Calculate general relativity-related tensors using metric tensor as input

Narendra Bhatkar
UID : 229126
Roll No : 17

Department of Physics, St. Xavier's College

Abstract

In mathematics and physics, a tensor is a mathematical object that generalizes the concepts of scalars, vectors, and matrices to higher dimensions.Tensors find applications in various fields, including physics (where they describe physical properties like stress, strain, and electromagnetic fields) and differential geometry (where they are used to describe the geometry of curved spaces in general relativity). In this project we will try to numerically calculate the general relativity tensors by taking the number of dimensions of a manifold, the coordinate labels being used, and the components of a metric as input and calculate the non-zero components of the metric, derivatives of the metric, the Christoffel symbols, the derivatives of the Christoffel symbols, the Riemann curvature tensor, the Ricci curvature tensor, the Ricci scalar, and the Einstein tensor as output.

Aim

To calculate the general relativity tensors using metric tensor as input.

Introduction

In the context of general relativity, tensors are mathematical objects that play a crucial role in describing the geometry of spacetime and the equations that govern the behavior of gravity. Tensors are used to represent various physical quantities and mathematical relationships within this theory. Here are some important tensors in general relativity:

1. **Metric Tensor ($g_{\mu\nu}$)** : The metric tensor is perhaps the most central tensor in general relativity. It describes the geometry of spacetime at every point. This symmetric rank-2 tensor assigns a metric or distance measure to each pair of spacetime coordinates (t, x, y, z), capturing how spacetime is curved due to the presence of mass and energy. The components of the metric tensor $g_{\mu\nu}$ define the line element (the infinitesimal distance) in spacetime.

2. **Christoffel Symbols ($\Gamma_{\alpha\beta\gamma}$)** : They play a crucial role in describing how coordinate systems change as one moves through curved spacetime. Christoffel symbols are vital for understanding the paths that objects follow under the influence of gravity, as described by the geodesic equation. In general relativity, spacetime is curved due to the presence of mass and energy, as described by Einstein's theory of gravity. Unlike flat spacetime, where we can use Cartesian coordinates, curved spacetime requires the use of curvilinear coordinates. However, curvilinear coordinates can be challenging to work with, especially when dealing with derivatives. To work with derivatives and differentiation in curved spacetime, one needs a way to define a derivative that respects the curved geometry. This is where Christoffel symbols come into play. They define a connection or covariant derivative, which is a way to take derivatives while accounting for the curvature of spacetime. Given by

$$\Gamma^{\alpha}_{\beta\gamma} = \frac{1}{2} g^{\alpha\sigma} \left(\frac{\partial g_{\sigma\beta}}{\partial x^{\gamma}} + \frac{\partial g_{\sigma\gamma}}{\partial x^{\beta}} - \frac{\partial g_{\beta\gamma}}{\partial x^{\sigma}} \right) \quad (1)$$

Here, $g^{\alpha\sigma}$ represents the components of the inverse metric tensor.

3. **Riemann Tensor ($R_{\alpha\beta\gamma\delta}$):** The Riemann tensor is a rank-4 tensor that encodes the curvature of spacetime where each index can take values from 0 to 3. It is derived from the metric tensor and its derivatives. The components of the Riemann tensor describe how spacetime is curved in a non-Euclidean manner. This tensor is fundamental to understanding the effects of gravity and is used to derive the Einstein field equations.

4. **Ricci Tensor ($R_{\mu\nu}$)**: The Ricci tensor is a rank-2 tensor, which means it has two indices, typically represented as μ and v. Each index can take on values from 0 to 3, corresponding to spacetime dimensions.While the full Riemann tensor encodes all the detailed curvature information about spacetime, the Ricci tensor represents a contracted version of the Riemann tensor. In other words, it summarizes some key aspects of spacetime curvature. The Ricci tensor is derived by contracting two indices of the Riemann tensor. Specifically, it is obtained by taking the contraction $R_{\mu\nu} = R^{\alpha}_{\mu\alpha\nu}$, where $R^{\alpha}_{\mu\alpha\nu}$ is a component of the Riemann tensor. The Ricci tensor plays a central role in Einstein's field equations, which are the fundamental equations of general relativity. These equations relate the curvature of spacetime (represented by the Ricci tensor) to the distribution of matter and energy in the universe.

5. **Ricci Scalar (R)** : The Ricci scalar, often denoted as R or R_{ic}, is a scalar quantity derived from the components of the Riemann tensor and the metric tensor in the context of general relativity. It summarizes certain aspects of the curvature of spacetime in a more compact form. Unlike tensors, which have multiple indices and represent multidimensional quantities, the Ricci scalar is a scalar, which means it is a single number or value associated with a specific point in spacetime. The Ricci scalar is defined as the contraction (trace) of the Ricci tensor. Mathematically, it is expressed as

$$R = g^{\mu\nu} R_{\mu\nu},$$

where $g^{\mu\nu}$ represents the components of the inverse metric tensor and $R_{\mu\nu}$ are the components of the Ricci tensor.

6. **Einstein tensor ($G_{\mu\nu}$)** : The Einstein tensor, is a mathematical object used in Einstein's theory of general relativity. It is a symmetric rank-2 tensor that plays a central role in the field equations of general relativity, known as the Einstein field equations. These equations describe how gravity works by relating the curvature of spacetime to the distribution of matter and energy. It is a rank-2 tensor, which means it has two indices, each of which can take on values from 0 to 3, corresponding to spacetime dimensions.The Einstein tensor is constructed from the components of the Ricci tensor ($R_{\mu\nu}$) and the metric tensor ($g_{\mu\nu}$). It represents the curvature of spacetime as described by the Ricci tensor.

   The Einstein field equations are the core equations of general relativity. They relate the Einstein tensor ($G_{\mu\nu}$) to the stress-energy-momentum tensor ($T_{\mu\nu}$), which represents the distribution of matter, energy, and momentum in spacetime. The equations are written as:

$$G_{\mu\nu} = 8\pi G T_{\mu\nu},$$

   where G is the gravitational constant.

Procedure

1. First we import all the necessary libraries in python.

2. Define all the parameters so that the code prompts the user to input the number of dimensions for spacetime and labels for coordinates.

3. The Tensor class is defined using data classes, which is a way to create simple classes to hold data.

   The class includes attributes for the tensor's name, symbol, key (indicating contravariant and covariant indices), and components. Methods in the class allow for determining the rank of the tensor, creating tensors filled with zeros, mapping tensor indices to coordinate labels, and displaying the tensor.

4. The `assign` function is defined to assign components to a tensor, and the `fix_input` function is defined to process user input, converting ^ to ** for exponentiation and adding * between numbers and letters.

5. Then all the tensors are defined and the code is performed.

. Inverts the metric tensor to obtain its inverse.

. Computes the first derivatives of the metric components.

. Calculates the Christoffel symbols based on the Levi-Civita connection and Computes the Riemann curvature tensor using Christoffel symbols.

. Derives the Ricci curvature tensor from the Riemann tensor.

. Computes the Ricci scalar (if it is nonzero) and Computes the Einstein tensor based on the Ricci tensor and the metric tensor.

6. The code then displays the various tensors and quantities, including the metric tensor, its inverse, derivatives, Christoffel symbols, Riemann tensor, Ricci tensor, Ricci scalar (if nonzero), and the Einstein tensor.

Conclusion

In this project we were able to calculate the general relativity tensors by specifying the input metric tensors, The code used the user inputs to define the metric tensor for a 4-dimensional spacetime with coordinates (t, l, θ, ϕ). It then performed a series of calculations to derive these tensors and quantities which include the Christoffel symbols, Riemann curvature tensor, Ricci curvature tensor, Ricci scalar, and the Einstein tensor. And consequently the outputs were obtained which can be seen in the python file.

In [1]:
```
from sympy import *
from dataclasses import dataclass
from IPython.display import display as Idisplay
from IPython.display import Math

greek = ['alpha', 'beta', 'gamma', 'Gamma', 'delta', 'Delta', 'epsilon',
         'varepsilon', 'zeta', 'eta', 'theta', 'vartheta', 'Theta', 'iota',
         'kappa', 'lambda', 'Lambda', 'mu', 'nu', 'xi', 'Xi', 'pi', 'Pi',
         'rho', 'varrho', 'sigma', 'Sigma', 'tau', 'upsilon', 'Upsilon',
         'phi', 'varphi', 'Phi', 'chi', 'psi', 'Psi', 'omega', 'Omega']

n = int(input('Enter the number of dimensions:\n'))
coords = []
for i in range(n):
    coords.append(Symbol(str(input('Enter coordinate label %d:\n' % i))))


@dataclass(frozen=False, order=True)
class Tensor:
    name: str
    symbol: str
    key: str
    components: list

    def rank(self):
        return self.key.count('*')

    def tensor_zeros(self, t=0):
        for i in range(self.rank()):
            t = [t,] * n
        return MutableDenseNDimArray(t)

    def coord_id(self, o):
        a = []
        for i in range(self.rank()):
            c = int(o/(n**(self.rank() - i - 1)))
            a.append(str(coords[c]))
            if any(letter in a[i] for letter in greek) is True:
                a[i] = '\\' + a[i] + ' '
            o -= c * (n**(self.rank() - i - 1))
        x = self.key
        w = 0
        for i in x:
            if i == '*':
                x = x.replace('*', a[w], 1)
                w += 1
        return self.symbol + x

    def print_tensor(self):
        for o in range(len(self.components)):
            if self.components[o] != 0:
                Idisplay(Math(latex(Eq(Symbol(self.coord_id(o)),
                                       self.components[o]))))
        print('\n\n')


def assign(instance, thing):
    instance.components = thing.reshape(len(thing)).tolist()


def fix_input(expr):
```

```
    expr = expr.replace('^', '**')
    for i in range(len(expr)-1):
        if expr[i].isnumeric() and (expr[i+1].isalpha() or
                                    expr[i+1] == '('):
            expr = expr[:i+1] + '*' + expr[i+1:]
    return expr


metric = Tensor('metric tensor', 'g', '_**', [])
metric_inv = Tensor('inverse of metric tensor', 'g', '__**', [])
metric_d = Tensor('partial derivative of metric tensor', 'g', '_**,*', [])
Christoffel = Tensor('Christoffel symbol - 2nd kind', 'Gamma', '__*_**', []
Christoffel_d = Tensor('partial derivative of Christoffel symbol',
                       'Gamma', '__*_**,*', [])
Riemann = Tensor('Riemann curvature tensor', 'R', '__*_***', [])
Ricci = Tensor('Ricci curvature tensor', 'R', '_**', [])
Einstein = Tensor('Einstein tensor', 'G', '_**', [])
Einstein_alt = Tensor('Einstein tensor', 'G', '__*_*', [])

# user inputs metric:
g = eye(n)
while True:
    diag = str(input('Is metric diagonal? y for yes, n for no\n')).lower()
    if diag == 'y':
        for i in range(n):
            g[i, i] = sympify(fix_input(str(input(
                'What is g_[%s%s]?\n' % (str(coords[i]), str(coords[i])
                )))))
    else:
        for i in range(n):
            for j in range(i, n):
                g[i, j] = sympify(fix_input(str(input(
                    'What is g_[%s%s]?\n' % (str(coords[i]), str(coords[j])
                    )))))
                g[j, i] = g[i, j]
    if g.det() == 0:
        print('\nMetric is singular, try again!\n')
        continue
    else:
        break

# calculate everything:
# inverse metric:
g_inv = MutableDenseNDimArray(g.inv())
assign(metric_inv, g_inv)
g = MutableDenseNDimArray(g)
assign(metric, g)
# first derivatives of metric components:
g_d = metric_d.tensor_zeros()
for i in range(n):
    for j in range(i):
        for d in range(n):
            g_d[i, j, d] = g_d[j, i, d]
    for j in range(i, n):
        for d in range(n):
            g_d[i, j, d] = diff(g[i, j], coords[d])
assign(metric_d, g_d)
# Christoffel symbols for Levi-Civita connection (Gam^i_jk):
Gamma = Christoffel.tensor_zeros()
for i in range(n):
    for j in range(n):
```

```
        for k in range(j):
            Gamma[i, j, k] = Gamma[i, k, j]
        for k in range(j, n):
            for l in range(n):
                Gamma[i, j, k] += S(1)/2 * g_inv[i, l] * (
                    -g_d[j, k, l] + g_d[k, l, j] + g_d[l, j, k]
                    )
assign(Christoffel, Gamma)
# first derivatives of Christoffel symbols (Gam^i_jk,d):
Gamma_d = Christoffel_d.tensor_zeros()
for i in range(n):
    for j in range(n):
        for k in range(j):
            for d in range(n):
                Gamma_d[i, j, k, d] = Gamma_d[i, k, j, d]
        for k in range(j, n):
            for d in range(n):
                Gamma_d[i, j, k, d] = simplify(diff(Gamma[i, j, k],
                                                    coords[d]))
assign(Christoffel_d, Gamma_d)
# Riemann curvature tensor (R^i_jkl):
Rie = Riemann.tensor_zeros()
for i in range(n):
    for j in range(n):
        for k in range(n):
            for l in range(k):
                Rie[i, j, k, l] = -Rie[i, j, l, k]
            for l in range(k, n):
                Rie[i, j, k, l] = Gamma_d[i, j, l, k] - Gamma_d[i, j, k, l]
                for h in range(n):
                    Rie[i, j, k, l] += (Gamma[h, j, l] * Gamma[i, h, k]
                                        - Gamma[h, j, k] * Gamma[i, h, l])
                    Rie[i, j, k, l] = simplify(Rie[i, j, k, l])
assign(Riemann, Rie)
# Ricci curvature tensor (R_jl):
Ric = simplify(tensorcontraction(Rie, (0, 2)))
assign(Ricci, Ric)
# Ricci curvature scalar:
R = 0
for i in range(n):
    for j in range(n):
        R += g_inv[i, j] * Ric[i, j]
R = simplify(R)
# Einstein tensor (G_ij):
G = Einstein.tensor_zeros()
for i in range(n):
    for j in range(i):
        G[i, j] = G[j, i]
    for j in range(i, n):
        G[i, j] = simplify(Ric[i, j] - S(1)/2 * R * g[i, j])
assign(Einstein, G)
# G^i_j:
G_alt = Einstein_alt.tensor_zeros()
for i in range(n):
    for j in range(n):
        for k in range(n):
            G_alt[i, j] += g_inv[i, k] * G[k, j]
        G_alt[i, j] = simplify(G_alt[i, j])
assign(Einstein_alt, G_alt)

# print it all
```

```
print()
metric.print_tensor()
metric_inv.print_tensor()
metric_d.print_tensor()
Christoffel.print_tensor()
Christoffel_d.print_tensor()
Riemann.print_tensor()
Ricci.print_tensor()
if R != 0:
    Idisplay(Math(latex(Eq(Symbol('R'), R))))
    print('\n\n')
Einstein.print_tensor()
Einstein_alt.print_tensor()
```

```
Enter the number of dimensions:
4
Enter coordinate label 0:
t
Enter coordinate label 1:
l
Enter coordinate label 2:
theta
Enter coordinate label 3:
phi
Is metric diagonal? y for yes, n for no
y
What is g_[tt]?
-1
What is g_[ll]?
1
What is g_[thetatheta]?
r(l)^2
What is g_[phiphi]?
r(l)^2sin(theta)^2
```

$$g_{tt} = -1$$

$$g_{ll} = 1$$

$$g_{\theta\theta} = r^2(l)$$

$$g_{\phi\phi} = r^2(l)\sin^2(\theta)$$

$$g^{tt} = -1$$

$$g^{ll} = 1$$

$$g^{\theta\theta} = \frac{1}{r^2(l)}$$

$$g^{\phi\phi} = \frac{1}{r^2(l)\sin^2(\theta)}$$

$$g_{\theta\theta,l} = 2r(l)\frac{d}{dl}r(l)$$

$$g_{\phi\phi,l} = 2r(l)\sin^2(\theta)\frac{d}{dl}r(l)$$

$$g_{\phi\phi,\theta} = 2r^2(l)\sin(\theta)\cos(\theta)$$

$$\Gamma^l_{\theta\theta} = -r(l)\frac{d}{dl}r(l)$$

$$\Gamma^l_{\phi\phi} = -r(l)\sin^2(\theta)\frac{d}{dl}r(l)$$

$$\Gamma^\theta_{l\theta} = \frac{\frac{d}{dl}r(l)}{r(l)}$$

$$\Gamma^\theta_{\theta l} = \frac{\frac{d}{dl}r(l)}{r(l)}$$

$$\Gamma^\theta_{\phi\phi} = -\sin(\theta)\cos(\theta)$$

$$\Gamma^\phi_{l\phi} = \frac{\frac{d}{dl}r(l)}{r(l)}$$

$$\Gamma^\phi_{\theta\phi} = \frac{\cos(\theta)}{\sin(\theta)}$$

$$\Gamma^\phi_{\phi l} = \frac{\frac{d}{dl}r(l)}{r(l)}$$

$$\Gamma^\phi_{\phi\theta} = \frac{\cos(\theta)}{\sin(\theta)}$$

$$\Gamma^l_{\theta\theta,l} = -r(l)\frac{d^2}{dl^2}r(l) - \left(\frac{d}{dl}r(l)\right)^2$$

$$\Gamma^l_{\phi\phi,l} = \left(-r(l)\frac{d^2}{dl^2}r(l) - \left(\frac{d}{dl}r(l)\right)^2\right)\sin^2(\theta)$$

$$\Gamma^l_{\phi\phi,\theta} = -r(l)\sin(2\theta)\frac{d}{dl}r(l)$$

$$\Gamma^\theta_{l\theta,l} = \frac{r(l)\frac{d^2}{dl^2}r(l) - \left(\frac{d}{dl}r(l)\right)^2}{r^2(l)}$$

$$\Gamma^\theta_{\theta l,l} = \frac{r(l)\frac{d^2}{dl^2}r(l) - \left(\frac{d}{dl}r(l)\right)^2}{r^2(l)}$$

$$\Gamma^{\theta}_{\phi\phi,\theta} = -\cos(2\theta)$$

$$\Gamma^{\phi}_{l\phi,l} = \frac{r(l)\frac{d^2}{dl^2}r(l) - \left(\frac{d}{dl}r(l)\right)^2}{r^2(l)}$$

$$\Gamma^{\phi}_{\theta\phi,\theta} = -\frac{1}{\sin^2(\theta)}$$

$$\Gamma^{\phi}_{\phi l,l} = \frac{r(l)\frac{d^2}{dl^2}r(l) - \left(\frac{d}{dl}r(l)\right)^2}{r^2(l)}$$

$$\Gamma^{\phi}_{\phi\theta,\theta} = -\frac{1}{\sin^2(\theta)}$$

$$R^{l}_{\theta l\theta} = -r(l)\frac{d^2}{dl^2}r(l)$$

$$R^{l}_{\theta\theta l} = r(l)\frac{d^2}{dl^2}r(l)$$

$$R^{l}_{\phi l\phi} = -r(l)\sin^2(\theta)\frac{d^2}{dl^2}r(l)$$

$$R^{l}_{\phi\phi l} = r(l)\sin^2(\theta)\frac{d^2}{dl^2}r(l)$$

$$R^{\theta}_{ll\theta} = \frac{\frac{d^2}{dl^2}r(l)}{r(l)}$$

$$R^{\theta}_{l\theta l} = -\frac{\frac{d^2}{dl^2}r(l)}{r(l)}$$

$$R^{\theta}_{\phi\theta\phi} = \left(1 - \left(\frac{d}{dl}r(l)\right)^2\right)\sin^2(\theta)$$

$$R^{\theta}_{\phi\phi\theta} = -\left(1 - \left(\frac{d}{dl}r(l)\right)^2\right)\sin^2(\theta)$$

$$R^{\phi}_{ll\phi} = \frac{\frac{d^2}{dl^2}r(l)}{r(l)}$$

$$R^{\phi}_{l\phi l} = -\frac{\frac{d^2}{dl^2}r(l)}{r(l)}$$

$$R^{\phi}_{\theta\theta\phi} = \left(\frac{d}{dl}r(l)\right)^2 - 1$$

$$R^{\phi}_{\theta\phi\theta} = 1 - \left(\frac{d}{dl}r(l)\right)^2$$

$$R_{ll} = -\frac{2\frac{d^2}{dl^2}r(l)}{r(l)}$$

$$R_{\theta\theta} = -r(l)\frac{d^2}{dl^2}r(l) - \left(\frac{d}{dl}r(l)\right)^2 + 1$$

$$R_{\phi\phi} = \left(-r(l)\frac{d^2}{dl^2}r(l) - \left(\frac{d}{dl}r(l)\right)^2 + 1\right)\sin^2(\theta)$$

$$R = \frac{2\left(-2r(l)\frac{d^2}{dl^2}r(l) - \left(\frac{d}{dl}r(l)\right)^2 + 1\right)}{r^2(l)}$$

$$G_{tt} = \frac{-2r(l)\frac{d^2}{dl^2}r(l) - \left(\frac{d}{dl}r(l)\right)^2 + 1}{r^2(l)}$$

$$G_{ll} = \frac{\left(\frac{d}{dl}r(l)\right)^2 - 1}{r^2(l)}$$

$$G_{\theta\theta} = r(l)\frac{d^2}{dl^2}r(l)$$

$$G_{\phi\phi} = r(l)\sin^2(\theta)\frac{d^2}{dl^2}r(l)$$

$$G_t^t = \frac{2r(l)\frac{d^2}{dl^2}r(l) + \left(\frac{d}{dl}r(l)\right)^2 - 1}{r^2(l)}$$

$$G_l^l = \frac{\left(\frac{d}{dl}r(l)\right)^2 - 1}{r^2(l)}$$

$$G_\theta^\theta = \frac{\frac{d^2}{dl^2}r(l)}{r(l)}$$

$$G_\phi^\phi = \frac{\frac{d^2}{dl^2}r(l)}{r(l)}$$