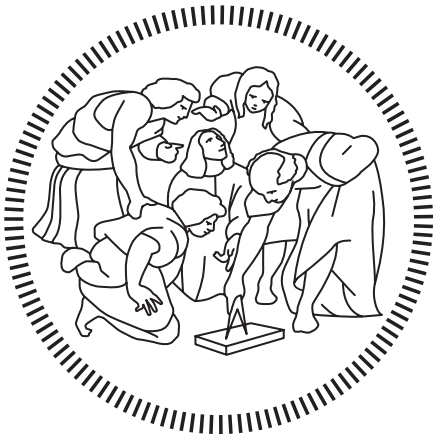
POLITECNICO DI MILANO



SCHOOL OF INDUSTRIAL AND INFORMATION ENGINEERING
Master of Science – Aeronautical Engineering

A multilayer SPH formulation for free surface flows

Supervisor
Prof. Paolo Francesco BARBANTE

Candidate
Luca MACAVERO – 920267

Academic Year 2019 – 2020

Abstract

In the present thesis, a Smoothed Particle Hydrodynamics (SPH) modeling of the first order layer-averaged (multi-layer) Navier-Stokes equation in the hydrostatic approximation is presented. The multi-layer approach allows to recover the velocity profile of the flow, this being impossible in classical one-layer formulation such as the Saint-Venant model. The mathematical formulation is first detailed. Its SPH implementation is then described, including specific procedures to deal with open boundary conditions, which are not straightforward to be implemented in the SPH framework; furthermore, some terms of the governing equations required to be partly treated in an Eulerian manner and their treatment is discussed. A dedicated code has been developed and a number of validation tests have been finally performed against both analytical and experimental results; its capability to recover reference solutions is demonstrated.

Sommario

In questa tesi, le equazioni di Navier-Stokes mediate in spessore (multi-strato) vengono modellate tramite il metodo Smoothed Particle Hydrodynamics (SPH), nell'ipotesi di approssimazione idrostatica. L'approccio multi-strato, a differenza di quello classico monostrato di Saint-Venant, permette di ricavare il profilo di velocità del flusso. In primo luogo viene presentata la formulazione matematica del problema. Viene poi descritta la sua implementazione tramite il metodo SPH, che include la possibilità di usare condizioni al contorno aperte, la cui trattazione non è banale; inoltre, la formulazione multi-strato necessita un trattamento euleriano per alcuni termini delle equazioni di governo e tale approccio viene discusso. Un codice dedicato è stato sviluppato per validare i risultati numerici, confrontandoli sia con alcuni risultati analitici, sia con risultati sperimentali; viene quindi dimostrato che la formulazione proposta approssima correttamente le soluzioni di riferimento.

Ringraziamenti

Ringrazio il professor Barbante per avermi guidato con grande disponibilità durante il lavoro nonostante l'impossibilità nel vederci; grazie per i numerosi e preziosi consigli e per la infinita pazienza nel rispondere ai miei dubbi.

Grazie a mamma e papà per avermi sempre sostenuto in tutti questi anni: non sarei dove sono senza di voi.

Grazie a Michi per tutti i momenti passati insieme e per tutta la felicità che hai portato nella mia vita.

Grazie agli amici con cui ho passato le mie serate: Quaglio, Fede, Jaco, Carlo, Mike, Marti, Tecla, Bea, Fede e Olto. Grazie agli amici che mi hanno reso le giornate di lezioni più leggere: Gio, Marco, Dave, Ale e Beppe.

Contents

List of Figures

List of Tables

Introduction

Several examples of free surface flows can be found in physics and in engineering: rivers, oceans, channels, continental ice sheet flows, avalanche flows and many others are of great interest in hydrology, oceanography, geology and climate modeling and are characterized by a free surface; free surface flows, such as the tear-film flow, can be found in biophysics; furthermore, free surface flows have many technological applications, from chemical engineering (heat exchangers, evaporators, photobioreactors) to microfluidics and nanofluidics, from coating technologies to aerospace processes (de-icing of aircraft wings); other examples are the thin film flows in atomizers used in liquid fuel injection, fire suppression and air conditioning.

The fluid motion is described by the Navier-Stokes equations, coupled with the continuity and the energy equations, and in order to treat them numerically, a wide amplitude of numerical models have been proposed: for example in the case of free surface flows in the shallow water approximation, the Saint-Venant model with hydrostatic pressure is employed. The Saint-Venant model is efficient in terms of numerical solution, since it reduces the dimensions of the problem. A 3D problem treated with Saint-Venant equations becomes a 2D problem, and this is a clear advantage in terms of computational time and complexity of the algorithm. One issue related to this model is that its range of application is questionable in presence of large friction coefficients, of significant water depth, when wind plays an important role and, in general, when the flow exhibits an appreciable velocity profile; in all these situations, the velocity profile is wrongly approximated by a constant velocity over all the depth.

In Ref. [1], the author propose a modification of the Saint-Venant model, which is able to overcome these issues, based on a multi-layer formulation of the shallow water equations. The derivation of the model consists in layer-averaging the Navier-Stokes equations in the approximation of hydrostatic pressure. A slight modification of the multi-layer model is used in order to overcome the loss of hyperbolicity that occurs in some situations, and to discretise it in the finite volume framework. In this model every layer is advected by the flow in such a way that mass exchange between adjacent layers is not allowed. By allowing mass exchange between layers it is possible to resolve flows in which also the vertical component of the velocity is strong (e.g. Ref. [3], [14]).

The multi-layer approach can describe typical applications of free surface flows: for example, Ref. [13] employed the same model for solving granular column collapse with $\mu(I)$ rheology, Ref. [5] used it for understanding the dark/light cycle in raceway ponds for the cultivation of oleaginous microalgae; several papers are dedicated to hydrodynamical applications, for example Ref. [6] is devoted to the simulation of hydrodynamical processes in both the Strait

of Gibraltar and the Alboran Sea using the multi-layer Saint-Venant equations. A multi-layer approach can be generalized also to generic flows: for example it is used in Ref. [16] for solving blood flow in large arteries.

The multi-layer Saint-Venant equations are usually solved using FVM or lattice Boltzmann method. These are grid based methods. In this thesis, the smoothed particle hydrodynamics (SPH) method is employed: it is a meshless, particle method, whose development was started in the '70s by Monaghan [17] and Lucy [25] for astrophysical purposes. The SPH is nowadays used in engineering, even if it has intrinsic problems related to its Lagrangian nature, for example the difficulty on the treatment of open boundary conditions. However, we show how it gives good results both in the case of analytical and experimental comparison. To the author's best knowledge the only previous application of the SPH method to multi-layer Saint-Venant system is found in Ref. [35]. However the proposed method is limited to the two layer Saint-Venant system, whereas our formulation can handle an arbitrary number of layers.

In this thesis we use a model similar to the one in Ref. [1], in which the possibility of mass exchange between layers is not covered: the mathematical formulation is described in Chapter 1.

In order to solve the equations, a particular form of the SPH method is used, based on the works of Vila [42] and de Leffe et al. [8]. The former proposed a technique for solving systems of conservation laws with SPH, the latter employed Vila's approach for solving the one layer Saint-Venant equations. We generalize the approach to the case of several layers in Chapter 2.

A code has been implemented, which is called MultiLayerSPH and is written in Fortran 2003; its structure is described in Chapter 3. The code is uploaded in the `github` repository `https://github.com/lu-maca/MultiLayersSPH`.

Finally, in Chapter 4 the code is validate by comparison against several analytical results and four experimental data, where it is shown how the method is able to recover good solutions in terms of accuracy in both cases. In particular, introducing a simple model of turbulence, it is able to recover the velocity profile in accordance with experimental measurements.

1. Mathematical formulation

The motion of a fluid respects the Navier-Stokes equations, together with the continuity and energy equations. In their conservative form they read:

$$\begin{cases} \partial_t \rho + \boldsymbol{\nabla} \cdot (\rho \mathbf{u}) = 0 \\ \partial_t (\rho \mathbf{u}) + \boldsymbol{\nabla} \cdot (\rho \mathbf{u} \otimes \mathbf{u}) = -\boldsymbol{\nabla} p + \boldsymbol{\nabla} \cdot \boldsymbol{\tau} + \rho \mathbf{f} \\ \partial_t (\rho E^t) + \boldsymbol{\nabla} \cdot (\rho E^t \mathbf{u}) = -\boldsymbol{\nabla} \cdot (p \mathbf{u}) + \boldsymbol{\nabla} \cdot (\mathbf{T} \cdot \mathbf{u}) + \rho \mathbf{f} \cdot \mathbf{u} - \boldsymbol{\nabla} \cdot \mathbf{q} \end{cases} \tag{1.1}$$

where $\mathbf{u} = \mathbf{u}(\mathbf{x}, t)$ is the velocity field, $p = p(\mathbf{x}, t)$ is the pressure field, $\rho = \rho(\mathbf{x}, t)$ is the density field, $E^t = e_{int} + \frac{1}{2}\mathbf{u}^2$ is the total energy and $\boldsymbol{\tau}$ is a viscous stress tensor; $\mathbf{q} = \mathbf{q}(\mathbf{x}, t))$ is the heat vector; $\mathbf{f}$ is a volume force. They must be paired with equation of state and constitutive law, boundary conditions and initial conditions.

From Equation (1.1), the multilayer formulation can be derived.

1.1 Layer-averaged Navier-Stokes equations

Consider a free surface flow and divide the thickness into N layers, represented schematically in Figure 1.1 in which the nomenclature proposed by Ref. [3] is used. In the following we will

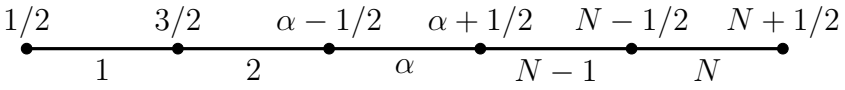


Figure 1.1: Schematic representation of the multi-layer framework.

derive the multi-layer equations in the general case; it is convenient to define the operator

$$\boldsymbol{\nabla}_{xy} = (\partial_x, \partial_y, 0)$$

and to introduce the nomenclature of Ref. [1]. We define ϕ_α and h_α as the average quantity ϕ (see Equation (1.2)) and the thickness of the α-th layer; $\phi_{\alpha+1/2}$ and $\phi_{\alpha-1/2}$ as the property ϕ respectively at the upper and lower interface of the α-th layer; finally, $z_{\alpha+1/2}$ and $z_{\alpha-1/2}$ are the coordinates of the interfaces $\alpha \pm 1/2$. The average of the quantity ϕ is defined on the α-th layer as:

$$\phi_\alpha = \frac{1}{h_\alpha} \int_{z_{\alpha-1/2}}^{z_{\alpha+1/2}} \phi \, \mathrm{d}z \tag{1.2}$$

Note that here the bottom is assumed impermeable and still, and at the free surface mass exchange is not allowed: this means we will only deal with mono-phase fluids.

Continuity equation. Consider the continuity equation and integrate it along the z direction between two generic depths, $z_{\alpha-\frac{1}{2}}$ and $z_{\alpha+\frac{1}{2}}$:

$$\int_{z_{\alpha-1/2}}^{z_{\alpha+1/2}} \partial_t \rho \, \mathrm{d}z + \int_{z_{\alpha-1/2}}^{z_{\alpha+1/2}} \boldsymbol{\nabla} \cdot (\rho \mathbf{u}) \, \mathrm{d}z = 0$$

Using the definition of $\boldsymbol{\nabla}_{xy}$:

$$\int_{z_{\alpha-1/2}}^{z_{\alpha+1/2}} \partial_t \rho \, \mathrm{d}z + \int_{z_{\alpha-1/2}}^{z_{\alpha+1/2}} \boldsymbol{\nabla}_{xy} \cdot (\rho \mathbf{u}_{xy}) \, \mathrm{d}z + \int_{z_{\alpha-1/2}}^{z_{\alpha+1/2}} \partial_z (\rho w) \, \mathrm{d}z = 0$$

Applying the Leibniz theorem[1], the kinematic boundary condition for an interface (which states that the interface between two layers is a material line), and the definition of average quantity given in Equation (1.2), the previous equation can be written as:

$$\partial_t (\rho_\alpha h_\alpha) + \boldsymbol{\nabla}_{xy} \cdot (\rho_\alpha \mathbf{u}_\alpha h_\alpha) = 0 \tag{1.3}$$

which is valid for every layer.

Momentum equation. The same procedure can be applied for the momentum equation. Here the normal vector[2] to the generic interface $\mathbf{n}_{\alpha\pm1/2}$ is used:

$$\mathbf{n}_{\alpha\pm1/2} = \left(-\partial_x z_{\alpha\pm1/2}, -\partial_y z_{\alpha\pm1/2}, 1\right)$$

The momentum equation is then:

$$\begin{aligned}\partial_t (\rho_\alpha \mathbf{u}_\alpha h_\alpha) + \boldsymbol{\nabla}_{xy} \cdot (\rho_\alpha \mathbf{u}_\alpha \otimes \mathbf{u}_\alpha h_\alpha) = &- \boldsymbol{\nabla}_{xy} \cdot [(p_\alpha \mathbf{I} + \boldsymbol{\tau}_\alpha) h_\alpha] + \mathbf{n}_{\alpha+1/2} \cdot \left(-p_{\alpha+1/2}\mathbf{I} + \boldsymbol{\tau}_{\alpha+1/2}\right) + \\ &- \mathbf{n}_{\alpha-1/2} \cdot \left(-p_{\alpha-1/2}\mathbf{I} + \boldsymbol{\tau}_{\alpha-1/2}\right) + \rho_\alpha \mathbf{f} h_\alpha\end{aligned} \tag{1.4}$$

where $\mathbf{I}$ is the identity matrix; this equation is valid in every layer.

[1] The Leibniz theorem states that

$$\int_{\alpha(x)}^{\beta(x)} \partial_x f(x, x') \, \mathrm{d}x' = \partial_x \int_{\alpha(x)}^{\beta(x)} f(x, x') \, \mathrm{d}x' + f(x, \alpha) \, \partial_x \alpha - f(x, \beta) \, \partial_x \beta$$

[2] It is not normalized with respect to its length.

Energy equation. In the same way, the depth-integrated energy equation reads:

$$\begin{aligned}\partial_t\left(\rho_\alpha E_\alpha^t h_\alpha\right)+\boldsymbol{\nabla}_{xy}\cdot\left(\rho_\alpha \mathbf{u}_\alpha E_\alpha^t h_\alpha\right) &= -\boldsymbol{\nabla}_{xy}\cdot\left(p_\alpha \mathbf{u}_\alpha h_\alpha\right)+\boldsymbol{\nabla}_{xy}\cdot\left(\boldsymbol{\tau}_\alpha\cdot\mathbf{u}_\alpha\right)+\\ &-\boldsymbol{\nabla}_{xy}\left(\mathbf{q}_\alpha h_\alpha\right)+\mathbf{n}_{\alpha+1/2}\cdot\left[\left(-p_{\alpha+1/2}\mathbf{I}\cdot\mathbf{u}_{\alpha+1/2}+\boldsymbol{\tau}_{\alpha+1/2}\cdot\mathbf{u}_{\alpha+1/2}\right)-\mathbf{q}_{\alpha+1/2}\right]+\\ &-\mathbf{n}_{\alpha-1/2}\cdot\left[\left(-p_{\alpha-1/2}\mathbf{I}\cdot\mathbf{u}_{\alpha-1/2}+\boldsymbol{\tau}_{\alpha-1/2}\cdot\mathbf{u}_{\alpha-1/2}\right)-\mathbf{q}_{\alpha-1/2}\right]+\rho_\alpha\mathbf{f}_\alpha\cdot\mathbf{u}_\alpha h_\alpha\end{aligned} \tag{1.5}$$

Equations (1.3), (1.4) and (1.5) are the depth-integrated Navier-Stokes equations; note that a new unknown has been introduced, namely h_α. Resolving these equation would be computationally expensive. Furthermore, the system needs a closure relation, since the balance between unknowns and equations is not respected due to the new unknown. One possibility is to impose the incompressibility of the flow, which is physically a good approximation in many cases. As explained in Ref. [4], if one supposes that the spatial distributions of $\mathbf{u}$ and other flow quantities are characterized by a length scale L, and that the variations of $|\mathbf{u}|$ with respect both to time and space have the magnitude U, then the velocity field can be supposed to be solenoidal if

$$|\boldsymbol{\nabla}\cdot\mathbf{u}| \ll \frac{U}{L} \quad \text{i.e. if} \quad \left|\frac{1}{\rho}\frac{D\rho}{Dt}\right| \ll \frac{U}{L}$$

If ρ and the entropy per unit mass S are chosen as independent state parameters, so that $p=p(\rho,S)$, one can differentiate:

$$\frac{Dp}{Dt}=\left(\frac{\partial p}{\partial\rho}\right)_S\frac{D\rho}{Dt}+\left(\frac{\partial p}{\partial S}\right)_\rho\frac{DS}{Dt}$$

Solving the relation for $\frac{D\rho}{Dt}$ and knowing from thermodynamics that $\left(\frac{\partial p}{\partial\rho}\right)_S=c^2$, where c is the speed of sound in the material, we obtain:

$$\left|\frac{1}{\rho c^2}\frac{Dp}{Dt}-\frac{1}{\rho c^2}\left(\frac{\partial p}{\partial S}\right)_\rho\frac{DS}{Dt}\right| \ll \frac{U}{L} \tag{1.6}$$

Condition (1.6) is satisfied if each of the two terms are small compared to U/L. In particular if the first term is much less than U/L, the fluid behaves as if it were incompressible, because the change in density of a material element due to pressure variations would be negligible. Let us consider the first term:

$$\left|\frac{1}{\rho c^2}\frac{Dp}{Dt}\right| \ll \frac{U}{L} \tag{1.7}$$

Note that the momentum equation for a Newtonian fluid, assuming small the effects of viscosity and heat conductivity, can be written as

$$\rho\frac{D\mathbf{u}}{Dt}=-\boldsymbol{\nabla}p+\rho\mathbf{f} \tag{1.8}$$

Then, multiplying Equation (1.8) by $\mathbf{u}$ we obtain:

$$\frac{\rho}{2}\frac{D|\mathbf{u}|^2}{Dt}=-\mathbf{u}\cdot\boldsymbol{\nabla}p+\rho\mathbf{u}\cdot\mathbf{f}$$

from which, by using the Lagrangian derivative definition for the pressure, the following holds:

$$\frac{Dp}{dt} = -\frac{\rho}{2}\frac{D|\mathbf{u}|^2}{Dt} + \frac{\partial p}{\partial t} + \rho \mathbf{u} \cdot \mathbf{f} \tag{1.9}$$

Assuming the flow to be isentropic $\left(\frac{DS}{Dt} \approx 0\right)$, and substituting Equation (1.9) in Equation (1.7), the latter becomes (see Ref. [4, Chapt. 3.6]):

$$\left| \frac{1}{\rho c^2}\frac{\partial p}{\partial t} - \frac{1}{2c^2}\frac{D|\mathbf{u}|^2}{Dt} + \frac{\mathbf{u}\cdot\mathbf{f}}{c^2} \right| \ll \frac{U}{L}$$

The order of magnitude of l.h.s. terms can be studied:

1. $\frac{D|\mathbf{u}|^2}{Dt}$ is of order of magnitude U^3/L, so that for the second term in the l.h.s. the following must be valid:
$$\frac{U^2}{c^2} = \mathrm{Ma}^2 \ll 1$$
and one can expect that this condition is well satisfied for free surface flows arising in nature;

2. the first term is linked to unsteadiness of the flow: if for example the flow is oscillatory and f is a measure of the dominant frequency, the velocity fluctuates with a period f^{-1}. The order of magnitude of pressure is $\rho L^2 f^2$ and the condition for incompressibility becomes
$$\frac{f^2 L^2}{c^2} \ll 1$$
which is the same condition above if $f \sim U/L$, but is more restrictive if $f \gg U/L$. This term is important in gas dynamics, while in the case under study it can be neglected: in free surface flows, $f \sim U/L$;

3. the third term comes from body forces: body forces are due to gravity or electromagnetic effects, which are important, for example, when studying thin films (see Ref. [7]). Considering only gravity effects:
$$\frac{\mathbf{u}\cdot\mathbf{f}}{c^2} \sim \frac{Ug}{c^2} \ll \frac{U}{L}$$
which is valid if $gL/c^2 \ll 1$. This condition can be not satisfied in some fields like dynamical meteorology, while for liquid as water it is fulfilled, being $g \sim 10\,\mathrm{m\,s^{-1}}$ and $c^2 \sim 10^6\ \mathrm{m^2/s^2}$.

Considering now the second term in (1.6), Batchelor [4] proved that it is tipically small compared to U/L.

In conclusion, if the fluid is Newtonian, the flow field can be considered solenoidal under our assumptions; thus energy equation will be uncoupled from continuity and momentum equations. Furthermore, when a first order approximation of the Navier-Stokes equations is considered, as will be done in next sections, pressure is linear along the depth (hydrostatic

approximation); in this case the ratio between density variation to the the initial density, induced by pressure variation, is of the order of gH/c^2, which is much less than one, when a depth $H \leq 10^2 - 10^3$ m is considered, depending on the fluid properties (namely his speed of sound c); so, the dependence of density on pressure can be neglected. On the other hand, temperature variations is much more important, since it is proportional to the thermal expansion coefficient β_T which is of the order of 10^{-3} K^{-1} for the more common fluids. In case of a thin film flow on a heated wall, for example, in which the ΔT at the wall can reach $50 - 100$ K, the contribution to the density variation given by temperature gradient can be significant.

Being the proposed method a novelty, we prefer to consider the most simple case, without temperature gradients, so that $\mathbf{q} = \mathbf{0}$; also, the density will be considered constant and equal in each layer. Note however that the model is valid also if ρ_α is constant in the α-th layer, but different in every layer. These assumptions allow to compare numerical results to analytical ones. Thus, the flow field is solenoidal and this condition will allow to close the balance between equations and unknowns. Also, the fluid will be supposed to be Newtonian and since temperature is supposed to be constant, the viscosity will be constant too; $\boldsymbol{\tau}$ will be given by:

$$\boldsymbol{\tau} = \mu \left(\boldsymbol{\nabla} \mathbf{u} + \boldsymbol{\nabla} \mathbf{u}^{\mathrm{T}} \right)$$

With these hypotesis in mind, it is convenient to derive an approximated formulation of Equations (1.3) and (1.4), which is called the Multi-layer Saint-Venant model.

1.2 Multi-layer Saint-Venant model

In the following, the geometry shown in Figure 1.2 will be considered, and this will be the general setup for every problem to be studied in this thesis. The first interface, called $z_{1/2}$ represents the bottom topography. As already pointed out, Equations (1.3) and (1.4) are

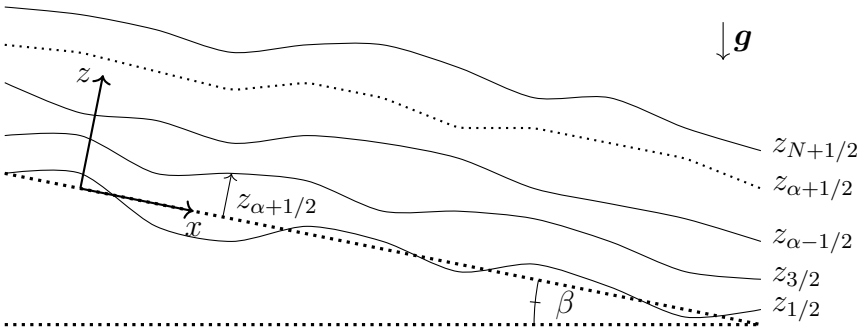


Figure 1.2: Multi-layer setup.

decoupled from the energy equation: therefore there are four equations for each layer (continuity equation and three scalar momentum equations). Under the assumptions depicted in the previous section ($\rho =$ constant and $\mu =$ constant, so that in each layer, $\rho_\alpha = \rho$), the number of unknowns is five for each layer: three components of $\mathbf{u}_\alpha$, h_α and p_α. Note that

the thickness of each layer is given by:

$$h_\alpha = z_{\alpha+1/2} - z_{\alpha-1/2}$$

and the total depth is:

$$h = \sum_{\alpha=1}^{N} h_\alpha \tag{1.10}$$

Finally, β is the angle of inclination of the plane formed by x and y with respect to the horizontal.

Several physical processes are characterized by the presence of a dimension much more important than the others: the long ocean or tidal wave, the falling of a fluid down an inclined surface and the boundary layer around an airfoil are examples of long wave phenomena. For describing mathematically this kind of phenomena, it is natural to introduce a small parameter $\epsilon = \frac{H}{L} \ll 1$, where H is a characteristic depth and L is the characteristic wavelength of disturbances to the free surface. This assumption is called the *shallow water assumption*; we introduce the following nondimensional variables:

$$h = H\tilde{h}, \quad z = H\tilde{z}, \quad (x, y) = L\left(\tilde{x}, \tilde{y}\right), \quad u = U\tilde{u},$$

$$v = U\tilde{v}, \quad w = W\tilde{w}, \quad t = \frac{L}{U}\tilde{t}, \quad p = \rho U^2 \tilde{p}$$

where $\tilde{(\cdot)}$ is the nondimensional variable and capital letters refer to characteristic quantities of the flow. It is then possible to make equations (1.3) and (1.4) nondimensional. The Saint-Venant approximation consists in neglecting all the terms of order greater than ϵ^n, where $n = 0, 1, 2, \ldots$. In the present case n is set equal to 1 and this is a first order approximation of the layer-averaged Navier-Stokes equations, for which, as will be shown later, the pressure is approximated as hydrostatic, i.e. linear along the depth.

Continuity equation. Equation (1.3), recalling that ρ is constant in space and time, can be rewritten as:

$$\partial_t h_\alpha + \boldsymbol{\nabla}_{xy} \cdot (\mathbf{u}_\alpha h_\alpha) = 0$$

which, in nondimensional form is:

$$\frac{HU}{L}\partial_{\tilde{t}} \tilde{h}_\alpha + \frac{HU}{L}\boldsymbol{\nabla}_{\tilde{x}\tilde{y}} \cdot \left(\tilde{\mathbf{u}}_\alpha \tilde{h}_\alpha\right) = 0$$

Finally, the nondimensional continuity equation reads exactly as the dimensional one:

$$\partial_{\tilde{t}} \tilde{h}_\alpha + \boldsymbol{\nabla}_{\tilde{x}\tilde{y}} \cdot \left(\tilde{\mathbf{u}}_\alpha \tilde{h}_\alpha\right) = 0 \tag{1.11}$$

Furthermore, from $\boldsymbol{\nabla} \cdot \mathbf{u} = 0$, one can find that $W = \epsilon U$.

Momentum equations. It is convenient to study momentum equations in x, y separately from momentum equation in z. Furthermore, the "planar" directions, namely the ones that lie onto the plane which forms an angle β with the horizontal, are treated in the same exact way. Thus only the x-momentum will be nondimensionalized, and the y-momentum will follow; Equation (1.4) becomes in the x component:

$$\begin{aligned}\partial_t(u_\alpha h_\alpha) + \partial_x\left(u_\alpha^2 h_\alpha\right) + \partial_y\left(u_\alpha v_\alpha h_\alpha\right) + \partial_x\left(p_\alpha h_\alpha\right) = &\left(\partial_x z_{\alpha+1/2} p_{\alpha+1/2} - \partial_x z_{\alpha-1/2} p_{\alpha-1/2}\right) + \\ &- 2\nu\left(\partial_x z_{\alpha+1/2}\partial_x u_{\alpha+1/2} - \partial_x z_{\alpha-1/2}\partial_x u_{\alpha-1/2}\right) + \\ &- \nu\left[\partial_y z_{\alpha+1/2}\left(\partial_y u + \partial_x v\right)_{\alpha+1/2} - \partial_y z_{\alpha-1/2}\left(\partial_y u + \partial_x v\right)_{\alpha-1/2}\right] + \\ &+ \nu\left(\partial_z u_{\alpha+1/2} - \partial_z u_{\alpha-1/2}\right) + \nu\left(\partial_x w_{\alpha+1/2} - \partial_x w_{\alpha-1/2}\right) + \\ &+ g\sin\beta h_\alpha + 2\nu\partial_x\left(h_\alpha\partial_x u_\alpha\right) + \nu\partial_y\left[\left(h_\alpha\partial_y u + \partial_x v\right)_\alpha\right]\end{aligned}$$

where $\nu = \frac{\mu}{\rho}$. Putting them in nondimensional form and considering one further assumption, that is that the Reynolds number computed with L as reference length is of order ϵ^{-1} or higher

$$\mathrm{Re}_L = \frac{UL}{\nu} \sim \frac{1}{\epsilon} \text{ or } \frac{1}{\epsilon^2} \tag{1.12}$$

one can find the first order approximation of the layer-averaged Navier-Stokes equation in the x and y component by neglecting terms of order greater than ϵ. This assumption is realistic in situations typical in geophysics, meteorology and in the physics of thin films. Proceeding with nondimensionalization:

$$\begin{aligned}\epsilon\left[\partial_{\tilde t}(\tilde u_\alpha\tilde h_\alpha) + \partial_{\tilde x}\left(\tilde u_\alpha^2\tilde h_\alpha\right) + \partial_{\tilde y}\left(\tilde u_\alpha\tilde v_\alpha\tilde h_\alpha\right) + \partial_{\tilde x}\left(\tilde p_\alpha\tilde h_\alpha\right)\right] = &\ \epsilon\left(\partial_{\tilde x}\tilde z_{\alpha+1/2}\tilde p_{\alpha+1/2} - \partial_{\tilde x}\tilde z_{\alpha-1/2}\tilde p_{\alpha-1/2}\right) + \\ &- \frac{\epsilon}{\mathrm{Re}_L}\Big\{2\left(\partial_{\tilde x}\tilde z_{\alpha+1/2}\partial_{\tilde x}\tilde u_{\alpha+1/2} - \partial_{\tilde x}\tilde z_{\alpha-1/2}\partial_{\tilde x}\tilde u_{\alpha-1/2}\right) + \\ &+ \left[\partial_{\tilde y}\tilde z_{\alpha+1/2}\left(\partial_{\tilde y}\tilde u + \partial_{\tilde x}\tilde v\right)_{\alpha+1/2} - \partial_{\tilde y}\tilde z_{\alpha-1/2}\left(\partial_{\tilde y}\tilde u + \partial_{\tilde x}\tilde v\right)_{\alpha-1/2}\right] + \\ &+ \left(\partial_{\tilde x}\tilde w_{\alpha+1/2} - \partial_{\tilde x}\tilde w_{\alpha-1/2}\right) + 2\partial_{\tilde x}\left(\tilde h_\alpha\partial_{\tilde x}\tilde u_\alpha\right) + \partial_{\tilde y}\left[\left(\tilde h_\alpha\partial_{\tilde y}\tilde u + \partial_{\tilde x}\tilde v\right)_\alpha\right]\Big\} + \\ &+ \frac{1}{\mathrm{Fr}^2}\sin\beta\tilde h_\alpha + \frac{1}{\epsilon\mathrm{Re}_L}\left(\partial_{\tilde z}\tilde u_{\alpha+1/2} - \partial_{\tilde z}\tilde u_{\alpha-1/2}\right)\end{aligned}$$

where the Froude number is defined as

$$\mathrm{Fr} = \sqrt{\frac{U^2}{gH}}$$

and has the same meaning of Mach number in gas dynamics. For example, flows at a supercritical Froude (i.e. $\mathrm{Fr} > 1$) can develop hydraulic jumps, which are the analogous of shock waves.

Using assumption (1.12), all terms multiplied by ϵ/Re_L are of order ϵ^2 or higher, while all the others are of order ≤ 1. The body force term

$$\frac{1}{\mathrm{Fr}^2}\sin\beta\tilde h_\alpha \tag{1.13}$$

depends on the order of the Froude number. One of the available analytical results for the full Navier-Stokes equations for incompressible flows is the so called Nusselt flow, which is a semiparabolical velocity profile, given by a gravity-driven flow on an inclined plate. The exact solution, valid in the case of laminar flow, is:

$$u(z) = \frac{g}{2\nu}(2h - z)z \sin\beta \tag{1.14}$$

and its mean value is:

$$u_{ave} = \frac{g}{3\nu} \sin\beta h^2 \tag{1.15}$$

Supposing that $U = u_{ave}$, the Froude number is:

$$\mathrm{Fr} = \sqrt{\frac{\epsilon \mathrm{Re}_L}{3} \sin\beta} \tag{1.16}$$

In our assumption the term (1.13) becomes:

$$\frac{3}{\epsilon \mathrm{Re}_L} \tilde{h}_\alpha \tag{1.17}$$

which is of order 1 if $\epsilon \mathrm{Re}_L \sim 1$, and of order ϵ if $\epsilon \mathrm{Re}_L \sim \epsilon^{-1}$.

Then, in the dimensional form and at the first order in ϵ, the x-momentum (and the y-momentum identically) is:

$$\begin{aligned}\partial_t \left(h_\alpha u_\alpha\right) + \partial_x \left(h_\alpha u_\alpha^2\right) + \partial_y \left(h_\alpha u_\alpha v_\alpha\right) + \frac{1}{\rho}\partial_x \left(h_\alpha p_\alpha\right) = &\frac{1}{\rho}\left[p_{\alpha+1/2}\partial_x\left(z_{\alpha+1/2}\right) - p_{\alpha-1/2}\partial_x\left(z_{\alpha-1/2}\right)\right] + \\ &+ \nu\left[\partial_z u_{\alpha+1/2} - \partial_z u_{\alpha-1/2}\right] + g\sin\beta h_\alpha\end{aligned} \tag{1.18}$$

It is now possible to study the nondimensional z-momentum and with the same arguments that have been used above, one can note that all the acceleration terms are of second order, and a relation for pressure can be found. This relation can be written in the following way:

$$p_{\alpha-1/2} = p_{\alpha+1/2} + \rho g h_\alpha + \epsilon\left(...\right) + \epsilon^2\left(...\right) + ...$$

It is possible to neglect terms of order ϵ, since pressure is a first order term in the x and y momentum equations: thus those terms would become of second order and they would be neglected anyhow. So the z-momentum assures that in the shallow water approximation, the pressure is hydrostatic.

The energy equation will not be studied, since it has been assumed that density is constant in space and time and changes in temperature are negligible.

In order to make the numerical treatment of the problem simpler and to be able to perform a study of the proposed numerical method, it has been chosen to consider the one dimensional problem that corresponds to two-dimensional non-integrated Navier-Stokes

equations. The equations that will be used are the following, namely the one dimensional multi-layer equations at order one with hydrostatic pressure:

$$\begin{cases} \partial_t h_\alpha + \partial_x(h_\alpha u_\alpha) = 0 \\ \\ \partial_t \left(h_\alpha u_\alpha\right) + \partial_x \left(h_\alpha u_\alpha^2\right) + \frac{1}{\rho}\partial_x \left(h_\alpha p_\alpha\right) = \frac{1}{\rho}\left[p_{\alpha+1/2}\partial_x\left(z_{\alpha+1/2}\right) - p_{\alpha-1/2}\partial_x\left(z_{\alpha-1/2}\right)\right] + \\ \qquad\qquad +\nu\left[\partial_z u_{\alpha+1/2} - \partial_z u_{\alpha-1/2}\right] + g\sin\beta h_\alpha \\ p_{\alpha-1/2} = p_{\alpha+1/2} + \rho g h_\alpha \cos\beta \end{cases} \tag{1.19}$$

$$\begin{cases} u_{1/2} = 0 \\ \partial_z u_{N+1/2} = 0 \\ p_{N+1/2} = p_{\text{interf}} \end{cases} \tag{1.20}$$

and must be valid for $\alpha = 1, \ldots, N$. In (1.20) are reported the boundary conditions at the bottom and the stress condition at the free surface. In particular, the first implies that the no slip condition is respected; the second one is also referred to as the *no wind condition*, since if wind exists the shear stress at the free surface would be non zero; the third states that the pressure at the free surface, interface $z_{N+1/2}$ is equal to the external pressure p_{interf}. More general boundary conditions at the bottom and at the free surface are possible, but are not studied in this thesis.

The same system of equations in (1.19) can be derived using the Reynolds averaged Navier-Stokes equations, including a turbulent viscosity under the Boussinesq hypotesis. This can be useful when turbulence plays an important role. Details are reported in Appendix A.

1.3 Comments on the multi-layer model

Initially the problem is three dimensional, because the w component of velocity is an unknown of the system; however, the integration along the depth and the assumption of first order approximation in ϵ make it a two dimensional problem, since the dependence of the solution on w drops down. This concept can be represented schematically, as in Figure 1.3. The shallow water approximation reduces the order of the system by one by imposing that the horizontal velocity is constant along the depth. For this reason, the classical shallow water approximation with only one layer cannot approximate stratified flows or flows characterized by viscosity effects, because they present an horizontal velocity profile along the depth which cannot be well represented by a single average velocity. This issue is caused by the following fact: in order to derive the layer-averaged Navier-Stokes equations, and subsequently the Saint-Venant equations, we have approximated the convective term as

$$\int_{z_{\alpha-1/2}}^{z_{\alpha+1/2}} \rho_\alpha \mathbf{u}_\alpha \otimes \mathbf{u}_\alpha \, \mathrm{d}z \approx \rho_\alpha \mathbf{u}_\alpha \otimes \mathbf{u}_\alpha h_\alpha \tag{1.21}$$

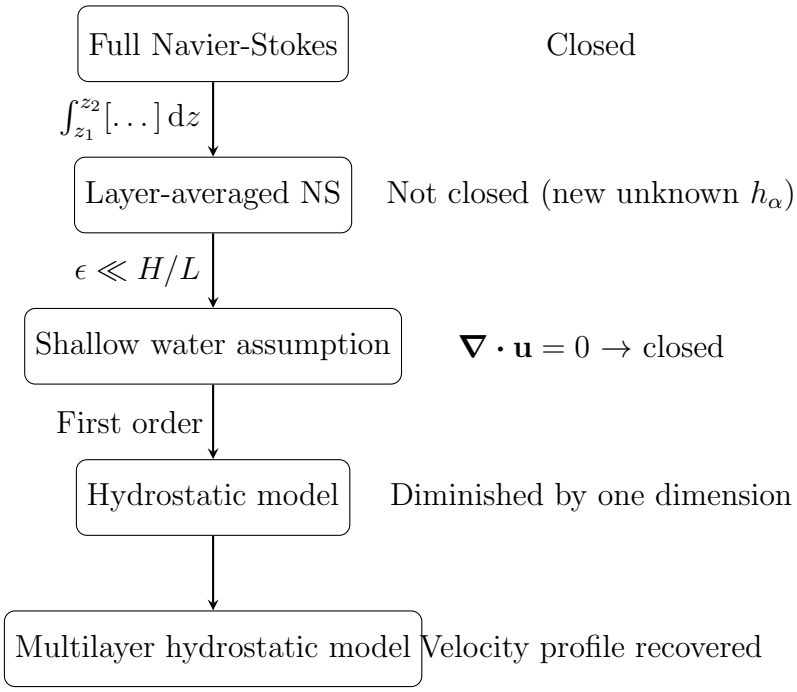


Figure 1.3: Summary of the models for free surface flows.

Consider now the one dimensional case with a single layer; the approximation is

$$\int_{z_{bottom}}^{h} \rho u^2 \,\mathrm{d}z \approx \rho u_{ave}^2 h$$

that is not exact. In fact, it should be written as:

$$\int_{z_{bottom}}^{h} \rho u^2 \,\mathrm{d}z \approx \Gamma \rho u_{ave}^2 h$$

where Γ is a shape factor given by

$$\Gamma = 1 + \frac{1}{h}\int_{z_{bottom}}^{h} \left(1 - \frac{u}{u_{ave}}\right)^2 \mathrm{d}z$$

and its importance has been widely discussed in Ref. [19]. Usually, Γ is set equal to 1, but this may have significant effects on the nature of the predicted flow, being $\Gamma \geq 1$.

It is known that an analytical solution to the Navier-Stokes equations for falling thin films is given by Equations (1.14) and (1.15). In order to reach the same solution reported in (1.15) by solving the shallow water model, Γ should be set equal to 6/5. Other examples about the dependence of the solution from the shape factor can be found in Ref. [19].

In those cases, namely the ones studied with the classical shallow water model, the issue raises because of the presence of only one layer and the approximation of $\Gamma = 1$. In the multi-layer shallow water system we can expect that this inconvenient will not happen even if $\Gamma = 1$: by computing the average velocity in each layer, we will be able to recover the

right velocity profile even if the velocity profile in the single layer is flat (namely $\Gamma = 1$). Thus we expect that adding more and more layers the velocity profile will converge to the right solution.

In conclusion, in this thesis we study two dimensional problems through the multi-layer shallow water model with hydrostatic pressure. As seen above, averaging along one dimension allows to drop down the dimension of the problem by 1; this means that if we are able to solve N one dimensional problems (one per layer), we expect to find a solution that well approximates the real one. These problems are not completely decoupled, since they talk to each other through the remaining stress tensor terms in Equation (1.19), which are grouped together in the r.h.s. of the momentum equation.

In the next chapter we discuss the strategy of solution for the N problems, including a description of the approximation of transverse derivative terms (namely the ∂_z's terms).

2. Smoothed particle hydrodynamics

Smoothed particle hydrodynamics (SPH) is a numerical method for solving the equations of fluid dynamics first introduced by Gingold and Monaghan [17] and Lucy [25] in 1977. Originally it was developed in the context of astrophysical studies, but it turned out that the method is applicable also to classical fluid mechanics problems. Classical computational fluid dynamics is based on computing quantities in some points in the domain, usually applying a grid based discretization; thus it is necessary to discretize the spatial domain in cells and to develop numerical methods able to deal with the presence of the cells while conserving mass, momentum and energy. The spatial discretization of the domain is a big deal, since the accuracy of the solution strongly depends on the quality of the grid. Furthermore in very complex geometry the process of generation of the grid can be even more complicated than the resolution of the problem, in such a way that the time spent on the grid generation can be higher than the effective computational time. In addition, if there is a free surface, appropriate techniques need to be used (such as the volume of fluids). All those methods which are based on the presence of a grid (fixed or moving) are called grid based methods: finite elements, finite differences and finite volumes belong to this class.

The smoothed particle hydrodynamics is a particle, mesh-free method, which can naturally solve free surface problems. SPH uses moving particles as interpolation points from which the properties of the fluid can be computed, reason why the grid is no longer necessary. Examples of mesh-free methods are the element free Galerkin method, the point interpolation method, the meshless local Petrov-Galerkin method and obviously the SPH. These methods are quite young compared to other grid based methods; however it turned out that they can be useful in simulating many engineering problems as described in Ref. [36].

2.1 Evaluation of a function through integral representation

The value of a function f at $\mathbf{x}$ can be computed by the integral

$$\langle f(\mathbf{x}) \rangle = \int_{\Omega} f(\mathbf{x}')\delta(\mathbf{x} - \mathbf{x}')\,\mathrm{d}\mathbf{x}' \tag{2.1}$$

where $\delta(\mathbf{x} - \mathbf{x}')$ is the Dirac delta function and $\langle(\cdot)\rangle$ is the so called kernel approximation of $(\cdot)$; Ω is the domain of the problem. Numerically it is not possible to represent the delta

function and usually a function with a finite support is used. Such a function is called the *interpolation kernel* $W = W(\mathbf{x} - \mathbf{x}', l)$, which depends on the *smoothing length* l, i.e. an indication about the range of influence of the kernel. When W is the delta function the interpolation is exact and $\langle f(\mathbf{x})\rangle = f(\mathbf{x})$; therefore the kernel W should tend to the delta function as l tends to zero. In addition, the kernel is normalized to one in order to make the interpolation of a constant exact, it is even, symmetric, positive, with compact support and smooth enough. The general form of the kernel function is:

$$W(x, l) = \frac{1}{l^n} W\left(\frac{x}{l}\right) \tag{2.2}$$

in which n is the normalization exponent and depends on the number of dimensions of the problem. Different kernels have been studied, such as the Gaussian, the quartic, the quintic and so on (see for example Ref. [24]). The approximation of the interpolation of f in a point $\mathbf{x}_A$ is then given by

$$\langle f(\mathbf{x}_A)\rangle = \int_\Omega f(\mathbf{x}') W(\mathbf{x}_A - \mathbf{x}', l)\, \mathrm{d}\mathbf{x}' \tag{2.3}$$

The SPH approximation of a function is of second order accuracy [24], as proved in Section 2.2.1.

2.2 SPH applied to fluid mechanics: the particle approximation

Equation (2.3) is valid only in the continuous case, while SPH uses a set of discrete particles distributed in space. This is the particle approximation: consider a domain of fluid and divide it into N small elements; the i-th element has mass m_i, density ρ_i and position $\mathbf{x}_i$. Each element carries its mass and a certain number of properties, such as density, velocity, energy. The i-th element occupies a volume V_i and keeps its mass constant, such that

$$m_i = \rho_i V_i$$

A generic quantity $\Phi = \Phi(\mathbf{x}_i)$ is given by (2.3) as:

$$\langle \Phi(\mathbf{x}_i)\rangle = \int \Phi(\mathbf{x}') W(\mathbf{x}_i - \mathbf{x}', l)\, \mathrm{d}\mathbf{x}'$$

Multiplying and dividing by $\rho(\mathbf{x}')$:

$$\langle \Phi(\mathbf{x}_i)\rangle = \int \rho(\mathbf{x}') \frac{\Phi(\mathbf{x}')}{\rho(\mathbf{x}')} W(\mathbf{x}_i - \mathbf{x}', l)\, \mathrm{d}\mathbf{x}'$$

and knowing that $\rho(\mathbf{x}')\, \mathrm{d}\mathbf{x}' = \mathrm{d}m(\mathbf{x}')$, discretizing the integral:

$$\langle \Phi(\mathbf{x}_i)\rangle = \int \rho(\mathbf{x}') \frac{\Phi(\mathbf{x}')}{\rho(\mathbf{x}')} W(\mathbf{x}_i - \mathbf{x}', l)\, \mathrm{d}\mathbf{x}' \approx \sum_{j=1}^{N} m_j \frac{\Phi(\mathbf{x}_j)}{\rho(\mathbf{x}_j)} W(\mathbf{x}_i - \mathbf{x}_j, l)$$

Finally the SPH interpolation of the quantity Φ is:

$$\langle \Phi(\mathbf{x}_i) \rangle \approx \sum_{j=1}^{N} m_j \frac{\Phi(\mathbf{x}_j)}{\rho(\mathbf{x}_j)} W(\mathbf{x}_i - \mathbf{x}_j, l) \tag{2.4}$$

If Φ is the density, then:

$$\langle \rho(\mathbf{x}_i) \rangle \approx \sum_{j=1}^{N} m_j W(\mathbf{x}_i - \mathbf{x}_j, l) \tag{2.5}$$

which means that mass is conserved in an exact manner. From Equation (2.4) it is natural to choose a kernel function with a compact support: as said above, the kernel function must tend to the delta function when $l \to 0$; this means that only the contribution of near particles is important, while farther particles weigh much less. The choice of a kernel function with compact support allows to consider only closer particles drastically reducing the computational time.

The SPH approximation allows the computation of spatial derivatives of the function only by knowing the derivatives of the kernel; following Monaghan [28], derivatives can be computed as:

$$\langle \boldsymbol{\nabla} f(\mathbf{x}_i) \rangle = \frac{1}{\rho_i} \left[\sum_{j=1}^{N} m_j \left[f(\mathbf{x}_j) - f(\mathbf{x}_i) \right] \boldsymbol{\nabla} W_{ij} \right] \tag{2.6}$$

or

$$\langle \boldsymbol{\nabla} f(\mathbf{x}_i) \rangle = \rho_i \left[\sum_{j=1}^{N} m_j \left[\frac{f(\mathbf{x}_j)}{\rho_j^2} + \frac{f(\mathbf{x}_i)}{\rho_i^2} \right] \boldsymbol{\nabla} W_{ij} \right] \tag{2.7}$$

where $\boldsymbol{\nabla} W_{ij}$ is the gradient of W evaluated in $\mathbf{x}_i - \mathbf{x}_j$ (and so for the divergence). It is then possible [24] to discretize the Navier-Stokes equations with an SPH approach: the method is instrisically Lagrangian, since particles move under the action of forces and transport their properties through the domain. Density of each particle changes in time as prescribed by the continuity equation

$$\frac{\mathrm{d}\rho_i}{\mathrm{d}t} = f(\rho_i, \mathbf{u}_i)$$

Changes in velocity are given by the momentum equation

$$\frac{\mathrm{d}\mathbf{u}_i}{\mathrm{d}t} = g(\rho_i, \mathbf{u}_i, e_i)$$

and energy varies in time as

$$\frac{\mathrm{d}e_i}{\mathrm{d}t} = h(\rho_i, \mathbf{u}_i, e_i)$$

where the kernel approximation brackets have been dropped down and $\mathrm{d}/\mathrm{d}t$ indicates the convective derivative. Here f, g and h represent particle approximation of the the r.h.s. of the Navier-Stokes equations. Also, pressure and all the other properties of the fluid can be found by applying the relevant equations. At each time step, new velocity, density and energy for every single particle are computed (time integration can be performed by several

temporal integration schemes, either implicit or explicit); velocity is integrated in order to find new particle position, while density and energy are needed for updating pressure and the properties of the fluid (such as viscosity, thermal conductivity, etc.). The fields of interest are found by using the SPH approximation, Equation (2.4), when it is necessary.

2.2.1 Some notes about the smoothing length

Since the kernel function has a compact support and the number of particles is finite, in the approximation of a field given by Equation (2.4) it is not necessary to consider the summation for $j = 1, \dots, N$, but only the summation over particles j that are located within the support of the kernel function centered in the position of particle i. Being the support of W compact (i.e. $W = 0$ when $|\mathbf{x} - \mathbf{x}'| > kl$, with kl the effective length of the smoothing function), when a particle is outside the radius of influence of the smoothing function its contribution to the interpolation is null. This property avoids time consuming computations.

The kernel approximation of a field is second order accurate in space, in the sense that the approximation becomes more and more accurate as the smoothing length l decreases and W tends to the Dirac's delta function. Consider a domain Ω and a function $f(\mathbf{x})$ to be interpolated in Ω; the interpolation, in the continuum, is given by Equation (2.3):

$$\langle f(\mathbf{x}) \rangle = \int_\Omega f(\mathbf{x}') W(\mathbf{x} - \mathbf{x}', l) \, \mathrm{d}\mathbf{x}'$$

By expanding $f(\mathbf{x}')$ in a Taylor series around $\mathbf{x}$:

$$\begin{aligned} \langle f(\mathbf{x}) \rangle &= \int_\Omega \left[f(\mathbf{x}) + f'(\mathbf{x})(\mathbf{x}' - \mathbf{x}) + \mathcal{O}\big((\mathbf{x} - \mathbf{x}')^2\big) \right] W(\mathbf{x} - \mathbf{x}', l) \, \mathrm{d}\mathbf{x}' \\ &= f(\mathbf{x}) \int_\Omega W(\mathbf{x} - \mathbf{x}', l) \, \mathrm{d}\mathbf{x}' + f'(\mathbf{x}) \int_\Omega (\mathbf{x}' - \mathbf{x}) W(\mathbf{x} - \mathbf{x}', l) \, \mathrm{d}\mathbf{x}' + \mathcal{O}\big(l^2\big) \end{aligned}$$

It is known that the kernel function is normalized so that its integral over the whole domain is unity; furthermore it is even with respect to $\mathbf{x}$ and if multiplied by $(\mathbf{x}' - \mathbf{x})$ it becomes odd: hence the second term on the r.h.s. is null. Then:

$$\langle f(\mathbf{x}) \rangle = f(\mathbf{x}) + \mathcal{O}\big(l^2\big) \tag{2.8}$$

It is possible to prove that also the derivative kernel approximation is second order accurate, so one can conclude that the SPH method is accurate at second order. This is true only when $\int_\Omega W(\mathbf{x} - \mathbf{x}', h) \, \mathrm{d}\mathbf{x}' = 1$. Consider for example the 1D case shown in Figure 2.1: the dashed kernel is relative to a particle that is in the centre of the domain $\Omega = [-4, 4]$, while the other one is relative to a particle located near the left boundary. The unity property of the integral is not respected when considering the particle near the boundary, and clearly second order accuracy is not reached in this case.

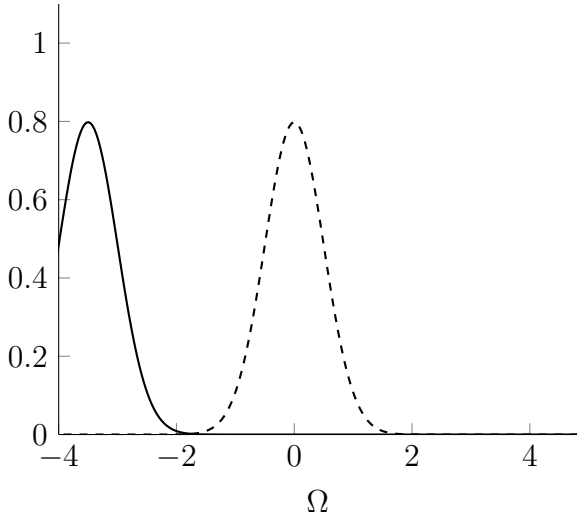


Figure 2.1: The same kernel (Gaussian) at different particle position.

Thus the treatment of boundary particles is of primary importance: a solution to deal with solid walls is the use of ghost or virtual particles on the wall to produce a repulsive force to the particles near the wall, avoiding penetration; other solutions exist, for example specular particles (see Ref. [28] for details).

Until now the smoothing length l was considered constant as done in the early SPH simulations [17]. However, since particles move and can depart or get closer, the number of particles in the support of the smoothing function varies with time. In fact, it is a good rule to maintain constant the number of particles in the support of the smoothing function in order to resolve both clustered and sparse regions: thus the smoothing length of every single particle needs to be updated at each time step, for taking into account of the relative movements of the particles. The minimum number of neighbours in 1D is 5 particles [11]. Various techniques have been studied to achieve this goal: for example, a way to maintain the number of neighbouring particles constant during the whole simulations is to impose the mass conservation [28]:

$$l_i^n = l_i^0 \left(\frac{\rho_i^0}{\rho_i^n} \right)^{\frac{1}{d}} \tag{2.9}$$

where d is the number of space dimension of the problem, n indicates the time step t^n and 0 the initial condition. This is *equivalent to keeping the mass inside the smoothing sphere constant* [32]: the mass contained in the smoothing sphere of particle i is given by

$$M_i^{tot} = \int_{V_i} \rho \, \mathrm{d}V \approx \frac{4}{3} \pi R_{kern}^3 \rho_i = \text{const.}$$

Knowing the kernel radius R_{kern} ($2l$ for the cubic and the Wendland kernel, for example) it is possible to write:

$$\frac{32}{3} \pi l_i^3 \rho_i = \text{const.} \quad \text{i.e.} \quad l_i^3 \rho = \text{const.}$$

and this proves the condition in Equation (2.9). Considering a variable smoothing length both in time and space creates subtle effects: take for example two particles A and B with

two different smoothing lengths. If particle A is contained in the support of the kernel of particle B, but the opposite is not valid, the third Newton's law is not respected, since A will perturb B, while B will not perturb A; thus in general, total momentum (and total energy) is not exactly conserved.

A possible solution to this issue, caused by the so-called $\boldsymbol{\nabla} l$ terms, is to derive the SPH formulation of the equation of motion using a variational approach [38]. Another possible approach was presented by Vila [42], and insures global conservation of physical quantities; this second option is the one we are going to explore.

2.3 SPH for systems of conservation laws

Vila [42] proposed a smoothed particle hydrodynamics method for solving systems of conservation laws in the form:

$$\frac{\partial \boldsymbol{\Phi}}{\partial t} + \sum_{l=1,d} \frac{\partial u^l \boldsymbol{\Phi}}{\partial x^l} + \boldsymbol{\nabla} \cdot \mathbf{F}(\mathbf{x}, t, \boldsymbol{\Phi}) = \mathbf{S} \tag{2.10}$$

where d is the number of space dimensions of the problem. His model is able to conserve total momentum and total energy when considering a variable smoothing length. The fluid is described by a set of moving particles with position $\mathbf{x}_i(t)$ and with a certain weight $w_i(t)$. Solving system (2.10) with a particle method is equivalent to solve the following system of equations [26]:

$$\begin{cases} \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_i = \mathbf{u}(\mathbf{x}_i, t) \\ \frac{\mathrm{d}}{\mathrm{d}t} w_i = w_i \boldsymbol{\nabla} \cdot \mathbf{u}(\mathbf{x}_i, t) \\ \frac{\mathrm{d}}{\mathrm{d}t} w_i \boldsymbol{\Phi}_i + w_i \sum_j w_j \left(\mathbf{F}_i + \mathbf{F}_j + \boldsymbol{\Pi}_{ij}\right) \cdot \boldsymbol{\nabla} W_{ij} = w_i \mathbf{S}_i \end{cases} \tag{2.11}$$

where $\boldsymbol{\Pi}_{ij}$ is a term of artificial viscosity, needed to make the time integration scheme stable; $\mathbf{F}_i$ and $\mathbf{F}_j$ stands for $\mathbf{F}(\mathbf{x}_i, t, \boldsymbol{\Phi}_i)$ and $\mathbf{F}(\mathbf{x}_j, t, \boldsymbol{\Phi}_j)$ respectively.

In order to improve accuracy of the derivative, a renormalization procedure can be applied in the approximation of derivatives [42]. A correction term $\mathbf{B}$, called the kernel gradient correction, is added in the derivative:

$$\boldsymbol{\nabla} \tilde{W}_{ij} = \mathbf{B}_i \boldsymbol{\nabla} W_{ij}$$

This term reads:

$$\mathbf{B}_i = \left[\sum_j w_j \boldsymbol{\nabla} W(x_\cdot - x_j, h_\cdot) \otimes (\mathbf{x}_j - \mathbf{x}_i) \right]^{-1}$$

where $\otimes$ is the tensorial product. Finally, considering the variable smoothing length, Equation (2.11) is modified as follows [42]:

$$\begin{cases} \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}_i = \mathbf{u}_i \\ \frac{\mathrm{d}}{\mathrm{d}t} w_i = w_i \sum_j w_j \boldsymbol{\nabla} \cdot \mathbf{u}_i \\ \frac{\mathrm{d}}{\mathrm{d}t} w_i \boldsymbol{\Phi}_i + w_i \sum_j w_j \left[\left(\mathbf{F}_i + \frac{1}{2}\boldsymbol{\Pi}_{ij}\right) \otimes \boldsymbol{\nabla} \tilde{W}_{ij,i} + \left(\mathbf{F}_j + \frac{1}{2}\boldsymbol{\Pi}_{ij}\right) \otimes \boldsymbol{\nabla} \tilde{W}_{ij,j} \right] = w_i S \end{cases} \tag{2.12}$$

where the term $\tilde{W}_{ij,i}$ stands for $\tilde{W}(\mathbf{x}_i - \mathbf{x}_j, l_i)$ and so for $\tilde{W}_{ij,j}$.
As an alternative to the artificial viscosity, it is possible to compute the fluxes ($\mathbf{F}(\mathbf{x}, t, \mathbf{\Phi})$) by using a Riemann solver [42]. However in this thesis only the artificial viscosity approach is implemented, even if it requires the estimation of an empirical coefficient. The same formulation has been used in Ref. [8] for modeling coastal flows through the nonlinear shallow water model of Saint-Venant, and we are going to take inspiration from their work.

2.4 SPH method for multi-layer Saint-Venant equations

The multi-layer Saint-Venant equations are given by Equations (1.19) and (1.20), and are written in the conservative form reported in Equations (2.10). In the particular case under study:

$$\mathbf{u} = \begin{bmatrix} u_\alpha \\ 0 \end{bmatrix}, \qquad \boldsymbol{\nabla} \cdot (\cdot) = \partial_x(\cdot), \qquad \mathbf{\Phi} = \begin{bmatrix} h_\alpha \\ u_\alpha h_\alpha \end{bmatrix}, \qquad \mathbf{F} = \begin{bmatrix} 0 & 0 \\ \frac{1}{\rho} p_\alpha h_\alpha & 0 \end{bmatrix}$$

and

$$\mathbf{S} = \begin{bmatrix} 0 \\ g \sin\beta h_\alpha + \frac{1}{\rho}\left[p_{\alpha+1/2}\partial_x\left(z_{\alpha+1/2}\right) - p_{\alpha-1/2}\partial_x\left(z_{\alpha-1/2}\right)\right] + \nu\left[\partial_z u_{\alpha+1/2} - \partial_z u_{\alpha-1/2}\right] \end{bmatrix}$$

with pressure given by the hydrostatic approximation:

$$p_{\alpha-1/2} = p_{\alpha+1/2} + \rho g h_\alpha$$

The source term contains non conservative terms, which technically are not source terms. However, in our method they are computed as if they were source terms. Boundary conditions are imposed at the free surface and at the wall by Equation (1.20).
With these variables, one can write Equations (2.12) as:

$$\begin{cases} \frac{\mathrm{d}}{\mathrm{d}t} x_{\alpha_i} = u_{\alpha_i} \\ \frac{\mathrm{d}}{\mathrm{d}t} w_{\alpha_i} = w_{\alpha_i} \sum_j w_{\alpha_j} \partial_x u(x_{\alpha_i}, t) \\ \frac{\mathrm{d}}{\mathrm{d}t} w_{\alpha_i} h_{\alpha_i} = 0 \\ \frac{\mathrm{d}}{\mathrm{d}t} w_{\alpha_i} h_{\alpha_i} u_{\alpha_i} = -w_{\alpha_i} \sum_j w_{\alpha_j} \left[\left(\frac{1}{\rho} h_{\alpha_i} p_{\alpha_i} + \frac{1}{2}\Pi_{ij} \right) B_i \frac{\mathrm{d}W_{ij,i}}{\mathrm{d}x} + \left(\frac{1}{\rho} h_{\alpha_j} p_{\alpha_j} + \frac{1}{2}\Pi_{ij} \right) B_j \frac{\mathrm{d}W_{ij,j}}{\mathrm{d}x} \right] + w_{\alpha_i} S_{\alpha_i} \end{cases} \tag{2.13}$$

where S_{α_i} is the source term for the α-th layer evaluated at particle i:

$$S = g \sin\beta h_{\alpha_i} + \frac{1}{\rho}\left[p_{\alpha+1/2}\partial_x\left(z_{\alpha+1/2}\right) - p_{\alpha-1/2}\partial_x\left(z_{\alpha-1/2}\right)\right]\Big|_i + \nu\left[\partial_z u_{\alpha+1/2} - \partial_z u_{\alpha-1/2}\right]\Big|_i$$

and is discussed in Section 2.4.3. The meaning of the equations of system (2.13) is discussed in Section 2.4.1.

Using the third equation, the fourth one can be simplified and the final system reads:

SPH for multi-layer Saint-Venant

$$\begin{cases} \frac{\mathrm{d}}{\mathrm{d}t}x_{\alpha_i} = u_{\alpha_i} \\ \frac{\mathrm{d}}{\mathrm{d}t}w_{\alpha_i} = w_{\alpha_i}\sum_j w_{\alpha_j}\partial_x u(x_{\alpha_i},t) \\ \frac{\mathrm{d}}{\mathrm{d}t}w_{\alpha_i}h_{\alpha_i} = 0 \\ \frac{\mathrm{d}}{\mathrm{d}t}u_{\alpha_i} = -\frac{1}{h_{\alpha_i}}\sum_j w_{\alpha_j}\left[\left(\frac{1}{\rho}h_{\alpha_i}p_{\alpha_i}+\frac{1}{2}\Pi_{ij}\right)B_i\frac{\mathrm{d}W_{ij,i}}{\mathrm{d}x}+\right. \\ \qquad\qquad \left.+\left(\frac{1}{\rho}h_{\alpha_j}p_{\alpha_j}+\frac{1}{2}\Pi_{ij}\right)B_j\frac{\mathrm{d}W_{ij,j}}{\mathrm{d}x}\right]+\frac{1}{h_{\alpha_i}}S_{\alpha_i} \end{cases} \tag{2.14}$$

2.4.1 Physical interpretation

A physical interpretation of system (2.14) can be performed [8]. The first equation is simply the kinematic equation for each particle in the α-th layer.

The third equation derives from mass conservation: it states that the quantity $w_{\alpha_i}h_{\alpha_i}$ is constant following the particle; this quantity is a surrogate of mass. If one thinks of w_{α_i} as the horizontal surface of the particle, the quantity $w_{\alpha_i}h_{\alpha_i}$ assumes the form of a volume of a water column: thus w_{α_i} is the horizontal surface of a particle, h_{α_i} its associated height and the volume transported by the particle cannot change in time, being the fluid incompressible. A visual interpretation was given by de Leffe et al. [8] and is reported in Figure 2.2. This framework is usually called the SWE-SPH, namely Shallow Water Equation SPH, since each particle transports an associated height. Table 2.1 contains the correspondence between typical SPH quantities and the ones of the model proposed here (SWE-SPH):

Table 2.1: Typical SPH vs SWE-SPH.

Typical SPH	SWE-SPH
ρ (kg/m^3)	h (m)
V (m^3)	w (m^{d-1})
m (kg)	V (m^d)

In the SWE-SPH, Equation (2.5) allows to compute the height associated to each particle of the α-th layer, knowing the correspondance between quantities summarized in Table 2.1:

$$h_{\alpha_i} = \sum_{j=1}^{N} w_{\alpha_j}h_{\alpha_j}W(\mathbf{x}_i-\mathbf{x}_j,l_i) = \sum_{j=1}^{N} V_{\alpha_j}W(\mathbf{x}_i-\mathbf{x}_j,l_i) \tag{2.15}$$

where l_i indicates the smoothing length associated to the i-th particle. Thus, the thickness of the layer at particle i position does depend only on the volume associated with the

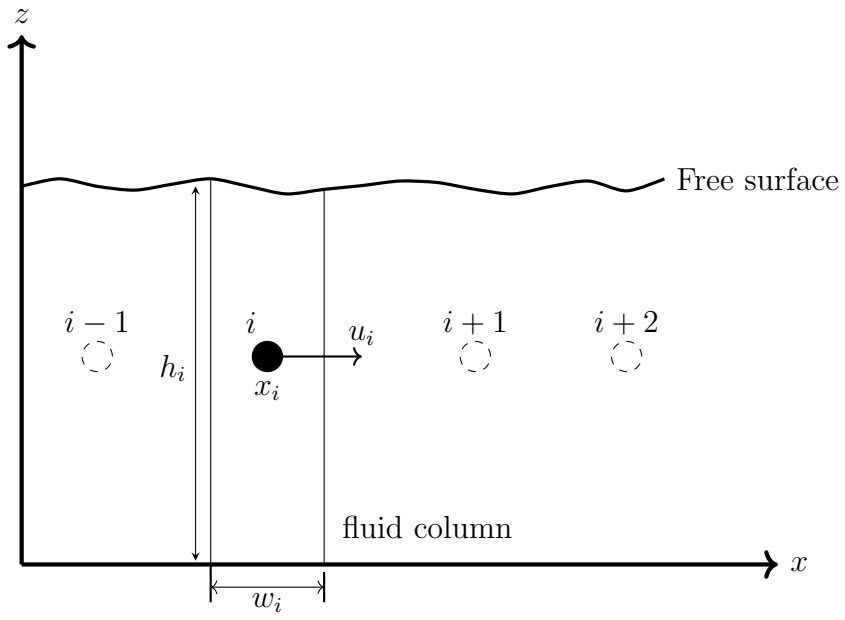


Figure 2.2: 1D visualization of the water column in the SWE-SPH method with one layer: the fluid column advected by particle i has an horizontal surface w_i and an associated height h_i, such that its volume is $V_i = w_i h_i$. In one dimension, particles can move along a straight line, that is the x-axis, accordingly to their velocity u_i.

particles inside the support of the smoothing function of that particle. Since volume is fixed by mass conservation and incompressibility, the thickness can be updated at a given time only by knowing the position of the particles at that time. Note also that Equation (2.9) for the evolution of the smoothing length and Equation (2.15) for the interpolation of the height associated to the particle are coupled at the n-th time step:

$$h^n_{\alpha_i} = \sum_{j=1}^{N} V_{\alpha_j} W\left(\mathbf{x}_i - \mathbf{x}_j, l^0_i \left(\frac{h^0_{\alpha_i}}{h^n_{\alpha_i}}\right)^{\frac{1}{d}}\right) \tag{2.16}$$

This is a nonlinear equation in $h^n_{\alpha_i}$ which can be solved with bisection, secant or Newton-Raphson methods. Here we choose the secant method, since for the Newton-Raphson method we would need an approximation of the derivative of the residual, which should be computed with the SPH approximation: in Ref. [34], the authors derive the SPH approximation of the shallow water equations with variable smoothing length using a variational formulation, and this allows to find a simple relation for the derivative of the residual and to solve Equation 2.15 through the Newton-Raphson method. In our approach, this relation has not been found, thus we use the secant method, for which the exact expression of the residual derivative is not needed. After having computed h_{α_i}, the computation of w_{α_i} is straightforward using the "volume conservation"

$$w_{\alpha_i} = \frac{V_{\alpha_i}}{h_{\alpha_i}} \tag{2.17}$$

In the multi-layer framework which has been studied in this thesis, every layer is described by a line (in two physical dimensions) or by a plane (in three physical dimensions) where particles are free to move and to interact accordingly to the fourth equation, that is the momentum equation in the x direction; with three physical dimensions, the y-momentum equation would be added to the system. Therefore, the r.h.s. of the fourth equation in system (2.14) is the Lagrangian acceleration of particle i of the α-th layer. Adjacent layers talk to each other only by means of the stress tensor terms contained in the source term S. The artificial viscosity term Π_{ij} is given by [8]:

$$\begin{cases} \Pi_{ij} = -\gamma \frac{h_{ij} l_{ij} c_{ij} u_{ij} \cdot x_{ij}}{\|x_{ij}\|^2}, & \text{if} \quad u_{ij} x_{ij} < 0 \\ \Pi_{ij} = 0, & \text{if} \quad u_{ij} \cdot x_{ij} \geq 0 \end{cases} \tag{2.18}$$

where $h_{ij} = 0.5(h_{\alpha_i} + h_{\alpha_j})$, $l_{ij} = 0.5(l_i + l_j)$, $c_{ij} = 0.5(c_i + c_j)$ and $c = \sqrt{gh}$, $u_{ij} = u_{\alpha_i} - u_{\alpha_j}$ and $x_{ij} = x_{\alpha_i} - x_{\alpha_j}$. The use of the artificial viscosity aims to avoid numerical instabilities and to allow the computation of shocks (see for example Sec. 4.1.1).

Finally, the second equation of system (2.14) is an evolution equation for the horizontal surface transported by each particle, that must be stretched in order to follow expansions or compressions of the particles. This equation is solved in order to estimate w_{α_i} at the next time step, in order to start the secant method. It is an ODE in the form:

$$\frac{\mathrm{d}}{\mathrm{d}t} w(x_i, t) = \lambda w(x_i, t)$$

whose solution is

$$w(x_i, t) = w(x_i, 0) \exp\{\lambda t\}$$

Thus, the solution to the second equation in system (2.14) is found as:

$$w_{\alpha_i}^{n+1} = w_{\alpha_i}^{n} \exp\left\{ \Delta t \sum_j w_{\alpha_j} \partial_x u(x_{\alpha_i}, t) \right\} \tag{2.19}$$

When the smoothing length is constant, height is computed by solving Equation (2.15); then w_{α_i} is found using Equation (2.17).

Another point of view on this equation can be recovered by multiplying it by h_{α_i}:

$$h_{\alpha_i} \frac{\mathrm{d}}{\mathrm{d}t} w_{\alpha_i} = h_{\alpha_i} w_{\alpha_i} \sum_j w_{\alpha_j} \partial_x u(x_{\alpha_i}, t) \tag{2.20}$$

Using the third equation in (2.14), the l.h.s. can be written as:

$$h_{\alpha_i} \frac{\mathrm{d}}{\mathrm{d}t} w_{\alpha_i} = \frac{\mathrm{d} h_{\alpha_i} w_{\alpha_i}}{\mathrm{d}t} - w_{\alpha_i} \frac{\mathrm{d}}{\mathrm{d}t} h_{\alpha_i} = -w_{\alpha_i} \frac{\mathrm{d}}{\mathrm{d}t} h_{\alpha_i}$$

Finally, Equation (2.20) becomes

$$w_{\alpha_i} \frac{\mathrm{d}}{\mathrm{d}t} h_{\alpha_i} = -h_{\alpha_i} w_{\alpha_i} \sum_j w_{\alpha_j} \partial_x u(x_{\alpha_i}, t)$$

This equation assumes the form of an evolution equation for the height associated to every particles and reads:

$$\frac{\mathrm{d}}{\mathrm{d}t}h_{\alpha_i} = -h_{\alpha_i}\sum_j w_{\alpha_j}\partial_x u(x_{\alpha_i}, t) \tag{2.21}$$

One can observe that Equation (2.21) is analogous to the equation for the time evolution of the density, that is:

$$\frac{\mathrm{d}\rho}{\mathrm{d}t} = -\rho\boldsymbol{\nabla}\cdot\mathbf{u}$$

Indeed in our model the thickness is a surrogate for the density (see Table 2.1) and the model is fully consistent with SWE-SPH existing formulations (e.g. Ref. [34]). A physical interpretation of Equations (2.20) and (2.21) is the following: when $\boldsymbol{\nabla}\cdot\mathbf{u} > 0$ there is an expansion, the horizontal surface stretches and the height decreases for the mass conservation; when $\boldsymbol{\nabla}\cdot\mathbf{u} < 0$ a compression is occurring, the horizontal surface diminishes and the height increases for the mass conservation.

2.4.2 Kernel function

A kernel function must be chosen for the solution of Equations (2.14) and (2.15). A multitude of kernel functions exists: historically, the first to be used was the Gaussian kernel [17], which, however, has non compact support. With non compact support kernel one has to consider not only the nearer particles to the point in which the interpolation takes place, but all the particles in the domain. This is unnecessarily time consuming because the Gaussian kernel becomes rapidly small with increasing distance and the most of the computation process would be spent for distant particles that are not really interacting with the point. For this reason, new kernels with a compact support had been developed, so that the time for each SPH interpolation is reduced.

In the present case, in particular, the one dimensional cubic spline kernel is employed [27]:

$$W(r,l) = \frac{2}{3l}\begin{cases} 1-\frac{3}{2}q^2+\frac{3}{4}q^3 & \text{if } 0\le q\le 1 \\ \frac{1}{4}(2-q)^3 & \text{if } 1<q\le 2 \\ 0 & \text{otherwise} \end{cases} \tag{2.22}$$

where $r = x - x_j$ and $q = r/l$. The radius of the smoothing kernel is then $2l$. Figure 2.3 shows both the kernel function and its derivative.

2.4.3 Computation of the source term

From the x-momentum equation in system (2.14) one can compute the velocity of the particle at the next time step and in order to solve it the source term S has to be estimated. S_i is composed by three different terms:

(i) the first one

$$g\sin\beta h_{\alpha_i}$$

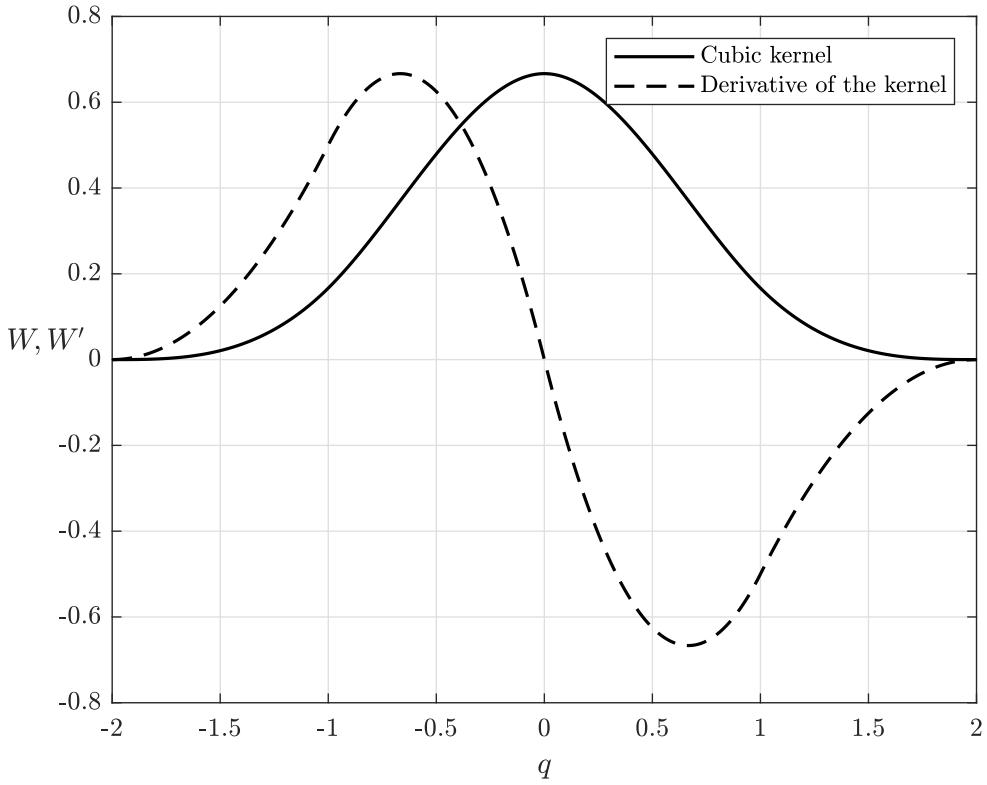


Figure 2.3: Cubic kernel function and its derivative.

is relative to the component of the gravity field parallel to the inclined plane and can been easily computed for each particle, since the thickness h_{α_i} is known;

(ii) the second

$$\frac{1}{\rho}\left[p_{\alpha+1/2}\partial_x\left(z_{\alpha+1/2}\right) - p_{\alpha-1/2}\partial_x\left(z_{\alpha-1/2}\right)\right]\Big|_i$$

contains part of the stress tensor. This term is more tricky, since it contains the derivatives of the coordinate of the interface between two different layers and it would be computationally expensive to compute it with the SPH approximation; we choose to impose a fixed grid on the domain and to compute the pressure and $z_{\alpha\pm\frac{1}{2}}$ at the grid point by applying the SPH interpolation, and then compute derivatives on the grid nodes by using a finite differences (FD) approach: second order accurate FD are employed[1] (see Appendix B for a computational cost comparison between the SPH classical approach for derivatives and the finite differences approach used here);

[1]The thickness of the α-th layer h_α at a grid node is computed by imposing a fictitious particle at the position of the node and using Equation (2.15); the coordinate of the $\alpha + 1/2$ interface is then computed by summing the thicknesses of layers $1, ..., \alpha$ (supposing the bottom is flat). Pressure is finally computed by the hydrostatic approximation, and derivatives of $z_{\alpha+1/2}$ are approximated by central finite differences at the grid nodes.

(iii) the third term is

$$\nu \left[\partial_z u_{\alpha+1/2} - \partial_z u_{\alpha-1/2}\right]\Big|_i$$

and also in this case the use of a fixed grid is necessary (see Appendix B); average velocity at grid nodes is computed by SPH interpolation[2]. Then, the term is approximated on the grid nodes with a second order approximation:

$$\begin{cases} \partial_z u_{1/2} = 2\frac{u_1 - u_{1/2}}{h_1} = 2\frac{u_1}{h_1} & \text{if } \alpha = 1 \\ \partial_z u_{\alpha+1/2} = 2\frac{u_{\alpha+1} - u_\alpha}{h_{\alpha+1} + h_\alpha} & \text{if } \alpha = 2, \dots, N \\ \partial_z u_{N+1/2} = 0 & \text{if } \alpha = N, \text{ no wind shear condition (BC)} \end{cases} \tag{2.23}$$

Once the derivatives and the pressure at the grid nodes have been computed, a process of interpolation at the particle position is performed. A linear interpolation gives good results.

2.4.4 Boundary conditions

One of the most problematic issues in particle methods is the imposition of boundary conditions: the Lagrangian nature of these methods makes more difficult to impose them than in classical grid based methods, in which, usually, quantities or fluxes are directly prescribed at the boundary. The SPH community is focusing on this issue only in these last years, considering it as a "grand challenge" [41]; when the SPH was initially proposed, it was adopted for astrophysical simulations, and boundaries are not present in that contest. However, in the last two decades SPH has assumed a role in other fields, such as coastal engineering. In order to simulate the behaviour of a fluid bounded in a closed domain or to introduce fluid in the computational domain (for example at an inflow), it is necessary to develop new techniques. A good review on the existing methods is presented in Ref. [41]: the research is mainly focused in developing robust treatment of boundary conditions for complex geometries, and formulating models with high order of accuracy. At the boundary many different procedures can be applied: for example, ghost or mirror particles, semi-analytical approach, fixed particles able to generate a force in order to enforce the boundary conditions [28]–[30], [41].

In this thesis, two different boundary conditions have been implemented:

(i) Wall boundary conditions: they are simulated by the use of mirror particles (MP) to avoid the truncation of the kernel of particles near the boundary. Fluid particles that are contained in the support of the particle at the boundary are mirrored with respect

[2]Placing a fictitious particle at a grid node, average velocity u_α is computed using Equation (2.4), which for average velocity becomes:

$$u_{\alpha_{\text{GN}}} = \sum_{j=1}^{N} w_{\alpha_j} u_{\alpha_j} W(\mathbf{x}_i - \mathbf{x}_j, l_i)$$

where l_i is a smoothing length associated with the grid node.

to the boundary itself [24]. One issue related to the MP method is that complex geometries are difficult to implement. Since our boundaries are simple walls, it is straightforward to implement the mirror particles method. The treatment is only used as lateral boundary, since particles can only move along a line (or a plane when three physical dimensions are considered) parallel to the bottom, and the vertical velocity is null: they cannot feel the presence of the bottom if no topography exists. The mirror particle method is usually used in order to respect the no-penetration condition and to make acceleration of the particles at the boundary to be null, so that the particles that are placed in correspondence of a wall with zero velocity will remain there and will not pass through the wall. However this is true only when all the acceleration terms in the momentum equation in system (2.14) are discretized through the SPH approximation. As we have seen, in our approach this would lead to expensive computational costs, and in the present formulation the term S_i comes from interpolation. In this case the mirror particles are not able to strictly impose the no-penetration condition, because their contribution does not appear in the source term. For this reason we complement the MP with a simple model of elastic collision between the wall and the particles to enforce the no-penetration condition. Furthermore, the velocity at the grid nodes located at the walls is imposed to be always null. However the mirror particles are still needed in order to achieve the second order accuracy of the SPH approximation at the wall, which would be otherwise deteriorated if they did not exist.

(ii) Open boundary conditions: in Ref. [29] the authors describe several possible options for open boundary conditions; the main idea is to introduce two zones, called the *buffer zones*, at the inflow and at the outflow, in which the so called open boundary particles (OBP) are located. These OBP are needed in order to: i) maintain the domain filled of particles and ii) avoid the truncation of the kernels of the particles close to the boundary. OBPs are free to move and to transport properties. In particular, at the inflow specific quantities are imposed, while at the outflow a do-nothing approach has been implemented: particles properties in the outflow buffer zone are frozen and their velocity is extrapolated from the domain. In this way the outflow boundary will not reflect outgoing disturbances. Looking at Figure 2.4, in which a three dimensional configuration is represented (with only one layer), one can understand how the process works: when an inlet particle passes the inlet boundary it becomes a fluid particle and a new particle is created in the inlet buffer zone, such that $x_{\text{new}} = -L_{BZ} + x_{\text{fluid}}$, where L_{BZ} is the length of the buffer zone (tipically 2 times the kernel radius) and x_{fluid} is the position of the particle that passes into the fluid region. In this way the inlet buffer zone is always filled with particles. The properties of new inlet particles are chosen in order to respect the inlet boundary condition. When a fluid particle passes the outlet boundary, it becomes a boundary particle and when it passes the extreme right boundary of the buffer zone it will be deleted.

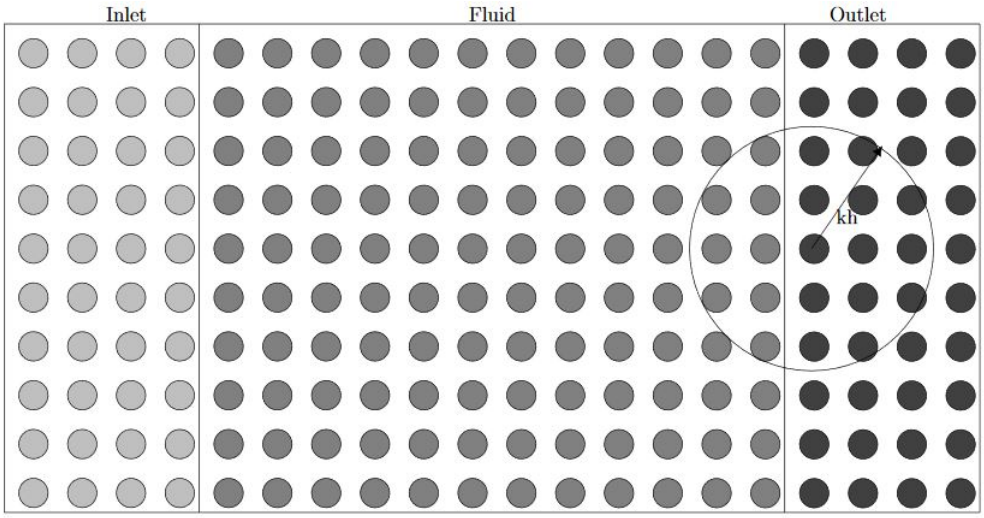


Figure 2.4: Inlet, outlet and fluid particles; the filled kernel is shown. From Ref. [29].

2.4.5 Time integration

The final step to be performed after having computed the accelerations on all particles is the time integration of the momentum and of the kinematic equations. This step can be done using several numerical methods: for example in Ref. [8] a fourth order Runge-Kutta scheme is employed. In this thesis the Euler-Cromer method is implemented: its advantages are the lower memory storage and computational effort compared to a multistep integrator, since only one force evaluation per particle is needed at every time step.

The integrator reads:

$$u^{n+1} = u^n + a^n \Delta t \tag{2.24}$$

$$x^{n+1} = x^n + u^{n+1} \Delta t \tag{2.25}$$

where the acceleration a^n is computed by solving the r.h.s. of the fourth equation in system (2.14). The method can be used with both constant and variable time step, by specifying Δt or the CFL number.

In order keep the solution stable, the time step should satisfy the CFL condition and the stability criterion for diffusion problems:

$$\frac{\Delta t}{\min\left(\frac{l_i}{c_i + ||u_i||}\right)} < 1 \tag{2.26}$$

$$\Delta t < \frac{\min\left(h_\alpha\right)^2}{2\nu} \tag{2.27}$$

where $c_i = \sqrt{g h_i}$, ν is the kinematic viscosity coefficient. The CFL condition states that, in order to maintain nnumerical stability, the space travelled by a wave must be lower than the smoothing lenght of a particle, while Equation (2.27) is the stability criterion for central finite differences method.

2.5 Solution procedures: summary

All the procedures that have been described above can be assembled in order to depict the process as a whole. It can be summarized as follows (in the case of multiple layers):

(I) Impose initial conditions and boundary conditions;

(II) Start time iteration; for each layer:

(i) Compute the acceleration for each particle, i.e. the r.h.s. of the momentum equation in system (2.14): this step involves the estimation of pressures $p_{\alpha+1/2}$, derivatives $\partial_x z_{\alpha+1/2}$ and $\partial_z u_{\alpha+1/2}$ at the particle position, through the procedure described in Section 2.4.3;

(ii) Integrate momentum equation using Equations (2.24) and update and store the position using Equation (2.25);

(iii) Apply boundary conditions:

– wall BC, create new mirror particles in order to estimate the acceleration at the next time step;

– open BC, update the arrays of fluid and boundary particles;

(iv) If smoothing length is allowed to change, update $h_{\alpha_i}^{n+1}$ and $l_{\alpha_i}^{n+1}$ solving Equation (2.16) by the secant method; once $h_{\alpha_i}^{n+1}$ and $l_{\alpha_i}^{n+1}$ are known, from the conservation of mass, Equation (2.17), it is possible to find $w_{\alpha_i}^{n+1}$ and to store it;

(v) Write the output (if needed), then return to step (i).

In the next chapter this process will be explained in a more complete way, also presenting its implementation.

3. The code

Several software exist for the numerical solution of the Navier-Stokes equations with the SPH method (e.g. DualSPHysics, GPUSPH, PySPH), but to the author's best knowledge, none of them solves the first order Saint-Venant problem with an arbitrary number of layers; therefore a dedicated program has been implemented. The code is called MultiLayerSPH and is written in Fortran 2003, both for its simplicity and flexibility in treating arrays; the code can be run serially, but it could be parallelized in future works, being the problem itself a set of N parallel problems which talk to each other only through the stress terms at the interfaces.

We remember that the problem to be solved has the geometry shown in Figure 1.2 and the set of governing equation is (2.14).

3.1 Main structure of the code

The code is organised as shown in Figure 3.1: a pre-processor allows to read the input file (`.inp`) and to set initial conditions for the problem. In particular, it allows to generate the grid for the computation of derivatives and to initialize position and velocity of particles. All the information needed for starting the time integration are stored in arrays, one for each layer. Initial conditions are also set in the boundary zone, where mirrored particles or open boundary particles are needed to make the calculation of accelerations possible. The solver block contains all those routines which are needed for the time integration as explained in Section 2.5 in point (II). In particular, firstly accelerations of all particles are computed, by computing the terms in the r.h.s. of the momentum equation in (2.14); then the time integration is possible, through the Euler-Cromer integrator, depending on the time step, as already described in Section 2.4.5. The integration is performed if the time step respects the conditions in Equations (2.26) and (2.27), otherwise Δt is computed in order to respect the stability conditions. With the time integration, new particles' positions are stored. If the domain in Figure 1.2 has open boundaries, the open boundaries routines are called: these routines make the exchange of particles between the buffer layer and the domain possible, dynamically updating the arrays containing all the information about OBPs and real particles. If the domain is bounded, then all the particles that moved outside the domain hit the walls and bounce back in order to fulfill the no penetration condition; note that in this case, if there are particles on the wall, they need to stay still, i.e. their acceleration must be null. Finally, if the smoothing length is allowed to vary in space, the smoothing length updating routine is called, otherwise the step is skipped. The final step

consists in calling the output process. It is composed by two different routines: the first one computes all the needed quantities at the grid nodes, which will be used in order to estimate derivatives and pressures at the next time step; the second one writes the results on `.dat` files, which can be post-processed. The process continues in loop while $t \leq T_{fin}$.

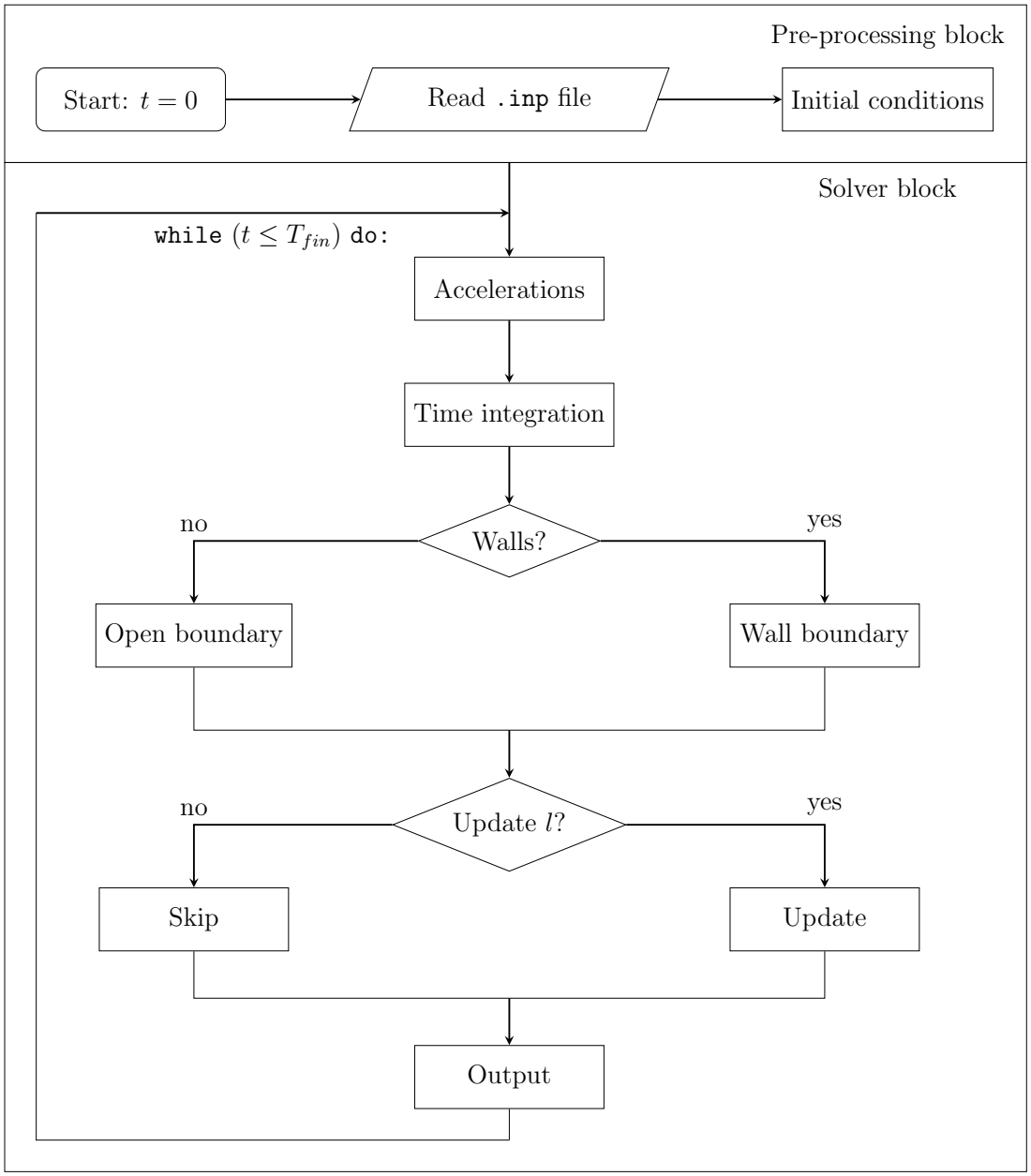


Figure 3.1: Flow chart of the general procedure.

The code is subdivided in modules, following the nomenclature of the Fortran language. The modules are grouped in the `MultiLayerSPH/src` directory:

– `mod_global.f03`, in which all global variables are defined;

- `mod_eulerian.f03` contains all the routines related to the auxiliary grid;
- `mod_sph.f03` groups all the routines related to SPH;
- `mod_integration.f03` contains the integrators and the routines related to stability conditions;
- `mod_boundary.f03` with all boundary conditions related routines;

The `main.f03` file (in directory `MultiLayerSPH/src`) includes the whole process described in the flow chart in Figure 3.1.

3.2 Pre-processing block

The program starts by reading the input file (`.inp`) written in the following format, which is based on the `namelist` I/O format (a namelist is anything that appears between `&GROUP_NAME` and `/`):

```
&probl_domain
case = # test case
beta = angle of inclination of the bottom
length = length of the domain
boundary = .TRUE./.FALSE. (closed/open)
nlayer = # layers
tfin = final time of simulation
/

&SPH
numpar = # particles
ninterp = # grid nodes
dt = time step
CFL_fixed = fixed CFL number, if variable time step is needed ...
            ... (0 = constant dt)
art_visc_coeff = artificial viscosity coefficient
SLupd = .TRUE./.FALSE. (variable smoothing length/fixed smoothing length)
/

&physics
patm = atmospheric pressure
rho = density
nu = kinematic viscosity coefficient
turbulence = .TRUE./.FALSE. (turbulence modelling ON/OFF)
/

&results
```

```
res_freq = output printing frequency
/
```

The `test_case` keyword accepts different test cases such as the dam breaks (dry bed 1, wet bed 2), the Nusselt flow (3), the Gaussian hump (4).

The `read_input` routine (`MultiLayerSPH/src/mod_global.f03`) is a simple procedure that is called once in the whole simulation and reads the input file. After `read_input`, the program calls `init_grid` (`MultiLayerSPH/src/mod_eulerian.f03`), which generates the auxiliary mesh, composed by one line for each layer. Depending on the boundary conditions, the grid can extend outside the physical domain; this happens if the `boundary` variable is set to `.FALSE.` in the input file. The mesh is described by a Fortran derived type:

Mesh derived type

```
type mesh
  real*8, allocatable :: x(:), u(:), have(:), l(:), pint(:)
  real*8, allocatable :: pave(:), zcoord(:), dudz(:), dhdx(:)
end type mesh
```

Every layer is described by a structure of this type; the N layers are grouped into an array of `mesh` derived types, such that element 1 of the array is the lower layer and element N is the upper one. Element 0 of the array describes the topography and the physical quantities at interface $z_{1/2}$, referring to Figure 1.2. Every layers contains allocatable arrays for:

– nodes position (`x(:)`);

– velocity at nodes position (`u(:)`);

– smoothing length associated to the nodes (`l(:)`)

– the pressure at the upper interface of the layer in correspondence of the nodes position (`pint(:)`);

– average pressure in the layer at nodes position (`pave(:)`);

– z coordinate of the upper interface of the layer in correspondence of the nodes position (`zcoord(:)`);

– velocity z derivative at the upper interface of the layer evaluated at nodes position (`dudz(:)`);

– x derivative of the upper interface of the layer in correspondence of the nodes position (`dhdx(:)`).

The `init_grid` routine sets nodes positions in every layer and is called once, since the grid is fixed.

Before the simulation begins, fluid particles (and boundary particles) need to be placed through `set_particles` and `set_bparticles` routines (`MultiLayerSPH/src/mod_sph.f03`).

The first one introduces equispaced particles in the domain such that initial conditions are respected; furthermore it imposes the initial smoothing length (usually equal to $2\Delta x$, where Δx is the initial particle spacing) and w_i, namely the horizontal surface of each particle (set initially equal to Δx). The second one creates mirrored particles, if wall boundaries are used, and OBPs if inflow/outflow conditions are set. Both fluid and boundary particles are described by an array of `particles` derived type:

Particles derived type

```
type particles
  real*8, allocatable :: x(:), u(:), a(:), V(:), w(:), nu_t(:)
  real*8, allocatable :: have(:), l(:), B(:), inhave(:), inl(:)
  real*8, allocatable :: xold(:), uold(:)
  real*8, allocatable :: pave(:), pplus(:), pmin(:)
  real*8, allocatable :: dudzplus(:), dudzmin(:), dhdxplus(:), dhdxmin(:)

  integer, allocatable :: loc(:)
  integer              :: np
end type particles
```

Element 1 of the array of structures contains information about particles of the lower layer, and so on. Fluid particles are stored in the array of structures `p(1:N)` and boundary particles in the array of structures `bp(1:N)`, where N is the number of layers. Using Fortran formalism, we declare the array of structures for fluid particles and boundary particles as:

Particles structure definitions

```
type(particles), allocatable :: p(:)
type(particles), allocatable :: bp(:)
```

The components of the structures can be accessed using the component selector character %. For example, the position of particle i in the k-th layer can be accessed through `p(k)%x(i)` (and the same for the components of the array of mesh derived type, in order to access properties at the grid nodes). Each particle is described by:

– its position (`x`), velocity (`u`), acceleration (`a`), volume (`V`), horizontal surface (`w`) and associated heights (`have`);

– its smoothing length (`l`) and, the turbulent viscosity (`nu_t`), if turbulence modelling is required;

– a term (`B`) that is used as gradient correction in SPH gradient approximation;

– its initial heights (`inhave`) and initial smoothing length (`inl`), that are used if the smoothing length is allowed to change;

– previous time step position (`xold`) and velocity (`uold`), that are nedeed for the integrator;

– the average pressure in its position (`pave`);

– other quantities evaluated at its position and used in the computation of accelerations (pressure at upper/lower interface `pplus`, `pmin`, velocity z derivative at upper/lower interface `dudzplus`, `dudzmin` and x derivative of the upper/lower interface `dhdxplus`, `dhdxmin`);

– an integer number (`loc`), which describes the location of the particle with respect to grid nodes. If the particle is between node i and $i+1$, `loc` assumes the value i. This variable is used for the interpolation process.

Finally, the number of particles per layer is stored in `np`. Note that `set_particles` routine sets only position, volume, horizontal surface, height, smoothing length and initial height and initial smoothing length. The remaining variables are computed during the simulation, before acceleration is estimated. The `set_bparticles` routine imposes the same quantities for boundary particles. When open boundary conditions are used we set the size of particles related arrays (for the k-th layer: `p(k)%x`, `p(k)%u`,...) greater than the effective number of particles: some elements are empty (set equal to 0). In this way boundary particles (OBPs) can be transformed into fluid particles by switching between `bp` to empty elements of `p` and vice versa, without reshaping the size of the arrays, as explained in Section 3.3.3.
When the initialization process is concluded, the solver can start the time integration.

3.3 Solver block

The solver block is formed by the routines that compute the numerical solution.

3.3.1 Accelerations

Being initial conditions imposed and initial heights and velocity at the grid nodes known, average pressures are computed by calling the `avepressure` routine and pressures at the interfaces between adjacent layers are found with the `pressure` routine, which uses the information about the heights of each layer at the grid node to estimate the position of the interface and then applies the definition of hydrostatic pressure (average and interface pressures are both computed at grid nodes, as explained in Section 2.4.3); also, the `derivatives` routine is called, which calculates velocity derivatives in x and z by using finite differences, as pointed out in Section 2.4.3.
Before applying the SPH approximation for particles' accelerations, we need to estimate derivatives, average pressure and interface pressure at particle position: this is done by the `interp` routine through a linear interpolation; in particular, the `locpar` routine determines the location (`p(k)%loc(i) = j`) of every particle with respect to the nodes; then, the `interp` routine interpolates quantities at the particle position, knowing that particle i is contained in the region between grid nodes j and $j+1$[1]. The linear interpolation is performed as follows:

$$\phi(x_i) = \phi_j + \frac{\phi_{j+1} - \phi_j}{\Delta x_{\text{mesh}}} (x_i - x_{\text{mesh},j}) \tag{3.1}$$

[1]Particle location (`p(k)%loc(i)`) is an integer number j if the particle is between node j and $j+1$.

where $\phi(x_i)$ is the interpolated quantity at particle i position x_i, ϕ_j and ϕ_{j+1} are the quantity values at the grid nodes j and $j+1$, Δx_{mesh} is the nodes spacing and $x_{\text{mesh},j}$ is the x-coordinate of the j-th grid node.

Routines described above are all contained in `mod_eulerian.f03` module.

Thus, all the information needed for the estimation of accelerations are available. Note that if wall boundary conditions are imposed, the grid is fully contained in the physical domain and the average pressures for mirrored particles are simply set equal to the average pressures on the mirrored particle; note also that other quantities as the derivatives are not needed for boundary particles, since derivatives participate in the computation of the source term S, which is necessary only for real particles. If open boundaries are present, the grid extends beyond the physical boundaries, for all the buffer layer length: this allows the interpolation of average pressure for boundary particles. Before accelerations can be computed the `grad_corr` routine is called, which computes the gradient correction term B_i for each particle and stores it in `p(k)%B` (and `bp(k)%B` for boundary particles). When a closed boundary is chosen the mirrored particles assume the same values for B_i of the real particles. When open boundaries are considered, the gradient correction term is estimated in the following way: a virtual node, which is the mirroring of the OBP with respect to the boundary (so it is in the physical domain), is created and B_i for the OBP is computed in correspondence of this position[2].

If the `turbulence` keyword in the input file is set on `.TRUE.`, the `turbulent_viscosity` routine is called and the turbulent viscosity associated with every particle (`p%nu_t`) is computed (see for details Appendix A).

Accelerations are then computed by the `acceleration` routine that can be found in the file `MultiLayerSPH/src/mod_sph.f03` and gives the r.h.s. of the momentum equation in (2.14).

3.3.2 Time integration

Particles move under the action of forces and their motion trasports quantities, such as the heights of each layer. The motion is resolved by integrating the Newton's second law, namely the momentum equation, with the integrator described in Section 2.4.5. The integrator is located in the directory `MultiLayerSPH/src/mod_integration`. In case of closed boundaries, particles that initially are placed in contact with the walls must not move for respecting the no-penetration condition: however, as explained in Section 2.4.4 the no-penetration condition cannot be stricly fulfilled by means of mirrored particles in this particular formulation, and thus they begin to move. Their displacements can be an issue when mirrored particles are placed, because, as explained in Figure 3.2, if the particle that initially is at the wall begins to move, non physical perturbations will grow.

This phenomenon is fixed by imposing null displacement for those fluid (real) particles that are in correspondence of the wall at time t^0. On the other hand, for open boundary

[2]If B_{OBP} were computed in the actual position of the OBP, the kernel would be cut and the SPH approximation would be less accurate.

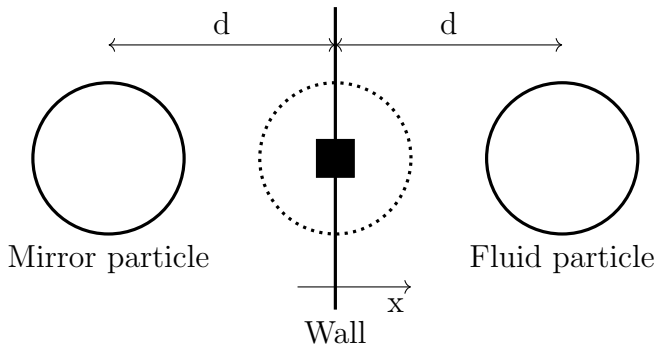


Figure 3.2: The particle in its initial configuration is the dotted one and d is its displacement in the domain ($x > 0$) at time t^n; the mirrored particle is placed at $x = -d$. If $d \neq 0$, the quantity computed at $x = 0$ in the grid node represented as a black square through the SPH approximation will be affected by a non physical disturbance.

conditions this issue does not occur because OBPs are allowed to come into the domain and to become real particles (and vice versa).

Before the time integration, it is important to verify if the stability conditions of Equations (2.26) and (2.27). This is done by the `stability_check` routine, which is contained in `MultiLayerSPH/src/mod_sph.f03`.

3.3.3 Boundary conditions

The file `MultiLayerSPH/src/mod_boundary.f03` collects boundary conditions related routines.

When particles move, they can collide with walls if closed boundaries are set, or they can simply leave or enter the domain if open boundaries are chosen. The kind of boundary routines that are called depends on the value assumed by the `boundary` keyword in the input file (`.inp`):

– `boundary = .TRUE.`: the `collision` routine makes particles collide with the walls through elastic collisions, then a simple `closed_boundary` routine is called, which mirrors all the particles in the vicinity of the boundary;

– `boundary = .FALSE.`: if particle i of the k-th layer is a OBP, it belongs to the array of structures `bp(k)` and its properties are stored in `bp(k)%x(i)`, `bp(k)%u(i)` and so on; when the particle passes the inflow boundary, the following procedure is performed:

  (a) Find empty elements in `p(k)%V`; if the element `idx` of the array `p(k)%V` is null, its index is stored as an "empty index", meaning that element `idx` of each component of the structure `p(k)` contains a zero; that is, if `p(k)%V(idx)` $= 0$, no particle is stored in that element. This is done by calling the `empty_index` routine.

  (b) Transfer particle i from `bp(k)` to index `idx` of the arrays contained in `p(k)` (which is empty and can host the new particle):

```
bp(k)%x(i) -> p(k)%x(idx)
bp(k)%u(i) -> p(k)%u(idx)
bp(k)%a(i) -> p(k)%a(idx)
bp(k)%l(i) -> p(k)%l(idx)
bp(k)%V(i) -> p(k)%V(idx)
...
```

and so on.

A similar procedure is followed for outgoing particles, transferring them from particles data structures `p` to boundary particles data structures `bp`. The procedure to be performed if particle i passes through the outflow boundary is the following:

(a) Transfer particle information from particles data `p(k)` to an empty element `idx` of boundary particles structure `bp(k)` (which can host the outgoing particle):

```
p(k)%x(i) -> bp(k)%x(idx)
p(k)%u(i) -> bp(k)%u(idx)
p(k)%a(i) -> bp(k)%a(idx)
p(k)%l(i) -> bp(k)%l(idx)
p(k)%V(i) -> bp(k)%V(idx)
...
```

(b) Set `p(k)%V(i)` $= 0$, such that the element `i` of particles data structure `p(k)` will be read as empty, and can host inflow particles.

Finally, open boundary conditions must be updated for all those particles entering the buffer layers, through the `open_boundary` routine following what explained in Section 2.4.4.

3.3.4 Smoothing length update

Routines described in this section are contained in `MultiLayerSPH/src/mod_sph.f03`.

If the smoothing length update keyword (`SLupd`) is set to `.FALSE.`, the smoothing length is kept constant and the `update_w` routine is called. It updates h_{α_i} for each particle, by solving Equation (2.15); finally, w_{α_i} is then given by Equation (2.17).

If instead `SLupd` is set to `.TRUE.`, the `secant` routine is called: this is the most time consuming part of the program so far. However in some problems variable smoothing length is strongly necessary in order to solve them correctly: one example is the dam break problem with dry bed. The tolerance of the secant method is set to the machine precision, which, in case of double precision, is 10^{-15}. This routine could be easily parallelised in order to make it much quicker, since it solves $N_{\text{particles}}$ decoupled problems for each layer: one can for example distribute layers over more cores. As explained in Section 2.4.1, after having computed with Equation (2.16) the new heights h_{α_i} transported by the i-th particle using

the secant method, the estimation of its smoothing length l_{α_i} comes from Equation (2.9) and the new "horizontal surface" of the particle w_{α_i} is found inverting the mass conservation as done in Equation (2.17). In order to initialize the secant method, an estimation of w_{α_i} is performed using Equation (2.19).

3.3.5 Output

The last procedure to be called composes the output block in the flow chart in Figure 3.1. It is formed by two main routines, the first one is needed at each time step and the second one only when the user decides to print the output files. In particular, the first is named `quantities_grid` and it is called in order to restart the process: its aim is to compute the heights and the velocity at each grid node. In case of open boundaries, the grid extends beyond the boundaries, and it is necessary to implement a Shepard filter in order to compute quantities in the buffer layer's nodes. The Shepard filter allows to apply the SPH approximation when the kernel is truncated, as in the case of the nodes which are in the buffer layers. The filter allows to have a zero-th order consistency in the approximation and is given by [30]:

$$\phi_i(x) = \frac{\sum_j w_j \phi_j W_{ij,i}}{\sum_j w_j W_{ij,i}} \tag{3.2}$$

Note that in order to compute quantities at the grid nodes with the SPH approximation, a smoothing length must be assigned to the grid nodes: here we assumed that l for the node is equal to the one of the nearest particle to the node. This is not strictly correct when the smoothing length varies in space, but it has been found that it behaves well in the studied cases[3]. On the contrary, when the smoothing length is assumed constant, the method is consistent. The routine is located at `MultiLayerSPH/src/mod_eulerian.f03`.

Finally, the `output` routine (`MultiLayerSPH/src/mod_sph.f03`) creates `.dat` files in which results are stored. The process can now restart from the acceleration block by computing pressures and derivatives, and the time step is increased, until it reaches the requested one.

[3]One possible more consistent solution is to use an SPH approximation, namely Equation (2.4), for computing the smoothing length at the grid nodes. In this way, the smoothing length associated to grid nodes would be:

$$\langle l_i \rangle \approx \sum_{j=1}^{N} w_j l_j W(x_i - x_j, l_i)$$

where $\langle (\cdot)_i \rangle$ refers to the quantity $(\cdot)$ at the node position, and $(\cdot)_j$ is the quantity related to particles close to the node i. This relation is non linear in the smoothing length associated to the i-th grid node and in order to solve it, a root-finding algorithm is necessary, for example the same secant algorithm used to solve Equation (2.16). Even if this solution is more consistent than the one used in this work, our choice does not affect the solution in an appreciable way and saves computational time.

4. Validation and results

A set of benchmark solutions for the shallow water equations have been proposed in literature for the verification of the accuracy of numerical methods. For example, Ref. [10] collects a compilation of analytic solutions for the shallow water equations. Such solutions are based on the classical form of the shallow water equations, with only one layer. In the present model, viscosity effects can be appreciated in the presence of a velocity profile.

We considered some test cases, to verify the accuracy of both the inviscid and the viscous SPH formulation of the multilayer shallow water equations:

(i) inviscid:

- Dam break with dry bed: analytical solution available;
- Dam break with wet bed: analytical solution available;
- Gaussian hump: no analytical solution, compared to a different numerical method;

(ii) viscous (necessarily with more than one layer):

- Nusselt flow: one of the simplest exact solutions of the Navier-Stokes equations;
- Comparison with experimental results:
  (a) Dam break with dry bed;
  (b) Dam break with wet bed;
  (c) Roll wave train.

The testing platform is a personal computer with 4GB RAM and the code is run on a single 2.40 GHz CPU.

4.1 Inviscid analysis

The next test cases are typical examples of solution of the classic (one layer) shallow water equations. In inviscid cases the multi-layer approach is not strictly needed, since its major goal is to approximate the velocity profile induced by viscosity. However, we will show that our formulation gives good results compared against analytic solutions, both in the mono-layer and multi-layer cases.

4.1.1 Dam break

The inviscid shallow water equations are the equivalent of the Euler equations for gasdynamics. It is then possible to solve a Riemann problem for the dam break test case. It is an initial value problem with two piecewise constant initial depths:

$$h(x,0) = \begin{cases} h_L & \text{for} \quad x \leq \bar{x} \\ h_R & \text{for} \quad x > \bar{x} \end{cases} \tag{4.1}$$

Let us suppose h_L to be non zero; the case with $h_R = 0$ is referred to as the dry bed case, while the case with $h_R \neq 0$ as the wet bed one.

Dry bed

The analytical solution was found by Ritter [33] and is reported in Ref. [10, Sec. 4.1.2]. The dam break is istantaneous, the bottom is flat and $\nu = 0$. The initial conditions are such that the left state is fixed, representing a reservoir of constant height h_L:

$$h(x,0) = \begin{cases} h_L & \text{for } 0 < x \leq \bar{x} \\ 0 & \text{for } \bar{x} < x \leq L \end{cases} \tag{4.2}$$

where L is the length of the domain. The analytical solution is given by:

$$h(x,t) = \begin{cases} h_L & \text{for } 0 \leq x \leq \bar{x} - t\sqrt{gh_L} \\ \frac{4}{9g}\left(\sqrt{gh_L} - \frac{x-\bar{x}}{2t}\right)^2 & \text{for } \bar{x} - t\sqrt{gh_L} < x \leq \bar{x} + 2t\sqrt{gh_L} \\ 0 & \text{for } x > \bar{x} + 2t\sqrt{gh_L} \end{cases} \tag{4.3}$$

In our test case, we consider the following settings: $L = 10$ m, $h_L = 1$ m, $\bar{x} = 3$ m. The analytical solution is self-similar and if initial conditions are appropriately scaled the solution will be always the same: thus the choice of initial condition is unimportant. All the input file settings are reported in Appendix C.

The angle of inclination of the plate is null, and we consider only one layer in order to verify that the method is able to reproduce the solution in the classical SPH-SWE (see for comparison Ref. [8, Sec. 5.2]). `boundary` is set to `.TRUE.`, meaning that wall boundary conditions are imposed. The time step is fixed to $\Delta t = 0.001$ s. It is fundamental to update the smoothing length, otherwise the wave front would not been resolved because the particles close to it would move away from the neighbours and they would remain alone, without being influenced by the others. Initial smoothing length is set $l = 2.5\Delta x$, i.e. 2.5 times the initial particle spacing: in this way, initially, the number of neighbour particles is 8, being consistent with the requirement of a minimum of 5 neighbours.

We have simulated eight different cases, summarized in Table 4.1:

Table 4.1: Dam break with dry bed: summary of simulations.

# grid nodes	# particles
100	50
	100
	200
	400
200	50
	100
	200
	400

A convergence analysis is made both for the case with 100 grid nodes and the case with 200 grid nodes, with increasing number of particles (i.e. diminishing the initial spacing between particles). In Table 4.2 it is reported the L_2 error of the non dimensional water depth, as defined in Ref. [40]:

$$||\text{err}_h||_2 = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\frac{h_i - h_i^{ex}}{h_L}\right)^2} \tag{4.4}$$

where N is the number of grid points in the layer, h_i is the numerical solution at the i-th grid node.

Table 4.2: Dam break with dry bed: L_2 depth errors at $t = 0.5$ s.

# grid nodes	# particles	L_2 error
100	50	0.0162
	100	0.0092
	200	0.0057
	400	0.0047
200	50	0.0161
	100	0.0089
	200	0.0053
	400	0.0034

We can observe how the error decreases when increasing the number of particles and the number of grid nodes, as expected. The increase of the number of grid nodes and, consequently, the increase of the number of particles makes the computation more demanding in terms of CPU time. The convergence rate with respect to the number of particles is less than one. The SPH is second order accurate if a constant smoothing length is considered, while for the case of varying smoothing length no theories exist, to the author's best knowledge.

Figure 4.1 shows the numerical solution with 200 grid points and 400 particles. The analytical solution is well approximated during the transient.

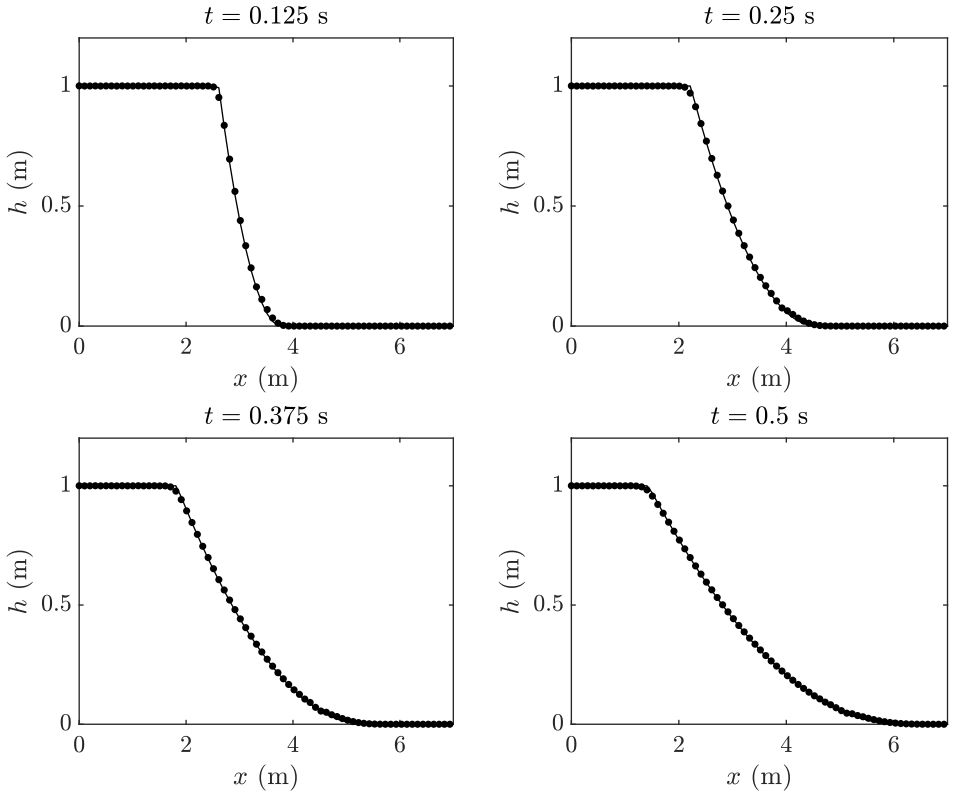


Figure 4.1: Dam break with dry bed at different times; − exact, · numerical.

The multi-layer approach is here explored by setting the number of particles at 200 (for each layer), the number of grid points at 100 (for each layer) and by varying the number of layers from 1 to 10. Results are summarized in Table 4.3: we see that the L_2 depth error

Table 4.3: Dam break with dry bed, multi-layer approach: L_2 depth errors at $t = 0.5$ s.

# layers	L_2 error
1	0.0164
5	0.0157
10	0.0158

remains almost constant, so the use of more and more layers does not change the accuracy of the result, as it can be expected in this case, because viscosity plays no role.

Wet bed

The dam break with wet bed is the equivalent of the Sod problem for the shock tube. Its solution has been found by Stoker [39] and its reported in Ref. [10, Sec. 4.1.1]. The dam break is supposed to be instantaneous, the bottom is flat and there is no friction. For the following initial conditions (where L is the length of the domain)

$$h(x,0) = \begin{cases} h_L & \text{for } 0 \leq x \leq \bar{x} \\ h_R & \text{for } \bar{x} < x \leq L \end{cases} \tag{4.5}$$

with $h_L \geq h_R$ and $u(x,0) = 0$, the analytical solution is found using the method of characteristics: at time $t \geq 0$, a left-going rarefaction wave reduces the initial depth h_L to an intermediate value h_M and a right-going shock increases h_R to the intermediate value h_M. Thus:

$$h(x,t) = \begin{cases} h_L & \text{for } 0 \leq x \leq \bar{x} - t\sqrt{gh_L} \\ \frac{4}{9g}\left(\sqrt{gh_L} - \frac{x-\bar{x}}{2t}\right)^2 & \text{for } \bar{x} - t\sqrt{gh_L} < x \leq \bar{x} + t\left(2\sqrt{gh_L} - 3c_M\right) \\ \frac{c_M^2}{g} & \text{for } \bar{x} + t\left(2\sqrt{gh_L} - 3c_M\right) < x \leq \bar{x} + t\frac{2c_M^2\left(\sqrt{gh_L} - c_M\right)}{c_M^2 - gh_R} \\ h_R & \text{for } x > \bar{x} + t\frac{2c_M^2\left(\sqrt{gh_L} - c_M\right)}{c_M^2 - gh_R} \end{cases} \tag{4.6}$$

where $c_M = \sqrt{gh_M}$ is the solution of

$$-8gh_Rc_M^2\left(\sqrt{gh_L} - c_M\right)^2 + \left(c_M^2 - gh_R\right)^2\left(c_M^2 + gh_R\right) = 0$$

The solution in term of average velocity is given by:

$$u(x,t) = \begin{cases} 0 & \text{for } 0 \leq x \leq \bar{x} - t\sqrt{gh_L} \\ \frac{2}{3}\left(\sqrt{gh_L} + \frac{x-\bar{x}}{t}\right) & \text{for } \bar{x} - t\sqrt{gh_L} < x \leq \bar{x} + t\left(2\sqrt{gh_L} - 3c_M\right) \\ 2\left(\sqrt{gh_L} - c_M\right) & \text{for } \bar{x} + t\left(2\sqrt{gh_L} - 3c_M\right) < x \leq \bar{x} + t\frac{2c_M^2\left(\sqrt{gh_L} - c_M\right)}{c_M^2 - gh_R} \\ 0 & \text{for } x > \bar{x} + t\frac{2c_M^2\left(\sqrt{gh_L} - c_M\right)}{c_M^2 - gh_R} \end{cases} \tag{4.7}$$

For the test case, we consider $L = 10$ m, $h_L = 1$ m, $h_R = 0.5$ m and $\bar{x} = 5$ m. Also in this case the solution is self-similar, and it does not depend on initial conditions if properly scaled. The input file is reported in Appendix C.

Also in this case we set wall boundaries. Here we have simulated both the case with variable and fixed smoothing length. The simulations that have been run are summarized in Table 4.4. Here only the case with 200 grid nodes has been considered, because in the 100 grid nodes case, the shock wave was too smeared whatever the number of particle used. Table 4.5 reports the L_2 errors on the non dimensional depth, defined as

$$||\text{err}_h||_2 = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\frac{h_i - h_i^{ex}}{h_i^{ex}}\right)^2} \tag{4.8}$$

Table 4.4: Dam break with wet bed: summary of simulations.

# grid nodes	# particles
200	100
	200
	400
	800

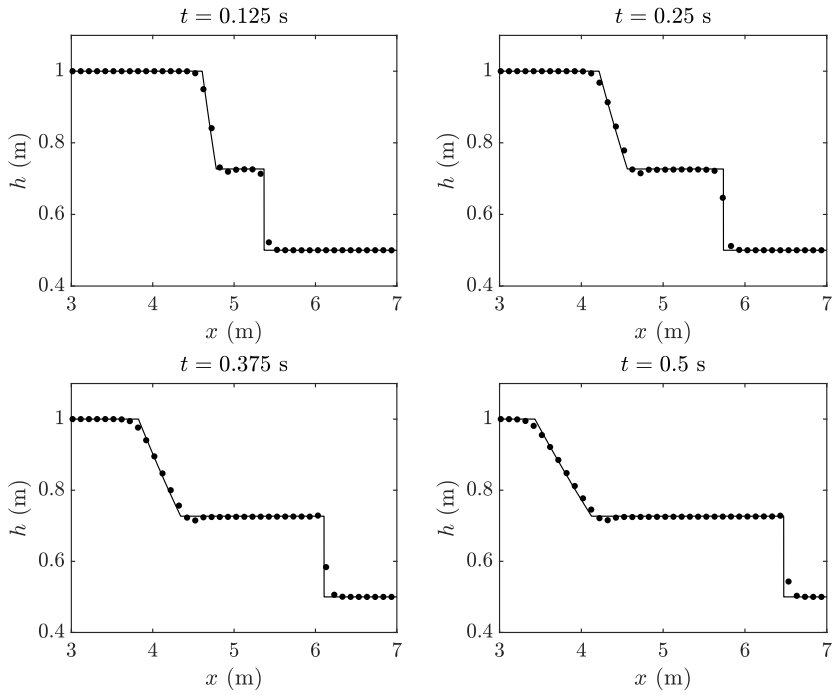
Table 4.5: Dam break with wet bed: L_2 depth errors at $t = 0.5$ s.

# grid nodes	# particles	L_2 error (depth)	
		`SLupd = .TRUE.`	`SLupd = .FALSE.`
200	100	0.0376	0.0289
	200	0.0285	0.0218
	400	0.0230	0.0163
	800	0.0205	0.0121

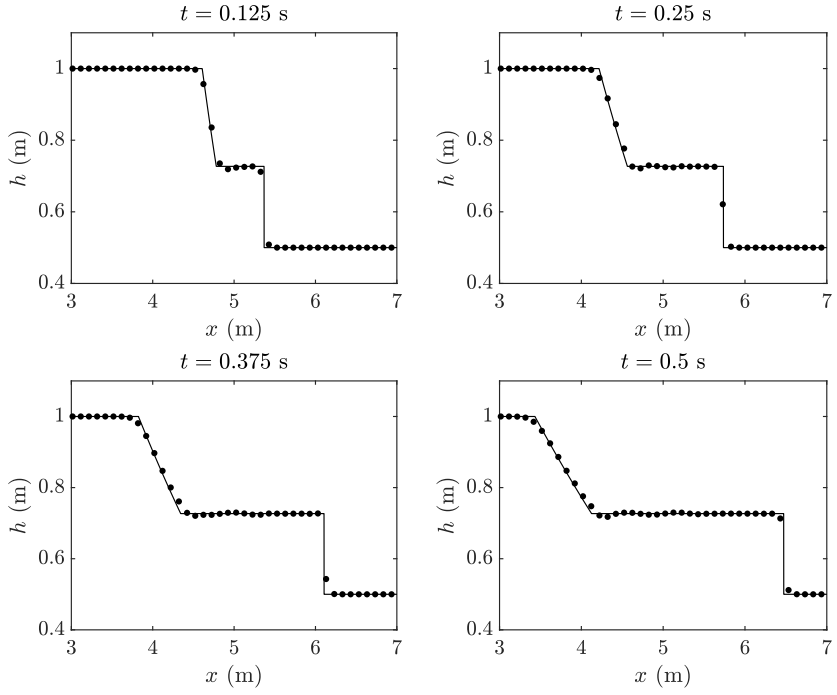
where N is the number of grid nodes, h_i and h_i^{ex} are the numerical and exact solution at the grid nodes, with variable (`SLupd = .TRUE.`) and constant (`SLupd = .FALSE.`) smoothing length:

If in the previous test case it was mandatory to have a variable smoothing length in order to compute an acceptable solution, here it seems not necessary, at a first glance. Actually, it would seem that the best results in terms of L_2 error on the non dimensional depth are obtained by imposing $l = \text{const}$. Figure 4.2 compares the two cases of variable and constant smoothing length for different times. It can be noted that in 4.3a, which is the one with the variable smoothing length, the shock is more smeared with respect to the case with constant smoothing length, shown in Figure 4.2b. Figure 4.3 shows the average velocity comparison between the case of constant and variable smoothing length.

In Figure 4.4 is shown a solution in the absence of artificial viscosity (`art_visc_coeff = 0`), with smoothing length update, 800 particles and 200 grid nodes: upstream of the shock, the solution presents some numerical oscillations, and these instability phenomena explain the introduction of an artificial viscosity. As already underlined, the choice of the artificial viscosity is suboptimal, since it depends on the coefficient α (the `art_visc_coeff` keyword in the input file), which is chosen empirically. One possible solution would be to implement the method proposed by Vila [42]: it essentially consists in a different estimation of the flux term in the momentum equation of system (2.14) by the use of a Riemann solver, which allows to avoid the introduction of an artificial viscosity term at a cost of a major implementation effort; the method would become a sort of ALE (arbitrarily Lagrangian-Eulerian) method.

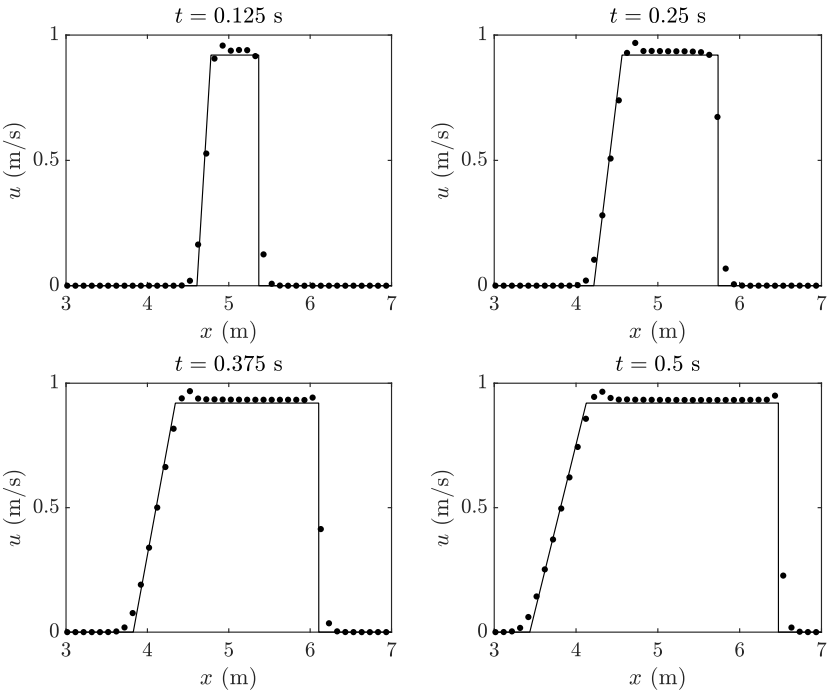


(a) Variable length update.

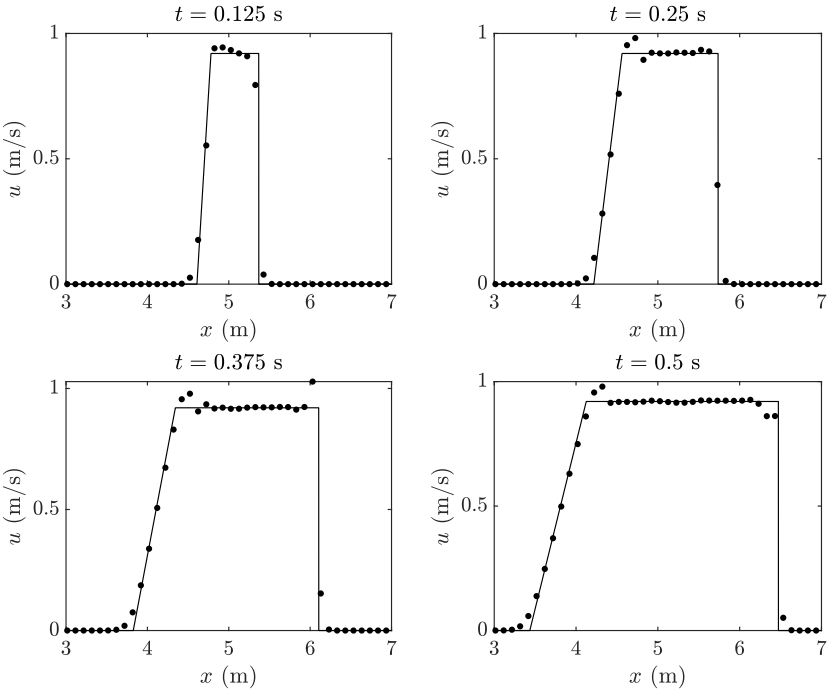


(b) Constant smoothing length.

Figure 4.2: Dam break with wet bed: depth comparison with 800 particles; — exact, • numerical.



(a) Variable smoothing length.



(b) Constant smoothing length.

Figure 4.3: Dam break with wet bed: average velocity comparison with 800 particles; − exact, · numerical.

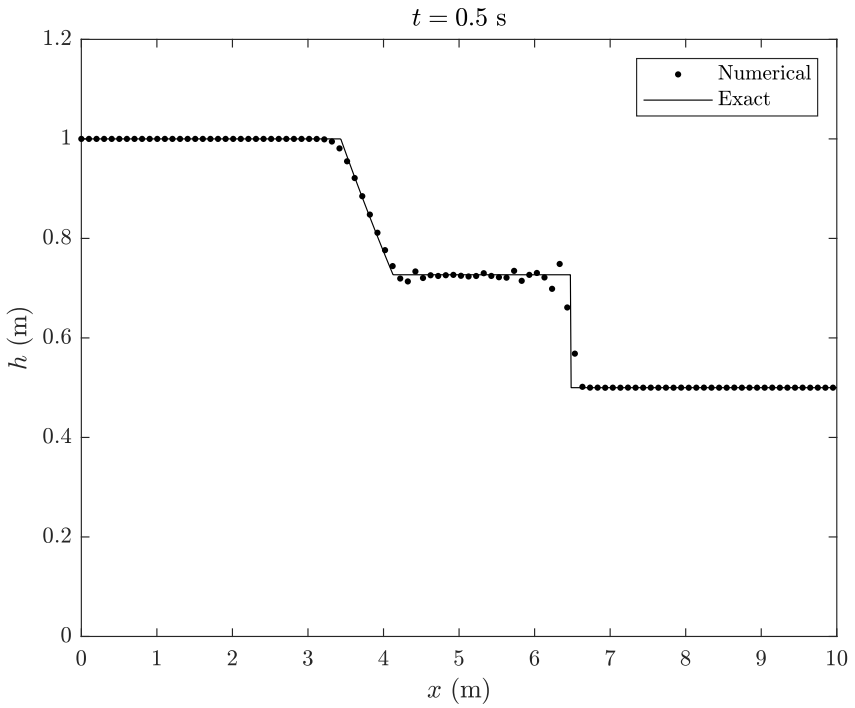


Figure 4.4: Null artificial viscosity in the dam break with wet bed: instabilities grow near the shock.

The multi-layer approach in this case is explored as above, by setting a certain number of particles and grid nodes (200 and 200 respectively) for each layer and increasing the number of layers from 1 to 10. Note that the smoothing length update is enabled. Results are summarized in Table 4.6: we see that the L_2 depth error remains almost constant, as in the previous case.

Table 4.6: Dam break with wet bed, multi-layer approach: L_2 depth errors at $t = 0.5$ s.

# layers	L_2 error
1	0.0289
5	0.0263
10	0.0262

4.1.2 Gaussian hump

The Gaussian hump is an initial disturbance of the free surface which propagates in space. From the initial Gaussian peak, two waves travel in opposite directions, with velocity

$$\sqrt{g\left(h_0 + \frac{h_d - h_0}{2}\right)}$$

where h_0 is the undisturbed depth, h_d is the disturbance peak depth and $(h_d - h_0)$ is the disturbance amplitude. The initial conditions are given by:

$$\begin{cases} h(x,0) = h_0 + (h_d - h_0)\exp\left\{-\left(\frac{x-\bar{x}}{\sigma}\right)^2\right\} & \text{for } 0 \leq x \leq L \\ u(x,0) = 0 & \text{for } 0 \leq x \leq L \end{cases} \tag{4.9}$$

In the following, we choose $h_0 = 1$ m, $h_d = 1.5$ m, the domain length $L = 10$ m, the centre of the peak $\bar{x} = 5$ m and $\sigma^2 = 0.5$ m. The input file settings are reported in Appendix C. Table 4.7 reports all the test cases that have been simulated with the associated L_2 non dimensional depth error, computed as:

$$||\text{err}_h||_2 = \sqrt{\frac{1}{N_{\text{gr}}}\sum_j \left(\frac{h_j^{\text{num}} - h_j^{\text{ref}}}{h_j^{\text{ref}}}\right)^2} \tag{4.10}$$

Since an exact solution does not exist, we use the `PyClaw` software [22] with the high order solver `SharpClaw` [21]. The reference (namely, the high order) solution has been computed using FVM with 5-th order WENO reconstruction, 4th-order strong stability preserving method as time integrator[1] and 19999 cells.

Table 4.7: Gaussian hump: L_2 depth errors

# grid nodes	# particles	L_2 error (depth) $t = 0.4$ s	L_2 error (depth) $t = 1$ s
200	200	0.0072	0.0207
	400	0.0048	0.0160
	800	0.0041	0.0138
	1600	0.0039	0.0135

We can observe how at time $t = 1$ s, the L_2 errors are higher than at time $t = 0.4$ s. This is because two shock waves develop, as shown in Figure 4.5, and the solution there cannot approximate the discontinuity perfectly. However results are quite good. Figure 4.5 shows the comparison between the numerical solution computed with the SPH method and the numerical solution computed with the higher order FV method, in the case of 1600 particles and 200 grid nodes. The computational time was as expected higher in the case of the SPH simulation: the finite volume method took ≈ 150 s, while the SPH code took ≈ 380 s with a maximum Courant number ≈ 0.47 and a fixed $\Delta t = 0.001$ s.

[1] see documentation `https://www.clawpack.org/pyclaw/solvers.html#pyclaw-sharpclaw`.

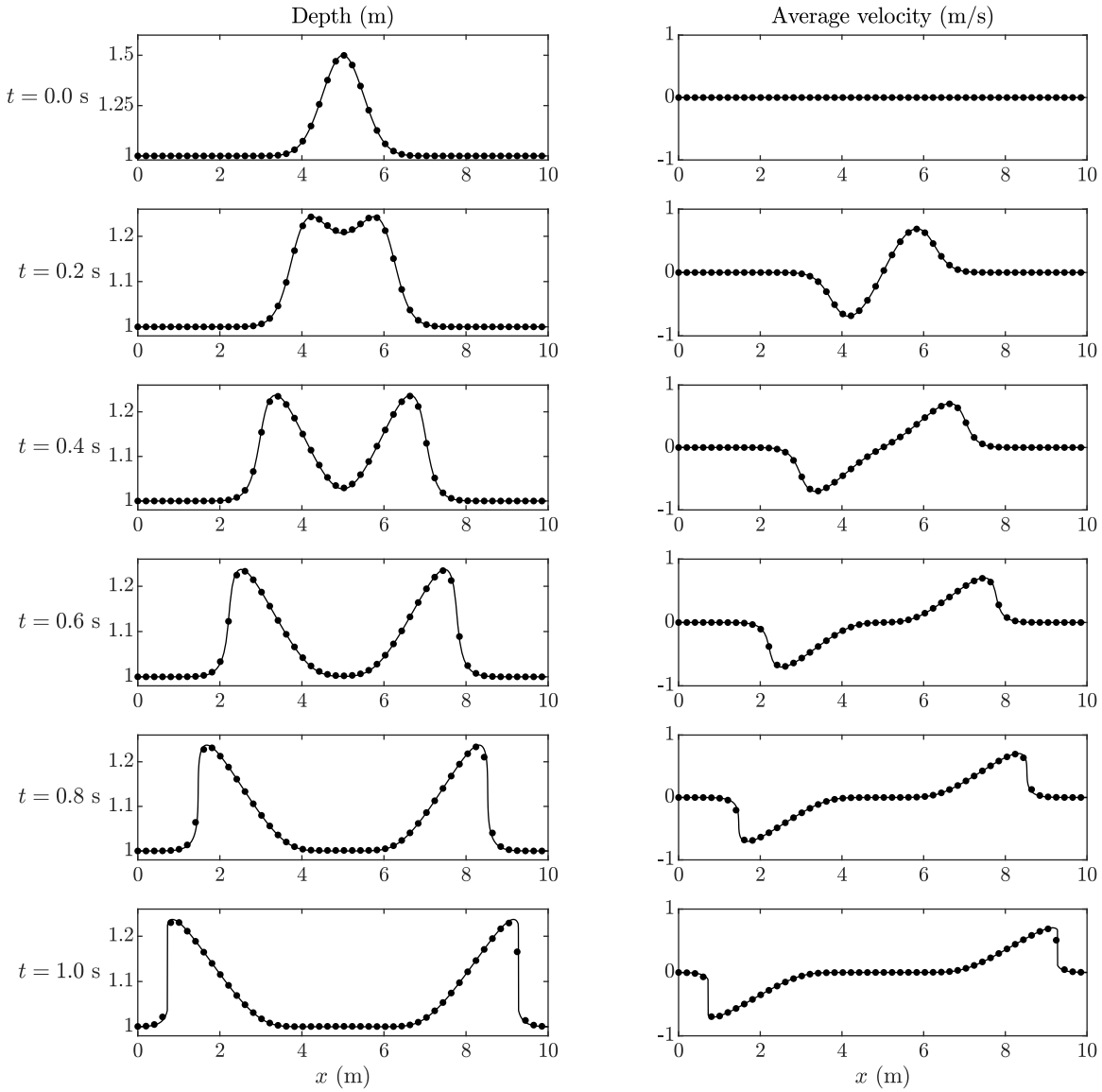


Figure 4.5: Gaussian hump: time progression of the solution (1600 particles, 200 nodes); − reference, • SPH.

Figure 4.6 shows in detail the depth solution at time $t = 1$ s, in which the two shocks are present. We can note that the SPH solution slightly oscillates near the shock, and this explains why the L_2 depth error is higher when a shock develops.

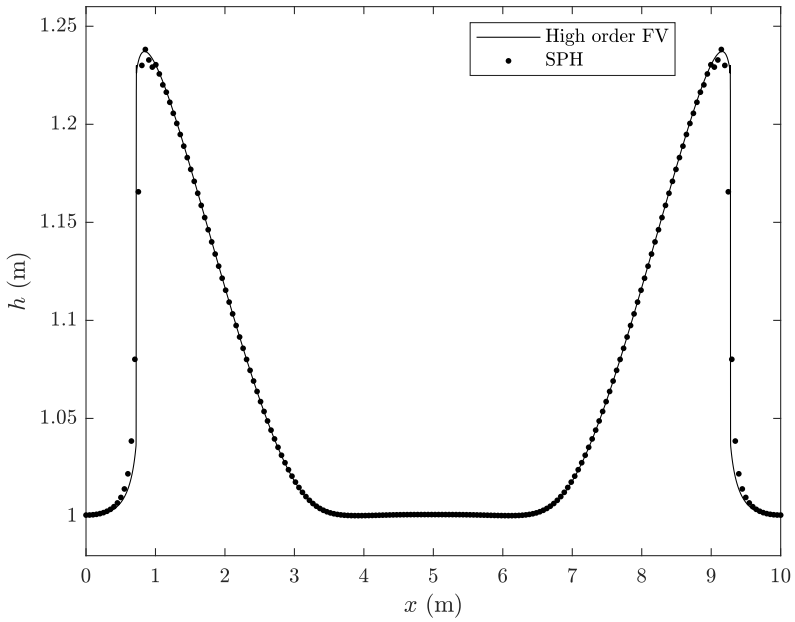


Figure 4.6: Gaussian hump: SPH solution (1600 particles, 200 nodes) vs. high order FV solution.

As already done, the multi-layer approach is tested by setting 400 particles and 200 grid nodes in each layer, and increasing the number of layers. Table 4.8 shows the results in terms of L_2 depth error against the reference solution at time $t = 0.4$ s and $t = 1.0$ s: L_2 errors are not affected by the use of more than one layer.

Table 4.8: Gaussian hump, multi-layer approach: L_2 depth errors

# layers	L_2 error (depth)	
	$t = 0.4$ s	$t = 1$ s
1	0.0048	0.0160
5	0.0063	0.0172
10	0.0062	0.0167

4.2 Viscous analysis

In the previous test cases the multi-layer formulation is not strictly needed, and we can approximate solutions in a satisfactory way also with only one layer. We have also shown that good solutions can be found considering more layers, even if this is not necessary since viscosity is null. However, our formulation finds its complete use when viscosity effects are important, so that the multi-layer model is needed in order to resolve the velocity profile.

4.2.1 Nusselt flow

The Nusselt flow is one of the simplest exact solution of the Navier-Stokes equations. Consider an inclined plate over which a a fluid is flowing under the effect of gravity without any disturbance. The solution of the 2D Navier-Stokes equations in the hypothesis of incompressible ($\boldsymbol{\nabla} \cdot \mathbf{u} = 0$), stationary ($\partial_t = 0$) and fully developed ($\partial_x = 0$) flow is given by Equations (1.14) and (1.15). This is a classical test case: for example, in Ref. [12] the authors use the weakly compressible SPH for simulating a Nusselt flow under the hypothesis of hydrostatic pressure.

If one considers periodic boundary conditions, which are the simplest in term of implementation for open boundaries, particles that leave the domain passing through an outflow boundary are instantaneously inserted on the other side (i.e. the inflow boundary), so some instabilities can grow due to particles disorder. Thus we decided to develop the inflow and outflow boundary conditions already described in Section 2.4.4, which should be much more stable for long time evolution than the periodic ones. All the results presented here are computed by using the OBP's approach.

The settings of the simulation are resumed in the input file reported in Appendix C, following the test case reported in Ref. [12, Sec. 3]. The angle of inclination β is such that $\sin\beta = 0.001$ and the length of the domain is 2 m. The thickness of falling film is 1 m, in agreement with Ref. [12, Sec. 3] in which $L = 2H$; open boundary conditions are imposed. Here the number of layers varies from 1 to 20. The simulation is run for $t = 10\,\mathrm{s}$ in order to make exchange of particles happen: in this way, we can ensure that the solution is affected only by new particles and not by the initial particles, proving that the open boundary routines work correctly. The number of particles per layer is 50, the initial distance between particles is $\Delta x = 0.0392\,\mathrm{m}$ and the number of grid points per layer is 50. The time step is fixed such that the two stability conditions (2.26) and (2.27) are satisfied: $\Delta t = 0.01\,\mathrm{s}$. Kinematic viscosity is set to $0.01\ \mathrm{m}^2/\mathrm{s}$, such that its effect can be easily appreciated. Initial conditions are such that velocity in a layer is equal to the analytical velocity value, at the inflow velocity and layer thickness are prescribed equal to the initial ones, while at the outflow a do-nothing condition is applied.

Uniform flows down an inclined plane (described by the Nusselt solution) are unstable if the inclination angle is higher than a certain threshold. The stability of the Nusselt flow can be studied with a Orr-Sommerfeld stability analysis to streamwise perturbations [20]. The stability threshold is given by the condition:

$$\mathrm{Re}_{h,\mathrm{c}} < \frac{5}{6}\cot\beta \tag{4.11}$$

where $\mathrm{Re}_{h,\mathrm{c}}$ is the critical Reynolds number computed with respect to the total depth of the flow using the average velocity reported in Equation (1.15). Here, $\sin\beta = 0.001$ and $\mathrm{Re}_{h,\mathrm{c}} \approx 800$, the actual Reynolds number is $\mathrm{Re}_h = 32.7$ and the flow is always stable to any streamwise perturbation. The flow is equivalent to a water thin film flowing down an inclined plate, whose depth is 1 mm and with an average velocity of about $3\,\mathrm{cm\,s^{-1}}$.

Constant smoothing length

Smoothing length update is not enabled here, since this is a fully developed flow and particles should remain equispaced during their movement. This allows to maintain a low computational time, without loosing accuracy, when the number of layers is high. The value of the smoothing length is set to $2\Delta x$.

Table 4.9 contains the CPU time and the L_2 error on the velocity profile at $x = L/2$, computed as:

$$||\text{err}_h||_2 = \sqrt{\frac{1}{N_{\text{gr}}} \sum_j \left(\frac{u_j^{\text{num}} - u_j^{\text{ex}}}{u_j^{\text{ex}}} \right)^2} \tag{4.12}$$

where u_j^{ex} is the averaged velocity computed with the analytical solution (1.14) and (1.15) at the grid nodes in each layer.

Table 4.9: Nusselt flow: L_2 average velocity errors at $t = 10\,\text{s}$ (constant smoothing length).

# layers	L_2 error	CPU time (s)
1	0.1232	1.77
2	0.0404	2.92
4	0.0192	5.87
6	0.0136	8.76
8	0.0104	11.66
10	0.0087	14.64
12	0.0071	17.37
14	0.0059	21.00
16	0.0049	23.39
18	0.0042	25.99
20	0.0042	29.48

The CPU time grows linearly with the number of layers. Note how, increasing the number of layers, the solution becomes more accurate. Looking at Figure 4.8 we note that the convergences rate is approximately 1. We can conclude that the multi-layer formulation is able to catch the right solution when more and more layers are considered. Figure 4.7 shows the solution when considering 16 layers, compared with the exact one. The solution is plotted at $L/2$.

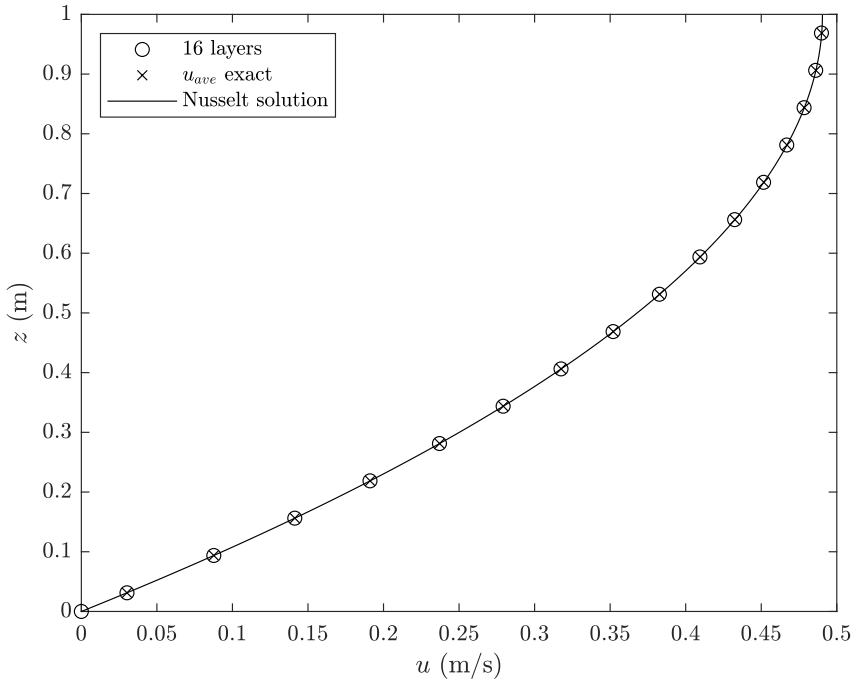


Figure 4.7: Nusselt flow (16 layers), $t = 10\,\mathrm{s}$: comparison multi-layer shallow water vs. exact in the constant smoothing length case.

Variable smoothing length

Letting the smoothing length vary allows to deal with unsteady problems. Here we report the same test case above, but with a variable smoothing length. Initially $l = 2\Delta x$ as in the previous case. Table 4.10 contains the L_2 errors on velocity profile computed as in Equation (4.12) at $x = L/2$ and the CPU time when the smoothing length is variable. We see that the accuracy in terms of the velocity L_2 error and the convergence do not depend on the smoothing length nature (constant or variable), as shown in Figure 4.8. The only drawback of the variable smoothing length approach is its more demanding computational time. However, allowing it to change is the only way to simulate unsteady phenomena.

Table 4.10: Nusselt flow: L_2 average velocity errors at $t = 10\,\mathrm{s}$ (variable smoothing length).

# layers	L_2 error	CPU time (s)
1	0.1231	2.03
2	0.0400	4.66
4	0.0190	8.71
6	0.0132	11.06
8	0.0112	15.64
10	0.0087	19.13
12	0.0070	22.53
14	0.0057	25.98
16	0.0048	29.36
18	0.0041	33.61
20	0.0035	37.03

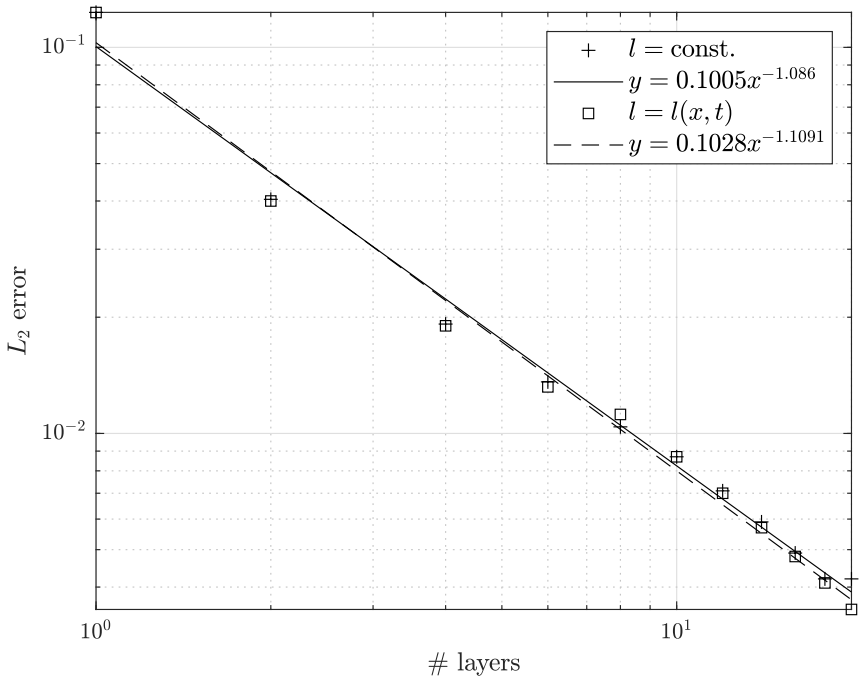


Figure 4.8: Nusselt flow: error against number of layers (constant vs. variable smoothing length.

4.2.2 Unsteady flow: roll waves

Consider the same geometry of the previous test case with the following settings:

Table 4.11: Roll waves: simulation settings.

H (m)	L (m)	β (°)	ν (Pa s)
0.2	25	4	0.006

In this simulation, the inlet boundary conditions on velocity are imposed as:

$$u(0,t) = u_{\text{Nus}}\left[1 + 0.1\sin\left(2\pi t\right)\right] \tag{4.13}$$

where u_{Nus} is the Nusselt solution given by Equation (1.14). Here 5 layers has been considered and the simulation has been run for 10 seconds. The input file for this simulation is reported in Appendix C.
Considering the settings reported in Table 4.11 and the critical condition (4.11):

$$\text{Re}_{h,\text{c}} = 11.9 < 50.6 = \text{Re}_h$$

and consequently the flow will be unstable to streamwise perturbations. A periodic perturbation at the inlet of the flow, for example the one given in Equation (4.13), will transform the flow into a system of breaking waves called *roll waves*.

In this simulation it is fundamental to employ a variable smoothing length, so that the solution in regions of low and high particles density can be found without loosing in accuracy. Results at different times are shown in Figure 4.9: the solution is qualitatively similar to a laminar roll waves train. In Section 4.3.3 we report a comparison against experimental results, showing that the multi-layer method is able to capture the experimental results with a good accuracy.

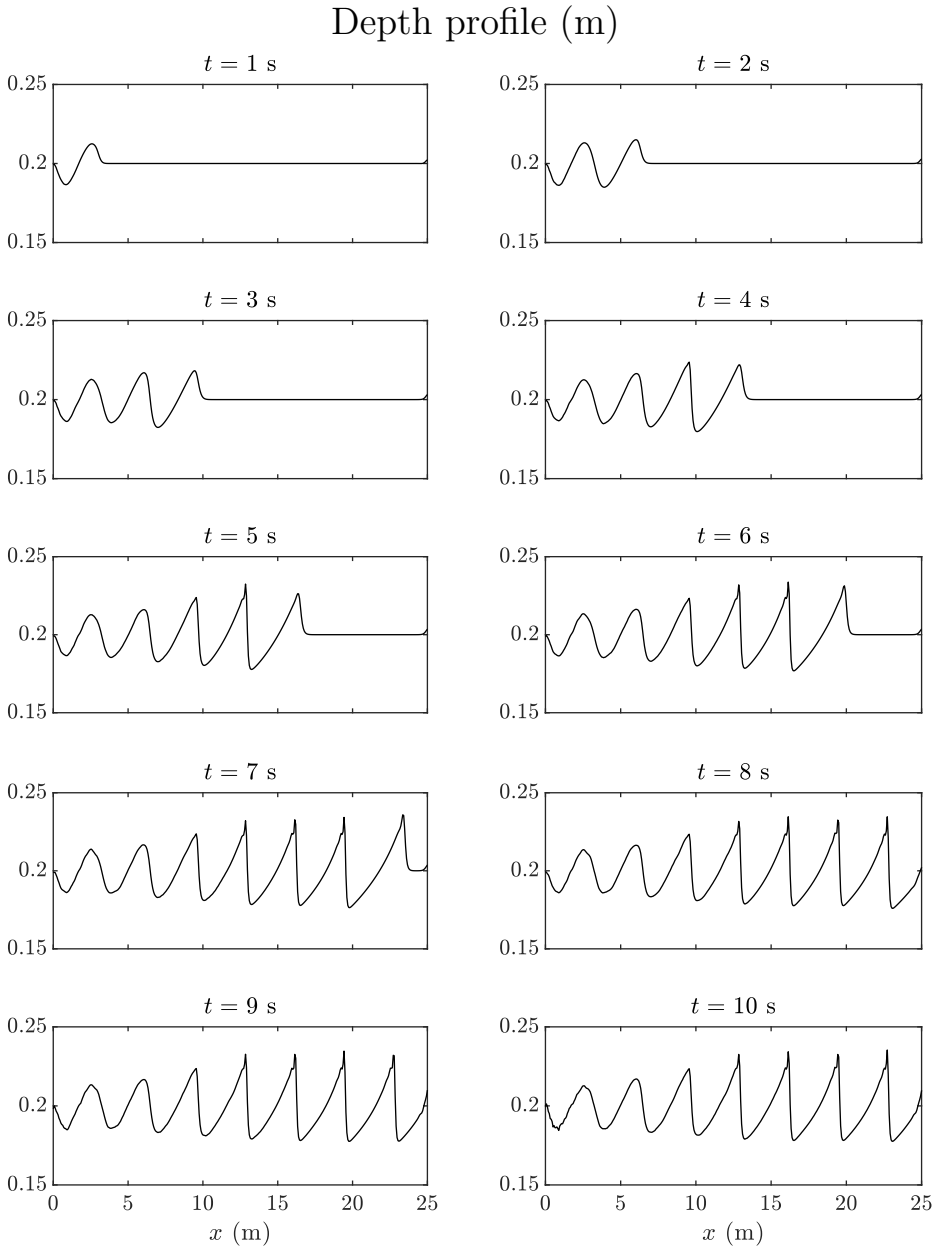


Figure 4.9: Roll waves: depth profile at different times. One can observe the disturbances growing while being advected downstream by the flow, until a shock is formed. Roll waves are well represented.

4.3 Comparison with experimental results

Having confirmed that for the Nusselt flow the multi-layer formulation gives accurate solutions in comparison to the available analytical ones, and that in the presence of viscosity the solution converges to the analytical one when the number of layer increases, we can now compare the formulation with experimental results. The following experiments have been used for the comparison:

(i) dam break with dry bed, using both a low viscosity fluid (water [23]), and a high viscosity fluid (glucose-syrup [9]);

(ii) dam break with wet bed, using water [43];

(iii) roll waves in laminar flows [15].

4.3.1 Comparison against dry bed dam break

Low viscosity test case

LaRocque et al. [23] measure velocity profiles in the case of a dam break with dry bed using water as working fluid, in a 7.31 m long and 0.18 m wide channel with a bottom slope of 0.93%. The measurements are performed through ultrasonic velocity profiling.

In the vicinity of the wet-dry front, turbulence plays an important role; citing directly Ref. [23]: "*It is commonly accepted that turbulence does not play an important role in highly transient gravity-dominated flows such as dam-break flows. In the upstream reservoir, the flow can be described by the potential flow theory. However, in the downstream part, the flow can be highly turbulent*". We need then a turbulent formulation for the multi-layer model, which can be derived from the Reynolds averaged Navier-Stokes equations (see Appendix A). Before the computation of particles' acceleration, the `turb_visc` routine is called in order to compute the turbulent viscosity through the Smagorinsky model introduced in Equation (A.5); the Smagorinsky coefficient is usually set between 0.1 and 0.2. Here we have found that $C_s = 0.2$ gives good results.

We consider the following cases:

$$h(x,0) = \begin{cases} 0.25/0.3/0.35 \text{ m} & \text{for } -3.37\,\text{m} \leq x \leq 0\,\text{m} \\ 0\,\text{m} & \text{for } 0\,\text{m} < x \leq 3.94\,\text{m} \end{cases} \tag{4.14}$$

where $x = 0\,\text{m}$ is placed at the initial position of the dam. Numerical simulation is run with settings reported in Appendix C. Note that 50 layers are employed, in order to recover the velocity profile. The CPU time for the simulation is about 120 seconds.

Figure 4.10 shows the numerical results for the water profile compared with experimental data. Figure 4.11 reports the velocity profiles upstream of the dam ($x = -0.3, -0.6\,\text{m}$) at $t = 1.25\,\text{s}$ and Figure 4.12 shows the velocity profiles at different times (x=$-0.3\,\text{m}$). We see how the multi-layer model is able to recover them with a quite good agreement

with respect to experimental results. However, for times around 2 s after the dam failure, velocity values are underestimated at $x = -0.3$ m, as one can see in Figure 4.12 and 4.13. The height profile is well approximated (also at times around 2 s), since viscosity does not play an important role except for the region of the wet-dry front, for which experimental data are not available.

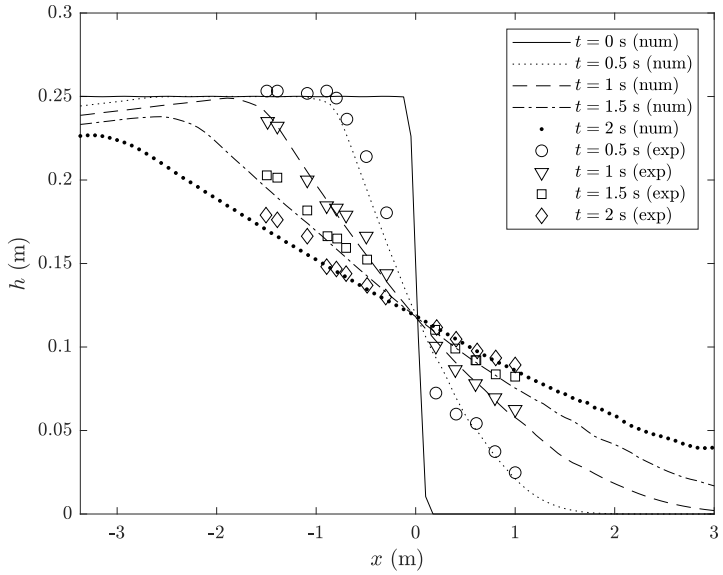
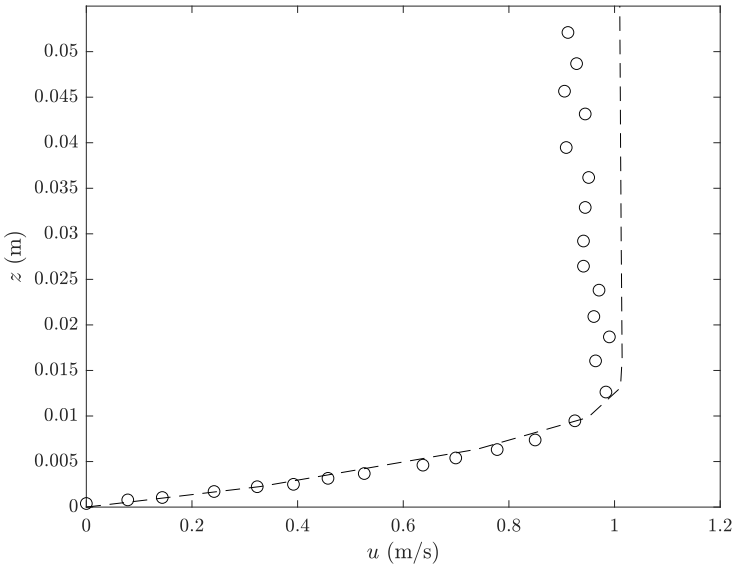
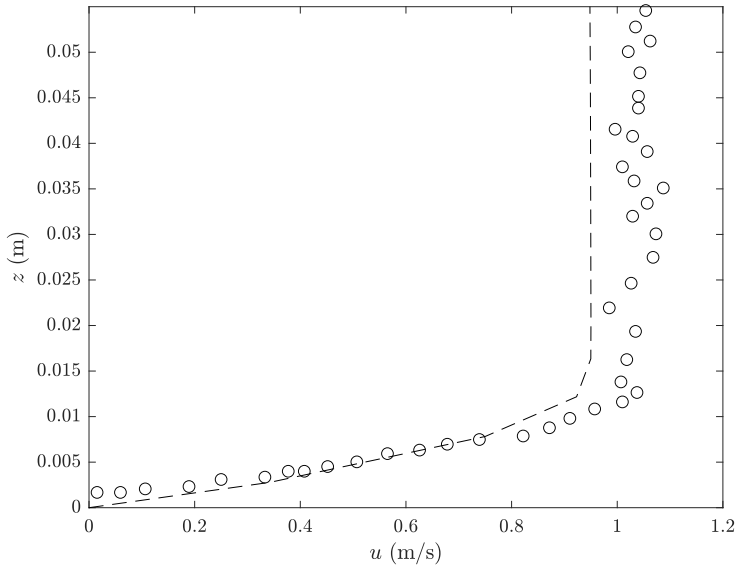


Figure 4.10: Dam break with dry bed: height profile for water, numerical vs. experimental ($h_L = 0.25$ m); the behaviour of the free surface is well approximated by the multi-layer model. Experimental results from Ref. [23, Fig. 5b].



(a) $h_L = 0.3$ m, $x = -0.3$ m (continued in next page).



(b) $h_L = 0.35\,\mathrm{m}$, $x = -0.6\,\mathrm{m}$.

Figure 4.11: Dry bed dam break: velocity profiles at time $t = 1.25\,\mathrm{s}$; experimental results from Ref. [23, Fig. 3]; − numerical, ∘ experimental. Multi-layer approach recovers the velocity profile in both cases with a certain accuracy.

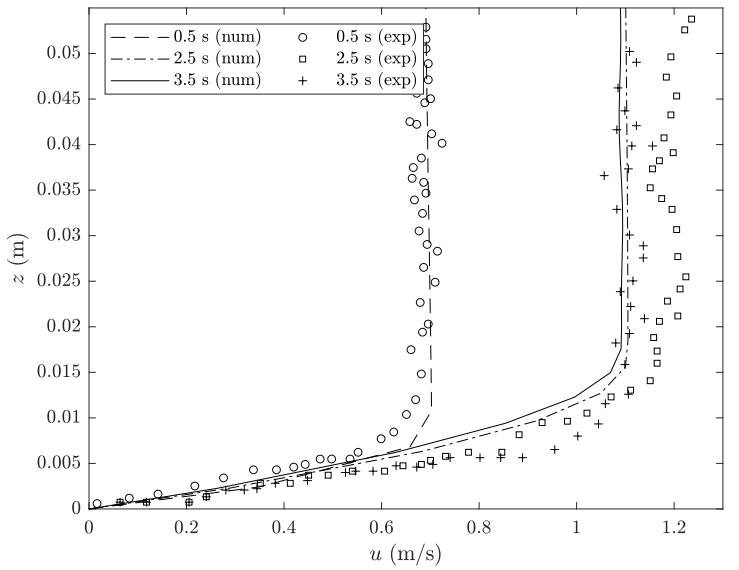


Figure 4.12: Dam break with dry bed: velocity profiles at $x = -0.3\,\mathrm{m}$ at different times ($h_L = 0.3\,\mathrm{m}$); experimental results from [23, Fig. 6a]. The velocity derivative at the wall is well approximated, but the numerical velocity profile at $t = 2.5\,\mathrm{s}$ is slightly shifted with respect to the experimental value.

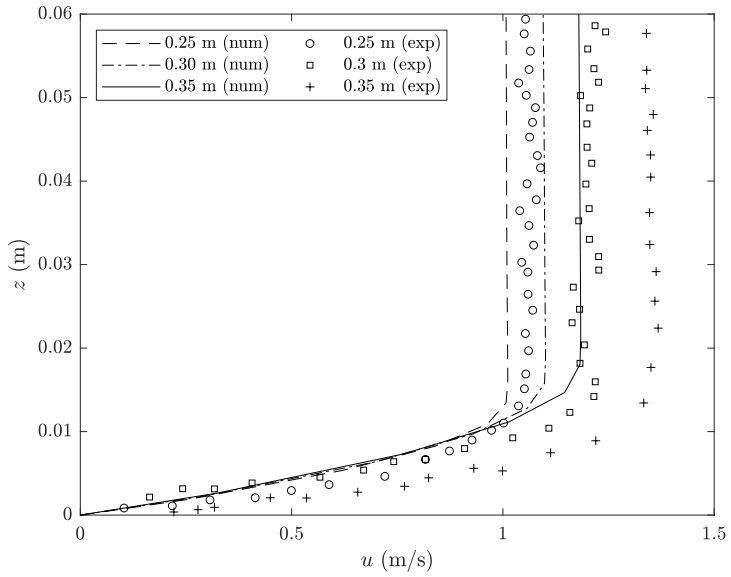
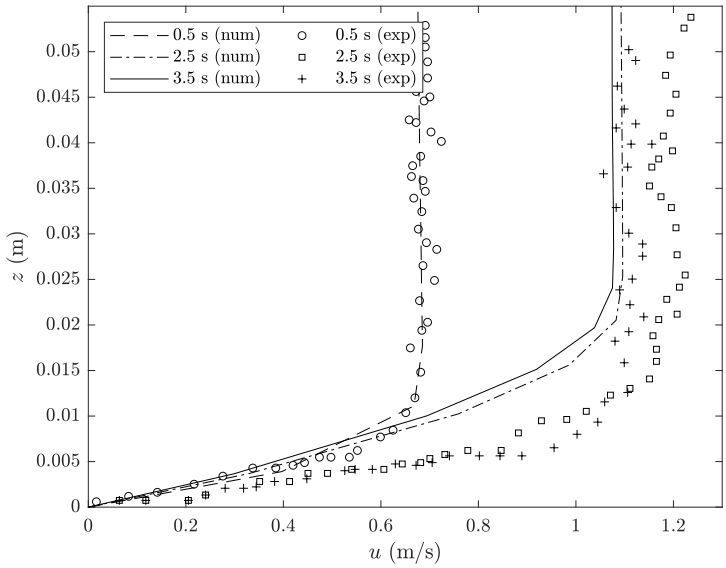
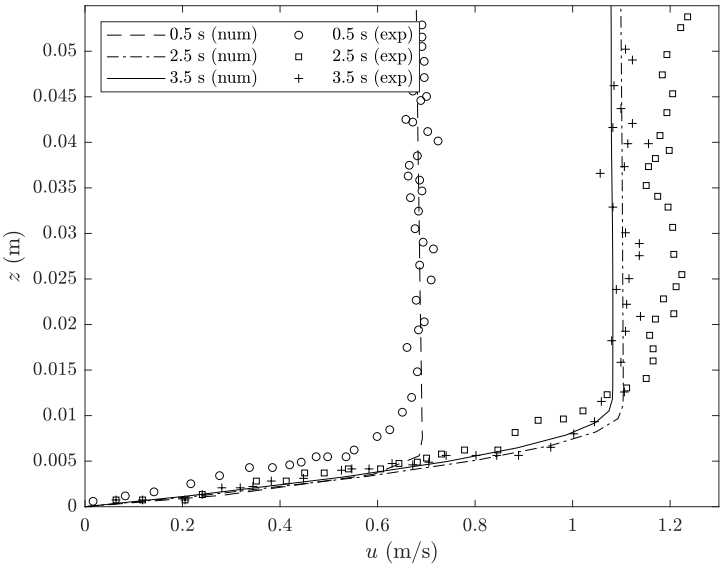


Figure 4.13: Dam break with dry bed: velocity profiles at $x = -0.3\,\mathrm{m}$ at time $t = 2.25\,\mathrm{s}$ for different h_L; experimental results from [23, Fig. 8a]. The asymptotic velocity value is slightly underestimated by the multi-layer formulation.

Citing Ref. [23]: "*The plots also show that the nondimensional shear layer thickness is approximately 0.05*", which is equivalent to $\approx 0.01\,\mathrm{m}$. Using less than 50 layers results in a bad approximation of the velocity profile, as one can see from Figure 4.14a, in which 30 layers have been used for the case of $h_L = 0.30\,\mathrm{m}$. On the other hand, when more than 50 layers are used, results are essentially unchanged, as one can see from Figure 4.14b. This comparison proves that the higher the number of layers, the higher the accuracy of the multi-layer method in approximating the velocity profile.



(a) 30 layers (continued in next page).



(b) 100 layers.

Figure 4.14: Dam break with dry bed: velocity profiles at $x = -0.3\,\mathrm{m}$ at different times ($h_L = 0.3\,\mathrm{m}$); experimental results from [23, Fig. 6a]. Left: 30 layers; right: 100 layers.

High viscosity test case

Debiane [9] presented a set of experimental data for laminar dam break in horizontal and inclined channels, using a 3 m long and 0.3 m wide channel and glucose-syrup solutions with dynamic viscosities between 12 and 170 Pa s. The dam break is simulated by raising a gate in the quickest way possible, in order to avoid flow disturbances. However, for the highest viscosities, Debiane found that the flow is influenced by the opening of the gate, and by resistance side effects, so that experimental results are not reliable enough. We then choose to simulate the cases with viscosity of 85 Pa s, reported as Glus. 9 in Ref. [9].

Numerical simulations are performed using the multi-layer approach, in order to catch the effects of viscosity. The parameters of the experiment are:

Table 4.12: Glus. 9 from [9]

H (m)	L (m)	β (°)	ρ (kg/m^3)	μ (Pa s)
0.054	0.44	0	1413	85

where μ is the dynamic viscosity, ρ the density, H is the initial height of the fluid and L the length of the reservoir. Figure 4.16 shows the comparison between experiment and numerical results in terms of the depth profile in the non dimensional variables:

$$\hat{h} = \frac{h}{H}, \qquad \hat{x} = \frac{x}{L}, \qquad \hat{t} = \frac{\rho g H^3}{12 \mu L^2} t$$

The simulation has been run with 5 layers, 200 particles per layer and 100 grid nodes per layer; non dimensional time $\hat{t} = 0.55$ is equivalent to $t = 49.8\,\mathrm{s}$ and the CPU time for simulating 50 s is more than 4 hours with $\Delta t = 2.5 \times 10^{-4}\,\mathrm{s}$. For this reason we cannot increase too much the number of layers. The input file is reported in the Appendix C.

The Reynolds number of the flow is very low, in this case. Being $L \sim 0.1\,\mathrm{m}$, $\nu \sim 0.01\,\mathrm{m}^2/\mathrm{s}$ and knowing from experimental data that $U \sim 0.001\,\mathrm{m\,s}^{-1}$:

$$\mathrm{Re}_L = \frac{UL}{\nu} \approx 0.01$$

This Reynolds number does not fulfill the hypothesis of the multi-layer model, for which Re_L should be of order ϵ^{-1} or higher. Furthermore, our multi-layer formulation does not allow mass exchange between layers, meaning that, since lower layers will have very low velocity in order to enforce the no-slip condition at the bottom, they will remain still in their initial condition. This can be seen in Figure 4.15, where interfaces between layers are shown at time $t = 49.8\,\mathrm{s}$: the first layer front is at $\hat{x} \approx 0.2$, while the fifth layer front is at $\hat{x} \approx 0.8$. The velocity derivative that appears in the source term S in the momentum equation in system (2.14) is therefore not well estimated. In the multi-layer formulation with mass exchange between adjacent layers, the thickness of every layer is everywhere a fixed fraction of the total thickness and in this way the previous issue can be overcome. However, we shall underline that the problem under consideration does not fulfill the fundamental hypothesis for using the first order approximation of the Navier-Stokes equations, and in principle the full Navier-Stokes equations have to be used.

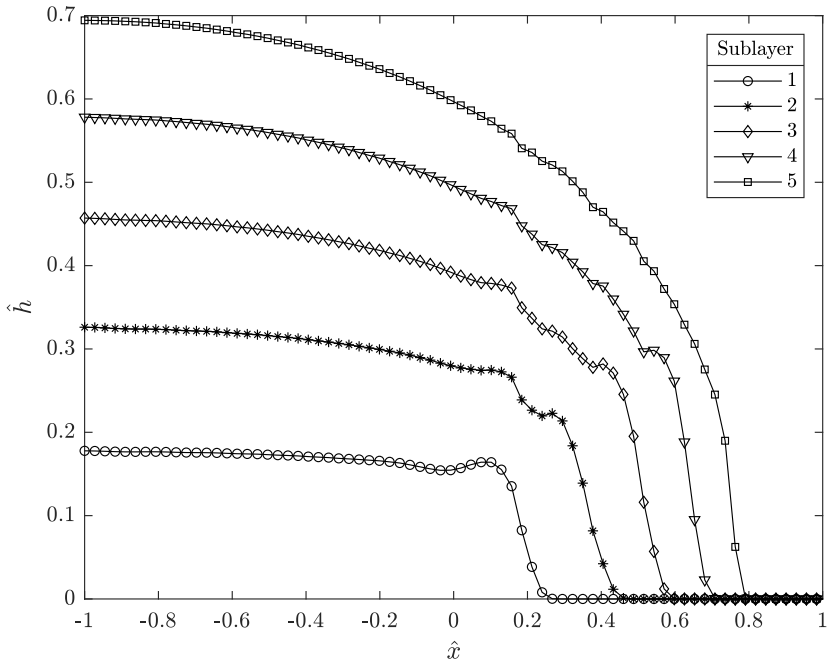


Figure 4.15: Dam break with dry bed (numerical result): sub-layers representation at time $t = 49.8\,\mathrm{s}$ for the case reported in Ref. [9].

The front position is slightly overestimated by the numerical method as shown in Figure 4.15. However, considering the above observations the agreement is good, being the maximum error on the wet-dry front around 25%. Overall, the experimental solution is well approximated in the upstream region, as one can observe in Figure 4.16.

Better results are expected in that cases in which a wet-dry front does not exist and the Reynolds number is high enough to fulfill the hypothesis of the multi-layer model. In the next section we present an experimental case of dam break with wet bed.

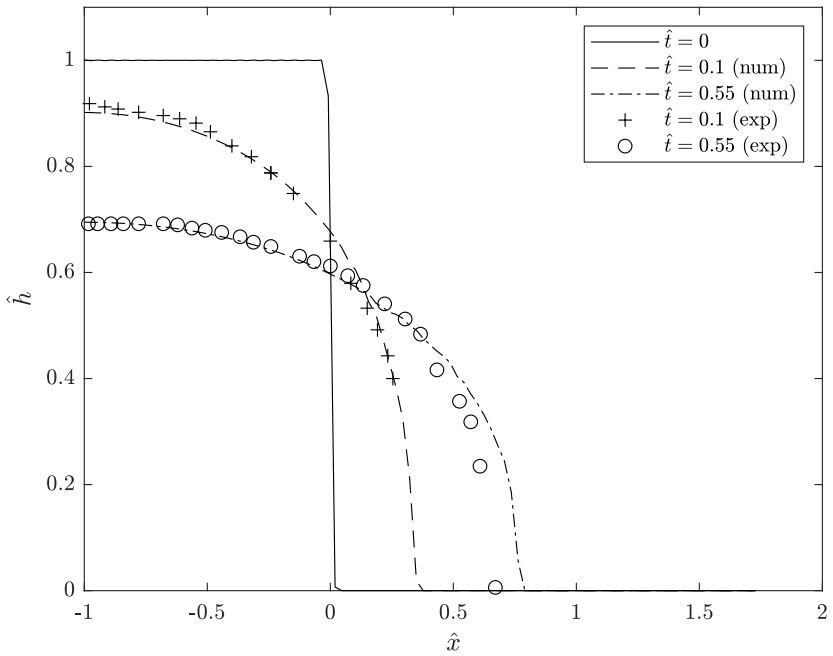


Figure 4.16: Dam break with dry bed: height profile, experimental vs. numerical results (non dimensional variables); results from Ref. [9].

4.3.2 Comparison against wet bed dam break

As stated above, removing the wet-dry front, we expect a better accuracy of the method in comparison to experiments. We will refer to Wang et al. [43], in which the authors focus on the self-similarity nature of the solution for the wet bed dam break by using both numerical simulations and laboratory tests with water as working fluid. The closure model the authors employ in the paper for the solution of the Navier-Stokes equations is the large eddy simulation. We will compare experimental results with numerical ones using the less demanding multi-layer formulation with turbulent viscosity.

Referring to Equation (4.5), the initial conditions here considered are:

$$h(x,0) = \begin{cases} 0.4\,\mathrm{m} & \text{for } -8.37\,\mathrm{m} \leq x \leq 0\,\mathrm{m} \\ 0.08/0.12/0.2/0.28\,\mathrm{m} & \text{for } 0\,\mathrm{m} < x \leq 9.63\,\mathrm{m} \end{cases} \tag{4.15}$$

These cases are reported as $\alpha = h_R/h_L = 0.2, 0.3, 0.5, 0.7$ in Ref. [43]. The Reynolds number of the flow is high enough for the development of turbulent structures. A rough estimate gives:

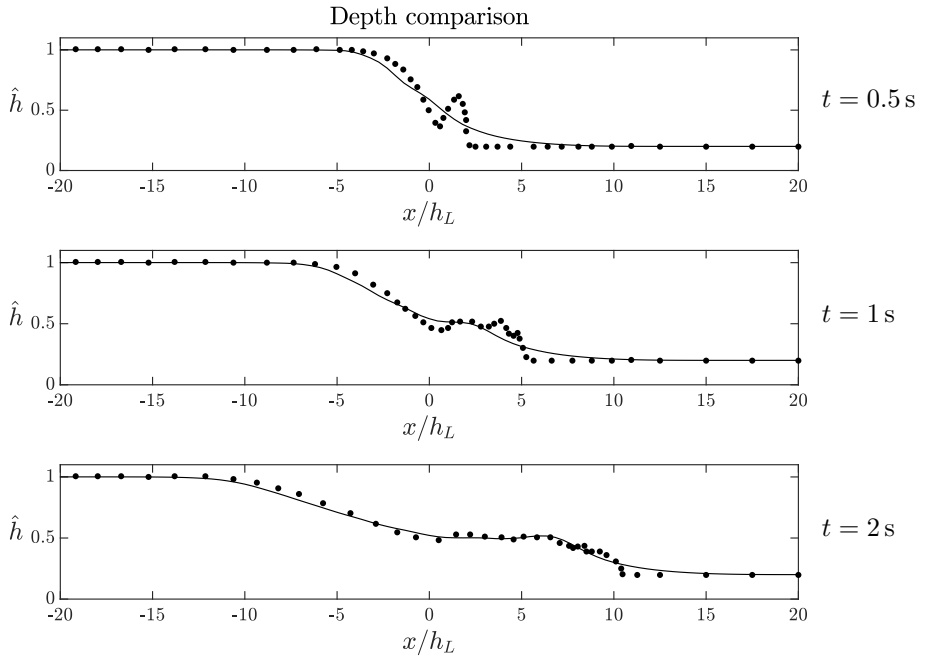
$$\mathrm{Re}_H = \frac{UH}{\nu} \approx 10^5$$

with $\nu = 1 \times 10^{-6}\,\mathrm{m}^2/\mathrm{s}$, $U = 1\,\mathrm{m\,s}^{-1}$ and $H = 0.1\,\mathrm{m}$. Thus, also in this case, turbulence modelling is needed, in order to recover the velocity profiles correctly. We choose a Smagorinsky coefficient $C_s = 0.2$. Note that the presence of an additional viscosity smears the depth profile in the correspondence of the shock that one can typically find in the dam break with wet bed. However the velocity of propagation of the shock is not influenced.

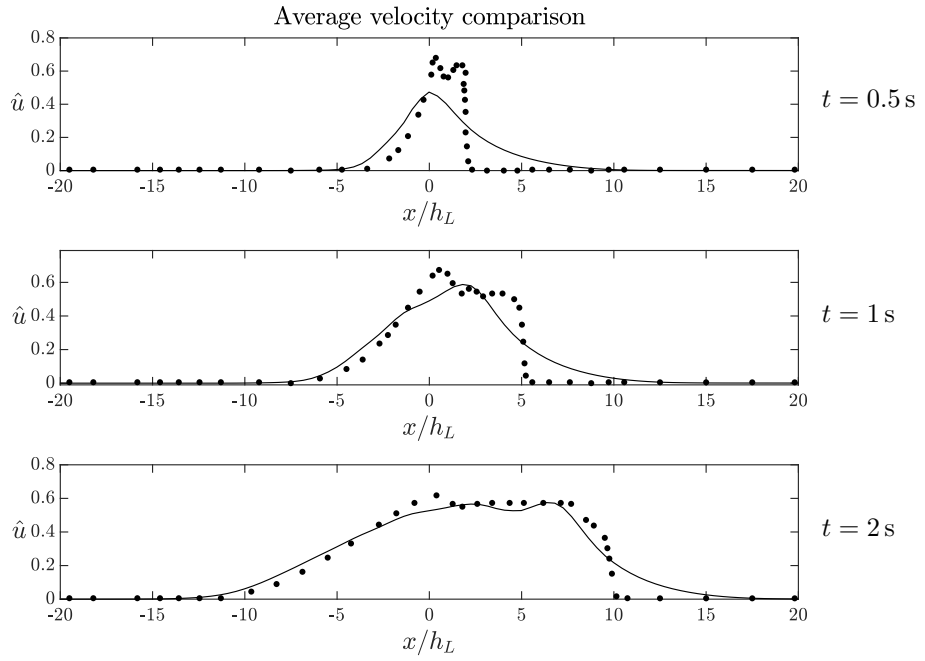
The input file is reported in Appendix C. The smoothing length is allowed to vary; the initial smoothing length is set to $2\Delta x$, where Δx is the initial particle spacing. In this test case, using 20 layers and 100 particles per layer results in a CPU time of about 80 s.

Experimental and numerical results in terms of non dimensional depth $\hat{h} = h/h_L$ and average velocity $\hat{u} = u/\sqrt{gh_L}$ for different α are shown in Figure 4.17, 4.18, 4.19 and 4.20. The dam is located at $\hat{x} = 0$. We note that in general numerical results are smoother than experimental ones. The experimental steep front in the numerical solution is smeared, and oscillations that occur in experimental data are not well captured by the multi-layer formulation. An increase in the number of particles and grid nodes with the same number of layers results in a better approximation of the shock: for example, Figure 4.21 shows the non dimensional water depth and average velocity when the number of particles is 200 (per layer) and the number of grid nodes is 200 (per layer). Oscillations close to the shock are a result of turbulent effects in that region; they are smoothed by the multi-layer approach, which is a first order approximation of the full Navier-Stokes equations. However, the moving front position and speed are always in good agreement with the experimental results. This is in agreement with the dam break case simulated with the multi-layer approach by Audusse et al. [2, Sec. 4.2]. When α grows, that is when the water jump is lower, turbulent effects close to the moving front are less important and the multi-layer formulation can approximate the solution slightly better, as shown in Figure 4.20.

The multi-layer approach allows to recover the velocity profile. Figure 4.22a, 4.22b, 4.23a and 4.23b report the velocity profiles at different positions at time $t = 2.83\,\mathrm{s}$, compared with numerical results from Ref. [43] found solving the full Navier-Stokes equations using LES model for turbulence. The accordance is generally good.

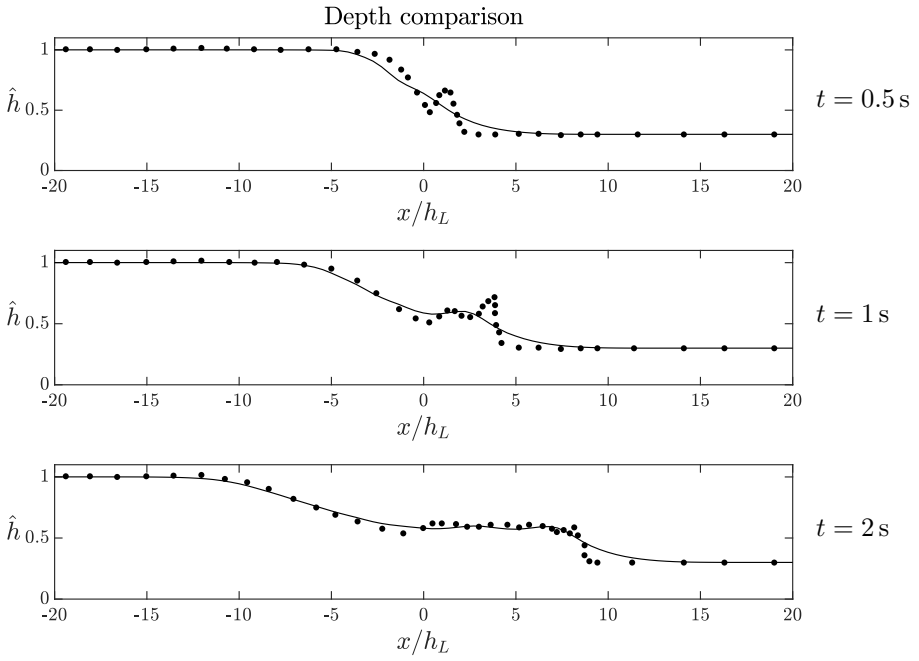


(a) Non dimensional water depth.

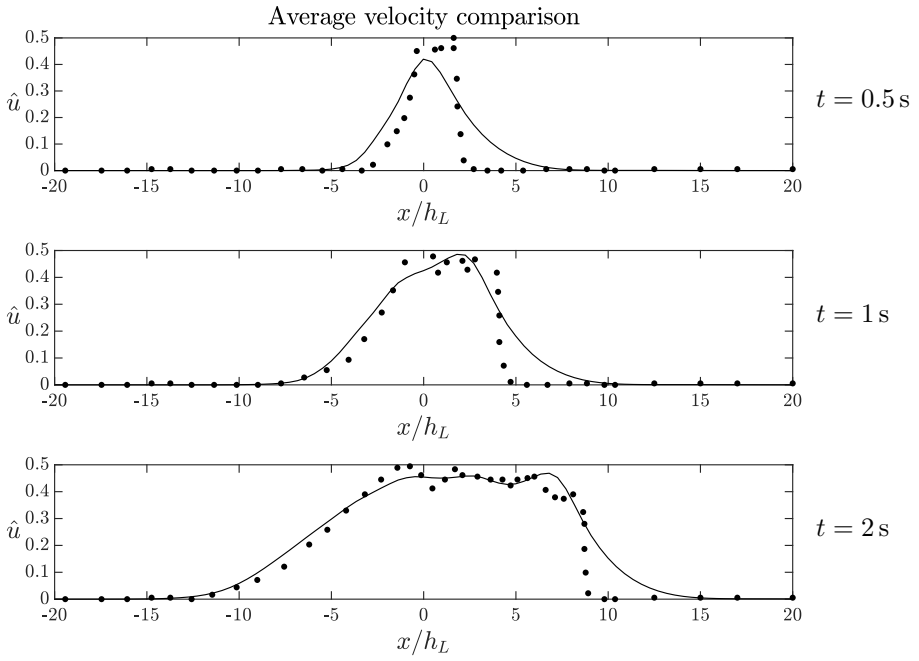


(b) Non dimensional average velocity.

Figure 4.17: Dam break with wet bed ($\alpha = 0.2$): experimental ($\cdot$), numerical ($-$).

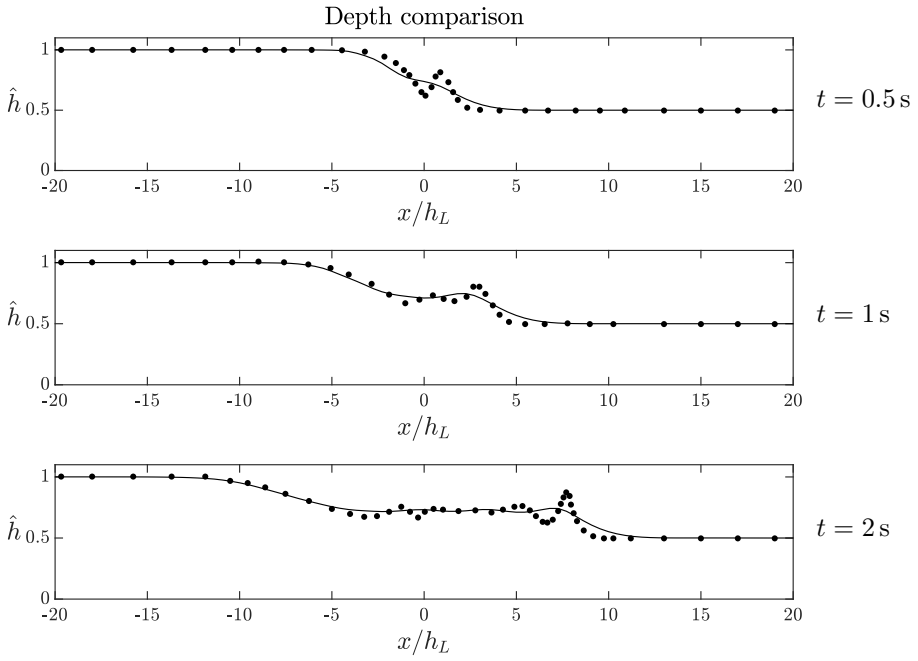


(a) Non dimensional water depth

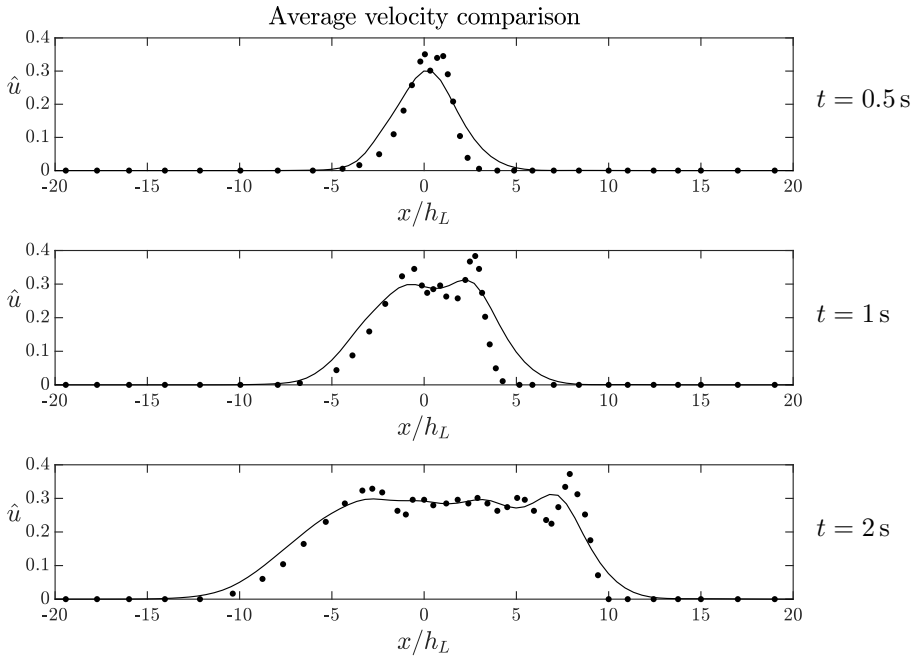


(b) Non dimensional average velocity.

Figure 4.18: Dam break with wet bed ($\alpha = 0.3$): experimental ($\cdot$), numerical ($-$).

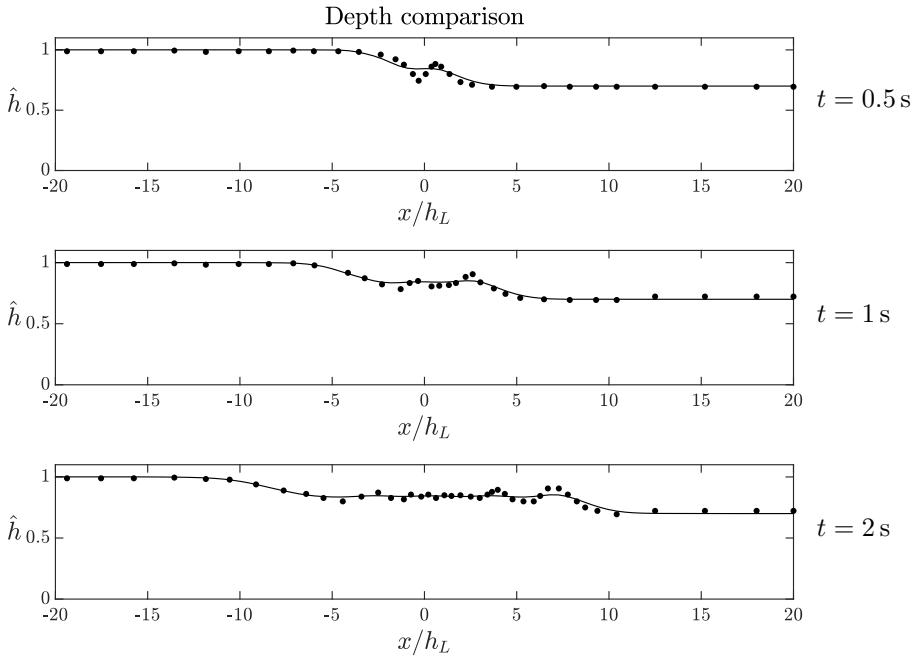


(a) Non dimensional water depth.

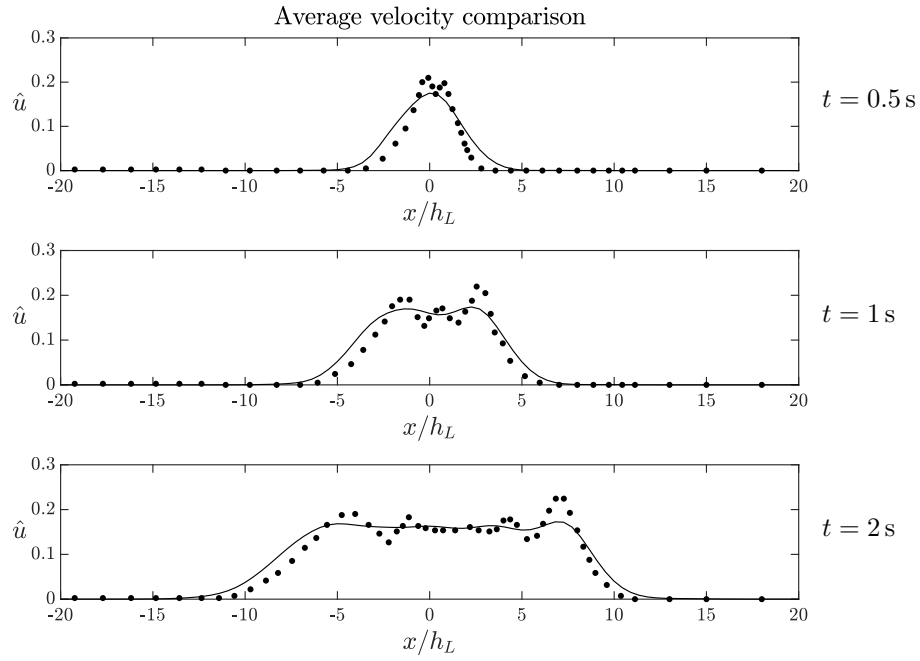


(b) Non dimensional average velocity.

Figure 4.19: Dam break with wet bed ($\alpha = 0.5$): experimental ($\cdot$), numerical ($-$).

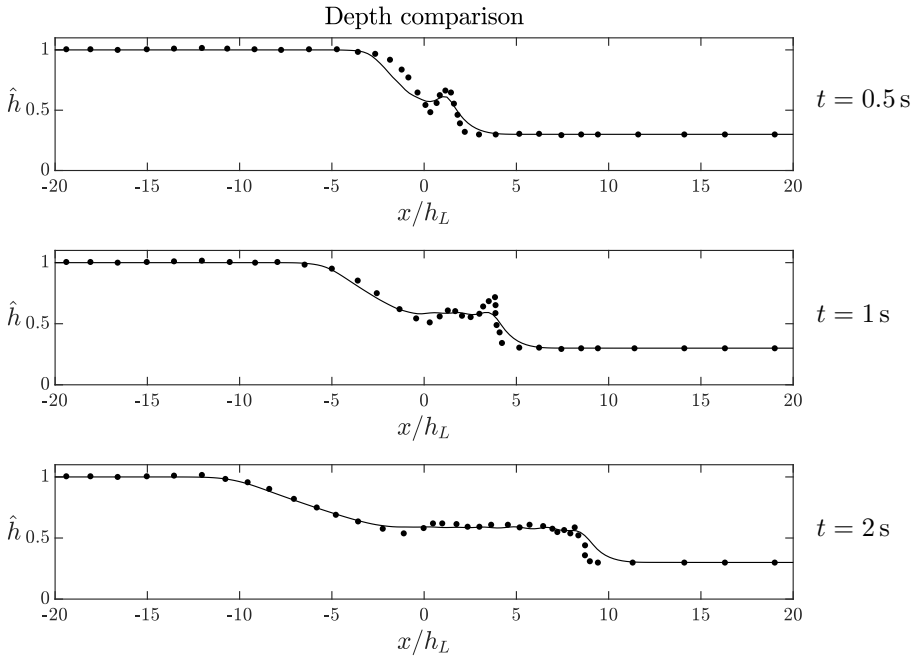


(a) Non dimensional water depth.

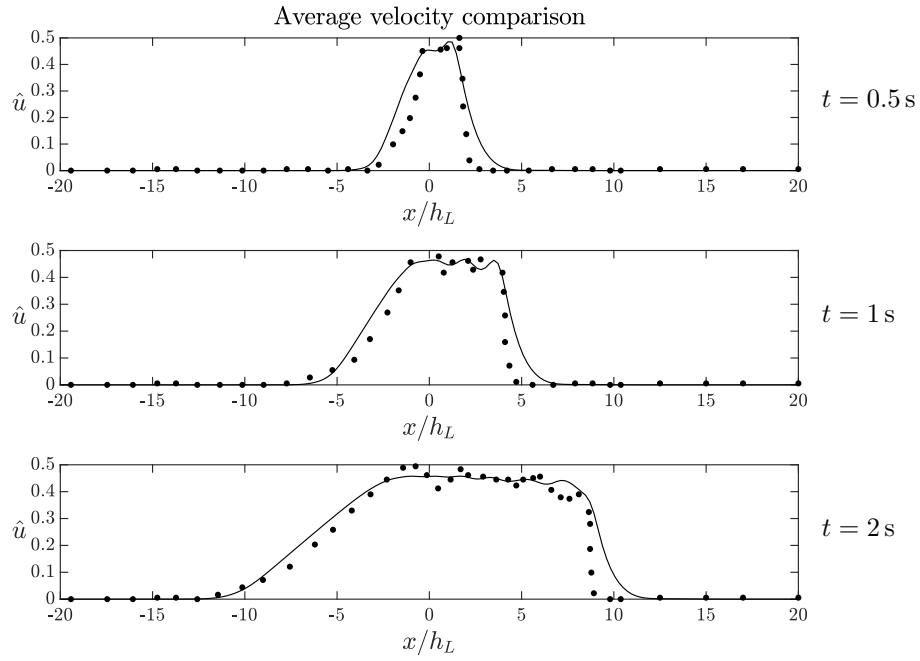


(b) Non dimensional average velocity.

Figure 4.20: Dam break with wet bed ($\alpha = 0.7$): experimental ($\cdot$), numerical ($-$).

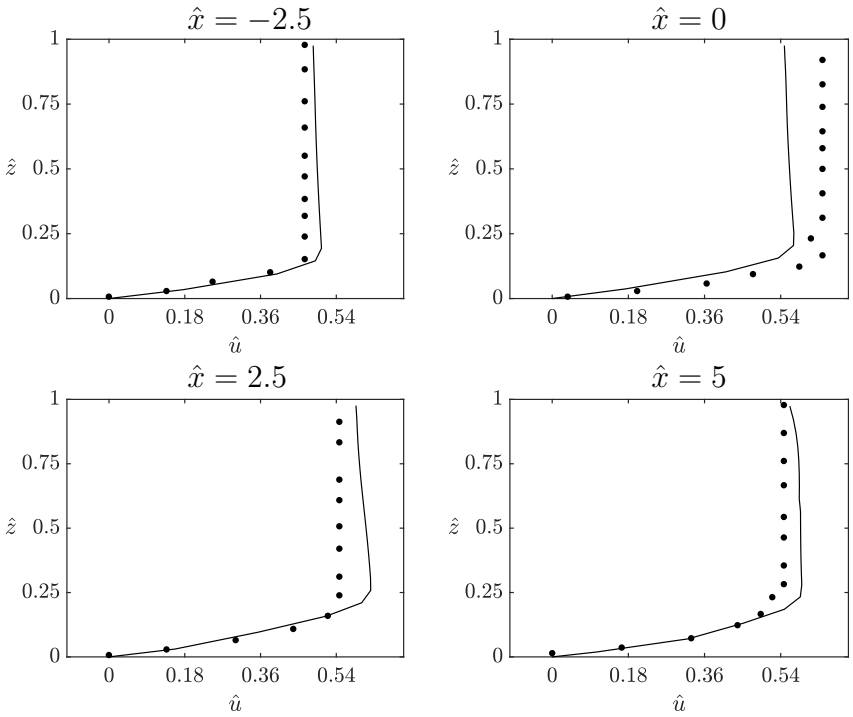


(a) Non dimensional water depth.

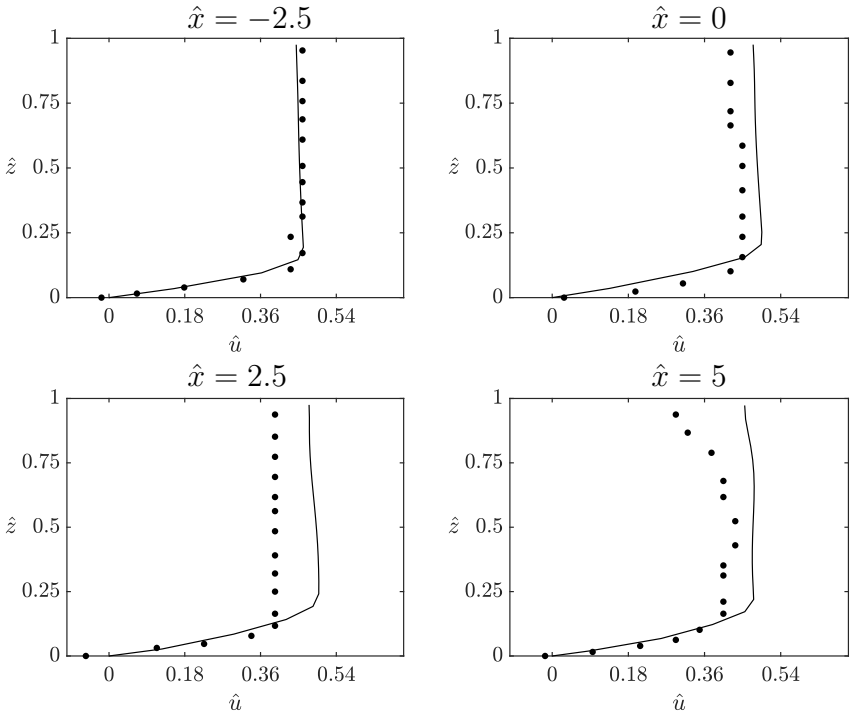


(b) Non dimensional average velocity.

Figure 4.21: Dam break with wet bed ($\alpha = 0.3$), 200 particles and 200 grid nodes per layer: experimental ($\cdot$), numerical ($-$). Solutions are more resolved than the ones in Figure 4.18a.

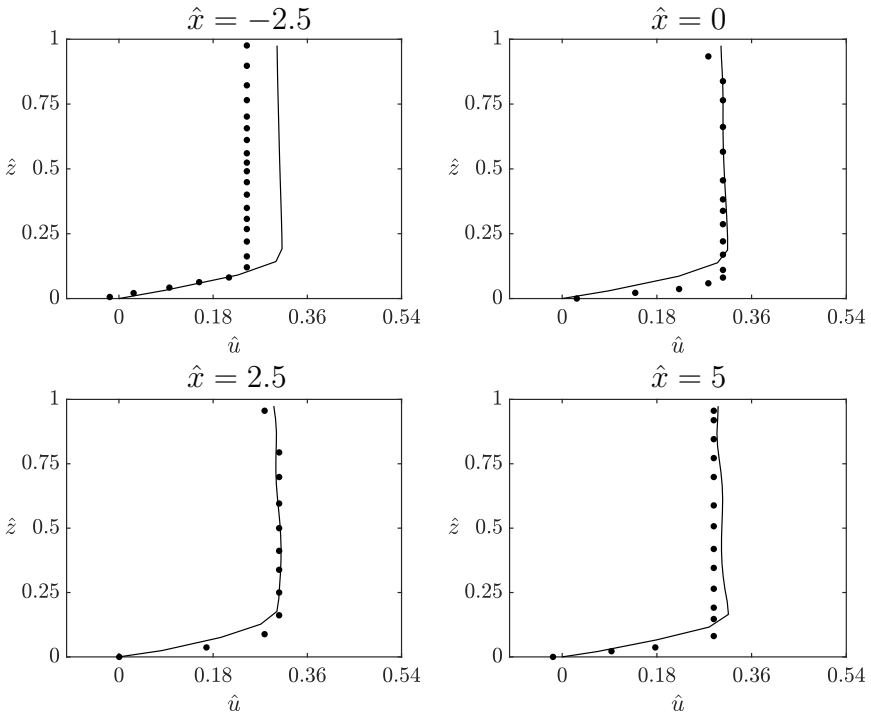


(a) $\alpha = 0.2$.

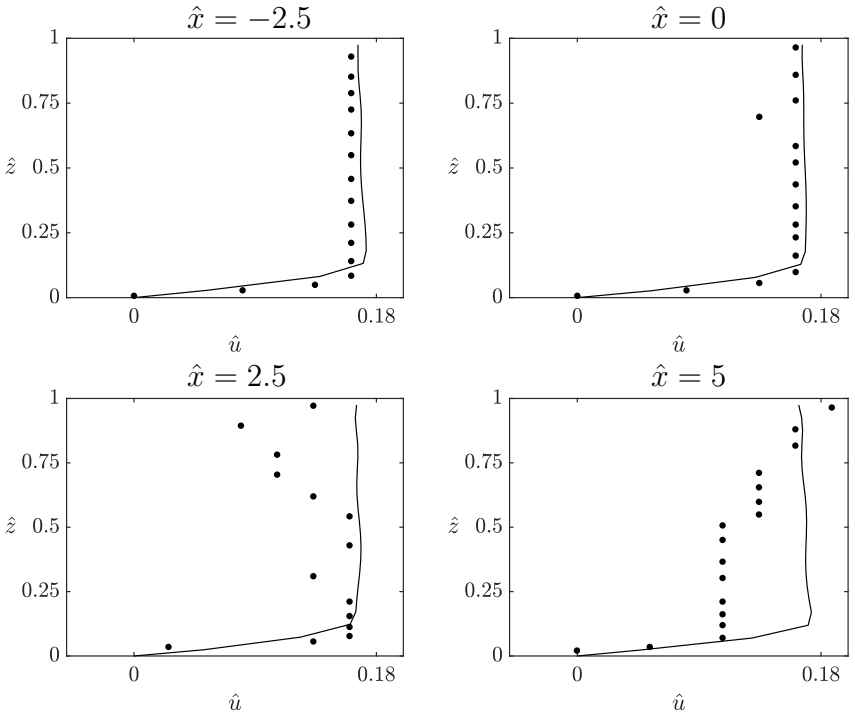


(b) $\alpha = 0.3$.

Figure 4.22: Dam break with wet bed: velocity profile at different positions; − multi-layer, · full NS.



(a) $\alpha = 0.5$



(b) $\alpha = 0.7$

Figure 4.23: Dam break with wet bed: velocity profileat different positions; − multi-layer, · full NS.

4.3.3 Laminar roll waves

Fiorot et al. [15] detail a methodology for measuring roll waves in laminar viscous flows. In their apparatus the stationary flow (Nusselt flow) is perturbed at the inlet with a periodic disturbance. If the Reynolds number exceeds the critical value above which the flow becomes unstable to certain perturbations, they are amplificated and a roll waves train grows. The critical Reynolds number is given by Equation (4.11), and by substitution in Equation (1.16) one can find the critical Froude number $\mathrm{Fr_c}$:

$$\mathrm{Fr_c} = \sqrt{\frac{\epsilon \mathrm{Re}_{L,\mathrm{c}}}{3} \sin\beta} = \sqrt{\frac{\mathrm{Re}_{h,\mathrm{c}}}{3} \sin\beta} = \sqrt{\frac{5}{18} \cos\beta}$$

which, for $\beta \ll 1$ is:

$$\mathrm{Fr_c} = \sqrt{\frac{5}{18}} \approx 0.53 \tag{4.16}$$

This result is valid for laminar flow in wide amplitude channel.

The authors study three different test cases, resumed in Table 4.13. The angle of inclination is fixed at $\beta = 8°$, the viscosity of the working fluid (glycerin) is $\mu = 212\,\mathrm{mPa\,s}$ and its density $\rho = 1273\,\mathrm{kg/m^3}$.

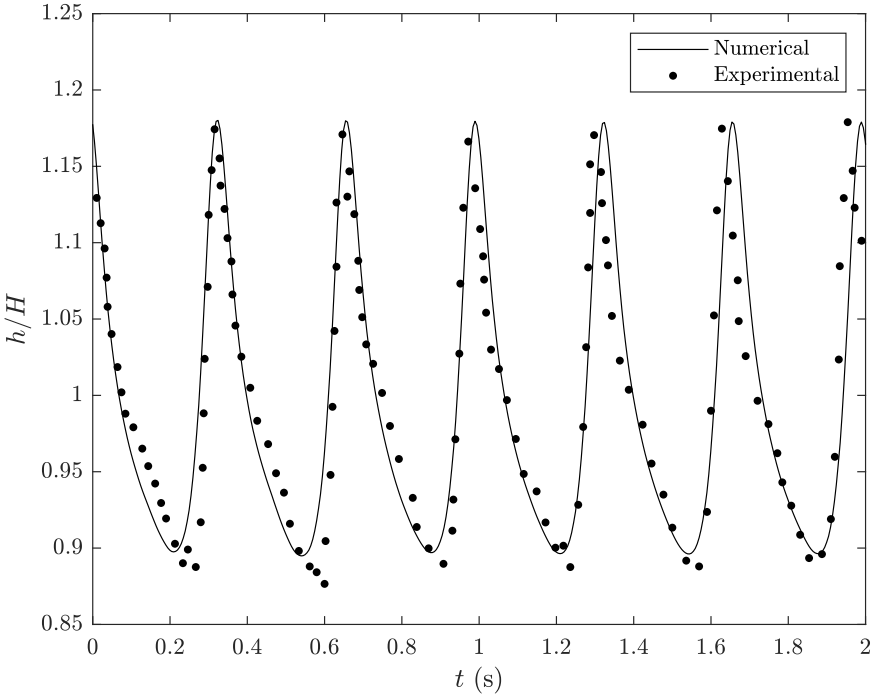
Table 4.13: Roll waves: simulation settings.

Test case	H (mm)	u_{ave} ($\mathrm{m\,s^{-1}}$)	Re_h	Fr	Figure
(a)	9.83	0.26	15	0.83	4.24a
(b)	10.73	0.3	19	0.92	4.24b
(c)	11.57	0.33	23	0.98	4.24c

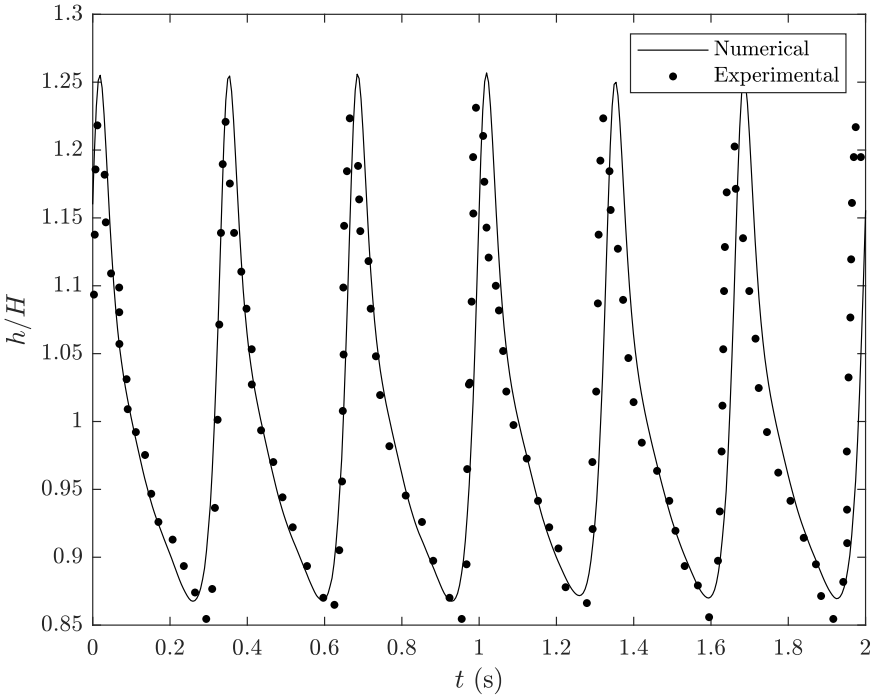
The measuring point is located at $x = 2\,\mathrm{m}$ from the inlet, where disturbances are introduced.

Concerning the numerical simulation, we use the multi-layer method with 5 layers, 300 particles per layer and 200 grid nodes per layer. A sinusoidal disturbance at the inlet with frequency equal to 3 Hz, accordingly with the experimental setup, and amplitude 2.5% is imposed. The simulation is run for 6 s, in order to have a fully developed roll wave train. The input file is reported in Appendix B. In this case, artificial viscosity needs to be tuned in order to have a good accuracy: in particular, for test case (a), $\gamma = 1.25$, for test case (b), $\gamma = 1.5$, while $\gamma = 2$ for test case (c). For this purpose, we remark that it could be convenient to treat the flux terms in the momentum equation in system (2.14) with an approximated Riemann solver, such that there is no need for manual tuning of the viscosity coefficient.

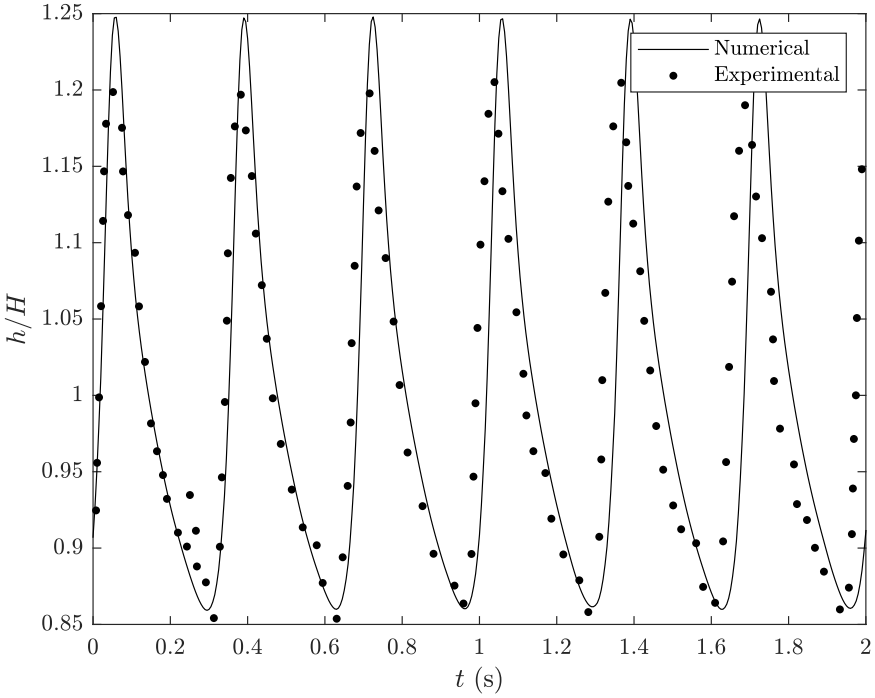
Results are compared with experimental ones in Figure 4.24. We can observe that test cases (a) and (b) are very well approximated by our method, while in test case (c) it overestimates the peaks of the roll wave. Overall, this comparison confirms that the proposed multi-layer SPH formulation can deal with open boundary conditions as well. This result is not trivial, since their treatment is a critical point of the SPH method.



(a) $H_{\text{inlet}} = 9.83\,\text{mm}$.



(b) $H_{\text{inlet}} = 10.73\,\text{mm}$ (continued in next page).



(c) $H_{\text{inlet}} = 11.57\,\text{mm}$.

Figure 4.24: Roll wave train ($x = 2\,\text{m}$): test cases proposed in Ref. [15, Fig. 13]. Results are accurate for the first two cases; peaks are overestimated in the third one.

Conclusions

A formulation for the multi-layer Saint-Venant system has been proposed, and a Lagrangian particle method (SPH) has been applied to solve it. The approach relies on the resolution of a system of conservation laws, coupled only in the source term S, which is discretized using a fixed cartesian grid. Several aspects related to the implementation of the numerical method have been discussed, such as the necessity of a variable smoothing length, and the treatment of open boundaries, which is problematic in particle methods.

A code (MultiLayerSPH) has been implemented and validated against analytical results, such as the dam break cases, the Gaussian perturbation of a free surface at rest, the Nusselt flow and experimental results. Concerning experimental results, we have shown that if a wet-dry front exists, the multi-layer approach achieves good results in approximating both the height profile and the velocity profile. For high viscosity cases, the approach is able to recover the height profile with a certain error in the determination of the wet-dry front. Instead, in the case of wet bed dam break, the multi-layer approach gives accurate estimates of the flow field. The method is able to reproduce with good approximation the experimental velocity profiles. Finally, it can reproduce with good accuracy a laminar roll waves train. We have proved that the SPH method could be used in this framework, although as well know in the literature the computational time can be higher than in the case of a grid based method.

Future works could be focused on different topics, such as:

- introduce new approximations for the velocity derivative in the source term S: for example, using spline instead of finite differences;

- use a higher order time integrator, for example second order Runge-Kutta;

- avoid the employment of artificial viscosity, approximating the flux terms in momentum equation in system (2.14) with a Riemann solver;

- allow mass exchange between layers; this is a difficult task to deal with, since in our SPH method one cannot simply impose a mass flux between layers as in a Eulerian method, but has to develop a technique for the mass exchange through particles of different layers: one possible idea can be to redistribute the volume that leaves a layer on different particles of the adjacent layers, by using a variable volume SPH method. The redistribution of the volume over many particles can be performed by weighting their contribution depending on their position with respect to the point where the new volume enters the layer. When the volume associated to a particle becomes too high (or too low), a fission (or fusion) process of particles should be performed [40];

- introduce a different rheology model: this task is simpler than the previous, since it would affect only the computation of the stress tensor;

- extend the code to 3D geometries: as already pointed out, this requires the implementation of a two dimensional solver. Probably this task would require parallel computing approach in order to maintain the computational time low enough to make the method affordable. One possibility could be to run the simulations in parallel on GPU: the speedup would be very pronounced. Directly citing Ref. [18]: *The use of graphic cards and CUDA has allowed much finer details [...] thanks to the ability to run computations with hundreds of times more particles in far shorter times than required for similar code run on a single CPU or even many clusters. This speedup makes high-resolution modeling more accessible and useable for research and design.*

A. Turbulence modelling

Consider the Navier-Stokes equation under the hypothesis summarized in Chapter 1; the 2D Reynolds averaged Navier-Stokes equations are then [31]:

$$\partial_t \bar{u}_j + \bar{u}_i \partial_i \bar{u}_j = -\frac{1}{\rho}\partial_j \bar{p} + g_j + \partial_i \left(2\nu \bar{S}_{ij}\right) - \partial_i \overline{u_i' u_j'} \tag{A.1}$$

where the over bar indicates the average of the quantity, the prime indicates its fluctuation $(\cdot)' = (\cdot) - \bar{(\cdot)}$ and $\bar{S}_{ij}$ is the mean rate of strain tensor given by

$$\bar{S}_{ij} = \frac{1}{2}\left(\partial_i u_j + \partial_j u_i\right)$$

The last term on the r.h.s. of Equation (A.1) is called the Reynolds stress tensor and is not closed: indeed, the evolutionary equation for the Reynolds stress tensor contains higher order correlations of the fluctuations and it is not closed as well. A possible solution is to introduce the Boussinesq hypothesis, which states that the deviatoric Reynolds stress is proportional to the mean rate of strain through a viscosity coefficient ν_T, called *turbulent viscosity*:

$$\frac{2}{3}k\delta_{ij} - \overline{u_i' u_j'} = 2\nu_{\mathrm{T}} \bar{S}_{ij} \tag{A.2}$$

The mean-momentum equation incorporating the turbulent-viscosity hypothesis will be:

$$\partial_t \bar{u}_j + \bar{u}_i \partial_i \bar{u}_j = -\frac{1}{\rho}\partial_j \tilde{p} + g_j + \partial_i \left(2(\nu + \nu_{\mathrm{T}}) \bar{S}_{ij}\right) \tag{A.3}$$

where $\tilde{p}$ is a modified pressure which contains the turbulent kinetic energy k.

Averaging along the z direction and considering a first order approximation, we will end up with the same equations found in (1.19), with the effective viscosity in place of the fluid viscosity, namely:

$$\begin{cases} \partial_t h_\alpha + \partial_x (h_\alpha u_\alpha) = 0 \\ \\ \partial_t \left(h_\alpha u_\alpha\right) + \partial_x \left(h_\alpha u_\alpha^2\right) + \frac{1}{\rho}\partial_x \left(h_\alpha p_\alpha\right) = \frac{1}{\rho}\left[p_{\alpha+1/2}\partial_x\left(z_{\alpha+1/2}\right) - p_{\alpha-1/2}\partial_x\left(z_{\alpha-1/2}\right)\right] + \\ \qquad\qquad + \nu_{\text{eff}}\left[\partial_z u_{\alpha+1/2} - \partial_z u_{\alpha-1/2}\right] + g\sin\beta h_\alpha \\ p_{\alpha-1/2} = p_{\alpha+1/2} + \rho g h_\alpha \end{cases} \tag{A.4}$$

where $\nu_{\text{eff}} = \nu + \nu_{\text{T}}$ is called *effective viscosity* and the over bars have been dropped down. Even if for many flows the accuracy of the hypothesis is poor (see Ref. [31] for further details), here the turbulent viscosity hypothesis is accepted as an adequate approximation.

Turbulence modelling is needed in order to estimate the value of the turbulent viscosity ν_T. The turbulent viscosity can be written as a product of a characteristic length scale and a characteristic velocity scale, and in order to estimate them, several models exist. In this work we use the Smagorinsky model [37], for which, in our case:

$$\nu_{\text{T}} = C_s^2 \Delta^2 \sqrt{2 \left(\frac{\mathrm{d}u}{\mathrm{d}z} \right)^2} \tag{A.5}$$

where C_s is a dimensionless coefficient (Smagorinsky coefficient) to be tuned and Δ is, in mesh based methods, the size of the elements of the mesh. In our model, Δ is considered to be the thickness of each layer.

B. Derivatives: SPH approach vs. FD approach

In Section 2.4.3 we say that the cost of the classical SPH approach for the computation of derivatives is much more expensive than the one using FD approach. One can try to quantify the computational cost of the former compared to the latter. The comparison is performed for the computation of the derivative of interfaces between adjacent layers, but is also valid for the estimation of velocity derivatives, with the appropriate corrections. One can observe that the information needed for the computation of interface derivative are the same needed for the computation of interface pressure and average pressure on a particle. So we focus only on the terms

$$\partial_x \left(z_{\alpha+1/2} \right) \quad \text{and} \quad \partial_x \left(z_{\alpha-1/2} \right)$$

for which we prove the computational cost of the SPH approach to be much higher than the one of FD. Finally, one can point out that the value of $z_{\alpha-1/2}$ is closely linked to the value of $z_{\alpha+1/2}$ by the relation

$$z_{\alpha-1/2}|_i = z_{\alpha+1/2}|_i - h_{\alpha_i}$$

where i indicates the i-th particle of the α-th layer. Then the following analysis on the computational costs can be performed only for $z_{\alpha+1/2}|_i$ without any loss of information.

Let us introduce the number of particles N_p (per layer), the number of grid nodes N_g (per layer) and the number of layers N_l.

B.1 SPH approach: interfaces coordinates computed at particles position

We need to compute the value of $z_{\alpha+1/2}$ at the particle i location, in order to apply the approximation of the derivative in the sense of SPH, i.e. Equations (2.6) or (2.7).

The thickness of the first layer at the particle location is directly the coordinate of the interface between layers 1 and 2: N_p SPH approximations are required. For the coordinate of the interface of the second layer (between layers 2-3), we need to compute the thickness of the first layer and of the second layer at the particle i position belonging to layer 2: $2N_p$ SPH approximations are required. For the interface between layers 3-4 we need the thickness

of layers 1, 2, 3 at the particle i position belonging to layer 3: $3N_p$ SPH approximations required. And so on, until the last interface, namely the free surface of the flow, is computed: $N_l N_p$ SPH approximations are required in this last case.

Table B.1 sums up the approximate number of SPH approximations required in each layer:

Table B.1: SPH approximations required in each layer (SPH approach).

layer	# SPH approximations
1	N_p
2	$2N_p$
3	$3N_p$
i-th	iN_p
N_s	$N_l N_p$

The total number of SPH approximations is:

$$N_{\text{tot,SPH}} = N_p + 2N_p + 3N_p + ... + N_l N_p = N_p \sum_{i=1}^{N_l} i = N_p \frac{N_l(N_l+1)}{2} \tag{B.1}$$

Then one can apply the SPH approximation of derivatives through Equations (2.6) or (2.7).

B.2 FD approach: interfaces coordinates computed at grid nodes

We compute the value of $z_{\alpha+1/2}$ at the grid nodes and we use finite differences for the derivative.

The coordinate of the interface between layers 1-2 at grid nodes is simply the thickness of first layer at their position: N_g SPH approximations are required. Having the thickness of the first layer at grid nodes, the coordinate of the interface between layers 2-3 ($z_{5/2}$) at grid nodes is simply given by:

$$z_{5/2} = h_1 + h_2$$

i.e. is given by the sum of the thicknesses of the two first layers at the grid nodes. Having already the thickness of the first layer from the previous step, we only need for the thickness of the second layer at the grid nodes: N_g SPH approximations are required. By using information from the previous steps, the number of SPH approximations per layer is simply N_g.

Table B.2 sums up the approximate number of SPH approximations required in each layer:

Table B.2: SPH approximations required in each layer (FD approach).

layer	# SPH approximations
1	N_g
2	N_g
3	N_g
i-th	N_g
N_l	N_g

The total number of SPH approximations is:

$$N_{\text{tot,FD}} = N_g + N_g + ... + N_g = N_l N_g \tag{B.2}$$

and clearly this is less expensive than the previous case in Equation (B.1), when N_l grows. For example, if $N_p = 200$ (per layer), $N_l = 10$ and $N_g = 200$ (per layer), at each time step:

$$N_p \frac{N_l(N_l+1)}{2} = 11000 > 2000 = N_l N_g$$

Furthermore, usually the number of particles is higher than the number of grid nodes, so this estimate is quite optimistic.

C. Input files

C.1 Inviscid dam break with dry bed

```
&prob_domain
case = 1
beta = 0
length = 10
boundary = .TRUE.
nlayer = 1
tfin = 0.5
/

&SPH
numpar = 400
ninterp = 200
dt = 0.001
CFL_fixed = 0
art_visc_coeff = 0.5
SLupd = .TRUE.
/

&physics
patm = 0
rho = 1000
nu = 0
turbulence = .FALSE.
/

&results
res_freq = 5
/
```

C.2 Inviscid dam break with wet bed

```
&prob_domain
case = 2
beta = 0
length = 10
boundary = .TRUE.
nlayer = 1
tfin = 0.5
/

&SPH
numpar = 400
ninterp = 200
dt = 0.001
CFL_fixed = 0
art_visc_coeff = 0.5
SLupd = .TRUE.
/

&physics
patm = 0
rho = 1000
nu = 0
turbulence = .FALSE.
/

&results
res_freq = 5
/
```

C.3 Gaussian hump

```
&prob_domain
case = 4
beta = 0
length = 10
boundary = .TRUE.

tfin = 1
/
```

```
&SPH
numpar = 400
ninterp = 200
dt = 0.001
CFL_fixed = 0
art_visc_coeff = 0.5
SLupd = .TRUE.
/

&physics
patm = 0
rho = 1000
nu = 0
turbulence = .FALSE.
/

&results
res_freq = 5
/
```

C.4 Nusselt flow

```
&prob_domain
case = 3
beta = 0.0572957
length = 2
boundary = .FALSE.
nlayer = 1, 2, 4, 6, 8, 10, 12, 14, 16
tfin = 10
/

&SPH
numpar = 50
ninterp = 50
dt = 0.01
CFL_fixed = 0
art_visc_coeff = 0.5
SLupd = .FALSE.
/

&physics
patm = 0
```

```
rho = 1000
nu = 0.01
turbulence = .FALSE.
/

&results
res_freq = 500
/
```

C.5 Roll waves

```
&probl_domain
testcase = 3
beta = 4
length = 25.
boundary = .false.
nlayer = 5
tfin = 10
/

&SPH
numpar = 800
ninterp = 400
dt = 0.01
CFL_fixed = 0
art_visc_coeff = 0.5
SLupd = .true.
/

&physics
patm = 0.
rho = 1000
nu = 0.006
turbulence = .false.
/

&results
res_freq = 20
/
```

C.6 LaRocque et al. [23] experiment

```
    &prob_domain
testcase = 1
beta = 0.4185
length = 7.31
boundary = .true.
nlayer = 50
tfin = 3.5
/

&SPH
numpar = 100
ninterp = 100
dt = 0.01
CFL_fixed = 0
art_visc_coeff = 0.5
SLupd = .true.
/

&physics
patm = 0.
rho = 1000.
nu = 0.000001
turbulence = .TRUE.
/

&results
res_freq = 5
/
```

C.7 Debiane [9] experiment

```
    &prob_domain
testcase = 1
beta = 0.
length = 0.7
boundary = .TRUE.
nlayer = 5
tfin = 50.
/
```

```
&SPH
numpar = 200
ninterp = 100
dt = 0.00025
CFL_fixed = 0
art_visc_coeff = 0.5
SLupd = .TRUE.
/

&physics
patm = 0.
rho = 1413.
nu = 0.0601
turbulence = .FALSE.
/

&results
res_freq = 5
/
```

C.8 Wang et al. [43] experiment

```
    &prob_domain
testcase = 2
beta = 0.
length = 18
boundary = .TRUE.
nlayer = 50
tfin = 3.
/

&SPH
numpar = 100
ninterp = 100
dt = 0.01
CFL_fixed = 0
art_visc_coeff = 0.5
SLupd = .TRUE.
/

&physics
patm = 0.
```

```
rho = 1000.
nu = 0.000001
turbulence = .TRUE.
/

&results
res_freq = 5
/
```

C.9 Fiorot et al. [15] experiment

```
    &probl_domain
testcase = 3
beta = 8
length = 2.5
boundary = .false.
nlayer = 5
tfin = 6
/

&SPH
numpar = 300
ninterp = 200
dt = 0.005
CFL_fixed = 0
art_visc_coeff = 1
SLupd = .true.
/

&physics
patm = 0.
rho = 1273
nu = 1.66d-4
turbulence = .false.
/

&results
res_freq = 1
/
```

Bibliography

[1] E. Audusse, "A multilayer Saint-Venant model: Derivation and numerical validation," *Discrete and Continuous Dynamical Systems-series B*, vol. 5, pp. 189–214, 2005.

[2] E. Audusse *et al.*, "A fast finite volume solver for multi-layered shallow water flows with mass exchange," *Journal of Computational Physics*, vol. 272, pp. 23–45, 2014.

[3] E. Audusse *et al.*, "A multilayer Saint-Venant system with mass exchanges for shallow water flows. Derivation and numerical validation," *ESAIM Mathematical Modelling and Numerical Analysis*, vol. 45, pp. 169–200, 2011.

[4] G. K. Batchelor, *An Introduction to Fluid Dynamics*, ser. Cambridge Mathematical Library. Cambridge University Press, 2000.

[5] O. Bernard *et al.*, "A 2D model for hydrodynamics and biology coupling applied to algae growth simulations," *ESAIM Mathematical Modelling and Numerical Analysis*, vol. 47, pp. 1387–1412, 2013.

[6] M. J. Castro *et al.*, "A multi-layer shallow-water model: Applications to the Strait of Gibraltar and the Alboran Sea," in *The Mathematics of Models for Climatology and Environment*, Springer Berlin Heidelberg, 1997, pp. 367–394.

[7] R. V. Craster *et al.*, "Dynamics and stability of thin liquid films," *Review of Modern Physics*, vol. 81, pp. 1131–1198, 2009.

[8] M. De Leffe *et al.*, "SPH modeling of shallow-water coastal flows," *Journal of Hydraulic Research*, vol. 48, pp. 118–125, 2010.

[9] K. Debiane, "Hydraulique des écoulements laminaires à surface libre dans un canal pour des milieux visqueux ou viscoplastiques: Régimes uniforme, graduellement varié, et rupture de barrage," Ph.D. dissertation, Université Joseph Fourier – Laboratoire de Rhéologie de Grenoble, 2000.

[10] O. Delestre *et al.*, "SWASHES: A compilation of shallow water analytic solutions for hydraulic and environmental studies," *International Journal for Numerical Methods in Fluids*, vol. 72, pp. 269–300, 2013.

[11] C. A. Dutra Fraga Filho, *Smoothed Particle Hydrodynamics – Fundamentals and Basic Applications in Continuum Mechanics*. Springer, 2019.

[12] I. Federico *et al.*, "Simulating 2D open-channel flows through an SPH model," *European Journal of Mechanics - B/Fluids*, vol. 34, pp. 35–46, 2012.

[13] E. Fernández-Nieto *et al.*, "2D granular flows with the $\mu(I)$ rheology and side walls friction: A well balanced multilayer discretization," *Journal of Computational Physics*, vol. 356, 2017.

[14] E. D. Fernández-Nieto *et al.*, "A multilayer method for the hydrostatic Navier-Stokes equations: A particular weak solution," *Journal of Scientific Computing*, vol. 60, pp. 408–437, 2014.

[15] G. Fiorot *et al.*, "Experimental setup for measuring roll waves on laminar open channel flows," *Flow Measurement and Instrumentation*, vol. 41, pp. 149–157, 2015.

[16] A. R. Ghigo *et al.*, "A 2D nonlinear multiring model for blood flow in large elastic arteries," *Journal of Computational Physics*, vol. 350, pp. 136–165, 2017.

[17] R. A. Gingold *et al.*, "Smoothed particle hydrodynamics: Theory and application to non-spherical stars," *Monthly Notices of the Royal Astronomical Society*, vol. 181, no. 3, pp. 375–389, 1977.

[18] A. Hérault *et al.*, "SPH on GPU with CUDA," *Journal of Hydraulic Research*, vol. 48, pp. 74–79, 2010.

[19] A. J. Hogg *et al.*, "The effects of hydraulic resistance on dam-break and other shallow inertial flows," *Journal of Fluid Mechanics*, vol. 501, pp. 179–212, 2004.

[20] S. Kalliadasis *et al.*, *Falling Liquid Films*, ser. Applied Mathematical Sciences 176. Springer-Verlag London, 2012.

[21] D. I. Ketcheson *et al.*, "High-order wave propagation algorithms for hyperbolic systems," *SIAM Journal on Scientific Computing*, vol. 35, A351–A377, 2013.

[22] D. I. Ketcheson *et al.*, "PyClaw: Accessible, extensible, scalable tools for wave propagation problems," *SIAM Journal on Scientific Computing*, vol. 34, pp. C210–C231, 2012.

[23] L. A. LaRocque *et al.*, "Experimental and numerical investigations of two-dimensional dam-break flows," *Journal of Hydraulic Engineering*, vol. 139, pp. 569–579, 2013.

[24] M. Liu *et al.*, "Smoothed particle hydrodynamics (SPH): An overview and recent developments," *Archives of Computational Methods in Engineering*, vol. 17, pp. 25–76, 2010.

[25] L. Lucy, "A numerical approach to the testing of the fission hypothesis," *The Astronomical Journal*, vol. 82, pp. 1013–1024, 1977.

[26] S. Mas-Gallic *et al.*, "A particle method for first-order symmetric systems," *Numerische Mathematik*, vol. 51, pp. 323–352, 1987.

[27] J. J. Monaghan, "Smoothed particle hydrodynamics," *Annual Review of Astronomy and Astrophysics*, vol. 30, pp. 543–574, 1992.

[28] ——, "Smoothed particle hydrodynamics," *Reports on Progress in Physics*, vol. 68, no. 8, pp. 1703–1759, 2005.

[29] P. Negi *et al.*, "An improved non-reflecting outlet boundary condition for weakly-compressible SPH," *Computer Methods in Applied Mechanics and Engineering*, vol. 367, pp. 113–119, 2020.

[30] X. Ni *et al.*, "Simulation of free-surface flow using the smoothed particle hydrodynamics (SPH) method with radiation open boundary conditions," *Journal of Atmospheric and Oceanic Technology*, vol. 33, 2016.

[31] S. B. Pope, *Turbulent flows*. Cambridge University Press, 2000.

[32] D. J. Price, "Smoothed particle hydrodynamics and magnetohydrodynamics," *Journal of Computational Physics*, vol. 231, pp. 759–794, 2012.

[33] A. Ritter, "Die fortpflanzung der wasserwellen," *Zeitschrift des Vereines Deurscher Ingenieure*, vol. 36, pp. 947–954, 1892.

[34] M. Rodriguez-Paz *et al.*, "A corrected smooth particle hydrodynamics formulation of the shallow-water equations," *Computers and Structures*, vol. 83, no. 17, pp. 1396–1410, 2005.

[35] G. Rossi *et al.*, "A well-balanced path conservative SPH scheme for nonconservative hyperbolic systems with applications to shallow water and multi-phase flows," *Computers & Fluids*, vol. 154, pp. 102–122, 2017.

[36] M. Shadloo *et al.*, "Smoothed particle hydrodynamics method for fluid flows, towards industrial applications: Motivations, current state, and challenges," *Computers & Fluids*, vol. 136, pp. 11–34, 2016.

[37] J. Smagorinsky, "General circulation experiments with the primitive equations: I. The basic experiment," *Monthly Weather Review*, vol. 91, pp. 99–164, 1963.

[38] V. Springel *et al.*, "Cosmological smoothed particle hydrodynamics simulations: The entropy equation," *Monthly Notices of the Royal Astronomical Society*, vol. 333, pp. 649–664, 2002.

[39] J. J. Stoker, *Water Waves: The Mathematical Theory with Applications*. Wiley, 1992.

[40] R. Vacondio *et al.*, "Accurate particle splitting for smoothed particle hydrodynamics in shallow water with shock capturing," *International Journal for Numerical Methods in Fluids*, vol. 69, 2012.

[41] R. Vacondio *et al.*, "Grand challenges for smoothed particle hydrodynamics numerical schemes," *Computational Particle Mechanics*, 2020.

[42] J. P. Vila, "On particle weighted methods and smooth particle hydrodynamics," *Mathematical Models and Methods in Applied Sciences*, vol. 9, pp. 161–209, 1999.

[43] B. Wang *et al.*, "Experimental and numerical investigations of similarity for dambreak flows on wet bed," *Journal of Hydrology*, vol. 583, 2020.