Introduction

In this notebook we will go through the basics of running and analysing `Cactus/Einstein Toolkit` data. This will be a very basic introduction. For more details, see for instance the notes An Introduction to the Einstein Toolkit.

Notebook setup

```
# this allows you to use "cd" in cells to change directories instead of requiring "%cd"
%automagic on

# override IPython's default %%bash to not buffer all output
from IPython.core.magic import register_cell_magic

# use python strings within bash cells
@register_cell_magic
def bash(line, cell): get_ipython().system(cell)

import os
import scrolldown
```

```
Automagic is ON, % prefix IS NOT needed for line magics.
```

```
<IPython.core.display.Javascript object>
```

Prerequisites

The very first step in running the Einstein Toolkit is obviously downloading and compiling it. The compiling stage is usually quite time-intensive, and sometimes requires some fiddling around (in particular on new clusters). So, in order to save time, for this workshop we're providing a Virtual Machine with an already compiled exectutable of the ET.

Obtaining the Einstein Toolkit

This part will not be necessary for this session (it is already provided in the Virtual Machine), but just for reference, the procedure to obtain the ET is to follow the instructions here: https://einsteintoolkit.org/download.html.

kuibit

`kuibit` is a very useful analysis tool. It can be obtained via

```
pip install --user -U kuibit==1.3.5 # requires Python3 version 3.6.1 or greater
```

Building the Einstein Toolkit

This part will also not be necessary for this session (since an already compiled executable is provided in the Virtual Machine), but just for reference, the typical procedure to compile an ET executable without using `Simfactory` is as follows. First we create a *configuration*, which in this example (and in the provided VM) is called `ET`:

```
cd Cactus
make ET-config options=<machine config file> THORNLIST=<thornlist>
```

The machine configuration file needs to be prepared for each individual machine. Several examples of known machines (with extension `.cfg`) can be found in the folder `Cactus/simfactory/mdb/optionlists/`. For regular laptops, the `generic.cfg` file typically works well. The specific configuration file used for the executable provided in the VM (for a Fedora 36 operating system) is provided in the VM as well. Once the configuration is done, the compilation process is simply

```
make -j <number of processes> ET
```

After this, if everything is compiled correctly, an executable called `cactus_ET` will be created under the folder `Cactus/exe/`. These steps need to be repeated for every different configuration (typically, with different thornlists) built. As mentioned above, this is not needed for this session and you will find the executable `cactus_ET` already in the `Cactus/exe/` folder.

For convenience, let us store a variable with the path to this executable:

```
HOME = os.environ['HOME']
BASEDIR = os.path.join(HOME, "./ET/Cactus")
EXE = os.path.join(BASEDIR, "exe/cactus_ET"); EXE

'/home/mzilhao/./dev/ET/Cactus/exe/cactus_ET'
```

Running `Hello World`

To test the configuration, let us run the `HelloWorld` parameter file. The command for running the ET is similar to that of other MPI executables,

```
mpirun -np <num procs> ./exe/cactus_ET <parameter file>
```

so for our particular case we do:

```
%%bash
export OMP_NUM_THREADS=1
mpirun -np 2 $EXE
$BASEDIR/arrangements/CactusExamples/HelloWorld/par/HelloWorld.par
```

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

       10
  1   0101       ************************
  01  1010 10      The Cactus Code V4.11.0
 1010 1101 011      www.cactuscode.org
  1001 100101    ************************
    00010101
     100011     (c) Copyright The Authors
      0100      GNU Licensed. No Warranty
      0101
----------------------------------------------------------------------
----------

Cactus version:    4.11.0
Compile date:      Aug 01 2022 (10:36:07)
Run date:          Nov 14 2022 (16:18:55+0100)
Run host:          relayer (pid=16322)
Working directory: /home/mzilhao/01-Projectos/2022-11_Meudon/apr
Executable:        /home/mzilhao/./dev/ET/Cactus/exe/cactus_ET
Parameter file:
/home/mzilhao/./dev/ET/Cactus/arrangements/CactusExamples/HelloWorld/
par/HelloWorld.par
----------------------------------------------------------------------
----------

Activating thorn Cactus...Success -> active implementation Cactus
Activation requested for
--->HelloWorld<---
Activating thorn HelloWorld...Success -> active implementation
helloworld
----------------------------------------------------------------------
----------
  if (recover initial data)
    Recover parameters
  endif

  Startup routines
    [CCTK_STARTUP]

  Startup routines which need an existing grid hierarchy
    [CCTK_WRAGH]
  Parameter checking routines
    [CCTK_PARAMCHECK]

  Initialisation
    if (NOT (recover initial data AND recovery_mode is 'strict'))
      [CCTK_PREREGRIDINITIAL]
      Set up grid hierarchy
      [CCTK_POSTREGRIDINITIAL]
      [CCTK_BASEGRID]
      [CCTK_INITIAL]
      [CCTK_POSTINITIAL]
      Initialise finer grids recursively
      Restrict from finer grids
      [CCTK_POSTRESTRICTINITIAL]
      [CCTK_POSTPOSTINITIAL]
      [CCTK_POSTSTEP]
    endif
    if (recover initial data)
      [CCTK_BASEGRID]
      [CCTK_RECOVER_VARIABLES]
      [CCTK_POST_RECOVER_VARIABLES]
    endif
    if (checkpoint initial data)
      [CCTK_CPINITIAL]
    endif
    if (analysis)
      [CCTK_ANALYSIS]
  endif
  Output grid variables

  do loop over timesteps
    [CCTK_PREREGRID]
    Change grid hierarchy
    [CCTK_POSTREGRID]
    Rotate timelevels
    iteration = iteration+1
    t = t+dt
    [CCTK_PRESTEP]
    [CCTK_EVOL]
      HelloWorld::HelloWorld: Print message to screen
    Evolve finer grids recursively
    Restrict from finer grids
    [CCTK_POSTRESTRICT]
    [CCTK_POSTSTEP]
    if (checkpoint)
      [CCTK_CHECKPOINT]
    endif
    if (analysis)
      [CCTK_ANALYSIS]
    endif
    Output grid variables
    enddo

  Termination routines
    [CCTK_TERMINATE]

  Shutdown routines
```

```
    [CCTK_SHUTDOWN]

  Routines run after changing the grid hierarchy:
    [CCTK_POSTREGRID]
----------------------------------------------------------------------
----------
----------------------------------------------------------------------
----------

INFO (HelloWorld): Hello World!
INFO (HelloWorld): Hello World!
INFO (HelloWorld): Hello World!
INFO (HelloWorld): Hello World!
INFO (HelloWorld): Hello World!
INFO (HelloWorld): Hello World!
INFO (HelloWorld): Hello World!
INFO (HelloWorld): Hello World!
INFO (HelloWorld): Hello World!
INFO (HelloWorld): Hello World!
----------------------------------------------------------------------
----------
Done.
```

The above command will run the example "HelloWorld.par" and display its log output. If you see

```
INFO (HelloWorld): Hello World!
```

it has run correctly.

Running the wave equation

Let us now run an example with the `WaveMoL` `thorn`

```
!mkdir parfiles
```

first we create a simple parameter file under the folder "parfiles"

```
%%writefile parfiles/gaussian-RK4-2.par
# gaussian-RK4.par
# Evolve the scalar wave equation with the RK4 integrator

ActiveThorns = "
   Boundary
   Carpet
   CarpetIOASCII
   CarpetIOBasic
   CarpetIOScalar
   CarpetLib
   CarpetReduce
   CartGrid3D
   CoordBase
   GenericFD
   IOUtil
   LoopControl
   ML_WaveToy
   MoL
   SymBase
   Time
"

Carpet::domain_from_coordbase = yes
CartGrid3D::type              = "coordbase"

CoordBase::domainsize = "minmax"
CoordBase::spacing    = "numcells"
CoordBase::xmin       = -15.0
CoordBase::ymin       = -5.0
CoordBase::zmin       = -5.0
CoordBase::xmax       = +5.0
CoordBase::ymax       = +5.0
CoordBase::zmax       = +5.0
CoordBase::ncells_x   = 100
CoordBase::ncells_y   = 50
CoordBase::ncells_z   = 50

CoordBase::boundary_size_x_lower = 2
CoordBase::boundary_size_y_lower = 2
CoordBase::boundary_size_z_lower = 2
CoordBase::boundary_size_x_upper = 2
CoordBase::boundary_size_y_upper = 2
CoordBase::boundary_size_z_upper = 2
Carpet::ghost_size               = 2

Cactus::cctk_itlast = 100

MoL::ODE_method             = "RK4"
MoL::MoL_Intermediate_Steps = 4
MoL::MoL_Num_Scratch_Levels = 1

Time::dtfac = 0.5


ML_WaveToy::initial_data = "Gaussian"
ML_WaveToy::WT_u_bound   = "newrad"
ML_WaveToy::WT_rho_bound = "newrad"


IO::out_dir      = $parfile
```

```
#IO::out_fileinfo = "none"

IOBasic::outInfo_every = 1
IOBasic::outInfo_vars  = "ML_WaveToy::u"

IOScalar::outScalar_reductions = "norm1 norm2 minimum maximum
norm_inf"
IOScalar::outScalar_every      = 1
IOScalar::outScalar_vars       = "ML_WaveToy::WT_u"

IOASCII::out1D_every = 1
IOASCII::out1D_vars  = "ML_WaveToy::WT_u ML_WaveToy::WT_rho
ML_WaveToy::WT_eps"

CarpetIOASCII::compact_format = yes
CarpetIOASCII::output_ghost_points = no

Writing parfiles/gaussian-RK4.par
```

and now we can run it, just like the "Hello World" example

```
%%bash
export OMP_NUM_THREADS=1
mpirun -np 2 $EXE parfiles/gaussian-RK4.par

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

       10
  1   0101       ************************
  01  1010 10      The Cactus Code V4.11.0
 1010 1101 011      www.cactuscode.org
  1001 100101    ************************
    00010101
     100011     (c) Copyright The Authors
      0100      GNU Licensed. No Warranty
      0101
----------------------------------------------------------------------
----------

Cactus version:    4.11.0
Compile date:      Aug 01 2022 (10:36:07)
Run date:          Nov 14 2022 (16:24:48+0100)
Run host:          relayer (pid=16590)
Working directory: /home/mzilhao/01-Projectos/2022-11_Meudon/apr
Executable:        /home/mzilhao/./dev/ET/Cactus/exe/cactus_ET
Parameter file:    parfiles/gaussian-RK4.par
----------------------------------------------------------------------
----------
```

```
Activating thorn Cactus...Success -> active implementation Cactus
Activation requested for
--->Boundary Carpet CarpetIOASCII CarpetIOBasic CarpetIOScalar
CarpetLib CarpetReduce CartGrid3D CoordBase GenericFD IOUtil
LoopControl ML_WaveToy MoL SymBase Time<---
Thorn Carpet requests automatic activation of MPI
Thorn Carpet requests automatic activation of Timers
Thorn CarpetLib requests automatic activation of Vectors
Thorn CarpetLib requests automatic activation of CycleClock
Thorn LoopControl requests automatic activation of hwloc
Thorn hwloc requests automatic activation of zlib
Activating thorn Boundary...Success -> active implementation boundary
Activating thorn Carpet...Success -> active implementation Driver
Activating thorn CarpetIOASCII...Success -> active implementation
IOASCII
Activating thorn CarpetIOBasic...Success -> active implementation
IOBasic
Activating thorn CarpetIOScalar...Success -> active implementation
IOScalar
Activating thorn CarpetLib...Success -> active implementation
CarpetLib
Activating thorn CarpetReduce...Success -> active implementation
reduce
Activating thorn CartGrid3D...Success -> active implementation grid
Activating thorn CoordBase...Success -> active implementation
CoordBase
Activating thorn CycleClock...Success -> active implementation
CycleClock
Activating thorn GenericFD...Success -> active implementation
GenericFD
Activating thorn hwloc...Success -> active implementation hwloc
Activating thorn IOUtil...Success -> active implementation IO
Activating thorn LoopControl...Success -> active implementation
LoopControl
Activating thorn ML_WaveToy...Success -> active implementation
ML_WaveToy
Activating thorn MoL...Success -> active implementation MethodOfLines
Activating thorn MPI...Success -> active implementation MPI
Activating thorn SymBase...Success -> active implementation SymBase
Activating thorn Time...Success -> active implementation time
Activating thorn Timers...Success -> active implementation Timers
Activating thorn Vectors...Success -> active implementation Vectors
Activating thorn zlib...Success -> active implementation zlib
----------------------------------------------------------------------
----------
  if (recover initial data)
    Recover parameters
  endif
  Startup routines
    [CCTK_STARTUP]
      Carpet::MultiModel_Startup: Multi-model Startup routine
      CycleClock::CycleClock_Setup: Set up CycleClock
      LoopControl::LC_setup: Set up LoopControl
      Timers::Timer_Startup: Prepare hierarchical timers
      Carpet::Driver_Startup: Startup routine
      CarpetReduce::CarpetReduceStartup: Startup routine
      CartGrid3D::SymmetryStartup: Register GH Extension for
GridSymmetry
      CoordBase::CoordBase_Startup: Register a GH extension to store
the coordinate system handles
      IOUtil::IOUtil_Startup: Startup routine
      CarpetIOASCII::CarpetIOASCIIStartup: [global] Startup routine
      CarpetIOScalar::CarpetIOScalarStartup: [global] Startup routine
      ML_WaveToy::ML_WaveToy_Startup: [meta] create banner
      MoL::MoL_Startup: Startup banner
      SymBase::SymBase_Startup: Register GH Extension for SymBase
      CarpetIOBasic::CarpetIOBasicStartup: [global] Startup routine
      Vectors::Vectors_Startup: Print startup message
      GROUP hwloc_startup: hwloc startup group
        hwloc::hwloc_version: Output hwloc version

  Startup routines which need an existing grid hierarchy
    [CCTK_WRAGH]
      Boundary::Boundary_RegisterBCs: [global] Register boundary
conditions that this thorn provides
      CartGrid3D::RegisterCartGrid3DCoords: [meta] Register
coordinates for the Cartesian grid
      MoL::MoL_SetupIndexArrays: Set up the MoL bookkeeping index
arrays
      MoL::MoL_SetScheduleStatus: [global] Set the flag so it is ok to
register with MoL
      GROUP MoL_Register: The group where physics thorns register
variables with MoL
        ML_WaveToy::ML_WaveToy_RegisterVars: [meta] Register Variables
for MoL
      MoL::MoL_ReportNumberVariables: [meta] Report how many of each
type of variable there are
      GROUP SymBase_Wrapper: Wrapper group for SymBase
        GROUP SymmetryRegister: Register your symmetries here
          CartGrid3D::CartGrid3D_RegisterSymmetryBoundaries: [meta]
Register symmetry boundaries
          ML_WaveToy::ML_WaveToy_RegisterSymmetries: [meta] register
symmetries
        SymBase::SymBase_Statistics: Print symmetry boundary face
descriptions
  Parameter checking routines
    [CCTK_PARAMCHECK]
```

```
      Boundary::Boundary_Check: Check dimension of grid variables
      Carpet::CarpetParamCheck: Parameter checking routine
      CarpetLib::CarpetLib_test_prolongate_3d_rf2: [global] Test
prolongation operators
      CartGrid3D::ParamCheck_CartGrid3D: Check coordinates for
CartGrid3D
      MoL::MoL_ParamCheck: Basic parameter checking
      Vectors::Vectors_Test: Run correctness tests.

  Initialisation
    if (NOT (recover initial data AND recovery_mode is 'strict'))
      [CCTK_PREREGRIDINITIAL]
      Set up grid hierarchy
      [CCTK_POSTREGRIDINITIAL]
        CartGrid3D::SpatialCoordinates: Set Coordinates after
regridding
        GROUP MaskBase_SetupMask: Set up the weight function
          GROUP MaskBase_SetupMaskAll: Set up the weight function
            CarpetReduce::MaskBase_AllocateMask: [global] Allocate the
weight function
            CarpetReduce::MaskBase_InitMask: [global] [loop-local]
Initialise the weight function
            GROUP SetupIMaskInternal: Set up the integer weight
function (schedule other routines in here)
              CarpetReduce::CoordBase_SetupMask: [global] [loop-local]
Set up the outer boundaries of the weight function
              CarpetReduce::CarpetMaskSetup: [global] [loop-singlemap]
Set up the weight function for the restriction regions
            GROUP SetupIMask: Set up the integer weight function
(schedule other routines in here)
            CarpetReduce::MaskBase_SetMask: [global] [loop-local] Set
the weight function
            GROUP SetupMask: Set up the real weight function (schedule
other routines in here)
            CarpetReduce::MaskBase_TestMask: [global] Test the weight
function
        GROUP MoL_PseudoEvolutionBoundaries: Apply boundary conditions
to pseudo-evolved quantities
      [CCTK_BASEGRID]
        CartGrid3D::SpatialSpacings: Set up ranges for spatial 3D
Cartesian coordinates (on all grids)
        CartGrid3D::SpatialCoordinates: Set up spatial 3D Cartesian
coordinates on the GH
        GROUP MaskBase_SetupMask: Set up the weight function
          GROUP MaskBase_SetupMaskAll: Set up the weight function
            CarpetReduce::MaskBase_AllocateMask: [global] Allocate the
weight function
            CarpetReduce::MaskBase_InitMask: [global] [loop-local]
Initialise the weight function
```

```
            GROUP SetupIMaskInternal: Set up the integer weight function (schedule other routines in here)
              CarpetReduce::CoordBase_SetupMask: [global] [loop-local] Set up the outer boundaries of the weight function
              CarpetReduce::CarpetMaskSetup: [global] [loop-singlemap] Set up the weight function for the restriction regions
            GROUP SetupIMask: Set up the integer weight function (schedule other routines in here)
            CarpetReduce::MaskBase_SetMask: [global] [loop-local] Set the weight function
            GROUP SetupMask: Set up the real weight function (schedule other routines in here)
            CarpetReduce::MaskBase_TestMask: [global] Test the weight function
        ML_WaveToy::ML_WaveToy_CheckBoundaries: [meta] check boundaries treatment
        SymBase::SymBase_Check: Check whether the driver set up the grid consistently
        Time::Time_Initialise: [global] Initialise Time variables
        Time::TemporalSpacings: [singlemap] Set timestep based on Courant condition (courant_static)
      [CCTK_INITIAL]
        CarpetIOASCII::CarpetIOASCIIInit: [global] Initialisation routine
        CarpetIOBasic::CarpetIOBasicInit: [global] Initialisation routine
        CarpetIOScalar::CarpetIOScalarInit: [global] Initialisation routine
        ML_WaveToy::WT_Gaussian: WT_Gaussian
        MoL::MoL_StartLoop: [level] Initialise the step size control
      [CCTK_POSTINITIAL]
        GROUP MoL_PostStepModify: The group for physics thorns to schedule enforcing constraints
        GROUP MoL_PostStep: Ensure that everything is correct after the initial data have been set up
          ML_WaveToy::ML_WaveToy_SelectBoundConds: [level] select boundary conditions
          GROUP ML_WaveToy_ApplyBCs: Apply boundary conditions controlled by thorn Boundary
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested local physical boundary conditions
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary conditions
            Boundary::Boundary_ClearSelection: [level] Unselect all grid variables for boundary conditions
        GROUP MoL_PseudoEvolution: Calculate pseudo-evolved quantities
      Initialise finer grids recursively
      Restrict from finer grids
      [CCTK_POSTRESTRICTINITIAL]
        GROUP MoL_PostStep: Ensure that everything is correct after
restriction
          ML_WaveToy::ML_WaveToy_SelectBoundConds: [level] select
boundary conditions
          GROUP ML_WaveToy_ApplyBCs: Apply boundary conditions
controlled by thorn Boundary
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
            Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
        GROUP MoL_PseudoEvolutionBoundaries: Apply boundary conditions
to pseudo-evolved quantities
      [CCTK_POSTPOSTINITIAL]
      [CCTK_POSTSTEP]
    endif
    if (recover initial data)
      [CCTK_BASEGRID]
        CartGrid3D::SpatialSpacings: Set up ranges for spatial 3D
Cartesian coordinates (on all grids)
        CartGrid3D::SpatialCoordinates: Set up spatial 3D Cartesian
coordinates on the GH
        GROUP MaskBase_SetupMask: Set up the weight function
          GROUP MaskBase_SetupMaskAll: Set up the weight function
            CarpetReduce::MaskBase_AllocateMask: [global] Allocate the
weight function
            CarpetReduce::MaskBase_InitMask: [global] [loop-local]
Initialise the weight function
            GROUP SetupIMaskInternal: Set up the integer weight
function (schedule other routines in here)
              CarpetReduce::CoordBase_SetupMask: [global] [loop-local]
Set up the outer boundaries of the weight function
              CarpetReduce::CarpetMaskSetup: [global] [loop-singlemap]
Set up the weight function for the restriction regions
            GROUP SetupIMask: Set up the integer weight function
(schedule other routines in here)
            CarpetReduce::MaskBase_SetMask: [global] [loop-local] Set
the weight function
            GROUP SetupMask: Set up the real weight function (schedule
other routines in here)
            CarpetReduce::MaskBase_TestMask: [global] Test the weight
function
        ML_WaveToy::ML_WaveToy_CheckBoundaries: [meta] check
boundaries treatment
        SymBase::SymBase_Check: Check whether the driver set up the
grid consistently
```

```
        Time::Time_Initialise: [global] Initialise Time variables
        Time::TemporalSpacings: [singlemap] Set timestep based on
Courant condition (courant_static)
      [CCTK_RECOVER_VARIABLES]
      [CCTK_POST_RECOVER_VARIABLES]
        GROUP MaskBase_SetupMask: Set up the weight function
          GROUP MaskBase_SetupMaskAll: Set up the weight function
            CarpetReduce::MaskBase_AllocateMask: [global] Allocate the
weight function
            CarpetReduce::MaskBase_InitMask: [global] [loop-local]
Initialise the weight function
            GROUP SetupIMaskInternal: Set up the integer weight
function (schedule other routines in here)
              CarpetReduce::CoordBase_SetupMask: [global] [loop-local]
Set up the outer boundaries of the weight function
              CarpetReduce::CarpetMaskSetup: [global] [loop-singlemap]
Set up the weight function for the restriction regions
            GROUP SetupIMask: Set up the integer weight function
(schedule other routines in here)
            CarpetReduce::MaskBase_SetMask: [global] [loop-local] Set
the weight function
            GROUP SetupMask: Set up the real weight function (schedule
other routines in here)
            CarpetReduce::MaskBase_TestMask: [global] Test the weight
function
        GROUP MoL_PostStep: Ensure that everything is correct after
recovery
          ML_WaveToy::ML_WaveToy_SelectBoundConds: [level] select
boundary conditions
          GROUP ML_WaveToy_ApplyBCs: Apply boundary conditions
controlled by thorn Boundary
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
            Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
    endif
    if (checkpoint initial data)
      [CCTK_CPINITIAL]
    endif
    if (analysis)
      [CCTK_ANALYSIS]
        CarpetLib::CarpetLib_printtimestats: [global] Print timing
statistics if desired
        CarpetLib::CarpetLib_printmemstats: [global] Print memory
statistics if desired
        LoopControl::LC_statistics_analysis: [meta] Output LoopControl
```

```
statistics
        ML_WaveToy::WT_Dirichlet: WT_Dirichlet
        ML_WaveToy::WT_Energy: WT_Energy
        ML_WaveToy::WT_EnergyBoundary: WT_EnergyBoundary
  endif
  Output grid variables

  do loop over timesteps
    [CCTK_PREREGRID]
    Change grid hierarchy
    [CCTK_POSTREGRID]
      CartGrid3D::SpatialCoordinates: Set Coordinates after regridding
      GROUP MaskBase_SetupMask: Set up the weight function
        GROUP MaskBase_SetupMaskAll: Set up the weight function
          CarpetReduce::MaskBase_AllocateMask: [global] Allocate the
weight function
          CarpetReduce::MaskBase_InitMask: [global] [loop-local]
Initialise the weight function
          GROUP SetupIMaskInternal: Set up the integer weight function
(schedule other routines in here)
            CarpetReduce::CoordBase_SetupMask: [global] [loop-local]
Set up the outer boundaries of the weight function
            CarpetReduce::CarpetMaskSetup: [global] [loop-singlemap]
Set up the weight function for the restriction regions
          GROUP SetupIMask: Set up the integer weight function
(schedule other routines in here)
          CarpetReduce::MaskBase_SetMask: [global] [loop-local] Set
the weight function
          GROUP SetupMask: Set up the real weight function (schedule
other routines in here)
          CarpetReduce::MaskBase_TestMask: [global] Test the weight
function
      GROUP MoL_PostStep: Ensure that everything is correct after
regridding
        ML_WaveToy::ML_WaveToy_SelectBoundConds: [level] select
boundary conditions
        GROUP ML_WaveToy_ApplyBCs: Apply boundary conditions
controlled by thorn Boundary
          GROUP BoundaryConditions: Execute all boundary conditions
            Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
            CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
          Boundary::Boundary_ClearSelection: [level] Unselect all grid
variables for boundary conditions
      GROUP MoL_PseudoEvolutionBoundaries: Apply boundary conditions
to pseudo-evolved quantities
    Rotate timelevels
    iteration = iteration+1
    t = t+dt
    [CCTK_PRESTEP]
      LoopControl::LC_steer: [meta] Update LoopControl algorithm
preferences
    [CCTK_EVOL]
      MoL::MoL_StartLoop: [level] Initialise the step size control
      while (MoL::MoL_Stepsize_Bad)
        GROUP MoL_Evolution: A single Cactus evolution step using MoL
          GROUP MoL_StartStep: MoL internal setup for the evolution
step
            MoL::MoL_SetCounter: [level] Set the counter for the ODE
method to loop over
            MoL::MoL_SetTime: [level] Ensure the correct time and
timestep are used
            MoL::MoL_AllocateScratchSpace: [level] Allocate storage
for scratch levels
          GROUP MoL_PreStep: Physics thorns can schedule preloop setup
routines in here
          MoL::MoL_AllocateScratch: Allocate sufficient space for
array scratch variables
          MoL::MoL_InitialCopy: Ensure the data is in the correct
timelevel
          while (MoL::MoL_Intermediate_Step)
            GROUP MoL_Step: The loop over the intermediate steps for
the ODE integrator
              MoL::MoL_InitRHS: Initialise the RHS functions
              GROUP MoL_CalcRHS: Physics thorns schedule the
calculation of the discrete spatial operator in here
                ML_WaveToy::WT_RHS: WT_RHS
                ML_WaveToy::WT_Dirichlet: WT_Dirichlet
              GROUP MoL_PostRHS: Modify RHS functions
              GROUP MoL_RHSBoundaries: Any 'final' modifications to
the RHS functions (boundaries etc.)
              MoL::MoL_Add: Updates calculated with the efficient
Runge-Kutta 4 method
              MoL::MoL_DecrementCounter: [level] Alter the counter
number
              MoL::MoL_ResetTime: [level] If necessary, change the
time
              GROUP MoL_PostStepModify: The group for physics thorns
to schedule enforcing constraints
              GROUP MoL_PostStep: The group for physics thorns to
schedule boundary calls etc.
                ML_WaveToy::ML_WaveToy_SelectBoundConds: [level]
select boundary conditions
                GROUP ML_WaveToy_ApplyBCs: Apply boundary conditions
controlled by thorn Boundary
                  GROUP BoundaryConditions: Execute all boundary
conditions
```

```
                    Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                    CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                  Boundary::Boundary_ClearSelection: [level] Unselect
all grid variables for boundary conditions
              MoL::MoL_ResetDeltaTime: [level] If necessary, change
the timestep
          end while
          MoL::MoL_FinishLoop: [level] Control the step size
          MoL::MoL_RestoreSandR: Restoring the Save and Restore
variables to the original state
          MoL::MoL_FreeScratchSpace: [level] Free storage for scratch
levels
      end while
      GROUP MoL_PseudoEvolution: Calculate pseudo-evolved quantities
    Evolve finer grids recursively
    Restrict from finer grids
    [CCTK_POSTRESTRICT]
      GROUP MoL_PostStep: Ensure that everything is correct after
restriction
        ML_WaveToy::ML_WaveToy_SelectBoundConds: [level] select
boundary conditions
        GROUP ML_WaveToy_ApplyBCs: Apply boundary conditions
controlled by thorn Boundary
          GROUP BoundaryConditions: Execute all boundary conditions
            Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
            CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
          Boundary::Boundary_ClearSelection: [level] Unselect all grid
variables for boundary conditions
      GROUP MoL_PseudoEvolutionBoundaries: Apply boundary conditions
to pseudo-evolved quantities
    [CCTK_POSTSTEP]
    if (checkpoint)
      [CCTK_CHECKPOINT]
    endif
    if (analysis)
      [CCTK_ANALYSIS]
      CarpetLib::CarpetLib_printtimestats: [global] Print timing
statistics if desired
      CarpetLib::CarpetLib_printmemstats: [global] Print memory
statistics if desired
      LoopControl::LC_statistics_analysis: [meta] Output LoopControl
statistics
      ML_WaveToy::WT_Dirichlet: WT_Dirichlet
      ML_WaveToy::WT_Energy: WT_Energy
      ML_WaveToy::WT_EnergyBoundary: WT_EnergyBoundary
    endif
    Output grid variables
    enddo

  Termination routines
    [CCTK_TERMINATE]
      LoopControl::LC_statistics_terminate: [meta] Output LoopControl
statistics
      MoL::MoL_FreeIndexArrays: Free the MoL bookkeeping index arrays

  Shutdown routines
    [CCTK_SHUTDOWN]
      Timers::Timer_Shutdown: Prepare hierarchical timers

  Routines run after changing the grid hierarchy:
    [CCTK_POSTREGRID]
      CartGrid3D::SpatialCoordinates: Set Coordinates after regridding
      GROUP MaskBase_SetupMask: Set up the weight function
        GROUP MaskBase_SetupMaskAll: Set up the weight function
          CarpetReduce::MaskBase_AllocateMask: [global] Allocate the
weight function
          CarpetReduce::MaskBase_InitMask: [global] [loop-local]
Initialise the weight function
          GROUP SetupIMaskInternal: Set up the integer weight function
(schedule other routines in here)
            CarpetReduce::CoordBase_SetupMask: [global] [loop-local]
Set up the outer boundaries of the weight function
            CarpetReduce::CarpetMaskSetup: [global] [loop-singlemap]
Set up the weight function for the restriction regions
          GROUP SetupIMask: Set up the integer weight function
(schedule other routines in here)
          CarpetReduce::MaskBase_SetMask: [global] [loop-local] Set
the weight function
          GROUP SetupMask: Set up the real weight function (schedule
other routines in here)
          CarpetReduce::MaskBase_TestMask: [global] Test the weight
function
      GROUP MoL_PostStep: Ensure that everything is correct after
regridding
        ML_WaveToy::ML_WaveToy_SelectBoundConds: [level] select
boundary conditions
        GROUP ML_WaveToy_ApplyBCs: Apply boundary conditions
controlled by thorn Boundary
          GROUP BoundaryConditions: Execute all boundary conditions
            Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
            CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
          Boundary::Boundary_ClearSelection: [level] Unselect all grid
variables for boundary conditions
```

```
      GROUP MoL_PseudoEvolutionBoundaries: Apply boundary conditions
to pseudo-evolved quantities
----------------------------------------------------------------------
----------
INFO (Carpet): Multi-Model listing:
   model 0: "world"
INFO (Carpet): Multi-Model process distribution:
   processes 0-1: model 0 "world"
INFO (Carpet): Multi-Model: This is process 0, model 0 "world"
Current core file size limit: hard=[unlimited], soft=[unlimited]
Current addres space size limit: hard=[unlimited], soft=[unlimited]
Current data segment size limit: hard=[unlimited], soft=[unlimited]
Current resident set size limit: hard=[unlimited], soft=[unlimited]

INFO (CycleClock): Measuring CycleClock tick via OpenMP...
INFO (CycleClock): Calibrated CycleClock: 0.501999 ns per clock tick
(1.99204 GHz)
INFO (Vectors): Using vector size 1 for architecture scalar (no
vectorisation, 64-bit precision)
INFO (hwloc): library version 2.5.0, API version 0x20500
----------------------------------------------------------------------
----------
AMR driver provided by Carpet
----------------------------------------------------------------------
----------
AMR 0D ASCII I/O provided by CarpetIOASCII
----------------------------------------------------------------------
----------
AMR 1D ASCII I/O provided by CarpetIOASCII
----------------------------------------------------------------------
----------
AMR 2D ASCII I/O provided by CarpetIOASCII
----------------------------------------------------------------------
----------
AMR 3D ASCII I/O provided by CarpetIOASCII
----------------------------------------------------------------------
----------
AMR scalar I/O provided by CarpetIOScalar
----------------------------------------------------------------------
----------
ML_WaveToy
----------------------------------------------------------------------
----------
MoL: Generalized time integration.
----------------------------------------------------------------------
----------
AMR info I/O provided by CarpetIOBasic
----------------------------------------------------------------------
----------
```

```
INFO (Carpet): MPI is enabled
INFO (Carpet): Carpet is running on 2 processes
INFO (Carpet): This is process 0
INFO (Carpet): OpenMP is enabled
INFO (Carpet): This process contains 2 threads, this is thread 0
INFO (Carpet): There are 4 threads in total
INFO (Carpet): There are 2 threads per process
INFO (Carpet): This process runs on host relayer, pid=16590
INFO (Carpet): This process runs on 2 cores: 0, 4
INFO (Carpet): Thread 0 runs on 2 cores: 0, 4
INFO (Carpet): Thread 1 runs on 2 cores: 0, 4
INFO (Carpet): This simulation is running in 3 dimensions
INFO (Carpet): Boundary specification for map 0:
   nboundaryzones: [[2,2,2],[2,2,2]]
   is_internal   : [[0,0,0],[0,0,0]]
   is_staggered  : [[0,0,0],[0,0,0]]
   shiftout      : [[0,0,0],[0,0,0]]
INFO (Carpet): CoordBase domain specification for map 0:
   physical extent: [-5,-5,-5] : [5,5,5]   ([10,10,10])
   interior extent: [-4.8,-4.8,-4.8] : [4.8,4.8,4.8]   ([9.6,9.6,9.6])
   exterior extent: [-5.2,-5.2,-5.2] : [5.2,5.2,5.2]
([10.4,10.4,10.4])
   base_spacing   : [0.2,0.2,0.2]
INFO (Carpet): Adapted domain specification for map 0:
   convergence factor: 2
   convergence level : 0
   physical extent   : [-5,-5,-5] : [5,5,5]   ([10,10,10])
   interior extent   : [-4.8,-4.8,-4.8] : [4.8,4.8,4.8]
([9.6,9.6,9.6])
   exterior extent   : [-5.2,-5.2,-5.2] : [5.2,5.2,5.2]
([10.4,10.4,10.4])
   spacing           : [0.2,0.2,0.2]
INFO (Carpet): Base grid specification for map 0:
   number of grid points             : [53,53,53]
   number of coarse grid ghost points: [[2,2,2],[2,2,2]]
INFO (Carpet): Buffer zone counts (excluding ghosts):
   [0]: [[0,0,0],[0,0,0]]
INFO (Carpet): Overlap zone counts:
   [0]: [[0,0,0],[0,0,0]]
INFO (Carpet): Group and variable statistics:
INFO (Carpet):    There are 449 grid functions in 17 groups
INFO (Carpet):    There are 66 grid scalars in 18 groups
INFO (Carpet):    There are 11 1-dimensional grid arrays in 4 groups
INFO (Carpet):    There are 1 2-dimensional grid arrays in 1 groups
INFO (Carpet):    There are 0 3-dimensional grid arrays in 0 groups
INFO (Carpet):    (The number of variables counts all time levels)
INFO (CarpetIOASCII): I/O Method 'IOASCII_0D' registered: 0D AMR
output of grid variables to ASCII files
INFO (CarpetIOASCII): I/O Method 'IOASCII_1D' registered: 1D AMR
```

```
output of grid variables to ASCII files
INFO (CarpetIOASCII): Periodic 1D AMR output requested for:
   ML_WAVETOY::u
   ML_WAVETOY::rho
   ML_WAVETOY::eps
INFO (CarpetIOASCII): I/O Method 'IOASCII_2D' registered: 2D AMR
output of grid variables to ASCII files
INFO (CarpetIOASCII): I/O Method 'IOASCII_3D' registered: 3D AMR
output of grid variables to ASCII files
INFO (CarpetIOScalar): Periodic scalar output requested for:
   ML_WAVETOY::u
INFO (MoL): Using Runge-Kutta 4 as the time integrator.
INFO (MoL): The maximum number of evolved variables is 123. 2 are
registered.
INFO (MoL): The maximum number of slow evolved variables is 123. 0 are
registered.
INFO (MoL): The maximum number of constrained variables is 123. 0 are
registered.
INFO (MoL): The maximum number of SandR variables is 123. 0 are
registered.
INFO (MoL): The maximum number of evolved array variables is 123. 0
are registered.
INFO (MoL): The maximum number of constrained array variables is 123.
0 are registered.
INFO (MoL): The maximum number of SandR array variables is 123. 0 are
registered.
INFO (MoL): The maximum size of any array variables is 0.
INFO (Vectors): Testing vectorisation... [errors may result in
segfaults]
INFO (Vectors): 101/101 tests passed
INFO (CartGrid3D): Grid Spacings:
INFO (CartGrid3D): dx=>2.0000000e-01  dy=>2.0000000e-01
dz=>2.0000000e-01
INFO (CartGrid3D): Computational Coordinates:
INFO (CartGrid3D): x=>[-5.200, 5.200]  y=>[-5.200, 5.200]  z=>[-5.200,
5.200]
INFO (CartGrid3D): Indices of Physical Coordinates:
INFO (CartGrid3D): x=>[0,52]  y=>[0,52]  z=>[0,52]
INFO (Time): Timestep set to 0.1 (courant_static)
-----------------------------------------------
Iteration      Time |              ML_WAVETOY::u
                    |      minimum      maximum
-----------------------------------------------
        0     0.000 | 5.175555e-17    1.0000000
        1     0.100 | 5.175555e-17    0.9850663
        2     0.200 | 5.175555e-17    0.9410074
        3     0.300 | 5.175555e-17    0.8699936
        4     0.400 | 5.175555e-17    0.7754760
        5     0.500 | 5.175555e-17    0.6619523
        6     0.600 | 5.175555e-17    0.5346690
        7     0.700 | 5.175555e-17    0.3992833
        8     0.800 | 5.175555e-17    0.2782196
        9     0.900 | 5.175555e-17    0.2172637
       10     1.000 | 5.175555e-17    0.1844159
       11     1.100 |   -0.1146476    0.1644039
       12     1.200 |   -0.2141817    0.1509274
       13     1.300 |   -0.2964451    0.1410715
       14     1.400 |   -0.3603860    0.1333093
       15     1.500 |   -0.4059329    0.1268877
       16     1.600 |   -0.4338761    0.1213124
       17     1.700 |   -0.4457075    0.1163501
       18     1.800 |   -0.4434385    0.1118611
       19     1.900 |   -0.4294114    0.1077410
-----------------------------------------------
Iteration      Time |             ML_WAVETOY::u
                    |      minimum      maximum
-----------------------------------------------
       20     2.000 |   -0.4061180    0.1039254
       21     2.100 |   -0.3760395    0.1003960
       22     2.200 |   -0.3415142    0.0970919
       23     2.300 |   -0.3046379    0.0940191
       24     2.400 |   -0.2692337    0.0911307
       25     2.500 |   -0.2416508    0.0884202
       26     2.600 |   -0.2198471    0.0858689
       27     2.700 |   -0.2021654    0.0834559
       28     2.800 |   -0.1875220    0.0811847
       29     2.900 |   -0.1751271    0.0790232
       30     3.000 |   -0.1644791    0.0769933
       31     3.100 |   -0.1551767    0.0750550
       32     3.200 |   -0.1469914    0.0732159
       33     3.300 |   -0.1396946    0.0714696
       34     3.400 |   -0.1331414    0.0697998
       35     3.500 |   -0.1272295    0.0682091
       36     3.600 |   -0.1218532    0.0666793
       37     3.700 |   -0.1170254    0.0651732
       38     3.800 |   -0.1127754    0.0637106
       39     3.900 |   -0.1087135    0.0623253
-----------------------------------------------
Iteration      Time |             ML_WAVETOY::u
                    |      minimum      maximum
-----------------------------------------------
       40     4.000 |   -0.1058024    0.0609972
       41     4.100 |   -0.1035130    0.0597379
       42     4.200 |   -0.1018068    0.0585189
       43     4.300 |   -0.1025582    0.0572997
       44     4.400 |   -0.1049331    0.0559763
       45     4.500 |   -0.1089452    0.0546397
       46     4.600 |   -0.1135065    0.0532975
       47     4.700 |   -0.1193913    0.0516636
       48     4.800 |   -0.1232103    0.0500149
       49     4.900 |   -0.1255127    0.0483666
       50     5.000 |   -0.1265008    0.0466617
       51     5.100 |   -0.1249368    0.0443153
       52     5.200 |   -0.1220594    0.0419306
       53     5.300 |   -0.1192784    0.0395087
       54     5.400 |   -0.1169934    0.0361057
       55     5.500 |   -0.1156997    0.0327472
       56     5.600 |   -0.1152458    0.0294298
       57     5.700 |   -0.1169892    0.0249518
       58     5.800 |   -0.1229219    0.0207183
       59     5.900 |   -0.1283221    0.0167178
-----------------------------------------------
Iteration      Time |            ML_WAVETOY::u
                    |      minimum      maximum
-----------------------------------------------
       60     6.000 |   -0.1328251    0.0118365
       61     6.100 |   -0.1364733    0.0081641
       62     6.200 |   -0.1376141    0.0063958
       63     6.300 |   -0.1362589    0.0146691
       64     6.400 |   -0.1343881    0.0220344
       65     6.500 |   -0.1333202    0.0289748
       66     6.600 |   -0.1397765    0.0348713
       67     6.700 |   -0.1456408    0.0391332
       68     6.800 |   -0.1496222    0.0420545
       69     6.900 |   -0.1522635    0.0447583
       70     7.000 |   -0.1537758    0.0460384
       71     7.100 |   -0.1523184    0.0460146
       72     7.200 |   -0.1475696    0.0462221
       73     7.300 |   -0.1394179    0.0452860
       74     7.400 |   -0.1278376    0.0436437
       75     7.500 |   -0.1128998    0.0474083
       76     7.600 |   -0.1058810    0.0623618
       77     7.700 |   -0.1047101    0.0763079
       78     7.800 |   -0.1036498    0.0876189
       79     7.900 |   -0.1029353    0.0960362
-----------------------------------------------
Iteration      Time |            ML_WAVETOY::u
                    |      minimum      maximum
-----------------------------------------------
       80     8.000 |   -0.1032067    0.1017677
       81     8.100 |   -0.1056260    0.1067423
       82     8.200 |   -0.1118774    0.1087837
       83     8.300 |   -0.1245356    0.1079294
       84     8.400 |   -0.1375977    0.1096677
       85     8.500 |   -0.1499178    0.1325744
       86     8.600 |   -0.1609802    0.1525642
       87     8.700 |   -0.1702419    0.1689433
       88     8.800 |   -0.1771561    0.1812100
       89      8.900 |   -0.1812009    0.1889942
       90      9.000 |   -0.1819094    0.1950914
       91      9.100 |   -0.1789004    0.1975371
       92      9.200 |   -0.1719078    0.1966989
       93      9.300 |   -0.1608049    0.1951874
       94      9.400 |   -0.1456230    0.1905209
       95      9.500 |   -0.1265615    0.1863564
       96      9.600 |   -0.1039884    0.1794257
       97      9.700 |   -0.0823688    0.1735933
       98      9.800 |   -0.0706553    0.1651859
       99      9.900 |   -0.0560750    0.1572037
-----------------------------------------------
Iteration      Time |                ML_WAVETOY::u
                    |      minimum      maximum
-----------------------------------------------
      100     10.000 |   -0.0533831    0.1479392
----------------------------------------------------------------------
----------
Done.
```

we should now have a folder called "gaussian-RK4" with the output

Plotting the output

The output for this example consists of simple ascii files. The `rho` function (which maps to either u or v in the equations above), is shown in the next cell. The "d" is for diagonal.

```
!head -20 ./gaussian-RK4/u.x.asc
```

```
# 1D ASCII output created by CarpetIOASCII
# created on relayer by mzilhao on Nov 14 2022 at 16:24:48+0100
# parameter filename: "parfiles/gaussian-RK4.par"
#
# u x (u)
#
# iteration 0   time 0
# time level 0
# refinement level 0   multigrid level 0   map 0   component 0
# column format: 1:it 2:ix 3:iy 4:iz  5:time     6:x 7:y 8:z
     9:data
0     0 26 26    0     -5.2 0 0   1.34381227763152e-06
0     1 26 26    0     -5 0 0     3.72665317207867e-06
0     2 26 26    0     -4.8 0 0   9.92950430585108e-06
0     3 26 26    0     -4.6 0 0   2.54193465161992e-05
0     4 26 26    0     -4.4 0 0   6.25215037748202e-05
0     5 26 26    0     -4.2 0 0   0.000147748360232034
0     6 26 26    0     -4 0 0     0.000335462627902512
0     7 26 26    0     -3.8 0 0   0.000731802418880472
```

```
0     8 26 26    0     -3.6 0 0   0.00153381067932446
0     9 26 26    0     -3.4 0 0   0.00308871540823677
```

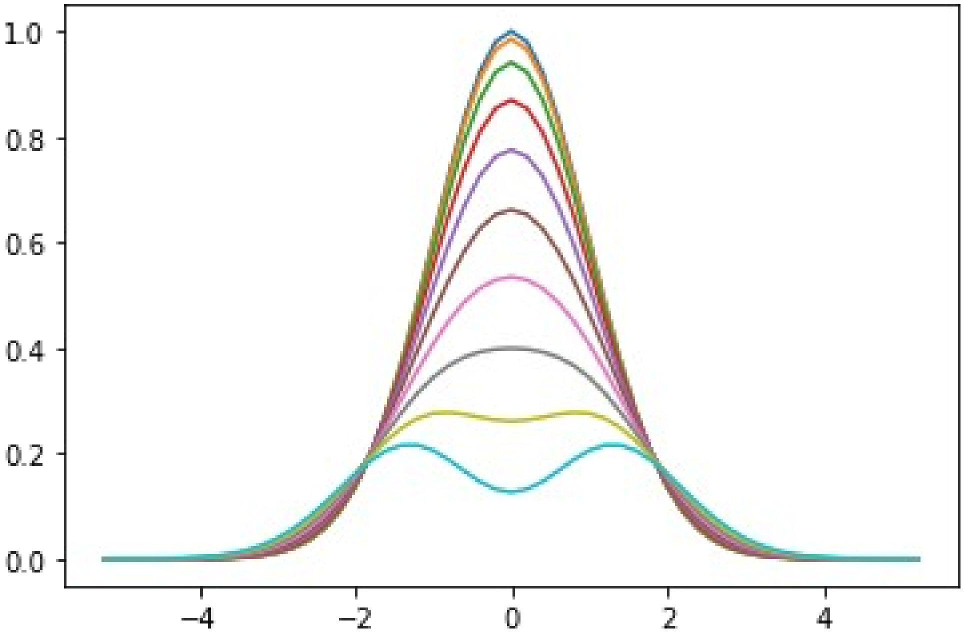
```
# Load the data using numpy
import os
import numpy as np
data = np.genfromtxt(os.path.join("gaussian-RK4","u.x.asc"))
print(data.shape)
```

```
(5353, 9)
```

```
# Load the timesteps from the output data
time_steps = np.unique(data[:,0])
print(time_steps)
```

```
[  0.   1.   2.   3.   4.   5.   6.   7.   8.   9.  10.  11.  12.  13.
  14.  15.  16.  17.  18.  19.  20.  21.  22.  23.  24.  25.  26.  27.
  28.  29.  30.  31.  32.  33.  34.  35.  36.  37.  38.  39.  40.  41.
  42.  43.  44.  45.  46.  47.  48.  49.  50.  51.  52.  53.  54.  55.
  56.  57.  58.  59.  60.  61.  62.  63.  64.  65.  66.  67.  68.  69.
  70.  71.  72.  73.  74.  75.  76.  77.  78.  79.  80.  81.  82.  83.
  84.  85.  86.  87.  88.  89.  90.  91.  92.  93.  94.  95.  96.  97.
  98.  99. 100.]
```

```
# Plot the first 10 timesteps
import matplotlib.pyplot as plt
x = data[data[:,0] == time_steps[0]][:,5]
for time_step in time_steps[:10]:
    step_data = data[data[:,0] == time_step][:,-1]
    plt.plot(x,step_data)
```



```
!mkdir figs
```

```
cd figs
```

```
/home/mzilhao/01-Projectos/2022-11_Meudon/apr/figs
```

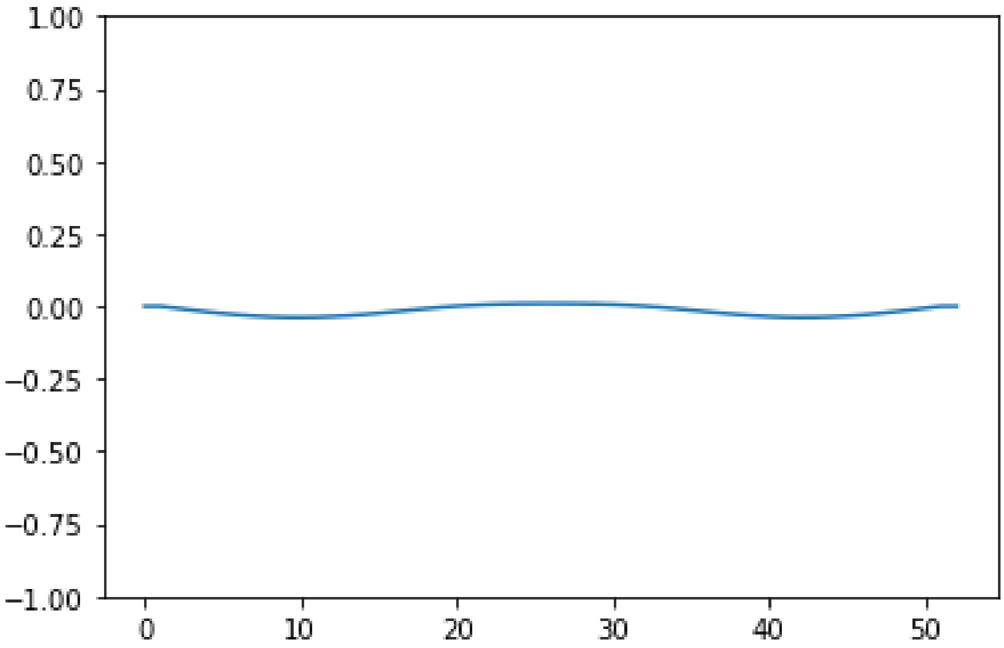
```
# Create a movie using all the timesteps
import matplotlib.pyplot as plt
frameno = 0
for time_step in time_steps:
    plt.clf()
    step_data = data[data[:,0] == time_step][:,-1]
    plt.ylim([-1,1])
    plt.plot(step_data)
    frameno += 1
    plt.savefig("plot%03d.png" % frameno)
```

```
from subprocess import call
call(["ffmpeg", "-i", "plot%03d.png", "-i", "plot001.png",
      "-filter_complex", "[1:v] palettegen [p];[0:v][p] paletteuse",
      "-y", "output.gif"])
```

```
ffmpeg version 5.0.1 Copyright (c) 2000-2022 the FFmpeg developers
  built with gcc 12 (GCC)
  configuration: --prefix=/usr --bindir=/usr/bin
--datadir=/usr/share/ffmpeg --docdir=/usr/share/doc/ffmpeg
--incdir=/usr/include/ffmpeg --libdir=/usr/lib64
--mandir=/usr/share/man --arch=x86_64 --optflags='-O2 -flto=auto -
ffat-lto-objects -fexceptions -g -grecord-gcc-switches -pipe -Wall -
Werror=format-security -Wp,-D_FORTIFY_SOURCE=2 -Wp,-
D_GLIBCXX_ASSERTIONS -specs=/usr/lib/rpm/redhat/redhat-hardened-cc1 -
fstack-protector-strong -specs=/usr/lib/rpm/redhat/redhat-annobin-cc1
-m64 -mtune=generic -fasynchronous-unwind-tables -fstack-clash-
protection -fcf-protection' --extra-ldflags='-Wl,-z,relro -Wl,--as-
needed -Wl,-z,now -specs=/usr/lib/rpm/redhat/redhat-hardened-ld -
specs=/usr/lib/rpm/redhat/redhat-annobin-cc1 -Wl,--build-id=sha1 ' --
extra-cflags=' -I/usr/include/rav1e' --enable-libopencore-amrnb --
enable-libopencore-amrwb --enable-libvo-amrwbenc --enable-version3 --
enable-bzlib --enable-chromaprint --disable-crystalhd --enable-
fontconfig --enable-frei0r --enable-gcrypt --enable-gnutls --enable-
ladspa --enable-libaom --enable-libdav1d --enable-libass --enable-
libbluray --enable-libbs2b --enable-libcdio --enable-libdrm --enable-
libjack --enable-libfreetype --enable-libfribidi --enable-libgsm --
enable-libilbc --enable-libmp3lame --enable-libmysofa --enable-nvenc
--enable-openal --enable-opencl --enable-opengl --enable-libopenjpeg
--enable-libopenmpt --enable-libopus --enable-libpulse --enable-
librsvg --enable-librav1e --enable-librubberband --enable-libsmbclient
--enable-version3 --enable-libsnappy --enable-libsoxr --enable-
libspeex --enable-libsrt --enable-libssh --enable-libsvtav1 --enable-
libtheora --enable-libtwolame --enable-libvorbis --enable-libv4l2 --
```

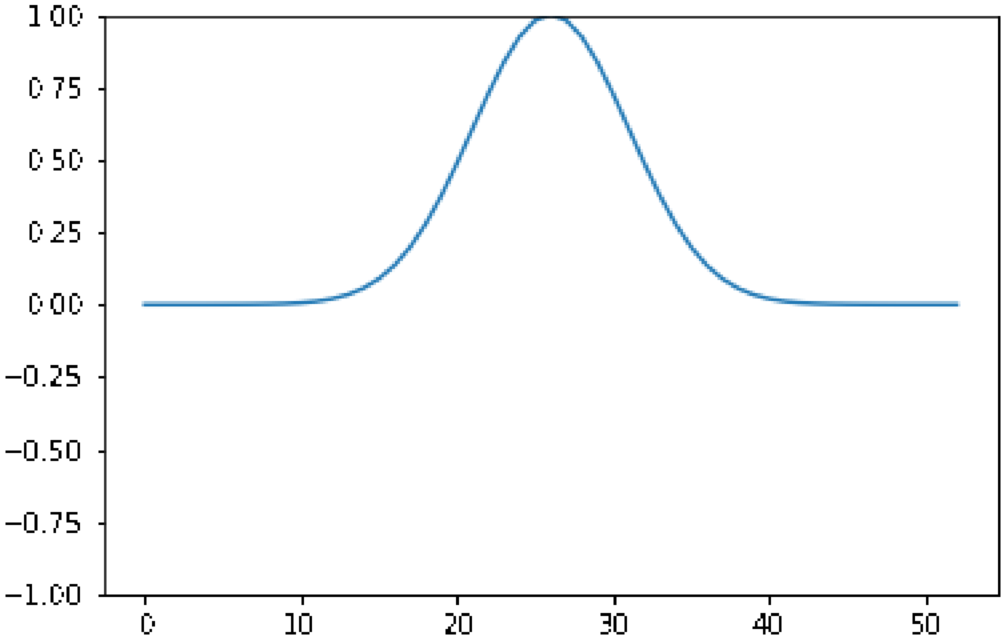
```
enable-libvidstab --enable-libvmaf --enable-version3 --enable-
vapoursynth --enable-libvpx --enable-vulkan --enable-libglslang --
enable-libwebp --enable-libx264 --enable-libx265 --enable-libxvid --
enable-libxml2 --enable-libzimg --enable-libzmq --enable-libzvbi --
enable-lv2 --enable-avfilter --enable-libmodplug --enable-postproc --
enable-pthreads --disable-static --enable-shared --enable-gpl --
disable-debug --disable-stripping --shlibdir=/usr/lib64 --enable-lto
--enable-libmfx --enable-runtime-cpudetect
  libavutil      57. 17.100 / 57. 17.100
  libavcodec     59. 18.100 / 59. 18.100
  libavformat    59. 16.100 / 59. 16.100
  libavdevice    59.  4.100 / 59.  4.100
  libavfilter     8. 24.100 /  8. 24.100
  libswscale      6.  4.100 /  6.  4.100
  libswresample   4.  3.100 /  4.  3.100
  libpostproc    56.  3.100 / 56.  3.100
Input #0, image2, from 'plot%03d.png':
  Duration: 00:00:04.04, start: 0.000000, bitrate: N/A
  Stream #0:0: Video: png, rgba(pc), 432x288 [SAR 2835:2835 DAR 3:2],
25 fps, 25 tbr, 25 tbn
Input #1, png_pipe, from 'plot001.png':
  Duration: N/A, bitrate: N/A
  Stream #1:0: Video: png, rgba(pc), 432x288 [SAR 2835:2835 DAR 3:2],
25 fps, 25 tbr, 25 tbn
Stream mapping:
  Stream #0:0 (png) -> paletteuse
  Stream #1:0 (png) -> palettegen:default
  paletteuse:default -> Stream #0:0 (gif)
Press [q] to stop, [?] for help
[image2 @ 0x55d47ae42300] Thread message queue blocking; consider
raising the thread_queue_size option (current value: 8)
[Parsed_palettegen_0 @ 0x55d47ae4fe00] 255(+1) colors generated out of
427 colors; ratio=0.597190
[Parsed_palettegen_0 @ 0x55d47ae4fe00] Duped color: FF000000
    Last message repeated 2 times
[Parsed_palettegen_0 @ 0x55d47ae4fe00] Duped color: FFFFFFFF
Output #0, gif, to 'output.gif':
  Metadata:
    encoder         : Lavf59.16.100
  Stream #0:0: Video: gif, pal8(pc, gbr/unknown/unknown, progressive),
432x288 [SAR 1:1 DAR 3:2], q=2-31, 200 kb/s, 25 fps, 100 tbn
    Metadata:
      encoder         : Lavc59.18.100 gif
frame=  101 fps=0.0 q=-0.0 Lsize=     283kB time=00:00:04.01 bitrate=
578.5kbits/s speed=12.2x
video:283kB audio:0kB subtitle:0kB other streams:0kB global
headers:0kB muxing overhead: 0.006898%

0
```



```
# Show the movie
from IPython.display import Image

Image("output.gif")
```



```
cd ..
```

```
/home/mzilhao/01-Projectos/2022-11_Meudon/apr
```

Exercises

- Change the parameters of the `WaveMoL` `thorn` and compare the output.
  - Recall that you can check the names of all the provided parameters in the file `WaveMoL/param.ccl`. All the parameters that are not explicitly set at runtime take the default values given therein.

TOV

Let us now simulate a static TOV star. Below we construct a Cactus parameter file to simulate a single, spherical symmetric star using the Einstein Toolkit.

```
%%writefile parfiles/tov.par
# tov.par

# Example parameter file for a static TOV star. Everything is evolved, but
# because this is a solution to the GR and hydro equations, nothing changes
# much. What can be seen is the initial perturbation (due to numerical errors)
# ringing down (look at the density maximum), and later numerical errors
# governing the solution. Try higher resolutions to decrease this error.

# Some basic stuff
ActiveThorns = "Time MoL"
ActiveThorns = "Coordbase CartGrid3d Boundary StaticConformal"
ActiveThorns = "SymBase ADMBase TmunuBase HydroBase InitBase ADMCoupling ADMMacros"
ActiveThorns = "IOUtil"
ActiveThorns = "SpaceMask CoordGauge Constants LocalReduce aeilocalinterp LoopControl"
ActiveThorns = "Carpet CarpetLib CarpetReduce CarpetRegrid2 CarpetInterp"
ActiveThorns = "CarpetIOASCII CarpetIOScalar CarpetIOHDF5 CarpetIOBasic"

# Finalize
Cactus::terminate            = "time"
Cactus::cctk_final_time      = 80 #400 # divide by ~203 to get ms

# Termination Trigger
```

```
ActiveThorns = "TerminationTrigger"
TerminationTrigger::max_walltime = 24          # hours
TerminationTrigger::on_remaining_walltime = 0  # minutes
TerminationTrigger::check_file_every = 512
TerminationTrigger::termination_file = "TerminationTrigger.txt"
TerminationTrigger::termination_from_file   = "yes"
TerminationTrigger::create_termination_file = "yes"

# grid parameters
Carpet::domain_from_coordbase = "yes"
CartGrid3D::type         = "coordbase"
CartGrid3D::domain       = "full"
CartGrid3D::avoid_origin = "no"
CoordBase::xmin =  0.0
CoordBase::ymin =  0.0
CoordBase::zmin =  0.0
CoordBase::xmax = 24.0
CoordBase::ymax = 24.0
CoordBase::zmax = 24.0
# Change these parameters to change resolution. The ?max settings
above
# have to be multiples of these. 'dx' is the size of one cell in x-
direction.
# Making this smaller means using higher resolution, because more
points will
# be used to cover the same space.
CoordBase::dx   =   2.0
CoordBase::dy   =   2.0
CoordBase::dz   =   2.0

CarpetRegrid2::regrid_every =   0
CarpetRegrid2::num_centres  =   1
CarpetRegrid2::num_levels_1 =   2
CarpetRegrid2::radius_1[1]  = 12.0


CoordBase::boundary_size_x_lower        = 3
CoordBase::boundary_size_y_lower        = 3
CoordBase::boundary_size_z_lower        = 3
CoordBase::boundary_size_x_upper        = 3
CoordBase::boundary_size_y_upper        = 3
CoordBase::boundary_size_z_upper        = 3
CoordBase::boundary_shiftout_x_lower    = 1
CoordBase::boundary_shiftout_y_lower    = 1
CoordBase::boundary_shiftout_z_lower    = 1
CoordBase::boundary_shiftout_x_upper    = 0
CoordBase::boundary_shiftout_y_upper    = 0
CoordBase::boundary_shiftout_z_upper    = 0
```

```
ActiveThorns = "ReflectionSymmetry"

ReflectionSymmetry::reflection_x = "yes"
ReflectionSymmetry::reflection_y = "yes"
ReflectionSymmetry::reflection_z = "yes"
ReflectionSymmetry::avoid_origin_x = "no"
ReflectionSymmetry::avoid_origin_y = "no"
ReflectionSymmetry::avoid_origin_z = "no"

# storage and coupling
TmunuBase::stress_energy_storage = yes
TmunuBase::stress_energy_at_RHS  = yes
TmunuBase::timelevels            =  1
TmunuBase::prolongation_type     = none



HydroBase::timelevels            = 3

ADMMacros::spatial_order = 4

SpaceMask::use_mask      = "yes"

Carpet::enable_all_storage       = no
Carpet::use_buffer_zones         = "yes"

Carpet::poison_new_timelevels    = "yes"
Carpet::check_for_poison         = "no"

Carpet::init_3_timelevels        = no
Carpet::init_fill_timelevels     = "yes"

CarpetLib::poison_new_memory = "yes"
CarpetLib::poison_value      = 114

# system specific Carpet paramters
Carpet::max_refinement_levels    = 10
driver::ghost_size               = 3
Carpet::prolongation_order_space = 3
Carpet::prolongation_order_time  = 2

# Time integration
time::dtfac = 0.25

MoL::ODE_Method             = "rk4"
MoL::MoL_Intermediate_Steps = 4
MoL::MoL_Num_Scratch_Levels = 1

# check all physical variables for NaNs
#  This can save you computing time, so it's not a bad idea to do this
#  once in a whioe.
```

```
ActiveThorns = "NaNChecker"
NaNChecker::check_every = 16384
NaNChecker::action_if_found = "terminate" #"terminate", "just warn", "abort"
NaNChecker::check_vars = "ADMBase::metric ADMBase::lapse ADMBase::shift HydroBase::rho HydroBase::eps HydroBase::press HydroBase::vel"

# Hydro paramters

ActiveThorns = "EOS_Omni GRHydro"

HydroBase::evolution_method      = "GRHydro"

GRHydro::riemann_solver         = "Marquina"
GRHydro::GRHydro_eos_type       = "Polytype"
GRHydro::GRHydro_eos_table      = "2D_Polytrope"
GRHydro::recon_method           = "ppm"
GRHydro::GRHydro_stencil        = 3
GRHydro::bound                  = "none"
GRHydro::rho_abs_min            = 1.e-10
# Parameter controlling finite difference order of the Christoffel symbols in GRHydro
GRHydro::sources_spatial_order  = 4

# Curvature evolution parameters

ActiveThorns = "GenericFD NewRad"
ActiveThorns = "ML_BSSN ML_BSSN_Helper"
ADMBase::evolution_method        = "ML_BSSN"
ADMBase::lapse_evolution_method  = "ML_BSSN"
ADMBase::shift_evolution_method  = "ML_BSSN"
ADMBase::dtlapse_evolution_method= "ML_BSSN"
ADMBase::dtshift_evolution_method= "ML_BSSN"

ML_BSSN::timelevels = 3

ML_BSSN::harmonicN           = 1      # 1+log
ML_BSSN::harmonicF           = 2.0    # 1+log
ML_BSSN::evolveA             = 1
ML_BSSN::evolveB             = 1
# NOTE: The following parameters select geodesic slicing. This choice only enables you to evolve stationary spacetimes.
#       They will not allow you to simulate a collapsing TOV star.
ML_BSSN::ShiftGammaCoeff     = 0.0
ML_BSSN::AlphaDriver         = 0.0
ML_BSSN::BetaDriver          = 0.0
ML_BSSN::advectLapse         = 0
ML_BSSN::advectShift         = 0
ML_BSSN::MinimumLapse        = 1.0e-8
```

```
ML_BSSN::my_initial_boundary_condition = "extrapolate-gammas"

ML_BSSN::my_rhs_boundary_condition     = "NewRad"

# Some dissipation to get rid of high-frequency noise
ActiveThorns = "SphericalSurface Dissipation"
Dissipation::verbose   = "no"
Dissipation::epsdis   = 0.01
Dissipation::vars = "
        ML_BSSN::ML_log_confac
        ML_BSSN::ML_metric
        ML_BSSN::ML_curv
        ML_BSSN::ML_trace_curv
        ML_BSSN::ML_Gamma
        ML_BSSN::ML_lapse
        ML_BSSN::ML_shift
"


# init parameters
InitBase::initial_data_setup_method = "init_some_levels"

# Use TOV as initial data
ActiveThorns = "TOVSolver"

HydroBase::initial_hydro         = "tov"
ADMBase::initial_data            = "tov"
ADMBase::initial_lapse           = "tov"
ADMBase::initial_shift           = "tov"
ADMBase::initial_dtlapse         = "zero"
ADMBase::initial_dtshift         = "zero"

# Parameters for initial star
TOVSolver::TOV_Rho_Central[0] = 1.28e-3
TOVSolver::TOV_Gamma          = 2
TOVSolver::TOV_K              = 100

# Set equation of state for evolution
EOS_Omni::poly_gamma                   = 2
EOS_Omni::poly_k                       = 100
EOS_Omni::gl_gamma                     = 2
EOS_Omni::gl_k                         = 100


# I/O

# Use (create if necessary) an output directory named like the
# parameter file (minus the .par)
IO::out_dir             = ${parfile}
```

```
# Write one file overall per output (variable/group)
# In production runs, comment this or set to "proc" to get one file
# per MPI process
IO::out_mode            = "onefile"

# Some screen output
IOBasic::outInfo_every = 512
IOBasic::outInfo_vars  = "Carpet::physical_time_per_hour
HydroBase::rho{reductions='maximum'}"

# Scalar output
IOScalar::outScalar_every    = 512
IOScalar::one_file_per_group = "yes"
IOScalar::outScalar_reductions = "norm1 norm2 norm_inf sum maximum
minimum"
IOScalar::outScalar_vars     = "
 HydroBase::rho{reductions='maximum'}
 HydroBase::press{reductions='maximum'}
 HydroBase::eps{reductions='minimum maximum'}
 HydroBase::vel{reductions='minimum maximum'}
 HydroBase::w_lorentz{reductions='minimum maximum'}
 ADMBase::lapse{reductions='minimum maximum'}
 ADMBase::shift{reductions='minimum maximum'}
 ML_BSSN::ML_Ham{reductions='norm1 norm2 maximum minimum norm_inf'}
 ML_BSSN::ML_mom{reductions='norm1 norm2 maximum minimum norm_inf'}
 GRHydro::dens{reductions='minimum maximum sum'}
 Carpet::timing{reductions='average'}
"

# 1D ASCII output. Disable for production runs!
IOASCII::out1D_every        = 2048
IOASCII::one_file_per_group = yes
IOASCII::output_symmetry_points = no
IOASCII::out1D_vars         = "
 HydroBase::rho
 HydroBase::press
 HydroBase::eps
 HydroBase::vel
 ADMBase::lapse
 ADMBase::metric
 ADMBase::curv
 ML_BSSN::ML_Ham
 ML_BSSN::ML_mom
"

# 2D HDF5 output
CarpetIOHDF5::output_buffer_points = "no"

CarpetIOHDF5::out2D_every = 2048
```

```
CarpetIOHDF5::out2D_vars = "
 HydroBase::rho
 HydroBase::eps
 HydroBase::vel
 HydroBase::w_lorentz
 ADMBase::lapse
 ADMBase::shift
 ADMBase::metric
 ML_BSSN::ML_Ham
 ML_BSSN::ML_mom
 "

Writing parfiles/tov.par
```

run the simulation

```
%%bash
export OMP_NUM_THREADS=1
mpirun -np 2 $EXE parfiles/tov.par

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

       10
  1   0101       ************************
  01  1010 10      The Cactus Code V4.11.0
 1010 1101 011      www.cactuscode.org
  1001 100101    ************************
    00010101
     100011     (c) Copyright The Authors
      0100      GNU Licensed. No Warranty
      0101
----------------------------------------------------------------------
----------

Cactus version:    4.11.0
Compile date:      Aug 01 2022 (10:36:07)
Run date:          Nov 14 2022 (16:45:41+0100)
Run host:          relayer (pid=17680)
Working directory: /home/mzilhao/01-Projectos/2022-11_Meudon/apr
Executable:        /home/mzilhao/./dev/ET/Cactus/exe/cactus_ET
Parameter file:    parfiles/tov.par
----------------------------------------------------------------------
----------

Activating thorn Cactus...Success -> active implementation Cactus
Activation requested for
--->Time MoL Coordbase CartGrid3d Boundary StaticConformal SymBase
ADMBase TmunuBase HydroBase InitBase ADMCoupling ADMMacros IOUtil
SpaceMask CoordGauge Constants LocalReduce aeilocalinterp LoopControl
```

```
Carpet CarpetLib CarpetReduce CarpetRegrid2 CarpetInterp CarpetIOASCII
CarpetIOScalar CarpetIOHDF5 CarpetIOBasic TerminationTrigger
ReflectionSymmetry NaNChecker EOS_Omni GRHydro GenericFD NewRad
ML_BSSN ML_BSSN_Helper SphericalSurface Dissipation TOVSolver<---
Thorn Carpet requests automatic activation of MPI
Thorn Carpet requests automatic activation of Timers
Thorn CarpetIOHDF5 requests automatic activation of HDF5
Thorn CarpetLib requests automatic activation of Vectors
Thorn CarpetLib requests automatic activation of CycleClock
Thorn GRHydro requests automatic activation of EOS_Polytrope
Thorn LoopControl requests automatic activation of hwloc
Thorn EOS_Polytrope requests automatic activation of EOS_Base
Thorn HDF5 requests automatic activation of zlib
Activating thorn ADMBase...Success -> active implementation ADMBase
Activating thorn ADMCoupling...Success -> active implementation
ADMCoupling
Activating thorn ADMMacros...Success -> active implementation
ADMMacros
Activating thorn aeilocalinterp...Success -> active implementation
AEILocalInterp
Activating thorn Boundary...Success -> active implementation boundary
Activating thorn Carpet...Success -> active implementation Driver
Activating thorn CarpetInterp...Success -> active implementation
interp
Activating thorn CarpetIOASCII...Success -> active implementation
IOASCII
Activating thorn CarpetIOBasic...Success -> active implementation
IOBasic
Activating thorn CarpetIOHDF5...Success -> active implementation
IOHDF5
Activating thorn CarpetIOScalar...Success -> active implementation
IOScalar
Activating thorn CarpetLib...Success -> active implementation
CarpetLib
Activating thorn CarpetReduce...Success -> active implementation
reduce
Activating thorn CarpetRegrid2...Success -> active implementation
CarpetRegrid2
Activating thorn CartGrid3d...Success -> active implementation grid
Activating thorn Constants...Success -> active implementation
Constants
Activating thorn Coordbase...Success -> active implementation
CoordBase
Activating thorn CoordGauge...Success -> active implementation
CoordGauge
Activating thorn CycleClock...Success -> active implementation
CycleClock
Activating thorn Dissipation...Success -> active implementation
Dissipation
```

```
Activating thorn EOS_Base...Success -> active implementation EOS_Base
Activating thorn EOS_Omni...Success -> active implementation EOS_Omni
Activating thorn EOS_Polytrope...Success -> active implementation
EOS_2d_Polytrope
Activating thorn GenericFD...Success -> active implementation
GenericFD
Activating thorn GRHydro...Success -> active implementation GRHydro
Activating thorn HDF5...Success -> active implementation HDF5
Activating thorn hwloc...Success -> active implementation hwloc
Activating thorn HydroBase...Success -> active implementation
HydroBase
Activating thorn InitBase...Success -> active implementation InitBase
Activating thorn IOUtil...Success -> active implementation IO
Activating thorn LocalReduce...Success -> active implementation
LocalReduce
Activating thorn LoopControl...Success -> active implementation
LoopControl
Activating thorn ML_BSSN...Success -> active implementation ML_BSSN
Activating thorn ML_BSSN_Helper...Success -> active implementation
ML_BSSN_Helper
Activating thorn MoL...Success -> active implementation MethodOfLines
Activating thorn MPI...Success -> active implementation MPI
Activating thorn NaNChecker...Success -> active implementation
NaNChecker
Activating thorn NewRad...Success -> active implementation NewRad
Activating thorn ReflectionSymmetry...Success -> active implementation
ReflectionSymmetry
Activating thorn SpaceMask...Success -> active implementation
SpaceMask
Activating thorn SphericalSurface...Success -> active implementation
SphericalSurface
Activating thorn StaticConformal...Success -> active implementation
StaticConformal
Activating thorn SymBase...Success -> active implementation SymBase
Activating thorn TerminationTrigger...Success -> active implementation
TerminationTrigger
Activating thorn Time...Success -> active implementation time
Activating thorn Timers...Success -> active implementation Timers
Activating thorn TmunuBase...Success -> active implementation
TmunuBase
Activating thorn TOVSolver...Success -> active implementation
TOVSolver
Activating thorn Vectors...Success -> active implementation Vectors
Activating thorn zlib...Success -> active implementation zlib
----------------------------------------------------------------------
----------
  if (recover initial data)
    Recover parameters
  endif
  Startup routines
    [CCTK_STARTUP]
      Carpet::MultiModel_Startup: Multi-model Startup routine
      CycleClock::CycleClock_Setup: Set up CycleClock
      LoopControl::LC_setup: Set up LoopControl
      ML_BSSN_Helper::ML_BSSN_SetGroupTags: [meta] Set checkpointing
and prolongation group tags
      Timers::Timer_Startup: Prepare hierarchical timers
      Carpet::Driver_Startup: Startup routine
      IOUtil::IOUtil_Startup: Startup routine
      CarpetInterp::CarpetInterpStartup: Startup routine
      CarpetReduce::CarpetReduceStartup: Startup routine
      CartGrid3D::SymmetryStartup: Register GH Extension for
GridSymmetry
      CoordBase::CoordBase_Startup: Register a GH extension to store
the coordinate system handles
      AEILocalInterp::AEILocalInterp_U_Startup: register
CCTK_InterpLocalUniform() interpolation operators
      EOS_Omni::EOS_Omni_Startup: [global] Set up conversion factors
and other fun stuff
      EOS_Polytrope::EOS_Polytrope_Startup: Setup the polytropic EOS
      GRHydro::GRHydro_RegisterMask: Register the hydro masks
      HydroBase::HydroBase_StartUp: Startup banner
      CarpetIOASCII::CarpetIOASCIIStartup: [global] Startup routine
      LocalReduce::LocalReduce_Startup: Startup routine
      CarpetIOBasic::CarpetIOBasicStartup: [global] Startup routine
      ML_BSSN::ML_BSSN_Startup: [meta] create banner
      ML_BSSN_Helper::ML_BSSN_RegisterSlicing: [meta] Register slicing
      CarpetIOHDF5::CarpetIOHDF5_Startup: Startup routine
      MoL::MoL_Startup: Startup banner
      SymBase::SymBase_Startup: Register GH Extension for SymBase
      TerminationTrigger::TerminationTrigger_StartSignalHandler: Start
signal handler
      CarpetIOScalar::CarpetIOScalarStartup: [global] Startup routine
      Vectors::Vectors_Startup: Print startup message
      GROUP hwloc_startup: hwloc startup group
        hwloc::hwloc_version: Output hwloc version

  Startup routines which need an existing grid hierarchy
    [CCTK_WRAGH]
      ADMBase::Einstein_InitSymBound: [global] Set up GF symmetries
      Boundary::Boundary_RegisterBCs: [global] Register boundary
conditions that this thorn provides
      CarpetRegrid2::CarpetRegrid2_Initialise: [global] Initialise
locations of refined regions
      CartGrid3D::RegisterCartGrid3DCoords: [meta] Register
coordinates for the Cartesian grid
      CoordGauge::Einstein_ActivateSlicing: Initialize slicing, setup
```

```
priorities for mixed slicings
      CoordGauge::Einstein_SetNextSlicing: Identify the slicing for
the next iteration
      GRHydro::GRHydro_Startup: Startup banner
      GRHydro::GRHydro_ClearLastMoLPostStep: [global] Initialize
InLastMoLPostStep to zero
      ML_BSSN_Helper::ML_BSSN_ParamCompat: [meta] Handle parameter
backward compatibility
      MoL::MoL_SetupIndexArrays: Set up the MoL bookkeeping index
arrays
      MoL::MoL_SetScheduleStatus: [global] Set the flag so it is ok to
register with MoL
      TmunuBase::TmunuBase_SetStressEnergyState: [global] Set the
stress_energy_state variable
      GROUP MoL_Register: The group where physics thorns register
variables with MoL
        GRHydro::GRHydro_Register: Register variables for MoL
        ML_BSSN::ML_BSSN_RegisterVars: [meta] Register Variables for
MoL
        ML_BSSN_Helper::ML_BSSN_RegisterConstrained: [meta] Register
ADMBase variables as constrained
      SpaceMask::MaskSym: [global] Set grid symmetries for mask
      SpaceMask::MaskSym_emask: [global] Set grid symmetries for emask
(compatibility mode)
      GROUP SymBase_Wrapper: Wrapper group for SymBase
        GROUP SymmetryRegister: Register your symmetries here
          CartGrid3D::CartGrid3D_RegisterSymmetryBoundaries: [meta]
Register symmetry boundaries
          ML_BSSN::ML_BSSN_RegisterSymmetries: [meta] register
symmetries
          ReflectionSymmetry::ReflectionSymmetry_Register: Register
reflection symmetry boundaries
        SymBase::SymBase_Statistics: Print symmetry boundary face
descriptions
      TOVSolver::TOV_C_AllocateMemory: [global] Allocate memory for
TOVSolver_C
      MoL::MoL_ReportNumberVariables: [meta] Report how many of each
type of variable there are
  Parameter checking routines
    [CCTK_PARAMCHECK]
      ADMBase::ADMBase_ParamCheck: [global] Check consistency of
parameters
      Boundary::Boundary_Check: Check dimension of grid variables
      Carpet::CarpetParamCheck: Parameter checking routine
      CarpetLib::CarpetLib_test_prolongate_3d_rf2: [global] Test
prolongation operators
      CarpetRegrid2::CarpetRegrid2_ParamCheck: Check parameters
      CartGrid3D::ParamCheck_CartGrid3D: Check coordinates for
CartGrid3D
```

```
      Dissipation::dissipation_paramcheck: Check dissipation
parameters for consistency
      GRHydro::GRHydro_ParamCheck: Check parameters
      ML_BSSN_Helper::ML_BSSN_ParamCheck: [meta] Check parameters
      MoL::MoL_ParamCheck: Basic parameter checking
      SphericalSurface::SphericalSurface_ParamCheck: [global] Check
that all surface names are unique
      TOVSolver::TOV_C_ParamCheck: [global] Check parameters
      TerminationTrigger::TerminationTrigger_ParamCheck: Check
consitency of parameters
      TmunuBase::TmunuBase_ParamCheck: [global] Check that no
deprecated parameters are used.
      Vectors::Vectors_Test: Run correctness tests.

  Initialisation
    if (NOT (recover initial data AND recovery_mode is 'strict'))
      [CCTK_PREREGRIDINITIAL]
      Set up grid hierarchy
      [CCTK_POSTREGRIDINITIAL]
        CartGrid3D::SpatialCoordinates: Set Coordinates after
regridding
        GROUP MaskBase_SetupMask: Set up the weight function
          GROUP MaskBase_SetupMaskAll: Set up the weight function
            CarpetReduce::MaskBase_AllocateMask: [global] Allocate the
weight function
            CarpetReduce::MaskBase_InitMask: [global] [loop-local]
Initialise the weight function
            GROUP SetupIMaskInternal: Set up the integer weight
function (schedule other routines in here)
              CarpetReduce::CoordBase_SetupMask: [global] [loop-local]
Set up the outer boundaries of the weight function
              CarpetReduce::CarpetMaskSetup: [global] [loop-singlemap]
Set up the weight function for the restriction regions
            GROUP SetupIMask: Set up the integer weight function
(schedule other routines in here)
            CarpetReduce::MaskBase_SetMask: [global] [loop-local] Set
the weight function
            GROUP SetupMask: Set up the real weight function (schedule
other routines in here)
            CarpetReduce::MaskBase_TestMask: [global] Test the weight
function
        Dissipation::setup_epsdis: Setup spatially varying dissipation
        SpaceMask::MaskZero: Initialise mask to zero
        GRHydro::GRHydro_RefinementLevel: Calculate current refinement
level
        GROUP HydroBase_ExcisionMaskSetup: Set up hydro excision mask
          HydroBase::HydroBase_InitExcisionMask: Initialize hydro
excision mask to 'no excision everywhere'
        GROUP MoL_PseudoEvolutionBoundaries: Apply boundary conditions
```

```
to pseudo-evolved quantities
          GROUP ML_BSSN_ConstraintsEverywhere_bc_group:
ML_BSSN_ConstraintsEverywhere
            ML_BSSN::ML_BSSN_ConstraintsEverywhere_SelectBCs: [level]
ML_BSSN_ConstraintsEverywhere_SelectBCs
            GROUP ML_BSSN_ConstraintsEverywhere_ApplyBCs: Apply BCs
for groups set in ML_BSSN_ConstraintsEverywhere
              GROUP BoundaryConditions: Execute all boundary
conditions
                Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
              Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
          GROUP ML_BSSN_ConstraintsInterior_bc_group:
ML_BSSN_ConstraintsInterior
            ML_BSSN::ML_BSSN_ConstraintsInterior_SelectBCs: [level]
ML_BSSN_ConstraintsInterior_SelectBCs
            GROUP ML_BSSN_ConstraintsInterior_ApplyBCs: Apply BCs for
groups set in ML_BSSN_ConstraintsInterior
              GROUP BoundaryConditions: Execute all boundary
conditions
                Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
              Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
        SpaceMask::MaskOne: Set mask to one
        GRHydro::GRHydro_SetupMask: Initialize the atmosphere mask
      [CCTK_BASEGRID]
        ADMBase::ADMBase_SetShiftStateOn: Set the shift_state variable
to 1
        ADMBase::ADMBase_SetDtLapseStateOn: Set the dtlapse_state
variable to 1
        ADMBase::ADMBase_SetDtShiftStateOn: Set the dtshift_state
variable to 1
        ADMMacros::ADMMacros_SetLocalSpatialOrder: Initialize the
local_spatial_order
        CartGrid3D::SpatialSpacings: Set up ranges for spatial 3D
Cartesian coordinates (on all grids)
        CartGrid3D::SpatialCoordinates: Set up spatial 3D Cartesian
coordinates on the GH
        SphericalSurface::SphericalSurface_SetupRes: [global] [loop-
local] Set surface resolution automatically
        Dissipation::dissipation_basegrid: Ensure that there are
enough ghost zones
        GRHydro::GRHydro_Reset_Execution_Flags: [global] Initially set
execution flags to 'YEAH, Execute'!
        GRHydro::GRHydro_InitSymBound: Schedule symmetries and check
shift state
        GRHydro::reset_GRHydro_C2P_failed: Initialise the mask
function that contains the points where C2P has failed (at BASEGRID)
        ML_BSSN::ML_BSSN_CheckBoundaries: [meta] check boundaries
treatment
        NaNChecker::NaNChecker_ResetCounter: [global] Reset the
NaNChecker::NaNsFound counter
        SpaceMask::MaskZero: Initialise mask to zero
        SpaceMask::MaskOne: Set old style mask to one
        SphericalSurface::SphericalSurface_Setup: [global] Calculate
surface coordinate descriptors
        GROUP MaskBase_SetupMask: Set up the weight function
          GROUP MaskBase_SetupMaskAll: Set up the weight function
            CarpetReduce::MaskBase_AllocateMask: [global] Allocate the
weight function
            CarpetReduce::MaskBase_InitMask: [global] [loop-local]
Initialise the weight function
            GROUP SetupIMaskInternal: Set up the integer weight
function (schedule other routines in here)
              CarpetReduce::CoordBase_SetupMask: [global] [loop-local]
Set up the outer boundaries of the weight function
              CarpetReduce::CarpetMaskSetup: [global] [loop-singlemap]
Set up the weight function for the restriction regions
            GROUP SetupIMask: Set up the integer weight function
(schedule other routines in here)
            CarpetReduce::MaskBase_SetMask: [global] [loop-local] Set
the weight function
            GROUP SetupMask: Set up the real weight function (schedule
other routines in here)
            CarpetReduce::MaskBase_TestMask: [global] Test the weight
function
        SphericalSurface::SphericalSurface_Set: [global] Set surface
radii to be used for initial setup in other thorns
        GROUP SphericalSurface_HasBeenSet: Set the spherical surfaces
before this group, and use it afterwards
          SphericalSurface::SphericalSurface_CheckState: [global] Test
the state of the spherical surfaces
        SymBase::SymBase_Check: Check whether the driver set up the
grid consistently
        TerminationTrigger::TerminationTrigger_ResetTrigger: Clear
trigger state
        TerminationTrigger::TerminationTrigger_StartTimer: Start timer
        TerminationTrigger::TerminationTrigger_CreateFile: Create
```

```
termination file
        Time::Time_Initialise: [global] Initialise Time variables
        Time::TemporalSpacings: [singlemap] Set timestep based on
Courant condition (courant_static)
      [CCTK_INITIAL]
        StaticConformal::StaticConformal_InitialiseState: Set the
conformal_state variable to 0
        GROUP ADMBase_InitialData: Schedule group for calculating ADM
initial data
        GRHydro::GRHydro_EOSHandle: [global] Set the EOS number
        CarpetIOASCII::CarpetIOASCIIInit: [global] Initialisation
routine
        CarpetIOBasic::CarpetIOBasicInit: [global] Initialisation
routine
        CarpetIOHDF5::CarpetIOHDF5_Init: [global] Initialisation
routine
        CarpetIOScalar::CarpetIOScalarInit: [global] Initialisation
routine
        GRHydro::GRHydro_Rho_Minima_Setup: Set up minimum for the
rest-mass density in the atmosphere (before intial data)
        GRHydro::GRHydro_SetupMask: Initialize the atmosphere mask
        GRHydro::GRHydro_RefinementLevel: Calculate current refinement
level
        GROUP ADMBase_InitialGauge: Schedule group for the ADM initial
gauge condition
          ADMBase::ADMBase_DtLapseZero: Set the dtlapse to 0 at all
points
          ADMBase::ADMBase_DtShiftZero: Set the dtshift to 0 at all
points
        GROUP HydroBase_Initial: HydroBase initial data group
          GROUP GRHydro_Initial: GRHydro initial data group
          GROUP HydroBase_ExcisionMaskSetup: Set up hydro excision
mask
            HydroBase::HydroBase_InitExcisionMask: Initialize hydro
excision mask to 'no excision everywhere'
          GROUP TOV_Initial_Data: Group for the TOV initial data
            TOVSolver::TOV_C_Integrate_RHS: [global] Integrate the 1d
equations for the TOV star
            TOVSolver::TOV_C_Exact: Set up the 3d quantities for the
TOV star
        GROUP ADMBase_PostInitial: Schedule group for modifying the
ADM initial data, such as e.g. adding noise
        GRHydro::GRHydro_SqrtSpatialDeterminant: Calculate sdetg
        GRHydro::GRHydro_InitialAtmosphereReset: Use mask to enforce
atmosphere at initial time
        ML_BSSN::ML_BSSN_InitialADMBase1Everywhere:
ML_BSSN_InitialADMBase1Everywhere
        ML_BSSN::ML_BSSN_InitialADMBase2Interior:
ML_BSSN_InitialADMBase2Interior
```

```
        ML_BSSN::ML_BSSN_InitialADMBase2BoundaryScalar:
ML_BSSN_InitialADMBase2BoundaryScalar
        ML_BSSN_Helper::ML_BSSN_ExtrapolateGammas: Extrapolate Gammas
and time derivatives of lapse and shift
        MoL::MoL_StartLoop: [level] Initialise the step size control
        GROUP HydroBase_Prim2ConInitial: Recover the conservative
variables from the primitive variables
          GRHydro::Primitive2ConservativePolyCells: Convert initial
data given in primive variables to conserved variables
      [CCTK_POSTINITIAL]
        CarpetIOHDF5::CarpetIOHDF5_CloseFiles: [global] Close all
filereader input files
        GRHydro::GRHydro_Scalar_Setup: Set up and check scalars for
efficiency
        GROUP MoL_PostStepModify: The group for physics thorns to
schedule enforcing constraints
          ML_BSSN::ML_BSSN_EnforceEverywhere:
ML_BSSN_EnforceEverywhere
        GROUP MoL_PostStep: Ensure that everything is correct after
the initial data have been set up
          ML_BSSN::ML_BSSN_SelectBoundConds: [level] select boundary
conditions
          GRHydro::GRHydro_RefinementLevel: Calculate current
refinement level
          GRHydro::GRHydro_SetLastMoLPostStep: [level] Set grid scalar
InLastMoLPostStep if this is the last MoL PostStep call
          GROUP ML_BSSN_ApplyBCs: Apply boundary conditions controlled
by thorn Boundary
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
              ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
            Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
          ML_BSSN::ML_BSSN_ADMBaseInterior: ML_BSSN_ADMBaseInterior
          ML_BSSN::ML_BSSN_ADMBaseBoundaryScalar:
ML_BSSN_ADMBaseBoundaryScalar
          ML_BSSN::ML_BSSN_ADMBaseEverywhere:
ML_BSSN_ADMBaseEverywhere
          ML_BSSN_Helper::ML_BSSN_ADMBase_SelectBCs: [level] Select
boundary conditions for ADMBase variables
          GROUP ML_BSSN_ADMBase_ApplyBCs: Apply boundary conditions to
ADMBase variables
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
```

```
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
              ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
            Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
          GROUP ADMBase_SetADMVars: Set the ADM variables before this
group, and use them afterwards
          GROUP HydroBase_PostStep: Post step tasks for hydro thorns
            GROUP GRHydro_PostStep: Post step tasks for GRHydro
            GROUP GRHydro_AtmosphereMaskBoundaries: Apply boundary
conditions to primitives
              GRHydro::GRHydro_SelectAtmosphereMaskBoundaries: [level]
Select atmosphere mask for boundary conditions
              GROUP GRHydro_ApplyAtmosphereMaskBCs: Apply boundary
conditions to real-valued atmosphere mask
                GROUP BoundaryConditions: Execute all boundary
conditions
                  Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                  CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                  ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
                Boundary::Boundary_ClearSelection: [level] Unselect
all grid variables for boundary conditions
            GRHydro::GRHydroPostSyncAtmosphereMask: Set integer
atmosphere mask from synchronized real atmosphere mask
            if (GRHydro::InLastMoLPostStep)
              GRHydro::GRHydro_AtmosphereReset: Reset the atmosphere
            GROUP HydroBase_Boundaries: HydroBase-internal Boundary
conditions group
              GROUP Do_GRHydro_Boundaries: GRHydro Boundary conditions
group
              GROUP HydroBase_Select_Boundaries: Group to schedule the
boundary condition functions
                if (GRHydro::execute_MoL_PostStep)
                  GRHydro::GRHydro_Bound: [level] Select GRHydro
boundary conditions
              GROUP HydroBase_ApplyBCs: Apply the boundary conditions
of HydroBase
                GROUP BoundaryConditions: Execute all boundary
conditions
                  Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                  CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                  ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
                Boundary::Boundary_ClearSelection: [level] Unselect
```

```
all grid variables for boundary conditions
            GROUP HydroBase_Con2Prim: Convert from conservative to
primitive variables
              if (GRHydro::execute_MoL_Step)
                GRHydro::GRHydro_SqrtSpatialDeterminant: Calculate
sdetg
              if (GRHydro::execute_MoL_PostStep)
                GRHydro::Con2Prim: Convert back to primitive variables
(polytype)
          GROUP SetTmunu: Group for calculating the stress-energy
tensor
            TmunuBase::TmunuBase_ZeroTmunu: Initialise the stress-
energy tensor to zero
            GROUP AddToTmunu: Add to the stress-energy tensor here
              GRHydro::GRHydro_Tmunu: Compute the energy-momentum
tensor
        GROUP MoL_PseudoEvolution: Calculate pseudo-evolved quantities
          GROUP ADMBase_SetADMVars: Set the ADM variables before this
group, and use them afterwards
          GROUP ML_BSSN_ConstraintsEverywhere_group:
ML_BSSN_ConstraintsEverywhere
            ML_BSSN::ML_BSSN_ConstraintsEverywhere:
ML_BSSN_ConstraintsEverywhere
            GROUP ML_BSSN_ConstraintsEverywhere_bc_group:
ML_BSSN_ConstraintsEverywhere
              ML_BSSN::ML_BSSN_ConstraintsEverywhere_SelectBCs:
[level] ML_BSSN_ConstraintsEverywhere_SelectBCs
              GROUP ML_BSSN_ConstraintsEverywhere_ApplyBCs: Apply BCs
for groups set in ML_BSSN_ConstraintsEverywhere
                GROUP BoundaryConditions: Execute all boundary
conditions
                  Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                  CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                  ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
                Boundary::Boundary_ClearSelection: [level] Unselect
all grid variables for boundary conditions
          GROUP ML_BSSN_ConstraintsInterior_group:
ML_BSSN_ConstraintsInterior
            ML_BSSN::ML_BSSN_ConstraintsInterior:
ML_BSSN_ConstraintsInterior
            GROUP ML_BSSN_ConstraintsInterior_bc_group:
ML_BSSN_ConstraintsInterior
              ML_BSSN::ML_BSSN_ConstraintsInterior_SelectBCs: [level]
ML_BSSN_ConstraintsInterior_SelectBCs
              GROUP ML_BSSN_ConstraintsInterior_ApplyBCs: Apply BCs
for groups set in ML_BSSN_ConstraintsInterior
```

```
                GROUP BoundaryConditions: Execute all boundary
conditions
                  Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                  CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                  ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
                Boundary::Boundary_ClearSelection: [level] Unselect
all grid variables for boundary conditions
      Initialise finer grids recursively
      Restrict from finer grids
      [CCTK_POSTRESTRICTINITIAL]
        GROUP MoL_PostStep: Ensure that everything is correct after
restriction
          ML_BSSN::ML_BSSN_SelectBoundConds: [level] select boundary
conditions
          GRHydro::GRHydro_RefinementLevel: Calculate current
refinement level
          GRHydro::GRHydro_SetLastMoLPostStep: [level] Set grid scalar
InLastMoLPostStep if this is the last MoL PostStep call
          GROUP ML_BSSN_ApplyBCs: Apply boundary conditions controlled
by thorn Boundary
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
              ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
            Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
          ML_BSSN::ML_BSSN_ADMBaseInterior: ML_BSSN_ADMBaseInterior
          ML_BSSN::ML_BSSN_ADMBaseBoundaryScalar:
ML_BSSN_ADMBaseBoundaryScalar
          ML_BSSN::ML_BSSN_ADMBaseEverywhere:
ML_BSSN_ADMBaseEverywhere
          ML_BSSN_Helper::ML_BSSN_ADMBase_SelectBCs: [level] Select
boundary conditions for ADMBase variables
          GROUP ML_BSSN_ADMBase_ApplyBCs: Apply boundary conditions to
ADMBase variables
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
              ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
            Boundary::Boundary_ClearSelection: [level] Unselect all
```

```
grid variables for boundary conditions
          GROUP ADMBase_SetADMVars: Set the ADM variables before this
group, and use them afterwards
          GROUP HydroBase_PostStep: Post step tasks for hydro thorns
            GROUP GRHydro_PostStep: Post step tasks for GRHydro
            GROUP GRHydro_AtmosphereMaskBoundaries: Apply boundary
conditions to primitives
              GRHydro::GRHydro_SelectAtmosphereMaskBoundaries: [level]
Select atmosphere mask for boundary conditions
              GROUP GRHydro_ApplyAtmosphereMaskBCs: Apply boundary
conditions to real-valued atmosphere mask
                GROUP BoundaryConditions: Execute all boundary
conditions
                  Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                  CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                  ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
                Boundary::Boundary_ClearSelection: [level] Unselect
all grid variables for boundary conditions
            GRHydro::GRHydroPostSyncAtmosphereMask: Set integer
atmosphere mask from synchronized real atmosphere mask
            if (GRHydro::InLastMoLPostStep)
              GRHydro::GRHydro_AtmosphereReset: Reset the atmosphere
            GROUP HydroBase_Boundaries: HydroBase-internal Boundary
conditions group
              GROUP Do_GRHydro_Boundaries: GRHydro Boundary conditions
group
              GROUP HydroBase_Select_Boundaries: Group to schedule the
boundary condition functions
                if (GRHydro::execute_MoL_PostStep)
                  GRHydro::GRHydro_Bound: [level] Select GRHydro
boundary conditions
              GROUP HydroBase_ApplyBCs: Apply the boundary conditions
of HydroBase
                GROUP BoundaryConditions: Execute all boundary
conditions
                  Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                  CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                  ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
                Boundary::Boundary_ClearSelection: [level] Unselect
all grid variables for boundary conditions
            GROUP HydroBase_Con2Prim: Convert from conservative to
primitive variables
              if (GRHydro::execute_MoL_Step)
```

```
                GRHydro::GRHydro_SqrtSpatialDeterminant: Calculate
sdetg
              if (GRHydro::execute_MoL_PostStep)
                GRHydro::Con2Prim: Convert back to primitive variables
(polytype)
          GROUP SetTmunu: Group for calculating the stress-energy
tensor
            TmunuBase::TmunuBase_ZeroTmunu: Initialise the stress-
energy tensor to zero
            GROUP AddToTmunu: Add to the stress-energy tensor here
              GRHydro::GRHydro_Tmunu: Compute the energy-momentum
tensor
        GROUP MoL_PseudoEvolutionBoundaries: Apply boundary conditions
to pseudo-evolved quantities
          GROUP ML_BSSN_ConstraintsEverywhere_bc_group:
ML_BSSN_ConstraintsEverywhere
            ML_BSSN::ML_BSSN_ConstraintsEverywhere_SelectBCs: [level]
ML_BSSN_ConstraintsEverywhere_SelectBCs
            GROUP ML_BSSN_ConstraintsEverywhere_ApplyBCs: Apply BCs
for groups set in ML_BSSN_ConstraintsEverywhere
              GROUP BoundaryConditions: Execute all boundary
conditions
                Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
              Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
          GROUP ML_BSSN_ConstraintsInterior_bc_group:
ML_BSSN_ConstraintsInterior
            ML_BSSN::ML_BSSN_ConstraintsInterior_SelectBCs: [level]
ML_BSSN_ConstraintsInterior_SelectBCs
            GROUP ML_BSSN_ConstraintsInterior_ApplyBCs: Apply BCs for
groups set in ML_BSSN_ConstraintsInterior
              GROUP BoundaryConditions: Execute all boundary
conditions
                Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
              Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
      [CCTK_POSTPOSTINITIAL]
        GRHydro::GRHydro_Rho_Minima_Setup_Final: Set the value of the
rest-mass density of the atmosphere which will be used during the
```

```
evolution
        GRHydro::GRHydro_InitialAtmosphereReset: Use mask to enforce
atmosphere at initial time
        GROUP Con2Prim: Convert from conservative to primitive
variables (might be redundant)
          if (GRHydro::execute_MoL_Step)
            GRHydro::GRHydro_SqrtSpatialDeterminant: Calculate sdetg
          if (GRHydro::execute_MoL_PostStep)
            GRHydro::Con2Prim: Convert back to primitive variables
(polytype)
        GROUP SetTmunu: Calculate the stress-energy tensor
          TmunuBase::TmunuBase_ZeroTmunu: Initialise the stress-energy
tensor to zero
          GROUP AddToTmunu: Add to the stress-energy tensor here
            GRHydro::GRHydro_Tmunu: Compute the energy-momentum tensor
        GROUP ADMConstraintsGroup: Evaluate ADM constraints, and
perform symmetry boundary conditions
        TOVSolver::TOV_C_FreeMemory: [global] Free memory from
TOVSolver_C
      [CCTK_POSTSTEP]
        SphericalSurface::SphericalSurface_Set: [global] Set surface
radii
        GRHydro::GRHydro_RefinementLevel: Calculate current refinement
level (for the check of the C2P mask)
        GRHydro::check_GRHydro_C2P_failed: Check the mask function
that contains the points where C2P has failed and report an error in
case a failure is found
        GROUP HydroBase_ExcisionHasBeenSet: Group to schedule thorns
changing the mask before and thorns using the mask after
        GROUP zzz_NaNChecker_NaNCheck: Check for NaNs and count them
in NaNChecker::NaNsFound
          NaNChecker::NaNChecker_NaNCheck_Prepare: [level] Prepare
data structures to check for NaNs
          NaNChecker::NaNChecker_NaNCheck_Check: [local] Check for
NaNs
          NaNChecker::NaNChecker_NaNCheck_Finish: [level] Count NaNs
in NaNChecker::NaNsFound
        NaNChecker::NaNChecker_TakeAction: [global] [loop-level]
Output NaNChecker::NaNmask and take action according to
NaNChecker::action_if_found
        SpaceMask::CheckMask: Ensure that all mask values are legal
        GROUP SphericalSurface_HasBeenSet: Set the spherical surfaces
before this group, and use it afterwards
          SphericalSurface::SphericalSurface_CheckState: [global] Test
the state of the spherical surfaces
        Dissipation::setup_epsdis: Setup spatially varying dissipation
    endif
    if (recover initial data)
      [CCTK_BASEGRID]
```

```
        ADMBase::ADMBase_SetShiftStateOn: Set the shift_state variable
to 1
        ADMBase::ADMBase_SetDtLapseStateOn: Set the dtlapse_state
variable to 1
        ADMBase::ADMBase_SetDtShiftStateOn: Set the dtshift_state
variable to 1
        ADMMacros::ADMMacros_SetLocalSpatialOrder: Initialize the
local_spatial_order
        CartGrid3D::SpatialSpacings: Set up ranges for spatial 3D
Cartesian coordinates (on all grids)
        CartGrid3D::SpatialCoordinates: Set up spatial 3D Cartesian
coordinates on the GH
        SphericalSurface::SphericalSurface_SetupRes: [global] [loop-
local] Set surface resolution automatically
        Dissipation::dissipation_basegrid: Ensure that there are
enough ghost zones
        GRHydro::GRHydro_Reset_Execution_Flags: [global] Initially set
execution flags to 'YEAH, Execute'!
        GRHydro::GRHydro_InitSymBound: Schedule symmetries and check
shift state
        GRHydro::reset_GRHydro_C2P_failed: Initialise the mask
function that contains the points where C2P has failed (at BASEGRID)
        ML_BSSN::ML_BSSN_CheckBoundaries: [meta] check boundaries
treatment
        NaNChecker::NaNChecker_ResetCounter: [global] Reset the
NaNChecker::NaNsFound counter
        SpaceMask::MaskZero: Initialise mask to zero
        SpaceMask::MaskOne: Set old style mask to one
        SphericalSurface::SphericalSurface_Setup: [global] Calculate
surface coordinate descriptors
        GROUP MaskBase_SetupMask: Set up the weight function
          GROUP MaskBase_SetupMaskAll: Set up the weight function
            CarpetReduce::MaskBase_AllocateMask: [global] Allocate the
weight function
            CarpetReduce::MaskBase_InitMask: [global] [loop-local]
Initialise the weight function
            GROUP SetupIMaskInternal: Set up the integer weight
function (schedule other routines in here)
              CarpetReduce::CoordBase_SetupMask: [global] [loop-local]
Set up the outer boundaries of the weight function
              CarpetReduce::CarpetMaskSetup: [global] [loop-singlemap]
Set up the weight function for the restriction regions
            GROUP SetupIMask: Set up the integer weight function
(schedule other routines in here)
            CarpetReduce::MaskBase_SetMask: [global] [loop-local] Set
the weight function
            GROUP SetupMask: Set up the real weight function (schedule
other routines in here)
            CarpetReduce::MaskBase_TestMask: [global] Test the weight
```

```
function
        SphericalSurface::SphericalSurface_Set: [global] Set surface
radii to be used for initial setup in other thorns
        GROUP SphericalSurface_HasBeenSet: Set the spherical surfaces
before this group, and use it afterwards
          SphericalSurface::SphericalSurface_CheckState: [global] Test
the state of the spherical surfaces
        SymBase::SymBase_Check: Check whether the driver set up the
grid consistently
        TerminationTrigger::TerminationTrigger_ResetTrigger: Clear
trigger state
        TerminationTrigger::TerminationTrigger_StartTimer: Start timer
        TerminationTrigger::TerminationTrigger_CreateFile: Create
termination file
        Time::Time_Initialise: [global] Initialise Time variables
        Time::TemporalSpacings: [singlemap] Set timestep based on
Courant condition (courant_static)
      [CCTK_RECOVER_VARIABLES]
      [CCTK_POST_RECOVER_VARIABLES]
        CarpetIOHDF5::CarpetIOHDF5_InitCheckpointingIntervals:
[global] Initialisation of checkpointing intervals after recovery
        GROUP MaskBase_SetupMask: Set up the weight function
          GROUP MaskBase_SetupMaskAll: Set up the weight function
            CarpetReduce::MaskBase_AllocateMask: [global] Allocate the
weight function
            CarpetReduce::MaskBase_InitMask: [global] [loop-local]
Initialise the weight function
            GROUP SetupIMaskInternal: Set up the integer weight
function (schedule other routines in here)
              CarpetReduce::CoordBase_SetupMask: [global] [loop-local]
Set up the outer boundaries of the weight function
              CarpetReduce::CarpetMaskSetup: [global] [loop-singlemap]
Set up the weight function for the restriction regions
            GROUP SetupIMask: Set up the integer weight function
(schedule other routines in here)
            CarpetReduce::MaskBase_SetMask: [global] [loop-local] Set
the weight function
            GROUP SetupMask: Set up the real weight function (schedule
other routines in here)
            CarpetReduce::MaskBase_TestMask: [global] Test the weight
function
        GRHydro::GRHydro_EOSHandle: [global] Set the EOS number
        GRHydro::GRHydroCopyIntegerMask: Initialize the real valued
atmosphere mask after checkpoint recovery
        GRHydro::GRHydro_SqrtSpatialDeterminant: Calculate sdetg
        GROUP HydroBase_ExcisionMaskSetup: Set up hydro excision mask
          HydroBase::HydroBase_InitExcisionMask: Initialize hydro
excision mask to 'no excision everywhere'
        GROUP MoL_PostStep: Ensure that everything is correct after
```

```
recovery
          ML_BSSN::ML_BSSN_SelectBoundConds: [level] select boundary
conditions
          GRHydro::GRHydro_RefinementLevel: Calculate current
refinement level
          GRHydro::GRHydro_SetLastMoLPostStep: [level] Set grid scalar
InLastMoLPostStep if this is the last MoL PostStep call
          GROUP ML_BSSN_ApplyBCs: Apply boundary conditions controlled
by thorn Boundary
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
              ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
            Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
          ML_BSSN::ML_BSSN_ADMBaseInterior: ML_BSSN_ADMBaseInterior
          ML_BSSN::ML_BSSN_ADMBaseBoundaryScalar:
ML_BSSN_ADMBaseBoundaryScalar
          ML_BSSN::ML_BSSN_ADMBaseEverywhere:
ML_BSSN_ADMBaseEverywhere
          ML_BSSN_Helper::ML_BSSN_ADMBase_SelectBCs: [level] Select
boundary conditions for ADMBase variables
          GROUP ML_BSSN_ADMBase_ApplyBCs: Apply boundary conditions to
ADMBase variables
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
              ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
            Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
          GROUP ADMBase_SetADMVars: Set the ADM variables before this
group, and use them afterwards
          GROUP HydroBase_PostStep: Post step tasks for hydro thorns
            GROUP GRHydro_PostStep: Post step tasks for GRHydro
            GROUP GRHydro_AtmosphereMaskBoundaries: Apply boundary
conditions to primitives
              GRHydro::GRHydro_SelectAtmosphereMaskBoundaries: [level]
Select atmosphere mask for boundary conditions
              GROUP GRHydro_ApplyAtmosphereMaskBCs: Apply boundary
conditions to real-valued atmosphere mask
                GROUP BoundaryConditions: Execute all boundary
conditions
                  Boundary::Boundary_ApplyPhysicalBCs: Apply all
```

```
requested local physical boundary conditions
                  CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                  ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
                Boundary::Boundary_ClearSelection: [level] Unselect
all grid variables for boundary conditions
            GRHydro::GRHydroPostSyncAtmosphereMask: Set integer
atmosphere mask from synchronized real atmosphere mask
            if (GRHydro::InLastMoLPostStep)
              GRHydro::GRHydro_AtmosphereReset: Reset the atmosphere
            GROUP HydroBase_Boundaries: HydroBase-internal Boundary
conditions group
              GROUP Do_GRHydro_Boundaries: GRHydro Boundary conditions
group
              GROUP HydroBase_Select_Boundaries: Group to schedule the
boundary condition functions
                if (GRHydro::execute_MoL_PostStep)
                  GRHydro::GRHydro_Bound: [level] Select GRHydro
boundary conditions
              GROUP HydroBase_ApplyBCs: Apply the boundary conditions
of HydroBase
                GROUP BoundaryConditions: Execute all boundary
conditions
                  Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                  CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                  ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
                Boundary::Boundary_ClearSelection: [level] Unselect
all grid variables for boundary conditions
            GROUP HydroBase_Con2Prim: Convert from conservative to
primitive variables
              if (GRHydro::execute_MoL_Step)
                GRHydro::GRHydro_SqrtSpatialDeterminant: Calculate
sdetg
              if (GRHydro::execute_MoL_PostStep)
                GRHydro::Con2Prim: Convert back to primitive variables
(polytype)
          GROUP SetTmunu: Group for calculating the stress-energy
tensor
            TmunuBase::TmunuBase_ZeroTmunu: Initialise the stress-
energy tensor to zero
            GROUP AddToTmunu: Add to the stress-energy tensor here
              GRHydro::GRHydro_Tmunu: Compute the energy-momentum
tensor
        GROUP zzz_NaNChecker_NaNCheck: Check for NaNs and count them
in NaNChecker::NaNsFound
```

```
          NaNChecker::NaNChecker_NaNCheck_Prepare: [level] Prepare
data structures to check for NaNs
          NaNChecker::NaNChecker_NaNCheck_Check: [local] Check for
NaNs
          NaNChecker::NaNChecker_NaNCheck_Finish: [level] Count NaNs
in NaNChecker::NaNsFound
        NaNChecker::NaNChecker_TakeAction: [global] [loop-level]
Output NaNChecker::NaNmask and take action according to
NaNChecker::action_if_found
        TerminationTrigger::TerminationTrigger_ResetMinutes: [global]
Reset Watchtime
    endif
    if (checkpoint initial data)
      [CCTK_CPINITIAL]
        CarpetIOHDF5::CarpetIOHDF5_InitialDataCheckpoint: [meta]
Initial data checkpoint routine
    endif
    if (analysis)
      [CCTK_ANALYSIS]
        CarpetLib::CarpetLib_printtimestats: [global] Print timing
statistics if desired
        CarpetLib::CarpetLib_printmemstats: [global] Print memory
statistics if desired
        LoopControl::LC_statistics_analysis: [meta] Output LoopControl
statistics
        GROUP ML_BSSN_EvolutionAnalysis: Calculate RHS at analysis
          ML_BSSN::ML_BSSN_EvolutionAnalysisInit:
ML_BSSN_EvolutionAnalysisInit
          ML_BSSN::ML_BSSN_EvolutionAnalysisInterior:
ML_BSSN_EvolutionAnalysisInterior
          ML_BSSN_Helper::ML_BSSN_NewRad: Apply NewRad boundary
conditions to RHS
        TerminationTrigger::TerminationTrigger_CheckWalltime: Check
elapsed job walltime
        TerminationTrigger::TerminationTrigger_CheckSignal: Check if
we received a termination signal
        TerminationTrigger::TerminationTrigger_CheckFile: Check
termination file
  endif
  Output grid variables

  do loop over timesteps
    [CCTK_PREREGRID]
    Change grid hierarchy
    [CCTK_POSTREGRID]
      CartGrid3D::SpatialCoordinates: Set Coordinates after regridding
      GROUP MaskBase_SetupMask: Set up the weight function
        GROUP MaskBase_SetupMaskAll: Set up the weight function
          CarpetReduce::MaskBase_AllocateMask: [global] Allocate the
```

```
weight function
          CarpetReduce::MaskBase_InitMask: [global] [loop-local]
Initialise the weight function
          GROUP SetupIMaskInternal: Set up the integer weight function
(schedule other routines in here)
            CarpetReduce::CoordBase_SetupMask: [global] [loop-local]
Set up the outer boundaries of the weight function
            CarpetReduce::CarpetMaskSetup: [global] [loop-singlemap]
Set up the weight function for the restriction regions
          GROUP SetupIMask: Set up the integer weight function
(schedule other routines in here)
          CarpetReduce::MaskBase_SetMask: [global] [loop-local] Set
the weight function
          GROUP SetupMask: Set up the real weight function (schedule
other routines in here)
          CarpetReduce::MaskBase_TestMask: [global] Test the weight
function
      Dissipation::setup_epsdis: Setup spatially varying dissipation
      SpaceMask::MaskZero: Initialise mask to zero
      GROUP HydroBase_ExcisionMaskSetup: Set up hydro excision mask
        HydroBase::HydroBase_InitExcisionMask: Initialize hydro
excision mask to 'no excision everywhere'
      SpaceMask::MaskOne: Set mask to one
      GRHydro::GRHydro_SetupMask: Initialize the atmosphere mask
      GROUP MoL_PostStep: Ensure that everything is correct after
regridding
        ML_BSSN::ML_BSSN_SelectBoundConds: [level] select boundary
conditions
        GRHydro::GRHydro_RefinementLevel: Calculate current refinement
level
        GRHydro::GRHydro_SetLastMoLPostStep: [level] Set grid scalar
InLastMoLPostStep if this is the last MoL PostStep call
        GROUP ML_BSSN_ApplyBCs: Apply boundary conditions controlled
by thorn Boundary
          GROUP BoundaryConditions: Execute all boundary conditions
            Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
            CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
            ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
          Boundary::Boundary_ClearSelection: [level] Unselect all grid
variables for boundary conditions
        ML_BSSN::ML_BSSN_ADMBaseInterior: ML_BSSN_ADMBaseInterior
        ML_BSSN::ML_BSSN_ADMBaseBoundaryScalar:
ML_BSSN_ADMBaseBoundaryScalar
        ML_BSSN::ML_BSSN_ADMBaseEverywhere: ML_BSSN_ADMBaseEverywhere
        ML_BSSN_Helper::ML_BSSN_ADMBase_SelectBCs: [level] Select
boundary conditions for ADMBase variables
        GROUP ML_BSSN_ADMBase_ApplyBCs: Apply boundary conditions to
```

```
ADMBase variables
          GROUP BoundaryConditions: Execute all boundary conditions
            Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
            CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
            ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
          Boundary::Boundary_ClearSelection: [level] Unselect all grid
variables for boundary conditions
        GROUP ADMBase_SetADMVars: Set the ADM variables before this
group, and use them afterwards
        GROUP HydroBase_PostStep: Post step tasks for hydro thorns
          GROUP GRHydro_PostStep: Post step tasks for GRHydro
          GROUP GRHydro_AtmosphereMaskBoundaries: Apply boundary
conditions to primitives
            GRHydro::GRHydro_SelectAtmosphereMaskBoundaries: [level]
Select atmosphere mask for boundary conditions
            GROUP GRHydro_ApplyAtmosphereMaskBCs: Apply boundary
conditions to real-valued atmosphere mask
              GROUP BoundaryConditions: Execute all boundary
conditions
                Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
              Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
          GRHydro::GRHydroPostSyncAtmosphereMask: Set integer
atmosphere mask from synchronized real atmosphere mask
          if (GRHydro::InLastMoLPostStep)
            GRHydro::GRHydro_AtmosphereReset: Reset the atmosphere
          GROUP HydroBase_Boundaries: HydroBase-internal Boundary
conditions group
            GROUP Do_GRHydro_Boundaries: GRHydro Boundary conditions
group
            GROUP HydroBase_Select_Boundaries: Group to schedule the
boundary condition functions
              if (GRHydro::execute_MoL_PostStep)
                GRHydro::GRHydro_Bound: [level] Select GRHydro
boundary conditions
            GROUP HydroBase_ApplyBCs: Apply the boundary conditions of
HydroBase
              GROUP BoundaryConditions: Execute all boundary
conditions
                Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
```

```
                CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
              Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
          GROUP HydroBase_Con2Prim: Convert from conservative to
primitive variables
            if (GRHydro::execute_MoL_Step)
              GRHydro::GRHydro_SqrtSpatialDeterminant: Calculate sdetg
            if (GRHydro::execute_MoL_PostStep)
              GRHydro::Con2Prim: Convert back to primitive variables
(polytype)
        GROUP SetTmunu: Group for calculating the stress-energy tensor
          TmunuBase::TmunuBase_ZeroTmunu: Initialise the stress-energy
tensor to zero
          GROUP AddToTmunu: Add to the stress-energy tensor here
            GRHydro::GRHydro_Tmunu: Compute the energy-momentum tensor
      GROUP MoL_PseudoEvolutionBoundaries: Apply boundary conditions
to pseudo-evolved quantities
        GROUP ML_BSSN_ConstraintsEverywhere_bc_group:
ML_BSSN_ConstraintsEverywhere
          ML_BSSN::ML_BSSN_ConstraintsEverywhere_SelectBCs: [level]
ML_BSSN_ConstraintsEverywhere_SelectBCs
          GROUP ML_BSSN_ConstraintsEverywhere_ApplyBCs: Apply BCs for
groups set in ML_BSSN_ConstraintsEverywhere
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
              ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
            Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
        GROUP ML_BSSN_ConstraintsInterior_bc_group:
ML_BSSN_ConstraintsInterior
          ML_BSSN::ML_BSSN_ConstraintsInterior_SelectBCs: [level]
ML_BSSN_ConstraintsInterior_SelectBCs
          GROUP ML_BSSN_ConstraintsInterior_ApplyBCs: Apply BCs for
groups set in ML_BSSN_ConstraintsInterior
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
              ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
            Boundary::Boundary_ClearSelection: [level] Unselect all
```

```
grid variables for boundary conditions
    Rotate timelevels
    iteration = iteration+1
    t = t+dt
    [CCTK_PRESTEP]
      CoordGauge::Einstein_SetNextSlicing: Identify the slicing for
the next iteration
      GRHydro::reset_GRHydro_C2P_failed: Reset the mask function that
contains the points where C2P has failed (at PRESTEP)
      LoopControl::LC_steer: [meta] Update LoopControl algorithm
preferences
      NaNChecker::NaNChecker_ResetCounter: [global] Reset the
NaNChecker::NaNsFound counter
    [CCTK_EVOL]
      MoL::MoL_StartLoop: [level] Initialise the step size control
      while (MoL::MoL_Stepsize_Bad)
        GROUP MoL_Evolution: A single Cactus evolution step using MoL
          GROUP MoL_StartStep: MoL internal setup for the evolution
step
            MoL::MoL_SetCounter: [level] Set the counter for the ODE
method to loop over
            MoL::MoL_SetTime: [level] Ensure the correct time and
timestep are used
            MoL::MoL_AllocateScratchSpace: [level] Allocate storage
for scratch levels
          GROUP MoL_PreStep: Physics thorns can schedule preloop setup
routines in here
            GRHydro::GRHydro_Scalar_Setup: Set up and check scalars
for efficiency
          MoL::MoL_AllocateScratch: Allocate sufficient space for
array scratch variables
          MoL::MoL_InitialCopy: Ensure the data is in the correct
timelevel
          while (MoL::MoL_Intermediate_Step)
            GROUP MoL_Step: The loop over the intermediate steps for
the ODE integrator
              MoL::MoL_InitRHS: Initialise the RHS functions
              GROUP MoL_CalcRHS: Physics thorns schedule the
calculation of the discrete spatial operator in here
                GROUP HydroBase_RHS: Groups for scheduling tasks for
calculating RHS of hydro variables
                  if (GRHydro::execute_MoL_Step)
                    GROUP GRHydroRHS: Calculate the update terms
                      GRHydro::SourceTerms: Source term calculation
                      GRHydro::GRHydroStartLoop: [level] Set the
flux_direction variable
                      while (GRHydro::flux_direction)
                        GROUP FluxTerms: Calculation of intercell
fluxes
```

```
                          GRHydro::GRHydro_RefinementLevel: Calculate
current refinement level
                          GRHydro::Reconstruct: Reconstruct the
functions at the cell boundaries
                          GRHydro::Riemann: Solve the local Riemann
problems
                          GRHydro::UpdateCalcul: Calculate the update
term from the fluxes
                          GRHydro::GRHydroAdvanceLoop: [level]
Decrement the flux_direction variable
                      end while
                      GRHydro::GRHydroUpdateAtmosphereMask: Alter the
update terms if inside the atmosphere region
                ML_BSSN::ML_BSSN_EvolutionBoundaryScalar:
ML_BSSN_EvolutionBoundaryScalar
                GROUP ML_BSSN_EvolutionInteriorSplitBy:
                  ML_BSSN::ML_BSSN_EvolutionInteriorSplitBy1:
ML_BSSN_EvolutionInteriorSplitBy1
                  ML_BSSN::ML_BSSN_EvolutionInteriorSplitBy2:
ML_BSSN_EvolutionInteriorSplitBy2
                  ML_BSSN::ML_BSSN_EvolutionInteriorSplitBy3:
ML_BSSN_EvolutionInteriorSplitBy3
                ML_BSSN_Helper::ML_BSSN_NewRad: Apply NewRad boundary
conditions to RHS
              GROUP MoL_PostRHS: Modify RHS functions
                Dissipation::dissipation_add: Add Kreiss-Oliger
dissipation to the right hand sides
              GROUP MoL_RHSBoundaries: Any 'final' modifications to
the RHS functions (boundaries etc.)
              MoL::MoL_Add: Updates calculated with the efficient
Runge-Kutta 4 method
              MoL::MoL_DecrementCounter: [level] Alter the counter
number
              MoL::MoL_ResetTime: [level] If necessary, change the
time
              GROUP MoL_PostStepModify: The group for physics thorns
to schedule enforcing constraints
                ML_BSSN::ML_BSSN_EnforceEverywhere:
ML_BSSN_EnforceEverywhere
              GROUP MoL_PostStep: The group for physics thorns to
schedule boundary calls etc.
                ML_BSSN::ML_BSSN_SelectBoundConds: [level] select
boundary conditions
                GRHydro::GRHydro_RefinementLevel: Calculate current
refinement level
                GRHydro::GRHydro_SetLastMoLPostStep: [level] Set grid
scalar InLastMoLPostStep if this is the last MoL PostStep call
                GROUP ML_BSSN_ApplyBCs: Apply boundary conditions
controlled by thorn Boundary
```

```
                  GROUP BoundaryConditions: Execute all boundary
conditions
                    Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                    CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                    ReflectionSymmetry::ReflectionSymmetry_Apply:
Apply reflection symmetries
                  Boundary::Boundary_ClearSelection: [level] Unselect
all grid variables for boundary conditions
                ML_BSSN::ML_BSSN_ADMBaseInterior:
ML_BSSN_ADMBaseInterior
                ML_BSSN::ML_BSSN_ADMBaseBoundaryScalar:
ML_BSSN_ADMBaseBoundaryScalar
                ML_BSSN::ML_BSSN_ADMBaseEverywhere:
ML_BSSN_ADMBaseEverywhere
                ML_BSSN_Helper::ML_BSSN_ADMBase_SelectBCs: [level]
Select boundary conditions for ADMBase variables
                GROUP ML_BSSN_ADMBase_ApplyBCs: Apply boundary
conditions to ADMBase variables
                  GROUP BoundaryConditions: Execute all boundary
conditions
                    Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                    CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                    ReflectionSymmetry::ReflectionSymmetry_Apply:
Apply reflection symmetries
                  Boundary::Boundary_ClearSelection: [level] Unselect
all grid variables for boundary conditions
                GROUP ADMBase_SetADMVars: Set the ADM variables before
this group, and use them afterwards
                GROUP HydroBase_PostStep: Post step tasks for hydro
thorns
                  GROUP GRHydro_PostStep: Post step tasks for GRHydro
                  GROUP GRHydro_AtmosphereMaskBoundaries: Apply
boundary conditions to primitives
                    GRHydro::GRHydro_SelectAtmosphereMaskBoundaries:
[level] Select atmosphere mask for boundary conditions
                    GROUP GRHydro_ApplyAtmosphereMaskBCs: Apply
boundary conditions to real-valued atmosphere mask
                      GROUP BoundaryConditions: Execute all boundary
conditions
                        Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                        CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                        ReflectionSymmetry::ReflectionSymmetry_Apply:
Apply reflection symmetries
```

```
                      Boundary::Boundary_ClearSelection: [level]
Unselect all grid variables for boundary conditions
                  GRHydro::GRHydroPostSyncAtmosphereMask: Set integer
atmosphere mask from synchronized real atmosphere mask
                  if (GRHydro::InLastMoLPostStep)
                    GRHydro::GRHydro_AtmosphereReset: Reset the
atmosphere
                  GROUP HydroBase_Boundaries: HydroBase-internal
Boundary conditions group
                    GROUP Do_GRHydro_Boundaries: GRHydro Boundary
conditions group
                    GROUP HydroBase_Select_Boundaries: Group to
schedule the boundary condition functions
                      if (GRHydro::execute_MoL_PostStep)
                        GRHydro::GRHydro_Bound: [level] Select GRHydro
boundary conditions
                    GROUP HydroBase_ApplyBCs: Apply the boundary
conditions of HydroBase
                      GROUP BoundaryConditions: Execute all boundary
conditions
                        Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                        CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                        ReflectionSymmetry::ReflectionSymmetry_Apply:
Apply reflection symmetries
                      Boundary::Boundary_ClearSelection: [level]
Unselect all grid variables for boundary conditions
                  GROUP HydroBase_Con2Prim: Convert from conservative
to primitive variables
                    if (GRHydro::execute_MoL_Step)
                      GRHydro::GRHydro_SqrtSpatialDeterminant:
Calculate sdetg
                    if (GRHydro::execute_MoL_PostStep)
                      GRHydro::Con2Prim: Convert back to primitive
variables (polytype)
                GROUP SetTmunu: Group for calculating the stress-
energy tensor
                  TmunuBase::TmunuBase_ZeroTmunu: Initialise the
stress-energy tensor to zero
                  GROUP AddToTmunu: Add to the stress-energy tensor
here
                    GRHydro::GRHydro_Tmunu: Compute the energy-
momentum tensor
              GRHydro::GRHydro_ClearLastMoLPostStep: [level] Reset
InLastMoLPostStep to zero
              MoL::MoL_ResetDeltaTime: [level] If necessary, change
the timestep
          end while
          MoL::MoL_FinishLoop: [level] Control the step size
          MoL::MoL_RestoreSandR: Restoring the Save and Restore
variables to the original state
          MoL::MoL_FreeScratchSpace: [level] Free storage for scratch
levels
      end while
      GRHydro::sync_GRHydro_C2P_failed: Syncronise the mask function
that contains the points where C2P has failed
      GROUP MoL_PseudoEvolution: Calculate pseudo-evolved quantities
        GROUP ADMBase_SetADMVars: Set the ADM variables before this
group, and use them afterwards
        GROUP ML_BSSN_ConstraintsEverywhere_group:
ML_BSSN_ConstraintsEverywhere
          ML_BSSN::ML_BSSN_ConstraintsEverywhere:
ML_BSSN_ConstraintsEverywhere
          GROUP ML_BSSN_ConstraintsEverywhere_bc_group:
ML_BSSN_ConstraintsEverywhere
            ML_BSSN::ML_BSSN_ConstraintsEverywhere_SelectBCs: [level]
ML_BSSN_ConstraintsEverywhere_SelectBCs
            GROUP ML_BSSN_ConstraintsEverywhere_ApplyBCs: Apply BCs
for groups set in ML_BSSN_ConstraintsEverywhere
              GROUP BoundaryConditions: Execute all boundary
conditions
                Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
              Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
        GROUP ML_BSSN_ConstraintsInterior_group:
ML_BSSN_ConstraintsInterior
          ML_BSSN::ML_BSSN_ConstraintsInterior:
ML_BSSN_ConstraintsInterior
          GROUP ML_BSSN_ConstraintsInterior_bc_group:
ML_BSSN_ConstraintsInterior
            ML_BSSN::ML_BSSN_ConstraintsInterior_SelectBCs: [level]
ML_BSSN_ConstraintsInterior_SelectBCs
            GROUP ML_BSSN_ConstraintsInterior_ApplyBCs: Apply BCs for
groups set in ML_BSSN_ConstraintsInterior
              GROUP BoundaryConditions: Execute all boundary
conditions
                Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
```

```
              Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
    Evolve finer grids recursively
    Restrict from finer grids
    [CCTK_POSTRESTRICT]
      GROUP MoL_PostStep: Ensure that everything is correct after
restriction
        ML_BSSN::ML_BSSN_SelectBoundConds: [level] select boundary
conditions
        GRHydro::GRHydro_RefinementLevel: Calculate current refinement
level
        GRHydro::GRHydro_SetLastMoLPostStep: [level] Set grid scalar
InLastMoLPostStep if this is the last MoL PostStep call
        GROUP ML_BSSN_ApplyBCs: Apply boundary conditions controlled
by thorn Boundary
          GROUP BoundaryConditions: Execute all boundary conditions
            Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
            CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
            ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
          Boundary::Boundary_ClearSelection: [level] Unselect all grid
variables for boundary conditions
        ML_BSSN::ML_BSSN_ADMBaseInterior: ML_BSSN_ADMBaseInterior
        ML_BSSN::ML_BSSN_ADMBaseBoundaryScalar:
ML_BSSN_ADMBaseBoundaryScalar
        ML_BSSN::ML_BSSN_ADMBaseEverywhere: ML_BSSN_ADMBaseEverywhere
        ML_BSSN_Helper::ML_BSSN_ADMBase_SelectBCs: [level] Select
boundary conditions for ADMBase variables
        GROUP ML_BSSN_ADMBase_ApplyBCs: Apply boundary conditions to
ADMBase variables
          GROUP BoundaryConditions: Execute all boundary conditions
            Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
            CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
            ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
          Boundary::Boundary_ClearSelection: [level] Unselect all grid
variables for boundary conditions
        GROUP ADMBase_SetADMVars: Set the ADM variables before this
group, and use them afterwards
        GROUP HydroBase_PostStep: Post step tasks for hydro thorns
          GROUP GRHydro_PostStep: Post step tasks for GRHydro
          GROUP GRHydro_AtmosphereMaskBoundaries: Apply boundary
conditions to primitives
            GRHydro::GRHydro_SelectAtmosphereMaskBoundaries: [level]
Select atmosphere mask for boundary conditions
```

```
            GROUP GRHydro_ApplyAtmosphereMaskBCs: Apply boundary
conditions to real-valued atmosphere mask
              GROUP BoundaryConditions: Execute all boundary
conditions
                Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
              Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
          GRHydro::GRHydroPostSyncAtmosphereMask: Set integer
atmosphere mask from synchronized real atmosphere mask
          if (GRHydro::InLastMoLPostStep)
            GRHydro::GRHydro_AtmosphereReset: Reset the atmosphere
          GROUP HydroBase_Boundaries: HydroBase-internal Boundary
conditions group
            GROUP Do_GRHydro_Boundaries: GRHydro Boundary conditions
group
            GROUP HydroBase_Select_Boundaries: Group to schedule the
boundary condition functions
              if (GRHydro::execute_MoL_PostStep)
                GRHydro::GRHydro_Bound: [level] Select GRHydro
boundary conditions
            GROUP HydroBase_ApplyBCs: Apply the boundary conditions of
HydroBase
              GROUP BoundaryConditions: Execute all boundary
conditions
                Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
              Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
          GROUP HydroBase_Con2Prim: Convert from conservative to
primitive variables
            if (GRHydro::execute_MoL_Step)
              GRHydro::GRHydro_SqrtSpatialDeterminant: Calculate sdetg
            if (GRHydro::execute_MoL_PostStep)
              GRHydro::Con2Prim: Convert back to primitive variables
(polytype)
        GROUP SetTmunu: Group for calculating the stress-energy tensor
          TmunuBase::TmunuBase_ZeroTmunu: Initialise the stress-energy
tensor to zero
          GROUP AddToTmunu: Add to the stress-energy tensor here
            GRHydro::GRHydro_Tmunu: Compute the energy-momentum tensor
```

```
      GROUP MoL_PseudoEvolutionBoundaries: Apply boundary conditions to pseudo-evolved quantities
        GROUP ML_BSSN_ConstraintsEverywhere_bc_group: ML_BSSN_ConstraintsEverywhere
          ML_BSSN::ML_BSSN_ConstraintsEverywhere_SelectBCs: [level] ML_BSSN_ConstraintsEverywhere_SelectBCs
          GROUP ML_BSSN_ConstraintsEverywhere_ApplyBCs: Apply BCs for groups set in ML_BSSN_ConstraintsEverywhere
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested local physical boundary conditions
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary conditions
              ReflectionSymmetry::ReflectionSymmetry_Apply: Apply reflection symmetries
            Boundary::Boundary_ClearSelection: [level] Unselect all grid variables for boundary conditions
        GROUP ML_BSSN_ConstraintsInterior_bc_group: ML_BSSN_ConstraintsInterior
          ML_BSSN::ML_BSSN_ConstraintsInterior_SelectBCs: [level] ML_BSSN_ConstraintsInterior_SelectBCs
          GROUP ML_BSSN_ConstraintsInterior_ApplyBCs: Apply BCs for groups set in ML_BSSN_ConstraintsInterior
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested local physical boundary conditions
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary conditions
              ReflectionSymmetry::ReflectionSymmetry_Apply: Apply reflection symmetries
            Boundary::Boundary_ClearSelection: [level] Unselect all grid variables for boundary conditions
    [CCTK_POSTSTEP]
      SphericalSurface::SphericalSurface_Set: [global] Set surface radii
      GRHydro::GRHydro_RefinementLevel: Calculate current refinement level (for the check of the C2P mask)
      GRHydro::check_GRHydro_C2P_failed: Check the mask function that contains the points where C2P has failed and report an error in case a failure is found
      GROUP HydroBase_ExcisionHasBeenSet: Group to schedule thorns changing the mask before and thorns using the mask after
      GROUP zzz_NaNChecker_NaNCheck: Check for NaNs and count them in NaNChecker::NaNsFound
        NaNChecker::NaNChecker_NaNCheck_Prepare: [level] Prepare data structures to check for NaNs
        NaNChecker::NaNChecker_NaNCheck_Check: [local] Check for NaNs
        NaNChecker::NaNChecker_NaNCheck_Finish: [level] Count NaNs in NaNChecker::NaNsFound
      NaNChecker::NaNChecker_TakeAction: [global] [loop-level] Output
NaNChecker::NaNmask and take action according to
NaNChecker::action_if_found
      SpaceMask::CheckMask: Ensure that all mask values are legal
      GROUP SphericalSurface_HasBeenSet: Set the spherical surfaces
before this group, and use it afterwards
        SphericalSurface::SphericalSurface_CheckState: [global] Test
the state of the spherical surfaces
      Dissipation::setup_epsdis: Setup spatially varying dissipation
    if (checkpoint)
      [CCTK_CHECKPOINT]
      CarpetIOHDF5::CarpetIOHDF5_EvolutionCheckpoint: [meta] Evolution
checkpoint routine
    endif
    if (analysis)
      [CCTK_ANALYSIS]
      CarpetLib::CarpetLib_printtimestats: [global] Print timing
statistics if desired
      CarpetLib::CarpetLib_printmemstats: [global] Print memory
statistics if desired
      LoopControl::LC_statistics_analysis: [meta] Output LoopControl
statistics
      GROUP ML_BSSN_EvolutionAnalysis: Calculate RHS at analysis
        ML_BSSN::ML_BSSN_EvolutionAnalysisInit:
ML_BSSN_EvolutionAnalysisInit
        ML_BSSN::ML_BSSN_EvolutionAnalysisInterior:
ML_BSSN_EvolutionAnalysisInterior
        ML_BSSN_Helper::ML_BSSN_NewRad: Apply NewRad boundary
conditions to RHS
      TerminationTrigger::TerminationTrigger_CheckWalltime: Check
elapsed job walltime
      TerminationTrigger::TerminationTrigger_CheckSignal: Check if we
received a termination signal
      TerminationTrigger::TerminationTrigger_CheckFile: Check
termination file
    endif
    Output grid variables
    enddo

  Termination routines
    [CCTK_TERMINATE]
      CarpetIOHDF5::CarpetIOHDF5_TerminationCheckpoint: [meta]
Termination checkpoint routine
      LoopControl::LC_statistics_terminate: [meta] Output LoopControl
statistics
      MoL::MoL_FreeIndexArrays: Free the MoL bookkeeping index arrays

  Shutdown routines
    [CCTK_SHUTDOWN]
      Timers::Timer_Shutdown: Prepare hierarchical timers
  Routines run after changing the grid hierarchy:
    [CCTK_POSTREGRID]
      CartGrid3D::SpatialCoordinates: Set Coordinates after regridding
      GROUP MaskBase_SetupMask: Set up the weight function
        GROUP MaskBase_SetupMaskAll: Set up the weight function
          CarpetReduce::MaskBase_AllocateMask: [global] Allocate the
weight function
          CarpetReduce::MaskBase_InitMask: [global] [loop-local]
Initialise the weight function
          GROUP SetupIMaskInternal: Set up the integer weight function
(schedule other routines in here)
            CarpetReduce::CoordBase_SetupMask: [global] [loop-local]
Set up the outer boundaries of the weight function
            CarpetReduce::CarpetMaskSetup: [global] [loop-singlemap]
Set up the weight function for the restriction regions
          GROUP SetupIMask: Set up the integer weight function
(schedule other routines in here)
          CarpetReduce::MaskBase_SetMask: [global] [loop-local] Set
the weight function
          GROUP SetupMask: Set up the real weight function (schedule
other routines in here)
          CarpetReduce::MaskBase_TestMask: [global] Test the weight
function
      Dissipation::setup_epsdis: Setup spatially varying dissipation
      SpaceMask::MaskZero: Initialise mask to zero
      GROUP HydroBase_ExcisionMaskSetup: Set up hydro excision mask
        HydroBase::HydroBase_InitExcisionMask: Initialize hydro
excision mask to 'no excision everywhere'
      SpaceMask::MaskOne: Set mask to one
      GRHydro::GRHydro_SetupMask: Initialize the atmosphere mask
      GROUP MoL_PostStep: Ensure that everything is correct after
regridding
        ML_BSSN::ML_BSSN_SelectBoundConds: [level] select boundary
conditions
        GRHydro::GRHydro_RefinementLevel: Calculate current refinement
level
        GRHydro::GRHydro_SetLastMoLPostStep: [level] Set grid scalar
InLastMoLPostStep if this is the last MoL PostStep call
        GROUP ML_BSSN_ApplyBCs: Apply boundary conditions controlled
by thorn Boundary
          GROUP BoundaryConditions: Execute all boundary conditions
            Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
            CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
            ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
          Boundary::Boundary_ClearSelection: [level] Unselect all grid
```

```
variables for boundary conditions
        ML_BSSN::ML_BSSN_ADMBaseInterior: ML_BSSN_ADMBaseInterior
        ML_BSSN::ML_BSSN_ADMBaseBoundaryScalar:
ML_BSSN_ADMBaseBoundaryScalar
        ML_BSSN::ML_BSSN_ADMBaseEverywhere: ML_BSSN_ADMBaseEverywhere
        ML_BSSN_Helper::ML_BSSN_ADMBase_SelectBCs: [level] Select
boundary conditions for ADMBase variables
        GROUP ML_BSSN_ADMBase_ApplyBCs: Apply boundary conditions to
ADMBase variables
          GROUP BoundaryConditions: Execute all boundary conditions
            Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
            CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
            ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
          Boundary::Boundary_ClearSelection: [level] Unselect all grid
variables for boundary conditions
        GROUP ADMBase_SetADMVars: Set the ADM variables before this
group, and use them afterwards
        GROUP HydroBase_PostStep: Post step tasks for hydro thorns
          GROUP GRHydro_PostStep: Post step tasks for GRHydro
          GROUP GRHydro_AtmosphereMaskBoundaries: Apply boundary
conditions to primitives
            GRHydro::GRHydro_SelectAtmosphereMaskBoundaries: [level]
Select atmosphere mask for boundary conditions
            GROUP GRHydro_ApplyAtmosphereMaskBCs: Apply boundary
conditions to real-valued atmosphere mask
              GROUP BoundaryConditions: Execute all boundary
conditions
                Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
                CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
                ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
              Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
          GRHydro::GRHydroPostSyncAtmosphereMask: Set integer
atmosphere mask from synchronized real atmosphere mask
          if (GRHydro::InLastMoLPostStep)
            GRHydro::GRHydro_AtmosphereReset: Reset the atmosphere
          GROUP HydroBase_Boundaries: HydroBase-internal Boundary
conditions group
            GROUP Do_GRHydro_Boundaries: GRHydro Boundary conditions
group
            GROUP HydroBase_Select_Boundaries: Group to schedule the
boundary condition functions
              if (GRHydro::execute_MoL_PostStep)
        GRHydro::GRHydro_Bound: [level] Select GRHydro
boundary conditions
      GROUP HydroBase_ApplyBCs: Apply the boundary conditions of
HydroBase
       GROUP BoundaryConditions: Execute all boundary
conditions
        Boundary::Boundary_ApplyPhysicalBCs: Apply all
requested local physical boundary conditions
        CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry
boundary conditions
        ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
       Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
     GROUP HydroBase_Con2Prim: Convert from conservative to
primitive variables
      if (GRHydro::execute_MoL_Step)
       GRHydro::GRHydro_SqrtSpatialDeterminant: Calculate sdetg
      if (GRHydro::execute_MoL_PostStep)
       GRHydro::Con2Prim: Convert back to primitive variables
(polytype)
    GROUP SetTmunu: Group for calculating the stress-energy tensor
     TmunuBase::TmunuBase_ZeroTmunu: Initialise the stress-energy
tensor to zero
     GROUP AddToTmunu: Add to the stress-energy tensor here
      GRHydro::GRHydro_Tmunu: Compute the energy-momentum tensor
   GROUP MoL_PseudoEvolutionBoundaries: Apply boundary conditions
to pseudo-evolved quantities
    GROUP ML_BSSN_ConstraintsEverywhere_bc_group:
ML_BSSN_ConstraintsEverywhere
     ML_BSSN::ML_BSSN_ConstraintsEverywhere_SelectBCs: [level]
ML_BSSN_ConstraintsEverywhere_SelectBCs
     GROUP ML_BSSN_ConstraintsEverywhere_ApplyBCs: Apply BCs for
groups set in ML_BSSN_ConstraintsEverywhere
      GROUP BoundaryConditions: Execute all boundary conditions
       Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
       CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
       ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
      Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
    GROUP ML_BSSN_ConstraintsInterior_bc_group:
ML_BSSN_ConstraintsInterior
     ML_BSSN::ML_BSSN_ConstraintsInterior_SelectBCs: [level]
ML_BSSN_ConstraintsInterior_SelectBCs
     GROUP ML_BSSN_ConstraintsInterior_ApplyBCs: Apply BCs for
groups set in ML_BSSN_ConstraintsInterior
```

```
            GROUP BoundaryConditions: Execute all boundary conditions
              Boundary::Boundary_ApplyPhysicalBCs: Apply all requested
local physical boundary conditions
              CartGrid3D::CartGrid3D_ApplyBC: Apply symmetry boundary
conditions
              ReflectionSymmetry::ReflectionSymmetry_Apply: Apply
reflection symmetries
            Boundary::Boundary_ClearSelection: [level] Unselect all
grid variables for boundary conditions
----------------------------------------------------------------------
----------
INFO (Carpet): Multi-Model listing:
   model 0: "world"
INFO (Carpet): Multi-Model process distribution:
   processes 0-1: model 0 "world"
INFO (Carpet): Multi-Model: This is process 0, model 0 "world"
Current core file size limit: hard=[unlimited], soft=[unlimited]
Current addres space size limit: hard=[unlimited], soft=[unlimited]
Current data segment size limit: hard=[unlimited], soft=[unlimited]
Current resident set size limit: hard=[unlimited], soft=[unlimited]
INFO (CycleClock): Measuring CycleClock tick via OpenMP...

INFO (CycleClock): Calibrated CycleClock: 0.501984 ns per clock tick
(1.99209 GHz)
INFO (Vectors): Using vector size 1 for architecture scalar (no
vectorisation, 64-bit precision)
INFO (hwloc): library version 2.5.0, API version 0x20500
----------------------------------------------------------------------
----------
AMR driver provided by Carpet
----------------------------------------------------------------------
----------
HydroBase: Let it flow.
----------------------------------------------------------------------
----------
AMR 0D ASCII I/O provided by CarpetIOASCII
----------------------------------------------------------------------
----------
AMR 1D ASCII I/O provided by CarpetIOASCII
----------------------------------------------------------------------
----------
AMR 2D ASCII I/O provided by CarpetIOASCII
----------------------------------------------------------------------
----------
AMR 3D ASCII I/O provided by CarpetIOASCII
----------------------------------------------------------------------
----------
AMR info I/O provided by CarpetIOBasic
----------------------------------------------------------------------
----------
```

```
ML_BSSN
----------------------------------------------------------------------
----------
AMR HDF5 I/O provided by CarpetIOHDF5
----------------------------------------------------------------------
----------
AMR 0D HDF5 I/O provided by CarpetIOHDF5
----------------------------------------------------------------------
----------
AMR 1D HDF5 I/O provided by CarpetIOHDF5
----------------------------------------------------------------------
----------
AMR 2D HDF5 I/O provided by CarpetIOHDF5
----------------------------------------------------------------------
----------
AMR 3D HDF5 I/O provided by CarpetIOHDF5
----------------------------------------------------------------------
----------
MoL: Generalized time integration.
----------------------------------------------------------------------
----------
AMR scalar I/O provided by CarpetIOScalar
----------------------------------------------------------------------
----------

INFO (Carpet): MPI is enabled
INFO (Carpet): Carpet is running on 2 processes
INFO (Carpet): This is process 0
INFO (Carpet): OpenMP is enabled
INFO (Carpet): This process contains 2 threads, this is thread 0
INFO (Carpet): There are 4 threads in total
INFO (Carpet): There are 2 threads per process
INFO (Carpet): This process runs on host relayer, pid=17680
INFO (Carpet): This process runs on 2 cores: 0, 4
INFO (Carpet): Thread 0 runs on 2 cores: 0, 4
INFO (Carpet): Thread 1 runs on 2 cores: 0, 4
INFO (Carpet): This simulation is running in 3 dimensions
INFO (Carpet): Boundary specification for map 0:
   nboundaryzones: [[3,3,3],[3,3,3]]
   is_internal   : [[0,0,0],[0,0,0]]
   is_staggered  : [[0,0,0],[0,0,0]]
   shiftout      : [[1,1,1],[0,0,0]]
INFO (Carpet): CoordBase domain specification for map 0:
   physical extent: [0,0,0] : [24,24,24]   ([24,24,24])
   interior extent: [0,0,0] : [22,22,22]   ([22,22,22])
   exterior extent: [-6,-6,-6] : [28,28,28]   ([34,34,34])
   base_spacing   : [2,2,2]
INFO (Carpet): Adapted domain specification for map 0:
   convergence factor: 2
   convergence level : 0
   physical extent   : [0,0,0] : [24,24,24]   ([24,24,24])
   interior extent   : [0,0,0] : [22,22,22]   ([22,22,22])
   exterior extent   : [-6,-6,-6] : [28,28,28]   ([34,34,34])
   spacing           : [2,2,2]
INFO (Carpet): Base grid specification for map 0:
   number of grid points             : [18,18,18]
   number of coarse grid ghost points: [[3,3,3],[3,3,3]]
INFO (Carpet): Buffer zone counts (excluding ghosts):
   [0]: [[0,0,0],[0,0,0]]
   [1]: [[9,9,9],[9,9,9]]
   [2]: [[9,9,9],[9,9,9]]
   [3]: [[9,9,9],[9,9,9]]
   [4]: [[9,9,9],[9,9,9]]
   [5]: [[9,9,9],[9,9,9]]
   [6]: [[9,9,9],[9,9,9]]
   [7]: [[9,9,9],[9,9,9]]
   [8]: [[9,9,9],[9,9,9]]
   [9]: [[9,9,9],[9,9,9]]
INFO (Carpet): Overlap zone counts:
   [0]: [[0,0,0],[0,0,0]]
   [1]: [[0,0,0],[0,0,0]]
   [2]: [[0,0,0],[0,0,0]]
   [3]: [[0,0,0],[0,0,0]]
   [4]: [[0,0,0],[0,0,0]]
   [5]: [[0,0,0],[0,0,0]]
   [6]: [[0,0,0],[0,0,0]]
   [7]: [[0,0,0],[0,0,0]]
   [8]: [[0,0,0],[0,0,0]]
   [9]: [[0,0,0],[0,0,0]]
INFO (Carpet): Group and variable statistics:
INFO (Carpet):    There are 1066 grid functions in 133 groups
INFO (Carpet):    There are 230 grid scalars in 71 groups
INFO (Carpet):    There are 100 1-dimensional grid arrays in 10 groups
INFO (Carpet):    There are 1 2-dimensional grid arrays in 2 groups
INFO (Carpet):    There are 0 3-dimensional grid arrays in 0 groups
INFO (Carpet):    (The number of variables counts all time levels)
INFO (CarpetIOASCII): I/O Method 'IOASCII_0D' registered: 0D AMR
output of grid variables to ASCII files
INFO (CarpetIOASCII): I/O Method 'IOASCII_1D' registered: 1D AMR
output of grid variables to ASCII files
INFO (CarpetIOASCII): Periodic 1D AMR output requested for:
   ADMBASE::gxx
   ADMBASE::gxy
   ADMBASE::gxz
   ADMBASE::gyy
   ADMBASE::gyz
   ADMBASE::gzz
   ADMBASE::kxx
   ADMBASE::kxy
   ADMBASE::kxz
   ADMBASE::kyy
   ADMBASE::kyz
   ADMBASE::kzz
   ADMBASE::alp
   HYDROBASE::rho
   HYDROBASE::press
   HYDROBASE::eps
   HYDROBASE::vel[0]
   HYDROBASE::vel[1]
   HYDROBASE::vel[2]
   ML_BSSN::H
   ML_BSSN::M1
   ML_BSSN::M2
   ML_BSSN::M3
INFO (CarpetIOASCII): I/O Method 'IOASCII_2D' registered: 2D AMR
output of grid variables to ASCII files
INFO (CarpetIOASCII): I/O Method 'IOASCII_3D' registered: 3D AMR
output of grid variables to ASCII files
INFO (CarpetIOHDF5): I/O Method 'IOHDF5' registered: AMR output of
grid variables to HDF5 files
INFO (CarpetIOHDF5): I/O Method 'IOHDF5_0D' registered: 0D AMR output
of grid variables to HDF5 files
INFO (CarpetIOHDF5): I/O Method 'IOHDF5_1D' registered: 1D AMR output
of grid variables to HDF5 files
INFO (CarpetIOHDF5): I/O Method 'IOHDF5_2D' registered: 2D AMR output
of grid variables to HDF5 files
INFO (CarpetIOHDF5): Periodic 2D AMR output requested for:
   ADMBASE::gxx
   ADMBASE::gxy
   ADMBASE::gxz
   ADMBASE::gyy
   ADMBASE::gyz
   ADMBASE::gzz
   ADMBASE::alp
   ADMBASE::betax
   ADMBASE::betay
   ADMBASE::betaz
   HYDROBASE::rho
   HYDROBASE::eps
   HYDROBASE::vel[0]
   HYDROBASE::vel[1]
   HYDROBASE::vel[2]
   HYDROBASE::w_lorentz
   ML_BSSN::H
   ML_BSSN::M1
   ML_BSSN::M2
   ML_BSSN::M3
```

```
INFO (CarpetIOHDF5): I/O Method 'IOHDF5_3D' registered: 3D AMR output
of grid variables to HDF5 files
INFO (CarpetIOScalar): Periodic scalar output requested for:
   ADMBASE::alp
   ADMBASE::betax
   ADMBASE::betay
   ADMBASE::betaz
   CARPET::physical_time_per_hour
   CARPET::current_physical_time_per_hour
   CARPET::time_total
   CARPET::time_evolution
   CARPET::time_computing
   CARPET::time_communicating
   CARPET::time_io
   CARPET::evolution_steps_count
   CARPET::local_grid_points_per_second
   CARPET::total_grid_points_per_second
   CARPET::local_grid_point_updates_count
   CARPET::total_grid_point_updates_count
   CARPET::local_interior_points_per_second
   CARPET::total_interior_points_per_second
   CARPET::local_interior_point_updates_count
   CARPET::total_interior_point_updates_count
   CARPET::io_per_second
   CARPET::io_bytes_per_second
   CARPET::io_bytes_ascii_per_second
   CARPET::io_bytes_binary_per_second
   CARPET::io_count
   CARPET::io_bytes_count
   CARPET::io_bytes_ascii_count
   CARPET::io_bytes_binary_count
   CARPET::comm_per_second
   CARPET::comm_bytes_per_second
   CARPET::comm_count
   CARPET::comm_bytes_count
   CARPET::time_levels
   CARPET::current_walltime
   CARPET::syncs_count
   GRHYDRO::dens
   HYDROBASE::rho
   HYDROBASE::press
   HYDROBASE::eps
   HYDROBASE::vel[0]
   HYDROBASE::vel[1]
   HYDROBASE::vel[2]
   HYDROBASE::w_lorentz
   ML_BSSN::H
   ML_BSSN::M1
   ML_BSSN::M2
   ML_BSSN::M3
WARNING[L1,P1] (ML_BSSN_Helper): Forcing
ML_BSSN::initial_boundary_condition="extrapolate-gammas" because
ML_BSSN::my_initial_boundary_condition="extrapolate-gammas"
WARNING[L1,P1] (ML_BSSN_Helper): Forcing
ML_BSSN::rhs_boundary_condition="NewRad" because
ML_BSSN::my_rhs_boundary_condition="NewRad"
WARNING[L1,P1] (ML_BSSN_Helper): Forcing ML_BSSN::evolveB=0 because
ML_BSSN::shiftGammaCoeff=0.0
WARNING[L1,P0] (ML_BSSN_Helper): Forcing
ML_BSSN::initial_boundary_condition="extrapolate-gammas" because
ML_BSSN::my_initial_boundary_condition="extrapolate-gammas"
WARNING[L1,P0] (ML_BSSN_Helper): Forcing
ML_BSSN::rhs_boundary_condition="NewRad" because
ML_BSSN::my_rhs_boundary_condition="NewRad"
WARNING[L1,P0] (ML_BSSN_Helper): Forcing ML_BSSN::evolveB=0 because
ML_BSSN::shiftGammaCoeff=0.0
INFO (MoL): Using Runge-Kutta 4 as the time integrator.
INFO (SymBase): Symmetry on lower x-face: reflection_symmetry
INFO (SymBase): Symmetry on lower y-face: reflection_symmetry
INFO (SymBase): Symmetry on lower z-face: reflection_symmetry
WARNING[L1,P1] (ML_BSSN_Helper): Parameter
ML_BSSN::my_initial_boundary_condition is outdated; please update the
parameter file. Do not use this parameter, and set up initial boundary
conditions as usual.
WARNING[L1,P1] (ML_BSSN_Helper): Parameter
ML_BSSN::my_rhs_boundary_condition is outdated; please update the
parameter file. Do not use this parameter, and set up RHS boundary
conditions as usual.
INFO (MoL): The maximum number of evolved variables is 664. 29 are
registered.
INFO (MoL): The maximum number of slow evolved variables is 664. 0 are
registered.
INFO (MoL): The maximum number of constrained variables is 664. 38 are
registered.
INFO (MoL): The maximum number of SandR variables is 664. 0 are
registered.
INFO (MoL): The maximum number of evolved array variables is 664. 0
are registered.
INFO (MoL): The maximum number of constrained array variables is 664.
0 are registered.
INFO (MoL): The maximum number of SandR array variables is 664. 0 are
registered.
INFO (MoL): The maximum size of any array variables is 0.
WARNING[L1,P0] (ML_BSSN_Helper): Parameter
ML_BSSN::my_initial_boundary_condition is outdated; please update the
parameter file. Do not use this parameter, and set up initial boundary
conditions as usual.
WARNING[L1,P0] (ML_BSSN_Helper): Parameter
```

```
ML_BSSN::my_rhs_boundary_condition is outdated; please update the
parameter file. Do not use this parameter, and set up RHS boundary
conditions as usual.
INFO (Vectors): Testing vectorisation... [errors may result in
segfaults]
INFO (Vectors): 101/101 tests passed
INFO (CarpetRegrid2): Enforcing grid structure properties, iteration 0
INFO (CarpetRegrid2): Enforcing grid structure properties, iteration 1

INFO (Carpet): Grid structure (superregions, grid points):
   [0][0][0]   exterior: [0,0,0] : [17,17,17]   ([18,18,18] + PADDING)
5832
   [1][0][0]   exterior: [3,3,3] : [32,32,32]   ([30,30,30] + PADDING)
27000
INFO (Carpet): Grid structure (superregions, coordinates):
   [0][0][0]   exterior: [-6,-6,-6] : [28,28,28] : [2,2,2]
   [1][0][0]   exterior: [-3,-3,-3] : [26,26,26] : [1,1,1]
INFO (Carpet): Global grid structure statistics:
INFO (Carpet): GF: rhs: 10k active, 10k owned (+0%), 24k total
(+147%), 3 steps/time
INFO (Carpet): GF: vars: 252, pts: 4M active, 4M owned (+0%), 10M
total (+158%), 1.0 comp/proc
INFO (Carpet): GA: vars: 289, pts: 0M active, 0M total (+0%)
INFO (Carpet): Total required memory: 0.080 GByte (for GAs and
currently active GFs)
INFO (Carpet): Load balance:  min     avg     max     sdv     max/avg-
1
INFO (Carpet): Level  0:       0M      0M      0M      0M owned
0%
INFO (Carpet): Level  1:       2M      2M      2M      0M owned
0%
INFO (CartGrid3D): Grid Spacings:
INFO (CartGrid3D): dx=>2.0000000e+00  dy=>2.0000000e+00
dz=>2.0000000e+00
INFO (CartGrid3D): Computational Coordinates:
INFO (CartGrid3D): x=>[-6.000,28.000]  y=>[-6.000,28.000]  z=>[-
6.000,28.000]
INFO (CartGrid3D): Indices of Physical Coordinates:
INFO (CartGrid3D): x=>[0,17]  y=>[0,17]  z=>[0,17]
INFO (TerminationTrigger): Reminding you every 60 minutes about
remaining walltime
INFO (Time): Timestep set to 0.5 (courant_static)
INFO (GRHydro): Trying to get EOS handles
INFO (GRHydro): Trying to get EOS handles
INFO (GRHydro): GRHydro will use the 2D_Polytrope equation of state.
INFO (GRHydro): Setting up the atmosphere mask: all points are
not_atmosphere
INFO (TOVSolver): Integrated TOV equation
INFO (TOVSolver): Information about the TOVs used:
INFO (): TOV    radius    mass  bary_mass mass(g) cent.rho rho(cgi)
```

```
K   K(cgi)    Gamma
INFO ():   1   8.12502  1.40016  1.50618 2.78e+33  0.00128 7.92e+14
100 1.45e+05       2
INFO (TOVSolver): Not using old matter initial data
INFO (TOVSolver): Done interpolation.
INFO (TerminationTrigger): Reminding you every 60 minutes about
remaining walltime
INFO (Time): Timestep set to 0.25 (courant_static)
INFO (GRHydro): Setting up the atmosphere mask: all points are
not_atmosphere
INFO (TOVSolver): Not using old matter initial data
INFO (TOVSolver): Done interpolation.
-------------------------------------------------
Iteration      Time | *me_per_hour | *ROBASE::rho
                    |              |      maximum
-------------------------------------------------
        0     0.000 |    0.0000000 |    0.0012800
      512     0.500 | 3.245418e+03 |    0.0012770
     1024     1.000 | 3.152295e+03 |    0.0012742
     1536     1.500 | 3.094625e+03 |    0.0012715
     2048     2.000 | 2.954479e+03 |    0.0012690
     2560     2.500 | 2.875534e+03 |    0.0012666
     3072     3.000 | 2.919954e+03 |    0.0012645
     3584     3.500 | 2.921631e+03 |    0.0012627
     4096     4.000 | 2.836578e+03 |    0.0012613
     4608     4.500 | 2.799077e+03 |    0.0012602
     5120     5.000 | 2.812504e+03 |    0.0012594
     5632     5.500 | 2.820583e+03 |    0.0012590
     6144     6.000 | 2.801438e+03 |    0.0012589
     6656     6.500 | 2.752911e+03 |    0.0012590
     7168     7.000 | 2.763712e+03 |    0.0012595
     7680     7.500 | 2.768784e+03 |    0.0012602
     8192     8.000 | 2.766293e+03 |    0.0012610
     8704     8.500 | 2.751914e+03 |    0.0012620
     9216     9.000 | 2.760190e+03 |    0.0012632
     9728     9.500 | 2.766596e+03 |    0.0012644
-------------------------------------------------
Iteration      Time | *me_per_hour | *ROBASE::rho
                    |              |      maximum
-------------------------------------------------
    10240    10.000 | 2.755021e+03 |    0.0012656
    10752    10.500 | 2.671216e+03 |    0.0012668
    11264    11.000 | 2.593298e+03 |    0.0012680
    11776    11.500 | 2.592937e+03 |    0.0012692
    12288    12.000 | 2.537341e+03 |    0.0012703
    12800    12.500 | 2.537811e+03 |    0.0012713
    13312    13.000 | 2.549237e+03 |    0.0012722
    13824    13.500 | 2.508780e+03 |    0.0012731
    14336    14.000 | 2.443597e+03 |    0.0012738
    14848    14.500 | 2.396693e+03 |    0.0012744
    15360    15.000 | 2.390157e+03 |    0.0012749
    15872    15.500 | 2.398947e+03 |    0.0012753
    16384    16.000 | 2.403105e+03 |    0.0012756
    16896    16.500 | 2.407073e+03 |    0.0012758
    17408    17.000 | 2.420602e+03 |    0.0012759
    17920    17.500 | 2.434481e+03 |    0.0012759
    18432    18.000 | 2.436937e+03 |    0.0012759
    18944    18.500 | 2.438352e+03 |    0.0012758
    19456    19.000 | 2.447061e+03 |    0.0012757
    19968    19.500 | 2.456141e+03 |    0.0012755
-------------------------------------------------
Iteration      Time | *me_per_hour | *ROBASE::rho
                    |              |      maximum
-------------------------------------------------
    20480    20.000 | 2.457703e+03 |    0.0012753
    20992    20.500 | 2.458487e+03 |    0.0012750
    21504    21.000 | 2.466478e+03 |    0.0012748
    22016    21.500 | 2.474543e+03 |    0.0012745
    22528    22.000 | 2.475739e+03 |    0.0012742
    23040    22.500 | 2.476337e+03 |    0.0012739
    23552    23.000 | 2.478415e+03 |    0.0012735
    24064    23.500 | 2.486718e+03 |    0.0012732
    24576    24.000 | 2.487413e+03 |    0.0012728
    25088    24.500 | 2.486976e+03 |    0.0012724
    25600    25.000 | 2.492932e+03 |    0.0012719
    26112    25.500 | 2.499049e+03 |    0.0012715
    26624    26.000 | 2.499248e+03 |    0.0012709
    27136    26.500 | 2.494796e+03 |    0.0012704
    27648    27.000 | 2.498173e+03 |    0.0012698
    28160    27.500 | 2.502967e+03 |    0.0012691
    28672    28.000 | 2.503284e+03 |    0.0012685
    29184    28.500 | 2.503400e+03 |    0.0012677
    29696    29.000 | 2.510523e+03 |    0.0012670
    30208    29.500 | 2.515153e+03 |    0.0012662
-------------------------------------------------
Iteration      Time | *me_per_hour | *ROBASE::rho
                    |              |      maximum
-------------------------------------------------
    30720    30.000 | 2.516472e+03 |    0.0012653
    31232    30.500 | 2.513356e+03 |    0.0012643
    31744    31.000 | 2.519716e+03 |    0.0012634
    32256    31.500 | 2.523896e+03 |    0.0012624
    32768    32.000 | 2.511796e+03 |    0.0012613
    33280    32.500 | 2.512942e+03 |    0.0012603
    33792    33.000 | 2.516910e+03 |    0.0012592
    34304    33.500 | 2.513745e+03 |    0.0012580
    34816    34.000 | 2.513095e+03 |    0.0012569
    35328    34.500 | 2.512740e+03 |    0.0012557
    35840    35.000 | 2.517035e+03 |    0.0012546
    36352    35.500 | 2.520782e+03 |    0.0012534
    36864    36.000 | 2.520578e+03 |    0.0012523
    37376    36.500 | 2.519634e+03 |    0.0012511
    37888    37.000 | 2.523053e+03 |    0.0012500
    38400    37.500 | 2.522687e+03 |    0.0012488
    38912    38.000 | 2.522146e+03 |    0.0012478
    39424    38.500 | 2.521181e+03 |    0.0012467
    39936    39.000 | 2.524628e+03 |    0.0012457
    40448    39.500 | 2.524983e+03 |    0.0012447
------------------------------------------------
Iteration      Time | *me_per_hour | *ROBASE::rho
                    |              |      maximum
------------------------------------------------
    40960    40.000 | 2.524441e+03 |    0.0012438
    41472    40.500 | 2.523601e+03 |    0.0012429
    41984    41.000 | 2.526843e+03 |    0.0012420
    42496    41.500 | 2.530090e+03 |    0.0012413
    43008    42.000 | 2.530580e+03 |    0.0012405
    43520    42.500 | 2.530110e+03 |    0.0012398
    44032    43.000 | 2.533281e+03 |    0.0012391
    44544    43.500 | 2.533445e+03 |    0.0012385
    45056    44.000 | 2.532588e+03 |    0.0012380
    45568    44.500 | 2.533004e+03 |    0.0012374
    46080    45.000 | 2.535408e+03 |    0.0012370
    46592    45.500 | 2.538170e+03 |    0.0012366
    47104    46.000 | 2.537317e+03 |    0.0012362
    47616    46.500 | 2.536005e+03 |    0.0012359
    48128    47.000 | 2.538833e+03 |    0.0012356
    48640    47.500 | 2.541343e+03 |    0.0012354
    49152    48.000 | 2.540494e+03 |    0.0012352

    49664    48.500 | 2.538036e+03 |    0.0012351
    50176    49.000 | 2.541910e+03 |    0.0012350
    50688    49.500 | 2.535229e+03 |    0.0012349
------------------------------------------------
Iteration      Time | *me_per_hour | *ROBASE::rho
                    |              |      maximum
------------------------------------------------
    51200    50.000 | 2.513844e+03 |    0.0012349
    51712    50.500 | 2.489527e+03 |    0.0012349
    52224    51.000 | 2.475593e+03 |    0.0012349
    52736    51.500 | 2.447746e+03 |    0.0012350
    53248    52.000 | 2.423180e+03 |    0.0012351
    53760    52.500 | 2.404377e+03 |    0.0012352
    54272    53.000 | 2.390085e+03 |    0.0012353
    54784    53.500 | 2.380294e+03 |    0.0012355
    55296    54.000 | 2.372424e+03 |    0.0012357
    55808    54.500 | 2.351751e+03 |    0.0012359
    56320    55.000 | 2.325405e+03 |    0.0012361
    56832    55.500 | 2.314331e+03 |    0.0012364
    57344    56.000 | 2.299807e+03 |    0.0012366
    57856    56.500 | 2.287892e+03 |    0.0012369
    58368    57.000 | 2.287512e+03 |    0.0012372
    58880    57.500 | 2.290579e+03 |    0.0012375
    59392    58.000 | 2.291587e+03 |    0.0012378
    59904    58.500 | 2.292564e+03 |    0.0012381
    60416    59.000 | 2.296295e+03 |    0.0012385
    60928    59.500 | 2.298538e+03 |    0.0012388
-------------------------------------------------
Iteration      Time | *me_per_hour | *ROBASE::rho
                    |              |      maximum
-------------------------------------------------
    61440    60.000 | 2.299955e+03 |    0.0012391
    61952    60.500 | 2.301327e+03 |    0.0012395
    62464    61.000 | 2.304055e+03 |    0.0012398
    62976    61.500 | 2.306987e+03 |    0.0012402
    63488    62.000 | 2.307755e+03 |    0.0012405
    64000    62.500 | 2.309424e+03 |    0.0012409
    64512    63.000 | 2.312258e+03 |    0.0012413
    65024    63.500 | 2.314901e+03 |    0.0012416
    65536    64.000 | 2.315428e+03 |    0.0012420
    66048    64.500 | 2.316508e+03 |    0.0012424
    66560    65.000 | 2.319884e+03 |    0.0012427
    67072    65.500 | 2.320721e+03 |    0.0012431
    67584    66.000 | 2.321220e+03 |    0.0012434
    68096    66.500 | 2.321914e+03 |    0.0012438
    68608    67.000 | 2.324455e+03 |    0.0012441
    69120    67.500 | 2.327771e+03 |    0.0012445
    69632    68.000 | 2.328113e+03 |    0.0012448
    70144    68.500 | 2.328414e+03 |    0.0012452
    70656    69.000 | 2.329179e+03 |    0.0012455
    71168    69.500 | 2.332227e+03 |    0.0012458
-------------------------------------------------
Iteration      Time | *me_per_hour | *ROBASE::rho
                    |              |      maximum
-------------------------------------------------
    71680    70.000 | 2.333359e+03 |    0.0012461
    72192    70.500 | 2.333550e+03 |    0.0012464
    72704    71.000 | 2.336329e+03 |    0.0012467
    73216    71.500 | 2.338478e+03 |    0.0012470
    73728    72.000 | 2.339340e+03 |    0.0012473
    74240    72.500 | 2.339413e+03 |    0.0012476
    74752    73.000 | 2.341746e+03 |    0.0012478
    75264    73.500 | 2.343706e+03 |    0.0012481
    75776    74.000 | 2.344153e+03 |    0.0012483
    76288    74.500 | 2.344470e+03 |    0.0012486
    76800    75.000 | 2.346570e+03 |    0.0012488
    77312    75.500 | 2.348636e+03 |    0.0012490
    77824    76.000 | 2.348413e+03 |    0.0012493
    78336    76.500 | 2.348090e+03 |    0.0012495
    78848    77.000 | 2.350201e+03 |    0.0012497
    79360    77.500 | 2.352352e+03 |    0.0012499
    79872    78.000 | 2.352658e+03 |    0.0012501
    80384    78.500 | 2.352716e+03 |    0.0012503
    80896    79.000 | 2.355040e+03 |    0.0012505
    81408    79.500 | 2.356425e+03 |    0.0012507
-----------------------------------------------
Iteration      Time | *me_per_hour | *ROBASE::rho
                    |              |      maximum
-----------------------------------------------
    81920    80.000 | 2.356485e+03 |    0.0012509
INFO (Carpet): Terminating due to cctk_final_time at t = 80.000000
----------------------------------------------------------------------
----------
Done.
```

Plotting the output

This time let us use kuibit to analyse and plot the data.

```
import matplotlib.pyplot as plt
import numpy as np
from kuibit.simdir import SimDir
from kuibit.grid_data import UniformGrid

%matplotlib inline
```

let's start by looking at the available data

```
sim = SimDir("./tov")
gf  = sim.gf

print(gf)
```

```

Available grid data of dimension 1D (x):
['rho', 'H', 'gxx', 'gxy', 'gxz', 'gyy', 'gyz', 'gzz', 'vel[0]',
'vel[1]', 'vel[2]', 'kxx', 'kxy', 'kxz', 'kyy', 'kyz', 'kzz', 'alp',
'M1', 'M2', 'M3', 'eps', 'press']


Available grid data of dimension 1D (y):
['gxx', 'gxy', 'gxz', 'gyy', 'gyz', 'gzz', 'H', 'press', 'kxx', 'kxy',
'kxz', 'kyy', 'kyz', 'kzz', 'M1', 'M2', 'M3', 'vel[0]', 'vel[1]',
'vel[2]', 'rho', 'alp', 'eps']
```

```
Available grid data of dimension 1D (z):
['H', 'rho', 'kxx', 'kxy', 'kxz', 'kyy', 'kyz', 'kzz', 'M1', 'M2',
'M3', 'eps', 'press', 'vel[0]', 'vel[1]', 'vel[2]', 'gxx', 'gxy',
'gxz', 'gyy', 'gyz', 'gzz', 'alp']


Available grid data of dimension 2D (xy):
['M1', 'betaz', 'alp', 'w_lorentz', 'gyz', 'H', 'gyy', 'eps', 'betay',
'M3', 'vel[1]', 'vel[2]', 'M2', 'gzz', 'rho', 'gxz', 'gxx', 'betax',
'vel[0]', 'gxy']


Available grid data of dimension 2D (xz):
['eps', 'gxy', 'rho', 'betay', 'betaz', 'w_lorentz', 'gxx', 'vel[2]',
'gyy', 'gyz', 'vel[0]', 'M1', 'M2', 'alp', 'M3', 'vel[1]', 'H', 'gxz',
'gzz', 'betax']


Available grid data of dimension 2D (yz):
['gyz', 'rho', 'M3', 'H', 'vel[2]', 'vel[0]', 'M1', 'gzz', 'betaz',
'gxy', 'M2', 'eps', 'gxx', 'gxz', 'betay', 'alp', 'gyy', 'w_lorentz',
'betax', 'vel[1]']


Available grid data of dimension 3D (xyz):
[]
```

and let's focus on those available in the $z=0$ plane

```
vars2D = gf.xy
print(vars2D)


Available grid data of dimension 2D (xy):
['M1', 'betaz', 'alp', 'w_lorentz', 'gyz', 'H', 'gyy', 'eps', 'betay',
'M3', 'vel[1]', 'vel[2]', 'M2', 'gzz', 'rho', 'gxz', 'gxx', 'betax',
'vel[0]', 'gxy']
```

let us work with the density `rho`

```
rho = vars2D.fields.rho

print(rho)

<kuibit.cactus_grid_functions.OneGridFunctionH5 object at
0x7f18b37e6c80>
```

```
# iteration 0:
rho0 = rho[0]

# print available iterations
print(rho.iterations)
```

```
[0, 2048, 4096, 6144, 8192, 10240, 12288, 14336, 16384, 18432, 20480,
22528, 24576, 26624, 28672, 30720, 32768, 34816, 36864, 38912, 40960,
43008, 45056, 47104, 49152, 51200, 53248, 55296, 57344, 59392, 61440,
63488, 65536, 67584, 69632, 71680, 73728, 75776, 77824, 79872, 81920]
```

or the available times

```
print(rho.available_times)
```

```
[0.0, 2.0, 4.0, 6.0, 8.0, 10.0, 12.0, 14.0, 16.0, 18.0, 20.0, 22.0,
24.0, 26.0, 28.0, 30.0, 32.0, 34.0, 36.0, 38.0, 40.0, 42.0, 44.0,
46.0, 48.0, 50.0, 52.0, 54.0, 56.0, 58.0, 60.0, 62.0, 64.0, 66.0,
68.0, 70.0, 72.0, 74.0, 76.0, 78.0, 80.0]
```

let's see what information the object `rho0` holds

```
print(rho0)
```

```
Available refinement levels (components):
0 (1)
1 (1)
Spacing at coarsest level (0): [2. 2.]
Spacing at finest level (1): [1. 1.]
```

```
type(rho0)
```

```
kuibit.grid_data.HierarchicalGridData
```

we see that it's of the type `HierarchicalGridData`, storing the data in all available Carpet refinement levels. For now let us focus on the data in the inner level:

```
rho0_1, = rho0[1]; print(rho0_1)
```

```
<kuibit.grid_data.UniformGridData object at 0x7f18b37e6d40>
```

we can access the data at specific point

```
rho0_1[1,1]
```

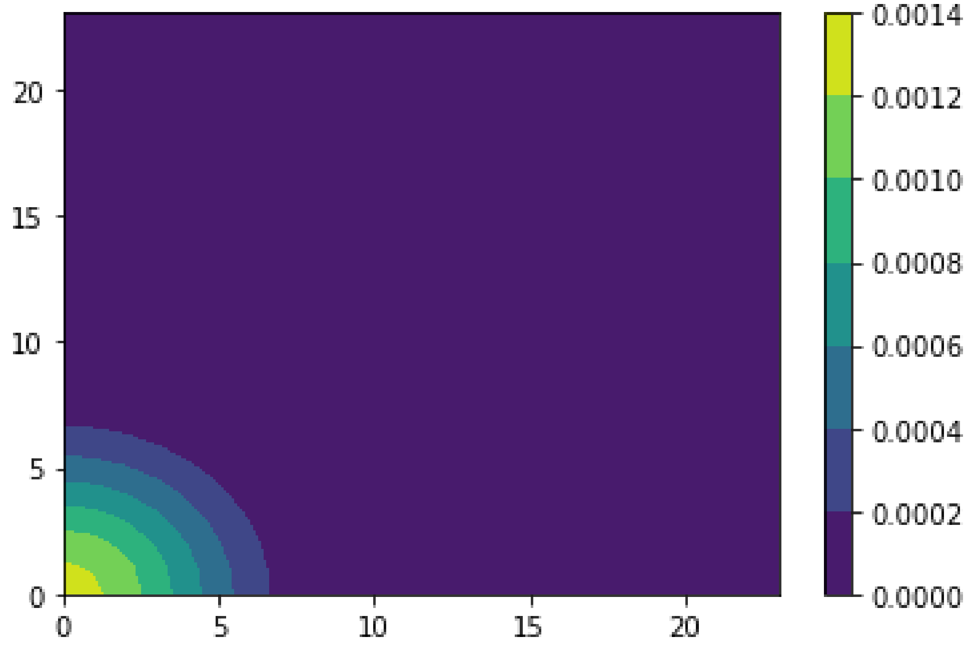
```
0.0011813782894413533
```

and plot the data

```
cf = plt.contourf(*rho0_1.coordinates_meshgrid(), rho0_1.data)

plt.colorbar(cf)

<matplotlib.colorbar.Colorbar at 0x7f18b2b5eef0>
```

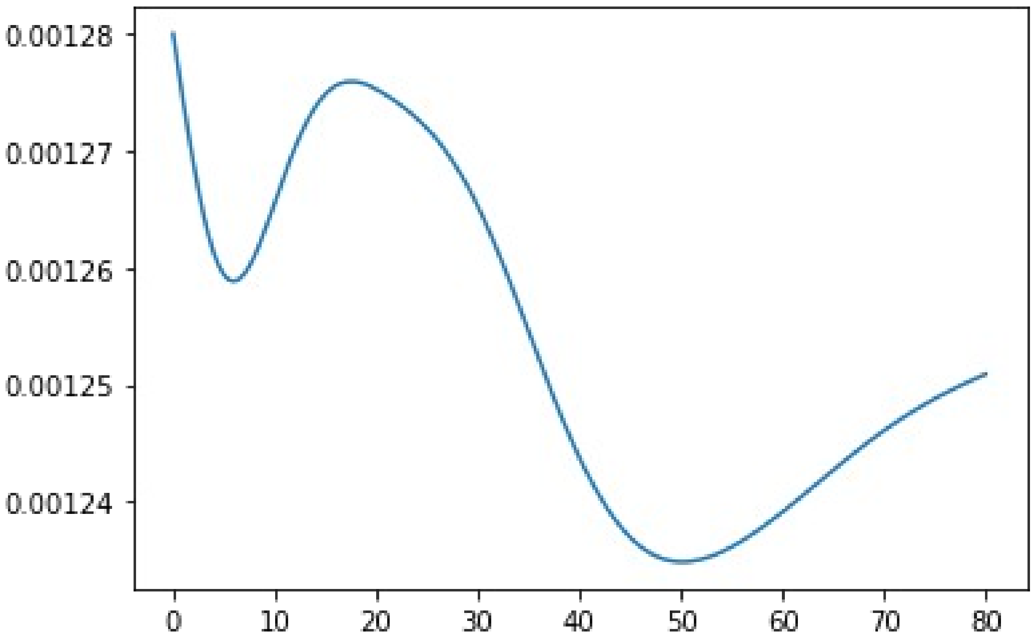


we can also analyse scalar reductions using kuibit's TimeSeries

```
timeseries = sim.ts
rho_max = timeseries.maximum['rho']

plt.plot(rho_max.t, rho_max.values)

[<matplotlib.lines.Line2D at 0x7f18b23521a0>]
```


0.00128
0.00127
0.00126
0.00125
0.00124
0
10
20
30
40
50
60
70
80