Using Multiple Grid Patches for Black Hole Excision

Jonathan Thornburg <jthorn@aei.mpg.de>

Max-Planck-Institut für Gravitationsphysik
Albert-Einstein-Institut
Golm, Germany

How to do BH Excision in 3D?

xyz grid, cubical excision region
$\Rightarrow$ continuum causality problems at corners,
finite differencing is bad too

could also try xyz grid,
"cube with corners cut off"
excision region
(generalizes 2D octagon)

xyz grid, lego-sphere excision region
$\Rightarrow$ very hard to stablely finite difference at excision surface

(r, θ, ϕ) grid, spherical excision region
$\Rightarrow$ nice finite differencing at excision surface
easy to use nonuniform grid in r to put outer boundary farther out
spherical outer boundary for radiation BCs
z axis coordinate singularity

multiple grid patches, spherical excision region
$\Rightarrow$ potentially same advantages as (r, θ, ϕ) grid
need inter-patch interpolation (stability?)
tricky updates for ghost-zone corners
maybe tricky to use existing unigrid software

(all of the above, multiple grid patches) $\Rightarrow$ this project

Project Overview/Context

Goals:

- try 3D multiple grid patches for BH excision
- see how serious the problems (stability, tricky coding) are in practice

Simplifying Assumptions:

- every slice contains a (single) BH at the origin $\Rightarrow$ excised
- vacuum Einstein equations only – no matter, no shocks

Code:

- grid is $r \times \{$**multiple patches covering** $\boldsymbol{S^2}\}$
- free evolution, ADM or BSSN formulation of the Einstein equations
- 4th order finite differencing (3rd or 4th order for radial boundaries)
- standalone code, not (yet?) in Cactus
- main test case so far is evolution of Kerr ($J/m^2 = 0.6$) (coords $\Rightarrow \partial_t \mathsf{u} = 0$)
- in the future, should be able to evolve distorted rotating BH

Multipatch Design Issues

General issues:

- What formulation of the Einstein equations?
- How many patches (what coordinates/grids) to cover the domain?
- What tensor basis for the Einstein equations?

Issues more specific to this project:

- Should adjacent patches have a Δx gap between them, just touch, or overlap?
- How to coordinate-transform BSSN $\tilde{\Gamma}^i$ in ghost zones?
- How to handle ghost-zone corners?
- How to handle "triple" ghost-zone corners?
- How to do multiple patches in Cactus, or more generally, how to reuse/interoperate with existing uni-patch software?

How Many Patches (What Coordinates) to Cover S^2?

2 patches (stereographic coordinates)

- relatively simple: only one interpatch boundary
- can use eth formalism
- relatively large coordinate distortion near patch boundaries
  $\Rightarrow$ less accurate finite differencing
- either the equator isn't a grid line (irregular overlap between patches)
  or (maybe?) there's a z axis coordinate singularity

6 patches ("inflated cube" coordinates)

- relatively complex: many interpatch boundaries
- relatively low coordinate distortion near patch boundaries
  $\Rightarrow$ accurate finite differencing
- can have adjacent patches share $\perp$ and r coordinates
  $\Rightarrow$ only need 1D ($\|$) interpatch interpolation

6-Patch "Inflated Cube" Coordinates and Topology

I use 6 angular patches to cover S^2, with angular coordinates …
in neighborhood of $\pm z$ axis: (rotation about x axis, rotation about y axis)
in neighborhood of $\pm x$ axis: (rotation about y axis, rotation about z axis)
in neighborhood of $\pm y$ axis: (rotation about x axis, rotation about z axis)

r coordinate is common to all patches
and adjacent patches share $\perp$ coordinates
$\Rightarrow$ only need 1D ($\|$) interpatch interpolation
to fill in ghost-zone values (ghost zones
always overlap neighboring patches)

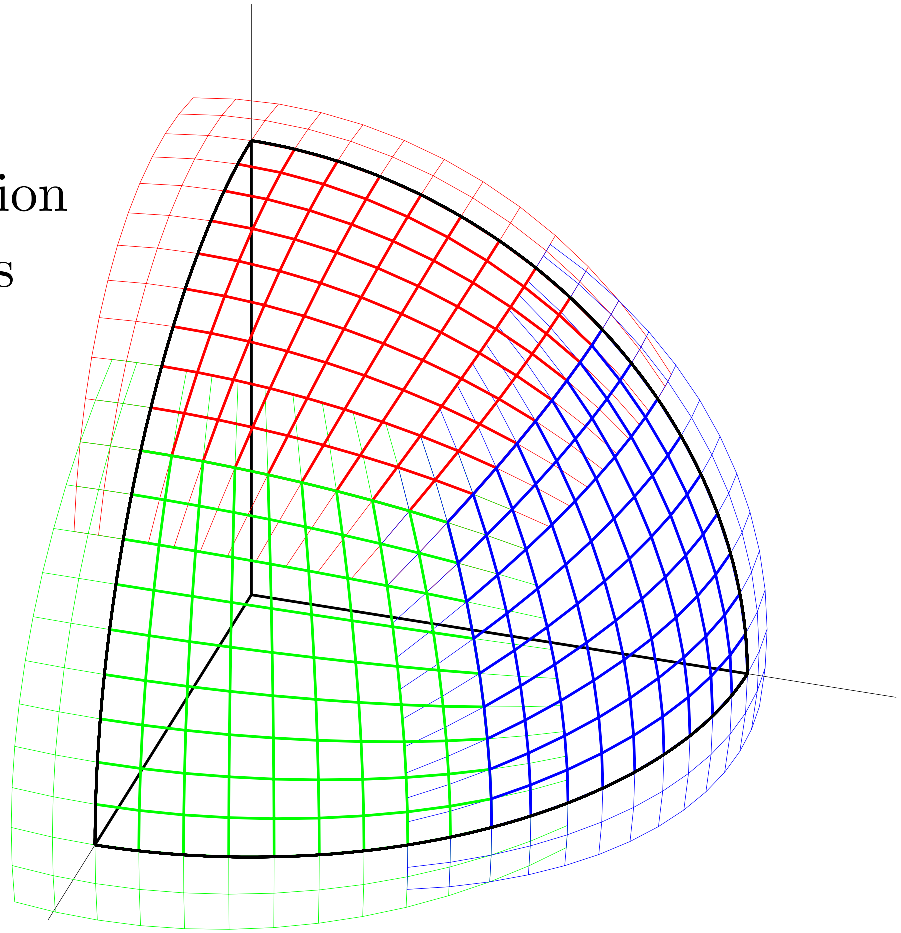


figure shows 3-patch system covering
$(+,+,+)$ octant of S^2; 4/5/6 patches
cover quadrant/hemisphere/full sphere

figure shows adjacent patches just
touching (ghost zones overlap)

What Tensor Basis for the Einstein Equations?

xyz basis everywhere

- no need for interpatch coordinate transformation
- maybe easier to interface to other (xyz-grid) software?
- must transform grid finite differences into xyz partial derivatives ($\equiv$ use a non-coordinate basis) everywhere in the grid $\Rightarrow$ may be slow
- harder to treat r specially (either for excision FD or for radiation BCs)

each patch uses its own local coordinate basis

- must do interpatch coordinate transformations (only tensors for ADM, but BSSN $\tilde{\Gamma}^i$ requires $\partial_k \phi$ in ghost zones!)
- but only need these at patch boundaries $\Rightarrow$ patch interiors ($=$ most grid points) can run at full speed
- easy to treat r specially (either for excision FD or for radiation BCs)

Interpatch Coordinate Transformations

Assume we know the interpatch coordinate transformation analytically. Then...

- ADM g_{ij} and K_{ij} are tensors $\Rightarrow$ easy to transform: $K_{ab}(p) = Y^i{}_a Y^j{}_b K_{ij}(q)$
- BSSN $\tilde{g}_{ij}$ and $\tilde{A}_{ij}$ are tensor densities (and $\phi = \log(\text{tensor density})$)

  $\Rightarrow$ still easy to transform: $\tilde{A}_{ab}(p) = \left|Y^{\|}{}_{\|}\right|^{-2/3} Y^i{}_a Y^j{}_b \tilde{A}_{ij}(q)$
- BSSN $\tilde{\Gamma}^i$ is not a tensor density $\Rightarrow$ hard to transform:

$$\begin{aligned}\tilde{\Gamma}^a(p) \quad &= \quad \left|Y^{\|}{}_{\|}\right|^{2/3} X^a{}_k \tilde{\Gamma}^k(q) + X^a{}_k Y^k{}_{bc}\, \tilde{g}^{bc}(p) \\ &\qquad +2\tilde{g}^{ab}(p)\partial_b\phi(p) - 2\left|Y^{\|}{}_{\|}\right|^{2/3} X^a{}_k\, \tilde{g}^{k\ell}(q)\partial_\ell\phi(q)\end{aligned}$$

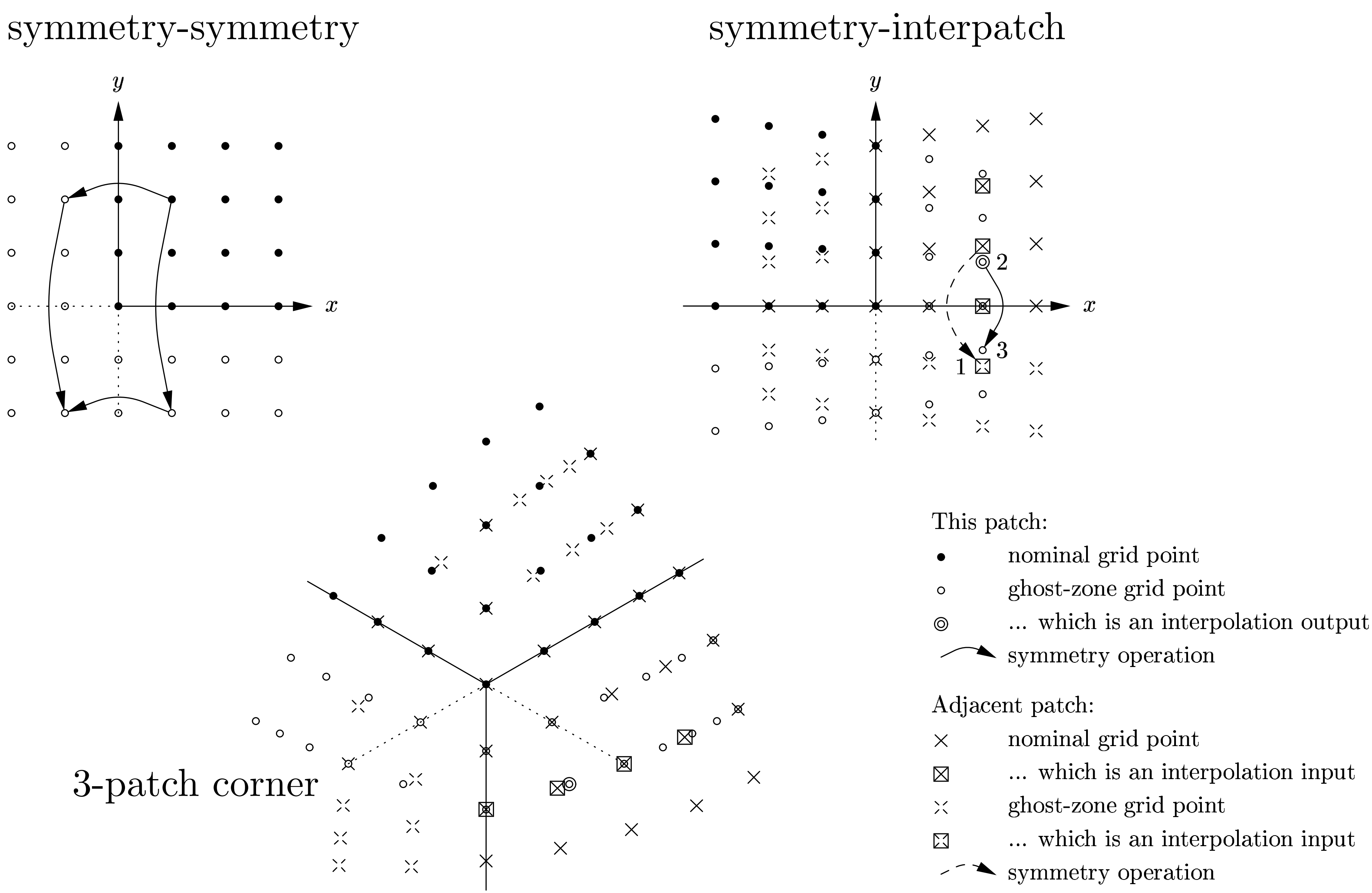
  $\Rightarrow$ Need spatial partial derivatives of ϕ, in ghost zones! I can see 2 options:

  - We are in p's ghost zone, which is presumably in q's interior, so $\partial_\ell\phi(q)$ is easy. We know the $\phi(p) \leftrightarrow \phi(q)$ transformation, so we can write $\partial_b\phi(p)$ in terms of $\partial_\ell\phi(q)$. (Actually this is only needed for $\partial_{\perp(p)}\phi(p)$, since $\partial_r\phi(p)$ and $\partial_{\|(p)}\phi(p)$ are $\|$ to boundary $\Rightarrow$ ok for finite differencing.)

    $\Rightarrow$ Problem: grids are incommensurate $\Rightarrow$ need more interpolation.
  - Have a double-width ghost zone for ϕ.

    $\Rightarrow$ Conceptually simple, but a bit messy to implement.

Ghost Zone Updates at Patch Corners are Tricky



Multiple Patches in Cactus

It would be nice to be able to do multi-patch in Cactus. How to do this?

In order from "clean/elegant, but lots of work to implement" to "quick-n-dirty":

- multi-models: Cactus knows about multiple grid patches
  problem: multi-models aren't planned for Cactus until ≥ 4.1
  (probably $\gtrsim 18$ months from now in practice)
- write a multi-patch driver
  problem: Cactus driver interface isn't that well documented
  though in practice this may not be a severe problem
  problem: must "reinvent the wheel" for lots of stuff in existing drivers
- have each patch be a Carpet grid or grids,
  do interpatch interpolation as Carpet "boundary conditions"
- tile patches onto Cactus grid, fixup interpatch boundaries by hand
- interpolate Cactus initial data into multi-patch grid,
  interpolate multi-patch results back to Cactus for analysis

Project Status

Old version of the code (ADM only):

- as expected for ADM, there is an overall constraint-violating instability which seems to be unrelated to the multipatch structure
- finite difference instability at “triple corner” where 3 patches meet

(Very) New version of the code (slightly different interpatch interpolation):

- ADM does **not** show the “triple corner” instability, but instead a medium-wavelength instability along interpatch boundaries.
- BSSN shows no signs of any multipatch instabilities out to $t = 100m$
  code currently uses Dirichlet outer BC $\Rightarrow$ outer boundary instability
  (appear to be unrelated to multipatch)
- there is no sign of any instability at the “triple corner”
- there is no sign of any instability at the excision boundary
- 4th order convergence in patch interiors, 3rd order near interpatch boundaries

Sample Results: BSSN, Kerr $a = 0.6 \to t = 100m$

This constraint violation does **not** seem to grow with time (at least for $t \leq 100m$)

$$C_{\rm rel} \equiv \frac{{}^{(3)}R - K_{ij}K^{ij} + K^2}{|{}^{(3)}R| + |K_{ij}K^{ij}| + K^2}$$