ProSe-PA 0.77
PROgram SEine with Parallel calculation and data Assimilation

User guide

Shuaitao WANG
Aurélien BORDET
Masihullah HASANYAR
Nicolas GALLOIS
Nicolas FLIPO

Mines Paris/ARMINES - PSL University
Center for geosciences and geoengineering
77 300 Fontainebleau, France

May 2, 2024

Contents

User license v

The ProSe-PA software : General information 1

Install and quick start 3

The ProSe-PA master command file 7

1 General setup: input folders and output folder 9

- **1.1 Block FOLD-IN : Definition of folders containing model input files** 11
- **1.2 Block FOLD-OUT: Definition of folder containing model output files** 11

2 Simulation definition 13

- **2.1 General structure** 16
- **2.2 Block SIMTIME: Temporal aspects of the simulation** 17
- **2.3 Block SIMSET: Settings of the simulation** 17
  - 2.3.1 Block SIMSET-GSET: General settings 18
  - 2.3.2 Block SIMSET-SPE: Definition of biogeochemical species 20
  - 2.3.3 Block SIMSET-MSPE: Definition of macrospecies 20
  - 2.3.4 Block SIMSET-HYD: Hydraulic settings 21
    - 2.3.4.1 General settings of hydraulics 22
    - 2.3.4.2 Initialization of Q and Z: `INITIALIZATION` 23
    - 2.3.4.3 Description of hydraulic network: `NETWORK_DESCRIPTION` 24
      - Definition of singularities 24
      - Definition of river reaches and their connections 26
      - Definition of cross-sections in a river reach 27
      - Definition of Neumann boundary conditions: inflows and outflows 28
  - 2.3.5 Block SIMSET-MTO: Spatiotemporal meteorological data setting 29
    - 2.3.5.1 Settings of spatiotemporal meteorological data 29
    - 2.3.5.2 Reading of spatiotemporal meteorological data 30
  - 2.3.6 Block SIMSET-HT: heat transport settings 31
  - 2.3.7 Block SIMSET-ST: solute transport settings 31
  - 2.3.8 Block SIMSET-BIO: biogeochemistry settings 32
    - 2.3.8.1 General settings 32

2.3.8.2 Definition of simulated layers . . . 33
Definition of water layer . . . 33
Definition of vase layer . . . 34
2.3.8.3 Predefined biogeochemical species . . . 35
Description of biogeochemical species . . . 36
2.3.8.4 Meteorological data settings . . . 53
User-provided data . . . 53
Program calculated data . . . 53
2.3.8.5 SIMSET-DA: settings of parameter ranges for data assimilation . . . 54

3 Model outputs **57**
3.1 `Outputs` **: Syntax overview** . . . 57
3.2 **Output types** . . . 58
3.2.1 `PRINT_PK`: Saving the PK (kilometric point) of singularities and inflows . . . 58
3.2.2 `FINAL_STATE` : Saving the final state of a run . . . 58
3.2.3 `MB_ELEMENTS`: Water balance element by element . . . 59
3.2.4 `TIME_SERIES`: Transversal time series of asked variables . . . 60
3.2.5 `LONGITUDINAL_PROFILES`: longitudinal profiles of asked variables . . . 62
3.2.6 `TUBE_OUTPUT`: Printing tube coordinates and associated hydraulic properties . . . 64
3.2.7 `MASS_BALANCES_BIO`: Printing mass balances of biogeochemical species . . . 65
3.2.7.1 Biogeochemical processes in `MASS_BALANCES_BIO` output files . . . 66
3.2.8 `ENERGY_BALANCES_HEAT`: energy balances . . . 67
3.2.8.1 Energy balances terms in `ENERGY_BALANCES_HEAT` output files . . . 68

Appendixes **69**
A **The** `.COMM` **file : Fully detailed template** . . . 71
B **List of units recognized in the command** `.COMM` **file** . . . 79

References **80**

List of Figures

Figure 2.1 Illustration of the connecting river network in ProSe-PA. 24

List of Tables

Table 1 ProSe-PA install from the main branch of the git repository. 3
Table 2 Installation of ProSe-PA from any branch of the git repository. 4
Table 3 List of all main building blocks of a ProSe-PA command file. 8
Table 4 Simulation types available in ProSe-PA0.7x. 8

Table 2.1 List of all building blocks of a simulation settings 18
Table 2.2 Elements to be defined in the command file 22
Table 2.3 keywords for predefined biogeochemical species in ProSe-PA0.77 35
Table 2.4 Elements to be defined in the command file 53
Table 2.5 Parameters considered in data assimilation 54

Table 3.1 ProSe-PA0.7x output type files. 58
Table 3.2 ProSe-PA0.7x time series variable list. 62

User license

This user guide describes how to run a ProSe-PA 0.7x simulation. It also aims to detail the structure and content of a command file, as well as its sub-files used to define geometries and parameters for a ProSe-PA application.

ProSe-PA 0.7x (*PROgram SEine with Parallel calculation and data Assimilation*) is a software developed at the Center for geosciences and geoengineering, the joint research center of MINES Paris and ARMINES, PSL University, Fontainebleau, France.

Source code is available at https://gitlab.com/prose-pa/prose-pa, under an open source Eclipse Public License (EPL) *v*2.0. It couples several C-ANSI libraries (also under EPL v2.0 licence), available here :

- https://gitlab.com/ghydro
- https://gitlab.com/gtransp
- https://gitlab.com/gutil

The ProSe-PA software : General information

The ProSe-PA is a software for simulating the hydro-thermal-biogeochemical functioning of rivers, particularly heavily urbanised rivers, and streams. It can be used to simulate the anthropisation of environments, through the explicit representation of developments such as navigation dams, sluice gates and river navigation, as well as discharges into the environment, such as those from wastewater treatment plants or combined sewer overflows. It is based on the river continuum concept introduced by Vannote *et al.* (1980) and explicitly simulates the fate of micro-organisms in interaction with macro-elements, including organic carbon.

The current version of ProSe-PA0.7x couples five main libraries to simulate hydraulics, transport, heat energy and biogeochemical function of a river system.

- **libhyd** (**??-A**) simulates river and streams discharge and water height by resolving one-dimensional (1-D) Saint-Venant equations.
- **libtube** simulates pseudo-2D hydraulics of river and streams (discharge, water height) using output of *libhyd*.
- **libttc** (**??-B**), which solves a generic formulation of the advection-dispersion transport equation. It can be applied to a solute and/or heat transport problem. To solve the transport problem, the discharge calculated by *libhyd* library is used.
- **libseb** (**??-C**), computes the overall balance of thermal exchanges between water and the atmosphere based on spatiotemporal meteorological data. The balance is composed of four terms: solar radiation (short-wave), long-wave radiation (atmosphere/environment), evapo-condensation (latent heat of phase transition), and convecto-conduction (sensible heat).
- **librive** (**??-D**) simulates the functioning of the water column in contact with a benthic compartment made up of unconsolidated sediments. It explicitly simulates the growth of micro-organisms in the water column and in the benthic compartment, enabling the carbon, oxygen and nutrient (nitrogen, phosphorus, silica) cycles associated with these biological processes, as well as particulate and dissolved exchanges between the water column and the benthic compartment.

A specificity of ProSe-PA0.7x is the inclusion of data assimilation procedures and memory-shared parallelization processes based on `openmp` for the simulation of transport and biogeochemical processes.

Two filters (particle filter and ensemble Kalman filter) are implemented into ProSe-PA0.77 for assimilating high frequency dissolved oxygen data and estimating microorganism parameters (Wang *et al.*, 2024).

The ProSe-PA0.7x modular approach provides the possibility to activate or disable each library according to the user's specific needs. In addition, the library-based architecture of the software gives it considerable scalability.

PROSE-PA install and quick start

Program install

PROSE-PA can be run under Linux operating system. Before installing PROSE-PA, install the required dependencies. If you use an apt-based system, such as Ubuntu, use the following command :

```
sudo apt install flex bison gfortran make git gcc
```

Then, go to the folder where you want to install PROSE-PA. In `https` mode:

```
git clone https://gitlab.com/prose-pa/prose-pa
cd prose-pa
```

The options for installing from the main branch and all libraries are listed in Table 1.

Command	Installation type
`./make_prose-pa.sh all`	Re-installs and compiles each PROSE-PA library at `$LIB_HYDROSYSTEM_PATH` location. This command compiles `prose-pa0.x` at . location.
`./make_prose-pa.sh PATH`	Installs and compiles each PROSE-PA library at `$PATH` location. Compiles `prose-pa0.x` at . location.
`./make_prose-pa.sh PATH update`	Compiles `prose-pa0.x` at . location, assuming that all libraries needed for compiling are located at `$PATH`, and recompiles them.
`./make_prose-pa.sh`	Re-installs and compiles each library at `./.` location. Compiles `prose-pa0.x` at . location. Equivalent to `./make_Cawaqs.sh ./`

Table 1: PROSE-PA install from the main branch of the git repository.

The options for installing from another branch are referenced in Table 2.

In that specific case, please make sure beforehand that :

- the name of the branch is the same in each library. If the branch does not exist in the library, then the main branch is used by default.
- the branch does exist in the ProSe-PA project. To do so, use the command `git checkout <branch_name>`.

Command	Installation type
`./make_prose-pa_from_branches.sh -b <branch_name>`	Re-installs and compiles each ProSe-PA library at `$LIB_HYDROSYSTEM_PATH` location, using `<branch_name>` as a branch name if found, `main` otherwise. Compiles `prose-pa0.x` at the . location.
`./make_prose-pa_from_branches.sh -b <branch_name> PATH`	Installs and compiles each ProSe-PA library at `$PATH` location, using `<branch_name>` as a branch name if found, `main` otherwise. Compiles `prose-pa0.x` at . location.
`./make_prose-pa_from_branches.sh all`	Calls `./make_prose-pa.sh all`. Re-installs and compiles each library at `$LIB_HYDROSYSTEM_PATH` location. Compiles `prose-pa0.x` at . location.
`./make_prose-pa_from_branches.sh PATH`	Calls `./make_prose-pa.sh all`. Installs and compiles each ProSe-PA library at `$PATH` location. Compiles `prose-pa0.x` at . location.

Table 2: Installation of ProSe-PA from any branch of the git repository.

Parallel computation compilation

The parallel computation is designed differently for transport/biogeochemistry and for data assimilation. In direct calculation mode (without data assimilation), the calculation is parallelized at species level for transport and at cell level for biogeochemistry. In data assimilation mode, an ensemble simulations are running. The parallel calculation is designed thus at simulation (particle) level.

By default, the installation scripts compile a data assimilation mode, `prose-pa0.7x`, which means the parallel computation by particles (see line 144 in `Makefile` or `Makefile`).

```
OPTD = -DOMP $(OPTDOMP) -DCOUPLED_RIVE -DCDA
```

To compile a direct calculation mode (forward model), go to the folder where the ProSe-PA is installed (`../prose-pa/src`), then comment the line 144 (above) and activate the line 143 (below) in `Makefile_tmp` file:

```
OPTD = -DOMP $(OPTDOMP) -DCOUPLED_RIVE
```

Then, re-compile ProSe-PA in the `/prose-pa/src` folder:

```
make -f Makefile_tmp clean
make -f Makefile_tmp all
```

For greater ease, a tip consists in putting the executable `prose-pa0.7x` in the `/usr/local/bin/` folder. Then, we can run `prose-pa0.7x` software everywhere on the system. To do so, from the folder (`/prose-pa/src`) where ProSe-PA is installed:

```
sudo cp prose-pa<version\_number> /usr/local/bin/prose-pa<version\_number>
```

Program launch

Presuming you have copied `prose-pa0.7x` in the `/usr/local/bin/` folder, use the command below to launch ProSe-PA everywhere. Although not mandatory, the `<command_file_name>` file usually has a `.COMM` extension, while the log file uses the `.log` suffix.

```
prose-pa0.7x <path_to_dataset>/<command_file_name> <log_file_name>
```

The ProSe-PA master command file

Generally speaking, the program is divided into 2 distinct phases: an input file reading step, which defines and sets needed simulation computer objects, then followed by a calculation phase. This user guide focuses only on the syntax of the command file, as well as all the information needed to run a full simulation. All the necessary details on the calculation phase (*i.e.* equations, concepts, code structure, etc.) are provided in the ProSe-PA technical guide[1].

General structure

A ProSe-PA command file (usually using the `.COMM` extension) is built using successively embedded setup instruction blocks. Each block is introduced using a specific keyword (Tab. 3) which launches the parameterization of a given library (or simulation module). A command file is divided into two master blocks: `SIMULATION` and `OUTPUTS`. The parts `SIMULATION` consist of a series of sub-blocks defining the parameters and all needed data of a simulation. All blocks and embedded sub-blocks are enclosed by braces, except `INPUT_FOLDERS` and `OUTPUT_FOLDER`. Opening and closing braces indicate the start and end of a block or sub-block, respectively. The blocks are identified by blue texts.

```
INPUT_FOLDERS    = ...                          ! Block FOLD-IN
OUTPUT_FOLDER    = ...                          ! Block FOLD-OUT
SIMULATION       = { ... }                      ! Block SIM
OUTPUTS          = { ... }                      ! Blcok OUTS
```

ProSe-PA keywords are not letter case dependent. Each keyword can be freely written in uppercase, lowercase, or a combination of both. All sentences start with a # or ! are considered as comments, and will not be read by the program.

[1] Available at : **To be updated when done**

Table 3: List of all main building blocks of a ProSe-PA command file.

Main block	Master keyword	Description	Guide section
FOLD-IN	`INPUT_FOLDERS`	Lists all directories containing ProSe-PA input files	1.1
FOLD-OUT	`OUTPUT_FOLDER`	Defines the folder in which the application outputs will be stored	1.2
SIM	`SIMULATION`	Defines a simulation	2
OUTS	`OUTPUTS`	Defines the properties of the simulation output files	3.1

Simulation types

The current ProSe-PA0.77 allows four main types of simulation, as listed in table 4. It is important to note that, for each type of simulation, the order of the main blocks cannot be changed.

Table 4: Simulation types available in ProSe-PA0.7x.

Simulation type	Settings
Hydraulics only	Sec. 2.3.4
Hydraulics and transport (heat, solute)	Sec. 2.3.6 and Sec. 2.3.7
Hydraulics, transport (heat, solute), biogeochemistry	Sec. 2.3.8
Hydraulics, transport (heat, solute), biogeochemistry with data assimilation	Sec. A

CHAPTER 1

General setup: input folders and output folder

Content

1.1 Block FOLD-IN : Definition of folders containing model input files 11
1.2 Block FOLD-OUT: Definition of folder containing model output files 11

1.1 Block FOLD-IN : Definition of folders containing model input files

An example of the syntax of the model input folder definition block is shown below. Although not mandatory, it is common to store input files in dedicated folders according to the properties. These paths define the folders that contain the input file of the model. Up to 25 paths can be defined. It is recommended to set the absolute path to the folders that contain the input files. Subsequently, for each input file defined in the command file, the existence of the input file will be tested in each listed folder until it is found. If not, the program will stop and display an error message.

```
Input_folders    = /my_home/MY_PROSEPA_DATA/MY_MODEL/Cmd_files/
                 = /my_home/MY_PROSEPA_DATA/MY_MODEL/DATA_HYD/
                 = /my_home/MY_PROSEPA_DATA/MY_MODEL/DATA_MTO/
                 = /my_home/MY_PROSEPA_DATA/MY_MODEL/DATA_BIO/
                 = /my_home/MY_PROSEPA_DATA/MY_MODEL/DATA_HYD_BIO/
                 = ...
```

1.2 Block FOLD-OUT: Definition of folder containing model output files

Similarly, an output folder need to be defined. If the folder doesn’t exist, the program will create it. An example of the syntax of the model output folder definition is shown below.

```
Output_folder    = /my_home/MY_PROSEPA_OUPUTS/
```

CHAPTER 2

Simulation definition

Content

2.1 General structure . . . 16

2.2 Block SIMTIME: Temporal aspects of the simulation . . . 17

2.3 Block SIMSET: Settings of the simulation . . . 17

2.3.1 Block SIMSET-GSET: General settings . . . 18

2.3.2 Block SIMSET-SPE: Definition of biogeochemical species . . . 20

2.3.3 Block SIMSET-MSPE: Definition of macrospecies . . . 20

2.3.4 Block SIMSET-HYD: Hydraulic settings . . . 21

2.3.4.1 General settings of hydraulics . . . 22

2.3.4.2 Initialization of Q and Z: `INITIALIZATION` . . . 23

2.3.4.3 Description of hydraulic network: `NETWORK_DESCRIPTION` . . . 24

Definition of singularities . . . 24

Definition of river reaches and their connections . . . 26

Definition of cross-sections in a river reach . . . 27

Definition of Neumann boundary conditions: inflows and outflows . . . 28

2.3.5 Block SIMSET-MTO: Spatiotemporal meteorological data setting . . . 29

2.3.5.1 Settings of spatiotemporal meteorological data . . . 29

2.3.5.2 Reading of spatiotemporal meteorological data . . . 30

2.3.6 Block SIMSET-HT: heat transport settings . . . 31

2.3.7 Block SIMSET-ST: solute transport settings . . . 31

2.3.8 Block SIMSET-BIO: biogeochemistry settings . . . 32

2.3.8.1 General settings . . . 32

2.3.8.2 Definition of simulated layers . . . 33

Definition of water layer . . . 33

Definition of vase layer . . . 34

2.3.8.3 Predefined biogeochemical species . . . 35

Description of biogeochemical species 36
2.3.8.4 Meteorological data settings 53
User-provided data 53
Program calculated data 53
2.3.8.5 SIMSET-DA: settings of parameter ranges for data assimilation 54

2.1 General structure

A simulation is defined by the block SIM, initiated by the simulation keyword `Simulation`, is divided into two sub-blocks and a run name:

- `run_name`, to define the name of the simulation
- `time`, to define the temporal aspects of the simulation (time format, range, time steps),
- `settings`, to set a simulation type and all necessary parameters/data.

```
SIMULATION = {                                     ! Block SIM
    simulation_name

    TIME     = { ... }                             ! Block SIMTIME

    SETTINGS = { ... }                             ! Block SIMSET
}
```

ProSe-PA keywords are not letter case dependent. Each keyword can be freely written in uppercase, lowercase, or a combination of both. All sentences start with a # are considered as comments, and will not be read by the program.

2.2 Block SIMTIME: Temporal aspects of the simulation

The temporal aspects of the simulation are defined in the TIME block.

```
TIME   = {                                               ! Block SIMTIME
    date_format   = FR_TS
    year0_julian = 2007
    t_init        = [d] 0.      # 01/01/2007 00:00:00
    t_end         = [d] 30.
    dt            = [min] 15.
    nstep_da      = 1
}
```

The parameters are explained below:

- date_format: Format of the calendar date (FR_TS/US_TS) when using calendar date time series in input data. Two formats of calendar date are readable in PROSE-PA0.7x: FR_TS (dd/mm/yyyy HH:MM:SS) or US_TS (mm/dd/yyyy HH:MM:SS). The HH:MM:SS is not mandatory. Set to FR_TS by default.
- year0_julian: The calendar date time series need to be convert into Julian day in the calculation phase. The syntax year0_julian defines the year from which the Julian days are calculated, with the first day of the year0_julian (for example, 01/01/2007 00:00:00) as 0. Set to 1850 by default.
- t_init: The simulation starting time. It can be either a Julian day or a calendar date. For Julian day, time unit ([d] for day or [h] for hour or [min] for minute or [s] for second) must be given following the value.
- t_end: The simulation ending time. The setting of t_end is the same as t_init.
- dt: The time step of calculation. A unit ([d] for day or [h] for hour or [min] for minute or [s] for second) and a value must be given.
- nstep_da: The number of steps to calculate the data assimilation time step ($dt_{da} = nstep_{da} \times dt$). Set to 1 by default.

2.3 Block SIMSET: Settings of the simulation

The settings of the simulation is initiated by the keyword Settings, including general setting, species, hydraulics, transport, biogeochemistry etc.

```
SETTINGS = {                                             ! Block SIMSET
    ...                                                  ! Block SIMSET-GSET
    species         = { ... }                            ! Blcok SIMSET-SPE
    macrospecies    = { ... }                            ! Blcok SIMSET-MSPE
    hydraulic       = { ... }                            ! Block SIMSET-HYD
    meteo           = { ... }                            ! Block SIMSET-MTO
    heat_transport = { ... }                             ! Block SIMSET-HT
    transport       = { ... }                            ! Block SIMSET-ST
    biology         = { ... }                            ! Block SIMSET-BIO
    param_range_da = { ... }                        ! Block SIMSET-DA
}
```

Table 2.1: List of all building blocks of a simulation settings

Blocks	Master keyword	Description	Guide section
SIMSET-GSET	-	Gerneral setting of the simulation	Sec. 2.3.1
SIMSET-SPE	`species`	Defines the simulated species	Sec. 2.3.2
SIMSET-MSPE	`macrospecies`	Defines the macrospecies	Sec. 2.3.3
SIMSET-HYD	`hydraulic`	Sets all parameters and data used for hydraulic simulation	Sec. 2.3.4
SIMSET-MTO	`meteo`	Sets spatiotemporal meteorological data for heat simulation	Sec. 2.3.5
SIMSET-HT	`heat_transport`	Sets heat transport	Sec. 2.3.6
SIMSET-ST	`transport`	Sets solute transport	Sec. 2.3.7
SIMSET-BIO	`biology`	Sets biogeochemistry	Sec. 2.3.8
SIMSET-DA	`param_range_da`	Sets parameter ranges for data assimilation	Sec. 2.3.8.5

2.3.1 Block SIMSET-GSET: General settings

General simulation parameters are defined directly after the definition of SIMSET block, which means after the open brace. The order of keywords can be changed.

```
regime            = steady
hyd               = yes
ttc               = no
rive              = no
tube              = no
dynamic_T         = no
da                = no
da_method         = PF
num_thread        = 1
num_particules    = 1
num_thread_par    = 1
num_tube_def      = 5
alpha_da          = 0.3
error_obs         = 0.10
s_percent         = 0.10
random_walk       = loop
solver            = sp_prose
default_t_inflows = [°C] 20.
nb_comp_phy       = 3
dbo_oxy           = [molO2/molC] 1.0
```

It includes :

- `regime`: The regime of the simulation: steady-state or transient-state. This parameter can be set to `TRANSIENT` or `STEADY`. It is optional. Set to `STEADY` by default.
- `hyd`: Boolean option (`YES/NO`) to launch hydraulics calculations (call *libhyd* library). Set to `YES` by default.
- `ttc`: Boolean option (`YES/NO`) to launch solute transport-based calculations (call *libttc* library). Set to `NO` by default. The transport calculations necessity the hydraulics calculations (discharge).
- `rive`: Boolean option (`YES/NO`) to launch biogeochemical calculations (call *librive* library). Set to `NO` by default. The biogeochemical calculations necessity the hydraulics and transport calculations (water height, water velocity, advection-dispersion).
- `tube`: Boolean option (`YES/NO`) to launch pseudo-2D hydraulics calculations (call *libtube* library). The Set to `NO` by default. It needs the hydraulics calculations.
- `dynamic_T`: Boolean option (`YES/NO`) to launch heat transport-based calculations (call *libttc* library). Set to `NO` by default. The transport calculations necessity the hydraulics calculations (discharge).
- `da`: Boolean option (`YES/NO`) to launch oxygen data assimilation based calculations. Set to `NO` by default.
- `da_method`: The data assimilation method, useful only when keyword `da` set to `YES`. Two methods are implemented: particle filter `PF` and ensemble Kalman filter (`PF`). Set to `PF` by default.
- `num_thread`: The number of threads for parallel computation with `OpenMP`. It is designed for transport at species level and for biogeochemisrty at cell level. Set to `1` by default. It should be 1 when running a deterministic (direct) simulation.
- `num_particules`: The number of ensemble simulations when data assimilation is active. Set to `1` by default.
- `num_thread_par`: The number of threads for parallel computation with `OpenMP`. It is designed for data assimilation at particle level (`num_particules`). Set to `1` by default.
- `alpha_da`: Used for particle filter (`PF`), to define a minimum effective sample size (`alpha_da` × `num_particules`) below which a resampling procedure proceeds. Set to 0.3 by default.
- `error_obs`: To define the standard deviations of observation errors (σ = `error_obs`$\times [O_2]_{obs}$). Set to 0.01 by default.
- `s_percent`: Used for particle filter (`PF`), to define the standard deviation of parameter perturbation (random walk) after resampling procedure ($\eta \sim N(0, s_{percent}(max - min))$). Set to 0.10 by default.
- `random_walk`: Option of random walk method (`LOOP/NOT_LOOP`). Set to `LOOP` by default.
- `solver`: Option of solver library. Two libraries are available: `SP_PROSE` (call *sparse* library) and `GC_PROSE` (call *libgc* library). Set to `SP_PROSE` by default. Parallel computation works only with `SP_PROSE`.
- `default_t_inflows`: User defined water temperature for inflows where temperature data is not available. It will replace the `CODE_TS` or -9999 defined in the time series data files. Set to 20 °C by default.
- `nb_comp_phy`: number of constituents for phytoplankton. `RIVE` model is based on three constituents (F, S, R, (Wang *et al.*, 2024)). Set to 3 by default.

- dbo_oxy: oxygen/carbon share for oxygen carbon conversion in respiration and photosynthesis processes. Set to 1.2 molO2/molC. However, it is recommended to use 1.0 molO2/molC.

2.3.2 Block SIMSET-SPE: Definition of biogeochemical species

The block SIMSET-SPE, initiated by SPECIES, defines all simulated biogeochemical species including biogeochemical processes, related parameters' values. It is possible to define the different species properties in water layer (WATER) and sediment layer (VASE). If species are defined in only 1 compartment, and other compartments are present, they will have the same properties in these other compartments.

```
SPECIES = {                                  ! Blcok SIMSET-SPE
    WATER = {
        #PARTICULATE
        include phy
        include pom
        include hb
        include bif
        include nb
        include mes
        include zoo
        ...

        #DISSOLVED
        include o2
        include mod
        include no3
        include nh4
        ...
    }
    VASE  = { ... }
}
```

It includes:

- WATER: Definition of species in water layer.
- VASE: Definition of species in sediment layer.
- include: To open a species file (such as phy) and read the properties of the species.

This syntax of species files will be detailed in section 2.3.8.3.

The particulate species must be defined before dissolved species if adsorption process is simulated!

2.3.3 Block SIMSET-MSPE: Definition of macrospecies

The macrospecies are indeed other species that exist in the river but are not considered under the species section. This could be parameters of the repartition of the total organic carbon (TOC) that could be used to divide TOC first of all into dissolved organic carbon (DOC) and particulate organic carbon (POC) and then into their smaller fractions according to biodegradability. For better understanding of these parameters, please visit here

An example of macrospecies (section 2.3.3) is below. The total organic carbon (TOC) can be treated as a macrospecies and characterized by its phases (dissolved and particulate) or biodegradability (fast, slowly, refractory).

```
macrospecies = {
TOC = {
    total_organic_carbon                       ! macrospecies name
    related_to MOD MOP , threshold = 0.7   ! fraction of dissolved phase
    share_mop = {
        biodegradable      = {
            val     = 0.3                      ! fraction of biodegradable MOP
            ! range = { 0.15, 0.5 }            ! a value or a range
        }
        fast_biodegradable = { val = 0.7 } ! fraction of fast biodegradable MOP
    }
   share_mod = {
        biodegradable      = {
            val     = 0.3                      ! fraction of biodegradable MOD
            ! range = { 0.1 , 0.5 }            ! a value or a range
        }
        fast_biodegradable = { val = 0.7 } ! fraction of fast biodegradable MOD
    }
}
}
```

The list of parameters which could be defined under this item is in Table: 2.2.

2.3.4 Block SIMSET-HYD: Hydraulic settings

The hydraulic settings are initiated by the keyword HYDRAULIC. The block SIMSET-HYD can be divided in three parts: general settings, initialization, and network description.

```
HYDRAULIC = {
    ndim                = 1
    calculate_curvature = no
    curvature           = 0
    dx                  = [m] 500
    eps_Q               = [m^3/s] 0.000001
    eps_Z               = [m] 0.000001
    global_strickler    = 20.
    dz                  = [m] 0.1
    upstream_Hmin       = [m] 0.
    downstream_Hmax     = [m] 20.
    theta               = 0.9
```

Table 2.2: Elements to be defined in the command file

	Macrospecies parameters		
name	description	unit	value
	Parameters of TOC macrospecie		
`threshold`	value of ratio between MOD and TOC (t)		
`share_mop`	definition of parameters for repartition of MOP		
`share_mod`	definition of parameters for repartition of MOD		
`biodegradable`	the ratio between biodegradable portion of MOP (BMOP) and MOP itself (b_2 = BMOP/MOP; similarly its the ratio between biodegradable portion of MOD and MOD itself (b_1 = BMOD/MOD)		
`fast_biodegradable`	the ratio between fast biodegradable portion of MOP (MOP1) and BMOP (s_2 = MOP1/BMOP; similarly its the ratio between fast biodegradable portion of MOD (MOD1) and BMOD (s_1 = MOD1/BMOD		
`val`	mean value of the parameter		
`range`	variation range of the parameter		

```
    schem_type          = ST_VENANT
    initialization      = { ... }
    network_description = { ... }
}
```

2.3.4.1 General settings of hydraulics

It includes,

- `ndim`, dimension of calculation, 1D for the moment. Set to 1 by default.
- `calculate_curvature`, Boolean option `YES/NO` to define the calculation of curvature by *libhyd* library if the cross-sections are georeferenced. Set to `NO` by default.
- `curvature`, to define overall curvature value. Set to 0 by default.
- `dx`, longitudinal discretization ([unit] value). No default value. It is optional.
- `eps_Q`, convergence of precision of discharge Q. Set to 10^{-6} [m^3 s^{-1}] by default.
- `eps_Z`, convergence of precision of water height Z. Set to 10^{-6} [m] by default.
- `global_strickler`, to define global value of the strickler friction coefficient. Set to 40 by default.
- `dz`, vertical space step. Set to 0.1 [m] by default
- `upstream_Hmin`, minimum water height upstream of the network. Set to 0 [m] by default.
- `downstream_Hmax`, maximum water height downstream of the network. Set to 10 [m] by default.
- `theta`, degree of implicitness of the equations' discretisation. Set to 0.9 by default.
- `schem_type`, numerical method for equation solving. No default value.

2.3.4.2 Initialization of Q and Z: `INITIALIZATION`

The initialization of hydraulics (discharge and water level) starts with the keyword `INITIALIZATION` and is optional. If the initial discharge and water level are not provided, ProSe-PA calculates approximate Q and Z according to the boundary conditions. Normally, a steady-state simulation is run first to get the converged discharge and water level, which can be used to initialize a transient state simulation.

```
HYDRAULIC = {
    ...
    initialization      = {
        init_Z_file         = include ini_Z
        init_Q_file         = include ini_Q
    }
    ...
}
```

It includes,

- `init_Z_file`, to initialize the water level at cell center from the ASCII text file `init_Z` which contains two columns separated by space: `[cell_identifier] [water_level]`. The unit of water level is [m].

```
! CELL_ID    Z
        0    9.937318
        1    9.930820
        ...  ...
        199  9.000000
```

- `init_Q_file`, to initialize the discharge at cell faces from the ASCII text file `init_Q` which contains two columns separated by a space: `[face_identifier] [discharge]`. The unit of `Q` is [m^3 s^{-1}].

```
! FACE_ID    Q
        0    95.000000
        1    95.000000
        ...  ...
        201  95.000000
```

- `include`, to define a filename which is read by the program.

❗ The identifier of cell/face starts at 0!

2.3.4.3 Description of hydraulic network: NETWORK_DESCRIPTION

The NETWORK_DESCRIPTION sub-block, which defines the connecting hydraulic network as well as inflows. The river network is divided into river reaches which are connected by singularities (Fig. 2.1). Each river reach is discretized in model cells (called also as elements), according to the dx (section 2.3.4.1).

```
HYDRAULIC = {
    ...
    initialization       = { ... }
    network_description = {
        singularities        = { include sings_example }
        reaches              = { include reaches_example }
        profiles             = { include profiles_example }
        inflows              = { include inflows_example }
    }
}
```

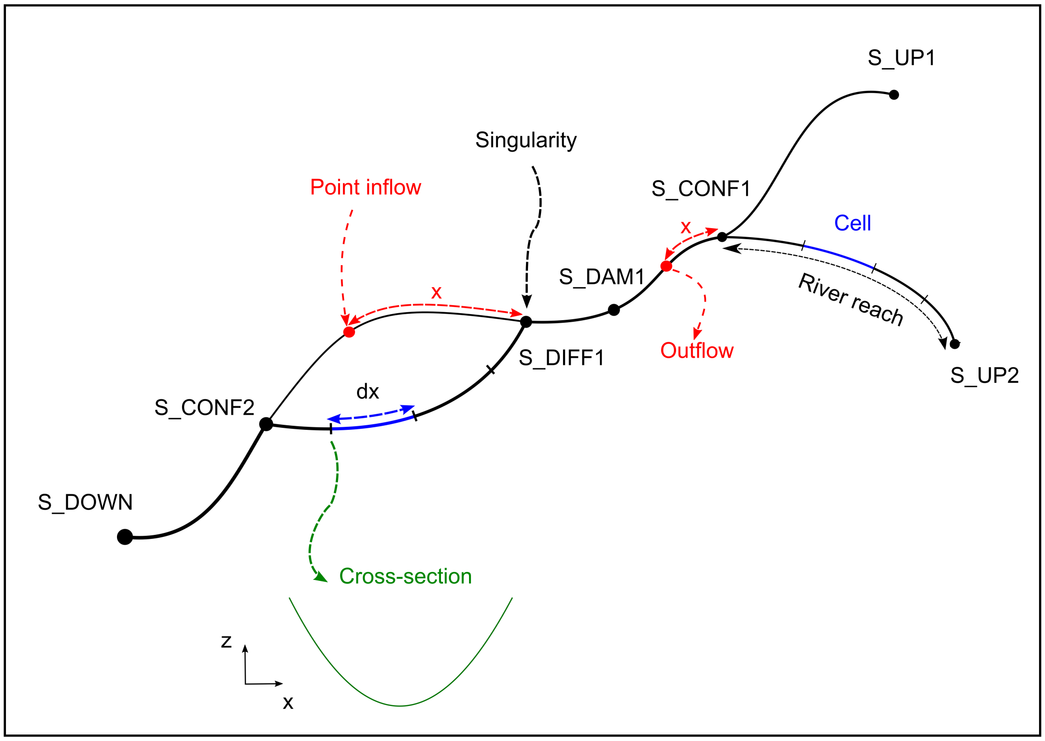

Figure 2.1: Illustration of the connecting river network in ProSe-PA.

Definition of singularities

The singularity (black points, Fig. 2.1) in ProSe-PA represents a section point where the geometry of river reach changes fast or where presents a hydraulic works (weirs, gates). This changing in geometry results in changing of water velocity. The definition of singulartities is initiated by the keyword `SINGULARITIES`. The syntax of singularities is shown below. A singularity is initiated by singularity name and terminated by a semicolon: `[singularity_name;]`. Upstream, "continuous" singularities and confluences or diffluences without hydraulic structures are simply named (such as `S_UP1;`, `S_CONF1;`, `S_DIFF1;`).

```
! In file sings_example
S_UP1;
S_UP2;
S_CONF1;
S_DAM1 {
    ...
};
S_DIFF1;
S_CONF2;
S_DOWN {
   type     = WATER_LEVEL
   pk       = [km] 100
   z(t)     = {
        [d] [m]
        0.0 8.38
   }
};
```

A hydraulic structure is defined by specifying its type `HYDWORK` and operating lows. It can be `weir` and/or `gate`.

```
S_DAM1 {
    type            = HYDWORK
    holler          = 0.21
    hyd_structures  = {
        weir = {
            mu    = 0.6
            width = [m] 5.
            Zw(t) = {
                [d] [m]
                 0. 19.9
            }
        }

        gate = {
            mu    = 0.6
            width = [m] 5.
            H(t)  = {
                [d] [m]
                 0. 0.2
            }
        }
    }
};
```

It includes,

- type, the type of the singularity. It can be either WATER_LEVEL or HYDWORK. WATER_LEVEL specifies Dirichlet boundary condition (constant water level) and HYDWORK for weir and gate.
- holler, Holler's coefficient for reoxygenation. 0.21 is recommended.
- hyd_structures, to defined the properties of the hydraulic structure when type is HYDWORK.
- weir, to defined a weir with the discharge constant for the weir (mu), width of the weir (width), height of the weir (Zw(t)).
- gate, to defined a gate with the discharge constant for the gate (mu), width of the gate (width), height of the gate (H(t)).

A Dirichlet boundary condition (constant water level) is defined by the type keyword WATER_LEVEL and water level (Z(t)). When it is located at the downstream of river network (for example, S_DOWN), a PK (point kilometric) value must be provided. This PK value is used to calculate the locations of all other singularities.

```
S_DOWN {
   type     = WATER_LEVEL
   pk       = [km] 100
   z(t)     = {
        [d] [m]
        0.0 9.00
   }
};
```

Definition of river reaches and their connections

The definition of river reaches is initiated by the keyword REACHES. A river reach is construct by linking two singularities from upstream to downstream. The syntax is [S_UPSTREAM -> S_DOWNSTREAM BRANCH_NUM;]. The definition of strickler friction coefficient (strickler) as function of discharge ([m^3 s^{-1}]) is possible. If strickler is not defined here, the global strickler friction coefficient defined in general settings (section 2.3.4.1) is applied.

```
! In file reaches_example
S_UP1 -> S_CONF1 1;
S_UP2 -> S_CONF1 1;
S_CONF1 -> S_DAM1 1 {
   strickler = {
      300 27
      400 33
      500 31.5
   }
};
S_DAM1 -> S_DIFF1 1;
```

```
S_DIFF1 -> S_CONF2 1;
S_DIFF1 -> S_CONF2 2;
S_CONF2 -> SDOWN 1;
```

The numbering of branches (BRANCH_NUM) are from left to right and start at 1.

Definition of cross-sections in a river reach

The definition of cross-sections is initiated by the keyword PROFILES. A cross-section is described by its name (S0, S1, S2), the upstream singularity (S_UP1) and the number of the branch to which the ross-section belongs (1). The syntax of a cross-section is [SECTION_NAME <- UPSTREAM_SINGULARITY BRANCH_NUM { ... };].

```
! In file profiles_example
S0 <- S_UP1 1
{
    dx   = [m] 1000                      ! Distance between S0 and S_UP1
    type = ABSC Z
        0   10
        10  4.7
        110 4.7
        120 10
};

S1 <- S_UP1 1                            ! Distance between S1 and S0
{
    dx   = [m] 800
    type = ABSC Z
        0   10
        10  4.6
        110 4.6
        120 10
};

S2 <- S_UP1 1                            ! Distance between S1 and S2
{
    dx   = [m] 900
    type = X Y Z
        include geoS2
};
```

It includes,

- dx, the longitudinal distance to the upstream cross-section or upstream singularity. It is different from the longitudinal discretization defined in section 2.3.4.1.

- `type`, the type of geometry data. It can be abscissa/ordinate system (`ABSC Z`) or georeferenced data (`X Y Z`).

```
! In file geoS2
!             X               Y    Z
605346.451193 123700.467180 4.5
605353.380000 123700.120000 4.5
605358.070000 123699.890000 4.5
...
```

The geometry data is organized from left river bank to right river bank!

Definition of Neumann boundary conditions: inflows and outflows

The Neumann boundary conditions here represent imposed flows boundary conditions. The block inflows is initiated by the keyword `INFLOWS`. The syntax is `[FLOW_NAME : UPSTREAM_SINGULARITY BRANCH_NUM { ... };]`.

```
! In file inflows_example
SEINE : S_UP2 1 {
    x      = 0.000000 [km]
    y      = 0.000000
    type   = UPSTREAM_INFLOW
    q      = { [d] [m^3/s] 0. 95 }
    mes 1 = { [d] [mg/l] 0. 5. }
    mod 1 = { [d] [mgC/l] 0. 2.0 }
    ...
    T = { [d] [°C] 0. 25 }
};
...
inflow1 : S_DIFF1 2 {
    x      = 15.3 [km]
    y      = 1.000000
    type   = EFFLUENT
    q      = { [d] [m^3/s] 0. 5 }            ! positive discharge for inflows
    mes 1 = { [d] [mg/l] 0. 5. }
    mod 1 = { [d] [mgC/l] 0. 2.0 }
    ...
    T = { [d] [°C] 0. 25 }
};
...
outflow1 : S_CONF 1 {
    x     = 7.5 [km]
    y     = 0.000000
    type = EFFLUENT
    q    = { [d] [m^3/s] 0. -4 }          ! negative discharge for outflows (uptake)
};
```

It includes,

- x, longitudinal distance to the upstream singularity.
- y, transversal position, 0 for left bank and 1 for right bank. type, type of the flow. It can be UPSTREAM_INFLOW, INFLUENT or EFFLUENT. UPSTREAM_INFLOW describes the flow entering the river network at upstream. INFLUENT identifies a river tributary not explicitly described. EFFLUENT identifies a discharge or withdrawals (negative discharge).
- q, time series of discharge.
- SPECIES SPECIES_NUM, associated concentrations of biogeochemical species (such as mes 1, mod 1) defined in section 2.3.2. For a uptake, we don't need to define the concentrations. It treated as the concentrations in the river.
- T, associated water temperatures. For a uptake, we don't need to define the water temperatures. It treated as the water temperatures in the river.

> Conceptually, in the code, there is no difference in treatment between the two types INFLUENT and EFFLUENT. This distinction is not clear. We need to rework the algorithm slightly and distinguish EFFLUENT into two types: EFFLUENT and UPTAKE, so that the user no longer has to define negative discharges for withdrawals. In addition, all these cases must be clearly distinguished in the columns of the mass balance output files.

2.3.5 Block SIMSET-MTO: Spatiotemporal meteorological data setting

The block SIMSET-MTO is initiated by the keyword METEO. In this block, the ProSe-PA reads the spatiotemporal meteorological data (air temperature, atmospheric pressure, shortwave radiation, longwave radiation, relative humidity, wind speed). The reading procedure is designed for SAFRAN-type data, which means the river network is covered by SAFRAN cells.

```
METEO = {
    met_corresp = forcing_to_reach_corresponding.txt
    met_data    = {
        folder      = /my_home/MY_METEO_DATA/
        year0_meteo = 1850
        forcings    = { ... }
   }
}
```

2.3.5.1 Settings of spatiotemporal meteorological data

It includes,

- `met_corresp`, defined a filename. In the ASCII file, we link each river reach to SAFRAN cell.

```
! In file forcing_to_reach_corresponding.txt
UPSTREAM_SING DOWNSTREAM_SIN BRANCH_NUM SAFRAN_CELL_ID
S_UP1 S_CONF1 1 1563
S_UP2 S_CONF1 1 1563
S_CONF1 S_DAM1 1 1453
S_DAM1 S_DIFF1 1 1453
S_DIFF1 S_CONF1 1 976
S_DIFF1 S_CONF1 2 976
S_CONF2 SDOWN 1 1 890

```

> The separator in this `met_corresp` file must be a space. The header is mandatory and will be skipped in reading procedure.

- `met_data`, initiate data settings.
- `folder`, the path where store all the meteorological data
- `year0_meteo`, the year from which the Julian days are calculated, with the first day of the `year0_meteo` (for example, 01/01/1840 00:00:00) as 0. Set to 1850 by default.

2.3.5.2 Reading of spatiotemporal meteorological data

The keyword `forcings` initiates meteorological data reading.

```
forcings = {
    T_air   = TMPH_case_study.txt
    P_atm   = PRESSURE_grid_safran.dat
    H_sw    = SWDH_case_study.txt
    H_lw    = LWDH_case_study.txt
    Humid   = HRLH_case_study.txt
    U_wind  = VNTH_case_study.txt
}
```

It includes,

- `T_air`, air temperature, [K].
- `P_atm`, atmospheric pressure, [Pa]
- `H_sw`, shortwave radiation, [W m^{-2}].
- `H_lw`, longwave radiation, [W m^{-2}].
- `Humid`, relative humidity, [%].
- `U_wind`, wind speed at 2 meters high, [m s^{-1}]

The atmospheric data is the mean absolute value, and contains only one column for all SAFRAN cells. Except for atmospheric pressure (`P_atm`), the format of data file is same. Conceptually, only hourly and formatted data can be read by ProSe-PA0.77. The first line is formatted as `[Juilan_Day Hour SAFRAN_CELL_ID1 SAFRAN_CELL_ID2 SAFRAN_CELL_ID3 ...]`. The other lines are formatted as `[day_val hour_val VAL1 VAL2 VAL3 ...]`.

For example in case of air temperature,

```
! In the file TMPH_case_study.txt
Juilan_Day Hour 890 976 1453 1563 ...
1461 0 282.55 282.55 282.55 282.55 ...
1461 1 282.55 282.55 282.55 282.55 ...
1461 2 282.76 282.76 282.76 282.76 ...
...
```

> Conceptually, the reading of spatiotemporal meteorological data depends on the number of SAFRAN cells in the data files. For the moment, the number of SAFRAN cells is fixed to 26, which should contain all SAFRAN cells mentioned in `met_corresp` (section 2.3.5.1). To have more SAFRAN cells, a recompilation of ProSe-PA is needed!

2.3.6 Block SIMSET-HT: heat transport settings

```
heat_transport  = {
    theta_t     = 1.0
    init_T_file = [°C] include init_T
}
```

It includes,

- `theta_t`, degrees of implicitness of the equations' discretisation. Set to 1.0 by default.
- `init_T_file`, to initialize the water temperature at cell center from the ASCII text file `init_T` which contains two columns separated by space: `[cell_identifier water_temp]`.

> The block SIMSET-HT is mandatory for initializing heat transport related structures and launching heat simulation.

2.3.7 Block SIMSET-ST: solute transport settings

```
transport = {
    theta_ttc = 1.0
}
```

It includes,

- `theta_ttc`, degrees of implicitness of the equations' discretisation. Set to 0.8 by default.

> The block SIMSET-ST is mandatory for initializing solute transport related structures and launching solute transport simulation.

> For the moment, since *libttc* library doesn't store the boundary conditions at previous time step, it is recommended to use 1.0 for `theta_t` and `theta_ttc`. It is then a fully implicit calculation.

2.3.8 Block SIMSET-BIO: biogeochemistry settings

The block SIMSET-BIO, initiated by the keyword `biology`, sets biogeochemical calculation and activates .

2.3.8.1 General settings

The settings mainly includes three parts: `LAYERS`, sediment erosion (`EXCHANGES`) and meteorological data `METEO`.

```
biology = {                                               ! Block SIMSET-BIO
    numerical_method = EXPLICIT
    max_div_dt       = 1.0
    dz_rive          = [m] 0.1
    layers           = {
        include waterlayer
        include vaselayer
    }
    exchanges        = {
        calc_sedim_eros = SIMULTANEOUS_PROSE
        eta_hyd         = [g/cm^3] 0.0013
        pnavig          = [g/m^2/s] 0.0005
    }
    meteo            = {
        temperature = { ... }
        wind        = { ... }
        radiation   = { ... }
        photoperiod = { ... }
   }
}
```

It includes,

- `numerical_method`, two numerical methods are available : explicit method (`EXPLICIT`) and Fourth Order Runge-Kutta method (`RUNGE_KUTTA`). Set to `RUNGE_KUTTA` by default. It is recommended to use `EXPLICIT` method, which spends less time.
- `max_div_dt`, maximum re-division factor of the calculation time step, set to 1.0 by default.
- `dz_rive`, vertical discretization, set to 0.1 [m] by default.
- `layers`, definition of simulated layers, detailed in section 2.3.8.2.
- `exchanges`, parameterization of sediment erosion
  - `calc_sedim_eros`, method to calculate sedimentation-erosion, only `SIMULTANEOUS_PROSE` is available for the moment.
  - `eta_hyd`, η, fraction of total hydraulic power dissipated by the flow to maintain particles in suspension in the water column (Vilmin *et al.*, 2015; Wang et Flipo, 2020). Here we put $\eta \times \rho_{water}$, that' why the unit is [g cm^{-3}].

    The code should be modified to put directly η, but not $\eta \times \rho_{water}$!

  - `Pnavig`, estimated average power transmitted by river navigation (Vilmin *et al.*, 2015; Wang et Flipo, 2020).
- `meteo`, sets meteorological data used in *librive*, detailed in section 2.3.8.4

2.3.8.2 Definition of simulated layers

The definition of simulated layers is initiated by the keyword `LAYERS`.

```
LAYERS = {
        include waterlayer
        include vaselayer
}
```

The layer periphyton (`PERIPHYTON`) is not tested in ProSe-PA0.7x.

Definition of water layer

The definition of a layer initiated by the layer type (`WATER/VASE`) and followed by the layer name (a string). We can set the initial concentrations of the simulated biogeochemical species in the layer. The initialization starts with the keyword `initial_concentrations`. The syntax of initialization is `[SPECIES SPECIES_NUM = [UNIT] CONC_INIT]`. A biogeochemical species can have several subspecies (for example, heterotrophic bacteria distinct by their size). The `SPECIES_NUM` presents the number of the subspecies, which starts at 1.

```
! In file waterlayer
WATER                     ! LAYER TYPE (WATER)
water1_s1 = {             ! LAYER_NAME = { ... }
    initial_concentrations = {
        ! SPECIES SPECIES_NUM = [UNIT]  CONC_INIT
          bact    1           = [mgC/l] include bact1_init
          bact    2           = [mgC/l] include bact2_init
          bactn   1           = [mgC/l] include bactn1_init
          phy     1           = [mgC/l] include phy1_init
          phy     2           = [mgC/l] 0 0.
          ...
    }
}
```

The initialization file contains two columns: [CELL_ID CONC_VAL].

```
! Exp. file bact1_init
! CELL_ID CONC_VAL
1 0.02
2 0.02
3 0.02
...
200 0.02
```

For a homogeneous initialization, we can simply write one line: [unit] 0 value. The program sets 0 to all cells. It is useful for a steady-state simulation.

Definition of vase layer

In case of sediment layer, apart of initial concentrations, we can define the porosity of the layer (porosity) and the sediment volume of each cell (volume_layer). The sediment mass of each cell then can be calculated. It is recommended to use a porosity of 0.9.

```
! Exp. file vaselayer
VASE                      ! LAYER TYPE (VASE)
vase1_s1 = {              ! LAYER_NAME = { ... }
    porosity = 0.9
    volume_layer = [m^3] include volume_vase
    initial_concentrations = {
        ! SPECIES SPECIES_NUM = [UNIT]  CONC_INIT
          bact    1           = [mgC/l] 0 0.
          bact    2           = [mgC/l] 0 0.
          bactn   1           = [mgC/l] 0 0.
          phy     1           = [mgC/l] 0 0.
          phy     2           = [mgC/l] 0 0.
          ...
    }
}
```

Normally, the sediment layer can be initialized to null and the program simulates an accumulation of sediment. The initialization of sediment layer allows for starting a simulation with the output of another simulation.

2.3.8.3 Predefined biogeochemical species

20 biogeochemical main species are implemented in *librive* (Tab. 2.3), among which five species (`SODA, TA, DIC, PH, CO2`) need a compilation of the unified RIVE v1.0 (Wang *et al.*, 2024). Each main species can have several subspecies For example, they can be heterotrophic bacteria distinct by their size (`BACT`: small one and large one), different phytoplankton species (`PHY`: diatoms, green algae, and cyanobacteria), different nitrifying bacteria (`BACTN`: ammonia-oxidizing bacteria and nitrite-oxidizing bacteria).

keyword	Species
`PHY`	phytoplankton
`ZOO`	zooplankton
`BACT`	heterotrophic bacteria
`BACTN`	nitrifying bacteria
`BIF`	indicator bacteria for fecal contamination
`MES`	mineral suspended matter
`MOP`	particulate organic matter
`MOD`	dissolved organic matter
`SI_D`	dissolved silica
`NH4`	ammonium
`NO2`	nitrite
`NO3`	nitrate
`PO4`	orthophosphate
`O2`	dissolved oxygen
`N2O`	nitrous oxide
code	Species defined in unified RIVE v1.0
`SODA`	small monomeric substrate
`TA`	total alkalinity
`DIC`	dissolved inorganic carbon
`PH`	acidity
`CO2`	carbon dioxide

Table 2.3: keywords for predefined biogeochemical species in ProSe-PA0.77

Description of biogeochemical species

A species description file should start with the species keyword (see Table 2.3), followed by descriptions of one or more subspecies (`SUBSPECIES_NAME = ...`). For each subspecies, various parameters (properties) can be defined according to their type: parameters for particulate species (density and porosity), for living species (optimum temperature for growth), biochemical composition (`C/N`, `C/P`, `C/Si` and `C/chla` ratios), half-saturation constants for Monod-Michaelis-Menten kinetics (`kmich` ...). Each living subspecies associates several biogeochemical processes such as growth, photosynthesis, respiration, mortality. Advection-diffusion properties is also defined for each subspecies (`transport_type`, `media_type`, `advection`, `diffusion`).

An example of phytoplankton (`PHY`) is shown below,

```
! Exp. phytoplankton
PHY                                  ! Species keyword

dia = {                              ! Subspecies name, SPECIES_NUM = 1
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Particulate properties
    rho  = [g/cm^3] 1.2
    phi  = 0.9
    vsed = [m/h] 0.004

    ! Living species properties
    Topt  = [◦C] 21.
    sigma = [◦C] 13.

    ! Monod-Michaelis-Menten kinetics
    kmich phys = 0.06
    kmich N    = [ugN/l] 14
    kmich P    = [ugP/l] 15.5
    kmich NH4  = [umol/l] 5.
    kmich Si   = [mgSi/l] 0.196

    ! Biochemical composition
    C/ P    = [mgC/mgP] 40.
    C/ N    = [mgC/mgN] 7.
    C/ si   = [mgC/mgSI] 1.071429
    C/ chla = [mgC/ugchla] 0.035

    dbo     = [molO2/molC] 1.


    ! Biogeochemical processes
    photosynthesis = {
        eta      = [/m] 0.2
        eta_mes  = [l/mg/m] 0.042
        eta_chla = [l/ugchla/m] 0.02
        alpha    = [m^2s/uE/h] 0.0012
        beta     = [m^2s/uE/h] 0
        pmax     = [/h] 0.2
    }
   growth = {
        sr    = [/h] 0.15
        cr    = [/h] 0.2
        mumax = [/h] 0.05
    }

   excretion = {
        excr_cst  = [/h] 0.001085714
        excr_phot = 0.0005714286
    }

   respiration = {
        maint = [/h] 0.002
        energ = 0.5
    }

   mortality = {
        mort         = [/h] 0.025
        phi_lim_mort = [mgC/l] 7
        delta        = 1.
    }
}

green_algae = { ... }                          ! Subspecies name, SPECIES_NUM = 2
cyanobacteria = { ... }                        ! Subspecies name, SPECIES_NUM = 3
...
```

An example of zooplankton (ZOO) is shown below,

```
ZOO

zor =                                          ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Particulate properties
    rho  = [g/cm^3] 1.2
    phi  = 0.9
    vsed = [m/h] 0.005 ! a voir

    ! Living species properties
    Topt  = [$\circ$C] 25.
    sigma = [$\circ$C] 10.

    ! Monod-Michaelis-Menten kinetics
    kmich phy = 0.1

    ! Biochemical composition
    C/ P    = [mgC/mgP] 40.
    C/ N    = [mgC/mgN] 7.

    ! Biogeochemical processes
    growth = {
        mumax      = [/h] 0.025
        phi_lim_gr = [mgC/l] 0.1
   }
    respiration = { }
    mortality   = {
        mort = [/h] 0.007
    }
   grazing = {
        graz = [/h] 0.1
    }
}

zoc = { ... }  ! Subspecies name, SPECIES_NUM = 1
...            ! Subspecies name, SPECIES_NUM = ...
```

An example of heterotrophic bacteria (BACT) is shown below,

```
! Exp. heterotrophic bacteria
BACT                                           ! Species keyword

shb =                                          ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Particulate properties
    rho  = [g/cm^3] 1.2
    phi  = 0.9
    vsed = [m/h] 0.005

    ! Living species properties
    Topt  = [◦C] 20.
    sigma = [◦C] 17.

    ! Biochemical composition
    C/ P =  [mgC/mgP] 40.
    C/ N =  [mgC/mgN] 7.

    ! Monod-Michaelis-Menten kinetics
    kmich O2  = [mgO2/l] 0.5
    kmich mod = [mgC/l] 0.1

    ! Biogeochemical processes
    growth = {
        mumax  = [/h] 0.04
        yield  = [-] 0.25
        k_bact = [mgC/l] 0.03
    }
    respiration = { }                              ! Mandatory
    mortality  = {
        mort = [/h] 0.02
    }
}
lhb = { ... }  ! Subspecies name, SPECIES_NUM = 2
...            ! Subspecies name, SPECIES_NUM = ...
```

An example of nitrifying bacteria (BACTN) is shown below,

```
! Exp. nitrifying bacteria
BACTN                                              ! Species keyword

ammonia_oxidizing =                                ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Particulate properties
    rho   = [g/cm^3] 1.2
    phi   = 0.9
    vsed = [m/h] 0.1

    ! Living species properties
    Topt  = [◦C] 23.
    sigma = [◦C] 18.

    ! Monod-Michaelis-Menten kinetics
    kmich nh4 = [mgN/l] 0.75
    kmich O2  = [mgO2/l] 0.64

    ! Biochemical composition
    C/ P = [mgC/mgP] 40.
    C/ N = [mgC/mgN] 7.

    ! Biogeochemical processes
    growth = {
        mumax        = [/h] 0.07
        yield_nit    = [mgC/mgN] 0.07
        stoechio_nit = [molO2/molN] 1.5
        k_bact = [mgC/l] 0.00003 ! limitation for sediment layer
    }

    mortality = {
        mort = [/h] 0.005
    }
}

nitrite_oxidizing =                                 ! Subspecies name, SPECIES_NUM = 2
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Particulate properties
    rho   = [g/cm^3] 1.2
    phi   = 0.9
    vsed = [m/h] 0.1

    ! Living species properties
    Topt  = [○C] 23.
    sigma = [○C] 18.

    ! Monod-Michaelis-Menten kinetics
    kmich no2 = [mgN/l] 0.05
    kmich O2  = [mgO2/l] 1.088

    ! Biochemical composition
    C/ P =  [mgC/mgP] 40.
    C/ N =  [mgC/mgN] 7.

    ! Biogeochemical processes
    growth = {
        mumax        = [/h] 0.05
        yield_nit    = [mgC/mgN] 0.02
        stoechio_nit = [molO2/molN] 0.5
        k_bact = [mgC/l] 0.00003 ! limitation for sediment layer
    }

    mortality = {
        mort = [/h] 0.005
    }
}
...  ! Subspecies name, SPECIES_NUM = ...
```

An example of fecal bacteria (BIF) is below,

```
BIF                                                ! Species keyword

afc =                                              ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Particulate properties
    rho  = [g/cm^3] 1.2
    phi  = 0.9
    vsed = [m/h] 0.02

    ! Living species properties
    Topt  = [○C] 25.
    sigma = [○C] 20.

    ! Biochemical composition
    C/ P =  [mgC/mgP] 40.
    C/ N =  [mgC/mgN] 7.

    ! Biogeochemical processes
    mortality = {
        mort      = [/h] 0.0225
        kd        = [/m] 5
        kp        = [l/mg/m] 0.0
        alpha_bif = [m^2/J] 0.009
        m_bif     = [-] 0.186
        c_ref     = [mgc/l] 0.00000075
    }
}

ffc = { ... }                                 ! Subspecies name, SPECIES_NUM = 2
...                                           ! Subspecies name, SPECIES_NUM = ...
```

An example of mineral suspended matter (MES) is below,

```
MES                                             ! Species keyword

mes1 =                                          ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Particulate properties
    rho   = [g/cm^3] 2.0
    phi   = 0.9
    vsed = [m/h] 1.
}
```

An example of particulate organic matter (MOP) is below,

```
MOP                                           ! Species keyword

mop1 =                                        ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Particulate properties
    rho   = [g/cm^3] 1.2
    phi   = 0.9
    vsed = [m/h] 1.

    ! Ratio of mop1 in death of living  species
    compose_mo = 0.3

    ! Biochemical composition
    C/ P = [mgC/mgP] 40.
    C/ N = [mgC/mgN] 7.

    ! Biogeochemical processes
    hydrolysis = {
        kc_hydr = [/h] 0.005
    }
}
```

```

mop2 =                                       ! Subspecies name, SPECIES_NUM = 2
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Particulate properties
    rho  = [g/cm^3] 1.2
    phi  = 0.9
    vsed = [m/h] 1.

    ! Ratio of mop2 in death of living  species
    compose_mo = 0.15

    ! Biochemical composition
    C/ P = [mgC/mgP] 40.
    C/ N = [mgC/mgN] 7.

    ! Biogeochemical processes
    hydrolysis = {
        kc_hydr = [/h] 0.000225
    }
}

mop3 =                                       ! Subspecies name, SPECIES_NUM = 3
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Particulate properties
    rho  = [g/cm^3] 1.2
    phi  = 0.9
    vsed = [m/h] 1.

    ! Ratio of mop3 in death of living  species
    compose_mo = 0.05

    ! Biochemical composition
    C/ P = [mgC/mgP] 40.
    C/ N = [mgC/mgN] 7.

    ! Biogeochemical processes
    hydrolysis = {
        kc_hydr = [/h] 0.0
    }
}
```

An example of dissolved organic matter (MOD) is below,

```
MOD                                               ! Species keyword

mod1 = {                                          ! Subspecies name, SPECIES_NUM = 1

    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Ratio of mod1 in death of living  species
    compose_mo   = 0.3

    ! Biochemical composition
    C/ N = [mgC/mgN] 7.
    C/ P = [mgC/mgP] 40.

    ! Diffusion coefficient at water-sediment interface
    ds   = [m^2/s] 0.0015

    ! Monod-Michaelis-Menten kinetics
    kmich mod = [mgC/l] 0.25

    ! To consider mod1 as substrat for heterotrophic bacteria uptake, a highest hydrolysis rate is defined
    hydrolysis = {
        kc_hydr = [/h] 0.75
    }
}


mod2 =  {                                           ! Subspecies name, SPECIES_NUM = 2
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type = FREE_WATER
    advection = yes
    diffusion = no
    diff_mol = [m^2/s] 0.

    ! Ratio of mod1 in death of living  species
    compose_mo   = 0.15

    ! Biochemical composition
    C/ N =  [mgC/mgN] 7.
    C/ P = [mgC/mgP] 40.

    ! Diffusion coefficient at water-sediment interface
    ds  = [m^2/s] 0.0015

    ! Monod-Michaelis-Menten kinetics
    kmich mod = [mgC/l] 0.200000

    ! Hydrolysis process
    hydrolysis = {
        kc_hydr = [/h] 0.550000
    }
}

mod3 = {                                           ! Subspecies name, SPECIES_NUM = 3
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type = FREE_WATER
    advection = yes
    diffusion = no
    diff_mol = [m^2/s] 0.

    ! Biochemical composition
    compose_mo  = 0.05

    ! Biochemical composition
    C/ N = [mgC/mgN] 7.
    C/ P =  [mgC/mgP] 40.

    ! Diffusion coefficient at water-sediment interface
    ds  = [m^2/s] 0.0015

    ! Monod-Michaelis-Menten kinetics
    kmich mod =  [mgC/l] 0.0

    ! Hydrolysis process
    hydrolysis = {
        kc_hydr = [/h] 0.0
    }
}
```

An example of dissolved silica (SI_D) is below,

```
SI_D                                       ! Species keyword

sio2 =                                     ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Diffusion coefficient at water-sediment interface
    ds  = [m^2/s] 0.0015
}
```

An example of ammonium (NH4) is below,

```
NH4                                              ! Species keyword

ammonium =                                        ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Diffusion coefficient at water-sediment interface
    ds  = [m^2/s] 0.0015
}
```

An example of nitrite (NO2) is below,

```
NO2                                              ! Species keyword

nitrite =                                        ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Diffusion coefficient at water-sediment interface
    ds  = [m^2/s] 0.0015

    ! reactions = {                          ! Optional, example for decay
    !    reaction = {
    !        decay          = [/h] 0.005
    !        other_reactors = { oxygene 0.5 }
    !        products       = { nitrate 1. }
    !    }
    !}
}
```

An example of nitrate (NO3) is below,

```
NO3                                               ! Species keyword

nitrate =                                         ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Diffusion coefficient at water-sediment interface
    ds   = [m^2/s] 0.0015
}
```

An example of orthophosphate (PO4) is below,

```
PO4                                                ! Species keyword

phosphate = {                                      ! Subspecies name, SPECIES_NUM = 1
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Diffusion coefficient at water-sediment interface
    ds   = [m^2/s] 0.0015

    ! Adsorption-desorption
    ads_desorption = {
        Kps        = [mgP/l] 0.682
        Pac        = [mgP/mg] 0.0058
        damping    = [s] 600
        adsorbs_on = { mes 1 }
    }
}
```

An example of oxygen (O2) is below,

```
O2                                               ! Species keyword

```

```
oxygen =                                          ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Diffusion coefficient at water-sediment interface
    ds  = [m^2/s] 0.0015

    ! Oxygen reaeration at air-water interface
    reaeration = {
        rea_navig = [m/h] 0.
        rea_wind  = [cm/h] 118.
        d_richey  = [cm^2/s] 0.000026
    }
}
```

N2O and the species defined in unified RIVE v1.0 have not yet been tested in PROSE-PA0.7x.

An example of small monomeric substrate (SODA) is below,

```
SODA                                                ! Species keyword

substrate =                                         ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Diffusion coefficient at water-sediment interface
    ds  = [m^2/s] 0.0015

    ! Biochemical composition
    C/ N = [mgC/mgN] 7.
    C/ P =  [mgC/mgP] 40.
}
```

An example of total alkalinity (TA) is below,

```
TA                                              ! Species keyword

total_alkalinity =                                       ! Subspecies name,
    SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Diffusion coefficient at water-sediment interface
    ds   = [m^2/s] 0.0 ! a voir
}
```

An example of dissolved inorganic carbon (DIC) is below,

```
DIC                                              ! Species keyword

dic1 =                                        ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Diffusion coefficient at water-sediment interface
    ds   = [m^2/s] 0.0015
}
```

An example of acidity (PH) is below,

```
PH                                            ! Species keyword

acidity =                                        ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Diffusion coefficient at water-sediment interface
    ds   = [m^2/s] 0.0 ! a voir

    ! Culberson (1980) formula
    kb   = [mol/l] 5.81e-10
    bor0 = [mmol/l] 0. ! freshwater=0mM and Sea water = 0.41mM or 4-5 ppm

}
```

An example of carbon dioxide (`CO2`) is below,

```
CO2                                              ! Species keyword

carbon_dioxide =                                 ! Subspecies name, SPECIES_NUM = 1
{
    ! Advection-diffusion properties
    transport_type = SOLUTE
    media_type     = FREE_WATER
    advection      = yes
    diffusion      = no
    diff_mol       = [m^2/s] 0.

    ! Diffusion coefficient at water-sediment interface
    ds  = [m^2/s] 0.0015 ! a voir

    ! Exchanges at air-water interface
    reaeration = {
        k600_method = User_defined ! Reservoirs, Strahler_Order, User_defined
        k600        = [m/h] 0.08 ! user defined k600 value
      }

}
```

The list of parameters that can be set for each process is given in Table 2.4.

Parameters of biogeochemical species

name	description	unit	default value
	parameters of particulate species		
`rho`	Density	[g m^{-3}]	1000 g m^{-3}
`phi`	Porosity		0.
`Vsed`	Sinking velocity	[m h^{-1}]	
`scouring`	Phytoplankton removal parameter		
	parameters of dissolved species		
`Ds`	Diffusion coefficient at water-sediment interface	[m^2 s^{-1}]	
	parameters of living species		

Topt	Optimum temperature	°C	20 °C
sigma	Standard deviation of temperature function	°C	1 °C
prod_N2O	Nitrous oxide production rate for denitrification	[h^{-1}]	0.
parameters of mineral species			
compose_mo	Portion of organic matter in which degraded living species during mortality		
photosynthesis parameters (photosynthesis)			
eta	Basic light extinction term	[L^{-1}]	
eta_chla	Linear algal self-shading light extinction coefficient	[L^{2}M^{-1}]	
eta_mes	Light extinction coefficient related to suspended solids	[L^{2}M^{-1}]	
alpha	Photosynthetic efficiency	[h^{-1} (μE m^{-2} s^{-1})$^{-1}$]	
beta	Photoinhibition capacity	[h^{-1} (μE m^{-2} s^{-1})$^{-1}$]	
pmax	Maximum rate of photosynthesis	[h^{-1}]	
Growth parameters (growth)			
mumax	Maximal growth rate	[h^{-1}]	
yield	Growth yield	[-]	
phy_lim_gr	critical concentration above which growth is reduced	[mgC L^{-1}]	
sr	Maximal rate of reserve products synthesis for phytoplankton	[h^{-1}]	
cr	Maximal rate of reserve products catabolism for phytoplankton	[h^{-1}]	
yield_nit	growth yield for nitrifying bacteria	[mgC/mgNH4]	
kmich_no2	half-saturation constant with respect to nitrites during nitrous oxide production		
cmin_o2	concentration in oxygen dissolved below which nitrous oxide can be produced during nitrification		
stoechio_nit	stoichiometry for O2 and N in nitrification	[molO2/molN]	
fbis	growth correction parameter for nitrous oxide production		
Respiration parameters (respiration)			
maint	Maintenance respiration coefficient for phytoplankton	[h^{-1}]	

`energ`	Energetic cost of growth for phytoplankton	[-]
	Grazing parameters for zooplankton (`grazing`)	
`graz`	Maximal grazing rate	[h^{-1}]
	Excretion parameters for phytoplankton (`excretion`)	
`excr_cst`	Basic excretion rate	[h^{-1}]
`excr_phot`	Excretion constant related to photosynthesis	[-]
	Mortality parameters (`mortality`)	
`mort`	Mortality rate	[h^{-1}]
`phi_lim_mort`	Concentration above which mortality is accelerated	[mgC L^{-1}]
`delta`	Mortality multiplication factor for a concentration greater than `phi_lim_mort`	[-]
`kd`	light extinction related to dissolved for bif light degradation	[m^{-1}]
`kp`	light extinction related to particulate (MSS) for bif light degradation	[m^{-1} mg]
`alpha_bif`	Photoinhibition coefficient	[m^{2} J^{-1}]
`C_ref`	Reference concentration of bif	[mgC L^{-1}]
`m_bif`	Exponent of $\frac{C}{C_ref}$ for bif light degradation	[-]
	Chemical reaction parameters (`reaction`)	
`decay`	Radioactive decay rate (order 1)	[h^{-1}]
	Hydrolysis parameters (`hydrolysis`)	
`kc_hydr`	half-saturation constant for hydrolysis of organic matter	[mgC L^{-1}]
	Adsorption-desorption parameters (`ads_desorption`)	
`Kps`	Half saturation adsorption constant	[mgP L^{-1}]
`Pac`	Maximum adsorption capacity of mineral suspended solids	[mgP/mg]
`damping`	Adsorption damping coefficient	[s]
	Oxygen reaeration parameters (`reaeration`)	
`rea_navig`	Reaeration coefficient due to navigation	[m h^{-1}]
`rea_wind`	Reaeration coefficient due to wind speed	[m h^{-1}]
`D_Richey`	Diffusion coefficient for stagnant water	[m h^{-1}]

	Transport parameters (`Advection-dispersion`)		
`transport_type`	Type of heat or chemical substance (HEAT or SOLUTE), for the moment no specific temperature species		SOLUTE
`media_type`	FREE_WATER for river advection		
`advection`	Advection process (YES or NO)		
`diffusion`	Molecular diffusion (YES or NO)		
`diff_mol`	Molecular diffusion coefficient (YES or NO)	[m^2 s^{-1}]	

Table 2.4: Elements to be defined in the command file

2.3.8.4 Meteorological data settings

User-provided data When water temperature, wind speed, radiation data (time series) are available, they can be included as following,

```
temperature = { [d] [°C]
                include temp
}
wind        = { [d] [m/s]
                0. 2.0
}
radiation   = { [d] [J/cm^2/h]
                include RayTrappes2007_2012
}
```

Program calculated data If no data are available, the user can provide the `mean`, `amplitude`, `delay` and `attenuation` parameters corresponding to water `temperature`, solar `radiation` and `photoperiod`. The water temperature and solar radiation can be calculated according to sinusoidal function.

```
temperature = {
    mean      = [°C] 12.
    amplitude = [°C] 20.
    delay     = [h] 29.
}
radiation   = {
    mean               = [uE/m^2/s] 542
    amplitude          = [uE/m^2/s] 288
    attenuation_factor = 1
}
photoperiod = {
    mean      = [h] 12.
    amplitude = [h] 4.2
}
```

The user-provided and sinusoidal data are spatially homogeneous. When heat transport is activated, the program uses water temperature calculated by heat transport, which allows for spatially heterogeneous data.

2.3.8.5 SIMSET-DA: settings of parameter ranges for data assimilation

All parameters considered in data assimilation should be defined here. The block SIMSET-DA is initiated by the keyword `param_range_da`. The syntax is `[PARAM_NAME UNIT VAL_MIN VAL_MAX S_PERCENT]`.

```
param_range_da = { includes param_range }
```

In total, 13 parameters can be defined (Tab. 2.5).

Table 2.5: Parameters considered in data assimilation

Parameters	Description
Physical parameters	
η_{water}	Light extinction coefficient for pure water
K_{navig}	Reaeration coefficient related to the navigation
Bacterial parameters	
$\mu_{max,hb}$	Maximum growth rate of bacteria
$mort_{hb}$	Maximum mortality rate of bacteria
$T_{opt,hb}$	Optimal temperature for bacterial growth
Y_{hb}	Bacterial growth yield
Phytoplanktonic parameters	
α_{pp}	Photosynthetic capacity
$\eta_{chla,pp}$	Light extinction coefficient by algal self-shading
$C_{pp}/\mathrm{Chl}a$	Ratio of carbon to chlorophyll *a*
$P_{max,pp}$	Maximum photosynthesis rate
$R_{m,pp}$	Respiration of maintenance
$T_{opt,pp}$	Optimal temperature for growth of phytoplankton
Organic matter parameters	
b_1	Fraction of biodegradable `MOD`

The `S_PERCENT`, which is used for particle filter and defines the degree of random walk after resampling procedure, is optional. If no defined, the global `S_PERCENT` defined in general setting is used (section 2.3.1).

```
! Exp. param_range
! PARAM_NAME    UNIT          VAL_MIN    VAL_MAX    S_PERCENT
Rm_phy          [/h]          0.001      0.021      0.10
a_phy           [m^2s/uE/h]   0.0003     0.0018     0.10
pmax_phy        [/h]          0.09       0.546      0.10
eta_chla_phy    [l/ugchla/m]  0.006      0.054      0.10
c_chla_phy      [mgC/ugchla]  0.02       0.13       0.10
eta_water       [/m]          0.2        0.8        0.10
Topt_phy        [°C]          10.        37.        0.10
mumax_bact      [/h]          0.01       0.13       0.10
yield_bact      [-]           0.03       0.5        0.04
mort_bact       [/h]          0.01       0.08       0.04
Topt_bact       [°C]          10         35         0.10
Krea_navig      [m/h]         0.         0.05       0.10
b1_river        [-]           0.1        0.5        0.04
```

In ProSe-PA0.7x, it is possible to run a simulation with time-varying values of these parameters. The syntax to define time-varying parameter values is `[TIME_UNIT PARAM_UNIT TIME VALUE TIME VALUE ... ...]`, which can be included by the keyword `INCLUDE`.

CHAPTER 3

Model outputs

3.1 `Outputs` : Syntax overview

The syntax template provided below describes the "outputs" section of the command file, initiated by the keyword `Outputs` and designed to set up the output characteristics. For each output type available, a dedicated sub-block sets the properties of the corresponding output files. The set of keywords for the ProSe-PA output types (`[OUTPUT-TYPE]`) are detailed in table 3.1.

```
Outputs = {
    OUTPUT-TYPE = {
         ...
    }
    OUTPUT-TYPE = {
         ...
    }
    ...
}
```

The subblocks `[OUTPUT-TYPE]` are initiated by a dedicated keyword that defines a given output file type. Recall that all outputs are stored in the master output folder targeted by the `Output_folder` keyword in the command file (Sec. 1.2). Within this folder, output files from each ProSe-PA module are stored in a dedicated subfolder that contains the name of the corresponding module.

Table 3.1: ProSe-PA0.7x output type files.

Output types	Sub-folder name	Output keyword
PK files	`$OUTPUT_FOLDER/`	`PRINT_PK`
Final states	`$OUTPUT_FOLDER/`	`FINAL_STATE`
Water balance by elements	`$OUTPUT_FOLDER/`	`MB_ELEMENTS`
Transversal states	`$OUTPUT_FOLDER/time_series`	`TIME_SERIES`
Longitudinal states	`$OUTPUT_FOLDER/longitudinal_profiles`	`LONGITUDINAL_PROFILES`
Tube calculations	`$OUTPUT_FOLDER/tube`	`TUBE_OUTPUT`
Biogeochemical mass balance	`$OUTPUT_FOLDER/mass_balance_bio`	`MASS_BALANCES_BIO`
Energy balance	`$OUTPUT_FOLDER/energy_balance_heat`	`ENERGY_BALANCES_HEAT`

3.2 Output types

3.2.1 PRINT_PK: Saving the PK (kilometric point) of singularities and inflows

The kilometric point (PK) of each singularity and inflow (Sec. 2.3.4.3) can be printed. The default choice is `NO` for printing pks. The `pks` file is written directly into the master `$OUTPUT_FOLDER`.

```
PRINT_PK = {
    yes
    FILE_NAME = pks
}
```

3.2.2 FINAL_STATE : Saving the final state of a run

```
FINAL_STATE = {
    yes
    FILE_NAME = seine_final
}
```

This option saves its different states (hydraulic, heat, biogeochemical) at the end of the simulation in different files, which can be used to initialize the status of a compartment in a new simulation. These files are written directly into the master `$OUTPUT_FOLDER`.

For a full simulation, this option creates the following files,

- `seine_final_Z`, final water surface levels at each cell (element) center, can be used for initialization (Sec. 2.3.4.2).
- `seine_final_Q`, final discharges at each cell (element) face, can be used for initialization (Sec. 2.3.4.2).

- seine_final_T, final water temperature at each cell (element) center, can be used for initialization (Sec. 2.3.6).
- under steady-state
    - conc_init, final concentrations of biogeochemical species in water column (mmol m^{-3} or g m^{-3}), can be used for initialization (Sec. 2.3.8.2). No sediment concentrations are calculated under steady-state simulation.
- under transient-state
    - conc_final_water, final concentrations of biogeochemical species in water column (mmol m^{-3} or g m^{-3}), can be used for initialization (Sec. 2.3.8.2).
    - conc_final_vase, final concentrations of biogeochemical species in sediment layer (mmol m^{-3} or g m^{-3}), can be used for initialization (Sec. 2.3.8.2).
    - volume_final_vase, final volume of each cell (element) in sediment layer (m^{-3}), can be used for initialization (Sec. 2.3.8.2).

The filenames of concentrations and volume are not user-defined, but in ProSe-PA. It should be changed as other final filenames.

3.2.3 MB_ELEMENTS: Water balance element by element

```
MB_ELEMENTS = { yes }
```

This option is used to debug a water conservation problem and create the file mb_elements.txt. The default option is NO. This option prints, for each cell (element), the difference between water mass at time t and t-1 (deltaM), difference between upstream and downstream fluxes (deltaF), and the relative error (%).

```
!Exp. mb_elements.txt
#File generated by libhyd0.40

Mass balance at time t = 900.000000 s :
element 0 : deltaM = 0.026778 ; deltaF = 0.026778 ; error = 0.000001 %
element 1 : deltaM = -0.028199 ; deltaF = -0.028199 ; error = -0.000001 %
element 2 : deltaM = -0.006358 ; deltaF = -0.006358 ; error = -0.000058 %
...
```

It is recommended to activate this option only when debugging a water conservation problem.

3.2.4 TIME_SERIES: Transversal time series of asked variables

```
TIME_SERIES = {
    TIME      = { DT = [min] 15 }
    TIME_UNIT = [s]
    VAR       = {
        hh  [m]
        zfs [m]
        ...

    }
    POINTS    = {
        Seine
        [km] 50. 1
        [km] 90. 1
        ...
    }
    POINTS_OBS = {
        Seine
        [km] 60. 1 o2 1 [s] [mgO2/l] include o2_obs_pk60
        [km] 90. 1 o2 1 [s] [mgO2/l] include o2_obs_pk60
        ...
    }
}
```

The block TIME_SERIES is initiated by the keyword TIME_SERIES. It includes,

- TIME includes DT, which defines the timespan for printing the time series. The syntax is TIME = { dt = [UNIT] VALUE }. It is optional. Set as the time step of simulation (Sec. 2.2) by default.
- TIME_UNIT, the unit of time (Julian day) in output file. Set to second ([s]) by default. In ProSe-PA0.7x, a corresponding date is also printed.
- VAR, defines the desired variables to be printed. The syntax is below and the VAR_KEYWORD is listed in the table 3.2,

```
VAR = {
    VAR_KEYWORD [VAR_UNIT]
    VAR_KEYWORD [VAR_UNIT]
    ...
}

```

- POINTS, defines the locations (transversal) at where the values of each above variable should be printed. The syntax is,

```
POINTS = {
    RIVER_NAME
    [PK_UNIT] PK BRANCH_NUM
    [PK_UNIT] PK BRANCH_NUM
    ...
}
```

  It includes,

  - RIVER_NAME, the river name on which the points are located.
  - [PK_UNIT], the unit of the PK, normally in [km].
  - PK, the value of the PK (Sec. 2.3.4.3 and Sec. 3.2.1).
  - NUM_BRANCH, the branch number of the reach on which the points are located (Sec. 2.3.4.3)

- POINTS_OBS, defines the locations (transversal) at where the observed concentrations of a biogeochemical species are available. The syntax is,

```
POINTS_OBS = {
    RIVER_NAME
    [PK_UNIT] PK BRANCH_NUM SPECIES SPECIES_NUM [TIME_UNIT] [CONC_UNIT] TIME CONC
   TIME CONC ...
    [PK_UNIT] PK BRANCH_NUM SPECIES SPECIES_NUM [TIME_UNIT] [CONC_UNIT] TIME CONC
   TIME CONC ...
    ...
}
```

  It includes,

  - SPECIES, species keyword (Tab. 2.3).
  - SPECIES_NUM, the number of the subspecies.
  - TIME_UNIT, the unit of time.
  - CONC_UNIT, the unit of observed concentrations.
  - TIME CONC, values of time and observed concentrations. The observed concentrations can be put in a file using the keyword include.

The observed concentrations (POINTS_OBS) are used only for data assimilation in ProSe-PA0.7x.

Table 3.2: ProSe-PA0.7x time series variable list.

Variable keywords	Description	Unit
	Hydraulics variables	
`hh`	Water height	[m]
`zfs`	Free surface level	[m]
`wfs`	Free surface width	[m]
`surf`	Wetted area	[m^2]
`disch`	Discharge	[m^3/s]
`vel`	Water velocity	[m/s]
`peri`	Wetted perimeter	[m]
`rh`	Hydraulic radius	[m]
	Biogeochemical species	
`phy`	Phytoplankton	[mgC/l]
`zoo`	Zooplankton	[mgC/l]
`bact`	Heterotrophic bacteria	[mgC/l]
`bactn`	Nitrifying bacteria	[mgC/l]
...	Reference to table 2.3	...
	Sediment layer	
`sedvol`	Volume of sediment per length of element	[m^3/m]
`sedh`	Sediment layer thickness	[cm]
	Temperature	
`tw`	Water temperature	[°C]
`ta`	Air temperature	[°C]

3.2.5 LONGITUDINAL_PROFILES: longitudinal profiles of asked variables

The block LONGITUDINAL_PROFILES allows for printing of all asked variables' values in an user-defined longitudinal EXTENT and time period.

```
LONGITUDINAL_PROFILES =
{
TIME       = {
    T_INI  = 01/01/2007 00:00:00 ! [d] 0.
    T_END  = 02/01/2007 00:00:00 ! [d] 1.
    DT     = [h] 12.
}
TIME_UNIT  = [d]
VAR        = {
    hh  [m]
    zfs [m]
    ...
}
EXTENT     = {
    Seine
    [km] 0.   1
    [km] 50.  1
}
}
LONGITUDINAL_PROFILES = { ... }
...
```

It includes,

- TIME, defines the starting time (T_INI), ending time (T_END), and timespan DT of the printing. The T_INI and T_END can be either a date or a Julian day ([UNIT] DAY).
- TIME_UNIT, unit for printing Julian day.
- VAR, list of variables to be printed, see table 3.2
- EXTENT, defines the upstream point and downstream point, between which the asked variables' values should be printed. The syntax is similar to the subblock POINTS,

```
EXTENT = {
    RIVER_NAME
    [PK_UNIT] PK BRANCH_NUM ! upstream point
    [PK_UNIT] PK BRANCH_NUM ! downstream point
}

```

The asked variables are printed branch by branch between the upstream point and downstream point (EXTENT). They are stored in the folder $OUTPUT_FOLDER/longitudinal_profiles. An example (main branch, 0) is below,

```
! Exp. lp_WATER_t1.00_pk0.00_pk50.00_seine_0.txt
#File generated by ProSe-PA0.77
#Longitudinal profile (mean transversal values) between PK 0.00 and 50.00 at time
    1.000000 in WATER
#PK POINT(XC YC) BACT 1 O2 1
0.000000;POINT(0.000000 0.000000);1.020115e-02;9.101778e+00;
0.750000;POINT(0.000000 0.000000);1.036221e-02;9.096843e+00;
1.250000;POINT(0.000000 0.000000);1.052569e-02;9.091870e+00;
...
50.000000;POINT(0.000000 0.000000);1.500461e-02;8.966302e+00;
```

The POINT(XC XY) gives coordinates of the PK when using georeferenced geometry data (Sec. 2.3.4.3). The POINT(0.000000 0.000000) means the geometry data is not georeferenced.

It is possible to defined several LONGITUDINAL_PROFILES = { ... }.

In ProSe-PA0.7x, LONGITUDINAL_PROFILES is not printed in data assimilation mode.

3.2.6 TUBE_OUTPUT: Printing tube coordinates and associated hydraulic properties

The pseudo-2D hydraulic simulation is realized by creating tubes (TUBE). The output of tube properties is divided into geometry data MESH_TUBE and hydraulics data HYD_TUBE. They are stored in the folder $OUTPUT_FOLDER/tube. The data can be imported easily into Qgis.

```
TUBE_OUTPUT = {
    MESH_TUBE = {
        T_INI      = 01/01/2007 ! [d] 0.
        T_END      = 02/01/2007 ! [d] 1.
        DT         = [h] 12.
        TIME_UNIT  = [d]
    }

    HYD_TUBE = {
        T_INI      = 01/01/2007 ! [d] 0.
        T_END      = 02/01/2007 ! [d] 1.
        DT         = [h] 12.
        TIME_UNIT  = [d]
    }
}
```

An example of tube hydraulics data (at tube center) is below,

```
! Exp. hyd_geotube_t1760.50.txt
TIME;ID_TUBE;H_WATER;U_WATER;ALTI_WATER
1760.500000;0;3.119991;0.164062;29.716803
1760.500000;1;4.877327;0.193025;29.716803
1760.500000;2;5.050491;0.193190;29.716803
...
```

It includes,

- Column TIME, gives the time and the unit is difined in the subblock HYD_TUBE.
- Column ID_TUBE, identifier of the tube.
- Column H_WATER, water height at tube center.
- Column U_WATER, water velocity at tube center.
- Column ALTI_WATER, altitude of water free surface at tube center.

And the coordinates of tubes,

```
! Exp. coord_geotube_t1760.50.txt
ID_TUBE;CELL_CENTRE;CELL_VERTICES
```

```
0;POINT(605491.235047 118494.921083);POLYGON((605476.069273 118478.714249, 605512.454113 118487.162273, 605506.374574 118510.101356, 605470.042226 118503.706454, 605476.069273 118478.714249))
1;POINT(605519.298641 118500.646506);POLYGON((605512.454113 118487.162273, 605532.032120 118491.707998, 605526.333758 118513.614397, 605506.374574 118510.101356, 605512.454113 118487.162273))
2;POINT(605538.726230 118504.600284);POLYGON((605532.032120 118491.707998, 605550.500405 118495.996061, 605546.038638 118517.082678, 605526.333758 118513.614397, 605532.032120 118491.707998))
...
```

It includes,

- Column ID_TUBE, identifier of the tube.
- Column CELL_CENTRE, coordinates of the tube center.
- Column CELL_VERTICES, coordinates of the tube vertices.

> In ProSe-PA0.7x, the tube calculation works only with user defined geometric data (cross-sections, Sec. 2.3.4.3), but not interpolated data (dx, longitudinal discretization, Sec. 2.3.4.1).

3.2.7 MASS_BALANCES_BIO: Printing mass balances of biogeochemical species

The printing of mass balances of biogeochemical species can be activated by the keyword MASS_BALANCES_BIO. Several mass balances can be defined and the order is numbering in the output filename. Each mass balance is initiated by the user-defined mass balance name (mb1, mb2, and ...). The syntax is below,

```
MASS_BALANCES_BIO = {
    mb1 = {
        T_INI    = 01/01/2007 ! [d] 0.
        T_END    = 02/01/2007 ! [d] 1.
        N_STEPS = 1.
        SPECIES = {
            phy [mol]
            o2  [mol]
            ...
            mes [kg]
        }
        EXTENT = {
            Seine
            [km] 0.  1 ! upstream point
            [km] 60. 1 ! downstream point
        }
    }
    mb2 = { ... }
    ...
}
```

It includes,

- mb1, a user-defined string for mass balance name.
- T_INI, starting time, can be either a data or a Julian day.
- T_END, ending time, can be either a data or a Julian day.
- N_STEPS, number of time steps for calculating mass balance, timespan of mass balance (dt_mb) is $dt_{mb} = n_{steps} \times dt$. It is recommended to use 1.
- SPECIES, list of biogeochemical species to be printed (see table 2.3).
- EXTENT, defines the upstream point and downstream point, between which the mass balance should be calculated. It is optional. If no defined, the mass balance is calculated on entire simulated domain.

```
EXTENT = {
    RIVER_NAME
    [PK_UNIT] PK BRANCH_NUM ! upstream point
    [PK_UNIT] PK BRANCH_NUM ! downstream point
}

```

The PK of downstream point in EXTENT should not be exactly the PK of the last face in a reach. It is recommended to defined the upstream point and downstream point on the reach, which has only one branch.

It is not recommended to print MASS_BALANCES_BIO in data assimilation mode.

3.2.7.1 Biogeochemical processes in MASS_BALANCES_BIO output files

The MASS_BALANCES_BIO output files contain the following columns,

- Date, date (format FR_TS or US_TS).
- day, Julian day in second.
- HYDROLYSIS, hydrolysis of biodegradable organic matter (MOD, MOP).
- RAD_DECAY, user-defined reaction or radiactive decay.
- ADS_DESORPTION, adsorption - desorption, for example PO4.
- MORTALITY, mortality of living species.
- GRAZING, grazing of zooplankton.
- GROWTH, growth of living species.
- RESP, respiration.
- NITROS, ammonia oxidizing.
- NITRAT, nitrite oxidizing.

- RECYCLING, release of nutrients by mineralization.
- EXCR, excretion of phytoplankton.
- PHOT, photsynthesis of phytoplankton.
- CR_SR, catabolysis and synthesis of reserve products R, (Wang *et al.*, 2024).
- REA, reaeration at air-water interface.
- DEGAS, degassing.
- REOX, reoxygenation at weirs.
- DIFFUSION, dissolved exchanges at water-sediment interface (turbulent diffusion).
- SEDIMENTATION_EROSION, sedimentation - erosion.
- SCOURING, scouring of periphyton.
- FIN, fluxes inflow at upstream point.
- FOUT, fluxes outflow at downstream point.
- FIN_LATERAL, all lateral fluxes.
- MINI, initial mass at starting of timespan.
- MEND, final mass at end of timespan.
- EXCEPTION, exception procedure.
- ERROR, mass balance error.

3.2.8 ENERGY_BALANCES_HEAT: energy balances

The block ENERGY_BALANCES_HEAT is similar to the block MASS_BALANCES_BIO. The printing of energy balances can be activated by the keyword ENERGY_BALANCES_HEAT. Several energy balances can be defined and the order is numbering in the output filename. Each energy balance is initiated by the user-defined energy balance name (eb1, eb2, and ...). The syntax is below,

```
ENERGY_BALANCES_HEAT = {
    eb1 = {
        T_INI      = 01/01/2007 ! [d] 0
        T_END      = 02/01/2007 ! [d] 1.
        N_STEPS    = 1
        TIME_UNIT = [s]
        HEAT_UNIT = [J]
        EXTENT     = {
            Seine
            [km] 0. 1
            [km] 60. 1
        }
    }
    eb2 = { ... }
    ...
}
```

It includes,

- eb1, a user-defined string for energy balance name, used to name output filename. It is possible to defined several energy balances.

- `T_INI`, starting time, can be either a data or a Julian day.
- `T_END`, ending time, can be either a data or a Julian day.
- `N_STEPS`, number of time steps for calculating energy balance, timespan of energy balance (`dt_mb`) is $dt_{eb} = n_{steps} \times dt$. It is recommended to use 1.
- `TIME_UNIT`, unit for printing Julian day.
- `HEAT_UNIT`, unit for heat fluxes in output files.
- `EXTENT`, defines the upstream point and downstream point, between which the energy balance should be calculated. It is optional. If no defined, the energy balance is calculated on entire simulated domain.

```
EXTENT = {
    RIVER_NAME
    [PK_UNIT] PK BRANCH_NUM ! upstream point
    [PK_UNIT] PK BRANCH_NUM ! downstream point
}

```

> The `PK` of downstream point in `EXTENT` should not be exactly the `PK` of the last face in a reach. It is recommended to defined the upstream point and downstream point on the reach, which has only one branch.

3.2.8.1 Energy balances terms in `ENERGY_BALANCES_HEAT` output files

The `ENERGY_BALANCES_HEAT` output files contain the following columns separated by semi-colon (`;`),

- `Date`, date (format `FR_TS` or `US_TS`).
- `t`, Julian day, the unit is set by `TIME_UNIT`.
- `FIN_LATERAL`, lateral energy fluxes.
- `SHORTWAVE`, shortwave radiation.
- `LONGWAVE`, longwave radiation.
- `EVAPO_LATENT`, latent heat.
- `CONVECT_SENSIBLE`, sensible heat.
- `FIN`, energy fluxes inflow at upstream point.
- `FOUT`, energy fluxes outflow at downtream point.
- `EINIT`, initial energy at starting of timespan.
- `EEND`, final energy at end of timespan.
- `ERROR`, energy balance error.

Appendixes

Appendixes

A The `.COMM` file : Fully detailed template 71

B List of units recognized in the command `.COMM` file 79

A The .COMM file : Fully detailed template

A complete .COMM file template including all PROSE-PA0.7x features is detailed below, showing the entire set of embedded bracket sub-blocks. Overall, this example corresponds to the two_reaches case study.

```
INPUT_FOLDERS   = $HOME/Programmes/prose-pa/test_cases/two_reaches/
                = $HOME/Programmes/prose-pa/test_cases/two_reaches/Cmds_files/
                = $HOME/Programmes/prose-pa/test_cases/two_reaches/layers
                = $HOME/Programmes/prose-pa/test_cases/two_reaches/param_bio/
    waterspecies
                = $HOME/Programmes/prose-pa/test_cases/two_reaches/Singularities
                = $HOME/Programmes/prose-pa/test_cases/two_reaches/Reaches
                = $HOME/Programmes/prose-pa/test_cases/two_reaches/bathymetry
                = $HOME/Programmes/prose-pa/test_cases/two_reaches/inflows

OUTPUT_FOLDER   = $HOME/OUTPUT_ProSe-PA/OUTPUT_FULL_TEMPLATE/

! - - - - - - - - - - - - - - - - - - - - - - - End of FLOD-IN and FOLD-OUT blocks
SIMULATION = {
    test_simulation

    TIME = {
        DATE_FORMAT  = FR_TS
        YEAR0_JULIAN = 2007
        T_INI        = 01/01/2007 00:00:00 ! [d] 0.
        T_END        = 01/01/2008 00:00:00 ! [d] 365.
        DT           = [min] 15.
        NSTEP_DA     = 1
    }
! - - - - - - - - - - - - - - - - - - - - - - - End of SIMTIME block
    SETTINGS = {
        REGIME            = transient
        HYD               = yes
        TTC               = yes
        RIVE              = yes
        TUBE              = no
        DYNAMIC_T         = yes
        DA                = yes
        DA_METHOD         = PF
        NUM_THREAD        = 20
        NUM_PARTICULES    = 10
        NUM_THREAD_PAR    = 1
        NUM_TUBE_DEF      = 1
        ALPHA_DA          = 0.3
        ERROR_OBS         = 0.10
        S_PERCENT         = 0.10
        RANDOM_WALK       = loop
        SOLVER            = sp_prose
        DEFAULT_T_INFLOWS = [°C] 20.
        NB_COMP_PHY       = 3
        DBO_OXY           = [molO2/molC] 1.0
! - - - - - - - - - - - - - - - - - - - - - - - End of SIMSET-GSET block
        SPECIES = {
            WATER = {
                ! Particulate species
                include phy
                include mop
                include bact
                include bactn
                include mes
                include zoo
                include bif
                ! Dissolved species
                include o2
                include no3
                include mod
                include nh4
                include po4
                include no2
                include si_d
            }
        }
! - - - - - - - - - - - - - - - - - - - - - - - End of SIMSET-SPE block
        MACROSPECIES = {
            TOC = {
                total_organic_carbon
                SHARE_MOP = {
                    BIODEGRADABLE      = {
                        VAL     = 0.3
                    }
                    FAST_BIODEGRADABLE = { val = 0.7 }
                }
                SHARE_MOD = {
                    BIODEGRADABLE      = {
                        VAL     = 0.3
                    }
                    FAST_BIODEGRADABLE = { val = 0.7 }
                }
            }
        }
! - - - - - - - - - - - - - - - - - - - - - - - End of SIMSET-MSPE block
        HYDRAULIC = {
            NDIM                = 1
            CALCULATE_CURVATURE = no
            CURVATURE           = 0
            DX                  = [m] 500
            EPS_Q               = [m^3/s] 0.000001
            EPS_Z               = [m] 0.000001
            GLOBAL_STRICKLER    = 20.
            DZ                  = [m] 0.1
            UPSTREAM_HMIN       = [m] 0.
            DOWNSTREAM_HMAX     = [m] 20.
            THETA               = 0.9
            SCHEM_TYPE          = ST_VENANT
            INITIALIZATION      = {
                INIT_Z_FILE = include init_Z
                INIT_Q_FILE = include init_Q
            }
            NETWORK_DESCRIPTION = {
                SINGULARITIES = { include sing_case_study }
                REACHES       = { include reaches_case_study }
                PROFILES      = { include profils_case_study }
                INFLOWS       = { include inflows_case_study }
            }
        }
! - - - - - - - - - - - - - - - - - - - - - - - End of SIMSET-HYD block
        METEO = {
            MET_CORRESP = forcing_to_reach_corresponding.txt
            MET_DATA    = {
                FOLDER      = $HOME/Programmes/prose-pa/test_cases/two_reaches/DATA_METEO/
                YEAR0_METEO = 1850
                FORCINGS    = {
                    T_AIR  = TMPH_case_study.txt
                    P_ATM  = PRESSURE_GRID_SAFRAN.dat
                    H_SW   = SWDH_case_study.txt
                    H_LW   = LWDH_case_study.txt
                    HUMID  = HRLH_case_study.txt
                    U_WIND = VNTH_case_study.txt
                }
            }
        }
! - - - - - - - - - - - - - - - - - - - - - - - End of SIMSET-MTO block
        HEAT_TRANSPORT  = {
            THETA_T     = 1.0
            INIT_T_FILE = [°C] include init_T
        }
! - - - - - - - - - - - - - - - - - - - - - - - End of SIMSET-HT block
        TRANSPORT =  {
            THETA_TTC = 0.9
        }
! - - - - - - - - - - - - - - - - - - - - - - - End of SIMSET-ST block
        BIOLOGY = {
            NUMERICAL_METHOD = EXPLICIT
            MAX_DIV_DT       = 1.0
            DZ_RIVE          = [m] 0.1
            LAYERS           = {
                include watercomp
                include vasecomp
            }
            EXCHANGES          = {
                CALC_SEDIM_EROS        = SIMULTANEOUS_PROSE
                ETA_HYD                = [g/cm^3] 0.0013
                PNAVIG                 = [g/m^2/s] 0.0005
            }
            METEO             = {
                TEMPERATURE = {
                    MEAN               = [°C] 12.
                    AMPLITUDE          = [°C] 20.
                    DELAY              = [h]  29.
                }
                WIND        = { [d] [m/s]
                                0. 2.
                }
                RADIATION   = {
                    MEAN               = [uE/m^2/s] 542
                    AMPLITUDE          = [uE/m^2/s] 288
                    ATTENUATION_FACTOR = 1
                }
                PHOTOPERIOD = {
                    MEAN               = [h] 12.
                    AMPLITUDE          = [h] 4.2
                }
            }
        }
! - - - - - - - - - - - - - - - - - - - - - - - End of SIMSET-BIO block
        param_range_da = { include param_range }
! - - - - - - - - - - - - - - - - - - - - - - - End of SIMSET-DA block
    } ! - - - - - - - - - - - - - - - - - - - - - - - End of SIMSET block
} ! - - - - - - - - - - - - - - - - - - - - - - - End of SIM block

OUTPUTS = {
    PRINT_PK    = {
        yes
        FILE_NAME = pks
    }
    FINAL_STATE = {
        yes
        FILE_NAME = seine_final }
    MB_ELEMENTS = { yes }
    TIME_SERIES = {
        TIME        = { DT = [min] 15. }
        TIME_UNIT   = [s]
        VAR         = {
            ! Hydraulic variables
            hh    [m]
            zfs   [m]
            wfs   [m]
            surf  [m^2]
            Disch [m^3/s]
            Vel   [m/s]
            peri  [m]
            rh    [m]
            ! Biogeochemical species
            phy   [mgC/l]
            zoo   [mgC/l]
            o2    [mgO2/l]
            nh4   [mgN/l]
            no3   [mgN/l]
            no2   [mgN/l]
            bif   [mgC/l]
            bact  [mgC/l]
            bactn [mgC/l]
            po4   [mgN/l]
            mod   [mgC/l]
            mop   [mgC/l]
            mes   [mg/l]
            si_d  [mgSI/l]
            ! Temperature
            tw    [°C]
            ta    [°C]
            ! Sediment layer realted variables
            sedvol [m^3/m]
            sedh   [cm]
        }
        POINTS     = {
            Seine
            [km] 50. 1
            [km] 60. 1
            [km] 90. 1
        }
        POINTS_OBS = {
            Seine
            [km] 60. 1 o2 1 [s] [mgO2/l] include o2_obs_pk60
            [km] 90. 1 o2 1 [s] [mgO2/l] include o2_obs_pk90
        }
    }
! - - - - - - - - - - - - - - - - - - - - - - - End of TIME_SERIES block
    LONGITUDINAL_PROFILES = {
        TIME        = {
            T_INI = 01/01/2007 ! [d] 0.
            T_END = 02/01/2007 ! [d] 1.
            DT    = [h] 12.
        }
        TIME_UNIT   = [d]
        VAR         = {
            ! Hydraulic variables
            hh    [m]
            zfs   [m]
            wfs   [m]
            surf  [m^2]
            Disch [m^3/s]
            Vel   [m/s]
            peri  [m]
            rh    [m]
            ! Biogeochemical species
            phy   [mgC/l]
            zoo   [mgC/l]
            o2    [mgO2/l]
            nh4   [mgN/l]
            no3   [mgN/l]
            no2   [mgN/l]
            bif   [mgC/l]
            bact  [mgC/l]
            bactn [mgC/l]
            po4   [mgN/l]
            mod   [mgC/l]
            mop   [mgC/l]
            mes   [mg/l]
            si_d  [mgSI/l]
            ! Temperature
            tw    [°C]
            ta    [°C]
            ! Sediment layer related variables
            sedvol [m^3/m]
            sedh   [cm]
        }

        EXTENT = {
            Seine
            [km] 0.      1
            [km] 99.999 1
        }
    }
! - - - - - - - - - - - - - - - - - - - - - - - End of LONGITUDINAL_PROFILES block
    TUBE_OUTPUT = {
        MESH_TUBE = {
            T_INI     = 01/01/2007 ! [d] 0.
            T_END     = 02/01/2007 ! [d] 1.
            DT        = [h] 12.
            TIME_UNIT = [d]
        }
        HYD_TUBE = {
            T_INI     = 01/01/2007 ! [d] 0.
            T_END     = 02/01/2007 ! [d] 1.
            DT        = [h] 12.
            TIME_UNIT = [d]
        }
    }
! - - - - - - - - - - - - - - - - - - - - - - - End of TUBE_OUTPUT block
    MASS_BALANCES_BIO = {
        mb1 = {
            T_INI   = 01/01/2007 ! [d] 0.
            T_END   = 02/01/2007 ! [d] 1.
            N_STEPS = 1.
            SPECIES = {
                phy [mol]
                o2 [mol]
                nh4 [mol]
                no3 [mol]
                no2 [mol]
                bact [mol]
                bactn [mol]
                po4 [mol]
                mod [mol]
                mop [mol]
                mes [kg]
            }
            EXTENT = {
                Seine
                [km] 0. 1
                [km] 49.99 1
            }
        }
        mb2 = {
            T_INI   = 01/01/2007 ! [d] 0.
            T_END   = 02/01/2007 ! [d] 1.
            N_STEPS = 1.
            SPECIES = {
                phy [mol]
                o2 [mol]
                nh4 [mol]
                no3 [mol]
                no2 [mol]
                bact [mol]
                bactn [mol]
                po4 [mol]
                mod [mol]
                mop [mol]
                mes [kg]
            }
            EXTENT = {
                Seine
                [km] 0. 1
                [km] 49.99 1
            }
        }
    }
! - - - - - - - - - - - - - - - - - - - - - - - End of MASS_BALANCES_BIO block
    ENERGY_BALANCES_HEAT = {
        eb1 = {
            T_INI     = 01/01/2007 ! [d] 0
            T_END     = 02/01/2007 ! [d] 1.
            N_STEPS   = 1
            TIME_UNIT = [s]
            HEAT_UNIT = [J]
            EXTENT    = {
                Seine
                [km] 0. 1
                [km] 99.999 1
            }
        }
    }
! - - - - - - - - - - - - - - - - - - - - - - - End of ENERGY_BALANCES_HEAT block
}  ! ----------------------- End of general OUTPUTS block
```

B List of units recognized in the command .COMM file

Several specifiers can be used to declare units of input parameters or time series, with the most suitable unit from the user's point of view. The list below features units recognized and authorized by PROSE-PA0.77 in order to define inputs and/or outputs units. All combinations (especially for division and power operations) of these letters or strings (*e.g.* kg and ha to form the kg/ha unit) will be interpreted when parsing the command file and its sub-files targeted by the `include` command, to determine the appropriate multiplicative factor in order to transpose all input values into a reference unit, valid for all PROSE-PA code and its libraries. These reference units are as follows:

- the second [s] for a time unit,
- the meter [m], for a length unit,
- the gram [g], for a mass unit,
- the [mmol], for a matter quantity,
- the [K] and [°C], for a temperature.
- the micro-Einstein[uE], for energy

The following characters and/or strings are recognized by PROSE-PA0.77:

– TIME : [min] = minute, [s] = second, [h] = hour, [d] = day, [yr] = year (365 days only),

– LENGTH/SURFACE : [m] = meter, [km] = kilometer, [cm] = centimeter, [mm] = millimeter. The powered units are also interpreted, such as [m^X] = squared meters, cube meters, etc. ($X \in 2,3,...$), as well as [km^X] = squared kilometers, etc.

– MASS: [ug] = microgram, [mg] = milligram, [g] = gram, [kg] = kilogram, and [t] : ton,

– MOLAR MASS of C, N, NH4, NO2, NO3, P, PO4, O2, CHLA, SI, SIO2 : same as MASS with the code of the molecule directly paste in capital letters to the mass unit, examples mgC, gNO3, mgO2, ...

– VOLUME : [l] = liter,

– MATTER QUANTITY : [mol] = mol, [mmol] = millimol, and [umol] = micromol.

– TEMPERATURE : [K] = Kelvin, [°C] = Celsius degree.

– ENERGY, [uE] = micro Einstein, [J] = Joule, and [cal] = Calorie.

References

VANNOTE, R., MINSHALL, G. W., CUMMINS, K., SEDELL, J. et CUSHING, C. (1980). The River Continuum Concept. *Can. J. Fish. Aquat. Sci.*, 37(1):130–137. 1

VILMIN, L., FLIPO, N., de FOUQUET, C. et POULIN, M. (2015). Pluri-annual sediment budget in a navigated river system: The Seine River (France). *Sciences of Total Environment*, 502:48–59. 33

WANG, S. et FLIPO, N. (2020). Influence des flux de sédimentation-érosion sur les flux de carbone dans la Seine. Rapport technique, PIREN Seine. 33

WANG, S., THIEU, V., BILLEN, G., GARNIER, J., SILVESTRE, M., MARESCAUX, A., YAN, X. et FLIPO, N. (2024). The community-centered freshwater biogeochemistry model unified rive v1.0: a unified version for water column. *Geoscientific Model Development*, 17(1):449–476. 2, 19, 35, 67