

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

excision region

“cube with corners cut off”

could also try  $xyz$  grid,

(generalizes 2D octagon)

$xyz$  grid, lego-sphere excision region

$\Rightarrow$  very hard to stably finite difference at excision surface

$(r, \theta, \phi)$  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  $(r, \theta, \phi)$  grid

need inter-patch interpolation (stability?)

tricky updates for ghost-zone corners

maybe tricky to use existing unigrid software

$\Leftrightarrow$  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  $r \times \{\text{multiple patches covering } S^2\}$
- 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  $\Rightarrow \partial_t u = 0$ )
- in the future, should be able to evolve distorted rotating BH

# Multi-patch 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  $\Delta x$  gap between them, just touch, or overlap?
- How to coordinate-transform BSSN  $\tilde{T}^i$  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 ( $\parallel$ ) 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

$\Leftrightarrow$  only need 1D (||) interpolation

to fill in ghost-zone values (ghost zones

always overlap neighboring patches)

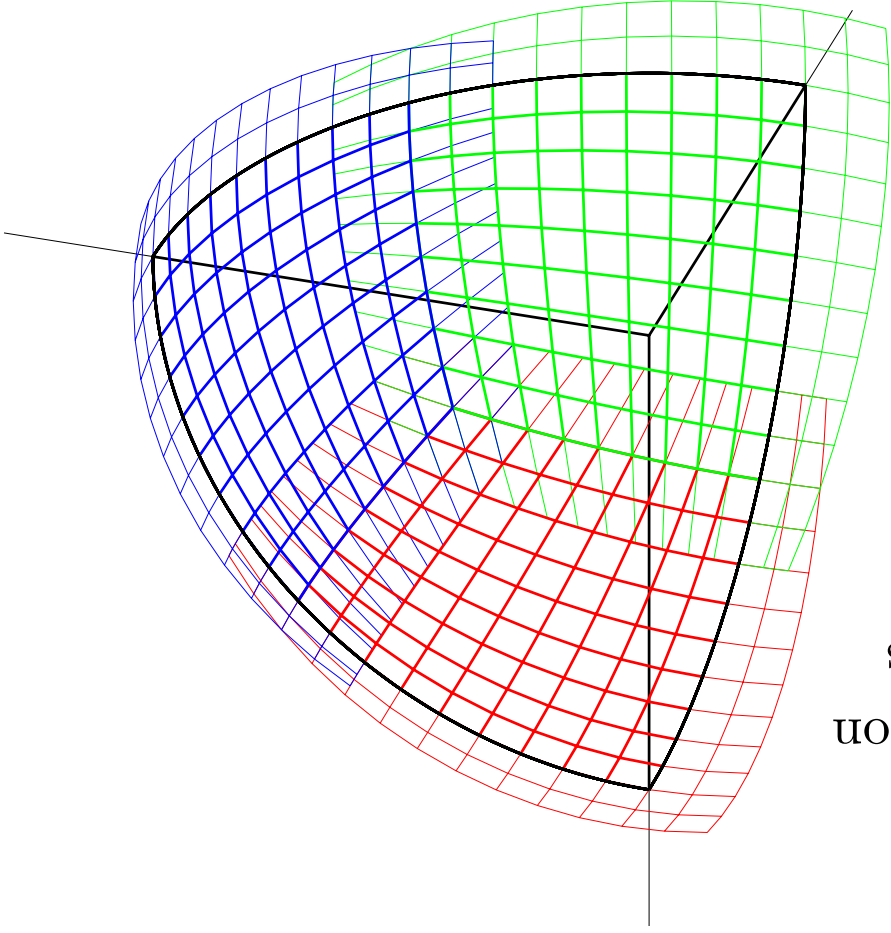
figure shows 3-patch system covering

(+, +, +) 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  $\tilde{T}^i$  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:  $K^{ab}(d) = Y^i{}_a Y^j{}_b K^{ij}(q)$
- BSSN  $\tilde{g}_{ij}$  and  $\tilde{A}_{ij}$  are **tensor densities** (and  $\phi = \log(\text{tensor density})$ )  $\Rightarrow$  still easy to transform:  $\tilde{A}^{ab}(d) = |Y_\parallel|^{-2/3} Y^i{}_a Y^j{}_b \tilde{A}^{ij}(q)$

- BSSN  $\tilde{\Gamma}^i$  is **not a tensor density**  $\Rightarrow$  hard to transform:

$$\tilde{\Gamma}^a(d) = |Y_\parallel|^{2/3} X^a{}_k \tilde{\Gamma}^k(b) + X^a{}_k Y^{bc} \tilde{g}^{bc}(d) + 2\tilde{g}^{ab}(d) \partial_b \phi(d) - 2|Y_\parallel|^{2/3} X^a{}_k \tilde{g}^{kl}(b) \partial_l \phi(b)$$

$\Rightarrow$  Need **spatial partial derivatives of  $\phi$** , 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  $\partial_b \phi(d)$  in terms of  $\partial_\ell \phi(q)$ . (Actually this is only needed for  $\partial_\perp \phi(d)$ , since  $\partial_r \phi(d)$  and  $\partial_\parallel \phi(d)$  are  $\parallel$  to boundary  $\Rightarrow$  ok for finite differencing.)

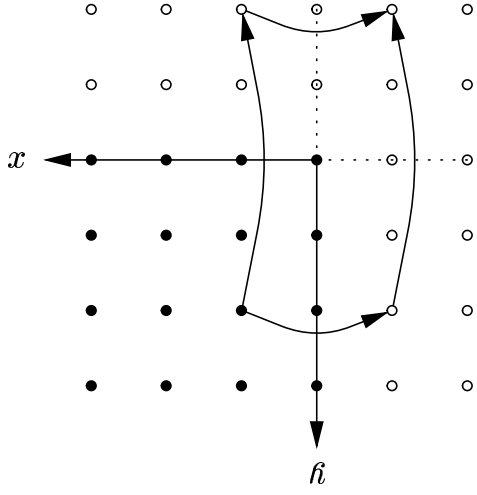
$\Rightarrow$  Problem: grids are incommensurate  $\Rightarrow$  **need more interpolation**.

- Have a **double-width ghost zone** for  $\phi$ .

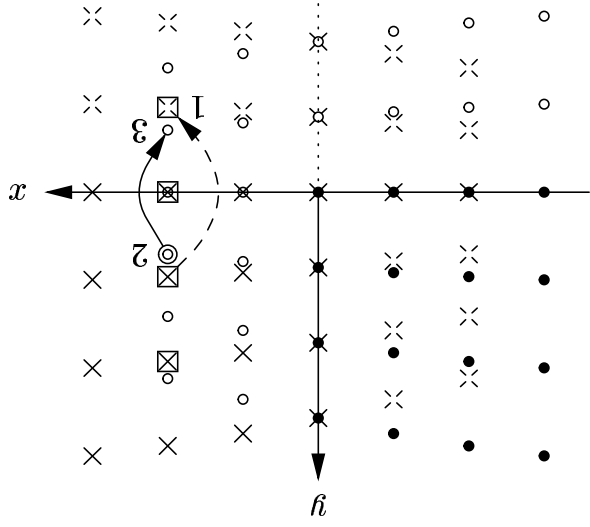
$\Rightarrow$  Conceptually simple, but a bit messy to implement.

# Ghost Zone Updates at Patch Corners are Tricky

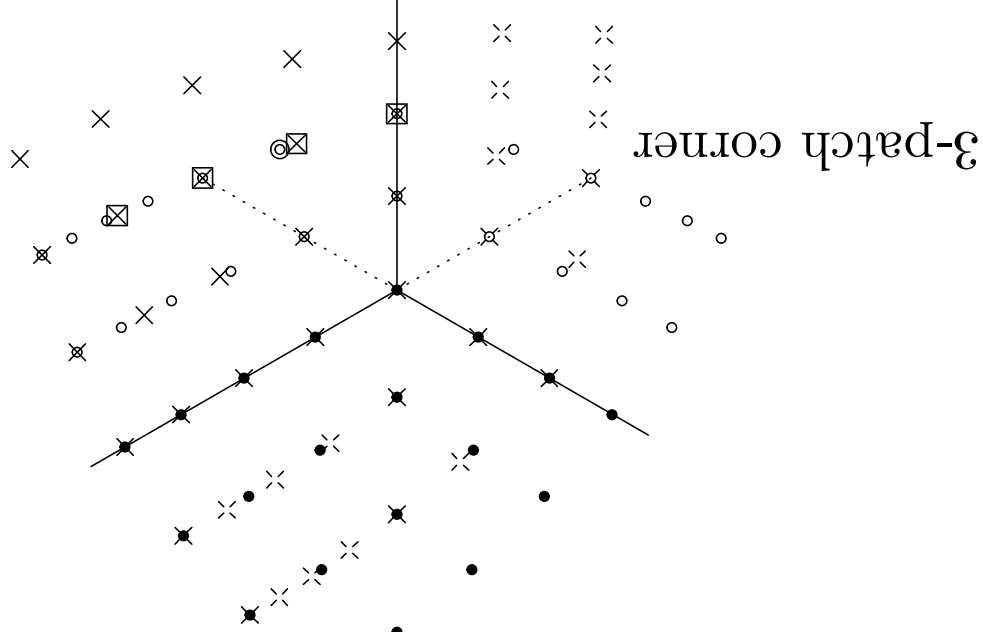
symmetry-symmetry



symmetry-interpatch



- This patch:
- nominal grid point
  - ghost-zone grid point
  - ⊙ ... which is an interpolation output
- Adjacent patch:
- × nominal grid point
  - ⊠ ... which is an interpolation input
  - ⊗ ghost-zone grid point
  - ⊠ ... which is an interpolation input
- symmetry operation
- symmetry operation



3-patch corner

## 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  $\geq 4.1$

(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 interpolation):

- ADM does **not** show the “triple corner” instability, but instead a

medium-wavelength instability along interface 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 interface boundaries

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

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

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

