GR computations with the Python-based free computer algebra system SageMath

Éric Gourgoulhon[1], Florentin Jaffredo[2], Marco Mancini[1], Travis Scrimshaw[3], Karim Van Aelst[1]

[1] Laboratoire Univers et Théories, Observatoire de Paris, CNRS, Université PSL, Université Paris Diderot, Meudon, France

[2] École Polytechnique, Palaiseau, France

[3] School of Mathematics and Physics, University of Queensland, St. Lucia, Australia

22nd International Conference on General Relativity and Gravitation

Valencia, Spain

11 July 2019

SageMath in a few words

SageMath (*nickname:* Sage) is a **free open-source** computer algebra system

SageMath in a few words

SageMath (*nickname:* Sage) is a **free open-source** computer algebra system

SageMath is free (GPL v2)

Freedom means

1. everybody can use it, by downloading the software from `http://sagemath.org`
2. everybody can examine the source code and improve it

SageMath in a few words

SageMath (*nickname:* Sage) is a **free open-source** computer algebra system

SageMath is free (GPL v2)

Freedom means

1. everybody can use it, by downloading the software from http://sagemath.org
2. everybody can examine the source code and improve it

SageMath is based on Python

- no need to learn any specific syntax to use it
- Python is a very powerful *object oriented language*, with a neat syntax
- SageMath benefits from the Python ecosystem (e.g. Jupyter notebook)

SageMath in a few words

SageMath (*nickname:* Sage) is a **free open-source** computer algebra system

SageMath is free (GPL v2)

Freedom means

1. everybody can use it, by downloading the software from `http://sagemath.org`
2. everybody can examine the source code and improve it

SageMath is based on Python

- no need to learn any specific syntax to use it
- Python is a very powerful *object oriented language*, with a neat syntax
- SageMath benefits from the Python ecosystem (e.g. Jupyter notebook)

SageMath is developed by an enthusiastic community

- mostly composed of mathematicians
- welcoming newcomers

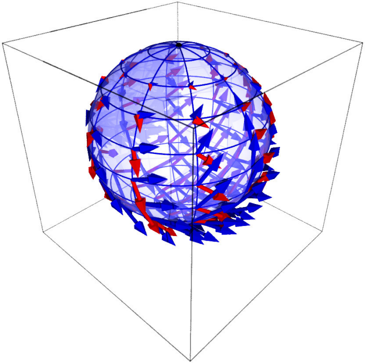
Tensor calculus with SageMath

`SageMath` is well developed in group theory and graph theory

Tensor calculus with SageMath

SageMath is well developed in group theory and graph theory

SageManifolds project: extends SageMath towards **differential geometry** and **tensor calculus**



Stereographic-coordinate frame on $\mathbb{S}^2$

- https://sagemanifolds.obspm.fr
- fully included in SageMath
- $\sim$ 15 contributors (developers and reviewers) cf. https://sagemanifolds.obspm.fr/authors.html
- dedicated mailing list
- help: https://ask.sagemath.org

Everybody is very welcome to contribute
$\Longrightarrow$ visit https://sagemanifolds.obspm.fr/contrib.html

Current status

Already present (SageMath 8.8):

- differentiable manifolds: tangent spaces, vector frames, tensor fields, curves, pullback and pushforward operators, submanifolds
- standard tensor calculus (tensor product, contraction, symmetrization, etc.), even on non-parallelizable manifolds, and with all monoterm tensor symmetries taken into account
- Lie derivatives of tensor fields
- differential forms: exterior and interior products, exterior derivative, Hodge duality
- multivector fields: exterior and interior products, Schouten-Nijenhuis bracket
- affine connections (curvature, torsion)
- pseudo-Riemannian metrics
- computation of geodesics (numerical integration)

Current status

Already present *(cont'd)*:

- some plotting capabilities (charts, points, curves, vector fields)
- parallelization (on tensor components) of CPU demanding computations
- extrinsic geometry of pseudo-Riemannian submanifolds
- tensor series expansions

Future prospects:

- more symbolic backends (Giac, FriCAS, ...)
- more graphical outputs
- symplectic forms, fibre bundles, spinors, integrals on submanifolds, variational calculus, etc.
- connection with numerical relativity: use SageMath to explore numerically-generated spacetimes

A short example:

Near-horizon geometry of the extremal Kerr black hole

This notebook derives the near-horizon geometry of the extremal (i.e. maximally spinning) Kerr black hole. It is based on SageMath tools developed through the [SageManifolds project](#).

First we set up the notebook to display maths using LaTeX rendering and to perform computations in parallel on 8 threads:

In [1]:
```
%display latex
Parallelism().set(nproc=8)
```

Spacetime manifold

We declare the Kerr spacetime (or more precisely the part of it covered by Boyer-Lindquist coordinates) as a 4-dimensional Lorentzian manifold $\mathcal{M}$:

In [2]:
```
M = Manifold(4, 'M', latex_name=r'\mathcal{M}', structure='Lorentzian')
print(M)
```

```
4-dimensional Lorentzian manifold M
```

We then introduce the standard **Boyer-Lindquist coordinates** (t, r, θ, ϕ) as a chart `BL` (for *Boyer-Lindquist*) on $\mathcal{M}$:

In [3]:
```
BL.<t,r,th,ph> = M.chart(r"t r th:(0,pi):\theta ph:(0,2*pi):periodic:\phi")
print(BL); BL
```

```
Chart (M, (t, r, th, ph))
```

Out[3]: $(\mathcal{M}, (t, r, \theta, \phi))$

Metric tensor of the extremal Kerr spacetime

The metric is set by its components in the coordinate frame associated with Boyer-Lindquist coordinates, which is the current manifold's default frame:

In [4]:
```
m = var('m', domain='real')
a = m  # extremal Kerr
rho2 = r^2 + (a*cos(th))^2
Delta = r^2 -2*m*r + a^2
g = M.metric()
g[0,0] = -(1-2*m*r/rho2)
g[0,3] = -2*a*m*r*sin(th)^2/rho2
g[1,1], g[2,2] = rho2/Delta, rho2
g[3,3] = (r^2+a^2+2*m*r*(a*sin(th))^2/rho2)*sin(th)^2
g.display()
```

Out[4]:

$$g = \left(\frac{2\,mr}{m^2\cos(\theta)^2 + r^2} - 1\right) \mathrm{d}t \otimes \mathrm{d}t + \left(-\frac{2\,m^2 r \sin(\theta)^2}{m^2\cos(\theta)^2 + r^2}\right) \mathrm{d}t \otimes \mathrm{d}\phi + \left(\frac{m^2\cos(\theta)^2 + r^2}{m^2 - 2\,mr + r^2}\right) \mathrm{d}r \otimes \mathrm{d}r + \left(m^2\cos(\theta)^2 + r^2\right) \mathrm{d}\theta \otimes \mathrm{d}\theta + \left(-\frac{2\,m^2 r \sin(\theta)^2}{m^2\cos(\theta)^2 + r^2}\right) \mathrm{d}\phi \otimes \mathrm{d}t + \left(\frac{2\,m^3 r \sin(\theta)^2}{m^2\cos(\theta)^2 + r^2} + m^2 + r^2\right) \sin(\theta)^2 \mathrm{d}\phi \otimes \mathrm{d}\phi$$

Check that we are dealing with a solution of the vacuum Einstein equation:

In [5]:
```
g.ricci().display()
```

Out[5]: $\mathrm{Ric}(g) = 0$

Near-horizon coordinates

Let us introduce the chart NH of the near-horizon coordinates $(\bar{t}, \bar{r}, \theta, \bar{\phi})$:

In [6]:
```
NH.<tb,rb,th,phb> = M.chart(r"tb:\bar{t} rb:\bar{r} th:(0,pi):\theta phb:(0,2*pi):periodic:\
print(NH)
NH
```

```
Chart (M, (tb, rb, th, phb))
```

Out[6]: $(\mathcal{M}, (\bar{t}, \bar{r}, \theta, \bar{\phi}))$

Following J. Bardeen and G. T. Horowitz, Phys. Rev. D **60**, 104030 (1999), the near-horizon coordinates $(\bar{t}, \bar{r}, \theta, \bar{\phi})$ are related to the Boyer-Lindquist coordinates by

$$\bar{t} = \epsilon t, \quad \bar{r} = \frac{r-m}{\epsilon}, \quad \theta = \theta, \quad \bar{\phi} = \phi - \frac{t}{2m},$$

where ϵ is a constant parameter. The horizon of the extremal Kerr black hole is located at $r = m$, which corresponds to $\bar{r} = 0$.

We implement the above relations as a transition map from the chart BL to the chart NH :

In [7]:
```
eps = var('eps', latex_name=r'\epsilon')
BL_to_NH = BL.transition_map(NH, [eps*t, (r-m)/eps, th, ph - t/(2*m)])
BL_to_NH.display()
```

Out[7]:

$$\left\{\begin{array}{lcl} \bar{t} & = & \epsilon t \\ \bar{r} & = & -\frac{m-r}{\epsilon} \\ \theta & = & \theta \\ \bar{\phi} & = & \phi - \frac{t}{2\,m} \end{array}\right.$$

The inverse relation is

In [8]:
```
BL_to_NH.inverse().display()
```

Out[8]:

$$\left\{\begin{array}{lcl} t & = & \frac{\bar{t}}{\epsilon} \\ r & = & \epsilon \bar{r} + m \\ \theta & = & \theta \\ \phi & = & \frac{2\,\epsilon m \bar{\phi} + \bar{t}}{2\,\epsilon m} \end{array}\right.$$

The metric components with respect the coordinates $(\bar{t}, \bar{r}, \theta, \bar{\phi})$ are computed by passing the chart `NH` to the method `display()`:

In [9]:
```
g.display(NH)
```

Out[9]:

$$g = \left(-\frac{m^2\bar{r}^2\cos(\theta)^4 - \epsilon^2\bar{r}^4 - 4\,\epsilon m\bar{r}^3 - 3\,m^2\bar{r}^2 + \left(\epsilon^2\bar{r}^4 + 4\,\epsilon m\bar{r}^3 + 6\,m^2\bar{r}^2\right)\cos(\theta)^2}{4\left(\epsilon^2 m^2\bar{r}^2 + m^4\cos(\theta)^2 + 2\,\epsilon m^3\bar{r} + m^4\right)}\right) \mathrm{d}\bar{t}\otimes\mathrm{d}\bar{t}$$
$$+ \left(-\frac{\epsilon m^2\bar{r}^2\sin(\theta)^4 - \left(\epsilon^3\bar{r}^4 + 4\,\epsilon^2 m\bar{r}^3 + 8\,\epsilon m^2\bar{r}^2 + 4\,m^3\bar{r}\right)\sin(\theta)^2}{2\left(\epsilon^2 m\bar{r}^2 + m^3\cos(\theta)^2 + 2\,\epsilon m^2\bar{r} + m^3\right)}\right) \mathrm{d}\bar{t}\otimes\mathrm{d}\bar{\phi}$$
$$+ \left(\frac{\epsilon^2\bar{r}^2 + m^2\cos(\theta)^2 + 2\,\epsilon m\bar{r} + m^2}{\bar{r}^2}\right) \mathrm{d}\bar{r}\otimes\mathrm{d}\bar{r} + \left(\epsilon^2\bar{r}^2 + m^2\cos(\theta)^2 + 2\,\epsilon m\bar{r} + m^2\right) \mathrm{d}\theta\otimes\mathrm{d}\theta$$
$$+ \left(-\frac{\epsilon m^2\bar{r}^2\sin(\theta)^4 - \left(\epsilon^3\bar{r}^4 + 4\,\epsilon^2 m\bar{r}^3 + 8\,\epsilon m^2\bar{r}^2 + 4\,m^3\bar{r}\right)\sin(\theta)^2}{2\left(\epsilon^2 m\bar{r}^2 + m^3\cos(\theta)^2 + 2\,\epsilon m^2\bar{r} + m^3\right)}\right) \mathrm{d}\bar{\phi}\otimes\mathrm{d}\bar{t}$$
$$+ \left(-\frac{\epsilon^2 m^2\bar{r}^2\sin(\theta)^4 - \left(\epsilon^4\bar{r}^4 + 4\,\epsilon^3 m\bar{r}^3 + 8\,\epsilon^2 m^2\bar{r}^2 + 8\,\epsilon m^3\bar{r} + 4\,m^4\right)\sin(\theta)^2}{\epsilon^2\bar{r}^2 + m^2\cos(\theta)^2 + 2\,\epsilon m\bar{r} + m^2}\right) \mathrm{d}\bar{\phi}\otimes\mathrm{d}\bar{\phi}$$

From now on, we use the near-horizon coordinates as the default ones on the spacetime manifold:

In [10]:
```
M.set_default_chart(NH)
M.set_default_frame(NH.frame())
```

The near-horizon metric h as the limit $\epsilon \to 0$ of the Kerr metric g

Let us define the *near-horizon metric* as the metric h on $\mathcal{M}$ that is the limit $\epsilon \to 0$ of the Kerr metric g. The limit is taken by asking for a series expansion of g with respect to ϵ up to the 0-th order (i.e. keeping only ϵ^0 terms). This is acheived via the method `truncate`:

In [11]:
```
h = M.lorentzian_metric('h')
h.set(g.truncate(eps, 0))
h.display()
```

Out[11]:

$$h = \left(-\frac{\bar{r}^2 \cos(\theta)^4 + 6\,\bar{r}^2 \cos(\theta)^2 - 3\,\bar{r}^2}{4\left(m^2 \cos(\theta)^2 + m^2\right)} \right) \mathrm{d}\bar{t} \otimes \mathrm{d}\bar{t} + \left(\frac{2\,\bar{r} \sin(\theta)^2}{\cos(\theta)^2 + 1} \right) \mathrm{d}\bar{t} \otimes \mathrm{d}\bar{\phi} + \left(\frac{m^2 \cos(\theta)^2 + m^2}{\bar{r}^2} \right) \mathrm{d}\bar{r} \otimes \mathrm{d}\bar{r} + \left(m^2 \cos(\theta)^2 + m^2 \right) \mathrm{d}\theta \otimes \mathrm{d}\theta + \left(\frac{2\,\bar{r} \sin(\theta)^2}{\cos(\theta)^2 + 1} \right) \mathrm{d}\bar{\phi} \otimes \mathrm{d}\bar{t} + \left(\frac{4\,m^2 \sin(\theta)^2}{\cos(\theta)^2 + 1} \right) \mathrm{d}\bar{\phi} \otimes \mathrm{d}\bar{\phi}$$

We note that the metric h is not asymptotically flat.

Killing vectors of the near-horizon geometry

Let us first consider the vector field $\eta := \frac{\partial}{\partial\bar{\phi}}$:

In [12]:
```
eta = M.vector_field(0, 0, 0, 1, name='eta', latex_name=r'\eta')
eta.display()
```

Out[12]:

$$\eta = \frac{\partial}{\partial\bar{\phi}}$$

It is a Killing vector of the near-horizon metric, since the Lie derivative of h along η vanishes:

In [13]:
```
h.lie_derivative(eta).display()
```

Out[13]: 0

This is not surprising since the components of h are independent from $\bar{\phi}$.

Similarly, we can check that $\xi_1 := \frac{\partial}{\partial\bar{t}}$ is a Killing vector of h, reflecting the independence of the components of h from $\bar{t}$:

In [14]:
```
xi1 = M.vector_field(1, 0, 0, 0, name='xi2', latex_name=r'\xi_{1}')
xi1.display()
```

Out[14]:

$$\xi_1 = \frac{\partial}{\partial\bar{t}}$$

In [15]:
```
h.lie_derivative(xi1).display()
```

Out[15]: 0

The above two Killing vectors correspond respectively to the **axisymmetry** and the **pseudo-stationarity** of the Kerr metric. A third symmetry, which is not present in the original Kerr metric, is the invariance under the **scaling** $(\bar{t}, \bar{r}) \mapsto (\alpha\bar{t}, \bar{r}/\alpha)$, as it is clear on the metric components in Out[11]. The corresponding Killing vector is

In [16]:
```
xi2 = M.vector_field(tb, -rb, 0, 0, name='xi2', latex_name=r'\xi_{2}')
xi2.display()
```

Out[16]: $\xi_2 = \bar{t}\frac{\partial}{\partial\bar{t}} - \bar{r}\frac{\partial}{\partial\bar{r}}$

In [17]:
```
h.lie_derivative(xi2).display()
```

Out[17]: 0

Finally, a fourth Killing vector is

In [18]:
```
xi3 = M.vector_field(tb^2/2 + 2*m^4/rb^2, -tb*rb, 0, -2*m^2/rb,
                     name='xi3', latex_name=r'\xi_{3}')
xi3.display()
```

Out[18]: $\xi_3 = \left(\frac{2\,m^4}{\bar{r}^2} + \frac{1}{2}\,\bar{t}^2\right)\frac{\partial}{\partial\bar{t}} - \bar{r}\bar{t}\frac{\partial}{\partial\bar{r}} - \frac{2\,m^2}{\bar{r}}\frac{\partial}{\partial\bar{\phi}}$

In [19]:
```
h.lie_derivative(xi3).display()
```

Out[19]: 0

Symmetry group

We have four independent Killing vectors, η, ξ_1, ξ_2 and ξ_3, which implies that the symmetry group of the near-horizon geometry is a 4-dimensional Lie group G. Let us determine G by investigating the **structure constants** of the basis $(\eta, \xi_1, \xi_2, \xi_3)$ of the Lie algebra of G. First of all, we notice that η commutes with the other Killing vectors:

In [20]:
```
for xi in [xi1, xi2, xi3]:
    show(eta.bracket(xi).display())
```

$[\eta, \xi_1] = 0$

$[\eta, \xi_2] = 0$

$[\eta, \xi_3] = 0$

Since η generates the rotation group $\mathrm{SO}(2) = \mathrm{U}(1)$, we may write that $G = \mathrm{U}(1) \times G_3$, where G_3 is a 3-dimensional Lie group, whose generators are (ξ_1, ξ_2, ξ_3). Let us determine the structure constants of these three vectors. We have

In [21]:
```
xi1.bracket(xi2).display()
```

Out[21]:

$$[\xi_1, \xi_2] = \frac{\partial}{\partial \bar{t}}$$

In [22]:
```
xi1.bracket(xi3).display()
```

Out[22]:

$$[\xi_1, \xi_3] = \bar{t}\frac{\partial}{\partial \bar{t}} - \bar{r}\frac{\partial}{\partial \bar{r}}$$

In [23]:
```
xi2.bracket(xi3).display()
```

Out[23]:

$$[\xi_2, \xi_3] = \left(\frac{4\,m^4 + \bar{r}^2\bar{t}^2}{2\,\bar{r}^2}\right)\frac{\partial}{\partial \bar{t}} - \bar{r}\bar{t}\frac{\partial}{\partial \bar{r}} - \frac{2\,m^2}{\bar{r}}\frac{\partial}{\partial \bar{\phi}}$$

To summarize, we have

In [24]:
```
all([xi1.bracket(xi2) == xi1,
     xi1.bracket(xi3) == xi2,
     xi2.bracket(xi3) == xi3])
```

Out[24]: True

To recognize a standard Lie algebra, let us perform a slight change of basis:

In [25]:
```
vE = -sqrt(2)*xi3
vF = sqrt(2)*xi1
vH = 2*xi2
```

We have then the following commutation relations:

In [26]:
```
all([vE.bracket(vF) == vH,
     vH.bracket(vE) == 2*vE,
     vH.bracket(vF) == -2*vF])
```

Out[26]: True

We recognize the Lie algebra $\mathfrak{sl}(2,\mathbb{R})$. Indeed, we have

In [27]:
```
sl2 = lie_algebras.sl(RR, 2)
E,F,H = sl2.gens()
all([E.bracket(F) == H,
     H.bracket(E) == 2*E,
     H.bracket(F) == -2*F])
```

Out[27]: True

Hence, we have

$$\mathrm{Lie}(G_3) = \mathfrak{sl}(2,\mathbb{R}).$$

At this stage, G_3 could be $\mathrm{SL}(2,\mathbb{R})$, $\mathrm{PSL}(2,\mathbb{R})$ or $\overline{\mathrm{SL}(2,\mathbb{R})}$ (the universal covering group of $\mathrm{SL}(2,\mathbb{R})$). One can show that actually $G_3 = \mathrm{SL}(2,\mathbb{R})$. We conclude that the full isometry group of the near-horizon geometry is

$$G = \mathrm{U}(1) \times \mathrm{SL}(2,\mathbb{R}).$$

The full notebook is available at
https://nbviewer.jupyter.org/github/sagemanifolds/SageManifolds/blob/master/Notebooks/SM_extremal_Kerr_near_horizon.ipynb

Hence, we have

$$\mathrm{Lie}(G_3) = \mathfrak{sl}(2, \mathbb{R}).$$

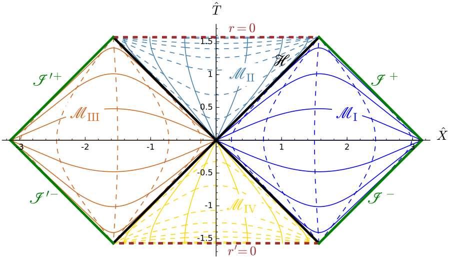
At this stage, G_3 could be $\mathrm{SL}(2, \mathbb{R})$, $\mathrm{PSL}(2, \mathbb{R})$ or $\overline{\mathrm{SL}(2, \mathbb{R})}$ (the universal covering group of $\mathrm{SL}(2, \mathbb{R})$). One can show that actually $G_3 = \mathrm{SL}(2, \mathbb{R})$. We conclude that the full isometry group of the near-horizon geometry is

$$G = \mathrm{U}(1) \times \mathrm{SL}(2, \mathbb{R}).$$

The full notebook is available at
https://nbviewer.jupyter.org/github/sagemanifolds/SageManifolds/blob/master/Notebooks/SM_extremal_Kerr_near_horizon.ipynb

Many other examples are posted at
https://sagemanifolds.obspm.fr/examples.html

Carter-Penrose diagram computed and drawn with SageMath →



Want to join the project or simply to stay tuned?

visit https://sagemanifolds.obspm.fr/
(download, documentation, example notebooks, mailing list)