TELEMAC-3D

Theory guide

Version v8p3
December 6, 2021

Authors and contributions

This theory guide was created by Agnès Leroy (EDF R&D, LNHE), based on the previous available TELEMAC-3D documentation. In particular, most of the material came from documents written by Jean-Michel Hervouet (EDF R&D, LNHE), be it his book, *Hydrodynamics of Free-Surface Flows*, which also lists a number of contributors, or the TELEMAC-3D release notes. The release notes also included contributions by Chi-Tuân Pham (EDF R&D, LNHE). This guide, as well as the abovementioned sources, rely on previous work by Jacek Jankowski and Astrid Decoene during their respective PhD theses. It also benefits from contributions by Lamia Abbas, Pierrick Quemar, Antoine Joly and Jacques Fontaine.

Work under construction

We would like to warn the reader that this document is still under construction and could be improved in many ways. Any remarks, suggestions, corrections, etc. are welcome and can be submitted to the TELEMAC developing team through the forum in the Documentation category: `http://www.opentelemac.org/index.php/kunena/10-documentation`.

Abstract

This guide aims at providing a comprehensive overlook of the theory behind TELEMAC-3D. It is based on Jean-Michel Hervouet's book, *Hydrodynamics of Free-Surface Flows* [36], as well as on the previous TELEMAC-3D release notes. It is organised into four chapters: the first two describe the equations and modelling choices in a continuous framework, as well as the principles of the Finite Elements method. The latter was nearly directly taken from *Hydrodynamics of Free Surface Flows*, but also includes some contributions from Jacek Jankowski's PhD thesis. After this, the time discretisation and then the space-time discretisation of the equations are described in two separate chapters. While the first two chapters are rather classical, the last two are specific to TELEMAC-3D. They benefitted from a lot of material from the former release notes and from *Hydrodynamics of Free-Surface Flows*. They are still unfinished at this stage. In particular, they contain some information regarding the resolution of the advection in TELEMAC-3D and the treatment of tidal flats, but these two sections directly come from the last release notes and have not been adapted to this guide yet. Nevertheless, we hope that these chapters will help the TELEMAC-3D users and developers better understand what is at the core of the module.

The main contributions in this theory guide, compared to previous works, lie in the new format of the document, which aims at gathering all the theoretical background of TELEMAC-3D. It includes highlighted remarks regarding the limits and possible improvements of the models used in the code. The chapter 3 is new: it aims at giving the reader an overview of the time scheme, even though all the demonstrations of consistency, conservation, etc. are only valid when written for the space-time discretised equations. New notations are adopted compared to previous works, with the hope that they will facilitate the understanding of the numerical schemes.

Contents

1. Equations and modelling choices in a continuous framework

1.1 Notations and concepts in geometry

The domain of study Ω in the referential $R = (O,x,y,z)$ (Oz along the vertical) is limited below by the bed described by the equation $z = b(x,y,t)$, or in most situations only $z = b(x,y)$, and above by the free surface described by the equation $z = \eta(x,y,t)$. It is limited laterally by vertical lines. Its boundary is Γ. The projection of Ω on the horizontal plane (O,x,y), that will be later our two-dimensional domain, is denoted by Ω_{2D} (see the figure 1.1). The domain of calculation is thus defined by:

$$\Omega = \left\{(x,y,z) \in \mathbb{R}^3 \text{ such that } (x,y) \in \Omega_{2D} \text{ and } b(x,y,t) \leq z \leq \eta(x,y,t)\right\} \tag{1.1}$$

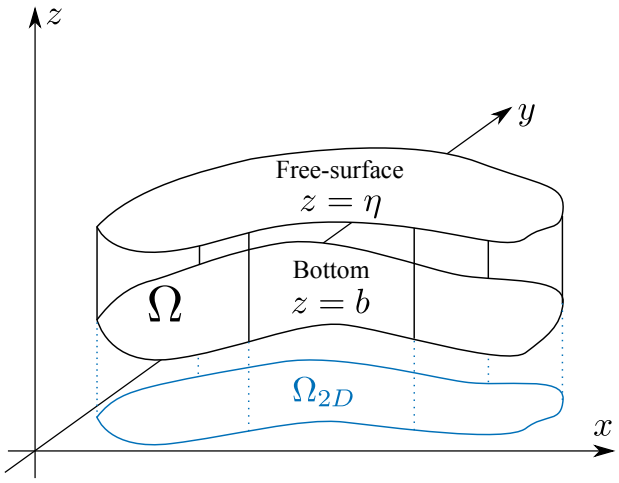

Figure 1.1: Sketch of the calculation domain (inspired by [36]).

Remark:

At the moment, bed motion is possible only when modelling the bed evolution due to sediment transport in TELEMAC-3D. Then, b is really a function of time, which modifies the equations presented in this chapter, but we did not detail the corresponding changes in the present version of the document.

Gravity is directed towards decreasing z. The height of the water column, or water depth, will be denoted by h and is equal to $\eta - b$. The bed elevation is given data. The water depth is generally an unknown quantity. The time will be denoted by t. Throughout the document, vectors with the subscript $2D$ correspond to the 2D horizontal vector of x and y components: let $\boldsymbol{u} = u\boldsymbol{e}_x + v\boldsymbol{e}_y + w\boldsymbol{e}_z$ be the 3D velocity, then $\boldsymbol{u}_{2D} = u\boldsymbol{e}_x + v\boldsymbol{e}_y$.

Let us now define the normal vectors to the boundaries. There are four types of boundaries to be considered: the bed Γ_b, the free-surface Γ_η, liquid lateral boundaries Γ_l and solid lateral boundaries Γ_s. Let $\boldsymbol{n}$ be the normal to a boundary, oriented outwards the domain. We denote by $\boldsymbol{n}_b$, $\boldsymbol{n}_\eta$, $\boldsymbol{n}_l$ and $\boldsymbol{n}_s$ the outward normal to the bed, the free-surface, the liquid lateral boundaries and the solid lateral boundaries. The free surface is a univalent function of coordinates x and y, which excludes calculation of waves on the verge of breaking, and varies with time. Its equation is in the form of $z = \eta(x,y,t)$. It can also be written as $\Phi(x,y,z,t) = 0$ with:

$$\Phi(x,y,z,t) = z - \eta(x,y,t) \tag{1.2}$$

The normal to the free surface, directed towards increasing z (hence external in relation to the volume containing water), is then the vector (not normalised):

$$\boldsymbol{n}_\eta = \boldsymbol{\nabla}(\Phi) \tag{1.3}$$

This vector has the following components:

$$\boldsymbol{n}_\eta = \left(-\frac{\partial \eta}{\partial x}, -\frac{\partial \eta}{\partial y}, 1\right) \tag{1.4}$$

The normal to the bed would be expressed in the same way but by replacing η by b, and with a minus sign to make it external to the volume of water. That is:

$$\boldsymbol{n}_b = \left(\frac{\partial b}{\partial x}, \frac{\partial b}{\partial y}, -1\right) \tag{1.5}$$

Note that $\boldsymbol{n}_\eta$ and $\boldsymbol{n}_b$ are not defined as unit vectors. In TELEMAC-3D, the bed elevation is a univocal function of coordinates x and y.

Possible improvement:

The fact that the function $b(x,y)$ is univocal is imposed by the choices in the numerical scheme of TELEMAC-3D. It implies that vertical sections of the bed can only be represented through a steep slope. Investigations regarding how to extend the TELEMAC-3D software to non univocal beds have been undertaken but are still in progress (see *e.g.* [71]).

1.2 The Navier-Stokes equations

The Navier–Stokes equations for incompressible flows consist of two equations: the continuity and the momentum equations. We first consider a possibly compressible flow in a domain Ω of dimension 3. The compressible continuity equation represents the mass conservation in a continuous medium and reads:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}) = 0 \tag{1.6}$$

where ρ is the density, $\boldsymbol{u}$ the velocity, $\boldsymbol{g}$ is the gravity and t is the time. The continuous gradient operator is denoted by $\boldsymbol{\nabla}$ and the divergence by $\nabla\cdot$. The Navier–Stokes momentum equation

is obtained from the Cauchy momentum equation for a continuous medium where a behaviour law is introduced to model the stress tensor. We recall the Cauchy equation:

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u}\cdot\boldsymbol{\nabla})\boldsymbol{u} = \frac{1}{\rho}\boldsymbol{\nabla}\cdot\boldsymbol{\sigma} + \boldsymbol{g} + \boldsymbol{F} \tag{1.7}$$

In this equation, $\boldsymbol{\sigma}$ is the Cauchy stress tensor, $\boldsymbol{F}$ are external forces other than the gravity (*e.g.* the Coriolis and centrifugal forces, see the section 1.8). On the other hand, the behaviour law used for a Newtonian fluid reads:

$$\begin{aligned} \boldsymbol{\sigma} &= -p\boldsymbol{I} + \boldsymbol{\sigma}_\tau \\ \boldsymbol{\sigma}_\tau &= \lambda\ \nabla\cdot\boldsymbol{u}\ \boldsymbol{I} + 2\mu\boldsymbol{s} \end{aligned} \tag{1.8}$$

where $\boldsymbol{I}$ is the identity tensor in dimension 3, $\boldsymbol{\sigma}_\tau$ is the shear-stress tensor, $\boldsymbol{s}$ is the strain-rate tensor, p is the pressure, μ is the dynamic molecular viscosity and λ is equal to $\zeta - 2/3\mu$ where ζ is the bulk viscosity. The tensor $\boldsymbol{s}$ is defined as:

$$\boldsymbol{s} = \frac{1}{2}\left[\boldsymbol{\nabla}\boldsymbol{u} + \boldsymbol{\nabla}\boldsymbol{u}^T\right] \tag{1.9}$$

where ^{T} denotes the transpose of a vector or tensor. The behaviour law (1.8) was obtained by introducing a model for viscosity in fluids based on an analogy with a model for particle friction.

Most of the time, the flows considered here involve buoyancy effects due to active scalars. Density variations in buoyant flows mainly affect the flow dynamics through the gravity term. We consider that the Boussinesq approximation is valid for the flows considered here, *i.e.* $\Delta\rho/\rho_0 \ll 1$ where ρ_0 is the reference density and $\Delta\rho$ the fluctuations around it[1]. This enables the treatment of buoyancy affecting the fluid motion by means of the gravity term only (see the section 1.5.3). Then, the fluid density is considered as constant and the continuity equation (1.6) can be rewritten as:

$$\nabla\cdot\boldsymbol{u} = 0 \tag{1.10}$$

The equation (1.10) can be used to simplify the shear-stress tensor $\boldsymbol{\sigma}_\tau$. Considering that the density ρ is constant, and neglecting the term $\boldsymbol{\nabla}\cdot\left(\mu\boldsymbol{\nabla}\boldsymbol{u}^T\right)$, the Navier–Stokes equations then read:

$$\begin{cases} \nabla\cdot\boldsymbol{u} = 0 \\ \dfrac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u}\cdot\boldsymbol{\nabla})\boldsymbol{u} = -\dfrac{1}{\rho}\boldsymbol{\nabla}p + \dfrac{1}{\rho}\boldsymbol{\nabla}\cdot(\mu\boldsymbol{\nabla}\boldsymbol{u}) + \boldsymbol{g} + \tilde{\boldsymbol{F}} \end{cases} \tag{1.11}$$

where $\tilde{\boldsymbol{F}}$ represents the external forces plus the buoyancy terms (see the section 1.10, equation (1.77) for their definition).

Warning:

The term $\boldsymbol{\nabla}\cdot\left(\mu\boldsymbol{\nabla}\boldsymbol{u}^T\right)$ is always neglected in TELEMAC-3D, but it may have an influence in case of turbulent flows, when the viscosity is not constant (see the section 1.6).

[1] The upper limit for the Boussinesq approximation validity is considered in [96] to be $\Delta\rho/\rho < 0.1$.

1.3 Boundary conditions

The closure of the system (1.11) involves the imposition of initial and boundary conditions on the velocity[2]. The pressure is the Lagrangian multiplier that stems from the minimisation of the momentum equation under the constraint $\nabla \cdot \boldsymbol{u} = 0$ [6]. Its computation is a key-point in the resolution of the incompressible Navier-Stokes equations. Many methods were developed to compute it and the strategy adopted in TELEMAC-3D is described in the section 1.4. It is noteworthy that at this stage of the considerations, the pressure does not require any boundary conditions.

1.3.1 Bed and lateral solid boundaries

On solid boundaries, the kinematic condition expressing the imperviousness is:

$$\boldsymbol{u}|_{\Gamma_s} \cdot \boldsymbol{n}_s = 0 \qquad \text{and} \quad \boldsymbol{u}|_{\Gamma_b} \cdot \boldsymbol{n}_b = 0 \tag{1.12}$$

Where $\boldsymbol{u}|_{\Gamma_i}$ is the fluid velocity on the solid boundary Γ_i (either the bed or a lateral boundary). On the lateral solid boundaries, this is equivalent to:

$$\boldsymbol{u}|_{\Gamma_s} \cdot \boldsymbol{n}_{s2D} = 0 \qquad \text{on } \Gamma_s \tag{1.13}$$

since they are vertical. On the bed, the condition (1.12) is equivalent to:

$$\frac{d}{dt}(z-b) = 0 \qquad \text{on } \Gamma_b \tag{1.14}$$

This gives:

$$w|_{z=b} - \frac{\partial b}{\partial t} - \boldsymbol{u}_{2D}|_{\Gamma_b} \cdot \boldsymbol{\nabla}_{2D} b = 0 \qquad \text{on } \Gamma_b \tag{1.15}$$

Where $w|_{z=b}$ is the vertical component of fluid velocity of the bed, $\boldsymbol{u}|_{\Gamma_b}$ is the fluid velocity on the boundary Γ_b. The condition (1.15) corresponds to the fact that a point of the bed will stay on the bed: it is a kinematic condition. On the other hand, the dynamic condition at the solid boundaries accounts for the shear-stress $\boldsymbol{\tau}$:

$$\boldsymbol{\tau} = -\mu \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{n}} \qquad \text{on } \Gamma_b \cup \Gamma_s \tag{1.16}$$

The bed shear stress $\boldsymbol{\tau}$ acting on the fluid is oriented opposite to the tangential velocity to the bed: it is the opposite to the stress applied by the fluid onto the bed, which is in the direction of the current. Knowledge of this stress requires knowledge of the flow. The turbulence models will provide the modelling, based on knowledge of the current in the vicinity of the bed. In one dimension the shear stress is written as:

$$\tau = -\rho u_*^2 \tag{1.17}$$

which is a definition of a shear velocity u_*. The shear-stress will then be prescribed as a boundary condition in turbulence models (see the section 1.6.5).

Warning:
In case no turbulence model is used, the friction is still always prescribed as that of a turbulent flow.

[2]Note that in fact, with these initial and boundary conditions the existence and uniqueness of a global solution to the 3-D Navier–Stokes equations (with any source term and on any time interval) was not proved, although it was proved on various particular cases [52].

1.3.2 Free-surface boundary

The kinematic condition on the free-surface reads:

$$\frac{d}{dt}(z-\eta)=0 \qquad \text{on } \Gamma_\eta \tag{1.18}$$

As before, this represents the fact that at the free-surface, the velocity of the fluid is equal to the velocity of the free-surface. If z is on the free surface, this gives:

$$w|_{z=\eta}-\frac{\partial \eta}{\partial t}-\boldsymbol{u}_{2D}|_{\Gamma_\eta}\cdot\boldsymbol{\nabla}_{2D}\eta=0 \qquad \text{on } \Gamma_\eta \tag{1.19}$$

Where $w|_{z=\eta}$ is the vertical component of fluid velocity of the free-surface, $\boldsymbol{u}|_{\Gamma_\eta}$ is the fluid velocity on the boundary Γ_η. On the other hand, the dynamic condition at the free-surface reads:

$$(\boldsymbol{\sigma}_\tau\cdot\boldsymbol{n}_\eta)_{fluid}=(\boldsymbol{\sigma}_\tau\cdot\boldsymbol{n}_\eta)_{air} \tag{1.20}$$

Assuming that the pressure is equal to the atmospheric pressure p_{atm} and the viscous air stress is small enough to be disregarded (which is not the case when wind effects are taken into account), the condition (1.20) comes to:

$$p=p_{atm} \quad \text{and} \quad \boldsymbol{\sigma}\cdot\boldsymbol{n}_\eta=0 \qquad \text{on } \Gamma_\eta \tag{1.21}$$

Wind-induced stress on the free-surface

In case the influence of wind is taken into account, wind forcing appears in the equations through the two-dimensional condition at the surface:

$$\boldsymbol{\sigma}\cdot\boldsymbol{n}_\eta=\mu\frac{\partial \boldsymbol{u}_{2D}}{\partial n}=\rho_{air}a_{wind}u_{wind}\boldsymbol{u}_{wind} \tag{1.22}$$

ρ_{air}, the air density, is a function of the air temperature. However, in TELEMAC-3D a constant value of 1.3 kg/m^3 is used for the wind-induced friction on the free-surface. $\boldsymbol{u}_{2D}$ is the horizontal velocity at the surface and $\boldsymbol{u}_{wind}$ the wind velocity, usually measured 10 m above the water, and u_{wind} is its norm. The coefficient a_{wind} (dimensionless) is given by Flather [31]:

$$\begin{cases} a_{wind}=0.565\ 10^{-3} & \text{if} \quad \|\boldsymbol{w}\|\leq 5 \text{ m/s} \\ a_{wind}=(-0.12+0.137\,\|\boldsymbol{w}\|)\ 10^{-3} & \text{if} \quad 5 \text{ m/s}\leq\|\boldsymbol{w}\|\leq 19.22 \text{ m/s} \\ a_{wind}=2.513\ 10^{-3} & \text{if} \quad \|\boldsymbol{w}\|\geq 19.22 \text{ m/s} \end{cases} \tag{1.23}$$

The coefficient a_{wind} hides complex phenomena and the above values are only a guideline. In fact, the influence of the wind depends on the roughness of the free surface, which is itself dependent on the wind and the distance over which it is applied (called “fetch”).

1.3.3 Liquid boundaries

On the open boundaries, additional information is required on the pressure, the water depth, the velocity, the discharge, etc. These boundaries, which are purely artificial, usually cause difficulties in numerical modelling. In particular, certain choices of boundary conditions will not necessarily lead to well-defined problems (that depends on the nature of the flow – sub-critical / super-critical, and its direction). In TELEMAC-3D, the most common ways to prescribe open boundaries are:

- to prescribe a water level by imposing a hydrostatic pressure profile along the vertical;

- to prescribe a discharge by imposing a constant horizontal velocity profile along the vertical;
- to prescribe tidal boundary conditions (see the paragraph below: the horizontal velocities and/or water level are prescribed);
- to prescribe the scalars values.

When the open boundary is experiencing an incoming mass flux, the values specified by the user for the flow rate, water height or scalars are prescribed. When it is experiencing an outgoing mass flux, a homogeneous Neumann boundary condition is applied by interpolating the fields based on the interior values.
It is possible to program more complex open boundary conditions in a dedicated subroutine. In general, what is to be prescribed onto an open boundary is highly dependent on the flow itself.

Remark:
At the moment in TELEMAC-3D it is not possible to prescribe the dynamic pressure at an open boundary – it did not prove necessary until now for the targeted applications.

Tidal boundary conditions

The theory used to model tides in TELEMAC can be read in [43] and [86]. It is partly described here. When modelling tides, open boundary conditions for water depth and/or horizontal components of velocity can vary spatially and temporally and can be calculated at each time step for every boundary node as a sum of harmonic constituents.
For each harmonic constituent, the water depth h and horizontal components of velocity u and v are calculated as below, at point $\boldsymbol{x}$ and time t:

$$f(\boldsymbol{x},t) = \sum_i f_i(\boldsymbol{x},t) \tag{1.24}$$

with:

$$f_i(\boldsymbol{x},t) = f_i(t) A_{f_i}(\boldsymbol{x}) \cos\left(2\pi \frac{t}{T_i} - \phi_{f_i}(\boldsymbol{x}) + \phi_i^0 + g_i(t)\right) \tag{1.25}$$

where f is the water depth h or one of the horizontal components of velocity u or v, i refers to the considered constituent, T_i is the period of the constituent, A_{f_i} is the amplitude of the water depth or one of the horizontal components of velocity of the constituent, ϕ_{f_i} is the phase, $f_i(t)$ and $g_i(t)$ are the nodal factors (see below) and ϕ_i^0 is the phase at the original time of the simulation.

Remark:
In previous versions of TELEMAC-3D release notes and in the TELEMAC-2D documentation, different notations were used for the initial phase and nodal factors. To avoid confusions in the notations, they are slightly different in the present document: v_i (from previous notations) is denoted here by g_i and u_i^0 is denoted by ϕ_i^0.

The water depth and velocities of each constituent are then summed to obtain water depths and velocities to prescribe at open boundary conditions:

$$h = \sum_i h_i - b + z_{mean} \tag{1.26}$$

$$u = \sum_i u_i \tag{1.27}$$

$$v = \sum_i v_i \tag{1.28}$$

where z_{mean} is the level used to calibrate the sea levels. The coefficients A_{f_i} and ϕ_{f_i} are constant in time and only depend on the location. This information is highly sought after in the modelling of tides and some databases do exist for different areas. Among the different databases of harmonic constants available, four of them can be used with the TELEMAC system to force the open boundary conditions:

- Jean-Marc Janin (JMJ) and Xavier Blanchard's model of the English Channel and the near Atlantic Ocean including the Continental Shelf (4 harmonic constituents) [43],
- LEGOS atlases such as the regional NEA (North East Atlantic) atlas processed by NOVELTIS/LEGOS in the frame of the COMAPI project funded by CNES [59], [60] [25] which covers an area from Mauritania to the south of Norway (15 or 47 harmonic constituents if the solution is assimilated with satellite observations or not - the pure hydrodynamic solution has been modelled with T-UGOm finite element software) or the global atlas FES,
- TPXO global tidal solution and regional/local tidal solutions from OSU (Oregon State University) [28], such as the Atlantic Ocean and the European Shelf (around 11 harmonic constituents depending on the area) or areas all over the world,
- PREVIMER atlases with 7 embedded atlases covering the North East Atlantic (3 different resolutions) with 17 or 38 harmonic constituents.

The different databases provide amplitudes and phases for the tidal elevation and for the two horizontal components of the current most of the time for most of them.

The nodal factors ($f_i(t)$, $g_i(t)$) and the phase at the original time of the simulation (ϕ_i^0) are to be taken into account to correct the slow variations induced by the tilting of the moon orbit on the Equator. For solar waves such as S2, there is no correction ($f_i(t) = 1$ and $g_i(t) = \phi_i^0 = 0$ every time). On the contrary, for lunar waves or combined waves with at least one lunar wave, these factors have to be calculated, for example with the formulae coming from Schureman [86] or Pugh [70]. These formulae need the computation of some previous parameters (mean longitude of moon, mean longitude of sun, longitude of the lunar perigee, longitude of lunar ascending node).
Three calibration parameters are available for adjusting the results when calculating tidal open boundary conditions.

- one multiplier coefficient to calibrate tidal ranges ($CTIDE$),
- one multiplier coefficient to calibrate velocities ($CTIDEV$),
- one multiplier coefficient to calibrate sea level (MSL).

In practise, the previous formulae become:

$$h = CTIDE \sum_i h_i - b + MSL \tag{1.29}$$

$$u = CTIDEV \sum_i u_i \tag{1.30}$$

$$v = CTIDEV \sum_i v_i \tag{1.31}$$

For more information, the reader can read [67] and [66].

Thompson formulation for radiative open boundary conditions

The existence of artificial boundaries in the calculation domains, *e.g.* in the open sea, often raises the question of boundary conditions and leads to ill-posed problems. In the open sea, generally only the free-surface elevation is known. It is tempting to adopt a condition with imposed elevation and free velocity. When the current is directed inwards on these boundaries, the problem is ill-posed and the solution could diverge. The theory of characteristics in one dimension, which would help to emphasise the required conditions, cannot be applied strictly in two dimensions. The characteristic curves highlighted in one dimension are transformed into surfaces (see [21]). The "radiation" boundary conditions use the theory of characteristics with a few assumptions. In particular, this is the case for the Thompson method [94]. In practise, the following approach is used:

- all the available data at the open boundaries (elevation and velocities) are provided and declared as imposed values;
- the local conditions at the boundaries are studied in the light of the theory of characteristics and the Riemann invariants, to determine which information is really necessary and which should be kept aside. The new elevations and velocities to be imposed are deduced. These are not necessarily the ones provided by the user.

The Thompson method, adapted to the shallow water equations, is detailed in the Appendix A (5), but a quick description is given here. The idea of the Thompson method is to consider the shallow water equations in matricial form and to change referential so as to place oneself in a referential local to a boundary point.

Important:
Since the Thompson formulation is based on the shallow water equations, it is adapted to flows which are quasi-horizontal close to the open boundaries (*i.e.* the vertical velocity is about zero in these areas). When this is not the case, using the Thompson formulation may lead to inconsistencies (*e.g.* increased numerical dissipation, spurious velocity fields, etc.).

Let (ξ, ζ) be the coordinates in this new referential. It is then possible to re-write the shallow water system in this referential, and to derive the following system (see [36], pp. 105–108):

$$\frac{\partial W}{\partial t} + \Lambda \frac{\partial W}{\partial \xi} = LS_\xi \tag{1.32}$$

where W is defined by $dW = LdF$, with:

$$F = \begin{pmatrix} h \\ hU \\ hV \end{pmatrix}, \quad L = \begin{pmatrix} U_\zeta & \frac{\partial \xi}{\partial y} & -\frac{\partial \xi}{\partial x} \\ c - U_\xi & \frac{\partial \xi}{\partial x} & \frac{\partial \xi}{\partial y} \\ c + U_\xi & -\frac{\partial \xi}{\partial x} & -\frac{\partial \xi}{\partial y} \end{pmatrix} \tag{1.33}$$

On the other hand, Λ is defined by:

$$\Lambda = \begin{pmatrix} U_\xi & 0 & 0 \\ 0 & U_\xi + c & 0 \\ 0 & 0 & U_\xi - c \end{pmatrix} \tag{1.34}$$

with $c = \sqrt{gh}$. Thompson then proposes to consider that L is constant in the vicinity of a boundary point, and to write $W = \overline{L}F$, with $\overline{L}$ defined by:

$$\overline{L} = \begin{pmatrix} \overline{U_\zeta} & \dfrac{\partial \xi}{\partial y} & -\dfrac{\partial \xi}{\partial x} \\ \overline{c} - \overline{U_\xi} & \dfrac{\partial \xi}{\partial x} & \dfrac{\partial \xi}{\partial y} \\ \overline{c} + \overline{U_\xi} & -\dfrac{\partial \xi}{\partial x} & -\dfrac{\partial \xi}{\partial y} \end{pmatrix} \tag{1.35}$$

where the over-line means that the field is taken as constant. The Riemann invariants of the vector W are then:

$$\begin{cases} W_1 = -h\left(U_\zeta - \overline{U_\zeta}\right) & \text{(advection with velocity } U_\xi\text{)} \\ W_2 = h\left(\overline{c} + U_\xi - \overline{U_\xi}\right) & \text{(advection with velocity } U_\xi + c\text{)} \\ W_3 = h\left(-\overline{c} + U_\xi - \overline{U_\xi}\right) & \text{(advection with velocity } U_\xi - c\text{)} \end{cases} \tag{1.36}$$

The method of characteristics is then used to propagate the invariants. Source terms, if any, are taken into account in a fractional step. Three (or more, with scalars) characteristic curves per open boundary point thus have to be calculated. This makes this method more time consuming than only prescribing the field values at the boundary, but it sometimes is the only way to stabilise a simulation. Once the Riemann invariants are known, the shallow water variables (h, U, V, T) are recovered with the following formulae:

$$\begin{cases} h = \dfrac{W_1 - W_3}{2\overline{c}} \\ U = \dfrac{(\partial\xi/\partial y)W_1 + (\partial\xi/\partial x)W_2 - (\partial\xi/\partial x)h\overline{c}}{h} + \overline{U} \\ V = \dfrac{(\partial\xi/\partial y)W_2 - (\partial\xi/\partial x)W_1 - (\partial\xi/\partial y)h\overline{c}}{h} + \overline{V} \\ T = -\dfrac{W_4}{h} + \overline{T} \end{cases} \tag{1.37}$$

The choice of the local referential system is very important. In the original method by Thompson, the normal and tangent to the boundary were used for the referential. However, in this case characteristics of the same family stemming from two different points may cross because they have a different original direction. In the TELEMAC system, it was chosen to use the direction of the velocity field itself.

Remark:

An important consequence of this choice is that the velocity $\overline{U}_\zeta$ is always 0 by definition, which would lead to $W_1 = 0$. This is true in fact only if we consider that the direction of the referential changes along characteristics. It is false if we keep the original direction, which would be consistent with the assumption of constant fields close to the boundary point, leading to $\overline{L}$. Tests show that it is better to consider that U_ζ is indeed not 0, thus sticking to the assumption of constant fields. A possibility that remains to be tested would be to consider that U_ζ is indeed 0, and taking the norm of the velocity for the component U_ξ.

In case the velocity is zero at the boundary, the vector $(-g, \boldsymbol{\nabla}\eta)$ is chosen for the direction. It is indeed the driving term in the momentum equation that will induce a velocity at the next time step. In this way, water level gradients are taken into account in the computation of the boundary fields values. For more details about the implementation, see [36] pp.105–108 and the release notes for TELEMAC version 6.1 [38].

1.4 Treatment of the pressure field in TELEMAC-3D

1.4.1 Decomposition of the pressure

In TELEMAC-3D the pressure is decomposed into a hydrostatic and a dynamic part, *i.e.*:

$$p = p_h + p_d \tag{1.38}$$

Where p_h represents the hydrostatic pressure and p_d the dynamic pressure.
The hydrostatic pressure is defined by the following equation:

$$p_h = \rho g(\eta - z) + p_{atm} \tag{1.39}$$

Under the Boussinesq approximation, this gives:

$$p_h = g\rho_0(\eta - z) + p_{atm} + g\rho_0 \int_z^\eta \frac{\Delta\rho}{\rho_0}\, dz \tag{1.40}$$

The hydrostatic pressure is thus a function of the free-surface elevation η, and it is necessary to compute the latter. This is done by solving an equation on η that is obtained from the integration of the continuity equation on the vertical. The derivation of this equation is described in the subsection 1.4.2. On the other hand, the dynamic pressure is still coupled to the velocity field, and the strategy adopted in TELEMAC-3D is to compute it using the Chorin and Temam projection scheme [17, 93]. This is described in the subsection 1.4.3. Let us now focus on the computation of the hydrostatic pressure, and so on the computation of the free-surface elevation.

1.4.2 The free-surface equation – solving for the hydrostatic pressure

It is possible to show that if the continuity equation (1.10) and the kinematic condition on the bed (1.15) are satisfied, then the kinematic condition at the free-surface (1.19) is equivalent to:

$$\frac{\partial \eta}{\partial t} + \nabla_{2D} \cdot \left(\int_b^\eta \boldsymbol{u}_{2D}\, dz \right) = 0 \tag{1.41}$$

The implication from (1.19) to (1.41) is demonstrated by integrating the continuity equation (1.10) from the bed to the free-surface:

$$\int_b^\eta \nabla \cdot \boldsymbol{u}\, dz = 0 \tag{1.42}$$

If we develop the equation (1.42) to separate the horizontal and vertical directions, then we get:

$$\begin{aligned}\int_b^\eta \nabla \cdot \boldsymbol{u}\, dz &= \int_b^\eta \nabla_{2D} \cdot \boldsymbol{u}_{2D}\, dz + \int_b^\eta \frac{\partial w}{\partial z}\, dz = 0 \\ &= \int_b^\eta \nabla_{2D} \cdot \boldsymbol{u}_{2D}\, dz + w|_{z=\eta} - w|_{z=b} = 0\end{aligned} \tag{1.43}$$

The Leibniz's theorem then gives:

$$\int_b^\eta \nabla_{2D} \cdot \boldsymbol{u}_{2D}\, dz = \nabla_{2D} \cdot \left(\int_b^\eta \boldsymbol{u}_{2D}\, dz \right) - \boldsymbol{u}_{2D}|_{z=\eta} \cdot \boldsymbol{\nabla}_{2D}\eta + \boldsymbol{u}_{2D}|_{z=b} \cdot \boldsymbol{\nabla}_{2D} b \tag{1.44}$$

Using the equation (1.44) to rewrite (1.43), we get:

$$\nabla_{2D} \cdot \left(\int_b^\eta \boldsymbol{u}_{2D}\, dz \right) - \boldsymbol{u}_{2D}|_{z=\eta} \cdot \boldsymbol{\nabla}_{2D}\eta + \boldsymbol{u}_{2D}|_{z=b} \cdot \boldsymbol{\nabla}_{2D} b + w|_{z=\eta} - w|_{z=b} = 0 \tag{1.45}$$

Using the kinematic condition at the free-surface (1.19), the equation (1.45) can be rewritten as:

$$\frac{\partial \eta}{\partial t} + \nabla_{2D} \cdot \left(\int_b^\eta \boldsymbol{u}_{2D}\, dz \right) = F_b \tag{1.46}$$

Where the right hand-side of the equation (1.46) is defined as the conditions on the bed boundary:

$$F_b = w|_{z=b} - \boldsymbol{u}_{2D}|_{z=b} \cdot \boldsymbol{\nabla}_{2D} b \tag{1.47}$$

In the case of a fixed, impermeable bed then the condition (1.15) implies that $F_b = 0$. The equation (1.45) simplifies into the following equation on the free-surface elevation:

$$\frac{\partial \eta}{\partial t} + \nabla_{2D} \cdot \left(\int_b^\eta \boldsymbol{u}_{2D}\, dz \right) = 0 \tag{1.48}$$

The boundary conditions in the equation (1.48) are prescribed through the imposition of a water level or the prescription of horizontal velocities at the boundary.
In TELEMAC-3D, the hydrostatic pressure is obtained through the resolution of (1.48), with values for $\boldsymbol{u}_{2D}$ computed according to the advection, diffusion and source terms of the momentum equation.

1.4.3 Solving for the dynamic pressure

The resolution of (1.11) is made complex by the pressure-velocity coupling that prevents the use of sequential schemes. Indeed, as said earlier the pressure is the Lagrangian constraint that enforces the divergence-free constraint [6]. In 1968, Chorin [17] and Temam [93] introduced a projection-method for the approximate resolution of (1.11), which made it possible to solve a sequence of decoupled equations on the velocity and on the pressure at each time-step. Such

an algorithm is very interesting in terms of computational cost. Its theory is based on the Helmholtz-Hodge decomposition, which states that any vector $\boldsymbol{v}$ can be decomposed into the sum of a curl-free vector and a divergence-free vector. Indeed, let us consider two Euclidean vectorial spaces E and F:

$$\begin{aligned} E &= \left\{ \boldsymbol{v} \in \mathscr{C}^1(\Omega, \mathbb{R}^3),\ \boldsymbol{v} \cdot \boldsymbol{n}|_{\partial\Omega_w} = 0 \right\} \\ F &= \left\{ p \in \mathscr{C}^1(\Omega, \mathbb{R}),\ p|_{\partial\Omega_\eta} = 0 \right\} \end{aligned} \tag{1.49}$$

with the following scalar products on E and F:

$$\begin{aligned} \langle \boldsymbol{u}, \boldsymbol{v} \rangle &= \int_\Omega \boldsymbol{u} \cdot \boldsymbol{v}\, d\Omega \\ (p, q) &= \int_\Omega pq\, d\Omega \end{aligned} \tag{1.50}$$

Considering that $\int_\Omega \nabla \cdot (p\boldsymbol{v})\, d\Omega = (p, \nabla \cdot \boldsymbol{v}) + \langle \boldsymbol{\nabla} p, \boldsymbol{v} \rangle$, the following relation is found:

$$(p, \nabla \cdot \boldsymbol{v}) + \langle \boldsymbol{\nabla} p, \boldsymbol{v} \rangle = \oint_{\partial\Omega} p\boldsymbol{v} \cdot \boldsymbol{n} d\Gamma = 0 \tag{1.51}$$

which shows that the gradient and divergence operators are skew-adjoint in these spaces. An important consequence is that the kernel of $(\nabla\cdot)$, denoted by K, is orthogonal to the image of $(-\boldsymbol{\nabla})$. Thus, the following property holds:

$$\forall \boldsymbol{v} \in E, \exists ! \, (\tilde{\boldsymbol{v}}, p) \in K \times F,\ \boldsymbol{v} = \tilde{\boldsymbol{v}} + \boldsymbol{\nabla} p \tag{1.52}$$

Applying the divergence operator to this equation gives:

$$\nabla \cdot \boldsymbol{v} = \nabla^2 p \tag{1.53}$$

where ∇^2 is the Laplacian operator. This then yields:

$$\tilde{\boldsymbol{v}} = \boldsymbol{v} - \boldsymbol{\nabla} \left[\left(\nabla^2 \right)^{-1} \nabla \cdot \boldsymbol{v} \right] \tag{1.54}$$

This shows that the projection operator defined by:

$$\boldsymbol{P} = \left(\boldsymbol{I}_d - \boldsymbol{\nabla} \left[\left(\nabla^2 \right)^{-1} \nabla \cdot \right] \right) \tag{1.55}$$

projects any vector of E onto the space of divergence-free vector fields, provided that the divergence and gradient operators are skew-adjoint.

The idea of projection methods is thus to split the resolution of the momentum equation into two sub-steps. In the first one, an estimation of the velocity is computed, which does not satisfy the incompressibility constraint. Then, the projection operator $\boldsymbol{P}$ is used to project this velocity field onto the space of divergence-free vector fields.

In all the variants of pressure-correction schemes, the estimated velocity is computed based on the viscous and external forces. Some variants also take the pressure gradient of the former time-step into account at this stage. In TELEMAC-3D, the gradient of the hydrostatic pressure is taken into account in the prediction. In the second sub-step, the estimated velocity is corrected through its projection onto the vectorial space K of divergence-free vectors.

Here we only describe the projection scheme used in TELEMAC-3D. For more details about projection schemes see [33] or the summary of that paper in the first Chapter of [51]. In the

first sub-steps, the velocity estimation is based on the viscous and external forces, and on the hydrostatic pressure gradient computed through the resolution of (1.48):

$$\frac{\tilde{\boldsymbol{u}}^{n+1}-\boldsymbol{u}^n}{\delta t} = -(\boldsymbol{u}^n\cdot\boldsymbol{\nabla})\boldsymbol{u}^n - \frac{1}{\rho}\boldsymbol{\nabla}p_h^{n+1} + \frac{1}{\rho}\boldsymbol{\nabla}\cdot\left(\mu\boldsymbol{\nabla}\tilde{\boldsymbol{u}}^{n+1}\right) + \boldsymbol{g} + \tilde{\boldsymbol{F}} \tag{1.56}$$

$\tilde{\boldsymbol{u}}^{n+1}$ is the estimated velocity field, δt is the time-step size and the superscripts n correspond to the time iteration number. The dynamic pressure gradient then intervenes in the last sub-step:

$$\frac{\boldsymbol{u}^{n+1}-\tilde{\boldsymbol{u}}^{n+1}}{\delta t} = -\frac{1}{\rho}\boldsymbol{\nabla}p_d^{n+1} \tag{1.57}$$

which corresponds to the projection of $\tilde{\boldsymbol{u}}^{n+1}$ onto the divergence-free vectorial space. Indeed, the pressure p_d^{n+1} involved in (1.57) was previously computed through a pressure Poisson equation:

$$\nabla^2 p_d^{n+1} = \frac{\rho}{\delta t}\nabla\cdot\tilde{\boldsymbol{u}}^{n+1} \tag{1.58}$$

which corresponds to the enforcement of the incompressibility constraint $\nabla\cdot\boldsymbol{u}^{n+1}=0$ on (1.57). In this scheme, wall boundary conditions are applied to the predicted and final velocity field, respectively. Since the friction part of the viscous term (boundary shear-stress) is treated implicitly, it is necessary to impose the no-slip condition on the predicted velocity:

$$\tilde{\boldsymbol{u}}^{n+1}|_{\Gamma_s\cup\Gamma_b} = 0 \tag{1.59}$$

We consider that there are no moving walls in the simulation, hence the zero value. In order to better estimate velocity gradients close to the solid boundaries, when using a turbulence model a non-zero tangential velocity is assigned, through a wall function. Then, the shear-stress is also prescribed at the solid boundaries through a Neumann boundary condition on the predicted velocity. The only condition prescribed onto the final velocity field is the impermeability condition:

$$\boldsymbol{u}^{n+1}\cdot\boldsymbol{n}|_{\Gamma_s\cup\Gamma_b} = 0 \tag{1.60}$$

On the other hand, pressure boundary conditions are now necessary for the resolution of (1.58), which involves second derivatives of the pressure. The pressure wall boundary condition is obtained by projecting the equation (1.57) onto the normal to the wall, which yields:

$$\boldsymbol{\nabla}p_d^{n+1}\cdot\boldsymbol{n}|_{\Gamma_s\cup\Gamma_b} = -\frac{\rho}{\delta t}\left(\boldsymbol{u}^{n+1}-\tilde{\boldsymbol{u}}^{n+1}\right)\cdot\boldsymbol{n}|_{\Gamma_s\cup\Gamma_b} = 0 \tag{1.61}$$

due to the conditions (1.60). This homogeneous Neumann condition is artificial and was shown to induce a numerical boundary layer which deteriorates the scheme convergence [74].
Note that the correct pressure wall boundary condition is obtained by projecting the momentum equation (2nd line of (1.11)) onto the normal to the wall, which yields:

$$\frac{\partial}{\partial\boldsymbol{n}}\left(\frac{u^2}{2}+\frac{p_d}{\rho}\right)\bigg|_{\Gamma_s\cup\Gamma_b} = \boldsymbol{\nabla}\cdot(\nu\boldsymbol{\nabla}\boldsymbol{u})\cdot\boldsymbol{n}|_{\Gamma_s\cup\Gamma_b} \tag{1.62}$$

At the free-surface and open boundaries the pressure boundary conditions are imposed through:

$$\begin{cases} p_d|_{\Gamma_\eta} = 0 \\ \left.\dfrac{\partial p_d}{\partial\boldsymbol{n}}\right|_{\Gamma_l} = 0 \end{cases} \tag{1.63}$$

1.5 Modelling scalars in the flows

The scalar represents a temperature or a physical quantity contained in water (colouring agent, sediment). If the scalar interacts with the hydrodynamics of the flow, it is called active; if not, it is passive. Active scalars are mostly temperature and salinity, sometimes sediment. Passive scalars can serve for water quality studies.
Physically, the evolution in time of the scalar depends:

- On the current which carries it (advection).
- On diffusion (which is almost always turbulent).
- On source or sink terms.

It is unnecessary to specify here in which unit the scalar is expressed.

1.5.1 Advection–diffusion equation of a scalar

In three dimensions, the scalar equation in a conservative form is as follows:

$$\frac{\partial}{\partial t}(\rho T) + \nabla \cdot (\rho T \boldsymbol{u} + \boldsymbol{q}) = F_{source} \tag{1.64}$$

where F_{source} is the rate of creation and $\boldsymbol{q}$ is the flux due to the molecular diffusion (and, later in this document, turbulence) as:

$$\boldsymbol{q} = -\rho K \boldsymbol{\nabla} T \tag{1.65}$$

K is the molecular diffusion coefficient of the scalar. In the non-conservative form, by using the mass conservation equation, one obtains the equation known as the transport–diffusion equation:

$$\frac{\partial T}{\partial t} + (\boldsymbol{u} \cdot \boldsymbol{\nabla}) T = \nabla \cdot (K \boldsymbol{\nabla} T) + Q \tag{1.66}$$

where Q is a source or sink term.

1.5.2 Boundary conditions

The boundary conditions of the scalars on the walls are:

- No-flux boundary condition :

$$K \frac{\partial T}{\partial n} = 0 \tag{1.67}$$

  where the notation $\partial T / \partial n$ represents $\boldsymbol{\nabla} T \cdot \boldsymbol{n}$, where $\boldsymbol{n}$ is the perpendicular pointing outwards from the boundary.

- Robin boundary condition:

$$K \boldsymbol{\nabla} T \cdot \boldsymbol{n} = aT + b \tag{1.68}$$

1.5.3 Modelling active scalars effects

Many industrial and environmental flows involve fluids which density varies as a function of the temperature or of a scalar concentration like salinity. In many of these flows the Mach number is low (< 0.3) so that they are weakly-compressible (the variations of density due to velocity variations can be neglected as a first approximation). Such flows are subject to buoyancy effects due to gravity, which may generate density currents and stratifications. Besides, in most cases they are turbulent. It is then important for numerical models to represent the buoyancy effects in combination with turbulence effects. where the active scalar is the temperature and where

it is a scalar concentration. non-isothermal flows was chosen in this work, although the model also applies to other active scalars.

The density variations in buoyant flows mainly affect the flow dynamics through the gravity term. In a numerical model one possibility is to let the density vary in the Navier–Stokes equations. Then the expression of the momentum equation is not modified but care must be taken when solving the Navier–Stokes equations that the density is a varying quantity. An alternative approach is to apply the so-called Boussinesq approximation for flows where $\Delta\rho/\rho \ll 1$, which enables the treatment of buoyancy affecting the fluid motion by means of the gravity term only[3]. Then, the fluid density is considered as constant.

The difference in density $\Delta\rho$ is then given by an equation of state which connects the reference density ρ_0 of the fluid to the scalars' concentrations of active scalars. This equation of state has the form:

$$\frac{\Delta\rho}{\rho_0} = f(T_1, ..., T_i, ..., T_n) \tag{1.69}$$

where T_i is the ith active scalar transported. The increase in density is proportional to the differences in concentration, by the intermediary of volumetric expansion coefficients β_i:

$$\frac{\Delta\rho}{\rho_0} = -\sum_i \beta_i \left(T_i - T_i^0\right)_i \tag{1.70}$$

where β_i can be positive (temperature) or negative (salinity, sediment in suspension). T_i^0 is the reference value of the scalar i.

Remark:

In TELEMAC-3D, there are several implemented density laws. By default the reference density is the one of fresh water at 4°C: 999.972kg m^{-3}. The density law 1 corresponds to variations due to the temperature only, with an expansion coefficient $\beta = -7.10^{-6\circ}C^{-1}$ (it is thus a contraction coefficient). The density law 2 corresponds to variations due to the salinity only, with an expansion coefficient of $-0.75.10^{-3} psu^{-1}$. The density law 3 corresponds to variations due to both salinity and temperature (with the same expansion coefficients). The density law 4 corresponds to a user-defined density law and is always used in the case of a sediment concentration effect on the density. For example, to represent the density variations due to the salinity and temperature on sea-water at 10°C (with a reference salinity of 35 psu), the reference density would be 1025 kg m^{-3}, the thermal expansion coefficient $2.10^{-4\circ}C^{-1}$ and the haline expansion coefficient -2.10^{-4}psu^{-1}.

The pressure gradient term in the momentum equations is then developed to the first order:

$$\begin{aligned}\frac{1}{\rho}\boldsymbol{\nabla} p &= \frac{1}{\rho_0}\frac{1}{1+\dfrac{\Delta\rho}{\rho_0}}\boldsymbol{\nabla} p \\ &\approx \frac{1}{\rho_0}\left(1-\frac{\Delta\rho}{\rho_0}\right)\boldsymbol{\nabla} p\end{aligned} \tag{1.71}$$

The second line is a obtained by developing $1/(1+\Delta\rho/\rho_0)$ to the first order in $\Delta\rho/\rho_0$. It is then supposed that the density variations only intervene through the gravity term involved in the

[3]The upper limit for the Boussinesq approximation validity is considered in [96] to be $\Delta\rho/\rho < 0.1$.

hydrostatic pressure. The latter is defined by:

$$\begin{cases} p = p_h + p_d \\ \dfrac{\partial p_h}{\partial z} = -\rho g \end{cases} \tag{1.72}$$

or even, by writing $\rho = \rho_0 + \Delta\rho$:

$$\frac{\partial p_h}{\partial z} = -\rho_0 g \left(1 + \frac{\Delta\rho}{\rho_0}\right) \tag{1.73}$$

from which one obtains the expression of the hydrostatic pressure at elevation z:

$$p_h = p_{atm} + \rho_0 g\,(\eta - z) + \rho_0 g \int_z^\eta \frac{\Delta\rho}{\rho_0}\, dz' \tag{1.74}$$

The equation (1.71) thus becomes:

$$\frac{1}{\rho}\boldsymbol{\nabla} p = \frac{1}{\rho_0}\left(1 - \frac{\Delta\rho}{\rho_0}\right)\boldsymbol{\nabla} p_h + \frac{1}{\rho_0}\boldsymbol{\nabla} p_d \tag{1.75}$$

and replacing p_h by its expression:

$$\begin{cases} \dfrac{1}{\rho}\dfrac{\partial p}{\partial x} \approx g\left(1 - \dfrac{\Delta\rho}{\rho_0}\right)\dfrac{\partial \eta}{\partial x} + g\dfrac{\partial}{\partial x}\left(\displaystyle\int_z^\eta \frac{\Delta\rho}{\rho_0}\, dz\right) + \frac{1}{\rho_0}\frac{\partial p_d}{\partial x} \\ \dfrac{1}{\rho}\dfrac{\partial p}{\partial y} \approx g\left(1 - \dfrac{\Delta\rho}{\rho_0}\right)\dfrac{\partial \eta}{\partial y} + g\dfrac{\partial}{\partial y}\left(\displaystyle\int_z^\eta \frac{\Delta\rho}{\rho_0}\, dz\right) + \frac{1}{\rho_0}\frac{\partial p_d}{\partial y} \\ \dfrac{1}{\rho}\dfrac{\partial p}{\partial z} \approx -g + \dfrac{1}{\rho_0}\dfrac{\partial p_d}{\partial z} \end{cases} \tag{1.76}$$

In the third line, the terms involving $(\Delta\rho/\rho_0)^2$ were neglected (second order terms).

The terms of (1.76) that involve the quantity $\Delta\rho/\rho_0$ are grouped with the momentum buoyancy forces in the source term $\boldsymbol{F}$ of the equation (1.7); therefore:

$$\begin{cases} \tilde{\boldsymbol{F}}_{2D} = \boldsymbol{F}_{2D} - g\dfrac{\Delta\rho}{\rho_0}\boldsymbol{\nabla}_{2D}\eta + g\boldsymbol{\nabla}_{2D}\left(\displaystyle\int_z^\eta \frac{\Delta\rho}{\rho_0}\, dz\right) \\ \tilde{F}_z = F_z \end{cases} \tag{1.77}$$

Possible improvement:

For non-hydrostatic flows, the Boussinesq approximation consists of considering that buoyancy only affects the fluid motion by means of the gravity term. By multiplying the momentum equation by ρ and considering that ρ is constant, equal to the reference density ρ_0, except in the gravity term. The momentum equation then becomes:

$$\rho_0 \frac{\partial \boldsymbol{u}}{\partial t} + \rho_0 (\boldsymbol{u} \cdot \boldsymbol{\nabla}) \boldsymbol{u} = -\boldsymbol{\nabla} p + \boldsymbol{\nabla} \cdot (\mu_E \boldsymbol{\nabla} \boldsymbol{u}) + \rho \boldsymbol{g} + \rho_0 \boldsymbol{F} \tag{1.78}$$

with ρ written as: $\rho = \rho_0 + \Delta\rho$, the above equation divided by ρ_0 becomes:

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \boldsymbol{\nabla}) \boldsymbol{u} = -\frac{1}{\rho_0} \boldsymbol{\nabla} p + \frac{1}{\rho_0} \boldsymbol{\nabla} \cdot (\mu_E \boldsymbol{\nabla} \boldsymbol{u}) + \boldsymbol{g} + \frac{\Delta\rho}{\rho_0} \boldsymbol{g} + \boldsymbol{F} \tag{1.79}$$

Then, the buoyancy terms only affect the z-equation and considering the law (1.70) for the density variations, the source terms in the momentum equation become:

$$\begin{cases} \tilde{\boldsymbol{F}}_{2D} = \boldsymbol{F}_{2D} \\ \tilde{F}_z = F_z - \sum_i \beta_i \left(T_i - T_i^0 \right)_i \boldsymbol{g} \end{cases} \tag{1.80}$$

instead of the ones given by the equation (1.77), where the buoyancy effects are reported onto the x and y equations. This alternative formulation would simplify the treatment of the buoyancy terms compared to the current model. Some preliminary tests on the lock-exchange case were done, where no significant difference was observed between the two formulations.

1.5.4 Modelling heat exchanges with the atmosphere

Two models to compute heat exchange between water and atmosphere are available in TELEMAC-3D: a linearised formula of the balance of heat exchange fluxes at the free surface that is first described below and a model that computes the complete balance of exchanged fluxes then described.

In the first model, the thermal power liberated into the atmosphere per surface unit, denoted by P, is supposed to be proportional to $T - T_{air}$ where T is the temperature of water on the surface and T_{air} that of air; one gets $P = A(T - T_{air})$, where A is the exchange coefficient in W/m^2/°C. The heat flux leaving the liquid domain is written as:

$$\Phi = -\rho C_P K \boldsymbol{\nabla} T . \boldsymbol{n} = -\rho C_P K \frac{\partial T}{\partial z} \tag{1.81}$$

with $C_P = 4180$ J/kg/°C. Recall that K (m^{2}s^{-1} is the coefficient of molecular heat diffusion in water.
By equating the two formulations, one deduces the boundary conditions:

$$K \frac{\partial T}{\partial z} = -\frac{A}{\rho C_p} (T - T_{air}) \tag{1.82}$$

The coefficient A should include phenomena like radiation, convection of air in contact with water and latent heat produced by evaporation of water. Sweers [91] expresses the coefficient A, in W/m^2/°C, according to the temperature of water T and the wind velocity u_{wind} measured at the point under consideration, in m/s:

$$A = (4.48 + 0.049T) + 2021.5\,b\,(1 + u_{wind})\,(1.12 + 0.018\,T + 0.00158\,T^2) \tag{1.83}$$

The parameter b varies depending on the location. Its average value on the shores of the Channel and of the Atlantic is 0.0025. It is higher in the Mediterranean, where it reaches approximately 0.0035.

A much more elaborated model is implemented in TELEMAC-3D. This module calculates the complete balance of exchanged fluxes involved:

- the solar radiation RS,
- the atmospheric radiation RA,
- the water radiation RE,
- the latent heat due to evaporation CE,
- the sensitive heat of convective origin CV.

Whereas the long wave radiation (atmospheric radiation RA) is absorbed in the first centimetres of the water column, the short wave radiation (solar radiation RS) penetrates the water column. Evaporation is calculated by TELEMAC-3D. Heat exchanges with the atmosphere are taken into account at two levels:

- the complete balance of exchanged fluxes at the free surface is calculated at the boundary conditions of the surface for temperature:

$$K \left.\frac{\partial T}{\partial z}\right|_{z=\eta} = \frac{RA - RE - CE - CV}{\rho_{water} C_p}, \tag{1.84}$$

- the penetration of the solar radiation in the water column is taken into account in the source term of the advection-diffusion equation of the temperature:

$$S = \frac{1}{\rho_{water} C_p} \frac{\partial Q(z, RS)}{\partial z}, \tag{1.85}$$

where ρ_{water} (kg.m^{-3}) is the water density and the function $Q(z, RS)$ is the residual solar radiation at the elevation z.

The atmosphere emits a long wave radiation corresponding to the reemission of a part of direct solar energy. It emits like a black body. Clouds and albedo at the surface determine the atmospheric radiation RA penetrating the water:

$$RA = (1 - alb_{lw}) \varepsilon_{air} \sigma (T_{air} + 273.15)^4 (1 + Nua.C^2), \tag{1.86}$$

where:

- alb_{lw} = 0.03 is the albedo of water for long radiative waves (common value used in the literature [41], [35]),
- $\varepsilon_{air} = 0.937.10^{-5} (T_{air} + 273.15)^2$ is the air emissivity,
- $\sigma = 5.67.10^{-8}$ W.m^{-2}.K^{-4} is Stefan-Boltzmann's constant,

- C is the nebulosity (tenths). Some meteorological services like Météo France provide these data in octas that then need to be converted into tenths,
- Nua (dimensionless) is a parameter that characterises the type of cloud. In practise, it is difficult to know the type of cloud during the period of simulation and a mean value of 0.17 is often used [8], [41] but other choices are possible (see the table 1.1).

Table 1.1: Coefficient Nua characterising the type of cloud

Type of cloud	Cirrus	Cirro Stratus	Alto Cumulus	Alto Stratus	Cumulus	Stratus
Nua	0.04	0.08	0.17	0.20	0.20	0.24

The water radiation RE is calculated considering that the water surface emits like a grey body:

$$RE = \varepsilon_{water}\sigma(T_{surface} + 273.15)^4, \tag{1.87}$$

with $\varepsilon_{water} = 0.97$ (dimensionless) is water emissivity, $\sigma = 5.67.10^{-8}$ W.m^{-2}.K^{-4} is Stefan-Boltzmann's constant, $T_{surface}$ (°C) is the water temperature at the surface. The latent heat flux CE comes from evaporation. It is calculated by:

$$CE = \rho_{air} Le(T_{surface}) f(u_2)(HS_{water} - HS_{air}), \tag{1.88}$$

where:

- ρ_{air} is the air density,
- $Le(T_{surface}) = (2500.9 - 2.365T_{surface}).10^3$ (J.kg^{-1}) is the latent heat of evaporation at the surface temperature,
- u_2 is the wind velocity at 2 m high,
- $f(u_2)$ is a function of the wind velocity at 2 m high u_2,
- HS_{water} is the specific humidity of the saturated air at the surface temperature,
- HS_{air} is the specific humidity of air.

The specific humidity of the saturated air HS_{water} is calculated by:

$$HS_{water} = \frac{0.622 Pvs_{water}}{P_{atm} - 0.378 Pvs_{water}} \tag{1.89}$$

The specific humidity of air HS_{air} is calculated by:

$$HS_{air} = \frac{0.622 \dfrac{H_{rel}}{100} Pvs_{air}}{P_{atm} - 0.378 \dfrac{H_{rel}}{100} Pvs_{air}} \tag{1.90}$$

The saturated pressure are calculated by Magnus-Tetens's formula:

$$Pvs = \exp\left(2.3026\left(\frac{7.5T}{T + 237.3} + 0.7858\right)\right) \tag{1.91}$$

The surface evaporation flow rate is then calculated by:

$$Deb = \frac{CE}{\rho_{water} L_e}, \tag{1.92}$$

The sensitive heat flux CV corresponds to the heat transfer by convection:

$$CV = \rho_{air} Cp_{air} f(u_2)(T_{surface} - T_{air}), \tag{1.93}$$

where:

- $Cp_{air} = 1005$ J.kg^{-1}.°C^{-1} is the air specific heat,
- $T_{surface}$ (°C) is the water temperature at the surface.

The solar flux penetrating the water RS is calculated with:

- the solar radiation reaching the surface with a clear sky. It depends on time and the location of the site. Perrin de Brichambaut's method [64], [65] is used,
- cloud coverage: a corrective term depending on the nebulosity is applied (Berliand's formulae [9], [10]),
- albedo that enables to compute the effective part of solar radiation penetrating the water.

Solar radiation RS is written:

$$RS = AA.\sin(h)^{BB}(1 - 0.65C^2)(1 - Alb), \tag{1.94}$$

where:

- AA (W.m^{-2}) and BB (dimensionless) are coefficients related to luminosity and sky colour and have to be chosen with respect to the considered area. Three possibilities are suggested (see the table 1.2) but the mean values are used by default in TELEMAC-3D,
- h is the angular height of the sun (rad), which depends on the latitude and longitude of the location and changes with day and hour,
- Alb albedo of water for short waves that may vary every month [63].

Table 1.2: Values of the coefficients AA and BB for the calculation of solar radiation related to luminosity and sky colour

Type of sky	AA (W.m^{-2})	BB
Very pure sky (type 1)	1130	1.15
Mean pure sky (type 2)	1080	1.22
Industrial area (type 3)	995	1.25

The penetration of solar radiation in the water column is taken into account in the advection-diffusion equation of temperature by introducing a source term. It depends on the water turbidity and is linked to dissolved and suspended particles (mineral or organic). Two laws to model the solar radiation penetration are suggested and described below:

- the first one uses two exponential laws that may be difficult to calibrate and require an estimation of the type of water from a turbidity,

- the second one uses the *in situ* measurements of Secchi length and is recommended if available.

In the first case, the source term reads:

$$Q(z,RS) = RS.\left(R\exp\left(-\frac{z_s - z}{\zeta_1}\right) + (1-R)\exp\left(-\frac{z_s - z}{\zeta_2}\right)\right), \tag{1.95}$$

with RS (W.m^{-2}) is the solar radiation penetration the water, z_s (m) the free surface elevation, ζ_1 and ζ_2 (m) are attenuation lengths and R is a dimensionless coefficient. This equation means the penetration of solar radiation in the water column with a law of double exponentials. It takes into account a selective absorption of the solar spectrum by water, separating the radiation quickly attenuated in the water column and the one penetrating deeply. In the second case, the source term is written from Beer-Lambert's law:

$$Q(z,RS) = RS.\exp\left(-\frac{1.7(z_s - z)}{L_S}\right) \tag{1.96}$$

with RS (W.m^{-2}) is also the solar radiation penetration the water, z_s (m) the free surface elevation, L_s (m) is the Secchi length provided by *in situ* measurements. The coefficients ζ_1, ζ_2 and R are characteristics of optical properties of water, have to be adjusted to the studied water. Examples of combination are suggested in the table 1.3. This classification has been done for coastal and marine flows and is widely used in numerical modelling.

Table 1.3: Examples of values of the coefficients R, ζ_1, ζ_2 with respect to water turbidity ([61], [44], [46])

Type of water	R	ζ_1 (m)	ζ_2 (m)
Very clear (Kraus)	0.40	5	40
Type I (Jerlov)	0.58	0.35	23
Type IA (Jerlov)	0.62	0.6	20
Type IB (Jerlov)	0.67	1.0	17
Type II (Jerlov)	0.77	1.5	14
Very turbid - Type III (Jerlov)	0.78	1.4	7.9

The formulae require to provide additional data (wind magnitude and direction, air temperature, atmospheric pressure, relative humidity, nebulosity and rainfall). The site of a study may not be equipped for local wind measurements and these kinds of data are available at a different location (even far from the studied site). A wind function is then used, that is a linear function with a single coefficient of calibration b:

$$f(u_2) = b(1+u_2) \tag{1.97}$$

with u_2 that is the wind velocity at 2 m high. As for the linearised formula (1.83), this parameter b varies depending on location. Its average value on the shores of the Channel and of the Atlantic is 0.0025. It is higher in the Mediterranean, where it reaches approximately 0.0035 and can be around 0.0017 [85].
Wind data are often provided at 10 m high whereas latent and sensible fluxes need wind velocities at 2 m high. Velocities at 2 m high need to be calculated from wind velocities at 10 m high. A logarithmic wind velocity vertical profile is then considered, with a roughness length of $z_0 = 0.0002$ m. This leads to $u_2 = 0.85u_{10}$. This value of 0.85 (or the roughness length) may

be changed if needed.

$$u_2 = \frac{u_{10} \log\left(\frac{2}{z_0}\right)}{\log\left(\frac{10}{z_0}\right)} \tag{1.98}$$

1.6 Modelling turbulence and dispersion

When the Reynolds number, defined as $Re = UL/\nu$, where U and L are a characteristic velocity and length scale of the flow, becomes greater than around 2000, physical quantities like the velocity and the pressure show rapid variations in time and space. The value 2000 is actually an order of magnitude, valid for canals, and varies greatly according to the flows. It also depends on the choice of characteristic quantities U and L. For large Reynolds numbers, *i.e.* for a majority of applications, regular or laminar solutions, are no longer stable and are replaced by chaotic solutions highly variable in space and time, at different time and space scales. These fluctuations are not described in TELEMAC: only the average values are of interest in most industrial or environmental problems. These averages are done over the same experiment conducted a certain number of times (definition of the Reynolds average), but for the sake of commodity in the experimental setup they may be time averages in case of a steady flow, or space averages in case of a homogeneous flow. Unfortunately, due to the non-linearity of advection terms, the averaged flow is no longer the solution for the Navier–Stokes equations and the equations of this averaged flow are called "Reynolds-Averaged Navier–Stokes equations".

1.6.1 Reynolds-Averaged Navier–Stokes models

While a turbulent field is highly variable in space and time, it presents a smooth and less variable mean. This observation led to the idea of applying a statistical mean operator to the equations. For N occurrences of a flow the Reynolds average applied to a field A is defined as:

$$\overline{A} = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} A^i \tag{1.99}$$

This newly defined field $\overline{A}$ is called the mean field and is a flow feature: it is not sensitive to small perturbations of the initial conditions. Considering an instantaneous field A (dropping the superscript i which referred to the instance number), it is then written as:

$$A = \overline{A} + A' \tag{1.100}$$

where A' is a fluctuating field which is different between two instances of the flow. The mean field, contrarily to the instantaneous one, is smooth and reproducible. It may happen to be constant over time or invariant along a direction, while the instantaneous field is always time-variable and three-dimensional. Thus, an approach for modelling turbulent flows is to try to simulate the mean fields. Applying the Reynolds average operator to the Navier–Stokes equations (1.11) yields the Reynolds-Averaged Navier–Stokes (RANS) equations for incompressible flows:

$$\begin{cases} \nabla \cdot \overline{\boldsymbol{u}} = 0 \\ \dfrac{\partial \overline{\boldsymbol{u}}}{\partial t} + (\overline{\boldsymbol{u}} \cdot \boldsymbol{\nabla})\overline{\boldsymbol{u}} = -\dfrac{1}{\rho}\boldsymbol{\nabla}\overline{p} + \dfrac{1}{\rho}\boldsymbol{\nabla} \cdot (\mu \boldsymbol{\nabla}\overline{\boldsymbol{u}}) + \boldsymbol{g} - \boldsymbol{\nabla} \cdot \boldsymbol{R} + \tilde{\boldsymbol{F}} \end{cases} \tag{1.101}$$

The application of the Reynolds average operator to the non-linear convection term in the momentum equation (2nd term in the left-hand side of (1.7)) led to an additional stress tensor $\boldsymbol{R}$

called the Reynolds stress tensor and defined as:

$$\boldsymbol{R} = \overline{\boldsymbol{u}' \otimes \boldsymbol{u}'} \tag{1.102}$$

A transport equation on the components of the Reynolds stress tensor (called Reynolds stresses) can be obtained by subtracting (1.101) to (1.11), tensorially multiplying the result by $\boldsymbol{u}'$ and then applying the Reynolds average operator. Though, this does not close the problem since this transport equation involves new unknown terms, in particular third order moments of $\boldsymbol{u}'$. It is thus necessary to find a heuristic closure law in order to solve the system.

A first-order closure of the system consists in writing a closure law linking the second order moments of the fluctuating velocity to its first order moments, without solving the transport equation on $\boldsymbol{R}$. In other words, it relies on the construction of a behaviour law that expresses $\boldsymbol{R}$ as a function of $\overline{\boldsymbol{u}}$. Such a behaviour law was proposed by Boussinesq through a model similar to the Stokes model (1.8) for the Cauchy stress, which aims at representing the diffusion and dissipation effects of the turbulent eddies through an eddy viscosity, as well as the additional "pressure" they induce in the flow. The Boussinesq model reads:

$$\boldsymbol{R} = \frac{2}{3} k \boldsymbol{I}_d - 2\nu_T \boldsymbol{S} \tag{1.103}$$

where $\boldsymbol{S}$ is the mean strain rate tensor:

$$\boldsymbol{S} \doteq \overline{\boldsymbol{s}} = \frac{1}{2}\left(\boldsymbol{\nabla}\overline{\boldsymbol{u}} + (\boldsymbol{\nabla}\overline{\boldsymbol{u}})^T\right) \tag{1.104}$$

k is the kinetic energy of the fluctuating velocity field per unit mass (called turbulent kinetic energy):

$$k = \frac{1}{2} tr\boldsymbol{R} = \frac{1}{2}\overline{|\boldsymbol{u}'|^2} \tag{1.105}$$

and ν_T is the eddy viscosity. With this model, and neglecting again the term $\boldsymbol{\nabla} \cdot \left(\mu_E \boldsymbol{\nabla} \boldsymbol{u}^T\right)$, the RANS equations are written as:

$$\begin{cases} \nabla \cdot \overline{\boldsymbol{u}} = 0 \\ \dfrac{\partial \overline{\boldsymbol{u}}}{\partial t} + (\overline{\boldsymbol{u}} \cdot \boldsymbol{\nabla})\overline{\boldsymbol{u}} = -\dfrac{1}{\rho}\boldsymbol{\nabla}\tilde{p} + \dfrac{1}{\rho}\boldsymbol{\nabla} \cdot [\mu_E \boldsymbol{\nabla} \boldsymbol{u}] + \boldsymbol{g} + \tilde{\boldsymbol{F}} \end{cases} \tag{1.106}$$

where $\tilde{p} = \overline{p} + 2/3\rho k$ and $\mu_E = \mu + \mu_T$ is an effective viscosity, with $\mu_T = \rho\nu_T$. The remaining task is then to build a model for the eddy viscosity computation.

A variety of models is available to compute the eddy viscosity, from the simplest and coarsest (zero equation or one equation models) to more complex ones like the $k-\varepsilon$ model or even Reynolds-stress models. Turbulence models with zero, one, or two equations are all based on a hypothesis of isotropic turbulence. However, in certain applications a tensor of isotropic turbulent viscosity is not adapted. For example, when the mesh is hundreds of metres in the horizontal direction, but of the order of one meter in the vertical direction, or when dispersion

along the vertical is taken into account.

Remark:

In the scalar equation (1.66), an additional correlation term appears: $-\overline{u_i' T'}$. The Boussinesq model is then expressed by the formula:

$$-\overline{u_i' T'} = \frac{\nu_T}{Pr_T} \frac{\partial T}{\partial x_i} \tag{1.107}$$

where Pr_T is the turbulent Prandtl number (if T is the temperature; with salinity, it would be the turbulent Schmidt number, σ_T). The turbulent Prandtl and Schmidt numbers range generally between 0.8 and 1.2 depending on the flow, but other values are sometimes favoured, particularly in the presence of sediments. In TELEMAC-3D they are supposed equal to each other and with a constant value in the flow. This value can be chosen by the user: by default it is taken equal to 1. The transport–diffusion equation for a scalar, (1.66), then becomes:

$$\frac{\partial T}{\partial t} + (\boldsymbol{u} \cdot \boldsymbol{\nabla}) T = \nabla \cdot (K_E \boldsymbol{\nabla} T) + Q \tag{1.108}$$

where:

$$K_E = K + K_T \tag{1.109}$$

with K_T defined by:

$$K_T = \frac{\nu_T}{Pr_T} \tag{1.110}$$

We shall now examine, in increasing complexity, the various turbulence models available in TELEMAC-3D. The different models presented later will differ by the value ν_T that they provide. In some cases, in order to take into account the effect of dispersion, turbulent viscosity will be considered as an anisotropic quantity, with different values along each direction. The coefficient ν_T is dimensionally the product of a velocity and a length. It is considered roughly to represent the product of velocity of an eddy with the distance along which it mixes momentum. This interpretation demonstrates the anisotropy of the process. On the other hand, in the Boussinesq model, the Reynolds tensor is always analogous to the deformation tensor, with an isotropic behaviour as per elastic material. After presenting "zero-equation" models, we shall detail the most commonly used two-equation model: the k–ε model, and finally specify the values of shear stress near walls due to turbulence.

1.6.2 Zero-equation models

These very simple models assume that turbulent viscosity is constant, or directly depend on known or easily calculable parameters.

Constant viscosity

Constant viscosity is sufficient when flow is governed by the pressure gradient and by advection, in the tide flow regime for example, and especially for modelling oceanic circulation on a large scale. In 2D, this constant viscosity should include dispersion. The values of ν_T provided in the literature (see [30]) extend from 0.12 m^2/s (transversal diffusion in the Mackenzie River) to 1500 m^2/s (longitudinal diffusion in the Missouri River).

Nezu and Nakagawa model (56)

This model, published in 1993, gets its inspiration from the expression $\nu_T = \kappa u_* z$ in the boundary layer above the bed. But it also takes into account that turbulent viscosity decreases to zero towards the free surface (assuming the absence of wind). Nezu and Nakagawa [56] then propose:

$$\nu_T = \kappa u_* z \left(1 - \frac{z}{h}\right) \tag{1.111}$$

This model is very useful for representing diffusion along the vertical.

Mixing length models

Standard mixing length models:

this model, proposed by Prandtl [69] in 1925 gives the value of the viscosity coefficient as:

$$\nu_T = L_m^2 \sqrt{2\boldsymbol{S} : \boldsymbol{S}} \tag{1.112}$$

where $\boldsymbol{S}$ is the strain rate tensor of average motion, with:

$$S_{ij} = \frac{1}{2}\left(\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i}\right) \tag{1.113}$$

We recall that given two tensors $\boldsymbol{A}$ and $\boldsymbol{B}$, $\boldsymbol{A} : \boldsymbol{B} = tr(\boldsymbol{A}\boldsymbol{B}^T) = A_{ij}B_{ij}$ with the Einstein notation, tr being the trace operator. In the equation (1.112), L_m is the "mixing length" parameter equal to κy at a distance y from the wall, with $\kappa = 0.41$ (von Karman constant), till the size of eddies is no longer influenced by the wall and remains constant (in the case of flow between two walls, the limit is situated at 1/10th of the space between the walls). This model is used to describe the velocity profile in the vicinity of a wall. It is obtained by a hypothesis of equilibrium turbulence (for details see the section 1.6.3).

Mixing length models along the vertical:

still according to the Prandtl model, applied this time along the vertical, the model provides vertical viscosities to be used for velocities or scalars:

$$\nu_z = f(Ri)\, L_m^2 \sqrt{\left(\frac{\partial u}{\partial z}\right)^2 + \left(\frac{\partial v}{\partial z}\right)^2} \tag{1.114}$$

and:

$$\nu_{zT} = f_T(Ri)\, L_m^2 \sqrt{\left(\frac{\partial u}{\partial z}\right)^2 + \left(\frac{\partial v}{\partial z}\right)^2} \tag{1.115}$$

where:

$$Ri = -\frac{g}{\rho_0} \frac{\dfrac{\partial \rho}{\partial z}}{\left(\dfrac{\partial u}{\partial z}\right)^2 + \left(\dfrac{\partial v}{\partial z}\right)^2} \tag{1.116}$$

is the Richardson number. f and f_T are the damping functions of the velocity and scalar which help to refine the model and adapt it to special cases, particularly for stratifications. If the stratification is stable, which corresponds to a positive Richardson number, f is near zero. If the stratification is unstable, which corresponds to a negative Richardson number, f takes a value between 1 and ∞, depending on the authors. There are many expressions for damping functions,

which always show a tendency to decrease with Ri. One way of obtaining an expression is to start with the Reynolds tensor equations and assume a local equilibrium of turbulence, so certain terms can be ignored. The remaining terms are then modelled. We get an algebraic system dependent on the Richardson number. By choosing the models described in reference [88], we arrive at the graph in the figure 1.2 for damping functions of velocity and temperature. The function defined for temperature does not exceed 1 for $R_i = 0$ but it does for the velocity. So there is a turbulent Prandtl number different from 1 in a neutral situation ($Ri = 0$). In the case of a scalar, different from a temperature, other damping functions can be used. For example:

$$f(Ri) = (1 + aRi)^b \tag{1.117}$$

a and b being two constants depending on the nature of the scalar. This is a totally empirical law, but it is sometimes used in the presence of sediment.

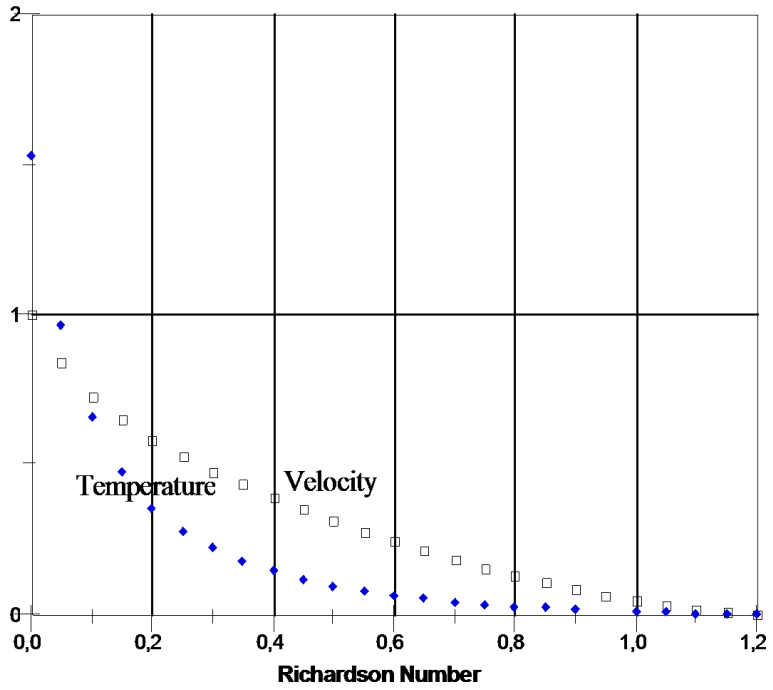

Figure 1.2: Damping functions on the velocity and scalar

The mixing length L_m can be modelled in different ways:
Classical Prandtl model (see [82]):
$L_m = \kappa z$ if $z \leq 0.2h$ (here, z is the distance to the bed and h the water depth).
$L_m = 0.2\kappa h$ if $z \geq 0.2h$.
Nezu and Nakagawa model (see [56]):

$$L_m = \kappa z \sqrt{1 - \frac{z}{h}}$$

In this case the mixing length vanishes at the free surface. This model can give a logarithmic velocity profile from bed to free surface.
Quetin model (1977, see [72]):

$$L_m = \frac{1}{\frac{1}{\kappa z} + \frac{1}{0.65d}}$$

where d is the distance to the free surface.
Tsanis model (see [95]):
$L_m = \kappa z$ if $z \leq 0.2h$ (here, z is the distance to the bed and h the water depth).
$L_m = 0.2\kappa h$ if $z \geq 0.2h$ and $z \leq 0.8h$.
$L_m = \kappa d$ if $z \geq 0.8h$ (d is the distance to the free surface).
In the production of turbulence expression, the horizontal gradients of u and v as well as the gradients of w are assumed negligible compared to the vertical gradients. The term:

$$\sqrt{\left(\frac{\partial u}{\partial z}\right)^2 + \left(\frac{\partial v}{\partial z}\right)^2}$$

is thus only an approximation of $\sqrt{2S_{ij}S_{ij}}$. For a calculation in the vicinity of a structure (quay, spikes, etc.), the gradients overlooked here can become important. However, their presence in the vertical viscosity expression would not be of much help to perform an efficient calculation around a structure. A mixing length model for horizontal viscosity would also become necessary. Hence a precise computation of flow around structures is not possible with this turbulence modelling.

Smagorinski model

The Smagorinski model (see [87]) has a meaning only in numerical modelling and is based on the mixing length model. It belongs to the category of sub-grid turbulence models. The principle is as follows: turbulence is the solution of the Navier–Stokes equations. It would naturally appear in the numerical solutions if the size of finite elements allowed the reproduction of all mechanisms including the viscous dissipation of very small vortices. Only in the formation of smaller vortices, where turbulence is inhibited by the mesh, does modelling take place in a numerical simulation. Smagorinski's idea is to add to the molecular viscosity a turbulent viscosity deduced from a mixing length model. This mixing length corresponds to the size of the vortices smaller than that of the mesh size. We simply arrive at the following formulation:

$$\nu_T = C_s^2 \Delta^2 \sqrt{2S_{ij}S_{ij}} \tag{1.118}$$

where C_s is a dimensionless coefficient to be calibrated and Δ the mesh size derived in 2D or 3D from the surface or from the volume of the elements. The values of C_s range from 0.1 (flow in a canal) to 0.2 (isotropic turbulence). This model is best understood in 3D. In 2D it does not take dispersion into account.

1.6.3 Two-equations k–ε model

This model is based on the calculation of physical quantities representing turbulence in the flow.

k and ε equations:

The transport equation on k is obtained by taking the trace of the transport equation on the Reynolds stresses, and reads:

$$\frac{\partial k}{\partial t} + \overline{\boldsymbol{u}} \cdot \boldsymbol{\nabla} k = \mathbb{P} + \nabla \cdot \boldsymbol{Q}^k - \varepsilon + \mathbb{G} \tag{1.119}$$

where $\mathbb{P}$ is a production term whose definition is $\mathbb{P} = -\boldsymbol{R} : \boldsymbol{S}$. By using the Boussinesq model (1.103), it can be written as[4]:

$$\mathbb{P} = \nu_T S^2 \tag{1.120}$$

[4] Strictly speaking, the term $-2/3k\nabla \cdot \boldsymbol{u}$ should be taken into account for flows that are not truly incompressible.

with S the scalar mean rate-of-strain defined as:

$$S = \sqrt{2\boldsymbol{S} : \boldsymbol{S}} \tag{1.121}$$

$\mathbb{G}$ is a source term due to the forces of gravity in the case of temperature gradients. It comes from the derivation of the Reynolds stress equation under the form: $\mathbb{G} = -\beta \boldsymbol{g} \cdot \overline{\boldsymbol{u}' T'}$, and thus also appears in the k equation. This term is modelled through a turbulent thermal diffusivity model, which yields:

$$\mathbb{G} = -\beta \frac{\nu_T}{Pr_T} g \frac{\partial \overline{T}}{\partial z} \tag{1.122}$$

with $\beta = -1/\rho(\partial\rho/\partial\overline{T})$, Pr_T being the turbulent Prandtl number.In (1.119), $\boldsymbol{Q}^k$ is the flux of k, which represents a transport of kinetic and potential energy by the eddies and the molecular viscosity. The flux of k can be modelled through a diffusion term:

$$\nabla \cdot \boldsymbol{Q}^k = \frac{1}{\rho} \nabla \cdot (\mu_k \boldsymbol{\nabla} k) \tag{1.123}$$

where μ_k is defined as $\mu_k = \mu + \mu_T/\sigma_k$, σ_k being a model constant. On the other hand, ε corresponds to a dissipation of turbulent kinetic energy (transformed into thermal energy) due to the viscosity, defined as:

$$\varepsilon = \nu \sum_i \sum_j \overline{\left(\frac{\partial u_i'}{\partial x_j} \right)^2} \tag{1.124}$$

where $\partial u_i'/\partial x_j$ denotes the derivative of the ith component of $\boldsymbol{u}'$ with respect to the jth coordinate. Although it is possible to write an exact transport equation on ε, the latter includes complex terms that cannot be explicitly calculated. This is why, in the $k-\varepsilon$ model ε is computed through a simplified equation that reproduces the k equation:

$$\frac{\partial \varepsilon}{\partial t} + \overline{\boldsymbol{u}} \cdot \boldsymbol{\nabla} \varepsilon = C_{1\varepsilon} \frac{\varepsilon}{k} (\mathbb{P} + (1 - C_{3\varepsilon}) \mathbb{G}) - C_{2\varepsilon} \frac{\varepsilon^2}{k} + \frac{1}{\rho} \nabla \cdot (\mu_\varepsilon \boldsymbol{\nabla} \varepsilon) \tag{1.125}$$

where μ_ε is defined as: $\mu_\varepsilon = \mu + \mu_T/\sigma_\varepsilon$, σ_ε being a constant. $C_{1\varepsilon}$ and $C_{2\varepsilon}$ are also constants of the model (see the table 1.4). The equation (1.125) has no theoretical background but relies on empirical considerations. The constant $C_{3\varepsilon}$ was introduced in order to represent the fact that stable stratifications weaken turbulence. It is thus taken as equal to one if $\mathbb{G}$ is negative, and zero otherwise. The term ε/k is the frequency of the large eddies, it ensures the equation homogeneity. The ε source terms are supposed proportional to the ones in the k equation.

Note that the $k-\varepsilon$ model is not accurate concerning non-inertial and streamline curvature effects, as well as severe deviation from local equilibrium. Besides, it was shown that computing the production term through (1.120) leads to over-estimations of k and thus of ν_T. In order to avoid this, Guimet & Laurence [34] proposed to restrict the production term to a linear behaviour for high values of the rate-of-strain, obtained from the equilibrium between $\mathbb{P}$ and ε for fully developed homogeneous turbulence. This yields a linear-quadratic model for the production[5]:

$$\mathbb{P} = \min\left(\sqrt{C_\mu} k S, \nu_T S^2\right) \tag{1.126}$$

Another issue is that the size of the large eddies, given by $L_t \sim k^{3/2}/\varepsilon$, may be predicted arbitrarily large, which is not physical since L_t should be bounded at least by L, the characteristic

[5]This essentially recovers the SST modification [55], although it does not include a low-Reynolds treatment.

size of the flow. To remedy this issue, Yap [100] proposed a modification of the $C_{2\varepsilon}$ coefficient in order to increase the dissipation of turbulent kinetic energy:

$$C_{2\varepsilon,Y} = \max\left(C_{2\varepsilon} - \max\left[0, 0.83\left(\frac{L_t}{L} - 1\right)\left(\frac{L_t}{L}\right)^2\right], 0\right) \tag{1.127}$$

Remark:

By default the Yap treatment for $C_{2\varepsilon}$ is not activated in TELEMAC-3D. This choice of option is hard-coded in the routine CSTKEP.f at the line 183.

The Kolmogorov dimensional analysis [45] leads to a definition of the eddy viscosity as a function of k and ε, which corresponds to the fact that the large turbulent eddies are the ones that most interact with the mean flow. ν_T is thus written as proportional to the length scale of the large eddies, $L_t \sim k^{3/2}/\varepsilon$, which yields:

$$\nu_T = C_\mu \frac{k^2}{\varepsilon} \tag{1.128}$$

C_μ is the Prandtl-Kolmogorov constant.

The constants of the k–ε model have been determined by comparison with simple situations. The constants C_μ and $C_{1\varepsilon}$ are obtained from data for turbulent flow near a rigid wall. The free decay of grid turbulence helps us to find a value for $C_{2\varepsilon}$ on the basis of well-documented experimental data. The constants σ_k and σ_ε have been "optimised", based on the performance of the model in both the test cases just mentioned. Obtaining $C_{3\varepsilon}$ is elaborated in references [49] and [97]. We retain that $C_{3\varepsilon}$ is equal to 1 for a stable situation, i.e. if $\mathbb{G}$ is negative, and equal to 0 for unstable stratifications. σ_k is sometimes called "Prandtl's turbulent number for the kinetic energy of turbulence". All these constants are summarised in the table 1.4.

Remarks:

- It is the equilibrium hypothesis between the creation of turbulence and dissipation which leads to the mixing length model. Actually, by assuming $\mathbb{P} = \varepsilon$, and as $\nu_T \approx k^2/\varepsilon$, we immediately find $\nu_T \approx k^2/\mathbb{P}$. Also $\nu_T \approx \sqrt{k} L_t$, where L_t is the characteristic size of vortices. We get:

  $$\nu_T^2 \approx \frac{k^2}{2 S_{ij} S_{ij}}$$

  and $\nu_T^4 \approx k^2 L_t^4$. By dividing the second equation by the first one, member by member, we finally get:

  $$\nu_T \approx L_t^2 \sqrt{2 S_{ij} S_{ij}}$$

- In internal flow modelling, the term $-2/3\, k\, \delta_{ij}$ is often omitted because it plays the same role as pressure. If we want to have the true pressure, it is not possible to integrate this term which should thus appear in numerical resolution.

Boundary conditions of the k–ε model

Rigid boundaries:

to define rigid boundary conditions, it is considered that turbulence is in local equilibrium at the wall, such that the production of turbulence (due to shear at the wall and at the bed) is equal

Table 1.4: constants of the k–ε model

κ	C_μ	Pr_T	$C_{1\varepsilon}$	$C_{2\varepsilon}$	$C_{3\varepsilon}$	σ_k	σ_ε
0.41	0.09	1.0	1.44	1.92	0 if $\mathbb{G} > 0$ and 1 if $\mathbb{G} \leq 0$	1.0	1.3

to dissipation and the velocity profile is locally logarithmic. This idea was exploited by Mellor and Yamada to establish a zero-equation turbulence model (see [54]). With the equilibrium hypothesis, at a distance δ from the wall, the k–ε model equations are:

$$k = \frac{u_*^2}{\sqrt{C_\mu}} \tag{1.129}$$

$$\varepsilon = \frac{u_*^3}{\kappa\delta} \tag{1.130}$$

The distance δ is chosen equal to 1/10th of the local mesh size, measured along the normal direction to the wall. In the presence of turbulence, the boundary condition for velocity is now written as:

$$\nu_T \frac{\partial u}{\partial n} = -u_*^2 \tag{1.131}$$

ν being replaced by ν_T.

Liquid boundaries:

without additional information on the value of k and ε at a liquid boundary, a "weak" condition is applied, i.e. $\nu_T(\partial k/\partial n) = 0$ and $\nu_T(\partial \varepsilon/\partial n) = 0$. Such a condition can, however, give rise to numerical problems at the entrance of a domain.

Free-surface:

The treatment of boundary conditions on k and ε at the free-surface is tricky since it is not clear what to do in this case. In TELEMAC-3D, the following is done:

- the production term is considered quadratic at the free-surface:

$$\mathbb{P} = \nu_T S^2 \tag{1.132}$$

- ν_T is calculated through the formula (1.128) but with a specific value of k. The later is calculated based on the definition of the friction velocity at the free-surface (obtained from the equation (1.22)):

$$u_*|_{z=\eta} = \sqrt{a_{wind}\rho_{air}/\rho}\; u_{wind} \tag{1.133}$$

and considering that k is defined by $k = u_*^2/\sqrt{C_\mu}$. The value of k at the free-surface is thus taken as:

$$k|_{z=\eta} = \left(\frac{a_{wind}\rho_{air}u_{wind}^2/\rho}{\sqrt{C_\mu}} \right)^2 \tag{1.134}$$

Remark:

In the absence of wind the boundary condition on k and ε at the free-surface is a zero-gradient. However, the presence of the free-surface should reduce the length scale of the large eddies [40]. A possibility to take this effect into account is to use the boundary condition on ε proposed in [15]:

$$\varepsilon_\eta = \frac{k_\eta^{3/2}}{\alpha h} \tag{1.135}$$

where $\alpha \approx 0.18$ is an empirical constant, k_η and ε_η are the values of k and ε at the free-surface and h is the water depth.

1.6.4 Other models

Though sufficiently difficult to implement numerically, the k–ε model has obvious limitations, particularly for eddies behind obstacles and in the presence of a high rate of deformation (tensor S_{ij}) near a stagnation point, for example. The Reynolds Stress Model (RSM) is a higher order model. The six Reynolds stress equations are then directly solved, and the ε equation, together with the equations of scalar turbulent flux of temperature and salinity. Only this type of model is capable of really taking into account turbulence anisotropy by dropping the Boussinesq model. However, the triple correlations of the equations are difficult to model and the boundary conditions difficult to justify. This model has not been implemented in TELEMAC-3D yet.

1.6.5 Turbulent stress on the walls

The presence of walls in turbulent flows makes the latter anisotropic and increases the production of turbulence due to shearing effects. Modelling near-wall turbulence is crucial in order to correctly reproduce the flows, since the no-slip condition leads to large values of the velocity gradient at the walls, which generates turbulence. Let us denote by y the distance to a wall and y^+ the dimensionless distance to a wall, defined as:

$$y^+ = \frac{y u_*}{\nu} \tag{1.136}$$

with u_* a friction velocity, defined by:

$$-\rho u_*^2 \frac{\boldsymbol{u}_\tau}{u_\tau} = \boldsymbol{\tau} = \mu \left.\frac{\partial \boldsymbol{u}_\tau}{\partial y}\right|_{y=0} \tag{1.137}$$

where $\boldsymbol{u}_\tau$ is the tangential velocity to the wall, u_τ its norm and $\boldsymbol{\tau}$ is the shear-stress, with units kg m^{-1}s^{-2}. The observation of the turbulent flow between two horizontal parallel plane walls (this configuration is called the plane Poiseuille channel) led to a sub-division of the near-wall region into three areas [97]:

- the viscous sub-layer: $0 < y^+ < 8$
- the buffer layer: $8 < y^+ < 30$
- the inertial sub-layer: $30 < y^+ < 0.2e^+$

where e^+ is the dimensionless half-height of the channel, defined by $e^+ = eu_*/\nu$ with e the half-height. The turbulence is negligible in the viscous sub-layer while the viscous effects are

small in the inertial sub-layer. In the latter, the velocity profile distribution along the normal to the wall follows a logarithmic law, so that this zone is also called the logarithmic zone. If the characteristic roughness size of the wall is greater than the thickness of the viscous sub-layer, this viscous sub-layer cannot develop and the flow is known to be hydraulically rough. If there is a viscous sub-layer the flow is known to be hydraulically smooth. In the inertial layer, the velocity profile takes the following form:

For hydraulically smooth flow:

$$\frac{u_\tau}{u_*} = \frac{1}{\kappa}\ln\left(y^+\right) + 5.2 \tag{1.138}$$

For hydraulically smooth flow the linear and logarithmic laws are unified by the Reichard law:

$$\frac{u_\tau}{u_*} = \frac{1}{\kappa}\ln(1+\kappa y^+) + 7.8(1-e^{-\frac{y^+}{11}}) - \frac{y^+}{11}e^{-0.33y^+} \tag{1.139}$$

For hydraulically rough flow:

$$\frac{u_\tau}{u_*} = \frac{1}{\kappa}\ln\left(\frac{33y}{k_s}\right) = \frac{1}{\kappa}\ln\left(\frac{y}{k_s}\right) + 8.5 \tag{1.140}$$

where k_s is the roughness size. An equivalent way of writing the velocity profile for the hydraulically rough flow is:

$$\frac{u_\tau}{u_*} = \frac{1}{\kappa}\ln\left(\frac{y}{y_o}\right) \tag{1.141}$$

where $y_o \approx k_s/33$. The reference [97] contains theoretical justifications of these formulae.

Directly simulating near-wall turbulence requires very fine meshes and modified turbulence models (so-called low-Reynolds turbulence models). Then, the computational points closest to the wall must be located in the viscous sub-layer. This is computationally expensive, especially for flows with high-Reynolds numbers. This led to the development of wall functions, based on semi-empirical formulae, which are used to reproduce near-wall effects with coarser discretisations. This corresponds to high-Reynolds-number turbulence models and requires the computational points closest to the wall to be located in the inertial layer. In Eulerian models, this can be done by designing the mesh so that the first calculation point is in the logarithmic zone. Another possibility is to solve the discretised equations on a 'classical' mesh, where the nodes located on the wall are treated in the same way as if they were shifted in the normal direction so as to be in the logarithmic zone, as for instance in [47]. This is done in TELEMAC-3D: the velocity at the wall is taken in the boundary layer at a distance y. y is arbitrarily chosen or made to depend on the local mesh size (*e.g.* the edge point is at a distance from the real wall equal to 1/10th of the size of the mesh). Then, it is possible to specify the laws to be adopted in numerical simulations to compute u_* at the walls.

Hydraulically smooth flow

By knowing u_τ and y, the Reichard law (Equation 1.139) gives u_* from the formula:

$$\frac{u_*}{u_\tau} = \frac{1}{\frac{1}{\kappa}\ln(1+\kappa y^+) + 7.8(1-e^{-\frac{y^+}{11}}) - \frac{y^+}{11}e^{-0.33y^+}} \tag{1.142}$$

This law is implicit as y^+ depends on u_*, and u_* is found by successive iterations (10 iterations are done). We then write:

$$\nu\frac{\partial u_\tau}{\partial n} = -{u_*}^2 \tag{1.143}$$

Hydraulically rough flow

Dimensional analysis shows that $\boldsymbol{\tau}$ has the form:

$$\boldsymbol{\tau} = -\frac{1}{2}\rho C_f \boldsymbol{u}_\tau \boldsymbol{u}_\tau \tag{1.144}$$

where C_f is a dimensionless friction coefficient. This general formula for shear forces supposes that $\boldsymbol{u}_\tau$ is sufficiently far from the wall. It can also be written as:

$$\mu \frac{\partial \boldsymbol{u}_\tau}{\partial \boldsymbol{n}} = -\frac{1}{2}\rho C_f \boldsymbol{u}_\tau \boldsymbol{u}_\tau \tag{1.145}$$

The specification of the stress $\mu \partial \boldsymbol{u}_\tau / \partial \boldsymbol{n}$ will naturally appear in the variational formulation of the diffusion terms in finite elements. The bed shear stress condition will thus be ensured:

- Either by a turbulence model which will provide a formula for the stress $\mu(\partial \boldsymbol{u}_\tau / \partial \boldsymbol{n}$, based on the expression of the roughness at the bed and flow in the vicinity of the bed. Most often, the turbulence model gives the shear velocity u_*, or the friction coefficient C_f.
- Or by knowledge of a friction coefficient C_f and its associated velocity $\boldsymbol{u}_\tau$, which, due to the laws of bed shear stress in two dimensions, will be the average velocity on the vertical (the friction coefficients on the bed will thus be the same in two and three dimensions). This is the case of the Chézy, Strickler and Manning formulae:

  Chézy formula: (C is the Chézy coefficient):

  $$C_f = \frac{2g}{C^2} \tag{1.146}$$

  Strickler formula: (S is the Strickler coefficient):

  $$C_f = \frac{2g}{h^{1/3} S^2} \tag{1.147}$$

  Manning formula: (m is the Manning coefficient, equal to the inverse of the Strickler coefficient):

  $$C_f = \frac{2g\, m^2}{h^{1/3}} \tag{1.148}$$

When the roughness size k_s is known, the following laws can also be used:

Nikuradse formula: This law stems from a depth-averaged value of the logarithmic turbulent velocity profile (1.140). The corresponding Chézy coefficient is obtained by the formula:

$$C = 7.83 \ln\left(12\frac{h}{k_s}\right) \tag{1.149}$$

Haaland formula:

$$C_f = \frac{1}{4\left[0.6 \ln\left(\left(\frac{6.9\nu}{4h\sqrt{u^2+v^2}}\right)^3 + \frac{k_s}{14.8h}\right)\right]^2} \tag{1.150}$$

Ramette formula: according to Ramette [73], the roughness size k_s can also be connected to the Chézy coefficient C by the expression:

$$C = \frac{26.4}{k_s^{1/6}} \left(\frac{k_s}{h} \right)^{1/24} h^{1/6} \tag{1.151}$$

This last empirical formula is not dimensionless and requires a strict respect of IS units.

Remark:

In TELEMAC-3D, there are options relying on the prescription of a friction coefficient, with association to the depth-averaged velocity, because it seemed important to be able to build a 3D model based on a 2D model without changing the calibration – the Haaland, Chézy, Strickler and Manning options. The Nikuradse option, on the other hand, makes it possible to prescribe a friction based on a logarithmic velocity profile for a given bed roughness, without any averaging of the velocity along the vertical. It thus seems more appropriate than the depth-averaged models for the simulation of non quasi-horizontal flows.

Possible improvement for active scalars modelling:

At the moment, in TELEMAC-3D there is no treatment for the temperature boundary condition when a Dirichlet condition is prescribed. However, the temperature gradients close to the walls may be large in turbulent mode, which generates turbulence as in the case of the velocity gradients. If one were to consider applications where the scalar come close to a solid boundary, it would be necessary to impose a wall function on the temperature with the $k-\varepsilon$ model. This is probably marginal but nevetheles the formulation is reported here. Considering a 1-D fully developed flow field and thermal field in a channel, the integration of the temperature equation along the normal to the wall, from the wall to the centre of the channel reads:

$$-Q_w = K_T \frac{d\overline{T}}{dy} \tag{1.152}$$

where y is the normal distance to the wall and Q_w the heat flux applied at the wall. Integrating once more yields:

$$\int_{T_w}^{\overline{T}} d\overline{T} = -Q_w \int_0^y \frac{dy}{K_T} \tag{1.153}$$

where T_w is the wall temperature. Defining the dimensionless variable $T^+ = (T_w - \overline{T})u_*/Q_w$, the equation (1.153) can be written as:

$$T^+ = \int_0^{y^+} \frac{\nu dy^+}{K_T} \tag{1.154}$$

where y^+ is defined by (1.136). The integration of this equation can be done assuming a decomposition of the near-wall region into a laminar layer where T^+ varies linearly with y^+ and a turbulent layer where it follows a logarithmic law, as in [13]. It is also possible to use a three-layers model (see [27]) through:

$$\begin{cases} T^+ = Pr\, y^+ & \text{if } y^+ < y_1^+ \\ T^+ = a_2 - \dfrac{Pr_T}{2a_1 (y^+)^2} & \text{if } y_1^+ \le y^+ < y_2^+ \\ T^+ = \dfrac{Pr_T}{\kappa} \ln y^+ + a_3 & \text{if } y^+ > y_2^+ \end{cases} \tag{1.155}$$

where the following constants were defined:

$$\begin{cases} y_1^+ = \left(\dfrac{a_4}{Pr}\right)^{1/3}, \quad y_2^+ = \sqrt{\dfrac{a_4 \kappa}{Pr_T}}, \quad a_1 = \dfrac{Pr_T}{a_4}, \\ a_2 = 15 Pr^{2/3}, \quad a_3 = 15 Pr^{2/3} - \dfrac{Pr_T}{2\kappa}\left(1 + \ln \dfrac{a_4 \kappa}{Pr_T}\right), \quad a_4 = 1000 \end{cases} \tag{1.156}$$

Recall that κ is defined in the table 1.4 and Pr_T is the turbulent Prandtl number. Finally, $Pr = \dfrac{\nu}{K}$ is the molecular Prandtl number.

1.7 Modelling culverts in the flow

1.7.1 An example of application: a flood control area in the Scheldt estuary

Flood Control Areas (FCA) together with a Controlled Reduced Tide (CRT) system are implemented in the Scheldt estuary to reduce the risk of flooding. The former is defined by an area specifically located in the regions where the bottom elevation is lower than the mean tide elevation. This area is surrounded by an outer higher dyke and in the interface with the river, it has a lower dyke that allows the flow to overtop the structure during storm surges. The CRT is based on the construction of inlet and outlet sluices that control the flow between the river and the floodplain depending on the water levels on both sides (see the figure 1.3) [92].

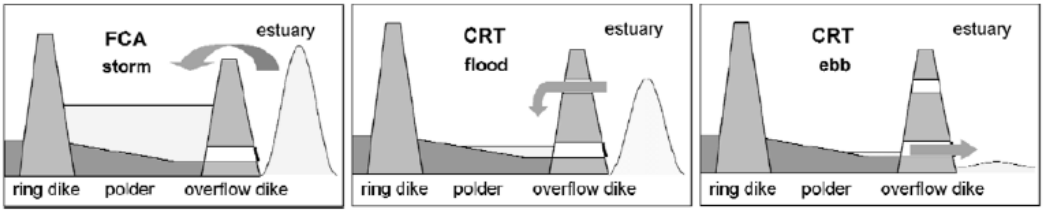

Figure 1.3: Water movements between the FCA and river without CRT (on the left) and with CRT (on the middle and right pannels). (Source: De Mulder et al. (2013)).

The incorporation of these structures in numerical models is essential to better predict and describe the flow hydrodynamics going to and coming from these areas. The inlet and outlet sluices act like a weir when they are not fully submerged and when water levels rise above the inlet ceiling pressurised flow formulae are used. The calibration of the head loss coefficients for the inlet and outlet sluices was done comparing model results with measured water levels and discharges of one specific CFA/CRT, called Lippenbroek. Later these values were validated using the measurements from the CFA/CRT Bergenmeersen. The coefficients found in this calibration/validation exercise are used for all the other inlet and outlet sluices for the other FCA, FCA/CRT areas in the 3D model.

1.7.2 Flow through a culvert: theoretical background

A number of studies regarding the description of flows through the culverts refer to Bodhaine's work [12]. Bodhaine categorized the flow through a culvert into six types, and for each type the discharge is calculated in a different way. The equations are deduced from the continuity and energy equations between the approach section (see the figure 1.4) and the exit (downstream) section of the culvert. The type of flow depends on whether the culvert flows full and whether the flow is controlled by the entrance or exit part of the culvert. The figure 1.4 shows a sketch for culvert flow definition.

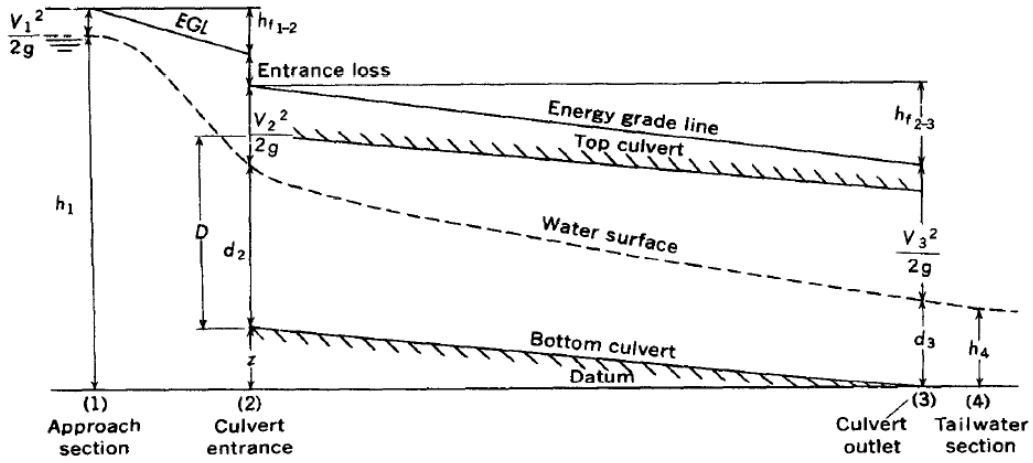

Figure 1.4: Sketch of general flow through a culvert [12].

Z gives the elevation of the culvert entrance relative to the datum through the culvert exit. The gravitational constant is given by g and h_{f12} is the head loss due to friction from the approach section to the culvert entrance; h_{f23} is the head loss due to friction inside the culvert, d_2 and d_3 are the water depths at the culvert entrance and exit, respectively; V_1, V_2 and V_3 are the velocities at the approach section, culvert entrance and culvert exit, respectively; D is the culvert height; and h_1 and h_4 are the water depths upstream and downstream of the culvert structure. The six types of flow classified by Bodhaine [12] depend on the water depths upstream and downstream of the culvert. The figure 1.5 gives a schematization of the different flow types made according to the equation for each type of flow. The different equations are presented below.

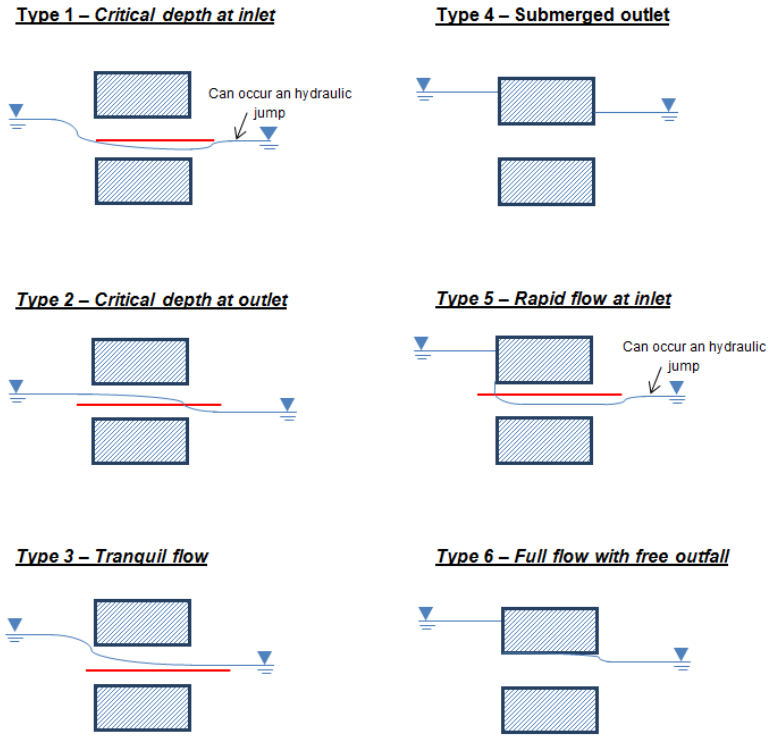

Figure 1.5: Sketch of the 6 different types of flow that occur through culverts according to Bodhaine [12]. The red line represents the critical water depth.

Type 1 – Critical depth at inlet- supercritical flow at the entrance

In flow type 1 the flow is supercritical inside the culvert and the critical depth occurs at the entrance of the culvert. The culvert slope (S_0) has to be greater than the critical slope (S_c) and the culvert flows partially full. For the Froude number $Fr = 1$ (which is the case of flow type 1), the discharge coefficient is typically $C_D = 0.95$. The discharge is then calculated according to the following formula:

$$Q = C_D A_c \sqrt{2g\left(h_1 - z - h_c - h_{f12} + \alpha \frac{\overline{V_1}^2}{2g}\right)} \tag{1.157}$$

with:
C_D the discharge coefficient;
A_c the flow area at critical water depth;
g the gravitational constant;
h_1 the upstream water depth;
z elevation of the culvert entrance;
h_c the critical water depth;
h_{f12} the head loss due to friction from the approach section to the culvert entrance;
α the kinetic energy correction coefficient for the approach section;
V_1 the average flow velocity at the approach section of the culvert.

Type 2 – Critical depth at outlet – supercritical flow at the exit

In flow type 2 the flow is tranquil inside the culvert. The critical depth is located at the culvert outlet. The culvert flows partially full. Here the culvert slope has to be smaller than the critical slope. The discharge coefficient is similar to flow type 1. The discharge is then calculated according to the following formula:

$$Q = C_D A_c \sqrt{2g\left(h_1 - h_c - h_{f12} - h_{f23} + \alpha \frac{\overline{V_1}^2}{2g}\right)} \tag{1.158}$$

with:
h_{f23} the head loss due to friction inside the culvert.

Type 3 – Tranquil flow – subcritical flow

In flow type 3 the flow is subcritical inside the culvert. There is no critical depth. The culvert flows partially full. Like flow types 1 and 2, the discharge coefficient varies with respect to the Froude number, being typically between $C_D = 0.82 - -0.95$. The discharge is calculated according to the following formula:

$$Q = C_D A_3 \sqrt{2g\left(h_1 - d_3 - h_{f12} - h_{f23} + \alpha \frac{\overline{V_1}^2}{2g}\right)} \tag{1.159}$$

with:
A_3 the flow area at the culvert outlet;
d_3 the water depth at the culvert outlet.

Type 4 – Submerged inlet and outlet

In flow type 4 the culvert inlet and outlet are submerged. The culvert flows full. The discharge coefficient varies with respect to the culvert geometry, ranging typically between $C_D = 0.75$ and $C_D = 0.95$. The discharge is calculated according to the following formula:

$$Q = C_D A_0 \sqrt{2g \frac{h_1 - h_4}{1 + 29 C_D^2 n^2 L / R^{4/3}}} \tag{1.160}$$

with:
A_0 the flow area at the culvert entrance;
h_4 the downstream water depth;
n the Manning coefficient;
L the length of the culvert;
R the hydraulic radius.

Type 5 – Rapid flow at inlet

In flow type 5, the flow is supercritical at the inlet to the culvert. The culvert flows partially full. Type 5 flow does not usually occur. When it does, the discharge coefficient is in general lower than the other types.

$$Q = C_D A_0 \sqrt{2g(h_1 - z)} \tag{1.161}$$

Type 6 – Full flow with free outfall

In flow type 6 the culvert flows full. The discharge coefficient is similar to the one obtained for the flow type 4. The discharge is calculated according to the following formula:

$$Q = C_D A_0 \sqrt{2g(h_1 - d_3 - h_{f23})} \tag{1.162}$$

The indices of the different variables might seem a bit confusing, but it was chosen to take the formulae from Bodhaine as they were and not to make any changes to them. Bodhaine differentiated between these six flow types based on conditions given in the table 1.5.

Table 1.5: Conditions for each type of flow defined by Bodhaine [12].

	$\frac{\mathbf{h_1 - z}}{\mathbf{D}}$		
Type 1	< 1.5	$\frac{h_4}{h_c} < 1.0$	$S_0 > S_c$
Type 2	< 1.5	$\frac{h_4}{h_c} > 1.0$	$S_0 < S_c$
Type 3	< 1.5	$\frac{h_4}{D} \leq 1.0$	
Type 4	> 1.0	$\frac{h_4}{D} > 1.0$	
Type 5	≥ 1.5	$\frac{h_4}{D} \leq 1.0$	
Type 6	≥ 1.5	$\frac{h_4}{D} > 1.0$	

All the different culvert geometric features will affect the presence of culvert flow type 5 or 6. To differentiate the two types, Bodhaine [12] suggests to use the relations given in the figure 1.6. r is the radius of entering rounded and w is the measure of the length of a wingwall or chamfer. First a curve corresponding to r/D, w/D is chosen. Then a point is set using the value for the culvert slope and for the ratio between the culvert length and height. If the point lies to the right of the chosen curve, the flow is type 6, if it lies to the left of the curve, the flow is type 5.

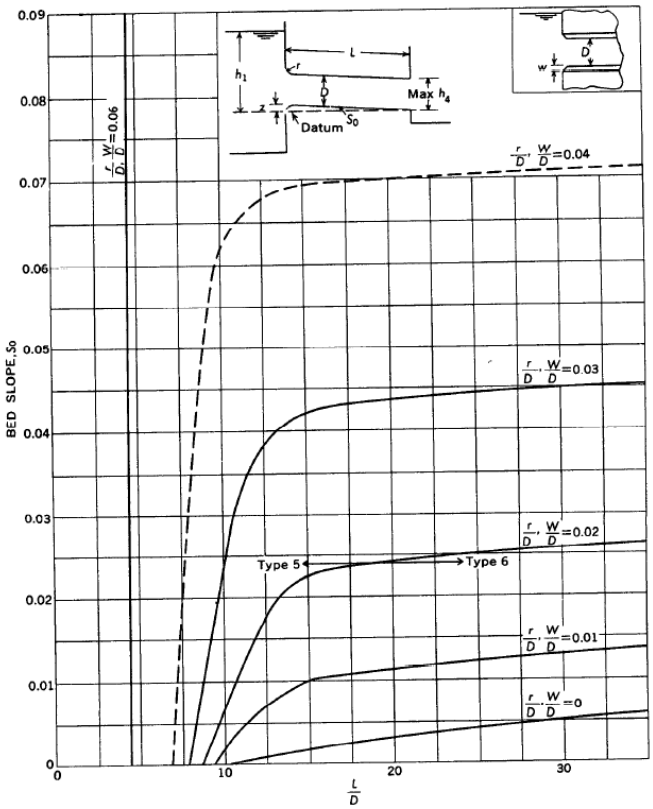

Figure 1.6: Criterion for classifying flow types 5 and 6 in concrete box or pipe culverts with square, rounded, or bevelled entrances, either with or without wingwalls [12].

The head loss coefficients are subject of different studies made in laboratory experiments. A number of authors have arrived to different values or empirical relationships for the head loss coefficients. For instance, Bodhaine [12] suggests different values for the discharge coefficient (C_D) for each type of flow and depending on a number of geometric features from the culvert. The discharge coefficients can vary from 0.39 to 0.98. Another example is Carlier [14] who proposes a non-dimensional coefficient μ, that for hydraulic structures made of only one culvert can be written as follows:

$$\mu = \frac{1}{\sqrt{C_1 + C_2 + C_3}} \tag{1.163}$$

with:
C_1 the head loss coefficient at the entrance of the hydraulic structure;
C_2 the head loss coefficient in the hydraulic structure;
C_3 the head loss coefficient at the exit of the hydraulic structure.
If the general expression for the discharge $Q = \mu A\sqrt{2g\Delta H}$ proposed by Carlier [14] is compared with the formulae given by Bodhaine [12], it can be seen that the non-dimensional head loss coefficient (μ), incorporates both the effect of the discharge coefficient (C_D) and the continuous and local head losses. ΔH is the head loss for each type of flow.

1.7.3 Formulations for culvert simulation in TELEMAC

TELEMAC-2D and 3D give the possibility of modelling hydraulic structures, such as bridges, discharges under a dike or tubes in which free-surface or pressurized flows may occur during the total simulation time. This is done by a couple of points between which flow may occur with

respect to the water levels in the river and in the floodplain. The subroutine BUSE is called to model this kind of structures. Each kind of flow has its own type of discharge calculation. The kind of formulation used to compute the flow rates can be chosen with the keyword OPTBUSE.

Option 1 – following Carlier's formulation

With this first option, the different equations implemented to calculate the discharges are dependent on the flow regime and follow Carlier [14]. Then the velocities are deduced from the discharges and are taken into account as source terms both in the continuity and momentum equations. The critical water depth (h_c) is approximated to $h_c \approx 2/3h_1$ [14]. The figure 1.7 presents the different variables used to calculate the discharges. S_1 and S_2 are the upstream and downstream water elevations, respectively, z_1 and z_2 the upstream and downstream culvert bottom elevations, D the culvert height and $h_1 = S_1 - z_1$ and $h_2 = S_2 - z_2$ the upstream and downstream water depths, respectively.

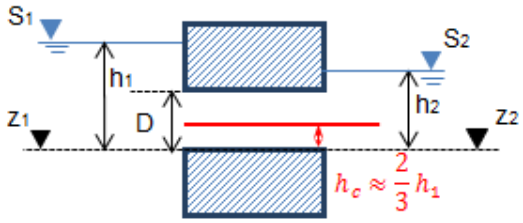

Figure 1.7: Representation of the different variables used to calculate the discharges for each type of flow.

The equations coded with this first option are described below. Between brackets the corresponding flow type according to Bodhaine [12] is given.

- Free surface flow equations:
  - Submerged weir (Bodhaine type 3)

$$Q = \mu(S_2 - z_2)W\sqrt{2g(S_1 - S_2)} = (S_2 - z_2)W\sqrt{\frac{2g(S_1 - S_2)}{C_1 + C_2 + C_3}} \tag{1.164}$$

  - Unsubmerged weir(Bodhaine type 2)

$$Q = 0.385W\sqrt{2g}(S_1 - z_1)^{3/2} \tag{1.165}$$

- Pressurised flow equations:
  - Submerged orifice law (Bodhaine type 4)

$$Q = \mu DW\sqrt{2g(S_1 - S_2)} = DW\sqrt{\frac{2g(S_1 - S_2)}{C_1 + C_2 + C_3}} \tag{1.166}$$

  - Unsubmerged orifice law

$$Q = \mu DW\sqrt{2g(S_1 - S_2)} = DW\sqrt{\frac{2g(S_1 - S_2)}{C_1 + C_2}} \tag{1.167}$$

The user has the possibility of assigning different values for C_1, C_2 and C_3. The flow direction is imposed, *e.g.*, the user can specify if the flow is going in only one direction or in both directions and in which direction. The keyword CLP specifies this behaviour.
CLP=0, flow is allowed in both directions;
CLP=1, flow is only allowed from section 1 to section 2
CLP=2, flow is only allowed from section 2 to section 1
CLP=3, no flow allowed.
A relaxation parameter (θ) is introduced so that the discharge is calculated in an explicit, implicit, or semi-implicit way. If $\theta = 1$ the calculation of the discharge is explicit while if $\theta = 0$ the discharge calculation is implicit:

$$Q^n = \theta Q^n + (1-\theta)Q^{n-1} \tag{1.168}$$

Relaxation gives slower convergence speed to get the final solution but smoothes some instabilities. If the solution does not converge because of instabilities, the coefficient can be lowered.

Option 2 – following Bodhaine's formulation

With this option the discharges are calculated based on the equations proposed in [12] and similar to the ones incorporated in DELFT 3D model. The flow type 1 conditions were not incorporated since they only occur when the culvert slope is larger than the critical flow slope. This only happens in very rare occasions if the culvert slope is very steep. The equations used to compute the discharges are given below.

Type 2 – Critical depth at outlet

$$Q = \mu h_c W \sqrt{2g(S_1 - (z_2 + h_c))} \tag{1.169}$$

with:

$$\mu = C_{D1} / \sqrt{1 + \left[\frac{2gLn^2}{R^{4/3}} + C_v\right] C_{D1}^2 \frac{h_c}{h_s}^2} \tag{1.170}$$

$$h_s = 0.5h_c + 0.5(S_1 - z) \tag{1.171}$$

$$R = \frac{h_s W}{2h_s + W} \tag{1.172}$$

Type 3 – Tranquil flow

$$Q = \mu (S_2 - z_2) W \sqrt{2g(S_1 - S_2)} \tag{1.173}$$

with:

$$\mu = C_{D1} / \sqrt{1 + \left[\frac{2gLn^2}{R^{4/3}} + C_v\right] C_{D1}^2 \left(\frac{S_2 - z_2}{h_s}\right)^2} \tag{1.174}$$

$$h_s = 0.5(S_1 - z) + 0.5(S_2 - z) \tag{1.175}$$

$$R = \frac{h_s W}{2h_s + W} \tag{1.176}$$

Type 4 – Submerged outlet

$$Q = \mu D W \sqrt{2g(S_1 - S_2)} \tag{1.177}$$

with:

$$\mu = C_{D2} / \sqrt{1 + \left[\frac{2gLn^2}{R^{4/3}} + C_v\right] C_{D2}^2} \tag{1.178}$$

$$h_s = D \tag{1.179}$$

$$R = \frac{h_s W}{2h_s + 2W} \tag{1.180}$$

Type 5 – Rapid flow at inlet

$$Q = \mu DW\sqrt{2gh_1} \tag{1.181}$$

with:

$$\mu = C_{D3} \tag{1.182}$$

$$h_s = D \tag{1.183}$$

$$R = \frac{h_s W}{2h_s + 2W} \tag{1.184}$$

Type 6 – Full flow with free outfall

$$Q = \mu DW\sqrt{2g(S_1 - (z_2 + D))} \tag{1.185}$$

with:

$$\mu = C_{D2}/\sqrt{1 + \left[\frac{2gLn^2}{R^{4/3}} + C_v\right] C_{D2}^2} \tag{1.186}$$

$$h_s = D \tag{1.187}$$

$$R = \frac{h_s W}{2h_s + 2W} \tag{1.188}$$

The head loss coefficient expressions were obtained from experimental studies made at Flanders Hydraulic Research. The discharge coefficient, C_D, is dependent of each type of flow, being the same for types 1, 2 and 3 (C_{D1}), then for types 4 and 6 (C_{D2}) and finally for type 5 (C_{D3}). The conditions at which a certain type of flow occurs, are presented in the table 1.6.

Table 1.6: Conditions for each type of flow used in TELEMAC with OPTBUSE = 2.

	$\frac{\mathbf{S_1 - z_1}}{\mathbf{D}}$	$\frac{\mathbf{S_2 - z_2}}{\mathbf{D}}$	$\mathbf{S_2 - z_2}$	L/D
Type 2	< 1.5		$< h_c$	
Type 3	< 1.5	≤ 1.0	$> h_c$	
Type 4	> 1.0	> 1.0		
Type 5	≥ 1.5	≤ 1.0		$< h_c$
Type 6	≥ 1.5	≤ 1.0		$\geq h_c$

The equations presented above are written to describe flow conditions through a culvert system with a single pipe. Nevertheless, additional features are sometimes incorporated in the hydraulic structures, such as weirs in the vicinity of the culvert entrance or exit. Such combined structures have to be taken into account. Then the geometric features of the culvert presented in the figure 1.7 are modified (see the figure 1.8). It was decided that an equivalent culvert bottom elevation should be used, which replaces both the bottom elevations z_1 and z_2 in the formulae decribed above. The equivalent bottom culvert elevation is then equal to the mean between z_1 and z_2. The diameter used in the equations will be the one corresponding to the entrance of the culvert, *i.e.*, regarding the figure 1.8, if the flow goes from left to the right D will be replaced by D_1 and on the opposite direction, the value D_2 will be used.

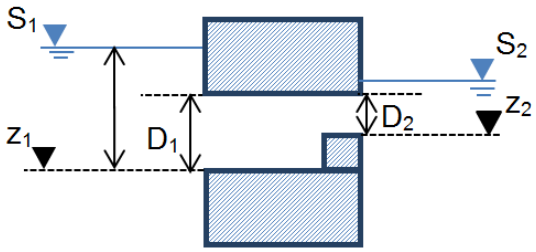

Figure 1.8: Representation of the different variables used to calculate the discharges for each type of flow.

The head loss coefficient (μ was adapted from the one calculated with the first option, based on Carlier [14] and is used as main head loss coefficient). Structures that caused additional head loss, like valves, grilles (trash screens) or pilars were added in the calculation of this main coefficient. In this way these additional features that can be present in culvert structures of different geometric configurations are taken into account and contribute to the flexibility of the implementation of many types of culvert structures. The head loss due to singularities can be obtained by the general relation [50], [14]:

$$\Delta H = C\frac{U^2}{2g} \text{ or } U = \mu\sqrt{2g\Delta H} \tag{1.189}$$

with:

$$\mu = \frac{1}{\sqrt{C}} \tag{1.190}$$

The coefficient C represents the sum of the different contributions for the head loss due to singularities:

$$C = C_1 + C_p + C_2 + C_3 + C_v + C_T \tag{1.191}$$

The different contributions to this head loss coefficient C will be discussed separately and in detail here below.

Coefficient C_1

C_1 represents the head loss due to the contraction of the flow at the entrance of the hydraulic structure. Usually it is equal to 0.5 (figure 1.9). Usually, there is an abrupt contraction that will cause a head loss due to the decelaration of the flow immediately after the culvert entrance.

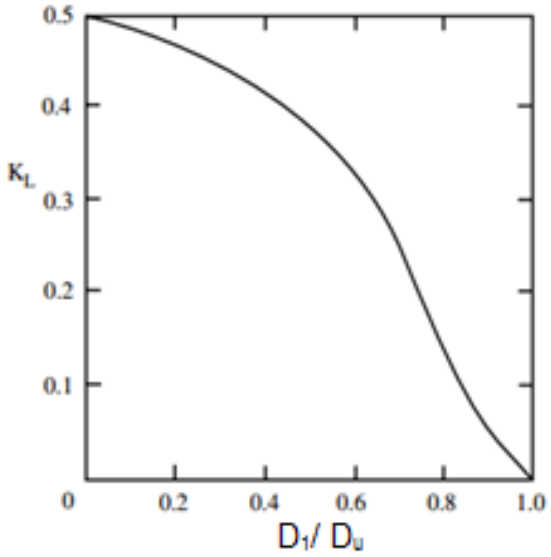

Figure 1.9: Local loss coefficient for a sudden contraction as a function of diameter ratio between the diameter after the contraction (D_1) and before the contraction D_u [48].

Already in the past, Bodhaine [12] noticed that the discharge coefficient (C_D) for type 5 flow had to be lowered comparetively with the other flow types. It seems that the calculated discharge tends to be overestimated when the default equation is applied. In order to take into account that effect, a correction coefficient (α_1^5) is applied to C_1 when type 5 flow occurs, such that:

$$\Delta H_{1.5} = \alpha_1^5 C_1 \frac{U^2}{2g} \tag{1.192}$$

Comparing with the values proposed in [12], $4 \leq \alpha_1^5 \leq 10$.

Coefficient C_p

Sometimes at the entrance of culverts the flow is divided into two sections caused by two entrance boxes instead of one but than the flow converges into a single culvert pipe. In other words a kind of pilar is dividing the flow at the entrance. This causes additional head loss and is taken into account. Following Carlier [14] the head loss through parallel pillars is given by:

$$\Delta H_p = C_p \frac{U^2}{2g} \tag{1.193}$$

where $C_p = \beta \left(Lp/b\right)^{4/3} \sin\theta$ represents the head loss coefficient due to the presence of pillars. Lp is the thickness of the pillars, b the free thickness between two consecutive pillars and β a coefficient dependent on cross-shore section of the pillar.

Coefficient C_2

C_2 represents the head loss coefficient due to the friction in the structure and is expressed by (Lencastre, 1972):

$$\Delta H_2 = C_2 \frac{U^2}{2g} = \frac{2gLn^2}{R^{4/3}} \frac{U^2}{2g} \tag{1.194}$$

where L is the length of the structure, n the Manning Strickler coefficient of the structure and R the wet cross-shore section in the structure calculated in the code for each type of flow. The table

1.8 presents the expressions for the calculation of R, following what was done in the DELFT 3D model. Here an assumption is made when calculating the hydraulic radius since the code does not make any kind of backwater analysis to get the precise water depths that occur in the culvert (like Mike 11 does).

Table 1.7: Different parameters for each type of flow to calculate the hydraulic radius in TELEMAC-3D.

Type of flow	$\mathbf{h_s}$	**R**
Type 2	$0.5h_c + 0.5(S_1 - z)$	$R = \frac{h_s W}{2h_s + W}$
Type 3	$0.5(S_1 - z_1) + 0.5(S_2 - z_2)$	$R = \frac{h_s W}{2h_s + W}$
Type 4	D	$R = \frac{h_s W}{2h_s + 2W}$
Type 5	D	$R = \frac{h_s W}{2h_s + 2W}$
Type 6	D	$R = \frac{h_s W}{2h_s + 2W}$

Coefficient C_3

C_3 is the head loss coefficient due to expansion of the flow exiting the culvert. It can be given by [50]:

$$\Delta H_3 = \left(1 - \frac{A_s}{A_{s2}}\right)^2 \frac{U^2}{2g} = C_3 \frac{U^2}{2g} \tag{1.195}$$

where A_s and A_{s2} are the sections in and at the downstream part of the structure. Usually is equal to the unity for a sudden enlargement.

Coefficient C_V

C_V is the head loss coefficient due to the presence of a valve. The head loss due to valves (ΔH_v) can also be estimated:

$$\Delta H_v = C_V \frac{U^2}{2g} \tag{1.196}$$

where C_V depends on the type of valve and the degree of opening. For a gate valve, some values were obtained experimentally, and they depend on the opening of the valve [48]: see the table 1.8.

Table 1.8: Values for the head loss coefficient depending on the opening of a gate valve.

	C_V
Wide open	0.2
3/4 open	1.0
1/2 open	5.6
1/4 open	17

Again, a correction coefficient (α_v^5) is applied to the head loss coefficient due to a valve in order to take into account the increase of the head loss when type 5 flow occurs. Through a number

of laboratory experiences (IMDC Report 613_9_1), it can be seen that when type 5 flow occurs, there is a greater influence of having a valve: the associated head loss coefficient is in general much higher than for the other types of flow (see the figure 1.10). Please note that the variable α in the figure 1.10 is not the same as α_v^5. The figure 1.10 is given here just to show the influence of the valve in the different types of flow.

$$\Delta H_{v5} = \alpha_v^5 C_v \frac{U^2}{2g} \tag{1.197}$$

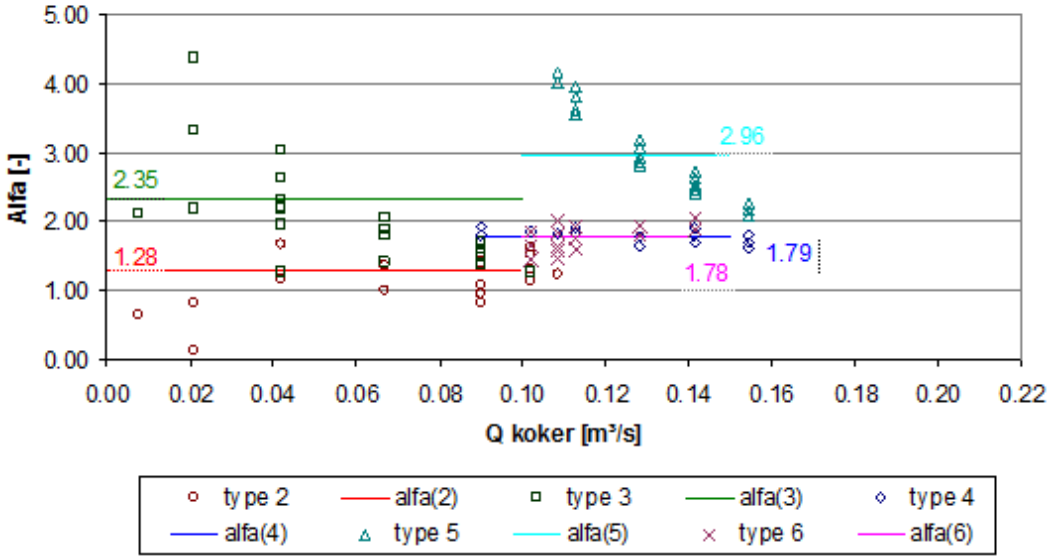

Figure 1.10: Discharge coefficient (Alfa) due to the presence of open valves for each type of flow (source: IMDC Report 613_9_1)

Coefficient C_t

Trash screens are usually present at the inlet of culverts to prevent garbage from entering or blocking the culvert. The head loss due to trash screens (ΔH_t) can be estimated by its relationship with the velocity head through the net flow area. A number of expressions were obtained in the past by several authors. We use the expression given by the Bureau of Reclamation [98]:

$$\Delta H_t = (1.45 - 0.45A_t - A_t^2)\frac{U^2}{2g} = C_t \frac{U^2}{2g} \tag{1.198}$$

where $A_t = \frac{Anet}{Agross}$ gives the ratio of net flow area to gross rack area. U is the net flow velocity. The value for C_t can vary between $C_t = 0$ equivalent to not having any trash screens to approximately $C_t = 1.4$, for which the net flow area is almost equal to the gross rack area.
Note that a very similar fomulation to this second option is included in DELFT 3D. The main difference between the second culvert functionality in TELEMAC and the one included in DELFT 3D is the way how the head loss coefficient is calculated. While in TELEMAC, the Carlier reference [14] was followed, in DELFT 3D they refer to experiments executed by Flanders Hydraulic Research.

Transport of scalars through culverts

With the implementation of the culvert functionality, some modifications had to be done in the code such that it would be possible to model the passage of the scalar through the culvert structure. Following the same idea implemented to model the flow through culverts, the concentration of the scalar in the model domain is assigned to source and sink terms for scalars. When the flow goes from the river to the floodplain, there is a source point in the floodplain with a scalar concentration equal to the one in the river. At the same time in the river there is a sink term with the same scalar concentration. The opposite happens when the flow goes from the floodplain to the river. Please note that it is the scalar concentration that is assigned to the source term and not the scalar concentration per second. In its structure, TELEMAC deals with that concentration and associates it to the discharge and volume of fluid at the source terms. In order to take into account the transport of scalars in the model the user has only to specify in the steering file the keywords relative to the scalars.

1.7.4 Input data for TELEMAC-3D

In order to take culverts into account in TELEMAC-3D the user has to define in the steering file two keywords:
NUMBER OF CULVERTS
CULVERTS DATA FILE

The number of culverts has to be assigned to the keyword NUMBER OF CULVERTS (please note that this is the number of culverts and not the number of sources/sink terms: one culvert has two sink/source terms). The keyword CULVERTS DATA FILE refers to an ASCII file where the geometric characteristics and all head loss coefficients are given to be used by the code. The text file has to follow a strict structure of the input parameters in order for the software to read the right values for the right parameter. Here below an example is given:

Relaxation, Number of culverts
0.05 2
I1 I2 CE1 CE2 CS1 CS2 LRG HAUT1 CLP LBUS Z1 Z2 CV C56 CV5 C5 CT HAUT2 FRIC LENGTH CIRC D1 D2 A1 A2 AA
data culvert 1 ...
data culvert 2 ...
...

The index number 1 refers to the river side and the index 2 refers to the floodplain side. I1 and I2 correspond to the index of the source/sink terms in the river side and in the floodplain that represent the beginning and end of the culvert. CE1, CE2 and CS1, CS2 are the head loss coefficients for the inlet and outlet sluice entrance (C_1) and exit (C_3), respectively. LRG represents the width of the culvert. HAUT1 and HAUT2 the height of the culvert at the river and floodplain side. The flow direction is also imposed through the keyword CLP and a relaxation parameter (variable RELAXB) is incorporated in the code. CLP=0, flow is allowed in both directions; CLP=1, flow is only allowed from section 1 to section 2; CLP=2, flow is only allowed from section 2 to section 1; CLP=3, no flow allowed. LBUS is the linear head loss in the culvert. Z1 and Z2 the elevations of the culvert extremities in the river side and in the floodplain. CV refers to the loss coefficient due to the presence of a valve (C_v) and C56 (c) is the constant used to differentiate flow types 5 and 6. C5 and CV5 represent correction coefficients to C1 and to CV coefficients due to the occurrence of the type 5 flow. CT is the loss coefficient due to the presence of trash screens (C_t). FRIC is the Manning Strikler coefficient. The length of the

culvert can be set (LENGTH parameter), and its shape can be specified through the parameter CIRC (equal to 1 in case of a circular section, 0 for a rectangular section). D1 and D2 are the angles that the pipe makes with respect to the bottom, in degrees. A1 and A2 are the angles with respect to the x axis. For a vertical intake, the angle with the bottom will therefore be 90°. They are used to account for the current direction at source or sink point. AA is a parameter which allows the user to choose whether A1 and A2 are automatically computed by TELEMAC or whether the data file values are used to set these angles: AA=1 – automatic angle; AA=0 – user-set angle.

1.8 Coriolis force and centrifugal force

These forces are due to the rotation of the Earth on its own axis. We establish their expressions below. Let $\boldsymbol{k}$ be the unit vector oriented along the rotational axis of the Earth, from South to North. The rotation vector of the Earth can then be written as:

$$\boldsymbol{\Omega} = \Omega \boldsymbol{k} \tag{1.199}$$

with $\Omega = 2\pi/T$ where T is equal to 86164: the duration in seconds of the sidereal day.

Let R_a be a fixed referential with its origin at the centre of the Earth and R_T a referential which turns with the Earth. In this referential, the axis Ox is directed eastwards, Oy northwards and Oz upwards. For any vector $\boldsymbol{A}$, the following relation holds:

$$\left.\frac{d\boldsymbol{A}}{dt}\right|_{R_A} = \left.\frac{d\boldsymbol{A}}{dt}\right|_{R_T} + \boldsymbol{\Omega} \wedge \boldsymbol{A} \tag{1.200}$$

The relation between the velocities expressed in the two referentials then reads:

$$\boldsymbol{u}_A = \boldsymbol{u} + \boldsymbol{\Omega} \wedge \boldsymbol{r}_A \tag{1.201}$$

which is equivalent to:

$$\left(\frac{d\boldsymbol{r}_A}{dt}\right)_{R_A} = \left(\frac{d\boldsymbol{r}}{dt}\right)_{R_T} + \boldsymbol{\Omega} \wedge \boldsymbol{r}_A \tag{1.202}$$

By deriving $\boldsymbol{u}_A$ with respect to time, we get:

$$\begin{aligned} \left(\frac{d\boldsymbol{u}_A}{dt}\right)_{R_A} &= \left(\frac{d\boldsymbol{u}_A}{dt}\right)_{R_T} + \boldsymbol{\Omega} \wedge \boldsymbol{u}_A \\ &= \left(\frac{d\boldsymbol{u}}{dt}\right)_{R_T} + \left(\frac{d\boldsymbol{\Omega}}{dt}\right)_{R_T} \wedge \boldsymbol{r} + \boldsymbol{\Omega} \wedge \left(\frac{d\boldsymbol{r}}{dt}\right)_{R_T} + \boldsymbol{\Omega} \wedge \boldsymbol{u} + \boldsymbol{\Omega} \wedge (\boldsymbol{\Omega} \wedge \boldsymbol{r}) \end{aligned} \tag{1.203}$$

The second line is obtained by replacing $\boldsymbol{u}_A$ with its expression (1.201) in the first line. The term $(d\boldsymbol{\Omega}/dt)_{R_T}$ is equal to 0 and $(d\boldsymbol{r}/dt)_{R_T}$ is equal to $\boldsymbol{u}$ by definition. We then have:

$$\left(\frac{d\boldsymbol{u}_A}{dt}\right)_{R_A} = \left(\frac{d\boldsymbol{u}}{dt}\right)_{R_T} + 2\boldsymbol{\Omega} \wedge \boldsymbol{u} + \boldsymbol{\Omega} \wedge (\boldsymbol{\Omega} \wedge \boldsymbol{r}) \tag{1.204}$$

The last term of this equation is oriented along the vertical. To write the equations of motion in the referential linked to the surface of the Earth, it is then necessary to include the terms:

$$-2\boldsymbol{\Omega} \wedge \boldsymbol{u} \tag{1.205}$$

which is called the Coriolis force, as well as:

$$-\boldsymbol{\Omega}\wedge(\boldsymbol{\Omega}\wedge\boldsymbol{r}) \tag{1.206}$$

which is called the centrifugal force.
In the referential linked to the surface of the Earth, the vector $\boldsymbol{\Omega}$ reads:

$$\boldsymbol{\Omega}=\begin{pmatrix}\Omega_x\\ \Omega_y\\ \Omega_z\end{pmatrix}=\begin{pmatrix}0\\ \Omega\cos(\lambda)\\ \Omega\sin(\lambda)\end{pmatrix} \tag{1.207}$$

where λ is the latitude at the considered point, counted positively from the equator to the North pole and negatively from the equator to the South pole. The Coriolis force then reads:

$$-2\boldsymbol{\Omega}\wedge\boldsymbol{u}=-2\Omega\begin{pmatrix}\cos(\lambda)w-\sin(\lambda)v\\ \sin(\lambda)u\\ \cos(\lambda)u\end{pmatrix} \tag{1.208}$$

The Coriolis force only has an effect on large scale flows, and is thus usually considered in the frame of quasi-horizontal flows. In this context, the vertical velocities are neglected. Besides, the effect of the Coriolis force along the vertical are not taken into account since they are negligible compared to the gravity term. Thus, the equation (1.209) becomes:

$$-2\boldsymbol{\Omega}\wedge\boldsymbol{u}=-2\Omega\begin{pmatrix}-\sin(\lambda)v\\ \sin(\lambda)u\\ 0\end{pmatrix} \tag{1.209}$$

By definition, the Coriolis coefficient is defined by:

$$\gamma=2\Omega\sin(\lambda) \tag{1.210}$$

It is about equal to $10^{-4}s^{-1}$ at a latitude of 45°N. The Coriolis forces have a significant impact on the fluid motion when their magnitude is of the same order as the inertia forces. This is measured by the Rossby number, that corresponds to the ratio between inertia and Coriolis forces:

$$R_0=\frac{U}{\gamma L} \tag{1.211}$$

where U and L are characteristic horizontal velocity and length of flow. The Coriolis force has a significant impact when $R_0\geq 1$; that is, for large oceanic or atmospheric areas: for example, considering an oceanic flow with characteristic horizontal velocity 1 m s^{-1} and γ equal to 10^{-4}, the Coriolis force becomes significant for horizontal length scales larger than 10 km. It is however negligible in rivers and lakes (and even more so in kitchen sinks...).

Remark:
The term due to the vertical velocity is ignored in TELEMAC-3D; however, it may have some importance in oceanic flows on the mesoscale, as pointed out by Mahadevan *et al.* [53].

For a geophysical flow, the centrifugal force is not added to the Navier–Stokes equations. In fact, one considers that the free surface of a fluid at rest coincides with an equipotential surface of apparent gravity, which already integrates the effect of centrifugal force.

1.9 Arbitrary Lagrangian-Eulerian formulation

1.9.1 Principle

The Arbitrary Lagrangian-Eulerian (ALE) formulation was developed in order to simplify the treatment of problems with moving boundaries, in particular for fluid-structure interaction problems and for flows with moving interfaces (see [16, 26, 39], for example). It is reviewed and interpreted in a Finite Elements context, with a view of using it in TELEMAC-3D, in [23]. In [22], Decoene *et al.* demonstrated the equivalence between the ALE formulation and the classical sigma-transform with a particular type of ALE mapping. They introduced a generalised form of the sigma transformation, which can now be used in TELEMAC-3D.
The idea of the ALE formulation is to define a reference configuration – generally chosen to be fixed, and a mapping that gives the correspondence between the real domain and the reference domain. The mapping can be arbitrarily chosen, but it has to be conforming to the evolution of the domain boundaries.
The idea is then to compute the time-derivatives in the reference domain, which is simpler than in the real (moving) domain. Indeed, in the moving domain, the discretisation of a time derivative of a field $\boldsymbol{u}$ at position $\boldsymbol{x}$ cannot be written as:

$$\frac{\partial \boldsymbol{u}}{\partial t}(\boldsymbol{x},t) \approx \frac{\boldsymbol{u}(\boldsymbol{x},t^{n+1}) - \boldsymbol{u}(\boldsymbol{x},t^{n})}{\delta t} \tag{1.212}$$

since $\boldsymbol{x}$ may not be in the computational domain anymore at time t^{n+1}.

Throughout this document, the reference mesh is denoted by Ω^*, and the fields in the reference mesh are denoted with a $*$ superscript. We denote by $\mathscr{A}^*$ the mapping that associates to each point $\boldsymbol{x}^*$ in Ω^* a point $\boldsymbol{x}$ in Ω, and $\mathscr{A}$ the mapping that associates to each point $\boldsymbol{x}$ in Ω a point $\boldsymbol{x}^*$ in Ω^*. Here, we dropped the time-dependence in the notations to simplify, but keep in mind that Ω varies in time, as well as $\mathscr{A}^*$. $\boldsymbol{x}^*$ represents the coordinates in the reference domain. It is assumed in TELEMAC-3D that $\mathscr{A}^*$ is invertible with continuous inverse $\mathscr{A}$.

Warning:
Note that this assumption makes it impossible for TELEMAC-3D to treat the breaking of the free-surface based on this ALE formulation.

The time-derivative of a field f in the reference domain is denoted by $(\partial f/\partial t)_{x^*,y^*,z^*}(\boldsymbol{x},t)$. In TELEMAC-3D the mesh motion is only allowed along the vertical and in the reference mesh, only the z coordinate is modified. The other coordinates and time remain unchanged. In other words, $t^* = t$, $x^* = x$ and $y^* = y$. The instantaneous domain velocity is defined by:

$$\boldsymbol{c}(x,y,z^*,t) = \frac{\partial \mathscr{A}^*}{\partial t} \tag{1.213}$$

Considering that, by definition of the mapping, $\boldsymbol{x} = \mathscr{A}^*(\boldsymbol{x}^*)$, the domain velocity can also be written as follows:

$$\boldsymbol{c}(x,y,z^*,t) = \left(\frac{\partial \boldsymbol{x}}{\partial t}\right)_{x,y,z^*} = \left(\frac{\partial z}{\partial t}\right)_{x,y,z^*} \boldsymbol{e}_z = (0,0,c) \tag{1.214}$$

The Jacobian matrix of $\mathscr{A}^*$ is denoted by $\boldsymbol{J}^*$ and defined by:

$$J^*_{ij}(x,y,z^*,t) = \frac{\partial \mathscr{A}^*(x^*_i)}{\partial x^*_j} \tag{1.215}$$

The continuous expression of the Jacobian determinant is then:

$$J^*(x,y,z^*,t) = \left(\frac{\partial z}{\partial z^*}\right)_{x,y,t} \tag{1.216}$$

In TELEMAC-3D, the domain motion is managed through the sigma transformation, either in its classical or its generalised form. It is necessary here to introduce one notion of the space discretisation in TELEMAC-3D, which is that the mesh is structured on the vertical. It is built as an extrusion of a 2D mesh along the vertical, then divided into layers. A transformation of the variable z is then done for each layer, so that z^* is equal to 0 at the bottom of a layer and 1 at the top. With the generalised sigma transform, the change of variable consists in defining z^* as:

$$z^* = \frac{z - z_{ip}}{z_{ip+1} - z_{ip}} = \frac{z - z_{ip}}{\Delta z} \tag{1.217}$$

where z_{ip} is the elevation of the bottom of layer p at point i, z_{ip+1} is the elevation of the top of layer p at point i and Δz is the height of the layer p, which is piecewise constant vertically.

Classical version of the sigma transformation:

The classical sigma transformation consists in changing the vertical coordinates so that the bed elevation becomes 0 and the free surface elevation becomes 1. The following change of variables is then used:

$$z^* = \frac{z-b}{\eta - b} = \frac{z-b}{h} \tag{1.218}$$

This is an affine transformation giving a transformed domain Ω^*. The elevations are shifted and multiplied by a coefficient $1/h$. For a function f, the volume integrals are thus modified in the following manner:

$$\int_\Omega f \, d\Omega = \int_{\Omega^*} \left(\frac{\partial z}{\partial z^*}\right)_{x,y,t} f^* \, d\Omega^* = \int_{\Omega^*} h f^* \, d\Omega^* \tag{1.219}$$

From now on we shall denote the Jacobian by Δz:

$$J^* = \left(\frac{\partial z}{\partial z^*}\right)_{x,y,t} = \Delta z \tag{1.220}$$

As a matter of fact, it will appear in the section 4.2, when we deal with the generalized sigma transformation on 3D meshes, that the Jacobian is the local height of prisms, the depth h being then the sum of these heights along the vertical.

1.9.2 Writing the Navier–Stokes equations in the reference domain

Space and time derivatives:

To change referentials it is necessary to express the derivatives in the reference domain as functions of the derivatives in the real domain. Be careful that a partial derivative with respect to x in the reference domain implies that one derives with constant y, z^* and t. In what follows the subscripts indicate what variables remain constant in the derivation. Then, $(\partial f/\partial x)_{y,z^*,t}$ and $(\partial f/\partial x)_{y,z,t}$ do not have the same meaning. For space derivatives, we shall use the following basic relations:

$$\left(\frac{\partial f}{\partial x}\right)_{y,z,t} = \left(\frac{\partial f}{\partial x}\right)_{y,z^*,t} + \left(\frac{\partial f}{\partial z^*}\right)_{x,y,t} \left(\frac{\partial z^*}{\partial x}\right)_{y,z,t} \tag{1.221}$$

$$\left(\frac{\partial f}{\partial y}\right)_{x,z,t} = \left(\frac{\partial f}{\partial y}\right)_{x,z^*,t} + \left(\frac{\partial f}{\partial z^*}\right)_{x,y,t}\left(\frac{\partial z^*}{\partial y}\right)_{x,z,t} \tag{1.222}$$

$$\left(\frac{\partial f}{\partial z}\right)_{x,y,t} = \left(\frac{\partial f}{\partial z^*}\right)_{x,y,t}\left(\frac{\partial z^*}{\partial z}\right)_{x,y,t} \tag{1.223}$$

Demonstration of the relation (1.221):

$$\left(\frac{\partial f}{\partial x}\right)_{y,z,t} = \left(\frac{\partial f}{\partial x^*}\right)_{y,z^*,t}\left(\frac{\partial x^*}{\partial x}\right)_{y,z,t} + \left(\frac{\partial f}{\partial y^*}\right)_{x,z^*,t}\left(\frac{\partial y^*}{\partial x}\right)_{y,z,t} + \left(\frac{\partial f}{\partial z^*}\right)_{x,y,t}\left(\frac{\partial z^*}{\partial x}\right)_{y,z,t}$$

Since $x^* = x$ and $y^* = y$, $(\partial x^*/\partial x)_{y,z,t} = 1$ and $(\partial y^*/\partial x)_{y,z,t} = 0$. Equation (1.221) is thus obtained. The equations (1.222) and (1.223) are obtained in the same way.

Regarding the time-derivatives, the following basic formula will be used:

$$\left(\frac{\partial f}{\partial t}\right)_{x,y,z} = \left(\frac{\partial f}{\partial t}\right)_{x,y,z^*} + \left(\frac{\partial f}{\partial z^*}\right)_{x,y,t}\left(\frac{\partial z^*}{\partial t}\right)_{x,y,z} \tag{1.224}$$

Remark:

The equation (1.224) is equivalent to:

$$\left(\frac{\partial f}{\partial t}\right)_{x,y,z^*} = \left(\frac{\partial f}{\partial t}\right)_{x,y,z} + \boldsymbol{c}\cdot\boldsymbol{\nabla} f \tag{1.225}$$

which is a more classical way to write this transport equation.

By taking $f = z$ in the equations (1.221) to (1.224), and using our notation for the Jacobian, this gives properties that will be used to write the Navier – Stokes equations in the reference domain:

$$\left(\frac{\partial z^*}{\partial x}\right)_{y,z,t} = -\frac{1}{\Delta z}\left(\frac{\partial z}{\partial x}\right)_{y,z^*,t} \tag{1.226}$$

$$\left(\frac{\partial z^*}{\partial y}\right)_{x,z,t} = -\frac{1}{\Delta z}\left(\frac{\partial z}{\partial y}\right)_{x,z^*,t} \tag{1.227}$$

$$\left(\frac{\partial z^*}{\partial t}\right)_{x,y,z} = -\frac{1}{\Delta z}\left(\frac{\partial z}{\partial t}\right)_{x,y,z^*} \tag{1.228}$$

Let $\boldsymbol{c}^*$ be defined by:

$$\boldsymbol{c}^* = \left(\frac{\partial z^*}{\partial t}\right)_{x,y,z} = (0,0,c^*) \tag{1.229}$$

$\boldsymbol{c}^*$ can be interpreted as the velocity of the reference domain. With the definition of the domain's velocity given by (1.213), the equation (1.228) reads:

$$\boldsymbol{c} + \Delta z\boldsymbol{c}^* = 0 \tag{1.230}$$

The following equality holds:

$$\frac{\partial}{\partial t}\left(\frac{\partial z}{\partial z^*}\right)_{x,y,t} = \frac{\partial}{\partial z^*}\left(\frac{\partial z}{\partial t}\right)_{x,y,z^*} \tag{1.231}$$

which reads, with our notations:

$$\frac{\partial \Delta z}{\partial t} - \frac{\partial c_z}{\partial z^*} = \frac{\partial \Delta z}{\partial t} - \nabla^* \cdot \boldsymbol{c} = 0 \tag{1.232}$$

Using the equation (1.230) to replace $\boldsymbol{c}$ in the previous relation, it is equivalent to:

$$\frac{\partial \Delta z}{\partial t} + \nabla^* \cdot (\Delta z \boldsymbol{c}^*) = 0 \tag{1.233}$$

In the same way, we find:

$$\frac{\partial}{\partial t}\left(\frac{1}{\Delta z}\right) + \frac{\partial}{\partial z^*}\left(\frac{\boldsymbol{c}}{\Delta z}\right) = \frac{\partial}{\partial t}\left(\frac{1}{\Delta z}\right) + \nabla \cdot \left(\frac{\boldsymbol{c}}{\Delta z}\right) = 0 \tag{1.234}$$

This can be interpreted as an evolution law for the Jacobian determinant. It ensures the conservation of the domain's volume when changing referentials and will have to be fulfilled in the discrete framework.

Advection terms:

The advection terms must be carefully examined in a moving grid because the advected functions vary not only under the effect of the flow, but also because they are defined on moving points. The derivative df/dt, written in the real domain, reads:

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} + v\frac{\partial f}{\partial y} + w\frac{\partial f}{\partial z} \tag{1.235}$$

From now on, for the sake of simplicity, we drop the subscripts when the derivatives are estimated in the real domain. Using the formula (1.221), this equation becomes:

$$\begin{aligned} \frac{df}{dt} &= \frac{\partial f}{\partial t} + u\left(\frac{\partial f}{\partial x}\right)_{y,z^*,t} + u\frac{\partial f}{\partial z^*}\frac{\partial z^*}{\partial x} + v\left(\frac{\partial f}{\partial y}\right)_{x,z^*,t} + v\frac{\partial f}{\partial z^*}\frac{\partial z^*}{\partial y} + w\frac{\partial f}{\partial z^*}\frac{\partial z^*}{\partial z} \\ &= \frac{\partial f}{\partial t} + u\left(\frac{\partial f}{\partial x}\right)_{y,z^*,t} + v\left(\frac{\partial f}{\partial y}\right)_{x,z^*,t} + \left(u\frac{\partial z^*}{\partial x} + v\frac{\partial z^*}{\partial y} + w\frac{\partial z^*}{\partial z}\right)\frac{\partial f}{\partial z^*} \end{aligned} \tag{1.236}$$

On the other hand, the velocities in the transformed mesh are defined by:

$$u_* = \frac{dx^*}{dt} = \frac{dx}{dt} = u \tag{1.237}$$

$$v^* = \frac{dy^*}{dt} = \frac{dy}{dt} = v \tag{1.238}$$

$$w^* = \frac{dz^*}{dt} \tag{1.239}$$

The vertical velocity in the transformed mesh, w^*, is also equal to:

$$w^* = \frac{\partial z^*}{\partial t} + u\frac{\partial z^*}{\partial x} + v\frac{\partial z^*}{\partial y} + w\frac{\partial z^*}{\partial z} \tag{1.240}$$

The equation (1.236) thus becomes:

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + u\left(\frac{\partial f}{\partial x}\right)_{y,z^*,t} + v\left(\frac{\partial f}{\partial y}\right)_{x,z^*,t} + (w^* - c^*)\frac{\partial f}{\partial z^*} \tag{1.241}$$

When the mesh is moving, f^{n+1} and f^n correspond to points having the same x, y and z^*, but not the same z. The partial derivatives in time in our equations are thus expressed using $(\partial f/\partial t)_{x,y,z^*}$. Using the equation (1.224), the equation (1.241) then becomes:

$$\frac{df}{dt} = \left(\frac{\partial f}{\partial t}\right)_{x,y,z^*} + u\left(\frac{\partial f}{\partial x}\right)_{y,z^*,t} + v\left(\frac{\partial f}{\partial y}\right)_{x,z^*,t} + w^*\frac{\partial f}{\partial z^*} \tag{1.242}$$

Advection equations, in the transformed mesh and taking into account the motion of the real mesh, will then have the simple form:

$$\left(\frac{\partial f}{\partial t}\right)_{x,y,z^*} + u\left(\frac{\partial f}{\partial x}\right)_{y,z^*,t} + v\left(\frac{\partial f}{\partial y}\right)_{x,z^*,t} + w^*\frac{\partial f}{\partial z^*} = 0 \tag{1.243}$$

Equivalence between the sigma transformation and the ALE formulation

Comparing the equations (1.241) and (1.235) shows that:

$$u\frac{\partial f}{\partial x} + v\frac{\partial f}{\partial y} + w\frac{\partial f}{\partial z} = u\left(\frac{\partial f}{\partial x}\right)_{y,z^*,t} + v\left(\frac{\partial f}{\partial y}\right)_{x,z^*,t} + (w^* - c^*)\frac{\partial f}{\partial z^*} \tag{1.244}$$

while the equation (1.230), combined with (1.221) and (1.240), also yields:

$$u\frac{\partial f}{\partial x} + v\frac{\partial f}{\partial y} + (w - c)\frac{\partial f}{\partial z} = u\left(\frac{\partial f}{\partial x}\right)_{y,z^*,t} + v\left(\frac{\partial f}{\partial y}\right)_{x,z^*,t} + w^*\frac{\partial f}{\partial z^*} \tag{1.245}$$

This shows that using the sigma transformation is equivalent to writing the equations in the ALE framework at the continuous level. In the section 1.10, the Navier–Stokes equations will thus be written in the ALE framework, using $\boldsymbol{u} - \boldsymbol{c}$ instead of $\boldsymbol{u}$ for the advection terms.

Divergence:

We now want to write $\nabla \cdot \boldsymbol{u}$ in the transformed mesh. Using the formula (1.221), we have:

$$\nabla \cdot \boldsymbol{u} = \left(\frac{\partial u}{\partial x}\right)_{y,z*,t} + \frac{\partial u}{\partial z^*}\frac{\partial z^*}{\partial x} + \left(\frac{\partial v}{\partial y}\right)_{x,z^*,t} + \frac{\partial v}{\partial z^*}\frac{\partial z^*}{\partial x} + \frac{\partial w}{\partial z^*}\frac{\partial z^*}{\partial z} \tag{1.246}$$

By multiplying the previous equation by $\Delta z = (\partial z/\partial z^*)_{x,y,t}$, which depends on x, y and t, but not on z or z^*, and using the fact that:

$$\Delta z\left(\frac{\partial u}{\partial x}\right)_{y,z^*,t} = \left(\frac{\partial \Delta z u}{\partial x}\right)_{y,z^*,t} - u\frac{\partial \Delta z}{\partial x} \tag{1.247}$$

we then get:

$$\begin{aligned}\Delta z \nabla \cdot \boldsymbol{u} \quad &= \left(\frac{\partial \Delta z u}{\partial x}\right)_{y,z^*,t} - u\frac{\partial \Delta z}{\partial x} + \frac{\partial \Delta z u}{\partial z^*}\frac{\partial z^*}{\partial x} + \left(\frac{\partial \Delta z v}{\partial y}\right)_{x,z^*,t} \\ &\quad - v\frac{\partial \Delta z}{\partial y} + \frac{\partial \Delta z v}{\partial z^*}\frac{\partial z^*}{\partial y} + \frac{\partial \Delta z w}{\partial z^*}\frac{\partial z^*}{\partial z}\end{aligned} \tag{1.248}$$

The term $\partial \Delta z w^* / \partial z^*$ can be calculated using the equation (1.240):

$$\begin{aligned}\frac{\partial \Delta z w^*}{\partial z^*} &= \frac{\partial}{\partial z^*}\left(\Delta z \frac{\partial z^*}{\partial t} + \Delta z u \frac{\partial z^*}{\partial x} + \Delta z v \frac{\partial z^*}{\partial y} + \Delta z w \frac{\partial z^*}{\partial z}\right) \\ &= \frac{\partial \Delta z u}{\partial z^*}\frac{\partial z^*}{\partial x} + \frac{\partial \Delta z v}{\partial z^*}\frac{\partial z^*}{\partial y} + \frac{\partial \Delta z w}{\partial z^*}\frac{\partial z^*}{\partial z} \\ &+ \Delta z \frac{\partial}{\partial z^*}\left(\frac{\partial z^*}{\partial t}\right) + \Delta z u \frac{\partial}{\partial z^*}\left(\frac{\partial z^*}{\partial x}\right) + \Delta z v \frac{\partial}{\partial z^*}\left(\frac{\partial z^*}{\partial y}\right) + \Delta z w \frac{\partial}{\partial z^*}\left(\frac{\partial z^*}{\partial z}\right)\end{aligned} \tag{1.249}$$

The equations (1.226) to (1.228) give:

$$\frac{\partial}{\partial z^*}\left(\frac{\partial z^*}{\partial t}\right)_{x,y,z} = -\frac{1}{\Delta z}\frac{\partial \Delta z}{\partial t} \tag{1.250}$$

$$\frac{\partial}{\partial z^*}\left(\frac{\partial z^*}{\partial x}\right)_{y,z,t} = -\frac{1}{\Delta z}\frac{\partial \Delta z}{\partial x} \tag{1.251}$$

$$\frac{\partial}{\partial z^*}\left(\frac{\partial z^*}{\partial y}\right)_{x,z,t} = -\frac{1}{\Delta z}\frac{\partial \Delta z}{\partial y} \tag{1.252}$$

We now assume that the Jacobian Δz is constant vertically (which will be our case on every finite element), and we thus have:

$$\frac{\partial}{\partial z^*}\left(\frac{\partial z^*}{\partial z}\right)_{x,y,t} = 0 \tag{1.253}$$

Warning:
Here the assumption of a constant Jacobian along the vertical (it will actually be constant per layer) is strong and prevents us from using quadratic elements along the vertical of the domain. To use such elements, it would be necessary to properly write the corresponding equation on $(\Delta z\, \nabla \cdot \boldsymbol{u})$ in the reference mesh.

All this leads to:

$$\begin{aligned}\frac{\partial \Delta z w^*}{\partial z^*} &= \frac{\partial \Delta z u}{\partial z^*}\frac{\partial z^*}{\partial x} + \frac{\partial \Delta z v}{\partial z^*}\frac{\partial z^*}{\partial y} + \frac{\partial \Delta z w}{\partial z^*}\frac{\partial z^*}{\partial z} \\ &- \frac{\partial \Delta z}{\partial t} - u\frac{\partial \Delta z}{\partial x} - v\frac{\partial \Delta z}{\partial y}\end{aligned} \tag{1.254}$$

We deduce from this that:

$$\begin{aligned}\frac{\partial \Delta z w}{\partial z^*}\frac{\partial z^*}{\partial z} &= \frac{\partial \Delta z w^*}{\partial z^*} + \frac{\partial \Delta z}{\partial t} + u\frac{\partial \Delta z}{\partial x} + v\frac{\partial \Delta z}{\partial y} \\ &- \frac{\partial \Delta z u}{\partial z^*}\frac{\partial z^*}{\partial x} - \frac{\partial \Delta z v}{\partial z^*}\frac{\partial z^*}{\partial y}\end{aligned} \tag{1.255}$$

By introducing this expression into the equation (1.248), we get:

$$\Delta z\, \nabla \cdot \boldsymbol{u} = \frac{\partial \Delta z}{\partial t} + \left(\frac{\partial \Delta z u}{\partial x}\right)_{y,z^*,t} + \left(\frac{\partial \Delta z v}{\partial y}\right)_{x,z^*,t} + \frac{\partial \Delta z w^*}{\partial z^*} \tag{1.256}$$

Other terms in the Navier–Stokes equations, such as diffusion terms, are modified in an even more cumbersome manner.

Remark by J.-M. Hervouet:
The numerical resolution of equations with the help of this sigma transformation is widely used but sometimes leads to awkward numerical artefacts (creeping diffusion etc.). We shall try to avoid this wherever possible, in particular during the diffusion stage.

1.10 Final set of equations

1.10.1 Navier–Stokes equations in Cartesian coordinates

The Navier-Stokes equations are now written as follows, in the ALE framework, with the decomposition of p into p_h and p_d and where ρ is constant, equal to ρ_0:

$$\begin{cases} \dfrac{\partial \boldsymbol{u}}{\partial t} + ((\boldsymbol{u}-\boldsymbol{c})\cdot\boldsymbol{\nabla})\boldsymbol{u} = -\dfrac{1}{\rho}\boldsymbol{\nabla}(p_h+p_d) + \dfrac{1}{\rho}\boldsymbol{\nabla}\cdot(\mu_E\boldsymbol{\nabla}\boldsymbol{u}) + \boldsymbol{g} + \boldsymbol{F} \\ p_h = g\rho(\eta - z) + p_{atm} + g\rho\displaystyle\int_z^\eta \frac{\Delta\rho}{\rho}\,dz \\ \dfrac{\partial \eta}{\partial t} + \nabla_{2D}\cdot\left(\displaystyle\int_b^\eta \boldsymbol{u}_{2D}\,dz\right) = F_b \\ \nabla\cdot\boldsymbol{u} = 0 \end{cases} \tag{1.257}$$

Here we wrote the system in the real domain Ω but without specifying any time-evolution of the domain yet, otherwise the notations would be much more complex. Bear in mind that in the chapter 4, we will write the space-time discretisation of this system, written in a moving domain. From now on we define:

$$\boldsymbol{F}^{visc} = \frac{1}{\rho}\boldsymbol{\nabla}\cdot(\mu_E\boldsymbol{\nabla}\boldsymbol{u}) \tag{1.258}$$

Let us expand the system of equations (1.257) in order to seperate the horizontal components and the vertical components:

$$\begin{cases} \dfrac{\partial u}{\partial t} = -((\boldsymbol{u}-\boldsymbol{c})\cdot\boldsymbol{\nabla})u + F_x^{visc} - g\dfrac{\partial \eta}{\partial x} - \dfrac{1}{\rho}\dfrac{\partial p_d}{\partial x} + \tilde{F}_x \\ \dfrac{\partial v}{\partial t} = -((\boldsymbol{u}-\boldsymbol{c})\cdot\boldsymbol{\nabla})v + F_y^{visc} - g\dfrac{\partial \eta}{\partial y} - \dfrac{1}{\rho}\dfrac{\partial p_d}{\partial y} + \tilde{F}_y \\ \dfrac{\partial w}{\partial t} = -((\boldsymbol{u}-\boldsymbol{c})\cdot\boldsymbol{\nabla})w + F_z^{visc} - \dfrac{1}{\rho}\dfrac{\partial p_d}{\partial z} + F_z \\ p_h = g\rho(\eta - z) + p_{atm} + g\rho\displaystyle\int_z^\eta \frac{\Delta\rho}{\rho}\,dz \\ \dfrac{\partial \eta}{\partial t} = -\nabla_{2D}\cdot\left(\displaystyle\int_b^\eta \boldsymbol{u}_{2D}\,dz\right) + F_b \\ \nabla\cdot\boldsymbol{u} = 0 \end{cases}$$

where the buoyancy terms are included in $\tilde{F}_x$ and $\tilde{F}_y$ (see the section 1.5.3). Besides this system of equations, we also solve a scalar equation:

$$\frac{\partial T}{\partial t} + ((\boldsymbol{u}-\boldsymbol{c})\cdot\boldsymbol{\nabla})T - \nabla\cdot(K_E\boldsymbol{\nabla}T) + Q = 0 \tag{1.259}$$

1.10.2 Navier–Stokes equations in Mercator projection

Storm surges and storm-induced currents, differences between the actual levels of the sea and currents and their values if the tides alone were considered, are due to the variations of atmospheric pressure and the effect of the wind on large stretches of the ocean. To carry out numerical simulations, vast maritime areas are discretised on the basis of maps which are Mercator projections of portions of the Earth's sphere. It is then necessary to find out how this Mercator projection modifies the Navier–Stokes equations.
The Mercator projection belongs to the family of conformal projections which ensure the local conservation of angles, but not distances. In other words, the scale of the map is not the same at every point, but at a given point the scale is the same in all directions. The domain to be represented in plane form is on the surface of a sphere, whose points are identified by their latitude λ and their longitude φ (see the figure 1.11). Henceforth, it is assumed that the latitude and longitude are angles expressed in radians, counted positively in the northern hemisphere and to the east of the Greenwich Meridian and negatively otherwise.

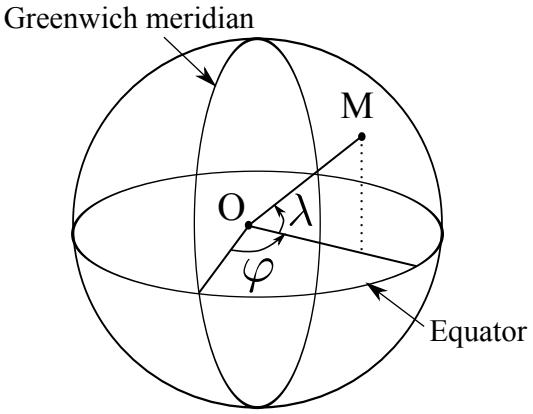

Figure 1.11: Definition of the longitude φ and latitude λ (the figure is taken from [36], figure 2.4).

On the basis of a sphere, we arrive at a plane called the Mercator plane, with the help of the projection, which to any point M of the sphere associates a point M_p of a cylinder tangent to the Earth at the equator (see the figure 1.12).
The coordinates x_p and y_p on this plane are deduced from the geographical coordinates by:

$$x_p = R\,\varphi$$

$$y_p = R\tan(\lambda)$$

However, the coordinates written in this referential do not correspond to a conformal projection because:

- for an east–west displacement, dx, on the surface of the Earth, we obtain a displacement dx_p in this referential such as:

$$dx = \cos(\lambda)\,dx_p \tag{1.260}$$

- while for a north–south displacement, dy, we obtain a displacement dy_p such as:

$$dy = \cos^2(\lambda)\,dy_p \tag{1.261}$$

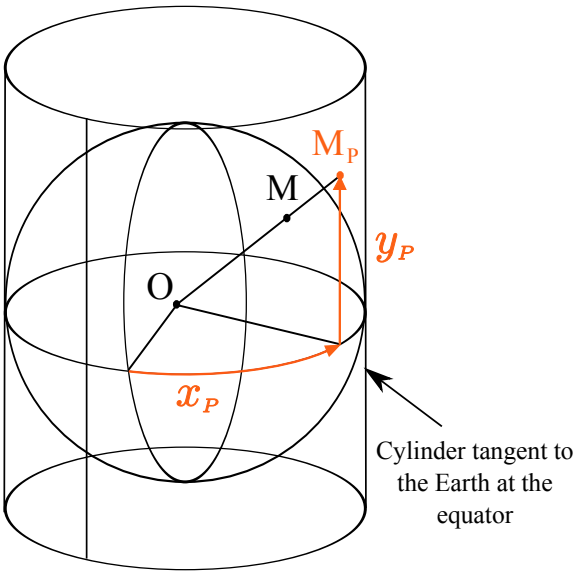

Figure 1.12: Sketch of the first phase of the Mercator projection (this figure is inspired by the figure 2.5 in [36]).

This distortion, which is greater in the north–south direction than in the east–west direction, makes the projection non-conformal. We get a conformal projection by a simple change of scale in the y direction. The new ordinate Y is then deduced from y_p, then from λ, by:

$$dY = \cos(\lambda)\, dy_p = \frac{R}{\cos(\lambda)} d\lambda \tag{1.262}$$

Moreover, it is possible to carry out a change in the origin at one point (φ_0, λ_0). Along the horizontal axis we immediately obtain:

$$X = R(\varphi - \varphi_0) \tag{1.263}$$

Along the vertical axis, we obtain by integration:

$$Y = R\left(\ln\left[\tan\left(\frac{\lambda}{2} + \frac{\pi}{4}\right)\right] - \ln\left[\tan\left(\frac{\lambda_0}{2} + \frac{\pi}{4}\right)\right]\right) \tag{1.264}$$

These coordinates X and Y are coordinates taken on maps and input to the software for numerical simulation. Hence, one should know how to express the Navier–Stokes equations in these coordinates. X and Y being functions of spherical coordinates, it is therefore practical to express the equations in these spherical coordinates at the very beginning. For this, we use the divergence and gradient operators expressed in these coordinates:

$$\nabla \cdot \boldsymbol{F} = \frac{1}{R\cos(\lambda)} \frac{\partial F_\varphi}{\partial \varphi} + \frac{1}{R\cos(\lambda)} \frac{\partial (F_\lambda \cos(\lambda))}{\partial \lambda} \tag{1.265}$$

$$\boldsymbol{grad}(f) = \frac{1}{R\cos(\lambda)} \frac{\partial f}{\partial \varphi} \boldsymbol{e}_\varphi + \frac{1}{R} \frac{\partial f}{\partial \lambda} \boldsymbol{e}_\lambda \tag{1.266}$$

Given the nature of the physical quantities and of the projection, the scalar and vectorial quantities (depth, velocity) are unchanged. We have for example $F_\varphi = F_x$ and $F_\lambda = F_y$. The derivatives

of the functions with respect to φ and λ are then replaced by the derivatives along X and Y, the Mercator coordinates:

$$\frac{\partial f}{\partial \varphi} = \frac{\partial f}{\partial X}\frac{\partial X}{\partial \varphi} + \frac{\partial f}{\partial Y}\frac{\partial Y}{\partial \varphi} = R\frac{\partial f}{\partial X} \tag{1.267}$$

$$\frac{\partial f}{\partial \lambda} = \frac{\partial f}{\partial X}\frac{\partial X}{\partial \lambda} + \frac{\partial f}{\partial Y}\frac{\partial Y}{\partial \lambda} = \frac{R}{\cos(\lambda)}\frac{\partial f}{\partial Y} \tag{1.268}$$

From this, new expressions of divergence and gradient are deduced:

$$\begin{aligned}\nabla \cdot \boldsymbol{F} &= \frac{1}{\cos(\lambda)}\frac{\partial F_x}{\partial X} + \frac{1}{\cos(\lambda)}\frac{\partial F_y}{\partial Y} - \frac{\tan(\lambda)}{R}F_y \\ &= \frac{1}{\cos^2(\lambda)}\frac{\partial}{\partial X}(F_x\cos(\lambda)) + \frac{1}{\cos^2(\lambda)}\frac{\partial}{\partial Y}(F_y\cos(\lambda))\end{aligned} \tag{1.269}$$

$$\boldsymbol{\nabla}(f) = \frac{1}{\cos(\lambda)}\begin{pmatrix} \dfrac{\partial f}{\partial X} \\ \dfrac{\partial f}{\partial Y} \end{pmatrix} \tag{1.270}$$

The equations could be solved by using all these formulae, but, in finite elements, this would induce a problem with the approximation space. As a matter of fact, if λ is a polynomial function, what becomes of $\cos(\lambda)$? If $\cos(\lambda)$ and $\sin(\lambda)$ are polynomials, what is $\tan(\lambda)$? These questions are fundamental when mass conservation has to be proved.
Now, if we take *e.g.* a conservative advection equation:

$$\frac{\partial f}{\partial t} + \nabla \cdot (f\,\boldsymbol{u}) = 0 \tag{1.271}$$

it becomes in Mercator projection:

$$\frac{\partial (f\cos^2(\lambda))}{\partial t} + \nabla \cdot (\cos(\lambda) f\,\boldsymbol{u}) = 0 \tag{1.272}$$

where $\nabla\cdot$ is now the Cartesian divergence operator. We shall see that integration into the Mercator domain of this equation will give terms such as:

$$\frac{\partial}{\partial t}\int_\Omega f\cos^2(\lambda)\,dXdY \quad \text{and} \quad \int_\Omega \nabla \cdot (f\,\boldsymbol{u}\cos(\lambda))\,dXdY$$

this last term being equal to:

$$\int_\Gamma f\,\boldsymbol{u}\,.\,\boldsymbol{n}\cos(\lambda)\,d\Gamma$$

In these terms, each length dX, dY or $d\Gamma$ is multiplied by $\cos(\lambda)$, which restores a real local scale. The volumes or flux calculated are thus real quantities. In the same way, the gradient terms would give real slopes. As in the case of all operators, the coordinates therefore appear multiplied by $\cos(\lambda)$. The idea is to carry out this operation once and for all before the start of calculations. This can be formalised by choosing a constant latitude λ per element. The advantage is that the equations to be worked out are then strictly identical to the Cartesian equations (this is actually due to the local character of the operators and is not a general property). This choice of constant latitude per element is naturally an approximation, but it enables us to retain strict finite element formalism, the approximation being somehow transferred to the coordinates. During a computation using spherical coordinates with a meshing in the Mercator projection, we shall build up a set of coordinates multiplied by $\cos(\lambda)$. If we use the coordinates modified in this manner, the Mercator projection is naturally taken into consideration

and no other modification appears again in the Navier-Stokes equations. All the mass and flow calculations are coherent and provide real values. The error in taking a constant latitude per element can be evaluated by finding the form which the Navier-Stokes equations should take with the coordinates X' and Y' such as $X' = X\cos(\lambda)$ and $Y' = Y\cos(\lambda)$. The difference at the continuum level between the equations makes the ignored terms, which are of the order of $1/R$, appear. They take into consideration the variations in latitude within the element.

Important note by J.-M. Hervouet:

The multiplication of coordinates of the points by the cosine of a constant latitude per element leads to a paradoxical situation: a point which belongs to several elements has as many sets of different coordinates which should be stored element by element. In fact this does not constitute a technical problem and does not lead to any contradiction. All calculations of vectors or matrices in finite elements start at the elementary level, where coherence is required only between the coordinates of the points of the element. It should even be possible, without any topological problem, to build a "two-dimensional" meshing of the Earth for calculating world tides, each element having coordinates corresponding to a local projection (the poles should be excluded, however, having infinite coordinates in Mercator projection).

2. Principles of the Finite Elements method

2.1 Introduction

A very simple description of the Finite Elements method is given here. It might require further development, but a list of reference works is given if more details are needed. In particular, we recommend the book by Ern & Guermond [29].
The domain of computation Ω is an open and bounded domain in $\mathbb{R}^N$, $\mathbb{R}$ being the set of real numbers. N is the space dimension, equal to 3 in TELEMAC-3D. Γ is the boundary, regular (a normal vector can be defined on it), of Ω. At the continuum level, physical quantities (pressure, temperature, components of the velocity vector, etc.) are functions of Ω in $\mathbb{R}$. Moreover, we require that these functions are square integrable and $L^2(\Omega)$ denotes the set of square integrable functions. A dot product can be defined on this set as:

$$(u,v)_0 = \int_\Omega uv \, d\Omega \tag{2.1}$$

A subset of $L^2(\Omega)$ is composed of functions whose derivatives along the N directions of space are also square integrable. This guarantees that a function gradient would also be square integrable. This subset is denoted $H^1(\Omega)$. A new dot product is defined in $H^1(\Omega)$ as:

$$(u,v)_1 = \int_\Omega uv + \overrightarrow{grad}(u) \cdot \overrightarrow{grad}(v) \, d\Omega \tag{2.2}$$

The spaces $L^2(\Omega)$ and $H^1(\Omega)$ with their dot product are called Hilbert spaces. Hilbert spaces are very similar to the Euclidean vector spaces. On both of them, and with the help of the dot product, we can define:

- a norm:
$$\|u\| = \sqrt{(u,u)} \tag{2.3}$$
- a distance:
$$d(u,v) = \|u-v\| \tag{2.4}$$
- and an angle:
$$\cos(\alpha) = \frac{(u,v)}{\|u\| \, \|v\|} \tag{2.5}$$

but their dimension is infinite. $H^1(\Omega)$ has been described with derivatives of the first order. $H^k(\Omega)$ can be described in the same way, with derivatives of the kth order, like the set of

functions whose derivatives up to kth order are all square integrable. The $H^k(\Omega)$ spaces are known as Sobolev spaces.
As for the equations being worked out, only linear equations with partial derivatives are considered (fluid mechanics equations are generally non-linear, but the resolution methods used for solving them will generally be written in a linear form). Here are some of the linear equations that we shall come across later:
Helmholtz equation:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \lambda u = 0 \tag{2.6}$$

Poisson equation:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f \tag{2.7}$$

In this chapter, we consider that our equation is in the form:

$$L(u) = f \tag{2.8}$$

Interpolation and the variational method will now be described more precisely.

2.2 Interpolation in finite elements

The general idea is to replace an unknown function u, which belongs to an infinite dimensional space, by an approximation u_h defined on a finite dimensional space N. This is an approximation and we cannot always ensure that $L(u_h) = f$. We simply try to minimize $L(u_h) - f$ and this will be the goal of the variational method.
In finite elements, a function u is represented by n real numbers which are the exact values of u at special points of the domain known as degrees of freedom. For the other points, an interpolation is sufficient. To define a unique interpolation function, e.g. a polynomial one, with exact values of u for the n degrees of freedom would be very complex, or would lead at least to a high-order function. So, finite elements give up the univocal nature of the interpolation function and choose a locally simple interpolation function (e.g. constant, linear, quadratic). A tessellation of space into segments, triangles, quadrilaterals, tetrahedra, prisms, etc., is executed. The degrees of freedom are assigned to these elements, on the vertices, at the centre of gravity, in the middle of the sides, etc., for example. Thus, each element is described both by the coordinates of its geometric nodes and by those of its interpolation nodes. This is followed by a simple definition of interpolation within each element. For example, in one dimension, taking the vertices of segments as degrees of freedom, a linear interpolation will be chosen. The function u_h approximating u can then be written as:

$$u_h = \sum_{i=1}^{n} u_i \Psi_i \tag{2.9}$$

where Ψ_i is a basis function defined differently, but in a linear manner, on each segment ending at the point i. Outside these segments, Ψ_i is zero. Figure 2.1 shows this basis function in one dimension, on a non-regular mesh. The value of Ψ_i is 1 at point i, and 0 on all the other degrees of freedom.

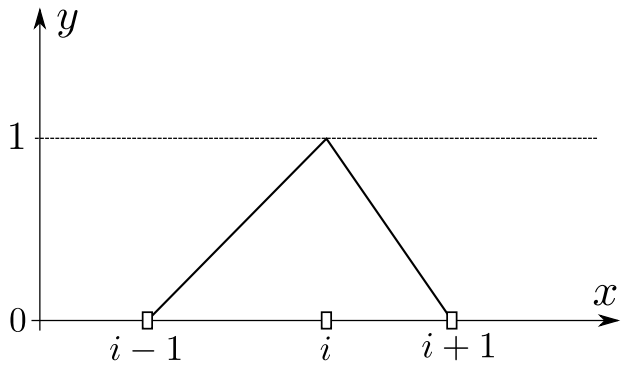

Figure 2.1: Shape of a linear basis in one dimension.

The property $\sum_{i=1}^{n} \Psi_i = 1$ is very important and will be the key to several demonstrations. In two dimensions, with triangles and a linear interpolation, each basis $\Psi_i(x,y)$ will be written as $ax+by+c$, a, b and c depending on the triangle, so as to obtain the value 1 at point i and 0 at the other ones. Figure 2.2 shows the extent and the nodal values of a linear basis on a finite element mesh.

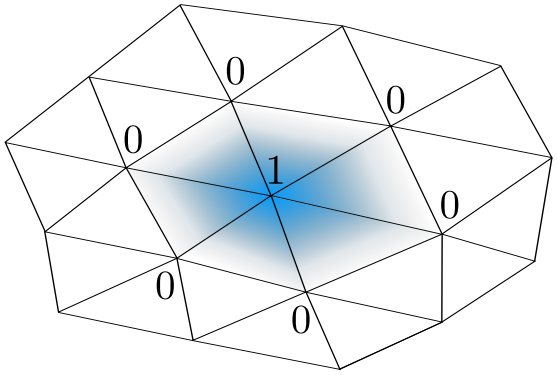

Figure 2.2: Extent of a basis function on a triangular mesh. The blue color represents higher values of the basis function.

These basis functions are sometimes called “shape functions”. With a linear interpolation, a linear function u will be represented exactly. It is said to be in the approximation space. The interpolation can also be quadratic, or of any other order which enables representation of the function on the degrees of freedom but also some of its derivatives. Sometimes, and it is so for quadrilaterals, the element is too complex to get a simple definition of the interpolation. So, we go back to a more regular reference element (a square in the case of a quadrilateral) in a “reference” space and define a geometric transformation between the two spaces. Interpolation is simple only in the reference space, except in the case of simplicial elements: segments, triangles and tetrahedra with linear interpolation. The description of different finite elements and the use of a reference space for building finite element matrices are explained in Reference [36]. A precise description of a large number of other elements can also be found in reference [20].

In TELEMAC-3D, Lagarange finite elements are used: P1 bases on prismatic 3D elements. The bases we use can be decomposed into the poduct of 2D basis with vertical bases:

$$\Psi = \Psi^H \Psi^V \tag{2.10}$$

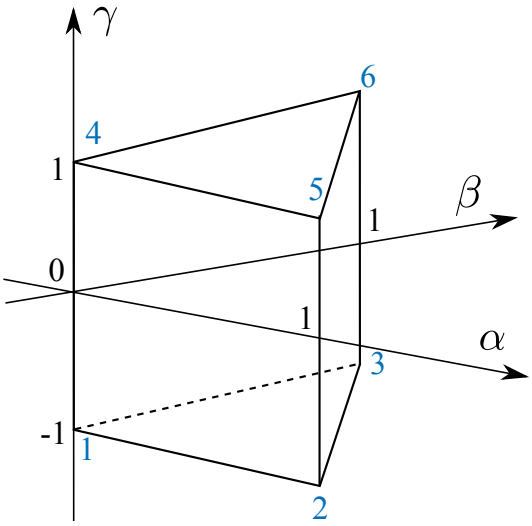

Figure 2.3: Sketch of the reference element used in TELEMAC-3D. The nodes indices are indicated in blue whereas the coordinates in the (α,β,γ) system are indicated in black.

with:

$$\sum_{i=1}^{n} \Psi_i = 1 \ , \quad \sum_{i=1}^{npoin2} \Psi_i^H = 1 \ , \quad \sum_{i=1}^{nplan} \Psi_i^V = 1 \tag{2.11}$$

where $npoin2$ is the number of points of the 2D mesh, $nplan$ is the number of points along the vertical, Ψ^H is represented in the figure 2.2 and Ψ^V is a 1-D P1 basis along z, as represented in the figure 2.1, but defined on the quadrangles composing the lateral faces of the prisms. In order to simplify the computations of the matrices stemming from the Finite Eelements formulations, they are calculated in reference elements and with reference bases. Each element of the real mesh is transformed into a reference element, and the elementary contributions are computed in each element in a decoupled way. In TELEMAC-3D, the Edge-by-Edge technique is applied and there is no global matrix assembly. The reference element used in TELEMAC-3D is shown in the figure 2.3.
The basis functions Φ_i corresponding to the 6 nodes of the reference element read:

$$\begin{aligned} \Phi_1 &= (1-\alpha-\beta)(1-\gamma)/2 & \Phi_4 &= (1-\alpha-\beta)(1+\gamma)/2 \\ \Phi_2 &= \alpha(1-\gamma)/2 & \Phi_5 &= \alpha(1+\gamma)/2 \\ \Phi_3 &= \beta(1-\gamma)/2 & \Phi_6 &= \beta(1+\gamma)/2 \end{aligned} \tag{2.12}$$

The basis functions Ψ_i in the real mesh are obtained by an iso-parametric transformation of Φ_i. The nodes of the prismatic elements in the real mesh have coordinates (x_i,y_i,z_i). The coordinates of a point $P(x,y,z)$ within an element of the real mesh are related to the coordinates in the reference element by:

$$x = \sum_{i=1}^{6} x_i \Phi_i(\alpha,\beta,\gamma) \ , \quad y = \sum_{i=1}^{6} y_i \Phi_i(\alpha,\beta,\gamma) \ , \quad z = \sum_{i=1}^{6} z_i \Phi_i(\alpha,\beta,\gamma) \tag{2.13}$$

Since the lateral faces of the prisms are vertical, some coordinates coincide: $x_1 = x_4$, $y_1 = y_4$,

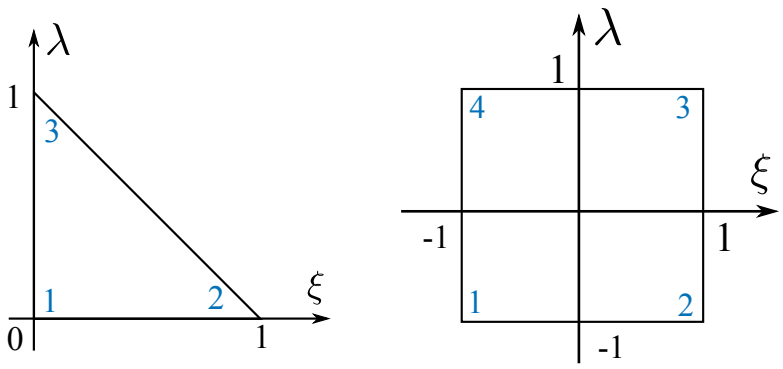

Figure 2.4: Sketch of the 2D reference elements used in TELEMAC-3D. The nodes indices are indicated in blue whereas the coordinates in the local referential of each element are indicated in black.

$x_2 = x_5$, $y_2 = y_5$, $x_3 = x_6$, $y_3 = y_6$, so that the relations (2.13) are simplified into:

$$\begin{aligned} x &= (1-\alpha-\beta)x_1 + \alpha x_2 + \beta x_3 \\ y &= (1-\alpha-\beta)y_1 + \alpha y_2 + \beta y_3 \\ z &= \frac{1}{2}\left[(1-\alpha-\beta)z_1 + \alpha z_2 + \beta z_3\right](1-\gamma) + \frac{1}{2}\left[(1-\alpha-\beta)z_4 + \alpha z_5 + \beta z_6\right](1+\gamma) \end{aligned} \tag{2.14}$$

The Jacobian of this transformation, denoted by $|\boldsymbol{J}|$, reads:

$$\begin{aligned} |\boldsymbol{J}| \quad &= \frac{1}{2}\left[(x_2-x_1)(y_3-y_1) + (x_1-x_3)(y_2-y_1)\right] \\ &\quad \left[(1-\alpha-\beta)z_4 + \alpha z_5 + \beta z_6 - (1-\alpha-\beta)z_1 - \alpha z_2 - \beta z_3\right] \end{aligned} \tag{2.15}$$

Let F be the transformation that from (α,β,γ) yields (x,y,z). The bases in the real mesh Ψ_i are obtained from the bases in the reference element through:

$$\Psi_i(x,y,z) = \Phi_i(F^{-1}(x,y,z)) \tag{2.16}$$

In some parts of the algorithm a two-dimensional approach is needed, as *e.g.* in the free-surface step or for the computation of the friction in the diffusion step. In the case of the 2-D integrals, triangular elements with linear interpolation are used. For vertical lateral boundaries, quadrilaterals with linear interpolation are used. The reference elements are: a triangle with nodes at $(0,0)$, $(0,1)$, $(1,0)$ and a square with nodes $(-1,-1)$, $(1,-1)$, $(1,1)$, $(-1,1)$, as represented in the figure 2.4. The linear interpolation functions for the triangle are:

$$\Phi_1 = (1-\xi-\lambda)\Phi_2 = \xi\Phi_3 = \lambda \tag{2.17}$$

and for the quadrilaterals:

$$\Phi_1 = (1-\xi-\lambda+\xi\lambda)/4 \Phi_2 = (1+\xi-\lambda-\xi\lambda/4 \Phi_3 = (1+\xi+\lambda+\xi\lambda)/4 \Phi_4 = (1-\xi+\lambda-\xi\lambda)/4 \tag{2.18}$$

2.3 Variational principle for boundary value problems

Having restricted our function u to a representation u_h, based on a discrete number of unknowns u_i, we now wish to minimize $L(u_h) - f$. The variational method states that the dot product $\int_\Omega (L(u_h) - f)\varphi_i \, d\Omega$ is zero for some functions φ_i called test functions. The choice of these functions defines the variants of the finite element method. The most classical technique, the Galerkin technique, consists in choosing test functions equal to the basis functions: $\varphi_i = \Psi_i$. In this case, if u_h is decomposed as $\sum_{i=1}^{n} u_i \Psi_i$, the variational method will clearly lead to a linear system of unknowns u_i.
Moreover, to each linear operator L is associated a bilinear form a such as:

$$a(u,v) = \int_\Omega (L(u) - f)\, v \, d\Omega \tag{2.19}$$

The finite element theory aims at determining the conditions on L and on a for which the problem would have a unique solution. An important condition is that, for every non-zero function u, there exists a real α positive, such that $\|L(u)\| \geq \alpha \|u\|$. If the norm L is defined by:

$$\|L\| = \sup_{u \neq 0} \left(\frac{\|L(u)\|}{\|u\|} \right)$$

this condition signifies that L should have a strictly positive norm. Otherwise, if a non-zero function u_p verifies $L(u_p) = 0$, the solution will not be unique because if u is a solution of the equation $L(u) = f$, $u + u_p$ will also be one. Within the variational method frame, the condition is written as:

$$\inf_{u \neq 0} \left(\sup_{v \neq 0} \left(\frac{a(u,v)}{\|u\| \, \|v\|} \right) \right) \geq \alpha \tag{2.20}$$

This is known as the "inf–sup condition". It prevents the occurrence of parasitic solutions which could arise from the bilinear form. For example, for a one-dimensional domain with regularly spaced degrees of freedom, if L is the gradient operator, and if the basis functions Ψ_i are linear and equal to the test functions φ_i, a function u whose values at the nodes are a succession of 1 and -1 (see Figure 2.5) would lead to $\int_\Omega L(u)\varphi_i d\Omega = 0$ for every test function φ_i and the inf–sup condition would not be verified.

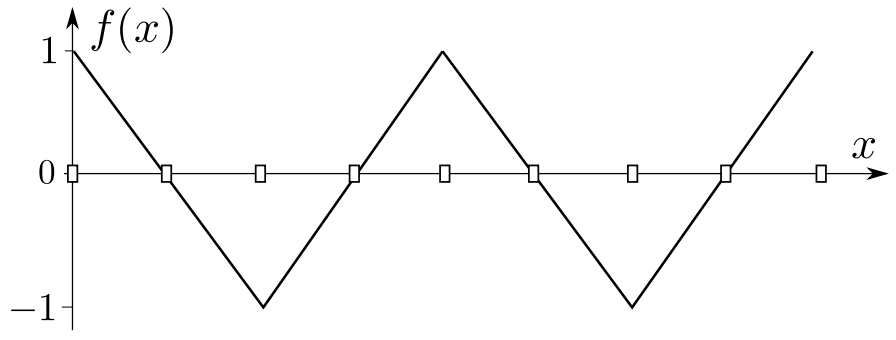

Figure 2.5: An example of invisible function for the gradient operator.

This situation arises when the boundary conditions are not considered, but it can take place also far from the boundaries of a calculation domain. The bilinear forms resulting from the variational formulations of the Navier–Stokes equations do not always satisfy the inf–sup condition. A well-known condition for internal flows of incompressible fluids consists in choosing

different elements to represent pressure and velocity: for example, a quadratic velocity and a linear pressure. Another solution, efficient with regular mesh and the finite difference method, is to have two different and staggered meshes for pressure and velocity. Finally, for elliptic equations, we can use a discretization of the Laplacian, known as "non-compatible", for which the Laplacian is not, at the discrete level, equal to the divergence of the gradient.
A more thorough examination of these questions, especially for the Navier–Stokes equations, can be found in references [18] and [32], and in the more recent work by Pironneau [68] on the methods of finite elements for fluids.

3. Time discretisation in TELEMAC-3D

In this chapter, we describe the time-scheme used in TELEMAC-3D in a continuous-space framework. The aim is to help the reader understand what comes in the chapter 4, where the space-time discretisation is described[1]. In TELEMAC-3D, a fractional-steps method is used for the time discretisation. An important aspect of the algorithm is that the horizontal and vertical velocities are not treated in the same way. Moreover, there are two definitions of the velocity field in the algorithm: the one calculated on the basis of the continuity equation so as to ensure mass conservation and the one calculated on the basis of the momentum equation. Later on, we refer to them as the conservative velocity field (denoted by $\boldsymbol{u}^C$) and the momentum velocity field (denoted by $\boldsymbol{u}$), respectively. We recall that the 2D vector corresponding to the x and y components of a 3D vector is denoted with a $2D$ subscript. On the other hand, we denote with an A superscript the fields obtained after an advection step. The algorithm then consists of several steps, which order depends on the option `DYNAMIC PRESSURE IN WAVE EQUATION`. In TELEMAC-3D, we call wave equation the equation solved to calculate the new water depth values. It is indeed an equation of wave propagation since we do not solve the coupled system linking h^{n+1} and $\boldsymbol{u}^{Cn+1}$, but rather write $\boldsymbol{u}^{Cn+1}$ as a function of h^{n+1} and introduce this formula into the equation on h^{n+1}. There are two options in TELEMAC-3D: either the dynamic pressure is calculated after the values of h^{n+1} and $\boldsymbol{u}^{Cn+1}$ are known, or it is calculated once $\boldsymbol{u}^{Cn+1}$ has been written as a function of h^{n+1} based on an intermediate velocity, which is thereafter corrected through a projection step, and then only h^{n+1} is calculated. We list the different steps below, for each value of the key-word.

3.1 Dynamic pressure calculated after the wave equation resolution

This corresponds to the option `DYNAMIC PRESSURE IN WAVE EQUATION=NO`, which is the one by default in TELEMAC-3D. The algorithm then consists of the following steps:

1. **an advection step for the momentum horizontal velocities**, using the conservative velocity for the advection:

$$\frac{\boldsymbol{u}_{2D}^{A}-\boldsymbol{u}_{2D}^{n}}{\delta t}+\left(\boldsymbol{u}^{Cn}\cdot\boldsymbol{\nabla}\right)\boldsymbol{u}_{2D}^{*}=0 \tag{3.1}$$

[1] It is important to bear in mind that demonstrations of conservation can only be done with the full space-time discretisation.

Remarks about the advection term $\left(\boldsymbol{u}^{C}\cdot\boldsymbol{\nabla}\right)\boldsymbol{u}^{*}_{2D}$:

- $\boldsymbol{u}^{Cn}$ is used as the advecting velocity because it is specifically computed so as to be divergence-free at step 5. When using a conservative advection scheme, this makes it possible to ensure mass-conservation at machine precision, with any discretisation. It is a very important feature of TELEMAC-3D, especially for dilution studies where scalar conservation is of primary importance[a].
- The value of $\boldsymbol{u}^{C}$ from the current time-step, $\boldsymbol{u}^{Cn}$, is used for the advection of the velocity, while the value of the next time-step, $\boldsymbol{u}^{Cn+1}$, is calculated when solving the wave equation. However, for the sake of simplicity in the notations, in the next chapter about space-time discretisation we will not write $\boldsymbol{u}^{Cn}$ or $\boldsymbol{u}^{Cn+1}$, but only $\boldsymbol{u}^{C}$ to have lighter notations.
- the time-discretisation of the advected field $\boldsymbol{u}_{2D}$ is not specified yet, which is accounted for by the $*$ superscript. Indeed, in TELEMAC-3D there are various methods for the computation of the advection term: the methods of characteristics, distributive schemes or finite elements can be used. The basic forms of distributive schemes – N, PSI, are explicit, while more complex forms can be semi-implicited in time. On the other hand, the SUPG scheme is implicit and the characteristics method consists of an exact time-integration between two time-steps. This is dealt with in the section 4.10.

[a] In the studies, the meshes are usually quite coarse, on the one hand because bathymetric data have a limited accuracy, but also to limit computational times, so that this is really a key-feature.

2. **an advection-diffusion step for the momentum vertical velocity**, using the conservative velocity field for the advection and the momentum velocity for the diffusion:

$$\frac{\tilde{w}^{aux}-w^{n}}{\delta t}+\left(\boldsymbol{u}^{Cn}\cdot\boldsymbol{\nabla}\right)w^{*}=-g+\tilde{F}_{z}+\nabla\cdot\left(\nu_{E}w^{*}\cdot\boldsymbol{e}_{z}\right) \tag{3.2}$$

Remark:

In this system, the term $\tilde{F}_z$ contains the buoyancy forces, Coriolis force, tidal force, source points inputs and outputs (isolated or linked through the culvert fomulation), rain and evaporation. If present, these terms are treated explicitly.

Then, the diffusion term can be partially or fully implicited using a coefficient of implicitation for the diffusion θ_d, so that this step reads:

$$\frac{\tilde{w}^{aux}-w^{n}}{\delta t}+\left(\boldsymbol{u}^{Cn}\cdot\boldsymbol{\nabla}\right)w^{*}=-g+\tilde{F}_{z}+\nabla\cdot\left(\nu_{E}\theta_{d}\boldsymbol{\nabla}\tilde{w}^{aux}+\nu_{E}(1-\theta_{d})\boldsymbol{\nabla}w^{n}\right) \tag{3.3}$$

Remark:

When applying the Finite Elements space-discretisation to this step, some diffusion boundary terms appear (friction terms), which may actually be treated implicitely or explicitely. This is described in the section 4.4.

3. **a hydrostatic step**, which consists in computing the new depth together with the conservative horizontal velocities:

$$\begin{cases} \dfrac{\eta^{n+1}-\eta^n}{\delta t} + \nabla_{2D} \cdot \displaystyle\int_b^\eta \tilde{\boldsymbol{u}}_{2D}^{n+1}\, dz = 0 \\ \dfrac{\tilde{\boldsymbol{u}}_{2D}^{n+1}-\boldsymbol{u}_{2D}^{A}}{\delta t} = -g\boldsymbol{\nabla}_{2D}\eta^{n+1} + \tilde{\boldsymbol{F}}_{2D} + \boldsymbol{\nabla}\cdot(\nu_E \boldsymbol{\nabla}\boldsymbol{u}_{2D}^n) \end{cases} \tag{3.4}$$

This system corresponds to the lines 1, 2 and 4 of (1.259), excluding the advection terms since they have already been treated in the previous fractional step. The diffusion term was written here in a fully explicit form for the sake of simplicity but the user actually has the choice to partially implicit it using a coefficient θ_d. This will be described with more details in the section 4.4.2.

By replacing η by $h+b$ in the second line of (3.4) and partially impliciting the water depth with a coefficient θ_h, the term $\boldsymbol{\nabla}_{2D}\eta^{n+1}$ of the second line is actually re-written:

$$\boldsymbol{\nabla}_{2D}(\theta_h\eta^{n+1} + (1-\theta_h)\eta^n) = \boldsymbol{\nabla}_{2D}\left(\theta_h h^{n+1} + (1-\theta_h)h^n + b\right) \tag{3.5}$$

We used the fact that b is considered constant in time here. The user also has the choice to partially implicit the velocity field during this step, using a coefficient θ_u in the first line of the system (3.4). The system (3.4) is then modified and reads:

$$\begin{cases} \dfrac{h^{n+1}-h^n}{\delta t} + \nabla_{2D} \cdot \displaystyle\int_b^\eta \left(\theta_u \tilde{\boldsymbol{u}}_{2D}^{n+1} + (1-\theta_u)\boldsymbol{u}_{2D}^n\right)\, dz = 0 \\ \dfrac{\tilde{\boldsymbol{u}}_{2D}^{n+1}-\boldsymbol{u}_{2D}^{A}}{\delta t} = -g\theta_h\boldsymbol{\nabla}_{2D}(h^{n+1}-h^n) - g\boldsymbol{\nabla}_{2D}\eta^n + \tilde{\boldsymbol{F}}_{2D} + \boldsymbol{\nabla}\cdot(\nu_E \boldsymbol{\nabla}\boldsymbol{u}_{2D}^n) \end{cases} \tag{3.6}$$

For the resolution of this system, the expression of $\tilde{\boldsymbol{u}}_{2D}^{n+1}$ given by the second line is injected into the first line. This yields a propagation equation on η^{n+1}. The system (3.4) is then modified and reads:

$$\begin{cases} \dfrac{h^{n+1}-h^n}{\delta t} + \nabla_{2D} \cdot \displaystyle\int_b^\eta -\delta t\,\theta_u g\theta_h \boldsymbol{\nabla}_{2D}(h^{n+1}-h^n)\, dz = \\ \qquad\qquad -\nabla_{2D} \cdot \displaystyle\int_b^\eta \left[\theta_u \boldsymbol{u}_{2D}^{aux} + (1-\theta_u)\boldsymbol{u}_{2D}^n\right]\, dz \\ \dfrac{\boldsymbol{u}_{2D}^{aux}-\boldsymbol{u}_{2D}^{A}}{\delta t} = -g\boldsymbol{\nabla}_{2D}\eta^n + \tilde{\boldsymbol{F}}_{2D} + \boldsymbol{\nabla}\cdot(\nu_E\theta_d\boldsymbol{\nabla}\boldsymbol{u}_{2D}^{aux} + \nu_E(1-\theta_d)\boldsymbol{\nabla}\boldsymbol{u}_{2D}^n) \\ \tilde{\boldsymbol{u}}_{2D}^{n+1} = \boldsymbol{u}_{2D}^{aux} - g\theta_h\delta t\boldsymbol{\nabla}_{2D}(h^{n+1}-h^n) \end{cases} \tag{3.7}$$

where we display the scheme using θ_d for partial implicitation of the diffusion. Solving the first line of this system provides the value of h^{n+1}. The field $\boldsymbol{u}_{2D}^{Cn+1}$ to be used for the advection at the next time-step is then defined by:

$$\boldsymbol{u}_{2D}^{Cn+1} = \theta_u \boldsymbol{u}_{2D}^{aux} + (1-\theta_u)\boldsymbol{u}_{2D}^n \tag{3.8}$$

so as to be consistent with the conservation of the scalars (see the section 4.8.2).

4. **a step of mesh motion** where the new layers elevations are calulcated based on the new values of water depth.

5. **a pressure step** for computing the dynamic pressure and calculating the momentum velocity field:

$$\begin{cases} \nabla^2 p_d^{n+1} = \dfrac{\rho}{\delta t} \nabla \cdot \tilde{\boldsymbol{u}}^{n+1} \\ \dfrac{\boldsymbol{u}^{n+1} - \tilde{\boldsymbol{u}}^{n+1}}{\delta t} = -\dfrac{1}{\rho} \boldsymbol{\nabla} p_d^{n+1} \end{cases} \tag{3.9}$$

where $\tilde{\boldsymbol{u}}^{n+1}$ is defined by:

$$\tilde{\boldsymbol{u}}^{n+1} = \boldsymbol{u}_{2D}^{aux} - \delta t g \theta_h \boldsymbol{\nabla}_{2D}(h^{n+1} - h^n) + \tilde{w}^{aux} \boldsymbol{e}_z \tag{3.10}$$

> **Remark:**
> This step corresponds to the resolution of the Navier–Stokes equations using a Chorin-Temam type projection method (see the section 1.4.3) where $\tilde{\boldsymbol{u}}^{n+1}$ is the predicted velocity field.

6. a step for the **computation of the conservative vertical velocity**:

$$\frac{\partial w^{Cn+1}}{\partial z} = -\frac{\partial u^{Cn+1}}{\partial x} - \frac{\partial v^{Cn+1}}{\partial y}, \; w^C|_b = 0 \tag{3.11}$$

7. **a final advection-diffusion step for scalars** (including k and ε when the $k-\varepsilon$ turbulence model is used), using the conservative velocity field for the advection:

$$\begin{cases} \dfrac{k^{n+1} - k^n}{\delta t} + \boldsymbol{u}^{Cn+1} \cdot \boldsymbol{\nabla} k^* = \mathbb{P}^n + \mathbb{G}^n - \varepsilon^n \dfrac{k^{n+1}}{k^n} + \nabla \cdot (\nu_k^n \theta_d \boldsymbol{\nabla} k^{n+1} + \nu_k^n (1-\theta_d) \boldsymbol{\nabla} k^n) \\ \dfrac{\varepsilon^{n+1} - \varepsilon^n}{\delta t} + \boldsymbol{u}^C \cdot \boldsymbol{\nabla} \varepsilon^* = \dfrac{\varepsilon^n}{k^n} \left(C_{\varepsilon_1} \mathbb{P}^n + C_{\varepsilon_3} \mathbb{G}^n - C_{\varepsilon_2,Y}^n \varepsilon^{n+1} \right) + \nabla \cdot (\nu_\varepsilon^n \theta_d \boldsymbol{\nabla} \varepsilon^{n+1} + \nu_\varepsilon^n (1-\theta_d) \boldsymbol{\nabla} \varepsilon^n) \end{cases} \tag{3.12}$$

See the section 1.6.3 for the definitions of $\mathbb{P}$, $\mathbb{G}$, μ_k, μ_ε, etc. ν_T^{n+1} is then calculated through the equation (1.128) and for remaining scalars (temperature, salinity, etc), the following time-discretised equation is solved:

$$\frac{T^{n+1} - T^n}{\delta t} + \boldsymbol{u}^{Cn+1} \cdot \boldsymbol{\nabla} T^* = F_{source} + \nabla \cdot (K_T^{n+1} \theta_d \boldsymbol{\nabla} T^{n+1} + K_T^{n+1} (1-\theta_d) \boldsymbol{\nabla} T^n) \tag{3.13}$$

> **Hydrostatic model**
> In TELEMAC-3D, it is possible to solve the Navier–Stokes equations with the assumption of hydrostatic pressure: the pressure step (step 5) is then entirely skipped. However, we highly recommend the use the non-hydrostatic option given the poor results obtained with the hydrostatic option on some cases, in particular when active scalars have a significant effect on the flow or for wave simulations.

3.2 Dynamic pressure calculated during the wave equation resolution

This corresponds to the option `DYNAMIC PRESSURE IN WAVE EQUATION=YES`. The algorithm is then modified compared to the one above:

1. **the advection step for the momentum horizontal velocities** is unchanged;

2. **the advection-diffusion step for the momentum vertical velocity** is also unchanged;

3. **the hydrostatic step**, is modified. The equations to be solved now read:

$$\left\{\begin{array}{ll} \dfrac{h^{n+1}-h^n}{\delta t}+\nabla_{2D}\cdot\displaystyle\int_b^\eta \left(\theta_u \boldsymbol{u}_{2D}^{n+1}+(1-\theta_u)\boldsymbol{u}_{2D}^n\right)\,dz=0 & \\ \dfrac{\boldsymbol{u}_{2D}^{n+1}-\boldsymbol{u}_{2D}^n}{\delta t} & =-g\theta_h\boldsymbol{\nabla}_{2D}(h^{n+1}-h^n)-g\boldsymbol{\nabla}_{2D}\eta^n+\tilde{\boldsymbol{F}}_{2D} \\ & +\boldsymbol{\nabla}\cdot(\nu_E\theta_d\boldsymbol{\nabla}\tilde{\boldsymbol{u}}_{2D}^{aux}+\nu_E(1-\theta_d)\boldsymbol{\nabla}\boldsymbol{u}_{2D}^n)-\dfrac{1}{\rho}\boldsymbol{\nabla}p_d^n \end{array}\right. \tag{3.14}$$

First, a predicted velocity field $\tilde{\boldsymbol{u}}^{aux}$ is calculated:

$$\frac{\tilde{\boldsymbol{u}}_{2D}^{aux}-\boldsymbol{u}_{2D}^A}{\delta t}=-g\boldsymbol{\nabla}_{2D}\eta^n+\tilde{\boldsymbol{F}}_{2D}+\boldsymbol{\nabla}\cdot(\nu_E\theta_d\boldsymbol{\nabla}\tilde{\boldsymbol{u}}_{2D}^{aux}+\nu_E(1-\theta_d)\boldsymbol{\nabla}\boldsymbol{u}_{2D}^n) \tag{3.15}$$

The projection method is then applied so as to calculate the dynamic pressure at this stage and to find a divergence-free velocity field $\boldsymbol{u}^{aux}$:

$$\left\{\begin{array}{l} \nabla^2 p_d^{n+1}=\dfrac{\rho}{\delta t}\nabla\cdot\tilde{\boldsymbol{u}}^{aux} \\ \dfrac{\boldsymbol{u}^{aux}-\tilde{\boldsymbol{u}}^{aux}}{\delta t}=-\dfrac{1}{\rho}\boldsymbol{\nabla}p_d^{n+1} \end{array}\right. \tag{3.16}$$

The expression of $\boldsymbol{u}_{2D}^{aux}$ is then used to replace $\boldsymbol{u}_{2D}^{n+1}$ in the first line of (3.14):

$$\begin{array}{c} \dfrac{h^{n+1}-h^n}{\delta t}+\nabla_{2D}\cdot\displaystyle\int_b^\eta -\delta t\,\theta_u g\theta_h\boldsymbol{\nabla}_{2D}(h^{n+1}-h^n)\,dz= \\ -\nabla_{2D}\cdot\displaystyle\int_b^\eta \left[\theta_u\boldsymbol{u}_{2D}^{aux}+(1-\theta_u)\boldsymbol{u}_{2D}^n\right]\,dz \end{array} \tag{3.17}$$

Solving this system provides the value of h^{n+1}. The value of $\boldsymbol{u}_{2D}^{Cn+1}$ is then obtained through:

$$\boldsymbol{u}_{2D}^{Cn+1}=\theta_u\boldsymbol{u}_{2D}^{aux}+(1-\theta_u)\boldsymbol{u}_{2D}^n \tag{3.18}$$

4. **a step of mesh motion**, which is unchanged;

5. a step for the **computation of the conservative vertical velocity**, which is unchanged;

6. **a final advection-diffusion step for scalars**, also unchanged.

4. Space-time discretisation in TELEMAC-3D

The chosen spatial discretization is a finite element discretization using prisms. Each prism has six nodes with quadrangular vertical sides. The prism with linear interpolation has been retained because its two-dimensional horizontal projection constitutes one of the finite elements (triangle with linear interpolation) used to resolve the Saint-Venant equations in two dimensions. It is thus possible to build a three-dimensional mesh from a two-dimensional mesh. It only requires to mesh the two-dimensional domain Ω_{2D} with triangles, then to duplicate this mesh vertically as sketched in the figure 4.1. Calculation in two and three dimensions with the same mesh is thus possible. In this chapter, we drop the time superscripts in the notation of the conservative velocity field $\boldsymbol{u}^{C}$ for the sake of simplicity in the notations. It is always $\boldsymbol{u}^{Cn}$ when involved in the advection of the velocity, and $\boldsymbol{u}^{Cn+1}$ everywhere else.

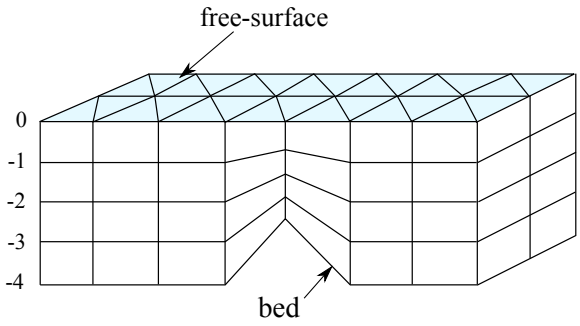

Figure 4.1: 3-D mesh built from a 2-D mesh with a constant proportionality coefficient per plane.

4.1 Building the three-dimensional mesh

To build the three-dimensional domain with prisms, it is possible to build the triangulation of the two-dimensional domain, then to repeat it along the vertical in superimposed layers (called "planes"). These "planes" are in fact surfaces with variable elevations. The only condition is that the elevation of the points belonging to a vertical line should increase from the bed to the free surface. Several simple formulae guarantee this condition, as in the following four methods.

4.1.1 Method 1: planes evenly spaced along the vertical

This first option, the most classic, consists of starting with the coordinate nodes (x,y) of the 2D mesh, then defining the elevation for each point of the 3D mesh with coordinates (x,y,z), according to the formula:

$$z(x,y,ip) = b(x,y) + \frac{ip-1}{np-1}\left(\eta(x,y) - b(x,y)\right) \tag{4.1}$$

b always being the bed elevation, η the free surface elevation and ip is the rank of the plane under consideration (planes range from 1 to np if we go from the bed to the free surface). This is the simplest method based on the sigma transformation (see the section 1.9. In this case the vertical coordinate z^* of each plane in the transformed mesh is simply $(ip-1)/(np-1)$. The figure 4.1 is a representation of such a mesh.

4.1.2 Method 2: data of a function θ constant for each plane

The coordinate z^* of plane ip in the transformed mesh is no longer $(ip-1)/(np-1)$ but a given number θ_{ip} such that $0 \leq \theta_{ip} \leq 1$ and, for intermediary planes, $\theta_{ip-1} < \theta_{ip} < \theta_{ip+1}$. We now have:

$$z(x,y,ip) = b(x,y) + \theta_{ip}\left(\eta(x,y) - b(x,y)\right) \tag{4.2}$$

The figure 4.2 shows an example of a 3D mesh built with the following values: $\theta_1 = 0$, $\theta_2 = 1/6$, $\theta_3 = 1/3$, $\theta_4 = 2/3$, $\theta_5 = 1$, where the planes are numbered from the bed to the free-surface. The choice of values of θ_{ip} is free, the only constraints being a value of 0 for the bed ($ip = 1$), a value of 1 for the free surface ($ip = np$) and a series of intermediate values in a strictly increasing order.

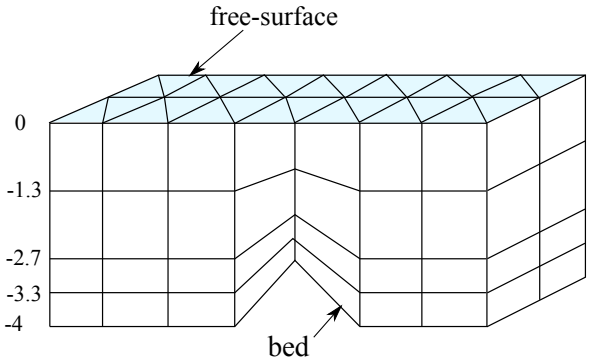

Figure 4.2: Mesh built with a constant proportionality coefficient per plane.

4.1.3 Method 3: double sigma transform

Here we prescribe a constant elevation for one plane, then the planes between the bed and this plane are evenly spaced, and the same is done between this plane and the free surface.

4.1.4 Method 4: horizontal planes

In order to better represent the temperature stratification zones (thermoclines), it is sometimes worth imposing a certain number of intermediate planes to be horizontal. In this case, it is possible to indicate the desired elevation for each plane, which will be denoted by z_{ip}. This elevation is not always reachable because it can be located below the bed or above the free

surface. In this case, the elevation of the point of the plane ip will be maintained slightly above the bed, or slightly below the free-surface. This can be done with the help of a "MinMod" limiter through the formula:

$$z = \min\left\{\max\left[b(x,y)+\frac{ip-1}{np-1}d_{\min}, z_{ip}\right], \eta(x,y,t)-\frac{np-ip}{np-1}d_{\min}\right\} \tag{4.3}$$

where d_{min} is an imposed minimum distance. We impose that plane 1 (the bed) is at a minimum distance $d_{\min}$ from the free surface, and that the plane np (the free surface) is at a minimum distance d_{min} from the bed. The figure 4.3 shows an example of a 3D mesh built with this method and the following prescribed elevations z_{ip}: $z_1 = -4m$, $z_2 = -3m$, $z_3 = -2m$, $z_4 = -1m$, $z_5 = 0m$.

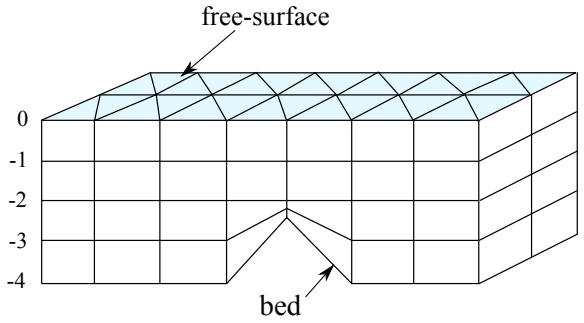

Figure 4.3: Example of a 3D mesh with a prescribed horizontal plane at $z = -2m$.

In conformity with the earlier formula, when the bed rises to the level of plane number 2 normally situated at elevation -3 m, this plane number 2 then adopts the form of the bed, at a distance $d_{min}/4$.

Comment by J.-M. Hervouet

The advantage of this distribution is the elimination of truncation errors in the calculation of buoyancy terms. These errors can generate unacceptable instability when these terms are dominant. Actually, when the planes are not horizontal, it is impossible to correctly represent a function f that only depends on z. Now, the density of the fluid should tend to this type of function corresponding to a position of equilibrium without horizontal pressure gradients. With this method, non-horizontal mesh planes are inevitable, but only at a distance less than d_{min} from the bed or from the surface. In these zones with quasi-zero volume, it is suggested that the forces arising from the parasitic horizontal pressure gradients (called hydrostatic inconsistencies) be cancelled by the intermediary of a filter which acts in places where the prisms are, such that a node of the lower base (number 1, 2 or 3) has an elevation higher than a node of the upper base (number 4, 5 or 6). This filter also cancels the different diffusion coefficients on these elements.

4.2 Sigma transformation of the 3D mesh

As the free surface evolves with time, the elevations z of the 3D mesh vary from one time step to another. However, it is possible to change referentials so as to fix the mesh during a time step. As we have already seen in the section 1.9, this can be particularly interesting in the advection step, which has a definite influence on the evolution of the free surface. The change of variables adopted in TELEMAC-3D is the sigma transformation, which consists in passing

from the coordinate z to the coordinate z^*, the latter being independent of time. For the classical sigma transformation, we recall the formula used:

$$z^* = \frac{z-b}{\eta-b} \tag{4.4}$$

When a plane has a prescribed constant elevation, a double sigma transform is preferred, with a function z^* varying between -1 and $+1$. If $z_{prescribed}$ is the required constant elevation, we have below the fixed plane:

$$z^* = \frac{z-z_{prescribed}}{z_{prescribed}-b} \tag{4.5}$$

and above:

$$z^* = \frac{z-z_{prescribed}}{\eta-z_{prescribed}} \tag{4.6}$$

For a generalized sigma transform, it is convenient to use the following function z^* between planes ip and $ip+1$:

$$z^* = ip - 1 + \frac{z-z_{ip}}{z_{ip+1}-z_{ip}} \tag{4.7}$$

where z_{ip} is the elevation of plane ip, and z_{ip+1} the elevation of plane $ip+1$. This function z^* now varies between 0 and $np-1$. The definition of z^* is thus univocal over the entire vertical.

Remark:

These forms of transformation are only necessary when dealing with the full 3D mesh, *e.g.* during an advection step with the method of characteristics. For local computations within a single finite element, a local definition is simpler, which reads:

$$z^* = \frac{z-z_{ip}}{z_{ip+1}-z_{ip}} \tag{4.8}$$

i.e. we have a classic sigma transformation between two planes.

It is interesting to notice that the four methods given above to build the mesh belong to two different families:

- the first two are such that $\partial z/\partial z^* = h$, they thus rely on the classic sigma transformation
- for the last two, which correspond to generalized sigma transformations:

$$\frac{\partial z}{\partial z^*} = z_{ip+1} - z_{ip} = \Delta z \tag{4.9}$$

i.e. the width between two planes, denoted by Δz. It is only piecewise constant vertically.

4.3 Space discretisation of the diffusion step

Depending on the chosen advection scheme, the diffusion and advection are either treated implicitly at the same time (SUPG scheme) or successively, first calculating the advection (with the characteristics method or a distributive scheme) and then adding the diffusion to the advected field. In this section we only consider the discretisation of the diffusion term. In this section, we denote with a D superscript the fields obtained after the diffusion has been calculated. Remember from the chapter 3 that the diffusion is hardly ever calculated on its own: it can only be the case for the vertical velocity prediction, when the advection and diffusion are treated separately.

For the conditions of existence and uniqueness of the solution for the diffusion problem, which is parabolic, the reader can refer to Raviart and Thomas [75] or to the book by Pironneau [68] (in French).

Given the complexity of a diffusion stage with the sigma transform, and the numerical artefacts which are its consequence, this stage is executed in the real mesh. The variational formulation of the diffusion step reads:

$$\begin{aligned}\int_\Omega \boldsymbol{u}^D \cdot \boldsymbol{\Psi}_i \, d\Omega - \delta t \theta_d \int_\Omega \nabla \cdot (\nu_E \boldsymbol{\nabla} \boldsymbol{u}^D) \cdot \boldsymbol{\Psi}_i \, d\Omega \quad &= \int_\Omega \boldsymbol{u}^A \cdot \boldsymbol{\Psi}_i \, d\Omega \\ &+ \delta t (1-\theta_d) \int_\Omega \nabla \cdot (\nu_E \boldsymbol{\nabla} \boldsymbol{u}^A) \cdot \boldsymbol{\Psi}_i \, d\Omega\end{aligned} \tag{4.10}$$

where θ_d is the coefficient of implicitation of the diffusion. The vectorial bases $\boldsymbol{\Psi}_i$ correspond to a P1 basis on the prism for each space direction. In the left-hand side, the second term is written as:

$$\begin{aligned}\int_\Omega \nabla \cdot (\nu_E \boldsymbol{\nabla} \boldsymbol{u}^D) \cdot \boldsymbol{\Psi}_i \, d\Omega \quad &= \int_\Omega \nabla \cdot (\nu_E \boldsymbol{\nabla} \boldsymbol{u}^D \cdot \boldsymbol{\Psi}_i) \, d\Omega - \int_\Omega \nu_E \boldsymbol{\nabla} \boldsymbol{\Psi}_i : \boldsymbol{\nabla} \boldsymbol{u}^D \, d\Omega \\ &= \int_\Gamma \nu_E \boldsymbol{\nabla} \boldsymbol{u}^D \cdot \boldsymbol{\Psi}_i \cdot \boldsymbol{n} \, d\Gamma - \int_\Omega \nu_E \boldsymbol{\nabla} \boldsymbol{\Psi}_i : \boldsymbol{\nabla} \boldsymbol{u}^D \, d\Omega\end{aligned} \tag{4.11}$$

The second line is obtained using the Gauss theorem on the integral of the divergence. The corresponding boundary integral corresponds to the friction on the boundaries, which is prescribed. The Finite Element discretisation of the term $\boldsymbol{\nabla} \boldsymbol{u}^D$ reads:

$$\boldsymbol{\nabla} \boldsymbol{u}^D = \sum_j \boldsymbol{u}_j^D \boldsymbol{\nabla} \boldsymbol{\Psi}_j \tag{4.12}$$

The Finite Element discretisation of the equation (4.10) then reads:

$$\begin{aligned}\boldsymbol{u}_j^D \int_\Omega \boldsymbol{\Psi}_i \cdot \boldsymbol{\Psi}_j \, d\Omega + \delta t \theta_d \nu_{Ej} \boldsymbol{u}_j^D \int_\Omega \boldsymbol{\nabla} \boldsymbol{\Psi}_i : \boldsymbol{\nabla} \boldsymbol{\Psi}_j \, d\Omega - \boldsymbol{u}_j^D \int_\Gamma \nu_E \boldsymbol{\nabla} \boldsymbol{\Psi}_j \cdot \boldsymbol{\Psi}_i \cdot \boldsymbol{n} \, d\Gamma \quad &= \boldsymbol{u}_j^A \int_\Omega \boldsymbol{\Psi}_i \cdot \boldsymbol{\Psi}_j \, d\Omega \\ &+ \delta t (1-\theta_d) \nu_{Ej} \boldsymbol{u}_j^A \int_\Omega \boldsymbol{\nabla} \boldsymbol{\Psi}_i : \boldsymbol{\nabla} \boldsymbol{\Psi}_j \, d\Omega\end{aligned} \tag{4.13}$$

for all degrees of freedom i, j. The construction of the diffusion matrix in the real mesh composed of prisms of arbitrary shape is not easy because the Jacobian of the isoparametric transformation between an arbitrary prism and the reference prism appears in the denominator in the calculation of the diffusion matrix. Indeed, the matrix terms corresponding to the second integral in the left-hand side are calculated in the reference element:

$$\int_\Omega \boldsymbol{\nabla} \boldsymbol{\Psi}_i : \boldsymbol{\nabla} \boldsymbol{\Psi}_j \, d\Omega = \int_{\hat{\Omega}} \boldsymbol{\nabla} \boldsymbol{\Psi}_i : \boldsymbol{\nabla} \boldsymbol{\Psi}_j |\boldsymbol{J}| \, d\hat{\Omega} \tag{4.14}$$

where $\hat{\Omega}$ is the transformation of Ω into a set of reference prisms through F, defined by (2.16) and $|\boldsymbol{J}|$ is the Jacobian of F, defined by (2.15). The terms $\boldsymbol{\nabla} \boldsymbol{\Psi}_i$ are then written (for example for the first coefficient of the matrix):

$$(\boldsymbol{\nabla} \boldsymbol{\Psi}_i)_{xx} = \frac{\partial \Psi_{xi}}{\partial \alpha} \frac{\partial \alpha}{\partial x} \tag{4.15}$$

and the term $\partial \alpha / \partial x$ is equal to $1/|\boldsymbol{J}|$. However, this Jacobian is not a constant, therefore we cannot get a polynomial expression for the matrix coefficients and their integration over $\hat{\Omega}$ is

tricky. It is however possible to find an approximation of the diffusion matrix on prisms, *e.g.* by considering a constant average Jacobian on every prism to get a polynomial expression of the coefficients. This is what is done at the moment in TELEMAC-3D.

This problem does not exist with tetrahedra and a linear interpolation, the Jacobian then being a constant. Now, a prism can be broken down into three tetrahedra (see the figure 4.4, where the tetrahedra are formed by the sets of four points (3,1,2,6), (4,6,5,1) and (5,2,1,6)). Then, let us consider that the interpolation on the prism is linear by pieces, exactly as in a quasi-bubble triangle. As no degree of freedom is added, the topology of a prismatic element matrix can be formally retained. After calculating the element matrices on the tetrahedra, a pre-assembly restores the element matrices prism by prism. Writing this pre-assembly shows that all the properties of a diffusion matrix (sum of the terms of each line equal to zero, sum of the terms of each column equal to zero) are actually transferred from the tetrahedra to the prisms. This technique has been implemented in TELEMAC-3D but is not used at the moment. The advantage of this technique compared to the approximate diffusion matrix on prisms is not obvious in preliminary numerical tests.

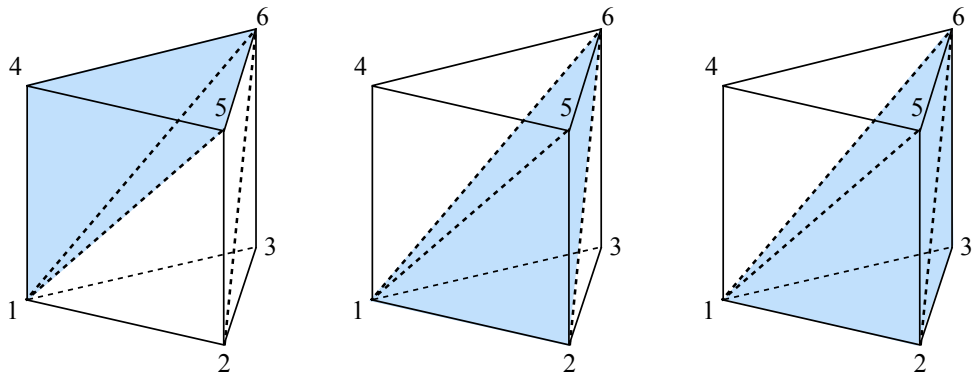

Figure 4.4: Decomposition of a prism into three tetrahedra.

Important remark:

The diffusion matrix should only possess negative extra-diagonal terms and the sum of each line should always yield zero in order to achieve monotonicity and conservation. However, from the computational point of view the extra-diagonal terms can happen to be positive. Thus, the diffusion matrix in TELEMAC-3D is modified: all extra-diagonal terms are clipped to a maximum value of zero and the diagonal terms are modified so as to always retrieve a zero-sum of each line.

With distributive schemes or when the user chooses to explicit the diffusion ($\theta_d = 0$), the first term in the left-hand side is mass-lumped and (4.13) reads:

$$\begin{aligned} \boldsymbol{u}_j^D \int_\Omega \boldsymbol{\Psi}_i \, d\Omega + \delta t \theta_d \nu_{Ej} \boldsymbol{u}_j^D \int_\Omega \boldsymbol{\nabla}\boldsymbol{\Psi}_i : \boldsymbol{\nabla}\boldsymbol{\Psi}_j \, d\Omega \quad &= \boldsymbol{u}_j^A \int_\Omega \boldsymbol{\Psi}_i \boldsymbol{\Psi}_j \, d\Omega \\ + \delta t (1-\theta_d) \boldsymbol{u}_j^A \int_\Omega \boldsymbol{\Psi}_i \nabla \cdot (\nu_{Ej} \boldsymbol{\nabla}\boldsymbol{\Psi}_j) \, d\Omega + \int_\Gamma \nu_E \boldsymbol{\nabla}\boldsymbol{u}^D \cdot \boldsymbol{\Psi}_i \cdot \boldsymbol{n} \, d\Gamma & \end{aligned} \tag{4.16}$$

Mass-lumping:
The mass-lumping corresponds to a modification of a matrix to obtain a diagonal matrix where all the extra-diagonal components were summed onto the diagonal, line by line. This constitutes an approximation of the original matrix but simplifies the resolution of the system since only a division by the diagonal is necessary.

Remark:
The calculation of $\int_\Omega \Psi_i \, d\Omega$ in the left-hand side is done in agreement with the choice of mass-lumping for the water heigth term in the wave equation (see the section 4.4.3).

4.3.1 Buoyancy terms

The source terms of buoyancy are one of the strategic points of 3D equations. Their correct treatment conditions the representation of stratifications, a current stumbling block of modelling. These source terms are, if we do not consider atmospheric pressure (see the section 1.5.3):

$$F_x = g\frac{\Delta\rho}{\rho_o}\left(\frac{\partial\eta}{\partial x}\right)_{y,t} - g\frac{\partial}{\partial x}\left(\int_z^\eta \frac{\Delta\rho}{\rho_o}\,dz\right)_{y,z,t} \tag{4.17}$$

$$F_y = g\frac{\Delta\rho}{\rho_o}\left(\frac{\partial\eta}{\partial y}\right)_{x,t} - g\frac{\partial}{\partial y}\left(\int_z^\eta \frac{\Delta\rho}{\rho_o}\,dz\right)_{x,z,t} \tag{4.18}$$

The terms F_x and F_y are calculated on the mesh at time t^n on the basis of differences in density $\Delta\rho$. These differences in density themselves have been evaluated earlier with the help of a state function in which the different concentrations of active scalars appear. Bijvelds [11] compares the treatment of these terms in sigma transform formulation, in different software. The term F_x can be calculated either in the real mesh or in the transformed mesh. In TELEMAC-3D, the calculation is done in the real mesh.

Calculation in the real mesh:
the second term of F_x cannot be directly expressed in the form of vectors in finite elements. It has to be transformed with the Leibnitz formula to obtain:

$$-g\frac{\partial}{\partial x}\left(\int_z^\eta \frac{\Delta\rho}{\rho_o}\,dz\right)_{y,z,t} = -g\left[\int_z^\eta \frac{\partial}{\partial x}\left(\frac{\Delta\rho}{\rho_o}\right)dz + \frac{\Delta\rho(\eta)}{\rho_o}\frac{\partial\eta}{\partial x}\right] \tag{4.19}$$

We finally get:

$$F_x = -g\int_z^\eta \frac{\partial}{\partial x}\left(\frac{\Delta\rho}{\rho_o}\right)dz + g\frac{\partial\eta}{\partial x}\left(\frac{\Delta\rho(z)}{\rho_o} - \frac{\Delta\rho(\eta)}{\rho_o}\right) \tag{4.20}$$

This formula is simply calculated by using vertical lines of the mesh for the integral along z and to find the value of $\Delta\rho(\eta)$ above one point of elevation z.

Remark:
The Appendix B 6 explains how these terms can be calculated in the transformed mesh. TELEMAC-3D contains the corresponding code but it is not used at the moment.

4.3.2 Friction terms

Compatibility between friction terms in 2D and 3D is discussed in [36]; however, this is only valid for formulae relying on a depth-integrated velocity, such as those of Strickler or Chézy. If we want to use a local roughness factor and assume a logarithmic profile near the bed, the friction velocity u_* has to be evaluated. We can use the fact that the mesh of prisms is made of piled-up meshes of triangles and consider the first plane above the bed. If we assume that this first plane is still in the logarithmic profile, we can write (the example of a hydraulically rough flow):

$$u_* = \frac{\kappa u_{plane\,1}}{\ln\left(\frac{33\Delta z}{k_s}\right)} \tag{4.21}$$

where $u_{plane\,1}$ is the velocity at the first point above the bed and Δz the altitude of this point above the bed.

4.3.3 Other source terms

The other source terms included in the terms F_x and F_y are also treated in the diffusion stage, explicitly (Coriolis, source points, culverts, rain and evaporation, tidal forcing, etc.).

4.4 Space discretisation of the hydrostatic step

Here we only consider the case where the dynamic pressure is calculated after the resolution of the hydrostatic step. We recall that we wrote the following system for the calculation of $h^{n+1}-h^n$ and $\tilde{\boldsymbol{u}}_{2D}^{n+1}$ in the section 3:

$$\begin{cases} \dfrac{h^{n+1}-h^n}{\delta t} + \nabla_{2D}\cdot\displaystyle\int_b^\eta \left(\theta_u \tilde{\boldsymbol{u}}_{2D}^{n+1} + (1-\theta_u)\boldsymbol{u}_{2D}^n\right)\, dz = 0 \\ \dfrac{\tilde{\boldsymbol{u}}_{2D}^{n+1}-\boldsymbol{u}_{2D}^A}{\delta t} = -g\theta_h \boldsymbol{\nabla}_{2D}(h^{n+1}-h^n) - g\boldsymbol{\nabla}_{2D}\eta^n + \tilde{\boldsymbol{F}}_{2D} + \nabla\cdot(\nu_E \boldsymbol{\nabla}\boldsymbol{u}_{2D}) \end{cases} \tag{4.22}$$

The time-discretisation for the diffusion term in the second line will be specified later. In the sub-sections 4.6.1 and 4.4.2, a variational formulation for the depth-integrated continuity equation (first line of this system) and for the horizontal momentum equation (second line) are derived. The latter will provide an expression of $\tilde{\boldsymbol{u}}_{2D}^{n+1}$, that is introduced in the first line in order to calculate h^{n+1}. Once h^{n+1} is known, the value of $\tilde{\boldsymbol{u}}_{2D}^{n+1}$ is retrieved through the horizontal momentum equation. In between, a conservative velocity $\boldsymbol{u}_{2D}^C$ will have been calculated, which is the one used for the advection terms.

4.4.1 Variational formulation of the depth-integrated continuity equation in the transformed mesh

The time-space discretised depth-averaged continuity equation is derived starting from the continuity equation $\nabla\cdot\boldsymbol{u}=0$ written in the transformed mesh (see the equation 1.256):

$$\frac{1}{\Delta z}\left[\frac{\partial \Delta z}{\partial t} + \left(\frac{\partial \Delta z u}{\partial x}\right)_{y,z^*,t} + \left(\frac{\partial \Delta z v}{\partial y}\right)_{x,z^*,t} + \left(\frac{\partial \Delta z w^*}{\partial z^*}\right)_{x,y,t}\right] = 0 \tag{4.23}$$

For each degree of freedom i we should thus solve:

$$\int_{\Omega^*}\left[\frac{\partial \Delta z}{\partial t} + \left(\frac{\partial \Delta z u}{\partial x}\right)_{y,z^*,t} + \left(\frac{\partial \Delta z v}{\partial y}\right)_{x,z^*,t} + \left(\frac{\partial \Delta z w^*}{\partial z^*}\right)_{x,y,t}\right]\Psi_i^*\, d\Omega^* = 0 \tag{4.24}$$

We shall go through the process of vertical integration of this equation, and then of injection of the velocity coming from the momentum equation, to derive the space-time discretised equation on $h^{n+1}-h^n$. A variational formulation of the 3D continuity equation in the transformed mesh (equation 1.256) can be written in the form:

$$\begin{aligned}\frac{1}{\delta t}\int_{\Omega^*}\left(\Delta z^{n+1}-\Delta z^n\right)\Psi_i^*\,d\Omega^* \quad &= FLUINT(i)+FLUVER(i)\\ &\quad -FLUEXT(i)+SOURCE(i)\end{aligned} \tag{4.25}$$

with the following notations:

$$FLUINT(i)=\int_{\Omega^*}\Delta z\boldsymbol{u}\cdot\boldsymbol{\nabla}_{2D}\Psi_i^*\,d\Omega^* \tag{4.26}$$

$$FLUVER(i)=\int_{\Omega^*}\Delta z w^*\frac{\partial\Psi_i^*}{\partial z^*}\,d\Omega^* \tag{4.27}$$

$$SOURCE(i)=\frac{\int_\Omega Q_{sce}\Psi_{isce}\Psi_i\,d\Omega}{\int_\Omega\Psi_{isce}\,d\Omega}\alpha(i) \tag{4.28}$$

$$FLUEXT(i)=\int_{\Gamma^*_{liq}}\Delta z\boldsymbol{u}\cdot\boldsymbol{n}\Psi_i^*\,d\Gamma^* \tag{4.29}$$

For all the degrees of freedom i. $\boldsymbol{u}\cdot\boldsymbol{\nabla}_{2D}$ means that only the first two components of the 3D velocity are taken into account. The velocity $\boldsymbol{u}$ is here:

$$\boldsymbol{u}=\theta_u\tilde{\boldsymbol{u}}^{n+1}+(1-\theta_u)\boldsymbol{u}^n \tag{4.30}$$

Let us now sum these integrals over the degrees of freedom located above each i of the 2D domain. It is actually easier to write the first term in the real domain Ω to simplify it. By definition of Δz, we have:

$$\int_\Omega\Psi_i\,d\Omega=\int_{\Omega^*}\Psi_i^*\frac{\partial z}{\partial z*}\,d\Omega^*=\int_{\Omega^*}\Psi_i^*\Delta z\,d\Omega^* \tag{4.31}$$

Let us sum all the values of $\int_\Omega\Psi_i d\Omega$ along the degrees of freedom j on the vertical of i:

$$\begin{aligned}\sum_{j\ above\ i}\left(\int_\Omega\Psi_j\,d\Omega\right) &= \sum_{j\ above\ i}\left(\int_{\Omega_{2D}}\left(\int_{z=b}^{\eta}\Psi_j^V\,dz\right)\Psi_i^H\,d\Omega_{2D}\right)\\ &= \int_{\Omega_{2D}}\left(\int_{z=b}^{\eta}\sum_{j\ above\ i}\Psi_j^V\,dz\right)\Psi_i^H\,d\Omega_{2D}\\ &= \int_{\Omega_{2D}}h\Psi_i^H\,d\Omega_{2D}\end{aligned} \tag{4.32}$$

The integral and sum symbols commute because the basis functions are always positive. The first term of the equation (4.25) thus becomes:

$$\frac{1}{\delta t}\int_{\Omega^*}\left(\Delta z^{n+1}-\Delta z^n\right)\Psi_i^*\,d\Omega^* \quad =\frac{1}{\delta t}\int_{\Omega_{2D}}(h^{n+1}-h^n)\Psi_i^H\,d\Omega_{2D} \tag{4.33}$$

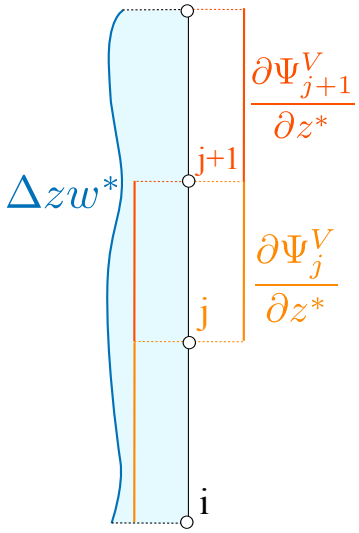

Figure 4.5: Sketch of the cancellation of $\sum_{j\ above\ i} \int_{\Omega^*} \Delta z w^* (\partial \Psi^{v*}_j / \partial z^*) \Psi^h_i \, d\Omega^*$: the integral above a point j cancels out with the one below $j+1$. The sum on the whole vertical is thus equal to zero.

On the other hand, the term $FLUVER(i)$ sums to zero on a vertical because of the properties of the basis function:

$$\sum_{j\ above\ i} \int_{\Omega^*} \Delta z w^* \frac{\partial \Psi^*_j}{\partial z^*} \, d\Omega^* \quad = \sum_{j\ above\ i} \int_{\Omega^*} \Delta z w^* \frac{\partial \Psi^{v*}_j}{\partial z^*} \Psi^h_i \, d\Omega^* = 0 \tag{4.34}$$

Indeed, the integrals cancel each other when summed along the vertical, as shown in the figure 4.5. The integration on the vertical of the equation (4.25) thus reads:

$$\int_{\Omega_{2D}} \Psi^h_i \left(\frac{h^{n+1} - h^n}{\delta t} \right) d\Omega_{2D} + \sum_{j\ above\ i} (-FLUINT(j) + FLUEXT(j) - SOURCE(j)) = 0 \tag{4.35}$$

or, more explicitely:

$$\left[\begin{array}{l} \displaystyle\int_{\Omega_{2D}} \Psi^h_i \left(\frac{h^{n+1} - h^n}{\delta t} \right) d\Omega_{2D} + \sum_{j\ above\ i} FLUEXT(j) \\ \displaystyle - \sum_{j\ above\ i} \int_{\Omega^*} \Delta z \left(\theta_u \tilde{\boldsymbol{u}}^{n+1} + (1-\theta_u) \boldsymbol{u}^n \right) \cdot \boldsymbol{\nabla}^* \Psi^*_j \, d\Omega^* \end{array} \right] = \sum_{j\ above\ i} SOURCE(j) \tag{4.36}$$

This corresponds to a variational formulation of the first line of the equation (3.6).

Remark:
It is important to write a variational formulation and only then sum the integrals over the vertical, otherwise the discrete conservation of mass is not achieved.

4.4.2 Estimation of the 2D conservative velocity

In what follows, we want to obtain a formula to calculate $\tilde{\boldsymbol{u}}^{n+1}$ that not involve having to solve a linear system: we need to then introduce that formula into the equation (4.36) to calculate h^{n+1}.

For this, we discretize the second line of the system (3.7), recalled below:

$$\frac{\boldsymbol{u}_{2D}^{aux}-\boldsymbol{u}_{2D}^{A}}{\delta t}=-g\boldsymbol{\nabla}_{2D}\eta^{n}+\tilde{\boldsymbol{F}}_{2D}+\nu_{E}\nabla^{2}\boldsymbol{u}_{2D} \tag{4.37}$$

Here the time discretisation of the diffusion term has not been specified: it is going to be specified below. Recall that we then have:

$$\tilde{\boldsymbol{u}}_{2D}^{n+1}=\boldsymbol{u}_{2D}^{aux}-g\theta_{h}\delta t\boldsymbol{\nabla}_{2D}(h^{n+1}-h^{n}) \tag{4.38}$$

We leave the framework of Finite Elements in this section, since we do not write a variational formulation of the equation but rather use or compute nodal values of each term so as to obtain $\boldsymbol{u}_{2D}^{aux}$ and then $\tilde{\boldsymbol{u}}_{2D}^{n+1}$. The gradient term $-g\theta_{h}\boldsymbol{\nabla}_{2D}(h^{n+1}-h^{n})$, written also $-g\theta_{h}\boldsymbol{\nabla}_{2D}\delta h$, shall be kept in this continuous form when being introduced in the depth-integrated continuity equation. When necessary, the nodal values of the fields or their gradients are estimated with the following formula:

$$[f]_{i}\approx\frac{\int_{\Omega}f\Psi_{i}\,d\Omega}{\int_{\Omega}\Psi_{i}\,d\Omega} \tag{4.39}$$

or for a 2D field:

$$[f]_{i}\approx\frac{\int_{\Omega_{2D}}f\Psi_{i}\,d\Omega_{2D}}{\int_{\Omega_{2D}}\Psi_{i}\,d\Omega_{2D}} \tag{4.40}$$

Note that this is only an approximation of the nodal value of f. Using this technique for the diffusion term yields:

$$\frac{\int_{\Omega}\nabla\cdot(\nu_{E}\boldsymbol{\nabla}\boldsymbol{u})\Psi_{i}\,d\Omega}{\int_{\Omega}\Psi_{i}\,d\Omega} \tag{4.41}$$

a form in which the discretisation in time is still not specified since we wish to retain the friction terms in an implicit form and the volumic term in an explicit, semi-implicit or implicit form. An integration by parts of the numerator yields:

$$\int_{\Omega}\nabla\cdot(\nu_{E}\boldsymbol{\nabla}\boldsymbol{u}_{2D})\Psi_{i}\,d\Omega=\int_{\Gamma}\Psi_{i}\nu_{E}\boldsymbol{\nabla}\boldsymbol{u}_{2D}\cdot\boldsymbol{n}\,d\Gamma-\int_{\Omega}\nu_{E}\boldsymbol{\nabla}\boldsymbol{u}_{2D}\cdot\boldsymbol{\nabla}\Psi_{i}\,d\Omega \tag{4.42}$$

Let us consider that the volumic term is treated explicitly for the sake of simplicity. The boundary term is treated implicitely. The above equation then becomes:

$$\int_{\Omega}\nabla\cdot(\nu_{E}\boldsymbol{\nabla}\boldsymbol{u}_{2D})\Psi_{i}\,d\Omega\simeq\int_{\Gamma}\Psi_{i}\nu_{E}\boldsymbol{\nabla}\boldsymbol{u}_{2D}^{aux}\cdot\boldsymbol{n}\,d\Gamma-\int_{\Omega}\nu_{E}\boldsymbol{\nabla}\boldsymbol{u}_{2D}^{n}\cdot\boldsymbol{\nabla}\Psi_{i}\,d\Omega \tag{4.43}$$

Now to deal with $\nu_{E}\boldsymbol{\nabla}\boldsymbol{u}_{2D}^{aux}\cdot\boldsymbol{n}$ we write that:

$$\nu_{E}\boldsymbol{\nabla}u^{aux}\cdot\boldsymbol{n}=aubor\;u^{aux}+bubor \tag{4.44}$$

and:

$$\nu_{E}\boldsymbol{\nabla}v^{aux}\cdot\boldsymbol{n}=avbor\;v^{aux}+bvbor \tag{4.45}$$

$avbor=aubor$ represents the stress due to friction and *bubor* and *bvbor* an explicit stress due for example to wind at the free surface or any other stress at the bed or the lateral boundaries. The term $\int_{\Gamma}\Psi_{i}\nu_{E}\boldsymbol{\nabla}\boldsymbol{u}_{2D}^{aux}\cdot\boldsymbol{n}\,d\Gamma$ finally takes the form of a diagonal matrix multiplied by the

velocity vector plus an explicit stress denoted by ***bubor***, with components *bubor* and *bvbor*. We now estimate the nodal value of *aubor* $\boldsymbol{u}_{2D}^{aux}$ with the formula (4.39) and define $fric3d$ as:

$$\frac{\int_\Gamma \Psi_i \; aubor \; \boldsymbol{u}_{2D}^{aux} \; d\Gamma}{\int_\Omega \Psi_i \; d\Omega} = -fric3d \; \boldsymbol{u}_{2D}^{aux} \tag{4.46}$$

We then use mass-lumping on $fric3d$ so as to have:

$$fric3d \simeq -\frac{\int_\Gamma \Psi_i \; d\Gamma}{\int_\Omega \Psi_i \; d\Omega} aubor = -\frac{aubor}{cos(\alpha)} \; \frac{\int_{\Omega_{2D}} \Psi_i^h \; d\Omega_{2D}}{\int_\Omega \Psi_i \; d\Omega} \tag{4.47}$$

Indeed, $1/cos(\alpha)$ is equal to $\sqrt{1+(\partial b/\partial x)^2+(\partial b/\partial y)^2}$ and as a matter of fact, we have:

$$d\Gamma = \sqrt{1+\left(\frac{\partial b}{\partial x}\right)^2+\left(\frac{\partial b}{\partial y}\right)^2} d\Omega_{2D} \tag{4.48}$$

on the bed, because the 2-dimensional domain is flat, whereas there is a slope of the 2-dimensional boundary of the 3-dimensional domain. On the other hand, we estimate the nodal value of $\nu_E \boldsymbol{\nabla} \boldsymbol{u}_{2D}^n$ with the formula (4.39) and define ***diff*** as:

$$\boldsymbol{diff} = \frac{\int_\Omega \nu_E \boldsymbol{\nabla} \boldsymbol{u}_{2D}^n \cdot \boldsymbol{\nabla} \Psi_i \; d\Omega}{\int_\Omega \Psi_i \; d\Omega} - \boldsymbol{bubor} \frac{\int_\Gamma \Psi_i \; d\Gamma}{\int_\Omega \Psi_i \; d\Omega} \tag{4.49}$$

The nodal values of ***bubor*** were also approximated and a mass-lumping was used to extract it from the integral, like for ***aubor***. Finally, we estimate the diffusion term as:

$$\frac{\int_\Omega \nabla \cdot (\nu_E \boldsymbol{\nabla} \boldsymbol{u}_{2D}) \Psi_i \; d\Omega}{\int_\Omega \Psi_i \; d\Omega} \approx -fric3d \; \boldsymbol{u}_{2D}^{aux} - \boldsymbol{diff} \tag{4.50}$$

The expression of $\tilde{\boldsymbol{u}}_{2D}^{n+1}$ to be injected in the equation (4.36) then reads:

$$\tilde{\boldsymbol{u}}_{2D}^{n+1} = D^{-1} \left(\boldsymbol{u}_{2D}^A + \delta t \left[-g\theta_h \boldsymbol{\nabla}_{2D} \delta h^{n+1} + \frac{\int_{\Omega_{2D}} [-g\boldsymbol{\nabla}_{2D}\eta^n] \Psi_i^h \; d\Omega_{2D}}{\int_{\Omega_{2D}} \Psi_i^h \; d\Omega_{2D}} + \boldsymbol{P}_{2D} + \boldsymbol{F}_{2D} - \boldsymbol{diff} \right] \right) \tag{4.51}$$

with:

$$D = 1 + \delta t (fric3d + s1u) \tag{4.52}$$

with the explicit treatment of the volumic integral for the diffusion term, we have:

$$D^{-1} = \frac{1}{1+\delta t(fric3d+s1u)} \tag{4.53}$$

but it is intentionally kept in the form D^{-1} because D will become a matrix when an implicit diffusion is taken into account (see the Finite Elements discretisation of the diffusion step described in the equation (4.13) for a better understanding of the implicit treatment of the diffusion). $\boldsymbol{P}_{2D}$ stands for $-\int_\Omega 1/\rho \boldsymbol{\nabla}_{2D} p_d \Psi_i \; d\Omega$.

In the equation (4.51), $\boldsymbol{\nabla}_{2D}\delta h^{n+1}$ is calculated as a nodal value, like $-g\,\boldsymbol{\nabla}_{2D}\eta^n$, that is:

$$\frac{\int_{\Omega_{2D}}[-g\theta_h\boldsymbol{\nabla}_{2D}\delta h^{n+1}-g\boldsymbol{\nabla}_{2D}\eta^n]\Psi_i^h\,d\Omega_{2D}}{\int_{\Omega_{2D}}\Psi_i^h\,d\Omega_{2D}} \tag{4.54}$$

This is provisional, we will see in the next subsection that in the continuity equation $\boldsymbol{\nabla}_{2D}\eta^n$ can be treated differently suppress wiggles.

4.4.3 Variational formulation of the equation on h^{n+1} - pseudo wave equation

We have defined:

$$\boldsymbol{u}^{aux}=D^{-1}\left(\boldsymbol{u}_{2D}^{A}+\delta t\left[-\frac{\int_{\Omega_{2D}}g\boldsymbol{\nabla}_{2D}\eta^n\Psi_i^h\,d\Omega_{2D}}{\int_{\Omega_{2D}}\Psi_i^h\,d\Omega_{2D}}+\boldsymbol{P}_{2D}+\boldsymbol{F}_{2D}-\boldsymbol{diff}\right]\right) \tag{4.55}$$

The depth-integrated continuity equation in variational formulation is then written as:

$$\begin{aligned}&\int_{\Omega_{2D}}\Psi_i^h\left(\frac{h^{n+1}-h^n}{\delta t}\right)d\Omega_{2D}+g\delta t\theta_h\theta_u\sum_{j\ above\ i}\left(\int_{\Omega^*}\Delta z\,D^{-1}\boldsymbol{\nabla}(h^{n+1}-h^n)\cdot\boldsymbol{\nabla}\Psi_j^*\,d\Omega^*\right)=\\&\sum_{j\ above\ i}\left(\int_{\Omega^*}\Delta z\,(\theta_u\boldsymbol{u}^{aux}+(1-\theta_u)\boldsymbol{u}^n)\cdot\boldsymbol{\nabla}\Psi_j^*\,d\Omega^*\right)+\sum_{j\ above\ i}[SOURCE(j)-FLUEXT(j)]\end{aligned} \tag{4.56}$$

where the variational formulation for the term containing δh in the conservative velocity is now consistent with the rest of the water-depth equation – a 3D integration followed by a summation on the vertical of each degree of freedom. The second term in the left-hand side is written as:

$$g\delta t\theta_h\theta_u\sum_{j\ above\ i}\left(\int_{\Omega^*}\Delta z\,D^{-1}\boldsymbol{\nabla}(h^{n+1}-h^n)\cdot\boldsymbol{\nabla}\Psi_j^*\,d\Omega^*\right)=\int_{\Omega_{2D}}\nu_{wave}\boldsymbol{\nabla}(h^{n+1}-h^n)\cdot\boldsymbol{\nabla}\Psi_i^h\,d\Omega^* \tag{4.57}$$

where ν_{wave} is a piece-wise constant function equal to:

$$\nu_{wave}(ielem)=g\delta t\ \theta_h\theta_u\sum_{prisms\ above\ ielem}\overline{(\Delta zD^{-1})} \tag{4.58}$$

where the bar stands for an average on the prism. *ielem* is a 2D element number.

Remark:

The choice of ν_{wave} as a piece-wise constant function stems from the fact that when summed over the vertical, the terms:

$$\sum_{j \ above \ i} \int_{\Omega^*} \Delta z g \delta t \theta_h \theta_u D^{-1} \nabla (h^{n+1} - h^n) \cdot \nabla \Psi_j^* \, d\Omega^* \tag{4.59}$$

should give:

$$\int_{\Omega_{2D}} \nu_{wave} \nabla (h^{n+1} - h^n) \cdot \nabla \Psi_i^h \, d\Omega_{2D} \tag{4.60}$$

In this way the treatment of this term is simpler and we recover a Saint-Venant like continuity equation. If we consider the integral:

$$\int_{\Omega^*} \Delta z \; g \delta t \theta_h \theta_u D^{-1} \nabla (h^{n+1} - h^n) \cdot \nabla \Psi_j^* \, d\Omega^* \tag{4.61}$$

on a prism, the only real 3-dimensional function is $\Delta z \; D^{-1}$, so that $\Delta z \; D^{-1}$ can be replaced by its average on the prism $\overline{\Delta z \; D^{-1}}$ without changing the result. Then by summing over the vertical we get to:

$$\int_{\Omega_{2D}} \nu_{wave} \nabla (h^{n+1} - h^n) \cdot \nabla \Psi_i^h \, d\Omega_{2D} \tag{4.62}$$

Finally, the variational formulation of the equation on h^{n+1} reads:

$$\int_{\Omega_{2D}} \left(\frac{h^{n+1} - h^n}{\delta t} \right) \Psi_i^h \, d\Omega_{2D} + \int_{\Omega_{2D}} \nu_{wave} \nabla (h^{n+1} - h^n) \cdot \nabla \Psi_i^h \, d\Omega_{2D} =$$

$$\sum_{j \ above \ i} \left(\int_{\Omega^*} \Delta z (\theta_u \boldsymbol{u}^{aux} + (1 - \theta_u) \boldsymbol{u}^n) \cdot \nabla \Psi_j^* \, d\Omega^* \right) + \sum_{j \ above \ i} (SOURCE(j) - FLUEXT(j)) \tag{4.63}$$

and $\boldsymbol{u}_{2D}^C$ is calculated through:

$$\boldsymbol{u}_{2D}^C = \theta_u \boldsymbol{u}_{2D}^{aux} + (1 - \theta_u) \boldsymbol{u}_{2D}^n \tag{4.64}$$

Remark:

In case the gradient of the dynamic pressure is included in $\boldsymbol{u}^{aux}$, a term $-(1/\rho)\nabla p_d$ is added to the momentum equation, where p_d is calculated by applying the Chorin projection method before calculating the new water depth. This corresponds to the key-word `DYNAMIC PRESSURE IN WAVE EQUATION` (see the section 3.2).

Some history:

In former versions of TELEMAC-3D this step was solved by calling TELEMAC-2D. It then appeared that this was a theoretical mistake: average on the vertical and discretisation do not commute. The right thing to do is to first discretise the equations in 3D, then to sum them over the vertical. This sum is thus discrete and done plane by plane. In the more recent versions the only technique proposed is the wave equation, which consists of eliminating the velocity from the depth-summed continuity equation, yielding an equation on the depth, which looks like a wave equation as the celerity of waves clearly appears in it.

Equivalence between the pseudo-wave equation and a Saint-Venant like continuity equation

The continuity equation, given bed and free-surface boundary conditions, yields a Saint-Venant like continuity equation. The classical version of the Saint-Venant continuity equation is recalled below, with a variational formulation and a time discretisation:

$$\int_{\Omega_{2D}} \left(\frac{h^{n+1} - h^n}{\delta t} \right) \Psi_i^h \, d\Omega_{2D} + \int_{\Gamma_{2D}} h\boldsymbol{U} \cdot \boldsymbol{n} \Psi_i^h \, d\Gamma_{2D} - \int_{\Omega_{2D}} h\boldsymbol{U} \cdot \boldsymbol{\nabla} \Psi_i^h \, d\Omega_{2D} = 0 \tag{4.65}$$

The continuity equation solved to obtain the values of water height, (4.63), can also be considered as:

$$\begin{aligned} &\int_{\Omega_{2D}} \left(\frac{h^{n+1} - h^n}{\delta t} \right) \Psi_i^h \, d\Omega_{2D} - \sum_{j \; above \; i} \left(\int_{\Omega^*} \Delta z \boldsymbol{u}^{comp} \cdot \boldsymbol{\nabla} \Psi_j^* \, d\Omega^* \right) \\ &- \sum_{j \; above \; i} (SOURCE(j) - FLUEXT(j)) = 0 \end{aligned} \tag{4.66}$$

with $\boldsymbol{u}^{comp}$ being here:

$$\boldsymbol{u}^{comp} = \theta_u \boldsymbol{u}^{aux} + (1 - \theta_u) \boldsymbol{u}^n - g \delta t \theta_h \theta_u D^{-1} \boldsymbol{\nabla} (h^{n+1} - h^n) \tag{4.67}$$

We have:

$$\sum_{j \; above \; i} \left(\int_{\Omega^*} \Delta z \boldsymbol{u}^{comp} \cdot \boldsymbol{\nabla} \Psi_j^* \, d\Omega^* \right) = \int_{\Omega_{2D}} \sum_{j \; above \; i} \Delta z \boldsymbol{u}^{comp} \cdot \boldsymbol{\nabla} \Psi_j^H \, d\Omega_{2D} \tag{4.68}$$

For the same reason as $FLUEXT = 0$, so that the equation (4.63) is a form of depth-integrated continuity equation, as expected.

Mass-lumping of the pseudo-wave equation

Mass-lumping may be used to solve the pseudo-wave equation. It is always the case when distributive schemes are used for the advection, otherwise it is a choice left to the user. In case mass-lumping is used, the left-hand side of the equation (4.63) is approximated in the form:

$$\frac{1}{\delta t} \int_{\Omega_{2D}} (h^{n+1} - h^n) \Psi_i^h \, d\Omega_{2D} \simeq \frac{(h_i^{n+1} - h_i^n)}{\delta t} \int_{\Omega_{2D}} \Psi_i^h \, d\Omega_{2D} \tag{4.69}$$

then the terms $\int_{\Omega^{n+1}} \Psi_i \, d\Omega^{n+1}$ and $\int_{\Omega^n} \Psi_i \, d\Omega^n$ must be modified accordingly everywhere they are used in the equations. Indeed, for a case like the one shown in the figure 4.6, the integral of the basis around i is not equal to zero while h_i is equal to 0, so that the mass-lumped terms

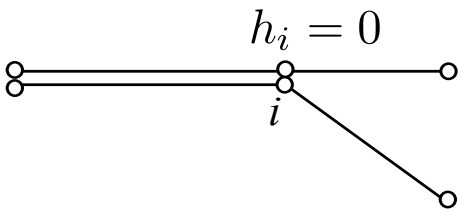

Figure 4.6: Sketch of a configuration where mass-lumping would lead to an incompatibility between the pseudo wave equation and the calculation of $\Delta z w^{C*}$ without a specific treatment of the terms $\int_\Omega \Psi_i \, d\Omega$ in the equations.

involving h_i will cancel out but not the ones involving other quantities. This leads to spurious mass transfers and must be avoided. As a matter of fact, we must write:

$$\int_\Omega \Psi_i \, d\Omega = \int_{\Omega^*} h \Psi_i^* \, d\Omega^* \simeq h_i \int_{\Omega^*} \Psi_i^* \, d\Omega^* \tag{4.70}$$

In this way, the calculation the diffusion stage (for the vertical velocity and scalars) and the velocity correction stage remain compatible with the mass-lumped pseudo wave equation.

Remark:
This corresponds to the calculation of `VOLU` with the formula `MASBAS 2` in the subroutine `vc0ppp.f`.

4.4.4 Suppressing wiggles: compatibility of the free surface gradient

In the equation (4.51) used to calculate $\boldsymbol{u}^{aux}$, the explicit free surface gradient is a linear function taken as:

$$\boldsymbol{\nabla} \eta^n_{linear} \simeq \frac{\int_\Omega \boldsymbol{\nabla} \eta^n \Psi_i \, d\Omega}{\int_\Omega \Psi_i \, d\Omega} \tag{4.71}$$

for every degree of freedom i with test function Ψ_i. This is the "compatible" way of computing the gradient. Oscillations of η are not "seen" by this way of discretising the gradient and do not trigger a velocity that would cause artificial maxima of free-surface elevation to smooth out. To damp such spurious oscillations, $\boldsymbol{\nabla} \eta^n$ should be considered as a piece-wise constant function, in a way consistant with the fact that η^n is a linear function. This is the "non compatible way" of computing the gradient.
On the other hand, the terms involving $\boldsymbol{\nabla}(h^{n+1} - h^n)$ in (4.63) are piece-wise constant functions, since they are the gradients of a piece-wise linear function. This is the “non-compatible” way of discretising the free-surface gradient. The “compatible” way of discretising $\boldsymbol{\nabla} \eta^n$ makes it possible to obtain a velocity field such as $\partial h / \partial t + \nabla \cdot (h \boldsymbol{u}) = 0$ is fulfilled. However, with this discretisation, oscillations of η are invisible because they do not trigger a velocity that would cause artificial maxima of free-surface elevation to smooth out. The “non-compatible” way of discretising η helps smoothing out spurious oscillations but the velocity field does not exactly obey the continuity equation anymore.

To sum up, a totally compatible $\boldsymbol{\nabla} \eta$ ensures a better compatibility of the continuity equation with the velocity field, while a totally non compatible $\boldsymbol{\nabla} \eta$ makes it possible to suppress spurious oscillations of the free-surface elevation. It happens that the non compatible treatment of

$\boldsymbol{\nabla}(h^{n+1}-h^n)$ is not always sufficient to suppress spurious oscillations, hence the idea of treating also the term $\boldsymbol{\nabla}\eta^n$ in a non-compatible way. This can be achieved by suppressing $\boldsymbol{\nabla}\eta^n$ in the computation of $\boldsymbol{u}^{aux}$ (equation (4.55)), and adding the following term to the right hand side of the continuity equation:

$$-\int_{\Omega_{2D}} \frac{\nu_{wave}}{\theta_h} [\boldsymbol{\nabla}\eta^n]_{piece-wise\ const.} \cdot \boldsymbol{\nabla}\Psi_i^h \, d\Omega_{2D} \tag{4.72}$$

Suppressing $\boldsymbol{\nabla}\eta^n$ in the computation of $\boldsymbol{u}^{aux}$ is actually equivalent to adding:

$$-\int_{\Omega} \frac{\nu_{wave}}{\theta_h} [\boldsymbol{\nabla}\eta^n]_{linear} \cdot \boldsymbol{\nabla}\Psi_i^h \, d\Omega \tag{4.73}$$

to the right-hand side of the continuity equation. The two terms do not sum to zero because $\boldsymbol{\nabla}\eta^n$ is not discretised in the same way. With a parameter θ^{comp} varying from 0 to 1 we can skip from compatible to non compatible $\boldsymbol{\nabla}\eta^n$. When recovering the final velocity with the formula (4.64), one must not forget that $\boldsymbol{u}^{aux}$ lacks the non compatible part of the free surface gradient, hence this lacking part must be added, and it can only be done in a compatible way to get a linear contribution.

Remark:

When $\boldsymbol{\nabla}\eta^n$ is referred to in the formulae, care must be taken that in the case of tidal flats with treatment by option 1 (correction of the free surface), the function η^n is piece-wise linear and must be stored in a specific array. BIEF subroutines must be written in order to deal with such a discretisation.

4.5 Space discretisation of the pressure step

We recall that in this step the following equations are solved:

$$\begin{cases} \nabla^2 p_d^{n+1} = \dfrac{\rho}{\delta t} \nabla \cdot \tilde{\boldsymbol{u}}^{n+1} \\ \dfrac{\boldsymbol{u}^{n+1} - \tilde{\boldsymbol{u}}^{n+1}}{\delta t} = -\dfrac{1}{\rho} \boldsymbol{\nabla} p_d^{n+1} \end{cases} \tag{4.74}$$

where $\tilde{\boldsymbol{u}}^{n+1}$ denotes the non divergence-free vector of components (u^C, v^C, w^D) (see the chapter 3). The variational formulation is written as:

$$\int_{\Omega} \nabla \cdot \left(\frac{\delta t}{\rho} \boldsymbol{\nabla} p_d \right) \Psi_i \, d\Omega = \int_{\Omega} \nabla \cdot \tilde{\boldsymbol{u}}^{n+1} \Psi_i \, d\Omega \tag{4.75}$$

The left-hand side of this equation is integrated by parts:

$$\int_{\Gamma} \Psi_i \frac{\delta t}{\rho} \boldsymbol{\nabla} p_d \cdot \boldsymbol{n} \, d\Gamma - \int_{\Omega} \frac{\delta t}{\rho} \boldsymbol{\nabla} p_d \cdot \boldsymbol{\nabla}\Psi_i \, d\Omega = \int_{\Omega} \nabla \cdot \tilde{\boldsymbol{u}}^{n+1} \Psi_i \, d\Omega \tag{4.76}$$

which is also:

$$\int_{\Omega} \frac{\delta t}{\rho} \boldsymbol{\nabla} p_d \cdot \boldsymbol{\nabla}\Psi_i \, d\Omega = \int_{\Gamma} \Psi_i \frac{\delta t}{\rho} \boldsymbol{\nabla} p_d \cdot \boldsymbol{n} \, d\Gamma - \int_{\Omega} \nabla \cdot \tilde{\boldsymbol{u}}^{n+1} \Psi_i \, d\Omega \tag{4.77}$$

and by admitting the condition $\partial p_d / \partial n = 0$ as a wall boundary condition for p_d, we arrive at the system:

$$\int_{\Omega} \frac{\delta t}{\rho} \boldsymbol{\nabla} p_d^{n+1} \cdot \boldsymbol{\nabla}\Psi_i \, d\Omega = -\int_{\Omega} \nabla \cdot \tilde{\boldsymbol{u}}^{n+1} \Psi_i \, d\Omega \tag{4.78}$$

This is the Poisson equation. The matrix of this system is a diffusion-type matrix built with 1 in the place of viscosity, the pressure found being in fact $(\delta t/\rho)p_d$. The boundary condition on the free surface is $p_d = 0$. Once p_d^{n+1} is known, $\boldsymbol{u}^{n+1}$ is retrieved with the second line of (4.74), which is also subjected to a variational formulation:

$$\int_\Omega \boldsymbol{u}^{n+1}\Psi_i \, d\Omega - \int_\Omega \widetilde{\boldsymbol{u}}^{n+1}\Psi_i \, d\Omega = -\frac{\delta t}{\rho}\int_\Omega \boldsymbol{\nabla} p_d^{n+1}\Psi_i \, d\Omega \tag{4.79}$$

which is discretised into:

$$\int_\Omega \boldsymbol{u}_j^{n+1}\Psi_j\Psi_i \, d\Omega - \int_\Omega \tilde{\boldsymbol{u}}_j^{n+1}\Psi_j\Psi_i \, d\Omega = -\frac{\delta t}{\rho}\int_\Omega p_{dj}^{n+1}\boldsymbol{\nabla}\Psi_j\Psi_i \, d\Omega \tag{4.80}$$

for all degrees of freedom i, j. In this equation, the pressure gradient was discretised through:

$$\boldsymbol{\nabla} p_d^{n+1} = \sum_j p_{dj}^{n+1}\boldsymbol{\nabla}\Psi_j \tag{4.81}$$

by deriving the discretised pressure, $p(x,y,z) = \sum_j p_j\Psi_j(x,y,z)$. The two terms of the left-hand side are mass-lumped in TELEMAC-3D (there is no option available yet to solve the full linear system), so that the velocity $\boldsymbol{u}^{n+1}$ is calculated through:

$$\boldsymbol{u}_j^{n+1} = \tilde{\boldsymbol{u}}_j^{n+1} - \frac{\delta t}{\rho}\frac{\int_\Omega p_{dj}^{n+1}\boldsymbol{\nabla}\Psi_j\Psi_i \, d\Omega}{\int_\Omega \Psi_j \, d\Omega} \tag{4.82}$$

for all degrees of freedom i, j. The term $\int_\Omega \Psi_j \, d\Omega$ in the denominator of the right-hand side is calculated in agreement with the choice of mass-lumping on the water-height term in the wave equation: if mass-lumping was activated, it is calculated through the equation (4.70).

4.6 Computing the conservative vertical velocity

The conservative vertical velocity w^C is calculated on the basis of the mass conservation equation. As the advection is executed in the fixed transformed mesh, it is actually w^{C*} that we ought to know. Indeed, w^C is only used during the calculation of the advection terms.

> **Remark:**
> All the demonstrations of conservation of mass for the distributive schemes were done considering that both the advection and the mass conservation equation (both in the form of the wave equation and in its classical form to retrieve w^C), are solved in the transformed mesh. They could be solved in the fixed mesh but this would increase the complexity of the algorithm.

In fact, except in the case of advection by the method of characteristics, where w^{C*} is really used, it is the $\Delta z w^{C*}$ grouping which appears often and which is homogeneous to a velocity (we recall that Δz is the depth between two planes, see the section 4.2). The formula for passing from w^* to w, i.e. $w = \Delta z w^{C*}$ + other terms (see the equation 4.103 below, we treat here the generalized sigma transform case), shows that $\Delta z w^{C*}$ should have the same discretization as w^C. It appears that in the fixed mesh, the real unknown should be $\Delta z w^{C*}$ and not w^{C*}. We shall now examine a method for calculating $\Delta z w^{C*}$, then how w^C is found on the basis of $\Delta z w^{C*}$. We first have to establish the form of the continuity equation in the transformed mesh.

4.6.1 Discretisation of the continuity equation

We recall here the continuity equation $div(\boldsymbol{u}^C) = 0$ written in the transformed mesh (equation 1.256):

$$\frac{1}{\Delta z}\left[\frac{\partial \Delta z}{\partial t} + \left(\frac{\partial\left(\Delta z u^C\right)}{\partial x}\right)_{y,z^*,t} + \left(\frac{\partial\left(\Delta z v^C\right)}{\partial y}\right)_{x,z^*,t} + \left(\frac{\partial \Delta z w^{C*}}{\partial z^*}\right)_{x,y,t}\right] = 0 \tag{4.83}$$

For each degree of freedom i we should thus solve:

$$\int_{\Omega^*}\left[\frac{\partial \Delta z}{\partial t} + \left(\frac{\partial \Delta z u^C}{\partial x}\right)_{y,z^*,t} + \left(\frac{\partial \Delta z v^C}{\partial y}\right)_{x,z^*,t} + \left(\frac{\partial \Delta z w^{C*}}{\partial z^*}\right)_{x,y,t}\right]\Psi_i^*\, d\Omega^* = 0 \tag{4.84}$$

With a simple sigma transformation, we would in fact get:

$$\int_{\Omega^*}\left[\left(\frac{\partial h}{\partial t}\right)_{x,y} + \left(\frac{\partial h u^C}{\partial x}\right)_{y,z^*,t} + \left(\frac{\partial h v^C}{\partial y}\right)_{x,z^*,t} + \left(\frac{\partial h w^{C*}}{\partial z^*}\right)_{x,y,t}\right]\Psi_i^*\, d\Omega^* = 0 \tag{4.85}$$

where we see the underlying depth-integrated continuity equation. It is actually this form of the mass conservation equation that is found while demonstrating the mass conservation of the scalar (see the section 4.8.2). Discretized as such, this equation whose unknown is w^{C*} or rather $\Delta z w^{C*}$ leads to a system that is ill-conditioned. The boundary conditions are $w^{C*} = 0$ at the bed and the free surface, if they are impermeable. The term:

$$\int_{\Omega^*}\left(\frac{\partial \Delta z w^{C*}}{\partial z^*}\right)_{x,y,t}\Psi_i^*\, d\Omega^* \tag{4.86}$$

with the unknown $\Delta z w^{C*}$ leads to a matrix A with general term:

$$A_{ij} = \int_{\Omega^*}\frac{\partial \Psi_j^*}{\partial z^*}\Psi_i^*\, d\Omega^* \tag{4.87}$$

Outside the boundaries, the diagonal terms of this matrix are zero and it is not invertible. From the boundary conditions at the bed and the free surface, we arrive at disconnected solutions between two successive planes (parasitic oscillations), or an impossibility. To circumvent this problem, a solution consists of changing unknowns or, to be more precise, of giving a specific definition of the unknown $\Delta z w^{C*}$. This definition, which will be given in the next paragraph, has been inspired by the necessity to get an exact mass conservation of scalars with distributive schemes. As a matter of fact, it will in due course appear that this definition of $\Delta z w^{C*}$ is entirely compatible with these distributive schemes.

4.6.2 Calculation of $\Delta z w^{C*}$:

For each point i of the mesh, the calculation of $\Delta z w^{C*}$ is done by solving:

$$\int_{\Omega^*}\left(\Delta z^{n+1} - \Delta z^n\right)\Psi_i^*\, d\Omega^* = \delta t\int_{\Omega^*}\Delta z\boldsymbol{u}^C\cdot\boldsymbol{\nabla}\Psi_i^*\, d\Omega^* - \delta t\int_{\Gamma^*}\Delta z\boldsymbol{u}^C\cdot\boldsymbol{n}\Psi_i^*\, d\Gamma^* \tag{4.88}$$

In the right-hand side, no time discretization of Δz is specified yet. Actually, it must be consistent with the 2D continuity equation, as will be explained later, which generally imposes that Δz is taken at time t^n. We shall now show that the problem is well-posed if we consider that the unknowns are the average of $\Delta z w^{C*}$ along the vertical of each prism. This corresponds to a definition of the vertical velocity on a staggered mesh, as represented in the figure 4.7. Let us

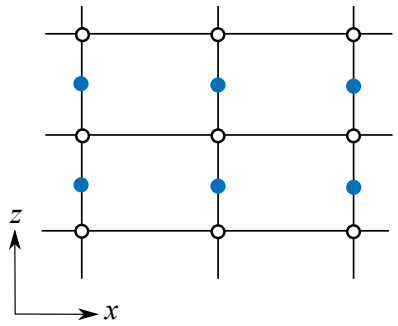

Figure 4.7: Definition of a vertical velocity on a staggered mesh: the blue dots represent the degrees of freedom for the vertical velocity and the white dots represent the degrees of freedom for the horizontal velocity.

separate the horizontal and vertical gradients in the continuity equation:

$$\begin{aligned}\int_{\Omega^*} \Delta z w^{C*} \frac{\partial \Psi_i^*}{\partial z^*}\, d\Omega^* &= \frac{1}{\delta t}\int_{\Omega^*} \left(\Delta z^{n+1} - \Delta z^n\right) \Psi_i^*\, d\Omega^* \\ &- \int_{\Omega^*} \Delta z \boldsymbol{u}^C \cdot \boldsymbol{\nabla}_{2D} \Psi_i^*\, d\Omega^* + \int_{\Gamma^*} \Delta z \boldsymbol{u}^C \cdot \boldsymbol{n} \Psi_i^*\, d\Gamma^*\end{aligned} \tag{4.89}$$

In this form, the right-hand side only contains known terms (assuming that the flux boundary conditions are given at the boundaries). So, we concentrate on the left-hand side which contains unknowns. For this, it should be considered that we work with prisms in the transformed fixed mesh, hence with horizontal base and top. In these conditions, prisms are homothetic to the reference prism and the 3D bases are products of one function of x and y and one function of z The function of x and y is the same as the basis functions of triangles with linear interpolation. This makes it possible to break down each basis as $\Psi_i^* = \Psi_i^h \Psi_i^v$ where Ψ_i^h only depends on x and y, while Ψ_i^v only depends on z. Thus:

$$\frac{\partial \Psi_i^*}{\partial z^*} = \Psi_i^h \frac{\partial \Psi_i^v}{\partial z^*} \tag{4.90}$$

For any elevation z^*, we discretize the function $\Delta z \boldsymbol{u}^{C*}$ on the 2D domain as follows:

$$\Delta z \boldsymbol{u}^{C*} = \sum_{j=1}^{npoin2} \left[\Delta z \boldsymbol{u}^{C*}(z^*)\right]_j \Psi_j^h(x,y) \tag{4.91}$$

where $npoin2$ is the number of degrees of freedom in the 2D mesh.

Remark:
This choice is very impacting and constitutes a decorrelation of the layers with regards to the vertical velocity.

The left-hand side of the equation 4.89 then becomes:

$$\sum_{j=1}^{npoin2} \left(\int_{\Omega_{2D}} \Psi_i^h \Psi_j^h\, d\Omega_{2D} \int_0^1 \left[\Delta z w^*(z^*)\right]_j \frac{\partial \Psi_i^v}{\partial z^*}\, dz^* \right) \tag{4.92}$$

The mass matrix of the 2D mesh appears in $\int_{\Omega_{2D}} \Psi_i^h \Psi_j^h\, d\Omega_{2D}$. According to the position of the point i, the value of $\partial \Psi_i^v / \partial z^*$ in a prism is equal to $\pm 1/\Delta z^*$, where Δz^* is the height of the prism (these bases are linear; their value is 1 at the top and 0 at the bed or vice versa).

For a basis located on the plane ip, a plane different from the bed and the free surface, we get:

$$\begin{aligned}\int_0^1 \left[\Delta z w^{C*}(z^*)\right]_j \frac{\partial \Psi_i^v}{\partial z^*}\, dz^* &= \frac{1}{z^*_{ip}-z^*_{ip-1}} \int_{z^*_{ip-1}}^{z^*_{ip}} \left[\Delta z w^{C*}(z^*)\right]_j \, dz^* \\ &- \frac{1}{z^*_{ip+1}-z^*_{ip}} \int_{z^*_{ip}}^{z^*_{ip+1}} \left[\Delta z w^{C*}(z^*)\right]_j \, dz^*\end{aligned} \tag{4.93}$$

The quantities retrieved here are the ones which help to calculate the coefficients of distributive schemes (see the section 4.10.2). Let us define:

$$\overline{(\Delta z w^{C*})}^j_{ip-\frac{1}{2}} = \frac{1}{(z^*_{ip}-z^*_{ip-1})} \int_{z^*_{ip-1}}^{z^*_{ip}} \left[\Delta z w^{C*}(z^*)\right]_j \, dz^* \tag{4.94}$$

we finally get:

$$\int_0^1 \left[\Delta z w^{C*}(z^*)\right]_j \frac{\partial \Psi_i^v}{\partial z^*}\, dz^* = \overline{(\Delta z w^{C*})}^j_{ip-1/2} - \overline{(\Delta z w^{C*})}^j_{ip+1/2} \tag{4.95}$$

For a point located on the bed, we have:

$$\int_0^1 \left[\Delta z w^{C*}(z^*)\right]_j \frac{\partial \Psi_i^v}{\partial z^*}\, dz^* = +\overline{(\Delta z w^{C*})}^j_{1+1/2} \tag{4.96}$$

and at the surface:

$$\int_0^1 \left[\Delta z w^{C*}(z^*)\right]_j \frac{\partial \Psi_i^v}{\partial z^*}\, dz^* = -\overline{(\Delta z w^{C*})}^j_{np-1/2} \tag{4.97}$$

We arrive at a series of linear systems (one per plane), which, after discretisation, read:

$$\begin{aligned}&\sum_{j=1}^{npoin2} \left(\int_{\Omega_{2D}} \Psi_i^h \Psi_j^h \, d\Omega_{2D} \left[\overline{(\Delta z w^{C*})}^j_{ip+1/2} - \overline{(\Delta z w^{C*})}^j_{ip-1/2} \right] \right) = \\ &-\frac{1}{\delta t} \int_{\Omega^*} \sum_{j=1}^{npoin3} \left(\Delta z_j^{n+1} - \Delta z_j^n \right) \Psi_j^* \Psi_i^* \, d\Omega^* + \int_{\Omega^*} \sum_{j=1}^{npoin2} \Delta z_j \Psi_j^* \sum_{k=1}^{npoin2} \Psi_k^* \boldsymbol{u}_k^C \cdot \boldsymbol{\nabla}_{2D} \Psi_i^* \, d\Omega^* \\ &- \int_{\Gamma^*_{liq}} \sum_{j=1}^{npoin2} \Delta z_j \Psi_j^* \sum_{k=1}^{npoin2} \boldsymbol{u}_k^C \Psi_k^* \cdot \boldsymbol{n} \Psi_i^* \, d\Gamma^*\end{aligned} \tag{4.98}$$

We deliberately kept the summation signs in the equations because they either run on the 2D points or on the 3D points. In this form, the new unknowns $\overline{(\Delta z w^{C*})}^j_{ip+1/2}$ can be found layer by layer starting from the bed, or in the same way from the surface. The system's matrix is always the same mass matrix and is easily invertible. On the other hand, there are still too many equations and it is likely that, if we start from the bed, the boundary condition at the free surface will not be retrieved. This does not happen since the depth-integrated continuity equation was resolved in a compatible way to calculate the free surface. In fact, the sum of all equations, layer after layer, eliminates all the unknowns $\overline{(\Delta z w^{C*})}^j_{ip+1/2}$ and restores the mass conservation expression of the pseudo-wave equation. If the latter has been correctly resolved, the linear system of the last plane is deduced from the others, and hence it cannot give rise to contradictions.

The equation 4.98 is actually simplified by mass lumping. In this case, the left-hand side reads:

$$\sum_{j=1}^{npoin2} \left(\int_{\Omega_{2D}} \Psi_i^h \Psi_j^h \, d\Omega_{2D} \right) \left[\overline{(\Delta z w^{C*})}^i_{ip+1/2} - \overline{(\Delta z w^{C*})}^i_{ip-1/2} \right] \tag{4.99}$$

and the system is solved by a mere division of the right-hand side by $\int_{\Omega_{2D}} \Psi_i^h \, d\Omega_{2D}$. The same mass lumping must also be done when computing the corresponding terms in the scalar equation (see the section 4.10.2 and the computation of the terms b_i) in order to ensure the mass conservation.

4.6.3 Nodal values of w^{C*}

One would now expect that we deduce w^{C*} from $\Delta z w^{C*}$ by dividing by Δz. This is not a good idea. As a matter of fact we need nodal values of w^{C*}, e.g. for the advection in the transformed mesh with the method of characteristics, but if we write w^C in the transformed mesh, we get:

$$w^C = \frac{dz}{dt} = \frac{\partial z}{\partial t} + u^C \frac{\partial z}{\partial x} + v^C \frac{\partial z}{\partial y} + w^{C*} \frac{\partial z}{\partial z^*} \tag{4.100}$$

and so we have:

$$\Delta z w^{C*} = -\frac{\partial z}{\partial t} - u^C \frac{\partial z}{\partial x} - v^C \frac{\partial z}{\partial y} + w^C \tag{4.101}$$

We can deduce from this latter equation that if w^C is a linear function, then $\Delta z w^{C*}$ is also a linear function. To get this linear function, we need to have nodal values of $\Delta z w^{C*}$, which can be obtained by the following formula using the layer-averaged values of the previous paragraph:

$$\overline{(\Delta z w^{C*})}^j_{ip} = \frac{\overline{(\Delta z w^{C*})}^j_{ip-1/2} + \overline{(\Delta z w^{C*})}^j_{ip+1/2}}{2} \tag{4.102}$$

At the boundaries, i.e. at the bed and at the free surface, we know that the impermeability condition imposes that $\Delta z w^{C*} = 0$. We now have a linear function $\Delta z w^{C*}$. When computing the characteristics in the transformed mesh, the vertical velocity in a layer will be this linear function $\Delta z w^{C*}$ divided by Δz which is locally a function of x and y.

4.6.4 Passing from $\Delta z w^{C*}$ to w^C – only for the hydrostatic option

As the advection is done in the transformed mesh, w^C is actually never used, but users generally expect to see it in the results, and so it must be built when the hydrostatic option is used because the momentum equation along z is not solved. This is done with the equation 4.101, which is discretized in time in the form:

$$w^{Cn+1} = (\Delta z w^{C*})^{n+1} + \frac{z^{n+1} - z^n}{\delta t} + u^C \frac{\partial z^n}{\partial x} + v^C \frac{\partial z^n}{\partial y} \tag{4.103}$$

This equation is integrated layer by layer over the vertical in the transformed mesh, and then multiplied by a test function and integrated on the 2D mesh. It gives:

$$\begin{aligned} \int_{\Omega_{2D}} \left(\int_{z^*_{ip}}^{z^*_{ip+1}} w^{Cn+1} \, dz^* \right) \Psi_i^h d\Omega_{2D} \quad &= \int_{\Omega_{2D}} \left(\int_{z^*_{ip}}^{z^*_{ip+1}} \left(\Delta z w^{C*}\right)^{n+1} \, dz^* \right) \Psi_i^h d\Omega_{2D} \\ &+ \int_{\Omega_{2D}} \left(\int_{z^*_{ip}}^{z^*_{ip+1}} \left(\frac{z^{n+1} - z^n}{\delta t} + u^C \frac{\partial z^n}{\partial x} + v^C \frac{\partial z^n}{\partial y} \right) \, dz^* \right) \Psi_i^h \, d\Omega_{2D} \end{aligned} \tag{4.104}$$

where ip is the number of the lower plane of the layer, and i is the global number of a node in the 2D mesh. As w is linear, we have:

$$\int_{z^*_{ip}}^{z^*_{ip+1}} w^{Cn+1} \, dz^* = (w^{Cn+1}_{ip} + w^{Cn+1}_{ip+1}) \frac{\Delta z^*_{ip+1/2}}{2} \tag{4.105}$$

By using similar formulae for $(z^{n+1}-z^n)/\delta t$, we get:

$$\begin{aligned}\int_{\Omega_{2D}}(w_{ip}^{Cn+1}+w_{ip+1}^{Cn+1})\Psi_i^h\,d\Omega_{2D} &= 2\int_{\Omega_{2D}}(\Delta z w^{C*})_{ip+1/2}^{n+1}\Psi_i^h\,d\Omega_{2D}\\ &+\frac{1}{\delta t}\int_{\Omega_{2D}}(z_{ip}^{n+1}+z_{ip+1}^{n+1}-z_{ip}^{n}-z_{ip+1}^{n})\Psi_i^h\,d\Omega_{2D}\\ &+\frac{2}{\Delta z_{ip+1/2}^*}\int_{\Omega_{2D}}\left(\int_{z_{ip}^*}^{z_{ip+1}^*}\left(u^C\frac{\partial z^n}{\partial x}+v^C\frac{\partial z^n}{\partial y}\right)\,dz^*\right)\Psi_i^h\,d\Omega_{2D}\end{aligned} \tag{4.106}$$

We recognize here a series of linear systems on the 2D mesh, the matrix being the mass matrix. For every layer in the mesh, we shall get $w_{ip}^{Cn+1}+w_{ip+1}^{Cn+1}$. If we start from $ip=1$, the first value w_1^{Cn+1} is given by the boundary condition on the bed, and then the boundary condition at the free surface should be naturally ensured. Another possibility is to start from the free surface, and going down to the bed. Lest the truncation errors or a lack of compatibility in the discretization should give a wrong boundary condition after going up or down, we build both solutions, creating a series of w_{ip}^{Cup} and w_{ip}^{Cdown}, where w_1^{Cup} is obtained by the boundary condition at the bed and w_{np}^{Cdown} is obtained from the boundary condition at the free surface. The final value of the vertical velocity is then obtained by the formula:

$$w_{ip}^{Cn+1}=\frac{(ip-1)w_{ip}^{Cdown}+(np-ip)w_{ip}^{Cup}}{np-1} \tag{4.107}$$

where np is still the number of superimposed meshes of triangles.

4.7 Space discretisation of the scalar equation

The scalar equation (1.259) in a moving mesh is recalled below:

$$\frac{\partial T}{\partial t}+((\boldsymbol{u}-\boldsymbol{c})\cdot\boldsymbol{\nabla})\,T-\nabla\cdot(K_E\boldsymbol{\nabla}(T))+Q=0 \tag{4.108}$$

The advection takes place in the fixed transformed mesh with the methods presented in the section 4.10. It is either coupled with the diffusion (the case of the SUPG scheme) or solved before the diffusion (the case of the characteristics and distributive schemes). The advection equation is written as:

$$\int_{\Omega^*}\Delta z(T^{n+1}-T^n)\Psi_i^*\,d\Omega^*+\delta t\int_{\Omega^*}\Delta z\boldsymbol{u}^*\cdot\boldsymbol{\nabla}T\Psi_i^*\,d\Omega^*=0 \tag{4.109}$$

We recall that the velocity of the mesh is also taken into account by this formula, as explained in the section 1.9. In 1995, Janin (see [42]) found a remarkable solution for building a monotonic and at the same time conservative scheme:

- the use of a distributive scheme called the MURD scheme (see the section 4.10.2);
- the choice of advection in a mesh fixed in time t^{n+1}, with T written in an explicit way in the term $\boldsymbol{\nabla}T$ (see the demonstration of conservation of the scalar in the section 4.8.2);
- a compatible resolution of the mass conservation equation, with a variable $\Delta z w^{C*}$ averaged per layer as seen in the section 4.6.2.

Only a good comprehension of these three points allows one to appreciate the interest and originality of this approach. Diffusion has fewer constraints and is calculated in the real mesh. In 2005 mass conservation of the scalar was extended to the SUPG advection scheme, using the same ideas.

4.8 The conservation of mass

4.8.1 Conservation of mass of water

The conservation of mass on a domain Ω is written as:

$$\int_\Omega \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right) d\Omega = 0 \tag{4.110}$$

With the calculation of $\Delta z w^{C*}$ through the resolution of this equation in the transformed mesh, we ensure that the conservative velocity field is indeed almost divergence-free, considering that the discretisation of the continuity equation induces some errors. Overall conservation of mass is actually possible without a local resolution of the continuity, hence without a divergence-free 3D velocity field. A more localized conservation will be imposed for the conservation of the scalar.

4.8.2 Conservation of scalar

The conservation of the scalar is written as:

$$\int_\Omega (T^{n+1} - T^n)\, d\Omega + \delta t \int_\Gamma T \boldsymbol{u} \cdot \boldsymbol{n}\, d\Gamma - \delta t \int_\Gamma K_E \frac{\partial T}{\partial n}\, d\Gamma - \delta t \int_\Omega Q\, d\Omega = 0 \tag{4.111}$$

where Q is a source term. If Dirichlet-type boundary conditions are imposed on the boundaries, this spoils the conservation and a correction is needed (see below in the section 4.9.3). The non-conservative equation resolved for each degree of freedom is as follows:

$$\int_\Omega \left(\frac{\partial T}{\partial t} + (\boldsymbol{u}^C - \boldsymbol{c}) \cdot \boldsymbol{\nabla} T - \nabla (K_E \nabla T) - Q \right) \Psi_i\, d\Omega = 0 \tag{4.112}$$

Considering only the advection of the scalar we have:

$$\int_\Omega (T^{n+1} - T^n) d\Omega + \delta t \int_\Omega (\boldsymbol{u}^C - \boldsymbol{c}) \cdot \boldsymbol{\nabla} T\, d\Omega = 0 \tag{4.113}$$

which, with (4.111), is equivalent to:

$$\int_\Omega (T^{n+1} - T^n) d\Omega + \delta t \int_\Gamma T \boldsymbol{u}^C \cdot \boldsymbol{n}\, d\Gamma = 0 \tag{4.114}$$

We dropped the diffusion term and the creation/destruction term Q which are not really a problem for the conservation of mass. This formulation in the real domain actually masks the problem of the evolution of Ω and the fact that the advection step of the scalar should be done in the fixed mesh if the advection was already done there. Written in the fixed mesh, the conservation of the scalar reads:

$$\int_{\Omega^*} (\Delta z^{n+1} T^{n+1} - \Delta z^n T^n) d\Omega^* + \delta t \int_{\Gamma^*} \Delta z T \boldsymbol{u}^{C*} \cdot \boldsymbol{n}\, d\Gamma^* = 0 \tag{4.115}$$

while the advection equation of the scalar in the fixed mesh at an instant taken between t^n and t^{n+1} gives:

$$\int_{\Omega^*} \Delta z (T^{n+1} - T^n)\, d\Omega^* + \delta t \int_{\Omega^*} \Delta z \boldsymbol{u}^* \cdot \boldsymbol{\nabla} T\, d\Omega^* = 0 \tag{4.116}$$

In this equation, if the fixed mesh is chosen at instant $t^n + \theta_h \delta t$, the following should be taken:

$$\Delta z = \theta_h \Delta z^{n+1} + (1 - \theta_h) \Delta z^n \tag{4.117}$$

We then need to choose an implicit expression of T for calculating its gradient. If we define:

$$T = \theta_T T^{n+1} + (1-\theta_T) T^n \tag{4.118}$$

the integral equation above, by adding the following null expression to it:

$$\delta t \int_{\Gamma^*} \Delta z T \boldsymbol{u}^{C*} \cdot \boldsymbol{n} \, d\Gamma^* - \delta t \int_{\Gamma^*} \Delta z T \boldsymbol{u}^{C*} \cdot \boldsymbol{n} \, d\Gamma^* \tag{4.119}$$

can then be written as:

$$\begin{aligned}
&\int_{\Omega^*} (\Delta z^{n+1} T^{n+1} - \Delta z^n T^n) \, d\Omega^* + \delta t \int_{\Gamma^*} \Delta z T \boldsymbol{u}^{C*} \cdot \boldsymbol{n} \, d\Gamma^* \\
&- \int_{\Omega^*} \theta_h T^n (\Delta z^{n+1} - \Delta z^n) \, d\Omega^* + (1-\theta_T) \delta t \int_{\Omega^*} \Delta z \boldsymbol{u}^{C*} \cdot \boldsymbol{\nabla} T^n \, d\Omega^* \\
&- (1-\theta_T) \delta t \int_{\Gamma^*} \Delta z T^n \boldsymbol{u}^{C*} \cdot \boldsymbol{n} \, d\Gamma^* - \int_{\Omega^*} (1-\theta_h) T^{n+1} (\Delta z^{n+1} - \Delta z^n) \, d\Omega^* \\
&+ \theta_T \delta t \int_{\Omega^*} \Delta z \boldsymbol{u}^{C*} \cdot \boldsymbol{\nabla} T^{n+1} \, d\Omega^* - \theta_T \delta t \int_{\Gamma^*} \Delta z T^{n+1} \boldsymbol{u}^{C*} \cdot \boldsymbol{n} \, d\Gamma^* = 0
\end{aligned} \tag{4.120}$$

It has to be proved that the last three lines are zero to guarantee the conservation of the scalar. By breaking down T^n and T^{n+1} on the basis Ψ_i^* of the transformed mesh, two conditions are sufficient:

$$\theta_h + \theta_T = 1 \tag{4.121}$$

and:

$$\int_{\Omega^*} (\Delta z^{n+1} - \Delta z^n) \Psi_i^* \, d\Omega^* - \delta t \int_{\Omega^*} \Delta z \boldsymbol{u}^{C*} \cdot \boldsymbol{\nabla} \Psi_i^* \, d\Omega^* + \delta t \int_{\Gamma^*} \Psi_i^* \Delta z \boldsymbol{u}^{C*} \cdot \boldsymbol{n} \, d\Gamma^* = 0 \tag{4.122}$$

At this point the variational formulation of the depth-integrated mass conservation equation appears. We arrive at **two fundamental conclusions**:

- the conservation of the mass of the scalar requires a compatible treatment of the depth-integrated continuity equation so that the condition (4.122) is satisfied;
- the condition $\theta_h + \theta_T = 1$ which signifies, for instance, that if an explicit scheme is used to calculate the value of $\int_{\Omega^*} \Delta z \boldsymbol{u} \cdot \boldsymbol{\nabla} T \, d\Omega^*$, the advection step should be done (paradoxically) in the mesh fixed at time t^{n+1}. A more detailed demonstration of scalar conservation will be given in the section 4.9.2 below, with the example of a distributive advection scheme.

4.9 Sources and sinks of fluid in the Navier–Stokes equations

In order to deal with water or scalar inputs into a domain, *e.g.* drainage pipes whose dimensions are less than the size of finite elements, we use the concept of sources and sink points in the domain. The fluid flow, with or without scalar, is considered as entering the calculation domain at a single point.

4.9.1 Sources of fluid

In 3D, it is now the continuity equation $\nabla \cdot \boldsymbol{u} = 0$ which is no longer verified locally. In the transformed mesh, starting again from the equation 1.256, we now have to solve:

$$\int_{\Omega^*}\left[\left(\frac{\partial \Delta z}{\partial t}\right)_{x,y}+\left(\frac{\partial \Delta z u^C}{\partial x}\right)_{y,z^*,t}+\left(\frac{\partial \Delta z v^C}{\partial y}\right)_{x,z^*,t}+\left(\frac{\partial \Delta z w^{C*}}{\partial z^*}\right)_{x,y,t}\right]\Psi_i^*\,d\Omega^*=\int_{\Omega^*}\Delta z\nabla\cdot\boldsymbol{u}^{C*}\Psi_i^*\,d\Omega^* \tag{4.123}$$

Recall that $\int_{\Omega^*}\Delta z\nabla\cdot\boldsymbol{u}^C\Psi_i^*\,d\Omega^*=\int_\Omega\nabla\cdot\boldsymbol{u}\Psi_i\,d\Omega$. With inputs our outputs of fluid in the domain, $\nabla\cdot\boldsymbol{u}^C$ will not be zero in certain points. The integral $\nabla\cdot\boldsymbol{u}^C$ around a domain surrounding a source point will be the discharge of the source Q_{sce}. To take this into account, one solution consists of adding to the right-hand side of the continuity equation the term:

$$\sum_{i3d\text{ above }i2d}\frac{\int_\Omega Q_{sce}\Psi_{isce}\Psi_{i3d}\,d\Omega}{\int_\Omega\Psi_{isce}\,d\Omega} \tag{4.124}$$

where *isce* is the point where the fluid is entering or leaving. It is worth noticing that:

$$\sum_{i3d\text{ above }i2d}\frac{\int_\Omega Q_{sce}\Psi_{isce}\Psi_{i3d}\,d\Omega}{\int_\Omega\Psi_{isce}\,d\Omega}\neq\frac{\int_{\Omega_{2D}}Q_{sce}\Psi^h_{isce}\Psi_{i2d}\,d\Omega_{2D}}{\int_{\Omega_{2D}}\Psi^h_{isce}\,d\Omega_{2D}} \tag{4.125}$$

On the other hand, the series of linear systems giving the vertical velocities averaged over each plane is then written for every point i in the 2D mesh as:

$$\begin{aligned}&\sum_{j=1}^{npoin2}\left\{\int_{\Omega_{2D}}\Psi_i^h\Psi_j^h\,d\Omega_{2D}\left[\overline{(\Delta z w^{C*})}^j_{ip+1/2}-\overline{(\Delta z w^{C*})}^j_{ip-1/2}\right]\right\}\\&=-\frac{1}{\delta t}\int_{\Omega^*}\left(\Delta z^{n+1}-\Delta z^n\right)\Psi_i^*\,d\Omega^*+\int_{\Omega^*}\Delta z\boldsymbol{u}\cdot\boldsymbol{\nabla}_{2D}\Psi_i^*\,d\Omega^*\\&-\int_{\Gamma^*_{liq}}\Delta z\boldsymbol{u}\cdot\boldsymbol{n}\Psi_i^*\,d\Gamma^*+\sum_{i3d\text{ above }i2d}\frac{\int_\Omega Q_{sce}\Psi_{isce}\Psi_{i3d}\,d\Omega}{\int_\Omega\Psi_{isce}\,d\Omega}\end{aligned} \tag{4.126}$$

The domain Ω is here considered at the same instant as the other flux terms.

4.9.2 Sources of scalar with a distributive scheme

Having seen both the conservation of scalar mass and the sources of water, we are now in a position to understand how to deal with sources of scalars. As a matter of fact it is necessary to rely on the proof of conservation of the scalar mass to find out which source terms to apply to it. The treatment of the advection terms has its importance and we take here the example of the MURD scheme which will be presented in the section 4.10.2. It is an explicit scheme and it is about the only thing to recall in what follows. We recall the following notations:

$$FLUINT(i)=\int_{\Omega^*}\Delta z\boldsymbol{u}^C\cdot\boldsymbol{\nabla}_{2D}\Psi_i^*\,d\Omega^* \tag{4.127}$$

$$FLUVER(i)=\int_{\Omega^*}\Delta z w^{C*}\frac{\partial\Psi_i^*}{\partial z^*}\,d\Omega^* \tag{4.128}$$

$$SOURCE(i)=\frac{\int_\Omega Q_{sce}\Psi_{isce}\Psi_i d\Omega}{\int_\Omega\Psi_{isce}\,d\Omega}\alpha(i) \tag{4.129}$$

$$FLUEXT(i) = \int_{\Gamma^*_{liq}} \Delta z \boldsymbol{u}^C \cdot \boldsymbol{n} \Psi_i^* \, d\Gamma^* \tag{4.130}$$

The continuity equation is then written as:

$$\frac{1}{\delta t} \int_{\Omega^*} \left(\Delta z^{n+1} - \Delta z^n \right) \Psi_i^* \, d\Omega^* = FLUINT(i) + FLUVER(i) - FLUEXT(i) + SOURCE(i) \tag{4.131}$$

The equation for the scalar T is as follows:

$$\frac{1}{\delta t} \int_{\Omega^*} \Delta z^{n+1} \left(T^{n+1} - T^n \right) \Psi_i^* \, d\Omega^* = - \int_{\Omega^*} \Delta z \boldsymbol{u} \cdot \boldsymbol{\nabla} T^n \Psi_i^* \, d\Omega^* + SOURCE\_TRAC(i) \tag{4.132}$$

where the term $\int_{\Omega^*} \Delta z \boldsymbol{u} . \boldsymbol{\nabla}(T^n) \Psi_i^* d\Omega^*$ is explicit and calculated by the distributive scheme. $SOURCE\_TRAC(i)$ is the source term we look for. The overall conservation of the scalar mass is shown by first multiplying all the continuity equations by T_i^n, then by adding them, and eventually by adding to them all the scalar equations, which gives:

$$\begin{aligned} &\frac{1}{\delta t} \int_{\Omega^*} \left(\Delta z^{n+1} - \Delta z^n \right) T^n \, d\Omega^* + \frac{1}{\delta t} \int_{\Omega^*} \Delta z^{n+1} \left(T^{n+1} - T^n \right) \, d\Omega^* \\ &= \sum_i FLUINT(i) T_i^n + \sum_i FLUVER(i) T_i^n - \sum_i FLUEXT(i) T_i^n \\ &+ \sum_i SOURCE(i) T_i^n - \int_{\Omega^*} \Delta z \boldsymbol{u} \cdot \boldsymbol{\nabla} T^n \, d\Omega^* + \sum_i SOURCE\_TRAC(i) \end{aligned} \tag{4.133}$$

Now, the distributive scheme ensures that:

$$\sum_i FLUINT(i) T_i^n + \sum_i FLUVER(i) T_i^n - \int_{\Omega^*} \Delta z \boldsymbol{u} \cdot \boldsymbol{\nabla} T^n \, d\Omega^* = 0 \tag{4.134}$$

because it simply performs a redistribution of the first two terms without changing its sum. We finally arrive at:

$$\begin{aligned} \frac{1}{\delta t} \int_{\Omega^*} \left(\Delta z^{n+1} T^{n+1} - \Delta z^n T^n \right) \, d\Omega^* \quad &= - \sum_i FLUEXT(i) T_i^n + \sum_i SOURCE(i) T_i^n \\ &+ \sum_i SOURCE\_TRAC(i) \end{aligned} \tag{4.135}$$

In the case of a source with a scalar value T_{sce}, the conservation of mass of scalar imposes that:

$$\sum_i SOURCE(i) T_i^n + \sum_i SOURCE\_TRAC(i) = Q_{sce} T_{sce} \tag{4.136}$$

One solution for a source is then:

$$SOURCE\_TRAC(i) = (T_{sce} - T_i^n) SOURCE(i) \tag{4.137}$$

If there are several sources, there should be a distinction between the $SOURCE(i)$ vectors for each source. If the source is actually a sink, then $SOURCE(i)$ is negative and a different treatment is necessary. As a matter of fact, in a sink, the value of the scalar now exiting the domain is not T_{sce}, but rather the local value T_i, so that $SOURCE\_TRAC(i)$ must in fact be cancelled. Everywhere T_{sce} appears, it must be replaced by T_i (here T_i^n because we have an explicit scheme, but it could as well be T_i^{n+1} with an implicit scheme). To take sinks into account as well as sources, we must thus consider that:

$$SOURCE\_TRAC(i) = (T_{sce} - T_i^n) MAX(SOURCE(i), 0) \tag{4.138}$$

4.9.3 Dirichlet boundary conditions with a distributive scheme

As the previous subsections give a full proof of the mass conservation of the scalar, we are now in a position to deal with the problem of imposed boundary conditions. The distributive schemes pay no attention to prescribed values at the boundaries; they just take the scalar at time t^n and ensure mass conservation and monotonicity. A user may want to prescribe given values at an entrance to the domain, or a given scalar flux. If nothing special is done the flux of the scalar through the boundaries (here positive if entering the domain) will be:

$$observed\ flux = -\int_{\Gamma^*_{liq}} \Delta z T^n \boldsymbol{u}.\boldsymbol{n}\, d\Gamma^* \tag{4.139}$$

$$= -\sum_i T_i^n \int_{\Gamma^*_{liq}} \Delta z \boldsymbol{u}\cdot\boldsymbol{n}\Psi_i^*\, d\Gamma^* = -\sum_i T_i^n FLUEXT(i) \tag{4.140}$$

and the imposed values will not be ensured. The summation on i must be understood as i being on the boundary with prescribed values. The user would in fact like to have:

$$wanted\ flux = -\int_{\Gamma^*_{liq}} \Delta z T^{imposed} \boldsymbol{u}\cdot\boldsymbol{n}\, d\Gamma^* \ \text{ and } \ T_i^{n+1} = T_i^{imposed} \text{ on boundaries} \tag{4.141}$$

Both conditions are not compatible. Brutally setting $T_i^{n+1} = T_i^{imposed}$ or $T_i^n = T_i^{imposed}$ on boundaries will add or remove scalar mass in the domain, because the basis functions on the boundaries go inside the domain, thus creating an artificial flux. We propose here a relaxation of the Dirichlet conditions which consists of replacing T_i^n by:

$$T_i^n + \theta(T_i^{imposed} - T_i^n) = T_i^n + \Delta T^n \tag{4.142}$$

on liquid boundaries with imposed values, with $0 < \theta < 1$. θ is computed so that the sum of the *a priori* added mass and the observed flux gives the wanted flux. The added mass is:

$$added\ mass = \theta(T_i^{imposed} - T_i^n)\int_{\Omega^*} \Delta z^n \Psi_i^*\, d\Omega^* = \theta(T_i^{imposed} - T_i^n)\int_{\Omega^n} \Psi_i\, d\Omega^n \tag{4.143}$$

the new observed flux is:

$$observed\ flux = -\left[T_i^n + \theta(T_i^{imposed} - T_i^n)\right] FLUEXT(i) \tag{4.144}$$

As we want:

$$added\ mass\ +\ \delta t * observed\ flux\ =\ \delta t * wanted\ flux \tag{4.145}$$

we have to choose:

$$\theta = \frac{-FLUEXT(i)}{\left[-FLUEXT(i) + \dfrac{1}{\delta t}\int_{\Omega^n}\Psi_i\, d\Omega^n\right]} \tag{4.146}$$

This local value of θ is considered only for entrances, i.e. when $FLUEXT(i)$ is negative, then θ is always between 0 and 1.
After applying the distributive scheme the value of the scalar initially set at the boundary, $T_i^n + \theta(T_i^{imposed} - T_i^n)$, will have been slightly modified by the algorithm but still ensures monotonicity, and the correct prescribed flux will be observed if a mass balance is done.

4.9.4 Sources of scalar with the SUPG advection scheme

We have taken the example of a distributive scheme for our explanations in the last two subsections. The theory with a SUPG advection scheme (see [36]) would be similar but with a few modifications detailed hereafter, due to the fact that this scheme is implicit, the scalar considered being $\theta_T T^{n+1} + (1-\theta_T)T^n$. The mesh used for the advection equation must be taken at time $(1-\theta_T)\,t^{n+1} + \theta_T\, t^n$. The derivative in time of the scalar is thus in the transformed mesh:

$$\frac{1}{\delta t}\int_{\Omega^*}\left[(1-\theta_T)\Delta z^{n+1} + \theta_T \Delta z^n\right]\left(T^{n+1}-T^n\right)\Psi_i^*\, d\Omega^* \tag{4.147}$$

Consequently, in the proof of scalar mass conservation, Equation 4.25 is no longer multiplied by T_i^n but by $\theta_T T_i^{n+1} + (1-\theta_T)T_i^n$ so that the combination and sum of all equations form on the left-hand side the term:

$$\frac{1}{\delta t}\int_{\Omega^*}(\Delta z^{n+1}T^{n+1} - \Delta z^n T)\, d\Omega^* \tag{4.148}$$

As will be seen later, the SUPG theory introduces an extra term into the scalar equation, in the form:

$$-\int_{\Omega^*}\Delta z \boldsymbol{u}.\boldsymbol{\nabla}(T^n)\, k\frac{\boldsymbol{u}}{|\boldsymbol{u}|}\,.\,\boldsymbol{\nabla}(\Psi_i^*)\, d\Omega^* \tag{4.149}$$

on the right-hand side, but this term is cancelled when summed over all bases Ψ_i^* because their sum equals 1.

As in our proof $SOURCE(i)$ in the continuity equation is now multiplied by $\theta_T T_i^{n+1} + (1-\theta_T)T_i^n$, the source of the scalar is accordingly modified in the scalar equation:

$$SOURCE\_TRAC(i) = \left[T_{sce} - \theta_T T_i^{n+1} + (1-\theta_T)T_i^n\right] SOURCE(i) \tag{4.150}$$

4.9.5 Dirichlet boundary conditions with a SUPG advection scheme

In the case of an implicit advection scheme, advection and diffusion are solved together. The difficulty of ensuring a correct flux corresponding to prescribed boundary conditions is that the flux cannot be *a priori* known, as it was with an explicit scheme. An *a posteriori* correction is thus necessary. However, we choose to start with the *a priori* correction of the distributive scheme, thus adding the increment ΔT^n to T^n as in Equation 4.142. Then the advection–diffusion algorithm is applied, starting from $T_i^n + \Delta T^n$, and yields a result denoted T^D. Eventually an increment ΔT^{n+1} is added to T^D, such that:

$$a\ posteriori\ added\ mass \tag{4.151}$$

$$= \Delta T^{n+1}\int_{\Omega^n}\Psi_i\, d\Omega^{n+1} = -\delta t\, FLUEXT(i)\,\theta\,(T_i^n + \Delta T^n - T_i^D) \tag{4.152}$$

As the observed flux is:

$$observed\ flux = -\delta t\left[T_i^n + \Delta T^n + \theta(T_i^D - T_i^n - \Delta T^n)\right] FLUEXT(i) \tag{4.153}$$

and as:

$$a\ priori\ added\ mass = \Delta T_i^n\int_{\Omega^n}\Psi_i\, d\Omega^n = -\delta t\, FLUEXT(i)\left[T_i^{imposed} - T_i^n - \Delta T^n\right] \tag{4.154}$$

we can check that we recognize the required property:

$$a\ priori\ added\ mass + a\ posteriori\ added\ mass + observed\ flux \tag{4.155}$$

$$= -T_i^{imposed}\,\delta t\, FLUEXT(i) \tag{4.156}$$

which is the wanted flux.

4.10 Solving transport equations in 3D

This section is focused on the resolution of the advection problem:

$$\frac{\partial c}{\partial t} + \boldsymbol{u}\cdot\boldsymbol{\nabla} c = 0 \tag{4.157}$$

with c a given field (which can be the velocity, a tracer, a turbulent quantity, etc.) and $\boldsymbol{u}$ the advecting velocity field. In TELEMAC-3D, there are three types of advection schemes available: the method of characteristics, the Streamline Upwind Petrov Galerkin scheme (SUPG) and the residual distribution schemes. There are actually 13 possible choices for the advection schemes, for each advected variable:

- the method of characteristics, with two variants: the strong and weak forms
- the SUPG scheme (with options for the upwinding);
- the MURD scheme N and the MURD scheme PSI, with three variants each: explicit, first-order predictor-corrector or second-order predictor-corrector
- a scheme by Leo Postma, with or without tidal flats treatment;
- the N scheme with tidal flats treatment, in its explicit (also known as NERD) or predictor-corrector (also known as LIPS) forms.

These schemes are described in more details in the subsections below, but before this, the Table 4.1 summarises their main properties:

- **conservation**: ability of the scheme to conserve the total quantity of the advected quantity during the transport;
- respect of the **maximum principle**: the ability of the scheme to respect the physical bounds of the quantity (*e.g.* always positive for a concentration or a water depth, etc.);
- **ability to handle tidal flats** or not;
- **order** of the scheme: rate of the error decrease with respect to the mesh refinement, for unsteady cases (the PSI scheme is actually a 2nd order scheme on steady cases);
- **numerical diffusion**: spurious diffusion introduced by the scheme, that tends to make fields more homogeneous than they should be.

The overall accuracy of a scheme depends on its order and on the numerical diffusion it introduces: the order itself is not sufficient to conclude on the accuracy. For example, the method of characteristics, which is a first-order scheme, usually proves more accurate than the classical N or PSI schemes, which are also first-order schemes. The predictor-corrector versions of the N and PSI schemes introduce much less numerical diffusion than any other advection scheme implemented in TELEMAC-3D (except maybe the characteristics, but this method does not

conserve the quantity of tracer). The predictor-corrector version of the N and PSI schemes was made able to deal with tidal flats by using a locally implicit treatment on dry zones. Anywhere else in the flow it comes down to the first-order predictor-corrector scheme. Comparisons between the schemes are available in the validation document associated with the latest version of TELEMAC (see for example the cone and lock-exchange examples). In general, a good recommandation would be to use either the method of characteristics or the first-order predictor-corrector PSI scheme for the advection of velocities and the first-order predictor-corrector PSI scheme for the advection of tracers. Nonetheless, each scheme has advantages and drawbacks that may result in different choices depending on the simulated case: we always seek a balance between the physical properties (conservation, respect of the maximum principle, tidal flats treatment), the accuracy and the computational time. Note that the respect of the maximum principle is important for the tracers advection to avoid unphysical values. The Table 4.1 is meant to help in the choice of the advection scheme.

Table 4.1: Main properties of the advection schemes available in TELEMAC-3D. The order column refers to the space and time order of the scheme and the indications about numerical diffusion are qualitative ($++$ designating the schemes with most numerical diffusion and $--$ the schemes with least numerical diffusion).

Scheme	Conservation	Max. principle	Tidal flats	Order	Num. diffusion
Characteristics	No	Yes	Yes	1	$-$
Weak characteristics	No	No	Yes	1	$--$
SUPG	No	No	No	1	$+$
Classical N or PSI	Yes	Yes	No	1	$++$
1st order pred/corr N or PSI	Yes	Yes	No	1	$--$
2nd order pred/corr N or PSI	Yes	Yes	No	2	$--$
Leo Postma wo/ tidal flats	Yes	Yes	No	1	$++$
Leo Postma w/ tidal flats	Yes	Yes	Yes	1	$++$
NERD	Yes	Yes	Yes	1	$++$
LIPS N or PSI	Yes	Yes	Yes	1	$--$

4.10.1 Method of characteristics

A lot of litterature can be found on the method of characteristics, but in the case of advection it merely consists in computing pathlines of the flow, called characteristics (actually trajectories or streamlines if the velocity is considered constant in time). Its simplicity makes it very popular, together with the fact that it is unconditionnaly stable and fullfils the monotonicity criterion. Besides, it can be very accurate in spite of being a first-order scheme. However, it suffers from several drawbacks:

- it is quite complex to come up with an effective implementation in parallel computations (this has been overcome in the TELEMAC system);
- it does not ensure conservation of the advected quantity: as a consequence, its use is not recommanded for the advection of tracers.

In the absence of terms other than advection, the advected variables are constant on the characteristics, and are thus taken at the previous time step at what is called the foot of the characteristic, namely the place where a particle of water, that is now on a given point of the mesh, was

at the previous time step. Basically the method of characteristics is thus a computation of trajectory followed by an interpolation. We shall not here describe the navigation in the mesh and how it works in parallel, but just deal with the specific 3D aspects, the fact that the advection in TELEMAC-3D is done in a transformed mesh to account for relocalisation. The theory behind the vertical displacements in the method of characteristics in 3D is not straightforward and we give hereafter some explanations.

The advection equation is solved in a transformed mesh with horizontal planes. The sigma transformation is used to take the mesh displacement into account. The physical values at any point of the mesh change due to advection terms (velocity of the flow) but also due to the mesh's displacement. It is shown in the reference [36] page 21, equation 2.90 that this can be simply taken into account by an advection in the transformed mesh. One would thus expect that, when computing the characteristics pathlines, the vertical displacements are of the form:

$$\Delta z^* = w^* \delta t \tag{4.158}$$

using the transformed coordinates and the transformed velocities, but we find in the implementation the following form:

$$\Delta z^* = w \, \frac{ZSTAR(IET+1) - ZSTAR(IET)}{\Delta z} \, \delta t \tag{4.159}$$

where Δz is the local real width of the given layer between two planes and $ZSTAR$ is the transformed elevation of planes (here $IET+1$ and IET) in the transformed mesh. The velocity w provided by TELEMAC-3D to the subroutine CHARAC is in fact $w^*\Delta z$. This quantity is provided by the subroutine TRIDW2 and has the dimension of a velocity, it is linked to the real vertical velocity by the formula:

$$w = \frac{\partial z}{\partial t} + u\frac{\partial z}{\partial x} + v\frac{\partial z}{\partial y} + w^*\frac{\partial z}{\partial z^*} \tag{4.160}$$

where $\partial z/\partial z^*$ is Δz when we apply a sigma transformation that changes a layer of width Δz into a layer of width 1. Our vertical displacements are thus in reality:

$$\Delta z^* = w^* \frac{\partial z}{\partial z^*} \, \frac{ZSTAR(IET+1) - ZSTAR(IET)}{\Delta z} \, \delta t \tag{4.161}$$

and $(ZSTAR(IET+1) - ZSTAR(IET))/\Delta z$ is here to give the value $\partial z^*/\partial z$ that retrieves the wanted formula $\Delta z^* = w^* \delta t$. At this level we understand that the values of ZSTAR must correspond to a sigma transformation that changes a layer Δz into a layer of width 1. This is linked to the choice done for the mesh displacement (classical sigma transformation or generalised sigma transformation). In case the classical sigma transformation is used, $(ZSTAR(IET+1) - ZSTAR(IET))/\Delta z$ can be determined on the whole mesh at once:

$$ZSTAR(IET) = \frac{IET-1}{NPLAN-1} \tag{4.162}$$

where NPLAN is the number of planes. In $(ZSTAR(IET+1) - ZSTAR(IET))/\Delta z$, the denominator is the real width of a layer, so the correction factor is actually a constant over the mesh if the values of ZSTAR are proportional to the real z coordinates. For example, this is the case if they are the values of the array ZSTAR used for building the mesh, i.e. a sigma transformation giving the whole 3D mesh a width of 1. On the other hand, in case the generalised sigma transformation is used, it is necessary to stop at every plane, and recompute a new displacement compatible with the new metrics.

4.10.2 MURD schemes in three dimensions

The MURD scheme of Janin [42], which signifies "Multidimensional Upwind Residual Distribution", is an adaptation to prisms of the N and PSI schemes that are described for triangular meshes in [36] and [62]. We shall first recall some elements on the N and PSI schemes on triangular meshes (based on Sara Pavan's PhD thesis [62]), then look successively at the two variants in 3D: N and PSI. These schemes belong to the family of residual distribution schemes (RD).

Residual distribution schemes (from (62))

The residual distribution (RD) technique is less popular and less developed than the FV and FE methods, but it has been an important research topic in TELEMAC, especially for the advection of tracers, to achieve the conservation and respect of the maximum principle with the highest possible accuracy. The overview given in this section focuses on the principle of RD schemes and on the technique of predictor-corrector schemes to achieve second (or higher) order in space and in time, for a general conservation law. In the framework of the RD method, the solution is approximated in the piecewise linear finite-element space over an arbitrary unstructured grid: $c_h(x,y,t) = \sum_{i}^{npoin} c(x_i,y_i,t)\psi_i(x,y)$ where $npoin$ is the total number of nodes in the mesh, $c(x_i,y_i,t)$ is the time dependent nodal value of the solution at node i and $\psi_i(x,y)$ is the piecewise linear shape function that benefits from the classical properties of linear P^1 FE basis functions.
The integration of the scalar advection equation (4.157) (without sources) over a single time step, using an explicit time integration, leads to the following form:

$$c_i^{n+1} = c_i^n - \frac{\delta t}{S_i} \sum_{T \ni i} \phi_i^E \tag{4.163}$$

where c_i^n is the nodal value at time n and c_i^{n+1} is the nodal value at time $n+1$, S_i represents the area of the cell around the point i, equal to $\sum_{T \ni i} \frac{T}{3}$ and ϕ_i^E is the splitting residual at node i, computed for every element that contains the node i and then summed up.
The quantity ϕ^E, called the residual, is computed as:

$$\phi^E = \int_E \boldsymbol{u} \cdot \nabla c_h \, dxdy \tag{4.164}$$

By construction, the consistency relation:

$$\sum_{i \in T} \phi_i^E = \phi^T \tag{4.165}$$

is satisfied. Another fundamental relation is $\phi_i^T = \beta_i \phi^T$, where β_i represents the distribution coefficient. The way to distribute the residual to the nodes (i.e. through the distribution coefficients) influences a series of properties of the scheme like the positivity, the linearity preservation and the multidimensional character. Hence different schemes with different properties can be created with a different residual distribution.
Every property characterizes the solution in a certain way and they are briefly recalled here below [24]:

- Positivity. A scheme of the form:

$$\phi_i^E = \sum_{j \in T} \lambda_{ij}^E (c_i^n - c_j^n) \qquad \text{with } \lambda_{ij}^E \geq 0 \tag{4.166}$$

and able to respect the maximum principle, under the time step condition:

$$S_i - \delta t \sum_{T \ni i} \sum_{j \in T} \lambda_{ij}^E \geq 0 \quad \forall i \in \mathscr{T}_h \tag{4.167}$$

is said to be positive. This property is thus related to the non-oscillatory character of the solutions.

- Linearity Preserving. A scheme is linearity preserving if its distribution coefficients β_i are uniformly *bounded* with respect to the solution and the data of the problem:

$$\max_{T \in \mathscr{T}_h} \max_{j \in T} |\beta_j| < C < \infty \quad \forall \phi^E, c_h, c_h^0 \tag{4.168}$$

where C is a constant. Linearity preserving schemes satisfy by construction the necessary condition for second order accuracy, hence this property is related to the accuracy of the scheme.

- Genuinely Multidimensional Upwind procedure: multidimensional upwind schemes only send portions of ϕ^E to downstream nodes. This property corresponds to the generalization of the 1D upwind idea. It is related to the stability of the scheme.

The origin of these schemes traces back to Ni [57] and Roe [83], who, in 1987, proposed the name of fluctuation splitting schemes. In [83], the upwind treatment of the scalar convection equation was generalized in two space dimensions, defining the multidimensional upwind character of these schemes. In the same work, also the most successful first order scheme was introduced. It is the N (narrow) scheme, also known as the optimum first order scheme, since among the first order schemes it is the least diffusive. The scheme is positive, multidimensional upwind and linear because of the Godunov theorem, it is limited to be only of first order accuracy.
However, due to its properties, the scheme plays an important role in the construction of second order schemes.

There are several ways to increase the order of a RD scheme. For more details, please consult Sara Pavan's PhD thesis [62]. Here we first focus on the PSI scheme, then on the predictor-corrector technique, which are the ones implemented in TELEMAC-3D. In the RD scheme, the truncation error is defined following a variational approach, see for example [1, 5, 24]. From the consistency analysis it is possible to show that, in two spaces, a RD scheme verifies the truncation error estimate:

$$TE(w_h) := \sum_{i \in \mathscr{T}_h} \varphi_i \sum_{T \ni i} \phi_i(w_h) = \mathscr{O}(\Delta x^k) \tag{4.169}$$

provided that the following condition is met:

$$\phi_i(w_h) = \mathscr{O}(\Delta x^{k+1}) \quad \forall i \in T \text{ and } T \in \mathscr{T}_h \tag{4.170}$$

with φ a smooth compact function $\varphi \in C_0^k(\Omega)$, w_h the $k-th$ order accurate continuous piecewise polynomial of degree $k-1$ interpolant of w, a smooth exact solution of the PDE. This condition guarantees formally that the scheme has an $\mathscr{O}(\Delta x^k)$ error [24].

The N scheme only preserves $\mathscr{O}(\Delta x^1)$ and does not guarantee the LP property since the distribution coefficients are unbounded, but it can be the basis for a second order scheme. Indeed, in order to obtain bounded distribution coefficients and thus meet the condition $\phi_i = \mathscr{O}(\Delta x^3)$, the

non linear limiter PSI is introduced. The original PSI scheme was presented in 1994 by Struijs Struijs [90] in his PhD, and it met a large success (see [58, 83, 84]) due to its good performance, which is better than FV schemes, especially on irregular grids [24] for steady scalar problems. The scheme is related to the N scheme since the limiter is applied on the distribution coefficients of the N scheme.

Later, a more general framework to construct non linear *limited* second order schemes was presented in [4, 5, 58]. In the RD method, the limiter is used in a completely different manner than in the FV method. In this case, the role of the limiter is to bound the coefficients and to preserve the positivity of the first order scheme coefficients, while in the FV context the limiter is used to limit the slopes and so to avoid oscillations in the solution. This technique, which is the one commonly used in practice, has been known for a very long time (1995), and improved constructions have not been published since then [24]. In the RD framework the first order scheme is critically important in the construction of the second order one. Beyond the N scheme, the Lax–Friedrichs scheme is often used as a basis for the construction of a limited non linear variant [2, 76, 79, 80]. However, all these formulations, which are based on the prototype (4.163) are only first order accurate in time dependent problems. This is due to an *inconsistency* in the spatial discretization [24], which is independent of the order of the time discretization. Essentially, whatever the choice of the discretization of the time derivative, the scheme has a discretization error bounded by $\mathscr{O}(\Delta x)$ [78]. This can be demonstrated by performing a time continuous consistency analysis (see [24, 77]).
To overcome this issue, several solutions have been tested based on a new concept of residual. The difference consists of including the time derivative in the computation of the residual, which thus becomes a space-time residual. This operation is necessary to recover second order accuracy in space and it leads to the formation of a mass-matrix of the time derivative, exactly like in the FE context, except that in this case the problem has *not* been formulated using a variational approach. The various approaches proposed to solve time dependent problems can be classified in three families of schemes:

- space-time schemes,
- implicit in space schemes,
- explicit predictor-corrector schemes.

Here we focus on the last one: the predictor-corrector schemes. It is a very attractive alternative since it does not lead to the resolution of a non-linear system, while the other two do. Instead, it is replaced by two explicit resolutions. It follows that the final scheme is cheaper and efficient. This method is the most recent with respect to the other schemes for time dependent problems: it was initially published in 2010 by Ricchiuto and Abgrall [76], then applied to the SW system in [79] and finally combined with an ALE formulation [7]. A discontinuous RD based version of the predictor-corrector scheme is also presented in [99]. The genuinely explicit RD approach consists of the two steps:

$$\left\{\begin{array}{rcl} |S_i|\dfrac{c_i^*-c_i^n}{\delta t} & = & -\displaystyle\sum_{T\ni i}\beta_i\int_T \boldsymbol{u}\cdot\nabla c_h^n \\ |S_i|\dfrac{c_i^{n+1}-c_i^*}{\delta t} & = & -\displaystyle\sum_{T\ni i}\beta_i\int_T\left(\dfrac{c_h^*-c_h^n}{\delta t}+\dfrac{1}{2}\boldsymbol{u}\cdot\nabla c_h^n+\dfrac{1}{2}\boldsymbol{u}\cdot\nabla c_h^*\right) \end{array}\right. \tag{4.171}$$

The unknown is initially approximated with a classical scheme for steady problem and then it is corrected in the second step. This formulation stems from a complex construction, detailed in [76], which is briefly recalled here:

1. formulation of a bubble stabilized Galerkin scheme;
2. construction of a modified semi-discrete residual guaranteeing that the overall accuracy is not reduced. In particular, the authors show that for a second order scheme a first order semi-discrete operator is sufficient;
3. RK time step formulation, consistent with the semi-discrete residual;
4. mass lumping to avoid the inversion of a mass-matrix.

Obviously, as for the other families of time dependent schemes, the distribution coefficient of expression (4.171) must be uniformly bounded. Thus they can be generated by the limitation of positive first order schemes, like the Lax Friedrichs or the N scheme, in order to have a positive second order scheme. Otherwise, other non positive first order schemes can be used if the goal is simply to have a second order scheme (not monotone). Concerning the stability of the scheme, the authors affirm that a Fourier analysis on structured triangulation is under way to have better estimates of the computational time step and the stability (or positivity) of the various RD formulations is currently being investigated. However, for the scalar homogeneous case, the theory of positive coefficients [24, 89] can be applied and precise conditions can be found to bound the numerical solution [79]. To conclude, the conservation issue for RD schemes is briefly addressed. Conservative formulations of RD schemes for systems of (or scalar) non linear conservation laws, are linked to the computation (and the existence) of conservative linearization of the multidimensional flux jacobian over the mesh cells. This problem was investigated in a series of papers [3, 19, 77, 81]. The classical solution is to compute the residual through *exact* mean values Jacobians. In [19], this procedure is called linearization-based RD. Unfortunately such a procedure can be too difficult or impossible to implement. This is the case for the shallow water system. Thus, a first solution is presented in [3], where the exact mean values Jacobians are replaced by approximated values obtained from adaptive quadrature of the quasi-linear form. With this approach, it is also possible to find a corresponding Lax Wendroff theorem for RDS and to show that these schemes converge to the correct weak solutions with some assumptions and for a certain (large) number of Gauss points. Another solution, which became the most popular, is to use a contour integration based RD [19, 77, 81]. Such a procedure is easier and less expensive than the previous one. It consists of approximating directly the contour integral of the convective fluxes over the boundaries of an element. This technique is in general the one used for SW discretization. The theoretical aspects regarding the 3D predictor-corrector RD schemes are based on the ones of the context of shallow water equations: for more details, please consult the Chapter 5 of Sara Pavan's PhD thesis [62].

N-type MURD scheme

We work with the transformed fixed mesh and we want to guarantee the global conservation of the quantity c by showing that one has:

$$\int_{\Omega^*} \Delta z^{n+1} \frac{c^{n+1}-c^n}{\delta t}\, d\Omega^* + \int_{\Omega^*} \Delta z \boldsymbol{u}^* \cdot \boldsymbol{\nabla} c^n \, d\Omega^* = 0 \tag{4.172}$$

based on our local equations. We recall that $\boldsymbol{u}^*$ is made up of the components (u, v, w^*).
The first term is initially written as:

$$\int_{\Omega^*} \Delta z^{n+1} \frac{c^{n+1}-c^n}{\delta t}\, d\Omega^* = \sum_{ip=1}^{np} \frac{c_{ip}^{n+1}-c_{ip}^n}{\delta t} \int_{\Omega^*} \Psi_{ip}^* \Delta z^{n+1}\, d\Omega^* \tag{4.173}$$

We later denote:

$$\int_{\Omega^*} \Psi_{ip}^* \Delta z^{n+1}\, d\Omega^* = S_{ip} \tag{4.174}$$

to obtain $\sum_{ip=1}^{np}(c_{ip}^{n+1}-c_{ip}^{n})S_{ip}/\delta t$, the first term of the equation of a point ip. For the second term, $\int_{\Omega^*}\Delta z\boldsymbol{u}\cdot\boldsymbol{\nabla}c^n\,d\Omega^*$, we shall calculate it element by element, and for each element share the contributions over the points it is made up of. The sum of all our equations will again give us the global equation desired. By limiting ourselves to a prism P^* of the transformed mesh, we shall thus try to calculate:

$$\Phi_P=\int_{P^*}\Delta z\boldsymbol{u}\cdot\boldsymbol{\nabla}c\,dP^* \tag{4.175}$$

by trying to put it into the form:

$$\Phi_P=\sum_{i=1}^{6}\sum_{j=1}^{6}\lambda_{ij}(c_i^n-c_j^n) \tag{4.176}$$

with the coefficients λ_{ij} positive or zero, as we did for the N scheme. By decomposing c, we already know that:

$$\Phi_P=\sum_{i=1}^{6}c_i^n\int_{P^*}\Delta z\boldsymbol{u}\cdot\boldsymbol{\nabla}\Psi_i\,dP^* \tag{4.177}$$

Now the terms:

$$\int_{P^*}\Delta z\boldsymbol{u}\cdot\boldsymbol{\nabla}\Psi_i\,dP^* \tag{4.178}$$

can be decomposed in horizontal or vertical gradients in the form a_i+b_i with:

$$a_i=\int_{P^*}\Delta zu\frac{\partial\Psi_i}{\partial x}\,dP^*+\int_{P^*}\Delta zv\frac{\partial\Psi_i}{\partial y}\,dP^* \tag{4.179}$$

and:

$$b_i=\int_{P^*}\Delta zw^*\frac{\partial\Psi_i}{\partial z^*}\,dP^* \tag{4.180}$$

As we did in Section 4.6.2, we now decompose the bases $\Psi_i(x,y,z^*)$ of the prism into the product $\Psi_i^h(x,y)\Psi_i^v(z^*)$, which gives:

$$a_i=\frac{\partial\Psi_i^h}{\partial x}\int_{P^*}\Delta zu\Psi_i^v\,dP^*+\frac{\partial\Psi_i^h}{\partial y}\int_{P^*}\Delta zv\Psi_i^v\,dP^* \tag{4.181}$$

and:

$$b_i=\frac{\partial\Psi_i^v}{\partial z^*}\int_{P^*}\Delta zw^*\Psi_i^h\,dP^* \tag{4.182}$$

The properties of the bases $\Psi_i^h(x,y)$ and $\Psi_i^v(z^*)$ then give us a group of properties for coefficients a_i and b_i:

- $a_1+a_2+a_3=0$ inferred by $\Psi_1^h+\Psi_2^h+\Psi_3^h=1$
- $a_4+a_5+a_6=0$ inferred by $\Psi_4^h+\Psi_5^h+\Psi_5^h=1$
- $b_1+b_4=0$ inferred by $\Psi_1^v+\Psi_4^v=1$
- $b_2+b_5=0$ inferred by $\Psi_2^v+\Psi_5^v=1$
- $b_3+b_6=0$ inferred by $\Psi_3^v+\Psi_6^v=1$

Here, we have again used the local numbering of the points in a prism. Based on the N scheme, which is built on triangles, a first set of coefficients λ_{ij} could be the following:

- $\lambda_{ij} = \max(\min(a_i, -a_j), 0)$ for the points i and j of the lower triangle of the prism, or for those of the upper triangle.
- $\lambda_{ii+3} = \max(b_i, 0)$ for the points i of the lower triangle.
- $\lambda_{i+3i} = \max(b_{i+3}, 0)$ for the points i of the upper triangle.
- $\lambda_{ij} = 0$ for the other terms (which are "crossed" terms, linking the points which are neither at the same level, nor on the same vertical line).

We already possess all the good properties to construct an N-type scheme (all the coefficients are positive or zero). However, one could obtain a less diffusive scheme by transforming the sums of horizontal and vertical fluxes into crossed fluxes, which can be done by noticing that for a set of three coefficients λ_{ij}, λ_{jk} and λ_{ik}, one does not change the value of Φ_P by carrying out the following series of operations:

- $\lambda_{ik} = \lambda_{ik} + \min(\lambda_{ij}, \lambda_{jk})$
- $\lambda_{ij} = \lambda_{ij} - \min(\lambda_{ij}, \lambda_{jk})$
- $\lambda_{jk} = \lambda_{jk} - \min(\lambda_{ij}, \lambda_{jk})$

One in fact observes that the coefficients of c_i^n, c_j^n and c_k^n remain unchanged. We now have a supplementary property, which is that if a coefficient λ_{ij} is strictly positive, then all the coefficients λ_{jk} are zero. We shall choose, as in two dimensions, the distribution coefficients β_i as:

$$\beta_i \Phi_P = \sum_{j=1}^{6} \lambda_{ij}(c_i^n - c_j^n) \tag{4.183}$$

in order to obtain an N-type scheme. Here again, we find a stability criterion in order to ensure the respect of the maximum principle:

$$\delta t \leq \frac{S_i}{\sum\limits_{\text{neighbouring prisms}} \lambda_{ij}} \tag{4.184}$$

PSI-type MURD scheme

Based on our newly found N-type scheme, we need to construct the PSI-type scheme that will also guarantee that the distribution coefficients fall between 0 and 1. To do this, in the calculation of Φ_P we need to separate the positive and negative contributions:

$$\Phi_P = \Phi_P^+ - \Phi_P^- = \sum_{j=1}^{6} \lambda_{ij} \max(f_i^n - f_j^n, 0) - \sum_{j=1}^{6} \lambda_{ij} \max(f_j^n - f_i^n, 0) \tag{4.185}$$

By comparing Φ_P^+ and Φ_P^-, one can modify the scheme N coefficients:
If $\Phi_P^+ \geq \Phi_P^-$:
    If $f_j^n \geq f_i^n$, then λ_{ij} is cancelled,
    otherwise λ_{ij} is replaced by $\lambda_{ij}\Phi_P/\Phi_P^+$.
If $\Phi_P^+ \leq \Phi_P^-$:
    If $f_j^n \geq f_i^n$, then λ_{ij} is replaced by $-\lambda_{ij}\Phi_P/\Phi_P^-$,

otherwise λ_{ij} is cancelled.
The following is an explanation of these manipulations. In the case where the positive contributions are the most important, we multiply them by Φ_P/Φ_P^+ and we cancel the negative contributions. What was Φ_P^- is thus cancelled, what was Φ_P^+ becomes $\Phi_P^+\Phi_P/\Phi_P^+ = \Phi_P$. The final expression for Φ_P remains unchanged. The case $\Phi_P^+ \leq \Phi_P^-$ can easily be deduced.
Following this modification of the coefficients λ_{ij}, all the terms $\lambda_{ij}(f_i^n - f_j^n)$ are positive and all the terms β_i fall between 0 and 1. Actually:

$$\beta_i = \frac{\sum_{j=1}^{6} \lambda_{ij}(f_i^n - f_j^n)}{\Phi_P} = \frac{\sum_{j=1}^{6} \lambda_{ij}(f_i^n - f_j^n)}{\sum_{l=1}^{6}\sum_{j=1}^{6} \lambda_{ij}(f_l^n - f_j^n)} \tag{4.186}$$

All the terms of the summations are positive and the numerator is smaller than the denominator.

Remarks on mass conservation with the MURD scheme at the discrete level
The MURD scheme is perfectly compatible with the proof of scalar mass conservation that was set out in the Section 4.8.2 and with the continuity step described in the Section 4.6.2. In fact, we saw in this section that:

$$\Delta z w^* = \sum_{j=1}^{npoin2} \left[\Delta z w^*(z^*)\right]_j \Psi_i^h(x,y) \tag{4.187}$$

and $\Delta z w^*$ is necessary here to calculate the coefficients:

$$b_i = \frac{\partial \Psi_i^v}{\partial z^*} \int_{P^*} \Delta z w^* \Psi_i^h \, dP^* \tag{4.188}$$

We thus find that:

$$b_i = \sum_{j=1}^{npoin2} \int_{\Omega_{2D}} \Psi_i^h \Psi_j^h \, d\Omega_{2D} \left\{ \frac{\partial \Psi_i^v}{\partial z^*} \int_{z_{ip}^*}^{z_{ip+1}^*} \left[\Delta z w^*\right]_j dz^* \right\} \tag{4.189}$$

We see that the term in brackets is (ignoring the sign):

$$\frac{1}{(z_{ip}^* - z_{ip+1}^*)} \int_{z_{ip}^*}^{z_{ip+1}^*} \left[\Delta z w^*(z^*)\right]_j \, dz^* \tag{4.190}$$

which we have denoted $\overline{(\Delta z w^*)}_{ip+1/2}^j$. These terms are produced to resolve the continuity equation. If mass lumping has been used when computing $\overline{(\Delta z w^*)}_{ip+1/2}^j$ (see the note at the end of the section 4.6.2), then the coefficients b_i must be computed in a compatible way, which is:

$$b_i = \int_{\Omega_{2D}} \Psi_i^h \, d\Omega_{2D} \left\{ \frac{\partial \Psi_i^v}{\partial z^*} \int_{z_{ip}^*}^{z_{ip+1}^*} \left[\Delta z w^*\right]_i dz^* \right\} \tag{4.191}$$

4.10.3 Streamline Upwind Petrov Galerkin (SUPG) scheme

The use of the method of characteristics on the fixed mesh arising from the sigma transform is not a problem but it may be with other techniques. In order to show the complications arising from the sigma transform, here is an example of advection with the SUPG method. This method consists of upwinding the basis functions in the direction of the flow:

$$\Psi_i' = \Psi_i + \frac{\delta t}{2} \boldsymbol{u} \cdot \boldsymbol{\nabla} \Psi_i \tag{4.192}$$

So, the new basis functions Ψ_i' remain linear but become discontinuous. The variational formulation is applied to the following equation:

$$\frac{c^A - c^n}{\delta t} + \boldsymbol{u}^* \cdot \boldsymbol{\nabla}^* \left(\theta c^A + (1-\theta) c^n\right) = 0 \tag{4.193}$$

where θ is the relaxation coefficient (TETASUPG, which cannot be chosen by the user), $\boldsymbol{u}^*$ the velocity in the fixed mesh and $\boldsymbol{\nabla}^*$ the gradient operator in the same mesh. The earlier equation is multiplied by the deformed basis function:

$$\Psi_i'^* = \Psi_i^* + \frac{\delta t}{2} \boldsymbol{u}^* \cdot \boldsymbol{\nabla} \Psi_i^* \tag{4.194}$$

and by using the Einstein notation $c^A = c_j^A \Psi_j^*$, we get:

$$\frac{c_j^A \Psi_j^* \Psi_i'^*}{\delta t} + \theta \Psi_i'^* \boldsymbol{u}^* \cdot \boldsymbol{\nabla}^* \left(c_j^A \Psi_j^*\right) = \frac{c_j^n \Psi_j^* \Psi_i'^*}{\delta t} + (\theta - 1) \Psi_i'^* \boldsymbol{u}^* \cdot \boldsymbol{\nabla}^* \left(c_j^n \Psi_j^*\right) \tag{4.195}$$

Strictly speaking, the deformed basis function should be used for all the terms in the equation. In practice, only the advection term is transformed, which does not alter the conservation property. After dividing by θ, we get:

$$\begin{aligned} &c_j^A \left(\frac{\Psi_j^* \Psi_i^*}{\theta \delta t} + \Psi_i^* \boldsymbol{u}^* \cdot \boldsymbol{\nabla}^* \left(\Psi_j^*\right) \right) + c_j^A \left(\frac{\delta t}{2} \boldsymbol{u}^* \cdot \boldsymbol{\nabla}^* \left(\Psi_i^*\right) \boldsymbol{u}^* \cdot \boldsymbol{\nabla}^* \left(\Psi_j^*\right) \right) = \\ &c_j^n \left(\frac{\Psi_j^* \Psi_i^*}{\theta \delta t} \right) + c_j^n \left(\frac{\theta - 1}{\theta} \left(\Psi_i^* \boldsymbol{u}^* \cdot \boldsymbol{\nabla}^* \left(\Psi_j^*\right) + \frac{\delta t}{2} \boldsymbol{u}^* \cdot \boldsymbol{\nabla}^* \left(\Psi_i^*\right) \boldsymbol{u}^* \cdot \boldsymbol{\nabla}^* \left(\Psi_j^*\right) \right) \right) \end{aligned} \tag{4.196}$$

By multiplying by h and by integrating on the total fixed domain we finally get:

$$c_j^A \left(\frac{M_{ij}}{\theta \delta t} + VGR_{ij} + UGUG_{ij} \right) = c_j^n \left(\frac{M_{ij}}{\theta \delta t} + \frac{\theta - 1}{\theta} \left(VGR_{ij} + UGUG_{ij}\right) \right) \tag{4.197}$$

with:

$$M_{ij} = \int_{\Omega^*} h \Psi_i^* \Psi_j^* \, d\Omega^* = \int_{\Omega} \Psi_i \Psi_j \, d\Omega \tag{4.198}$$

M, whose general term is M_{ij}, is the mass matrix. It may be estimated either in the real mesh or in the fixed mesh.

$$VGR_{ij} = \int_{\Omega^*} h \varphi_i^* \boldsymbol{u}^* \cdot \boldsymbol{\nabla}^* \left(\varphi_j^*\right) \, d\Omega^* \tag{4.199}$$

VGR, whose general term is VGR_{ij}, is the matrix obtained for classic advection by finite elements expressed in fixed mesh.

$$UGUG_{ij} = \frac{\delta t}{2} \int_{\Omega^*} h \boldsymbol{u}^* \cdot \boldsymbol{\nabla}^* \Psi_i^* \, \boldsymbol{u}^* \cdot \boldsymbol{\nabla}^* \Psi_j^* \, d\Omega^* \tag{4.200}$$

$UGUG$, whose general term is $UGUG_{ij}$, is the additional matrix from SUPG. We show [37] that $UGUG_{ij}$ is equivalent to an artificial diffusion which, in a dimension greater than one, is only produced in the direction of the current. In 2D, the calculation of the earlier matrices is relatively short for triangular elements. For prisms with a triangular base, the number of operations necessary to form the matrices then clearly increases. In order to simplify the calculation of their coefficients and observing that, in theory, it does not harm the conservation property, we can limit $\boldsymbol{u}^*$ in formula 4.194 to the horizontal components of velocity. In the case of essentially

horizontal flow, this approximation is not a problem. Only the generic coefficient of the $UGUG$ matrix is modified in this way. It becomes:

$$UGUG_{ij} = \frac{\delta t}{2} \int_{\Omega^*} h \boldsymbol{u}_h^* \cdot \boldsymbol{\nabla}^* \Psi_i^* \cdot \boldsymbol{u}^* \cdot \boldsymbol{\nabla}^* \Psi_j^* \, d\Omega^* \tag{4.201}$$

with $\boldsymbol{u}_h^* = (u, v, 0)$. When this approximation is made the matrix $UGUG$ is no longer symmetric. When advection of a variable is done with the help of this method, the steps of advection and diffusion are in fact treated at the same time.

4.10.4 The case of settling velocity

We explain here how is done the new advection matrix for the treatment of the settling velocity w_c. The treatment described here corresponds to the keyword ADVECTION-DIFFUSION SCHEME WITH SETTLING VELOCITY = 0, i.e. the settling velocity is taken into account in a fully implicit way in the diffusion step. The former treatment consisted of computing first a centered advection term (call to MATRIX with formula 'MATFGR Z'), and then upwinding (subroutine upwind, which merely adds a diffusion with a coefficient $\Delta z \, w_c/2$, where Δz is the local mesh size on the vertical). Though this technique would give a perfectly implicit upwind scheme in a regular one-dimensional mesh, it is more difficult to prove stability and monotonicity in our 3-dimensional context, with Δz depending on the layer and on the position in the underlying 2D mesh.
Our starting equation is in the continuum:

$$\frac{\partial c}{\partial t} + \mathrm{div}(-\boldsymbol{w}_c c) = 0 \tag{4.202}$$

As w_c is considered positive (version 7.0 on...), we have added the sign minus to recall that $\boldsymbol{w}_c$ is a vector pointing downwards. The equation is solved in the fixed mesh taken at the end of the time step (since this is also done with diffusion). It is important to start from an equation written in a conservative form, because $\boldsymbol{w}_c$ may be variable in space, and has no reason to be divergence free. Moreover it has nothing to see with the ambient flow, which would have to be used to simplify into a non conservative equation. After discretising the derivative in time and variational formulation, our equation gives for every point i:

$$\int_\Omega \Psi_i \frac{c^{n+1} - c^n}{\delta t} \, d\Omega + \int_\Omega \Psi_i \, \mathrm{div}(-\boldsymbol{w}_c c) \, d\Omega = 0 \tag{4.203}$$

Then we integrate by parts the divergence term:

$$\int_\Omega \Psi_i \frac{c^{n+1} - c^n}{\delta t} \, d\Omega - \int_\Gamma \Psi_i C \boldsymbol{w}_c \cdot \boldsymbol{n} \, d\Gamma + \int_\Omega \boldsymbol{w}_c c \boldsymbol{\nabla}(\Psi_i) \, d\Omega = 0 \tag{4.204}$$

The boundary term is treated in the boundary conditions, it is on one side the flux coming from the free surface (i.e. zero) and on the other side the deposition on the bed. The deposition is discretised with an implicit concentration, and with mass-lumping, i.e. with the approximation $\int_\Gamma \Psi_i c \boldsymbol{w}_c \cdot \boldsymbol{n} \, d\Gamma \simeq c_i^{n+1} \int_\Gamma \Psi_i \boldsymbol{w}_c \cdot \boldsymbol{n} \, d\Gamma$. If w_c is constant we see clearly that this term is $-w_c c_i^{n+1} \int_{\Omega_{2D}} \Psi_i \, d\Omega_{2D}$. Ω_{2D} refers to the 2D mesh and is horizontal. $\int_{\Omega_{2D}} \Psi_i \, d\Omega_{2D}$ is the area around point i (equivalent to the finite volume). This term is negative (seen from the suspension it is a loss of sediment), this is why it is treated implicitly. Now let us consider the matrix corresponding to the term $\int_\Omega \Psi_i \, \mathrm{div}(-\boldsymbol{w}_c c) \, d\Omega = 0$: this term represents the internal fluxes that change the concentration of a point (think that $\int_\Omega \Psi_i (c^{n+1} - c^n)/\delta t \, d\Omega$ is the variation of

mass). The matrix will be built element by element, so we restrict our considerations to a single prism. The area associated to points is $SURFAC/3$, where $SURFAC$ is the area of the triangle formed by a projection on the horizontal of upper and lower sides of the prism, or the corresponding triangle in the 2D mesh. Assembling all $SURFAC/3$ of elements around a point gives $\int_{\Omega_{2D}} \Psi_i \, d\Omega_{2D}$, the integral of the 2D basis of this point.

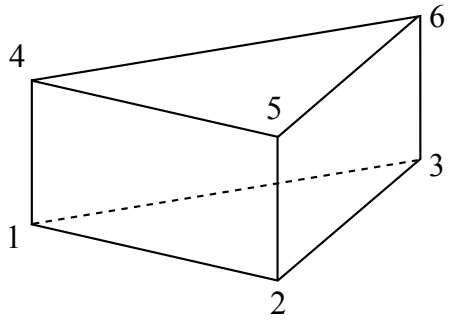

Figure 4.8: Sketch of the prismatic element used in TELEMAC-3D and of the local numbering convention for the nodes.

To have an upwind treatment of advection we decide that:

- point 4 looses a flux of $c^{n+1}(4)w_c(4)SURFAC/3$
- point 5 looses a flux of $c^{n+1}(5)w_c(5)SURFAC/3$
- point 6 looses a flux of $c^{n+1}(6)w_c(6)SURFAC/3$
- point 1 receives from point 4 a flux of $c^{n+1}(4)w_c(4)SURFAC/3$
- point 2 receives from point 5 a flux of $c^{n+1}(5)w_c(5)SURFAC/3$
- point 3 receives from point 6 a flux of $c^{n+1}(6)w_c(6)SURFAC/3$

If we restrict to fluxes into the prism this is all what we see. Of course the upper points also receive from the upper prism and the lower points give to a lower prism, but if we restrict here to what we see in the prism there will be no duplication, and only what receive the points at the free surface and what give the points at the bed will be forgotten, and it is actually the boundary terms which are treated elsewhere.
The six fluxes mentionned above result in 6 non zero terms of the element by element matrix, namely:
$-w_c(4)SURFAC/3$ (coefficient of point 4 in equation of point 1, which is XM(ielem,3))
$-w_c(5)SURFAC/3$ (coefficient of point 5 of equation of point 2, which is XM(ielem,8))
$-w_c(6)SURFAC/3$ (coefficient of point 6 of equation of point 3, which is XM(ielem,12))
$w_c(4)SURFAC/3$ (diagonal of 4, which is T(ielem,4))
$w_c(5)SURFAC/3$ (diagonal of 5, which is T(ielem,5))
$w_c(6)SURFAC/3$ (diagonal of 6, which is T(ielem,6))
This is what is done in subroutine mt15pp.f, then these terms may be assembled to give the final matrix.
The process of points giving a quantity that is received by others is a proof of mass-conservation. Note that the diagonals receive positive numbers and the off-diagonal terms receive negative numbers. The matrix thus built is then a M-matrix and this ensures the positivity. The matrix will remain always positive-definite.

It would be better to consider that the settling velocities are piece-wise constant on the vertical, so that we do not consider the settling velocity of upper points, but an average velocity on the way between upper points and lower points.

4.11 Prescription of the boundary conditions

Section under construction

This section has not been written yet. Actually, it will probably be spreaded over the previous subsections: for each sub-step of the time scheme, specific boundary conditions are prescribed.

4.12 Treatment of dry zones and smashed elements

Section under construction

This section directly comes from the last TELEMAC-3D release notes and has not been reshaped yet.

On dry zones, all elements of the mesh are smashed and have no volume. This can also happen without dry zones, when the user imposes a plane with constant elevation that would go under the bed, it is the first case treated here, before the real tidal flats or dry areas.

4.12.1 Smashed elements

Up to version 5.9, only smashed elements on tidal flats or dry zones were treated. The fact that such elements had no volume precluded the use of finite-volume like numerical schemes, hence only the method of characteristics could be applied for advection. It appeared that most of the treatments done in the case of tidal flats could be applied also in the case of smashed elements with water above. There is indeed a risk of such a situation when the generalised sigma transformation is used, and a constant elevation of planes is prescribed. When the bed goes above the prescribed elevation of a plane, a number of elements is smashed and this caused a crash of computations. To avoid this the parameter DISMIN in subroutine CALCOT guaranteed a minimum height in the elements. From version 6.0 on, it is now possible to have DISMIN=0. To be more precise there is now a DISMIN_BOT and a DISMIN_SUR, respectively for bed and free surface, and DISMIN_BOT may be 0. The key modification for achieving is an array of integers IPBOT, of size NPOIN2 (number of points in the 2D mesh), giving the rank of the last layer with no height. if NPLAN is the number of planes on the vertical, we have for a 2D point I:
IPBOT(I)=0 if all layers are normal (height greater than 1 mm)
IPBOT(I)=3 if plane 3 and 4 are closer than 1 mm and distance between plane 4 and plane 5 is greater than 1 mm.
IPBOT(I)=NPLAN-1 if all the planes are smashed (case of tidal flats). NPLAN is the number of planes.
This new array allowed a number of specific treatments listed below:

- Friction is applied at the level IPBOT(I)+1 and all points below.
- In diffusion all points below the real bed are treated as Dirichlet points, with the previous value as prescribed Dirichlet value. After solving the linear system the points below the real bed are given the value of the real bed.

- The Poisson equation for the non-hydrostatic pressure is treated in the same way as diffusion.

A general principle is that points with the same elevation on a vertical must eventually have the same physical value, so that no artificial infinite gradient is created. 1 mm is an arbitrary but reasonable choice and once it is done all the tests are on IPBOT.

4.12.2 Dry zones

Option 1: correction of free surface gradients

This option first used in 2D has been considered impossible for a long time in 3D, on account of severely distorted elements which appear when, in the absence of water, the free surface and the bed become identical. It appeared later that the work with these severely distorted elements was manageable on condition that some divisions by 0 were avoided. For example, to calculate the vertical average of the horizontal velocity, calculation with the trapezoidal rule should be replaced on the exposed beds by an arithmetic average. Apart from a few numerical problems, for which a remedy exists, the correction of the free surface gradients is formally the most elegant because, when a free surface gradient is treated with a resolution of the Saint-Venant equations, nothing else is left in the actual 3D part of the resolution for both the hydrostatic hypothesis and the non-hydrostatic equations. Plate 9, shown earlier, shows that exposure on a beach can be treated in 3D, in this case with 10 planes along the vertical, with zero water depth on the right side of the calculation domain. In the non-hydrostatic case, we notice a trough in the free surface. This trough becomes more prominent when the flow at the exit of the pool which is emptied, becomes supercritical. The water trapped in the puddle seems to take longer to return to rest. This is because the dynamic pressure gradient is not subject to any specific treatment on the exposed banks.

Option 2: masking of the uncovered elements

The columns of dry or partially dry elements have to be extracted from the calculation domain. This is done with the help of an array equal to 1 or 0, called a mask. The detailed description of this technique is mainly a catalogue of dissuasive difficulties.

Untimely masking–unmasking

In situations at the limit between covering and exposing, with fluxes of water changing signs from one time step to another, the elements balance endlessly between a normal situation and a situation where they are removed from the mesh. In the presence of a bed slope, the flow is guided by the slope and those elements whose bed elevation is the highest will dry first. On the other hand, on a flat bed, the flow is much more erratic and the problem is more acute. It is therefore necessary to mask the entire flat zone when one of the elements of this zone has the criterion for masking. This choice also partially helps in solving the following difficulty.

Occurrence of singular points

The removal of an element does not always give a topologically correct mesh. A point can form an isthmus with zero width where two different coasts touch but without a chance of flux. This type of point should be eliminated by extending the masked zone around the point at one of the two angular sectors not yet masked. This process should be iterated till a correct topology is obtained. A prior analysis of the topography helps to accelerate the process. In exposed zones, the free surface is considered locally quasi-horizontal, and by knowing its elevation at a given instant, all the dry or partially dry elements can be deduced from it. In fact, if an element is masked because the water depth is below a given criterion, all the neighbouring elements with a greater or equal bed elevation should also be masked and so on from one to the next.

Unfortunately, this hypothesis is not valid in zones with high variations in the free surface, but it helps to eliminate the risk of a singular point, if from the start certain regularity is assured in the bathymetry. In practice, it has to be ensured that, by locally changing a bed elevation if necessary, if we turn around an internal point in the domain by going over the elements to which it belongs, we pass over a single minimum and a single maximum of the bed.

Criterion for masking

The third difficulty encountered concerns determining the criterion for masking. There is no criterion for unmasking, since, in the algorithm, we start by unmasking all the elements. The criterion for masking needs the computation of the water depth per element which cannot be less than a given threshold value. This calculation of the height only requires an estimation of the free surface elevation for each element, which is compared with one bed elevation per element. The first idea would be to retain the elevation of the lowest free surface of the three vertices of each triangle in order to be certain of masking all the elements that require masking. However, if the vertex in question is surrounded by masked elements, there is no chance that its elevation will change and the masked elements will remain the same whatever the evolution of the situation around them. To avoid this, or to limit this risk as far as possible, the elevation of the free surface per element is taken as the average of the elevations of the three vertices. Thereupon, almost all the elements become unmasked when the level of the free surface rises, except in some specific places with which we have not yet dealt, such as basins with a single node of mesh at the centre. If some isolated elements have not been correctly unmasked at the time of rise in level of the free surface, we should refine the mesh locally (a very cumbersome solution for a study that is already under way) or resort to a simplification of the bathymetry.

Treatment of masked nodes

Formally, as the technique of masking presented here consists of excluding from the calculation domain those elements that fit the criterion mentioned above, there may be nodes surrounded by masked elements, called masked nodes, for which the equations are no longer resolved. The variables in these nodes (u, v, T, k, ε) are then arbitrarily modified. When unmasking these nodes, these values are once again taken into account in the calculation of the solution. To remedy this shortcoming, we should retain, during the entire duration of the masking of a node, the values of the variables calculated just before this masking. So, it is the increments of the variables and not the variables themselves which are arbitrarily forced to zero.

4.13 Hydrostatic inconsistencies

When a 3D mesh has planes which are not horizontal, spurious horizontal gradients of functions like salinity may appear in the variational formulation. Then a vertical stratification may wrongly be interpreted as unstable. This numerical problem is known as "hydrostatic inconsistencies". Up to version 5.7, there was in TELEMAC-3D a treatment of hydrostatic inconsistencies based on the shape of prismatic elements (key-word HYDROSTATIC INCONSISTENCY FILTER). It consisted in cancelling the salinity (or other scalars like temperature) gradients if the condition:

$$MAX(Z1,Z2,Z3) > MIN(Z4,Z5,Z6) \tag{4.205}$$

appears in an element, ZJ being the elevation of point J. Recall that the local numbering of the points in a prism was given in the figure 4.8. The drawback of this approach is that even in less distorted situations, wrong gradients may appear. What we want in fact is to have horizontal gradients equal to zero if there is a vertical stratification of a given quantity, i.e. if the iso-value surfaces of a given quantity are horizontal, then the gradients of this quantity must be zero. However it is impossible to be sure, only in view of the values at the six nodes of a prism, if

the iso-values are horizontal, the more so if the prism itself has a tilted top and bed. We thus relax the condition by looking only at "topological" possibilities. For example if all nodes at the top of a prism have a salinity greater than 32 g/l and all nodes at the bed less than 30 g/l, we deduce that there is a possibility that the iso-value surface of 31 g/l is horizontal and crosses the element. More generally, if the three verticals of a plane have points with the same salinity (even if their elevation is not exactly the same, which may be due to the linear interpolation), there is a risk of vertical stratification. If FJ denotes the salinity of point J, this ends up in the following test:

$$MIN(MAX(F1,F4),MAX(F2,F5),MAX(F3,F6)) > \tag{4.206}$$

$$MAX(MIN(F1,F4),MIN(F2,F5),MIN(F3,F6)) \tag{4.207}$$

which says in fact that the intersection between the three ranges of values corresponding to the three verticals is not void. This test is done in subroutine VC13PP in BIEF library and is quoted as option 3, which can be asked in subroutine TRISOU in TELEMAC-3D. Another slightly different way of seeing things would be to check that for every of the six points in the prism, its salinity value can be found on the two other verticals if they contain the same elevation. This shows that the iso-value surface could be horizontal. For point 1 and the vertical of points 3 and 6, this would give:

$$\text{if } Z1 > Z3 \text{ and } Z1 < Z6 \text{ then we should have:} \tag{4.208}$$

$$F1 > MIN(F3,F6) \text{ and } F1 < MAX(F3,F6) \tag{4.209}$$

otherwise a stratification is not possible. This gives 12 tests (6 points and two vertical per point), which is heavier than the previous idea. In VC13PP this choice is quoted as option 4.

5. Appendix A: Thompson formulation for radiative open boundary conditions – from *Hydrodynamics of Free-Surface Flows*

The original method [94] uses the theory of characteristics, linearised in a direction normal to the boundary, in the framework of the Saint-Venant equations. Here the depth-averaged velocity is denoted by $\boldsymbol{U}$, with components U and V:

$$\begin{cases} U = \dfrac{1}{h}\displaystyle\int_b^\eta u dz \\ V = \dfrac{1}{h}\displaystyle\int_b^\eta v dz \end{cases} \tag{5.1}$$

A detailed explanation of the original technique:

We explain here in more detail what is said in Reference [36] page 105 to 108. We neglect diffusion and start from the conservative form of Saint-Venant equations, put in the following form taken from Reference [36] at page 31, using the fact that the free surface η is the bottom topography plus the depth h:

$$\begin{cases} \dfrac{\partial h}{\partial t} + \nabla \cdot (h\boldsymbol{U}) = Sce \\ \dfrac{\partial (hU)}{\partial t} + \dfrac{\partial}{\partial x}(hUU + g\dfrac{h^2}{2}) + \dfrac{\partial}{\partial y}(hUV) = -gh\dfrac{\partial b}{\partial x} + hF_x \\ \dfrac{\partial (hV)}{\partial t} + \dfrac{\partial}{\partial x}(hUV) + \dfrac{\partial}{\partial y}(hVV + g\dfrac{h^2}{2}) = -gh\dfrac{\partial b}{\partial y} + hF_y \end{cases} \tag{5.2}$$

Let F, G_x, G_y and $S(F)$ be:

$$F = \begin{pmatrix} h \\ hU \\ hV \end{pmatrix}$$

$$G_x = \begin{pmatrix} hU \\ hU^2 + g\dfrac{h^2}{2} \\ hUV \end{pmatrix} \text{ and } G_y = \begin{pmatrix} hV \\ hUV \\ hV^2 + g\dfrac{h^2}{2} \end{pmatrix}$$

$$S(F) = \begin{pmatrix} Sce \\ -gh\dfrac{\partial b}{\partial x} + hF_x \\ -gh\dfrac{\partial b}{\partial y} + hF_y \end{pmatrix}$$

The Saint-Venant equations (5.2) can then be written in the following form:

$$\frac{\partial F}{\partial t} + \frac{\partial G_x}{\partial x} + \frac{\partial G_y}{\partial y} = S(F)$$

The Thompson method as implemented so far in Telemac consists in considering a local system of coordinates based on a local normal vector $\boldsymbol{n}$ (normal to the boundary) and a local tangent vector $\boldsymbol{t}$. If the new system of coordinates is denoted ξ and ζ, we have:

$$\boldsymbol{n} = \begin{pmatrix} \dfrac{\partial \xi}{\partial x} \\ \dfrac{\partial \xi}{\partial y} \end{pmatrix} \text{ and } \boldsymbol{t} = \begin{pmatrix} \dfrac{\partial \zeta}{\partial x} \\ \dfrac{\partial \zeta}{\partial y} \end{pmatrix} = \begin{pmatrix} -\dfrac{\partial \xi}{\partial y} \\ \dfrac{\partial \xi}{\partial x} \end{pmatrix}$$

We keep these notations here, but the directions $\boldsymbol{n}$ and $\boldsymbol{t}$ may not be linked to the boundary. The components of the velocity in the new system will be denoted by U_ξ and U_ζ. We have:

$$\begin{pmatrix} U_\xi \\ U_\zeta \end{pmatrix} = \begin{pmatrix} \dfrac{\partial \xi}{\partial x} & \dfrac{\partial \xi}{\partial y} \\ \dfrac{\partial \zeta}{\partial x} & \dfrac{\partial \zeta}{\partial y} \end{pmatrix} \begin{pmatrix} U \\ V \end{pmatrix} = \begin{pmatrix} U\dfrac{\partial \xi}{\partial x} + V\dfrac{\partial \xi}{\partial y} \\ -U\dfrac{\partial \xi}{\partial y} + V\dfrac{\partial \xi}{\partial x} \end{pmatrix}$$

and:

$$\begin{pmatrix} U \\ V \end{pmatrix} = \begin{pmatrix} \dfrac{\partial \xi}{\partial x} & -\dfrac{\partial \xi}{\partial y} \\ \dfrac{\partial \xi}{\partial y} & \dfrac{\partial \xi}{\partial x} \end{pmatrix} \begin{pmatrix} U_\xi \\ U_\zeta \end{pmatrix} = \begin{pmatrix} U_\xi\dfrac{\partial \xi}{\partial x} - U_\zeta\dfrac{\partial \xi}{\partial y} \\ U_\xi\dfrac{\partial \xi}{\partial y} + U_\zeta\dfrac{\partial \xi}{\partial x} \end{pmatrix}$$

We first want to put the system in the form:

$$\frac{\partial F}{\partial t} + A_x\frac{\partial F}{\partial x} + B_y\frac{\partial F}{\partial y} = S(F) \tag{5.3}$$

where A_x and B_y are matrices. For this goal:
In Equation 5.2:
$\dfrac{\partial}{\partial x}(g\dfrac{h^2}{2})$ is written $c^2\dfrac{\partial h}{\partial x}$
$\dfrac{\partial}{\partial x}(hUU)$ is written $U^2\dfrac{\partial h}{\partial x} + h\dfrac{\partial U^2}{\partial x} = U^2\dfrac{\partial h}{\partial x} + 2Uh\dfrac{\partial U}{\partial x} = U^2\dfrac{\partial h}{\partial x} + 2U\dfrac{\partial (hU)}{\partial x} - 2U^2\dfrac{\partial h}{\partial x}$
$\dfrac{\partial}{\partial y}(hUV)$ is written $V\dfrac{\partial}{\partial y}(hU) + hU\dfrac{\partial V}{\partial y} = V\dfrac{\partial}{\partial y}(hU) + U\dfrac{\partial (hV)}{\partial y} - UV\dfrac{\partial h}{\partial y}$
$\dfrac{\partial}{\partial y}(g\dfrac{h^2}{2})$ is written $c^2\dfrac{\partial h}{\partial y}$
$\dfrac{\partial}{\partial y}(hVV)$ is written $-V^2\dfrac{\partial h}{\partial y} + 2V\dfrac{\partial (hV)}{\partial y}$
$\dfrac{\partial}{\partial x}(hUV)$ is written $Vd\dfrac{\partial}{\partial x}(hU) + U\dfrac{\partial (hV)}{\partial x} - UV\dfrac{\partial h}{\partial x}$
We effectively get to Equation 5.3 with:

$$A_x = \begin{pmatrix} 0 & 1 & 0 \\ c^2-U^2 & 2U & 0 \\ UV & V & U \end{pmatrix} \text{ and } B_y = \begin{pmatrix} 0 & 0 & 1 \\ -UV & V & U \\ c^2-V^2 & 0 & 2V \end{pmatrix} \tag{5.4}$$

Now we change the coordinates by writing that for every function f we have:

$$\frac{\partial f}{\partial x} = \frac{\partial \xi}{\partial x}\frac{\partial f}{\partial \xi} + \frac{\partial \zeta}{\partial x}\frac{\partial f}{\partial \zeta} \text{ and } \frac{\partial f}{\partial y} = \frac{\partial \xi}{\partial y}\frac{\partial f}{\partial \xi} + \frac{\partial \zeta}{\partial y}\frac{\partial f}{\partial \zeta}$$

It gives us a system in the form:

$$\frac{\partial F}{\partial t} + A_\xi \frac{\partial F}{\partial \xi} + B_\zeta \frac{\partial F}{\partial \zeta} = S(F) \tag{5.5}$$

with:

$$A_\xi = \frac{\partial \xi}{\partial x}A_x + \frac{\partial \xi}{\partial y}B_y \text{ and } B_\zeta = \frac{\partial \zeta}{\partial x}A_x + \frac{\partial \zeta}{\partial y}B_y = -\frac{\partial \xi}{\partial y}A_x + \frac{\partial \xi}{\partial x}B_y$$

which gives:

$$A_\xi = \begin{pmatrix} 0 & \frac{\partial \xi}{\partial x} & \frac{\partial \xi}{\partial y} \\ \frac{\partial \xi}{\partial x}(c^2-U^2) - \frac{\partial \xi}{\partial y}UV & 2U\frac{\partial \xi}{\partial x} + V\frac{\partial \xi}{\partial y} & U\frac{\partial \xi}{\partial y} \\ \frac{\partial \xi}{\partial y}(c^2-V^2) - \frac{\partial \xi}{\partial x}UV & V\frac{\partial \xi}{\partial x} & U\frac{\partial \xi}{\partial x} + 2V\frac{\partial \xi}{\partial y} \end{pmatrix} \tag{5.6}$$

and:

$$B_\zeta = \begin{pmatrix} 0 & -\frac{\partial \xi}{\partial y} & \frac{\partial \xi}{\partial x} \\ -\frac{\partial \xi}{\partial y}(c^2-u^2) - \frac{\partial \xi}{\partial x}UV & -2U\frac{\partial \xi}{\partial y} + V\frac{\partial \xi}{\partial x} & U\frac{\partial \xi}{\partial x} \\ \frac{\partial \xi}{\partial x}(c^2-V^2) + \frac{\partial \xi}{\partial y}UV & -V\frac{\partial \xi}{\partial y} & -U\frac{\partial \xi}{\partial y} + 2V\frac{\partial \xi}{\partial x} \end{pmatrix} \tag{5.7}$$

or even, still denoting U_ξ as the normal component of velocity and U_ζ the tangential component:

$$A_\xi = \begin{pmatrix} 0 & \frac{\partial \xi}{\partial x} & \frac{\partial \xi}{\partial y} \\ \frac{\partial \xi}{\partial x}c^2 - UU_\xi & U\frac{\partial \xi}{\partial x} + U_\xi & U\frac{\partial \xi}{\partial y} \\ \frac{\partial \xi}{\partial y}c^2 - VU_\xi & V\frac{\partial \xi}{\partial x} & U_\xi + V\frac{\partial \xi}{\partial y} \end{pmatrix} \tag{5.8}$$

and:

$$B_\zeta = \begin{pmatrix} 0 & -\frac{\partial \xi}{\partial y} & \frac{\partial \xi}{\partial x} \\ -\frac{\partial \xi}{\partial y}c^2 - UU_\zeta & -U\frac{\partial \xi}{\partial y} + U_\zeta & U\frac{\partial \xi}{\partial x} \\ \frac{\partial \xi}{\partial x}c^2 - UU_\zeta & -V\frac{\partial \xi}{\partial y} & U_\zeta + V\frac{\partial \xi}{\partial x} \end{pmatrix} \tag{5.9}$$

Subsequently, we ignore the variations along the direction ζ and try to solve the system:

$$\frac{\partial F}{\partial t} + A_\xi \frac{\partial F}{\partial \xi} = S(F) \tag{5.10}$$

An open question is: which part of $S(F)$ should be kept in this equation ? We discard *Sce*, F_x and F_y, and keep only the variations of bottom along the direction ξ. It gives

$$S_\xi(F) = \begin{pmatrix} 0 \\ -gh\dfrac{\partial \xi}{\partial x}\dfrac{\partial b}{\partial \xi} \\ -gh\dfrac{\partial \xi}{\partial y}\dfrac{\partial b}{\partial \xi} \end{pmatrix} \tag{5.11}$$

For the time being, we call it $S_\xi(F)$ whatever its value and go on with the diagonalisation of A_ξ. Now A_ξ is diagonalized as $A_\xi = L^{-1}\Lambda L$ with:

$$L = \begin{pmatrix} U_\zeta & \dfrac{\partial \xi}{\partial y} & -\dfrac{\partial \xi}{\partial x} \\ c - U_\xi & \dfrac{\partial \xi}{\partial x} & \dfrac{\partial \xi}{\partial y} \\ c + U_\xi & -\dfrac{\partial \xi}{\partial x} & -\dfrac{\partial \xi}{\partial y} \end{pmatrix}$$

and:

$$\Lambda = \begin{pmatrix} U_\xi & 0 & 0 \\ 0 & U_\xi + c & 0 \\ 0 & 0 & U_\xi - c \end{pmatrix}$$

This can be controlled by checking that $LA_\xi = \Lambda L$.
By stating that $dW = LdF$, we then get back to the diagonalized system:

$$\frac{\partial W}{\partial t} + \Lambda \frac{\partial W}{\partial \xi} = LS_\xi \tag{5.12}$$

each of whose lines is a simple transport equation with source term. Thompson proposes to consider that L is constant in the vicinity of a boundary point, and to write $W = \overline{L}F$, where:

$$\overline{L} = \begin{pmatrix} \overline{U_\zeta} & \dfrac{\partial \xi}{\partial y} & -\dfrac{\partial \xi}{\partial x} \\ \overline{c} - \overline{U_\xi} & \dfrac{\partial \xi}{\partial x} & \dfrac{\partial \xi}{\partial y} \\ \overline{c} + \overline{U_\xi} & -\dfrac{\partial \xi}{\partial x} & -\dfrac{\partial \xi}{\partial y} \end{pmatrix} \tag{5.13}$$

the overbar values being considered as constant (these are the values deduced from the local conditions: h, U and V at the original starting point of the characteristics). The Riemann invariants of the vector W are thus:

- $W_1 = h(\overline{U_\zeta} - U_\zeta)$ (advection with velocity U_ξ).
- $W_2 = h(\overline{c} + U_\xi - \overline{U_\xi})$ (advection with velocity $U_\xi + c$).
- $W_3 = h(\overline{c} - U_\xi + \overline{U_\xi})$ (advection with velocity $U_\xi - c$).

and to which can be added, if a tracer T also has to be considered:

- $W_4 = h(T-\overline{T})$ (advection with velocity U_ξ).

Pure advection is treated with the method of characteristics. To be more precise, a first advection is done with velocity U_ξ. This is done backwards in time. For every boundary point of Thompson type, we compute the backward trajectory and find, at what is called the foot of the characteristic curve (starting point of the trajectory which will arrive at the boundary point after Δt), the values of depth and components of velocity which we call $\widetilde{h}_1$, $\widetilde{U}_1$, $\widetilde{V}_1$ and $\widetilde{T}_1$. If we neglect the source terms and take the invariants at this foot of characteristic pathline, we have:

- $W_1 = h(\overline{U_\zeta} - U_\zeta) = \widetilde{W}_1 = \widetilde{h}_1(\overline{U_\zeta} - \widetilde{U}_{\zeta 1})$ with $\overline{U_\zeta} = -U\dfrac{\partial \xi}{\partial y} + V\dfrac{\partial \xi}{\partial x}$ and $\widetilde{U}_{\zeta 1} = -\widetilde{U}_1\dfrac{\partial \xi}{\partial y} + \widetilde{V}_1\dfrac{\partial \xi}{\partial x}$
- $W_4 = h(T-\overline{T}) = \widetilde{h}(\widetilde{T}_1 - \overline{T})$ with $\overline{T} = T$

then, after an advection with velocity $U_\xi + c$, i.e. with results now called $\widetilde{h}_2$, $\widetilde{U}_2$ and $\widetilde{V}_2$:

- $W_2 = h(\overline{c} + U_\xi - \overline{U_\xi}) = \widetilde{W}_2 = \widetilde{h}_2(\overline{c} + \widetilde{U}_{\xi 2} - \overline{U_\xi})$ with $\overline{c} = \sqrt{gh}$, $\overline{U_\xi} = U\dfrac{\partial \xi}{\partial x} + V\dfrac{\partial \xi}{\partial y}$ and $\widetilde{U}_{\xi 2} = \widetilde{U}_2\dfrac{\partial \xi}{\partial x} + \widetilde{V}_2\dfrac{\partial \xi}{\partial y}$.

then, after an advection with velocity $U_\xi - c$, i.e. with yet other values denoted $\widetilde{h}_3$, $\widetilde{U}_3$ and $\widetilde{V}_3$:

- $W_3 = h(\overline{c} - U_\xi + \overline{U_\xi}) = \widetilde{W}_3 = \widetilde{h}_3(\overline{c} - \widetilde{U}_{\xi 3} + \overline{U_\xi})$ with $\overline{c} = \sqrt{gh}$, $\overline{U_\xi} = U\dfrac{\partial \xi}{\partial x} + V\dfrac{\partial \xi}{\partial y}$ and $\widetilde{U}_{\xi 3} = \widetilde{U}_3\dfrac{\partial \xi}{\partial x} + \widetilde{V}_3\dfrac{\partial \xi}{\partial y}$.

All this is valid only if the backwards characteristic goes inside the domain. This can be checked by the fact that $\boldsymbol{U}_{conv}.\boldsymbol{n} > 0$, where $\boldsymbol{U}_{conv}$ is the advection velocity field (i.e. based on U_ξ, $U_\xi + c$ or $U_\xi - c$, respectively for W_1, W_2 and W_3). If $\boldsymbol{U}_{conv}.\boldsymbol{n} < 0$, all variables with a tilde will be based on the boundary conditions prescribed by the user. For example, $\widetilde{U}_{\zeta 1}$ may be taken equal to $-U_{bor}\dfrac{\partial \xi}{\partial y} + V_{bor}\dfrac{\partial \xi}{\partial x}$, where U_{bor} and V_{bor} are the prescribed components of the velocity field. Source terms will be considered later. Once the Riemann invariants are known, the primitive variables can be restored by the following formulae:

$$h = \frac{W_2 + W_3}{2\overline{c}} \tag{5.14}$$

$$h(U - \overline{U}) = \frac{\partial \xi}{\partial y} W_1 + \frac{\partial \xi}{\partial x}(W_2 - W_3) \tag{5.15}$$

$$h(V - \overline{V}) = \frac{\partial \xi}{\partial y}(W_2 - W_3) - \frac{\partial \xi}{\partial x} W_1 \tag{5.16}$$

$$h(T - \overline{T}) = -W_4 \tag{5.17}$$

Equation 5.14 can be used to eliminate h from the 3 others, it yields:

$$h = \frac{W_2 + W_3}{2\overline{c}} \tag{5.18}$$

$$hU = \frac{W_2+W_3}{2\overline{c}}\overline{U} + \frac{\partial\xi}{\partial y}W_1 + \frac{\partial\xi}{\partial x}(W_2-W_3) \tag{5.19}$$

$$hV = \frac{W_2+W_3}{2\overline{c}}\overline{V} + \frac{\partial\xi}{\partial y}(W_2-W_3) - \frac{\partial\xi}{\partial x}W_1 \tag{5.20}$$

$$hT = \frac{W_2+W_3}{2\overline{c}}\overline{T} - W_4 \tag{5.21}$$

This form is not the most practical but readily gives, if necessary or for checking:

$$\overline{L}^{-1} = \begin{pmatrix} 0 & \frac{1}{2\overline{c}} & \frac{1}{2\overline{c}} \\ \frac{\partial\xi}{\partial y} & \frac{1}{2}\frac{\partial\xi}{\partial x} + \frac{\overline{U}}{2\overline{c}} & -\frac{1}{2}\frac{\partial\xi}{\partial x} + \frac{\overline{U}}{2\overline{c}} \\ -\frac{\partial\xi}{\partial x} & \frac{1}{2}\frac{\partial\xi}{\partial y} + \frac{\overline{V}}{2\overline{c}} & -\frac{1}{2}\frac{\partial\xi}{\partial y} + \frac{\overline{V}}{2\overline{c}} \end{pmatrix} \tag{5.22}$$

We will favour the following formulas for the implementation:

$$h = \frac{W_2+W_3}{2\overline{c}} \tag{5.23}$$

$$U = \frac{\frac{\partial\xi}{\partial y}W_1 + \frac{\partial\xi}{\partial x}(W_2-W_3)}{h} + \overline{U} \tag{5.24}$$

$$V = \frac{-\frac{\partial\xi}{\partial x}W_1 + \frac{\partial\xi}{\partial y}(W_2-W_3)}{h} + \overline{V} \tag{5.25}$$

$$T = -\frac{W_4}{h} + \overline{T} \tag{5.26}$$

If we do not neglect source terms, they have to be integrated along the characteristic curve. Assuming a constant $\overline{L}$ as done before we have:

$$\begin{pmatrix} W_1 \\ W_2 \\ W_3 \end{pmatrix} = \begin{pmatrix} \widetilde{W}_1 \\ \widetilde{W}_2 \\ \widetilde{W}_3 \end{pmatrix} + \Delta t \begin{pmatrix} \overline{U_\zeta}\ Sce + \frac{\partial\xi}{\partial y}(-gh\frac{\partial\xi}{\partial x}\frac{\partial b}{\partial\xi} + hF_x) - \frac{\partial\xi}{\partial x}(-gh\frac{\partial\xi}{\partial y}\frac{\partial b}{\partial\xi} + hF_y) \\ (\overline{c}-\overline{U_\xi})Sce + \frac{\partial\xi}{\partial x}(-gh\frac{\partial\xi}{\partial x}\frac{\partial b}{\partial\xi} + hF_x) + \frac{\partial\xi}{\partial y}(-gh\frac{\partial\xi}{\partial y}\frac{\partial b}{\partial\xi} + hF_y) \\ (c+U_\xi)Sce - \frac{\partial\xi}{\partial x}(-gh\frac{\partial\xi}{\partial x}\frac{\partial b}{\partial\xi} + hF_x) - \frac{\partial\xi}{\partial y}(-gh\frac{\partial\xi}{\partial y}\frac{\partial b}{\partial\xi} + hF_y) \end{pmatrix} \tag{5.27}$$

Neglecting again Sce, F_x and F_y, we are left with:

$$\begin{pmatrix} W_1 \\ W_2 \\ W_3 \end{pmatrix} = \begin{pmatrix} \widetilde{W}_1 \\ \widetilde{W}_2 \\ \widetilde{W}_3 \end{pmatrix} - gh\Delta t \begin{pmatrix} 0 \\ \frac{\partial b}{\partial\xi} \\ -\frac{\partial b}{\partial\xi} \end{pmatrix}$$

Though the source terms could be treated in an explicit way, we do the following approximation: $\frac{\partial b}{\partial\xi}$ is approximated as $\frac{b-\widetilde{Z}_f}{(U+c)\Delta t}$, i.e. the variation of b along the (backwards) characteristic

curve divided by the length of the curve, then U is neglected so that we have $\dfrac{\partial b}{\partial \xi} \simeq \dfrac{b-\widetilde{Z}_f}{c\Delta t}$, and eventually $-gh\Delta t\dfrac{\partial b}{\partial \xi}$ is simplified into $-c(\widetilde{Z}_f - b)$. It gives the following new formulas for W_2 and W_3:

$$W_2 = \overline{c}(\widetilde{h}_2 + \widetilde{Z}_{f2} - b) + \widetilde{h}_2(\widetilde{U}_{\xi 2} - \overline{U_\xi})$$

$$W_3 = \overline{c}(\widetilde{h}_3 + \widetilde{Z}_{f3} - b) + \widetilde{h}_3(-\widetilde{U}_{\xi 3} + \overline{U_\xi})$$

In Thompson publication finite differences are employed to solve the 3 advection problems of the method, this was done mainly because at that time regular grids were common practice. Eric David, at Sogreah, then resorted to the method of characteristics itself to solve these problems on unstructured grids. At that time (1999) it precluded parallelism. Then Jacek Jankowski (BAW Karlsruhe) wrote an amazing parallel version of the method of characteristics (module "streamline" in library BIEF). More recently, module streamline was adapted by Christophe Denis (Sinetics, EDF R&D) for dealing with a list of points that are not necessarily linked to mesh nodes, to enable the treatment of particles on one hand, and Thompson boundary points on the other hand. This was not the end of the story. As a matter of fact, the advection fields requested by Thompson boundary points depend on the starting point, and these specific fields must be defined for the whole domain. In parallel this implies that every Thompson boundary point has to send its advection fields to all processors, in case its characteristic pathlines would go to another sub-domain. This was considered too cumbersome, a dead end. Moreover, the Thompson theory leads to the fact that two nearby boundary points may have their characteristics pathlines crossing, because linearisation was done in two different directions. This is somewhat against the nature of characteristics that do not cross unless they carry the same invariant. For all these reasons it was considered that the theory had to be modified. It seems natural that the linearisation direction should be the direction of the flow. It is what is attempted here.

The new theory:

Linearisation in the direction of the flow:

All what has been said in previous section is valid up to version 6.0 if we choose for $\boldsymbol{n}$ the outward normal vector to the boundary. The problem is that in this case the 3 advections fields depend on the boundary point under treatment. This was heavy in scalar mode, where points with the same normal were grouped for optimization and shared the same advection field. It becomes even more heavy in parallel because these advection fields should be built for the whole domain, which implies that for every Thompson point, its normal vector must be exported to all sub-domains. It also appears very strange that characteristics of the same family stemming from two different points may cross because they have a different original direction.
The new theory consists in choosing advection fields that would not depend on a given boundary point. It seems very natural to choose, instead of the outward normal vector $\boldsymbol{n}$, the direction of the velocity field itself. We have then:

$$\boldsymbol{n} = \begin{pmatrix} \dfrac{\partial \xi}{\partial x} \\ \dfrac{\partial \xi}{\partial y} \end{pmatrix} = \begin{pmatrix} \dfrac{U}{\sqrt{U^2+V^2}} \\ \dfrac{V}{\sqrt{U^2+V^2}} \end{pmatrix} \tag{5.28}$$

An important consequence of this choice is that the velocity $\overline{U}_\zeta$ is always 0 by definition, which would lead to $W_1 = 0$. This is true in fact only if we consider that the direction $\boldsymbol{n}$ changes

along characteristics, it is false if we keep the original $\boldsymbol{n}$, which would be consistent with the linearisation leading to $\overline{L}$. Tests show that it is better to consider that U_ζ is indeed not 0, thus sticking to the linearisation. A possibility that remains to be tested would be considering that U_ζ is indeed 0, and taking the norm of velocity for the component U_ξ.
In any case there is an obvious problem when there is no velocity, the direction where to apply the celerity c is then undefined. A first idea is to cancel also the celerity c in this case, so that all variables will keep their original value. This is not possible, because a velocity equal to 0 for a given boundary point would then trigger that the depth and velocity at this point remain unchanged. This is valid only if there is no wave approaching the point, i.e. no velocity and no free surface slope. When there is a free surface slope, it seems then natural to choose the direction of the vector $-g\,\boldsymbol{grad}(\eta)$, which is the driving term in momentum equation that will create velocity at the next time step. This happens to be very important in tests, especially the gaussian hill test case.
Depth and velocity fields for interpolation:

An unexpected problem occured in the results, showing that the tests $\boldsymbol{U}_{conv}.\boldsymbol{n} > 0$, to decide whether we should take e.g. the depth h or the prescribed depth h_{bor} for computing $\widetilde{h}$, could happen to be wrong. As a matter of fact the method of characteristics itself is able to check if the pathline goes out of the domain, and in this case it stops and interpolates at this exit point. In a corner the average $\boldsymbol{n}$ of the corner point may lead to a different decision, thus leading to wrongly choose for example h instead of h_{bor}. Any case where the value $\boldsymbol{U}_{conv}.\boldsymbol{n}$ is very close to 0 will lead to a random choice, and then to large differences if h and h_{bor} are very different. It was thus decided to discard the tests $\boldsymbol{U}_{conv}.\boldsymbol{n} > 0$ and to use interpolation fields of h, U and V that already contain the prescribed boundary conditions. A characteristic pathline that exits a Thompson boundary will thus find naturally that $\widetilde{h} = h_{bor}$, without resorting to testing $\boldsymbol{U}_{conv}.\boldsymbol{n}$. A drawback is that for small Courant numbers, when the characteristics pathlines will not go far from boundaries, their interpolated values will be influenced by the prescribed values of the boundary. When prescribed values are correct, which is generally the case with box models and measurements, this could be also an advantage. With this new approach there can be no discontinuity of choice due to a truncation error.

6. Appendix B: Calculation of the buoyancy source terms in the transformed mesh

In the transformed mesh, we have:

$$F_x = g\frac{\Delta\rho}{\rho_o}\left(\frac{\partial\eta}{\partial x}\right)_{y,t} - g\left[\frac{\partial}{\partial x}\left(\int_z^{\eta}\frac{\Delta\rho}{\rho_o}dz\right)_{y,z^*,t} + \left(\frac{\partial z}{\partial x}\right)_{y,z^*,t}\frac{\Delta\rho}{\rho_o}\right] \tag{6.1}$$

The variable $zs = z - \eta$ is introduced; F_x is then simplified as:

$$F_x = -g\left[\frac{\partial}{\partial x}\left(\int_{zs}^{0}\frac{\Delta\rho}{\rho_o}dz\right)_{y,z^*,t} + \left(\frac{\partial zs}{\partial x}\right)_{y,z^*,t}\frac{\Delta\rho}{\rho_o}\right] \tag{6.2}$$

At this stage of reasoning, it is worth pursuing the calculation of F_x at the discrete level because the expression formulated above can be further simplified. The variation of this term will be calculated between two nodes of the mesh denoted as "inf" and "sup", one located just above the other, with the knowledge that on the surface F_x is zero:

$$\begin{aligned} F_x^{\inf} - F_x^{\sup} &= -g\left[\frac{\partial}{\partial x}\left(\int_{zs^{\inf}}^{zs^{\sup}}\frac{\Delta\rho}{\rho_o}dz\right)_{y,z^*,t}\right] \\ &- g\left[\left(\frac{\partial zs^{\inf}}{\partial x}\right)_{y,z^*,t}\frac{\Delta\rho^{\inf}}{\rho_o} - \left(\frac{\partial zs^{\sup}}{\partial x}\right)_{y,z^*,t}\frac{\Delta\rho^{\sup}}{\rho_o}\right] \end{aligned} \tag{6.3}$$

With linear functions along z, this gives:

$$\begin{aligned} F_x^{\inf} - F_x^{\sup} &= -g\frac{\partial}{\partial x}\left[\frac{1}{2}\left(zs^{\sup} - zs^{\inf}\right)\left(\frac{\Delta\rho^{\sup}}{\rho_o} + \frac{\Delta\rho^{\inf}}{\rho_o}\right)\right]_{y,z^*,t} \\ &- g\left[\left(\frac{\partial zs^{\inf}}{\partial x}\right)_{y,z^*,t}\frac{\Delta\rho^{\inf}}{\rho_o} - \left(\frac{\partial zs^{\sup}}{\partial x}\right)_{y,z^*,t}\frac{\Delta\rho^{\sup}}{\rho_o}\right] \end{aligned} \tag{6.4}$$

and we finally get:

$$\begin{aligned} F_x^{\inf} - F_x^{\sup} &= \frac{g}{2}\left[\left(\frac{\Delta\rho^{\sup}}{\rho_o} - \frac{\Delta\rho^{\inf}}{\rho_o}\right)\frac{\partial}{\partial x}\left(zs^{\sup} + zs^{\inf}\right)_{y,z^*,t}\right] \\ &- \frac{g}{2}\left[\left(zs^{\sup} - zs^{\inf}\right)\frac{\partial}{\partial x}\left(\frac{\Delta\rho^{\sup}}{\rho_o} + \frac{\Delta\rho^{\inf}}{\rho_o}\right)_{y,z^*,t}\right] \end{aligned} \tag{6.5}$$

The two derivatives along x are discontinuous functions at the nodes of the mesh, and are calculated as follows:

- For every node M, we can define the set $ELEM(M)$ of elements of the mesh having the node M among their vertices.
- For each of the elements i of $ELEM(M)$ we can calculate the $\left(\frac{\partial f}{\partial x}\right)_i$ derivative of f at M on the side of the element i.

The derivative of f at M is then estimated by:

$$\left(\frac{\partial f}{\partial x}\right)(M) = \frac{\sum\limits_{i \in ELEM(M)} \int\limits_{\Omega} \left(\frac{\partial f}{\partial x}\right)_i \Psi_M d\Omega}{\int\limits_{\Omega} \Psi_M d\Omega} \tag{6.6}$$

where Ψ_M is the basis function associated to node M.

[1] R. Abgrall. Toward the Ultimate Conservative Scheme: Following the Quest. *Journal of Computational Physics*, 167(2):277–315, March 2001. ISSN 00219991. doi: 10.1006/jcph.2000.6672. URL http://linkinghub.elsevier.com/retrieve/pii/S0021999100966725.

[2] R. Abgrall. Essentially non-oscillatory Residual Distribution schemes for hyperbolic problems. *Journal of Computational Physics*, 214(2):773–808, May 2006. ISSN 00219991. doi: 10.1016/j.jcp.2005.10.034. URL http://linkinghub.elsevier.com/retrieve/pii/S0021999105004730.

[3] Rémi Abgrall and Timothy Barth. Residual distribution schemes for conservation laws via adaptive quadrature. *SIAM Journal on Scientific Computing*, 24 (3):732–769, 2003. URL http://epubs.siam.org/doi/abs/10.1137/S106482750138592X.

[4] Rémi Abgrall and Mohamed Mezine. Construction of second-order accurate monotone and stable residual distribution schemes for steady problems. *Journal of Computational Physics*, 195(2):474–507, April 2004. ISSN 00219991. doi: 10.1016/j.jcp.2003.09.022. URL http://linkinghub.elsevier.com/retrieve/pii/S0021999103005400.

[5] Rémi Abgrall and Philip L. Roe. High order fluctuation schemes on triangular meshes. *Journal of Scientific Computing*, 19(1-3):3–36, 2003. URL http://link.springer.com/article/10.1023/A:1025335421202.

[6] R. Aris. *Vectors, Tensors and the Basic Equations of Fluid Mechanics*. Prentice-Hall Inc.: Engelwood Cliffs, NJ, 1962.

[7] L. Arpaia, M. Ricchiuto, and R. Abgrall. An ALE Formulation for Explicit Runge–Kutta Residual Distribution. *Journal of Scientific Computing*, 63(2):502–547, May 2015. ISSN 0885-7474, 1573-7691. doi: 10.1007/s10915-014-9910-5. URL http://link.springer.com/10.1007/s10915-014-9910-5.

[8] Tennessee Valley Authority. *Heat and mass transfer between a water surface and the atmosphere*. Water Resources Research Report No 0-68003, 1972.

[9] M.E. Berliand and T.G. Berliand. Measurement of the effective radiation of the earth with varying cloud amounts. *Izv. Akad. Nauk SSSR, Ser. Geofiz.*, 1, 1952.

[10] T.G. Berliand. Metodika kilamatologicheskikh naschetov summarnoi radiatsii. *Meteor. Hydrol.*, 6:9–12, 1960.

[11] M. Bijvelds. Numerical modelling of estuarine flow over steep topography. *Communications on Hydraulic and Geotechnical Engineering, TU Delft Faculty of Civil Engineering Report*, 01-2, March 2001.

[12] G.L. Bodhaine. *Measurement of peak discharge at culverts by indirect methods*, chapter A3, book 3. 1968.

[13] J. Bredberg. On the wall boundary condition for turbulence models. Technical report, Chalmers University of Technology, Göteborg, Sweden, 2000.

[14] M. Carlier. *Hydraulique générale et appliquée*. Paris, Eyrolles, 1972.

[15] I. Celik and W. Rodi. Simulation of free-surface effects in turbulent channel flows. *Physicochemical Hydrodynamics*, 4:217–227, 1984.

[16] R. K. Chan. A generalized arbitrary lagrangian-eulerian method for incompressible flows with sharp interfaces. *Journal of Computational Physics*, 17(3):311–331, 1975. URL www.scopus.com. Cited By :28.

[17] A.J. Chorin. Numerical solution of the Navier–Stokes equations. *Mathematics of Computation*, 22:745–762, 1968.

[18] P.G. Ciarlet. *The finite element method for elliptic problems*. North-Holland, 1978.

[19] Árpád Csík, Mario Ricchiuto, and Herman Deconinck. A Conservative Formulation of the Multidimensional Upwind Residual Distribution Schemes for General Nonlinear Conservation Laws. *Journal of Computational Physics*, 179(1):286–312, June 2002. ISSN 00219991. doi: 10.1006/jcph.2002.7057. URL http://linkinghub.elsevier.com/retrieve/pii/S0021999102970579.

[20] G. Datt and G. Touzot. *Une présentation de la méthode des éléments finis*. Collection Université de Compiègne, Presses de l'Université Laval, 1981.

[21] A. Daubert and O. Graffe. Quelques aspects des écoulements presque horizontaux à 2 dimensions en plan et non permanents. *La Houille Blanche*, 22:847–860, 1967.

[22] A. Decoene and J. F. Gerbeau. Sigma transformation and ale formulation for three-dimensional free surface flows. *International Journal for Numerical Methods in Fluids*, 59(4):357–386, 2009. URL www.scopus.com. Cited By :7.

[23] Astrid Decoene. *Hydrostatic model for three-dimensional free surface flows and numerical schemes*. PhD thesis, Université Pierre et Marie Curie - Paris VI ; Laboratoire Jacques-Louis Lions, 2006.

[24] Herman Deconinck and Mario Ricchiuto. Residual Distribution Schemes: Foundations and Analysis. In Erwin Stein, René de Borst, and Thomas J. R. Hughes, editors, *Encyclopedia of Computational Mechanics*. John Wiley & Sons, Ltd, Chichester, UK, October 2007. ISBN 0-470-84699-2 978-0-470-84699-5 0-470-09135-5 978-0-470-09135-7. URL http://doi.wiley.com/10.1002/0470091355.ecm054.

[25] Laboratoire d'études en Géophysique et Océanographie Spatiales, Accessed: 2016-04-05. http://www.legos.obs-mip.fr/recherches/equipes/ecola/outils-produits.

[26] J. Donea, P. Fasoli Stella, S. Giuliani, J. P. Halleux, and A. V. Jones. Arbitrary lagrangian eulerian finite element procedure for transient dynamic fluid-structure interaction problems. *Transactions of the International Conference on Structural Mechanics in Reactor Technology*, B(4), 1981. URL `www.scopus.com`.

[27] EDF R&D. Code_Saturne 3.0.0 Theory Guide, Accessed: 2013. `http://code-saturne.org/cms/sites/default/files/theory-3.0.pdf`.

[28] G.D. Egbert and L. Erofeeva. Osu tidal data inversion, Accessed: 2016-04-05. `http://volkov.oce.orst.edu/tides/`.

[29] A. Ern and J.-L. Guermond. *Theory and Practice of Finite Elements*. Applied Mathematical Sciences, Volume 159. Springer, 2004.

[30] H.B. Fischer. *Mixing in inland and coastal waters*. Academic Press, 1979.

[31] R.A. Flather. *Results from surge prediction model of the North-West European continental shelf for April, November and December 1973*. Institute of Oceanography (UK), Report # 24., 1976.

[32] V. Girault and P.A. Raviart. *Finite element methods for the Navier–Stokes equations*. Springer-Verlag, 1986.

[33] J. L. Guermond, P. Minev, and Jie Shen. An overview of projection methods for incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 195(44-47): 6011–6045, 2006.

[34] V. Guimet and D. Laurence. A linearised turbulent production in the k-ε model for engineering applications. In *Proc. Vth International Symposium on Engineering Turbulence Modelling and Measurements*, pages 157–166, 2002. Majorqua (Spain).

[35] B. Henderson-Sellers. Calculating the surface energy balance for lake and reservoir modeling: A review. *Reviews of Geophysics*, 24(3):625–649, 1986.

[36] J.-M. Hervouet. *Hydrodynamics of free surface flows, modelling with the finite element method*. John Wiley & Sons Ltd., 2007.

[37] J.-M. Hervouet and C. Moulin. Nouveaux schémas de convection dans telemac-2D version 2.3: apport de la méthode SUPG. Technical report, EDF, 1993. HE-43/93.27.

[38] J.-M. Hervouet and C.-T. Pham. Telemac version 6.1, release notes. Telemac-2D and Telemac-3D. Technical report, EDF R&D, LNHE, 2011.

[39] C. W. Hirt, A. A. Amsden, and J. L. Cook. An arbitrary lagrangian-eulerian computing method for all flow speeds. *Journal of Computational Physics*, 14(3):227–253, 1974. URL `www.scopus.com`. Cited By :1362.

[40] J.C.R. Hunt. *Turbulence structure and turbulent diffusion near gas-liquid interfaces*, pages 67–82. Reidel Pub., 1984. eds. W. Brutsaert & G.H. Jirka.

[41] A. Imerito. *Dynamic Reservoir Simulation Model DYRESM v4. V4.0 Science Manual*. Rapport du Centre for Water Research, University of Western Australia, 2007.

[42] J.-M. Janin. Conservativité et positivité dans un module de transport de scalaire écrit en éléments finis - application à telemac-3D. Technical report, EDF, 1995. HE-42/95/054/A.

[43] J.-M. Janin and X. Blanchard. *Simulation des courants de marée en Manche et Proche Atlantique*. EDF DER&LNH report HE-42/92.58, 1992.

[44] N.G. Jerlov. *Optical oceanography*. Ed. Elsevier, 1968.

[45] A. N. Kolmogorov. The local structure of turbulence in incompressible viscous fluid for very large Reynolds numbers. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 434(1890):9–13, 1991.

[46] E.B. Kraus. *Atmosphere - Ocean Interaction*. Oxford University Press, 1972.

[47] D. Kuzmin, O. Mierka, and S. Turek. *On the Implementation of the k-epsilon Turbulence Model in Incompressible Flow Solvers Based on a Finite Element Discretization*. Ergebnisberichte angewandte Mathematik. Tech. University of Dortmund, 2007.

[48] E. B. Larock, W. R. Jeppson, and Z. G. Watters. *Hydraulics of Pipeline Systems*. CRC Press, 2000.

[49] B.E. Launder and D.B. Spalding. The numerical computation of turbulent flows. *Computer Methods in Applied Mechanics and Engineering*, 3, 1974.

[50] A. Lencastre. *Manuel d'hydraulique générale*. Paris, Eyrolles, 1961.

[51] A. Leroy. *A New Incompressible SPH Model: Towards Industrial Applications*. PhD thesis, Université Paris-Est, Paris, France, 2014.

[52] P.-L. Lions. *Mathematical Topics in Fluid Mechanics. Volume 1: Incompressible Models*. Clarendon Press, Oxford Lecture Series in Mathematics and Its Applications 3, 1996.

[53] A. Mahadevan, J. Oliger, and R. Street. A non-hydrostatic meso-scale ocean model. *Journal of Physical Oceanography*, 26(9):1881–1900, 1996.

[54] G.L. Mellor and T. Yamada. A hierarchy of turbulence closure models for planetary boundary layers. *Journal of Atmospheric Sciences*, 31:1791–1806, 1974.

[55] F. R. Menter. Two-equation eddy-viscosity turbulence models for engineering applications. *AIAA Journal*, 32(8):1598–1605, 1994.

[56] I. Nezu and H. Nakagawa. *Turbulence in open-channel flows*. Balkema, 1993.

[57] R.-H. Ni. A multiple grid scheme for solving the Euler equation. *AIAA Journal*, (20): 1565–1571, 1981.

[58] Henri Paillère. *Multidimensional Upwind Residual Distribution Schemes for the Euler and Navier-Stokes Equations on Unstructured Grids*. PhD thesis, Université Libre de Bruxelles, 1995.

[59] I.L. Pairaud, F. Lyard, F. Auclair, T. Letellier, and P. Marsaleix. Dynamics of the semi-diurnal and fourth-diurnal internal tides in the bay of biscay. part 1: Barotropic tides. *Continental Shelf Research*, 28(1011):1294–1315, 2008.

[60] I.L. Pairaud, F. Auclair, P. Marsaleix, F. Lyard, and A. Pichon. Dynamics of the semi-diurnal and fourth-diurnal internal tides in the bay of biscay. part 2: Baroclinic tides. *Continental Shelf Research*, 30(3-4):253–269, 2010.

[61] C.A. Paulson and J.J. Simpson. Irradiance measurements in the upper ocean. *Journal of Physical Oceanography*, 7:952–956, 1977.

[62] Sara Pavan. *New advection schemes for free-surface flows*. PhD thesis, Université Paris-Est, 2016.

[63] R.E. Payne. Albedo of the sea surface. *Journal of Atmospheric Sciences*, 29:959–970, 1972.

[64] C. Perrin de Brichambaut. *Rayonnement solaire et échanges radiatifs naturels*. Ed. Gauthier-Villars, 1963.

[65] C. Perrin de Brichambaut. *Estimation des ressources énergétiques solaires en France*. Supplément aux cahiers A.F.E.D.E.S. numéro 1, 1975.

[66] C.-T. Pham and F. Lyard. Use of tidal harmonic constants databases to force open boundary conditions in TELEMAC. In 19th *TELEMAC-MASCARET User Conference*, Oxford, UK, 2012.

[67] C.-T. Pham, S. Bourban, N. Durand, and M. Turnbull. *Méthodologie pour la simulation de la marée en Manche et proche Atlantique avec TELEMAC-2D et TELEMAC-3D*. EDF R&D-LNHE report H-P74-2012-02534-FR, 2013.

[68] O. Pironneau. *Méthode des éléments finis pour les fluides*. Masson, 1988.

[69] L. Prandtl. Über die ausgebidete turbulenz. *Zeitschrift für augewandte Mathematik und Mechanik*, 5(136), 1925.

[70] D.T. Pugh. *Tides, Surges and Mean Sea-Level*. John Wiley & Sons, 1987, reprinted in 1996.

[71] P.Wang, A. Joly, and A. Leroy. Towards the simulation of submerged bottom structures with vertical walls using Telemac-3D. In *23rd TELEMAC-MASCARET User Conference*, 2016. Paris, France.

[72] B. Quetin. Modèles mathématiques de calcul des écoulements induits par le vent. In *17ième congrès de l'AIRH*, Baden-Baden, 15–19 Août, 1977.

[73] M. Ramette. *Guide d'hydraulique fluviale*. EDF Report-LNHE-40/81.04., 1981.

[74] R. Rannacher. On chorin's projection method for the incompressible navier-stokes equations. In *The Navier-Stokes Equations II — Theory and Numerical Methods*, volume 1530 of *Lecture Notes in Mathematics*, pages 167–183. Springer Berlin Heidelberg, 1992.

[75] P.A. Raviart and J.M. Thomas. *Introduction à l'analyse numérique des équations aux dérivées partielles*. Masson, 1983.

[76] M. Ricchiuto and R. Abgrall. Explicit Runge–Kutta residual distribution schemes for time dependent problems: Second order case. *Journal of Computational Physics*, 229(16):5653–5691, August 2010. ISSN 00219991. doi: 10.1016/j.jcp.2010.04.002. URL `http://linkinghub.elsevier.com/retrieve/pii/S0021999110001786`.

[77] M. Ricchiuto, R. Abgrall, and H. Deconinck. Application of conservative residual distribution schemes to the solution of the shallow water equations on unstructured meshes. *Journal of Computational Physics*, 222(1):287–331, March 2007. ISSN 00219991. doi: 10.1016/j.jcp.2006.06.024. URL http://linkinghub.elsevier.com/retrieve/pii/S0021999106002853.

[78] Mario Ricchiuto. *Contributions to the development of residual discretizations for hyperbolic conservation laws with application to shallow water flows*. PhD thesis, Université Sciences et Technologies-Bordeaux I, 2011. URL http://tel.archives-ouvertes.fr/tel-00651688/.

[79] Mario Ricchiuto. An explicit residual based approach for shallow water flows. *Journal of Computational Physics*, 280:306–344, January 2015. ISSN 00219991. doi: 10.1016/j.jcp.2014.09.027. URL http://linkinghub.elsevier.com/retrieve/pii/S0021999114006639.

[80] Mario Ricchiuto and Andreas Bollermann. Stabilized residual distribution for shallow water simulations. *Journal of Computational Physics*, 228(4):1071–1115, 2009. URL http://www.sciencedirect.com/science/article/pii/S0021999108005391.

[81] Mario Ricchiuto, Árpád Csík, and Herman Deconinck. Residual distribution for general time-dependent conservation laws. *Journal of Computational Physics*, 209(1):249–289, October 2005. ISSN 00219991. doi: 10.1016/j.jcp.2005.03.003. URL http://linkinghub.elsevier.com/retrieve/pii/S002199910500118X.

[82] W. Rodi. *Turbulence models and their applications in hydraulics. A state of the art review*. Edition AIRH, 1984.

[83] P. L. Roe. Linear advection schemes on triangular meshes. Technical report coa 8720, Cranfield Institute of Technology, 1987.

[84] P. L. Roe and D. Sidilkover. Optimum Positive Linear Schemes for Advection in Two and Three Dimensions. *SIAM Journal on Numerical Analysis*, 29(6):1542–1568, December 1992. ISSN 0036-1429, 1095-7170. doi: 10.1137/0729089. URL http://epubs.siam.org/doi/abs/10.1137/0729089.

[85] M.J. Salencon and J.M. Thébaud. *Modélisation d'écosystème lacustre*. Ed. Masson, 1997. ISBN 2-225-85627-3.

[86] P. Schureman. *Manual of harmonic analysis and prediction of tides*. U.S. Coast and Geodetic Survey, 1924, reprinted in 1971.

[87] J. Smagorinski. General circulation experiments with the primitive equations. i. the basic experiment. *Monthly Weather Review*, 91:99–164, 1963.

[88] J.-C. Soliva. *Modèle tridimensionnel d'écoulements méso-météorologiques. Etude de la convection-diffusion d'un polluant passif à cette échelle*. PhD thesis, Thèse ENPC, 1982.

[89] Stefan Spekreijse. Multigrid solution of monotone second-order discretizations of hyperbolic conservation laws. *Mathematics of Computation*, 49(179):135–135, September 1987. ISSN 0025-5718. doi: 10.1090/S0025-5718-1987-0890258-9. URL http://www.ams.org/jourcgi/jour-getitem?pii=S0025-5718-1987-0890258-9.

[90] R. Struijs. *A Multi-Dimensional Upwind Discretization Method for the Euler Equations on Unstructured Grids*. PhD thesis, University of Delft, Netherlands, 1994.

[91] H.E. Sweers. Monograms to estimate the heat-exchange coefficient at the air-water interface as a function of wind speed and temperature; a critical survey of some literature. *Journal of Hydrology*, 30:375–401, 1976.

[92] M. J. Teles, S. Smolders, T. Maximova, I. Rocabado, and J. Vanlede. Numerical modelling of flood control areas with controlled reduced tide. scheldt estuary physics and integrated management. In *Proc. of the 36th IAHR World Congress*, pages 90–99, 2015.

[93] R. Temam. Une méthode d'approximation des solutions des équations de Navier–Stokes. *Bulletin de la Société Mathématique de France*, 98:115–152, 1968. (in French).

[94] K.W. Thompson. Time dependent boundary conditions for hyperbolic systems. *Journal of Computational physics*, 68:1–24, 1987.

[95] I. Tsanis. Simulation of wind-induced water currents. *Journal of Hydraulic Engineering*, 115:1113–1134, 1989.

[96] P.-L. Viollet. *Mécanique des fluides à masse volumique variable*. Presses de l'Ecole Nationale des Ponts et Chaussées, 1997. (in French).

[97] P.-L. Viollet, J.-P. Chabard, P. Esposito, and D. Laurence. *Mécanique des fluides appliquée*. Presses de l'Ecole Nationale des Ponts et Chaussées, 2002. (in French).

[98] T.L. Wahl. *Trash control structures and equipment: a literature review and survey of bureau of reclamation experience*. United States Department of the Interior, Bureau of Reclamation, Hydraulics Branch, Research and Laboratory Services Division, Denver, Colorado, 1992.

[99] Andrzej Warzyński, Matthew E. Hubbard, and Mario Ricchiuto. Runge–Kutta Residual Distribution Schemes. *Journal of Scientific Computing*, 62(3):772–802, March 2015. ISSN 0885-7474, 1573-7691. doi: 10.1007/s10915-014-9879-0. URL `http://link.springer.com/10.1007/s10915-014-9879-0`.

[100] C. J. Yap. *Turbulent heat and momentum transfer in recirculating and impringing flows*. PhD thesis, University of Manchester, Manchester, UK, 1987.