Basic Computations in General Relativity with SageManifolds

A. Megías and J. J. Ruiz-Lorenzo
Departamento de Física, Universidad de Extremadura, E-06006 Badajoz, Spain

March 13, 2021

1 Spacetime Geometry Computations

This notebook is based on the original notebook: Schwatzschild spacetime.

In this notebook we will use the Schwarzschild spacetime to show the essential computations that we will need during the course. The corresponding tools have been developed within the SageManifolds project.

For a given metric $g_{\mu\nu}$ we can compute:

- The inverse metric: $g^{\mu\nu}$.
- Christoffel Symbols: ${\Gamma^\lambda}_{\mu\nu} = \frac{1}{2} g^{\lambda\sigma} \left(\partial_\mu g_{\sigma\nu} + \partial_\nu g_{\sigma\mu} - \partial_\sigma g_{\mu\nu} \right)$.
- Riemann tensor: ${R^\lambda}_{\mu\nu\sigma} = \partial_\nu {\Gamma^\lambda}_{\mu\sigma} - \partial_\sigma {\Gamma^\lambda}_{\mu\nu} + \Gamma^\eta_{\mu\sigma} \Gamma^\lambda_{\eta\nu} - \Gamma^\eta_{\mu\nu} \Gamma^\lambda_{\eta\sigma}$.
- Ricci tensor: $R_{\mu\nu} = {R^\lambda}_{\mu\lambda\nu}$.
- Scalar curvature: $R = g^{\mu\nu} R_{\mu\nu}$.
- Einstein tensor: $G_{\mu\nu} = R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R$.

Although the notebook is created for 1+3 dimensions it can be generalized to whatever 1+n dimensional Lorentzian metric.

A more advanced notebook about Schwarzschild spacetime, involving many coordinate charts, more tensor calculus and graphical outputs, is available here.

Click here to download the original notebook file (ipynb format). To run it, you must start SageMath with the Jupyter interface, via the command `sage -n jupyter`.

[1]:
```
version() #SageMath version
%display latex #To display LaTeX expressions in some outputs
```

1.1 Spacetime manifold

We define our manifold (spacetime). The method `Manifold()` must receive the following arguments: $1+n$ dimension of the spacetime; name of the manifold; `structure = 'Lorentzian'` to work with a Lorentzian manifold.

[2]:
```
n=3 #space dimensions
M = Manifold(1+n, 'M', structure='Lorentzian')
print(M)
```

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

1.2 List of coordinates

We must define our coordinates via the method `chart()` applied to the object `M` (our manifold). Note that the argument of `chart()` is a raw string (hence the prefix `r` in front of it), which defines the range of each coordinate, if different from $(-\infty, +\infty)$, as well as its LaTeX symbol, if different from the Python symbol to denote the coordinate. The Python variables for each coordinate are declared within the `<...>` operator on the left-hand side of the identity, `X` denoting the Python variable chosen for the coordinate chart.

As an example, the standard **Schwarzschild coordinates** are introduced. The coordinates are the following:

$$t, \quad r \in (0, +\infty), \quad \theta \in (0, \pi), \quad \phi \in (0, 2\pi).$$

[3]:
```
X.<t,r,th,ph> = M.chart(r"t r:(0,+oo) th:(0,pi):\theta ph:(0,2*pi):\phi")
X
```

[3]: $(M, (t, r, \theta, \phi))$

[4]:
```
X[:]
```

[4]: (t, r, θ, ϕ)

The coordinates follows the same indexing: $X^0 = t$, $X^1 = r$, $X^2 = \theta$, $X^3 = \phi$.

[5]:
```
X[0],X[1],X[2],X[3]
```

[5]: (t, r, θ, ϕ)

1.3 Metric tensor $g_{\mu\nu}$.

We introduce first the mass parameter m as a symbolic positive variable, via the function `var()`:

[6]:
```
m = var('m')
assume(m>=0)
```

The metric tensor of the Lorentzian manifold `M` is returned by the method `metric()`; we initialize its components in the chart `X`, which is the default (unique) chart on `M`:

[7]:
```
g = M.metric()
g[0,0] = -(1-2*m/r)
g[1,1] = 1/(1-2*m/r)
g[2,2] = r^2
g[3,3] = (r*sin(th))^2
```

```
g.display()
```

[7]:
$$g = \left(\frac{2\,m}{r} - 1\right) \mathrm{d}t \otimes \mathrm{d}t + \left(-\frac{1}{\frac{2\,m}{r} - 1}\right) \mathrm{d}r \otimes \mathrm{d}r + r^2 \mathrm{d}\theta \otimes \mathrm{d}\theta + r^2 \sin\left(\theta\right)^2 \mathrm{d}\phi \otimes \mathrm{d}\phi$$

To display the metric as a matrix:

[8]:
```
g[:]
```

[8]:
$$\left(\begin{array}{rrrr} \frac{2\,m}{r} - 1 & 0 & 0 & 0 \\ 0 & -\frac{1}{\frac{2\,m}{r} - 1} & 0 & 0 \\ 0 & 0 & r^2 & 0 \\ 0 & 0 & 0 & r^2 \sin\left(\theta\right)^2 \end{array}\right)$$

In order to access to a the component of the metric with components (μ, ν) we would write: `g[mu,nu]`. Where `mu` and `nu` are integer variables such that $\mu, \nu \in \{0, \ldots, n\}$, where $\{t, r, \theta, \phi\} \equiv \{0, 1, 2, 3\}$ for our case. Here, we display the component g_{tt}.

[9]:
```
g[0,0]
```

[9]:
$$\frac{2\,m}{r} - 1$$

The inverse metric can be computed via `g.inverse()`.

[10]:
```
ginv=g.inverse(); ginv
```

[10]:
$$g^{-1}$$

[11]:
```
ginv.display()
```

[11]:
$$g^{-1} = \left(\frac{r}{2\,m - r}\right) \frac{\partial}{\partial t} \otimes \frac{\partial}{\partial t} + \left(-\frac{2\,m - r}{r}\right) \frac{\partial}{\partial r} \otimes \frac{\partial}{\partial r} + \frac{1}{r^2} \frac{\partial}{\partial \theta} \otimes \frac{\partial}{\partial \theta} + \frac{1}{r^2 \sin\left(\theta\right)^2} \frac{\partial}{\partial \phi} \otimes \frac{\partial}{\partial \phi}$$

[12]:
```
ginv[:]
```

[12]:
$$\left(\begin{array}{rrrr} \frac{r}{2\,m - r} & 0 & 0 & 0 \\ 0 & -\frac{2\,m - r}{r} & 0 & 0 \\ 0 & 0 & \frac{1}{r^2} & 0 \\ 0 & 0 & 0 & \frac{1}{r^2 \sin\left(\theta\right)^2} \end{array}\right)$$

If we multiply both matrices, we should get the $(1 + n) \times (1 + n)$ identity matrix

[13]:
```
delta = g['_{ab}']*ginv['^{bc}']
```

[14]:
```
delta[:]
```

[14]:
$$\left(\begin{array}{rrrr} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right)$$

1.4 Christoffel symbols $\Gamma^{\lambda}{}_{\mu\nu}$.

The Christoffel symbols of g with respect to the given coordinates are printed by the method `christoffel_symbols_display()` applied to the metric object `g`. By default, only the nonzero symbols and the nonredundant ones (taking into account the symmetry of the last two indices) are displayed. Type `g.christoffel_symbols_display?` to see all possible options.

[15]:
```
g.christoffel_symbols_display()
```

[15]:
$$\begin{array}{lcl} \Gamma_{ \, t \, r }^{ \, t   } & = & -\frac{m}{2 \, m r - r^{2}} \\ \Gamma_{ \, t \, t }^{ \, r   } & = & -\frac{2 \, m^{2} - m r}{r^{3}} \\ \Gamma_{ \, r \, r }^{ \, r   } & = & \frac{m}{2 \, m r - r^{2}} \\ \Gamma_{ \, \theta \, \theta }^{ \, r   } & = & 2 \, m - r \\ \Gamma_{ \, \phi \, \phi }^{ \, r   } & = & {\left(2 \, m - r\right)} \sin\left(\theta\right)^{2} \\ \Gamma_{ \, r \, \theta }^{ \, \theta   } & = & \frac{1}{r} \\ \Gamma_{ \, \phi \, \phi }^{ \, \theta   } & = & -\cos\left(\theta\right) \sin\left(\theta\right) \\ \Gamma_{ \, r \, \phi }^{ \, \phi   } & = & \frac{1}{r} \\ \Gamma_{ \, \theta \, \phi }^{ \, \phi   } & = & \frac{\cos\left(\theta\right)}{\sin\left(\theta\right)} \end{array}$$

Accessing to a Christoffel symbol specified by its indices (e.g. $\Gamma^{t}{}_{t\,r}$):

[16]:
```
g.christoffel_symbols()[0,0,1]
```

[16]:
$$-\frac{m}{2 \, m r - r^{2}}$$

Checking the symmetry on the last two indices:

[17]:
```
g.christoffel_symbols()[0,0,1] == g.christoffel_symbols()[0,1,0]
```

[17]: True

1.5 Riemann curvature tensor

The Riemann curvature tensor is obtained by the method `riemann()`:

[18]:
```
Riem = g.riemann()
print(Riem)
```

```
Tensor field Riem(g) of type (1,3) on the 4-dimensional Lorentzian manifold M
```

Displaying its nonredundant components:

[19]:
```
Riem.display_comp(only_nonredundant=True)
```

[19]:

$$\begin{array}{lcl}
\mathrm{Riem}\left(g\right)^{t}_{r\,t\,r} & = & -\frac{2\,m}{2\,mr^{2}-r^{3}} \\
\mathrm{Riem}\left(g\right)^{t}_{\theta\,t\,\theta} & = & -\frac{m}{r} \\
\mathrm{Riem}\left(g\right)^{t}_{\phi\,t\,\phi} & = & -\frac{m\sin\left(\theta\right)^{2}}{r} \\
\mathrm{Riem}\left(g\right)^{r}_{t\,t\,r} & = & -\frac{2\left(2\,m^{2}-mr\right)}{r^{4}} \\
\mathrm{Riem}\left(g\right)^{r}_{\theta\,r\,\theta} & = & -\frac{m}{r} \\
\mathrm{Riem}\left(g\right)^{r}_{\phi\,r\,\phi} & = & -\frac{m\sin\left(\theta\right)^{2}}{r} \\
\mathrm{Riem}\left(g\right)^{\theta}_{t\,t\,\theta} & = & \frac{2\,m^{2}-mr}{r^{4}} \\
\mathrm{Riem}\left(g\right)^{\theta}_{r\,r\,\theta} & = & -\frac{m}{2\,mr^{2}-r^{3}} \\
\mathrm{Riem}\left(g\right)^{\theta}_{\phi\,\theta\,\phi} & = & \frac{2\,m\sin\left(\theta\right)^{2}}{r} \\
\mathrm{Riem}\left(g\right)^{\phi}_{t\,t\,\phi} & = & \frac{2\,m^{2}-mr}{r^{4}} \\
\mathrm{Riem}\left(g\right)^{\phi}_{r\,r\,\phi} & = & -\frac{m}{2\,mr^{2}-r^{3}} \\
\mathrm{Riem}\left(g\right)^{\phi}_{\theta\,\theta\,\phi} & = & -\frac{2\,m}{r}
\end{array}$$

We can **lower and raise all the indices** of the components $R^{\lambda}{}_{\mu\nu\sigma}$ of the Riemann tensor, via the metric g by the methods `down()` and `up()`.

[20]:
```
Riemdown = Riem.down(g);
Riemup = Riem.up(g);
```

1.6 Ricci tensor

We know that the Ricci tensor is computed via the Riemann curvature tensor: $R_{\mu\nu} = R^{\lambda}{}_{\mu\lambda\nu}$. However, SageMath can give us directly the Ricci tensor from the metric g with the method `g.ricci()`.

[21]:
```
Ric = g.ricci()
```

[22]:
```
Ric.display()
```

[22]:

$$\mathrm{Ric}\left(g\right) = 0$$

[23]:
```
Ric[:]
```

[23]:

$$\left(\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right)$$

Let us check that the definition of the Ricci tensor via the contraction of the Riemann tensor and the one given by the SageMath method `g.ricci()` coincides.

[24]:
```
Ric == Riem.down(g)['_{abcd}']*ginv['^{ac}']
```

[24]:

True

1.7 Calculating the Scalar Curvature

It is computed by the contraction of theinverse metric and the Ricci tensor, i.e., $R = g^{\mu\nu} R_{\mu\nu}$.

[25]:
```
ScalarCurvature=ginv['^{ab}']*Ric['_ab']; ScalarCurvature
```

[25]: 0

1.8 Einstein tensor

The Einstein tensor is found from the tensors already computed, $G_{\mu\nu} = R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R$.

[26]:
```
EinsteinTensor = Ric-g*ScalarCurvature/2
```

[27]:
```
EinsteinTensor[:]
```

[27]:

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

A zero matrix means that the vacuum Einstein equation is satisfied.

1.9 Kretschmann scalar

The Kretschmann scalar is the "square" of the Riemann tensor defined by

$$K = R_{\lambda\mu\nu\sigma} R^{\lambda\mu\nu\sigma}$$

To compute it, we must first form the tensor fields whose components are $R_{\lambda\mu\nu\sigma}$ and $R^{\lambda\mu\nu\sigma}$. They are obtained by respectively lowering and raising the indices of the components $R^{\lambda}{}_{\mu\nu\sigma}$ of the Riemann tensor, via the metric g. These two operations are performed by the methods `down()` and `up()`. The contraction is performed by summation on repeated indices:

[28]:
```
K = Riem.down(g)['_{abcd}'] * Riem.up(g)['^{abcd}']
K
```

[28]: Scalar field on the 4-dimensional Lorentzian manifold M

[29]:
```
K.display()
```

[29]:

$$\begin{array}{llll} M & \longrightarrow & \mathbb{R} \\ (t, r, \theta, \phi) & \longmapsto & \frac{48\, m^2}{r^6} \end{array}$$

[30]:
```
K.expr()
```

[30]: $\frac{48\, m^2}{r^6}$

1.10 Levi-Civita Connection

The Levi-Civita Connection ∇ associated with the metric g.

[31]:
```
nab = g.connection() ; print(nab)
```

```
Levi-Civita connection nabla_g associated with the Lorentzian metric g on the
4-dimensional Lorentzian manifold M
```

We check the compatibility between metric and connection (that is, we must get $\nabla_g g = 0$).

[32]:
```
nab(g).display()
```

[32]:

$$\nabla_g g = 0$$

[33]:
```
w = M.vector_field('w')
```

Compute the covariant derivative of the vector $w = (-t^2, r^2, 0, 0)$, $\nabla_\nu w^\nu$.

[34]:
```
w[:] = [-t^2,r^2,0,0]
```

[35]:
```
DW = (nab(w)['^a_b']*delta['_a^b'])
DW.expr()
```

[35]:

$$4\,r - 2\,t$$

Check that $\nabla_\nu w^\nu = \partial_\nu w^\nu + w^\gamma {\Gamma^\nu}_{\gamma\nu}$.

[36]:
```
sum([w[i].diff(i)+w[i]*sum([g.christoffel_symbols()[j,i,j] for j in M.irange()])
 for i in M.irange()])
```

[36]:

$$4\,r - 2\,t$$