Numerical Relativity and the Einstein Toolkit

Miguel Zilhão
Universidade de Aveiro

IX Amazonian Workshop on Gravity and Analogue Models,
Universidade Federal do Pará
June 2024

Link for VM image

Numerical Relativity

What is Numerical Relativity

Numerical Relativity:

solving numerically the full GR equations, typically for dynamical spacetimes in the strong field regime, where no approximations hold

Goals:

understanding gravity in its full non-linear glory

Challenges:

very difficult problem…

Why Numerical Relativity

Systems with strong and dynamical gravitational fields:

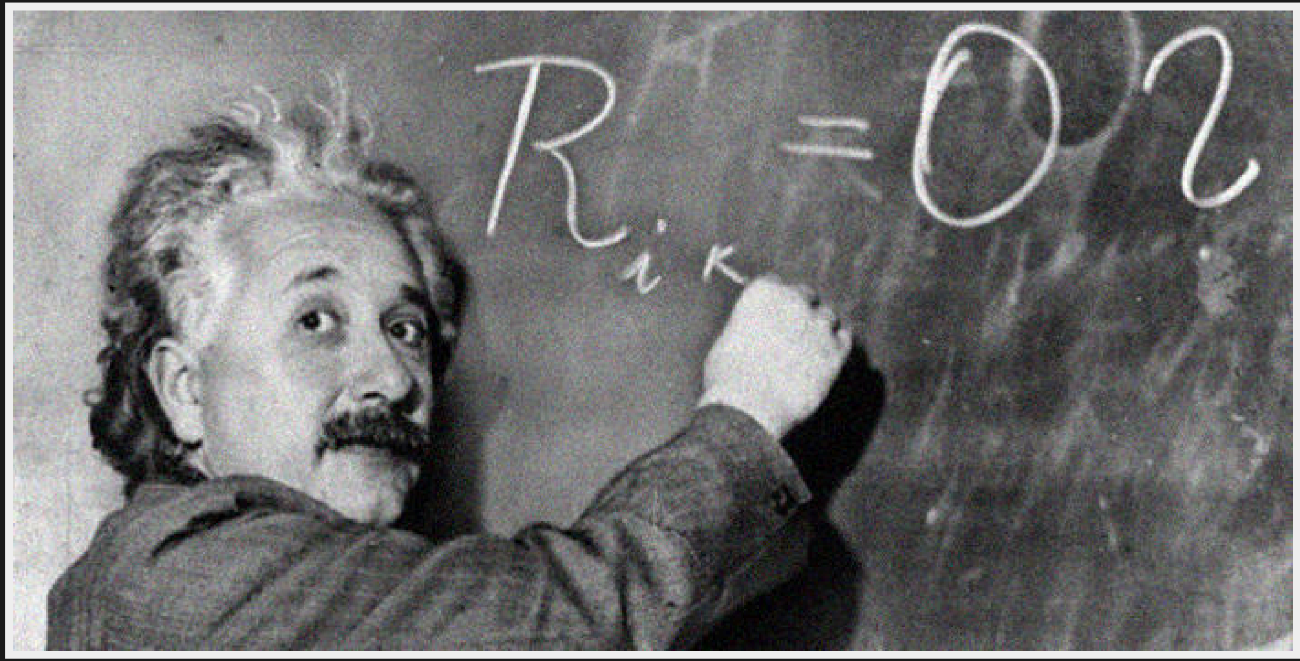
- Gravitational radiation
  - Astrophysics, gravitational wave astronomy
- Mathematical and theoretical Physics
  - Cosmic censorship
  - Instabilities (Black hole interior, Myers-Perry)
- High-energy particle systems
  - AdS/CFT correspondence

History and milestones

Year	Milestone	Group
1915	Einstein's equations are published	Einstein
1964	First documented attempts at numerical simulations	Hahn & Lindquist
1976	Head-on collision of two BHs (in axisymmetry)	Smarr & Eppley
1990s	"Binary Black Hole Grand Challenge Project"	Matzner et al
1993	Critical phenomena in gravitational collapse	Choptuik
1997	Release of Cactus 1.0	Seidel et al
1998	Generic (3D) single BH simulation	Gomez et al
1999	Baumgarte-Shapiro-Shibata-Nakamura evolution system	BSSN
2005	First simulations of BH binaries	Pretorius
2006	"Moving puncture" simulations	UTB/RIT; NASA Goddard

Formalism

Einstein's equations (1915)



$$R_{\mu\nu} - \frac{R}{2} g_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$$

$$\Gamma^{\alpha}_{\beta\gamma} = \frac{1}{2} \sum_{\delta=t,x^1,\ldots,x^{D-1}} g^{\alpha\delta} \left(\partial_\gamma g_{\delta\beta} + \partial_\beta g_{\delta\gamma} - \partial_\delta g_{\beta\gamma}\right)$$

$$8\pi T_{\alpha\beta} = \sum_\delta \left[\partial_\delta \Gamma^{\delta}_{\alpha\beta} - \partial_\alpha \Gamma^{\delta}_{\delta\beta} + \sum_\gamma (\Gamma^{\delta}_{\alpha\beta}\Gamma^{\gamma}_{\delta\gamma} - \Gamma^{\delta}_{\gamma\beta}\Gamma^{\gamma}_{\delta\alpha})\right]$$

$$- \frac{1}{2} g_{\alpha\beta} \sum_{\delta,\gamma} \left\{ g^{\delta\gamma} \sum_\mu \left[\partial_\mu \Gamma^{\mu}_{\delta\gamma} - \partial_\delta \Gamma^{\mu}_{\mu\delta} + \sum_\nu (\Gamma^{\mu}_{\delta\gamma}\Gamma^{\nu}_{\mu\nu} - \Gamma^{\mu}_{\nu\gamma}\Gamma^{\nu}_{\mu\delta})\right]\right\}$$

Generalized Harmonic Gauge (GHG)

$$R_{\mu\nu} = 8\pi \left(T_{\mu\nu} - \frac{1}{2} g_{\mu\nu} T \right)$$

Impose that coordinates are harmonic:

$$H^{\alpha} \equiv \Box x^{\alpha} = \frac{1}{\sqrt{-g}} \partial_{\lambda} \left(\sqrt{-g}\, g^{\lambda\alpha} \right)$$

and promote H^{α} to independently evolved variables

The Einstein equations take the form

$$g^{\mu\nu}\partial_\mu\partial_\nu g_{\alpha\beta} = -2\partial_\nu g_{\mu(\alpha}\,\partial_{\beta)}g^{\mu\nu} - 2\partial_{(\alpha}H_{\beta)} + 2H_\mu\Gamma^\mu_{\alpha\beta} - 2\Gamma^\mu_{\nu\alpha}\Gamma^\nu_{\mu\beta}$$
$$- 8\pi\left(2T_{\alpha\beta} - Tg_{\alpha\beta}\right) - 2\kappa\left[2n_{(\alpha}\mathcal{C}_{\beta)} - g_{\alpha\beta}n^\mu\mathcal{C}_\mu\right]$$

- Set of 2nd order **wave equations** for each component of spacetime metric:

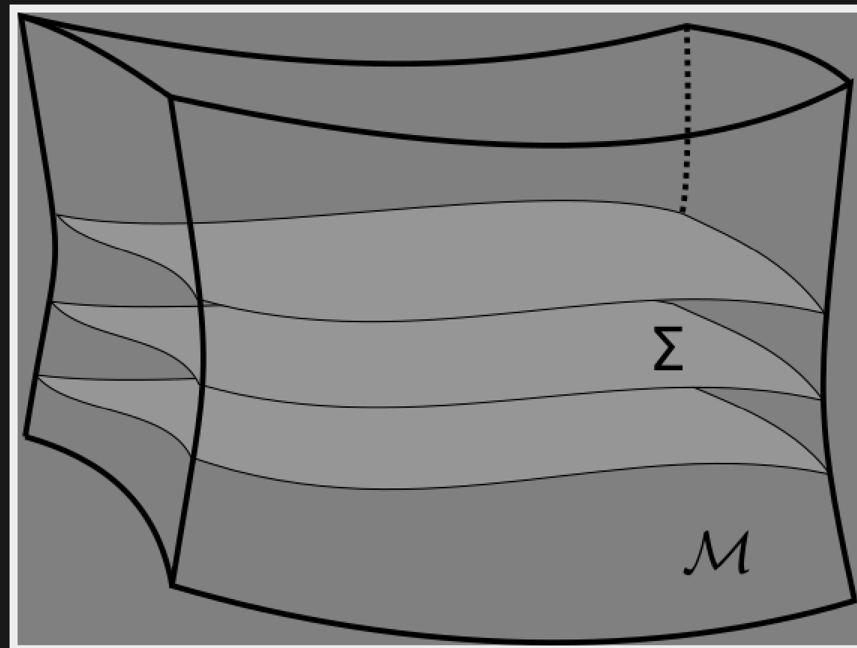
$$\Box^{\text{flat}} g_{\alpha\beta} \simeq \cdots$$

- Equations are *symmetric hyperbolic*
- Used in Pretorius' 2005 breakthrough simulations

$\mathcal{C}^\mu = H^\mu - \Box x^\mu$

3+1 decomposition

Foliations



- Consider a continuous set of hypersurfaces Σ covering the manifold $\mathcal{M}$

- Define:
  - *Eulerian observer*

$$n = -\alpha \nabla t$$

  - *Induced metric* (or *3-metric*)

$$\gamma_{\alpha\beta} = g_{\alpha\beta} + n_\alpha n_\beta$$

  - *Extrinsic curvature*

$$K_{\alpha\beta} = -\gamma_\alpha{}^\mu \gamma_\beta{}^\nu \nabla_\nu n_\mu$$

Projection of Einstein's equations

1. Full projection onto Σ (ie, act with $\gamma_\alpha{}^\mu \gamma_\beta{}^\nu$ on trace-reversed equations):

$$\gamma_\alpha{}^\mu \gamma_\beta{}^\nu R_{\mu\nu} = 8\pi \left(S_{\alpha\beta} - \frac{1}{2}(S - E)\gamma_{\alpha\beta} \right)$$

Trace-reversed form of the Einstein equations:

$$R_{\mu\nu} = 8\pi \left(T_{\mu\nu} - \frac{1}{2} g_{\mu\nu} T \right)$$

2. Full projection perpendicular to Σ (ie, act with $n^\mu n^\nu$):

$$n^\mu n^\nu R_{\mu\nu} + \frac{1}{2}R = 8\pi E$$

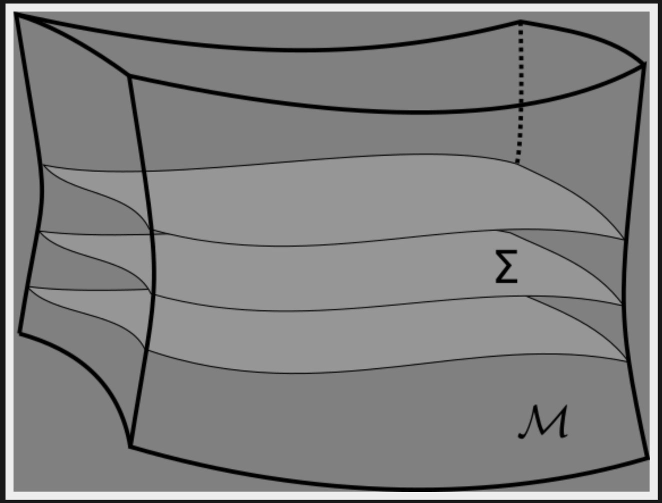
3. Mixed projection (ie, act with ${\gamma_\alpha}^\mu n^\nu$):

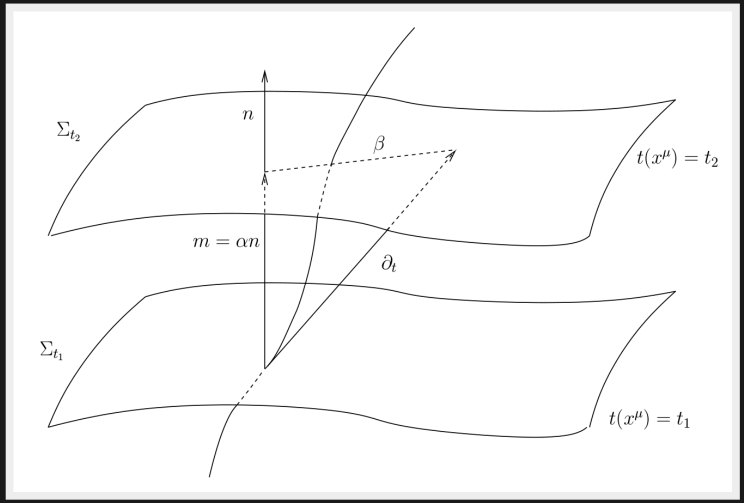
$${\gamma_\alpha}^\mu n^\nu R_{\mu\nu} = -8\pi p_\alpha$$

Einstein's equations:

$$R_{\mu\nu} - \frac{R}{2}g_{\mu\nu} = 8\pi T_{\mu\nu}$$

3+1 decomposition





We write the metric as

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

To learn more

- “3+1 formalism and bases of numerical relativity” (Gourgoulhon 2007)

ADM-York equations

Evolution equations

$$
\begin{aligned}
\left(\partial_t - \mathcal{L}_\beta\right)\gamma_{ij} &= -2\alpha K_{ij}, \\
\left(\partial_t - \mathcal{L}_\beta\right) K_{ij} &= -\nabla_i \nabla_j \alpha + \alpha \Big[R_{ij} + K K_{ij} - 2 K_{ik} K^k{}_j \\
&\qquad + 4\pi \left((S - E)\gamma_{ij} - 2 S_{ij} \right) \Big]
\end{aligned}
$$

Constraints

$$
\begin{aligned}
R + K^2 - K_{ij} K^{ij} &= 16\pi E \\
\nabla_j \left(K^{ij} - \gamma^{ij} K \right) &= 8\pi p^i
\end{aligned}
$$

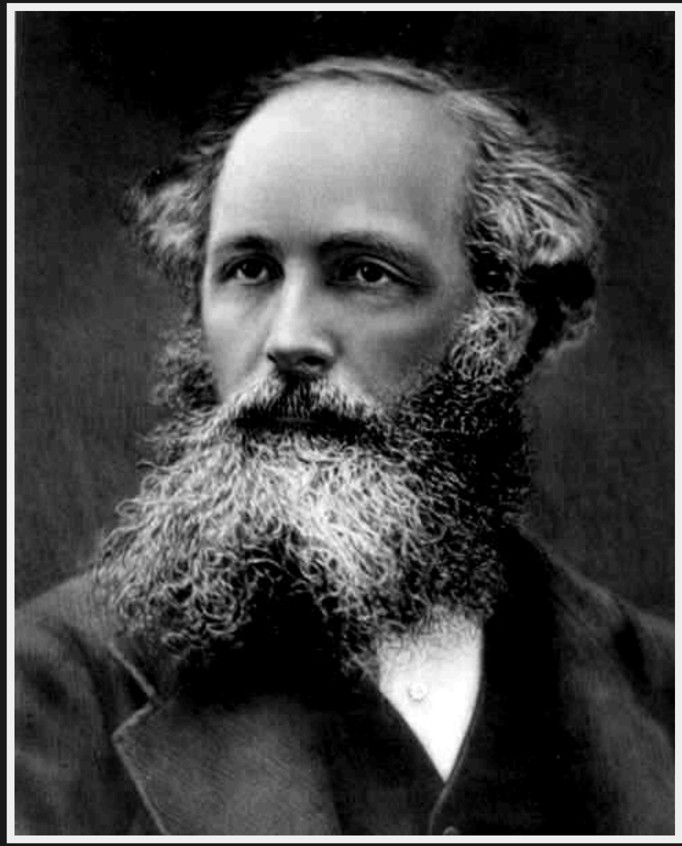
Electromagnetic analogy

Evolution equations

$$-\partial_t \vec{E} + \nabla \times \vec{H} = 4\pi \vec{j}$$
$$-\partial_t \vec{H} + \nabla \times \vec{E} = 0$$

Constraints

$$\nabla \cdot \vec{E} = 4\pi\rho$$
$$\nabla \cdot \vec{H} = 0$$



BSSN equations

$$
\begin{aligned}
\partial_t \tilde{\gamma}_{ij} &= \beta^k \partial_k \tilde{\gamma}_{ij} + 2\tilde{\gamma}_{k(i}\partial_{j)}\beta^k - \frac{2}{3}\tilde{\gamma}_{ij}\partial_k\beta^k - 2\alpha\tilde{A}_{ij}, \\
\partial_t \chi &= \beta^k \partial_k \chi + \frac{2}{3}\chi(\alpha K - \partial_k\beta^k), \\
\partial_t \tilde{A}_{ij} &= \beta^k \partial_k \tilde{A}_{ij} + 2\tilde{A}_{k(i}\partial_{j)}\beta^k - \frac{2}{3}\tilde{A}_{ij}\partial_k\beta^k + \chi(\alpha R_{ij} - \nabla_i\partial_j\alpha)^{\mathrm{TF}} \\
&\quad + \alpha\left(K\,\tilde{A}_{ij} - 2\tilde{A}_i{}^k\tilde{A}_{kj}\right) - 8\pi\alpha\left(\chi S_{ij} - \frac{S}{3}\tilde{\gamma}_{ij}\right), \\
\partial_t K &= \beta^k\partial_k K - \nabla^k\partial_k\alpha + \alpha\left(\tilde{A}^{ij}\tilde{A}_{ij} + \frac{1}{3}K^2\right) + 4\pi\alpha(E + S), \\
\partial_t \tilde{\Gamma}^i &= \beta^k\partial_k\tilde{\Gamma}^i - \tilde{\Gamma}^k\partial_k\beta^i + \frac{2}{3}\tilde{\Gamma}^i\partial_k\beta^k + 2\alpha\tilde{\Gamma}^i_{jk}\tilde{A}^{jk} + \frac{1}{3}\tilde{\gamma}^{ij}\partial_j\partial_k\beta^k \\
&\quad + \tilde{\gamma}^{jk}\partial_j\partial_k\beta^i - \frac{4}{3}\alpha\tilde{\gamma}^{ij}\partial_j K - \tilde{A}^{ij}\left(3\alpha\chi^{-1}\partial_j\chi + 2\partial_j\alpha\right) - 16\pi\alpha\chi^{-1}j^i
\end{aligned}
$$

Initial data

- *Mathematical problem*: solve the constraint equations

$$R + K^2 - K_{ij}K^{ij} = 16\pi E$$
$$\nabla_j \left(K^{ij} - \gamma^{ij} K\right) = 8\pi p^i$$

- *Physical problem*: make sure that the data represents the physical system in question

Example: momentarily static initial data

$$E = 0, \qquad p^i = 0, \qquad K_{ij} = 0$$

$$\gamma_{ij} dx^i dx^j = \left(1 + \frac{m_1}{2\vec{r}_1} + \frac{m_2}{2\vec{r}_2}\right)^4 (dx^2 + dy^2 + dz^2)$$

Interaction Energy in Geometrostatics

Dieter R. Brill*
Yale University, New Haven, Connecticut

and

Richard W. Lindquist
Adelphi College, Garden City, New York

(Received 4 March 1963)

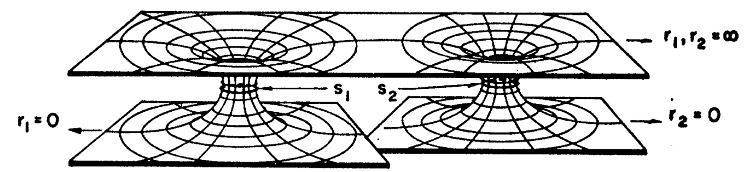
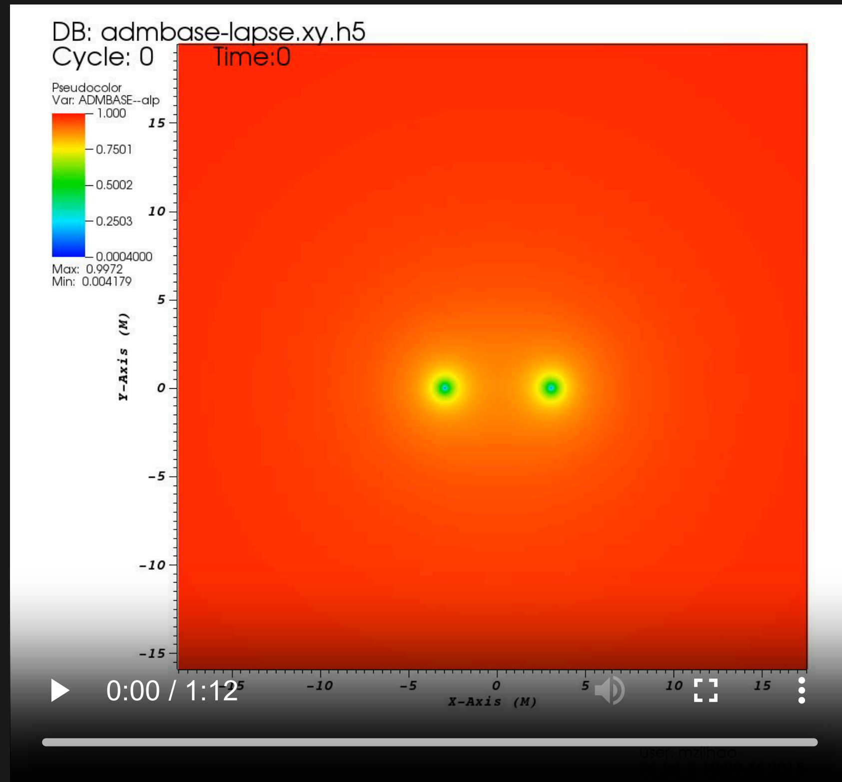


Fig. 2. A two-dimensional analog of the hypersurface of time symmetry of a manifold containing two "throats" is shown isometrically imbedded in flat three-space. The figure illustrates the curvature and topology for a system of two "particles" of equal mass m, and separation large compared to m, described by the metric

$$ds^2 = (1 + m/2r_1 + m/2r_2)^4 ds_F^2.$$

Applications

Black hole collisions

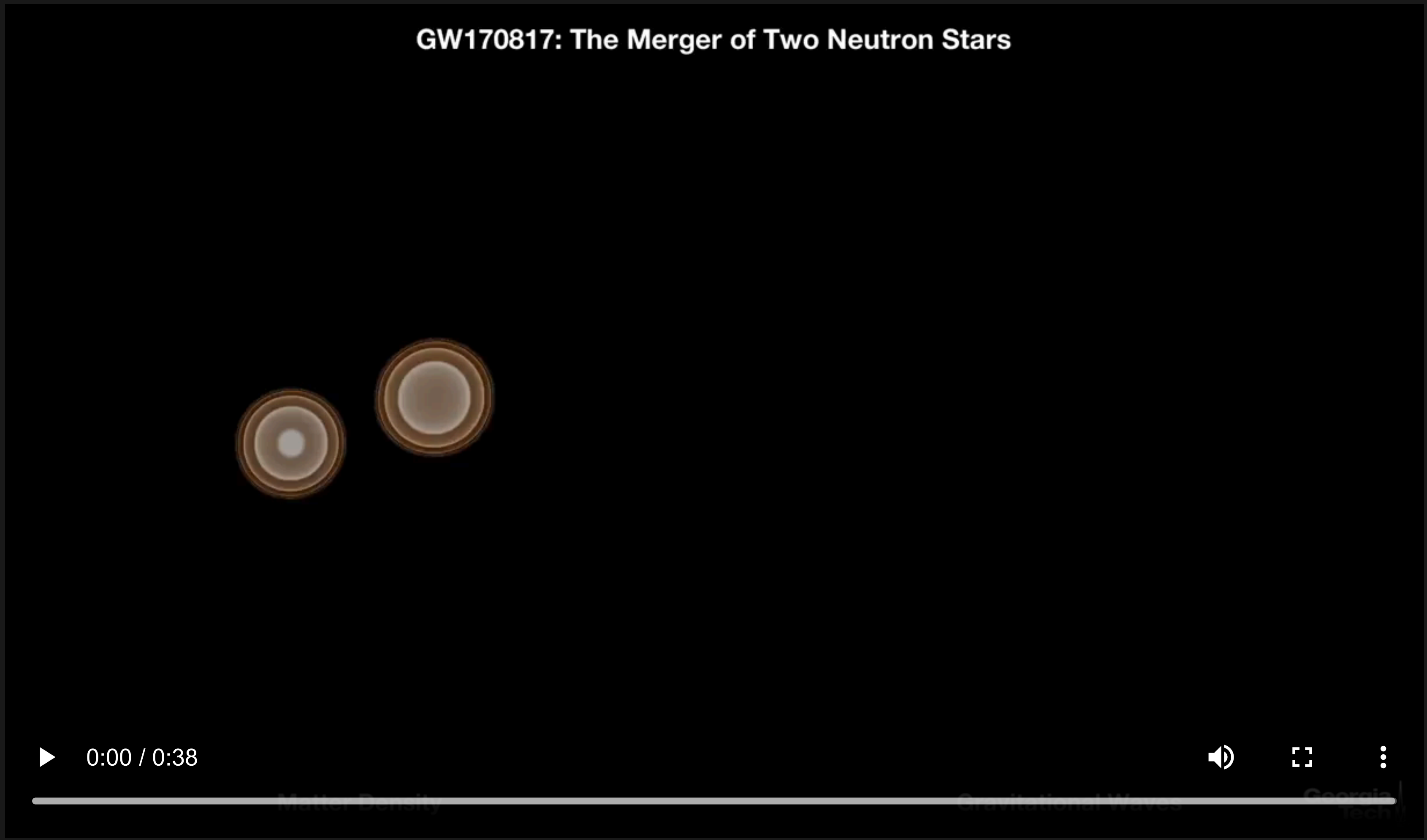


Simulation performed with the Einstein Toolkit

To know more

- “An Introduction to the Einstein Toolkit” (Zilhão and Löffler 2013)

Neutron star binaries



[Georgia Tech]

Boson star dynamics

Video and simulation by Pedro Ildefonso

Target practice with black holes

Video and simulation by Zhen Zhong (Cardoso et al. 2022)

The Einstein Toolkit

Before numerical evolution

1. Write Einstein's equations as a well-posed ***Initial Boundary Value Problem*** (IBVP):
    - solution's behaviour depends continuously with the initial data;
    - numerically suitable gauge conditions.
2. **Discretize** resulting PDEs
3. Specify (constraint preserving) physically correct, **boundary conditions**
4. Find a way to deal with singularities

During numerical evolution

- Compute constraint-satisfying **initial data** representing snapshot of physical system
- Implement **mesh refinement**, or similar, to efficiently handle different length scales (and parallelize resulting algorithms)
- **Extract physical results** in gauge-invariant fashion from numerical data

Why Cactus/Einstein Toolkit

Typical problem in Numerical Relativity:

- mesh refinement
- efficiently parallelize
- large input/output
- somewhat complex tools for analysis

Typical workflow:

1. Compute initial data
2. Evolve equations
3. Analysis

What is Cactus

- General framework for the development of portable, modular applications
- Programs are split into independent components called `thorns`
- `thorns` are developed independently and should be interchangeable with others with same functionality
- `thorns` don't directly interact with each other
  - `Cactus` framework (`flesh`) provides the "glue"
- Supports C, C++, Fortran

What is the Einstein Toolkit

ET $\approx$ Cactus $+$ collection of `thorns` and tools for numerical relativity:

- initial data
- vacuum spacetime solver
- hydrodynamics solver
- analysis tools (apparent and event horizon finder, wave extraction, ...)
- ...

Einstein Toolkit history

- First version (Bohr) released 2010-06-17
- New version released $\approx$ every 6 months
- Current version (Meitner) released 2023-12-14
- Many contributors over the past decade, from all over the world
- Currently made up by over 270 Cactus components

Einstein Toolkit goals

The Einstein Toolkit is a community-driven software platform of core computational tools to advance and support research in relativistic astrophysics and gravitational physics.

- Separating **Physics** from **Computational Science**
- Providing computational tools which are
  - maintained
  - of high quality
  - easy to use
  - *open source*
- User can focus on science

Obtaining the Einstein Toolkit

- https://einsteintoolkit.org/

- https://einsteintoolkit.org/download.html

Contents: `arrangements`

- Several Cactus `thorns` (I/O, Method of Lines, . . .)
- Carpet (Adaptive Mesh Refinement driver)
- EinsteinBase
- EinsteinInitial
- EinsteinAnalysis
- McLachlan, Lean (BSSN implementation)
- . . .

Contents: tools

- GetComponents
- Simfactory
- Formaline

Main core Cactus `arrangements`

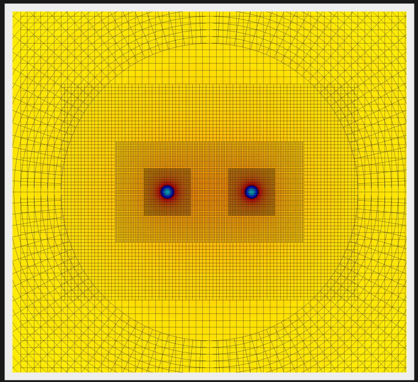
CactusBase
: Infrastructure thorns for boundary conditions, coordinates, IO, symmetries and time

CactusNumerical
: Numerical infrastructure thorns: time integration, dissipation, symmetry boundary conditions, spherical surfaces, local interpolation, Method of Lines (MoL), …

CactusUtils
: Utility thorns: formaline, nan-checking, termination triggering and timer reports

ExternalLibraries
: Provides external libraries: Lapack, GSL, HDF5, FFTW, Lorene (for initial data) and others

Carpet



Berger-Oliger Adaptive Mesh Refinement (AMR) driver:

- Splits grid functions and arrays among the MPI processes
- Setups mesh refinement grid hierarchy
- Communicates ghost cell information between MPI processes
- Communicates between refinement levels by prolongation and restriction
- Modifies grid hierarchy (regridding) when requested
- Performs parallel IO

ET `arrangement` EinsteinBase

Main `thorns`:

- ADMBase (defines spacetime variables)
- HydroBase (defines hydro variables)
- TmunuBase (defines “right-hand-side” of Einstein’s equations)
- EOS_Base (EOS registration)
- EOSG_Base (interface to generic EOS)

EinsteinBase: ADMBase

Holds everything else together:

1. Defines groups of grid functions for basic variables:
   - metric (`gxx`, `gxy`, `gxz`, `gyy`, `gyz`, `gzz`)
   - extrinsic curvature (`kxx`, `kxy`, `kxz`, `kyy`, `kyz`, `kzz`)
   - gauge: lapse (`alp`), shift (`betax`, `betay`, `betaz`)
2. Defines basic parameters to choose the initial data, evolution method and number of active timelevels that other thorns can use or extend where appropriate.
3. Defines a schedule group for other thorns to schedule their routines modifying the ADMBase variables.

EinsteinBase: TmunuBase

Defines grid functions for stress-energy tensor:

- The time component T_{00}: `eTtt`
- The mixed components T_{0i}: `eTtx`, `eTty`, `eTtz`
- The spatial components T_{ij}: `eTxx`, `eTxy`, `eTxz`, `eTyy`, `eTyz`, `eTzz`

Also sets up scheduling groups for other thorns to schedule routines that adds to the stress-energy tensor

ET `arrangement` EinsteinInitial

- **TwoPunctures**
  - Puncture binary black hole initial data
- **TOVSolver**
  - Single TOV star
- **GRHydro_InitData**
  - Test initial data for hydro evolutions

Metric evolution `thorns`

- **Lean**
  - BSSN evolution code
- **McLachlan**
  - BSSN and Z4 evolution code (machine generated)

EinsteinAnalysis

AHFinderDirect
: Find black hole apparent horizons (quickly)

AHfinder
: Find black hole apparent horizons (slowly)

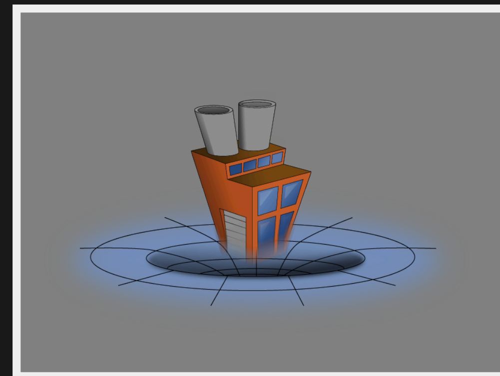
WeylScal4
: Calculate the Newman-Penrose scalar Ψ_4

Multipole
: Decompose arbitrary grid functions into spin-weighted spherical harmonics

ADMAnalysis
: Calculate several quantities from the ADM variables

EHFinder
: Find event horizons in numerical spacetimes

Tools: `Simfactory`



- Access remote systems, synchronise source code trees
- Configure and build on different systems semi-automatically
- Provide maintained list of supercomputer configurations
- Manage simulations (follow “best practices”, avoid human errors)

Hands-on session

Virtual Machine

Install VirtualBox. With my Fedora 36 installation I just had to run

```
sudo dnf install VirtualBox
```

1. Open VirtualBox
2. Go to

   File > Import Appliance
   Under “File”, choose the provided “ET_2022-05.ova” virtual appliance and cick “Next”.
3. Adjust any of the default settings and click “Import”

Your VM should now be ready. Select it and press “Start”. If all goes well, the system should login automatically. If you need to install anything, the credentials are as follows:

```
user: meudon

password: ET2022
```

WaveMoL `thorn`: wave equation

$$\partial_t^2 \phi = \partial_{x^i}^2 \phi^i$$

To illustrate usage of Cactus Method of Lines (MoL) thorn, we rewrite the equations in first order form:

$$\begin{aligned}
\partial_t \Phi &= \partial_{x^i} \Pi^i, \\
\partial_t \Pi^j &= \partial_{x^j} \Phi, \\
\partial_t \phi &= \Phi, \\
\partial_{x^j} \phi &= \Pi^j.
\end{aligned}$$

The first three equations (five separate PDEs) will be evolved. The final equation is used to set the initial data and can be thought of as a constraint.

Anatomy of a Cactus `thorn`

Dissecting `WaveMoL`:

```
WaveMoL
├── README
├── doc
├── configuration.ccl
├── interface.ccl
├── param.ccl
├── schedule.ccl
├── src
│   ├── InitSymBound.c
│   ├── make.code.defn
│   ├── Startup.c
│   ├── WaveMoL.c
│   └── WaveMoLRegister.c
└── test
```

```
WaveMoL
├── README
├── doc
├── configuration.ccl
├── interface.ccl
├── param.ccl
├── schedule.ccl
├── src
│   ├── InitSymBound.c
│   ├── make.code.defn
│   ├── Startup.c
│   ├── WaveMoL.c
│   └── WaveMoLRegister.c
└── test
```

interface.ccl

```
implements: wavemol       # Name of thorn as seen by other thorns

USES INCLUDE: Symmetry.h

CCTK_INT FUNCTION MoLRegisterEvolvedGroup(CCTK_INT IN EvolvedIndex, \
                                          CCTK_INT IN RHSIndex)
CCTK_INT FUNCTION MoLRegisterConstrained(CCTK_INT IN ConstrainedIndex)

REQUIRES FUNCTION MoLRegisterEvolvedGroup
REQUIRES FUNCTION MoLRegisterConstrained

CCTK_INT FUNCTION Boundary_SelectGroupForBC(CCTK_POINTER_TO_CONST IN GH, \
  CCTK_INT IN faces, CCTK_INT IN boundary_width, CCTK_INT IN table_handle, \
  CCTK_STRING IN var_name, CCTK_STRING IN bc_name)
REQUIRES FUNCTION Boundary_SelectGroupForBC

public

cctk_real scalarevolvemol_scalar type = GF Timelevels = 3 tags='tensortypealias="Scalar"'
{
  phi
  phit
} "The scalar field and time derivative"

cctk_real scalarevolvemol_vector type = GF Timelevels = 3 tags='tensortypealias="U"'
{
```

```
WaveMoL
├── README
├── doc
├── configuration.ccl
├── interface.ccl
├── param.ccl
├── schedule.ccl
├── src
│   ├── InitSymBound.c
│   ├── make.code.defn
│   ├── Startup.c
│   ├── WaveMoL.c
│   └── WaveMoLRegister.c
└── test
```

schedule.ccl

```
STORAGE: scalarevolvemol_scalar[3], scalarevolvemol_vector[3]
STORAGE: scalarrhsmol_scalar, scalarrhsmol_vector
STORAGE: energy

schedule WaveMoL_Startup at STARTUP
{
  LANG: C
} "Register Banner"

schedule WaveMoL_InitSymBound at BASEGRID
{
  LANG: C
  OPTIONS: META
} "Schedule symmetries"

schedule WaveMoL_RegisterVars in MoL_Register
{
  LANG: C
  OPTIONS: META
} "Register variables for MoL"

schedule WaveMoL_CalcRHS in MoL_CalcRHS
{
  LANG: C
} "Register RHS calculation for MoL"

```

```
WaveMoL
├── README
├── doc
├── configuration.ccl
├── interface.ccl
├── param.ccl
├── schedule.ccl
├── src
│   ├── InitSymBound.c
│   ├── make.code.defn
│   ├── Startup.c
│   ├── WaveMoL.c
│   └── WaveMoLRegister.c
└── test
```

param.ccl

```
shares: MethodOfLines

USES CCTK_INT MoL_Num_Evolved_Vars
USES CCTK_INT MoL_Num_Constrained_Vars
USES CCTK_INT MoL_Num_SaveAndRestore_Vars

restricted:

CCTK_INT WaveMoL_MaxNumEvolvedVars "The maximum number of evolved variables used by WaveMoL" ACCUMUL
{
  5:5        :: "Just 5: phi and the four derivatives"
} 5

CCTK_INT WaveMoL_MaxNumConstrainedVars "The maximum number of constrained variables used by WaveMoL"
{
  1:1        :: "The energy"
} 1

private:

KEYWORD bound "Type of boundary condition to use"
{
  "none"      :: "No boundary condition"
  "flat"      :: "Flat boundary condition"
  "radiation" :: "Radiation boundary condition"
} "none"
```

```
WaveMoL
├── README
├── doc
├── configuration.ccl
├── interface.ccl
├── param.ccl
├── schedule.ccl
├── src
│   ├── InitSymBound.c
│   ├── make.code.defn
│   ├── Startup.c
│   ├── WaveMoL.c
│   └── WaveMoLRegister.c
└── test
```

WaveMoLRegister.c

```c
void WaveMoL_RegisterVars(CCTK_ARGUMENTS) {

  DECLARE_CCTK_ARGUMENTS;
  DECLARE_CCTK_PARAMETERS;

  CCTK_INT ierr = 0, group, rhs, var;

  group = CCTK_GroupIndex("wavemol::scalarevolvemol_scalar");
  rhs = CCTK_GroupIndex("wavemol::scalarrhsmol_scalar");

  ierr += MoLRegisterEvolvedGroup(group, rhs);

  group = CCTK_GroupIndex("wavemol::scalarevolvemol_vector");
  rhs = CCTK_GroupIndex("wavemol::scalarrhsmol_vector");

  ierr += MoLRegisterEvolvedGroup(group, rhs);

  var = CCTK_VarIndex("wavemol::energy");

  ierr += MoLRegisterConstrained(var);
}
```

```
WaveMoL
├── README
├── doc
├── configuration.ccl
├── interface.ccl
├── param.ccl
├── schedule.ccl
├── src
│   ├── InitSymBound.c
│   ├── make.code.defn
│   ├── Startup.c
│   ├── WaveMoL.c
│   └── WaveMoLRegister.c
└── test
```

WaveMoL.c

```c
void  WaveMoL_CalcRHS(CCTK_ARGUMENTS) {

  DECLARE_CCTK_ARGUMENTS;

  int i,j,k, index, istart, jstart, kstart, iend, jend, kend;
  CCTK_REAL dx,dy,dz, hdxi, hdyi, hdzi;

  /* Set up shorthands */
  dx = CCTK_DELTA_SPACE(0); dy = CCTK_DELTA_SPACE(1); dz = CCTK_DELTA_SPACE(2);

  hdxi = 0.5 / dx; hdyi = 0.5 / dy; hdzi = 0.5 / dz;

  istart = 1; jstart = 1; kstart = 1;

  iend = cctk_lsh[0]-1; jend = cctk_lsh[1]-1; kend = cctk_lsh[2]-1;

  /* Calculate the right hand sides. */
  for (k=0; k<cctk_lsh[2]; k++) {
    for (j=0; j<cctk_lsh[1]; j++) {
      for (i=0; i<cctk_lsh[0]; i++) {
        index = CCTK_GFINDEX3D(cctkGH,i,j,k);
        phirhs[index] = phit[index];
        phitrhs[index] = 0;
        phixrhs[index] = 0;
        phiyrhs[index] = 0;
        phizrhs[index] = 0;
```

More information

- https://einsteintoolkit.org/
- “An Introduction to the Einstein Toolkit” (Zilhão and Löffler 2013)

Final tips and remarks

- Check available thorns; one may already be doing what you are looking for (or may be easily adapted/extended)
- Copy provided parameter files and adapt from there
- Drawbacks: relying on 3rd party tools…
- Use bug tracking system for bugs, issues or wish list
- Email questions to users mailing list users@einsteintoolkit.org

Acknowledgments

- UIDB/04106/2020 https://doi.org/10.54499/UIDB/04106/2020
- UIDP/04106/2020 https://doi.org/10.54499/UIDP/04106/2020
- 2022.00721.CEECIND https://doi.org/10.54499/2022.00721.CEECIND/CP1720/CT0001
- 2022.04560.PTDC https://doi.org/10.54499/2022.04560.PTDC
- NewFunFiCO 101086251

fct Fundação para a Ciência e a Tecnologia

CIDMA universidade de aveiro

References

Cardoso, Vitor, Taishi Ikeda, Zhen Zhong, and Miguel Zilhão. 2022. "Piercing of a boson star by a black hole." *Phys. Rev. D* 106 (4): 044030. https://doi.org/10.1103/PhysRevD.106.044030.

Gourgoulhon, Eric. 2007. "3+1 Formalism and Bases of Numerical Relativity." https://arxiv.org/abs/gr-qc/0703035.

Zilhão, Miguel, and Frank Löffler. 2013. "An Introduction to the Einstein Toolkit." *Int.J.Mod.Phys.* A28: 1340014. https://doi.org/10.1142/S0217751X13400149.