Detail information of BHAC Code

Yosuke Mizuno
Tsung-Dao Lee Institute,
Shanghai Jiao Tong University

BHAC tutorial III, October 17th, 2022

Table of Contents

- Job script for TDLI Astro cluster
- Information about parameter file
- Information about modules & post-process
- Information about src files

Job script for TDLI Astro cluster

- For TDLI Astro cluster user, need to submit job script
- Create "job.sh"

```
#!/bin/bash
#SBATCH --partition=normal (or student)
#SBATCH --ntasks-per-node=48
#SBATCH --nodes=1
#SBATCH --time=1-00:00:00
#SBATCH --job-name=2DFMtorus
#SBATCH --mail-type=ALL
#SBATCH -o output/out # STDOUT
#SBATCH -e output/err # STDERR

module load parallel_studio_xe/2020

mpirun -genv I_MPI_OFI_PROVIDER=tcp -np 48 ./bhac -i amrvac.par
```

- Then submit job: $sbatch job.sh

Job script for TDLI Astro cluster

- For TDLI Astro cluster user,
- Running job is managed by “slurm”
- Some useful command
- check available nodes: $sinfo
- check submitted job status: $squeue
- submit job script: $sbatch <job_script>
- cancel job: $scancel <job_ID>

Detail inside BHAC code (amrvac.par)

- amrvac.par : parameter set for initial setup

```
&filelist
   primnames       = 'rho u1 u2 u3 p b1 b2 b3 s se tr1 lfac xi' : primitive variable name
   autoconvert     = .false.
   saveprim        = .true. : save primitive variable
   convert_type    = 'vtuBCCmpi' : output file format (vtu file)
   slice_type      = 'vtuCC' : slice file format (vtu file)
   filenameini     = 'output/data' : input file name
   filenameout     = 'output/data' : output file name
   filenamelog     = 'output/amrvac' : output file log name
   nwauxio         = 26 : number of output variables
   typeparIO       = -2 : output IO type (parallel or serial)
 &end
```

Detail inside BHAC code (amrvac.par)

&savelist : output information

itsave(1,1) = 0 : amrvac.log file, itsave: initial time save
itsave(1,2) = 0 : output file
itsave(1,3) = 0 : slice file
itsave(1,4) = 0 : collapse file (integration of direction)
itsave(1,5) = 0 : integrated variables (mass accretion rate etc)

ditsave(1) = 10 : ditsave: save number of each iteration
dtsave(2) = 10 : dtsave: save number of simulation time (GM/c^2)
dtsave(3) = 10
dtsave(4) = 10
dtsave(5) = 1

itsave: save at exact iteration number
ditsave: save at each iteration
dtsave: save at each time step

Detail inside BHAC code (amrvacpar.t)

&savelist : output information

collapse(1) = .true. : collapse file information (1,2,3) means direction
collapse(2) = .true.
collapse(3) = .true.
collapseLevel = 1 : collapse level (if you have AMR)

nslices = 5 : number of slice output
slicedir(1) = 3 : slice direction (1,2,3)
slicecoord(1) = 0.0d0 : slice position
slicedir(2) = 2 :
slicecoord(2) = 1.570796d0
.....
&end

Detail inside BHAC code (amrvacpar.t)

&stoplist : simulation finish set
tmax = 15000 : maximum simulation time
dtmin = 1.d-6 : minimum time stepping (dt)
&end

- Others (itmax : maximum iteration number)

Detail inside BHAC code (amrvacpar.t)

&methodlist : choosing method information

wnames = 'd s1 s2 s3 tau b1 b2 b3 Ds Dse dtr1 lfac xi' : conserved variable name

typeadvance = 'twostep' : selected time advance scheme (advance.t)

typefull1 = 13*'tvdlf' : selected approximate Riemann solver scheme (tvdlf.t)

typelimiter1 = 13*'ppm' : selected reconstruction scheme (mod_limiter.t)

typeemf = 'uct2' : selected constrained transport scheme (tvdlf.t)

typepoly = 'gammie' : selected way to calculate of maximum wave speed (amrvacphys.t)

typeinversion = '2D1DEntropy' : selected inversion procedure scheme (con2prim.t)

maxitnr = 30 : maximum number of iteration for inversion procedure

....

&end

Detail inside BHAC code (amrvacpar.t)

&boundlist : boundary condition information (boundary_condition.t)

dixB = 4 : number of ghost cell each direction

typeB = : boundary condition setup

5*’noinflow',3*'cont',5*'noinflow' : x1 direction (inner)

5*’noinflow',3*'cont',5*'noinflow' : x1 direction (outer)

‘symm','symm','asymm',2*'symm','symm','asymm',6*'symm' : x2 direction (in)

‘symm','symm','asymm',2*'symm','symm','asymm',6*'symm' : x2 direction (out)

13*’periodic' : x3 direction (in)

13*’periodic' : x3 direction (out)

….

&end

Detail inside BHAC code (amrvacpar.t)

```
&amrlist : grid, simulation box, AMR setup
      mxnest          = 3 : number of AMR level
      errorestimate    = 0 : AMR error estimate (errest.t)
      nxlone1         = 96 : base grid number (x1 direction)
      nxlone2         = 48 : base grid number (x2 direction)
      nxlone3         = 48 : base grid number (x3 direction)
      xprobmin1        = 0.17 : inner boundary position of x1 direction
      xprobmax1         = 7.824046010856292 : outer boundary position of x1 direction
      xprobmin2        = 0.0d0 : inner boundary position of x2 direction
      xprobmax2         = 0.5d0 : outer boundary position of x2 direction
      xprobmin3        = 0.0d0 : inner boundary position of x3 direction
      xprobmax3         = 1.0d0 : outer boundary position of x3 direction
....
 &end
```

Detail inside BHAC code (amrvacpar.t)

For AMR usage

&amrlist : grid, simulation box, AMR setup

```
      mxnest          = 3 : number of AMR level
      errorestimate    = 3 : AMR error estimate: Lohner type criteria (errest.t)
      flags(13)       = 2 : AMR refine criteria total number (2 per 13 variables)
      flags(1)        = 1 : AMR refine criteria quantities (1 = density)
      flags(2)        = 5 : AMR refine criteria quantities (5 = pressure)
      wflags(1)       =0.5 : AMR refine criteria wait for flag(1)
      wflags(5)       =0.5 : AMR refine criteria wait for flag(2)
      tol             =13*0.1 : tolerance for AMR refine
      amr_wavefilter =13*0.05 : AMR wave filter (Lohner type criteria special constant)
      nbufferx1        = 0 : AMR refine in ghost cell (0=no, 1=yes)
      nbufferx2        = 0
....
 &end
```

Detail inside BHAC code (amrvacpar.t)

&paramlist : additional parameter setup

slowsteps = 0

typecourant = ‘maxsum' : method for calculate dt (setdt.t)

courantpar = 0.85d0 : CFL number

&end

Detail inside BHAC code (definitions.h)

- definitions.h : switch the specific physical module usage
  - e.g., GLM, HDF5, ENTROPY, STAGGERED, ELECTRONS, RADCOOL, COULOMB, PARTICLES etc
- In program files,
  - {#IFDEF XXXX
  - }
- If defined specific physical module, we read inside lines.
- If want to make new module, first define module name, and add switch in subroutine.

Detail inside BHAC code (postprocess)

- After run the simulation, you can do some post-process
- 1. restart simulation
  - $ mpiexec -n X ./bhac -i amrvac.par -restart 1000 (-slice 1001) (-collapse 1001)

- 2. post conversion, collapse, slice
  - autosave = .false. => save = .true.
  - set stoplist => itreset = .true., itmax = 0
  - $ mpiexec -n X ./bhac -i amrvac.par -restart 1000 (-slice 1000) (-collapse 1000)

Detail inside BHAC code (src)

- Inside $BAHC_DIR/src there are stored many main program files
  - amrvac.t : main routine
  - amrvacdef.t : common block (defined variables)
  - amrvacio/amrio.t : subroutine for I/O
  - amrvacio/convert.t : subroutine for convert files
  - advance.t : time evolution scheme
  - boundary_conditions.t : subroutine for boundary condition
  - mod_limiter.t : subroutine for reconstruction scheme
    - some of reconstruction schemes have separated file (e.g., ppm.t, weno5.t, mp5.t)
  - tvdlf.t : Approximate Riemann solver
  - setdt.t : subroutine for calculate nextstep dt

Detail inside BHAC code (src)

- errest.t : AMR criteria subroutine
- forest.t, coarsen.t : AMR related subroutine

- geometry directory includes metric related subroutine
  - geometry/mod_metric.t : subroutine for metric calculation
  - geometry/mod_coord_XXX.t : subroutine for each metrics
  - geometry/mod_transform.t : subroutine for coordinate transformation

- If you want to add new metric, make new “mod_coord_XXX.t” writing related subroutines

Detail inside BHAC code (src)

- rmhd directory includes subroutines for solving GRMHD equations
  - rmhd/amrvacpar.t : setup for parameters
  - rmhd/amrvacphys.t : subroutine for solving GRMHD equations (physics)
  - rmhd/con2prim.t : subroutine for inversion procedure (conserved to primitive)
  - rmhd/collectaux.t : subroutine for calculation Lorentz factor, xi
  - rmhd/eos.t : subroutine for equation of state
  - rmhd/fct.t : subroutine for constrained transport scheme
  - rmhd/mod_aux.t : subroutine for useful calculation
  - rmhd/postrad.t : subroutine for conversion to postrad file format (for GRRT calculation)

Detail inside BHAC code (src)

- rmhd directory includes subroutines for solving GRMHD equations
- For make new physical module => add subroutine in amrvacphys.t file.
- For make new EoS model => add subroutine in eos.t file

Detail inside BHAC code (model)

- Each simulation model needs following files in model directory:
  - amrvacsettings.t : AMR cell grid number setting
  - amrvacusr.t : the whole problem setup for initial conditions and boundary conditions
  - amrvacusrpar.t : model defined special parameter
  - amrvac.par : parameter file
  - definition.h : definition of physics module
  - makefile : makefile for compile

- Modified simulation setup (initial condition, output variables etc) needs to change amrvacusr.t

Detail inside BHAC code (model)

- Modified simulation setup (initial condition, output variables etc) needs to change amrvacusr.t
  - initglobaldata_usr : model specific parameter setup
  - initonegrid_usr : model setup
  - specialbound_usr : special boundary condition
  - specialvar_output : special variable output setting
  - specialvarnames_output
  - specialsource : special source variables
  - specialeta: special eta setup for resistive MHD (not used now)
  - specialrefine_grid : AMR refine setup for model specified
  - specialvarforerrest : special setup for AMR refile criteria
  - specialset_B0 : adding steady potential background field (not used)

BHAC Tips

- BHAC code using pre-compiler (perl), .t files will translate .f90 files before compile (vacpp.pl)
- Our written form is a bit complicated (need to refer similar treatment)
- *W* variables store conserved or primitive variables. Please carefully check which variables are stored in each subroutine.