global_star

The goal of this patch is to run MHD, global simulations of stellar magneto-convection. It's built on top of the 'mhdconv' patch and most of the work was actually done by Reese (?).

The main updates are:

- Spherical treatment of gravity (Reese patch)
- Spherical initial conditions (Reese patch)
- Treatment of boundary conditions in spherical symmetry (Reese patch)
- Heating/cooling treatment in spherical symmetry (this patch)

I'll just comment on the changes I made to the heating/cooling treatment.

In 'mhdconv', the heating and cooling rates were equal because the volume of material subject to heating/cooling was the same (box-in-a-star setup). In global models, the outer layer (cooling) occupies a much larger volume than the inner one (heating). To avoid the domain getting systematically cooled down, one need to normalize the rates. This is done in two steps:

1. First, the cooling and heating volumes are computed.
2. Then the cooling rate gets normalized: cooling_rate → cooling_rate * heat_volume/cooling_volume

This is implemented in the '*eneana*' routine. The first time the routine is called (in the '*energy_fine'* loop), the flag `compute_volumes` flag is set to `.true.`, therefore the volumes are computed. At the end of the '*energy_fine*' routine, the `compute_volumes` flag is set to `.false.` and an MPI all_reduce is used to compute the total heating and cooling volumes. After that, everytime the '*eneana'* routine is called, the normalized cooling rate will be used, so that the total energy is conserved.

- The heating/cooling rates are also normalized according to a cosine profile (see `compute_vol_energy` routine in '*condinit.f90*') to avoid 'energy jumps' at the boundaries of the heating/cooling layers.

- A variable `t_heat_delay` is used to delay and progressively increase the heating and cooling of the plasma. This is again done to avoid abrupt changes in the plasma conditions. As it is defined, it should delay the start of the heating/cooling by ~1 convective timescale, and then progressively increase it to its full value.
- As for the 'mhdconv' patch, the code units of the patch are : [time] → s; [length] → Mm; [density] → g/cc. This means that quantities must be normalized when given as input and when analyzed as outputs.

The patch has two namelists. One is the usual *HeFlash.nml* we used for the paper. It is ready to use in a 2D setup, I think the *`BOUNDARY_PARAMS`* need to be adjusted for 3D. The other is *modelS.nml* which contains the parameters to initialize the simulation box with a profile approximating the solar one. More info on how this profile was generated can be found in the modelS zip file.