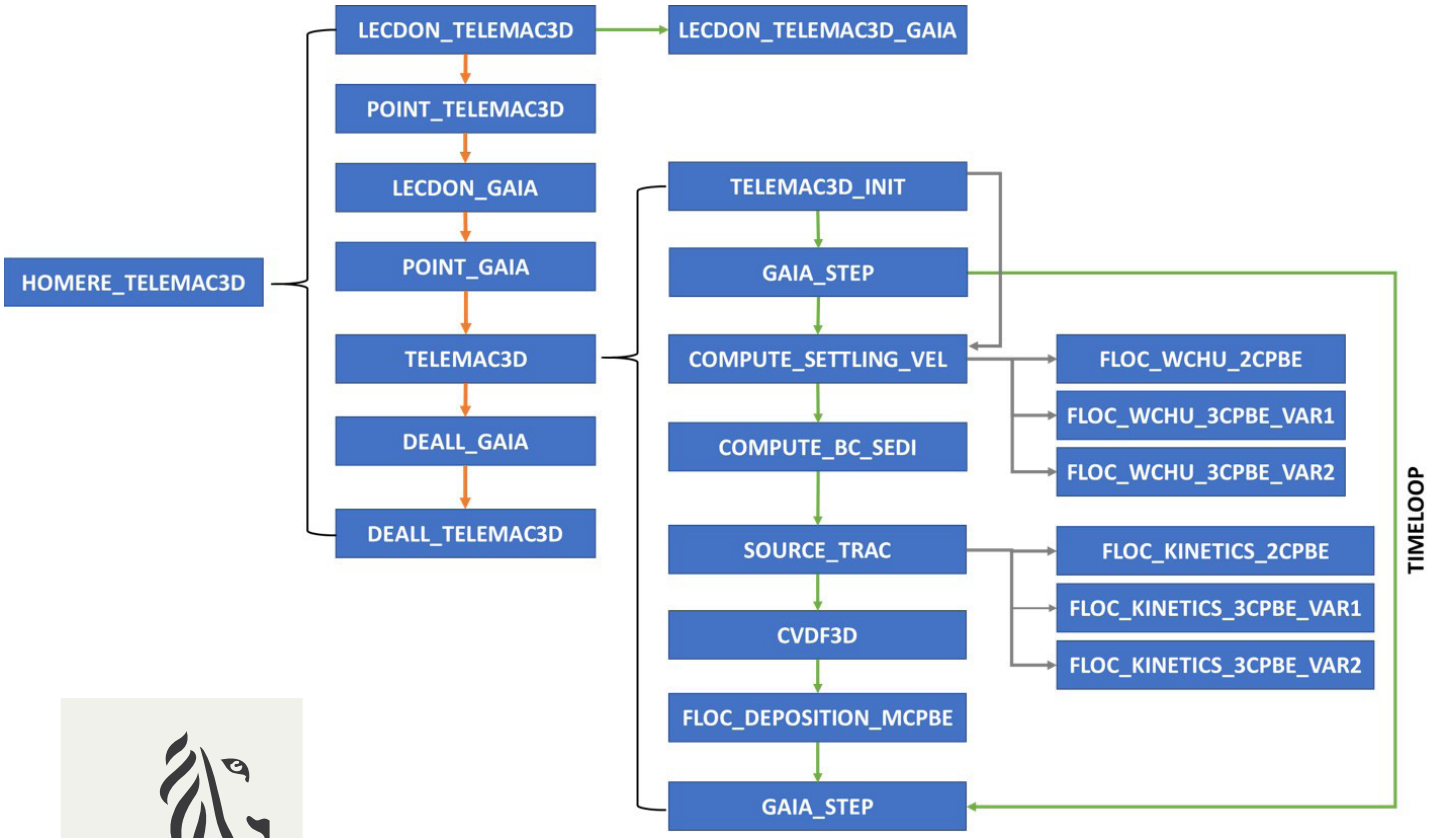

Flanders
State of
the Art

18_043_1
FHR reports

Implementation of Multi-class Flocculation Model in TELEMAC Modelling System

Sub report 1
The code development

DEPARTMENT
MOBILITY &
PUBLIC
WORKS

www.flandershydraulicsresearch.be

Implementation of Multi-class Flocculation Model in TELEMAC Modelling System

Sub report 1 – The code development

Bi, Q.; Vanlede, J.

Flanders Hydraulics Research
Flanders State of the Art

Cover figure © The Government of Flanders, Department of Mobility and Public Works, Flanders Hydraulics Research

Legal notice

Copyright and citation

This publication should be cited as follows:

Bi, Q.; Vanlede, J. (2022). Implementation of Multi-class Flocculation Model in TELEMAC Modelling System: Sub report 1 – The code development. Version 1.0. FHR Reports, 18_043_1. Flanders Hydraulics Research: Antwerp

Document identification

Customer:	Flanders Hydraulic Research	Ref.:	WL2022R18_043_1
Keywords (3-5):	Cohesive Sediment, Flocculation Models, TELEMAC-3D, GAIA		
Knowledge domains:	Hydraulics and sediment > Sediment > Cohesive sediment > Numerical modelling		
Text (p.):	40	Appendices (p.):	48
Confidential:	☒ No	☒ Available online	

Author(s):	Bi, Q.

Control

	Name	Signature
Reviser(s):	Vanlede, J.	Getekend door:Joris Vanlede (Signature) Getekend op:2023-01-20 10:45:54 +01:0 Reden:Ik keur dit document goed Joris Vanlede
Project leader:	Bi, Q.	Getekend door:Qilong Bi (Signature) Getekend op:2022-12-16 10:15:58 +01:0 Reden:Ik keur dit document goed Qilong Bi

Approval

Head of Division:	Bellafkih, K.	Getekend door:Abdelkarim Bellafkih (Sign Getekend op:2022-12-16 10:23:40 +01:0 Reden:Ik keur dit document goed Abdelkarim Bellafkih

Abstract

Implementation of multi-class population balance equation based flocculation model in TELEMAC-MASCARET system is an internal project in collaboration with KU Leuven, Kyungpook National University (South Korea), and the developers team at EDF R&D (France).

TELEMAC-MASCARET is a suite of sophisticated modelling tools used in the field of hydrodynamic and sediment transport modelling in coastal, estuarine and riverine systems. It is also one of the main modelling platform used in the Flanders Hydraulics Research for research and engineering projects.

The general objective of this internal project is to improve the functionality of modelling flocculation processes in the TELEMAC system by incorporating more advanced flocculation models based on two-class population balance equations (2CPBEs) (Lee et al., 2011) and three-class population balance equations (3CPBEs) (Shen et al., 2018a, 2018b) developed at KU Leuven, and make it available as a useful tool for future research and studies. Another important task in this project is to investigate the influence of flocculation on large-scale sediment transport, which will be elaborated in a separate validation report.

This report focuses on the code development of the MCPBE flocculation models and the procedures of adding a new module in TELEMAC-MASCARET.

Contents

List of tables

List of figures

1 Introduction

Flocculation is a key feature of cohesive sediment, e.g. mud particles, found in the natural environment, especially in marine and estuarine systems. Mud particles in an aquatic environment tend to stick together as the result of the van der Waals forces into aggregates or flocs (Toorman & Berlamont, 2018).

By implementing a sophisticate 3D flocculation model in TELEMAC system, the cohesive sediment processes of flocculation, advection, dispersion, erosion resuspension and sedimentation can be simulated more precisely in three dimensions by incorporating the flocculation kinetics (as source and sink terms which are tracked in time) than the existing empirical flocculation models (i.e. the currently adopted Soulsby flocculation model). This is due to the fact that the two-class TCPBE model actually determines the flocculation kinetics and the existing Souslby flocculation model assumes a quasi-equilibrium and adopts physics-based formulae for determination of the settling velocity and mass flux. It should come as no surprise that incorporating the actual flocculation kinetics is a major advantage in comparison with equilibrium flocculation models as it can differentiate various states in space and time to simulate the bimodal flocculation patterns (e.g. during a tidal cycle) (Lee et al., 2015).

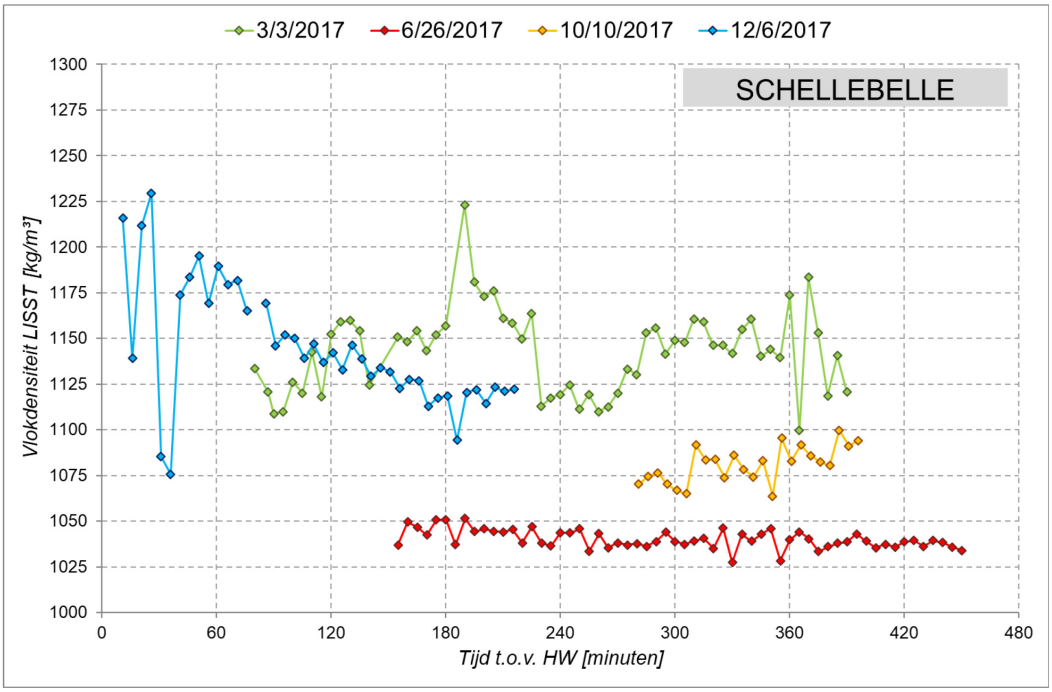

Figure 1 – Seasonal variation of sediment particle density measured in Schellebelle, which indicates the flocculation process and the influence from biological effects (data from Flanders Hydraulics Research).

Moreover, the mechanistic nature of the model would allow straightforward adaptation of the flocculation kinetics (i.e. through physicochemical properties) and their structure (i.e. excess density and fractal dimension) when incorporating biologically mediated flocculation effects. These biological effects occur seasonally and have been frequently observed in nature during biologically active seasons. Biologically mediated flocculation occurs as bio-polymers interact with mud flocs and are often related to algae bloom periods and the presence of phytoplankton. These effects are also highly likely to be related to the occurrence of megaflocs (to be distinguished from microflocs and macroflocs in the model) which have median sizes over 200 microns and are formed due to the gluing capacity of EPS (extracellular polymeric substances) and TEP

(transparent exopolymer particles) (Lee & Schlautman, 2015). As the presence of these megaflocs significantly alters the mass settling flux and as they occur seasonally, the model should be able to account for this class besides the microflocs and macroflocs. This would entail an expansion of the population balance and underlying flocculation kinetic processes, although more efficient and straightforward strategies may be used.

The current large-scale sediment transport models usually lack the insights in flocculation processes. They commonly consider size-fixed sediment particles in the system and use non-changeable properties, e.g. density and settling velocity during the whole simulations, or adopt empirical formulas which are not suitable for complex applications. Due to the simplified transport processes, the accuracy of prediction provided by these models are usually not satisfying. Hence, the purpose of this study is to include this missing piece in the large-scale numerical modelling of sediment transport.

Implementation of an advanced flocculation model in TELEMAC-MASCARET system is an internal project in collaboration with two other partners from KU Leuven and Kyungpook National University (South Korea).

TELEMAC-MASCARET is a suite of sophisticated modelling tools used in the field of hydrodynamic and sediment transport modelling in coastal and estuarine systems. It is also one of the main modelling platforms used by the Flanders Hydraulics Research in its research and projects.

The general objective of this internal project is to improve the functionality of modelling flocculation processes in the TELEMAC system by implementing more advanced flocculation models based on two-class population balance equations (2CPBEs) (Lee et al. 2011) and three-class population balance equations (3CPBEs) (Shen et al. 2018a, 2018b), and make it available as a useful tool for future research and projects.

2 The MCPBE Flocculation Model

Flocculation is a key feature of cohesive sediment, e.g. mud particles, found in the natural environment, especially in marine and estuarine systems. Mud particles in an aquatic environment tend to stick together as the result of the van der Waals forces into aggregates or flocs (Toorman & Berlamont, 2018).

Floc suspensions usually exist as bimodal distributions (i.e. microflocs or flocculi, and macroflocs) which is observed in the marine and estuarine environment. Bimodal flocculation of marine and estuarine sediments describes the aggregation and breakage process in which dense microflocs and floppy macroflocs change their relative mass fraction and develop a bimodal floc size distribution. To simulate bimodal flocculation of such sediments, a flocculation model based on Two-Class Population Balance Equations (2CPBEs), which includes both size-fixed microflocs (or flocculi) and size-varying macroflocs, was developed (Lee et al. 2011). Tri-modal distributions are also frequently found, especially around algae bloom periods. Shen et al. (2018a, 2018b) extended the 2CPBE model to 3CPBE model in order to account additional flocculation processes in such case.

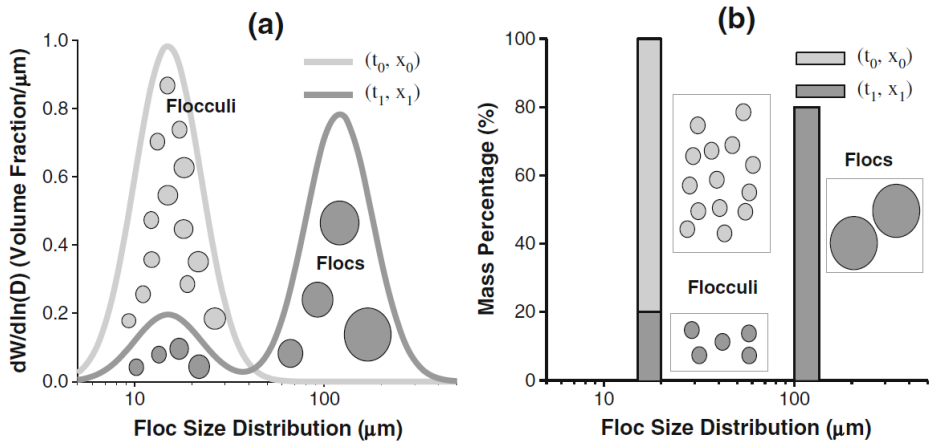

Figure 2 – The bimodal distributions of flocs in natural marine and estuarine environment (a), and the model scheme in 2CPBE using size-fixed microflocs and size-varying macroflocs (b).

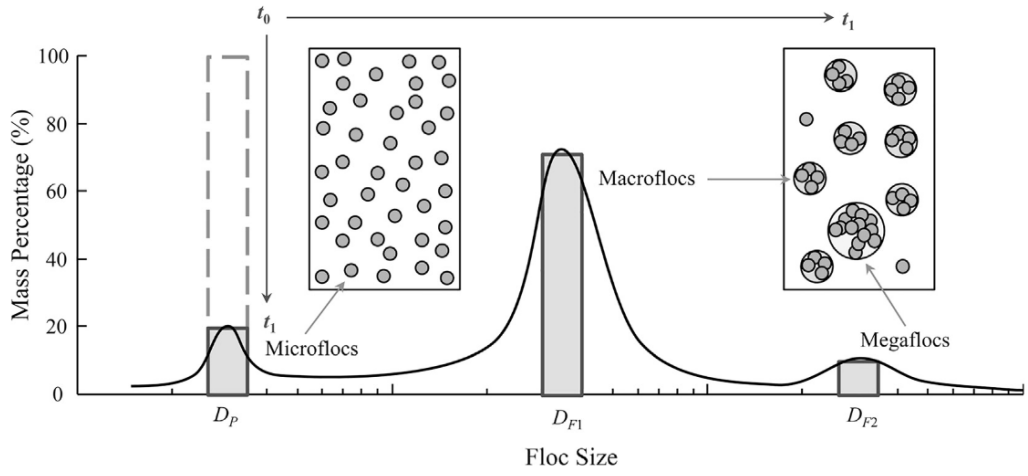

Figure 3 – Schematic diagram of the FSDs before and after flocculation. At time t_0, all particles are concentrated on microflocs. With time, macroflocs and megaflocs have appeared because of aggregation and breakage processes. (Shen et al. 2018a)

Single-class flocculation models do not suffice in quantifying sediment transport fluxes accurately and they are fundamentally incapable of simulating bimodal or multimodal flocculation patterns observed in nature as they are restricted to only one single class. Multi-class models (size or distribution based) have been used and investigated and provide accurate results in most cases, yet applying them to larger scale situations is often fundamentally limiting through computation time as the number of floc size classes should be very large in order to incorporate the entire particle size distribution observed in nature. Therefore, 2CPBE and 3CPBE flocculation models are developed with a minimum amount of classes that is necessary to simulate bimodal or tri-modal flocculation, and remain accurate yet computationally efficient by tracking the floc size and concentrations of all the classes (microflocs and macroflocs in 2CPBE, and an additional class of megafloc in 3CPBE) as function of space and time. A general mathematical model for the multi-class PBEs in a 3D fluid field may be written as:

$$\left[\frac{\partial N_i}{\partial t}\right] \quad \text{(I)}$$

$$+\left[\frac{\partial}{\partial x}(u_x N_i)+\frac{\partial}{\partial y}(u_y N_i)+\frac{\partial}{\partial z}(u_z N_i)\right] \quad \text{(II)}$$

$$-\left[\frac{\partial}{\partial x}\left(D_{tx}\frac{\partial N_i}{\partial x}\right)+\frac{\partial}{\partial y}\left(D_{ty}\frac{\partial N_i}{\partial y}\right)+\frac{\partial}{\partial z}\left(D_{tz}\frac{\partial N_i}{\partial z}\right)\right] \quad \text{(III)}$$

$$=(A_i+B_i)-\frac{\partial(w_{s,i}N_i)}{\partial z} \quad \text{(IV)} \qquad (1)$$

In the above equation, N_i = number concentration of the *i*th particle size class, *x*, *y*, *z*, *t* = position and time, u_x, u_y, u_z = mean fluid velocities, D_{tx}, D_{tx}, D_{tz} = turbulent dispersion coefficients, A_i and B_i = growth and decay kinetics of N_i by aggregation and breakage, respectively, and $w_{s,i}$ = settling velocity of the *i*th particle class due to gravity. On the left-hand side of the equation, the respective terms in brackets represent the storage change (I), the particle mean advection (II), and the turbulent diffusion of flocs (III), while on the right-hand side, the source/sink terms (IV) represent the net effects of floc aggregation, breakage and settling due to gravity. In case of 2CPBE model, in the above equation *i* = *P (microflocs)*, *F1 (macroflocs)* and *T1 (microflocs in macroflocs)*. Thus, N_P represents number concentration of microflocs in suspension, N_{F1} the number concentration of macroflocs in suspension, and the composition of macroflocs through N_{T1} the number concentration of microflocs in macroflocs. In case of 3CPBE model, there are two additional classes, *F2* and *T2*, representing megaflocs and microflocs in megaflocs, respectively.

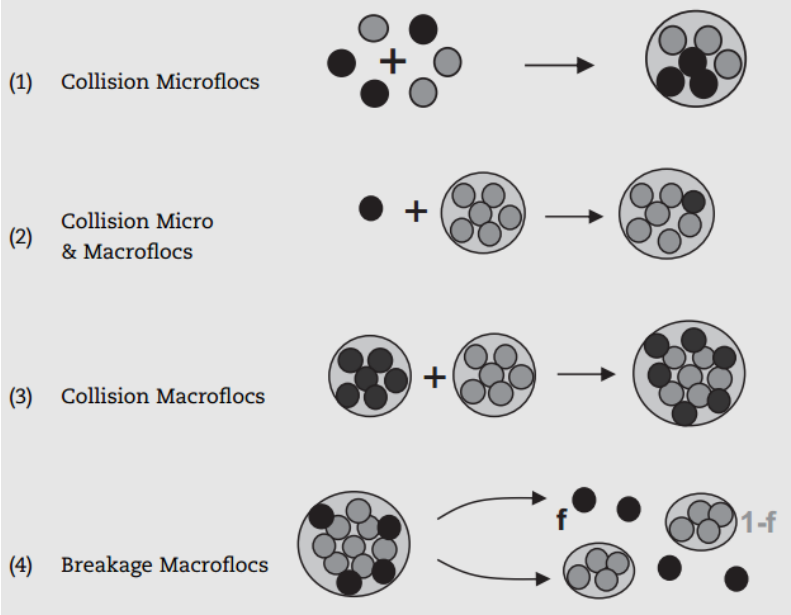

Figure 4 – The flocculation kinetics modelled in 2CPBE flocculation model, including aggregation and breakage (Lee et al. 2011).

In case of 2CPBE flocculation model, the aggregation and breakage terms describe 4 different mechanisms (Figure 4) and can be expressed as follows:

$$(A_P + B_P) = -\frac{1}{2}\alpha\beta_{PP}N_P N_P\left(\frac{N_{C1}}{N_{C1}-1}\right) - \alpha\beta_{PF}N_P N_{F1} + f_{P1}N_{C1}a_{F1}N_{F1} \quad (2)$$

$$(A_{F1} + B_{F1}) = \frac{1}{2}\alpha\beta_{PP}N_P N_P\left(\frac{1}{N_{C1}-1}\right) - \frac{1}{2}\alpha\beta_{F1F1}N_{F1}N_{F1} + a_{F1}N_{F1} \quad (3)$$

$$(A_{T1} + B_{T1}) = \frac{1}{2}\alpha\beta_{PP}N_P N_P\left(\frac{N_{C1}}{N_C-1}\right) + \alpha\beta_{PF1}N_P N_{F1} - fN_{C1}a_{F1}N_{F1} \quad (4)$$

where, N_{C1} = number of microflocs in a macrofloc (N_{T1}/N_{F1}), f_{P1} = fraction of microflocs generated by breakage of macroflocs, whereas (1-f_{P1}) = fraction of smaller macroflocs by breakage of larger macroflocs, α = collision efficiency, β_{ij} = collision frequency, and a_{F1} = breakage frequency of macroflocs.

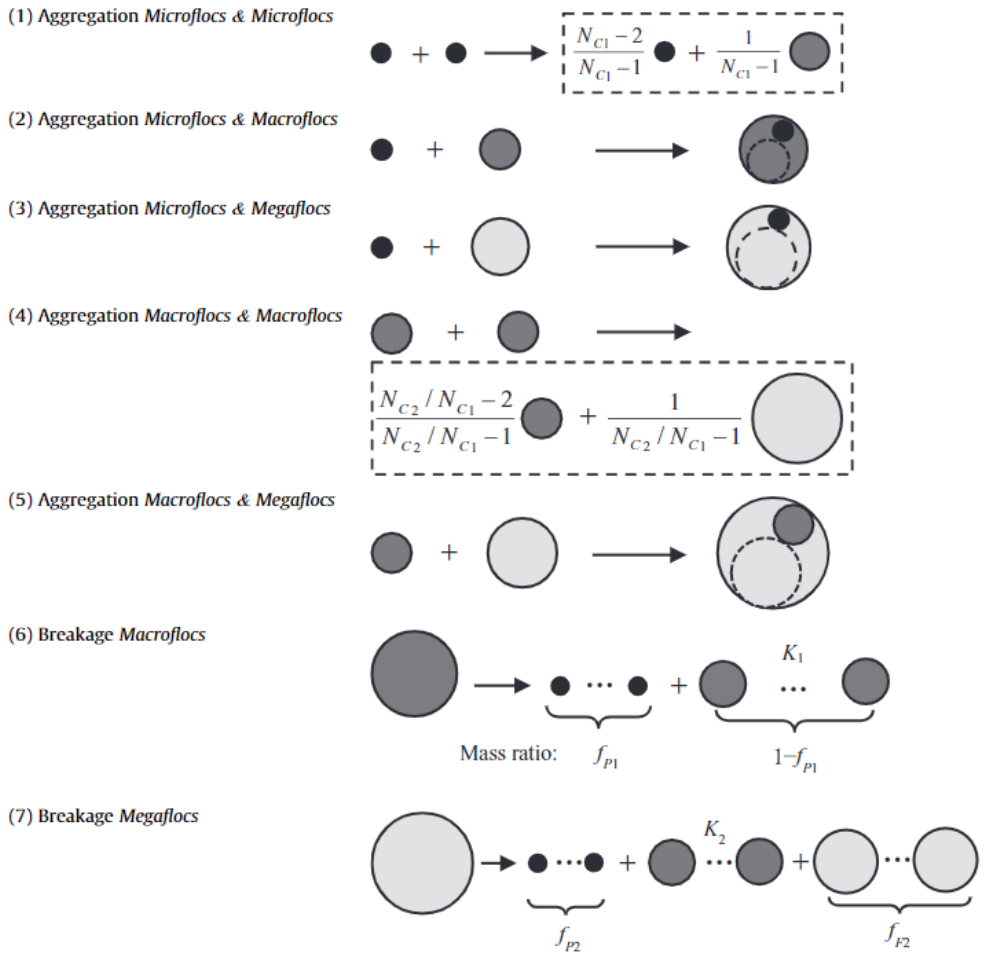

Figure 5 – The flocculation kinetics modelled in 3CPBE flocculation model with fixed megaflocs, including aggregation and breakage (Shen et al. 2018a).
For the 3CPBE with varying megaflocs, in additional to these mechanisms, aggregation of megaflocs and megaflocs is also considered (Shen et al. 2018b).

In case of 3CPBE flocculation models, the mechanisms in the flocculation kinetics are listed in Figure 5. The aggregation and breakage terms hence become more complicated:

$$(A_P + B_P) = -\frac{1}{2}\alpha\beta_{PP}N_P N_P\left(\frac{N_{C1}}{N_{C1}-1}\right) - \alpha\beta_{PF1}N_P N_{F1} - \alpha\beta_{PF2}N_P N_{F2} + f_{P1}N_{C1}a_{F1}N_{F1} + f_{P2}N_{C2}a_{F2}N_{F2} \quad (5)$$

$$(A_{F1} + B_{F1}) = \frac{1}{2}\alpha\beta_{PP}N_P N_P\left(\frac{1}{N_{C1}-1}\right) - \frac{1}{2}\alpha\beta_{F1F1}N_{F1}N_{F1}\left(\frac{N_{C2}/N_{C1}}{N_{C2}/N_{C1}-1}\right) - \alpha\beta_{F1F2}N_{F1}N_{F2} + (K_1-1)a_{F1}N_{F1} + K_2 a_{F2}N_{F2} \quad (6)$$

$$(A_{T1} + B_{T1}) = \frac{1}{2}\alpha\beta_{PP}N_P N_P\left(\frac{N_{C1}}{N_{C1}-1}\right) + \alpha\beta_{PF1}N_P N_{F1} - \frac{1}{2}\alpha\beta_{F1F1}N_{F1}N_{F1}\left(\frac{N_{C2}}{N_{C2}/N_{C1}-1}\right) - N_{C1}\alpha\beta_{F1F2}N_{F1}N_{F2} - f_{P1}N_{C1}a_{F1}N_{F1} + (1-f_{P2}-f_{F2})f_{P2}N_{C2}a_{F2}N_{F2} \quad (7)$$

$$(A_{F2} + B_{F2}) = \frac{1}{2}\alpha\beta_{F1F1}N_{F1}N_{F1}\left(\frac{N_{C1}}{N_{C2}/N_{C1}-1}\right) - \frac{1}{2}\alpha\beta_{F2F2}N_{F2}N_{F2} + (K_3-1)N_{C2}a_{F2}N_{F2} \quad (8)$$

$$(A_{T2} + B_{T2}) = \alpha\beta_{PF2}N_P N_{F2} + \frac{1}{2}\alpha\beta_{F1F1}N_{F1}N_{F1}\left(\frac{N_{C2}}{N_{C2}/N_{C1}-1}\right) + N_{C1}\alpha\beta_{F1F2}N_{F1}N_{F2} - (1-f_{F2})N_{C2}a_{F2}N_{F2} \quad (9)$$

where, N_{C2} = number of microflocs in a megafloc (N_{T2}/N_{F2}), f_{P2} = fraction of microflocs generated by breakage of macroflocs, f_{F2} = fraction of remaining megaflocs by breakage of a larger megafloc, and a_{F2} = breakage frequency of megaflocs. Note that eq.(8) only exists in the 3CPBE model with varying megaflocs.

In above equations (for both 2CPBE and 3CPBE cases), the collision efficiency α is a fitting parameter, and the collision frequency β_{ij} can be expressed as (Thomas et al., 1999; Maggi, 2005) with a linear combination of three mechanisms (terms):

$$\beta_{ij} = \frac{1}{6}G\left(D_i + D_j\right)^3 + \frac{\pi}{4}\left(D_i + D_j\right)^2\left|w_{s,i} - w_{s,j}\right| + \frac{2}{3}\frac{K_B T}{\mu}\frac{\left(D_i + D_j\right)^2}{D_i D_j} \quad (10)$$

where G is the shear rate, us is the settling velocity, K_B is the Boltzmann constant, T is the absolute temperature and μ is the fluid dynamic viscosity, D is the characteristic floc size, w_s is the settling velocity given by a fractal-corrected Stokes equation with hindered settling corrections (Winterwerp and van Kesteren, 2004), i and j are the indices for *P, F1, or F2*. It is important to note that the effect of turbulent shear (the first term in eq.10) is the main mechanism in natural environments (Winterwerp, 1998). The effect of differential settling (second term in eq.10) is important during slack tide when turbulence is low (Lick et al., 1993), while the effect of Brownian motion (the third term in eq.10) is generally low for large particles (Winterwerp, 1998). The breakup frequency a can be written as (Winterwerp, 1998):

$$a_i = E_b G\left(\frac{D_i - D_P}{D_P}\right)^{3-n_{fi}}\left(\frac{\mu G}{F_y/D_i^2}\right)^q \quad (11)$$

where E_b is the breakage coefficient. The floc strength F_y, although not a constant (Kranenburg, 1999), is assumed 10^{-10} Pa in this study (Maggi et al., 2007; Verney et al., 2011), D is the characteristic floc size, n_f is the fractal dimension, $i = F_1, or\ F_2$ is the indices for macroflocs or megaflocs respectively. In the studies of Lee

et al. (2011) and Shen et al. (2018a, 2018b), *q* in the eq.(11) is always kept constant. Kuprenas et al. (2018) proposed a closure for *q* based on the observations from the experiment (Tran et al. 2018):

$$q = c_1 + c_2 \frac{D_i}{\eta} \tag{12}$$

where, c1 and c2 are the calibration parameters, D is floc size and η is the Kolmogorov length scale and *i*=*P*, *F1* or *F2*. This new formula is adopted in this study.

The floc characteristics is spatial-temporal dependent as the floc composition changes in the MCPBE flocculation model. The floc size *D* is given by the relation $D_i = D_P N_{Ci}^{1/n_{fi}}, i = F_1 \; or \; F_2$ according to Matsoukas and Friedlander (1991). The floc density is given by $\rho_i = \rho_w + (\rho_P - \rho_w) N_{Ci}^{1-3/n_{fi}}$ (Maerz et al. 2011). The settling velocity associated with each floc classes is given by a fractal-corrected Stokes equation with hindered settling corrections (Winterwerp and vanKesteren, 2004):

$$\omega_{s,i} = (1-\phi_i)^{\gamma} \left(\frac{1}{18} \frac{(\rho_P - \rho_w) g}{\mu} D_P^{3-n_{fi}} \frac{D_i^{n_{fi}-1}}{1 + 0.15 Re_i^{0.687}} \right) \tag{13}$$

where $\omega_{s,i}$ is the settling velocity of floc class *i*, ϕ is the volumetric concentration, γ is fitted parameter for hindered settling, ρ_P is the density of microflocs, ρ_w is the water density, g is the acceleration of gravity, μ is the dynamic viscosity of water, *D* is the floc size, n_{fi} is the fractal dimension of floc class *i*, Re_i is the Reynolds number of floc class *i*.

The cohesive sediment processes, e.g. flocculation, advection, dispersion, erosion resuspension and sedimentation, can be simulated more precisely in 3D by incorporating the flocculation kinetics described by the MCPBE flocculation model, than using the existing flocculation formula in TELEMAC, i.e. the Soulsby flocculation model (Souslby et al. 2013). This is due to the fact that the MCPBE flocculation model actually computes the flocculation kinetics in a dynamic way following the local changes of flow and sediment concentrations and characteristics, while the Soulsby model assumes a quasi-equilibrium, and adopts simplified relations for determination of the settling velocity. It should come as no surprise that incorporating the actual flocculation kinetics is a major advantage in comparison with equilibrium flocculation models as it can differentiate various states in space and time to simulate the bimodal flocculation patterns (e.g. during a tidal cycle) (Lee et al., 2015).

Moreover, the mechanistic nature of the model would allow straightforward adaptation of the flocculation kinetics (i.e. through physicochemical properties) and their structure (i.e. excess density and fractal dimension) when incorporating biologically mediated flocculation effects. These biological effects occur seasonally and have been frequently observed in nature during biologically active seasons. Biologically mediated flocculation occurs as bio-polymers interact with mud flocs and are often related to algae bloom periods and the presence of phytoplankton. These effects are also highly likely to be related to the occurrence of megaflocs (to be distinguished from microflocs and macroflocs in the model) which have median sizes over 200 microns and are formed due to the gluing capacity of EPS (extracellular polymeric substances) and TEP (transparent exopolymer particles) (Lee et al., 2015). As the presence of these megaflocs significantly alters the mass settling flux and as they occur seasonally, the model should be able to account for this class besides the microflocs and macroflocs. A recent study of Shen et al. (2018b) enhances the knowledge on the dynamics of SPMs, especially the biophysical influences on the fate and transport of estuarine aggregates. With the expansion of the 2CPBE model and underlying flocculation kinetic processes, the 3CPBE flocculation model could be able to capture the dynamics of bio-mediated suspended particulate matters under the seasonal variations of biomass as observed in Belgian coastal waters (southern North Sea).

3 The TELEMAC System

The TELEMAC-MASCARET system consists of a complete processing chain for the calculation of water, solute and sediment motions in the fluvial, coastal, estuarine and lacustrine domains. It comprises pre-processors for digitizing the data and describing the problem, simulation programs and post-processors for displaying and analysing the results. One of the key assets of the system is the use of the finite element theory that comprises a rigorous theoretical framework and a flexibility for describing complex geometries. There are several modules in the TELEMAC system and the most commonly used modules are as follows:

- **TELEMAC-2D** is a horizontal depth-averaged hydrodynamics solver. It solves the Saint-Venant (or shallow water) equations in two dimensions.
- **TELEMAC-3D** solves the three-dimensional Navier-Stokes equations with a free surface, it also provides the functionality of solving three-dimensional transport equations for passive and active travers when coupling with GAIA or WAQTEL.
- **GAIA** uses the results of the TELEMAC-2D and TELEMAC-3D to undertake simulation of bedload and suspended sediment transport through coupling. It offers various sediment transport formulas and multi-layer consolidation model.
- **TOMAWAC** models the changes of the power spectrum of wind-driven waves and wave agitation for applications in the oceanic domain, in the intracontinental seas as well as in the coastal zone.
- **WAQTEL** focuses on the water quality aspects. It was developed to allow users to tackle water quality problems together with hydrodynamic. In the recent versions, the **AED2 library** (the Aquatic Ecodynamics Modelling Library v2 developed by UWA) has been integrated into WAQTEL as a fully functional module, which offers possibility to model complex water quality related processes in TELEMAC.

4 Framework of Implementation

This implementation work is designed to be aligned with the long-term plan of the TELEMAC development. Therefore, the developer team from EDF R&D has been involved into this stage.

From the feedback of the developers, the flocculation model will ideally be integrated as a new process in WAQTEL (water quality module within TELEMAC system). By implementing in such way, it allows further extension of the flocculation model, e.g. including biological effects from other processes in WAQTEL, but meantime it still remains a relatively independent module and can be used in a more general way.

Based on this suggestion, a implementation framework is made with the knowledge of the code structure in TELEMAC system. The details are described in the following subsections.

4.1 The Procedures for Solving MCPBE Flocculation Model

Here gives an overview of the procedures for solving MCPBE flocculation model.

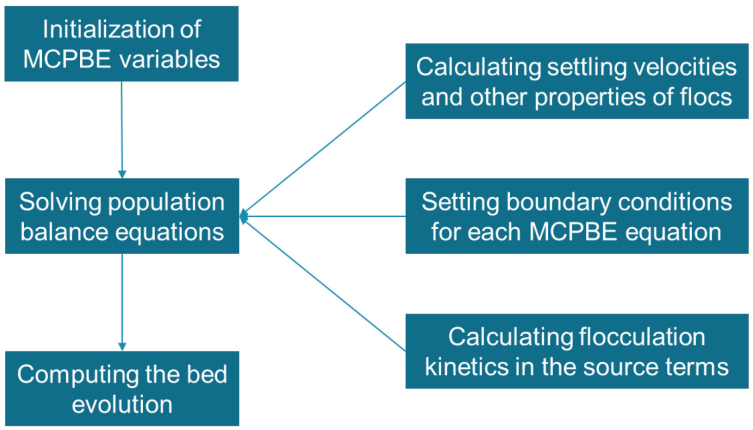

Figure 6 – The general procedures for solving TCPBE flocculation model in TELEMAC

Based on the procedures illustrated above, the implementation can be divided into 3 submodules:

- The initialization module will read the parameters from the steering file and initialize necessary variables for later use;
- The flocculation kinetics module will be responsible for computing aggregation/breakage kinetics, solving MCPBE and updating setting velocity and other properties of flocs;
- The bed layer module will be adapted to account for the interactions of flocculation process with bed layer, and update the bottom elevation at each time step.

4.2 Coupling TELEMAC-3D with GAIA

The TELEMAC-3D code solves three-dimensional Navier-Stokes equations (with or without the hydrostatic pressure hypothesis) and the transport-diffusion equations of intrinsic quantities (temperature, salinity, concentration). As one of the main modules in TELEMAC system, it can be coupled with other modules, e.g. GAIA, TOMAWAC and WAQTEL.

In the TELEMAC system, it is possible to allow TELEMAC-3D to be coupled simultaneously with the other three modules, which provides the possibility of using TELEMAC suite to model complex phenomena of sediment transport.

According to the plan from EDF R&D, a new framework of integrating sediment transport modules in TELEMAC system is in development. In this new framework, SEDI-3D (3D sediment transport functionality inside TELEMAC-3D) will be integrated into GAIA. This means the suspended load will be solved in the transport-diffusion equations in TELEMAC-3D, but bedload transport and bed-layer model will be moved into GAIA. In principle, the flocculation model does not only simulate flocculation kinetic in water column, it also needs a bed-layer model to determine its bottom boundary condition while solving MCPBEs. This interaction with the bed layer influences the bottom evolution and adds feedback to the hydrodynamics. Therefore, it is foreseen that solving flocculation kinetics will be implemented in TELEMAC-3D, and the interactions between the flocculation model and bed-layer will be implemented in GAIA.

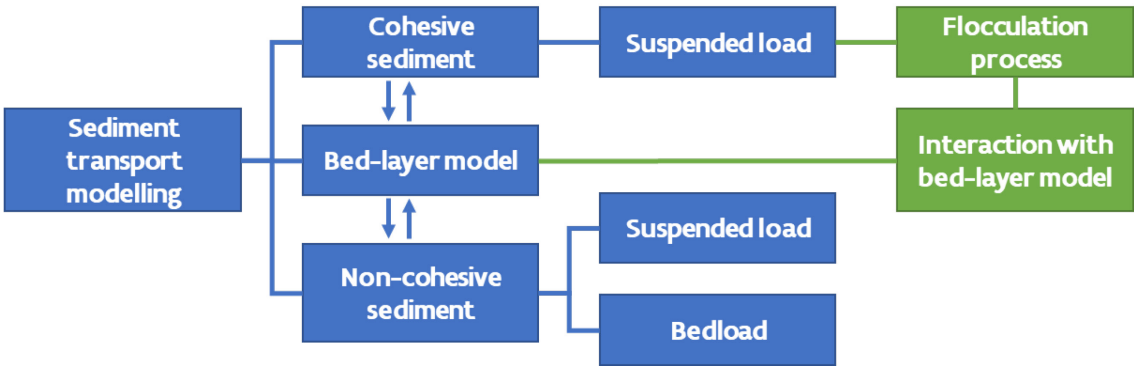

Figure 7 – The general scheme of modelling sediment transport

Figure 7 shows the a general framework for simulating sediment transport, which could be used the benchmark to test the functionality of modelling mixed sediment in TELEMAC system. In this framework, the flocculation module will be a key part for modelling the cohesive sediment, and it is expected to work under the presence of the non-cohesive sediment.

4.3 The general structure for MCPBE model

The basis of the MCPBE flocculation models (M=2 or 3) consists of multiple “advection-diffusion-reaction” equations. The flocculation kinetics are captured by the source terms devised in the models. Depends on the defined classes, the MCPBE flocculation model could require different number of tracers and equations.

For example, in 2CPBE flocculation model, there are 3 equations describing the aggregation/breakage kinetics between 2 classes of cohesive particles, microflocs and macroflocs. Thus, in the model it needs 3 tracers (2 for the microflocs and macroflocs, and 1 for tracking the composition of macroflocs).

In the 3CPBE flocculation model, the modelled flocculation kinetics become more complex and it requires more equations to capture the dynamic interactions between all the 3 classes, namely microflocs, macroflocs

and megaflocs. There are two difference versions of 3CPBE model, one with 4 equations and the other one with 5 equations, to emphasize on different aspects of the flocculation process.

To accommodate both 2CPBE and 3CPBE models in the implementation framework, it is better to choose a general structure based on the most complex model, and make another one as a special case of it. In this case, the 3CPBE model with 5 equations is considered when designing a general structure for the implementation of MCPBE flocculation models. The 2CPBE and 3CPBE with 4 equations will be considered as simplified cases.

Table 1 – The overview of input parameters required for the MCPBE flocculation model

Symbol	**Variable name in TELEMAC**	**Meaning**	**Default values**	**Remarks**
NP	CNUM_P	number of microflocs in suspension per unit volume	N/A	Tracer 1
NF1	CNUM_F1	number of macroflocs in suspension per unit volume	N/A	Tracer 2
NT1	CNUM_T1	number of microflocs in macroflocs per unit volume	N/A	Tracer 3
NF2	CNUM_F2	number of megaflocs in suspension per unit volume	N/A	Tracer 4 (not used for fixed 3rd class)
NT2	CNUM_T2	number of microflocs in megaflocs per unit volume	N/A	Tracer 5
NC1	NC1	number of microflocs in one macrofloc	N/A	NT1/NF1; Impose a min value, e.g., 2.0.
NC2	NC2	number of microflocs in one megafloc	N/A	NT2/NF2; Impose a min value, e.g., 2*NC1.
nf	FRACDIM_MAC; FRACDIM_MEG	Fractal dimension of macroflocs and megaflocs	2.0	1-3; Suggest set as a constant for all flocs in the first version
DP	FLOCMIC_DIA	Size of microflocs	15 µm	~ 4-30µm
DF1	FLOCMAC_DIA	Size of macroflocs	N/A	DP*NC1^(1/nf); At t = 0, 100 µm (~ 50-200 µm)
DF2	FLOCMEG_DIA	Size of megaflocs	N/A for varying 3rd class; 360 µm for fixed 3rd class	DP*NC2^(1/nf); At t = 0 (for varying 3rd class), 360 µm (~200-400 µm)
fP1	BRKFRAC_P1	mass fraction of created microflocs when a macrofloc breaks up	0.1	0-1; Suggest set as the same value as that in two-class model

fP2	BRKFRAC_P2	mass fraction of created microflocs when a megafloc breaks up	0.1	0-1; Suggest set as the same value as that in two-class model
fF2	BRKFRAC_F2	mass fraction of remaining megaflocs when a megafloc breaks up	0.0	0-1; Suggest set as the same value as that in two-class model
K1	BRK_K1	number of created macroflocs when a parent macrofloc breaks up	2	>1; Set as a real instead of an integer
K2	BRK_K2	number of created macroflocs when a megafloc breaks up	2	>1; Set as a real instead of an integer
K3	BRK_K3	number of created megaflocs when a parent megafloc breaks up	0.0	Set as a real instead of an integer
rho_P	FLOCMIC_DEN	microfloc density	2500 kg/m^3	1600 - 2650 kg/m^3
alpha	AGG_ALPHA	collision efficiency	0.2	< 1 ; Fitting parameter
Eb	BRK_EFF	breakup fitting parameter	1.00E-04	in orders of 1.0E-6 - 1.0E-4; Fitting parameter
Fy	BRK_FY	Floc strength	1.00E-10	Set as a constant in the first version
q	BRK_Q	Parameter in the floc breakup frequency function	5.00E-01	Winterwerp (1998):0.5; Lee et al (2014): 1.0
qc	BRK_QC	Parameter in the floc breakup frequency function	5.00E-01	Proposed by Kyle Strom (2018)
KB	KBOLZ	Boltzmann constant(J/K)	1.38E-23	Only used in the Brownian motion term
T	TEMPSTD	Temperature(k)	283	Only used in the Brownian motion term
mP		Mass percentage of microflocs	N/A	NP/(NP+NT1+NT2); At t = 0, 10%
mF1		Mass percentage of macroflocs	N/A	NT1/(NP+NT1+NT2); At t = 0, 10%
mF2		Mass percentage of megaflocs	N/A	NT2/(NP+NT1+NT2); At t = 0, 80%
ws,p	WCHU_P	Settling velocity of microflocs	N/A	Stokes equation
ws,F1	WCHU_F1	Settling velocity of macroflocs	N/A	Winertwerp equation; WCHU_T1 = WCHU_F1
ws,F2	WCHU_F2	Settling velocity of megaflocs	N/A	Winterwerp equation; WCHU_T2= WCHU_F2

The most complicated case in the MCPBE mode has 5 equations, which means it requires 5 tracers to be defined at the beginning, 3 of them accounting for the microflocs, macroflocs and megaflocs, and the rest 2 track the composition of macroflocs and megaflocs as results of the interactions among the different classes.

A general structure of implementation does not mean it always uses 5 tracers in the flocculation model. The simulation is still initialized with 3 tracers in 2CPBE model, and 4 or 5 tracers in 3CPBE model. This is the way to ensure the efficiency of the model. It also means a consistent way of defining variables across different versions, and a unified way of preparing the sink/source terms. Moreover, it should also provide possibilities for future extension, e.g. including biological effects.

In terms of defining variables, all the possible variables in the most complicated case should be declared in the TELEMAC but only the necessary ones are initialized in simulations (Table 1).

For the future extension, it also comes down to the defining of variables. Some variables should be foreseen defined as multi-dimensional matrixes, although it is only a spatial uniform constant in at the moment. This will leave rooms for introducing biological effects later, under which the same variable could become temporal and spatial varying quantities.

5 Code development

The MCPBE flocculation module will be implemented as a new functionality in TELEMAC-3D and GAIA (the replacement of SISYPHE in the TELEMAC suite). The following code development is based on the stable version of TELEMAC (v8p2r1). In the development of the MCPBE flocculation models, new subroutines are created and added to the standard TELEMAC source code:

- computing flocculation kinetics, i.e.
  - FLOC_WCHU_2CPBE
  - FLOC_WCHU_3CPBE_VAR1
  - FLOC_WCHU_3CPBE_VAR2
- updating settling velocities, i.e.
  - FLOC_KINETICS_2CPBE
  - FLOC_KINETICS_3CPBE_VAR1
  - FLOC_KINETICS_3CPBE_VAR2
- and treating the bottom boundary conditions
  - FLOC_DEPOSITION_MCPBE.

Moreover, some existing subroutines are also modified to accommodate the new functionalities. The entire structure of the MCPBE model is illustrated in Figure 8. The details will be discussed separately in the following subsections.

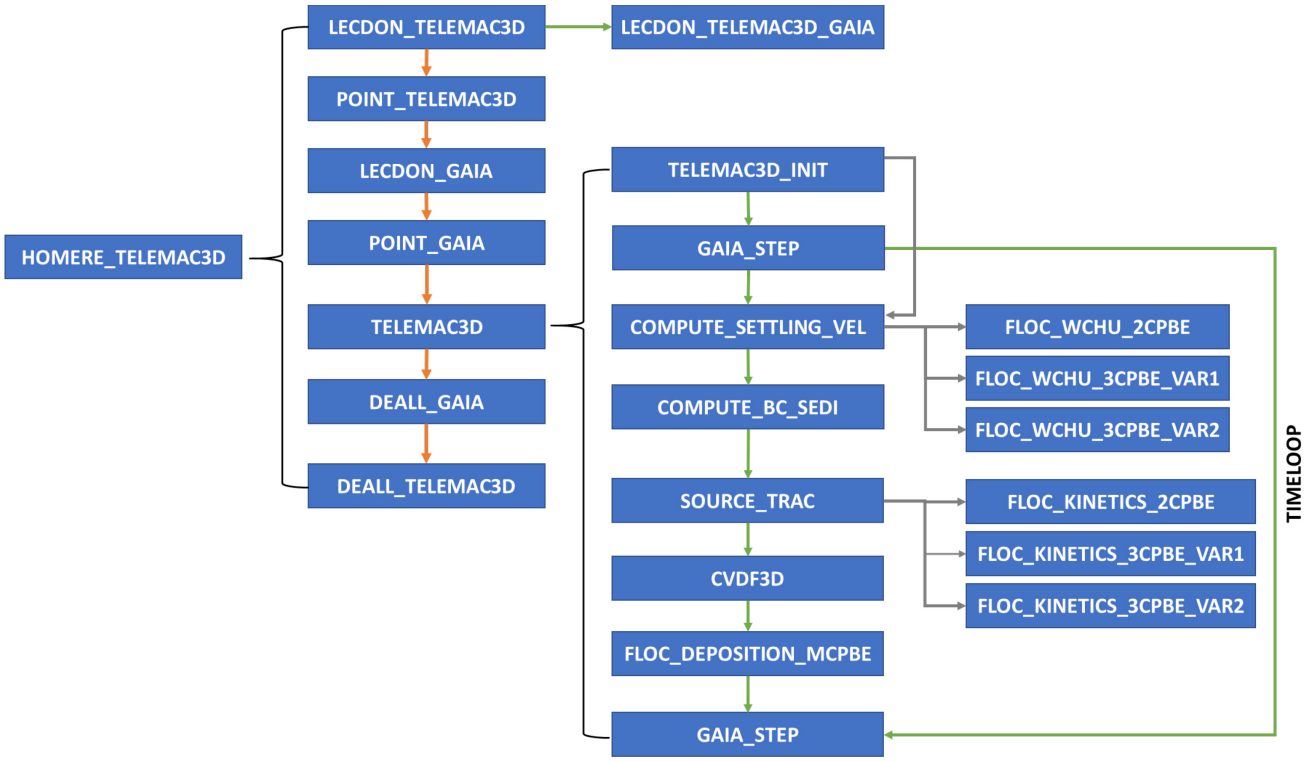

Figure 8 – The structure of the subroutines in the MCPBE flocculation model

5.1 Implementing new keywords

TELEMAC defines and reads the keywords in three steps as described in the example below.

Firstly, the definition of the new keywords should be added in **gaia.dico** (or other relevant **.dico** files, e.g. **telemac3d.dico**), and within the definition, the **INDEX** number and **MNEMO** (used as variable name in TELEMAC) should be given. Here is an example of adding a new keyword and read its value into the TELEMAC system.

```
NOM = 'FLOCULATION'
NOM1 = 'FLOCCULATION'
TYPE = LOGICAL
INDEX = 34
TAILLE = 1
DEFAUT = NON
DEFAUT1 = NO
MNEMO = 'FLOC'
RUBRIQUE = 'COHESIF';'VITESSE DE CHUTE';''
RUBRIQUE1 = 'COHESIVE';'SETTLING VELOCITY';''
NIVEAU = 1
AIDE = 'Decide si la formulation entravee doit etre utilisee pour calculer la vitesse de chute pour la vase.'
AIDE1 = 'Decides if hindered formulation is to be used to compute settling velocity for mud.'
```

Secondly, a logical variable (or other types of variable, depending on the purpose) associated with this keyword should be defined in **declarations_gaia.f** like:

```
!>  Include floculation effects
    LOGICAL :: FLOC
```

Finally, a line of code is added in **lecdon_gaia.f** to assign the value to the variable when it reads from the keyword:

```
FLOC = MOTLOG( ADRESS(3,34))
```

For different types of keywords, one can use different way of assigning values:

- Integer number keyword, MOTINT(ADRESS(1,INDEX));
- Real number keyword, MOTREA(ADRESS(2, INDEX));
- Logical keyword, MOTLOG(ADRESS(3, INDEX));
- Character keyword, MOTCAR(ADRESS(4, INDEX)).

5.1.1 Activating MCPBE flocculation models

In order to introduce a new functionality in TELEMAC system, new keywords related to the MCPBE flocculation model have to be defined in TELEMAC-3D and GAIA, on top of the existed code structures. This means the new keywords should be compatible with the existed keywords without creating conflicts with the other functionalities.

The main place for introducing the MCPBE flocculation functionality will be in GAIA. In the current version, there are already keywords related to the flocculation. In the **gaia.dico**, the keywords **TYPE OF SEDIMENT**, **FLOCCULATION** and **FLOCCULATION FORMULA** as listed in Table 2 are available for use if user need to activate the simpler flocculation model.

GAIA provides the possibility for modelling multiclass or mixed sediment transport. The type of sediment and the number of classes can be define in the keyword **TYPE OF SEDIMENT**. For example, if **TYPE OF SEDIMENT=CO;CO;CO;NCO;NCO**, it means the system will have three classes of cohesive sediment and two classes of non-cohesive sediment. In this case, if the keyword **FLOC = YES**, the flocculation model will only be applied to each cohesive sediment class, and **FLOCCULATION FORMULA** is used to select specific flocculation model (options can be seen in Table 2). Examples of setting up the MCPBE flocculation model (.cas files for GAIA) can be found in Appendix IX.

Table 2 – Existed and proposed changes in the keywords for selecting MCPBE flocculation model

Status	Keyword	Type	Variable name	Options
Existing	TYPE OF SEDIMENT	STRING	TYPE_SED	CO or NCO for each class
Existing	FLOCCULATION	LOGICAL	FLOC	NO or YES
Existing	FLOCCULATION FORMULA	INTEGER	FLOC_TYPE	1: Van Leussen; 2: Soulsby et al., 2013; 3: MCPBE flocculation model (this development).
Modified	FLOCCULATION FORMULA	INTEGER	FLOC_TYPE	1: Van Leussen; 2: Soulsby et al., 2013; 3: MCPBE flocculation model (KUL)
New	MCPBE VERSION	INTEGER	MCPBE_VER	1: 2-class model; 2: 3-class (fixed megaflocs); 3: 3-class (varying megaflocs).

For compatibility, an additional option is added in the keyword **FLOCCULATION FORMULA**, which is option 3: MCPBE flocculation model. For accommodating three different versions of MCPBE flocculation model, i.e. the 2-class 3-equation model, the 3-class 4-equation model and 3-class 5-equation model, another keyword should be added, which is called **MCPBE VERSION** (Table 2). User can choose the desired version using this newly defined keyword.

After selecting the MCPBE flocculation model (**MCPBE VERSION**), the values assigned to the keyword **TYPE OF SEDIMENT** will be checked in order to make sure the right number of cohesive sediment class are defined. This part will be programmed into subroutine **lecdon_telemac3d_gaia.f.** A rule should be followed: for using 2CPBE model, it is necessary to define 3 classes of cohesive sediment (**TYPE OF SEDIMENT = CO;CO;CO;…**), which are for microflocs, macroflocs and microflocs in macroflocs; for using 3CPBE model with fixed megaflocs, user has to define 4 classes of cohesive sediment (**TYPE OF SEDIMENT = CO;CO;CO;CO;…**), representing microflocs, macroflocs, microflocs in macroflocs and microflocs in megaflocs; for using 3CPBE model with varying megaflocs, user has to define 5 classes of cohesive sediment (**TYPE OF SEDIMENT = CO;CO;CO;CO;CO;…**), representing microflocs, macroflocs, microflocs in macroflocs, megaflocs and microflocs in megaflocs.

The implementation framework is designed for fitting into the existed structure of TELEMAC modules as much as possible. Because the number and type of sediment class is also used in the construction of the bed layer to determine bed composition and its mechanical properties (e.g. critical shear stress for erosion), it is crucial that the implementation does not break this consistency in the code. Additional efforts are made to differentiate the "real" cohesive sediment classes (microflocs, macroflocs and megaflocs) and the "auxiliary" sediment classes (microflocs in macroflocs, microflocs in megaflocs) used in the MCPBE flocculation models when dealing with bed layer. The details will be described in the next section.

At this stage, The implementation framework can allow the MCPBE flocculation model working with mixed sediment, in which the cohesive and non-cohesive sediments co-exist. In such case, the number of cohesive sediment class will depend on the choice of the MCPBE flocculation model, and user could also define the number of non-cohesive sediment class in addition to that. Therefore, the suspended sediment transport and bedload transport can be solved simultaneously and this is a more realistic situation in marine and estuarine environment.

5.1.2 Physical parameters required in MCPBE flocculation model

The MCPBE flocculation models require various physical parameters as input. They can be given through the additional keywords (Table 3). The names and default values of these new keywords are described in Table 3.

Table 3 – The new keywords proposed for the physical parameters required in MCPBE flocculation model

Keyword	Type	Variable name	Default value
INI. NUMBER OF MICROFLOCS BOUNDED IN MACROFLOCS	REAL	NC1_INI	20
INI. NUMBER OF MICROFLOCS BOUNDED IN MEGAFLOCS	REAL	NC2_INI	100
MAX NUMBER OF MICROFLOCS BOUNDED IN MACROFLOCS	REAL	NC1_MAX	1000
MAX NUMBER OF MICROFLOCS BOUNDED IN MEGAFLOCS	REAL	NC2_MAX	1000
FRACTAL DIMENSION OF MACROFLOCS	REAL	FRACDIM_MAC	2.0
FRACTAL DIMENSION OF MEGAFLOCS	REAL	FRACDIM_MEG	2.0
SIZE OF MICROFLOCS	REAL	FLOCMIC_DIAFIX	15 µm
SIZE OF MEGAFLOCS	REAL	FLOCMEG_DIAFIX	360 µm (only used for fixed 3rd class in the 3-class 4-equation model)
MICROFLOC DENSITY	REAL	FLOCMIC_DEN	2500 kg/m^3
FRACTION OF CREATED MICROFLOCS BY MACROFLOC BREAKUP	REAL	BRKFRAC_P1	0.1 (mass fraction)
FRACTION OF CREATED MICROFLOCS BY MEGAFLOC BREAKUP	REAL	BRKFRAC_P2	0.1 (mass fraction)
FRACTION OF REMAINING MEGAFLOCS DURING MEGAFLOC BREAKUP	REAL	BRKFRAC_F2	0.0 (mass fraction)

NUMBER OF CREATED MACROFLOCS BY MACROFLOC BREAKUP	REAL	BRK_K1	2.0
NUMBER OF CREATED MACROFLOCS BY MEGAFLOC BREAKUP	REAL	BRK_K2	2.0
NUMBER OF CREATED MEGAFLOCS BY MEGAFLOC BREAKUP	REAL	BRK_K3	0.0
COLLISION EFFICIENCY	REAL	AGG_ALPHA	0.2
BREAKUP EFFICIENCY	REAL	BRK_EFF	1.00E-04
FLOC YIELD STRENGTH	REAL	BRK_FY	1.00E-10
FLOC BREAKUP FREQUENCY	REAL	BRK_Q	5.00E-01
FLOC BREAKUP FREQUENCY COEFFICIENT	REAL	BRK_QC	5.00E-01
BOLTZMANN CONSTANT	REAL	KBOLZ	1.38E-23
TEMPERATURE USED IN MCPBE MODEL	REAL	TEMPSTD	283.0

The definitions of the above keywords are described in the Appendix I, which can be added directly into the relevant files in TELEMAC, i.e. **gaia.dico**.

5.1.3 Variables associated with the new keywords

Because of the added new keywords, the corresponding variables should be defined in order to store the values or options given to these keywords from the steering file. For this purpose, the following code should be added in **declarations_gaia.f**.

As integer:

```
!>   MCPBE Floculation model:
!!   1: 2-class 3-equation model; 2: 3-class 4-equation model; 3: 3-class 5-equation model
     INTEGER MCPBE_VER
!
!>   PRINT POINT INFO for MCPBE Floculation model:
!!   print info at given point, no printout if 0
     INTEGER DBPOIN_MCPBE
```

As real:

```
!
!>   MCPBE, Fractal dimension of macroflocs and megaflocs
!
     DOUBLE PRECISION :: FRACDIM_MAC, FRACDIM_MEG
!
!>   Size of microflocs and megaflocs
!
     DOUBLE PRECISION :: FLOCMIC_DIAFIX, FLOCMEG_DIAFIX
!
!>   Density of microflocs
!
```

```
    DOUBLE PRECISION :: FLOCMIC_DEN
!
!>   Fraction of created microflocs by macrofloc breakup, and by megafloc breakup
!
    DOUBLE PRECISION :: BRKFRAC_P1, BRKFRAC_P2
!
!>   Fraction of remaining megaflocs during a larger megafloc breakup
!
    DOUBLE PRECISION :: BRKFRAC_F2
!
!>   Number of created macroflocs by macrofloc breakup, and by megafloc breakup
!
    DOUBLE PRECISION :: BRK_K1, BRK_K2
!
!>   Number of created megaflocs by a larger megafloc breakup
!
    DOUBLE PRECISION :: BRK_K3
!
!>   Collision efficiency, breakup efficiency
!
    DOUBLE PRECISION :: AGG_ALPHA, BRK_EFF
!
!>   Floc yield strength
!
    DOUBLE PRECISION :: BRK_FY
!
!>   Floc breakup frequency coefficient
!
    DOUBLE PRECISION :: BRK_Q, BRK_QC
!
!>   Boltzmann constant
!
    DOUBLE PRECISION :: KBOLZ
!
!>   Temperature used in MCPBE model
!
    DOUBLE PRECISION :: TEMPSTD
!
!>   initial number conc. of microflocs bounded in macroflocs and megaflocs
!
    DOUBLE PRECISION :: NC1_INI,NC2_INI
!
!>   maximum number microflocs bounded in macroflocs and megaflocs
!
    DOUBLE PRECISION :: NC1_MAX,NC2_MAX
```

5.1.4 Reading new keywords

At the time of writing this report, the GAIA module is not fully completed yet. The keyword FLOC_TYPE is not read correctly into GAIA. Therefore, it is included in the code modifications. The following block should be added in **lecdon_gaia.f** for reading the new keywords from the steering file of GAIA**.** The meaning of these keywords and variables can be found in Table 1.

```
! For the MCPBE flocculation model
    HIND_TYPE =  MOTINT(ADRESS(1, 51))
    FLOC_TYPE =  MOTINT(ADRESS(1, 61))
    MCPBE_VER = MOTINT(ADRESS(1, 71))
    DBPOIN_MCPBE = MOTINT(ADRESS(1, 72))
    FRACDIM_MAC = MOTREA(ADRESS(2, 61))
    FRACDIM_MEG = MOTREA(ADRESS(2, 62))
    FLOCMIC_DIAFIX = MOTREA(ADRESS(2, 63))
    FLOCMEG_DIAFIX = MOTREA(ADRESS(2, 64))
    FLOCMIC_DEN = MOTREA(ADRESS(2, 65))
    BRKFRAC_P1 = MOTREA(ADRESS(2, 66))
    BRKFRAC_P2 = MOTREA(ADRESS(2, 67))
    BRKFRAC_F2 = MOTREA(ADRESS(2, 68))
    BRK_K1 = MOTREA(ADRESS(2, 69))
    BRK_K2 = MOTREA(ADRESS(2, 70))
    BRK_K3 = MOTREA(ADRESS(2, 71))
    AGG_ALPHA = MOTREA(ADRESS(2, 72))
    BRK_EFF = MOTREA(ADRESS(2, 73))
    BRK_FY = MOTREA(ADRESS(2, 74))
    BRK_Q = MOTREA(ADRESS(2, 75))
    BRK_QC = MOTREA(ADRESS(2, 82))
    KBOLZ = MOTREA(ADRESS(2, 76))
    TEMPSTD = MOTREA(ADRESS(2, 77))
    NC1_INI = MOTREA(ADRESS(2, 78))
    NC2_INI = MOTREA(ADRESS(2, 79))
    NC1_MAX = MOTREA(ADRESS(2, 80))
    NC2_MAX = MOTREA(ADRESS(2, 81))
```

5.2 Defining new global variables

Beside of the new keywords and their corresponding variables, the following variables should be defined globally in GAIA (**declarations_gaia.f**) and TELEMAC-3D (**declarations_telemac3d.f**) for being accessed by the other subroutines or TELEMAC modules.

There are five variables defined in GAIA (**declarations_gaia.f**) for identifying and indexing the tracers used in the MCPBE model throughout the computation.

```
!>   Index used in MCPBE flocculation model
     INTEGER IMICFLC,IMACFLC,IMEGFLC,IMICF_MACF,IMICF_MEGF
```

GAIA only has the declaration of BIEF_OBJ with the type of MESH2D. This means all the variables defined in GAIA only have 2D dimensions. Therefore, the global variables with 3D dimensions are added in TELEMAC-3D (**declarations_telemac3d.f**).

```
!>   Size of microflocs in case of 3CPBE with biological effects
!
     TYPE(BIEF_OBJ), TARGET :: FLOCMIC_DIA
!
!>   Size and density of macroflocs
!
     TYPE(BIEF_OBJ), TARGET :: FLOCMAC_DIA, FLOCMAC_DEN
!
!>   Size and density of megaflocs
!
     TYPE(BIEF_OBJ), TARGET :: FLOCMEG_DIA, FLOCMEG_DEN
!
!    Number of microflocs inside macroflocs and megaflocs
!
     TYPE(BIEF_OBJ), TARGET :: NC_MAC, NC_MEG
```

The above BIEF_OBJs have to be initialized in the subroutine **point_telemac3d.f** in order to give them proper dimensions. The modifications consist of two parts. The first part is for reading information from GAIA in the beginning of this subroutine:

```
     USE DECLARATIONS_GAIA, ONLY: FLOC,FLOC_TYPE,MCPBE_VER,
    &                     FLOCMIC_DIAFIX FLOCMEG_DIAFIX,
    &                     NC1_INI,NC2_INI,FRACDIM_MEG
```

The second part is for memory allocation:

```
!    FOR MCPBE FLOCCULATION MODEL
!>   Allocate variables for MCPBE flocculation model
     WRITE(LU,*) 'CHECK FLOCCULATION MODEL',FLOC,FLOC_TYPE
     IF (FLOC .AND. FLOC_TYPE.EQ.3) THEN
       WRITE(LU,*) 'ADD EXTRA VARIABLES FOR MCPBE FLOCCULATION MODEL'
       CALL BIEF_ALLVEC(1, FLOCMAC_DIA,'FMADIA',IELM3,1,1,MESH3D)
       CALL BIEF_ALLVEC(1, FLOCMAC_DEN,'FMADEN',IELM3,1,1,MESH3D)
       CALL BIEF_ALLVEC(1, NC_MAC,     'NC_MAC',IELM3,1,1,MESH3D)
```

```
      CALL OS('X=C     ', X=NC_MAC,C=NC1_INI)
      IF (MCPBE_VER .GT. 1) THEN
        CALL BIEF_ALLVEC(1, FLOCMIC_DIA,'FMIDIA',IELM3,1,1,MESH3D)
        CALL BIEF_ALLVEC(1, FLOCMEG_DIA,'FMEDIA',IELM3,1,1,MESH3D)
        CALL BIEF_ALLVEC(1, FLOCMEG_DEN,'FMEDEN',IELM3,1,1,MESH3D)
        CALL BIEF_ALLVEC(1, NC_MEG,     'NC_MEG',IELM3,1,1,MESH3D)
        IF (MCPBE_VER.EQ.2) THEN
          NC2_INI = (FLOCMEG_DIAFIX/FLOCMIC_DIAFIX)**FRACDIM_MEG
          WRITE(LU,*) 'NC2_INI FROM THE INITIAL FLOC SIZES: ',NC2_INI
        ENDIF
        CALL OS('X=C     ', X=FLOCMIC_DIA,C=FLOCMIC_DIAFIX)
        CALL OS('X=C     ', X=NC_MEG,C=NC2_INI)
      ELSE
        CALL BIEF_ALLVEC(1, FLOCMIC_DIA,'FMIDIA',0,1,0,MESH3D)
        CALL BIEF_ALLVEC(1, FLOCMEG_DIA,'FMEDIA',0,1,0,MESH3D)
        CALL BIEF_ALLVEC(1, FLOCMEG_DEN,'FMEDEN',0,1,0,MESH3D)
        CALL BIEF_ALLVEC(1, NC_MEG,     'NC_MEG',0,1,0,MESH3D)
      ENDIF
    ELSE
      CALL BIEF_ALLVEC(1, FLOCMIC_DIA,'FMIDIA',0,1,0,MESH3D)
      CALL BIEF_ALLVEC(1, FLOCMAC_DIA,'FMADIA',0,1,0,MESH3D)
      CALL BIEF_ALLVEC(1, FLOCMAC_DEN,'FMADEN',0,1,0,MESH3D)
      CALL BIEF_ALLVEC(1, FLOCMEG_DIA,'FMEDIA',0,1,0,MESH3D)
      CALL BIEF_ALLVEC(1, FLOCMEG_DEN,'FMEDEN',0,1,0,MESH3D)
      CALL BIEF_ALLVEC(1, NC_MAC,     'NC_MAC',0,1,0,MESH3D)
      CALL BIEF_ALLVEC(1, NC_MEG,     'NC_MEG',0,1,0,MESH3D)
      WRITE(LU,*) 'NO INITIALIZATION FOR MCPBE FLOCCULATION MODEL'
    ENDIF
```

The reason that we could use the variables FLOC, FLOC_TYPE and MCPBE_VER is that the subroutine **lecdon_telemac3d_gaia.f** is called inside **lecdon_telemac3d.f**, which is called before the subroutine **point_telemac3d.f** in the higher level main program **homere_telemac3d.f**. And these variables associated with the new keywords are read in **lecdon_telemac3d_gaia.f** with following modifications.

```
!   FOR MCPBE FLOCCULATION MODEL
    FLOC        = MOTLOG( ADRESS(3,34))
    FLOC_TYPE   = MOTINT(ADRESS(1, 61))
    MCPBE_VER   = MOTINT(ADRESS(1, 71))
    NC1_INI     = MOTREA(ADRESS(2, 78))
    NC2_INI     = MOTREA(ADRESS(2, 79))
    FRACDIM_MEG   = MOTREA(ADRESS(2, 62))
    FLOCMIC_DIAFIX = MOTREA(ADRESS(2, 63))
    FLOCMEG_DIAFIX = MOTREA(ADRESS(2, 64))
    WRITE(LU,*) 'INI. MICROFLOC SIZE',FLOCMIC_DIAFIX
    WRITE(LU,*) 'INI. MEGAFLOC SIZE',FLOCMEG_DIAFIX
    WRITE(LU,*) 'FRACDIM_MEG GIVEN BY USER',FRACDIM_MEG
```

5.3 Initialization

The initialization phase consists of the following procedures:

- identifying corresponding cohesive sediment class for microflocs, macroflocs and megaflocs (in case of 3CPBE model);
- identifying auxiliary classes for tracking the composition of macroflocs and megaflocs;
- and initializing concentrations and settling velocity of each sediment class.

The initialization of the MCPBE flocculation model is discussed in the following sections. For the general structure of the code, reader may refer to Figure 8.

5.3.1 Identifying sediment classes for MCPBE model

In the MCPBE flocculation models, auxiliary tracers or classes are always needed for accounting for the interactions between the real cohesive sediment classes and tracking the composition of the flocs according to the flocculation kinetics. The real sediment classes, i.e. microflocs, macroflocs and megaflocs, always have their own properties, e.g. size, density, settling velocity, which could affect the transport process. They are also presented in the bed composition as a fraction of bed material, and can be deposited to and eroded from the bed, according to a general framework implemented in GAIA for dealing with multiclass sediment transport.

On the other hand, the auxiliary classes (i.e. microflocs bounded in macroflocs, microflocs bounded in megaflocs) can be seen as quantities or indexes connecting the real classes and evolving due to their interactions. Together they form the complete set of equations describing the flocculation kinetics. But the auxiliary classes used in MCPBE models are not the same as the real sediment classes. Their properties are always related to the correspond real classes and they cannot directly appear in the bed or be eroded from the bottom. So different treatments have to be applied to these auxiliary classes, which requires that each class can be correctly identified and indexed throughout the computation.

The advantage of doing this is that it will not make the MCPBE model interfere with the bed model, which could avoid overcomplication and unnecessary code modifications.

To identify the different cohesive sediment classes used in the MCPBE flocculation model, five indexing variables are defined in **declaration_gaia.f** and used in **lecdon_telemac3d_gaia.f**. Besides, necessary GAIA keywords are also read from steering file in **lecdon_telemac3d_gaia.f** for later use. The modifications in **lecdon_telemac3d_gaia.f** are shown below.

```
!    FOR MCPBE FLOCCULATION MODEL
     FLOC        = MOTLOG( ADRESS(3,34))
     FLOC_TYPE    = MOTINT(ADRESS(1, 61))
     MCPBE_VER    = MOTINT(ADRESS(1, 71))
     NC1_INI      = MOTREA(ADRESS(2, 78))
     NC2_INI      = MOTREA(ADRESS(2, 79))
     FRACDIM_MAC    = MOTREA(ADRESS(2, 61))
     FRACDIM_MEG    = MOTREA(ADRESS(2, 62))
     FLOCMIC_DIAFIX = MOTREA(ADRESS(2, 63))
     FLOCMEG_DIAFIX = MOTREA(ADRESS(2, 64))
     WRITE(LU,*) 'INI. MICROFLOC SIZE',FLOCMIC_DIAFIX
     WRITE(LU,*) 'INI. MEGAFLOC SIZE',FLOCMEG_DIAFIX
     WRITE(LU,*) 'FRACDIM_MAC GIVEN BY USER',FRACDIM_MAC
     WRITE(LU,*) 'FRACDIM_MEG GIVEN BY USER',FRACDIM_MEG
```

```
!    LABELS FOR FLOC CLASSES FOR MCPBE MODEL
     IMICFLC = 0         ! Microflocs
     IMACFLC = 0         ! Macroflocs
     IMEGFLC = 0         ! Megaflocs
     IMICF_MACF = 0      ! Microflocs in macroflocs
     IMICF_MEGF = 0      ! Microflocs in megaflocs
```

After the above preparation, the rest part in **lecdon_telemac3d_gaia.f** searches for the cohesive sediment classes in all the defined tracers based on their tracer names. The cohesive sediment class always has the name 'COH SEDIMENT? ' or 'SEDIMENT COH? ' in TELEMAC-3D and GAIA (? can be 1, 2, … ,5, for the 1st, 2nd … 5th cohesive sediment class). Here we assume the order of the cohesive sediment class according to Table 4:

Table 4 – Name of cohesive sediment classes and their corresponding indexing variables

Name of tracer	Index variable in 2CPBE[1]	Index variable in 3CPBE with fixed megaflocs*	Index variable in 3CPBE with varying megaflocs*
COH SEDIMENT1	IMICFLC	IMICFLC	IMICFLC
COH SEDIMENT2	IMACFLC	IMACFLC	IMACFLC
COH SEDIMENT3	IMICF_MACF	IMICF_MACF	IMICF_MACF
COH SEDIMENT4	N/A	IMICF_MEGF	IMEGFLC
COH SEDIMENT5	N/A	N/A	IMICF_MEGF

[1] IMICFLC = microflocs, IMACFLC = macroflocs, IMICF_MACF = microflocs in macroflocs, IMEGFLC = megaflocs, IMICF_MEGF = microflocs in megaflocs.

Based on this convention, the index of each floc class will be assigned to **IMICFLC**, **IMACFLC**, **IMEGFLC**, **IMICF_MACF** and **IMICF_MEGF** during initialization. The names of the cohesive sediment classes are changed to the names of corresponding floc classes in the following code in **lecdon_telemac3d_gaia.f**.

```
!    FOR MCPBE MODEL
     IF(FLOC .AND. FLOC_TYPE.EQ.3) THEN
!      LABEL EACH FLOC CLASS FOR MCPBE MODEL
       DO I=1,NTRAC
         WRITE(LU,*) 'MCPBE VERSION IS ',MCPBE_VER
         WRITE(LU,*) 'TRACER',I,'IS ',NAMETRAC(I)
!        2CPBE model only needs 3 tracers
         IF(NAMETRAC(I)(1:16).EQ.'COH SEDIMENT1   ' .OR.
    &       NAMETRAC(I)(1:16).EQ.'SEDIMENT COH1   ') THEN
           IMICFLC = I
           NAMETRAC(I)(1:16) = 'MICROFLOCS      '
         ENDIF
         IF(NAMETRAC(I)(1:16).EQ.'COH SEDIMENT2   ' .OR.
    &       NAMETRAC(I)(1:16).EQ.'SEDIMENT COH2   ') THEN
           IMACFLC = I
           NAMETRAC(I)(1:16) = 'MACROFLOCS      '
         ENDIF
         IF(NAMETRAC(I)(1:16).EQ.'COH SEDIMENT3   ' .OR.
    &       NAMETRAC(I)(1:16).EQ.'SEDIMENT COH3   ') THEN
           IMICF_MACF = I
           NAMETRAC(I)(1:16) = 'MICFLC IN MACFLC'
         ENDIF
!        3CPBE with fixed megaflocs model needs 4 tracers
         IF(MCPBE_VER.EQ.2) THEN
           IF(NAMETRAC(I)(1:16).EQ.'COH SEDIMENT4   ' .OR.
    &         NAMETRAC(I)(1:16).EQ.'SEDIMENT COH4   ') THEN
             IMICF_MEGF = I
             NAMETRAC(I)(1:16) = 'MICFLC IN MEGFLC'
           ENDIF
!        3CPBE with varying megaflocs model needs 5 tracers
         ELSEIF(MCPBE_VER.EQ.3) THEN
           IF(NAMETRAC(I)(1:16).EQ.'COH SEDIMENT4   ' .OR.
    &         NAMETRAC(I)(1:16).EQ.'SEDIMENT COH4   ') THEN
             IMEGFLC = I
             NAMETRAC(I)(1:16) = 'MEGAFLOCS       '
           ENDIF
           IF(NAMETRAC(I)(1:16).EQ.'COH SEDIMENT5   ' .OR.
    &         NAMETRAC(I)(1:16).EQ.'SEDIMENT COH5   ') THEN
             IMICF_MEGF = I
             NAMETRAC(I)(1:16) = 'MICFLC IN MEGFLC'
           ENDIF
         ENDIF
         WRITE(LU,*) 'TRACER',I,'IS CHANGED TO ',NAMETRAC(I)
       ENDDO
       WRITE(LU,*) 'FLOCS FOUND IN TRACER',IMICFLC,IMACFLC,IMEGFLC,
    &                              IMICF_MACF,IMICF_MEGF
     ENDIF
```

5.3.2 Configure the solvers for MCPBE flocculation model

After initialization of the sediment classes (tracers) for the MCPBE flocculation model, the solvers and corresponding advection/diffusion schemes have to be configured for each class in the subroutine **lecdon_telemac3d_gaia.f**. Note that the code firstly initializes the solvers for the sediment classes with the default values predefined in the corresponding keywords, or uses the first value given to each keywords. Then if multiple values are given to those keywords, they will be read and assigned to the corresponding variables. This means for each of the following keywords:

- SCHEME FOR ADVECTION OF SUSPENDED SEDIMENTS
- SCHEME OPTION FOR ADVECTION OF SUSPENDED SEDIMENTS
- SOLVER FOR DIFFUSION OF SUSPENSION
- INITIAL SUSPENDED SEDIMENTS CONCENTRATION VALUES
- PRESCRIBED SUSPENDED SEDIMENTS CONCENTRATION VALUES
- SUSPENDED SEDIMENTS CONCENTRATION VALUES AT THE SOURCES
- VERTICAL PROFILES OF SUSPENDED SEDIMENTS
- COEFFICIENT FOR HORIZONTAL DIFFUSION OF SUSPENDED SEDIMENTS
- COEFFICIENT FOR VERTICAL DIFFUSION OF SUSPENDED SEDIMENTS

One has to assign 3 values to each of the above keywords in case of MCPBE_VER = 1, and 4 or 5 values in case of MCPBE_VER = 2 or 3 correspondingly. This also means user has to define the initial and boundary concentrations for all the “real” and “auxiliary” sediment classes. If there are point sources, one has to give the values for all the “real” and “auxiliary” sediment classes as well.

After the configurations being done in the subroutine **lecdon_telemac3d_gaia.f**, the same information of the solvers for both “real” and “auxiliary” sediment classes has to be passed to the corresponding variables in the subroutine **lecdon_telemac3d.f** in TELEMAC-3D, where all the tracers, including suspended sediments, are solved.

5.3.3 Initializing concentrations for sediment class

As briefly mentioned above, the initialization of sediment concentration is partially done in the subroutine **lecdon_telemac3d_gaia.f** in GAIA by reading the keyword **INITIAL SUSPENDED SEDIMENTS CONCENTRATION VALUES**. The values assigned to this keyword is then passed to TELEMAC-3D in **lecdon_telemac3d.f**. However, in order to compute/correct the initial concentrations for the auxiliary sediment classes later in **lecdon_telemac3d.f**, the related keywords (representing the necessary physical parameters during the initialization stage) have been read from steering file by the additional code in **lecdon_telemac3d_gaia.f** (see §5.3.1) before using them in **lecdon_telemac3d.f**.

Based on the initial concentrations of the “real” sediment class given in the GAIA keyword, the initial values for the auxiliary class used in MCPBE flocculation model is computed by the following code added in **lecdon_telemac3d.f**. This can be used as a check for the initial concentrations defined in the keyword **INITIAL SUSPENDED SEDIMENTS CONCENTRATION VALUES** in GAIA.

```
!     COMPUTED CONCENTRATION FOR MICFLOCS IN MACFLOCS AND MICFLOCS IN MEGAFLOCS
      IF(FLOC .AND. FLOC_TYPE.EQ.3) THEN
        TRAC0(IMICF_MACF) = NC1_INI*TRAC0(IMACFLC)
!       FOR MCPBE.EQ.2, THE INIT. CONC. OF IMICF_MEGF IS GIVEN IN KEYWORDS
        IF(MCPBE_VER.EQ.3) THEN
          TRAC0(IMICF_MACF) = NC1_INI*TRAC0(IMACFLC)
          TRAC0(IMICF_MEGF) = NC2_INI*TRAC0(IMEGFLC)
```

```
      ENDIF
      DO ITRAC=IND_SED,IND_SED+LOCAL_NSUSP_TEL-1
       WRITE(LU,*) 'INITIAL CONC. OF ',
     &            NAMETRAC(ITRAC)(1:16),'=',TRAC0(ITRAC)
      ENDDO

     ENDIF
```

5.3.4 Initializing settling velocity for sediment class

The settling velocity is calculated in TELEMAC-3D although the initial values are read from GAIA for each class of sediment.

The settling velocity is calculated in the subroutine **compute_settling_vel.f** in TELEMAC-3D, before the computation of the source terms for the MCPBE flocculation model. This subroutine is called during the initialization stage in **telemac3d_init.f** and also at each time iteration in **telemac3d.f**. By default, the settling velocity (variable **WCHU** in TELEMAC-3D) is declared for all the tracers, and accepts values given in the GAIA keyword **CLASSES SETTLING VELOCITIES** for the “real” and “auxiliary” sediment classes. But it cannot recognize the options of using the MCPBE flocculation model in **compute_settling_vel.f**. Therefore, an additional case is added in this subroutine after line 131.

```
      ELSEIF(FLOC_TYPE.EQ.3) THEN
       IF(LT.EQ.0) THEN
        WRITE(LU,*) 'INIT. SETTLING VELOCITY OF ',
     &       NAMETRAC(ITRAC)(1:16),XWC0(NUM_ISUSP_ICLA(ISUSP))
       ELSE
        CONTINUE
       ENDIF
```

Note that the above code only prints out the initial settling velocities. The following code is appended to the subroutine **compute_settling_vel.f** for initializing and updating the settling velocities at each time step. The details will be explained in the next section.

```
! 3. SETTLING VELOCITY FOR MCPBE FLOCCULATION MODEL
!   Compute settling velocity at each time step
    IF(FLOC .AND. FLOC_TYPE.EQ.3) THEN
     IF(MCPBE_VER.EQ.1) CALL FLOC_WCHU_2CPBE
     IF(MCPBE_VER.EQ.2) CALL FLOC_WCHU_3CPBE_VAR1
     IF(MCPBE_VER.EQ.3) CALL FLOC_WCHU_3CPBE_VAR2
     WRITE(LU,*) 'UPDATE SETTLING VELOCITY IN THE MCPBE MODEL'
     DO ITRAC=IND_SED,IND_SED+NSUSP_TEL-1
      WRITE(LU,*) 'MAX. AND MIN. WS OF ',NAMETRAC(ITRAC)(1:16),':',
    &      MAXVAL(WCHU%ADR(ITRAC)%P%R),MINVAL(WCHU%ADR(ITRAC)%P%R)
     ENDDO
    ENDIF
```

5.4 Solving flocculation kinetics

The flocculation kinetics is modelled with MCPBEs. In general, there are 4 major steps for solving MCPBE:

- computing settling velocity and floc characteristics,
- computing source terms or reaction terms,
- computing the erosion/deposition flux as bottom boundary conditions,
- and assembling those terms in the MCPBEs and solve them numerically.

This section describes the necessary code development for each major step.

5.4.1 Computing settling velocity and floc characteristics

In the MCPBE flocculation model, the settling velocity and flocs properties are subjected to the changes of flow filed, turbulent shear and concentrations as described in eq. (13). Hence, it is a necessary step to compute these quantities at each time step and update their values before solving the equations for the flocculation kinetics. As mentioned earlier in §5.3.4, the subroutine **compute_settling_vel.f** is called at each time step for updating the settling velocity.

TELEMAC-3D already has two simplified flocculation model implemented. They don't really model the flocculation kinetics, instead, they only update the settling velocity based on the empirical or semi-empirical formulas in the subroutine **compute_settling_vel.f**. In order to have a unified framework for all the flocculation models, including the MCPBE flocculation model, the subroutine **compute_settling_vel.f** has been adapted.

Here three additional subroutines are created, **floc_wchu_2cpbe.f**, **floc_wchu_3cpbe_var1.f** and **floc_wchu_3cpbe_var2.f**, for the 2CPBE and 3CPBE flocculation models, respectively. These new subroutines are called within **compute_settling_vel.f**. The content of these three subroutines can be found in Appendix II , Appendix III and Appendix IV.

5.4.1.1 Computing sink and source terms (flocculation kinetics)

In TELEMAC-3D, the sink and source terms for the advection-diffusion equations are solved during the diffusion step in the subroutine **cvdf3d.f**. The sink and source terms are prepared in the subroutine **source_trac.f** before passing to **cvdf3d.f**. Therefore, as an important part of the MCPBE flocculation model, the computation of the sink and source terms are appended in the end of **source_trac.f**.

```
!**********************************************************************
!    FOR COUPLING WITH GAIA, WHEN MCPBE FLOCCULATION MODEL IS ENABLED
!**********************************************************************
!
    IF(INCLUS(COUPLING,'GAIA'))THEN
      IF(FLOC .AND. FLOC_TYPE.EQ.3) THEN
        IF(MCPBE_VER.EQ.1) THEN
          IF(DEBUG.GT.0) WRITE(LU,*) 'CALL OF FLOC_KINETICS_2CPBE'
          CALL FLOC_KINETICS_2CPBE
          IF(DEBUG.GT.0) WRITE(LU,*) 'BACK FROM FLOC_KINETICS_2CPBE'
        ENDIF
        IF(MCPBE_VER.EQ.2) THEN
          IF(DEBUG.GT.0) WRITE(LU,*) 'CALL OF FLOC_KINETICS_3CPBE'
          CALL FLOC_KINETICS_3CPBE_VAR1
```

```
        IF(DEBUG.GT.0) WRITE(LU,*) 'BACK FROM FLOC_KINETICS_3CPBE'
       ENDIF
       IF(MCPBE_VER.EQ.3) THEN
        IF(DEBUG.GT.0) WRITE(LU,*) 'CALL OF FLOC_KINETICS_3CPBE'
        CALL FLOC_KINETICS_3CPBE_VAR2
        IF(DEBUG.GT.0) WRITE(LU,*) 'BACK FROM FLOC_KINETICS_3CPBE'
       ENDIF
      ENDIF
     ENDIF
!
```

Here we created three additional subroutines, **floc_kinetics_2cpbe.f**, **floc_kinetics_3cpbe_var1.f** and **floc_kinetics_3cpbe_var2.f**, for the different versions of 2CPBE and 3CPBE flocculation models, respectively. The content of these subroutines can be found in Appendix V, Appendix VI and Appendix VII.

In an early implementation, the sink and source terms in the MCPBE models were treated explicitly. This leads to a restriction on the time step and cannot ensure the positivity of concentrations. According to Breugem (2021) (personal communication), linearization techniques, namely the Picard method and Patankar method (1980), can be used in the treatment of sink and source terms to improve the stability and ensure the non-negative concentrations. The Picard method (a single Picard iteration) corresponds to the idea of approximating a nonlinear term at time step *n* like $(c^n)^2$ by $c^n c^{n-1}$. The Patankar method is used to deal with the sink term that does not involve the unknown.

Considering the following form as one of the equations in a system of PDEs:

$$\frac{\partial c_p}{\partial t} = -ac_p c_f - bc_p c_p + S + ADV + DIFF \tag{12}$$

where c_p and c_f are the non-negative unknowns (e.g. concentrations of flocs), a and b are the positive coefficients, hence $\left(-ac_p c_f - bc_p c_p\right)$ is considered as the sink term, S is the source term, ADV and $DIFF$ are the advection and diffusion terms. Since TELEMAC-3D uses fractional step method for solving advection-diffusion equations, the ADV and $DIFF$ terms will be taken care by different schemes, so here we drop these two terms.

The explicit sink and source terms in eq. (12) will give:

$$\frac{c_p^{n+1} - c_p^n}{\Delta T} = -ac_p^n c_f^n - b\, c_p^n c_p^n + S \tag{13}$$

$$c_p^{n+1} = c_p^n \left(1 + \Delta T \left(-ac_f^n - bc_p^n + \frac{S}{c_p^n}\right)\right) \tag{14}$$

As seen above, eq. (14) imposes a constraint on ΔT. If ΔT exceeds the required range, negative concentrations may appear in the calculation and lead to numerical instabilities in the model.

Alternatively, applying the Picard method to eq. (12) gives:

$$c_p^{n+1} = c_p^n + \Delta T \left(-ac_p^{n+1} c_f^n - b\, c_p^{n+1} c_p^n + S\right) \tag{15}$$

$$c_p^{n+1} = \frac{c_p^n + \Delta T \cdot S}{\left(1 + \Delta T\left(ac_f^n + bc_p^n\right)\right)} \tag{16}$$

Eq. (16) ensures positivity without constraint on ΔT. This also corresponds to the way of treating sink and source terms in TELEMAC-3D. The source term S is explicit (only dependent on the time step n), while the sink term $\left(-ac_p c_f - bc_p c_p\right)$ is semi-implicit. This also corresponds to the way of dealing with the sink and source terms in TELEMAC-3D. In this case, the source term S is put in the variable **S0TA**, while the sink term

after implicitation $\left(ac_f^n + bc_p^n\right)$ is assigned to **S1TA**. Both **S0TA** and **S1TA** will be used in the subroutine **cvdf3d.f** for solving the MCPBE model. Note that the sink term is positive in **S1TA**.

In case the sink term has a different form, as in the following equation:

$$\frac{\partial c_p}{\partial t} = -ac_f + S + ADV + DIFF \tag{17}$$

Dropping the advection and diffusion terms, and applying the Patankar method to the sink term, it becomes:

$$\frac{c_p^{n+1} - c_p^n}{\Delta T} = -ac_f^n \frac{c_p^{n+1}}{c_p^n} + S \tag{18}$$

$$c_p^{n+1} = \left(c_p^n + \Delta T \cdot S\right) \Big/ \left(1 + \Delta T \frac{ac_f^n}{c_p^n}\right) \tag{19}$$

The above techniques are applied to each of the sink and source terms in the MCPBE model. Note that the original unknowns in the MCPBE model are the number concentrations (N_i). However, in TELEMAC-3D they are rescaled to C_i by multiplying N_i with mass of a fix-sized microfloc particle (m_p). The same m_p has to be added to the sink and source terms before applying the linearization techniques.

The comparison between the explicit and semi-implicit methods will be illustrated and further discussed in the validation report.

5.4.2 Computing bottom boundary conditions

Bottom boundary conditions are important while solving the MCPBEs. It requires the computation of erosion and deposition fluxes at the interface between water and sediment bed. Then based on these fluxes, the net flux is usually computed and imposed as the bottom boundary conditions when the flocculation processes are solved for the water column.

It is worth pointing out that, an assumption of the erosion/deposition at the bed is adopted in the MCPBE flocculation model. It states that, all the sediment classes, including the auxiliary classes, can deposit to the bed, and the deposition flux is scaled to their settling velocity and local concentrations. When they are deposited to the bed, all the classes will be converted into “microflocs” due to large shear and bed forming process. Therefore, only one sediment class “microflocs” will be presented in the bed. During erosion, because the bed is entirely formed of “microflocs”, thus only “microflocs” can be eroded from the bottom. According to Forsberg et al. (2018), the cohesive bed sediment is eroded as aggregates. Therefore, the erosion flux of “microflocs” will be redistributed among other sediment classes based on their near bed fractions (Figure 9).

In the module GAIA, a new framework is adopted for dealing with multiclass sediment and composition of the bed layer. According to this new framework, when multiple sediment classes are defined, which is the case for MCPBE flocculation model, these sediment classes will automatically appear in the bed composition and the initial ratio of each class will be given through the keyword **CLASSES INITIAL FRACTION**. Moreover, each sediment class will have its erosion and deposition fluxes computed, resulting in a dynamically changing bed composition after computation of bed evolution at each time step.

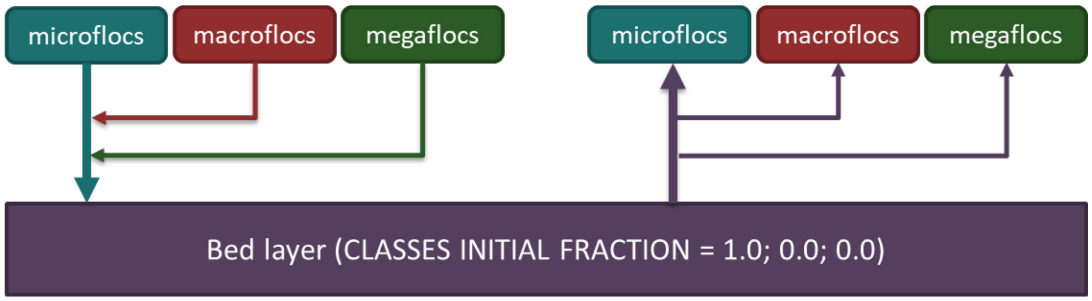

Figure 9 – Treatment of erosion and deposition fluxes in the MCPBE model

In order to be consistent with the assumptions in the MCPBE flocculation models, the following settings in the model (in the steering file) are required.

For computing the erosion flux, it is requested that the keyword **CLASSES INITIAL FRACTION = 1.0;0.0;0.0** (2CPBE) or **1.0;0.0;0.0;0.0;0.0** (3CPBE), assuming we only have cohesive sediment here. In such case, the first class is considered as the microflocs and it has 100% mass fraction in the bed composition. The rest cohesive sediment classes have 0% mass fraction in the bed, therefore, they are not able to be eroded from the bed, hence their erosion fluxes will be zero.

For ensuring the erosion flux for the other cohesive sediment classes ("macroflocs" and "megaflocs") to be zero, it requires the explicit deposition flux (**FLUDP**) from these two classes to be reset to zero while computing the bottom evolution and updating the bed composition, so that they will never appear in the bed layer. But the implicit deposition flux (**FLUDPT**) should be calculated and passed into the solver during the diffusion step (in **cvdf3d.f**) as part of the boundary conditions while solving the MCPBEs.

Fortunately, the implicit deposition flux (**FLUDPT**, which equals to the settling velocity) is required for solving the MCPBEs in **cvdf3d.f**, while the explicit deposition flux (**FLUDP**) is calculated later and only used in the bed evolution step and bed layer updating in **telemac3d.f**. This provides the possibility for treating them separately for the MCPBE flocculation models, in order to fulfill the above erosion/deposition assumptions.

After these treatments for the erosion and deposition fluxes, the bed evolution will be computed correctly and this part is taken care by GAIA.

(1). The variables for erosion and deposition

The variables **FLUDPT** (implicit deposition flux), **FLUDP** (explicit deposition flux) and **FLUER** (erosion flux) are declared for all the sediment classes, including both the "real" and "auxiliary" sediment classes. In some cases, total number of tracers may not be the same as the total number of sediment classes, hence, a variable in GAIA provides a way of associating the suspended sediment class and its deposition/erosion fluxes. In **lecdon_gaia.f**, a variable **I** = **NUM_ISUSP_ICLA(ITRAC-IND_SED+1)** takes the tracer number **ITRAC** as input and gives out the suspension number **I** that can be used as the index for erosion/deposition fluxes.

(2). Computing the erosion flux

The erosion flux (**FLUER**) is computed in the subroutine **compute_bc_sedi.f**, together with the implicit deposition flux. According to the assumptions adopted in the MCPBE model, the bed layer only consists of "microflocs". This determines that the original erosion flux (computed in GAIA) only contains "microflocs" as well. However, based on Forsberg et al. (2018), bed cohesive sediments can be eroded as aggregates. Hence, after the erosion of "microflocs" from the bed, the erosion flux will be immediately redistributed among all the classes based on their corresponding mass fraction near the bottom. The following code is added in the subroutine **compute_bc_sedi.f** for dealing with 2CPBE and 3CPBE models.

```
!    EROSION FLUX FOR MCPBE FLOCCULATION MODEL
     IF(FLOC.AND.FLOC_TYPE.EQ.3) THEN

      TATOT = 0.D0
      ERORAT1 = 0.D0
      ERORAT2 = 0.D0

     IF(MCPBE_VER.EQ.1) THEN
      IDMICF = NUM_ISUSP_ICLA(IMICFLC-IND_SED+1)
      IDMACF = NUM_ISUSP_ICLA(IMACFLC-IND_SED+1)
      IDMICMAC = NUM_ISUSP_ICLA(IMICF_MACF-IND_SED+1)
      DO IPOIN=1,NPOIN2
       TATOT(IPOIN)=MAX(1.D-6,
     &     TA%ADR(IMICFLC)%P%R(IPOIN)+TA%ADR(IMICF_MACF)%P%R(IPOIN))

       ERORAT1(IPOIN)=TA%ADR(IMICF_MACF)%P%R(IPOIN)/TATOT(IPOIN)
       FLUER%ADR(IDMICMAC)%P%R(IPOIN)=ERORAT1(IPOIN)
     &                        *FLUER%ADR(IDMICF)%P%R(IPOIN)
       FLUER%ADR(IDMACF)%P%R(IPOIN)=
     &             FLUER%ADR(IDMICMAC)%P%R(IPOIN)/NC_MAC%R(IPOIN)

       FLUER%ADR(IDMICF)%P%R(IPOIN)=FLUER%ADR(IDMICF)%P%R(IPOIN)
     &                       *(1.D0-ERORAT1(IPOIN))
      ENDDO
      IF(DBPOIN_MCPBE.GT.0.AND.MOD(LT,LISPRD).EQ.0) THEN
       WRITE(LU,*) 'TOTAL CONC.',TATOT(DBPOIN_MCPBE),
     &             'ERORAT1',ERORAT1(DBPOIN_MCPBE),
     &             'NSICLA',NSICLA,
     &             'NUM_ISUSP_ICLA(IMICFLC)',
     &              NUM_ISUSP_ICLA(IMICFLC-IND_SED+1),
     &             'NUM_ISUSP_ICLA(IMACFLC)',
     &              NUM_ISUSP_ICLA(IMACFLC-IND_SED+1),
     &             'NUM_ISUSP_ICLA(IMICF_MACF)',
     &              NUM_ISUSP_ICLA(IMICF_MACF-IND_SED+1)
       WRITE(LU,*) 'EROSION_FLUX_MICFLC',
     &                   FLUER%ADR(IDMICF)%P%R(DBPOIN_MCPBE),
     &             'EROSION_FLUX_MACFLC',
```

```
&                    FLUER%ADR(IDMACF)%P%R(DBPOIN_MCPBE),
&             'EROSION_FLUX_MIC_IN_MAC',
&                    FLUER%ADR(IDMICMAC)%P%R(DBPOIN_MCPBE)
  ENDIF
 ENDIF

 IF(MCPBE_VER.EQ.2) THEN
  IDMICF = NUM_ISUSP_ICLA(IMICFLC-IND_SED+1)
  IDMACF = NUM_ISUSP_ICLA(IMACFLC-IND_SED+1)
  IDMICMAC = NUM_ISUSP_ICLA(IMICF_MACF-IND_SED+1)
  IDMICMEG = NUM_ISUSP_ICLA(IMICF_MEGF-IND_SED+1)
  DO IPOIN=1,NPOIN2
   TATOT(IPOIN)=MAX(1.D-6,TA%ADR(IMICFLC)%P%R(IPOIN)
&                    +TA%ADR(IMICF_MACF)%P%R(IPOIN)
&                    +TA%ADR(IMICF_MEGF)%P%R(IPOIN))

   ERORAT1(IPOIN)=TA%ADR(IMICF_MACF)%P%R(IPOIN)/TATOT(IPOIN)
   FLUER%ADR(IDMICMAC)%P%R(IPOIN)=
&              ERORAT1(IPOIN)*FLUER%ADR(IDMICF)%P%R(IPOIN)
   FLUER%ADR(IDMACF)%P%R(IPOIN)=
&            FLUER%ADR(IDMICMAC)%P%R(IPOIN)/NC_MAC%R(IPOIN)

   ERORAT2(IPOIN)=TA%ADR(IMICF_MEGF)%P%R(IPOIN)/TATOT(IPOIN)
   FLUER%ADR(IDMICMEG)%P%R(IPOIN)=
&              ERORAT2(IPOIN)*FLUER%ADR(IDMICF)%P%R(IPOIN)

   FLUER%ADR(IDMICF)%P%R(IPOIN)=FLUER%ADR(IDMICF)%P%R(IPOIN)
&                   *(1.D0-ERORAT1(IPOIN)-ERORAT2(IPOIN))
  ENDDO
  IF(DBPOIN_MCPBE.GT.0.AND.MOD(LT,LISPRD).EQ.0) THEN
   WRITE(LU,*) 'TOTAL CONC.',TATOT(DBPOIN_MCPBE),
&             'ERORAT1',ERORAT1(DBPOIN_MCPBE),
&             'NSICLA',NSICLA,
&             'NUM_ISUSP_ICLA(IMICFLC)',
&              NUM_ISUSP_ICLA(IMICFLC-IND_SED+1),
&             'NUM_ISUSP_ICLA(IMACFLC)',
&              NUM_ISUSP_ICLA(IMACFLC-IND_SED+1),
```

```
&            'NUM_ISUSP_ICLA(IMICF_MACF)',
&             NUM_ISUSP_ICLA(IMICF_MACF-IND_SED+1),
&            'NUM_ISUSP_ICLA(IMICF_MEGF)',
&             NUM_ISUSP_ICLA(IMICF_MEGF-IND_SED+1)
    WRITE(LU,*) 'EROSION_FLUX_MICFLC',
&                   FLUER%ADR(IDMICF)%P%R(DBPOIN_MCPBE),
&            'EROSION_FLUX_MACFLC',
&                   FLUER%ADR(IDMACF)%P%R(DBPOIN_MCPBE),
&            'EROSION_FLUX_MICF_IN_MACF',
&                   FLUER%ADR(IDMICMAC)%P%R(DBPOIN_MCPBE),
&            'EROSION_FLUX_MICF_IN_MEGF',
&                   FLUER%ADR(IDMICMEG)%P%R(DBPOIN_MCPBE)
   ENDIF
  ENDIF

  IF(MCPBE_VER.EQ.3) THEN
   IDMICF = NUM_ISUSP_ICLA(IMICFLC-IND_SED+1)
   IDMACF = NUM_ISUSP_ICLA(IMACFLC-IND_SED+1)
   IDMEGF = NUM_ISUSP_ICLA(IMEGFLC-IND_SED+1)
   IDMICMAC = NUM_ISUSP_ICLA(IMICF_MACF-IND_SED+1)
   IDMICMEG = NUM_ISUSP_ICLA(IMICF_MEGF-IND_SED+1)
   DO IPOIN=1,NPOIN2
    TATOT(IPOIN)=MAX(1.D-6,TA%ADR(IMICFLC)%P%R(IPOIN)
&                   +TA%ADR(IMICF_MACF)%P%R(IPOIN)
&                   +TA%ADR(IMICF_MEGF)%P%R(IPOIN))

    ERORAT1(IPOIN)=TA%ADR(IMICF_MACF)%P%R(IPOIN)/TATOT(IPOIN)
    FLUER%ADR(IDMICMAC)%P%R(IPOIN)=
&             ERORAT1(IPOIN)*FLUER%ADR(IDMICF)%P%R(IPOIN)
    FLUER%ADR(IDMACF)%P%R(IPOIN)=
&           FLUER%ADR(IDMICMAC)%P%R(IPOIN)/NC_MAC%R(IPOIN)

    ERORAT2(IPOIN)=TA%ADR(IMICF_MEGF)%P%R(IPOIN)/TATOT(IPOIN)
    FLUER%ADR(IDMICMEG)%P%R(IPOIN)=
&             ERORAT2(IPOIN)*FLUER%ADR(IDMICF)%P%R(IPOIN)
    FLUER%ADR(IDMEGF)%P%R(IPOIN)=
&           FLUER%ADR(IDMICMEG)%P%R(IPOIN)/NC_MEG%R(IPOIN)
```

```
      FLUER%ADR(IDMICF)%P%R(IPOIN)=FLUER%ADR(IDMICF)%P%R(IPOIN)
     &                  *(1.D0-ERORAT1(IPOIN)-ERORAT2(IPOIN))
      ENDDO
      IF(DBPOIN_MCPBE.GT.0.AND.MOD(LT,LISPRD).EQ.0) THEN
       WRITE(LU,*) 'TOTAL CONC.',TATOT(DBPOIN_MCPBE),
     &            'ERORAT1',ERORAT1(DBPOIN_MCPBE),
     &            'NSICLA',NSICLA,
     &            'NUM_ISUSP_ICLA(IMICFLC)',
     &             NUM_ISUSP_ICLA(IMICFLC-IND_SED+1),
     &            'NUM_ISUSP_ICLA(IMACFLC)',
     &             NUM_ISUSP_ICLA(IMACFLC-IND_SED+1),
     &            'NUM_ISUSP_ICLA(IMICF_MACF)',
     &             NUM_ISUSP_ICLA(IMICF_MACF-IND_SED+1),
     &            'NUM_ISUSP_ICLA(IMEGFLC)',
     &             NUM_ISUSP_ICLA(IMEGFLC-IND_SED+1),
     &            'NUM_ISUSP_ICLA(IMICF_MEGF)',
     &             NUM_ISUSP_ICLA(IMICF_MEGF-IND_SED+1)
       WRITE(LU,*) 'EROSION_FLUX_MICFLC',
     &                  FLUER%ADR(IDMICF)%P%R(DBPOIN_MCPBE),
     &            'EROSION_FLUX_MACFLC',
     &                  FLUER%ADR(IDMACF)%P%R(DBPOIN_MCPBE),
     &            'EROSION_FLUX_MICF_IN_MACF',
     &                  FLUER%ADR(IDMICMAC)%P%R(DBPOIN_MCPBE),
     &            'EROSION_FLUX_MEGFLC',
     &                  FLUER%ADR(IDMEGF)%P%R(DBPOIN_MCPBE),
     &            'EROSION_FLUX_MICF_IN_MEGF',
     &                  FLUER%ADR(IDMICMEG)%P%R(DBPOIN_MCPBE)
      ENDIF
     ENDIF

     ENDIF
```

(2). Computing the implicit deposition flux

The implicit deposition flux (**FLUDPT**, equivalent to settling velocity) is used in the solver (**cvdf3d.f**) for the MCPBEs and it is computed in the subroutine **compute_bc_sedi.f**. The original code could already handle both the “real” and “auxiliary” sediment classes used in the MCPBE flocculation models. So no further modifications are needed here.

(3). Treatment for the explicit deposition flux

The explicit deposition flux (**FLUDP**) is used for the bed evolution and bed layer updating. The original code in **telemac3d.f** could compute the explicit deposition flux for all the sediment classes by multiplying the implicit deposition flux (**FLUDPT**, equivalent to settling velocity) with corresponding near bed concentration. In order to keep the bed composition unchanged throughout computation (only "microflocs" can enter or leave the bed layer), the explicit deposition fluxes (**FLUDP**) of all the other classes are converted into the explicit deposition flux of "microflocs", and then reset to zero before calculation of bed evolution, which will ensure the mass conservation. It is worth pointing out that when non-cohesive sediments are present in the model, the non-cohesive sediments can still mix with "microflocs" in the bed and in the water column. The treatment of explicit deposition flux only concerns the sediment classes used in the MCPBE flocculation model. A new subroutine **floc_deposition_mcpbe.f** is created (see Appendix VIII. Subroutine floc_deposition_mcpbe.f) for the treatment of the explicit deposition flux and called within the subroutine telemac3d.f at line 1957 just before the calculation of bed evolution.

```
!    TREATMENT OF DEPOSITION FLUX FOR MCPBE MODEL
     IF(FLOC .AND. FLOC_TYPE.EQ.3) CALL FLOC_DEPOSITION_MCPBE
!
```

In the subroutine **floc_deposition_mcpbe.f** the explicit deposition flux for "microflocs" is the sum of the explicit deposition fluxes for "microflocs" and "microflocs in macroflocs" (and "microflocs in megaflocs" in 3CPBE case). This is due to the fact that, in MCPBE flocculation models, the number concentration of each classes is rescaled by multiplying a constant (FLOCMIC_VOL*FLOCMIC_DEN) in order to avoid numerical instabilities and further modifications of the governing equations. Hence, this results in a special situation for the computed concentrations. The output concentrations for "microflocs" and "microflocs in macroflocs" (and "microflocs in megaflocs" in 3CPBE) are the correct mass concentrations for "microflocs" and "macroflocs" (and "megaflocs") respectively after the rescaling. But this is not the case for output concentration of "macroflocs" (and "megaflocs" in 3CPBE) since the multiplied constant does not correspond to its volume and density. Luckily we have already got its information from the class "microflocs in macroflocs" (and "microflocs in megaflocs" in 3CPBE).

Another important thing is to enforce the correct mass balance in the bed layer and it requires treating the implicit and explicit deposition fluxes separately. It should be pointed out that in MCPBEs all the sediment classes, including the auxiliary sediment classes, should have implicit deposition flux, since they will move out of the water column due to settling. However, it is a different story in terms of bed evolution and updating. The larger "macroflocs" and "megaflocs" will degrade to the "microflocs" when they settle down to the bed since their floc structures will be destroyed due to large shear near the bottom.

Thus, a special treatment for the explicit deposition flux is done in the above code to mimic this process. The explicit deposition fluxes of "microflocs in macroflocs" and "microflocs in megaflocs" are added to the explicit deposition fluxes of "microflocs", representing the deposited mass fluxes from "macroflocs" and "megaflocs" being converted into "microflocs" in the bed layer, then they are reset to zero. This ensures only microflocs are presented in the bed layer at the final stage of the settling process.

5.4.3 Solving MCPBEs

As mentioned before, the MCPBEs are eventually solved in the subroutine **cvdf3d.f**. It loops over all the tracers at each time step. Since each floc class is already correctly associated with its corresponding settling velocity, erosion/deposition fluxes, no more modifications are needed.

5.5 Deallocating variables after coupling

One thing that is necessary to be done at the end of simulation is to deallocate the memory for the declared BIEF objects. Because there are new global BIEF objects created for the MCPBE flocculation models, their memories have to be freed after use. This is done by inserting the following code in the subroutine **deall_telemac3d.f**.

```
    IF(FLOC.AND.FLOC_TYPE.EQ.3) THEN
      CALL BIEF_DEALLOBJ(FLOCMAC_DIA)
      CALL BIEF_DEALLOBJ(FLOCMAC_DEN)
      CALL BIEF_DEALLOBJ(NC_MAC)
      IF(MCPBE_VER.GT.1) THEN
        CALL BIEF_DEALLOBJ(FLOCMIC_DIA)
        CALL BIEF_DEALLOBJ(FLOCMEG_DEN)
        CALL BIEF_DEALLOBJ(FLOCMEG_DIA)
        CALL BIEF_DEALLOBJ(NC_MEG)
      ENDIF
    ENDIF
```

6 Conclusions

This report describes the implementation of the MCPBE flocculation models in TELEMAC system, specifically in TELEMAC-3D and GAIA.

Three different version of MCPBE flocculation models (2CPBE, 3CPBE with fixed megafloc size, 3CPBE with varying megafloc size), together with the existed empirical flocculation formulas in TELEMAC-3D are unified in this new framework. So one can choose freely between these models for the different applications.

The interaction with the bed layer is carefully treated, in order to fulfill the assumptions required by the MCPBE flocculation models. The implementation of this part does not conflict with the code structure in GAIA, so less modifications are needed.

The semi-implicit method is applied in the treatment of sink and source terms, in order to achieve better numerical stability. As demonstrated analytically, this method ensures the positivity of the concentrations, and allows the use of relatively larger time step. The semi-implicit method is the default option in the MCPBE model.

For the moment of writing this report, the module GAIA is still in development. The changes of the code in the GAIA are expected. Later those changes will also be adapted in the implementation of the MCPBE flocculation models if necessary.

7 References

Bi, Q., Shandro, E., Lee, B. J., & Toorman, E. (2016). Implementation of Two-class Population Balance Equation Bimodal Flocculation Model in TELEMAC-3D. TELEMAC-MASCARET User Conference. Paris (FR).

Fettweis, M., Baeye, M., Cardoso, C., Dujardin, A., Lauwaerts, B., Van den Eynde, D., Martens, C. (2016). The impact of disposal of fine grained sediments from maintenance dredging works on SPM concentration and fluid mud in and outside the harbor of Zeebrugge. Ocean Dynamics, 66, 1497-1516.

Forsberg, P. L., Skinnebach, K. H., Becker, M., Ernstsen, V. B., Kroon, A., & Andersen, T. J. (2018). The influence of aggregation on cohesive sediment erosion and settling. *Continental Shelf Research*, *171*, 52-62.

Kuprenas, R., Tran, D., & Strom, K. (2018). A shear-limited flocculation model for dynamically predicting average floc size. Journal of Geophysical Research: Oceans, 123(9), 6736-6752.

Lee, B. J., Toorman, E., Molz, F. J., & Wang, J. (2011). A two-class population balance equation yielding bimodal flocculation of marine or estuarine sediments. Water Research, 2131-2145.

Lee, B. J., & Schlautman, M. (2015). Effects of Polymer Molecular Weight on Adsorption and Flocculation in Aqueous Kaolinite Suspensions Dosed with Nonionic Polyacrylamides. Water, 5896-5909.

Kranenburg, C. (1999). Effects of floc strength on viscosity and deposition of cohesive sediment suspensions. Continental Shelf Research, 19(13), 1665-1680.

Maggi, F. (2005). Flocculation dynamics of cohesive sediment. Delft, The Netherlands: Technische Universiteit Delft.

Maggi, F., Mietta, F., Winterwerp, J.C., 2007. Effect of variable fractal dimension on the floc size distribution of suspended cohesive sediment. J. Hydrol. 343, 43-55.

Maerz, J., Verney, R., Wirtz, K., & Feudel, U. (2011). Modeling flocculation processes: Intercomparison of a size class-based model and a distribution-based model. Continental Shelf Research, 31(10), S84-S93.

Matsoukas, T., & Friedlander, S. K. (1991). Dynamics of aerosol agglomerate formation. Journal of Colloid and Interface Science, 146(2), 495-506.

Patankar, S.V. (1980), Numerical Heat Transfer and Fluid Flow, Hemisphere Publishing Corporation, Washington, DC.

Shen, X., Lee, B. J., Fettweis, M., & Toorman, E. A. (2018a). A tri-modal flocculation model coupled with TELEMAC for estuarine muds both in the laboratory and in the field. Water research(145), 473-486.

Shen, X., Toorman, E. A., Lee, B. J., & Fettweis, M. (2018b). Biophysical flocculation of suspended particulate matters in Belgian coastal zones. Journal of Hydrology, 238-252.

Soulsby, R. L., Manning, A. J., Spearman, J., & Whitehouse, R. J. S. (2013). Settling velocity and mass settling flux of flocculated estuarine sediments. Marine Geology, 339, 1-12.

Thomas, D. N., Judd, S. J., & Fawcett, N. (1999). Flocculation modelling: a review. Water research, 33(7), 1579-1592.

Toorman, E., & Berlamont, J. (2018). Dynamics of mud transport. Retrieved from Coastal Wiki: http://www.coastalwiki.org/wiki/Dynamics_of_mud_transport.

Tran, D., Kuprenas, R., & Strom, K. (2018). How do changes in suspended sediment concentration alone influence the size of mud flocs under steady turbulent shearing?. Continental Shelf Research, 158, 1-14.

Van Leussen, W. (1994). Estuarine Macroflocs: Their Role in Fine-grained Sediment.

Verney, R., Lafite, R., Brun-Cottan, J.C., Le Hir, P., 2011. Behaviour of a floc population during a tidal cycle: laboratory experiments and numerical modelling. Continent. Shelf Res. 31, S64-S83.

Winterwerp, J. C. (1998). A simple model for turbulence induced flocculation of cohesive sediment. Journal of hydraulic research, 36(3), 309-326.

Winterwerp, J. C., & Van Kesteren, W. G. (2004). Introduction to the physics of cohesive sediment dynamics in the marine environment. Elsevier.

Appendix I.
Proposed New Keywords in GAIA

NOM = 'FORMULE POUR FLOCULATION'

NOM1 = 'FLOCCULATION FORMULA'

TYPE = INTEGER

INDEX = 61

TAILLE = 1

DEFAUT = 1

DEFAUT1 = 1

MNEMO = 'FLOC_TYPE'

RUBRIQUE =

'COHESIF';'VITESSE DE CHUTE';''

RUBRIQUE1 =

'COHESIVE';'SETTLING VELOCITY';''

NIVEAU = 1

AIDE =

'Formule pour floculation :

\begin{itemize}

\item 1: Van Leussen ;

\item 2: Soulsby et al. (2013);

\item 3: MCPBE (KU Leuven).

\end{itemize}'

AIDE1 =

'Type of flocculation formula:

\begin{itemize}

\item 1: Van Leussen,

\item 2: Soulsby et al. (2013);

\item 3: MCPBE (KU Leuven).

\end{itemize}'

/=================================

NOM = 'MCPBE VERSION'

NOM1 = 'MCPBE VERSION'

TYPE = INTEGER

INDEX = 71

TAILLE = 1

DEFAUT = 1

DEFAUT1 = 1

MNEMO = 'MCPBE_VER'

RUBRIQUE =

‘FLOCULATION’;'FORMULE POUR FLOCULATION'

RUBRIQUE1 =

‘FLOCCULATION’;'FLOCCULATION FORMULA'

NIVEAU = 1

AIDE =

'Versions of MCPBE flocculation model:

\begin{itemize}

\item 1: TCPBE (3 eqs.);

\item 2: 3CPBE (4 eqs.);

\item 3: 3CPBE (5 eqs.).

\end{itemize}'

AIDE1 =

'Versions of MCPBE flocculation model:

\begin{itemize}

\item 1: TCPBE (3 eqs.);

\item 2: 3CPBE (4 eqs.);

\item 3: 3CPBE (5 eqs.).

\end{itemize}'

/=================================

NOM = 'PRINT POINT INFO MCPBE'

NOM1 = 'PRINT POINT INFO MCPBE'

TYPE = INTEGER

INDEX = 72

TAILLE = 1

DEFAUT = 0

DEFAUT1 = 0

MNEMO = 'DBPOIN_MCPBE'

RUBRIQUE =

'FORMULE POUR FLOCULATION';'MCPBE VERSION'

RUBRIQUE1 =

'FLOCCULATION FORMULA';'MCPBE VERSION'

NIVEAU = 1

AIDE =

'Printout info at specific point for MCPBE module'

AIDE1 =

'Printout info at specific point for MCPBE module'

/==================================

NOM = 'FRACTAL DIMENSION OF MACROFLOCS'

NOM1 = 'FRACTAL DIMENSION OF MACROFLOCS'

TYPE = REAL

INDEX = 61

TAILLE = 1

DEFAUT = 2.

DEFAUT1 = 2.

MNEMO = 'FRACDIM_MAC'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'ranges from 1 to 3, suggest set as a constant for all flocs at the moment'

AIDE1 =

'ranges from 1 to 3, suggest set as a constant for all flocs at the moment'

/==================================

NOM = 'FRACTAL DIMENSION OF MEGAFLOCS'

NOM1 = 'FRACTAL DIMENSION OF MEGAFLOCS'

TYPE = REAL

INDEX = 62

TAILLE = 1

DEFAUT = 2.

DEFAUT1 = 2.

MNEMO = 'FRACDIM_MEG'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'ranges from 1 to 3, suggest set as a constant for all flocs at the moment'

AIDE1 =

'ranges from 1 to 3, suggest set as a constant for all flocs at the moment'

/==================================

NOM = 'SIZE OF MICROFLOCS'

NOM1 = 'SIZE OF MICROFLOCS'

TYPE = REAL

INDEX = 63

TAILLE = 1

DEFAUT = 15.

DEFAUT1 = 15.

MNEMO = 'FLOCMIC_DIA'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'size of microflocs, ~4-30 micron'

AIDE1 =

'size of microflocs, ~4-30 micron'

/==================================

NOM = 'SIZE OF MEGAFLOCS'

NOM1 = 'SIZE OF MEGAFLOCS'

TYPE = REAL

INDEX = 64

TAILLE = 1

DEFAUT = 360.

DEFAUT1 = 360.

MNEMO = 'FLOCMEG_DIAFIX'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'size of megaflocs, ~200-400 micron, only used for fixed 3rd class in the 3-class 4-equation model'

AIDE1 =

'size of megaflocs, ~200-400 micron, only used for fixed 3rd class in the 3-class 4-equation model'

/=================================

NOM = 'MICROFLOC DENSITY'

NOM1 = 'MICROFLOC DENSITY'

TYPE = REAL

INDEX = 65

TAILLE = 1

DEFAUT = 2200.

DEFAUT1 = 2200.

MNEMO = 'FLOCMIC_DEN'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'density of microflocs, ~1600-2650 kg/m3'

AIDE1 =

'density of microflocs, ~1600-2650 kg/m3'

/=================================

NOM = 'FRACTION OF CREATED MICROFLOCS BY MACROFLOC BREAKUP'

NOM1 = 'FRACTION OF CREATED MICROFLOCS BY MACROFLOC BREAKUP'

TYPE = REAL

INDEX = 66

TAILLE = 1

DEFAUT = 0.1

DEFAUT1 = 0.1

MNEMO = 'BRKFRAC_P1'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'mass fraction of created microflocs when a macrofloc breaks up'

AIDE1 =

'mass fraction of created microflocs when a macrofloc breaks up'

/=================================

NOM = 'FRACTION OF CREATED MICROFLOCS BY MEGAFLOC BREAKUP'

NOM1 = 'FRACTION OF CREATED MICROFLOCS BY MEGAFLOC BREAKUP'

TYPE = REAL

INDEX = 67

TAILLE = 1

DEFAUT = 0.1

DEFAUT1 = 0.1

MNEMO = 'BRKFRAC_P2'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'mass fraction of created microflocs when a megafloc breaks up'

AIDE1 =

'mass fraction of created microflocs when a megafloc breaks up'

/=================================

NOM = 'FRACTION OF REMAINING MEGAFLOCS DURING MEGAFLOC BREAKUP’

NOM1 = 'FRACTION OF REMAINING MEGAFLOCS DURING MEGAFLOC BREAKUP'

TYPE = REAL

INDEX = 68

TAILLE = 1

DEFAUT = 0.0

DEFAUT1 = 0.0

MNEMO = 'BRKFRAC_F2'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'mass fraction of remaining megaflocs when a larger megafloc breaks up'

AIDE1 =

'mass fraction of remaining megaflocs when a larger megafloc breaks up'

/=================================

NOM = 'NUMBER OF CREATED MACROFLOCS BY MACROFLOC BREAKUP'

NOM1 = 'NUMBER OF CREATED MACROFLOCS BY MACROFLOC BREAKUP'

TYPE = REAL

INDEX = 69

TAILLE = 1

DEFAUT = 2.

DEFAUT1 = 2.

MNEMO = 'BRK_K1'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'number of created macroflocs when a parent macroflocs breaks up'

AIDE1 =

'number of created macroflocs when a parent macroflocs breaks up'

/=================================

NOM = 'NUMBER OF CREATED MACROFLOCS BY MEGAFLOC BREAKUP'

NOM1 = 'NUMBER OF CREATED MACROFLOCS BY MEGAFLOC BREAKUP'

TYPE = REAL

INDEX = 70

TAILLE = 1

DEFAUT = 2.

DEFAUT1 = 2.

MNEMO = 'BRK_K2'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'number of created macroflocs when a megaflocs breaks up'

AIDE1 =

'number of created macroflocs when a megaflocs breaks up'

/=================================

NOM = 'NUMBER OF CREATED MEGAFLOCS BY MEGAFLOC BREAKUP'

NOM1 = 'NUMBER OF CREATED MEGAFLOCS BY MEGAFLOC BREAKUP'

TYPE = REAL

INDEX = 71

TAILLE = 1

DEFAUT = 0.

DEFAUT1 = 0.

MNEMO = 'BRK_K3'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'number of created megaflocs when a larger megaflocs breaks up'

AIDE1 =

'number of created megaflocs when a larger megaflocs breaks up'

/==================================

NOM = 'COLLISION EFFICIENCY'

NOM1 = 'COLLISION EFFICIENCY'

TYPE = REAL

INDEX = 72

TAILLE = 1

DEFAUT = 0.2

DEFAUT1 = 0.2

MNEMO = 'AGG_ALPHA'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'collision efficiency based on the data fitting'

AIDE1 =

'collision efficiency based on the data fitting'

/==================================

NOM = 'BREAKUP EFFICIENCY'

NOM1 = 'BREAKUP EFFICIENCY'

TYPE = REAL

INDEX = 73

TAILLE = 1

DEFAUT = 1.E-04

DEFAUT1 = 1.E-04

MNEMO = 'BRK_EFF'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'breakup efficiency based on the data fitting'

AIDE1 =

'breakup efficiency based on the data fitting'

/==================================

NOM = 'FLOC YIELD STRENGTH'

NOM1 = 'FLOC YIELD STRENGTH'

TYPE = REAL

INDEX = 74

TAILLE = 1

DEFAUT = 1.E-10

DEFAUT1 = 1.E-10

MNEMO = 'BRK_FY'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'floc yield strength'

AIDE1 =

'floc yield strength'

/=================================

NOM = 'FLOC BREAKUP FREQUENCY COEFFICIENT'

NOM1 = 'FLOC BREAKUP FREQUENCY COEFFICIENT'

TYPE = REAL

INDEX = 75

TAILLE = 1

DEFAUT = 5.E-01

DEFAUT1 = 5.E-01

MNEMO = 'BRK_Q'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'Parameter in the floc breakup frequency function'

AIDE1 =

'Parameter in the floc breakup frequency function'

/=================================

NOM = 'BOLTZMANN CONSTANT'

NOM1 = 'BOLTZMANN CONSTANT'

TYPE = REAL

INDEX = 76

TAILLE = 1

DEFAUT = 1.38E-23

DEFAUT1 = 1.38E-23

MNEMO = 'KBOLZ'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'Boltzmann constant'

AIDE1 =

'Boltzmann constant'

/=================================

NOM = 'TEMPERATURE USED IN MCPBE MODEL'

NOM1 = 'TEMPERATURE USED IN MCPBE MODEL'

TYPE = REAL

INDEX = 77

TAILLE = 1

DEFAUT = 283.

DEFAUT1 = 283.

MNEMO = 'TEMPSTD'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'temperature used in mcpbe model'

AIDE1 =

'temperature used in mcpbe model'

/=================================

NOM = 'INITIAL NUMBER OF MICROFLOCS BOUNDED IN MACROFLOCS'

NOM1 = 'INITIAL NUMBER OF MICROFLOCS BOUNDED IN MACROFLOCS'

TYPE = REAL

INDEX = 78

TAILLE = 1

DEFAUT = 20.

DEFAUT1 = 20.

MNEMO = 'NC1_INI'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'initial number conc. of microflocs bounded in macroflocs'

AIDE1 =

'initial number conc. of microflocs bounded in macroflocs'

/==================================

NOM = 'INITIAL NUMBER OF MICROFLOCS BOUNDED IN MEGAFLOCS'

NOM1 = 'INITIAL NUMBER OF MICROFLOCS BOUNDED IN MEGAFLOCS'

TYPE = REAL

INDEX = 79

TAILLE = 1

DEFAUT = 100.

DEFAUT1 = 100.

MNEMO = 'NC2_INI'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'initial number conc. of microflocs bounded in megaflocs'

AIDE1 =

'initial number conc. of microflocs bounded in megaflocs'

/==================================

NOM = 'MAX NUMBER OF MICROFLOCS BOUNDED IN MACROFLOCS'

NOM1 = 'MAX NUMBER OF MICROFLOCS BOUNDED IN MACROFLOCS'

TYPE = REAL

INDEX = 80

TAILLE = 1

DEFAUT = 1000.

DEFAUT1 = 1000.

MNEMO = 'NC1_MAX'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'Maximum number of microflocs bounded in macroflocs'

AIDE1 =

'Maximum number of microflocs bounded in macroflocs'

/==================================

NOM = 'MAX NUMBER OF MICROFLOCS BOUNDED IN MEGAFLOCS'

NOM1 = 'MAX NUMBER OF MICROFLOCS BOUNDED IN MEGAFLOCS'

TYPE = REAL

INDEX = 81

TAILLE = 1

DEFAUT = 1000.

DEFAUT1 = 1000.

MNEMO = 'NC2_MAX'

RUBRIQUE =

‘FLOCULATION’;'MCPBE_VER'

RUBRIQUE1 =

‘FLOCCULATION’;'MCPBE_VER'

NIVEAU = 1

AIDE =

'Maximum number of microflocs bounded in megaflocs'

AIDE1 =

'Maximum number of microflocs bounded in megaflocs'

/==================================

NOM = 'FLOC BREAKUP FREQUENCY COEFFICIENT'

NOM1 = 'FLOC BREAKUP FREQUENCY
COEFFICIENT'

TYPE = REAL

INDEX = 82

TAILLE = 1

DEFAUT = 5.E-01

DEFAUT1 = 5.E-01

MNEMO = 'BRK_QC'

RUBRIQUE =

'FLOCULATION';'MCPBE_VER'

RUBRIQUE1 =

'FLOCCULATION';'MCPBE_VER'

NIVEAU = 1

AIDE =

'Parameter in the floc breakup frequency
function'

AIDE1 =

'Parameter in the floc breakup frequency
function'

/

Appendix II.
Subroutine floc_wchu_2cpbe.f

```
!              *************************
               SUBROUTINE FLOC_WCHU_2CPBE
!              *************************
!
!***********************************************************************
! TELEMAC3D
!***********************************************************************
!
!brief    COMPUTE SETTLING VELOCITY FOR MCPBE FLOCCULATION MODEL
!+           THIS SUBROUTINE IS ONLY USED FOR 2CPBE CASE.
!
!history  QILONG BI, BJ. LEE & X.SHEN
!+       09/05/19
!+       V8P1
!+
!
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!| NPOIN3         |-->| NUMBER OF POINTS IN 3D MESH
!| TA             |-->| TRACER (CONCENTRATIONS OF FLOCS)
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!
      USE BIEF
      USE DECLARATIONS_TELEMAC3D, ONLY: PRIVE,RHO0,NPOIN3,TA,WCHU,H,
     &                         FLOCMAC_DEN,FLOCMAC_DIA,NC_MAC,
     &                         NPOIN2,NPLAN,LT,LISPRD
      USE DECLARATIONS_GAIA, ONLY: HINDER,IMICFLC,IMACFLC,IMICF_MACF,
     &                    FRACDIM_MAC,FLOCMIC_DIAFIX,FLOCMIC_DEN,
     &                    DBPOIN_MCPBE,NC1_MAX
!
      USE DECLARATIONS_SPECIAL
      IMPLICIT NONE
!
!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!
      INTEGER IPOIN2,IPLAN,IPOIN
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: NC, CVOL, REP
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: PHI_HSP, PHI_HSF
      DOUBLE PRECISION, PARAMETER :: DVISC_W = 1.02D-3
!
!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
!
      DO IPLAN = 1,NPLAN
       DO IPOIN2 = 1,NPOIN2

        IF(H%R(IPOIN2).LT.1.D-2) THEN
         CONTINUE
        ELSE

        IPOIN=IPOIN2+(IPLAN-1)*NPOIN2

        NC(IPOIN) =
     &     TA%ADR(IMICF_MACF)%P%R(IPOIN)/TA%ADR(IMACFLC)%P%R(IPOIN)
        IF(ISNAN(NC(IPOIN)).OR.NC(IPOIN).GT.HUGE(NC(IPOIN))) THEN
          NC(IPOIN)=NC_MAC%R(IPOIN)
        ENDIF

        NC_MAC%R(IPOIN) = MIN(MAX(NC(IPOIN),2.D0),NC1_MAX)

        FLOCMAC_DEN%R(IPOIN) = RHO0+(FLOCMIC_DEN-RHO0)
     &                 *NC_MAC%R(IPOIN)**(1.D0-3.D0/FRACDIM_MAC)

        FLOCMAC_DIA%R(IPOIN) =
     &          NC_MAC%R(IPOIN)**(1.D0/FRACDIM_MAC)*FLOCMIC_DIAFIX

        CVOL(IPOIN) = (TA%ADR(IMICFLC)%P%R(IPOIN)
     &             +TA%ADR(IMICF_MACF)%P%R(IPOIN))/(FLOCMIC_DEN)

        IF(HINDER) THEN
          PHI_HSP(IPOIN) = (1.D0-CVOL(IPOIN))**4.D0
        ELSE
          PHI_HSP(IPOIN) = 1.D0
        ENDIF

        WCHU%ADR(IMICFLC)%P%R(IPOIN) =
     &                 (1.D0/18.D0*9.81*(FLOCMIC_DEN-RHO0)
     &                 *FLOCMIC_DIAFIX**2.D0)/DVISC_W

        WCHU%ADR(IMICFLC)%P%R(IPOIN) =
     &               PHI_HSP(IPOIN)*WCHU%ADR(IMICFLC)%P%R(IPOIN)

        IF(HINDER) THEN
          PHI_HSF(IPOIN) = (1.D0-CVOL(IPOIN))**4.D0
        ELSE
          PHI_HSF(IPOIN) = 1.D0
        ENDIF
```

```
      REP(IPOIN) = FLOCMAC_DIA%R(IPOIN)
     &                 *WCHU%ADR(IMACFLC)%P%R(IPOIN)*RHO0/DVISC_W
      WCHU%ADR(IMACFLC)%P%R(IPOIN) =
     &                 1.D0/18.D0*9.81*(FLOCMIC_DEN-RHO0)/DVISC_W
     &                 *FLOCMIC_DIAFIX**(3.D0-FRACDIM_MAC)
     &                 *FLOCMAC_DIA%R(IPOIN)**(FRACDIM_MAC-1.D0)
     &                 /(1.D0+0.15D0*REP(IPOIN)**0.687D0)

      WCHU%ADR(IMACFLC)%P%R(IPOIN) =
     &               PHI_HSF(IPOIN)*WCHU%ADR(IMACFLC)%P%R(IPOIN)

      WCHU%ADR(IMICF_MACF)%P%R(IPOIN)=WCHU%ADR(IMACFLC)%P%R(IPOIN)

!      WRITE INTO RESULTS
      PRIVE%ADR(1)%P%R(IPOIN)=FLOCMAC_DIA%R(IPOIN)
      PRIVE%ADR(2)%P%R(IPOIN)=FLOCMAC_DEN%R(IPOIN)
      PRIVE%ADR(3)%P%R(IPOIN)=WCHU%ADR(IMACFLC)%P%R(IPOIN)

     ENDIF

    ENDDO
   ENDDO

   IF (DBPOIN_MCPBE.GT.0.AND.MOD(LT,LISPRD).EQ.0) THEN
     WRITE(LU,*) 'WCHU_MICFLC',WCHU%ADR(IMICFLC)%P%R(DBPOIN_MCPBE),
   &           'WCHU_MACFLC',WCHU%ADR(IMACFLC)%P%R(DBPOIN_MCPBE),
   &      'WCHU_MIC_IN_MAC',WCHU%ADR(IMICF_MACF)%P%R(DBPOIN_MCPBE),
   &           'NC',NC_MAC%R(DBPOIN_MCPBE),'REP',REP(IPOIN),
   &           'TA_MICFLC',TA%ADR(IMICFLC)%P%R(DBPOIN_MCPBE),
   &           'TA_MACFLC',TA%ADR(IMACFLC)%P%R(DBPOIN_MCPBE),
   &           'TA_MICMAC',TA%ADR(IMICF_MACF)%P%R(DBPOIN_MCPBE),
   &           'FLOCMAC_DEN',FLOCMAC_DEN%R(DBPOIN_MCPBE),
   &           'FLOCMAC_DIA',FLOCMAC_DIA%R(DBPOIN_MCPBE)
    ENDIF
!
!-----------------------------------------------------------------------
!
!-----------------------------------------------------------------------
!
    RETURN
    END
```

Appendix III. Subroutine floc_wchu_3cpbe_var1.f

```
!              ******************************
               SUBROUTINE FLOC_WCHU_3CPBE_VAR1
!              ******************************
!
!***********************************************************************
! TELEMAC3D
!***********************************************************************
!
!brief    COMPUTE SETTLING VELOCITY FOR MCPBE FLOCCULATION MODEL
!+          THIS SUBROUTINE IS ONLY USED FOR 3CPBE CASE.
!
!history  QILONG BI, BJ. LEE & X.SHEN
!+      09/05/19
!+      V8P1
!+
!
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!| NPOIN3        |-->| NUMBER OF POINTS IN 3D MESH
!| TA            |-->| TRACER (CONCENTRATIONS OF FLOCS)
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!
      USE BIEF
      USE DECLARATIONS_TELEMAC3D, ONLY: NPOIN3,TA,WCHU,FLOCMIC_DIA,
     &                           FLOCMAC_DEN,FLOCMAC_DIA,PRIVE,
     &                           FLOCMEG_DEN,FLOCMEG_DIA,RHO0,
     &                           NPLAN,NPOIN2,NC_MEG,NC_MAC,
     &                           LT,LISPRD
      USE DECLARATIONS_GAIA, ONLY: MCPBE_VER,HINDER,IMICFLC,IMACFLC,
     &                 IMEGFLC,IMICF_MACF,IMICF_MEGF,FRACDIM_MAC,
     &                 FRACDIM_MEG,FLOCMIC_DEN,FLOCMEG_DIAFIX,
     &                 FLOCMIC_DIAFIX,DBPOIN_MCPBE,NC1_MAX,NC2_MAX
      USE DECLARATIONS_SPECIAL
      IMPLICIT NONE
!
!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!
      INTEGER IPOIN,IPLAN,IPOIN2
```

```
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: NC1,NC2,CVOL,PHI_HS
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: REP_MAC,REP_MEG
      DOUBLE PRECISION, PARAMETER :: DVISC_W = 1.02D-3
!
!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!
      DO IPLAN = 1,NPLAN
       DO IPOIN2 = 1,NPOIN2

        IPOIN=IPOIN2+(IPLAN-1)*NPOIN2
        FLOCMIC_DIA%R(IPOIN) = FLOCMIC_DIAFIX

!       3CPBE WITH FIXED MEGAFLOCS
        NC2(IPOIN)=(FLOCMEG_DIAFIX/FLOCMIC_DIA%R(IPOIN))**FRACDIM_MEG
        IF(ISNAN(NC2(IPOIN)).OR.NC2(IPOIN).GT.HUGE(NC2(IPOIN))) THEN
          NC2(IPOIN)=NC_MEG%R(IPOIN)
        ENDIF
        NC_MEG%R(IPOIN) =
     &            MIN(MAX(NC2(IPOIN),2.D0*NC_MAC%R(IPOIN)),NC2_MAX)
        FLOCMEG_DEN%R(IPOIN) = RHO0+(FLOCMIC_DEN-RHO0)
     &                    *NC_MEG%R(IPOIN)**(1.D0-3.D0/FRACDIM_MEG)
        FLOCMEG_DIA%R(IPOIN) = FLOCMEG_DIAFIX

        NC1(IPOIN) =
     &       TA%ADR(IMICF_MACF)%P%R(IPOIN)/TA%ADR(IMACFLC)%P%R(IPOIN)
        IF(ISNAN(NC1(IPOIN)).OR.NC1(IPOIN).GT.HUGE(NC1(IPOIN))) THEN
          NC1(IPOIN)=NC_MAC%R(IPOIN)
        ENDIF
        NC_MAC%R(IPOIN) = MIN(MAX(NC1(IPOIN),2.D0),NC1_MAX)
        FLOCMAC_DEN%R(IPOIN) = RHO0+(FLOCMIC_DEN-RHO0)
     &                    *NC_MAC%R(IPOIN)**(1.D0-3.D0/FRACDIM_MAC)
        FLOCMAC_DIA%R(IPOIN) =
     &       NC_MAC%R(IPOIN)**(1.D0/FRACDIM_MAC)*FLOCMIC_DIA%R(IPOIN)

        CVOL(IPOIN) = (TA%ADR(IMICFLC)%P%R(IPOIN)
     &             +TA%ADR(IMICF_MACF)%P%R(IPOIN)
     &             +TA%ADR(IMICF_MEGF)%P%R(IPOIN))/FLOCMIC_DEN

!       SETTLING VELOCITY WITH/WITHOUT HINDERED SETTLING
        IF(HINDER) THEN
          PHI_HS(IPOIN) = (1.D0-MIN(CVOL(IPOIN),0.7D0))**5.5D0
        ELSE
          PHI_HS(IPOIN) = 1.D0
```

```
      ENDIF

      REP_MAC(IPOIN) = FLOCMAC_DIA%R(IPOIN)
     &                 *WCHU%ADR(IMACFLC)%P%R(IPOIN)*RHO0/DVISC_W
      REP_MEG(IPOIN) = FLOCMEG_DIA%R(IPOIN)
     &                *WCHU%ADR(IMICF_MEGF)%P%R(IPOIN)*RHO0/DVISC_W

      WCHU%ADR(IMICFLC)%P%R(IPOIN) =
     &              1.D0/18.D0*9.81*(FLOCMIC_DEN-RHO0)/DVISC_W
     &                *(FLOCMIC_DIA%R(IPOIN)**2.D0)
      WCHU%ADR(IMACFLC)%P%R(IPOIN) =
     &              1.D0/18.D0*9.81*(FLOCMIC_DEN-RHO0)/DVISC_W
     &                *FLOCMIC_DIA%R(IPOIN)**(3.D0-FRACDIM_MAC)
     &                *FLOCMAC_DIA%R(IPOIN)**(FRACDIM_MAC-1.D0)
     &                /(1.D0+0.15D0*REP_MAC(IPOIN)**0.687D0)
      WCHU%ADR(IMICF_MEGF)%P%R(IPOIN) =
     &              1.D0/18.D0*9.81*(FLOCMIC_DEN-RHO0)/DVISC_W
     &                *FLOCMIC_DIA%R(IPOIN)**(3.D0-FRACDIM_MEG)
     &                *FLOCMEG_DIA%R(IPOIN)**(FRACDIM_MEG-1.D0)
     &                /(1.D0+0.15D0*REP_MEG(IPOIN)**0.687D0)

      WCHU%ADR(IMICFLC)%P%R(IPOIN) = PHI_HS(IPOIN)*
     &                          WCHU%ADR(IMICFLC)%P%R(IPOIN)
      WCHU%ADR(IMACFLC)%P%R(IPOIN) = PHI_HS(IPOIN)*
     &                          WCHU%ADR(IMACFLC)%P%R(IPOIN)
      WCHU%ADR(IMICF_MEGF)%P%R(IPOIN) = PHI_HS(IPOIN)*
     &                         WCHU%ADR(IMICF_MEGF)%P%R(IPOIN)

      WCHU%ADR(IMICF_MACF)%P%R(IPOIN) = WCHU%ADR(IMACFLC)%P%R(IPOIN)

!     WRITE INTO RESULTS
      PRIVE%ADR(1)%P%R(IPOIN)=FLOCMAC_DIA%R(IPOIN)
      PRIVE%ADR(2)%P%R(IPOIN)=FLOCMEG_DIA%R(IPOIN)
      PRIVE%ADR(3)%P%R(IPOIN)=WCHU%ADR(IMICF_MACF)%P%R(IPOIN)
      PRIVE%ADR(4)%P%R(IPOIN)=WCHU%ADR(IMICF_MEGF)%P%R(IPOIN)

     ENDDO
     ENDDO

     IF (DBPOIN_MCPBE.GT.0.AND.MOD(LT,LISPRD).EQ.0) THEN
      WRITE(LU,*) 'WCHU_MICFLC',WCHU%ADR(IMICFLC)%P%R(DBPOIN_MCPBE),
     &      'WCHU_MACFLC',WCHU%ADR(IMACFLC)%P%R(DBPOIN_MCPBE),
     &      'WCHU_MIC_IN_MAC',WCHU%ADR(IMICF_MACF)%P%R(DBPOIN_MCPBE),
```

```
     &      'WCHU_MIC_IN_MEG',WCHU%ADR(IMICF_MEGF)%P%R(DBPOIN_MCPBE),
     &      'NC_MAC',NC1(DBPOIN_MCPBE),'NC_MEG',NC2(DBPOIN_MCPBE),
     &      'REP_MAC',REP_MAC(DBPOIN_MCPBE),
     &      'REP_MEG',REP_MEG(DBPOIN_MCPBE),
     &      'TA_MICFLC',TA%ADR(IMICFLC)%P%R(DBPOIN_MCPBE),
     &      'TA_MACFLC',TA%ADR(IMACFLC)%P%R(DBPOIN_MCPBE),
     &      'TA_MIC_IN_MAC',TA%ADR(IMICF_MACF)%P%R(DBPOIN_MCPBE),
     &      'TA_MIC_IN_MEG',TA%ADR(IMICF_MEGF)%P%R(DBPOIN_MCPBE),
     &      'FLOCMAC_DEN',FLOCMAC_DEN%R(DBPOIN_MCPBE),
     &      'FLOCMAC_DIA',FLOCMAC_DIA%R(DBPOIN_MCPBE),
     &      'FLOCMEG_DEN',FLOCMEG_DEN%R(DBPOIN_MCPBE),
     &      'FLOCMEG_DIA',FLOCMEG_DIA%R(DBPOIN_MCPBE)
      ENDIF
!
!-----------------------------------------------------------------------
!
!-----------------------------------------------------------------------
!
      RETURN
      END
```

Appendix IV. Subroutine floc_wchu_3cpbe_var2.f

```
!              ******************************
               SUBROUTINE FLOC_WCHU_3CPBE_VAR2
!              ******************************
!
!***********************************************************************
! TELEMAC3D
!***********************************************************************
!
!brief    COMPUTE SETTLING VELOCITY FOR MCPBE FLOCCULATION MODEL
!+          THIS SUBROUTINE IS ONLY USED FOR 3CPBE CASE.
!
!history  QILONG BI, BJ. LEE & X.SHEN
!+      09/05/19
!+      V8P1
!+
!
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!| NPOIN3         |-->| NUMBER OF POINTS IN 3D MESH
!| TA             |-->| TRACER (CONCENTRATIONS OF FLOCS)
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!
      USE BIEF
      USE DECLARATIONS_TELEMAC3D, ONLY: NPOIN3,TA,WCHU,FLOCMIC_DIA,
     &                          FLOCMAC_DEN,FLOCMAC_DIA,PRIVE,
     &                          FLOCMEG_DEN,FLOCMEG_DIA,RHO0,
     &                          NPLAN,NPOIN2,NC_MEG,NC_MAC,
     &                          LT,LISPRD
      USE DECLARATIONS_GAIA, ONLY: MCPBE_VER,HINDER,IMICFLC,IMACFLC,
     &                IMEGFLC,IMICF_MACF,IMICF_MEGF,FRACDIM_MAC,
     &                FRACDIM_MEG,FLOCMIC_DEN,FLOCMEG_DIAFIX,
     &                FLOCMIC_DIAFIX,DBPOIN_MCPBE,NC1_MAX,NC2_MAX
      USE DECLARATIONS_SPECIAL
      IMPLICIT NONE
!
!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!
      INTEGER IPOIN,IPLAN,IPOIN2
```

```
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: NC1,NC2,CVOL,PHI_HS
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: REP_MAC,REP_MEG
      DOUBLE PRECISION, PARAMETER :: DVISC_W = 1.02D-3
!
!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!
      DO IPLAN = 1,NPLAN
       DO IPOIN2 = 1,NPOIN2

        IPOIN=IPOIN2+(IPLAN-1)*NPOIN2
        FLOCMIC_DIA%R(IPOIN) = FLOCMIC_DIAFIX

!       3CPBE WITH VARYING MEGAFLOCS
        NC1(IPOIN) =
     &        TA%ADR(IMICF_MACF)%P%R(IPOIN)/TA%ADR(IMACFLC)%P%R(IPOIN)
        IF(ISNAN(NC1(IPOIN)).OR.NC1(IPOIN).GT.HUGE(NC1(IPOIN))) THEN
          NC1(IPOIN)=NC_MAC%R(IPOIN)
        ENDIF
        NC_MAC%R(IPOIN) = MIN(MAX(NC1(IPOIN),2.D0),NC1_MAX)
        FLOCMAC_DEN%R(IPOIN) = RHO0+(FLOCMIC_DEN-RHO0)
     &                   *NC_MAC%R(IPOIN)**(1.D0-3.D0/FRACDIM_MAC)
        FLOCMAC_DIA%R(IPOIN) =
     &         NC_MAC%R(IPOIN)**(1.D0/FRACDIM_MAC)*FLOCMIC_DIA%R(IPOIN)

        NC2(IPOIN) =
     &        TA%ADR(IMICF_MEGF)%P%R(IPOIN)/TA%ADR(IMEGFLC)%P%R(IPOIN)
        IF(ISNAN(NC2(IPOIN)).OR.NC2(IPOIN).GT.HUGE(NC2(IPOIN))) THEN
          NC2(IPOIN)=NC_MEG%R(IPOIN)
        ENDIF
        NC_MEG%R(IPOIN) =
     &             MIN(MAX(NC2(IPOIN),2.D0*NC_MAC%R(IPOIN)),NC2_MAX)
        FLOCMEG_DEN%R(IPOIN) = RHO0+(FLOCMIC_DEN-RHO0)
     &                   *NC_MEG%R(IPOIN)**(1.D0-3.D0/FRACDIM_MEG)
        FLOCMEG_DIA%R(IPOIN) =
     &         NC_MEG%R(IPOIN)**(1.D0/FRACDIM_MEG)*FLOCMIC_DIA%R(IPOIN)

        CVOL(IPOIN) = (TA%ADR(IMICFLC)%P%R(IPOIN)
     &              +TA%ADR(IMICF_MACF)%P%R(IPOIN)
     &              +TA%ADR(IMICF_MEGF)%P%R(IPOIN))/FLOCMIC_DEN

!       SETTLING VELOCITY WITH/WITHOUT HINDERED SETTLING
        IF(HINDER) THEN
          PHI_HS(IPOIN) = (1.D0-MIN(CVOL(IPOIN),0.7D0))**5.5D0
```

```
      ELSE
        PHI_HS(IPOIN) = 1.D0
      ENDIF

      REP_MAC(IPOIN) = FLOCMAC_DIA%R(IPOIN)
     &                 *WCHU%ADR(IMACFLC)%P%R(IPOIN)*RHO0/DVISC_W
      REP_MEG(IPOIN) = FLOCMEG_DIA%R(IPOIN)
     &                 *WCHU%ADR(IMEGFLC)%P%R(IPOIN)*RHO0/DVISC_W

      WCHU%ADR(IMICFLC)%P%R(IPOIN) =
     &              1.D0/18.D0*9.81*(FLOCMIC_DEN-RHO0)/DVISC_W
     &                 *(FLOCMIC_DIA%R(IPOIN)**2.D0)
      WCHU%ADR(IMACFLC)%P%R(IPOIN) =
     &              1.D0/18.D0*9.81*(FLOCMIC_DEN-RHO0)/DVISC_W
     &                 *FLOCMIC_DIA%R(IPOIN)**(3.D0-FRACDIM_MAC)
     &                 *FLOCMAC_DIA%R(IPOIN)**(FRACDIM_MAC-1.D0)
     &                 /(1.D0+0.15D0*REP_MAC(IPOIN)**0.687D0)
      WCHU%ADR(IMEGFLC)%P%R(IPOIN) =
     &              1.D0/18.D0*9.81*(FLOCMIC_DEN-RHO0)/DVISC_W
     &                 *FLOCMIC_DIA%R(IPOIN)**(3.D0-FRACDIM_MEG)
     &                 *FLOCMEG_DIA%R(IPOIN)**(FRACDIM_MEG-1.D0)
     &                 /(1.D0+0.15D0*REP_MEG(IPOIN)**0.687D0)

      WCHU%ADR(IMICFLC)%P%R(IPOIN) = PHI_HS(IPOIN)*
     &                          WCHU%ADR(IMICFLC)%P%R(IPOIN)
      WCHU%ADR(IMACFLC)%P%R(IPOIN) = PHI_HS(IPOIN)*
     &                          WCHU%ADR(IMACFLC)%P%R(IPOIN)
      WCHU%ADR(IMEGFLC)%P%R(IPOIN) = PHI_HS(IPOIN)*
     &                          WCHU%ADR(IMEGFLC)%P%R(IPOIN)

      WCHU%ADR(IMICF_MACF)%P%R(IPOIN) = WCHU%ADR(IMACFLC)%P%R(IPOIN)
      WCHU%ADR(IMICF_MEGF)%P%R(IPOIN) = WCHU%ADR(IMEGFLC)%P%R(IPOIN)

!     WRITE INTO RESULTS
      PRIVE%ADR(1)%P%R(IPOIN)=FLOCMAC_DIA%R(IPOIN)
      PRIVE%ADR(2)%P%R(IPOIN)=FLOCMEG_DIA%R(IPOIN)
      PRIVE%ADR(3)%P%R(IPOIN)=WCHU%ADR(IMACFLC)%P%R(IPOIN)
      PRIVE%ADR(4)%P%R(IPOIN)=WCHU%ADR(IMEGFLC)%P%R(IPOIN)

     ENDDO
    ENDDO

    IF (DBPOIN_MCPBE.GT.0.AND.MOD(LT,LISPRD).EQ.0) THEN
```

```
      WRITE(LU,*) 'WCHU_MICFLC',WCHU%ADR(IMICFLC)%P%R(DBPOIN_MCPBE),
     &      'WCHU_MACFLC',WCHU%ADR(IMACFLC)%P%R(DBPOIN_MCPBE),
     &      'WCHU_MIC_IN_MAC',WCHU%ADR(IMICF_MACF)%P%R(DBPOIN_MCPBE),
     &      'WCHU_MEGFLC',WCHU%ADR(IMEGFLC)%P%R(DBPOIN_MCPBE),
     &      'WCHU_MIC_IN_MEG',WCHU%ADR(IMICF_MEGF)%P%R(DBPOIN_MCPBE),
     &      'NC_MAC',NC1(DBPOIN_MCPBE),'NC_MEG',NC2(DBPOIN_MCPBE),
     &      'REP_MAC',REP_MAC(DBPOIN_MCPBE),
     &      'REP_MEG',REP_MEG(DBPOIN_MCPBE),
     &      'TA_MICFLC',TA%ADR(IMICFLC)%P%R(DBPOIN_MCPBE),
     &      'TA_MACFLC',TA%ADR(IMACFLC)%P%R(DBPOIN_MCPBE),
     &      'TA_MIC_IN_MAC',TA%ADR(IMICF_MACF)%P%R(DBPOIN_MCPBE),
     &      'TA_MEGFLC',TA%ADR(IMEGFLC)%P%R(DBPOIN_MCPBE),
     &      'TA_MIC_IN_MEG',TA%ADR(IMICF_MEGF)%P%R(DBPOIN_MCPBE),
     &      'FLOCMAC_DEN',FLOCMAC_DEN%R(DBPOIN_MCPBE),
     &      'FLOCMAC_DIA',FLOCMAC_DIA%R(DBPOIN_MCPBE),
     &      'FLOCMEG_DEN',FLOCMEG_DEN%R(DBPOIN_MCPBE),
     &      'FLOCMEG_DIA',FLOCMEG_DIA%R(DBPOIN_MCPBE)
      ENDIF
!
!-----------------------------------------------------------------------
!
!-----------------------------------------------------------------------
!
      RETURN
      END
```

Appendix V. Subroutine floc_kinetics_2cpbe.f

```
!              *****************************
               SUBROUTINE FLOC_KINETICS_2CPBE
!              *****************************
!
!***********************************************************************
! TELEMAC3D
!***********************************************************************
!
!brief    COMPUTE FLOCCULATION KINETICS FOR MCPBE FLOCCULATION MODEL
!+             THIS SUBROUTINE IS ONLY USED FOR 2CPBE CASE.
!
!history  QILONG BI, BJ. LEE & X.SHEN
!+        09/05/19
!+        V8P1
!+
!
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!| NPOIN3         |-->| NUMBER OF POINTS IN 3D MESH
!| TA             |-->| TRACER (CONCENTRATIONS OF FLOCS)
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!
      USE BIEF
      USE DECLARATIONS_GAIA
      USE DECLARATIONS_TELEMAC3D, ONLY: NPOIN3,TA,S0TA,S1TA,DNUVIV,EP,
     &                            FLOCMAC_DIA,RHO0,WCHU,NPOIN2,
     &                            UETCAR,H,NC_MAC,NPLAN,LISPRD
      USE DECLARATIONS_SPECIAL
      IMPLICIT NONE

!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!
      INTEGER IPOIN2,IPLAN,IPOIN
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: CNUM_P,CNUM_F,CNUM_T,NC,
     &                           SHR_G,BETA_PP,BETA_FF,BETA_PF,
     &                           BRK_SH,ABKE_P0,ABKE_F0,ABKE_T0,
     &                           ABKE_P1,ABKE_F1,ABKE_T1
```

```
      DOUBLE PRECISION :: FLOCMIC_VOL,VMU,WCHU_P,WCHU_F
!
!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

      IF(FLOC .AND. FLOC_TYPE.EQ.3 .AND. MCPBE_VER.EQ.1) THEN
!       SOURCE TERM FOR MICROFLOCS
        S0TA%ADR(IMICFLC)%P%TYPR='Q'
        S1TA%ADR(IMICFLC)%P%TYPR='Q'
        CALL OS('X=C     ',S0TA%ADR(IMICFLC)%P,C=0.D0)
        CALL OS('X=C     ',S1TA%ADR(IMICFLC)%P,C=0.D0)
!       SOURCE TERM FOR MACROFLOCS
        S0TA%ADR(IMACFLC)%P%TYPR='Q'
        S1TA%ADR(IMACFLC)%P%TYPR='Q'
        CALL OS('X=C     ',S0TA%ADR(IMACFLC)%P,C=0.D0)
        CALL OS('X=C     ',S1TA%ADR(IMACFLC)%P,C=0.D0)
!       SOURCE TERM FOR MICROFLOCS INSIDE MACROFLOCS
        S0TA%ADR(IMICF_MACF)%P%TYPR='Q'
        S1TA%ADR(IMICF_MACF)%P%TYPR='Q'
        CALL OS('X=C     ',S0TA%ADR(IMICF_MACF)%P,C=0.D0)
        CALL OS('X=C     ',S1TA%ADR(IMICF_MACF)%P,C=0.D0)
      ENDIF

!     COMPUTE FLOCCULATION KINETICS

      FLOCMIC_VOL=(1.D0/6.D0)*3.14159D0*FLOCMIC_DIAFIX**3.D0
      VMU = DNUVIV * RHO0

      DO IPLAN = 1,NPLAN
       DO IPOIN2 = 1,NPOIN2

        IPOIN=IPOIN2+(IPLAN-1)*NPOIN2

        IF(H%R(IPOIN2).LT.1.D-2) THEN
          S0TA%ADR(IMICFLC)%P%R(IPOIN)=0.D0
          S0TA%ADR(IMACFLC)%P%R(IPOIN)=0.D0
          S0TA%ADR(IMICF_MACF)%P%R(IPOIN)=0.D0
          S1TA%ADR(IMICFLC)%P%R(IPOIN)=0.D0
          S1TA%ADR(IMACFLC)%P%R(IPOIN)=0.D0
          S1TA%ADR(IMICF_MACF)%P%R(IPOIN)=0.D0
        ELSE

        CNUM_P(IPOIN) =
     &        TA%ADR(IMICFLC)%P%R(IPOIN)/(FLOCMIC_VOL*FLOCMIC_DEN)
```

```
      CNUM_F(IPOIN) =
     &       TA%ADR(IMACFLC)%P%R(IPOIN)/(FLOCMIC_VOL*FLOCMIC_DEN)
      CNUM_T(IPOIN) =
     &     TA%ADR(IMICF_MACF)%P%R(IPOIN)/(FLOCMIC_VOL*FLOCMIC_DEN)

      NC(IPOIN) = CNUM_T(IPOIN)/CNUM_F(IPOIN)
      IF(ISNAN(NC(IPOIN)).OR.NC(IPOIN).GT.HUGE(NC(IPOIN))) THEN
       NC(IPOIN)=NC_MAC%R(IPOIN)
      ENDIF
      NC_MAC%R(IPOIN) = MIN(MAX(NC(IPOIN),2.D0),NC1_MAX)

!     Modification for testing TCBPE kinetics only
!     SHR_G(IPOIN)=50.0D0

!     Here, Please create a depth and shear velocity dependent "SHR_G"
      SHR_G(IPOIN) = SQRT(EP%R(IPOIN)/DNUVIV)

!     FOR THE BOTTOM NODE, EP COULD ALSO USE THEORETICAL VALUE (ML)
!     IF(IPOIN.LE.NPOIN2) THEN
!      SHR_G(IPOIN) = SQRT(UETCAR%R(IPOIN)**1.5/(KARMAN*0.001)
!  &                 *(1-0.001/MAX(H%R(IPOIN),0.001))/DNUVIV)
!     ENDIF
!     SHR_G(IPOIN) = MAX(SHR_G(IPOIN),1.D0)

      WCHU_P = WCHU%ADR(IMICFLC)%P%R(IPOIN)
      WCHU_F = WCHU%ADR(IMACFLC)%P%R(IPOIN)

      BETA_PP(IPOIN) = 2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)*4.D0
     &         +(1.D0/6.D0)*(2.D0*FLOCMIC_DIAFIX)**3.D0*SHR_G(IPOIN)

      BETA_FF(IPOIN) = 2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)*4.D0
     &     +(1.D0/6.D0)*(2.D0*FLOCMAC_DIA%R(IPOIN))**3.D0*SHR_G(IPOIN)

      BETA_PF(IPOIN) = 2.D0*KBOLZ*TEMPSTD
     &       *(1.D0/FLOCMIC_DIAFIX+1.D0/FLOCMAC_DIA%R(IPOIN))
     &       *(FLOCMIC_DIAFIX+FLOCMAC_DIA%R(IPOIN))/(3.D0*VMU)
     &       +(1.D0/6.D0)*(FLOCMIC_DIAFIX+FLOCMAC_DIA%R(IPOIN))**3.D0
     &       *SHR_G(IPOIN)+3.14159D0/4.D0
     &       *(FLOCMIC_DIAFIX+FLOCMAC_DIA%R(IPOIN))**2.D0
     &       *ABS(WCHU_P-WCHU_F)

!     Modification Proposed by Kyle Strom (2018)
      BRK_SH(IPOIN) = BRK_EFF*SHR_G(IPOIN)*((FLOCMAC_DIA%R(IPOIN)
```

```
     &        -FLOCMIC_DIAFIX)/FLOCMIC_DIAFIX)**(3.D0-FRACDIM_MAC)
     &        *(VMU*SHR_G(IPOIN)/(BRK_FY/FLOCMAC_DIA%R(IPOIN)**2.D0))
     &        **(BRK_Q+BRK_QC*FLOCMAC_DIA%R(IPOIN)/(DNUVIV**3.D0
     &        /EP%R(IPOIN))**0.25D0)

! CALCULATE THE AGGREGATION AND BREAKAGE KERNELS
      ABKE_P0(IPOIN) =
     &        BRKFRAC_P1*NC_MAC%R(IPOIN)*BRK_SH(IPOIN)*CNUM_F(IPOIN)

      ABKE_F0(IPOIN) = 0.5D0*AGG_ALPHA*BETA_PP(IPOIN)
     &    *(CNUM_P(IPOIN)*CNUM_P(IPOIN))*(1.D0/(NC_MAC%R(IPOIN)-1.D0))
     &    +BRK_SH(IPOIN)*CNUM_F(IPOIN)

      ABKE_T0(IPOIN) = 0.5D0*AGG_ALPHA*BETA_PP(IPOIN)*CNUM_P(IPOIN)
     &        *CNUM_P(IPOIN)*(NC_MAC%R(IPOIN)/(NC_MAC%R(IPOIN)-1.D0))
     &        +AGG_ALPHA*BETA_PF(IPOIN)*CNUM_P(IPOIN)*CNUM_F(IPOIN)

      ABKE_P1(IPOIN) = 0.5D0*AGG_ALPHA*BETA_PP(IPOIN)
     &        *CNUM_P(IPOIN)*(NC_MAC%R(IPOIN)/(NC_MAC%R(IPOIN)-1.D0))
     &        +AGG_ALPHA*BETA_PF(IPOIN)*CNUM_F(IPOIN)

      ABKE_F1(IPOIN) =
     &         0.5D0*AGG_ALPHA*BETA_FF(IPOIN)*CNUM_F(IPOIN)

      ABKE_T1(IPOIN) =
     &        BRKFRAC_P1*NC_MAC%R(IPOIN)*BRK_SH(IPOIN)
     &       *TA%ADR(IMACFLC)%P%R(IPOIN)
     &       /MAX(TA%ADR(IMICF_MACF)%P%R(IPOIN),1.D-15)

      S0TA%ADR(IMICFLC)%P%R(IPOIN) =
     &                 ABKE_P0(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN
      S0TA%ADR(IMACFLC)%P%R(IPOIN) =
     &                 ABKE_F0(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN
      S0TA%ADR(IMICF_MACF)%P%R(IPOIN) =
     &                 ABKE_T0(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN

      S1TA%ADR(IMICFLC)%P%R(IPOIN) =
     &                 ABKE_P1(IPOIN)
      S1TA%ADR(IMACFLC)%P%R(IPOIN) =
     &                 ABKE_F1(IPOIN)
      S1TA%ADR(IMICF_MACF)%P%R(IPOIN) =
     &                 ABKE_T1(IPOIN)
```

```
!      IF (ABKE_P(IPOIN).LE.0.D0) THEN
!        S0TA%ADR(IMICFLC)%P%R(IPOIN) = MAX(
!    &                  ABKE_P(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN,
!    &                  -MAX(0.D0,TA%ADR(IMICFLC)%P%R(IPOIN))/DT)
!        S0TA%ADR(IMACFLC)%P%R(IPOIN) =
!    &                  ABKE_F(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN
!        S0TA%ADR(IMICF_MACF)%P%R(IPOIN) =
!    &                  -S0TA%ADR(IMICFLC)%P%R(IPOIN)
!      ELSE
!        S0TA%ADR(IMICF_MACF)%P%R(IPOIN) = MAX(
!    &                ABKE_T(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN,
!    &                -MAX(0.D0,TA%ADR(IMICF_MACF)%P%R(IPOIN))/DT)
!        S0TA%ADR(IMACFLC)%P%R(IPOIN) = MAX(
!    &                ABKE_F(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN,
!    &                -MAX(0.D0,TA%ADR(IMACFLC)%P%R(IPOIN))/DT)
!        S0TA%ADR(IMICFLC)%P%R(IPOIN) =
!    &                 -S0TA%ADR(IMICF_MACF)%P%R(IPOIN)
!      ENDIF

     ENDIF

     ENDDO
    ENDDO

    IF(DBPOIN_MCPBE.GT.0.AND.MOD(LT,LISPRD).EQ.0) THEN
      WRITE(LU,*) 'H',H%R(DBPOIN_MCPBE),'SHR_G',SHR_G(DBPOIN_MCPBE),
   &      'EP',EP%R(DBPOIN_MCPBE),'UETCAR',UETCAR%R(DBPOIN_MCPBE),
   &           'S0TA_MICFLC',S0TA%ADR(IMICFLC)%P%R(DBPOIN_MCPBE),
   &           'S0TA_MACFLC',S0TA%ADR(IMACFLC)%P%R(DBPOIN_MCPBE),
   &      'S0TA_MICF_MACF',S0TA%ADR(IMICF_MACF)%P%R(DBPOIN_MCPBE),
   &           'S1TA_MICFLC',S1TA%ADR(IMICFLC)%P%R(DBPOIN_MCPBE),
   &           'S1TA_MACFLC',S1TA%ADR(IMACFLC)%P%R(DBPOIN_MCPBE),
   &      'S1TA_MICF_MACF',S1TA%ADR(IMICF_MACF)%P%R(DBPOIN_MCPBE),
   &           'NC',NC_MAC%R(DBPOIN_MCPBE)
    ENDIF

    RETURN
    END
```

Appendix VI.
Subroutine floc_kinetics_3cpbe_var1.f

```
!              ***********************************
               SUBROUTINE FLOC_KINETICS_3CPBE_VAR1
!              ***********************************
!
!***********************************************************************
! TELEMAC3D
!***********************************************************************
!
!brief    COMPUTE FLOCCULATION KINETICS FOR MCPBE FLOCCULATION MODEL
!+           THIS SUBROUTINE IS ONLY USED FOR 3CPBE CASE.
!
!history  QILONG BI, BJ. LEE & X.SHEN
!+       09/05/19
!+       V8P1
!+
!
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!| NPOIN3         |-->| NUMBER OF POINTS IN 3D MESH
!| TA             |-->| TRACER (CONCENTRATIONS OF FLOCS)
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!
      USE BIEF
      USE DECLARATIONS_GAIA
      USE DECLARATIONS_TELEMAC3D, ONLY: NPOIN3,TA,S0TA,S1TA,DNUVIV,EP,
     &            FLOCMIC_DIA,FLOCMAC_DIA,FLOCMEG_DIA,RHO0,WCHU,
     &            NPOIN2,UETCAR,H,NPLAN,NPOIN2,NC_MAC,NC_MEG,
     &            LISPRD
      USE DECLARATIONS_SPECIAL
      IMPLICIT NONE

!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!
      INTEGER IPOIN,IPOIN2,IPLAN
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: CNUM_P,CNUM_F1,CNUM_T1,
     &                 CNUM_F2,CNUM_T2,TA_MEGFLC,NC1,NC2,SHR_G
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: BETA_PP,BETA_PF1,BETA_PF2,
     &                      BETA_F1F1,BETA_F1F2,BETA_F2F2
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: BRK_SH1,BRK_SH2
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: ABSS_P0, ABSS_P1,ABSS_F10,
     &             ABSS_F11,ABSS_T10,ABSS_T11,ABSS_T20,ABSS_T21
```

```
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: DIN,KL_CAP,BIOGR
      DOUBLE PRECISION :: FLOCMIC_VOL,WCHU_P,WCHU_F1,WCHU_F2,VMU
!
      DOUBLE PRECISION, PARAMETER :: BIOGR_MAX =  0.D0
      DOUBLE PRECISION, PARAMETER :: OMG1      =  0.0467D0
      DOUBLE PRECISION, PARAMETER :: GAMMA_F   =  0.6D0
      DOUBLE PRECISION, PARAMETER :: DIN_HS    =  0.D0
      DOUBLE PRECISION, PARAMETER :: CPF2      =  1.D0
      DOUBLE PRECISION, PARAMETER :: CF1F2     =  1.D0
      DOUBLE PRECISION, PARAMETER :: CF2F2     =  1.D0
!
!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

      IF(FLOC .AND. FLOC_TYPE.EQ.3 .AND. MCPBE_VER.EQ.2) THEN
!       SOURCE TERM FOR MICROFLOCS
        S0TA%ADR(IMICFLC)%P%TYPR='Q'
        S1TA%ADR(IMICFLC)%P%TYPR='Q'
        CALL OS('X=C     ',S0TA%ADR(IMICFLC)%P,C=0.D0)
        CALL OS('X=C     ',S1TA%ADR(IMICFLC)%P,C=0.D0)
!       SOURCE TERM FOR MACROFLOCS
        S0TA%ADR(IMACFLC)%P%TYPR='Q'
        S1TA%ADR(IMACFLC)%P%TYPR='Q'
        CALL OS('X=C     ',S0TA%ADR(IMACFLC)%P,C=0.D0)
        CALL OS('X=C     ',S1TA%ADR(IMACFLC)%P,C=0.D0)
!       SOURCE TERM FOR MICROFLOCS INSIDE MACROFLOCS
        S0TA%ADR(IMICF_MACF)%P%TYPR='Q'
        S1TA%ADR(IMICF_MACF)%P%TYPR='Q'
        CALL OS('X=C     ',S0TA%ADR(IMICF_MACF)%P,C=0.D0)
        CALL OS('X=C     ',S1TA%ADR(IMICF_MACF)%P,C=0.D0)
!       SOURCE TERM FOR MICROFLOCS INSIDE MEGAFLOCS
        S0TA%ADR(IMICF_MEGF)%P%TYPR='Q'
        S1TA%ADR(IMICF_MEGF)%P%TYPR='Q'
        CALL OS('X=C     ',S0TA%ADR(IMICF_MEGF)%P,C=0.D0)
        CALL OS('X=C     ',S1TA%ADR(IMICF_MEGF)%P,C=0.D0)
!       WRITE(LU,*) 'TIME STEP IN GAIA IS ',DT
        TA_MEGFLC = 0.D0
      ENDIF

!     COMPUTE FLOCCULATION KINETICS
      VMU = DNUVIV * RHO0
!
      DO IPLAN = 1,NPLAN
       DO IPOIN2 = 1,NPOIN2

        IPOIN=IPOIN2+(IPLAN-1)*NPOIN2
```

```
!     Modification for testing TCBPE kinetics only
!     SHR_G(IPOIN) = 7.31D0 !FOR VAL LEUSSEN'S SETTLING COLUMN EXPERIMENTS

!     Here, Please create a depth and shear velocity dependent "SHR_G"
      SHR_G(IPOIN) = SQRT(EP%R(IPOIN)/DNUVIV)
!     FOR THE BOTTOM NODE, EP COULD ALSO USE THEORETICAL VALUE (ML)
!     IF(IPOIN.LE.NPOIN2) THEN
!       SHR_G(IPOIN) = SQRT(UETCAR%R(IPOIN)**1.5/(KARMAN*0.001)
!    &                 *(1-0.001/MAX(H%R(IPOIN),0.001))/DNUVIV)
!     ENDIF
!     SHR_G(IPOIN) = MAX(SHR_G(IPOIN),1.D0)

      IF(BIOGR_MAX .NE. 0.D0) THEN
        KL_CAP(IPOIN) = GAMMA_F*(1+OMG1)*(1.D-6/SHR_G(IPOIN))**0.5D0
     &                  /NC_MEG%R(IPOIN)**(1.D0/FRACDIM_MEG)
        DIN(IPOIN) = 2.D-3
        BIOGR(IPOIN) = BIOGR_MAX*DIN(IPOIN)/(DIN(IPOIN)+DIN_HS)
!       DT IN GAIA CAN BE DIFFERENT FROM DT_TEL IF MORFAC IS APPLIED
        FLOCMIC_DIA%R(IPOIN) = FLOCMIC_DIA%R(IPOIN)
     &             +DT*BIOGR(IPOIN)*FLOCMIC_DIA%R(IPOIN)
     &             *(1.D0-FLOCMIC_DIA%R(IPOIN)/KL_CAP(IPOIN))
      ELSE
        FLOCMIC_DIA%R(IPOIN) = FLOCMIC_DIAFIX
      ENDIF

      FLOCMIC_VOL=(1.D0/6.D0)*PI*FLOCMIC_DIA%R(IPOIN)**3.D0

      CNUM_P(IPOIN) =
     &       TA%ADR(IMICFLC)%P%R(IPOIN)/(FLOCMIC_VOL*FLOCMIC_DEN)
      CNUM_F1(IPOIN) =
     &       TA%ADR(IMACFLC)%P%R(IPOIN)/(FLOCMIC_VOL*FLOCMIC_DEN)
      CNUM_T1(IPOIN) =
     &     TA%ADR(IMICF_MACF)%P%R(IPOIN)/(FLOCMIC_VOL*FLOCMIC_DEN)
      CNUM_T2(IPOIN) =
     &     TA%ADR(IMICF_MEGF)%P%R(IPOIN)/(FLOCMIC_VOL*FLOCMIC_DEN)

      NC1(IPOIN) =
     &      TA%ADR(IMICF_MACF)%P%R(IPOIN)/TA%ADR(IMACFLC)%P%R(IPOIN)
      IF(ISNAN(NC1(IPOIN)).OR.NC1(IPOIN).GT.HUGE(NC1(IPOIN))) THEN
        NC1(IPOIN)=NC_MAC%R(IPOIN)
      ENDIF
      NC_MAC%R(IPOIN) = MIN(MAX(NC1(IPOIN),2.D0),NC1_MAX)

!     FIXED 3RD CLASS
      NC2(IPOIN)=(FLOCMEG_DIAFIX/FLOCMIC_DIA%R(IPOIN))**FRACDIM_MEG
      IF(ISNAN(NC2(IPOIN)).OR.NC2(IPOIN).GT.HUGE(NC2(IPOIN))) THEN
```

```
        NC2(IPOIN)=NC_MEG%R(IPOIN)
      ENDIF
      NC_MEG%R(IPOIN) =
     &           MIN(MAX(NC2(IPOIN),2.D0*NC_MAC%R(IPOIN)),NC2_MAX)

      CNUM_F2(IPOIN)=CNUM_T2(IPOIN)/NC_MEG%R(IPOIN)
      TA_MEGFLC(IPOIN)=TA%ADR(IMICF_MEGF)%P%R(IPOIN)/NC_MEG%R(IPOIN)

      WCHU_P  = WCHU%ADR(IMICFLC)%P%R(IPOIN)
      WCHU_F1 = WCHU%ADR(IMACFLC)%P%R(IPOIN)
      WCHU_F2 = WCHU%ADR(IMICF_MEGF)%P%R(IPOIN)

!     COLLISION FREQUENCY
      BETA_PP(IPOIN) = (1.D0/6.D0)*(FLOCMIC_DIA%R(IPOIN)
     &             +FLOCMIC_DIA%R(IPOIN))**3.D0*SHR_G(IPOIN) ! TURBULENT SHEAR
     &             +2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)*4.D0      ! BROWIAN MOTION

      BETA_PF1(IPOIN) = (1.D0/6.D0)*(FLOCMIC_DIA%R(IPOIN)
     &        +FLOCMAC_DIA%R(IPOIN))**3.D0*SHR_G(IPOIN)     ! TURBULENT SHEAR
     &        +2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)              ! BROWIAN MOTION
     &        *(1.D0/FLOCMIC_DIA%R(IPOIN)+1.D0/FLOCMAC_DIA%R(IPOIN))
     &        *(FLOCMIC_DIA%R(IPOIN)+FLOCMAC_DIA%R(IPOIN))
     &        +PI/4.D0*(FLOCMIC_DIA%R(IPOIN)+FLOCMAC_DIA%R(IPOIN))
     &        **2.D0*ABS(WCHU_P-WCHU_F1)                 ! DIFFERENTIAL SETTLING

      BETA_PF2(IPOIN) = CPF2*((1.D0/6.D0)*(FLOCMIC_DIA%R(IPOIN)
     &        +FLOCMEG_DIA%R(IPOIN))**3.D0*SHR_G(IPOIN)
     &        +2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)
     &        *(1.D0/FLOCMIC_DIA%R(IPOIN)+1.D0/FLOCMEG_DIA%R(IPOIN))
     &        *(FLOCMIC_DIA%R(IPOIN)+FLOCMEG_DIA%R(IPOIN))
     &        +PI/4.D0*(FLOCMIC_DIA%R(IPOIN)+FLOCMEG_DIA%R(IPOIN))
     &        **2.D0*ABS(WCHU_P-WCHU_F2))

      BETA_F1F1(IPOIN) = SHR_G(IPOIN)*(1.D0/6.D0)
     &    *(FLOCMAC_DIA%R(IPOIN)+FLOCMAC_DIA%R(IPOIN))**3.D0
     &    +2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)*4.D0

      BETA_F1F2(IPOIN) = CF1F2*(SHR_G(IPOIN)*(1.D0/6.D0)
     &    *(FLOCMAC_DIA%R(IPOIN)+FLOCMEG_DIA%R(IPOIN))**3.D0
     &    +2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)
     &    *(1.D0/FLOCMAC_DIA%R(IPOIN)+1.D0/FLOCMEG_DIA%R(IPOIN))
     &    *(FLOCMAC_DIA%R(IPOIN)+FLOCMEG_DIA%R(IPOIN))
     &    +PI/4.D0*(FLOCMAC_DIA%R(IPOIN)+FLOCMEG_DIA%R(IPOIN))
     &    **2.D0*ABS(WCHU_F1-WCHU_F2))

      BETA_F2F2(IPOIN) = CF2F2*(SHR_G(IPOIN)*(1.D0/6.D0)
```

```
     &       *(FLOCMEG_DIA%R(IPOIN)+FLOCMEG_DIA%R(IPOIN))**3.D0
     &       +2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)*4.D0)

!      BREAKAGE FREQUENCY
!      Modification Proposed by Kyle Strom (2018)
       BRK_SH1(IPOIN) = BRK_EFF*SHR_G(IPOIN)*((FLOCMAC_DIA%R(IPOIN)
     &              -FLOCMIC_DIA%R(IPOIN))/FLOCMIC_DIA%R(IPOIN))
     &              **(3.D0-FRACDIM_MAC)*(VMU*SHR_G(IPOIN)
     &              *FLOCMAC_DIA%R(IPOIN)**2.D0/BRK_FY)
     &              **(BRK_Q+BRK_QC*FLOCMAC_DIA%R(IPOIN)
     &              /(DNUVIV**3.D0/EP%R(IPOIN))**0.25D0)

       BRK_SH2(IPOIN) = BRK_EFF*SHR_G(IPOIN)*((FLOCMEG_DIA%R(IPOIN)
     &              -FLOCMIC_DIA%R(IPOIN))/FLOCMIC_DIA%R(IPOIN))
     &              **(3.D0-FRACDIM_MEG)*(VMU*SHR_G(IPOIN)
     &              *FLOCMEG_DIA%R(IPOIN)**2.D0/BRK_FY)
     &              **(BRK_Q+BRK_QC*FLOCMEG_DIA%R(IPOIN)
     &              /(DNUVIV**3.D0/EP%R(IPOIN))**0.25D0)

!      CALCULATE THE AGGREGATION AND BREAKAGE KERNELS
       ABSS_P0(IPOIN) =
     &       BRKFRAC_P1*NC_MAC%R(IPOIN)*BRK_SH1(IPOIN)*CNUM_F1(IPOIN)
     &      +BRKFRAC_P2*NC_MEG%R(IPOIN)*BRK_SH2(IPOIN)*CNUM_F2(IPOIN)

       ABSS_P1(IPOIN) = 0.5D0*AGG_ALPHA*BETA_PP(IPOIN)
     &      *CNUM_P(IPOIN)*(NC_MAC%R(IPOIN)/(NC_MAC%R(IPOIN)-1.D0))
     &      +AGG_ALPHA*BETA_PF1(IPOIN)*CNUM_F1(IPOIN)
     &      +AGG_ALPHA*BETA_PF2(IPOIN)*CNUM_F2(IPOIN)

       ABSS_F10(IPOIN) = 0.5D0*AGG_ALPHA*BETA_PP(IPOIN)
     &       *CNUM_P(IPOIN)*CNUM_P(IPOIN)/(NC_MAC%R(IPOIN)-1.D0)
     &       +(BRK_K1-1.D0)*BRK_SH1(IPOIN)*CNUM_F1(IPOIN)
     &       +BRK_K2*BRK_SH2(IPOIN)*CNUM_F2(IPOIN)

       ABSS_F11(IPOIN) = 0.5D0*AGG_ALPHA*BETA_F1F1(IPOIN)
     &       *CNUM_F1(IPOIN)*(NC_MEG%R(IPOIN)/NC_MAC%R(IPOIN))
     &       /(NC_MEG%R(IPOIN)/NC_MAC%R(IPOIN)-1.D0)
     &       +AGG_ALPHA*BETA_F1F2(IPOIN)*CNUM_F2(IPOIN)

       ABSS_T10(IPOIN) = 0.5D0*AGG_ALPHA*BETA_PP(IPOIN)
     &      *(CNUM_P(IPOIN)*CNUM_P(IPOIN))
     &      *NC_MAC%R(IPOIN)/(NC_MAC%R(IPOIN)-1.D0)
     &      +AGG_ALPHA*BETA_PF1(IPOIN)*CNUM_P(IPOIN)*CNUM_F1(IPOIN)
     &      +(1.D0-BRKFRAC_P2-BRKFRAC_F2)*NC_MEG%R(IPOIN)
     &      *BRK_SH2(IPOIN)*CNUM_F2(IPOIN)
```

```
      ABSS_T11(IPOIN) = (0.5D0*AGG_ALPHA*BETA_F1F1(IPOIN)
     &     *CNUM_F1(IPOIN)*TA%ADR(IMACFLC)%P%R(IPOIN)
     &     *(NC_MEG%R(IPOIN)/(NC_MEG%R(IPOIN)/NC_MAC%R(IPOIN)-1.D0))
     &     +NC_MAC%R(IPOIN)*AGG_ALPHA*BETA_F1F2(IPOIN)
     &     *TA%ADR(IMACFLC)%P%R(IPOIN)*CNUM_F2(IPOIN)
     &     +BRKFRAC_P1*NC_MAC%R(IPOIN)*BRK_SH1(IPOIN)
     &     *TA%ADR(IMACFLC)%P%R(IPOIN))
     &     /MAX(TA%ADR(IMICF_MACF)%P%R(IPOIN),1.D-15)

      ABSS_T20(IPOIN) =
     &     AGG_ALPHA*BETA_PF2(IPOIN)*CNUM_P(IPOIN)*CNUM_F2(IPOIN)
     &     +0.5D0*AGG_ALPHA*BETA_F1F1(IPOIN)*CNUM_F1(IPOIN)
     &     *CNUM_F1(IPOIN)*(NC_MEG%R(IPOIN)
     &     /(NC_MEG%R(IPOIN)/NC_MAC%R(IPOIN)-1.D0))
     &     +NC_MAC%R(IPOIN)*AGG_ALPHA*BETA_F1F2(IPOIN)
     &     *CNUM_F1(IPOIN)*CNUM_F2(IPOIN)

      ABSS_T21(IPOIN) = (1.D0-BRKFRAC_F2) !BRKFRAC_F2=0
     &     *NC_MEG%R(IPOIN)*BRK_SH2(IPOIN)*TA_MEGFLC(IPOIN)
     &     /MAX(TA%ADR(IMICF_MEGF)%P%R(IPOIN),1.D-15)

!     ASSEMBLE THE SINK AND SOURCE TERMS
      S0TA%ADR(IMICFLC)%P%R(IPOIN) =
     &                 ABSS_P0(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN
      S0TA%ADR(IMACFLC)%P%R(IPOIN) =
     &                 ABSS_F10(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN
      S0TA%ADR(IMICF_MACF)%P%R(IPOIN) =
     &                 ABSS_T10(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN
      S0TA%ADR(IMICF_MEGF)%P%R(IPOIN) =
     &                 ABSS_T20(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN

      S1TA%ADR(IMICFLC)%P%R(IPOIN) =    ABSS_P1 (IPOIN)
      S1TA%ADR(IMACFLC)%P%R(IPOIN) =    ABSS_F11(IPOIN)
      S1TA%ADR(IMICF_MACF)%P%R(IPOIN) = ABSS_T11(IPOIN)
      S1TA%ADR(IMICF_MEGF)%P%R(IPOIN) = ABSS_T21(IPOIN)

!     CONSERVATION OF MASS
!     S0TA%ADR(IMICFLC)%P%R(IPOIN) = MAX(
!    &                 ABSS_P(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN,
!    &                 -TA%ADR(IMICFLC)%P%R(IPOIN)/DT)
!     S0TA%ADR(IMACFLC)%P%R(IPOIN) = MAX(
!    &                 ABSS_F1(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN,
!    &                 -TA%ADR(IMACFLC)%P%R(IPOIN)/DT)
!     S0TA%ADR(IMICF_MACF)%P%R(IPOIN) = MAX(
!    &                 ABSS_T1(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN,
!    &                 -TA%ADR(IMICF_MACF)%P%R(IPOIN)/DT)
```

```
!     S0TA%ADR(IMICF_MEGF)%P%R(IPOIN) = MAX(
!    &                 ABSS_T2(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN,
!    &                 -TA%ADR(IMICF_MEGF)%P%R(IPOIN)/DT)

      ENDDO
     ENDDO

     IF(DBPOIN_MCPBE.GT.0.AND.MOD(LT,LISPRD).EQ.0) THEN
       WRITE(LU,*) 'H',H%R(DBPOIN_MCPBE),'SHR_G',SHR_G(DBPOIN_MCPBE),
    &      'EP',EP%R(DBPOIN_MCPBE),'UETCAR',UETCAR%R(DBPOIN_MCPBE),
    &          'S0TA_MICFLC',S0TA%ADR(IMICFLC)%P%R(DBPOIN_MCPBE),
    &          'S0TA_MACFLC',S0TA%ADR(IMACFLC)%P%R(DBPOIN_MCPBE),
    &      'S0TA_MICF_MACF',S0TA%ADR(IMICF_MACF)%P%R(DBPOIN_MCPBE),
    &      'S0TA_MICF_MEGF',S0TA%ADR(IMICF_MEGF)%P%R(DBPOIN_MCPBE),
    &          'S1TA_MICFLC',S1TA%ADR(IMICFLC)%P%R(DBPOIN_MCPBE),
    &          'S1TA_MACFLC',S1TA%ADR(IMACFLC)%P%R(DBPOIN_MCPBE),
    &      'S1TA_MICF_MACF',S1TA%ADR(IMICF_MACF)%P%R(DBPOIN_MCPBE),
    &      'S1TA_MICF_MEGF',S1TA%ADR(IMICF_MEGF)%P%R(DBPOIN_MCPBE),
    &                      'NC_MAC',NC_MAC%R(DBPOIN_MCPBE),
    &                      'NC_MEG',NC_MEG%R(DBPOIN_MCPBE)
       WRITE(LU,*) 'BETA_PF2',BETA_PF2(DBPOIN_MCPBE),
    &          'BETA_F1F1',BETA_F1F1(DBPOIN_MCPBE),
    &          'BETA_F1F2',BETA_F1F2(DBPOIN_MCPBE),
    &          'S0TA_MICF_MEGF term 1',
    &            AGG_ALPHA*BETA_PF2(DBPOIN_MCPBE)
    &            *CNUM_P(DBPOIN_MCPBE)*CNUM_F2(DBPOIN_MCPBE),
    &          'S0TA_MICF_MEGF term 2',
    &             0.5D0*AGG_ALPHA*BETA_F1F1(DBPOIN_MCPBE)
    &            *CNUM_F1(DBPOIN_MCPBE)*CNUM_F1(DBPOIN_MCPBE)
    &            *NC_MEG%R(DBPOIN_MCPBE)/(NC_MEG%R(DBPOIN_MCPBE)
    &            /NC_MAC%R(DBPOIN_MCPBE)-1.D0),
    &          'S0TA_MICF_MEGF term 3',
    &             NC_MAC%R(DBPOIN_MCPBE)*AGG_ALPHA
    &            *BETA_F1F2(DBPOIN_MCPBE)
    &            *CNUM_F1(DBPOIN_MCPBE)*CNUM_F2(DBPOIN_MCPBE)
       WRITE(LU,*) 'BRK_SH1',BRK_SH1(DBPOIN_MCPBE),
    &          'BRK_SH2',BRK_SH2(DBPOIN_MCPBE)

     ENDIF

     RETURN
     END
```

Appendix VII. Subroutine floc_kinetics_3cpbe_var2.f

```
!                 **********************************
                  SUBROUTINE FLOC_KINETICS_3CPBE_VAR2
!                 **********************************
!
!*********************************************************************
! TELEMAC3D
!*********************************************************************
!
!brief    COMPUTE FLOCCULATION KINETICS FOR MCPBE FLOCCULATION MODEL
!+             THIS SUBROUTINE IS ONLY USED FOR 3CPBE CASE.
!
!history  QILONG BI, BJ. LEE & X.SHEN
!+        09/05/19
!+        V8P1
!+
!
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!| NPOIN3         |-->| NUMBER OF POINTS IN 3D MESH
!| TA             |-->| TRACER (CONCENTRATIONS OF FLOCS)
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!
      USE BIEF
      USE DECLARATIONS_GAIA
      USE DECLARATIONS_TELEMAC3D, ONLY: NPOIN3,TA,S0TA,S1TA,DNUVIV,EP,
     &                FLOCMIC_DIA,FLOCMAC_DIA,FLOCMEG_DIA,RHO0,WCHU,
     &                NPOIN2,UETCAR,H,NPLAN,NPOIN2,NC_MAC,NC_MEG,
     &                LISPRD
      USE DECLARATIONS_SPECIAL
      IMPLICIT NONE

!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!
      INTEGER IPOIN,IPOIN2,IPLAN
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: CNUM_P,CNUM_F1,CNUM_T1,
     &                              CNUM_F2,CNUM_T2,NC1,NC2,SHR_G
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: BETA_PP,BETA_PF1,BETA_PF2,
     &                              BETA_F1F1,BETA_F1F2,BETA_F2F2
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: BRK_SH1,BRK_SH2
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: ABSS_P0, ABSS_P1,ABSS_F10,
     &                         ABSS_F11,ABSS_T10,ABSS_T11,ABSS_F20,
```

```
   &                        ABSS_F21,ABSS_T20,ABSS_T21
      DOUBLE PRECISION, DIMENSION(NPOIN3) :: DIN,KL_CAP,BIOGR
      DOUBLE PRECISION :: FLOCMIC_VOL,WCHU_P,WCHU_F1,WCHU_F2,VMU
!
      DOUBLE PRECISION, PARAMETER :: BIOGR_MAX =  0.D0
      DOUBLE PRECISION, PARAMETER :: OMG1      =  0.0467D0
      DOUBLE PRECISION, PARAMETER :: GAMMA_F   =  0.6D0
      DOUBLE PRECISION, PARAMETER :: DIN_HS    =  0.D0
      DOUBLE PRECISION, PARAMETER :: CPF2      =  1.D0
      DOUBLE PRECISION, PARAMETER :: CF1F2     =  1.D0
      DOUBLE PRECISION, PARAMETER :: CF2F2     =  1.D0
!
!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

      IF(FLOC .AND. FLOC_TYPE.EQ.3 .AND. MCPBE_VER.EQ.3) THEN
!       SOURCE TERM FOR MICROFLOCS
        S0TA%ADR(IMICFLC)%P%TYPR='Q'
        S1TA%ADR(IMICFLC)%P%TYPR='Q'
        CALL OS('X=C     ',S0TA%ADR(IMICFLC)%P,C=0.D0)
        CALL OS('X=C     ',S1TA%ADR(IMICFLC)%P,C=0.D0)
!       SOURCE TERM FOR MACROFLOCS
        S0TA%ADR(IMACFLC)%P%TYPR='Q'
        S1TA%ADR(IMACFLC)%P%TYPR='Q'
        CALL OS('X=C     ',S0TA%ADR(IMACFLC)%P,C=0.D0)
        CALL OS('X=C     ',S1TA%ADR(IMACFLC)%P,C=0.D0)
!       SOURCE TERM FOR MEGAFLOCS
        S0TA%ADR(IMEGFLC)%P%TYPR='Q'
        S1TA%ADR(IMEGFLC)%P%TYPR='Q'
        CALL OS('X=C     ',S0TA%ADR(IMEGFLC)%P,C=0.D0)
        CALL OS('X=C     ',S1TA%ADR(IMEGFLC)%P,C=0.D0)
!       SOURCE TERM FOR MICROFLOCS INSIDE MACROFLOCS
        S0TA%ADR(IMICF_MACF)%P%TYPR='Q'
        S1TA%ADR(IMICF_MACF)%P%TYPR='Q'
        CALL OS('X=C     ',S0TA%ADR(IMICF_MACF)%P,C=0.D0)
        CALL OS('X=C     ',S1TA%ADR(IMICF_MACF)%P,C=0.D0)
!       SOURCE TERM FOR MICROFLOCS INSIDE MEGAFLOCS
        S0TA%ADR(IMICF_MEGF)%P%TYPR='Q'
        S1TA%ADR(IMICF_MEGF)%P%TYPR='Q'
        CALL OS('X=C     ',S0TA%ADR(IMICF_MEGF)%P,C=0.D0)
        CALL OS('X=C     ',S1TA%ADR(IMICF_MEGF)%P,C=0.D0)
!       WRITE(LU,*) 'TIME STEP IN GAIA IS ',DT
      ENDIF

!     COMPUTE FLOCCULATION KINETICS
      VMU = DNUVIV * RHO0
!
```

```
      DO IPLAN = 1,NPLAN
       DO IPOIN2 = 1,NPOIN2

        IPOIN=IPOIN2+(IPLAN-1)*NPOIN2

!       Modification for testing TCBPE kinetics only
!       SHR_G(IPOIN) = 7.31D0 !FOR VAL LEUSSEN'S SETTLING COLUMN EXPERIMENTS

!       Here, Please create a depth and shear velocity dependent "SHR_G"
        SHR_G(IPOIN) = SQRT(EP%R(IPOIN)/DNUVIV)
!       FOR THE BOTTOM NODE, EP COULD ALSO USE THEORETICAL VALUE (ML)
!       IF(IPOIN.LE.NPOIN2) THEN
!        SHR_G(IPOIN) = SQRT(UETCAR%R(IPOIN)**1.5/(KARMAN*0.001)
!   &                  *(1-0.001/MAX(H%R(IPOIN),0.001))/DNUVIV)
!       ENDIF
!       SHR_G(IPOIN) = MAX(SHR_G(IPOIN),1.D0)

        IF(BIOGR_MAX .NE. 0.D0) THEN
         KL_CAP(IPOIN) = GAMMA_F*(1+OMG1)*(1.D-6/SHR_G(IPOIN))**0.5D0
     &                   /NC_MEG%R(IPOIN)**(1.D0/FRACDIM_MEG)
         DIN(IPOIN) = 2.D-3
         BIOGR(IPOIN) = BIOGR_MAX*DIN(IPOIN)/(DIN(IPOIN)+DIN_HS)
!        DT IN GAIA CAN BE DIFFERENT FROM DT_TEL IF MORFAC IS APPLIED
         FLOCMIC_DIA%R(IPOIN) = FLOCMIC_DIA%R(IPOIN)
     &          +DT*BIOGR(IPOIN)*FLOCMIC_DIA%R(IPOIN)
     &          *(1.D0-FLOCMIC_DIA%R(IPOIN)/KL_CAP(IPOIN))
        ELSE
         FLOCMIC_DIA%R(IPOIN) = FLOCMIC_DIAFIX
        ENDIF

        FLOCMIC_VOL=(1.D0/6.D0)*PI*FLOCMIC_DIA%R(IPOIN)**3.D0

        CNUM_P(IPOIN) =
     &    TA%ADR(IMICFLC)%P%R(IPOIN)/(FLOCMIC_VOL*FLOCMIC_DEN)
        CNUM_F1(IPOIN) =
     &    TA%ADR(IMACFLC)%P%R(IPOIN)/(FLOCMIC_VOL*FLOCMIC_DEN)
        CNUM_T1(IPOIN) =
     &    TA%ADR(IMICF_MACF)%P%R(IPOIN)/(FLOCMIC_VOL*FLOCMIC_DEN)
        CNUM_T2(IPOIN) =
     &    TA%ADR(IMICF_MEGF)%P%R(IPOIN)/(FLOCMIC_VOL*FLOCMIC_DEN)

        NC1(IPOIN) =
     &    TA%ADR(IMICF_MACF)%P%R(IPOIN)/TA%ADR(IMACFLC)%P%R(IPOIN)
        IF(ISNAN(NC1(IPOIN)).OR.NC1(IPOIN).GT.HUGE(NC1(IPOIN))) THEN
         NC1(IPOIN)=NC_MAC%R(IPOIN)
        ENDIF
```

```
      NC_MAC%R(IPOIN) = MIN(MAX(NC1(IPOIN),2.D0),NC1_MAX)

!     VARYING 3RD CLASS
      CNUM_F2(IPOIN) =
     &          TA%ADR(IMEGFLC)%P%R(IPOIN)/(FLOCMIC_VOL*FLOCMIC_DEN)
      NC2(IPOIN) =
     &       TA%ADR(IMICF_MEGF)%P%R(IPOIN)/TA%ADR(IMEGFLC)%P%R(IPOIN)
      IF(ISNAN(NC2(IPOIN)).OR.NC2(IPOIN).GT.HUGE(NC2(IPOIN))) THEN
        NC2(IPOIN)=NC_MEG%R(IPOIN)
      ENDIF
      NC_MEG%R(IPOIN) =
     &           MIN(MAX(NC2(IPOIN),2.D0*NC_MAC%R(IPOIN)),NC2_MAX)

      WCHU_P  = WCHU%ADR(IMICFLC)%P%R(IPOIN)
      WCHU_F1 = WCHU%ADR(IMACFLC)%P%R(IPOIN)
      WCHU_F2 = WCHU%ADR(IMEGFLC)%P%R(IPOIN)

!     COLLISION FREQUENCY
      BETA_PP(IPOIN) = (1.D0/6.D0)*(FLOCMIC_DIA%R(IPOIN)
     &              +FLOCMIC_DIA%R(IPOIN))**3.D0*SHR_G(IPOIN) ! TURBULENT SHEAR
     &              +2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)*4.D0       ! BROWIAN MOTION

      BETA_PF1(IPOIN) = (1.D0/6.D0)*(FLOCMIC_DIA%R(IPOIN)
     &         +FLOCMAC_DIA%R(IPOIN))**3.D0*SHR_G(IPOIN)     ! TURBULENT SHEAR
     &         +2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)               ! BROWIAN MOTION
     &         *(1.D0/FLOCMIC_DIA%R(IPOIN)+1.D0/FLOCMAC_DIA%R(IPOIN))
     &         *(FLOCMIC_DIA%R(IPOIN)+FLOCMAC_DIA%R(IPOIN))
     &         +PI/4.D0*(FLOCMIC_DIA%R(IPOIN)+FLOCMAC_DIA%R(IPOIN))
     &         **2.D0*ABS(WCHU_P-WCHU_F1)                   ! DIFFERENTIAL SETTLING

      BETA_PF2(IPOIN) = CPF2*((1.D0/6.D0)*(FLOCMIC_DIA%R(IPOIN)
     &         +FLOCMEG_DIA%R(IPOIN))**3.D0*SHR_G(IPOIN)
     &         +2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)
     &         *(1.D0/FLOCMIC_DIA%R(IPOIN)+1.D0/FLOCMEG_DIA%R(IPOIN))
     &         *(FLOCMIC_DIA%R(IPOIN)+FLOCMEG_DIA%R(IPOIN))
     &         +PI/4.D0*(FLOCMIC_DIA%R(IPOIN)+FLOCMEG_DIA%R(IPOIN))
     &         **2.D0*ABS(WCHU_P-WCHU_F2))

      BETA_F1F1(IPOIN) = SHR_G(IPOIN)*(1.D0/6.D0)
     &     *(FLOCMAC_DIA%R(IPOIN)+FLOCMAC_DIA%R(IPOIN))**3.D0
     &     +2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)*4.D0

      BETA_F1F2(IPOIN) = CF1F2*(SHR_G(IPOIN)*(1.D0/6.D0)
     &     *(FLOCMAC_DIA%R(IPOIN)+FLOCMEG_DIA%R(IPOIN))**3.D0
     &     +2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)
     &     *(1.D0/FLOCMAC_DIA%R(IPOIN)+1.D0/FLOCMEG_DIA%R(IPOIN))
```

```
     &     *(FLOCMAC_DIA%R(IPOIN)+FLOCMEG_DIA%R(IPOIN))
     &     +PI/4.D0*(FLOCMAC_DIA%R(IPOIN)+FLOCMEG_DIA%R(IPOIN))
     &     **2.D0*ABS(WCHU_F1-WCHU_F2))

          BETA_F2F2(IPOIN) = CF2F2*(SHR_G(IPOIN)*(1.D0/6.D0)
     &       *(FLOCMEG_DIA%R(IPOIN)+FLOCMEG_DIA%R(IPOIN))**3.D0
     &       +2.D0*KBOLZ*TEMPSTD/(3.D0*VMU)*4.D0)

!     BREAKAGE FREQUENCY
!     Modification Proposed by Kyle Strom (2018)
          BRK_SH1(IPOIN) = BRK_EFF*SHR_G(IPOIN)*((FLOCMAC_DIA%R(IPOIN)
     &               -FLOCMIC_DIA%R(IPOIN))/FLOCMIC_DIA%R(IPOIN))
     &               **(3.D0-FRACDIM_MAC)*(VMU*SHR_G(IPOIN)
     &               *FLOCMAC_DIA%R(IPOIN)**2.D0/BRK_FY)
     &               **(BRK_Q+BRK_QC*FLOCMAC_DIA%R(IPOIN)
     &               /(DNUVIV**3.D0/EP%R(IPOIN))**0.25D0)

          BRK_SH2(IPOIN) = BRK_EFF*SHR_G(IPOIN)*((FLOCMEG_DIA%R(IPOIN)
     &               -FLOCMIC_DIA%R(IPOIN))/FLOCMIC_DIA%R(IPOIN))
     &               **(3.D0-FRACDIM_MEG)*(VMU*SHR_G(IPOIN)
     &               *FLOCMEG_DIA%R(IPOIN)**2.D0/BRK_FY)
     &               **(BRK_Q+BRK_QC*FLOCMEG_DIA%R(IPOIN)
     &               /(DNUVIV**3.D0/EP%R(IPOIN))**0.25D0)

!     CALCULATE THE AGGREGATION AND BREAKAGE KERNELS
          ABSS_P0(IPOIN) =
     &       BRKFRAC_P1*NC_MAC%R(IPOIN)*BRK_SH1(IPOIN)*CNUM_F1(IPOIN)
     &      +BRKFRAC_P2*NC_MEG%R(IPOIN)*BRK_SH2(IPOIN)*CNUM_F2(IPOIN)

          ABSS_P1(IPOIN) = 0.5D0*AGG_ALPHA*BETA_PP(IPOIN)
     &      *CNUM_P(IPOIN)*(NC_MAC%R(IPOIN)/(NC_MAC%R(IPOIN)-1.D0))
     &      +AGG_ALPHA*BETA_PF1(IPOIN)*CNUM_F1(IPOIN)
     &      +AGG_ALPHA*BETA_PF2(IPOIN)*CNUM_F2(IPOIN)

          ABSS_F10(IPOIN) = 0.5D0*AGG_ALPHA*BETA_PP(IPOIN)
     &       *CNUM_P(IPOIN)*CNUM_P(IPOIN)/(NC_MAC%R(IPOIN)-1.D0)
     &       +(BRK_K1-1.D0)*BRK_SH1(IPOIN)*CNUM_F1(IPOIN)
     &       +BRK_K2*BRK_SH2(IPOIN)*CNUM_F2(IPOIN)

          ABSS_F11(IPOIN) = 0.5D0*AGG_ALPHA*BETA_F1F1(IPOIN)
     &       *CNUM_F1(IPOIN)*(NC_MEG%R(IPOIN)/NC_MAC%R(IPOIN))
     &       /(NC_MEG%R(IPOIN)/NC_MAC%R(IPOIN)-1.D0)
     &       +AGG_ALPHA*BETA_F1F2(IPOIN)*CNUM_F2(IPOIN)

          ABSS_T10(IPOIN) = 0.5D0*AGG_ALPHA*BETA_PP(IPOIN)
     &      *(CNUM_P(IPOIN)*CNUM_P(IPOIN))
```

```
&     *NC_MAC%R(IPOIN)/(NC_MAC%R(IPOIN)-1.D0)
&     +AGG_ALPHA*BETA_PF1(IPOIN)*CNUM_P(IPOIN)*CNUM_F1(IPOIN)
&     +(1.D0-BRKFRAC_P2-BRKFRAC_F2)*NC_MEG%R(IPOIN)
&     *BRK_SH2(IPOIN)*CNUM_F2(IPOIN)

  ABSS_T11(IPOIN) = (0.5D0*AGG_ALPHA*BETA_F1F1(IPOIN)
&     *CNUM_F1(IPOIN)*TA%ADR(IMACFLC)%P%R(IPOIN)
&     *(NC_MEG%R(IPOIN)/(NC_MEG%R(IPOIN)/NC_MAC%R(IPOIN)-1.D0))
&     +NC_MAC%R(IPOIN)*AGG_ALPHA*BETA_F1F2(IPOIN)
&     *TA%ADR(IMACFLC)%P%R(IPOIN)*CNUM_F2(IPOIN)
&     +BRKFRAC_P1*NC_MAC%R(IPOIN)*BRK_SH1(IPOIN)
&     *TA%ADR(IMACFLC)%P%R(IPOIN))
&     /MAX(TA%ADR(IMICF_MACF)%P%R(IPOIN),1.D-15)

  IF(BRK_K3.LT.1.D0) THEN
    ABSS_F20(IPOIN) = 0.5D0*AGG_ALPHA*BETA_F1F1(IPOIN)
&               *CNUM_F1(IPOIN)*CNUM_F1(IPOIN)
&               /(NC_MEG%R(IPOIN)/NC_MAC%R(IPOIN)-1.D0)

    ABSS_F21(IPOIN) = 0.5D0*AGG_ALPHA*BETA_F2F2(IPOIN)
&               *CNUM_F2(IPOIN)
&               +(1.D0-BRK_K3)*BRK_SH2(IPOIN)
  ELSE
    ABSS_F20(IPOIN) = 0.5D0*AGG_ALPHA*BETA_F1F1(IPOIN)
&               *CNUM_F1(IPOIN)*CNUM_F1(IPOIN)
&               /(NC_MEG%R(IPOIN)/NC_MAC%R(IPOIN)-1.D0)
&               +(BRK_K3-1.D0)*BRK_SH2(IPOIN)*CNUM_F2(IPOIN)

    ABSS_F21(IPOIN) = 0.5D0*AGG_ALPHA*BETA_F2F2(IPOIN)
&               *CNUM_F2(IPOIN)
  ENDIF

  ABSS_T20(IPOIN) = AGG_ALPHA*BETA_PF2(IPOIN)
&             *CNUM_P(IPOIN)*CNUM_F2(IPOIN)
&             +0.5D0*AGG_ALPHA*BETA_F1F1(IPOIN)
&             *CNUM_F1(IPOIN)*CNUM_F1(IPOIN)
&             *NC_MEG%R(IPOIN)
&             /(NC_MEG%R(IPOIN)/NC_MAC%R(IPOIN)-1.D0)
&             +NC_MAC%R(IPOIN)*AGG_ALPHA*BETA_F1F2(IPOIN)
&             *CNUM_F1(IPOIN)*CNUM_F2(IPOIN)

  ABSS_T21(IPOIN) = (1.D0-BRKFRAC_F2)*NC_MEG%R(IPOIN) !BRKFRAC_F2=0
&             *BRK_SH2(IPOIN)*TA%ADR(IMEGFLC)%P%R(IPOIN)
&             /MAX(TA%ADR(IMICF_MEGF)%P%R(IPOIN),1.D-15)
```

```
      S0TA%ADR(IMICFLC)%P%R(IPOIN) =
     &                 ABSS_P0(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN
      S0TA%ADR(IMACFLC)%P%R(IPOIN) =
     &                 ABSS_F10(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN
      S0TA%ADR(IMICF_MACF)%P%R(IPOIN) =
     &                 ABSS_T10(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN
      S0TA%ADR(IMEGFLC)%P%R(IPOIN) =
     &                 ABSS_F20(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN
      S0TA%ADR(IMICF_MEGF)%P%R(IPOIN) =
     &                 ABSS_T20(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN

      S1TA%ADR(IMICFLC)%P%R(IPOIN) =    ABSS_P1 (IPOIN)
      S1TA%ADR(IMACFLC)%P%R(IPOIN) =    ABSS_F11(IPOIN)
      S1TA%ADR(IMICF_MACF)%P%R(IPOIN) = ABSS_T11(IPOIN)
      S1TA%ADR(IMEGFLC)%P%R(IPOIN) =    ABSS_F21(IPOIN)
      S1TA%ADR(IMICF_MEGF)%P%R(IPOIN) = ABSS_T21(IPOIN)

!     CONSERVATION OF MASS
!     S0TA%ADR(IMICFLC)%P%R(IPOIN) = MAX(
!    &                 ABSS_P(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN,
!    &                 -TA%ADR(IMICFLC)%P%R(IPOIN)/DT)
!     S0TA%ADR(IMACFLC)%P%R(IPOIN) = MAX(
!    &                 ABSS_F1(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN,
!    &                 -TA%ADR(IMACFLC)%P%R(IPOIN)/DT)
!     S0TA%ADR(IMICF_MACF)%P%R(IPOIN) = MAX(
!    &                 ABSS_T1(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN,
!    &                 -TA%ADR(IMICF_MACF)%P%R(IPOIN)/DT)
!     S0TA%ADR(IMEGFLC)%P%R(IPOIN) = MAX(
!    &                 ABSS_F2(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN,
!    &                 -TA%ADR(IMEGFLC)%P%R(IPOIN)/DT)
!     S0TA%ADR(IMICF_MEGF)%P%R(IPOIN) = MAX(
!    &                 ABSS_T2(IPOIN)*FLOCMIC_VOL*FLOCMIC_DEN,
!    &                 -TA%ADR(IMICF_MEGF)%P%R(IPOIN)/DT)

      ENDDO
     ENDDO

     IF(DBPOIN_MCPBE.GT.0.AND.MOD(LT,LISPRD).EQ.0) THEN
      WRITE(LU,*) 'H',H%R(DBPOIN_MCPBE),'SHR_G',SHR_G(DBPOIN_MCPBE),
     &     'EP',EP%R(DBPOIN_MCPBE),'UETCAR',UETCAR%R(DBPOIN_MCPBE),
     &         'S0TA_MICFLC',S0TA%ADR(IMICFLC)%P%R(DBPOIN_MCPBE),
     &         'S0TA_MACFLC',S0TA%ADR(IMACFLC)%P%R(DBPOIN_MCPBE),
     &     'S0TA_MICF_MACF',S0TA%ADR(IMICF_MACF)%P%R(DBPOIN_MCPBE),
     &         'S0TA_MEGFLC',S0TA%ADR(IMEGFLC)%P%R(DBPOIN_MCPBE),
     &     'S0TA_MICF_MEGF',S0TA%ADR(IMICF_MEGF)%P%R(DBPOIN_MCPBE),
     &         'S1TA_MICFLC',S1TA%ADR(IMICFLC)%P%R(DBPOIN_MCPBE),
```

```
&           'S1TA_MACFLC',S1TA%ADR(IMACFLC)%P%R(DBPOIN_MCPBE),
&      'S1TA_MICF_MACF',S1TA%ADR(IMICF_MACF)%P%R(DBPOIN_MCPBE),
&           'S1TA_MEGFLC',S1TA%ADR(IMEGFLC)%P%R(DBPOIN_MCPBE),
&      'S1TA_MICF_MEGF',S1TA%ADR(IMICF_MEGF)%P%R(DBPOIN_MCPBE),
&                       'NC_MAC',NC_MAC%R(DBPOIN_MCPBE),
&                       'NC_MEG',NC_MEG%R(DBPOIN_MCPBE)
  WRITE(LU,*) 'BETA_PF2',BETA_PF2(DBPOIN_MCPBE),
&           'BETA_F1F1',BETA_F1F1(DBPOIN_MCPBE),
&           'BETA_F1F2',BETA_F1F2(DBPOIN_MCPBE),
&           'S0TA_MICF_MEGF term 1',
&             AGG_ALPHA*BETA_PF2(DBPOIN_MCPBE)
&             *CNUM_P(DBPOIN_MCPBE)*CNUM_F2(DBPOIN_MCPBE),
&           'S0TA_MICF_MEGF term 2',
&              0.5D0*AGG_ALPHA*BETA_F1F1(DBPOIN_MCPBE)
&             *CNUM_F1(DBPOIN_MCPBE)*CNUM_F1(DBPOIN_MCPBE)
&             *NC_MEG%R(DBPOIN_MCPBE)/(NC_MEG%R(DBPOIN_MCPBE)
&             /NC_MAC%R(DBPOIN_MCPBE)-1.D0),
&           'S0TA_MICF_MEGF term 3',
&              NC_MAC%R(DBPOIN_MCPBE)*AGG_ALPHA
&             *BETA_F1F2(DBPOIN_MCPBE)
&             *CNUM_F1(DBPOIN_MCPBE)*CNUM_F2(DBPOIN_MCPBE)
  WRITE(LU,*) 'BRK_SH1',BRK_SH1(DBPOIN_MCPBE),
&           'BRK_SH2',BRK_SH2(DBPOIN_MCPBE)

  ENDIF

  RETURN
  END
```

Appendix VIII. Subroutine floc_deposition_mcpbe.f

```
!              *******************************
               SUBROUTINE FLOC_DEPOSITION_MCPBE
!              *******************************
!
!***********************************************************************
! TELEMAC3D
!***********************************************************************
!
!brief   TREATMENT OF DEPOSITION FLUXES FOR MCPBE FLOCCULATION MODEL
!+       THIS SUBROUTINE ISUSED FOR BOTH 2CPBE AND 3CPBE CASES.
!
!history  QILONG BI
!+       01/09/21
!+       V8P1
!+
!
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!| NPOIN3        |-->| NUMBER OF POINTS IN 3D MESH
!| TA            |-->| TRACER (CONCENTRATIONS OF FLOCS)
!~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
!
      USE DECLARATIONS_GAIA, ONLY: NSUSP_TEL,DBPOIN_MCPBE,FLUDP,FLUER,
     &            NUM_ISUSP_ICLA,MCPBE_VER,IMICFLC,IMACFLC,IMEGFLC,
     &            IMICF_MACF,IMICF_MEGF
      USE DECLARATIONS_TELEMAC3D
!
      IMPLICIT NONE

!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!
      INTEGER IDMICF,IDMACF,IDMICMAC,IDMEGF,IDMICMEG
      INTEGER ITRAC,IPOIN
!
!+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!
!     TREATMENT OF DEPOSITION FLUX BEFORE BED EVOLUTION
!     ALL THE DEPOSITION FLUXES ARE AGGREGATED IN MICROFLOCS

      IF(NSUSP_TEL.GT.0.AND.DBPOIN_MCPBE.GT.0
     &                .AND.MOD(LT,LISPRD).EQ.0) THEN
```

```
      WRITE(LU,*) '-----------------------------------------'
      WRITE(LU,*) 'AT TIME',AT,'BEFORE TREATMENT'
      WRITE(LU,*) '-----------------------------------------'
      DO ITRAC=IND_SED, IND_SED+NSUSP_TEL-1
       ISUSP=ITRAC-IND_SED+1
       WRITE(LU,*) 'EROSION DEPOSITION FLUX OF TRACER ',ITRAC
       WRITE(LU,*) 'ISUSP',ISUSP,NAMETRAC(ITRAC)(1:16),
     &          'NUM_ISUSP_ICLA',NUM_ISUSP_ICLA(ISUSP),'FLUDP',
     &          FLUDP%ADR(NUM_ISUSP_ICLA(ISUSP))%P%R(DBPOIN_MCPBE),
     &          'TA',TA%ADR(ITRAC)%P%R(DBPOIN_MCPBE),
     &          'WCHU',WCHU%ADR(ITRAC)%P%R(DBPOIN_MCPBE),'FLUER',
     &          FLUER%ADR(NUM_ISUSP_ICLA(ISUSP))%P%R(DBPOIN_MCPBE)
      ENDDO
     ENDIF

!    TREATMENT OF DEPOSITION FLUX FOR 2CPBE MODEL
     IF(MCPBE_VER.EQ.1) THEN
       IDMICF = NUM_ISUSP_ICLA(IMICFLC-IND_SED+1)
       IDMACF = NUM_ISUSP_ICLA(IMACFLC-IND_SED+1)
       IDMICMAC = NUM_ISUSP_ICLA(IMICF_MACF-IND_SED+1)
      DO IPOIN=1,NPOIN2
!       MIC_IN_MAC deposition flux is added to MIC
       FLUDP%ADR(IDMICF)%P%R(IPOIN)=
     &          FLUDP%ADR(IDMICF)%P%R(IPOIN)
     &         +FLUDP%ADR(IDMICMAC)%P%R(IPOIN)
!       MAC and MIC_IN_MAC deposition fluxes are set to zero
       FLUDP%ADR(IDMACF)%P%R(IPOIN)=0.D0
       FLUDP%ADR(IDMICMAC)%P%R(IPOIN)=0.D0
      ENDDO
     ENDIF

     IF(MCPBE_VER.EQ.2) THEN
       IDMICF = NUM_ISUSP_ICLA(IMICFLC-IND_SED+1)
       IDMACF = NUM_ISUSP_ICLA(IMACFLC-IND_SED+1)
       IDMICMAC = NUM_ISUSP_ICLA(IMICF_MACF-IND_SED+1)
       IDMICMEG = NUM_ISUSP_ICLA(IMICF_MEGF-IND_SED+1)
      DO IPOIN=1,NPOIN2
!       MIC_IN_MAC and MIC_IN_MEG deposition flux is added to MIC
       FLUDP%ADR(IDMICF)%P%R(IPOIN)=
     &          FLUDP%ADR(IDMICF)%P%R(IPOIN)
     &         +FLUDP%ADR(IDMICMAC)%P%R(IPOIN)
     &         +FLUDP%ADR(IDMICMEG)%P%R(IPOIN)
!       MAC and MEG flocs deposition flux is set to zero
       FLUDP%ADR(IDMACF)%P%R(IPOIN)=0.D0
       FLUDP%ADR(IDMICMAC)%P%R(IPOIN)=0.D0
       FLUDP%ADR(IDMICMEG)%P%R(IPOIN)=0.D0
```

```
      ENDDO
     ENDIF

     IF(MCPBE_VER.EQ.3) THEN
       IDMICF = NUM_ISUSP_ICLA(IMICFLC-IND_SED+1)
       IDMACF = NUM_ISUSP_ICLA(IMACFLC-IND_SED+1)
       IDMEGF = NUM_ISUSP_ICLA(IMEGFLC-IND_SED+1)
       IDMICMAC = NUM_ISUSP_ICLA(IMICF_MACF-IND_SED+1)
       IDMICMEG = NUM_ISUSP_ICLA(IMICF_MEGF-IND_SED+1)
      DO IPOIN=1,NPOIN2
!       MIC_IN_MAC and MIC_IN_MEG deposition flux is added to MIC
       FLUDP%ADR(IDMICF)%P%R(IPOIN)=
     &                FLUDP%ADR(IDMICF)%P%R(IPOIN)
     &               +FLUDP%ADR(IDMICMAC)%P%R(IPOIN)
     &               +FLUDP%ADR(IDMICMEG)%P%R(IPOIN)
!       MAC and MEG flocs deposition flux is set to zero
       FLUDP%ADR(IDMACF)%P%R(IPOIN)=0.D0
       FLUDP%ADR(IDMEGF)%P%R(IPOIN)=0.D0
       FLUDP%ADR(IDMICMAC)%P%R(IPOIN)=0.D0
       FLUDP%ADR(IDMICMEG)%P%R(IPOIN)=0.D0
      ENDDO
     ENDIF

     IF(NSUSP_TEL.GT.0.AND.DBPOIN_MCPBE.GT.0
     &                .AND.MOD(LT,LISPRD).EQ.0) THEN
      WRITE(LU,*) '-----------------------------------------'
      WRITE(LU,*) 'AT TIME',AT,'AFTER TREATMENT'
      WRITE(LU,*) '-----------------------------------------'
      DO ITRAC=IND_SED, IND_SED+NSUSP_TEL-1
       ISUSP=ITRAC-IND_SED+1
       WRITE(LU,*) 'EROSION DEPOSITION FLUX OF TRACER ',ITRAC
       WRITE(LU,*) 'ISUSP',ISUSP,NAMETRAC(ITRAC)(1:16),
     &          'NUM_ISUSP_ICLA',NUM_ISUSP_ICLA(ISUSP),'FLUDP',
     &          FLUDP%ADR(NUM_ISUSP_ICLA(ISUSP))%P%R(DBPOIN_MCPBE),
     &          'TA',TA%ADR(ITRAC)%P%R(DBPOIN_MCPBE),
     &          'WCHU',WCHU%ADR(ITRAC)%P%R(DBPOIN_MCPBE),'FLUER',
     &          FLUER%ADR(NUM_ISUSP_ICLA(ISUSP))%P%R(DBPOIN_MCPBE)
      ENDDO
     ENDIF

     RETURN
     END
```

Appendix IX.
Example of .cas files for GAIA

2CPBE

```
/-----------------------------------------------------------------
/                       GAIA
/-----------------------------------------------------------------
/ GENERAL
/-----------------------------------------------------------------
/
TITLE = 'Schematize_Scheldt_Estuary'

GEOMETRY FILE          = 'geo_1DV_column_v2.slf'
BOUNDARY CONDITIONS FILE = 'bc_1DV_column_v2.cli'
RESULTS FILE           = 'r2D_mixing_cylinder_gaia_2CPBE_400mum.slf'

VARIABLES FOR GRAPHIC PRINTOUTS = 'E,TOB,M,QSBL'
MASS-BALANCE = YES
DEBUGGER = 0

/-----------------------------------------------------------------
/ NUMERICAL PARAMETERS
/-----------------------------------------------------------------
/
ZERO = 1e-12
MINIMAL VALUE OF THE WATER HEIGHT = 0.001

/-----------------------------------------------------------------
/ PHYSICAL PARAMETERS
/-----------------------------------------------------------------
CLASSES TYPE OF SEDIMENT    = CO;CO;CO
CLASSES SEDIMENT DENSITY    = 2500.0;1800.0;1800.0
CLASSES SEDIMENT DIAMETERS  = 0.000015;0.00007;0.00007
CLASSES INITIAL FRACTION    = 1.0;0.0;0.0

FLOCCULATION        = YES
FLOCCULATION FORMULA = 3
MCPBE VERSION       = 1
TREATMENT OF SINK AND SOURCE TERMS IN MCPBE = 1
PRINT POINT INFO MCPBE = 13

SIZE OF MICROFLOCS = 15.D-6
MICROFLOC DENSITY = 2500.0
FRACTAL DIMENSION OF MACROFLOCS = 2.2
COLLISION EFFICIENCY = 1.81
```

```
BREAKUP EFFICIENCY = 5.0E-05
FLOC BREAKUP FREQUENCY = 0.5
FLOC BREAKUP FREQUENCY COEFFICIENT = 0.5
FRACTION OF CREATED MICROFLOCS BY MACROFLOC BREAKUP = 0.1
NUMBER OF CREATED MACROFLOCS BY MACROFLOC BREAKUP = 2.0
INI. NUMBER OF MICROFLOCS BOUNDED IN MACROFLOCS = 2

NUMBER OF LAYERS FOR INITIAL STRATIFICATION = 1
LAYERS INITIAL THICKNESS                = 0.000

/Bedload computation
BED LOAD FOR ALL SANDS = NO
SLOPE EFFECT         = YES
BED-LOAD TRANSPORT FORMULA FOR ALL SANDS = 7
LAYERS NON COHESIVE BED POROSITY = 0.375
CLASSES SHIELDS PARAMETERS      = 0.2

/Suspended load computation
SUSPENSION FOR ALL SANDS =  NO
/EQUILIBRIUM INFLOW CONCENTRATION = YES

SCHEME FOR ADVECTION OF SUSPENDED SEDIMENTS=5;5;5
SCHEME OPTION FOR ADVECTION OF SUSPENDED SEDIMENTS=4;4;4
SOLVER FOR DIFFUSION OF SUSPENSION=1;1;1

INITIAL SUSPENDED SEDIMENTS CONCENTRATION VALUES=0.3999;0.00005;0.0001
PRESCRIBED SUSPENDED SEDIMENTS CONCENTRATION VALUES=
0.09;0.001;0.01;0.09;0.001;0.01
VERTICAL PROFILES OF SUSPENDED SEDIMENTS=1;1;1

COEFFICIENT FOR HORIZONTAL DIFFUSION OF SUSPENDED SEDIMENTS=
1.E-6;1.E-6;1.E-6
COEFFICIENT FOR VERTICAL DIFFUSION OF SUSPENDED SEDIMENTS=
2.E-1;2.E-1;2.E-1

MAXIMUM NUMBER OF ITERATIONS FOR SOLVER FOR SUSPENSION = 200
LAYERS MUD CONCENTRATION         = 500.0
LAYERS CRITICAL EROSION SHEAR STRESS OF THE MUD = 0.1
CLASSES CRITICAL SHEAR STRESS FOR MUD DEPOSITION = 1000.0

LAYERS PARTHENIADES CONSTANT = 2.D-03

SKIN FRICTION CORRECTION = 0

ADVECTION-DIFFUSION SCHEME WITH SETTLING VELOCITY = 0
/
```

3CPBE_var1

```
/-----------------------------------------------------------------
/                       GAIA
/-----------------------------------------------------------------
/ GENERAL
/-----------------------------------------------------------------
/
TITLE = 'Schematize_Scheldt_Estuary'

GEOMETRY FILE          = 'geo_1DV_column_v2.slf'
BOUNDARY CONDITIONS FILE = 'bc_1DV_column_v2.cli'
RESULTS FILE           = 'r2D_mixing_cylinder_gaia_3CPBE_400mum.slf'

VARIABLES FOR GRAPHIC PRINTOUTS = 'E,TOB,M,QSBL'
MASS-BALANCE = YES
DEBUGGER = 0

/-----------------------------------------------------------------
/ NUMERICAL PARAMETERS
/-----------------------------------------------------------------
/
ZERO = 1e-12
MINIMAL VALUE OF THE WATER HEIGHT = 0.001

/-----------------------------------------------------------------
/ PHYSICAL PARAMETERS
/-----------------------------------------------------------------
CLASSES TYPE OF SEDIMENT    = CO;CO;CO;CO
CLASSES SEDIMENT DENSITY    = 2500.0;1600.0;1600.0;1100.0
CLASSES SEDIMENT DIAMETERS  = 0.000015;0.00007;0.00007;0.00036
CLASSES INITIAL FRACTION    = 1.0;0.0;0.0;0.0

FLOCCULATION        = YES
FLOCCULATION FORMULA = 3
MCPBE VERSION       = 2
TREATMENT OF SINK AND SOURCE TERMS IN MCPBE = 1
PRINT POINT INFO MCPBE = 13

SIZE OF MICROFLOCS = 15.D-6
SIZE OF MEGAFLOCS  = 360.D-6
MICROFLOC DENSITY = 2500.0
FRACTAL DIMENSION OF MACROFLOCS = 2.2
FRACTAL DIMENSION OF MEGAFLOCS = 2.2
COLLISION EFFICIENCY = 2.5
BREAKUP EFFICIENCY = 5.E-05
FLOC BREAKUP FREQUENCY = 0.5
FLOC BREAKUP FREQUENCY COEFFICIENT = 0.5
FRACTION OF CREATED MICROFLOCS BY MACROFLOC BREAKUP = 0.1
FRACTION OF CREATED MICROFLOCS BY MEGAFLOC BREAKUP = 0.1
```

```
FRACTION OF REMAINING MEGAFLOCS DURING MEGAFLOC BREAKUP = 0.0
NUMBER OF CREATED MACROFLOCS BY MACROFLOC BREAKUP = 2.0
NUMBER OF CREATED MACROFLOCS BY MEGAFLOC BREAKUP = 2.0
NUMBER OF CREATED MEGAFLOCS BY MEGAFLOC BREAKUP = 0.0
INI. NUMBER OF MICROFLOCS BOUNDED IN MACROFLOCS = 2
INI. NUMBER OF MICROFLOCS BOUNDED IN MEGAFLOCS = 500

NUMBER OF LAYERS FOR INITIAL STRATIFICATION = 1
LAYERS INITIAL THICKNESS                = 0.000

/Bedload computation
BED LOAD FOR ALL SANDS = NO
SLOPE EFFECT         = YES
BED-LOAD TRANSPORT FORMULA FOR ALL SANDS = 7
LAYERS NON COHESIVE BED POROSITY = 0.375
CLASSES SHIELDS PARAMETERS      = 0.2

/Suspended load computation
SUSPENSION FOR ALL SANDS =  NO
/EQUILIBRIUM INFLOW CONCENTRATION = YES

SCHEME FOR ADVECTION OF SUSPENDED SEDIMENTS=5;5;5;5
SCHEME OPTION FOR ADVECTION OF SUSPENDED SEDIMENTS=4;4;4;4
SOLVER FOR DIFFUSION OF SUSPENSION=1;1;1;1
PRECONDITIONING FOR DIFFUSION OF SUSPENSION=2;2;2;2

INITIAL SUSPENDED SEDIMENTS CONCENTRATION VALUES=
0.3998;0.00005;0.0001;0.0001
PRESCRIBED SUSPENDED SEDIMENTS CONCENTRATION VALUES=
0.04;0.032;0.32;0.04;0.04;0.032;0.32;0.04
VERTICAL PROFILES OF SUSPENDED SEDIMENTS=1;1;1;1

COEFFICIENT FOR HORIZONTAL DIFFUSION OF SUSPENDED SEDIMENTS=
1.E-6;1.E-6;1.E-6;1.E-6
COEFFICIENT FOR VERTICAL DIFFUSION OF SUSPENDED SEDIMENTS=
2.E-1;2.E-1;2.E-1;2.E-1

MAXIMUM NUMBER OF ITERATIONS FOR SOLVER FOR SUSPENSION = 200
LAYERS MUD CONCENTRATION         = 500.0
LAYERS CRITICAL EROSION SHEAR STRESS OF THE MUD = 0.1
CLASSES CRITICAL SHEAR STRESS FOR MUD DEPOSITION = 1000.0

LAYERS PARTHENIADES CONSTANT = 2.D-03

SKIN FRICTION CORRECTION = 0

ADVECTION-DIFFUSION SCHEME WITH SETTLING VELOCITY = 0
/
```

3CPBE_var2

```
/-----------------------------------------------------------------
/                       GAIA
/-----------------------------------------------------------------
/ GENERAL
/-----------------------------------------------------------------
/
TITLE = 'Schematize_Scheldt_Estuary'

GEOMETRY FILE          = 'geo_1DV_column_v2.slf'
BOUNDARY CONDITIONS FILE = 'bc_1DV_column_v2.cli'
RESULTS FILE           = 'r2D_mixing_cylinder_gaia_3CPBE_400mgl.slf'

VARIABLES FOR GRAPHIC PRINTOUTS = 'E,TOB,M,QSBL'
MASS-BALANCE = YES
DEBUGGER = 0

/-----------------------------------------------------------------
/ NUMERICAL PARAMETERS
/-----------------------------------------------------------------
/
ZERO = 1e-12
MINIMAL VALUE OF THE WATER HEIGHT = 0.001

/-----------------------------------------------------------------
/ PHYSICAL PARAMETERS
/-----------------------------------------------------------------
CLASSES TYPE OF SEDIMENT    = CO;CO;CO;CO;CO
CLASSES SEDIMENT DENSITY    = 2500.0;1600.0;1600.0;1100.0;1100.0
CLASSES SEDIMENT DIAMETERS  = 0.000015;0.00007;0.00007;0.00036;0.00036
CLASSES INITIAL FRACTION    = 1.0;0.0;0.0;0.0;0.0

FLOCCULATION        = YES
FLOCCULATION FORMULA = 3
MCPBE VERSION       = 3
TREATMENT OF SINK AND SOURCE TERMS IN MCPBE = 1
PRINT POINT INFO MCPBE = 13

SIZE OF MICROFLOCS = 15.D-6
SIZE OF MEGAFLOCS  = 360.D-6
MICROFLOC DENSITY = 2500.0
FRACTAL DIMENSION OF MACROFLOCS = 2.2
FRACTAL DIMENSION OF MEGAFLOCS = 2.0
COLLISION EFFICIENCY = 2.93
BREAKUP EFFICIENCY = 5.E-05
FLOC BREAKUP FREQUENCY = 0.5
FLOC BREAKUP FREQUENCY COEFFICIENT = 0.5
FRACTION OF CREATED MICROFLOCS BY MACROFLOC BREAKUP = 0.1
FRACTION OF CREATED MICROFLOCS BY MEGAFLOC BREAKUP = 0.0
```

```
FRACTION OF REMAINING MEGAFLOCS DURING MEGAFLOC BREAKUP = 0.1
NUMBER OF CREATED MACROFLOCS BY MACROFLOC BREAKUP = 2.0
NUMBER OF CREATED MACROFLOCS BY MEGAFLOC BREAKUP = 2.0
NUMBER OF CREATED MEGAFLOCS BY MEGAFLOC BREAKUP = 0.2
INI. NUMBER OF MICROFLOCS BOUNDED IN MACROFLOCS = 2
INI. NUMBER OF MICROFLOCS BOUNDED IN MEGAFLOCS = 4
MAX NUMBER OF MICROFLOCS BOUNDED IN MEGAFLOCS = 2000

NUMBER OF LAYERS FOR INITIAL STRATIFICATION = 1
LAYERS INITIAL THICKNESS                = 0.000

/Bedload computation
BED LOAD FOR ALL SANDS = NO
SLOPE EFFECT         = YES
BED-LOAD TRANSPORT FORMULA FOR ALL SANDS = 7
LAYERS NON COHESIVE BED POROSITY = 0.375
CLASSES SHIELDS PARAMETERS      = 0.2

/Suspended load computation
SUSPENSION FOR ALL SANDS =  NO
/EQUILIBRIUM INFLOW CONCENTRATION = YES

SCHEME FOR ADVECTION OF SUSPENDED SEDIMENTS=5;5;5;5;5
SCHEME OPTION FOR ADVECTION OF SUSPENDED SEDIMENTS=4;4;4;4;4
SOLVER FOR DIFFUSION OF SUSPENSION=1;1;1;1;1
PRECONDITIONING FOR DIFFUSION OF SUSPENSION=2;2;2;2;2

INITIAL SUSPENDED SEDIMENTS CONCENTRATION VALUES=
0.3998;0.00005;0.0001;0.000025;0.0001
PRESCRIBED SUSPENDED SEDIMENTS CONCENTRATION VALUES=
0.04;0.032;0.32;0.00008;0.04;0.04;0.032;0.32;0.00008;0.04
VERTICAL PROFILES OF SUSPENDED SEDIMENTS=1;1;1;1;1

COEFFICIENT FOR HORIZONTAL DIFFUSION OF SUSPENDED SEDIMENTS=
1.E-6;1.E-6;1.E-6;1.E-6;1.E-6
COEFFICIENT FOR VERTICAL DIFFUSION OF SUSPENDED SEDIMENTS=
2.E-1;2.E-1;2.E-1;2.E-1;2.E-1

MAXIMUM NUMBER OF ITERATIONS FOR SOLVER FOR SUSPENSION = 200
LAYERS MUD CONCENTRATION         = 500.0
LAYERS CRITICAL EROSION SHEAR STRESS OF THE MUD = 0.1
CLASSES CRITICAL SHEAR STRESS FOR MUD DEPOSITION = 1000.0

LAYERS PARTHENIADES CONSTANT = 2.D-03

SKIN FRICTION CORRECTION = 0

ADVECTION-DIFFUSION SCHEME WITH SETTLING VELOCITY = 0
/
```

DEPARTMENT **MOBILITY & PUBLIC WORKS**
Flanders hydraulics Research

Berchemlei 115, 2140 Antwerp
T +32 (0)3 224 60 35
F +32 (0)3 224 60 36
waterbouwkundiglabo@vlaanderen.be
www.flandershydraulicsresearch.be