Startup
Outline

Sequence of procedures for starting or restarting CHIMERA3D

Program **radhyd_MPI** is the top level program that initiates and executes CHIMERA3D.

- The following sequence reads in the array dimensions, the data paths (where to find the restart files and where to send the output data), the header, the parameter 'nrst' (the cycle number from which to start or restart the run, the parameter 'nouttmp' which specifies the mode of the restart, and the model dimensions (imin - minimum radial zone, imax - maximum radial zone, jmin - minimum angular zone, jmax - maximum angular zone, kmin - minimum azimuthal zone, kmax - maximum azimuthal zone)

radhyd_MPI

1. **MPI_INIT** [Computes the total neutrino energy density and energy flux, needed to compute PN gravity.]
2. **MPI_COMM_RANK** [Computes the angular average of density, matter pressure, matter energy, radial velocity, angular velocity, velocity squared, neutrino energy density, and neutrino flux.]
3. **MPI_COMM_SIZE** [Using the initial grid, computes the gravitational potential and forces.]

   The following is performed on processor 0 only.
4. **read_pack_array_dimensions** [Controls the input of the array dimensions, data paths, and the number of active radial, angular, and azimuthal zones as given by the restart model.]
5. Opens Data3/Initial_Data/array_dimensions.d
   read_array_dimensions [Reads in the array dimensions and data paths from file array_dimensions.d in directory Data3/Initial_Data.]
   Closes Data3/Initial_Data/array_dimensions.d
6. Opens Data3/Initial_Data/reset.d
   read_pack_init [From file reset,d in directory Data3/Initial_Data reads the problem header, the parameter nrst, which, if nouttmp /= 1 or 2, tells which cycle number to pick up the restart, and the parameter nouttmp.]
   Closes Data3/Initial_Data/reset.d
7. The following directs the input of the problem dimensions, imin, imax, jmin, jmax, kmin, kmax. This depends on the values of nrst and nouttmp
   nrst:
   (a) nrst = 0 [New problem is initiated.]
      i. Opens Data3/Initial_Data/radhyd_keys.d
      read_model_dimensions [Reads in the model dimensions (radial, angular, and azimuthal dimensions) of the problem.]
      Closes Data3/Initial_Data/radhyd_keys.d

(b) nrst > 0 [Problem is being restarted]

(c) ny = nz = 1 (1-D problem):

(d) Opens Data3/Initial_Data/radhyd_keys.d
read_model_dimensions [Reads in the model dimensions (imin, imax, jmin, jmax, kmin, kmax).]
Closes Data3/Initial_Data/radhyd_keys.d

(e) ny > 0 and/or nz > 0 (Multi-D problem)

i. nouttmp = 1 [Problem is being restarted from temporary restart files "1"]

A. Opens Data_Path/Restart/rst_tmp1_keys.d
read_model_dimensions [Reads in the model dimensions (imin, imax, jmin, jmax, kmin, kmax).]
Closes Data_Path/Restart/rst_tmp1_keys.d

ii. nouttmp = 2 [Problem is being restarted from temporary restart files "2"]

A. Opens Data_Path/Restart/rst_tmp2_keys.d
read_model_dimensions [Reads in the model dimensions (imin, imax, jmin, jmax, kmin, kmax).]
Closes Data_Path/Restart/rst_tmp2_keys.d

iii. nouttmp = 3 [Problem is being restarted from permanent restart files "XXXXXXX"]

A. Opens Data_Path/Restart/restart_keysXXXXXXX.d
read_model_dimensions [Reads in the model dimensions (imin, imax, jmin, jmax, kmin, kmax).]
Closes Data_Path/Restart/restart_keysXXXXXXX.d

iv. nouttmp = 4 [Problem is being restarted from "final" restart files]

A. Opens Data_Path/Restart/restart_final_keys.d
read_model_dimensions [Reads in the model dimensions (imin, imax, jmin, jmax, kmin, kmax).]
Closes Data_Path/Restart/restart_final_keys.d

v. nouttmp = 5 [Multii-D problem is being restarted from a 1-D problem]

A. Opens Data_Path/Restart/restart_keysXXXXXXX.d
read_model_dimensions [Reads in the model dimensions (imin, imax, jmin, jmax, kmin, kmax).]
Closes Data_Path/Restart/restart_keysXXXXXXX.d

8. Checks that the number of processors requested (n_dim_data(7) = n_proc) is equal to the number of processors (num_procs) allocated to the run.

The following is performed on all processors.

9. Broadcast data paths and array dimensions to all processors.
10. **unpack_array_dimenisons** [Unpacks the array dimensions and domain decomposition on each processor.]
11. Assign the log file names for the directory Data_Path/Log_File
12. **load_array_module** [Store the array dimensions, domain decomposition and number of processors in the array module.]
13. Create communicators for the j and k domain decompositions

14. Check that the number of processors assigned to the y and z decompositions is the number requested, and the n_proc_y * n_proc_z, n_proc.

    The following continues the startup sequence.

15. **initialize**
    (a) **dimension_arrays** [Dimensions the arrays in all the modules and zeros or assigns default values to all the variables.]
        i. **dimension_mgfld_arrays**
            A. **dimension_abem_arrays** [Dimensions the arrays and initializes the variables and arrays in **abem_module**.]
            B. **dimension_abem_y_arrays** [Dimensions the arrays and initializes the variables and arrays in **abem_y_module**.]
            C. **dimension_abem_z_arrays** [Dimensions the arrays and initializes the variables and arrays in **abem_z_module**.]
            D. **dimension_brem_arrays** [Dimensions the arrays and initializes the variables and arrays in **brem_module**.]
            E. **dimension_incrmnt_arrays** [Dimensions the arrays and initializes the variables and arrays in **incrmnt_module**.]
            F. **dimension_mdl_cnfg_arrays** [Dimensions the arrays and initializes the variables and arrays in **mdl_cnfg_module**.]
            G. **dimension_mdl_cnfg_y_arrays** [Dimensions the arrays and initializes the variables and arrays in **mdl_cnfg_y_module**.]
            H. **dimension_mdl_cnfg_z_arrays** [Dimensions the arrays and initializes the variables and arrays in **mdl_cnfg_z_module**.]
            I. **dimension_nu_dist_arrays** [Dimensions the arrays and initializes the variables and arrays in **nu_dist_module**.]
            J. **dimension_nu_energy_grid_arrays** [Dimensions the arrays and initializes the variables and arrays in **nu_energy_grid_module**.]
            K. **dimension_pair_arrays** [Dimensions the arrays and initializes the variables and arrays in **pair_module**.]
            L. **dimension_scat_a_arrays** [Dimensions the arrays and initializes the variables and arrays in **scat_a_module**.]
            M. **dimension_scat_e_arrays** [Dimensions the arrays and initializes the variables and arrays in **scat_e_module**.]
            N. **dimension_scat_i_arrays** [Dimensions the arrays and initializes the variables and arrays in **scat_i_module**.]
            O. **dimension_scat_n_arrays** [Dimensions the arrays and initializes the variables and arrays in **scat_n_module**.]
            P. **dimension_scat_nA_arrays** [Dimensions the arrays and initializes the variables and arrays in **scat_nA_module**.]
            Q. **dimension_scat_nn_arrays** [Dimensions the arrays and initializes the variables and arrays in **scat_nn_module**.]
        ii. **dimension_edit_arrays** [Dimensions the arrays and initializes the variables and arrays in **edit_module**.]

iii. **dimension_eos_arrays**

A. **dimension_eos_snc_x_arrays** [Dimensions the arrays and initializes the variables and arrays in **eos_snc_x_module**.]

B. **dimension_eos_snc_y_arrays** [Dimensions the arrays and initializes the variables and arrays in **eos_snc_y_module**.]

C. **dimension_eos_snc_z_arrays** [Dimensions the arrays and initializes the variables and arrays in **eos_snc_z_module**.]

D. **dimension_eos_bck_arrays** [Dimensions the arrays and initializes the variables and arrays in **eos_bck_module**.]

E. **dimension_eos_ls_arrays** [Dimensions the arrays and initializes the variables and arrays in **eos_ls_module**.]

F. **dimension_eos_drv_arrays** [Dimensions the arrays and initializes the variables and arrays in **eos_drv_module**.]

iv. **dimension_nucbrn_arrays** [Dimensions the arrays and initializes the variables and arrays **innucbrn_module**.]

v. **dimension_e_advct_arrays** [Dimensions the arrays and initializes the variables and arrays **e_advct_module**.]

(b) **initialize_variables** [Initializes (assigns default values) to variables in modules not containing allocatable arrays, and therefore not initialized in the **dimension_array call**]

i. **initialize_global_var** [Initialize the global hydro variables in module **evh1_global**.]

ii. **initialize_cycle_arrays** [Initialize the variables in **cycle_module**.]

iii. **initialize_it_tol_arrays** [Initialize the variables in **it_tol_module**.]

iv. **initialize_bomb_arrays** [Initialize the variables in **bomb_module**.]

v. **initialize_rezone_arrays** [Initialize the variables in **rezone_module**.]

The following is performed on processor 0 only.

(c) **problem_read** [Orchestrate the reading in of the initial or restart model and problem keys.]

Zero all data buffers

Assign values to 'nread' and 'nprint'

Open **Data3/Initial_Data/reset.d** for reading in 'nrst' and 'nouttmp'
Open **Data_Path/Run_Log/superdump.d** for echoing the read in of problem keys, and for outputting cycle number.

i. **problem_pack** [Directs the reading in of the problem keys and, if the problem is being initiated or if it is 1D and being restarted, the model configuration.]

The following is performed if nrst = 0 (problem is being initiated)

A. **radhyd_read** [Directs the reading and packing of the keys in **radhyd_keys.d**.]
Open **Data3/Initial_Data/radhyd_keys.d**
Call **read_pack_radhyd_keys** to read and pack the keys in radhyd_keys.d.
Close **Data3/Initial_Data/radhyd_keys.d**

B. **eos_read** [Directs the reading and packing of the keys in **eos_keys.d**.]
Open **Data3/Initial_Data/eos_keys.d**
Call **read_pack_radhyd_keys** to read and pack the keys in **radhyd_keys.d**.
Close **Data3/Initial_Data/radhyd_keys.d**

C. **transport_read** [Directs the reading and packing of the keys in **transport_keys.d**.]
Open **Data3/Initial_Data/transport_keys.d**
Call **read_pack_transport_keys** to read and pack the keys in **transport_keys.d**.
Close **Data3/Initial_Data/transport_keys.d**

D. **e_advct_read** [Directs the reading and packing of the keys in **e_advct_keys.d**.]
Open **Data3/Initial_Data/e_advct_keys.d**
Call **read_pack_e_advct_keys** to read and pack the keys in **e_advct_keys.d**.
Close **Data3/Initial_Data/transport_keys.d**

E. **edit_read** [Directs the reading and packing of the keys in **edit_keys.d**.]
Open **Data3/Initial_Data/edit_keys.d**
Call **read_pack_edit_keys** to read and pack the keys in **edit_keys.d**.
Close **Data3/Initial_Data/edit_keys.d**

F. **hydro_read** [Directs the reading and packing of the keys in **hydro_keys.d**.]
Open **Data3/Initial_Data/hydro_keys.d**
Call **read_pack hydro_keys** to read and pack the keys in **hydro_keys.d**.
Close **Data3/Initial_Data/hydro_keys.d**

G. **nuclear_read** [Directs the reading and packing of the keys in **nuclear_keys.d**.]
Open **Data3/Initial_Data/nuclear_keys.d**.
Call **read_pack nuclear_keys** to read and pack the keys in **nuclear_keys.d**.
Close **Data3/Initial_Data/nuclear_keys.d**

H. **model_read** [Directs the reading and packing of the model configuration in **initial_model.d**.]
Open **Data3/model_Data/initial_model.d**.
Call **read_pack_initial_model** to read and pack the keys in **initial_model.d**.
Close **Data3/model_Data/initial_model.d**

The following is performed if nrst > 0 (problem is being restarted)

The following file is opened in order to read in the problem keys.

nouttmp = 1: Open **Data_Path/Restart/rst_tmp1_keys.d**
nouttmp = 2: Open **Data_Path/Restart/rst_tmp2_keys.d**
nouttmp = 3: Open **Data_Path/Restart/restart_keysXXXXXXX.d**
nouttmp = 4: Open **Data_Path//Restart/restart_final_keys.d**
nouttmp = 5: Open **Data_Path/Restart/restart_keysXXXXXXX.d**

From the above opened file, the following read commands are performed.

I. **read_pack_radhyd_keys** [Read and pack the radhyd keys from the above opened file.]

J. **read_pack_eos_keys** [Read and pack the eos keys from the above opened file.]

K. **read_pack_transport_keys** [Read and pack the eos keys from the above opened file.]

L. **read_pack_eos_keys** [Read and pack the transport keys from the above opened file.]

M. **read_pack_e_advect_keys** [Read and pack the e_advect keys from the above opened file.]

N. **read_pack_edit_keys** [Read and pack the edit keys from the above opened file.]

O. **read_pack_hydro_keys** [Read and pack the hydro keys from the above opened file.]

P. **read_pack_nuclear_keys** [Read and pack the nuclear keys from the above opened file.]

Close above opened file.

The following is performed if nrst > 0, ny = 1, nz = 1 (restarting a 1D problem

nouttmp = 1: Open unformated **Data_Path/Restart/rst_tmp1_model.d**
nouttmp = 2: Open unformated **Data_Path/Restart/rst_tmp2_model.d**
nouttmp = 3: Open unformated **Data_Path/Restart/restart_modelXXXXXXX.d**
nouttmp = 4: Open unformated **Data_Path//Restart/restart_final_mod.d**
nouttmp = 5: Open unformated **Data_Path/Restart/restart_modelXXXXXXX.d**

Q. **read_pack_restart_model** [Reads and packs the model configuration.]

Close above opened file.

The following is performed if nrst > 0, ny > 1 and/or nz > 1, and nouttmp = 5 (restarting a Multi-D problem from a 1-D precursor

Open unformated **Data_Path/Restart/restart_modelXXXXXXX.d**

R. **read_pack_restart_model** [Reads and packs the model configuration.]

Close above opened file.

The following is performed if nrst = 0 (problem is being initiated)

ii. Open **Data3/Initial_Data/reset_initial.d**
Write reset_initial.d for use as reset.d file if initiating the same simulation.

iii. Close **Data3/Initial_Data/reset_initial.d**

(d) **data_check** [Check array dimensions, problem dimensions, domain decomposition for cinsistency]

(e) **rezone** [Set up y-grid, z-grid]

The following is performed on all processors.

Broadcast radhyd_keys data to all processors.
Broadcast eos_keys data to all processors.
Broadcast transport_keys data to all processors.
Broadcast e_advect_keys data to all processors.
Broadcast edit_keys data to all processors.
Broadcast hydro_keys data to all processors.
Broadcast nuclear_keys data to all processors.
Broadcast rezone data to all processors.

(f) **unpack_arrays** [Directs the unpacking and loading into array modules of the problem keys, and for the case in which nrst = 0 (initialing a simulaion), or ny = nz = 1 (restarting a 1-D simulation), or nouttmp = 5 (restarting a multi-D simulation from a 1-D precursor) loading the model configuration into the radial_ray module.]

i. **unpack_init** [Unpacks nrst, nouttmp, and the problem header, packs the first into **cycle_module** and the second and third into **edit_module**]

ii. **unpack_radial_ray_keys** [Unpacks the radhyd_keys and loads into **radial_ray_module**]

iii. **unpack_rezone_arrays** [Unpacks the rezone parameters and arrays and loads into **radial_ray_module**, **evh1_sweep**. and **evh1_global**.]

iv. **unpack_eos_keys** [Unpacks the eos_keys and loads into **eos_snc_x_module**, **eos_snc_y_module**, **eos_snc_z_module**, **prb_cntl_module**, and **radial_ray_module**.]

v. **unpack_transport_keys** [Unpacks the transport_keys and loads into **cycle_module**, **eos_snc_x_module**, **it_tol_module**, **nu_dist_module**, **nu_energy_grid_module**, **prb_cntl_module**, and **t_cntrl_module**.]

vi. **unpack_e_advct_keys** [Unpacks the e_advct_keys and loads into **e_advct_module**.]

vii. **upack_edit_keys** [Unpacks the edit_keys and loads into **edit_module** and **radial_ray_module**.]

viii. **unpack_hydro_keys** [Unpacks the hydro_keys and loads into **bomb_module**, **convect_module**, **evh1_global**, **prb_cntl_module**, **shock_module**, and **t_cntrl_module**.]

ix. **unpack_nuclear_keys** [Unpacks the nuclear_keys and loads into **nucbrn_module**, **radial_ray_module**, **eos_snc_x_module**, **eos_snc_y_module**, and **eos_snc_z_module**.]

If nrst = 0 (initialing a simulaion), or ny = nz = 1 (restarting a 1-D simulation)

x. **unpack_initial_model** [Unpacks the initial model and loads into **eos_snc_x_module**, **mdl_cnfg_module**, and **radial_ray_module**.]

(g) If nouttmp = 5 (restarting a multi-D simulation from a 1-D precursor) **unpack_restart_model_to_node** [Unpacks the initial model on all processors and loads into **edit_module**, **eos_snc_x_module**, **mdl_cnfg_module**, **nucbrn_module**,

nu_dist_module, and **radial_ray_module**.]

If nrst /= 0 and ny > 1 or nz > 1 and nouttmp /= 5

nouttmp = 1: Open unformated **Data_Path//Restart/rst_tmp1_'myid',.d**
nouttmp = 2: Open unformated **Data_Path//Restart/rst_tmp1_'myid',.d**
nouttmp = 3: Open unformated **Data_Path/Restart/restart_mode'myid'_XXXXXXX.d**
nouttmp = 4: Open unformated **Data_Path//Restart/restart_final_mod'myid'.d**

read_restart_model_to_node [Reads in the model configuration on each processor and loads the data into **edit_module**, **eos_snc_x_module**, **mdl_cnfg_module**, **nucbrn_module**, **nu_dist_module**, and **radial_ray_module**.]

(h) **model_check** [Checks to ensure that the actual model dimensions are consistent with the declared model dimensions.]

(i) **nse_set** [Loads the nse array flags into **eos_snc_x_module**, **nucbrn_module**, and **radial_ray_module**.]

(j) **ye_fix** [Adjusts the electron fraction in nonNSE zones to be consistent with the composition.]

(k) **solid_angle** [Computes the solid angle subtended by each radial ray.]

(l) **problem_setup** [Directs the computation of the dependent variables needed to begin the simulation.]

i. The following is performed if nrst = 0:

ii. **setup_initial_hydro** [Computes the dependent variables in the hydro sector.]

A. Compute pseudoviscous multipliers (for edit purposes)

B. Set optional prescribed values of velocities.

C. **pblmst1** [Optionally modify problem before GR or transport sector dependent variables computed.]

D. **esrgnz_x** [Load eos table arrays.]

E. **eqstz_x** [Interpolate eos quantities.]

F. **gammaz_x** [Compute adiabatic exponents.]

G. Put NSE composition (protons, neutrons, helium, and representative heavy nucleus) in composition arrays.

H. Compute pressure boundary condition at outer edge.

iii. **angular_ave** [Compute the angular averages of quantities.]

iv. **regridder** [Adjust the radial grid to maintain constant Delta rho/rho from the center to the NSE-nonNSE boundary.]

v. The following is performed if nrst > 0:

vi. **setup_restart_hydro** [Computes the dependent variables in the hydro sector.]

A. Compute pseudoviscous multipliers (for edit purposes)

B. Set optional prescribed values of velocities.

C. **pblmst1** [Optionally modify problem before GR or transport sector dependent variables computed.]

D. **esrgnz_x** [Load eos table arrays.]

E. **eqstz_x** [Interpolate eos quantities.]

F. **gammaz_x** [Compute adiabatic exponents.]
G. Put NSE composition (protons, neutrons, helium, and representative heavy nucleus) in composition arrays.
H. Compute pressure boundary condition at outer edge.

vii. **angular_ave** [Compute the angular averages of quantities.]
viii. **initialize_grid** [Initialize final grid to initial grid.]
ix. **radhyd_to_poisson** [Evaluate and save the weights for solid angle integration of the Poisson equation for gravity.]
x. **e_zone** [Compute the energy zoning, as seen by a distant observer, for the neutrinos.]
xi. The following is performed if nrst = 0:
xii. **setup_initial_GR** [Computes the dependent variables in gravity sector when initiating a simulation.]
A. **agr_nu_cal** [Load neutrino lapse functions with preliminary values.]
B. **enu_cal** [Correct neutrino energies for lapse, and compute coefficients of the total neutrino number and energy.]

Iterate on the following three subroutine calls
C. **radhyd_to_nu_energy_flux** [Compute neutrino energy density and fluxes.]
D. **angular_ave** [Compute the angular averages of quantities.]
E. **radhyd_to_grav_i** [Direct the computation of gravity.]

. . . **grav_Newton** [Compute spherically symmetric Newtonian gravity]
. . . **greff** [Compute spherically symmetric Post-Newtonian gravity]
. . . **poisson** [Compute Spherical harmonics of Newtonian gravity; add PN spherically symmetric gravity to Spherical harmonics and subtract newtonian spherically symmetric gravity]
. . . **agr_cal** [Compute lapse functions]
. . . **agr_nu_cal** [Load neutrino lapse functions.]
. . .
F. **enu_cal** [Correct neutrino energies for lapse, and compute coefficients of the total neutrino number and energy.]

xiii. The following is performed if nrst > 0:
xiv. **setup_restart_GR** [Computes the dependent variables in gravity sector when restarting a simulation.]
A. **agr_nu_cal** [Load neutrino lapse functions with preliminary values.]
B. **enu_cal** [Correct neutrino energies for lapse, and compute coefficients of the total neutrino number and energy.]

Iterate on the following three subroutine calls
C. **radhyd_to_nu_energy_flux** [Compute neutrino energy density and fluxes.]
D. **angular_ave** [Compute the angular averages of quantities.]
E. **radhyd_to_grav_i** [Direct the computation of gravity.]

. . . **grav_Newton** [Compute spherically symmetric Newtonian gravity]

... **greff** [Compute spherically symmetric Post-Newtonian gravity]
... **poisson** [Compute Spherical harmonics of Newtonian gravity; add PN spherically symmetric gravity to Spherical harmonics and subtract newtonian spherically symmetric gravity]
... **agr_cal** [Compute lapse functions]
... **agr_nu_cal** [Load neutrino lapse functions.]
...

F. **enu_cal** [Correct neutrino energies for lapse, and compute coefficients of the total neutrino number and energy.]

xv. The following is performed if nrst = 0:

xvi. **setup_initial_trans** [Computes the dependent variables in transport sector when initiating a simulation.]

A. **gamgr_nu_cal** [Load relativistic gammas for neutrinos.]

B. **gamgr_nu_cal** [Load relativistic gammas for neutrinos.]
... **agr_nu_cal** [Load neutrino lapse functions.]
...

C. **enu_cal** [Correct neutrino energies for lapse, and compute coefficients of the total neutrino number and energy.]
...

D. **pre_trans** [Compute geometric factors and functions of the neutrino energy needed for transport.]
...

E. **pblmst2** [Optionally modify the problem given the neutrino energies.]
...

F. **read_ec_table** [Read in table of rates for electron capture on nuclei.]
...

G. **abemset** [Compute neutrino absorption and emission tables.]
...

H. **scateset** [Compute neutrino-electron scattering tables.]
...

I. **scatiset** [Compute neutrino-nucleon and nuclei isoenergetic scattering tables.]
...

J. **pairset** [Compute electron-positron pair annihilation rate tables.]
...

K. **bremset** [Compute neutrino-nucleon bremsstrahlung tables.]
...

L. **scatnset** [Compute neutrino-nucleon nonisoenergetic scattering tables.]
...

M. **abemrate** [Interpolate neutrino-electron scattering tables.]
...

N. **sctirate** [Interpolate neutrino-nucleon and nuclei isoenergetic scattering tables.]
...

O. **scterate** [Interpolate neutrino-electron scattering tables.]
...

P. **pairrate** [Interpolate electron-positron pair annihilation rate tables.]
...
Q. **bremrate** [Interpolate neutrino-nucleon bremsstrahlung tables.]
...
R. **sctnrate** [Interpolate neutrino-nucleon nonisoenergetic scattering tables.]
...
S. **mfp_cal** [Compute neutrino inverse mean free paths.]
...
T. **nu_sphere** [Compute neutrinospheres.]
...
U. **psi1_cal** [Compute the first angular moments of the neutrino occupation numbers.]
...
V. **nu_number** [Compute total neutrino numbers and energies,]
...
W. **nu_stress_x** [Compute the neutrino stresses,]
...
X. **eddington** [Compute total neutrino Eddington factors,]
...
Y. **nu_U** [Compute the isotropic neutrino pressure and energy,]

xvii. The following is performed if nrst > 0:

xviii. **setup_initial_trans** [Computes the dependent variables in transport sector when initiating a simulation.]

A. **gamgr_nu_cal** [Load relativistic gammas for neutrinos.]
B. **gamgr_nu_cal** [Load relativistic gammas for neutrinos.]
... **agr_nu_cal** [Load neutrino lapse functions.]
...
C. **enu_cal** [Correct neutrino energies for lapse, and compute coefficients of the total neutrino number and energy.]
...
D. **pre_trans** [Compute geometric factors and functions of the neutrino energy needed for transport.]
...
E. **pblmst2** [Optionally modify the problem given the neutrino energies.]
...
F. **read_ec_table** [Read in table of rates for electron capture on nuclei.]
...
G. **abemset** [Compute neutrino absorption and emission tables.]
...
H. **scateset** [Compute neutrino-electron scattering tables.]
...
I. **scatiset** [Compute neutrino-nucleon and nuclei isoenergetic scattering tables.]
...

J. **pairset** [Compute electron-positron pair annihilation rate tables.]
...
K. **bremset** [Compute neutrino-nucleon bremsstrahlung tables.]
...
L. **scatnset** [Compute neutrino-nucleon nonisoenergetic scattering tables.]
...
M. **abemrate** [Interpolate neutrino-electron scattering tables.]
...
N. **sctirate** [Interpolate neutrino-nucleon and nuclei isoenergetic scattering tables.]
...
O. **scterate** [Interpolate neutrino-electron scattering tables.]
...
P. **pairrate** [Interpolate electron-positron pair annihilation rate tables.]
...
Q. **bremrate** [Interpolate neutrino-nucleon bremsstrahlung tables.]
...
R. **sctnrate** [Interpolate neutrino-nucleon nonisoenergetic scattering tables.]
...
S. **mfp_cal** [Compute neutrino inverse mean free paths.]
...
T. **nu_sphere** [Compute neutrinospheres.]
...
U. **psi1_cal** [Compute the first angular moments of the neutrino occupation numbers.]
...
V. **nu_number** [Compute total neutrino numbers and energies,]
...
W. **nu_stress_x** [Compute the neutrino stresses,]
...
X. **eddington** [Compute total neutrino Eddington factors,]
...
Y. **nu_U** [Compute the isotropic neutrino pressure and energy,]

(m) **network_setup** [Reads in and broadcasts the nuclear and reaction rate data for the nuclear network.]

(n) **load_evh1_arrays** [loads EVH1 parameters into **radial_ray_module**, **evh1_bound** and **evh1_global**.]

(o) **time_step_check** [Computes the minimum Courant time]

16. **radhyd_to_edit** [Performs the initial edit of the model.]
17. **radhyd_to_monitor_MPI** [Computes initial values for monitoring energy and lepton conservation.]