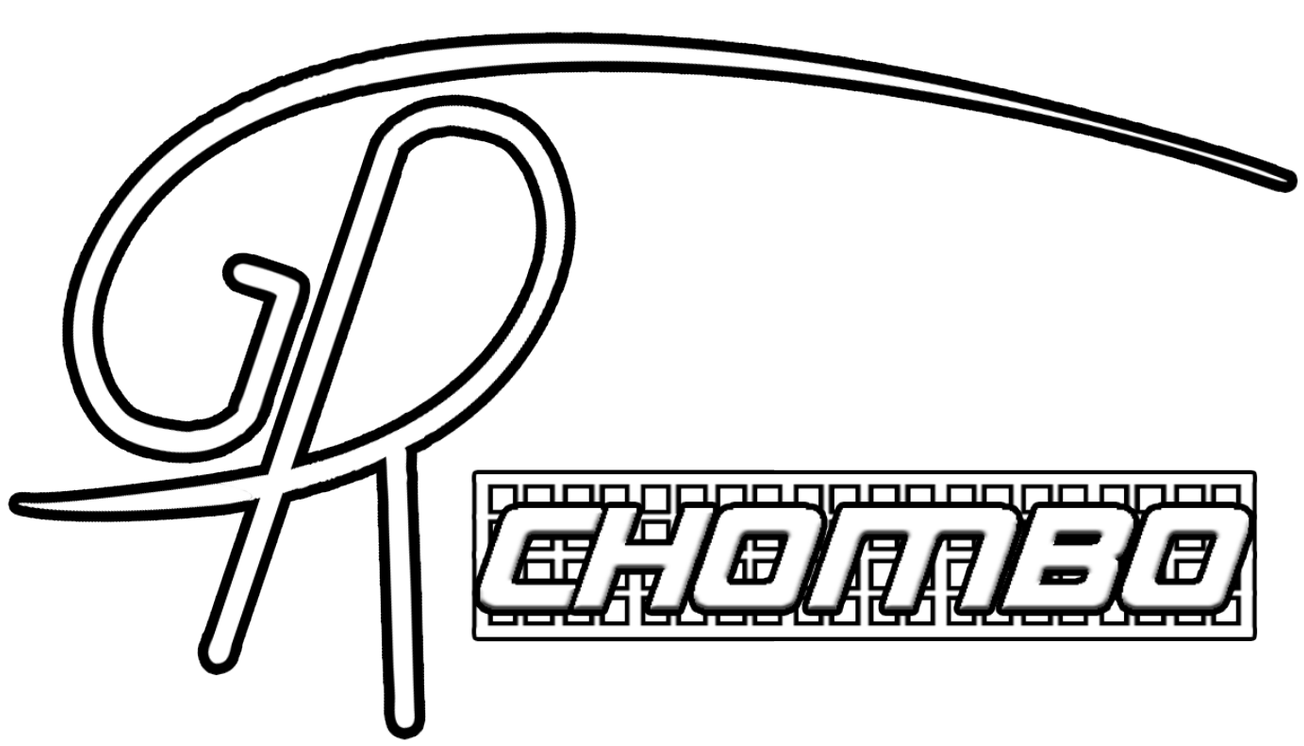
Parameters Guide v2.0

Amelia Drew



Includes:

- Parameters in BBH example params file
- Outline parameters in ChomboParameters
- How to add new parameters

Does not include:

- Details about GR parameters

BBH Params Example

```
# Filesystem parameters

verbosity = 0
chk_prefix = BinaryBH_
plot_prefix = BinaryBHPlot_
#restart_file = BinaryBH_000360.3d.hdf5
output_path = “”
```

Increase for more diagnostic info

Checkpoint from which to restart simulation. Default reads from run_dir/hdf5, otherwise must write full path starting with /

Path for output files, default reads to run_dir

```
# HDF5files are written every dt = L/N*dt_multiplier*checkpoint_interval
checkpoint_interval = 100
# set to 0 to turn off plot files (except at t=0 and t=stop_time)
# set to -1 to never ever print plotfiles
plot_interval = 10
num_plot_vars = 3
plot_vars = chi Weyl4_Re Weyl4_Im
```

Relatively self-explanatory!

```
# subpaths - specific directories for hdf5, pout, extraction data
# (these are created at runtime)
hdf5_subpath = "hdf5"
pout_subpath = "pout"
data_subpath = "data"

# change the name of output files
# pout_prefix = "pout"
print_progress_only_to_rank_0 = 1

# ignore_checkpoint_name_mismatch = 0
# write_plot_ghosts = 0
```

Housekeeping for output files

Important for visualisation

```
# Grid parameters

# 'N' is the number of subdivisions in each direction of a cubic box
# 'L' is the length of the longest side of the box, dx_coarsest = L/N
# NB - If you use reflective BC and want to specify the subdivisions and side
# of the box were there are no symmetries, specify 'N_full' and 'L_full' instead
# NB - if you have a non-cubic grid, you can specify 'N1' or 'N1_full',
# 'N2' or 'N2_full' and 'N3' or 'N3_full' ( then dx_coarsest = L/N(max) )
# NB - the N values need to be multiples of the block_factor
N_full = 64
L_full = 512
```

N - number of coarsest mesh point

For MPI - see later

```
# Maximum number of times you can regrid above coarsest level
max_level = 9 # There are (max_level+1) grids, so min is zero
```

Total number of refinement levels above coarsest base grid

```
# Frequency of regridding at each level and thresholds on the tagging
# Need one for each level except the top one, ie max_level items
# Generally you do not need to regrid frequently on every level
# Level Regridding: 0   1   2   3   4   5   6   7   8
regrid_interval =   0   0   0   64  64  64  64  64  64
regrid_threshold = 0.05
```

<u>Where</u> to regrid e.g. `GRChombo/Source/TaggingCriteria/PhiAndKTaggingCriterion.hpp`

```
data_t criterion = m_dx * (sqrt(mod_d1_phi) / m_threshold_phi +
                           sqrt(mod_d1_K) / m_threshold_K);
```

`GRChombo/Source/GRChomboCore/GRAMRLevel.cpp`

```
        IntVect iv(ix, iy, iz);
        if (tagging_criterion(iv, 0) >= m_p.regrid_threshold)
        {
```

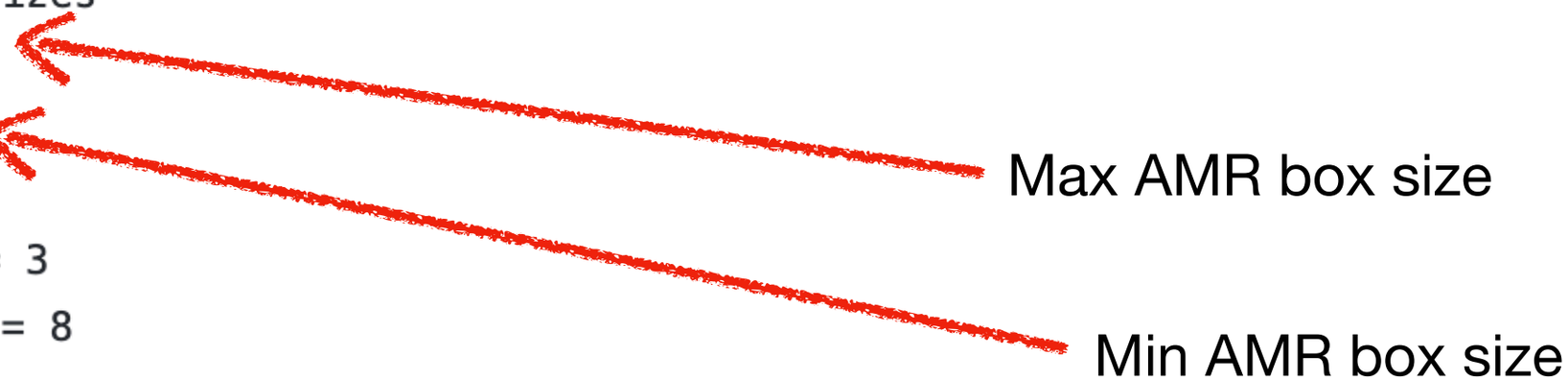
<u>How often</u> to regrid measured in time steps <u>on that</u> level. E.g.

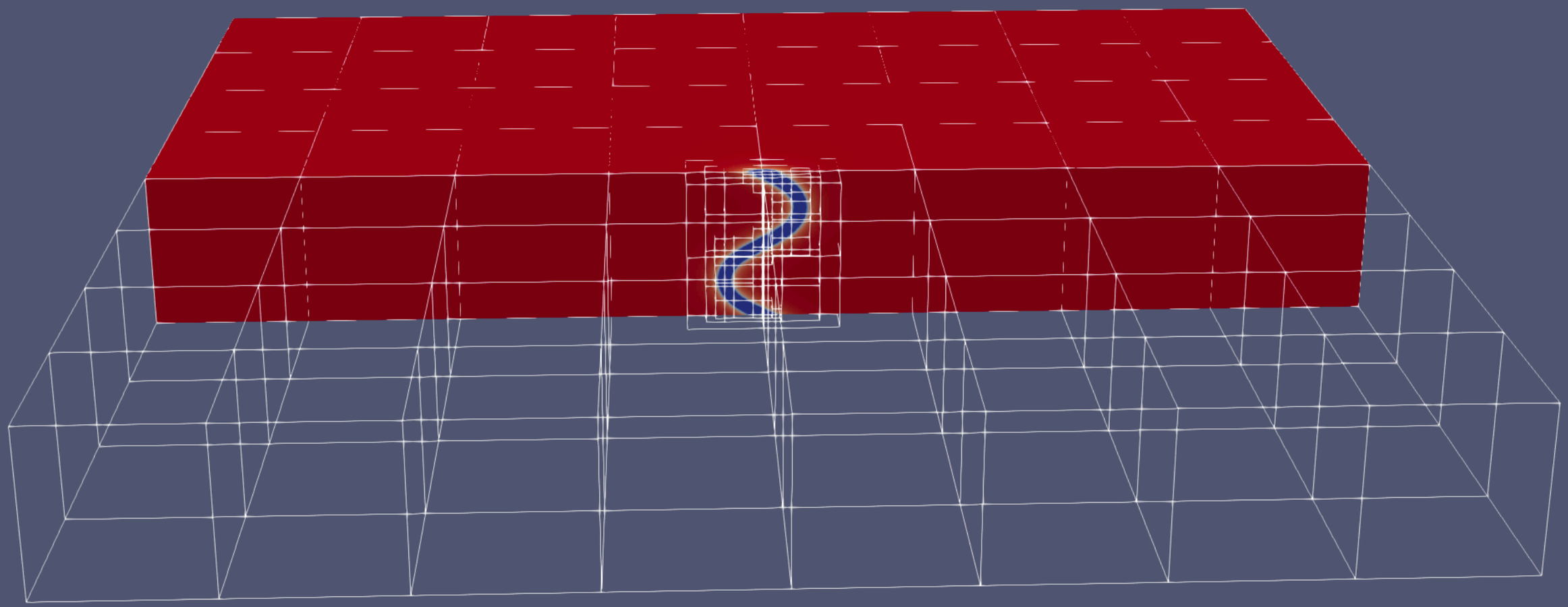
- 64 on regrid level 3 - regrid every 64 subsets on that level, corresponding to 64/(2^3) steps on coarsest level

See arXiv:2112.10567 for more information about different methods of regridding

```
# Max and min box sizes
max_box_size = 16
min_box_size = 16

# tag_buffer_size = 3
# grid_buffer_size = 8
# fill_ratio = 0.7
# num_ghosts = 3
# center = 256.0 256.0 256.0 # defaults to center of the grid
```





```
# Boundary Conditions parameters

# Periodic directions - 0 = false, 1 = true
isPeriodic = 0 0 0
# if not periodic, then specify the boundary type
# 0 = static, 1 = sommerfeld, 2 = reflective
# (see BoundaryConditions.hpp for details)
hi_boundary = 1 1 1
lo_boundary = 1 1 2

# if reflective boundaries selected, must set
# parity of all vars (in order given by UserVariables.hpp)
# 0 = even
# 1,2,3 = odd x, y, z
# 4,5,6 = odd xy, yz, xz
# 7     = odd xyz
vars_parity             = 0 0 4 6 0 5 0    #chi and hij
                          0 0 4 6 0 5 0    #K and Aij
                          0 1 2 3          #Theta and Gamma
                          0 1 2 3 1 2 3    #lapse shift and B
vars_parity_diagnostic = 0 1 2 3           #Ham and Mom
                          0 7              #Weyl
```

Periodic BCs - instructions in params file (see `BoundaryConditions.hpp` for details)

```
# Evolution parameters

# dt will be dx*dt_multiplier on each grid level
dt_multiplier = 0.25
stop_time = 2200.0
# max_steps = 100

# Spatial derivative order (only affects CCZ4 RHS)
max_spatial_derivative_order = 4 # can be 4 or 6

nan_check = 1
```

Self explanatory!

```
#coefficient for KO numerical dissipation
sigma = 0.3
```

Kreiss-Oliger coefficient for dissipating high frequency modes - max value depends on dt_multiplier

```
# Extraction parameters

# extraction_center = 256 256 256 # defaults to center
activate_extraction = 1
num_extraction_radii = 2
extraction_radii = 50.0 100.0
extraction_levels = 2 1
num_points_phi = 24
num_points_theta = 37
num_modes = 8
modes = 2 0 # l m for spherical harmonics
        2 1
        2 2
        4 0
        4 1
        4 2
        4 3
        4 4

# integral_file_prefix = "Weyl4_mode_"

# write_extraction = 0
# extraction_subpath = "data/extraction" # directory for 'write_extraction = 1'
# extraction_file_prefix = "Weyl4_extraction_"
```

Self explanatory!

Outputs extraction for each timestep in separate text file

Adding New Parameters:

1. Create `SimulationParameters.hpp`, inherit from from `SimulationParametersBase` or `ChomboParameters` (no GR)

2. Define parameters and collections of parameters if needed (cleaner passing to functions in ..Level.cpp), e.g. `GRChombo/Examples/AxionString/AxionStringLevel.cpp`

```
BoxLoops::loop(make_compute_pack(SetValue(0.0), AxionString(m_p.initial_params, m_dx, xsect)),
               m_state_new, m_state_new, FILL_GHOST_CELLS, disable_simd());
```

3. Load parameter values from params file

```
#ifndef SIMULATIONPARAMETERS_HPP_
#define SIMULATIONPARAMETERS_HPP_

//General includes
#include "ChomboParameters.hpp"
#include "GRParmParse.hpp"

//Problem specific includes
#include "AxionString.hpp"
#include "CylindricalExtraction.hpp"

class SimulationParameters : public ChomboParameters
{
public:
  SimulationParameters(GRParmParse &pp) : ChomboParameters(pp)
  {
    read_params(pp);
  }

  /// Read parameters from the parameter file
  void read_params(GRParmParse &pp)
  {
```

```
//Collection of parameters necessary for initial conditions
AxionString::params_t initial_params;

//Collection of parameters necessary for extraction
CylindricalExtraction::params_t extraction_params_cylinder;
```

```
// General parameters
int N3;
string path_to_ICs;
int damping;
Real regrid_threshold_phi;
double sigma; // Kreiss-Oliger dissipation parameter
int nan_check;
```

```
// Initial data
pp.load("amplitude", initial_params.amplitude);
pp.load("centerSF", initial_params.centerSF);
//        center; // Default centerSF to center in ChomboParameters

// Cross-section parameters
pp.load("lambda", initial_params.lambda);
pp.load("N3", N3);

initial_params.zlength = N3 * (L / max_N);
```

Note Default Parameters:

- Defined in `GRChombo/Source/GRChomboCore/ChomboParameters.hpp`

- Important: `ref_ratio` defaults to 2 - ratio of dx between refinement levels

```cpp
#ifndef CHOMBOPARAMETERS_HPP_
#define CHOMBOPARAMETERS_HPP_

// General includes
#include "BoundaryConditions.hpp"
#include "GRParmParse.hpp"

class ChomboParameters
{
  public:
    ChomboParameters(GRParmParse &pp) { read_params(pp); }

    void read_params(GRParmParse &pp)
    {
        pp.load("verbosity", verbosity, 0);
        // Grid setup
        pp.load("L", L, 1.0);
        pp.load("center", center,
                {0.5 * L, 0.5 * L, 0.5 * L}); // default to center
        pp.load("regrid_threshold", regrid_threshold, 0.5);
        pp.load("num_ghosts", num_ghosts, 3);
        pp.load("tag_buffer_size", tag_buffer_size, 3);
        pp.load("dt_multiplier", dt_multiplier, 0.25);
        pp.load("fill_ratio", fill_ratio, 0.7);

        // Periodicity and boundaries
        pp.load("isPeriodic", isPeriodic, {true, true, true});
        int bc = BoundaryConditions::STATIC_BC;
        pp.load("hi_boundary", boundary_params.hi_boundary, {bc, bc, bc});
        pp.load("lo_boundary", boundary_params.lo_boundary, {bc, bc, bc});
```

Note: this is only centre for cubic boxes

Questions?