# Affine Connection and Curvature

The Rindler Metric

## Initialization

In[ ]:=
```
Needs["xAct`xCoba`"]
```

```
------------------------------------------------------------

Package xAct`xPerm`  version 1.2.3, {2015, 8, 23}

CopyRight (C) 2003-2018, Jose M. Martin-Garcia, under the General Public License.

Connecting to external linux executable...

Connection established.

------------------------------------------------------------

Package xAct`xTensor`  version 1.1.3, {2018, 2, 28}

CopyRight (C) 2002-2018, Jose M. Martin-Garcia, under the General Public License.

------------------------------------------------------------

Package xAct`xCoba`  version 0.8.4, {2018, 2, 28}

CopyRight (C) 2005-2018, David Yllanes and
  Jose M. Martin-Garcia, under the General Public License.

------------------------------------------------------------

These packages come with ABSOLUTELY NO WARRANTY; for details type
  Disclaimer[]. This is free software, and you are welcome to redistribute
  it under certain conditions. See the General Public License for details.

------------------------------------------------------------
```

In[ ]:=
```
(*$PrePrint=ScreenDollarIndices;
$DefInfoQ=False;
$UndefInfoQ=False;*)
```

In[ ]:=
```
DefManifold[M, 2, {λ, μ, ν, ρ, σ, α, β, γ, δ}];
dimM = DimOfManifold[M];
dimM1 = dimM - 1;
```

```
** DefManifold: Defining manifold M.

** DefVBundle: Defining vbundle TangentM.
```

Here is the definition of the coordinate system, and the metric:

Simply define the list coords = {....} and the matrix gmatrix

In[ ]:=
```
coords = {η[], ξ[]};
(*DefConstantSymbol[a,PrintAs→"a"];*)
(*DefScalarFunction[ascale,PrintAs→"a"];
Use as e.g. ascale[t[],r[]] for a function of (t,r) *)
gmatrix = DiagonalMatrix[
   {-ξ[]^2, 1}
  ];
DefChart[ch, M, {0, 1}, coords, ChartColor → Blue];
g = CTensor[gmatrix, {-ch, -ch}];
SetCMetric[g, ch, SignatureOfMetric → {1, 1, 0}];
CD = CovDOfMetric[g];
```

```
** DefChart: Defining chart ch.
** DefTensor: Defining coordinate scalar η[].
** DefTensor: Defining coordinate scalar ξ[].
** DefMapping: Defining mapping ch.
** DefMapping: Defining inverse mapping ich.
** DefTensor: Defining mapping differential tensor dich[-a, ichα].
** DefTensor: Defining mapping differential tensor dch[-α, cha].
** DefBasis: Defining basis ch. Coordinated basis.
** DefCovD: Defining parallel derivative PDch[-α].
** DefTensor: Defining vanishing torsion tensor TorsionPDch[α, -β, -γ].
** DefTensor: Defining symmetric Christoffel tensor ChristoffelPDch[α, -β, -γ].
** DefTensor: Defining vanishing Riemann tensor RiemannPDch[-α, -β, -γ, δ].
** DefTensor: Defining vanishing Ricci tensor RicciPDch[-α, -β].
** DefTensor: Defining antisymmetric +1 density etaUpch[α, β].
** DefTensor: Defining antisymmetric -1 density etaDownch[-α, -β].
```

In[ ]:=
```
Print[
 "gμν= ", ComponentArray[g[{-μ, -ch}, {-ν, -ch}]] // MatrixForm, "     ,         ",
 "gμν= ", ComponentArray[g[{ μ,  ch}, { ν,  ch}]] // MatrixForm, "\n",
 "g = ", Determinant[g, ch][], " = ",
 Det[ComponentArray[g[{-μ, -ch}, {-ν, -ch}]]] // Simplify
]
```

$g_{\mu\nu}= \begin{pmatrix} -\xi^2 & 0 \\ 0 & 1 \end{pmatrix}$ , $g^{\mu\nu}= \begin{pmatrix} -\frac{1}{\xi^2} & 0 \\ 0 & 1 \end{pmatrix}$

$g = -\tilde{\tilde{\xi}}^2 = -\xi^2$

# Affine Connection

Print nonzero components:
$\Gamma^{\lambda}{}_{\mu\nu} = \frac{1}{2} g^{\lambda\sigma}(\partial_\mu g_{\sigma\nu} + \partial_\nu g_{\sigma\mu} - \partial_\sigma g_{\mu\nu})$
The components of the Christoffel symbols are collected by the ComponentArray[expr] function.

In[ ]:=
```
list = ComponentArray[Christoffel[CD, PDch][{α, ch}, {-β, -ch}, {-γ, -ch}]];
list = Table[
    If[
     UnsameQ[list[[i, j, k]], 0],
     {Subscript[Superscript["Γ", i - 1], j - 1, k - 1], list[[i, j, k]]}
    ],
    {i, 1, Length[list]}, {j, 1, Length[list]}, {k, 1, j}
   ];
Partition[DeleteCases[Flatten[list], Null], 2] // TableForm
```

Out[ ]//TableForm=

| | |
|---|---|
| $\Gamma^{0}{}_{1,0}$ | $\frac{1}{\xi}$ |
| $\Gamma^{1}{}_{0,0}$ | $\xi$ |

# Curvature

Print nonzero components of Riemann: $R^{\mu}{}_{\nu\rho\sigma}$
$R^{\lambda}{}_{\rho\mu\nu} = \partial_\mu \Gamma^{\lambda}{}_{\nu\rho} - \partial_\nu \Gamma^{\lambda}{}_{\mu\rho} + \Gamma^{\lambda}{}_{\mu\sigma} \Gamma^{\sigma}{}_{\nu\rho} - \Gamma^{\lambda}{}_{\nu\sigma} \Gamma^{\sigma}{}_{\mu\rho}$ (Carroll+Hartle’s convention)
xCoba has Wald’s convention, which for a Levi-Civita Connection gives the same reult after some index raising/lowering.

The components of the Riemann tensor are collected by the ComponentArray[expr] function.

In[ ]:=
```
list = ComponentArray[Riemann[CD][{α, ch}, {-β, -ch}, {-γ, -ch}, {-δ, -ch}]];
list = Table[
   If[
    UnsameQ[list[[i, j, k, l]], 0], {Subscript[Superscript["R", i - 1], j - 1, k - 1, l - 1]
     , list[[i, j, k, l]]}
   ],
   {i, 1, Length[list]}, {j, 1, Length[list]}, {k, 1, Length[list]}, {l, 1, k - 1}
  ];
Partition[DeleteCases[Flatten[list], Null], 2] // TableForm
```

Out[ ]//TableForm=
```
{}
```

The Riemann tensor with all lower indices: $R_{\mu\nu\rho\sigma}$

In[ ]:=
```
list = ComponentArray[Riemann[CD][{-α, -ch}, {-β, -ch}, {-γ, -ch}, {-δ, -ch}]];
list = Table[
   If[
    UnsameQ[list[[i, j, k, l]], 0], {Subscript["R", i - 1, j - 1, k - 1, l - 1]
     , list[[i, j, k, l]]}
   ],
   {i, 1, Length[list]}, {j, 1, i - 1}, {k, 1, Length[list]}, {l, 1, k - 1}
  ];
Partition[DeleteCases[Flatten[list], Null], 2] // TableForm
```

Out[ ]//TableForm=
```
{}
```

The Riemann tensor with all upper indices: $R^{\mu\nu\rho\sigma}$

In[∘ ]:=
```
list = ComponentArray[Riemann[CD][{α, ch}, {β, ch}, {γ, ch}, {δ, ch}]];
list = Table[
   If[
    UnsameQ[list[[i, j, k, l]], 0],
    {Superscript[Superscript[Superscript[Superscript["R", i - 1], j - 1], k - 1], l - 1]
      , list[[i, j, k, l]]}
   ],
   {i, 1, Length[list]}, {j, 1, i - 1}, {k, 1, Length[list]}, {l, 1, k - 1}
  ];
Partition[DeleteCases[Flatten[list], Null], 2] // TableForm
```

Out[∘ ]//TableForm=

```
{}
```

The Ricci tensor: $R_{\mu\nu}$

In[∘ ]:=
```
list = ComponentArray[Ricci[CD][{-α, -ch}, {-β, -ch}]];
list = Table[
   If[
    UnsameQ[list[[i, j]], 0], {Subscript["R", i - 1, j - 1]
      , list[[i, j]]}
   ],
   {i, 1, Length[list]}, {j, 1, i}
  ];
Partition[DeleteCases[Flatten[list], Null], 2] // TableForm
```

Out[∘ ]//TableForm=

```
{}
```

Ricci scalar:

In[∘ ]:=
```
Print["R = ", RicciScalar[CD][]]
```

```
R = 0
```

$R^2$ scalar

In[∘ ]:=
```
Print["R² = ", Kretschmann[CD][]]
```

```
R² = 0
```

Einstein tensor: $G_{\mu\nu}$

In[ ]:=
```
list = ComponentArray[Einstein[CD][{-α, -ch}, {-β, -ch}]];
list = Table[
    If[
     UnsameQ[list⟦i, j⟧, 0], {Subscript["G", i - 1, j - 1]
       , list⟦i, j⟧}
    ],
    {i, 1, Length[list]}, {j, 1, i}
   ];
Partition[DeleteCases[Flatten[list], Null], 2] // TableForm
```

Out[ ]//TableForm=
```
{}
```

Weyl tensor:

# Geodesic Equations

In[ ]:=
```
DefTensor[u[μ], M];
```

```
** DefTensor: Defining tensor u[μ].
```

The geodesic equations are:

$\frac{du^\mu}{d\tau} + \Gamma^\mu{}_{\nu\rho}\, u^\nu\, u^\rho = 0$

geqs is a list with the second term geqs[$\mu$+1]=$\Gamma^\mu{}_{\nu\rho}\, u^\nu\, u^\rho$ for each $\mu$=0,..,d-1.

(careful, in the expressions below, when e.g. $u^{3^2} = (u^3)^2$, the squared terms don't appear nicely)

In[ ]:=
```
geqs = ComponentArray[Christoffel[CD, PDch][{μ, ch}, {-ν, -ch}, {-ρ, -ch}]
       u[{ν, ch}] u[{ρ, ch}]] // ContractBasis // Simplify;
For[i = 1, i ≤ Length[geqs], i++, Print["d/dτ", u[{i - 1, ch}], "+(", geqs⟦i⟧, ")=0"]]
```

$\frac{d}{d\tau} u^0 + (\frac{2\, u^0\, u^1}{\xi}) = 0$

$\frac{d}{d\tau} u^1 + ( {u^0}^2\, \xi) = 0$

In[ ]:=
```
DefScalarFunction[dx0 , PrintAs → "η̇"];
DefScalarFunction[dx1 , PrintAs → "ξ̇"];
DefScalarFunction[ddx0, PrintAs → "η̈"];
DefScalarFunction[ddx1, PrintAs → "ξ̈"];
u  = CTensor[{dx0[]  , dx1[] }, {ch}];
du = CTensor[{ddx0[], ddx1[]}, {ch}];
{u, du}
```

```
** DefScalarFunction: Defining scalar function dx0.
** DefScalarFunction: Defining scalar function dx1.
** DefScalarFunction: Defining scalar function ddx0.
** DefScalarFunction: Defining scalar function ddx1.
```

Out[ ]=
```
{CTensor[{η̇[], ξ̇[]}, {ch}, 0], CTensor[{η̈[], ξ̈[]}, {ch}, 0]}
```

In[ ]:=
```
geqs = ComponentArray[Christoffel[CD, PDch][{μ, ch}, {-ν, -ch}, {-ρ, -ch}]
      u[{ν, ch}] u[{ρ, ch}]] // ContractBasis // Simplify;
For[i = 1, i ≤ Length[geqs], i++, Print[du[{i - 1, ch}], "+(", geqs[[i]], ")=0"]]
```

$\ddot{\eta}[\,]+\left(\frac{2\,\dot{\eta}[\,]\,\dot{\xi}[\,]}{\xi}\right)=0$

$\ddot{\xi}[\,]+\left(\dot{\eta}[\,]^2\,\xi\right)=0$

# Acknowledgements

This notebook has been programmed by Konstantinos Anagnostopoulos, Physics Department, National Technical University of Athens, Greece, while he was an instructor of the 4th year undergraduate course "General Relativity and Cosmology". It was created for fun, but it may turn out to be useful to everyone studying the General Theory of Relativity for the first time.

**Email**: konstant@mail.ntua.gr
**Web**: http://physics.ntua.gr/konstant