

```
In[ ]:= Get@FileNameJoin@{NotebookDirectory[], "BlackHoleEquations.m"};
```

Stefan's modified black hole equations

Connection to Part III project

The purpose of this script is to give a concrete set of tools for extracting the reduced field equations in a static, spherical spacetime. With these tools, it is hoped that the remainder of the project time can be spent on analysis of the physics.

```
-----  
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 1.2.0, {2021, 10, 17}
```

```
Copyright (C) 2002-2021, 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.  
-----  
-----
```

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

```
Copyright (C) 2003-2020, 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.  
-----  
-----
```

```
Connecting to external linux executable...
```

```
Connection established.
```

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.
** 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

Package xAct`Invar` version 2.0.5, {2013, 7, 1}
Copyright (C) 2006–2020, J. M. Martin-Garcia,
D. Yllanes and R. Portugal, under the General Public License.
** DefConstantSymbol: Defining constant symbol sigma.
** DefConstantSymbol: Defining constant symbol dim.
** Option CurvatureRelations of DefCovD changed from True to False
** Variable \$CommuteCovDsOnScalars changed from True to False

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.

Package xAct`SymManipulator` version 0.9.5, {2021, 9, 14}
Copyright (C) 2011–2021, Thomas Bäckdahl, under the General Public License.

Package xAct`xTras` version 1.4.2, {2014, 10, 30}
Copyright (C) 2012–2014, Teake Nutma, under the General Public License.
** Variable \$CovDFormat changed from Postfix to Prefix
** Option CurvatureRelations of DefCovD changed from False to True

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.

 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.

** DefManifold: Defining manifold M4.
 ** DefVBundle: Defining vbundle TangentM4.
 ** DefTensor: Defining symmetric metric tensor G[-a, -b].
 ** DefTensor: Defining antisymmetric tensor epsilonG[-a, -a1, -b, -b1].
 ** DefTensor: Defining tetrametric TetraG[-a, -a1, -b, -b1].
 ** DefTensor: Defining tetrametric TetraG†[-a, -a1, -b, -b1].
 ** DefCovD: Defining covariant derivative CD[-a].
 ** DefTensor: Defining vanishing torsion tensor TorsionCD[a, -a1, -b].
 ** DefTensor: Defining symmetric Christoffel tensor ChristoffelCD[a, -a1, -b].
 ** DefTensor: Defining Riemann tensor RiemannCD[-a, -a1, -b, -b1].
 ** DefTensor: Defining symmetric Ricci tensor RicciCD[-a, -a1].
 ** 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[-a, -a1].
 ** DefTensor: Defining Weyl tensor WeylCD[-a, -a1, -b, -b1].
 ** DefTensor: Defining symmetric TFRicci tensor TFRicciCD[-a, -a1].
 ** 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 symmetrized Riemann tensor SymRiemannCD[-a, -a1, -b, -b1].
 ** DefTensor: Defining symmetric Schouten tensor SchoutenCD[-a, -a1].
 ** DefTensor: Defining symmetric cosmological Schouten tensor SchoutenCCCD[LI[], -a, -a1].

```

** DefTensor: Defining symmetric cosmological Einstein tensor EinsteinCCCD[LI[_], -a, -a1].
** DefCovD: Defining covariant derivative CD[-a]. to be symmetrizable
** DefTensor: Defining weight +2 density DetG[]. Determinant.
** DefParameter: Defining parameter PerturbationParameterG.
** DefTensor: Defining tensor PerturbationG[LI[order], -a, -a1].

```

Define a Planck mass.

```

** DefConstantSymbol: Defining constant symbol MPl.

```

Define the Schwarzschild functions.

```

** DefScalarFunction: Defining scalar function B1.
** DefScalarFunction: Defining scalar function B2.

```

Define the chart for spherical polar coordinates.

```

** DefChart: Defining chart SphericalPolar.
** DefTensor: Defining coordinate scalar ct[].
** DefTensor: Defining coordinate scalar cr[].
** DefTensor: Defining coordinate scalar ctheta[].
** DefTensor: Defining coordinate scalar cphi[].
** DefMapping: Defining mapping SphericalPolar.
** DefMapping: Defining inverse mapping iSphericalPolar.
** DefTensor: Defining mapping differential tensor diSphericalPolar[-a, iSphericalPolara].
** DefTensor: Defining mapping differential tensor dSphericalPolar[-a, SphericalPolara].
** DefBasis: Defining basis SphericalPolar. Coordinated basis.
** DefCovD: Defining parallel derivative PDSphericalPolar[-a].
** DefTensor: Defining vanishing torsion tensor TorsionPDSphericalPolar[a, -a1, -b].
** DefTensor: Defining symmetric Christoffel tensor
  ChristoffelPDSphericalPolar[a, -a1, -b].
** DefTensor: Defining vanishing Riemann tensor RiemannPDSphericalPolar[-a, -a1, -b, b1].
** DefTensor: Defining vanishing Ricci tensor RicciPDSphericalPolar[-a, -a1].
** DefTensor: Defining antisymmetric +1 density etaUpSphericalPolar[a, a1, b, b1].
** DefTensor: Defining antisymmetric -1 density etaDownSphericalPolar[-a, -a1, -b, -b1].

```

Set the components of the metric to those of Schwarzschild.

Added independent rule $\overset{\circ}{g}_{00} \rightarrow \Theta[r]^2$ for tensor G
 Added independent rule $\overset{\circ}{g}_{01} \rightarrow 0$ for tensor G
 Added independent rule $\overset{\circ}{g}_{02} \rightarrow 0$ for tensor G
 Added independent rule $\overset{\circ}{g}_{03} \rightarrow 0$ for tensor G
 Added dependent rule $\overset{\circ}{g}_{10} \rightarrow \overset{\circ}{g}_{01}$ for tensor G
 Added independent rule $\overset{\circ}{g}_{11} \rightarrow -\Omega[r]^2$ for tensor G
 Added independent rule $\overset{\circ}{g}_{12} \rightarrow 0$ for tensor G
 Added independent rule $\overset{\circ}{g}_{13} \rightarrow 0$ for tensor G
 Added dependent rule $\overset{\circ}{g}_{20} \rightarrow \overset{\circ}{g}_{02}$ for tensor G
 Added dependent rule $\overset{\circ}{g}_{21} \rightarrow \overset{\circ}{g}_{12}$ for tensor G
 Added independent rule $\overset{\circ}{g}_{22} \rightarrow -r^2$ for tensor G
 Added independent rule $\overset{\circ}{g}_{23} \rightarrow 0$ for tensor G
 Added dependent rule $\overset{\circ}{g}_{30} \rightarrow \overset{\circ}{g}_{03}$ for tensor G
 Added dependent rule $\overset{\circ}{g}_{31} \rightarrow \overset{\circ}{g}_{13}$ for tensor G
 Added dependent rule $\overset{\circ}{g}_{32} \rightarrow \overset{\circ}{g}_{23}$ for tensor G
 Added independent rule $\overset{\circ}{g}_{33} \rightarrow -r^2 \text{Sin}[\theta]^2$ for tensor G

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

** DefTensor: Defining tensor ChristoffelCDPDSphericalPolar[a, -a1, -b].

Just to check the line element which we built here, let's have a look!

$$\overset{\circ}{g}_{\alpha\beta} \tag{1}$$

$$\begin{pmatrix} \overset{\circ}{g}_{00} & \overset{\circ}{g}_{01} & \overset{\circ}{g}_{02} & \overset{\circ}{g}_{03} \\ \overset{\circ}{g}_{10} & \overset{\circ}{g}_{11} & \overset{\circ}{g}_{12} & \overset{\circ}{g}_{13} \\ \overset{\circ}{g}_{20} & \overset{\circ}{g}_{21} & \overset{\circ}{g}_{22} & \overset{\circ}{g}_{23} \\ \overset{\circ}{g}_{30} & \overset{\circ}{g}_{31} & \overset{\circ}{g}_{32} & \overset{\circ}{g}_{33} \end{pmatrix} \tag{2}$$

$$\begin{pmatrix} \Theta[r]^y & 0 & 0 & 0 \\ 0 & -\Omega[r]^y & 0 & 0 \\ 0 & 0 & -r^y & 0 \\ 0 & 0 & 0 & -r^y \sin[\theta]^y \end{pmatrix}$$

Now how about the inverse metric?

$$g^{\alpha\beta} \tag{4}$$

$$\begin{pmatrix} g^{00} & g^{01} & g^{02} & g^{03} \\ g^{10} & g^{11} & g^{12} & g^{13} \\ g^{20} & g^{21} & g^{22} & g^{23} \\ g^{30} & g^{31} & g^{32} & g^{33} \end{pmatrix} \tag{5}$$

$$\begin{pmatrix} \frac{x}{\Theta[r]^2} & 0 & 0 & 0 \\ 0 & -\frac{x}{\Omega[r]^2} & 0 & 0 \\ 0 & 0 & -\frac{x}{r^2} & 0 \\ 0 & 0 & 0 & -\frac{\cos[\theta]^2}{r^2} \end{pmatrix} \tag{6}$$

Define scalar functions of the radial coordinate scalar, and functions which will represent the ADM quantities.

```
** DefTensor: Defining tensor Phi[].
** DefTensor: Defining tensor Psi[].
** DefScalarFunction: Defining scalar function Psis.
** DefScalarFunction: Defining scalar function Phis.
```

Connection to Part III project

Something I omitted in the video tutorial above was a discussion of how to set up reduced radial-function components of a tensor field which is not the metric. There were no such tensors in the torsionful effective theory I gave to Oliver, but of course in MoND we have the unit-timelike vector field. How to work with this?

We wish to define a unit-timelike one-form.

```
** DefTensor: Defining tensor A[-a].
```

$$\mathcal{A}_\alpha \quad (7)$$

$$\{\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3\} \quad (8)$$

We know that this one-form has the property of being unit-timelike, and that the Killing algebra is spherical. So, we define a couple of scalar functions.

** DefScalarFunction: Defining scalar function A1.

** DefScalarFunction: Defining scalar function A2.

Added independent rule $\mathcal{A}_0 \rightarrow \Phi[r]$ for tensor A

Added independent rule $\mathcal{A}_1 \rightarrow \sqrt{-1 + \frac{\Phi[r]^2}{\Theta[r]^2}} \Omega[r]$ for tensor A

Added independent rule $\mathcal{A}_2 \rightarrow 0$ for tensor A

Added independent rule $\mathcal{A}_3 \rightarrow 0$ for tensor A

$$\mathcal{A}_\alpha \quad (9)$$

$$\{\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3\} \quad (10)$$

$$\left\{ \Phi[r], \sqrt{-1 + \frac{\Phi[r]^2}{\Theta[r]^2}} \Omega[r], 0, 0 \right\} \quad (11)$$

Now check that a simple expression is carried properly to the component form.

$$g^{\alpha\beta} (\dot{\nabla}_\alpha \phi) (\dot{\nabla}_\beta \psi) \quad (12)$$

$$-\frac{\phi'[r] \psi'[r]}{\Omega[r]^2} \quad (13)$$

Next something more advanced: we want to show that our one-form really does the job.

$$\mathcal{A}_\alpha \mathcal{A}^\alpha \quad (14)$$

$$\mathcal{A}_\alpha \mathcal{A}_{\alpha'} g^{\alpha\alpha'} \quad (15)$$

$$1 \quad (16)$$

Great. Now let's check that this can be used for more complex expressions.

$$\mathcal{A}_\alpha \mathcal{A}_\beta R[\overset{\circ}{\nabla}]^{\alpha\beta} \quad (17)$$

$$\mathcal{A}_\alpha \mathcal{A}_\beta g^{\alpha\alpha'} g^{\beta\beta'} R[\overset{\circ}{\nabla}]_{\alpha'\beta'} \quad (18)$$

$$\frac{2\Phi[r]^2 (\Omega[r]\Theta'[r] + \Theta[r]\Omega'[r]) + \Theta[r]^2 (-2\Theta[r]\Omega'[r] - r\Theta'[r]\Omega'[r] + \Omega[r]r\Theta''[r])}{\Theta[r]^3 \Omega[r]^3 r} \quad (19)$$

Connection to Part III project

Brilliant, that computation took a fraction of a second. So with these tools all the field equations can be reduced to ODEs in the radius within no more than a couple of hours' tinkering: it is basically just data entry.