ppOpen-HPC:
Open Source Infrastructure for Development and Execution of Large-Scale Scientific Applications on Post-Peta-Scale Supercomputers with Automatic Tuning (AT).

ppohMATHMP

ver. 1.0

User’s guide

License

Change History

The change history lists the changes from version to version in the ppohMATHMP source code. We update this section as we add new features. Note that we tend to update the user's guide at the same time we make changes to ppohMATHMP.

Changes in release 1.0

1. The role of ppohMATHMP

PpohMATHMP ia a coupling library that enables weak coupling on various simulation models, such as an atmospheric model and an ocean model, a seismic model and a structure model. For getting very wide applicability, ppohMATHMP is designed so as that it is independent from grid structure. Instead of grid structure, PpohMATHMP requires a data set called "mapping table". Mapping table is composed of a correspondence table of grid indexes between a send model and a receive model and interpolation coefficients. A subroutine for making a mapping table file is provided by ppohMATHMP API.

2. Current status of ppohMATHMP

Current version of ppohMATHMP is ver.1.0 which targets scalar data exchange. An exchange code of vector data which requires rotation calculation is under development and will be released the next version.

3. Install

3.1 Making ppohMATHMP

Modify ppohMATHMP/src/Makefile and make. Mod files are installed in the directory ppohMATHMPP/include and a library libmathmp.a is generated in ppohMATHMP/lib.

3.2 Making test programn

Coupling test program is provided with ppohMATHMP. For building the test program, modify ppohMATHMP/test/src/Makefile and Mkinclude. After making process, executables "comp1" and "comp2" are made in the same directory.

4. How to run the test program

After compilation of ppohMAHTMP and test program, go to the directory ppohMATHMP/test/run and execute a shell "mpigh2_go".

5. How to use

5.1 Making a mapping table

PpohMATHMP requires grid index correspondence and interpolation coefficients as an input data. Therefore, user must prepare the data set referred as a mapping table. User can generate the mapping table by using of an API subroutine provided by ppohMATHMP as a table below. Subroutine ppoh_MATHMP_write_mapping_table_text generates a text format file and ppoh_MATHMP_write_mapping_table_binary generates a binary format file. The argument send_grid(:) is indexes of send model and recv_grid(:) is indexes of receive model. Coef(:) is interpolation coefficient. The array size of these three arguments must be same.

API name	arguments	meaning
ppoh_MATHMP_write_mapping_table_text	character(len=*) file_name integer send_grid(:) integer recv_grid(:) real(kind=8) coef(:)	table file name send model grid index recv model grid index coefficient
ppoh_MATHMP_write_mapping_table_binary	character(len=*) file_name integer send_grid(:) integer recv_grid(:) real(kind=8) coef(:)	table file name send model grid index recv model grid index coefficient

5.2 Making a configuration file

PpohMATHMP reads a configuration file. Format of the file is as follows. “run_mode” is an execution mode of the coupler. User must select “NORMAL” or “DEBUG”. When nun_mode = “DEBUG”, ppohMATHMP outputs coupling log file.
“send_comp” and “recv_comp” is a name of a send component(model) and a receive component respectively. “send_grid” and “recv_grid” is a name of grid, “map_file_name” is a mapping table file name, “map_file_type” sets type of mapping table file. When map_file_type = T, the table is written in text format, and map_file_type = B, the table is written in binary format.
“send_data” and “recv_data” is a name of send data and receive data respectively. “interval” is exchange interval. “flag” is “SNP” or “AVR”. When flag = SNP, snap shot data is exchanged and flag = “AVR”, average data is exchanged.

```
#rum_mode  : “DEBUG” or “NORMAL”

&nmcoupling
  run_mode = “NORMAL”
&end

#send_comp, recv_comp : send component name and receive component name
#send_grid, recv_grid : send grid name and receive grid name
#map_file_name : mapping table file name
#map_file_type : “T”  (=text) or “B” (=binary)

&nmcoupling
  send_comp = “FDM”, recv_comp = “FEM”
  send_grid = “FDM_GRID”, recv_grid = “FEM_GRID”
  map_file_name = “FDM_to_FEM_mapping.txt”, map_file_type = “T”
&end

#send_data, recv_data : send data name and receive data name
#interval : exchange interval (sec)
#flag : “SNP” (=snap shot) or “AVR” (=average)

&nmcoupling  send_data = “FDM_T”, recv_data = “FDM_T”, interval = 5, flag = “SNP”, &end

&nmcoupling
  send_comp = “FEM”, recv_comp = “FDM”
  send_grid = “FEM_GRID”, recv_grid = “FDM_GRID”
  map_file_name = “FEM_to_FDM_mapping.txt”, map_file_type = “T”
&end

&nmcoupling  send_data = “FEM_T”, recv_data = “FEM_T”, interval = 5, flag = “SNP”, &end
```

5.3 Insert API subroutines into the model code

Typical pattern of API call is as follows.

ppoh_MATHMP_init, ppoh_MATHMP_def_grid and ppoh_MATHMP_end_init are APIs called in initialization process. ppoh_MATHMP_set_time, ppoh_MATHMP_get_data, ppoh_MATHMP_put_data are called in the time integration loop. ppoh_MATHMP_finalize is a finalization API.

```
call ppoh_MATHMP_init(MY_NAME, CONF_FILE, “SYSOUT.PE”)
call ppoh_MATHMP_def_grid(MY_GRID, gnx, gny, gnz, is, ie, js, je, ks, ke)
call ppoh_MATHMP_end_init()
call ppoh_MATHMP_put_data(“FDM_T”, data_t)

do t = 1, 10
  call ppoh_MATHMP_set_time(1)
  call ppoh_MATHMP_get_data(“FEM_T”, recv_data)
  MAIN CALCULATION
  call ppoh_MATHMP_put_data(“FDM_T”, send_data)
end do
call ppoh_MATHMP_finalize()
```

Besides of these APIs above, use must call ppoh_MATHMP_get_mpi_parameters in many case to get a local communicator.

6. REFERRENCE

6.1 Initialization APIs

・ppoh_MATHM_init

This API initializes ppohMATHMP. The subroutine must be called at first

・ppoh_MATHMP_def_grid

This API defines the grid. It has three types of argument, one dimension grid index, tow dimension grid area and three dimension grid area. Global array size nx, ny (and nz) must be same in all processors of the component. PpohMATHMP can defined more than one grid for one component, so user can call ppoh_MATHMP_def_grid more than once.

・ppoh_MATHMP_end_init

This API finalizes coupler initialization process.

API name	arguments	meaning
ppoh_MATHMP_init	character(len=*) comp_name character(len=*) conf_file_name character(len=*) log_file_name	component name configuration file name log file name
ppoh_MATHMP_def_grid	character(len=*) grd_name integer grid_index(:)	grid name grid index
ppoh_MATHMP_def_grid	chacater(len=*) grid_name integer nx, ny integer is, ie, js, je	grid name global array size index of local area
ppoh_MATHMP_def_grid	chacater(len=*) grid_name integer nx, ny, nz integer is, ie, js, je, ks, ke	grid name global array size index of local area
ppoh_MATHMP_end_init		

6.2 Data exchange APIs

・ppoh_MATHMP_set_time

This API is expected to be called at the beginning of the time integration loop. Argument delta_t is delta t of every time step or the index of iteration loop. Data exchange is carried out in this subroutine.

・ppoh_MATHMP_get_data

This API gets the data sent from send component. The array size must be same as the size of local grid defined by the API ppoh_MATHMP_def_grid.

・ppoh_MATHMP_put_data
This API puts the data to the coupler. The array size must be same as the size of local grid defined by the API ppoh_MATHMP_def_grid.

API name	arguments	meaning
ppoh_MATHMP_set_time	integer delta_t	delta t
ppoh_MATHMP_get_data	character(len=*) data_name real(kind=8) data(*)	data name data array
ppoh_MATHMP_put_data	chacater(len=*) data_name real(kind=8) data(*)	data name data array

6.3 Finalization API

・ppoh_MATHMP_finalize
This API finalizes coupling processes. A mpi subroutine MPI_Finalize is called in this API. Therefore, user must not use MPI_Finalize to finalize your mpi processes.

API name	arguments	meaning
ppoh_MATHMP_finalize	NONE	NONE

6.4 Service APIs

・ppoh_MATHMP_get_mpi_parameters
User can get mpi parameters local communicator, local size and local rank through this API. User must use local communicator for intra communication in every component instead of MPI_COMM_WORLD.

・ppoh_MATHMP_put_log
This API outputs log string to the log file defined in ppoh_MATHMP_initialize.

・ppoh_MATHMP_error
This API aborts the process.

・ppoh_MATHMP_write_mapping_txext, ppoh_MATHMP_write_mapping_binary
These APIs create a mapping table file. The subroutine is not called in the coupled

component ordinary, but used in the program making a mapping table.

API name	arguments	meaning
ppoh_MATHMP_get_mpi_parameters	integer my_comm integer my_size integer my_rank	local communicator local mpi size local my mpi rank
ppoh_MATHMP_put_log	character(len=*) log_str	log string
ppoh_MATHMP_error	character(len=*) err_str	error string
ppoh_MATHMP_write_mapping_text	chacater(len=*) file_name integer send_grid(:) integer recv_grid(:) real(kind=8) coef(:)	mapping table file name grid index of send model grid index of recv model coefficient
ppoh_MATHMP_write_mapping_binary	chacater(len=*) file_name integer send_grid(:) integer recv_grid(:) real(kind=8) coef(:)	mapping table file name grid index of send model grid index of recv model coefficient