StaticConformal

Tom Goodale et al

Date: 2005/10/17 10:23:48

Abstract

Base thorn to provide the variables for the static conformal factor

1 Purpose

This thorn provides the variables defining a static conformal factor which is used to transform the physical metric. If this thorn is active and the `ADMBase::metric_type` parameter is set to `static conformal`, then the `ADMBase::g...` variables are the conformal values as opposed to the physical values.

The transformation is

$$g_{ij}^{\text{physical}} = \psi^4 g_{ij}^{\text{conformal}}$$

The extrinsic curvature is not transformed.

Memory is provided for the conformal factor `psi`, its first derivatives `psix`, `psiy`, `psiz`, and its second derivatives `psixx`, `psixy`, `psixz`, `psiyy`, `psiyz`, and `psizz` depending on the setting of the `conformal_storage` parameter.

Note that the first and second "derivative" grid functions have an additional factor of $1/\psi$ normalisation since this is the most common use of the derivative. I.e., the grid functions are

$$\begin{aligned} \texttt{psi} &= \psi, \\ \texttt{psix} &= \psi_x/\psi, \qquad \text{etc} \\ \texttt{psixx} &= \psi_{ij}/\psi \qquad \text{etc} \end{aligned}$$

Thorns need to check the value of the grid scalar `conformal_state` to determine how many levels of these variables have actually been calculated before using the conformal factor:

`conformal_state=0` No conformal factor has been calculated — thorns may assume the conformal factor is 1 at all points. (I.e., the metric is physical.)

`conformal_state=1` The conformal factor has been calulated, but no derivatives.

`conformal_state=2` The conformal factor and its first derivatives have been calculated.

`conformal_state=3` The conformal factor and its first and second derivatives have been calculated.

Note that this means that if you only want to know whether `psi` contains the values for the conformal factor you can check for `conformal_state > 0`.

2 Utilities

`StaticConformal` provides aliased functions to convert between physical and conformal 3-metric values. It is very important to understand that these functions apply the conversion *in place*. That is, if `gxx` contains the conformal metric value, when the routine is exited it will now contain the physical metric

value. These functions *do not* change the value of `conformal_state` and should be used with due care. (These functions are for example used by some analysis thorns who work only with the physical metric, they apply the transformation on entry to the analysis routine and switch it back on exit).

Convert from conformal to physical:

```
subroutine ConfToPhysInPlace (nx, ny, nz,
                              psi,
                              gxx, gxy, gxz, gyy, gyz, gzz)
   implicit none
   CCTK_INT,                          intent(in)    :: nx, ny, nz
   CCTK_REAL, dimension(nx, ny, nz), intent(in)    :: psi
   CCTK_REAL, dimension(nx, ny, nz), intent(inout) :: gxx, gxy, gxz, gyy, gyz, gzz
end subroutine ConfToPhysInPlace
```

Convert from physical to conformal:

```
subroutine PhysToConfInPlace (nx, ny, nz,
                              psi,
                              gxx, gxy, gxz, gyy, gyz, gzz)
   implicit none
   CCTK_INT,                          intent(in)    :: nx, ny, nz
   CCTK_REAL, dimension(nx, ny, nz), intent(in)    :: psi
   CCTK_REAL, dimension(nx, ny, nz), intent(inout) :: gxx, gxy, gxz, gyy, gyz, gzz
end subroutine ConfToPhysInPlace
```

3 Comments

The `StaticConformal` thorn itself does not calculate any conformal factor, but does initialise the `conformal_state` variable to 0.