kerr_sage

November 15, 2019

[21]:
```
%display latex
```

[22]:
```
Parallelism().set(nproc = 4)
```

1 Metric

[23]:
```
M = Manifold(4, 'M', r'\mathcal{M}');
```

[24]:
```
M0 = M.open_subset('M0', r'\mathcal{M}_0')
# BL = Boyer-Lindquist
BL.<t,r,th,ph> = M0.chart(r't r:(0,+oo) th:(0,pi):\theta ph:(0,2*pi):\phi')
```

[43]:
```
var('u', latex_name = '\\mu')
var('a')
var('q');

n = 4
```

[26]:
```
g = M.lorentzian_metric('g')
```

[27]:
```
p_sq = r^2 + (a*cos(th))^2
D = r^2 -2*u*r + a^2
E_sq = (r^2 + a^2)**2 - a^2*D*sin(th)^2

g[0,0] = (D - a^2*sin(th)^2)/p_sq
g[0,3] = 2*u*a*r*sin(th)^2/p_sq
g[1,1], g[2,2] = -p_sq/D, -p_sq
g[3,0] = 2*u*a*r*sin(th)^2/p_sq
g[3,3] = -E_sq*sin(th)^2/p_sq

g.display()
```

[27]:
```
g = -(a^2*sin(th)^2 - a^2 - r^2 + 2*r*u)/(a^2*cos(th)^2 + r^2) dt*dt +
2*a*r*u*sin(th)^2/(a^2*cos(th)^2 + r^2) dt*dph - (a^2*cos(th)^2 + r^2)/(a^2 +
r^2 - 2*r*u) dr*dr + (-a^2*cos(th)^2 - r^2) dth*dth +
```

```
2*a*r*u*sin(th)^2/(a^2*cos(th)^2 + r^2) dph*dt + ((a^2 + r^2 -
2*r*u)*a^2*sin(th)^2 - (a^2 + r^2)^2)*sin(th)^2/(a^2*cos(th)^2 + r^2) dph*dph
```

2 Christoffel symbols

[28]:
```
nab = g.connection()
```

[29]:
```
chris_latex = open('christoffel_sage.txt', 'w')

for i in range(n):
    for j in range(n):
        for k in range(n):
            if nab[i, j, k] != 0:
                chris = nab[i, j, k]

                chris_latex.write(str(i) + ';' + str(j) + ';' + str(k) + ';' +␣
↪'"' + str(chris) + '"' + '\n')
                #rim_latex.write(latex(rim) + '\n')

               #display([i, j, k, l], rim)

chris_latex.close()
```

3 Riemann tensor

[31]:
```
R = g.riemann()
```

[32]:
```
rim_latex = open('riemann_sage.txt', 'w')

for i in range(n):
    for j in range(n):
        for k in range(n):
            for l in range(n):
                    if R[i, j, k, l] != 0:
                        rim = R[i, j, k, l]

                        rim_latex.write(str(i) + ';' + str(j) + ';' + str(k) +␣
↪';' + str(l) + ';' + '"' + str(rim) + '"' + '\n')
                        #rim_latex.write(latex(rim) + '\n')

                       #display([i, j, k, l], rim)

rim_latex.close()
```

4 Ricci tensor

[33]:
```
Ric = g.ricci()
```

[34]:
```
Ric[:]
```

[34]:
```
[0 0 0 0]
[0 0 0 0]
[0 0 0 0]
[0 0 0 0]
```

5 Kretshmann scalar

[35]:
```
dR = R.down(g)
```

[36]:
```
uR = R.up(g)
```

[37]:
```
Kr_scalar = uR['^ijkl']*dR['_ijkl']
```

[38]:
```
Kr = Kr_scalar.coord_function()
Kr.factor()
```

[38]:
```
-48*(a^2*cos(th)^2 + 4*a*r*cos(th) + r^2)*(a^2*cos(th)^2 - 4*a*r*cos(th) +
r^2)*(a*cos(th) + r)*(a*cos(th) - r)*u^2/(a^2*cos(th)^2 + r^2)^6
```

[39]:
```
Kr_KN = 8/(r^2+(a*cos(th))^2)^6 *(6*u^2*(r^6 - 15*r^4*(a*cos(th))^2 +␣
↪15*r^2*(a*cos(th))^4 - (a*cos(th))^6)- 12*u*q^2*r*(r^4 - 10*(a*r*cos(th))^2␣
↪+ 5*(a*cos(th))^4) + q^4*(7*r^4 - 34*(a*r*cos(th))^2 + 7*(a*cos(th))^4))
```

[40]:
```
Kr == Kr_KN.subs(q = 0)
```

[40]:
```
True
```