SIMSTRAT

1D k-epsilon lake model

**Developer Manual**

1. Introduction 2

2. Code structure 2

3. External files 4

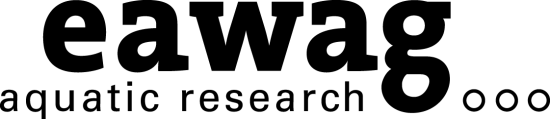
4. Obtaining a FORTRAN compiler 5

5. Installing the FABM libraries 5

6. Installing the HDF5/NetCDF libraries 5

7. Compiling SIMSTRAT 5

8. Further technical improvements 5



March 2015

Swiss Federal Institute of  
Aquatic Science and Technology

# Introduction

This developer manual is intended to be an exhaustive guide detailing the steps to obtain a working copy of the (physical) SIMSTRAT lake model coupled to the (biogeochemical) AED model. It has been written for Windows 7 (64 bit). Even though there are other possibilities, this manual aims at maximizing the use of open-source software.

# Code structure

The code of the model is organized as follows:

kepsmodel\_2014.f90

program **keps**

Calls to **Initialization**, **Form** and **keps\_simulation**

subroutine **keps\_simulation**

Simulation main loop, call to all the functions, coupling with FABM and management of the output

subroutine **Tridiagonal**

Tridiagonal matrix algorithm (solver)

subroutine **Coriolis**

Integration of the Coriolis force in the (x,y) velocities

subroutine **uvEquation**

Solving the advection-diffusion equations for the (x,y) velocities

subroutine **Buoyancy**

Calculation of the stratification coefficient

subroutine **Temperature**

Solving the temperature equation (transport, radiation, heat flux)

subroutine **TransportEquation**

Solving the compound equation (transport, source/sink)

subroutine **cmue\_cn** and **cmue\_qe**

Calculation of model parameters (cmue)

subroutine **StabilityFunctions**

Call to **cmue\_cn** or **cmue\_qe**

subroutine **Seiche**

Distributing the wind forcing into seiche energy

subroutine **Production**

Calculating shear stress and buoyancy energy production

subroutine **TKE**

Solving the turbulent kinetic energy (k) equation

subroutine **Dissipation**

Solving the TKE dissipation rate (ε) equation

subroutine **Advection**

Integration of the inflows and outflows to the model state

subroutine **Form**

Calculation of the morphologic parameters of the basin

keps\_initialization.f90

subroutine **Initialization**

Model initialization, calls to **ParameterList**, **Morph** and **InitCond**

subroutine **ParameterList**

Parameter initialization based on kepsilon.par

subroutine **Morph**

Morphology initialization based on the morphology file

subroutine **Grid**

Vertical grid initialization based on the grid file

subroutine **InitCond**

Applying the initial conditions based on the initial conditions file

subroutine **InitCond\_FABM**

Applying the biochemical initial conditions based on the corresponding files

subroutine **save\_ini**

Preparation of the output grid based on the output depths and output times files

subroutine **check\_advection**

Disabling of advection if all four inflow/outflow files are empty

keps\_utilities.f90

subroutine **Interp**

Linear interpolation (nearest neighbour for values out of range)

subroutine **Interp\_nan**

Linear interpolation (NaN for values out of range)

subroutine **Integrate**

Trapezoidal integration

subroutine **Forcing**

Calculation of all the relevant surface forcing parameters

subroutine **ReadForcing**

Reading the forcing file to obtain current values

subroutine **Absorption**

Reading the absorption file to obtain current values

subroutine **Lateral**

Reading the inflow/outflow files to obtain current values

subroutine **Lateral\_FABM**

Reading the biochemical inflow/outflow files to obtain current values

subroutine **Lateral\_rho**

Reading the inflow/outflow files to obtain current values, assuming that inflow placement is gravity-driven (not manual)

subroutine **Lateral\_FABM\_rho**

Reading the biochemical inflow/outflow files to obtain current values, assuming that inflow placement is gravity-driven (not manual)

# External files

During a simulation, the model opens several files for reading and writing. Until it is closed, each open file is associated to a user-defined file ID (unit). Direct output to the screen is associated to unit 6 on most platforms. A list of the used files is given in Table 1.

|  |  |  |  |
| --- | --- | --- | --- |
| **File ID** | **File description** | **Reference** | **Use** |
| 10 | Parameter | First command argument, or ‘kepsilon.par’ by default | Read once |
| 11 | Morphology | Parameter file, line 4 | Read once |
| 12 | Grid | Parameter file, line 3 | Read once |
| 13 | Initial conditions | Parameter file, line 2 | Read once |
| 14 | Biochemical initial conditions (several files) | Path: parameter file, line 14 | Read once |
| 15 | Output depths | Parameter file, line 8 | Read once |
| 15 | Output times | Parameter file, line 9 | Read once |
| 20 | Surface forcing | Parameter file, line 5 | Read continuously |
| 30 | Light absorption | Parameter file, line 6 | Read continuously |
| 41 | Water inflow | Parameter file, line 10 | Read continuously |
| 42 | Water outflow | Parameter file, line 11 | Read continuously |
| 43 | Temperature input | Parameter file, line 12 | Read continuously |
| 44 | Salinity input | Parameter file, line 13 | Read continuously |
| 45-\* | Biochemical input (several files) | Path: parameter file, line 14 | Read continuously |
| 80 | Physical output (binary) | Parameter file, line 7 | Write continuously |
| 81-93 | Physical output (text files) | Path: parameter file, line 7 | Write continuously |
| 101-\* | Biochemical output (text files) | Path: parameter file, line 15 | Write continuously |

Table 1 – External files used by the model;  
\* means the number depends on which biochemical modules are enabled

# Obtaining a FORTRAN compiler

The GNU Fortran compiler gfortran can be used. It can be obtained as part of the MinGW‑w64 project at <http://sourceforge.net/projects/mingw-w64/files/>, through the installer executable. Once mingw64 is installed, the path to its “bin” directory (inside the installation folder) must be added to the PATH environment variable of the user.

# Installing the FABM libraries

The procedure described at <https://sourceforge.net/p/fabm/wiki/cmake/> has been followed.

The FABM source code can be directly obtained at <http://sourceforge.net/p/fabm/code/ci/master/tree/> (“Download Snapshot” link). The CMake system is available at <http://www.cmake.org/download/> (installer executable or direct zip file). Once CMake has been installed, the path to its “bin” directory (inside the installation folder) must be added to the PATH environment variable of the user.

A “simstrat” folder must be created inside the “src/drivers” directory of the FABM installation folder. Inside this folder, the “fabm\_driver.h” file must be created and written. For SIMSTRAT, which simulates a one-dimensional water column, writing the following is enough:

#define \_FABM\_DIMENSION\_COUNT\_ 1

#define \_FABM\_DEPTH\_DIMENSION\_INDEX\_ 1

#define \_FABM\_VECTORIZED\_DIMENSION\_INDEX\_ 1

#include "fabm.h"

Thereafter, FABM can be build using mingw, according to the following instructions:

* Start CMake (cmake-gui).
* Set “Where is the source code” to the FABM src directory (inside the installation folder).
* Set “Where to build the binaries” to a specific folder inside which CMake will create the build-related files.
* Click "Configure". The first time, CMake asks to select a build system generator: choose "MinGW Makefiles" and check “Use default native compilers”, then click “Finish”.
* Now all configuration variables for the build system are listed and can be changed according to the user preferences. In particular, change “fabm\_host” to simstrat and change the “cmake\_install\_prefix” to a specific folder where the FABM libraries and module files for FORTRAN will be created. After every change, click "Configure" again; additional settings may appear.
* If all configuration variables are set correctly and clicking "Configure" brings nothing new, click “Generate”. This creates the MinGW make files.
* Open a command prompt and go into the build directory defined above.
* Type and run "mingw32-make install" to install the FABM libraries and module files. It may happen that unneeded modules are missing and that this prevents correct installation. In this case, open the “CMakeLists.txt” file (in the “src” directory in the FABM installation folder) and, in the “fabm\_institutes” list, comment out the names that cause trouble. Then, open the “fabm\_library.F90” file at the same location and comment out the lines that cause trouble.

If part of this process has to be redone, close CMake and delete the entire contents of the build directory before starting again from the start.

By using different build directories, one can build FABM with other settings in parallel. A different install directory should also be specified.

Whenever you update the FABM source code to the latest version, you have to rerun "mingw32 make install" in all build directories. It is not needed to rerun CMake itself.

Note: a problem has been encountered with recent versions of the FABM source code, with which the “fabm\_do” routine starts returning NaN values for no obvious reason. Early versions have also been found not to compute vertical velocities of the compounds. The version from the 5th of December 2014 has been used here.

# Installing the HDF5/NetCDF libraries

No success in performing this.

# Compiling SIMSTRAT

Code::Blocks is a text-editor and IDE compatible that supports Fortran. It can be obtained at <http://www.codeblocks.org/downloads/26#windows> (installer executable). When you open Code::Blocks for the first time, you have to select the compiler you want to use; choose gfortran.exe in the “bin” directory of the mingw64 installation folder. Then, go to the “Settings” -> “Compiler…” menu. In the “Linker settings” tab, add the FABM library as a link library (“libfabm.a” in the “lib” directory in the installation folder given when building FABM). In the “Search directories” tab, under “Compiler”, add the path to the FABM modules (“include” directory in the installation folder given when building FABM).

Open the main source file “kepsmodel\_2015.f90” from the SIMSTRAT source code. Clicking “build” compiles the current code and generates the model executable “kepsmodel\_2015.exe”.

# Further technical improvements

As additional improvements to the model package, the following could be done:

* Switching the output from many text files to a single NetCDF or HDF5 file. Libraries compatible with the gfortran compiler must be installed. The advantages would be faster writing and reading of the results and clearer output structure; the disadvantage would be having larger files to deal with.
* The model should still be rendered totally independent from MATLAB. For example, Python scripts for reading the results and processing for PEST could replace the MATLAB ones.
* Using a yaml file instead of a nml file for FABM configuration seems to be the latest standard in FABM. The name of this file could then be optionally given as the second argument when calling kepsmodel.exe from the command prompt.