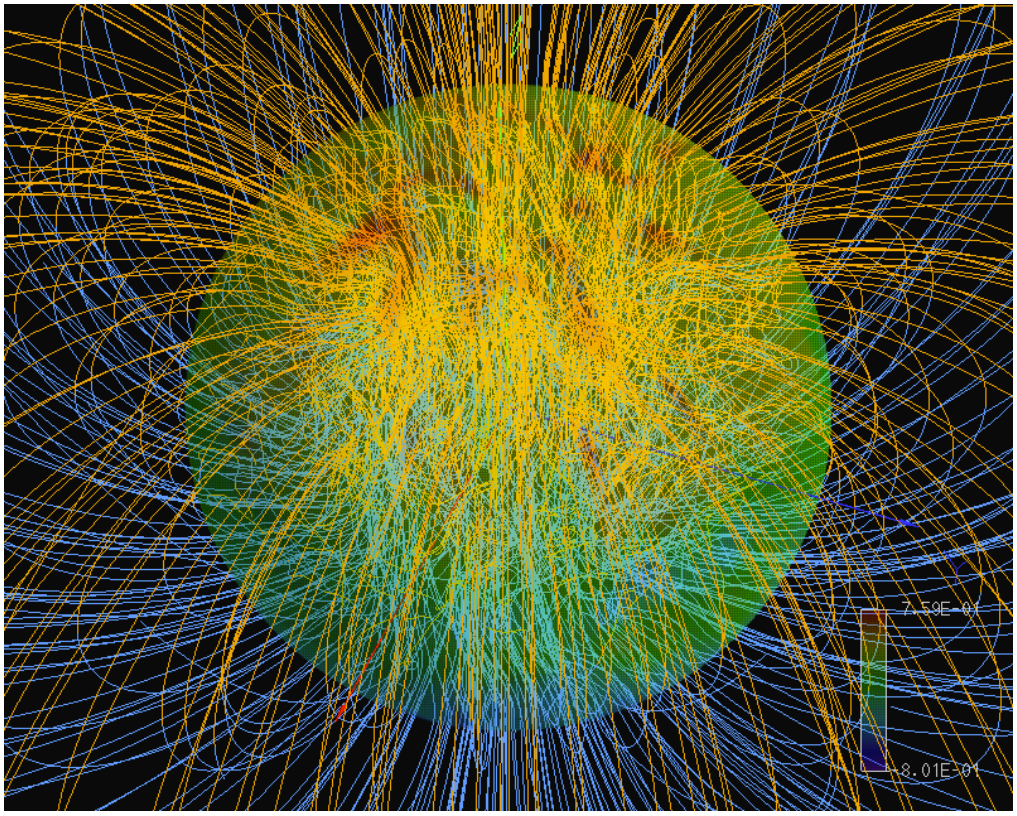
COMPUTATIONAL INFRASTRUCTURE FOR GEODYNAMICS (CIG)

Calypso

User Manual
Version 1.1



Hiroaki Matsui

www.geodynamics.org

Preface

Calypso is a program package of magnetohydrodynamics (MHD) simulations in a rotating spherical shell for geodynamo problems. This package consists of the simulation program, preprocessing program, post processing program to generate field data for visualization programs, and several small utilities. The simulation program runs on parallel computing systems using MPI and OpenMP parallelization.

Contents

1 Introduction

Calypso is a program package for magnetohydrodynamics (MHD) simulations in a rotating spherical shell for geodynamo problems. This package consists of the simulation program, preprocessing program, post processing program to generate field data for visualization programs, and several small utilities. The simulation program runs on parallel computing systems using MPI and OpenMP parallelization.

Calypso solves the equations that govern convection and magnetic-field generation in a rotating spherical shell. Flow is driven by thermal or compositional buoyancy in a Boussinesq fluid. Calypso also support various boundary conditions (e.g. fixed temperature, heat flux, composition, and compositional flux), and permits a conductive and rotatable inner core. Results are written as spherical harmonics coefficients, Gauss coefficients for the region outside of the fluid shell, and field data in Cartesian coordinate for easily visualization with a number of visualization programs.

This user guide describes the essentials of the magnetohydrodynamics theory and equations behind Calypso, and provides instructions for the configuration and execution of Calypso.

2 History

Calypso has its origins in two earlier projects. One is a dynamo simulation code written by Hiroaki Matsui in 1990's using a spectral method. This code solves for the poloidal and toroidal spectral coefficients, like Calypso, but it calculates the nonlinear terms in the spectral domain using a parallelization for SMP architectures. The other project is the thermal convection version of GeoFEM, which is Finite Element Method (FEM) platform for massively parallel computational environment, originally written by Hiroshi Okuda in 2000.

Hiroaki Matsui was responsible for adding routines to GeoFEM to perform magnetohydrodynamics simulation in a rotating frame. In 2002 this code successfully performed dynamo simulations in a rotating spherical shell using insulating magnetic boundary conditions. The following year Matsui implemented a subgrid scale (SGS) model in the FEM dynamo model in collaboration with Bruce Buffett. A module to solve for double diffusive convection was added to the FEM dynamo model by Hiroaki Matsui in 2009.

Progress in understanding the role of subgrid scale models in magnetohydrodynamic simulations relies on quantitative estimates for the transfer of energy between spatial scales. This information is most easily obtained from a spherical harmonic expansion of the simulation results, even when the simulation is performed by FEM. Hiroaki Matsui implemented the spherical harmonic transform in 2007 using a combination of MPI and

OpenMP, and later included the spherical harmonic transform routines into his old dynamo code to create Calypso. Additional software in the program package for visualization is based on data formats from the FEM model. In addition, the control parameter file format is adapted from the input formats used in GeoFEM.

Currently, Calypso Ver. 1.0 supports the following features and capabilities

- Magnetohydrodynamics simulation for a Boussinesq fluid in a rotating spherical shell.
- Convection driven by thermal and compositional buoyancy.
- Temperature or heat flux is fixed at boundaries
- Composition or compositional flux is fixed at boundaries
- Non-slip or free-slip boundary conditions
- Outside of the fluid shell is electrically insulated or pseudo vacuum boundary.
- A conductive inner core with the same conductivity as the surrounding fluid
- A rotating inner core driven by the magnetic and viscous torques.

2.1 Updates for Ver 1.1

In Version 1.1, a number of bug fixes and additional comments for Doxygen are completed. The following large bugs are fixed:

- `configure` command is updated to find appropriate GNU make command. (see Section 6.1)
- Label for radial grid type in the file `ctl_sph_shell` `raidal_grid_type_ctl` is changed to `radial_grid_type_ctl`. If the old name is used in the control file, program `gen_sph_grid` will crash.

And, the following features are implemented

- New ordering is used for spherical harmonics data to reduce communication time. The old version of spectrum indexing data, which is generated by `gen_sph_grids` in Ver. 1.0 is also supported in Ver. 1.1.

- Evaluation of Coriolis term is updated. Now, Adams-Gaunt integrals are evaluated in the initialization process in the simulation program `sph_mhd`, so the data file for Adams-Gaunt integrals which is made by `gen_sph_grids` is not required.

- Add a program `sph_add_initial_field`. to modify existed initial field data. This program is used to modify or add new fields in spectrum data. (See Section 13.)

- Heat and composition source terms are implemented. These source terms are fixed with time, and defined as spectrum data. The source terms are defined by using initial field generation program
`sph_initial_field` or `sph_add_initial_field`. (See section 12 and 13.)

- The boundary conditions for temperature and composition can be defined by using spherical harmonics coefficients. (i.e. inhomogeneous boundary conditions can be applied.) These boundary conditions are defined by using single external data file. (See Section 10.3)

3 Acknowledgements

Calypso was primarily developed by Dr. Hiroaki Matsui in collaboration with Prof. Bruce Buffett at the University of California, Berkeley. The following NSF grants supported the development of Calypso,

- B.A. Buffett, NSF EAR-0509893; Models of sub-grid scale turbulence in the Earths core and the geodynamo; 2005 - 2007.

- B.A. Buffett and D. Lathrop, NSF EAR-0652882; CSEDI Collaborative Research: Integrating numerical and experimental geodynamo models, 2007 - 2009

- B.A. Buffett, NSF EAR-1045277; Development and application of turbulence models in numerical geodynamo simulations ; 2010 - 2012

4 Citation

Computational Infrastructure for Geodynamics (CIG) and the Calypso developers are making the source code to Calypso available to researchers in the hope that it will aid their research and teaching. A number of individuals have contributed a significant amount of

time and energy into the development of Calypso. We request that you cite the appropriate papers and make acknowledgements as necessary. The Calypso development team asks that you cite the following papers:

Matsui, H., E. King, and B.A. Buffett, , Multi-scale convection in a geodynamo simulation with uniform heat flux along the outer boundary, to be submitted to *Geochemistry, Geophysics, Geosystems*.

5 Model of Simulation

5.1 Governing equations

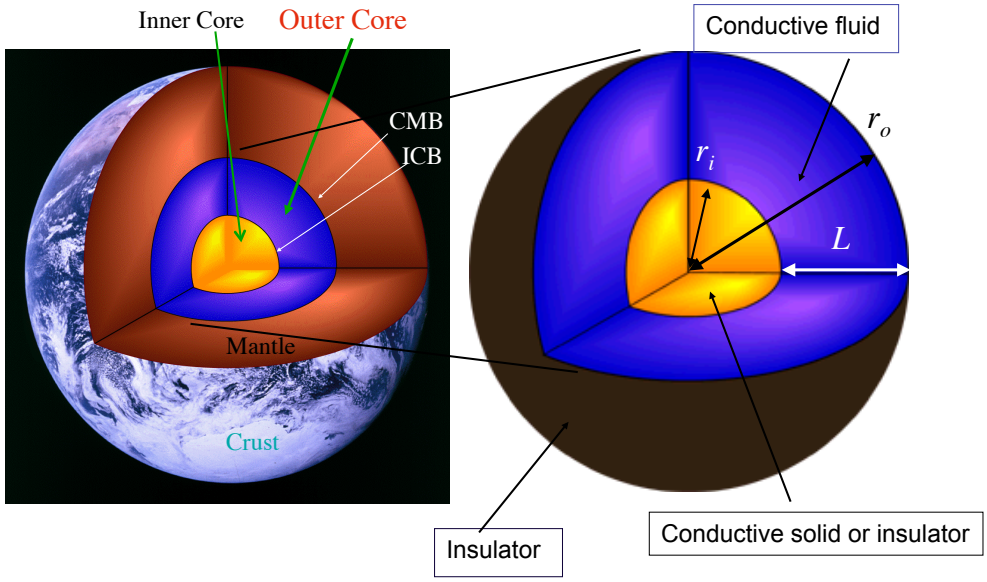


Figure 1: Rotating spherical shell modeled on the Earth's outer core.

This model performs a magnetohydrodynamics (MHD) simulation in a rotating spherical shell modeled on the Earth's outer core (see Figure 1). We consider a spherical shell from the inner core boundary (ICB) to the core mantle Boundary (CMB) in a rotating frame which constantly rotates with angular velocity $\boldsymbol{\Omega} = \Omega\hat{z}$. The fluid shell is filled with a conductive fluid with constant diffusivities (kinematic viscosity ν, magnetic diffusivity η, thermal diffusivity κ_T, and compositional diffusivity κ_C). The inner core ($0 < r < r_i$) is solid, and may be considered an electrical insulator or may have the same conductivity as the outer core. We assume that the region outside of the core is an electrical insulator. The rotating spherical shell is filled with Boussinesq modeled fluid. The governing equations of the MHD dynamo problem are the following,

$$\begin{aligned}\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{\omega} \times \boldsymbol{u}) &= -\nabla\left(P + \frac{1}{2}u^2\right) - \nu\nabla \times \nabla \times \boldsymbol{u} \\ &\quad -2\Omega\left(\hat{z} \times \boldsymbol{u}\right) + \left(\frac{\rho}{\rho_0}\boldsymbol{g}\right) + \frac{1}{\rho_0}\left(\boldsymbol{J} \times \boldsymbol{B}\right),\end{aligned}$$

$$
\begin{aligned}
\frac{\partial \boldsymbol{B}}{\partial t} &= -\eta \nabla \times \nabla \times \boldsymbol{B} + \nabla \times (\boldsymbol{u} \times \boldsymbol{B}), \\
\frac{\partial T}{\partial t} + (\boldsymbol{u} \cdot \nabla) T &= \kappa_T \nabla^2 T + q_T, \\
\frac{\partial C}{\partial t} + (\boldsymbol{u} \cdot \nabla) C &= \kappa_C \nabla^2 C + q_C, \\
\nabla \cdot \boldsymbol{u} &= \nabla \cdot \boldsymbol{B} = 0, \\
\boldsymbol{\omega} &= \nabla \times \boldsymbol{u},
\end{aligned}
$$

and

$$
\boldsymbol{J} = \frac{1}{\mu_0} \nabla \times \boldsymbol{B},
$$

where, $\boldsymbol{u}$, $\boldsymbol{\omega}$, P, $\boldsymbol{B}$, $\boldsymbol{J}$, T, C, q_T, and q_C are the velocity, vorticity, pressure, magnetic field, current density, temperature, compositional variation, heat source, and source of light element, respectively. Coefficients in the governing equations are the kinetic viscosity ν, thermal diffusivity κ_T, compositional diffusivity κ_C, and magnetic diffusivity η. The density ρ is written as a function of T, C, average density ρ_0, thermal expansion α_T, and density ratio of light element to main composition α_C,

$$
\rho = \rho_0 \left[1 - \alpha_T \left(T - T_0 \right) - \alpha_C \left(C - C_0 \right) \right]
$$

In Calypso, the vorticity equation and divergence of the momentum equation are used for solving $\boldsymbol{u}$, $\boldsymbol{\omega}$, and P as,

$$
\begin{aligned}
\frac{\partial \boldsymbol{\omega}}{\partial t} + \nabla \times (\boldsymbol{\omega} \times \boldsymbol{u}) &= -\nu \nabla \times \nabla \times \boldsymbol{\omega} - 2\Omega \nabla \times (\hat{z} \times \boldsymbol{u}) \\
&\quad + \nabla \times \left(\frac{\rho}{\rho_0} \boldsymbol{g} \right) + \frac{1}{\rho_0} \nabla \times (\boldsymbol{J} \times \boldsymbol{B}),
\end{aligned}
$$

and

$$
\begin{aligned}
\nabla \cdot (\boldsymbol{\omega} \times \boldsymbol{u}) &= -\nabla^2 \left(P + \frac{1}{2} u^2 \right) - 2\Omega \nabla \cdot (\hat{z} \times \boldsymbol{u}) \\
&\quad + \nabla \cdot \left(\frac{\rho}{\rho_0} \boldsymbol{g} \right) + \frac{1}{\rho_0} \nabla \cdot (\boldsymbol{J} \times \boldsymbol{B}).
\end{aligned}
$$

5.2 Spherical harmonics expansion

In Calypso, fields are expanded into spherical harmonics. A scalar field (for example, temperature $T(r, \theta, \phi)$) is expanded as

$$T(r,\theta,\phi) = \sum_{l=0}^{L}\sum_{m=-l}^{l} T_l^m(r) Y_l^m(\theta,\phi),$$

where Y_l^m are the spherical harmonics. Solenoidal fields (e.g. velocity $\boldsymbol{u}$, vorticity $\boldsymbol{\omega}$, magnetic field $\boldsymbol{B}$, and current density $\boldsymbol{J}$) are decomposed into poloidal and toroidal components. For example, the magnetic field is described as

$$\boldsymbol{B}(r,\theta,\phi) = \sum_{l=1}^{L}\sum_{m=-l}^{l} \left(\boldsymbol{B}_{Sl}^{\ m} + \boldsymbol{B}_{Tl}^{\ m}\right),$$

where

$$\begin{aligned}\boldsymbol{B}_{Sl}^{\ m}(r,\theta,\phi) &= \nabla\times\nabla\times\left(B_{Sl}^{\ m}(r) Y_l^m(\theta,\phi)\hat{r}\right),\\ \boldsymbol{B}_{Tl}^{\ m}(r,\theta,\phi) &= \nabla\times\left(B_{Tl}^{\ m}(r) Y_l^m(\theta,\phi)\hat{r}\right).\end{aligned}$$

The spherical harmonics are defined as real functions. $P_l^m\cos(m\phi)$ is assigned for positive m, $P_l^m\sin(m\phi)$ is assigned for negative m, where P_l^m are Legendre polynomials. Because Schmidt quasi normalization is used for the Legendre polynomials P_l^m, the orthogonality relation for the spherical harmonics is

$$\int Y_l^m Y_{l'}^{m'} \sin\theta d\theta d\phi = 4\pi\frac{1}{2l+1}\delta_{ll'}\delta_{mm'},$$

where, $\delta_{ll'}$ is Kronecker delta.

5.3 Evaluation of Coriolis term

The curl of the Coriolis force $-2\Omega\nabla\times(\hat{z}\times\boldsymbol{u})$ is evaluated in the spectrum space using the triple products of the spherical harmonics. These 3j-symbols (or Gaunt integral $G_{Lll'}^{Mmm'}$ and Elsasser integral $E_{Lll'}^{Mmm'}$) are written as

$$\begin{aligned}G_{Lll'}^{Mmm'} &= \int Y_L^M Y_l^m Y_{l'}^{m'} \sin\theta d\theta d\phi,\\ E_{Lll'}^{Mmm'} &= \int Y_L^M \left(\frac{\partial Y_l^m}{\partial\theta}\frac{\partial Y_{l'}^{m'}}{\partial\phi} - \frac{\partial Y_l^m}{\partial\phi}\frac{\partial Y_{l'}^{m'}}{\partial\theta}\right) d\theta d\phi.\end{aligned}$$

The Gaunt integral $1/(4\pi)G_{Lll'}^{Mmm'}$ and Elsasser integral $1/(4\pi)E_{Lll'}^{Mmm'}$ for the Coriolis terms are evaluated in the simulation program.

5.4 Boundary conditions

Calypso currently supports the following boundary conditions for velocity $\boldsymbol{u}$, magnetic field $\boldsymbol{B}$, temperature T, and composition variation C. These boundary conditions are defined in the control file `control_MHD`.

5.4.1 Non-slip boundary

The velocity $\boldsymbol{u}$ is set to be 0 at the boundary. For poloidal and toroidal coefficients of velocity, $U_{Sl}^{m}(r)$ and $U_{Tl}^{m}(r)$, the boundary condition can be described as

$$U_{Sl}^{m}(r) = \frac{\partial U_{Sl}^{m}}{\partial r} = 0,$$

and

$$U_{Tl}^{m}(r) = 0.$$

5.4.2 Free-slip boundary

For a free slip boundary, shear stress and radial flow vanish at the boundary. The boundary condition for poloidal and toroidal coefficients are described as

$$U_{Sl}^{m}(r) = \frac{\partial^2}{\partial r^2}\left(\frac{1}{r}U_{Sl}^{m}(r)\right) = 0,$$

and

$$\frac{\partial}{\partial r}\left(\frac{1}{r^2}U_{Tl}^{m}(r)\right) = 0.$$

5.4.3 Fixed rotation rate

If the boundary rotates with a rotation vector $\boldsymbol{\Omega}_b = (\Omega_{bx}, \Omega_{by}, \Omega_{bz})$, the boundary conditions for poloidal and toroidal coefficients are described as

$$\begin{aligned} U_{Sl}^{m}(r) &= \frac{\partial U_{Sl}^{m}}{\partial r} = 0, \\ U_{T1}^{1s}(r) &= r^2\Omega_{by}, \\ U_{T1}^{0}(r) &= r^2\Omega_{bz}, \\ U_{T1}^{1c}(r) &= r^2\Omega_{bx}, \end{aligned}$$

and

$$U_{Tl}^{m}(r) = 0 \text{ for } l > 2.$$

5.4.4 Fixed homogenous temperature

When a constant temperature T_b is is applied, the spherical harmonic coefficients are

$$T_0^0(r) \;=\; T_b,$$

and

$$T_l^m(r) \;=\; 0 \text{ for } l > 1.$$

5.4.5 Fixed homogenous heat flux

A constant heat flux is imposed by setting the radial temperature gradient to F_{Tb}. The spherical harmonic coefficients are

$$\frac{\partial T_0^0}{\partial r} \;=\; F_{Tb},$$

and

$$\frac{\partial T_l^m}{\partial r} \;=\; 0 \text{ for } l > 1.$$

5.4.6 Fixed composition

When a constant composition C_b is applied, the spherical harmonic coefficients are

$$C_0^0(r) \;=\; C_b,$$

and

$$C_l^m(r) \;=\; 0 \text{ for } l > 1.$$

5.4.7 Fixed composition flux

A constant composition flux is imposed by setting the radial composition gradient to F_{Cb}. The spherical harmonic coefficients are

$$\frac{\partial C_0^0}{\partial r} \;=\; F_{Cb},$$

and

$$\frac{\partial C_l^m}{\partial r} \;=\; 0 \text{ for } l > 1.$$

5.4.8 Connection to the magnetic potential field

If the regions outside the fluid shell are assumed to be electrical insulators, current density vanishes in the electric insulator

$$\boldsymbol{J}_{ext} = 0,$$

where the suffix $_{ext}$ indicates fields outside of the fluid shell. At the boundaries of the fluid shell, the magnetic field $\boldsymbol{B}_{fluid}$, current density $\boldsymbol{J}_{fluid}$, and electric field $\boldsymbol{E}_{fluid}$ in the conductive fluid satisfy:

$$(\boldsymbol{B}_{fluid} - \boldsymbol{B}_{ext}) = 0,$$
$$(\boldsymbol{J}_{fluid} - \boldsymbol{J}_{ext}) \cdot \hat{r} = 0,$$

and

$$(\boldsymbol{E}_{fluid} - \boldsymbol{E}_{ext}) \times \hat{r} = 0,$$

where, $\hat{r}$ is the radial unit vector (i.e. normal vector for the spherical shell boundaries). Consequently, radial current density $\boldsymbol{J}$ vanishes at the boundary as

$$\boldsymbol{J} \cdot \hat{r} = 0 \text{ at } r = r_i, r_o$$

In an electrical insulator the magnetic field can be described as a potential field

$$\boldsymbol{B}_{ext} = -\nabla W_{ext},$$

where W_{ext} is the magnetic potential. The boundary conditions can be satisfied by connecting the magnetic field in the fluid shell at boundaries to the potential fields. The magnetic field is connected to the potential field in an electrical insulator. At CMB ($r = r_o$), the boundary condition can be described by the poloidal and toroidal coefficients of the magnetic field as

$$\frac{l}{r} B_{Sl}^{m}(r) = -\frac{\partial B_{Sl}^{m}}{\partial r},$$

and

$$B_{Tl}^{m}(r) = 0.$$

If the inner core is also assumed to be an insulator, the magnetic boundary conditions for ICB ($r = r_i$) can be described as

$$\frac{l+1}{r} B_{Sl}^{m}(r) = \frac{\partial B_{Sl}^{m}}{\partial r},$$

and

$$B_{Tl}^{m}(r) = 0.$$

5.4.9 Magnetic boundary condition for center

If the inner core has the same conductivity as the outer core, we solve the induction equation for the inner core as for the outer core with the boundary conditions for the center. The poloidal and toroidal coefficients at center are set to

$$B_{Sl}^{m}(0) \quad = \quad B_{Tl}^{m}(0) = 0.$$

5.4.10 Pseudo-vacuum magnetic boundary condition

Under the pseudo-vacuum boundary condition, the magnetic field has only a radial component at the boundaries. Considering the conservation of the magnetic field, the magnetic boundary condition will be

$$\frac{\partial}{\partial r}\left(r^2 B_r\right) = B_\theta = B_\phi = 0 \text{ at } r = r_i, r_o.$$

The present boundary condition is also described by using the poloidal and toroidal coefficients as

$$\frac{\partial B_{Sl}^{m}}{\partial r} \quad = \quad B_{Tl}^{m}(r) = 0 \text{ at } r = r_i, r_o.$$

6 Installation

6.1 Library Requirements

Calypso requires the following libraries.

- GNU make
- MPI libraries (OpenMPI, MPICH, etc)
- FFTPACK Ver 5.1D (`http://people.sc.fsu.edu/~jburkardt/f_src/fftpack5.1d/fftpack5.1d.html`). The source files for FFTPACK are included in `src/EXTERNAL_libs` directory.

Linux and Max OS X use GNU make as a default 'make' command, but some system (e.g. BSD or SOLARIS) does not use GNU make as default. `configure` command searches and set correct GNU make command.

In addition, the following libraries can be used (optional).

- OpenMP
- FFTW version 3 (`http://www.fftw.org`) including Fortran wrapper
- PARALLEL HDF5 (`http://www.hdfgroup.org/HDF5/PHDF5`) including Fortran wrapper.

Note: Calypso does NOT use MPI and OpenMP features in FFTW3.

In the most of platforms, the Fourier transform is faster than that by FFTPACK.

HDF5 is used for field data output with XDMF format instead of VTK format. The comparison of field data format is described in section refsec:VTK.

OpenMP is used for the parallelization under the shared memory. Better choice to use both MPI and OpenMP parallelization (so-called Hybrid parallelization) or only using MPI (so-called flat MPI) is depends on the computational platform and compiler. For example, flat MPI has much better performance on Linux cluster with Intel Xeon processors and with Intel fortran compiler, but Hybrid model has better performance on Hitachi SR16000 with Power 6 processors.

6.2 Known problems

FFTPACK and Intel compiler

FFTPACK fails to compile with Intel fortran using the `'-warn all'` option. Currently the `'-warn all'` option is excluded by Makefile when FFTPACK is compiled.

Homebrew's FFTW3 on Mac OS X

Calypso uses Fortran wrappers in FFTW3. If FFTW3 is installed using Homebrew for Mac OS X (`http://mxcl.github.com/homebrew/`), the required fortran wrappers are not installed. In this case, please install FFTW3 with Fortran wrappers with another package manager (Macports (`http://www.macports.org`, for example), build FFTW3 by yourself including the Fortran wrapper, or turn off FFTW3 features in Calypso.

Cross compiler support

`configure` command in Calypso does not support cross compilation. If you want to compile with a cross compiler, please set the variables in Makefile manually (see section 6.6)

6.3 Directories

The top directory of Calypso (ex. `[CALYPSO_HOME]`) contains the following directories.

```
% cd [CALYPSO_HOME]
% ls
CMakeLists.txt Makefile.in configure.in examples
INSTALL bin doc src
LICENSE configure doxygen work
```

`bin`: directory for executable files

`cmake`: directory for cmake configurations

`cmake`: directory for document generated by doxygen

`doc`: documentations

`examples`: examples

`src`: source files

`work`: work directory. Compile is done in this directory.

6.4 Doxygen

Doxygen (`http://www.doxygen.org`) is an powerful document generation tool from source files. We only save a configuration file in this directory because thousands of html files generated by doxygen. The documents for source codes are generated by the following command:

```
% cd [CALYPSO_HOME]/doxygen
% doxygen ./Doxyfile_CALYPSO
```

The html documents can see by opening `[CALYPSO_HOME]/doxygen/html/index.html`. Automatically generated documentation is also available on the CIG website at `http://www.geodynamics.org/cig/software/calypso/`.

6.5 Install using `configure` command

6.5.1 Configuration using `configure` command

Calypso uses the configure script for configuration to install. The simplest way to install programs is the following process in the top directory of Calypso.

```
%pwd
[CALYPSO_HOME]
% ./configure
...
% make
...
% make install
```

After the installation, object modules can be deleted by the following command;

```
% make clean
```

./configure generates a Makefile in the current directory. Available options for `configure` can be checked using the `./configure --help` command. The following options are available in the `configure` command.

```
Optional Features:
  --disable-option-checking  ignore unrecognized --enable/--with options
  --disable-FEATURE       do not include FEATURE (same as --enable-FEATURE=no)
  --enable-FEATURE[=ARG]  include FEATURE [ARG=yes]
  --enable-fftw3          Use fftw3 library
```

```
 Optional Packages:
  --with-PACKAGE[=ARG]    use PACKAGE [ARG=yes]
  --without-PACKAGE       do not use PACKAGE (same as --with-PACKAGE=no)
  --with-hdf5=yes/no/PATH full path of h5pcc for parallel HDF5 configuration

Some influential environment variables:
  CC          C compiler command
  CFLAGS      C compiler flags
  LDFLAGS     linker flags, e.g. -L<lib dir> if you have libraries in a
              nonstandard directory <lib dir>
  LIBS        libraries to pass to the linker, e.g. -l<library>
  CPPFLAGS    (Objective) C/C++ preprocessor flags, e.g. -I<include dir> if
              you have headers in a nonstandard directory <include dir>
  FC          Fortran compiler command
  FCFLAGS     Fortran compiler flags
  MPICC       MPI C compiler command
  MPIFC       MPI Fortran compiler command
  PKG_CONFIG  path to pkg-config utility
  CPP         C preprocessor
  FFTW3_CFLAGS
              C compiler flags for FFTW3, overriding pkg-config
  FFTW3_LIBS  linker flags for FFTW3, overriding pkg-config
```

An example of usage of the configure command is the following;

```
% ./configure --prefix='/Users/matsui/local' \
? CFLAGS='-O -Wall -g' FCFLAGS='-O -Wall -g' \
? PKG_CONFIG_PATH='/Users/matsui/local/lib/pkgconfig' \
? --enable-fftw3 --with-hdf5='/Users/matsui/local/bin/h5pcc'
```

6.5.2 Compile

Compile is performed using the `make` command. The Makefile in the top directory is used to generate another Makefile in the `work` directory, which is automatically used to complete the compilation. The object file and libraries are compiled in the `work` directory. Finally, the executive files are assembled in `bin` directory. You should find the following programs in the `bin` directory.

`gen_sph_grids`: Preprocessing program for data transfer for spherical transform

`sph_mhd`: Simulation program

`sph_initial_field`: Example program to generate initial field

`sph_add_initial_field`: Example program to add initial field in existing spectum data

`sph_snapshot`: Data transfer from spectrum data to field data

`sph_dynamobench`: Data processing for dynamo benchmark test by Christensen *et. al.* (2002)

`sph_zm_snapshot`: Generate zonal mean field

`assemble_sph`: Data transfer program to change number of subdomains.

`make_f90depends`: Program to generate dependency of the source code (`make` command uses to generate `work/Makefile`)

The following library files are also made in `work` directory.

`libcalypso.a`: Calypso library

`libfftpack.5d.a`: FFTPACK 5.1 library

6.5.3 Clean

The object and fortran module files in `work` directory is deleted by typing

```
% make clean
```

This command deletes files with the extension `.o`, `.mod`, `.par`, `.diag`, and .

6.5.4 Install

The executive files are copied to the install directory `$(INSTDIR)/bin`. The install directory `$(INSTDIR)` is defined in Makefile, and can also set by `${--prefix}` option for `configure` command. Alternatively, you can use the programs in `${SRCDIR}/bin` directory without running `make install`. If directory `${PREFIX}` does not exist, `make install` creates `${PREFIX}`, `${PREFIX}/lib`, `${PREFIX}/bin`, and `${PREFIX}/include` directories. No files are installed in `${PREFIX}/lib` and `${PREFIX}/include`.

6.6 Install without using configure

It is possible to compile Calypso without using the `configure` command. To do this, you need to edit the `Makefile`. First, copy `Makefile` from template `Makefile.in` as

```
% cp Makefile.in Makefile
```

In Makefile, the following variables should be defined.

`SHELL` Name of shell command.

`SRCDIR` Directory of this Makefile.

`INSTDIR` Install directory.

`MPICHDIR` Directory names for MPI implementation. If you set fortran90 compiler name for MPI programs in `MPIF90`, you do not need to define this valuable.

`MPICHINCDIR` Directory names for include files for MPI implementation. If you set fortran90 compiler name for MPI programs in `MPIF90`, you do not need to define this valuable.

`MPILIBS` Library names for MPI implementation. If you set fortran90 compiler name for MPI programs in `MPIF90`, you do not need to define this valuable.

`F90_LOCAL` Command name of local Fortran 90 compiler to compile module dependency listing program.

`MPIF90` Command name of Fortran90 compiler and linker for MPI programs. If command does not have MPI implementation, you need to define the definition of MPI libraries `MPICHDIR`, `MPICHINCDIR`, and `MPILIBS`.

`AR` Command name for archive program (ex. `ar`) to generate libraries. If you need some options for archive command, options are also included in this valuable.

`RANLIB` Command name for `ranlib` to generate index to the contents of an archive. If system does not have `ranlib`, set `true` in this valuable. `true` command does not do anything for libraries.

`F90OPTFLAGS` Optimization flags for Fortran90 compiler (including OpenMP flags)

`FFTW3_CFLAGS` Option flags for FFTW3 (ex. `-I/usr/local/include`)

`FFTW3_LIBS` Library lists for FFTW3 (ex. `-L/usr/local/lib -lfftw3 -lm`)

`HDF5_FFLAGS` Option flags to compile with HDF5. This setting can be found by using hfd5 command `h5pfc -show`.

`HDF5_LDFLAGS` Option flags to link with HDF5. This setting can be found by using hfd5 command `h5pfc -show`.

`HDF5_FLIBS` Library lists for HDF5. This setting can be found by using hfd5 command `h5pfc -show`.

6.7 Install using cmake

CMake is a cross-platform, open-source build system. CMake can be downloaded from `http://www.cmake.org`. The following procedure is required to install.

1. Create working directory (you can also use `[CALYPSO_HOME]/work`).
2. Generate Makefile and working directories by `cmake` command.
3. Compile programs by `make` command.

In this section, `[CALYPSO\_HOME]/work` is used as the working directory. Options for CMake can be checked by `cmake -i [CALYPSO_HOME]` command at `[CALYPSO_HOME] /work`. There are a number of options can be found, but the following valuables are important settings for installation:

`CMAKE_INSTALL_PREFIX` Install directory

`CMAKE_Fortran_COMPILER` Fortran90 compiler.

`CMAKE_DISABLE_FIND_PACKAGE_OpenMP_Fortran` OpenMP is not used if 'yes' is set in this valuable.

`CMAKE_DISABLE_FIND_PACKAGE_FFTW` FFTW3 library does not linked if 'yes' is set in this valuable.

`CMAKE_LIBRARY_PATH` CMake library search paths. This directory is used to search FFTW3 library.

`CMAKE_INCLUDE_PATH` CMake include search paths. This directory is used to search include file for FFTW3.

`CMAKE_DISABLE_FIND_PACKAGE_FFTW` FFTW3 library does not linked if 'yes' is set in this valuable.

`HDF5_INCLUDE_DIRS` Include file directories to compile with HDF5. This setting can be found by using hfd5 command `h5pfc -show`.

`HDF5_LIBRARY_DIRS` Location of HDF5 library. This setting can be found by using hfd5 command `h5pfc -show`.

`HDF5_LIBRARIES` Library lists for HDF5. This setting can be found by using hfd5 command `h5pfc -show`.

`CMAKE_DISABLE_FIND_PACKAGE_HDF5` HDF5 library does not linked if 'yes' is set in this valuable.

An example of using CMake on Mac OS X is the following:

```
% cd work
% h5pfc -show
mpif90 -I/home/matsui/local/include -L/home/matsui/local/lib
 /home/matsui/local/lib/libhdf5hl_fortran.a
 /home/matsui/local/lib/libhdf5_hl.a
 /home/matsui/local/lib/libhdf5_fortran.a
 /home/matsui/local/lib/libhdf5.a
  -L/home/matsui/local/lib -lmpi -lz -ldl -lm
% cmake .. -DCMAKE_LIBRARY_PATH='/home/matsui/local/lib' \
? -DCMAKE_INCLUDE_PATH='/home/matsui/local/include' \
? -DHDF5_INCLUDE_DIRS='/home/matsui/local/include' \
? -DHDF5_LIBRARY_DIRS='/home/matsui/local/lib' \
? -DHDF5_LIBRARIES='/home/matsui/local/lib/libhdf5hl_fortran.a \
? /home/matsui/local/lib/libhdf5_hl.a \
? /home/matsui/local/lib/libhdf5_fortran.a \
? /home/matsui/local/lib/libhdf5.a'
```

After configuration, compile and install are started by

```
% make
...
% make install
```

After running `make` command, execute files are built in `[CALYPSO_HOME]/work/bin` directory.

7 Simulation procedure

Calypso consists of programs shown in Table 1. Because the serial programs do not use

Table 1: List of program and required control file name

Program	Control file name	Type
gen_sph_grids	control_sph_shell	Serial
sph_mhd	control_MHD	Parallel
sph_initial_field	control_MHD	Parallel
sph_add_initial_field	control_MHD	Parallel
sph_snapshot	control_snapshot	Parallel
sph_dynamobench	control_snapshot	Parallel
sph_zm_snapshot	control_snapshot	Parallel
assemble_sph	control_sph_assemble	Serial

MPI, they are simply invoked by

```
% [program]
```

Parallel programs must be invoked using MPI commands. On a Linux cluster using MPICH, parallel programs are invoked with

```
% mpirun -np [# of processes] [program]
```

This command will vary depending on the MPI implementation installed on the machine. Please consult with your sysadmin for details.

To perform simulations by Calypso, the following processes are required.

1. Generate grids and spherical harmonics indexing information by `gen_sph_grids`.
2. Make initial fields by `sph_initial_field` (if necessary).
3. Perform the simulation by `sph_mhd`.

4. Convert the parallel spectra data by `assemble_sph` to continue with changing number of processes (if necessary).

5. Data analysis by `sph_snapshot`, `sph_snapshot`, or `sph_dynamobench`.

6. Update initial fields by `sph_add_initial_field` for more simulations (if necessary).

The simulation program `sph_mhd` requires an indexing file for spherical transform. `sph_mhd` generates spectrum data and monitoring data, and field data in Cartesian coordinate as outputs. The data transform programs (`sph_snapshot` and `sph_zm_snapshot`) generate outputs data from parallel spectra data. The flow of data is shown in Figure 2.

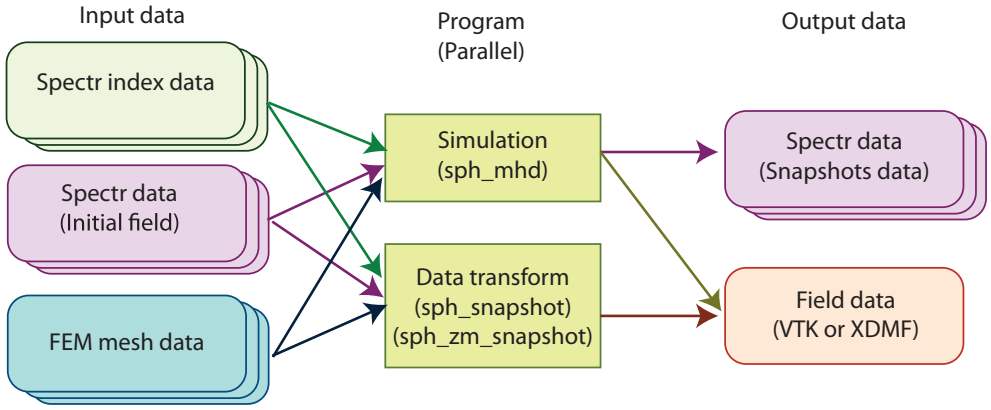


Figure 2: Data flow of the simulation. Simulations require index data for spherical harmonics transform, initial spectra (optional) data, and FEM mesh data. Simulation program also outputs spectra data, monitoring data and field data in Cartesian coordinate. Data transform program generates output data for simulation program from spectra data.

Each program needs one control file, the name of which is defined by the program. (Standard input is not supported by Fortran 90 so Calypso uses control files.) The appropriate control file names are shown in the Table 1. The following rules are used in the control files. An example of a control file is shown in Figure 3.

- Lines starting with '`#`' or '`!`' are treated as a comment lines and ignored.
- All control files consist of blocks which start with '`begin [name]`' and end with '`end [name]`'.

- The item name is shown first and the associated value/data is second.
- The order of items and blocks can be changed.
- If an item consists of multiple data, these should be listed in one line.
- If an item does not belong in the block it is ignored.
- An array block starts with '`begin array [name] [number of components]`' and ends with '`end array [name]`'.
- If `[number of components]` for an array is 0, '`end array [name]`' on the next line is not needed.
- In Fortran program, character '`/`' is recognized as an end of character valuable if text with '`/`' (*e.g.* file prefix including file paths) is not enclosed by `'` or `"`.
- Calypso's control file input is limited to 255 characters for each line.

8 Examples

Several examples are provided in the `examples` directory. There are three subdirectories as examples. README files are also provided to perform these examples in each subdirectory.

`assemble_sph` Examples for assembling program of spectrum data. (see section 11)

`dynamo_benchmark` Examples for dynamo benchmark by Christensen *et. al.* (2001)

`spherical_shell` Examples for preprocessing program (see Section 9)

8.1 Examples for preprocessing program

Four examples illustrate the use of the preprocessing program. The examples include

`Chebyshev_points` Example to generate indexing data using Chebyshev collocation points

`equidistance` Example to generate indexing data with equi-distance grid

`explicitly_defined` Example to generate indexing data with explicitly defined radial points

```
begin spherical_shell_ctl
!
  begin data_files_def
    num_subdomain_ctl    4
!
    sph_file_prefix             'sph_shell/in'
  end data_files_def
!
  begin num_grid_sph
    truncation_level_ctl     4
    ngrid_meridonal_ctl     12
    ngrid_zonal_ctl         24
!
    radial_grid_type_ctl   explicit
    array r_layer       4
      r_layer    1  0.5384615384615
      r_layer    2  0.5384615384615
      r_layer    3  1.038461538462
      r_layer    4  1.538461538462
    end array r_layer
!
  end num_grid_sph
end spherical_shell_ctl
```

Figure 3: Example of Control file

`with_inner_core` Example to generate indexing data including inner core and external of the fluid shell.

8.2 Examples of dynamo benchmark

There are four examples for simulations using dynamo benchmark test as following.

`Case_0` Example of dynamo benchmark case 0 (Thermally driven convection without magnetic field)

`Case_1` Example of dynamo benchmark case 1 (Dynamo model with co-rotating and electrically insulated inner core)

`Case_2` Example of dynamo benchmark case 2 (Dynamo model with rotatable and conductive inner core)

`Compositional_case_1` Example of dynamo benchmark case 1 using compositional variation instead of temperature

The process of the simulation is as following:

1. Change to the directory for Benchmark Case 1 (for example)

   ```
   [username]$ cd [CALYPSO_DIR]/examples/dynamo_benchmark/dynamobench_case1
   ```

2. Create the grid files for the simulation

   ```
   [dynamobench_case_1]$ [CALYPSO_DIR]/bin/gen_sph_grids
   ```

3. Run simulation program

   ```
   [dynamobench_case_1]$ mpirun -np 4 [CALYPSO_DIR]/bin/sph_mhd
   ```

4. To continue the simulation, change the parameter `rst_ctl` in `control_MHD` from `dynamo_benchmark_1` to `start_from_rst_file` and continue simulation by repeating step 2.

5. To check the results for dynamo benchmark, run

   ```
   [dynamobench_case_1]$ mpirun -np 4 [CALYPSO_DIR]/bin/sph_dynamobench
   ```

8.3 Example of data assembling program

An example for spectrum data assembling program is provided in `assemble_sph` directory.

8.4 Example of heat and compositional source

An example to perform a simulation with heat and compositional sources is given in `heat_composition_source` directory. To simplify the problem, only the thermal and compositional fields are evolved with no velocity (i.e. pure diffusion problem). A module to generate initial field data
`const_sph_initial_spectr` is copied to `src/programs/data_utilities/INITIAL_FIELD/` directory. The code must be recompiled after modifying this module. Initial field is generated by the program `sph_initial_field` after generating spherical harmonics information by `gen_sph_grid`. Aftrer the simulation, Y_0^0 component of temperature and composition as a function of radius and time is written in `picked_mode.dat`.

8.5 Example of thermal and compositional boundary conditions by external file

Heterogeneous boundary are input using an external file. An example to set thermal and compositional boundary conditions is given in `heterogineous_temp` directory. As in the heat source example, only the diffusion problem is solved in this example. In file `bc_spectr.btx`, temperature boundary conditions are defined for Y_0^0, Y_1^{1s}, Y_1^{1c}, and ,Y_2^{2c} component, and compositional boundary is defined for Y_0^0, Y_2^{2s}, and Y_2^{2c} components. The radial profile of these spherical harmonics coefficients are written in `picked_mode.dat`.

9 Preprocessing program (gen_sph_grid)

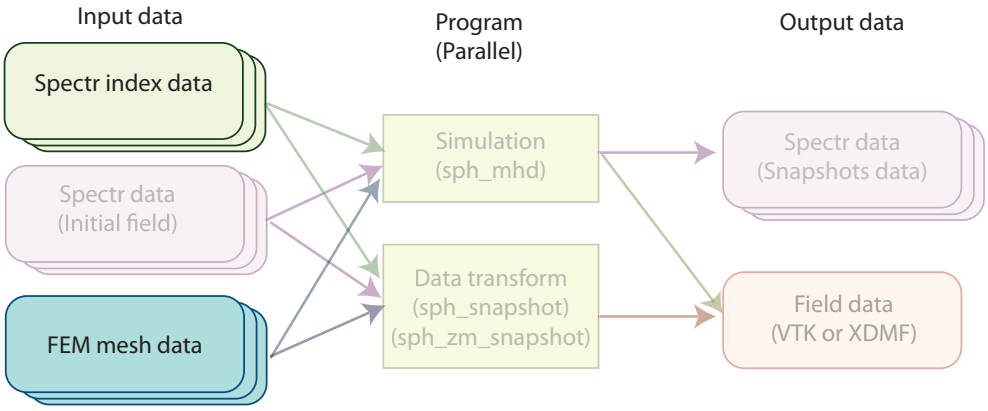


Figure 4: Generated files by preprocessing program in Data flow.

This program generates index table and a communication table for parallel spherical harmonics, table of integrals for Coriolis term, and FEM mesh information to generate visualization data (see Figure 4). This program needs control file for input. The output files include the indexing tables.

Table 2: List of files for gen_sph_grid

extension	Parallelization	I/O
control_sph_grid	Single	Input
[sph_prefix].[domain#].rj	Distributed	Output
[sph_prefix].[domain#].rlm	Distributed	Output
[sph_prefix].[domain#].rtm	Distributed	Output
[sph_prefix].[domain#].rtp	Distributed	Output
[sph_prefix].[domain#].gfm	Distributed	Output
radial_info.dat	Single	Output

9.1 Position of radial grid

The preprocessing program sets the radial grid spacing, either by a list in the control file or by setting an equidistant grid or Chebyshev collocation points.

In equidistance grid, radial grids are defined by

$$r(k) \;=\; r_i + (r_o - r_i)\,\frac{k - k_{ICB}}{N},$$

where, k_{ICB} is the grid points number at ICB. The radial grid set from the closest points of minimum radius defined by `[Min_radius_ctl]` in control file to the closest points of the maximum radius defined by `[Max_radius_ctl]` in control file, and radial grid number for the innermost points is set to $k = 1$.

In Chebyshev collocation points, radial grids in the fluid shell are defined by

$$r(k) \;=\; r_i + \frac{(r_o - r_i)}{2}\left[\frac{1}{2} - \cos\left(\pi\frac{k - k_{ICB}}{N}\right)\right],$$

For the inner core ($r < r_i$), grid points is defined by

$$r(k) \;=\; r_i - \frac{(r_o - r_i)}{2}\left[\frac{1}{2} - \cos\left(\pi\frac{k - k_{ICB}}{N}\right)\right],$$

and, grid points in the external of the shell ($r > r_o$) is defined by

$$r(k) \;=\; r_o + \frac{(r_o - r_i)}{2}\left[\frac{1}{2} - \cos\left(\pi\frac{k - k_{CMB}}{N}\right)\right],$$

where, k_{CMB} is the grid point number at CMB.

9.2 Control file (`control_sph_shell`)

Control file (`control_sph_shell`) consists the following items. Detailed description for each item can be checked by clicking "(Detail)" at the end of each item.
`spherical_shell_ctl`

- `data_files_def` (Detail)
  - `num_subdomain_ctl` `[Num_PE]` (Detail)
  - `sph_file_prefix` `[sph_prefix]` (Detail)
- `num_domain_ctl` (Detail)
  - `array num_domain_sph_grid [Direction] [Ndomain]` (Detail)

  - array num_domain_legendre [Direction] [Ndomain] (Detail)
  - array num_domain_spectr [Direction] [Ndomain] (Detail)

- num_grid_sph (Detail)
  - truncation_level_ctl [Lmax] (Detail)
  - ngrid_meridonal_ctl [Ntheta] (Detail)
  - ngrid_zonal_ctl [Nphi] (Detail)
  - radial_grid_type_ctl [explicit, Chebyshev, or equi_distance] (Detail)
  - num_fluid_grid_ctl [Nr_shell] (Detail)
  - fluid_core_size_ctl [Length] (Detail)
  - ICB_to_CMB_ratio_ctl [R_ratio] (Detail)
  - Min_radius_ctl [Rmin] (Detail)
  - Max_radius_ctl [Rmax] (Detail)

  - array r_layer [Layer #] [Radius] (Detail)
  - array boundaries_ctl [Boundary_name] [Layer #] (Detail)

9.3 Spectrum index data

gen_sph_grid generates indexing table of the spherical transform. To perform spherical harmonics transform with distributed memory computers, data communication table is also included in these files. Calypso needs four indexing data for the spherical transform.

[sph_prefix].[domain#].rj Indexing table for spectrum data $f(r,l,m)$ to calculate linear terms. In program, spherical harmonics modes (l,m) is indexed by $j = l(l+1)+m$. The spectrum data are decomposed by spherical harmonics modes j. Data communication table for Legendre transform is included. The data also have the radial index of the ICB and CMB.

[sph_prefix].[domain#].rlm Indexing table for spectrum data $f(r,l,m)$ for Legendre transform. The spectrum data are decomposed by radial direction r and

spherical harmonics order m. Data communication table to caricurate liner terms is included.

`[sph_prefix].[domain#].rtm` Indexing table for data $f(r, \theta, m)$ for Legendre transform. The data are decomposed by radial direction r and spherical harmonics order m. Data communication table for backward Fourier transform is included.

`[sph_prefix].[domain#].rtp` Indexing table for data $f(r, \theta, m)$ for Fourier transform and field data $f(r, \theta, \phi)$. The data are decomposed by radial direction r and meridional direction θ. Data communication table for forward Legendre transform is included.

9.4 Finite element mesh data

Calypso generates field data for visualization with XDMF or VTK format. To generate field data file, the preprocessing program generates FEM mesh data for each subdomain of spherical grid (r, θ, ϕ) under the Cartesian coordinate (x, y, z). The mesh data file is written as GeoFEM (`http://geofem.tokyo.rist.or.jp`) mesh data format, which consists of each subdomain mesh and communication table among overlapped nodes.

9.5 Radial grid data

The preprocessing program generates radius of each layer in `radial_info.dat` if `radial_grid_type_ctl` is set to `Chebyshev` or `equi_distance`. This file consists of blocks `array r_layer` and `array boundaries_ctl` for control file. This data may be useful if you want to modify radial grid spacing by yourself.

10 Simulation program (sph_mhd)

The name of the simulation program is sph_mhd. This program requires control_MHD as a Control file. This program performs with the indexing file for spherical harmonics and Coriolis term integration file generated by the preprocessing program gen_sph_grid. Data files for this program are listed in Table 3. Indexing data for spherical harmonics

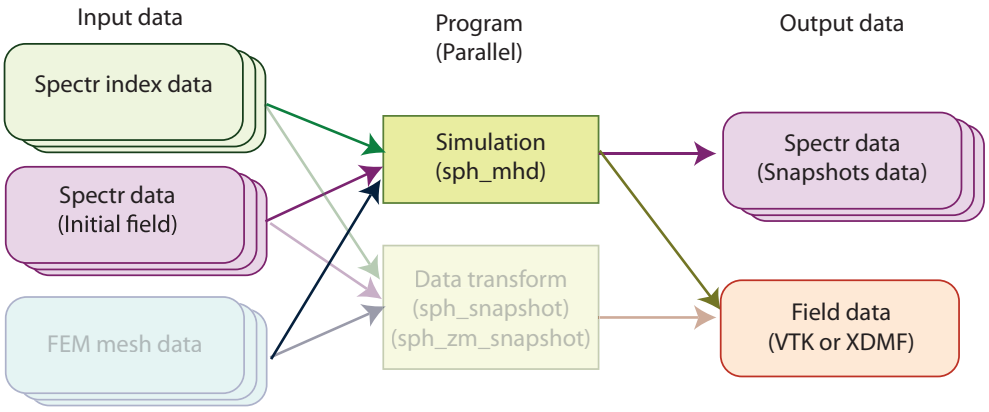


Figure 5: Data flow for the simulation program.

which starting with [sph_prefix] are obtained by the preprocessing program gen_sph_grid. The boundary condition data file [boundary_data_name] is optionally required if boundary conditions for temperature and composition are not homogenious.

Table 3: List of files for simulation sph_mhd

name	Parallelization	I/O
control_MHD	Serial	Input
[sph_prefix].[domain#].rj	Distributed	Input
[sph_prefix].[domain#].rlm	Distributed	Input
[sph_prefix].[domain#].rtm	Distributed	Input
[sph_prefix].[domain#].rtp	Distributed	Input
[sph_prefix].[domain#].gfm	Distributed	Input
[boundary_data_name]	Single	Input
[rst_prefix].[step#].[domain#].fst	Distributed	Input/Output
[vol_pwr_prefix].dat	Single	Output
[vol_pwr_prefix]_l.dat	Single	Output
[vol_pwr_prefix]_m.dat	Single	Output
[vol_pwr_prefix]_lm.dat	Single	Output
[vol_ave_prefix].dat	Single	Output
[layer_pwr_prefix]_l.dat	Single	Output
[layer_pwr_prefix]_m.dat	Single	Output
[layer_pwr_prefix]_lm.dat	Single	Output
[gauss_coef_prefix].dat	Single	Output
[picked_sph_prefix].dat	Single	Output
[fld_prefix].[step#].[domain#].[extension]	-	Output

10.1 Control file

The format of the control file `control_MHD` is described below. The detail of each block is described in section A. You can jump to detailed description by clicking "(Detail)".

`MHD_control` (Header of the control file)

- `data_files_def` (Detail)
  - `num_subdomain_ctl` `[Num_PE]` (Detail)
  - `num_smp_ctl` `[Num_Threads]` (Detail)
  - `sph_file_prefix` `[sph_prefix]` (Detail)
  - `boundary_data_file_name` `[boundary_data_name]` (Detail)
  - `restart_file_prefix` `[rst_prefix]` (Detail)
  - `field_file_prefix` `[fld_prefix]` (Detail)
  - `field_file_fmt_ctl` `[fld_format]` (Detail)
- `model`
  - `phys_values_ctl` (Detail)
    - `array nod_value_ctl` `[Field]` `[Viz_flag]` `[Monitor_flag]` (Detail)
  - `time_evolution_ctl` (Detail)
    - `array time_evo_ctl` `[Field]` (Detail)
  - `boundary_condition` (Detail)
    - `array bc_temperature` `[Group]` `[Type]` `[Value]` (Detail)
    - `array bc_velocity` `[Group]` `[Type]` `[Value]` (Detail)
    - `array bc_composition` `[Group]` `[Type]` `[Value]` (Detail)
    - `array bc_magnetic_field` `[Group]` `[Type]` `[Value]` (Detail)
  - `forces_define` (Detail)

    * array force_ctl [Force] (Detail)
- dimensionless_ctl (Detail)
    * array dimless_ctl [Name] [Value] (Detail)
- coefficients_ctl (Detail)
    * thermal (Detail)
        · array coef_4_termal_ctl [Name] [Power] (Detail)
        · array coef_4_t_diffuse_ctl [Name] [Power] (Detail)
        · array coef_4_heat_source_ctl [Name] [Power] (Detail)
    * momentum (Detail)
        · array coef_4_velocity_ctl [Name] [Power] (Detail)
        · array coef_4_press_ctl [Name] [Power] (Detail)
        · array coef_4_v_diffuse_ctl [Name] [Power] (Detail)
        · array coef_4_buoyancy_ctl [Name] [Power] (Detail)
        · array coef_4_Coriolis_ctl [Name] [Power] (Detail)
        · array coef_4_Lorentz_ctl [Name] [Power] (Detail)
        · array coef_4_composit_buoyancy_ctl [Name] [Power] (Detail)
    * induction (Detail)
        · array coef_4_magnetic_ctl [Name] [Power] (Detail)
        · array coef_4_m_diffuse_ctl [Name] [Power] (Detail)
        · array coef_4_induction_ctl [Name] [Power] (Detail)
    * composition (Detail)
        · array coef_4_composition_ctl [Name] [Power] (Detail)
        · array coef_4_c_diffuse_ctl [Name] [Power] (Detail)
        · array coef_4_composition_source_ctl [Name] [Power] (Detail)
- temperature_define (Detail)
    * ref_temp_ctl [REFERENCE_TEMP] (Detail)
    * low_temp_ctl (Detail)
        · depth [RADIUS] (Detail)
        · temperature [TEMPERATURE] (Detail)

        * high_temp_ctl (Detail)
            · depth [RADIUS] (Detail)
            · temperature [TEMPERATURE] (Detail)

- control
    - time_step_ctl (Detail)
        * elapsed_time_ctl [ELAPSED_TIME] (Detail)
        * i_step_init_ctl [ISTEP_START] (Detail)
        * i_step_finish_ctl [ISTEP_FINISH] (Detail)
        * i_step_check_ctl [ISTEP_MONITOR] (Detail)
        * i_step_rst_ctl [ISTEP_RESTART] (Detail)
        * i_step_field_ctl [ISTEP_FIELD] (Detail)
        * dt_ctl [DELTA_TIME] (Detail)
        * time_init_ctl [INITIAL_TIME] (Detail)
    - restart_file_ctl (Detail)
        * rst_ctl [INITIAL_TYPE] (Detail)
    - time_loop_ctl (Detail)
        * scheme_ctl [EVOLUTION_SCHEME] (Detail)
        * coef_imp_v_ctl [COEF_INP_U] (Detail)
        * coef_imp_t_ctl [COEF_INP_T] (Detail)
        * coef_imp_b_ctl [COEF_INP_B] (Detail)
        * coef_imp_c_ctl [COEF_INP_C] (Detail)
        * FFT_library_ctl [FFT_Name] (Detail)
        * Legendre_trans_loop_ctl [Leg_Loop] (Detail)

- sph_monitor_ctl (Detail)
    - volume_average_prefix [vol_ave_prefix] (Detail)
    - volume_pwr_spectr_prefix [vol_pwr_prefix] (Detail)
    - layered_pwr_spectr_prefix [layer_pwr_prefix] (Detail)

- picked_sph_prefix [picked_sph_prefix] (Detail)
- gauss_coefs_prefix [gauss_coef_prefix] (Detail)
- gauss_coefs_radius_ctl [gauss_coef_radius] (Detail)
- array pick_layer_ctl [Layer #] (Detail)
- array pick_sph_spectr_ctl [Degree] [Order] (Detail)
- array pick_sph_degree_ctl [Degree] (Detail)
- array pick_sph_order_ctl [Order] (Detail)
- array pick_gauss_coefs_ctl [Degree] [Order] (Detail)
- array pick_gauss_coef_degree_ctl [Degree] (Detail)
- array pick_gauss_coef_order_ctl [Order] (Detail)
- nphi_mid_eq_ctl [Nphi_mid_equator] (Detail)

10.2 Spectrum data for restarting

Spectrum data is used for restarting data and generating field data by Data transform program sph_snapshot, sph_zm_snapshot, or sph_dynamobench. This file is saved for each subdomain (MPI processes), then [step #] and [domain #] are added in the file name. The [step #] is calculated by time step / [ISTEP_RESTART].

10.3 Thermal and compositional boundary condition data file

Thermal and compositional heterogeneity at boundaries are defined by a external file named [boundary_data_name]. In this file, temperature, composition, heat flux, or compositional flux at ICB or CMB can be defined by spherical harmonics coefficients. To use boundary conditions in [boundary_data_name], file name is defined by boundary_data_file_name column in control file, and boundary condition type [type] is set to fixed_file or fixed_flux_file in bc_temperature or bc_composition column. By setting fixed_file or fixed_flux_file in control file, boundary conditions are copied from the file [boundary_data_name].

An example of the boundary condition file is shown in Figure 6. As for the control file, a line starting from '#' or '!' is recognized as a comment line. In `[boundary_data_name]`, boundary condition data is defined as following:

1. Number of total boundary conditions to be defined in this file.
2. Field name to define the first boundary condition
3. Place to define the first boundary condition (`ICB` or `CMB`)
4. Number of spherical harmonics modes for each boundary condition
5. Spectrum data for the boundary conditions (degree l, order m, and harmonics coefficients)
6. After finishing the list of spectrum data return to Step 2 for the next boundary condition

If harmonics coefficients of the boundary conditions are not listed in item 5, 0.0 is automatically applied for the harmonics coefficients of the boundary conditions. So, only non-zero components need to be listed in the boundary condition file.

10.4 Field data for visualization

Field data is used for the visualization processes. Field data are written with XDMF format (`http://www.xdmf.org/index.php/Main_Page`), merged VTK, or distributed VTK format (`http://www.vtk.org/VTK/img/file-formats.pdf`). The output data format is defined by `fld_format`. Visualization applications which we checked are listed in Table 4. Because the field data is written by using Cartesian coordinate (x, y, z) system, coordinate conversion is required to plot vector field in spherical coordinate (r, θ, ϕ) or cylindrical coordinate (s, ϕ, z). We will introduce a example of visualization process using ParaView in Section 17.

10.4.1 Distributed VTK data

Distributed VTK data have the following advantage and disadvantages to use:

- Advantage
  - Faster output
  - No external library is required

```
#
#  number of boundary conditions
      4
#
#   boundary condition data list
#
#    Fixed temperature at ICB
temperature
ICB
    3
  0  0   1.0E+00
  1  1   2.0E-01
  2  2   3.0E-01
#
#    Fixed heat flux at CMB
heat_flux
CMB
   2
  0  0    -0.9E+0
  1  -1    5.0E-1
#
#    Fixed composition flux at ICB
composite_flux
ICB
   2
  0  0    0.0E+00
  2  0   -2.5E-01
#
#    Fixed composition at CMB
composition
CMB
   2
  0   0   1.0E+00
  2  -2   5.0E-01
```

Figure 6: An example of boundary condition file.

Table 4: Checked visualization application

Format	Application
Distributed VTK	ParaView (`http://www.paraview.org`)
Merged VTK	ParaView, VisIt (`https://wci.llnl.gov/codes/visit/`) Mayavi (`http://mayavi.sourceforge.net/`)
XDMF	ParaView, VisIt

- Disadvantage
  - Many data files are generated
  - Total data file size is large
  - Only ParaView supports this format

Distributed VTK data consist files listed in Table 5. For ParaView, all subdomain data is read by choosing `[fld_prefix].[step#].pvtk` in file menu.

Table 5: List of written files for distributed VTK format

name	
`[fld_prefix].[step#].[domain#].vtk`	VTK data for each subdomain
`[fld_prefix].[step#].pvtk`	Subdomain file list for Paraview

10.4.2 Merged VTK data

Merged VTK data have the following advantage and disadvantages to use:

- Advantage
  - Merged field data is generated
  - No external library is required
  - Many applications support VTK format
- Disadvantage

  - Very slow to output
  - Total data file size is large

Merged VTK data generate files listed in Table 6.

Table 6: List of written files for merged VTK format

name	
`[fld_prefix].[step#].vtk`	Merged VTK data

10.4.3 Merged XDMF data

Merged XDMF data have the following advantage and disadvantages to use:

- Advantage
  - Fastest output
  - Merged field data is generated
  - File size is smaller than the VTK formats
- Disadvantage
  - Parallel HDF5 library should be required to use

Merged XDMF data generate files listed in Table 7. For ParaView, all subdomain data is read by choosing `[fld_prefix].solution.xdmf` in file menu.

Table 7: List of written files for XDMF format

name	
`[fld_prefix].mesh.h5`	HDF5 file for geometry data
`[fld_prefix].[step#].h5`	HDF5 file for field data
`[fld_prefix].solution.xdmf`	HDF5 file lists to be read

10.5 Mean square amplitude data

This program output mean square amplitude of the fields which is marked as `Monitor_ON` over the fluid shell at every `[increment_monitor]` steps. For vector fields, For the velocity $\boldsymbol{u}$ and magnetic field $\boldsymbol{B}$, the kinetic energy $1/2u^2$ and magnetic energy $1/2B^2$ are calculated instead of mean square amplitude. Labels on the first lines indicate following data. The data file have the following headers in the first 7 lines, and headers of the data and data are stored in the following lines. The header in the first 7 lines is the following. If these mean square amplitude data files exist before starting the simulation, programs append results at the end of files without checking constancy of the number of data and order of the field. If you change the configuration of data output structure, please move the existed data files to another directory before starting the programs.

`line 2:` Number of radial grid and truncation level

`line 4:` radial layer ID for ICB and CMB

`line 6:` Number of field of data, total number of components

`line 7:` Number of components for each field

Labels for data indicates as

`t_step` Time setp number

`time` Time

`K_ene_pol` Amplitude of poloidal kinetic energy

`K_ene_tor` Amplitude of toroidal kinetic energy

`K_ene` Amplitude of total kinetic energy

`M_ene_pol` Amplitude of poloidal magnetic energy

`M_ene_tor` Amplitude of toroidal magnetic energy

`M_ene` Amplitude of total magnetic energy

`[Field]_pol` Mean square amplitude of poloidal component of `[Field]`

`[Field]_tor` Mean square amplitude of toroidal component of `[Field]`

`[Field]` Mean square amplitude of `[Field]`

10.5.1 Volume average data

Volume average data are written by defining `volume_average_prefix` in control file. Volume average data are written in `[vol_ave_prefix].dat` with same format as RMS amplitude data. If you need the sphere average data for specific radial point, you can use picked spectrum data for $l = m = 0$ at specific radius.

10.5.2 Volume spectrum data

Volume spectrum data are written by defining `volume_pwr_spectr_prefix` in control file. By defining `volume_pwr_spectr_prefix`, following spectrum data averaged over the fluid shell is written. Data format is the same as the volume mean square data, but degree l, order m, or meridional wave number $l - m$ is added in the list of data.

`[vol_pwr_prefix_l.dat` Volume average of mean square amplitude of the fields as a function of spherical harmonic degree l. For scalar field, the spectrum is

$$f_{sq}(l) = \frac{1}{V} \sum_{m=-l}^{m=l} \int (f_l^m)^2 \, dV.$$

For vector field, spectrum for the poloidal and toroidal components are written by

$$B_{Ssq}(l) = \frac{1}{V} \sum_{m=-l}^{m=l} \int (\boldsymbol{B}_{Sl}^{\,m})^2 \, dV,$$

$$B_{Tsq}(l) = \frac{1}{V} \sum_{m=-l}^{m=l} \int (\boldsymbol{B}_{Tl}^{\,m})^2 \, dV.$$

If the vector field $\boldsymbol{F}$ is not solenoidal (i.e. $\nabla \cdot \boldsymbol{F} \neq 0$), The poloidal component of mean square data are included mean square field of the potential components as

$$F_{Ssq}(l) = \frac{1}{V} \sum_{m=-l}^{m=l} \int \left[(\boldsymbol{B}_{Sl}^{\,m})^2 + (-\nabla \phi_{Fl}^{\,m})^2 \right] dV.$$

`[vol_pwr_prefix]_m.dat` Volume average of mean square amplitude of the fields as a function of spherical harmonic order m. The zonal wave number is referred in this spectrum data. For scalar field, the spectrum is

$$f_{sq}(m) = \frac{1}{V} \sum_{l=0}^{l=m} \int \left[(f_l^m)^2 + \left(f_l^{-m}\right)^2 \right] dV.$$

For vector field, spectrum for the poloidal and toroidal components are written by

$$B_{Ssq}(m) = \frac{1}{V}\sum_{l=0}^{l=m}\int\left[\left(\boldsymbol{B}_{Sl}^{m}\right)^2+\left(\boldsymbol{B}_{Sl}^{-m}\right)^2\right]dV,$$
$$B_{Tsq}(m) = \frac{1}{V}\sum_{l=0}^{l=m}\int\left[\left(\boldsymbol{B}_{Tl}^{m}\right)^2+\left(\boldsymbol{B}_{Tl}^{-m}\right)^2\right]dV.$$

`[vol_pwr_prefix]_lm.dat` Volume average of mean square amplitude of the fields as a function of spherical harmonic order $n = l - m$. The wave number in the latitude direction is referred in this spectrum data. For scalar field, the spectrum is

$$f_{sq}(n) = \frac{1}{V}\sum_{l=n}^{l=l-n}\int\left[\left(f_l^{l-n}\right)^2+\left(f_l^{-l+n}\right)^2\right]dV.$$

For vector field, spectrum for the poloidal and toroidal components are written by

$$B_{Ssq}(n) = \frac{1}{V}\sum_{l=n}^{l=l-n}\int\left[\left(\boldsymbol{B}_{Sl}^{l-n}\right)^2+\left(\boldsymbol{B}_{Sl}^{-l+n}\right)^2\right]dV,$$
$$B_{Tsq}(n) = \frac{1}{V}\sum_{l=n}^{l=l-n}\int\left[\left(\boldsymbol{B}_{Tl}^{l-n}\right)^2+\left(\boldsymbol{B}_{Tl}^{-l+n}\right)^2\right]dV.$$

10.5.3 layered spectrum data

Spectrum data for the each radial position are written by defining `volume_pwr_spectr_prefix` in control file. By defining `layered_pwr_spectr_prefix`, following spectrum data averaged over the fluid shell is written. Data format is the same as the volume spectrum data, but radial grid point and radius of the layer is added in the list. The following files are generated.

`[layer_pwr_prefix]_l.dat` Surface average of mean square amplitude of the fields as a function of spherical harmonic degree l and radial grid id k. For scalar field, the spectrum is

$$f_{sq}(k,l) = \frac{1}{S}\sum_{m=-l}^{m=l}\int\left(f_l^m\right)^2 dS.$$

For vector field, spectrum for the poloidal and toroidal components are written by

$$\begin{aligned} B_{Ssq}(k,l) &= \frac{1}{S}\sum_{m=-l}^{m=l}\int (\boldsymbol{B}_{Sl}^{\,m})^2\, dS, \\ B_{Tsq}(k,l) &= \frac{1}{S}\sum_{m=-l}^{m=l}\int (\boldsymbol{B}_{Tl}^{\,m})^2\, dS. \end{aligned}$$

`[layer_pwr_prefix]_m.dat` Surace average of mean square amplitude of the fields as a function of spherical harmonic order m and radial grid id k. The zonal wave number is referred in this spectrum data. For scalar field, the spectrum is

$$f_{sq}(k,m) = \frac{1}{S}\sum_{l=m}^{l=L}\int \left[(f_l^m)^2 + \left(f_l^{-m}\right)^2\right] dS.$$

For vector field, spectrum for the poloidal and toroidal components are written by

$$\begin{aligned} B_{Ssq}(k,m) &= \frac{1}{S}\sum_{l=m}^{l=L}\int \left[(\boldsymbol{B}_{Sl}^{\,m})^2 + \left(\boldsymbol{B}_{Sl}^{\,-m}\right)^2\right] dS, \\ B_{Tsq}(k,m) &= \frac{1}{S}\sum_{l=m}^{l=L}\int \left[(\boldsymbol{B}_{Tl}^{\,m})^2 + \left(\boldsymbol{B}_{Tl}^{\,-m}\right)^2\right] dS. \end{aligned}$$

`[layer_pwr_prefix]_lm.dat` Surface average of mean square amplitude of the fields as a function of spherical harmonic order $n = l - m$ and radial grid id k. The wave number in the latitude direction is referred in this spectrum data. For scalar field, the spectrum is

$$f_{sq}(k,n) = \frac{1}{S}\sum_{l=n}^{l=L}\int \left[\left(f_l^{l-n}\right)^2 + \left(f_l^{-l+n}\right)^2\right] dS.$$

For vector field, spectrum for the poloidal and toroidal components are written by

$$\begin{aligned} B_{Ssq}(k,n) &= \frac{1}{S}\sum_{l=n}^{l=L}\int \left[\left(\boldsymbol{B}_{Sl}^{\,l-n}\right)^2 + \left(\boldsymbol{B}_{Sl}^{\,-l+n}\right)^2\right] dS, \\ B_{Tsq}(k,n) &= \frac{1}{S}\sum_{l=n}^{l=L}\int \left[\left(\boldsymbol{B}_{Tl}^{\,l-n}\right)^2 + \left(\boldsymbol{B}_{Tl}^{\,-l+n}\right)^2\right] dS. \end{aligned}$$

10.6 Gauss coefficient data `[gauss_coef_prefix].dat`

This program output selected Gauss coefficients of the magnetic field. Gauss coefficients is evaluated for radius defined by `[gauss_coef_radius]` every `[increment_monitor]` steps. Gauss coefficients are evaluated by using poloidal magnetic field at CMB $B_{Sl}^{m}(r_o)$ and radius defined by `[gauss_coef_radius]` r_e as

$$\begin{aligned} g_l^m &= \frac{l}{r_e^2}\left(\frac{r_o}{r_e}\right)^l B_{Sl}^{m}(r_o), \\ h_l^m &= \frac{l}{r_e^2}\left(\frac{r_o}{r_e}\right)^l B_{Sl}^{-m}(r_o). \end{aligned}$$

The data file has the following headers in the first three lines,

`line 2:` Number of saved Gauss coefficients and reference radius.

`line 3:` Labels of Gauss coefficients data.

The data consists of time step, time, and Gauss coefficients for each step in one line. If the Gauss coefficients data file exist before starting the simulation, programs append Gauss coefficients at the end of files without checking constancy of the number of data and order of the field. If you change the configuration of data output structure, please move the old Gauss coefficients file to another directory before starting the programs.

10.7 Spectrum monitor data `[picked_sph_prefix].dat`

This program outputs spherical harmonics coefficients at specified spherical harmonics modes and radial points in single text file. Spectrum data marked `[Monitor_On]` are written in our line for each spherical harmonics mode and radial point every `[increment_monitor]` steps. If the spectrum monitor data file exist before starting the simulation, programs append spectrum data at the end of files without checking constancy of the number of data and order of the field. If you change the configuration of data output structure, please move the old spectrum monitor file to another directory before starting the programs.

If a vector field $\boldsymbol{F}$ is not a solenoidal field, $\boldsymbol{F}$ is described by the spherical harmonics coefficients of the poloidal F_{Sl}^{m}, toroidal F_{Tl}^{m}, and potential φ_l^m components as

$$\boldsymbol{F}(r,\theta,\phi) = -\frac{1}{r^2}\frac{\partial \varphi_0^0}{\partial r}\hat{r} + \sum_{l=1}^{L}\sum_{m=-l}^{l}\left[\nabla\times\nabla\times(F_{Sl}^{m}\hat{r}) + \nabla\times(F_{Tl}^{m}) - \nabla(\varphi_l^m Y_l^m)\right].$$

In Calypso, the following coefficients are written for the non-solenoidal vector.

`[field_name]_pol :` $\begin{cases} F_{Sl}^{m} - \dfrac{r^2}{l\,(l+1)} \dfrac{\partial \varphi_l^m}{\partial r} & \text{for} \quad (l \neq 0) \\ -r^2 \dfrac{\partial \varphi_0^0}{\partial r} & \text{for} \quad (l = 0) \end{cases}$

`[field_name]_dpdr :` $\begin{cases} \dfrac{\partial F_{Sl}^{m}}{\partial r} - \varphi_l^m & \text{for} \quad (l \neq 0) \\ 0 & \text{for} \quad (l = 0) \end{cases}$

`[field_name]_tor :` F_{Tl}^{m}

11 Data transform program (`sph_snapshot` and `sph_zm_snapshot`)

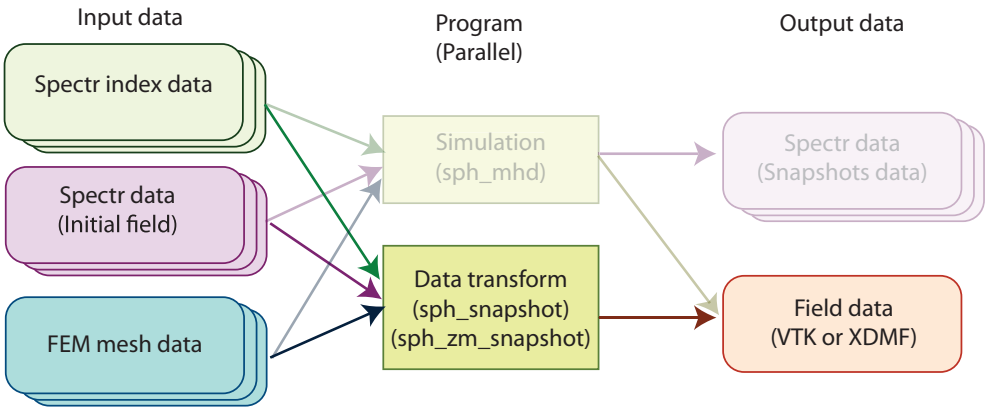


Figure 7: Data flow for data transform program.

Simulation program outputs spectrum data as a whole field data. This program generates field data from spectrum data for visualization. This program also can pick Gauss coefficients, mean square data over sphere or each surface from spectrum data as the simulation program.

This program requires control file `control_snapshot`. File format of the control file is same as the control field for simulation `control_MHD`.

The same files as the simulation program are read in this program, and field data are generated from the snapshots of spectrum data. The monitoring data for snapshots can also be generated. `[step #]` is added in the file name, and the `[step #]` is calculated by `time step` / `[ISTEP_FIELD]`.

Table 8: List of files for simulation sph_snap and sph_zm_snap

name	Parallelization	I/O
control_snapshot	Serial	Input
[sph_prefix].[domain#].rj	Distributed	Input
[sph_prefix].[domain#].rlm	Distributed	Input
[sph_prefix].[domain#].rtm	Distributed	Input
[sph_prefix].[domain#].rtp	Distributed	Input
[sph_prefix].[domain#].gfm	Distributed	Input
[boundary_data_name]	Single	Input
[rst_prefix].[step#].[domain#].fst	Distributed	Input
[vol_pwr_prefix].dat	Single	Output
[vol_pwr_prefix]_l.dat	Single	Output
[vol_pwr_prefix]_m.dat	Single	Output
[vol_pwr_prefix]_lm.dat	Single	Output
[vol_ave_prefix].dat	Single	Output
[layer_pwr_prefix]_l.dat	Single	Output
[layer_pwr_prefix]_m.dat	Single	Output
[layer_pwr_prefix]_lm.dat	Single	Output
[gauss_coef_prefix].dat	Single	Output
[picked_sph_prefix].dat	Single	Output
[fld_prefix].[step#].[domain#].[extension]	-	Output

12 Initial field generation program (sph_initial_field)

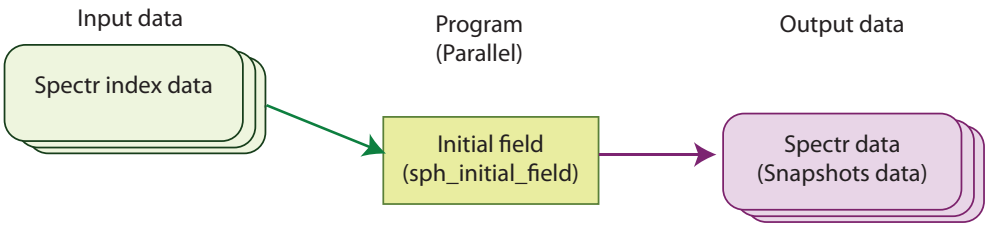


Figure 8: Data flow for initial field generation program.

The initial fields for dynamo benchmark can set in the simulation program by setting [INITIAL_TYPE] flag. This program is used to generate initial field by user. The heat source q_T and light element source q_C are also defined by this program because q_T and q_C are defined as scalar fields. The Fortran source file to define initial field const_sph_initial_spectr.f90 is saved in src/programs/data_utilities /INITIAL_FIELD/ directory, and please compile again after modifying this module. This program also needs the files listed in Table 9. This program generates the spectrum

Table 9: List of files for simulation sph_initial_field

name	Parallelization	I/O
control_MHD	Serial	Input
[sph_prefix].[domain#].rj	Distributed	Input
[sph_prefix].[domain#].rlm	Distributed	Input
[sph_prefix].[domain#].rtm	Distributed	Input
[sph_prefix].[domain#].rtp	Distributed	Input
[rst_prefix].0.[domain#].fst	Distributed	Input/Output

data files [rst_prefix].0.[domain#].fst. To use generated initial data file,

please set [ISTEP_START] to be 0 and [INITIAL_TYPE] to be start_from_rst_file.

13 Initial field modification program (sph_add_initial_field)

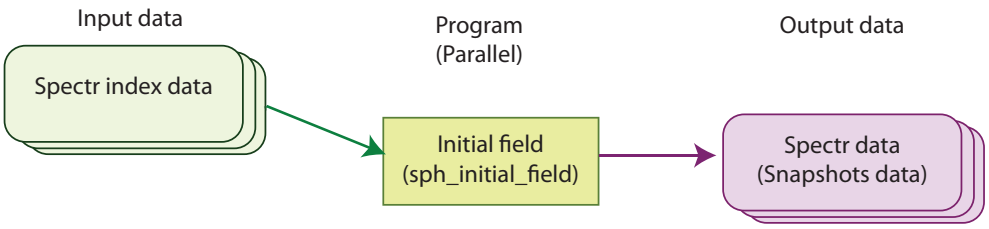


Figure 9: Data flow for initial field modification program.

Caution: This program overwrites existing initial field data. Please run it after taking a backup.

This program modifies or adds new data to an initial field file. It could be used to start a new geodynamo simulation by adding seed magnetic field or source terms to a non-magnetic convection simulation. The initial fields to be added are also defined in const_sph_initial_spectr.f90. data_utilities/INITIAL_FIELD/ directory. This program also needs the files listed in Table 10. This program generates the spectrum data files [rst_prefix].[step#].[domain#].fst. To use generated initial data file, set [ISTEP_START] and [ISTEP_RESTART] to be appropriate time step and increment, respectively. To read the original initial field data, [INITIAL_TYPE] is set to be start_from_rst_file in control_MHD. In other words, the [step #] in the file name, [ISTEP_START], and [ISTEP_RESTART] in the control file should be the consistent.

Table 10: List of files for simulation sph_add_initial_field

name	Parallelization	I/O
control_MHD	Serial	Input
[sph_prefix].[domain#].rj	Distributed	Input
[sph_prefix].[domain#].rlm	Distributed	Input
[sph_prefix].[domain#].rtm	Distributed	Input
[sph_prefix].[domain#].rtp	Distributed	Input
[rst_prefix].[step #].[domain#].fst	Distributed	Input/Output

14 Check program for dynamo benchmark (sph_dynamobench)

This program is only used to check solution for dynamo benchmark by Christensen *et. al*. The following files are used for this program.

Table 11: List of files for dynamo benchmark check sph_dynamobench

name	Parallelization	I/O
control_snapshot	Serial	Input
[sph_prefix].[domain#].rj	Distributed	Input
[sph_prefix].[domain#].rlm	Distributed	Input
[sph_prefix].[domain#].rtm	Distributed	Input
[sph_prefix].[domain#].rtp	Distributed	Input
[rst_prefix].[step#].[domain#].fst	Distributed	Input
dynamobench.dat	Single	Output

14.1 Dynamo benchmark data dynamobench.dat

In benchmark test by Christensen *et. al.*, both global values and local values are checked. As global results, Kinetic energy $\frac{1}{V}\int \frac{1}{2}u^2 dV$ in the fluid shell, magnetic energy in the fluid shell $\frac{1}{V}\frac{1}{EPm}\int \frac{1}{2}B^2 dV$ (for case 1 and 2), and magnetic energy in the solid

inner sphere $\frac{1}{V_i}\frac{1}{EPm}\int\frac{1}{2}B^2dV_i$ (for case 2 only). Benchmark also requests By increasing number of grid point at mid-dpeth of the fluid shell in the equatorial plane by `nphi_mid_eq_ctl`, program can find accurate solution for the point where $u_r = 0$ and $\partial u_r/\partial\phi > 0$. Angular frequency of the field pattern with respect to the ϕ direction is also required. The benchmark test also requires temperature and θ component of velocity. In the text file `dynamobench.dat`, the following data are written in one line for every `[i_step_rst_ctl]` step.

`t_step:` Time step number

`time:` Time

`KE_pol:` Poloidal kinetic energy

`KE_tor:` Toroidal kinetic energy

`KE_total:` Total kinetic energy

`ME_pol:` Poloidal magnetic energy (Case 1 and 2)

`ME_tor:` Toroidal magnetic energy (Case 1 and 2)

`ME_total:` Total magnetic energy (Case 1 and 2)

`ME_pol_ic:` Poloidal magnetic energy in inner core (Case 2)

`ME_tor_icore:` Toroidal magnetic energy in inner core (Case 2)

`ME_total_icore:` Total magnetic energy in inner core (Case 2)

`omega_ic_z:` Angular velocity of inner core rotation (Case 2)

`MAG_torque_ic_z:` Magnetic torque integrated over the inner core (Case 2)

`phi_1...4:` Longitude where $u_r = 0$ and $\partial u_r/\partial\phi > 0$ at mid-depth in equatorial plane.

`omega_vp44:` Drift frequency evaluated by V_{S4}^4 component

`omega_vt54:` Drift frequency evaluated by V_{T5}^4 component

`B_theta:` Θ component of magnetic field at requested point.

`v_phi:` ϕ component of velocity at requested point.

`temp:` Temperature at requested point.

```
t_step    time    KE_pol    KE_tor    KE_total    ME_pol    ME_t
or    ME_total    ME_pol_icore    ME_tor_icore    ME_total_icore
    omega_ic_z    MAG_torque_ic_z    phi_1    phi_2    phi_3
phi_4    omega_vp44    omega_vt54    B_theta    v_phi    temp
     20000   9.999999999998981E-001   1.534059732073072E+001   2
.431439471284618E+001   3.965499203357688E+001   2.4056940119550
09E+000   1.648662987055900E+000   4.054356999010911E+000   3.90
8687924452961E+001   4.812865754441352E-001   3.956816581997376E
+001   5.220517005592486E+000  -2.321885847438682E+002   3.59417
5626663308E-001   1.930213889461227E+000   3.501010216256124E+00
0   5.071806543051021E+000   7.808553595635292E-001  -1.64958344
1437563E-001  -5.136522824340612E+000  -8.047915942925034E+000
 3.752181234262930E-001
...
```

15 Data assemble program (`assemble_sph`)

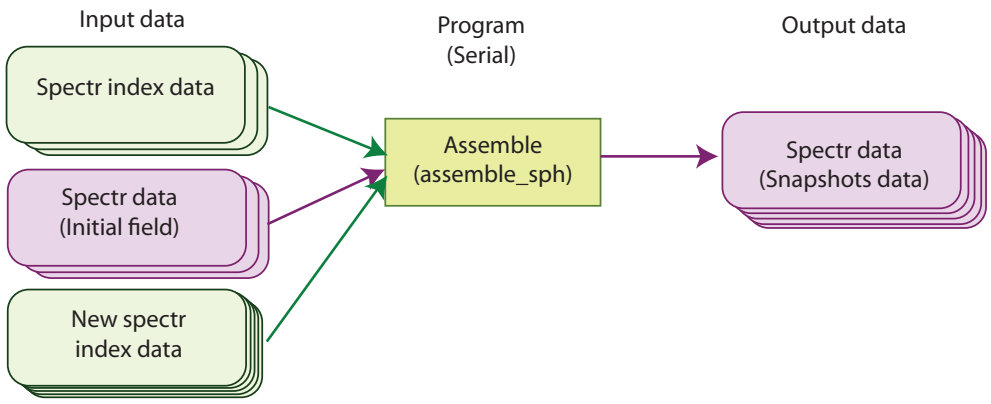


Figure 10: Data flow for spectrum data assemble program

Calypso uses distributed data files for simulations. This program is to generate new spectrum data for restarting with different spatial resolution or parallel configuration. This program organizes new spectral data by using specter indexing data using different domain decomposition. The following files used for data IO. If radial resolution is changed from the original data, the program makes new spectrum data by linear interpolation. If new

data have smaller or larger truncation degree, the program fills zero to the new spectrum data or truncates the data to fit the new spatial resolution, respectively. Data files for the program are shown In Table 12, and definition of `control_assemble_sph` is

Table 12: List of files for `assemble_sph`

extension	Distributed?	I/O
`control_sph_assemble`	Serial	Input
`[sph_prefix].[domain#].rj`	Distributed	Input
`[new_sph_prefix].[domain#].rj`	Distributed	Input
`[rst_prefix].[step#].[domain#].fst`	Distributed	Input
`[new_rst_prefix].[step#].[domain#].fst`	Distributed	Output

15.1 Format of control file

Control file consists the following groups.
`assemble_control`

- `data_files_def` (Detail)
  - `num_subdomain_ctl` `[Num_PE]` (Detail)
  - `sph_file_prefix` `[sph_prefix]` (Detail)
  - `restart_file_prefix` `[rst_prefix]` (Detail)
- `new_data_files_def` (Detail)
  - `num_new_domain_ctl` `[new_num_domain]` (Detail)
  - `new_sph_mode_prefix` `[new_sph_prefix]` (Detail)
  - `new_restart_prefix` `[new_rst_prefix]` (Detail)
  - `delete_original_data_flag` `[YES or NO]` (Detail)
- `control`
  - `time_step_ctl` (Detail)
    - `i_step_init_ctl` `[integer]` (Detail)
    - `i_step_finish_ctl` `[integer]` (Detail)

    * i_step_rst_ctl [integer] (Detail)

- newrst_magne_ctl (Detail)
  - magnetic_field_ratio_ctl [ratio] (Detail)

16 Module dependency program (module_dependency)

This program is only used to generate Makefile in work directory. Most of case, Fortran 90 modules have to compiled prior to be referred by another fortran90 routines. This program is generates dependency lists in Makefile. To use this program, the following limitation is required.

- One source code has to consist of one module.
- The module name should be the same as the file name.

17 Visualization using field data

The field data is written by XDMF or VTK data format using Cartesian coordinate. In this section we briefly introduce how to display the radial magnetic field using ParaView as an example.

After the starting Paraview, the file to be read is chosen in the file menu, and press ”apply”, button. Then, Paraview load the data from files (see Figure 11). Because the magnetic field is saved by the Cartesian coordinate, the radial magnetic field is obtained by the calculator tool. The procedure is as following (see Figure 12)

1. Push calculator button.
2. Choose ”Point Data” in Attribute menu
3. Input data name for radial magnetic field (”B_r” in Figure 12)
4. Enter the equation to evaluate radial mantic field $B_r = \boldsymbol{B} \cdot \boldsymbol{r}/|r|$.
5. Finally, push ”Apply” button.

After obtaining the radial mantric field, the image in figure 13 is obtained by using ”slice” and ”Contour” tools with appropriate color mapping.

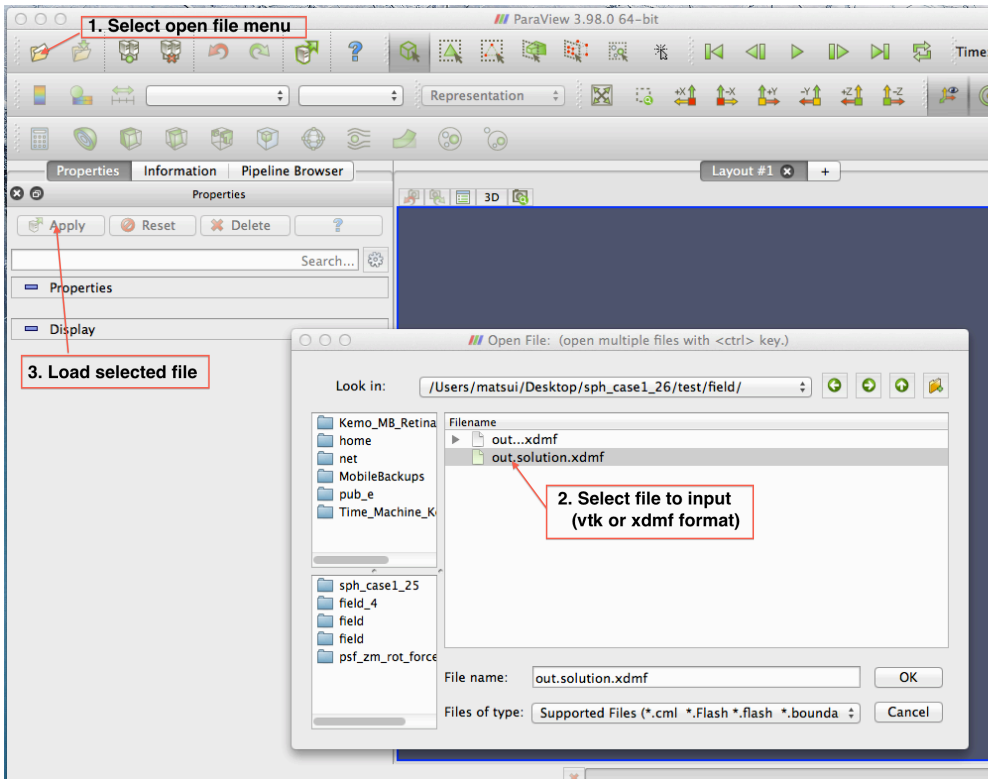


Figure 11: File open window for ParaView

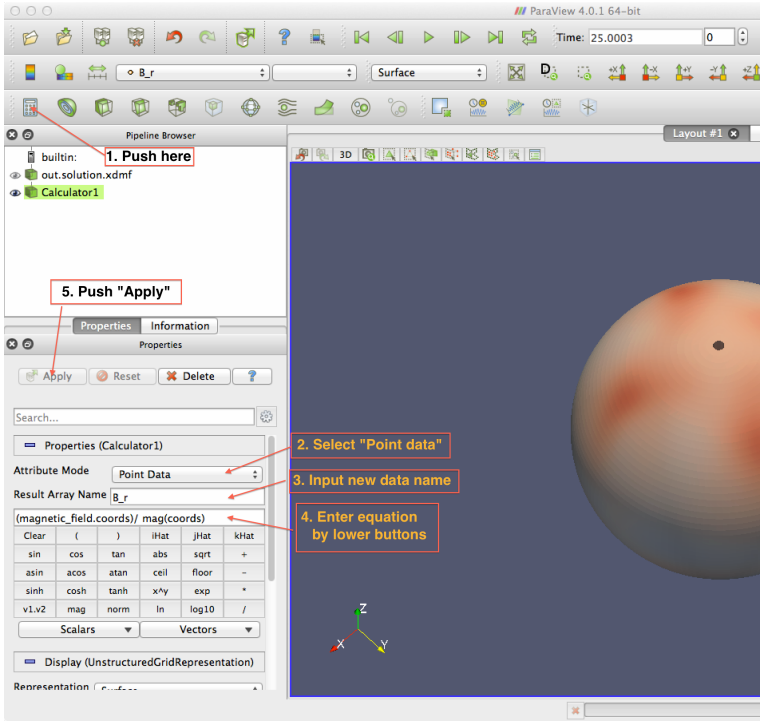


Figure 12: File open window for ParaView

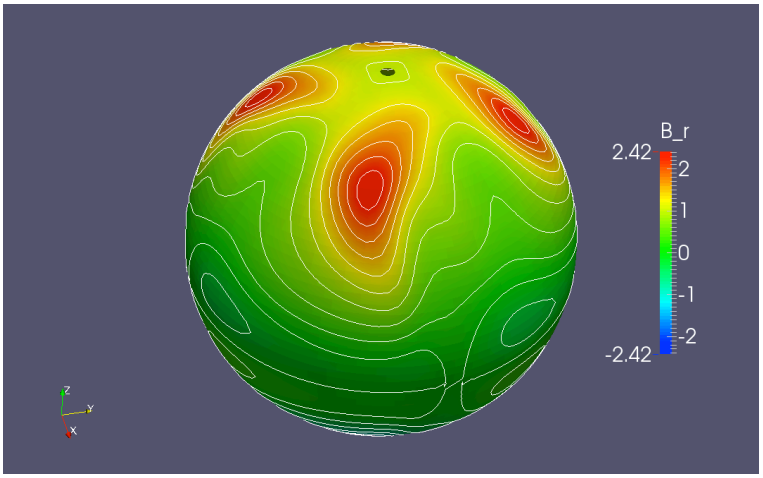


Figure 13: Visualization of radial magnetic field by Paraview.

References

[1] Bullard, E. C. and Gellman, H., Homogeneous dynamos and terrestrial magnetism, *Proc. of the Roy. Soc. of London*, **A247**, 213–278, 1954.

[2] Christensen, U.R., Aubert, J., Cardin, P., Dormy, E., Gibbons, S., Glatzmaier, G. A., Grote, E., Honkura, H., Jones, C., Kono, M., Matsushima, M., Sakuraba, A., Takahashi, F., Tilgner, A., Wicht, J. and Zhang, K., A numerical dynamo benchmark, *Physics of the Earth and Planetary Interiors*, **128**, 25–34, 2001.

Appendix A Definition of parameters for control files

A.1 data_files_def

File names and number of processes and threads are defined in this block.
(Back to control_MHD)
(Back to control_sph_shell)
(Back to control_assemble_sph)

num_subdomain_ctl [Num_PE]
Number of subdomain for the MPI program [Num_PE] is defined by integer. If number of processes in mpirun -np is different from number of subdomains, program will be stopped with message.

num_smp_ctl [Num_Threads]
Number of SMP threads for OpenMP [Num_Threads] is defined by integer. You can set larger number than the actual umber of thread to be used. If actual number of thread is less than this number, number of threads is set to the number which is defined in this field.

sph_file_prefix [sph_prefix]
File prefix of spherical harmonics indexing and FEM mesh file [sph_prefix] is defined by text. Process ID and extension are added after this file prefix.

boundary_data_file_name [boundary_data_name]
File name of boundary condition data file [boundary_data_name] is defined by text.

restart_file_prefix [rst_prefix]
File prefix of spectrum data for restarting and snapshots [rst_prefix] is defined by text. Step number, process ID, and extension are added after this file prefix.

field_file_prefix [fld_prefix]
File prefix of field data for visualize snapshots [fld_prefix] is defined by text. Step number and file extension are added after this file prefix.

field_file_fmt_ctl [fld_format]
Field data field format for visualize snapshots [fld_format] is defined by text. The following formats are currently supported.

`single_HDF5` Merged HDF5 file (Available if HDF5 library is linked)

`single_VTK` Merged VTK file (Default)

`VTK` Distributed VTK file

A.2 `phys_values_ctl`

Fields for the simulation are defined in this block.
(Back to `control_MHD`)

`array nod_value_ctl [Field] [Viz_flag] [Monitor_flag]`
Fields name `[Field]` for the simulation are listed in this array. If required fields for simulation are not in the list, simulation program adds required field in the list, but does not output any field data and monitoring data. `[Viz_flag]` is set to output of the field data for visualization by

`VIz_On` Write field data to VTK file

`VIz_Off` Do not write field data to VTK file.

In the `[Monitor_flag]`, output in the monitoring data is defined by

`Monitor_On` Write spectrum into monitoring data

`Monitor_Off` Do not write spectrum into monitoring data

Supported field in the present version is listed in Table 13

A.3 `time_evolution_ctl`

Fields for time evolution are defined in this block.
(Back to `control_MHD`)

`array time_evo_ctl [Field]`
Fields name for time evolution are listed in this array in `[Field]` by text. Available fields are listed in Table 14.

A.4 `boundary_condition`

Boundary condition are defined in this block.
(Back to `control_MHD`)

Table 13: List of field name

[Name]	field name	Description
velocity	Velocity	$\boldsymbol{u}$
vorticity	Vorticity	$\boldsymbol{\omega} = \nabla \times \boldsymbol{u}$
pressure	Pressure	P
temperature	Temperature	T
perturbation_temp	Perturbation of temperature	$\Theta = T - T_0$
heat_source	Heat source	q_T
composition	Composition variation	C
composition_source	Composition source	q_C
magnetic_field	Magnetic field	$\boldsymbol{B}$
current_density	Current density	$\boldsymbol{J} = \nabla \times \boldsymbol{B}$
electric_field	Electric field	$\boldsymbol{E} = \sigma\left(\boldsymbol{J} - \boldsymbol{u} \times \boldsymbol{B}\right)$
viscous_diffusion	Viscous diffusion	$-\nu\nabla \times \nabla \times \boldsymbol{u}$
buoyancy	Thermal buoyancy	$-\alpha_T T\boldsymbol{g}$
composite_buoyancy	Compositional buoyancy	$-\alpha_C C\boldsymbol{g}$
Lorentz_force	Lorentz force	$\boldsymbol{J} \times \boldsymbol{B}$
Coriolis_force	Coriolis force	$-2\Omega\hat{z} \times \boldsymbol{u}$
thermal_diffusion	Termal diffusion	$\kappa_T\nabla^2 T$
grad_temp	Temperature gradient	∇T
heat_flux	Advective heat flux	$\boldsymbol{u}T$
composition_diffusion	Compositional diffusion	$\kappa_C\nabla^2 C$
grad_composition	Composition gradient	∇C
composite_flux	Advective composition flux	$\boldsymbol{u}C$
magnetic_diffusion	Magnetic diffusion	$-\eta\nabla \times \nabla \times \boldsymbol{B}$
poynting_flux	Poynting flux	$\boldsymbol{E} \times \boldsymbol{B}$
rot_Lorentz_force	Curl of Lorentz force	$\nabla \times (\boldsymbol{J} \times \boldsymbol{B})$
rot_Coriolis_force	Curl of Coriolis force	$-2\Omega\nabla \times (\hat{z} \times \boldsymbol{u})$
rot_buoyancy	Curl of thermal buoyancy	$-\nabla \times (\alpha_T T\boldsymbol{g})$
rot_composite_buoyancy	Curl of compositional buoyancy	$-\nabla \times (\alpha_C C\boldsymbol{g})$
buoyancy_flux	Buoyancy flux	$-\alpha_T T\boldsymbol{g} \cdot \boldsymbol{u}$
Lorentz_work	Work of Lorentz force	$\boldsymbol{u} \cdot (\boldsymbol{J} \times \boldsymbol{B})$

Table 14: List of field name for time evolution

label	field name	Description
velocity	Velocity	$\boldsymbol{u}$
temperature	Temperature	T
composition	Composition variation	C
magnetic_field	Magnetic field	$\boldsymbol{B}$

array bc_temperature [Group] [Type] [Value]
Boundary conditions for temperature are defined by this array. Position of boundary is defined in [Group] column by ICB or CMB. The following type of boundary conditions are available for temperature in [Type] column.

fixed Fixed homogeneous temperature on the boundary. The fixed value is defined in [Value] by real.

fixed_file Fixed temperature defined by external file. [Value] in this line is ignored. See section 10.3.

fixed_flux Fixed homogeneous heat flux on the boundary. The value is defined in [Value] by real. Positive value indicates outward flux from fluid shell. (*e.g.* Flux to center at ICB and Flux to mantle at CMB are positive.)

fixed_flux_file Fixed heat flux defined by external file. [Value] in this line is ignored. See section 10.3.

array bc_velocity [Group] [Type] [Value]
Boundary conditions for velocity are defined by this array. Position of boundary is defined in [Group] by ICB or CMB. The following boundary conditions are available for velocity in [Type] column.

non_slip_sph Non-slip boundary is applied to the boundary defined in [Group]. Real value is required in [Value], but they value is not used in the program.

free_slip_sph Free-slip boundary is applied to the boundary defined in [Group]. Real value is required in [Value], but they value is not used in the program.

rot_inner_core If this condition is set, inner core ($r < r_i$) rotation is solved by using viscous torque and Lorentz torque. This boundary condition can be used for ICB,

and grid is filled to center. Real value is required in `[Value]`, but they value is not used in the program.

`rot_x` Set constant rotation around x-axis in `[Value]` by real. Rotation vector can be defined with `rot_y` and `rot_z`.

`rot_y` Set constant rotation around y-axis in `[Value]` by real. Rotation vector can be defined with `rot_z` and `rot_x`.

`rot_z` Set constant rotation around z-axis in `[Value]` by real. Rotation vector can be defined with `rot_x` and `rot_y`.

`array bc_magnetic_field [Group] [Type] [Value]`
Boundary conditions for magnetic field are defined by this array. Position of boundary is defined in `[Group]` by `to_Center`, `ICB`, or `CMB`. The following boundary conditions are available for magnetic field in `[Type]` column.

`insulator` Magnetic field is connected to potential field at boundary defined in [Group]. real value is required at `[Value]`, but they value is not used in the program.

`sph_to_center` If this condition is set, magnetic field in conductive inner core ($r < r_i$) is solved. This boundary condition can be used for `ICB`, and grid is filled to center. The value at `[Value]` does not used.

`array bc_composition [Group] [Type] [Value]`
Boundary conditions for composition variation are defined by this array. Position of boundary is defined in `[Group]` by `ICB` or `CMB`. The following boundary conditions are available for composition variation in `[Type]` column.

`fixed` Fixed homogeneous composition on the boundary. The fixed value is defined in `[Value]` by real.

`fixed_file` Fixed composition defined by external file. `[Value]` in this line is ignored. See section 10.3.

`fixed_flux` Fixed homogeneous compositional flux on the boundary. The value is defined in `[Value]` by real. Positive value indicates outward flux from fluid shell. (*e.g.* Flux to center at ICB and Flux to mantle at CMB are positive.)

`fixed_flux_file` Fixed compositional flux defined by external file. `[Value]` in this line is ignored. See section 10.3.

A.5 forces_define

Forces for the momentum equation are defined in this block.
(Back to control_MHD)

array force_ctl [Force]
Name of forces for momentum equation are listed in [Force] by text. The following fields are available.

Table 15: List of force

Label	Field name	Equation
Coriolis	Coriolis force	$-2\Omega\hat{z} \times \boldsymbol{u}$
Lorentz	Lorentz force	$\boldsymbol{J} \times \boldsymbol{B}$
gravity	Thermal buoyancy	$-\alpha_T T \boldsymbol{g}$
Composite_gravity	Compositional buoyancy	$-\alpha_C C \boldsymbol{g}$

A.6 dimensionless_ctl

Dimensionless numbers are defined in this block.
(Back to control_MHD)

array dimless_ctl [Name] [Value]
Dimensionless are listed in this array. The name is defined in [Name] by text, and value is defined in [Value] by real. These name of the dimensionless numbers are used to construct coefficients for each terms in governing equations. The following names can not be used because of reserved name in the program.

Table 16: List of reserved name of dimensionless numbers

label	field name	value
Zero	zero	0.0
One	one	1.0
Two	two	2.0
Radial_35	Ratio of outer core thickness to whole core	$0.65 = 1 - 0.35$

A.7 coefficients_ctl

Coefficients of each term in governing equations are defined in this block. Each coefficients are defined by list of name of dimensionless number [Name] and its power [Power]. For example, coefficient for Coriolis term for the dynamo benchmark $2E^{-1}$ is defined as

```
array coef_4_Coriolis_ctl   2
  coef_4_Coriolis_ctl       Two             1.0
  coef_4_Coriolis_ctl       Ekman_number  -1.0
end array coef_4_Coriolis_ctl
```

(Back to control_MHD)

A.7.1 thermal

Coefficients of each term in heat equation are defined in this block.
(Back to control_MHD)

coef_4_termal_ctl [Name] [Power]
Coefficient for evolution of temperature $\dfrac{\partial T}{\partial t}$ and advection of heat $(\boldsymbol{u} \cdot \nabla) T$ is defined by this array.

coef_4_t_diffuse_ctl [Name] [Power]
Coefficient for thermal diffusion $\kappa_T \nabla^2 T$ is defined by this array.

coef_4_heat_source_ctll [Name] [Power]
Coefficient for heat source q_T is defined by this array.

A.7.2 momentum

Coefficients of each term in momentum equation are defined in this block.
(Back to control_MHD)

coef_4_velocity_ctl [Name] [Power]
Coefficient for evolution of velocity $\dfrac{\partial \boldsymbol{u}}{\partial t}$ (or $\dfrac{\partial \boldsymbol{\omega}}{\partial t}$ for the vorticity equation) and advection $-\boldsymbol{\omega} \times \boldsymbol{u}$ (or $-\nabla \times (\boldsymbol{\omega} \times \boldsymbol{u})$ for the vorticity equation) is defined by this array.

`coef_4_press_ctl [Name] [Power]`
Coefficient for pressure gradient $-\nabla P$ is defined by this array. Pressure does not appear the vorticity equation which is used for the time integration. But this coefficient is used to evaluate pressure field.

`coef_4_v_diffuse_ctl [Name] [Power]`
Coefficient for viscous diffusion $-\nu\nabla \times \nabla \times \boldsymbol{u}$ is defined by this array.

`coef_4_buoyancy_ctl [Name] [Power]`
Coefficient for buoyancy $-\alpha_T T\boldsymbol{g}$ is defined by this array.

`coef_4_Coriolis_ctl [Name] [Power]`
Coefficient for Coriolis force $-2\Omega\hat{z} \times \boldsymbol{u}$ is defined by this array.

`coef_4_Lorentz_ctl [Name] [Power]`
Coefficient for Lorentz force $\rho_0^{-1}\boldsymbol{J} \times \boldsymbol{B}$ is defined by this array.

`coef_4_composit_buoyancy_ctl [Name] [Power]`
Coefficient for compositional buoyancy $-\alpha_C C\boldsymbol{g}$ is defined by this array.

A.7.3 `induction`

Coefficients of each term in magnetic induction equation are defined in this block.
(Back to `control_MHD`)

`coef_4_magnetic_ctl [Name] [Power]`
Coefficient for evolution of temperature $\dfrac{\partial \boldsymbol{B}}{\partial t}$ is defined by this array.

`coef_4_m_diffuse_ctl [Name] [Power]`
Coefficient for magnetic diffusion $-\eta\nabla \times \nabla \times \boldsymbol{B}$ is defined by this array.

`coef_4_induction_ctl [Name] [Power]`
Coefficient for magnetic induction $\nabla \times (\boldsymbol{u} \times \boldsymbol{B})$ is defined by this array.

A.7.4 composition

Coefficients of each term in composition equation are defined in this block.
(Back to control_MHD)

coef_4_composition_ctl [Name] [Power]
Coefficient for evolution of composition variation $\dfrac{\partial C}{\partial t}$ and advection of heat $(\boldsymbol{u} \cdot \nabla) C$ is defined by this array.

coef_4_c_diffuse_ctl [Name] [Power]
Coefficient for compositional diffusion $\kappa_C \nabla^2 C$ is defined by this array.

coef_4_composition_source_ctll [Name] [Power]
Coefficient for composition source q_C is defined by this array.

A.8 temperature_define

Reference of temperature T_0 is defined in this block. If reference of temperature is defined, perturbation of temperature $\Theta = T - T_0$ is used for time evolution and buoyancy.
(Back to control_MHD)

ref_temp_ctl [REFERENCE_TEMP]
Type of reference temperature is defined by text. The following options are available for [REFERENCE_TEMP].

none Reference of temperature is not defined. Temperature T is used to time evolution and thermal buoyancy.

spherical_shell Reference of temperature is set by

$$T_0 = \frac{1}{(r_h - r_l)} \left[r_l T_l - r_h T_h + \frac{r_l r_h}{r} (T_h - T_l) \right].$$

low_temp_ctl Amplitude of low reference temperature T_l and its radius r_l (Generally $r_l = r_o$) are defined in this block.

high_temp_ctl Amplitude of high reference temperature T_h and its radius r_h (Generally $r_h = r_i$) are defined in this block.

`depth [RADIUS]`
Radius for reference temperature is defined by real.

`temperature [TEMPERATURE]`
Temperature for reference temperature is defined by real.

A.9 time_step_ctl

Time stepping parameters are defined in this block.
(Back to `control_MHD`)
(Back to `control_assemble_sph`)

`elapsed_time_ctl [ELAPSED_TIME]`
Elapsed (wall clock) time (second) for simulation `[ELAPSED_TIME]` is defined by real. This parameter varies if end step `[ISTEP_FINISH]` is defined to `-1`. If simulation runs for given time, program output spectrum data `[rst_prefix].elaps.[process #].fst` immediately, and finish the simulation.

`i_step_init_ctl [ISTEP_START]`
Start step of simulation `[ISTEP_START]` is defined by integer. if `[ISTEP_START]` is set to `-1` and `[INITIAL_TYPE]` is set to `start_from_rst_file`, program read spectrum data file `[rst_prefix].elaps.[process #].fst` and start the simulation.

`i_step_finish_ctl [ISTEP_FINISH]`
End step of simulation `[ISTEP_FINISH]` is defined by integer. If this value is set to `-1`, simulation stops when elapsed time reaches to `[ELAPSED_TIME]`.

`i_step_check_ctl [ISTEP_MONITOR]`
Increment of time step for monitoring data `[ISTEP_MONITOR]` is defined by integer.

`i_step_rst_ctl [ISTEP_RESTART]`
Increment of time step to output spectrum data for restarting `[ISTEP_RESTART]` is defined by integer.

i_step_field_ctl [ISTEP_FIELD]
Increment of time step to output field data for visualization [ISTEP_FIELD] is defined by integer. If [ISTEP_FIELD] is set to be 0, no field data are written.

dt_ctl [DELTA_TIME]
Length of time step Δt is defined by real value.

time_init_ctl [INITIAL_TIME]
Initial time t_0 is defined by real value. This value is ignored if simulation starts from restart data.

A.10 restart_file_ctl

Initial field for simulation is defined in this block.
(Back to control_MHD)

rst_ctl [INITIAL_TYPE]
Type of Initial field is defined by text. The following parameters are available for [INITIAL_TYPE].

- No_data No initial data file. Small temperature perturbation and seed magnetic field are set as an initial field.
- start_from_rst_file Initial field is read from spectrum data file. File prefix is defined by restart_file_prefix.
- Dynamo_benchmark_0 Generate initial field for dynamo benchmark case 0
- Dynamo_benchmark_1 Generate initial field for dynamo benchmark case 1
- Dynamo_benchmark_2 Generate initial field for dynamo benchmark case 2
- Pseudo_vacuum_benchmark Generate initial field for pseudo vacuum dynamo benchmark

A.11 time_loop_ctl

Time evolution scheme is defined in this block.
(Back to control_MHD)

`scheme_ctl [EVOLUTION_SCHEME]`
Time evolution scheme is defined by text. Currently, Crank-Nicolson scheme is only available for diffusion terms.

`Crank_Nicolson` Crank-Nicolson scheme for diffusion terms and second order Adams-Bashforth scheme the other terms.

`coef_imp_v_ctl [COEF_INP_U]`
Coefficients for the implicit parts of the Crank-Nicolson scheme for viscous diffusion `[COEF_INP_U]` is defined by real.

`coef_imp_t_ctl [COEF_INP_T]`
Coefficients for the implicit parts of the Crank-Nicolson scheme for thermal diffusion `[COEF_INP_T]` is defined by real.

`coef_imp_b_ctl [COEF_INP_B]`
Coefficients for the implicit parts of the Crank-Nicolson scheme for magnetic diffusion `[COEF_INP_B]` is defined by real.

`coef_imp_c_ctl [COEF_INP_C]`
Coefficients for the implicit parts of the Crank-Nicolson scheme for compositional diffusion `[COEF_INP_C]` is defined by real.

`FFT_library_ctl [FFT_Name]`
FFT library name for Fourier transform is defined by text. The following libraries are available for `[FFT_Name]`. If this flag is not defined, program searches the fastest library in the initialization process.

`FFTW` Use FFTW

`FFTPACK` Use FFTPACK

`FFT_library_ctl [FFT_Name]`
Loop configuration for Legendre transform is defined by text. The following settings are available for `[Leg_Loop]`. If this flag is not defined, program searches the fastest approarch in the initialization process.

`Inner_radial_loop` Loop for the radial grids is set as the innermost loop

`Outer_radial_loop` Loop for the radial grids is set as the outermost loop

`Long_loop` Long one-dimentional loop is used

A.12 `sph_monitor_ctl`

Monitoring data is defined in this block. Monitoring data output root mean square, average, Gauss coefficients, or specific components of spectrum data which are flagged by `Monitor_On` in `nod_value_ctl` array.
(Back to `control_MHD`)

`volume_average_prefix [vol_ave_prefix]`
File prefix for volume average data `[vol_ave_prefix]` is defined by Text. Program add `.dat` extension after this file prefix. If this file prefix is not defined, volume average data are not generated.

`volume_pwr_spectr_prefix [vol_pwr_prefix]`
File prefix for mean square spectrum data averaged over the fluid shell `[vol_pwr_prefix]` is defined by Text.

Spectrum as a function of degree l is written in `[vol_pwr_prefix])_l.dat`, spectrum as a function of order m is written in `[vol_pwr_prefix]_m.dat`, and spectrum as a function of $(l-m)$ is written in `[vol_pwr_prefix]_lm.dat`. This prefix is also used for the file name of the volume mean square data as `[vol_pwr_prefix].dat`. If this file prefix is not defined, volume spectrum data are not generated and volume mean square data is written as `sph_pwr_volume.dat`.

`layered_pwr_spectr_prefix [layer_pwr_prefix]`
File prefix for mean square spectrum data averaged over each sphere surface `[layer_pwr_prefix]` is defined by Text.

Spectrum as a function of degree l is written in `[layer_pwr_prefix]_l.dat`, spectrum as a function of order m is written in `[layer_pwr_prefix]_m.dat`, and spectrum as a function of $(l-m)$ is written in `[layer_pwr_prefix]_lm.dat`. If this file prefix is not defined, sphere averaged spectrum data are not generated.

`picked_sph_prefix [picked_sph_prefix]`
File prefix for picked spectrum data `[picked_sph_prefix]` is defined by Text. Program add `.dat` extension after this file prefix. If this file prefix is not defined, picked spectrum data are not generated.

`gauss_coefs_prefix [gauss_coef_prefix]`
File prefix for Gauss coefficients `[gauss_coef_prefix]` is defined by Text. Program add `.dat` extension after this file prefix. If this file prefix is not defined, Gauss coefficients data are not generated.

`gauss_coefs_radius_ctl [gauss_coef_radius]`
Normalized radius to obtain Gauss coefficients `[gauss_coef_radius]` is defined by real. Gauss coefficients are evaluated from the poloidal magnetic field at CMB by assuming electrically insulated mantle. Do not set `[gauss_coef_radius]` less than the outer core radius r_o.

`array pick_layer_ctl [Layer #]` List of radial grid point number `[Layer #]` to output spectrum data by integer. If this array is not defined, picked spectrum data are written for all radial grid points.

`array pick_sph_spectr_ctl [Degree] [Order]`
List of spherical harmonics mode l and m of spectrum data to output. `[Degree]` and `[Order]` are defined by integer.

`array pick_sph_degree_ctl [Degree]`
Degrees l to output spectrum data are listed in `[Degree]` by integer. All spectrum data with listed degree l is output in file.

`array pick_sph_order_ctl [Order]`
Order m to output spectrum data are listed in `[Order]` by integer. All spectrum data with listed order m is output in file.

`array pick_gauss_coefs_ctl [Degree] [Order]`
List of spherical harmonics mode l and m of Gauss coefficients to output. `[Degree]` and `[Order]` are defined by integer.

`array pick_gauss_coef_degree_ctl [Degree]`
Degrees l to output Gauss coefficients are listed in `[Degree]` by integer. All Gauss coefficients with listed l is output in file.

`array pick_gauss_coef_order_ctl [Order]`
Orders m to output Gauss coefficients are listed in `[Order]` by integer. All Gauss coefficients with listed order m is output in file.

`nphi_mid_eq_ctl [Nphi_mid_equator]`
Number of grid points `[Nphi_mid_equator]` in longitudinal direction to evaluate mid-depth of the shell in the equatorial plane for dynamo benchmark is defined as integer. If `[Nphi_mid_equator]` is not defined or less than zero, `[Nphi_mid_equator]` is set set number grid as the input spherical transform data.

A.13 num_domain_ctl

Parallelization is defined in this block. Domain decomposition is defined for spectrum data, field data, and Legendre transform.
(Back to `control_sph_shell`)

`num_domain_sph_grid [Direction] [Ndomain]`
Definition of number of subdomains for physical data in spherical coordinate (r, θ, ϕ). Direction `radial` or `meridional` is set in `[Direction]`, and number of subdomains `[Ndomain]` are defined in the integer field.

`num_domain_legendre [Direction] [Ndomain]`
Definition of number of subdomains for Legendre transform between (r, θ, m) and (r, l, m). Direction `radial` or `zonal` is set in `[Direction]`, and number of subdomains `[Ndomain]` are defined in the integer field.

`num_domain_spectr [Direction] [Ndomain]`
Definition of number of subdomains for spectrum data in (r, l, m). Direction `modes` is set in the `[Direction]` field, and number of subdomains `[Ndomain]` are defined in the integer field.

A.14 num_grid_sph

Spatial resolution of the spherical shell is defined in this block.
(Back to `control_sph_shell`)

`truncation_level_ctl [Lmax]`
Truncation level L is defined by integer. Spherical harmonics is truncated by triangular $0 \leq l \leq L$ and $0 < m < l$.

`ngrid_meridonal_ctl [Ntheta]`
Number of grid in the meridional direction `[Ntheta]` is defined by integer.

`ngrid_zonal_ctl [Nphi]`
Number of grid in the zonal direction `[Nphi]` is defined by integer.

`raidal_grid_type_ctl [explicit, Chebyshev, or equi_distance]`
Type of the radial grid spacing is defined by text. The following types are supported in Calypso.

`explicit` Equi-distance grid

`Chebyshev` Chebyshev collocation points

`equi_distance` Set explicitly by `r_layer` array

`num_fluid_grid_ctl [Nr_shell]`
(This option works with `radial_grid_type_ctl` is `explicit` or `Chebyshev`.)
Number of layer in the fluid shell `[Nr_shell]` is defined by integer. Number of grids including CMB and ICB will be (`[Nr_shell]` + 1).

`fluid_core_size_ctl [Length]`
(This option works with `radial_grid_type_ctl` is `explicit` or `Chebyshev`.)
Size of the outer core `[Length]` ($= r_o - r_i$) is defined by real.

`ICB_to_CMB_ratio_ctl [R_ratio]`
(This option works with `radial_grid_type_ctl` is `explicit` or `Chebyshev`.)
Ratio of the inner core radius to outer core `[R_ratio]` ($= r_i/r_o$) is defined by real.

`Min_radius_ctl [Rmin]`
(This option works with `radial_grid_type_ctl` is `explicit` or `Chebyshev`.)
Minimum radius of the domains `[Rmin]` is defined by real. If this value is not defined, ICB becomes inner boundary of the domain.

Max_radius_ctl [Rmax]
(This option works with radial_grid_type_ctl is explicit or Chebyshev.) Maximum radius of the domains [Rmax] is defined by real. If this value is not defined, CMB becomes outer boundary of the domain.

r_layer [Layer #] [Radius]
(This option works with [radial_grid_type_ctl] is explicit.) List of the radial grid points in the simulation domain. Index of the radial point [Layer #] is defined by integer, and radius [Radius] is defined by real.

array boundaries_ctl [Boundary_name] [Layer #]
(This option works with [radial_grid_type_ctl] is explicit.) Boundaries of the simulation domain is defined by [Layer #] in [r_layer] array. The following boundary name can be defined for [Boundary_name].

to_Center Inner boundary of the domain to fill the center.

ICB ICB

CMB CMB

A.15 new_data_files_def

File names and number of processes for new domain decomposed data are defined in this block.
(Back to control_assemble_sph)

num_new_domain_ctl [new_num_domain]
Number of subdomain for new new decomposed data [new_num_domain] is defined by integer.

new_sph_mode_prefix [new_sph_prefix]
File prefix of new spherical harmonics indexing [new_sph_prefix] is defined by text.

new_restart_prefix [new_rst_prefix]
File prefix of new spectrum data [new_rst_prefix] is defined by text.

`delete_original_data_flag [delete_original_data_flag]`
If this flag set to `YES`, original specter data is deleted at the end of program.

A.16 `newrst_magne_ctl`

Parameters to modify magnetic field are defined in this block.
(Back to `control_assemble_sph`)

`magnetic_field_ratio_ctl [ratio]`
Ratio of new magnetic field data to original magnetic field `[ratio]` is defined by real.

Appendix B GNU GENERAL PUBLIC LICENSE

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You

must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

   The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

   A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

   The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

   The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and

to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

   All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

   You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

   Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

   No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

   You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

   You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

   You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

   (a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

   (b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

   (c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

   (d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

   You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

   (a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

   (b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

   (c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

   (d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of

what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

(e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification

itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

(a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

(b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

(c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

(d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

(e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

(f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights

have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

   You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

    Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

    An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

    You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

    A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

    A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be

infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered

work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

    If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

    Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

    The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General

Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

    THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

    IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

    If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local

law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>

Copyright (C) <textyear>  <name of author>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
```

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program>  Copyright (C) <year>  <name of author>

This program comes with ABSOLUTELY NO WARRANTY; for details type 'show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type 'show c' for details.
```

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see `http://www.gnu.org/licenses/`.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read `http://www.gnu.org/philosophy/why-not-lgpl.html`.