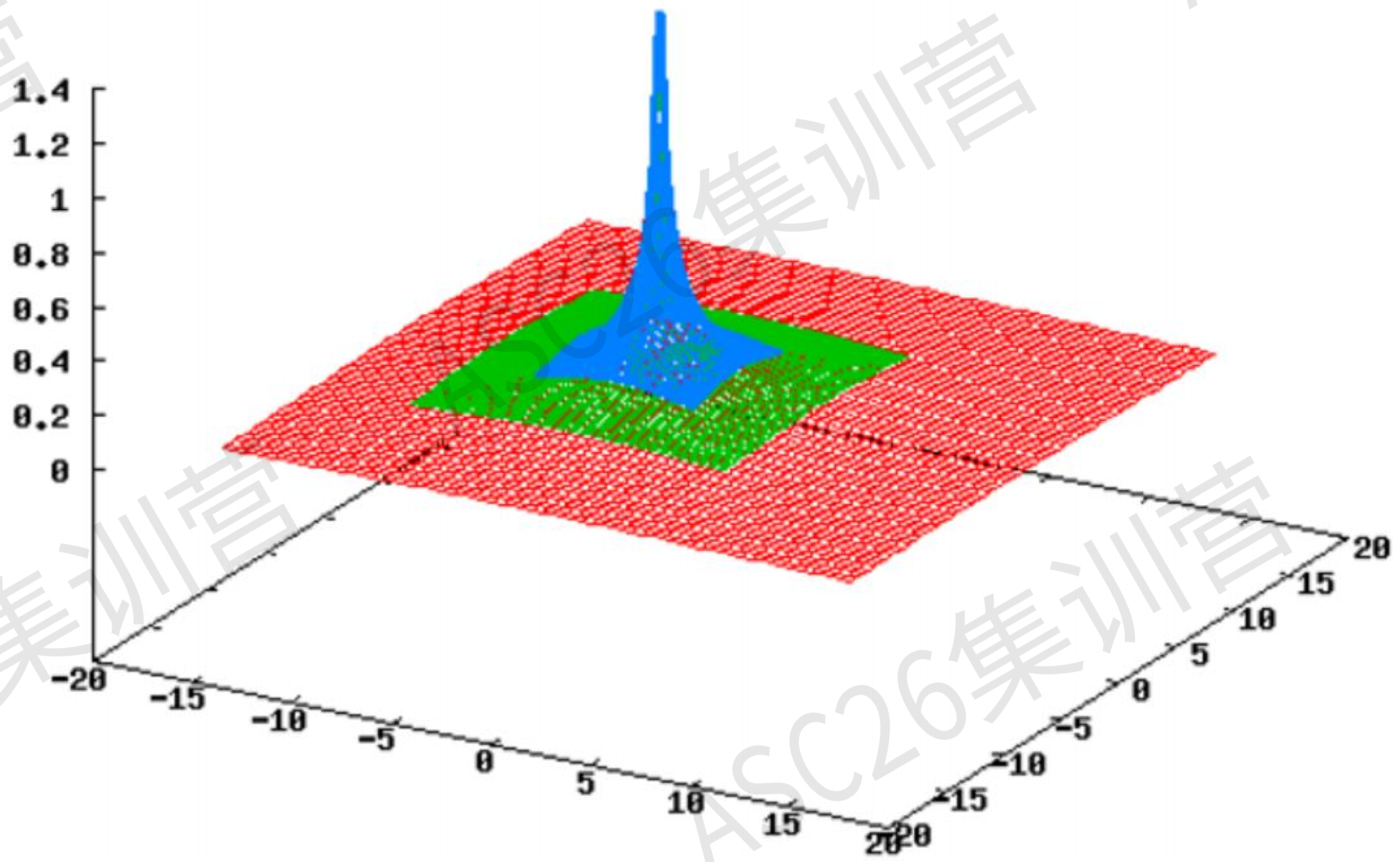
Why do we need AMR + parallelism?

- You cannot use the smallest Δx everywhere: the number of grid points would explode.
- AMR concentrates high resolution in the strong-field region (near the BHs), uses coarser grids in the wave zone / asymptotic region, and distributes grid blocks across many cores/nodes via domain decomposition.

Mesh refinement (example)

Example only, usually 12–16 levels

Take advantage of spacetime symmetry



Is the run stable? (online monitoring)

Most important: constraint monitoring (4 constraints)

- At each step, compute the Hamiltonian constraint H and the 3 momentum constraints M_i from the numerical solution $(\gamma_{ij}, K_{ij}, \alpha, \beta^i, \ldots)$.
- Monitor norms (e.g. L_2 and L_∞): a stable run should not blow up exponentially and should decrease systematically with increasing resolution.
- A stricter check: run multiple resolutions and verify $\|H\| \sim \mathcal{O}(\Delta x^p)$ (order p) convergence of constraint violations.

What else to monitor? (numerical + physical diagnostics)

- **Convergence**: convergence order of key quantities (waveform phase/amplitude, horizon mass/spin, orbital phase) across resolutions.
- **Conservation/consistency**: time evolution of ADM mass/angular momentum should match radiated energy/angular momentum (from waveform extraction), with no unphysical drift.
- **Gauge and boundaries**: lapse collapse, coordinate stretching, and outer-boundary reflections contaminate constraints/waveforms; monitor incoming constraint flux and reflected signals near the boundary.

Four constraints: Hamiltonian + Momentum

Constraint equations in the $3+1$ decomposition (vacuum RHS $=0$)

$$H \equiv R + K^2 - K_{ij}K^{ij} - 16\pi\rho \approx 0,$$
$$M_i \equiv D_j\left(K^j{}_i - \delta^j{}_i K\right) - 8\pi S_i \approx 0, \qquad i = 1,2,3.$$

Common numerical monitors (on a discrete grid)

$$\|H\|_2 \approx \left(\sum_{\text{grid}} H^2\,\Delta V\right)^{1/2}, \qquad \|M\|_2 \approx \left(\sum_{\text{grid}} \gamma^{ij} M_i M_j\,\Delta V\right)^{1/2}, \qquad \|H\|_\infty = \max_{\text{grid}} |H|.$$

- Often normalize (e.g. divide by a representative curvature/derivative scale) and compute level-weighted statistics across AMR levels.
- With excision, measure outside the excised region; with punctures, pay special attention to constraints near the strong-field region and in the wave zone.

Gravitational Waves

Fallback: open GW_GW150914.mp4.

Source tree and language stack

- Focus here: `amss-ncku-python/AMSS_NCKU_source`
- Typical split by language
  - C/C++: framework, class wrappers, Patch/AMR management, MPI orchestration, diagnostics workflow
  - Fortran 90: high-order finite differences, RK time stepping, prolong/restrict, dissipation and other numerical operators
  - CUDA: GPU acceleration for selected BSSN hot kernels
- Goal: stably and efficiently evolve BSSN/Z4c (and extensions) on AMR + MPI (optional GPU), and output physical diagnostics

Layered architecture: physics to parallelism

Evolution equations and physical models
`bssn_class.*`, `Z4c_class.*`, `bssnEM_class.*`, `bssnEScalar_class.*`

↓

Spatial discretization and stabilization
high-order centered FD: `diff_new*.f90` KO dissipation: `kodiss.f90`

↓

Time integration
explicit RK4: `rungekutta4_rout.f90`

↓

Grid/AMR and inter-level operators
`patch_system.*`, `patch.*`, `patch_interp.*`, `prolongrestrict*.f90`

↓

Parallelism and acceleration
MPI: `Parallel.*` GPU: `bssn_gpu.cu`, `bssn_gpu_class.*`

↓

Diagnostics and horizons
`BH_diagnostics.*`, `find_horizons.*`, `surface_integral.*`

Directory structure and functional layers (by module)

Evolution equations and physical models

- bssn_class.C/.h
- Z4c_class.C/.h
- bssnEM_class.*, bssnEScalar_class.*

Grid and AMR

- Patch management: patch_system.*, patch.*
- Interpolation: patch_interp.*
- Prolong/restrict: prolongrestrict*.f90

Time integration and finite differences

- RK4: rungekutta4_rout.f90
- Derivatives: diff_new*.f90
- Dissipation: kodiss.f90

Initial data, constraints, and diagnostics

- Initial data: TwoPunctures.*, initial_*.f90
- Constraints: bssn_constraint.f90, adm_constraint.f90
- Diagnostics/horizons: BH_diagnostics.*, find_horizons.*

Computational principles (high-level view)

- Numerical relativity in a 3+1 formulation (BSSN/Z4c): evolve metric and curvature variables on a discrete grid
- Spatial derivatives: high-order centered finite differences (accuracy) + Kreiss–Oliger dissipation (stability)
- Time stepping: explicit Runge–Kutta (typically RK4)
- AMR: block-structured patches; inter-level consistency via prolongation and restriction operators
- Parallelism: MPI domain decomposition; selected kernels have GPU implementations for speedups

Key algorithm flow (simplified)

Initialization

1. Read parameters and physical setup; construct initial data (e.g., Two-Puncture)
2. Build the block-structured AMR grid and register boundary/interpolation/inter-level rules

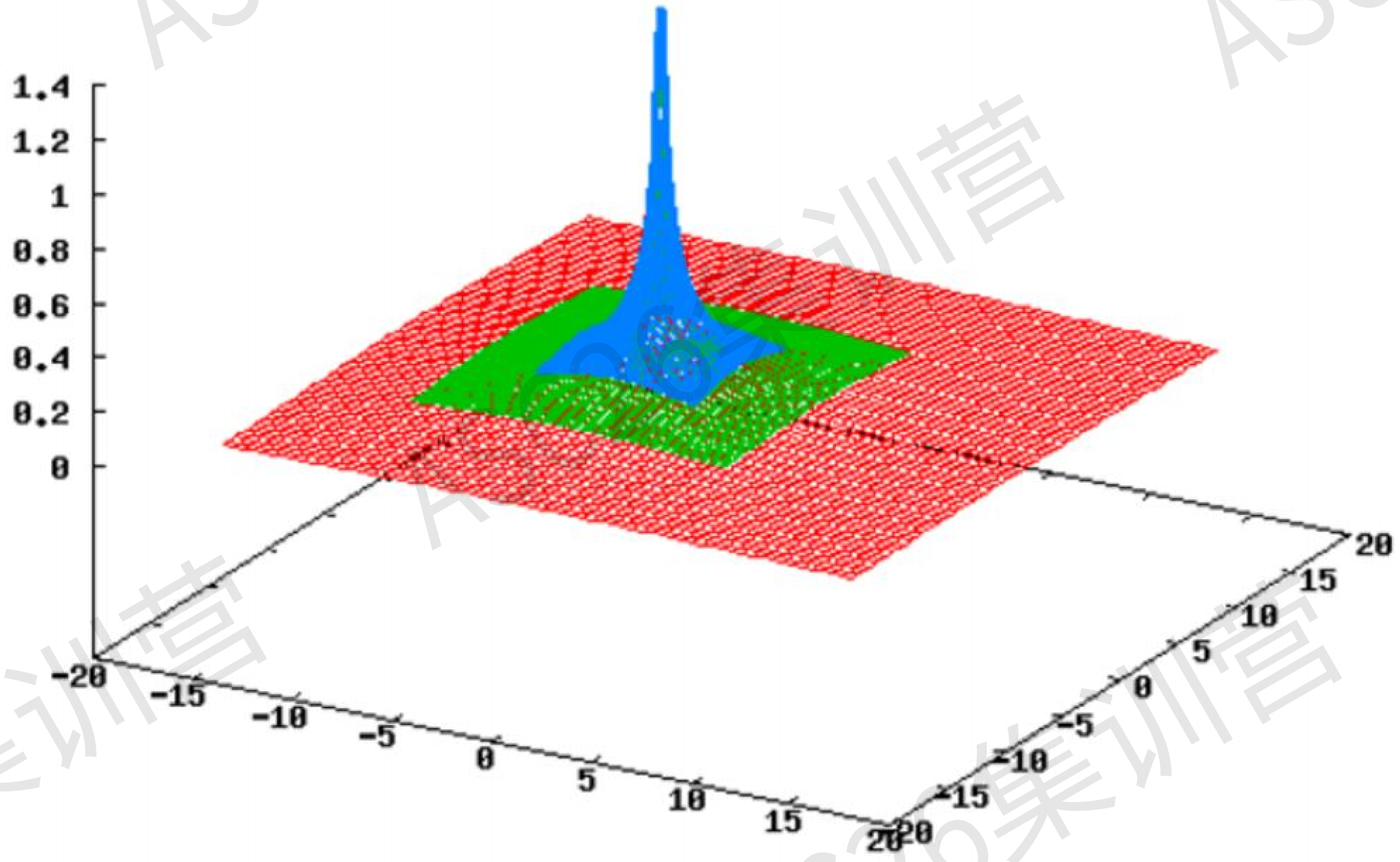
Per time step (main loop)

1. Compute spatial derivatives: high-order differences + numerical dissipation
2. Advance evolution equations with RK4 (multiple stages)
3. AMR operations: prolong/restrict; synchronize boundaries (ghost zones) across MPI ranks
4. Output/update diagnostics: constraints, horizons, waveforms, etc.

AMR: Patch system and inter-level operators

- Patch responsibilities: manage blocks per refinement level, neighbor relations, ghost zones, and boundary conditions
- Inter-level consistency: coarse$\rightarrow$fine prolongation and fine$\rightarrow$coarse restriction
- Implementation entry points: `patch_system.*`, `patch_interp.*`, `prolongrestrict*.f90`

AMR: schematic



Parallelism and acceleration: MPI + (optional) GPU

- MPI: decompose the domain by patches/grids; boundary exchange and synchronization typically appear in each RK stage
- GPU: offload selected BSSN hot kernels to CUDA (e.g., `bssn_gpu.cu`), integrated via `bssn_gpu_class.*`
- Typical bottlenecks: ghost-zone exchange (communication/synchronization) and FD/dissipation loops (memory bandwidth/compute)

Optimization levers (aligned with current implementation)

- Cache and vectorization: improve data layout (SoA or hybrid) and enable explicit vectorization for hot loops in `diff_new*.f90` and `bssn_*`
- Parallel scaling: add OpenMP for hot loops; use non-blocking MPI to hide halo-exchange latency
- GPU coverage: offload remaining constraint/dissipation operators; reduce CPU↔GPU transfers
- Time-step control: add an optional adaptive Courant factor (constraint/error driven) without changing default behavior
- Diagnostics scheduling: parallelize/async horizon and constraint measurements to reduce global synchronization

BH_Trajectory_XY.pdf: black-hole trajectories in the XY plane

- Data source: bssn_BH.dat from the simulation output directory
- What is plotted: each black hole's 2D trajectory $(X_i(t), Y_i(t))$
- Axes: $X[M]$ vs. $Y[M]$ (units normalized by the total mass M)
- Convention: different black holes are shown with different colored curves (e.g., BH1/BH2)
- How to read it: transitions from a slow inspiral to the post-merger settling/ringdown region

BH_Trajectory_21_XY.pdf: BH2 displacement relative to BH1

- Data source: bssn_BH.dat
- What is plotted: the relative displacement $\Delta \boldsymbol{r}_{21}(t)$ projected onto the XY plane
- Coordinates: $(\Delta X, \Delta Y) = (X_2 - X_1,\ Y_2 - Y_1)$
- Interpretation: removes overall drift and highlights the shrinking orbital separation and pre-merger relative motion
- Practical note: if BH1/BH2 are output at different times, interpolate to a common t-grid before differencing

ADM_Constraint_Grid_Level_0.pdf: constraint monitoring on the outer grid

- Data source: `bssn_constraint.dat`
- What is plotted: ADM constraint measures on grid level 0 (the outermost level) versus time $T[M]$
- Curves: H, P_x, P_y, P_z
- Physical meaning:
  - H: Hamiltonian constraint
  - P_x, P_y, P_z: momentum-constraint components along x, y, z
- Axes: time $T[M]$ vs. ADM constraint value
- How to read it: smaller is better; growth/spikes often indicate accumulated numerical error, boundary issues, or AMR synchronization problems