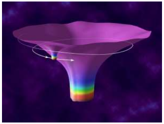
Self-Force for a Scalar Particle in Schwarzschild Spacetime: Time-Domain Calculations with Adaptive Mesh Refinement

Jonathan Thornburg

Department of Astronomy
Indiana University
Bloomington, Indiana, USA



Overall Strategy for Self-Force Calculations

Barack-Ori mode-sum regularization scheme

Overall Strategy for Self-Force Calculations

Barack-Ori mode-sum regularization scheme

Lorenz gauge:

- metric perturbation from point particle is relatively simple (locally isotropic)

Overall Strategy for Self-Force Calculations

Barack-Ori mode-sum regularization scheme

Lorenz gauge:

- metric perturbation from point particle is relatively simple (locally isotropic)

Decompose metric perturbation into spherical harmonics (ℓ, m):

- metric perturbations aren't separable
  $\Rightarrow$ compute modes via **time-domain numerical integration**

$$\Box\phi_{\ell m} + V_\ell(r)\phi_{\ell m} = S_{\ell m}(t)\,\delta\big(r(t) - r_{\text{particle}}(t)\big)$$

Overall Strategy for Self-Force Calculations

Barack-Ori mode-sum regularization scheme

Lorenz gauge:

- metric perturbation from point particle is relatively simple (locally isotropic)

Decompose metric perturbation into spherical harmonics (ℓ, m):

- metric perturbations aren't separable
  $\Rightarrow$ compute modes via **time-domain numerical integration**

$$\Box\phi_{\ell m} + V_\ell(r)\phi_{\ell m} = S_{\ell m}(t)\,\delta\big(r(t) - r_{\text{particle}}(t)\big)$$

Compute self-force via regularized mode sum:

$$F^c_{\text{self}} \sim \sum_\ell \left[\left(\sum_m \left[\nabla^c \phi_{\ell m}\right]_{\text{particle}} \right) - (\ell + \tfrac{1}{2})A - B \right]$$

Time-Domain Numerical Integration: Outline

We need to solve many wave equations (e.g., $\ell \leq 30 \Rightarrow 256$ modes) to very high accuracy, with time-dependent δ-function source terms.

Time-Domain Numerical Integration: Outline

We need to solve many wave equations (e.g., $\ell \leq 30 \Rightarrow 256$ modes) to very high accuracy, with time-dependent δ-function source terms.

The true BCs (representing the equilibrium metric perturbation) are unknown $\Rightarrow$ use arbitrary BCs ($\phi = 0$) and choose domain large enough for each mode so transients decay by end of integration.

Time-Domain Numerical Integration: Outline

We need to solve many wave equations (e.g., $\ell \leq 30 \Rightarrow 256$ modes) to very high accuracy, with time-dependent δ-function source terms.

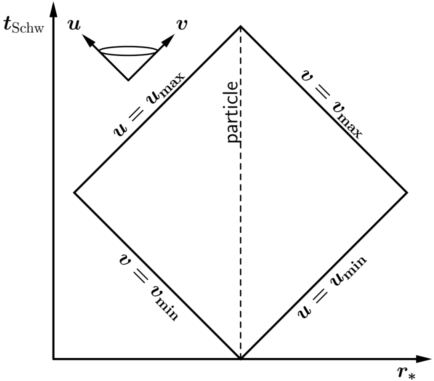
The true BCs (representing the equilibrium metric perturbation) are unknown $\Rightarrow$ use arbitrary BCs ($\phi = 0$) and choose domain large enough for each mode so transients decay by end of integration. (Domain size varies
strongly with ℓ.)

Time-Domain Numerical Integration: Outline

We need to solve many wave equations (e.g., $\ell \leq 30 \Rightarrow 256$ modes) to very high accuracy, with time-dependent δ-function source terms.

The true BCs (representing the equilibrium metric perturbation) are unknown $\Rightarrow$ use arbitrary BCs ($\phi = 0$) and choose domain large enough for each mode so transients decay by end of integration. (Domain size varies **strongly** with ℓ.)

Use **characteristic** grids, $\Rightarrow$ very easy to handle BCs



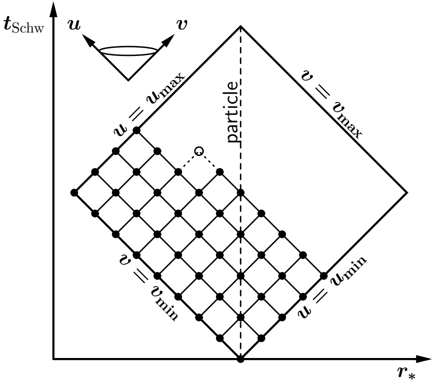
Time-Domain Numerical Integration: Outline

We need to solve many wave equations (e.g., $\ell \leq 30 \Rightarrow 256$ modes) to very high accuracy, with time-dependent δ-function source terms.

The true BCs (representing the equilibrium metric perturbation) are unknown $\Rightarrow$ use arbitrary BCs ($\phi = 0$) and choose domain large enough for each mode so transients decay by end of integration. (Domain size varies **strongly** with ℓ.)

Use **characteristic** grids, $\Rightarrow$ very easy to handle BCs

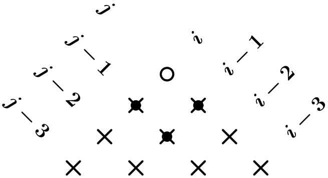
Integrate one $v =$ constant slice at a time via usual diamond-cell scheme (!)



4th Order Finite Differencing: Vacuum

Vacuum region:

- scheme described by Haas (2007):
- requires 3 past time levels, and 3 past points to start integrating a slice



4th Order Finite Differencing: Near the particle

δ-function source term $\Rightarrow$ ϕ is C^0 at the particle worldline

4th Order Finite Differencing: Near the particle

δ-function source term $\Rightarrow \phi$ is C^0 at the particle worldline

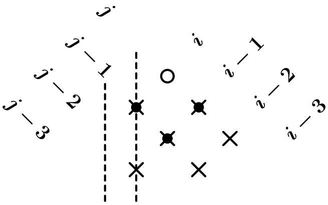
[Could use jump conditions like Sago.]

4th Order Finite Differencing: Near the particle

δ-function source term $\Rightarrow \phi$ is C^0 at the particle worldline
[Could use jump conditions like Sago.]

Use hybrid scheme:

- If evaluating at $i = i_{\text{particle}} \pm \{1, 2\}$, use Haas off-centered scheme.

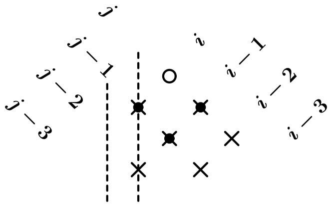


4th Order Finite Differencing: Near the particle

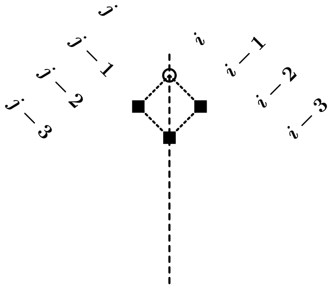
δ-function source term $\Rightarrow \phi$ is C^0 at the particle worldline
[Could use jump conditions like Sago.]

Use hybrid scheme:

- If evaluating at $i = i_{\text{particle}} \pm \{1, 2\}$, use Haas off-centered scheme.



- If evaluating at $i = i_{\text{particle}}$, compute 2nd order steps using cell sizes 1, 2, and 3 $\times$ normal,

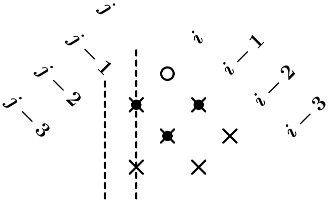


4th Order Finite Differencing: Near the particle

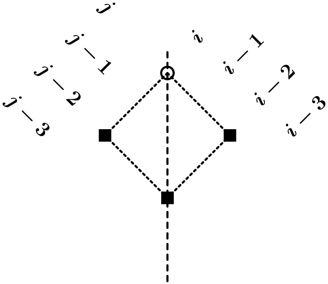
δ-function source term $\Rightarrow \phi$ is C^0 at the particle worldline
[Could use jump conditions like Sago.]

Use hybrid scheme:

- If evaluating at $i = i_{\text{particle}} \pm \{1, 2\}$, use Haas off-centered scheme.



- If evaluating at $i = i_{\text{particle}}$, compute 2nd order steps using cell sizes 1, 2, and 3 × normal,

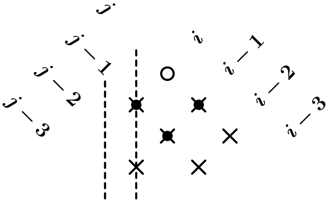


4th Order Finite Differencing: Near the particle

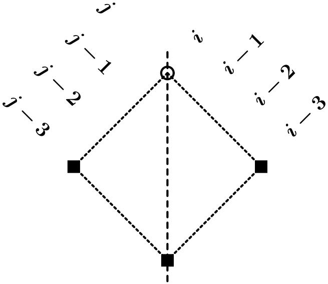
δ-function source term $\Rightarrow \phi$ is C^0 at the particle worldline
[Could use jump conditions like Sago.]

Use hybrid scheme:

- If evaluating at $i = i_{\text{particle}} \pm \{1, 2\}$, use Haas off-centered scheme.



- If evaluating at $i = i_{\text{particle}}$, compute 2nd order steps using cell sizes 1, 2, and 3 $\times$ normal,

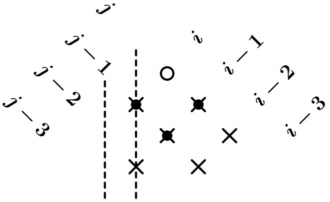


4th Order Finite Differencing: Near the particle

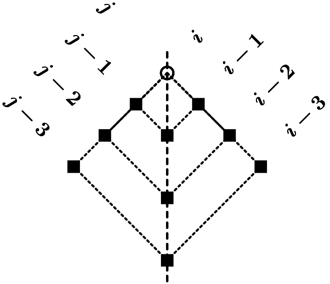
δ-function source term $\Rightarrow \phi$ is C^0 at the particle worldline
[Could use jump conditions like Sago.]

Use hybrid scheme:

- If evaluating at $i = i_{\text{particle}} \pm \{1, 2\}$, use Haas off-centered scheme.

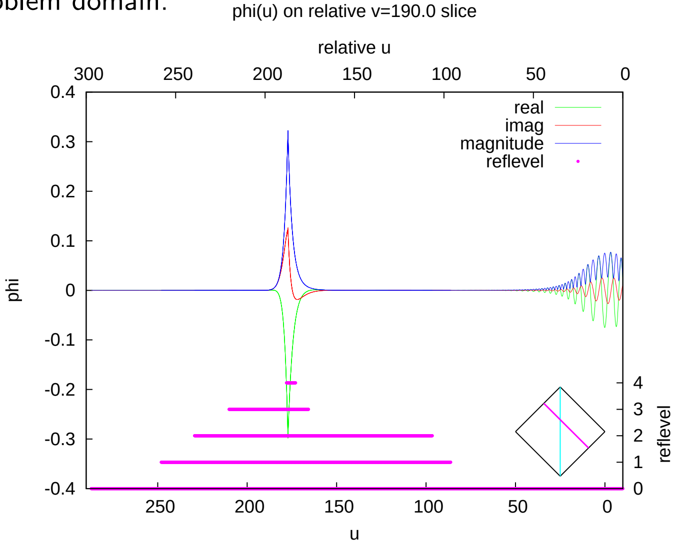


- If evaluating at $i = i_{\text{particle}}$, compute 2nd order steps using cell sizes 1, 2, and 3 $\times$ normal, then Richardson-extrapolate to 4th order.



Adaptive Mesh Refinement: Motivation

ϕ and its derivatives have very high dynamic range across the problem domain:



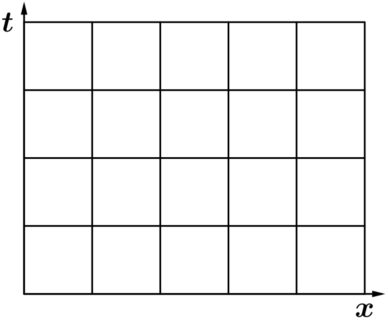
Adaptive Mesh Refinement: Cauchy Berger-Oliger

Berger & Oliger [*J. Comp. Phys.* **53**, 484 (1984)] defined what has become the standard approach to Eulierian finite-difference mesh refinement for "hyperbolic-like" time-evolution PDEs:

Adaptive Mesh Refinement: Cauchy Berger-Oliger

Berger & Oliger [*J. Comp. Phys.* **53**, 484 (1984)] defined what has become the standard approach to Eulierian finite-difference mesh refinement for "hyperbolic-like" time-evolution PDEs:

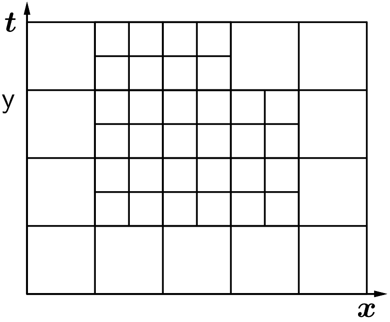
- use locally uniform grids (possibly in curvlinear coordinates)
- coarsest grid covers entire problem domain



Adaptive Mesh Refinement: Cauchy Berger-Oliger

Berger & Oliger [*J. Comp. Phys.* **53**, 484 (1984)] defined what has become the standard approach to Eulierian finite-difference mesh refinement for "hyperbolic-like" time-evolution PDEs:

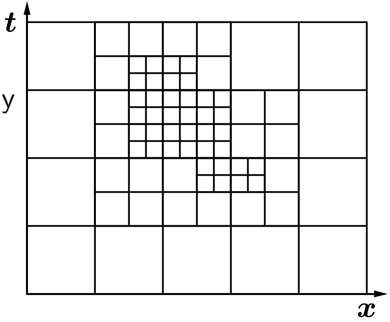
- use locally uniform grids (possibly in curvlinear coordinates)
- coarsest grid covers entire problem domain
- refine in both space and time
- fine grids overlay coarser grids
- integrate each grid independently



Adaptive Mesh Refinement: Cauchy Berger-Oliger

Berger & Oliger [*J. Comp. Phys.* **53**, 484 (1984)] defined what has become the standard approach to Eulierian finite-difference mesh refinement for "hyperbolic-like" time-evolution PDEs:

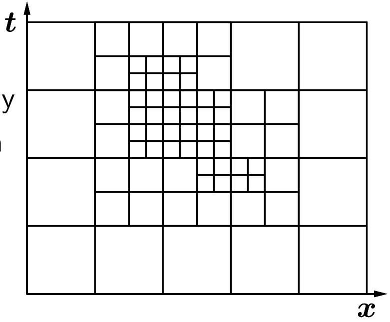
- use locally uniform grids (possibly in curvlinear coordinates)
- coarsest grid covers entire problem domain
- refine in both space and time
- fine grids overlay coarser grids
- integrate each grid independently



Adaptive Mesh Refinement: Cauchy Berger-Oliger

Berger & Oliger [*J. Comp. Phys.* **53**, 484 (1984)] defined what has become the standard approach to Eulierian finite-difference mesh refinement for "hyperbolic-like" time-evolution PDEs:

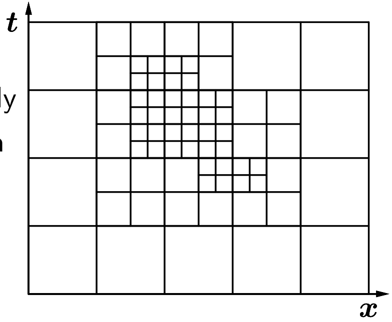
- use locally uniform grids (possibly in curvlinear coordinates)
- coarsest grid covers entire problem domain
- refine in both space and time
- fine grids overlay coarser grids
- integrate each grid independently
- fine-grid initial & boundary data interpolated from coarse grids



Adaptive Mesh Refinement: Cauchy Berger-Oliger

Berger & Oliger [*J. Comp. Phys.* **53**, 484 (1984)] defined what has become the standard approach to Eulierian finite-difference mesh refinement for "hyperbolic-like" time-evolution PDEs:

- use locally uniform grids (possibly in curvlinear coordinates)
- coarsest grid covers entire problem domain
- refine in both space and time
- fine grids overlay coarser grids
- integrate each grid independently
- fine-grid initial & boundary data interpolated from coarse grids
- inject fine-grid results back into coarse grid when/where points coincide [keeps coarse-grid solution from drifting away from (accurate) fine-grid solution]

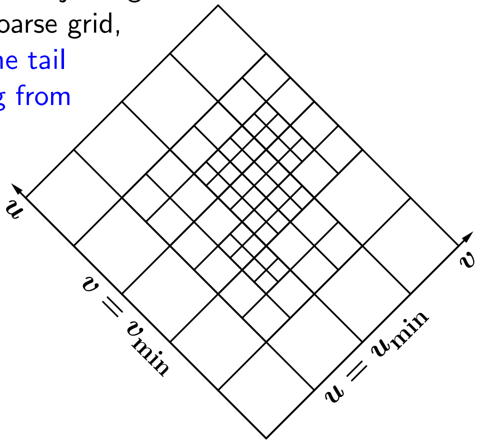


Adaptive Mesh Refinement: Characteristic Berger-Oliger

Standard Cauchy Berger-Oliger also works in characteristic coordinates, with one modification: after injecting fine-grid data back into coarse grid, we need to re-integrate the tail of the coarse slice starting from the injected data.

Adaptive Mesh Refinement: Characteristic Berger-Oliger

Standard Cauchy Berger-Oliger also works in characteristic coordinates, with one modification: after injecting fine-grid data back into coarse grid, we need to re-integrate the tail of the coarse slice starting from the injected data.



Tail Sum

Fit data points

$$\tfrac{1}{2}(F^{-}_{\ell,\mathrm{reg}} + F^{+}_{\ell,\mathrm{reg}}) \pm \tfrac{1}{2}|F^{+}_{\ell,\mathrm{reg}} - F^{-}_{\ell,\mathrm{reg}}|$$

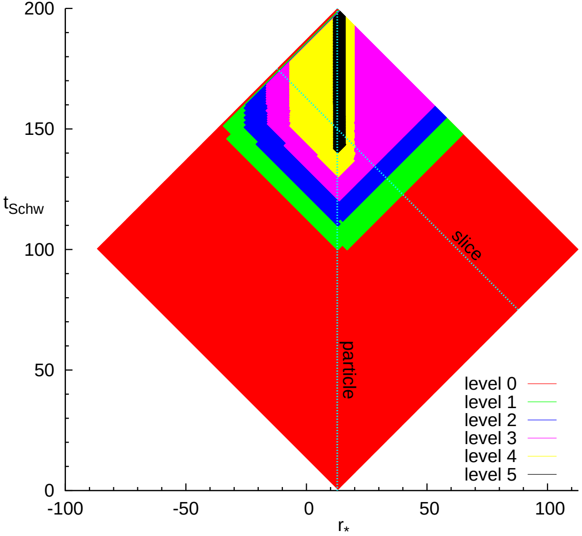
to series expansion [Detweiler, Messaritaki, and Whiting (2003)]:

$$\begin{aligned} F_{\ell,\mathrm{reg}} \quad = \quad & \frac{a_2}{(\ell-\frac{1}{2})(\ell+\frac{3}{2})} + \frac{a_4}{(\ell-\frac{3}{2})(\ell-\frac{1}{2})(\ell+\frac{3}{2})(\ell+\frac{5}{2})} \\ & + \frac{a_6}{(\ell-\frac{5}{2})(\ell-\frac{3}{2})(\ell-\frac{1}{2})(\ell+\frac{3}{2})(\ell+\frac{5}{2})(\ell+\frac{7}{2})} \\ & + \mathcal{O}(\ell^{-8}) \end{aligned}$$

to determine coefficients $\{a_2, a_4, a_6\}$, then evaluate $\sum_{\ell=K+1}^{\infty} F_{\ell,\mathrm{reg}}$ analytically (the sums telescope).
[Better would be to use analytically-known value of a_2.]

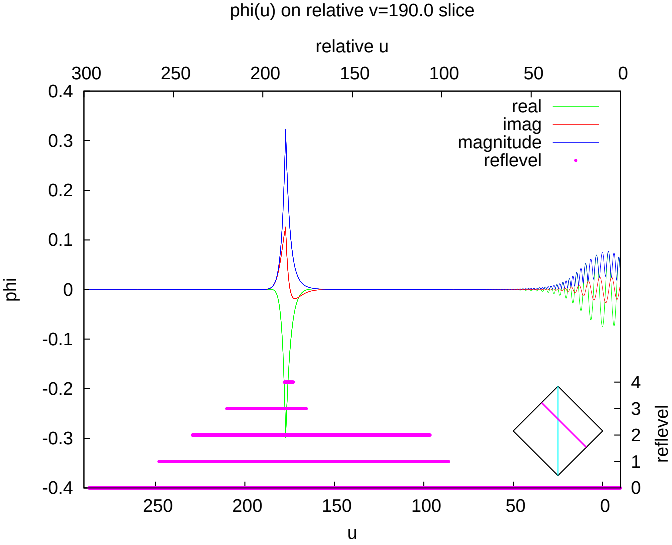
Sample Results: $(\ell, m) = (10, 10)$ Integration

Refinement-Level Map:



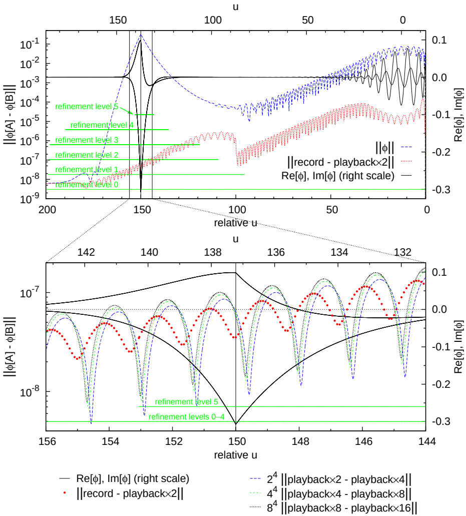
Sample Results: $(\ell, m) = (10, 10)$ Integration

Sample Frame $\Rightarrow$ Movie:



Sample Results: $(\ell, m) = (10, 10)$ Integration

4th Order Convergence with Grid Resolution:



Sample Results: Self-Force Calculation

Test case:

- Particle in circular orbit at $r = 10M$
- Modes computed numerically for $\ell \leq 30$ (256 modes).

Sample Results: Self-Force Calculation

Test case:

- Particle in circular orbit at $r = 10M$
- Modes computed numerically for $\ell \leq 30$ (256 modes).
- $\ell = 0$ domain size $30\,000M$
  $\ell = 1$ domain size $5\,000M$
  $\ell = 2$ domain size $1\,000M$
  $\ell = 3, 4$ domain size $500M$
  $\ell \geq 5$ domain size $400M$

Sample Results: Self-Force Calculation

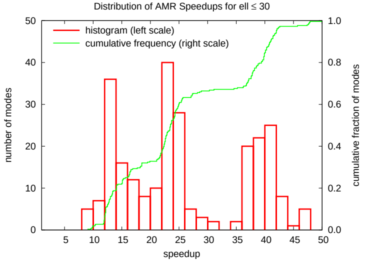
Test case:

- Particle in circular orbit at $r = 10M$
- Modes computed numerically for $\ell \leq 30$ (256 modes).
- $\ell = 0$ domain size $30\,000M$
  $\ell = 1$ domain size $5\,000M$
  $\ell = 2$ domain size $1\,000M$
  $\ell = 3, 4$ domain size $500M$
  $\ell \geq 5$ domain size $400M$
- AMR error tolerance
  $\|\phi^{\text{normal}} - \phi^{\frac{1}{2}\ \text{resolution}}\| \leq 10^{-15}$

Sample Results: Self-Force Calculation

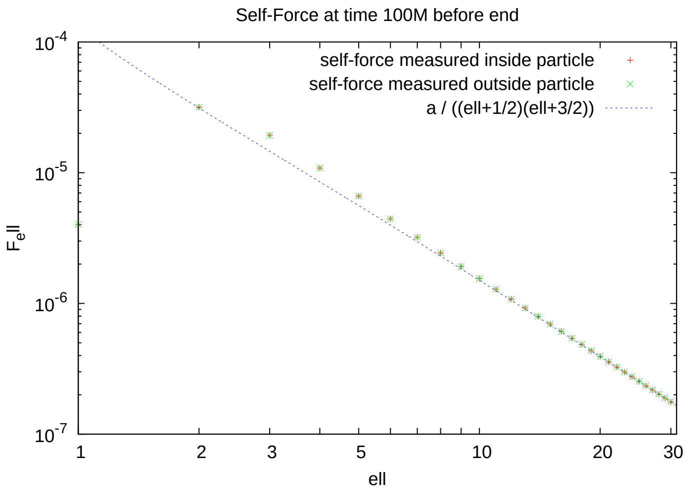
Test case:

- Particle in circular orbit at $r = 10M$
- Modes computed numerically for $\ell \le 30$ (256 modes).
- $\ell = 0$ domain size $30\,000M$
  $\ell = 1$ domain size $5\,000M$
  $\ell = 2$ domain size $1\,000M$
  $\ell = 3, 4$ domain size $500M$
  $\ell \ge 5$ domain size $400M$
- AMR error tolerance
  $\|\phi^{\text{normal}} - \phi^{\frac{1}{2}\text{ resolution}}\| \le 10^{-15}$
- Performance:
  $\ell = 0$: 108 minutes (1 proc),
  $\ell = 1$–30: $\sim$5 hours (10 procs)
  median AMR speedup ≈ 25



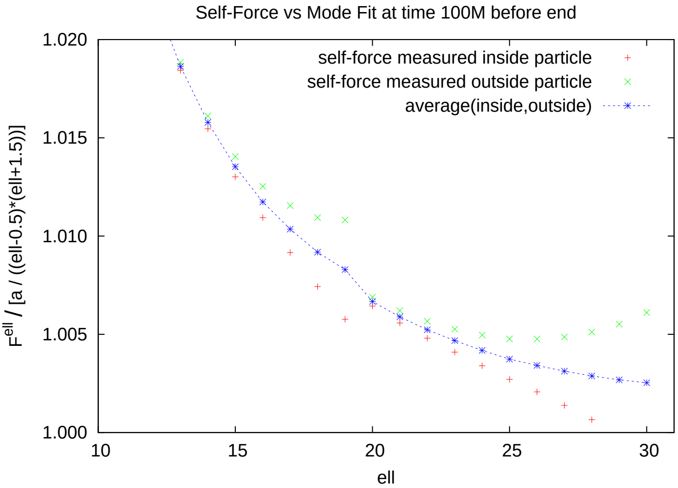
Sample Results: Self-Force Calculation

F_ℓ convergence with ℓ:



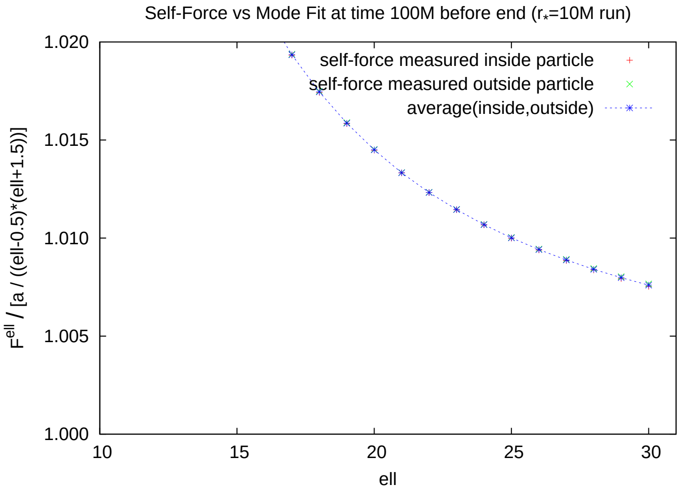
Sample Results: Self-Force Calculation

rescaled F_ℓ convergence with ℓ:



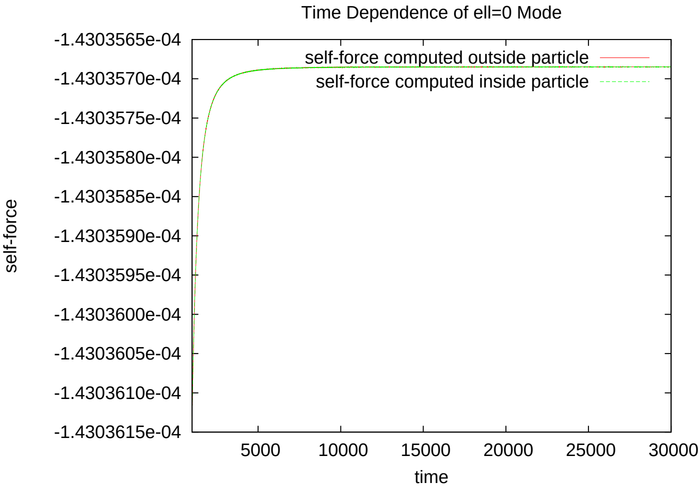
Sample Results: Self-Force Calculation #2

rescaled F_ℓ convergence with ℓ for $r_* = 10M$ run:



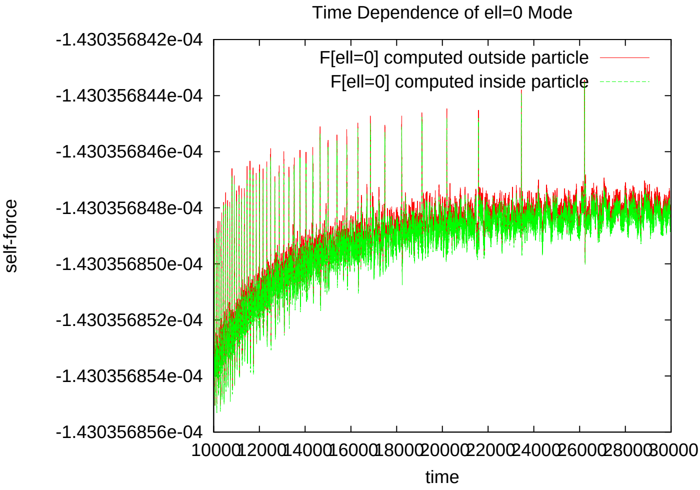
Sample Results: Self-Force Calculation #2

$F[\ell = 0]$ time dependence starting at $t = 1000M$:



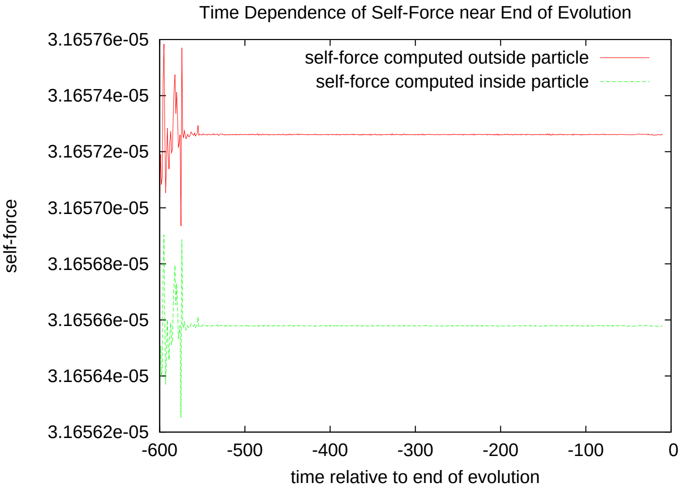
Sample Results: Self-Force Calculation #2

$F[\ell = 0]$ time dependence starting at $t = 10\,000M$:



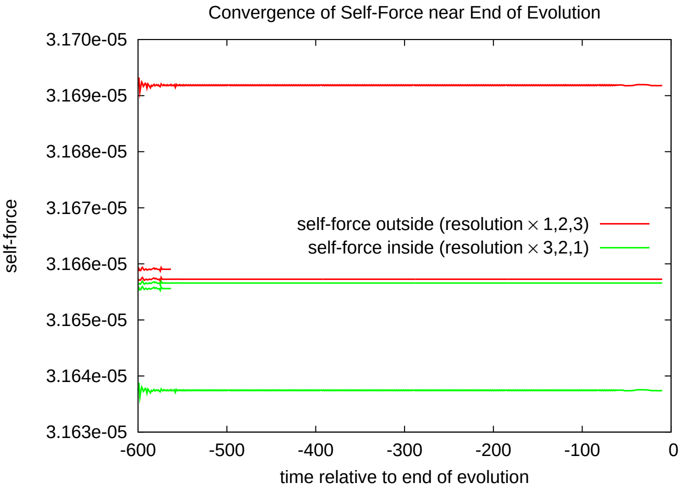
Sample Results: Self-Force Calculation #2

F time-and inside/outside-dependence near end of evolution:



Sample Results: Self-Force Calculation #2

F convergence with grid resolutions $\times 1$, $\times 2$, $\times 3$:



Sample Results: Self-Force Calculation #2

F (outside) time-dependence near end of evolution:

Conclusions

Current status:

- The characteristic Berger-Oliger algorithm works very well
- The combination of 4th order finite differencing and Berger-Oliger adaptive mesh refinement gives very high accuracy for self-force calculations

Conclusions

Current status:

- The characteristic Berger-Oliger algorithm works very well
- The combination of 4th order finite differencing and Berger-Oliger adaptive mesh refinement gives very high accuracy for self-force calculations

Near-term plans:

- Clean up tail fitting!

Conclusions

Current status:

- The characteristic Berger-Oliger algorithm works very well
- The combination of 4th order finite differencing and Berger-Oliger adaptive mesh refinement gives very high accuracy for self-force calculations

Near-term plans:

- Clean up tail fitting!

Longer-term plans:

- Extend finite differencing to handle eccentric orbits
- Implement "self-consistent orbit correction"