SENPR/NRPy: A Next-Generation, Dynamical Reference Metric Numerical Relativity Code

Zachariah B. Etienne
Ian Ruchlin

in collaboration with

Thomas W. Baumgarte
SENR/NRPy: Code Overview

- **NRPy**: Python+sympy code generation for NR
  - Similar to Kranc, but with no Mathematica!
    - Equations *at your fingertips*, even on HPC systems!
    - Input: Einstein notation + simple syntax Python code
    - Output: Efficient, compiler-vectorizable C code (AVX)

- **SENR**: Simple, Efficient Numerical Relativity code
  - Contains NRPy wrappers, diagnostics, MoL, BCs for solving BSSN equations in arbitrary coord systems
    - Log-Spherical Polar, Cylindrical, Cartesian, Bispherical-like

[https://tinyurl.com/senrcode](https://tinyurl.com/senrcode)
SENR/NRPy: Motivation

- SENR/NRPy = Simple, efficient, open (BSD-licensed, Python-based) infrastructure for numerical relativity codes and code generation
- Goal: When solving problem, choose the best coordinate grid for the task!
  - Black hole, neutron star: Log-Spherical polar coords
  - Compact binary: Dynamical, Bispherical-like coords
- Better coordinate grids = Giant efficiency gain over AMR!
  - At least ~160x decrease in # of gridpoints → use desktop for BHB
  - Single grid patch = ~25x better scalability than AMR!

https://tinyurl.com/senrcode
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

AMR
Adaptive Mesh Refinement
(Most Popular Method in NR)
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

AMR
Adaptive Mesh Refinement
(Most Popular Method in NR)
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations
Near-Spherical Object

- Highest res needed in radial dirn, need ~1/3 points in angular directions
  Cost: \( N_r \times N_{\theta} \times N_{\phi} \sim \frac{1}{9} N_r^3 \)
- Cartesian grid: need \( dx=dy=dz=dr \).
  Cost: \( N_x \times N_y \times N_z \sim N_r^3 \)
- So far, spherical polar grid ~9x more efficient than Cartesian

Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

Near-Spherical Object

- Highest res needed in radial dirn, need \( \sim \frac{1}{3} \) points in angular directions
  
  Cost: \( N_r \cdot N_{\theta} \cdot N_{\phi} \sim \frac{1}{9} N_r^3 \)

- Cartesian grid: need \( dx = dy = dz = dr \)
  
  Cost: \( N_x \cdot N_y \cdot N_z \sim N_r^3 \)

- So far, spherical polar grid \( \sim 9 \times \) more efficient than Cartesian

What about \( dr \) along diagonal?

- Cube diagonal = \( \sqrt{3} \cdot \text{sidelength} \) → to get \( dr \) resolution in all directions, need to reduce \( dx, dy, dz \) by \( \sqrt{3} \)

- Since cost in memory \( \sim \frac{1}{dx^3} \), “fitting the round peg in a square hole” increases cost by another factor of \( (\sqrt{3})^3 \sim 5.2x \)!
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

**Near-Spherical Object**

- Highest res needed in radial dirn, need ~1/3 points in angular directions
  Cost: $Nr*Ntheta*Nphi \sim 1/9 Nr^3$
- Cartesian grid: need $dx=dy=dz=dr$
  Cost: $Nx*Ny*Nz \sim Nr^3$
- So far, spherical polar grid ~9x more efficient than Cartesian

**What about $dr$ along diagonal?**

- Cube diagonal = $\sqrt{3}$*sidelength → to get $dr$ resolution in all directions, need to reduce $dx,dy,dz$ by $\sqrt{3}$
- Since cost in memory ~1/$dx^3$, “fitting the round peg in a square hole” increases cost by another factor of $(\sqrt{3})^3 \sim 5.2x$!

**Inefficiencies so far:** 

- ~47x
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

AMR Box
Boundary is a Cube...

- ... but fields fall off radially!
- \( \rightarrow \) region outside orange circle is over-resolved by 2x
- Total volume of over-resolved region = \( 8-\frac{4}{3} \pi \approx 3.8 \) = about half the cube!
- Bispherical coordinate system: Gain another \( \sim 1.7 \times \)

\( \text{AMR Box side-length} = 2 \)
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

AMR Box
Boundary is a Cube...

- ... but fields fall off radially!
- \( \text{region outside orange circle is over-resolved by } 2x \)
- Total volume of over-resolved region = \( 8 - \frac{4}{3} \pi \approx 3.8 = \text{about half the cube!} \)
- Bispherical coordinate system: Gain another \(~1.7x\)

AMR Box side-length = 2

Inefficiencies so far: \(~80x\)
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

AMR

Adaptive Mesh Refinement
(Most Popular Method in NR)

- Information must be interpolated across refinement boundaries.
- Interpolation → grids must overlap
- Overlap regions (grey) can take up 50% of overall computational domain!

Numerical Relativity (NR) Simulations
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

AMR
Adaptive Mesh Refinement (Most Popular Method in NR)

- Information must be interpolated across refinement boundaries.
- Interpolation → grids must overlap
- Overlap regions (grey) can take up 50% of overall computational domain!
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

**AMR Box**
Boundary is a Cube...

- ... but fields fall off radially!
- \( \rightarrow \) region outside orange circle is over-resolved by 2x
- Total volume of over-resolved region = 8-4/3 \( \pi \) \( \approx \) 3.8 = about half the cube!
- Bispherical coordinate system: Gain another \( \approx 1.7x \)

**High-order finite difference with AMR**

- \( \rightarrow \) Enormous number of ghost zones at refinement boundaries!
- Ghost zones can take up 50% of overall computational domain!
- Bispherical coordinate system: Gain another \( \approx 2x \)
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

AMR Box Boundary is a Cube...
- ... but fields fall off radially!
- → region outside orange circle is over-resolved by 2x
- Total volume of over-resolved region = 8-4/3 pi ~ 3.8 = about half the cube!
- Bispherical coordinate system: Gain another ~1.7x

High-order finite difference with AMR
- → Enormous number of ghost zones at refinement boundaries!
- Ghost zones can take up 50% of overall computational domain!
- Bispherical coordinate system: Gain another ~2x

AMR Inefficiencies: ~160x
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

AMR Inefficiencies: ~160x
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

AMR Inefficiencies: ~160x (estimated)
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

AMR Inefficiencies: ~160x (estimated)

Inefficiency Measurement
- Single black hole
- Moderate spin: $a/M = 0.5$
- Set up AMR (Carpet) grid, measure $H$ constraint violation
- Adjust SENR grids:
  - $H_{SENR} < H_{AMR}$ at all points
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

Inefficiency Measurement

- Single black hole
- Moderate spin: $a/M = 0.5$
- Set up AMR (Carpet) grid, measure H constraint violation
- Adjust SENR grids:
  - $H_{SENR} < H_{AMR}$ at all points

AMR Inefficiencies: $\sim 160x$ (estimated)

AMR Grid:
- 10 GB

SENR's Log-Spherical Polar Grid:
- 40 MB
  - (un-optimized grid structure, another 4-10x drop possible)
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

**AMR Inefficiencies:**

- Single black hole
- Moderate spin: $a/M = 0.5$
- Set up AMR (Carpet) grid, measure $H$ constraint violation
- Adjust SENR grids:
  - $H_{\text{SENR}} < H_{\text{AMR}}$ at all points

**Inefficiency Measurement**

**AMR Inefficiencies:**

- ~160x (estimated)

**AMR Grid:**
- 10 GB

**SENR's Log-Spherical Polar Grid:**
- 40 MB
  - (un-optimized grid structure, another 4-10x drop possible)

**AMR Inefficiencies:**

- 250x (measured)
Enormous Inefficiencies Exist in Numerical Relativity (NR) Simulations

Inefficiency Measurement

- Single black hole
- Moderate spin:
  - $a/M = 0.5$
- Set up AMR (Carpet) grid, measure $H$ constraint violation
- Adjust SENR grids:
  - $H_{SENR} < H_{AMR}$ at all points

AMR Inefficiencies:
- ~160x (estimated)

AMR Grid:
- 10 GB

SENR's Log-Spherical Polar Grid:
- 40 MB
  - (un-optimized grid structure, another 4-10x drop possible)

Stable long-term BH evolutions!
But does it converge?
SENR Results: Exponential convergence of numerical errors!

Simulating black hole without excision: Numerical errors converge to zero \textit{exponentially} with increased polynomial approximation order!
SENR/NRPy: Summary

- **Open Source, Open Development** → **Greater Adoption**
  - [http://tinyurl.com/senrcode](http://tinyurl.com/senrcode)

- **Algorithmic Simplicity** → **More Science Faster**
  - Easier to debug & extend
  - Build on tried & true algorithms
    - BSSN in Spherical Polar Coords techniques pioneered by T. Baumgarte et al
      - SENR: Extend ideas to support arbitrary, *dynamical* coords

- **Memory Efficiency Is Key Focus**: **Unlock the Desktop**
  - Get public involved → ~10,000x more GW throughput!

- **Bottom line**: **Maximize science with minimal human & computational resources**