Newman-Penrose basis and gravitational waves

Preliminaries

Load tensor packages

In[ ]:=
```
<< xAct`xPert`
```

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

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

CopyRight (C) 2003-2020, Jose M. Martin-Garcia, under the General Public License.

Connecting to external linux executable...

Connection established.

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

Package xAct`xTensor`  version 1.1.5, {2021, 2, 28}

CopyRight (C) 2002-2021, Jose M. Martin-Garcia, under the General Public License.

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

Package xAct`xPert`  version 1.0.6, {2018, 2, 28}

CopyRight (C) 2005-2020, David Brizuela, Jose M. Martin-Garcia
  and Guillermo A. Mena Marugan, 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.

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

** Variable $PrePrint assigned value ScreenDollarIndices

** Variable $CovDFormat changed from Prefix to Postfix

** Option AllowUpperDerivatives of ContractMetric changed from False to True

** Option MetricOn of MakeRule changed from None to All

** Option ContractMetrics of MakeRule changed from False to True
```

In[ ]:=
```
<< xAct`xCoba`
```

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

Package xAct`xCoba`  version 0.8.6, {2021, 2, 28}

CopyRight (C) 2005-2021, 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.

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

Nicer printing

In[ ]:=
```
$PrePrint = ScreenDollarIndices;
```

In[ ]:=
```
$CovDFormat = "Prefix";
```

In[ ]:=
```
$CVVerbose = False;
```

In[ ]:=
```
$CIndexForm = True;
```

Make rules from equations

In[ ]:=
```
mkrg[eq_Equal] := MakeRule[Evaluate[List @@ eq], MetricOn → All, ContractMetrics → True]
```

In[ ]:=
```
mkr0[eq_Equal] :=
 MakeRule[Evaluate[List @@ eq], MetricOn → None, ContractMetrics → False]
```

Object definitions

Spacetime manifold

The spacetime manifold M, on which tensors will be defined. Some Greek letters are defined as tangent space indices.

In[ ]:=
```
DefManifold[M, 4, {α, β, γ, μ, ν, ρ, σ, τ, ψ, κ, ζ}]
```

```
** DefManifold: Defining manifold M.

** DefVBundle: Defining vbundle TangentM.
```

Coordinates

In the following, we will use a coordinate chart $(t,\ x,\ y,\ z)$.

In[∘]:=
```
DefTensor[tt[], {M}, PrintAs → "t"]
```

```
** DefTensor: Defining tensor tt[].
```

In[∘]:=
```
DefTensor[xx[], {M}, PrintAs → "x"]
```

```
** DefTensor: Defining tensor xx[].
```

In[∘]:=
```
DefTensor[yy[], {M}, PrintAs → "y"]
```

```
** DefTensor: Defining tensor yy[].
```

In[∘]:=
```
DefTensor[zz[], {M}, PrintAs → "z"]
```

```
** DefTensor: Defining tensor zz[].
```

In[∘]:=
```
DefChart[coord, M, {0, 1, 2, 3}, {tt[], xx[], yy[], zz[]}, ChartColor → Blue]
```

```
** DefChart: Defining chart coord.
** DefMapping: Defining mapping coord.
** DefMapping: Defining inverse mapping icoord.
** DefTensor: Defining mapping differential tensor dicoord[-ạ, icoordα].
** DefTensor: Defining mapping differential tensor dcoord[-α, coordạ].
** DefBasis: Defining basis coord. Coordinated basis.
** DefCovD: Defining parallel derivative PDcoord[-α].
** DefTensor: Defining vanishing torsion tensor TorsionPDcoord[α, -β, -γ].
** DefTensor: Defining symmetric Christoffel tensor ChristoffelPDcoord[α, -β, -γ].
** DefTensor: Defining vanishing Riemann tensor RiemannPDcoord[-α, -β, -γ, ζ].
** DefTensor: Defining vanishing Ricci tensor RicciPDcoord[-α, -β].
** DefTensor: Defining antisymmetric +1 density etaUpcoord[α, β, γ, ζ].
** DefTensor: Defining antisymmetric -1 density etaDowncoord[-α, -β, -γ, -ζ].
```

Newman-Penrose basis

The most important step is the introduction of a new, complex, double-null basis, known as the Newman-Penrose basis. This is not a coordinate basis.

In[∘]:=
```
DefBasis[newpen, TangentM, {0, 1, 2, 3},
  BasisChange → CTensor[{{1, 0, 0, 1}, {1, 0, 0, -1} / 2, {0, 1, I, 0} / Sqrt[2],
     {0, 1, -I, 0} / Sqrt[2]}, {-newpen, coord}], BasisColor → Red]
```

```
** DefBasis: Defining basis newpen.

** DefCovD: Defining parallel derivative PDnewpen[-α].

** DefTensor: Defining torsion tensor TorsionPDnewpen[α, -β, -γ].

** DefTensor: Defining non-symmetric Christoffel tensor ChristoffelPDnewpen[α, -β, -γ].

** DefTensor: Defining vanishing Riemann tensor RiemannPDnewpen[-α, -β, -γ, ζ].

** DefTensor: Defining vanishing Ricci tensor RicciPDnewpen[-α, -β].

** DefTensor: Defining Jacobiancoordnewpen[].

** DefTensor: Defining tensor ChristoffelPDcoordPDnewpen[α, -β, -γ].

** DefTensor: Defining antisymmetric +1 density etaUpnewpen[α, β, γ, ζ].

** DefTensor: Defining antisymmetric -1 density etaDownnewpen[-α, -β, -γ, -ζ].
```

In[ ]:=
```
newpen /: CIndexForm[0, newpen] := "l";
```

In[ ]:=
```
newpen /: CIndexForm[1, newpen] := "n";
```

In[ ]:=
```
newpen /: CIndexForm[2, newpen] := "m";
```

In[ ]:=
```
newpen /: CIndexForm[3, newpen] := "m̅";
```

Metric

The metric g of signature (-,+,+,+). The Levi-Civita derivative of a tensor A_μ will be written as $\nabla_\mu A_\nu$ in prefix notation or $A_{\nu;\mu}$ in postfix notation.

In[ ]:=
```
DefMetric[{3, 1, 0}, Met[-μ, -ν], CD, {";", "▽"}, PrintAs → "g"]
```

```
** DefTensor: Defining symmetric metric tensor Met[-μ, -ν].
** DefTensor: Defining antisymmetric tensor epsilonMet[-α, -β, -γ, -ζ].
** DefTensor: Defining tetrametric TetraMet[-α, -β, -γ, -ζ].
** DefTensor: Defining tetrametric TetraMet†[-α, -β, -γ, -ζ].
** DefCovD: Defining covariant derivative CD[-μ].
** DefTensor: Defining vanishing torsion tensor TorsionCD[α, -β, -γ].
** DefTensor: Defining symmetric Christoffel tensor ChristoffelCD[α, -β, -γ].
** DefTensor: Defining Riemann tensor RiemannCD[-α, -β, -γ, -ζ].
** DefTensor: Defining symmetric Ricci tensor RicciCD[-α, -β].
** DefCovD:  Contractions of Riemann automatically replaced by Ricci.
** DefTensor: Defining Ricci scalar RicciScalarCD[].
** DefCovD:  Contractions of Ricci automatically replaced by RicciScalar.
** DefTensor: Defining symmetric Einstein tensor EinsteinCD[-α, -β].
** DefTensor: Defining Weyl tensor WeylCD[-α, -β, -γ, -ζ].
** DefTensor: Defining symmetric TFRicci tensor TFRicciCD[-α, -β].
** DefTensor: Defining Kretschmann scalar KretschmannCD[].
** DefCovD:  Computing RiemannToWeylRules for dim 4
** DefCovD:  Computing RicciToTFRicci for dim 4
** DefCovD:  Computing RicciToEinsteinRules for dim 4
** DefTensor: Defining weight +2 density DetMet[]. Determinant.
```

In[∘]:=
```
DefMetricPerturbation[Met, δMet, ϵ, PrintAs → "h"]
```

```
** DefParameter: Defining parameter ϵ.
** DefTensor: Defining tensor δMet[LI[order], -α, -β].
```

Wave covector

In the following we will study a plane wave, whose propagation is governed by a constant wave covector and frequency.

In[∘]:=
```
DefTensor[wav[-μ], {M}, PrintAs → "k"]
```

```
** DefTensor: Defining tensor wav[-μ].
```

In[∘]:=
```
DefConstantSymbol[freq, PrintAs → "ω"]
```

```
** DefConstantSymbol: Defining constant symbol freq.
```

Lorentz transformation

We further study Lorentz transformations, which leave the wave covector invariant.

```
In[ ]:= DefTensor[lor[μ, -ν], {M}, PrintAs → "Λ"]

** DefTensor: Defining tensor lor[μ, -ν].

In[ ]:= DefConstantSymbol[ar, PrintAs → "ℛα"]

** DefConstantSymbol: Defining constant symbol ar.

In[ ]:= DefConstantSymbol[ai, PrintAs → "ℐα"]

** DefConstantSymbol: Defining constant symbol ai.

In[ ]:= DefConstantSymbol[phi, PrintAs → "ϕ"]

** DefConstantSymbol: Defining constant symbol phi.
```

Newman-Penrose quantities

The following quantities serve as the relevant components of the curvature tensor in the Newman-Penrose basis.

```
In[ ]:= DefTensor[psi2[], {M}, PrintAs → "Ψ2"]

** DefTensor: Defining tensor psi2[].

In[ ]:= DefTensor[psi3[], {M}, PrintAs → "Ψ3"]

** DefTensor: Defining tensor psi3[].

In[ ]:= DefTensor[psi3b[], {M}, PrintAs → "Ψ̄3"]

** DefTensor: Defining tensor psi3b[].

In[ ]:= DefTensor[psi4[], {M}, PrintAs → "Ψ4"]

** DefTensor: Defining tensor psi4[].

In[ ]:= DefTensor[psi4b[], {M}, PrintAs → "Ψ̄4"]

** DefTensor: Defining tensor psi4b[].

In[ ]:= DefTensor[phi22[], {M}, PrintAs → "Φ22"]

** DefTensor: Defining tensor phi22[].
```

Scalar field

Finally, we also define a scalar field, so that we can study gravitational waves in scalar-tensor gravity.

In[ ]:=
```
DefTensor[scal[], {M}, PrintAs → "ψ"]
```

```
** DefTensor: Defining tensor scal[].
```

Component values

Basis transformation

Let us first have a look at the vectors which span the Newman-Penrose basis. Here are the components of the vectors *l*, *n*, *m* $\overline{m}$, expressed in the coordinate basis.

In[ ]:=
```
BasisValues[-newpen, coord]
```

Out[ ]=

FoldedRule[{},

$$\left\{e_l{}^t \to 1,\ e_l{}^x \to 0,\ e_l{}^y \to 0,\ e_l{}^z \to 1,\ e_n{}^t \to \frac{1}{2},\ e_n{}^x \to 0,\ e_n{}^y \to 0,\ e_n{}^z \to -\frac{1}{2},\ e_m{}^t \to 0,\ e_m{}^x \to \frac{1}{\sqrt{2}},\ e_m{}^y \to \frac{i}{\sqrt{2}},\ e_m{}^z \to 0,\ e_{\overline{m}}{}^t \to 0,\ e_{\overline{m}}{}^x \to \frac{1}{\sqrt{2}},\ e_{\overline{m}}{}^y \to -\frac{i}{\sqrt{2}},\ e_{\overline{m}}{}^z \to 0\right\}$$]

In[ ]:=
```
ToValues[ComponentArray[Basis[{-μ, -newpen}, {ν, coord}]]]
```

Out[ ]=

$$\left\{\{1, 0, 0, 1\}, \left\{\frac{1}{2}, 0, 0, -\frac{1}{2}\right\}, \left\{0, \frac{1}{\sqrt{2}}, \frac{i}{\sqrt{2}}, 0\right\}, \left\{0, \frac{1}{\sqrt{2}}, -\frac{i}{\sqrt{2}}, 0\right\}\right\}$$

Background metric

This is the Minkowski background metric in its familiar form in the coordinate basis.

In[ ]:=
```
AllComponentValues[Met[{-μ, -coord}, {-ν, -coord}], DiagonalMatrix[{-1, 1, 1, 1}]];
```

In[ ]:=
```
ToValues[ComponentArray[Met[{-μ, -coord}, {-ν, -coord}]]] // MatrixForm
```

Out[ ]//MatrixForm=

$$\begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Metric in Newman-Penrose basis

We now transform the metric into the Newman-Penrose basis. Note that the diagonal elements vanish, showing that the basis is composed of null vectors.

In[ ]:=
```
ChangeComponents[Met[{-μ, -newpen}, {-ν, -newpen}], Met[{-μ, -coord}, {-ν, -coord}]];
```

Computed $g_{\alpha\nu} \to e_{\nu}{}^{\beta}\ g_{\alpha\beta}$ in 0.136436 Seconds

Computed $g_{\mu\nu} \to e_{\mu}{}^{\alpha}\ g_{\alpha\nu}$ in 0.130780 Seconds

In[∘]:=
```
ToValues[ToValues[ComponentArray[Met[{-μ, -newpen}, {-ν, -newpen}]]]];
AllComponentValues[Met[{-μ, -newpen}, {-ν, -newpen}], %];
```

In[∘]:=
```
ToValues[ComponentArray[Met[{-μ, -newpen}, {-ν, -newpen}]]] // MatrixForm
```

Out[∘]//MatrixForm=

$$\begin{pmatrix} 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Dependent tensors

Additionally, we compute tensors derived from the metric, such as the Riemann tensor and inverse metric.

In[∘]:=
```
MetricCompute[Met, coord, All]
```

** DefTensor: Defining weight +2 density DetMetcoord[]. Determinant.

** DefTensor: Defining tensor ChristoffelCDPDcoord[α, -β, -γ].

In[∘]:=
```
ChangeComponents[Met[{μ, newpen}, {ν, newpen}], Met[{μ, coord}, {ν, coord}]];
```

Computed $g^{\alpha\nu} \to e_{\beta}{}^{\nu}\ g^{\alpha\beta}$ in 0.171582 Seconds

Computed $g^{\mu\nu} \to e_{\alpha}{}^{\mu}\ g^{\alpha\nu}$ in 0.154904 Seconds

In[∘]:=
```
ToValues[ToValues[ComponentArray[Met[{μ, newpen}, {ν, newpen}]]]];
AllComponentValues[Met[{μ, newpen}, {ν, newpen}], %];
```

Wave covector

Next, we write the wave covector in coordinates, and transform it into the Newman-Penrose basis.

In[∘]:=
```
AllComponentValues[wav[{-μ, -coord}], {1, 0, 0, -1} freq]
```

Out[∘]=

FoldedRule[{}, {$k_t \to \omega$, $k_x \to 0$, $k_y \to 0$, $k_z \to -\omega$}]

In[∘]:=
```
ChangeComponents[wav[{-μ, -newpen}], wav[{-μ, -coord}]];
```

Computed $k_{\mu} \to e_{\mu}{}^{\alpha}\ k_{\alpha}$ in 0.041498 Seconds

In[ ]:=
```
ToValues[ToValues[ComponentArray[wav[{-μ, -newpen}]]]];
AllComponentValues[wav[{-μ, -newpen}], %]
```
Out[ ]=

$\text{FoldedRule}\left[\{\}, \left\{k_{l} \to 0,\ k_{n} \to \omega,\ k_{m} \to 0,\ k_{\bar{m}} \to 0\right\}\right]$

In[ ]:=
```
ChangeComponents[wav[{μ, coord}], wav[{-μ, -coord}]];
```
Computed $k^{\mu} \to g^{\mu\alpha}\ k_{\alpha}$ in 0.022574 Seconds

In[ ]:=
```
ToValues[ToValues[ComponentArray[wav[{μ, coord}]]]];
AllComponentValues[wav[{μ, coord}], %]
```
Out[ ]=

$\text{FoldedRule}\left[\{\}, \left\{k^{t} \to -\omega,\ k^{x} \to 0,\ k^{y} \to 0,\ k^{z} \to -\omega\right\}\right]$

In[ ]:=
```
ChangeComponents[wav[{μ, newpen}], wav[{-μ, -coord}]];
```
Computed $k^{\beta} \to g^{\beta\alpha}\ k_{\alpha}$ in 0.019325 Seconds

Computed $k^{\mu} \to e_{\alpha}{}^{\mu}\ k^{\alpha}$ in 0.028670 Seconds

In[ ]:=
```
ToValues[ToValues[ComponentArray[wav[{μ, newpen}]]]];
AllComponentValues[wav[{μ, newpen}], %]
```
Out[ ]=

$\text{FoldedRule}\left[\{\}, \left\{k^{l} \to -\omega,\ k^{n} \to 0,\ k^{m} \to 0,\ k^{\bar{m}} \to 0\right\}\right]$

Plane wave

Finally, for the plane wave we will have the property that its derivative is proportional to the wave covector.

In[ ]:=
```
{CD[-ρ][δMet[LI[1], -μ, -ν]] == wav[-ρ] × δMet[LI[1], -μ, -ν] I,
  CD[-ρ][scal[]] == wav[-ρ] × scal[] I, CD[-ρ][wav[-μ]] == 0}
plawa = Flatten[mkrg /@ %];
```
Out[ ]=

$\left\{\nabla_{\rho} h^{1}{}_{\mu\nu} == i\ k_{\rho}\ h^{1}{}_{\mu\nu},\ \nabla_{\rho}\psi == i\ \psi\ k_{\rho},\ \nabla_{\rho} k_{\mu} == 0\right\}$

Perturbative expansion and Newman-Penrose variables

Riemann tensor

Let us now calculate the Riemann tensor for a linear perturbation around the Minkowki metric, which is given by a plane wave. Here we also use the fact that the Riemann tensor of the background vanishes to simplify the calculation.

In[ ]:=
```
RiemannCD[-σ, -ρ, -ν, -μ]
Perturbation[%];
ExpandPerturbation[%];
% /. RiemannCD → Zero;
% //. plawa;
ContractMetric[%];
ToCanonical[%]
ToBasis[newpen][%];
ComponentArray[%];
ToCanonical[%];
ToValues[%];
AllComponentValues[
  RiemannCD[-{σ, newpen}, -{ρ, newpen}, -{ν, newpen}, -{μ, newpen}], %];
%[[2]]
```

Out[ ]=

$$R[\nabla]_{\sigma\rho\nu\mu}$$

Out[ ]=

$$\frac{1}{2} k_\nu\ k_\sigma\ h^1{}_{\mu\rho} - \frac{1}{2} k_\nu\ k_\rho\ h^1{}_{\mu\sigma} - \frac{1}{2} k_\mu\ k_\sigma\ h^1{}_{\nu\rho} + \frac{1}{2} k_\mu\ k_\rho\ h^1{}_{\nu\sigma}$$

Out[ ]=

$$\Big\{R[\nabla]_{lnln} \to \frac{1}{2}\omega^2\ h^1{}_{ll},\ R[\nabla]_{lnlm} \to 0,\ R[\nabla]_{lnl\bar{m}} \to 0,\ R[\nabla]_{lnnm} \to -\frac{1}{2}\omega^2\ h^1{}_{lm},$$
$$R[\nabla]_{lnn\bar{m}} \to -\frac{1}{2}\omega^2\ h^1{}_{l\bar{m}},\ R[\nabla]_{lnm\bar{m}} \to 0,\ R[\nabla]_{lmlm} \to 0,\ R[\nabla]_{lml\bar{m}} \to 0,\ R[\nabla]_{lmnm} \to 0,\ R[\nabla]_{lmn\bar{m}} \to 0,$$
$$R[\nabla]_{lmm\bar{m}} \to 0,\ R[\nabla]_{l\bar{m}l\bar{m}} \to 0,\ R[\nabla]_{l\bar{m}nm} \to 0,\ R[\nabla]_{l\bar{m}n\bar{m}} \to 0,\ R[\nabla]_{l\bar{m}m\bar{m}} \to 0,\ R[\nabla]_{nmnm} \to \frac{1}{2}\omega^2\ h^1{}_{mm},$$
$$R[\nabla]_{nmn\bar{m}} \to \frac{1}{2}\omega^2\ h^1{}_{m\bar{m}},\ R[\nabla]_{nmm\bar{m}} \to 0,\ R[\nabla]_{n\bar{m}n\bar{m}} \to \frac{1}{2}\omega^2\ h^1{}_{\bar{m}\bar{m}},\ R[\nabla]_{n\bar{m}m\bar{m}} \to 0,\ R[\nabla]_{m\bar{m}m\bar{m}} \to 0\Big\}$$

Newman-Penrose quantities

We then introduce the definition of the Newman-Penrose quantities. Note the appearance of a factor $-\omega^2$, indicating a second order derivative of the metric components.

In[ ]:=
```
{psi2[] == -RiemannCD[{0, -newpen}, {1, -newpen}, {0, -newpen}, {1, -newpen}]/6,
  psi3[] == -RiemannCD[{3, -newpen}, {1, -newpen}, {0, -newpen}, {1, -newpen}]/2,
  psi3b[] == -RiemannCD[{2, -newpen}, {1, -newpen}, {0, -newpen}, {1, -newpen}]/2,
  psi4[] == -RiemannCD[{3, -newpen}, {1, -newpen}, {3, -newpen}, {1, -newpen}],
  psi4b[] == -RiemannCD[{2, -newpen}, {1, -newpen}, {2, -newpen}, {1, -newpen}],
  phi22[] == -RiemannCD[{3, -newpen}, {1, -newpen}, {2, -newpen}, {1, -newpen}]}
npndefeq = ToValues[%]
```

Out[ ]=

$$\left\{\Psi_{ɣ} == -\frac{1}{6}\ R[\nabla]_{lnln},\ \Psi_{ɤ} == -\frac{1}{2}\ R[\nabla]_{\bar{m}nln},\ \bar{\Psi}_3 == -\frac{1}{2}\ R[\nabla]_{mnln},\ \Psi_4 == -R[\nabla]_{\bar{m}n\bar{m}n},\ \bar{\Psi}_{ч} == -R[\nabla]_{mnmn},\ \Phi_{ɣɣ} == -R[\nabla]_{\bar{m}nmn}\right\}$$

Out[ ]=

$$\left\{\Psi_{ɣ} == -\frac{1}{12}\ \omega^{ɣ}\ h^1{}_{ll},\ \Psi_{ɤ} == -\frac{1}{4}\ \omega^{ɣ}\ h^1{}_{l\bar{m}},\ \bar{\Psi}_3 == -\frac{1}{4}\ \omega^2\ h^1{}_{lm},\ \Psi_4 == -\frac{1}{2}\ \omega^2\ h^1{}_{\bar{m}\bar{m}},\ \bar{\Psi}_4 == -\frac{1}{2}\ \omega^2\ h^1{}_{mm},\ \Phi_{22} == -\frac{1}{2}\ \omega^2\ h^1{}_{m\bar{m}}\right\}$$

These are the potentials and the relevant components of the metric perturbation in the Newman-Penrose basis:

In[ ]:=
```
nppot = npndefeq[[All, 1]]
```

Out[ ]=

$$\{\Psi_2,\ \Psi_3,\ \bar{\Psi}_3,\ \Psi_4,\ \bar{\Psi}_4,\ \Phi_{22}\}$$

In[ ]:=
```
metpot = Select[Flatten[ComponentArray[δMet[LI[1], -{μ, newpen}, -{ν, newpen}]]],
   Not[FreeQ[npndefeq, #]] &]
```

Out[ ]=

$$\{h^1{}_{ll},\ h^1{}_{lm},\ h^1{}_{l\bar{m}},\ h^1{}_{mm},\ h^1{}_{m\bar{m}},\ h^1{}_{\bar{m}\bar{m}}\}$$

For convenience, we introduce some rules to convert between Newman-Penrose variables and metric components.

In[ ]:=
```
nptomet = Solve[npndefeq, nppot][[1]]
```

Out[ ]=

$$\left\{\Psi_2 \to -\frac{1}{12}\ \omega^2\ h^1{}_{ll},\ \Psi_3 \to -\frac{1}{4}\ \omega^2\ h^1{}_{l\bar{m}},\ \bar{\Psi}_3 \to -\frac{1}{4}\ \omega^2\ h^1{}_{lm},\ \Psi_4 \to -\frac{1}{2}\ \omega^2\ h^1{}_{\bar{m}\bar{m}},\ \bar{\Psi}_4 \to -\frac{1}{2}\ \omega^2\ h^1{}_{mm},\ \Phi_{22} \to -\frac{1}{2}\ \omega^2\ h^1{}_{m\bar{m}}\right\}$$

In[ ]:=
```
mettonp = Solve[npndefeq, metpot][[1]]
```

Out[ ]=

$$\left\{h^1{}_{ll} \to -\frac{12\ \Psi_2}{\omega^2},\ h^1{}_{lm} \to -\frac{4\ \bar{\Psi}_3}{\omega^2},\ h^1{}_{l\bar{m}} \to -\frac{4\ \Psi_3}{\omega^2},\ h^1{}_{mm} \to -\frac{2\ \bar{\Psi}_4}{\omega^2},\ h^1{}_{m\bar{m}} \to -\frac{2\ \Phi_{22}}{\omega^2},\ h^1{}_{\bar{m}\bar{m}} \to -\frac{2\ \Psi_4}{\omega^2}\right\}$$

Lorentz transformation

Ansatz for Lorentz transformation

Let us make the following ansatz for a Lorentz transformation:

In[ ]:=
```
AllComponentValues[lor[{μ, newpen}, -{ν, newpen}],
  Transpose[{{1, 0, 0, 0}, {ar^2 + ai^2, 1, Exp[I phi] (ar - I ai), Exp[-I phi] (ar + I ai)},
    {ar + I ai, 0, Exp[I phi], 0}, {ar - I ai, 0, 0, Exp[-I phi]}}]]
```

Out[ ]=

FoldedRule[{}, { $\Lambda^{l}{}_{l} \to 1$, $\Lambda^{l}{}_{n} \to \mathcal{I}\alpha^2 + \mathcal{R}\alpha^2$, $\Lambda^{l}{}_{m} \to i\,\mathcal{I}\alpha + \mathcal{R}\alpha$, $\Lambda^{l}{}_{\bar{m}} \to -i\,\mathcal{I}\alpha + \mathcal{R}\alpha$,
$\Lambda^{n}{}_{l} \to 0$, $\Lambda^{n}{}_{n} \to 1$, $\Lambda^{n}{}_{m} \to 0$, $\Lambda^{n}{}_{\bar{m}} \to 0$, $\Lambda^{m}{}_{l} \to 0$, $\Lambda^{m}{}_{n} \to (-i\,\mathcal{I}\alpha + \mathcal{R}\alpha)\, e^{i\phi}$,
$\Lambda^{m}{}_{m} \to e^{i\phi}$, $\Lambda^{m}{}_{\bar{m}} \to 0$, $\Lambda^{\bar{m}}{}_{l} \to 0$, $\Lambda^{\bar{m}}{}_{n} \to (i\,\mathcal{I}\alpha + \mathcal{R}\alpha)\, e^{-i\phi}$, $\Lambda^{\bar{m}}{}_{m} \to 0$, $\Lambda^{\bar{m}}{}_{\bar{m}} \to e^{-i\phi}$ }]

Check Lorentz transformations

Check that the ansatz preserves the metric:

In[ ]:=
```
Met[-ρ, -σ] × lor[ρ, -μ] × lor[σ, -ν] - Met[-μ, -ν]
ToBasis[newpen][%];
TraceBasisDummy[%];
ComponentArray[%];
ToCanonical[%];
ToValues[%];
Simplify[%]
```

Out[ ]=

$$- g_{\mu\nu} + \Lambda^{\rho}{}_{\mu}\, \Lambda^{\sigma}{}_{\nu}\, g_{\rho\sigma}$$

Out[ ]=

{{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}}

Check that the ansatz preserves the wave covector:

In[ ]:=
```
wav[-ν] × lor[ν, -μ] - wav[-μ]
ToBasis[newpen][%];
TraceBasisDummy[%];
ComponentArray[%];
ToCanonical[%];
ToValues[%];
Simplify[%]
```

Out[ ]=

$$-k_{\mu} + \Lambda^{\nu}{}_{\mu}\, k_{\nu}$$

Out[ ]=

```
{0, 0, 0, 0}
```

Lorentz transformation of Newman-Penrose quantities

Now we can transform the metric using the Lorentz transformation defined above.

In[ ]:=
```
δMet[LI[1], -ρ, -σ] × lor[ρ, -μ] × lor[σ, -ν]
ToBasis[newpen][%];
TraceBasisDummy[%];
ComponentArray[%];
ToCanonical[%];
ToValues[%];
tramet = Simplify[%];
```

Out[ ]=

$$\Lambda^{\rho}{}_{\mu}\, \Lambda^{\sigma}{}_{\nu}\, h^{1}{}_{\rho\sigma}$$

This is the transformation of the relevant metric components.

In[ ]:=
```
TableForm[
 metru = MapThread[Rule, {metpot, Extract[tramet, Join @@ (Position[ComponentArray[
        δMet[LI[1], -{μ, newpen}, -{ν, newpen}]], #] & /@ metpot)]}, 1]]
```

Out[ ]//TableForm=

$$h^{1}{}_{ll} \to h^{1}{}_{ll}$$
$$h^{1}{}_{lm} \to (i\,\mathcal{I}\alpha + \mathcal{R}\alpha)\, h^{1}{}_{ll} + e^{i\phi}\, h^{1}{}_{lm}$$
$$h^{1}{}_{l\bar{m}} \to (-i\,\mathcal{I}\alpha + \mathcal{R}\alpha)\, h^{1}{}_{ll} + e^{-i\phi}\, h^{1}{}_{l\bar{m}}$$
$$h^{1}{}_{mm} \to (i\,\mathcal{I}\alpha + \mathcal{R}\alpha)^{2}\, h^{1}{}_{ll} + 2\,(i\,\mathcal{I}\alpha + \mathcal{R}\alpha)\, e^{i\phi}\, h^{1}{}_{lm} + e^{2i\phi}\, h^{1}{}_{mm}$$
$$h^{1}{}_{m\bar{m}} \to (\mathcal{I}\alpha - i\,\mathcal{R}\alpha)(\mathcal{I}\alpha + i\,\mathcal{R}\alpha)\, h^{1}{}_{ll} + (-i\,\mathcal{I}\alpha + \mathcal{R}\alpha)\, e^{i\phi}\, h^{1}{}_{lm} + (i\,\mathcal{I}\alpha + \mathcal{R}\alpha)\, e^{-i\phi}\, h^{1}{}_{l\bar{m}} + h^{1}{}_{m\bar{m}}$$
$$h^{1}{}_{\bar{m}\bar{m}} \to (-i\,\mathcal{I}\alpha + \mathcal{R}\alpha)^{2}\, h^{1}{}_{ll} + 2\,(-i\,\mathcal{I}\alpha + \mathcal{R}\alpha)\, e^{-i\phi}\, h^{1}{}_{l\bar{m}} + e^{-2i\phi}\, h^{1}{}_{\bar{m}\bar{m}}$$

We can express the rules above also in terms of the Newman-Penrose variables.

```
In[∘]:= TableForm[npru = MapThread[Rule, {nppot, Simplify[nppot /. nptomet /. metru /. mettonp]}, 1]]
```

Out[∘]//TableForm=

$$\Psi_2 \to \Psi_2$$

$$\Psi_3 \to -3\, i\, \mathcal{I}\alpha\, \Psi_2 + 3\, \mathcal{R}\alpha\, \Psi_2 + e^{-i\phi}\, \Psi_3$$

$$\overline{\Psi}_3 \to 3\, i\, \mathcal{I}\alpha\, \Psi_2 + 3\, \mathcal{R}\alpha\, \Psi_2 + e^{i\phi}\, \overline{\Psi}_3$$

$$\Psi_4 \to -6\, \mathcal{I}\alpha^2\, \Psi_2 + 6\, \mathcal{R}\alpha^2\, \Psi_2 + 4\, \mathcal{R}\alpha\, e^{-i\phi}\, \Psi_3 - 4\, i\, \mathcal{I}\alpha \left(3\, \mathcal{R}\alpha\, \Psi_2 + e^{-i\phi}\, \Psi_3\right) + e^{-2 i\phi}\, \Psi_4$$

$$\overline{\Psi}_4 \to -6\, \mathcal{I}\alpha^2\, \Psi_2 + 6\, \mathcal{R}\alpha^2\, \Psi_2 + 4\, \mathcal{R}\alpha\, e^{i\phi}\, \overline{\Psi}_3 + 4\, i\, \mathcal{I}\alpha \left(3\, \mathcal{R}\alpha\, \Psi_2 + e^{i\phi}\, \overline{\Psi}_3\right) + e^{2 i\phi}\, \overline{\Psi}_4$$

$$\Phi_{22} \to \Phi_{22} + 6\left(\mathcal{I}\alpha^2 + \mathcal{R}\alpha^2\right)\Psi_2 + 2\left(i\, \mathcal{I}\alpha + \mathcal{R}\alpha\right) e^{-i\phi}\, \Psi_3 + 2\left(-i\, \mathcal{I}\alpha + \mathcal{R}\alpha\right) e^{i\phi}\, \overline{\Psi}_3$$

The transformation is governed by the following matrix.

```
In[∘]:= MatrixForm[Transpose[FullSimplify[Coefficient[npru[[All, 2]], #] &/@ nppot]]]
```

Out[∘]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 3\left(-i\, \mathcal{I}\alpha + \mathcal{R}\alpha\right) & e^{-i\phi} & 0 & 0 & 0 & 0 \\ 3\left(i\, \mathcal{I}\alpha + \mathcal{R}\alpha\right) & 0 & e^{i\phi} & 0 & 0 & 0 \\ -6\left(\mathcal{I}\alpha + i\, \mathcal{R}\alpha\right)^2 & 4\left(-i\, \mathcal{I}\alpha + \mathcal{R}\alpha\right) e^{-i\phi} & 0 & e^{-2 i\phi} & 0 & 0 \\ -6\left(\mathcal{I}\alpha - i\, \mathcal{R}\alpha\right)^2 & 0 & 4\left(i\, \mathcal{I}\alpha + \mathcal{R}\alpha\right) e^{i\phi} & 0 & e^{2 i\phi} & 0 \\ 6\left(\mathcal{I}\alpha^2 + \mathcal{R}\alpha^2\right) & 2\left(i\, \mathcal{I}\alpha + \mathcal{R}\alpha\right) e^{-i\phi} & 2\left(-i\, \mathcal{I}\alpha + \mathcal{R}\alpha\right) e^{i\phi} & 0 & 0 & 1 \end{pmatrix}$$

E(2) class of gravity theories

General relativity

Let us now find the E(2) class of general relativity. We use the fact that we can write the vacuum field equations as $R_{\mu\nu} = 0$:

In[ ]:=
```
RicciCD[-μ, -ν]
Perturbation[%];
ExpandPerturbation[%];
% /. RicciCD → Zero;
% //. plawa;
SeparateMetric[][%];
ToCanonical[%]
ToBasis[newpen][%];
TraceBasisDummy[%];
ComponentArray[%];
ToCanonical[%];
ric = ToValues[%]
```

Out[ ]=

$$R[\nabla]_{\mu\nu}$$

Out[ ]=

$$\frac{1}{2} g^{\alpha\beta} k_\mu k_\nu h^1{}_{\alpha\beta} - \frac{1}{2} k^\alpha k_\nu h^1{}_{\mu\alpha} + \frac{1}{2} g_{\alpha\beta} k^\alpha k^\beta h^1{}_{\mu\nu} - \frac{1}{2} k^\alpha k_\mu h^1{}_{\nu\alpha}$$

Out[ ]=

$$\left\{\left\{0, \frac{1}{2} \omega^2 h^1{}_{ll}, 0, 0\right\}, \left\{\frac{1}{2} \omega^2 h^1{}_{ll}, \omega^2 h^1{}_{m\bar{m}}, \frac{1}{2} \omega^2 h^1{}_{lm}, \frac{1}{2} \omega^2 h^1{}_{l\bar{m}}\right\}, \left\{0, \frac{1}{2} \omega^2 h^1{}_{lm}, 0, 0\right\}, \left\{0, \frac{1}{2} \omega^2 h^1{}_{l\bar{m}}, 0, 0\right\}\right\}$$

Demanding that the above vanishes, we obtain the following conditions on the Newman-Penrose variables:

In[ ]:=
```
Reduce[# == 0 & /@ Flatten[ric /. mettonp], nppot]
```

Out[ ]=

$$\Psi_2 == 0 \;\&\&\; \Psi_3 == 0 \;\&\&\; \overline{\Psi}_3 == 0 \;\&\&\; \Phi_{22} == 0$$

We see that only the two tensor modes are allowed, while all other modes must vanish. The E(2) class is therefore N_2.

Scalar-tensor gravity

We then calculate the E(2) class for scalar-tensor gravity. Let us first check that a plane wave satisfies the scalar field equation $\nabla^\mu \nabla_\mu \psi = 0$.

In[ ]:=
```
CD[μ][CD[-μ][scal[]]]
% //. plawa;
ToBasis[newpen][%];
TraceBasisDummy[%];
ToValues[%]
```

Out[ ]=

$\nabla^{\mu}\nabla_{\mu}\psi$

Out[ ]=

0

We continue with the metric field equation $R_{\mu\nu} = \nabla_{\mu}\nabla_{\nu}\psi$. We can use the fact that we already calculated the Ricci tensor.

In[ ]:=
```
CD[-μ][CD[-ν][scal[]]]
% //. plawa;
ToBasis[newpen][%];
TraceBasisDummy[%];
ComponentArray[%];
ToCanonical[%];
scaleq = ToValues[%] - ric
```

Out[ ]=

$\nabla_{\nu}\nabla_{\mu}\psi$

Out[ ]=

$$\left\{\left\{0, -\frac{1}{2}\omega^2 h^1{}_{ll}, 0, 0\right\}, \left\{-\frac{1}{2}\omega^2 h^1{}_{ll}, -\omega^2\psi - \omega^2 h^1{}_{m\bar{m}}, -\frac{1}{2}\omega^2 h^1{}_{lm}, -\frac{1}{2}\omega^2 h^1{}_{l\bar{m}}\right\}, \left\{0, -\frac{1}{2}\omega^2 h^1{}_{lm}, 0, 0\right\}, \left\{0, -\frac{1}{2}\omega^2 h^1{}_{l\bar{m}}, 0, 0\right\}\right\}$$

These are the conditions on the Newman-Penrose variables.

In[ ]:=
```
Reduce[# == 0 & /@ Flatten[scaleq /. mettonp], nppot]
```

Out[ ]=

$$\Psi_2 == 0 \;\&\&\; \Psi_3 == 0 \;\&\&\; \overline{\Psi}_3 == 0 \;\&\&\; \Phi_{22} == \frac{\omega^2\psi}{2}$$

We see that now the breathing mode Φ_{22} comes with the amplitude of the scalar wave, which may be non-vanishing. The E(2) class is therefore N_3.