Numerical relativity in AdS/CFT by **spNDSolve**, a Mathematica package for solving PDEs by the pseudospectral method

Jie Ren

The Hebrew University of Jerusalem

Wüzburg

2018.07.31

Outline

Introduction to numerical relativity in AdS/CFT

The pseudospectral method and **spNDSolve**

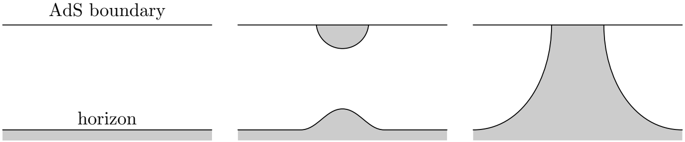
Some examples

Charged black funnel

Introduction

The absence of uniqueness theorems in AdS has following consequences

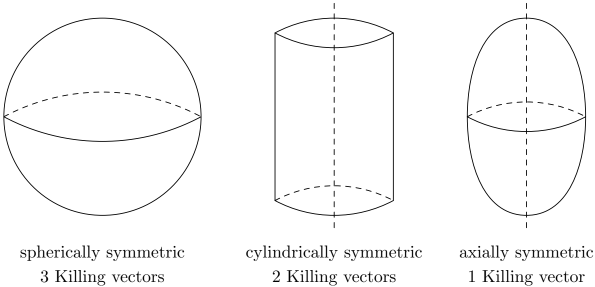
- Many matter fields can be added. Apply AdS/CFT to condensed matter physics.
-
  - The topology of the horizon is not necessarily a sphere.
  - The surface gravity of a horizon is not necessarily a constant.
  - The AdS boundary can be conformal to any spacetime $\mathcal{B}_d$.
  - e.g., Black droplets, black funnels, localized plasma balls



Solutions to Einstein's equation

$$R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R + \Lambda g_{\mu\nu} = 8\pi G T_{\mu\nu}$$

Symmetries in Spacetimes:



Einstein equations for stationary, axisymmetric spacetimes are completely integrable for $\Lambda = 0$, not for $\Lambda \neq 0$.

Solutions to Einstein equations

$$R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R + \Lambda g_{\mu\nu} = 8\pi G T_{\mu\nu}$$

Suppose (t, r, θ, ϕ) are spacetime coordinates, where ∂_t and ∂_ϕ are Killing vectors.

- When $\Lambda = 0$, the crossing term in the metric is only $dtd\phi$. However, when $\Lambda \neq 0$, $dtdr$ and $dtd\theta$ are possible.
- In the static case, Einstein equations are still not integrable when $\Lambda \neq 0$.
- Analytic solution exists in a very special case: the C-metric.

Conclusion: Numerical relativity in AdS/CFT is important.

Gauge fixing and the DeTurck method

$$g_{\mu\nu}: \qquad \frac{D(D+1)}{2} \text{ independent components}$$

$$R_{\mu\nu} + \frac{D-1}{L^2} g_{\mu\nu} = 0: \qquad \frac{D(D-1)}{2} \text{ independent equations}$$

$$R^H_{\mu\nu} + \frac{D-1}{L^2} g_{\mu\nu} = 0: \qquad \frac{D(D+1)}{2} \text{ independent equations}$$

$$R_{\mu\nu} \to R^H_{\mu\nu} = R_{\mu\nu} - \nabla_{(\mu}\xi_{\nu)},$$

where

$$\xi^\mu = g^{\lambda\sigma}[\Gamma^\mu_{\lambda\sigma}(g) - \bar{\Gamma}^\mu_{\lambda\sigma}(\bar{g})],$$

and $\bar{\Gamma}^\mu_{\lambda\sigma}(\bar{g})$ is the Levi-Civita connection associated with a reference metric $\bar{g}_{\mu\nu}$.

Conformal gauge v.s. DeTurck method

A stationary solution is a solution to a well-posed boundary value problem. [1510.02804]

Consider a codimension-2, static problem. In the conformal gauge, the metric anzatz has three unknown functions:

$$ds^2 = -A(r,x)dt^2 + B(r,x)(dr^2 + dx^2) + C(r,x)dy^2$$

However, there is still remaining gauge freedom, which is imposed as boundary conditions in this case.

By the DeTuck method, the metric ansatz must be closed under $r \to f_1(r,x)$ and $x \to f_2(r,x)$:

$$ds^2 = -Tdt^2 + Adr^2 + Bdx^2 + Sdy^2 + 2Fdrdx$$

There are five unknown functions of (r,x).

Holographic pair density waves (PDW)

$$\mathcal{L} = R + \frac{6}{L^2} - \frac{1}{2}(\partial\chi)^2 - \frac{Z_A}{4}F^2 - \frac{Z_B}{4}\tilde{F}^2 - \frac{Z_{AB}}{2}F\,\tilde{F} - \mathcal{K}(\chi)(\partial_\mu\theta - q_A A_\mu - q_B B_\mu)^2 - \frac{m^2}{2}\chi^2$$

The ansatz must be closed under coordinate transformations $z \to \tilde{z}(z,x)$ and $x \to \tilde{x}(z,x)$:

$$ds^2 = \frac{r_h^2}{L^2(1-z^2)^2}\Big[-F(z)Q_{tt}\,dt^2 + \frac{4z^2L^4Q_{zz}}{r_h^2F(z)}\,dz^2 + Q_{xx}(dx - 2z(1-z^2)^2Q_{xz}dz)^2 + Q_{yy}\,dy^2\Big]$$

$$A_t = \mu\,z^2\alpha, \quad B_t = z^2\beta, \quad \chi = (1-z^2)\phi,$$

where the eight functions $(\phi, \alpha, \beta, Q_{tt}, Q_{zz}, Q_{xx}, Q_{yy}, Q_{xz})$ are functions of z and x.

The pseudospectral method

- A function is vector in Hilbert space.
- A differential operator is a matrix.

In numerical calculations, we use a finite number of basis of Hilbert space to approximate the solution, so the differential operator is a finite-dimensional matrix.

- Construct the grid
- Calculate the differentiation matrix ($\frac{d}{dx}$, $\frac{d^2}{dx^2}$, *etc.*)
- Imposing boundary conditions by replacing rows corresponding to the boundaries
- Solve a linear system of equations
- Nonlinear equations = linear equations + iterations

spNDSolve

Goal: Develop a package that solves any stationary PDEs by the pseudospectral method (if applicable) by the same procedure. Only use equations and boundary conditions as inputs, just like *NDSolve*.

The basic idea of spNDSolve is as follows:

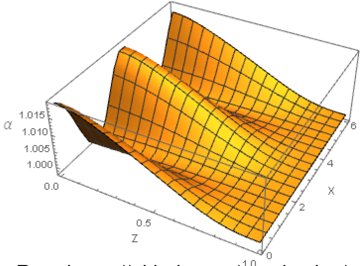
- At the first (symbolic) stage, expressions are automatically generated in *HoldForm*.
- At the second (numerical) stage, spNDSolve will assemble these expressions and execute them by *ReleaseHold*.

spNDSolve

Expressions generated at the first stage:

```
In[12]:= replrule

Out[12]= {g1[z, x] → g1, g1^(1,0)[z, x] → g1z, g1^(0,1)[z, x] → g1x, g1^(1,1)[z, x] → g1zx, g1^(2,0)[z, x] → g1zz,
  g1^(0,2)[z, x] → g1xx, g2[z, x] → g2, g2^(1,0)[z, x] → g2z, g2^(0,1)[z, x] → g2x, g2^(1,1)[z, x] → g2zx,
  g2^(2,0)[z, x] → g2zz, g2^(0,2)[z, x] → g2xx, g3[z, x] → g3, g3^(1,0)[z, x] → g3z, g3^(0,1)[z, x] → g3x,
  g3^(1,1)[z, x] → g3zx, g3^(2,0)[z, x] → g3zz, g3^(0,2)[z, x] → g3xx, g4[z, x] → g4, g4^(1,0)[z, x] → g4z,
  g4^(0,1)[z, x] → g4x, g4^(1,1)[z, x] → g4zx, g4^(2,0)[z, x] → g4zz, g4^(0,2)[z, x] → g4xx,
  g5[z, x] → g5, g5^(1,0)[z, x] → g5z, g5^(0,1)[z, x] → g5x, g5^(1,1)[z, x] → g5zx, g5^(2,0)[z, x] → g5zz,
  g5^(0,2)[z, x] → g5xx, At[z, x] → At, At^(1,0)[z, x] → Atz, At^(0,1)[z, x] → Atx, At^(1,1)[z, x] → Atzx,
  At^(2,0)[z, x] → Atzz, At^(0,2)[z, x] → Atxx, Bt[z, x] → Bt, Bt^(1,0)[z, x] → Btz, Bt^(0,1)[z, x] → Btx,
  Bt^(1,1)[z, x] → Btzx, Bt^(2,0)[z, x] → Btzz, Bt^(0,2)[z, x] → Btxx, phi[z, x] → phi, phi^(1,0)[z, x] → phiz,
  phi^(0,1)[z, x] → phix, phi^(1,1)[z, x] → phizx, phi^(2,0)[z, x] → phizz, phi^(0,2)[z, x] → phixx}

In[13]:= calcfderiv

Out[13]= {g1z = dz.g1, g1x = dx.g1, g1zx = dzx.g1, g1zz = dzz.g1, g1xx = dxx.g1, g2z = dz.g2,
  g2x = dx.g2, g2zx = dzx.g2, g2zz = dzz.g2, g2xx = dxx.g2, g3z = dz.g3, g3x = dx.g3, g3zx = dzx.g3,
  g3zz = dzz.g3, g3xx = dxx.g3, g4z = dz.g4, g4x = dx.g4, g4zx = dzx.g4, g4zz = dzz.g4, g4xx = dxx.g4,
  g5z = dz.g5, g5x = dx.g5, g5zx = dzx.g5, g5zz = dzz.g5, g5xx = dxx.g5, Atz = dz.At, Atx = dx.At,
  Atzx = dzx.At, Atzz = dzz.At, Atxx = dxx.At, Btz = dz.Bt, Btx = dx.Bt, Btzx = dzx.Bt, Btzz = dzz.Bt,
  Btxx = dxx.Bt, phiz = dz.phi, phix = dx.phi, phizx = dzx.phi, phizz = dzz.phi, phixx = dxx.phi}
```

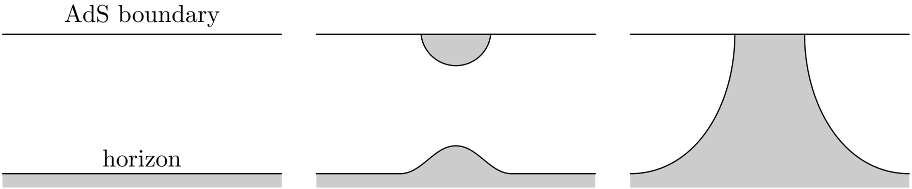
```
In[1]:= << spNDSolve.m

setdir;
flist = {g1[z, x], g2[z, x], g3[z, x], g4[z, x], g5[z, x], At[z, x], Bt[z, x], phi[z, x]};
brylist1 = {g1[z, x] - g2[z, x], D[g2[z, x], z], D[g3[z, x], z], D[g4[z, x], z], D[g5[z, x], z], D[At[z, x], z], D[Bt[z, x], z], D[ph
brylist2 = {g1[z, x] - 1, g2[z, x] - 1, g3[z, x] - 1, g4[z, x] - 1, g5[z, x], At[z, x] - 1, Bt[z, x], phi[z, x]};
brylist3 = {D[g1[z, x], x], D[g2[z, x], x], D[g3[z, x], x], D[g4[z, x], x], g5[z, x], D[At[z, x], x], D[Bt[z, x], x], D[phi[z, x], x]
calcbryind := ({bryind1, bryind2} = bryIndex[{1, 2}]; bryind3 = bryIndex[3, {5}];)
bryLinearize[{flist}, {{bryind1, brylist1}, {bryind2, brylist2}, {bryind3, brylist3}}];
get[eqlistmat, eqlist, xi2, z → zaux12];
calcfref := (g1 = one; g2 = one; g3 = one; g4 = one; g5 = 0 one; At = one; Bt = a0 (1 - z^2)^3 Cos[x]; phi = -a0 (1 - z^2)^3 Cos[x];)
output := (nxi2 = Norm@arrayDrop[xi2, {bryind1, bryind2}]; PrintTemporary["ξ^2=", nxi2]; {nxi2, rh, solsave});
Off[LinearSolve::luc];

In[14]:= Dynamic[changes]

Out[14]= {0.618555, 0.100396, 0.00170137, 4.54892×10^-7, 1.0657×10^-10}

In[16]:= spNDSolve[{z, x}, {(# + 1)/2 &, # &}, {20, {20, "periodic"}}, {a, 4}, {c, -2.34}, {rh, 0.1528}, {a0, 0.04}, {qB, 0}, eqlistmatQ → True]

Saved to outputs

In[25]:= ListPlot3D[{z, x, At}^T, AxesLabel → {Style["z", 14], Style["x", 14], Style["α ", 14]}, PlotRange → All]

Out[25]=
```



Charged black funnels

CFT at finite charge density in a black hole background.

$$ds^2 = \frac{L^2}{xyQH^2}\left[-x(1-y)QMdt^2 + \frac{xQH^2Ady^2}{4y(1-y)M} + \frac{r_0^2B[dx + x(1-x)^2Fdy]^2}{x(1-x)^4} + \frac{r_0^2S}{(1-x)^2}d\Omega_2^2\right]$$



Features of spNDSolve

- Grid construction
- Method of continuation
- Two-domain decomposition
- Arbitrary precision

To be improved:

- Iterative solver

Summary

spNDSolve: A Mathematica package for solving PDEs by the pseudospectral method

- Introduction to numerical relativity in AdS/CFT
  - Non-existence of uniqueness theorems
  - DeTurck method
  - Pseudospectral method
- The pseudospectral method and spNDSolve
  - Grid construction
  - Built-in continuation method
  - Two-domain decomposition
  - Arbitrary precision
- Some examples
  - Holographic superconductors
  - Charged black funnel

Summary

Thank you!

spNDSolve: A Mathematica package for solving PDEs by the pseudospectral method

- Introduction to numerical relativity in AdS/CFT
  - Non-existence of uniqueness theorems
  - DeTurck method
  - Pseudospectral method
- The pseudospectral method and spNDSolve
  - Grid construction
  - Built-in continuation method
  - Two-domain decomposition
  - Arbitrary precision
- Some examples
  - Holographic superconductors
  - Charged black funnel