

ShyBFM v1.0 – Quickstart manual

This short manual explains how to set up the required software, configure, compile, and run the ShyBFM v1.0 coupled circulation–biogeochemistry model. ShyBFM v1.0 code (<https://doi.org/10.5281/zenodo.17702067>) allow to couple advective and diffusive processes modelled by the physical core based on SHYFEM-MPI (Micaletto et al., 2021) with biogeochemical processes of the BFM (Vichi et al., 2023). More details on the coupled system can be found in the main ShyBFM paper of which this document is an attachment.

1. Prerequisites

1.1 System Requirements

- Linux environment (recommended: HPC cluster or workstation)
- Bash shell
- GNU Make (gmake)
- Fortran compiler (Intel recommended for the included architecture preset)
- MPI libraries (if running in mpi mode)

1.2 Required Software and Libraries

The coupled system depends on the source code of the core components of the circulation and biogeochemical models:

- ShyBFM v1.0 core source code
- BFM v. 5.2 source code (<https://www.bfm-community.eu/>)

External software is needed to manage the code compilation (mkmf), the vertical turbulence (GOTM) and the parallel solver (PETSc) and domain decomp (METIS):

- mkmf utility (<https://github.com/NOAA-GFDL/mkmf/>)
- GOTM (General Ocean Turbulence Model; Burchard et al., 1998)
- PETSc 3.7.5 (<https://petsc.org/>)

Additional tools are required for the domain decomposition (METIS) and the conversion of the binary output of the model to NetCDF.

- METIS 5.1.0 (<https://karypis.github.io/glaros/software/metis/overview.html>)
- NetCDF-fortran library (<https://docs.unidata.ucar.edu/netcdf-fortran/current/>)

-Memory (a shell script that can be found in the *tools* directory needed to define simulation name and basin files in the output conversion to NetCDF files).

2. Architecture Configuration

The build process uses an architecture preset that must be provided in the arch folder:

- *arch/architecture.env*
- *arch/architecture.inc*

The .env file define the environmental variables needed to compile and run the code, while the .inc file define the path for the ‘include’ and ‘lib’ directories and provide the flags for the compiler and the pre-processor. An example of architecture files can be found in the ShyBFM code repository.

3. Compilation Procedure

To compile the code a configuration script must be executed:

```
./configure.sh
```

The configuration script automatically creates build and library directories:

- *bin/build*
- *libs/*

3.1 Optional Step — Compile with BFM Coupling

Activate the coupling with BFM in the *configure.sh* setting:

```
use_bfm=true
```

In this case you need to set the path BFMDIR where you have the BFM code and choose a PRESET included in the BFM build.

If *use_bfm* is set to *false*, the coupling is not active and the model run with just the physical core (SHYFEM-MPI).

3.2 Core models compilation

The *configure.sh* script compiles the GOTM code and the main ShyBFM code. If the variable *use_bfm* is set to *true* the script compiles also the BFM code with the preset indicated in the PRESET variable. If the compilation ends without errors, it will produce an executable file in the *bin* directory:

```
-bin/shyfem-mpi.x
```

3.3 Tools compilation

In the *tools* directory another script can be found to compile a set of tools useful to run the model and for post-processing. It is possible to run the script with the command:

```
tools/configure-tools.sh
```

this will compile the code in the *partitioner*, *2nc* and *pre* directories and will produce 5 further executable files.

```
-bin/shypart
```

Shypart is needed to generate partitions for the domain decomposition and the parallel execution of the main code.

-bin/shypre

Shypre must be used to perform a mesh optimization using the Cuthill-McKee algorithm to reduce bandwidth and generating a *.bas* file that is used to run the model.

-bin/ous2nc

-bin/nos2nc

-bin/stm2nc

Ous2nc, *nos2nc* and *stm2nc* are the executables files that convert the model output from its original format to netcdf. We will talk more about these executables in section 7.

4. Pre-processing

There are few steps to do before running the model that consists in generating the partitions with the *shypart* executable and the *.bas* file with *shypre*.

4.1 Domain decomposition

the partitions can be generated executing *shypart* with MPI libraries as follows:

```
mpirun -np <N> bin/shypart namelistPart.str
```

where <N> is the number of partitions and *namelistPart.str* is a namelist with information needed for the partitioning, such the number of partitions (*nn_nparts*) and the vertical weights for the partitions (*nn_wvert*). In *namelistPart.str* the name of the *.bas* must be also indicated, together with the list of the vertical levels.

The structure of the *namelistPart.str* must be:

\$title

Project name

Simulation name

Basin.bas

\$end

\$para

nn_nparts = <N>

nn_wvert = X

\$end

\$levels

1.0 2.0 3.0 ...

\$end

The output of this procedure will be an ASCII file, *part_elems<N>* that contains the domain decomposition. This file must be in the same folder from which the main code is executed.

4.1 Mesh pre-processing

As mentioned in section 3.3, the mesh must be optimized to generate the *.bas* file and run the model. It is possible to do it with the *shypre* executable file as follows:

```
./shypre basin.grd
```

Where the *basin.grd* is the mesh description file.

6. Running the Model

After the *shyfem-mpi.x* executable has been generated, the model can be run with the following command:

```
mpirun -np N bin/shyfem-mpi.x namelist.str
```

the minimum requirement for a successful run is the presence of the partition file *part_elems<N>* and of the *.bas* file. In case the coupling with BFM is active, the transport module of ShyBFM must be activated setting the flag *use_tm = 1* in the *namelist.str* and indicating the path of the namelist of the transport module in *tm_nml*.

6.1 ShyBFM main namelist

The *namelist.str* file is the classic SHYFEM-MPI namelist where numerical parameters, atmospheric forcing, boundary and initial conditions can be set, with additional parameters to manage the transport module. We already mentioned the *use_tm* and *tm_nml*. In addition, tracers open boundary conditions may be defined in the open boundary sections as *trc3dn*. An example of the namelist can be found in the testcase folder *testcases/bench/shyfem.str*.

6.2 Transport module namelist

The transport module namelist (*namelist_tm.nml*) contains parameters and information about the number of variables, frequency and start of output files, horizontal diffusion coefficient and other parameters (*&run_settings*). Moreover, there are sections dedicated to the definition of the tracers used (*&tracer_definition*) and the definition of initial (*&ic_inputdata*) and surface boundary conditions (*&sbc_inputdata*). As already mentioned in section 6.1, the tracers open boundary conditions are defined in the main ShyBFM namelist.

In case the coupling with BFM is active the number and definition of variables are taken by the BFM preset used and the information on the number of variables (*nvar*) in *&run_settings* and definitions in *&tracer_definition* sections are not used.

6.3 BFM namelists

If the coupling with BFM is active, namelists relative to BFM parameters must be added in the main folder. These are generally 5 namelists. A general one (*BFM_General.nml*), two for pelagic ecology and environment (*Pelagic_Ecology.nml*; *Pelagic_Environment.nml*), one for the benthic environment (*Benthic_Environment.nml*) and one for the carbonate dynamics (*Carbonate_Dynamics.nml*).

6.4 Model output

The main model output consists in three files: a *.ous* file that contains sea level and velocity components, a *.nos* files with temperature and salinity and a *.stm* file with all the tracers from the transport module. These files are in the SHYFEM-MPI binary format and can be converted to NetCDF format for an easier data management and analysis.

7. Post-processing

The tools described in section 3.3 (*ous2nc*, *nos2nc* and *stm2nc*) can be used for output conversion to NetCDF. *Ous2nc* is used for *.ous* files, *nos2nc* for *.nos* files and *stm2nc* for *.stm* files. *Memory* script must be used to define the basin and simulation files. For instance, for the *basin.bas* file, if we want to convert the tracers output file *output.stm* we we can do:

```
memory -s output.stm
memory -b basin.bas
```

```
./stm2nc < apnfile.str
```

In case of *.stm* files, if BFM is active, an additional file (*tm_tracers.stm2nc*) must be present with a list of tracers metadata. *tm_tracers.stm2nc* is produced by the model and must be copied and named *info_tracers.stm2nc* in the same location from where the conversion script is called. This will generate a *out.nc* file. The *apnfile.str* contains information about the conversion. It is a six lines file where you indicate reference date, spatial limits and spatial step (if you want a regular file output), initial and final dates of output and frequency of output. The *apnfile.str* should be structured as:

```
YYY-mm-dd::HH:MM:SS # reference date
dx,dy,x0,y0,x1,y1   # spatial steps and limits (empty for output on native unstructured grid)
1                    # flag: empty or 0 for all the period, 1 to choose a specific period
YYY-mm-dd::HH:MM:SS # starting period
YYY-mm-dd::HH:MM:SS # end period
Frequency            # frequency of output
```

If all the spaces are left empty the default values are used, meaning that the whole domain is converted on the original unstructured mesh for the whole period of the simulation file.