A MULTILAYER SHALLOW WATER LIKE METHOD APPLIED TO THIN FILM

Jacopo Li Volsi

A.Y. 2020/2021

Supervisor: Prof. Paolo Barbante

POLITECNICO
MILANO 1863

Acknowledgements

Ringraziare tutte le persone che mi sono state vicine richiederebbe molte pagine, perché devo ammettere di essere una persona molto fortunata.
Sono fortunato ad avere una larga famiglia piena di personalità diverse e uniche.
Sono fortunato ad avere molti gruppi di amici, vicini e lontani.
Sono stato molto fortunato ad aver scelto un percorso di studi che, nonostante le difficoltà, mi ha permesso di approfondire il bellissimo linguaggio matematico.
Ringrazio in particolare il professor Barbante per essere stato un relatore paziente e disponibile.

Abstract

The aim of this work is to develop both theoretically and practically an alternative model for the simulation of thin film problems. In order to achieve a new set of equations, shallow water techniques are implied, due to the physical similarities of the two phenomena, which consist in the averaging across the thickness, divided in N layers, and the elimination of some terms through asymptotic analysis. In addition to the renowned primitive equation (PE) hypothesis, it is presented an alternative assumption, which includes a dissipative viscous part, named primitive equation with vertical viscosity (PEV^2). Recent numerical schemes, such as Kurganov central upwind, are then used for the finite volume discretization of the governing equations. The resulting system is implemented as an add-on to the open-source C++ code for computational fluid dynamics simulations SU2. The software is finally validated against cases well known in literature, like Nusselt solution or dam break problem.
Keywords: thin films, shallow waters, multilayer, Kurganov, SU2.

Contents

List of Figures

List of Symbols

- δ Total film thickness, [m]
- δ_0 Bottom topography height, [m]
- δ_i Cumulative thickness up to layer i, [m]
- δ_{ij} Thickness difference between layer i and j, [m]
- ρ Fluid density, $[\frac{kg}{m^3}]$
- F Implicitly defined surface,
- $\mathbf{n}$ Normal outward vector, [-]
- $\mathbf{v}$ Velocity vector, $[\frac{m}{s}]$
- p Pressure, [Pa]
- $\underline{\underline{\tau}}$ Cauchy stress tensor, [Pa]
- μ Dynamic viscosity, $[\frac{kg}{ms}]$
- ν Kinematic viscosity, $[\frac{m^2}{s}]$
- γ Surface tension, $[\frac{kg}{s^2}]$
- λ_∞ Wavelength characteristic dimension, [m]
- ϵ Shallow water parameter, [-]
- Re_λ Reynolds number for the wavelength, [-]
- Re_δ Reynolds number for the thickness, [-]
- Ca Capillarity, [-]
- We Weber number, [-]
- St Strouhal number, [-]
- f vortex shedding frequency, $[\frac{1}{s}]$
- Fr Froude number, [-]
- g Standard gravity, $[\frac{m}{s^2}]$
- $\mathbf{U}$ Vector of conserved quantities,
- ∇ gradient operator,
- $\nabla\cdot$ divergence operator,

- $\mathcal{F}$ Flux function in x direction,
- $\mathcal{G}$ Flux function in y direction,
- C Chezy constant, $[\frac{m^{\frac{1}{2}}}{s}]$
- $\frac{1}{n}$ Manning roughness coefficient, $[\frac{m^{\frac{1}{3}}}{s}]$
- R Hydraulic radius. [m]

1 Introduction

The flow of thin films is a subject which finds applications in a huge vastity of scientific fields, spanning from biophysics, where it can model the liquid membrane manifested during the blink of an eye [1], to geology, simulating the dynamic of large scale lava or continental ice sheet [2]. Even more are the engineering and industrial processes in which thin films are used for approximating the reality. For example, such model, used in the aerospace framework of the de-icing of aircraft wings [3], allows to predict where the film may freeze, and thus prevent serious consequences. Thin films are also found in diesel direct injection engines to account for non-vaporized fuel deposited on the head of the piston in the combustion chamber, as studied to a series of articles [4]-[5] published at the end of the nineties by Stanton and Rutland.
The dynamic of the liquid can be originated by various forces, such as gravity, capillarity or the shear of another fluid, and the consequent motion may present interesting features, for example regular or chaotic responses and, most importantly, shock and periodic waves. These considerations illustrate the reason why thin films arouse a great interest from the physical and mathematical point of view, but they also entail many complications from the modeling and the numerical prospective. The principal issue is caused by the non-linearity of the advection term in the mean evolution equation, which is the most commonly used treatment reserved to situation where one dimension is much smaller than the others. Indeed, starting from classical Navier-Stokes equations and following an approach similar to the one used in lubrication or in shallow waters, the unknown quantities are averaged across the direction in which the liquid result more subtle. Doing so, the mean of the product of the advected variables is obtained, which clearly cannot be immediatly substituted with the multiplication of the two separete mean unkowns. Most of the articles and research studies apply the same tecnique, which consist in the expansion of the aforementioned non-linear terms to obtain a closed model.
As exemplification of such method, we summarize the work done by Lavalle in its Ph.D. thesis [6]. In its discussion, each component of the velocity is approximated as the sum of functions corresponding to long waves of arbitrary amplitude, as first introduced by Benney in [7]. It is then chosen the desired accuracy order and the expression are substituted in the boundary layer set of equations. Integrating along the depth the equations allows to solve complex wave dynamics on falling liquid film, making possible to retrieve the streamwise velocity profile. It is interesting to stress that this procedure is boundary condition dependant, which are substituted during the integration process. Once the profile is computed, the non-linear term can be expanded obtaining, as one can imagine, an extensive sum of linear terms. In additon, the problem must be closed inserting the previous result for the velocity in the expression of the wall shear stress.
The procedure described, even if presented briefly and only in its key points, highlights the two main constraints of this approach. Firstly, the equations to be solved appear complex and often need much longer expression to increase the order of accuracy. The second, and most important issue, is the loss of generality due to the insertion of boundary conditions in the calculation of the profiles. The aim of this thesis is therefore to present an alternative strategy to remove

the non-linearity complication in an arbitrary way. The idea is to borrow from the shallow water theory, which shares similar hypothesis with thin film, the so-called multilayer method. The entire film heigth is divided in a number of layers, which still are fewer than an actual discretization in normal direction to save computational time, and on each of them the whole set of equations is resolved. This technique allows to exchange mean and product operations thanks to the smaller approximation error with respect to an integration along the complete thickness.
However, before arguing on the mathematical peculiarities, a general overview on the possible cases and application is presented in Chapter 2, which includes the hypothesis assumed throughout the whole discussion. To follow as much as possible a common thread between the shallow water and thin film world, the forces acting on the fluid taken into consideration are reduced to the shear of a gas interacting with the liquid and to the contribution of gravitational effects. The introductory section covers also the geometrical settings and restrictions to the problem faced by this work. Chapter 3 analizes the mathematical model developed accordingly to the multilayer technique and explains the two different assumption for the pressure typical of shallow water and their adaptation to this context. The fourth and fifth Chapters are mainly focused on the numerical method applied, which joins the recent work of Kurganov for the convective part with ad hoc discretization for viscous and source terms, and on the implementation of the extension of SU2, the CFD open source software born at Stanford University, which will take care of the envoirment outside the film. Lastly, the resulting mechanism is validated and analized on known test cases.

2 Problem Description

A thin film is a layer of fluid, typically liquid, which is deposed, sprayed or even created by melting on a surface. As hinted in the introduction, thin film theory and the study of its dynamics finds applications in countless fields, therefore it is an hard task to summarize all possible cases and even more difficult to propose a general model which takes into account every possible scenery and external contribution. In fact, even nowadays, many aspects related to thin film dynamics have not been clarified yet. On the other side, it is proper to present some phenomenology related to liquid film and to argue which hypothesis has been taken in consideration for this work as well as the reason behind the semplifications made.

2.1 Examples of Film Dynamics

There are many contributions to the dynamics of a liquid film and modeling them all would raise dramatically the complexity of the mathematical description. Therefore, as observed in Forte's PhD thesis [13], we state that the principal causes which allow the motion of a film can be summarized in the following list:

- Transport under the action of volumetric forces or shear stresses coming from the interaction with the world outside the film with the its surface.
- Variations of the momentum due to impingement of liquid drops, which can result in very different phenomena.
- Heat transfer with the wall or the surrounding gas and possible phase transition of the film itself or even of the solid bottom.

All those contributions can be visually represented and schematized in Fig. 2.1, which helps to have a clearer picture of the situation. In the following subsections each element listed will be documented, however we remind that the cases described are not an exhaustive list of all the possible reasons behind the motion of a thin film.

2.1.1 Falling and Sheared Film

The first phenomenon which was studied and on which were conducted experiments, is the falling film on an inclined plane and it immediately shows many important features. Already such case, which may look basic, presents some peculiarities. For example, the evolution of a thin liquid on a vertical flat plate without any forcing inlet is characterized by three different long-waves regimes, such as no perturbation in the higher part, 2D waves developing in the middle part to arrive at irregular 3D ripples in the furthest part form the inlet, as one can see in Fig. 2.2.

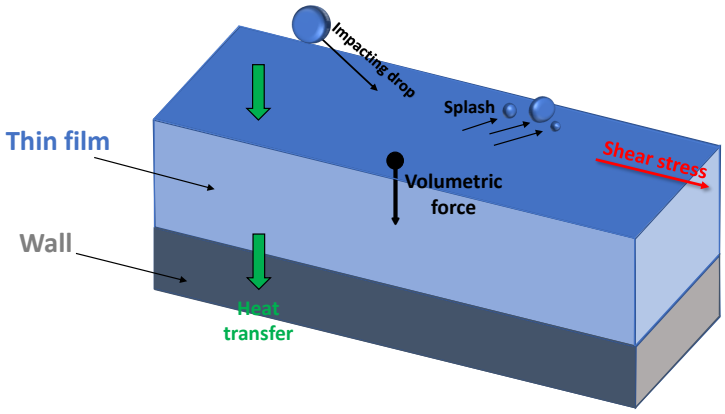

Figure 2.1: Sources of motion for a generic liquid film.

This, alongside with other experiments such as Nosoko & Miyara [15], make us conclude that the vertical case arises strongly non-linear waves on the surface of a liquid film. Indeed, in literature it is well established that surpassing a certain computable threshold, defined by the critical Reynolds number phenomena similar to the ones presented in the figure manifest. Reynolds number is a very important adimensional quantity in fluid dynamics theory, given by the formula

$$Re = \frac{UL\rho}{\mu}, \tag{2.1}$$

where U and L are respectively the characteristic velocity and length, which strongly depend on the typology of application that we are working with, while ρ and μ are intrinsic property of the substance such as density and dynamic viscosity, that can change only changing the fluid. The physical explanation of this expression is the ratio between inertial and viscous forces, indicating which aspect is privileged.
Continuing with the study of falling sheared film at small Reynolds number, it has been demonstrated that the height of wave crests is always small compared to the wavelength of the waves, which justify the introduction of a shallow water vertical parameter ϵ, that will be used in the deduction of the model in the following chapter. This assumption of long waves drives to the so-called long-wave theory, on which we will base, as well as the presented sources of motion, throughout this work.
Other models, like the one proposed by Nusselt [16] where the gravity force is fully balanced by the wall shear stress, introduced also the interaction with the surrounding world and found a waveless solution of the Navier-Stokes equations assuming a steady constant thickness film. The resulting profile is therefore assimilable to the parabolic solution of a channel flow. However, Nusselt's approach fails when unstable waves develop at the free surface of the liquid film.
The next two sparks which ignite the fluid dynamic will be presented apace with the reason why they have been omitted.

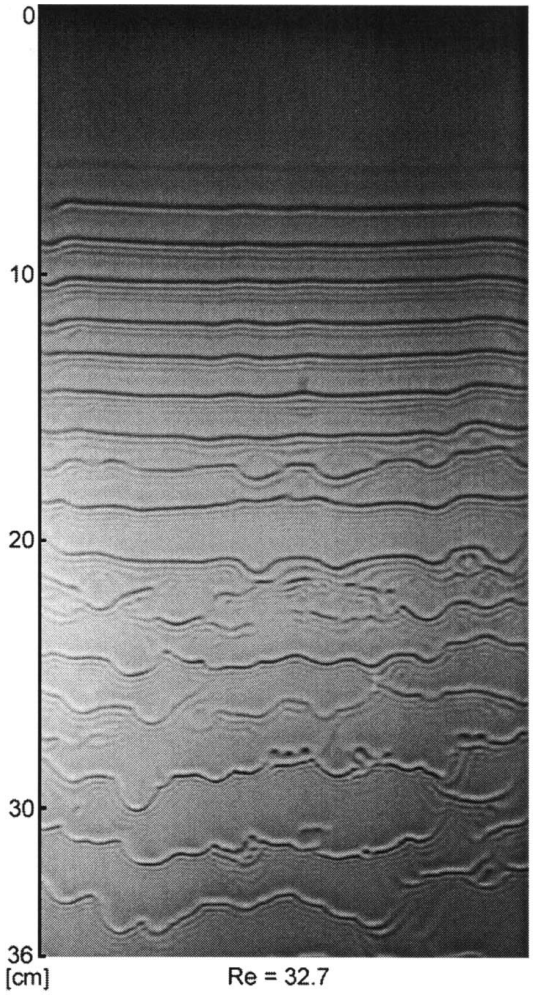

Figure 2.2: Development of waves on a vertical falling water film, experiment of Park & Nosoko [14].

2.1.2 Spray Impingement

Another typical situation that can be modeled with thin film theory is the dynamic of a liquid sprayed on a surface and again moved either by gravity or shear stress. The addition that this new point of view takes into consideration, is the impingement of drops of the same substance to the moving fluid, causing a significant variation in the momentum. An ulterior complexity is given by the fact that the impacting drop may have different behaviours. Hereafter, we give a brief explanation of each feasible phenomenon, using as discriminant parameter Weber's number, given by the ratio of inertial forces and surface tension, as proves the formula

$$We = \frac{\rho U^2 L}{\gamma} = Re\ Ca \tag{2.2}$$

where γ is the surface tension, strictly connected with the capillarity Ca, if we want to highlight the connection with Reynolds number.
The achievable effects are listed in Fig. 2.3 and are essentially of four types: stick, rebound,

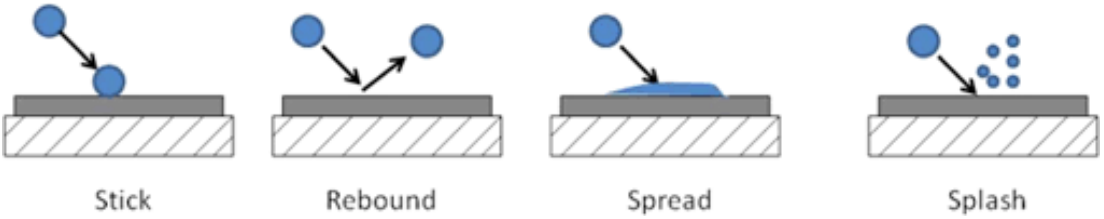

Figure 2.3: Typologies of drop impingement.

spread and splash.
Stick happens when Weber's number is less than 5 and the drop does not have the necessary energy neither to enter the film nor to change its shape, therefore it continues traveling attached to the surface.
If Weber is between 5 and 10, the impact has a sufficient energy which allows the **rebound**, but not the penetration in the liquid. Contrarily to the stick, at higher Weber's number the kinetic energy that characterizes the collision is enough to allow the shape to change and be embodied in the flowing film, causing the so-called **spread**.
Finally, for an impact energy higher than all previous cases, the phenomenon of **splash** manifests itself. It consists in the detachment of some droplet from a crown of liquid formed after the clash of the incoming particle.
Unfortunately, due the typology of software used for this work, which separates the zones with different fluids and it is not adaptable to the integration of smaller portion of volume proper of one area into the other, the decision made is to neglect this source of motion from the construction of the model.

2.1.3 Heat transfer

The contribution of heat may arise from many aspects, but most notably, like discussed by Oron [8] and Davis [17], from thermocapillary effect. Thermocapillary effect account for the emergence of shear stress, due to the variation of surface tension with temperature. In order to incorporate this reaction, one should add to the governing system an equation for the energy and appropriate boundary condition related to heat transfer. It has been proved by Oron [8] that, if gravity acts toward the base of the film, it has a stabilizing effect, while instead thermocapillarity creates instabilities especially at the interface. Moreover, if the temperature reaches the evaporation threshold, the repercussions heavily affect the dynamics of the film.
Therefore, the decision made was not to consider any passage of phase happening during the film evolution and consequently this motion mechanism, as well as the inclusion of an equation for the energy, has been completely avoided.

2.2 Geometrical Considerations

In many technological applications, e.g. direct injection engines, optical fiber coverage or painting textile filaments, one can encounter the situation of a thin liquid layer spread on a cylindrical surface, which introduces an extra component of curvature on the interface. To adapt the equations to this particular and often studied domain, in literature the choice of cylindrical coordinates is surely the most frequently used. Another avaible option, presented in [18] by Trujillo, is the conversion into Dupin coordinate system, which handles more efficently surface and normal components. Nevertheless, since the derived model, featuring vertical average and

multilayer technique to close unkwon terms, is not common, the geometry used during this work is rather simple, leaving space for complications on the equations and on the numerical solver. In particular, the coordinate system we refer to is Cartesian and the normal component for the film is always to be intended as the vertical direction of the classical referment, i.e. the ordinate in two dimension and the z-axis in three.
Another necessary and useful precisation regarding the terminology, especially involving the multilayer setting, has to be done. The most critical and influent condition are given at the boundary and in the case of thin film those are the interface or free surface and the bottom, i.e. the typically solid backdrop on which the liquid lays. Introducing the multilayer method, the different layers are distincted in 3 categories called bottom, top and k-th stratum. The first typology is not the basis at the boundary between the wall and the fluid, but it is the nearest to the physical bottom. On the other hand, the top layer represents exactly the free surface interface between film and, for example, the shearing gas around it. The remaining are simply the ones included between the previously defined strata.
To be more clear, we refer to the following image, Fig. 2.4, where it is introduced also the notation used throughout the rest of the work. The cumulative height of each layer is indicated

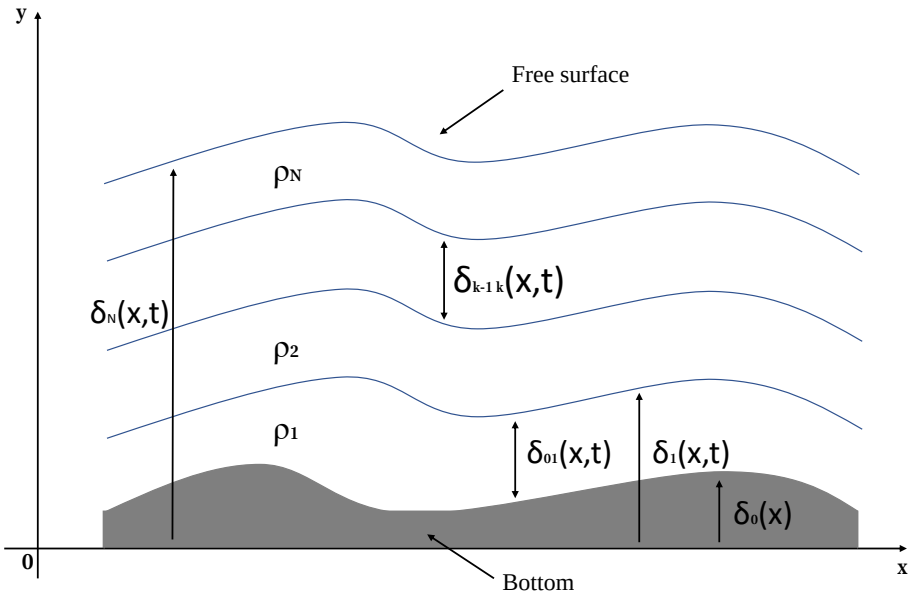

Figure 2.4: Multilayer technique domain.

with the greek letter delta either combined with a pedix δ_k denoting the stratum to which it belongs or δ without any subscript to specify the whole thickness, in such way that $\delta = \delta_N$ if the film is divided into N layers. The backdrop, usually indicated with $b(x)$, is called δ_0 and it is worth noticing that it does not depend on time, because we supposed that there is no sedimentation or similar phenomena, which may change in time the shape of the bottom. It is also important to define the layer thickness $\delta_{k-1k} = \delta_k - \delta_{k-1}$ for $k = 1..., N$, i.e. the difference of adjacent layer height, as this variable will pop out in the derivation of the mathematical model. One last remark can be done on the fluid density ρ, which may differ for distinct depth. This assumption is done more frequently in the shallow water framework, where it may indicate for example that fresher water flows over more saline, and consequently, heavier water.
Finally, also the distance intercurring among the layers plays an influent role on the efficency of the normal profile reconstruction, needed to close extra-diagonal components of the stress

tensor. To visualize in a clear way this consideration, the reader can think to the polynomial interpolation of a boundary layer velocity profile using equispaced nodes versus a distribution of the points closer to the wall. Indeed, for a function very steep in a small region of its domain, such as for $y^+ < 6$ in the flat plate case, knots placed at the same distance are not suitable.

3 Mathematical Model

The modeling of thin film can benefit a dimensional simplification due to the particular geometry of the problem. Through some semplifications we wil obtain a model that, once discretized, will help to reduce significantly the computational cost and the execution time of numerical simulations, which is an excellent advantage for real mathematical applications.
The path we chose to achieve the final model, that will avoid the use of the typical Taylor expansion to close the equations, is structured in two successive steps: averaging and asymptotic. The average, done in order to remove the dependency from the normal direction, is performed jointly with the application of the multilayer method, which is the substitutive plan to close the system of equations. Subsequently, additional terms popping out from the integration by part are analized and possibly simplified, if assimilable to infinitesimal order, by asymptotic analysis.
Finally, the resulting system's nature is studied, prelude for the discretization techniques described in the next chapter.

3.1 Multilayer Method

Multilayer method is a modelling approach frequently used in shallow water problems. Basically, it consists in dividing the water depth in multiple surfaces on which the same set of equation lives, plus some transfer term to connect each different stratum. For example, as presented by J. Macias *et al.* in [19], Saint-Venant equations are applied to a two layers problem aiming to simulate water dynamics in the Alborean sea, situated near the Strait of Gibraltar, where the Atlantic ocean and the Mediterranean sea meet. This approach is adopted to achieve two major benefits: a simpler set of integrated equation and a computational effort reduction, because the meshing is not performed along the whole depth, but only on the layer surface.
In the thin film setting, this method may be ulteriorly exploited to close the average terms that pop out from the integration in the direction normal to the wall. This happens because, unlike the classical film theory, where the mean terms are computed along the whole orthogonal direction and therefore need to be expanded and approximated locally, averaging across the smaller pieces of domain between the layers reduces the inaccuracy of this representation. Hence, the domain below the film must be divided in N initially given surfaces represented by implicit functions F_i, for i = 1,...,N, and the asymptotic equations need to be integrated along each interval created by the layers, paying particular attention to the first, which will be the nearest to the bottom of our domain, and the *N-th*, the one representing the free surface.

3.1.1 Average model

To lighten numerical computation of the 3-D configurations of the problem, we decide to perform an integration in normal direction and to consider average quantities for the vertical axis. In order to do this, we make the hypothesis that $\delta_i = \delta_i(x, y, t)$, i.e. each layer in the film is single valued for every x and y and it can be recast in the implicit form $F_i(x, y, z, t) = z - \delta_i(x, y, t) = 0$. The normal to this free surface is therefore defined through the gradient of F_i as $\mathbf{n}_i = \nabla F = -\frac{\partial \delta_i}{\partial x}\mathbf{i} - \frac{\partial \delta_i}{\partial y}\mathbf{j} + \mathbf{k}$, being $\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$ the unitary vector of our reference system. This is true only if the norm of the gradient is approximately 1, fact that can be checked by the result $(\frac{\partial \delta_i}{\partial x})^2$, $(\frac{\partial \delta_i}{\partial y})^2 \ll 1$, coherent with the subsequent asymptotic analysis.
The main idea is first to exchange integrals and derivatives, then to extract average quantities in order to make appear also the unknown δ_{ij}, representing the height between two generic adjacent layers. We therefore recall Liebniz formula, that is

$$\frac{\partial}{\partial t}\int_{a(t)}^{b(t)} f(x,t)dx = \int_{a(t)}^{b(t)} \frac{\partial f(x,t)}{\partial t}dx + f(b(t),t)\frac{db}{dt} - f(a(t),t)\frac{da}{dt}. \tag{3.1}$$

The final tool we need, since we are performing a mean analysis, is the mean value theorem for the integral, i.e. $\int_a^b f(x)dx = f(\zeta)(b-a)$, supposing f continuous and $\zeta \in (a, b)$. To shorten notation, mean quantities will be indicated with an overlaying bar, e.g. $\bar{f}$.
Another useful assumption we introduce is that the density is constant in normal direction, which is reasonable if we consider the thin film setting.

3.1.2 Multilayer Continuity Equation

Using the tools introduced in the average model section, we show the procedure integrating not along all the film height, but only between two generic neighbouring layers δ_i and δ_j, the continuity equation

$$\int_{\delta_i}^{\delta_j} \frac{\partial \rho}{\partial t}dz + \int_{\delta_i}^{\delta_j} \bigtriangledown \cdot (\rho \mathbf{v})dz = 0, \tag{3.2}$$

where ρ is the fluid density and $\bigtriangledown\cdot$ the divergence operator acting on the velocity vector $\mathbf{v}$. Via Liebniz formula, the result is the following:

$$\begin{aligned}\frac{\partial}{\partial t}\int_{\delta_i}^{\delta_j} \rho\ dz - \rho_{\delta_j}\frac{\partial \delta_j}{\partial t} + \rho_{\delta_i}\frac{\partial \delta_i}{\partial t} + \frac{\partial}{\partial x}\int_{\delta_i}^{\delta_j} \rho u\ dz - \rho_{\delta_j}u_{\delta_j}\frac{\partial \delta_j}{\partial x} + \rho_{\delta_i}u_{\delta_i}\frac{\partial \delta_i}{\partial x} \\ + \frac{\partial}{\partial y}\int_{\delta_i}^{\delta_j} \rho v\ dz - \rho_{\delta_j}v_{\delta_j}\frac{\partial \delta_j}{\partial y} + \rho_{\delta_i}v_{\delta_i}\frac{\partial \delta_i}{\partial y} + \int_{\delta_i}^{\delta_j} \frac{\partial \rho w}{\partial z}\ dz = 0.\end{aligned} \tag{3.3}$$

Of course in z direction the integration follows easily and directly. Using the implicit form of the surface, we see that $\frac{dF_j}{dt} = \frac{dz}{dt} - \frac{\partial \delta_j}{\partial t} - \frac{dx}{dt}\frac{\partial \delta_j}{\partial x} - \frac{dy}{dt}\frac{\partial \delta_j}{\partial y} = w_{s_j} - \frac{\partial \delta_j}{\partial t} - u_{s_j}\frac{\partial \delta_j}{\partial x} - v_{s_j}\frac{\partial \delta_j}{\partial y}$, where the pedix s_j indicates the velocities of the fluid evaluated at the implicit surface. The expression can be further simplified recalling that the normal to F_j is $\mathbf{n}_j = -\frac{\partial \delta_j}{\partial x}\mathbf{i} - \frac{\partial \delta_j}{\partial y}\mathbf{j} + \mathbf{k}$, resulting in $\frac{\partial \delta_j}{\partial t} = \mathbf{v}_{s_j} \cdot \mathbf{n}_j$. Substituting this geometric consideration and applying the mean theorem, we obtain:

$$\frac{\partial(\delta_{ij}\rho)}{\partial t} + \frac{\partial(\delta_{ij}\rho\bar{u}_j)}{\partial x} + \frac{\partial(\delta_{ij}\rho\bar{v}_j)}{\partial y} + \rho_j(\mathbf{v}_{\delta_j} - \mathbf{v}_{s_j}) \cdot \mathbf{n}_j + \rho_i(\mathbf{v}_{s_i} - \mathbf{v}_{\delta_i}) \cdot \mathbf{n}_i = 0, \tag{3.4}$$

adopting the semplified notation for the density $\rho_{\delta_j} = \rho_j$ here and for the rest of the work.
At contact with the free surface, the interaction term $(\mathbf{v}_{s_i} - \mathbf{v}_{\delta_i}) \cdot \mathbf{n}_i$ may survive since there

can happen phase passage or the impact of liquid drops, which causes a very different velocity to the fluid shearing the film.
Writing δ_{ij} we intend the time-variant thickness between layer j and i, i.e. $\delta_j - \delta_i$.
Then we adapt the described technique at the bottom, the generic k-th and the top layer because each of these cases have a peculiar behaviour and consequently different terms surviving.
Without loss of genrality, we suppose that the wall is flat and still, which results in δ_0 equal to constant in space and time annihilating the contribution of its temporal derivative, which correspondes to the term $\mathbf{v}_{s_0} \cdot \mathbf{n}_0$.

- Bottom layer

$$\frac{\partial(\delta_{01}\rho)}{\partial t} + \frac{\partial(\delta_{01}\rho\bar{u}_1)}{\partial x} + \frac{\partial(\delta_{01}\rho\bar{v}_1)}{\partial y} - \rho_0 w_0 = 0.$$

- K-th layer

$$\frac{\partial(\delta_{k-1k}\rho)}{\partial t} + \frac{\partial(\delta_{k-1k}\rho\bar{u}_k)}{\partial x} + \frac{\partial(\delta_{k-1k}\rho\bar{v}_k)}{\partial y} = 0.$$

For the strata in between, there is no exchange of velocity because the only source of motion for a layer is the velocity of the fluid and therefore when in the temporal derivative of the surface F_k arises the surface velocity $\mathbf{v}_{s_k}$, it is equal to the velocity evaluated at the k-th thickness $\mathbf{v}_{\delta_k}$, because we supposed that the surface is described by a streamline, therefore the instantaneous value is identical.

- Top layer

$$\frac{\partial(\delta_{N-1N}\rho)}{\partial t} + \frac{\partial(\delta_{N-1N}\rho\bar{u}_N)}{\partial x} + \frac{\partial(\delta_{N-1N}\rho\bar{v}_N)}{\partial y} + \rho_N(\mathbf{v}_{\delta_N} - \mathbf{v}_s) \cdot \mathbf{n}_N = 0.$$

Our new unknown are δ_{k-1k} and $(\bar{u}_k, \bar{v}_k, \bar{w}_k)$ for $k = 1, 2, ..., N$, where the mean velocity refers to the correspondent layer and it is indicated by the subscript $_k$. In addition, to compute the normal at the top layer, we need to compute the total film height given by the formula $\delta_N = \delta_0 + \delta_{01} + \delta_{12} + ... + \delta_{N-1N}$.

3.1.3 Multilayer Momentum Equation

Arguing as for the above continuity equation, we integrate the law for the conservation of momentum, for exemplificative purposes in x-direction and for the generic adjacent layers, with subscripts i and j. After applying Liebniz theorem and, as stated in the introduction to multilayer method, supposing that the mean cross term and the mean quadratic velocity can be simplified from average of product to product of average, introducing an hopefully neglectable approximation error, the resulting law is

$$\begin{aligned}
&\frac{\partial}{\partial t}\int_{\delta_i}^{\delta_j} \rho u \, dz - \rho_j u_{\delta_i}\frac{\partial \delta_j}{\partial t} + \rho_i u_{\delta_i}\frac{\partial \delta_i}{\partial t} + \frac{\partial}{\partial x}\int_{\delta_i}^{\delta_j} \rho u^2 \, dz + \frac{\partial}{\partial y}\int_{\delta_i}^{\delta_j} \rho uv \, dz \\
&\quad -\rho_j u_{\delta_j}(u_{\delta_j}\frac{\partial \delta_j}{\partial x} + v_{\delta_j}\frac{\partial \delta_j}{\partial y} - w_{\delta_j}) + \rho_i u_{\delta_i}(u_i\frac{\partial \delta_i}{\partial x} + v_{\delta_i}\frac{\partial \delta_i}{\partial y} - w_{\delta_i}) \\
&\quad + \frac{\partial}{\partial x}\int_{\delta_i}^{\delta_j} p dz - p_{\delta_j}\frac{\partial \delta_j}{\partial x} + p_{\delta_i}\frac{\partial \delta_i}{\partial x} = \frac{\partial}{\partial x}\int_{\delta_i}^{\delta_j} \tau_{xx} dz + \frac{\partial}{\partial y}\int_{\delta_i}^{\delta_j} \tau_{xy} dz \\
&\quad - \tau_{xx}|_{\delta_j}\frac{\partial \delta_j}{\partial x} + \tau_{xx}|_{\delta_i}\frac{\partial \delta_i}{\partial x} - \tau_{xy}|_{\delta_j}\frac{\partial \delta_j}{\partial y} + \tau_{xy}|_{\delta_i}\frac{\partial \delta_i}{\partial y} - \tau_{xz}|_{\delta_j} + \tau_{xz}|_{\delta_i} + \int_{\delta_i}^{\delta_j} f_x dz, \quad (3.5)
\end{aligned}$$

where we can again exploit the relation between the normal of the implicit surface and the derivative of the thickness and the relation with the temporal derivative, obtaining

$$\begin{aligned}\frac{\partial(\rho\bar{u}_j\delta_{ij})}{\partial t} &- \rho_j u_{\delta_j}\mathbf{v}_{s_j}\cdot\mathbf{n}_j + \rho_i u_{\delta_i}\mathbf{v}_{s_i}\cdot\mathbf{n}_i + \frac{\partial(\rho\bar{u}_j^2\delta_{ij})}{\partial x} + \frac{\partial(\rho\bar{u}_j\bar{v}_j\delta_{ij})}{\partial y} \\ &+\rho_j u_{\delta_j}\mathbf{v}_{\delta_j}\cdot\mathbf{n}_j - \rho_i u_{\delta_i}\mathbf{v}_{\delta_i}\cdot\mathbf{n}_i + \frac{\partial(\bar{p}_j\delta_{ij})}{\partial x} = \frac{\partial(\bar{\tau}^j_{xx}\delta_{ij})}{\partial x} + \frac{\partial(\bar{\tau}^j_{xy}\delta_{ij})}{\partial y} \\ &+[(-p_{\delta_j}\underline{\underline{I}} + \underline{\underline{\tau}}_{\delta_j})\cdot\mathbf{n}_j]_x - [(-p_{\delta_i}\underline{\underline{I}} + \underline{\underline{\tau}}_{\delta_i})\cdot\mathbf{n}_i]_x\delta_{ij} + \bar{f}_{x,j}.\end{aligned} \tag{3.6}$$

As in the previous section, we highlight the equality of $\mathbf{v}_{s_j}$ and $\mathbf{v}_{\delta_j}$, which allow the semplification of the velocity interaction terms.
It is worth noticing that also the stress tensor components are evaluated at the correspondent layer. Moreover, they have an implicit formulation, since for example $\bar{\tau}^j_{xx}\delta_{ij} = \int_{\delta_i}^{\delta_j}\mu\frac{\partial u}{\partial x}dz \neq \mu\frac{\partial\bar{u}_j}{\partial x}\delta_{ij}$, being μ the fluid dynamic viscosity. To solve this problem there exist two possible solutions. The first should provide a second use of Liebniz theorem that will generate extra boundary terms. Indeed, supposing the viscosity μ to be constant in the vertical axis, the expanded formulation results in $\bar{\tau}^j_{xx}\delta_{ij} = \mu\int_{\delta_i}^{\delta_j}\frac{\partial u}{\partial x}dz = \mu\frac{\partial(\bar{u}_j\delta_{ij})}{\partial x} - \mu u_{\delta_j}\frac{\partial\delta_j}{\partial x} + \mu u_{\delta_i}\frac{\partial\delta_i}{\partial x}$ and analogously for the other components. The second possibility is to use, once regained the mean velocity, a suitable interpolant which can give an expression of the stress tensor entries.
The resulting equations, expanded for each specific layer, are grouped in the following systems:

- Bottom layer

$$\begin{cases}\frac{\partial(\rho\bar{u}_1\delta_{01})}{\partial t} + \frac{\partial(\rho\bar{u}_1^2\delta_{01})}{\partial x} + \frac{\partial(\rho\bar{u}_1\bar{v}_1\delta_{01})}{\partial y} + \frac{\partial(\bar{p}_1\delta_{01})}{\partial x} = \frac{\partial(\bar{\tau}^1_{xx}\delta_{01})}{\partial x} + \frac{\partial(\bar{\tau}^1_{xy}\delta_{01})}{\partial y} + \\ \qquad +\delta_{01}\bar{f}_{x,1} + [(-p_{\delta_1}\underline{\underline{I}} + \underline{\underline{\tau}}_{\delta_1})\cdot\mathbf{n}_1]_x - \tau^0_{xz} + \rho_0 u_0 w_0 \\ \frac{\partial(\rho\bar{v}_1\delta_{01})}{\partial t} + \frac{\partial(\rho\bar{u}_1\bar{v}_1\delta_{01})}{\partial x} + \frac{\partial(\rho\bar{v}_1^2\delta_{01})}{\partial y} + \frac{\partial(\bar{p}_1\delta_{01})}{\partial y} = \frac{\partial(\bar{\tau}^1_{yx}\delta_{01})}{\partial x} + \frac{\partial(\bar{\tau}^1_{yy}\delta_{01})}{\partial y} + \\ \qquad +\delta_{01}\bar{f}_{y,1} + [(-p_{\delta_1}\underline{\underline{I}} + \underline{\underline{\tau}}_{\delta_1})\cdot\mathbf{n}_1]_y - \tau^0_{yz} + \rho_0 v_0 w_0 \\ \frac{\partial(\rho\bar{w}_1\delta_{01})}{\partial t} + \frac{\partial(\rho\bar{w}_1\bar{u}_1\delta_{01})}{\partial x} + \frac{\partial(\rho\bar{w}_1\bar{v}_1\delta_{01})}{\partial y} = \frac{\partial(\bar{\tau}^1_{zx}\delta_{01})}{\partial x} + \frac{\partial(\bar{\tau}^1_{zy}\delta_{01})}{\partial y} + \\ \qquad +\delta_{01}\bar{f}_{z,1} + [(-p_{\delta_1}\underline{\underline{I}} + \underline{\underline{\tau}}_{\delta_1})\cdot\mathbf{n}_1]_z + p_{\delta_0} - \tau^0_{zz} + \rho_0 w_0^2\end{cases}$$

- K-th layer

$$\begin{cases}\frac{\partial(\rho\bar{u}_k\delta_{k-1k})}{\partial t} + \frac{\partial(\rho\bar{u}_k^2\delta_{k-1k})}{\partial x} + \frac{\partial(\rho\bar{u}_k\bar{v}_k\delta_{k-1k})}{\partial y} + \frac{\partial(\bar{p}_k\delta_{k-1k})}{\partial x} = \frac{\partial(\bar{\tau}^k_{xx}\delta_{k-1k})}{\partial x} + \frac{\partial(\bar{\tau}^k_{xy}\delta_{k-1k})}{\partial y} + \\ \qquad +\delta_{k-1k}\bar{f}_{x,k} + [(-p_{\delta_k}\underline{\underline{I}} + \underline{\underline{\tau}}_{\delta_k})\cdot\mathbf{n}_k]_x - [(-p_{\delta_{k-1}}\underline{\underline{I}} + \underline{\underline{\tau}}_{\delta_{k-1}})\cdot\mathbf{n}_{k-1}]_x \\ \frac{\partial(\rho\bar{v}_k\delta_{k-1k})}{\partial t} + \frac{\partial(\rho\bar{u}_k\bar{v}_k\delta_{k-1k})}{\partial x} + \frac{\partial(\rho\bar{v}_k^2\delta_{k-1k})}{\partial y} + \frac{\partial(\bar{p}_k\delta_{k-1k})}{\partial y} = \frac{\partial(\bar{\tau}^k_{yx}\delta_{k-1k})}{\partial x} + \frac{\partial(\bar{\tau}^k_{yy}\delta_{k-1k})}{\partial y} + \\ \qquad +\delta_{k-1k}\bar{f}_{y,k} + [(-p_{\delta_k}\underline{\underline{I}} + \underline{\underline{\tau}}_{\delta_k})\cdot\mathbf{n}_k]_y - [(-p_{\delta_{k-1}}\underline{\underline{I}} + \underline{\underline{\tau}}_{\delta_{k-1}})\cdot\mathbf{n}_{k-1}]_y \\ \frac{\partial(\rho\bar{w}_k\delta_{k-1k})}{\partial t} + \frac{\partial(\rho\bar{w}_k\bar{u}_k\delta_{k-1k})}{\partial x} + \frac{\partial(\rho\bar{w}_k\bar{v}_k\delta_{k-1k})}{\partial y} = \frac{\partial(\bar{\tau}^k_{zx}\delta_{k-1k})}{\partial x} + \frac{\partial(\bar{\tau}^k_{zy}\delta_{k-1k})}{\partial y} + \\ \qquad +\delta_{k-1k}\bar{f}_{z,k} + [(-p_{\delta_k}\underline{\underline{I}} + \underline{\underline{\tau}}_{\delta_k})\cdot\mathbf{n}_k]_z - [(-p_{\delta_{k-1}}\underline{\underline{I}} + \underline{\underline{\tau}}_{\delta_{k-1}})\cdot\mathbf{n}_{k-1}]_z\end{cases}$$

• Top layer

$$\begin{cases} \frac{\partial(\rho\bar{u}_N\delta_{N-1N})}{\partial t}+\frac{\partial(\rho\bar{u}_N^2\delta_{N-1N})}{\partial x}+\frac{\partial(\rho\bar{u}_N\bar{v}_N\delta_{N-1N})}{\partial y}+\frac{\partial(\bar{p}_N\delta_{N-1N})}{\partial x}+\rho_N u_{\delta_N}(\mathbf{v}_{\delta_N}-\mathbf{v}_s)\cdot\mathbf{n}_N= \\ \qquad \frac{\partial(\bar{\tau}^N_{xx}\delta_{N-1N})}{\partial x}+\frac{\partial(\bar{\tau}^N_{xy}\delta_{N-1N})}{\partial y}+\delta_{N-1N}\bar{f}_{x,N}+[(-p_{\delta_N}\underline{\underline{I}}+\underline{\underline{\tau}}_{\delta_N})\cdot\mathbf{n}_N]_x-[(-p_{\delta_{N-1}}\underline{\underline{I}}+\underline{\underline{\tau}}_{\delta_{N-1}})\cdot\mathbf{n}_{N-1}]_x \\ \frac{\partial(\rho\bar{v}_N\delta_{N-1N})}{\partial t}+\frac{\partial(\rho\bar{u}_N\bar{v}_N\delta_{N-1N})}{\partial x}+\frac{\partial(\rho\bar{v}_N^2\delta_{N-1N})}{\partial y}+\frac{\partial(\bar{p}_N\delta_{N-1N})}{\partial y}+\rho_N v_{\delta_N}(\mathbf{v}_{\delta_N}-\mathbf{v}_s)\cdot\mathbf{n}_N= \\ \qquad \frac{\partial(\bar{\tau}^N_{yx}\delta_{N-1N})}{\partial x}+\frac{\partial(\bar{\tau}^N_{yy}\delta_{N-1N})}{\partial y}+\delta_{N-1N}\bar{f}_{y,N}+[(-p_{\delta_N}\underline{\underline{I}}+\underline{\underline{\tau}}_{\delta_N})\cdot\mathbf{n}_N]_y-[(-p_{\delta_{N-1}}\underline{\underline{I}}+\underline{\underline{\tau}}_{\delta_{N-1}})\cdot\mathbf{n}_{N-1}]_y \\ \frac{\partial(\rho\bar{w}_N\delta_{N-1N})}{\partial t}+\frac{\partial(\rho\bar{w}_N\bar{u}_N\delta_{N-1N})}{\partial x}+\frac{\partial(\rho\bar{w}_N\bar{v}_N\delta_{N-1N})}{\partial y}+\rho_N w_{\delta_N}(\mathbf{v}_{\delta_N}-\mathbf{v}_s)\cdot\mathbf{n}_N= \\ \qquad \frac{\partial(\bar{\tau}^N_{zx}\delta_{N-1N})}{\partial x}+\frac{\partial(\bar{\tau}^N_{zy}\delta_{N-1N})}{\partial y}+\delta_{N-1N}\bar{f}_{z,N}+[(-p_{\delta_N}\underline{\underline{I}}+\underline{\underline{\tau}}_{\delta_N})\cdot\mathbf{n}_N]_z-[(-p_{\delta_{N-1}}\underline{\underline{I}}+\underline{\underline{\tau}}_{\delta_{N-1}})\cdot\mathbf{n}_{N-1}]_z \end{cases} \tag{3.7}$$

Unfortunately even with the multilayer integrated technique, the system of equations comprehending continuity and momentum is not enough to match the 5 unknowns present in each layer, which are the 3 mean component of the velocity, mean pressure and thickness between layer. Therefore, we will exploit the dimensional considerations that can be carried out for the thin film setting and perform an asymptotic analysis, aiming to retrieve a system in closed form.

3.2 Asymptotic Analysis

In order to simplify the multilayer model governing the thin film, we perform an asymptotic analysis. The key hypothesis for such physics is that the height of the liquid in the direction normal to the wall is much smaller than the length in the other two directions. Therefore, in assigning characteristic length, we will assert that in the z-axis the typical height is δ_∞, whereas in x and y, coherently with the phenomenon, it will have the dimension of the wavelength, called λ_∞, supposed equal for both directions. To satisfy the geometric condition of normality of the vector $\mathbf{n}_j$ we need $\|\nabla F_j\| = \sqrt{(\frac{\partial\delta_j}{\partial x})^2+(\frac{\partial\delta_j}{\partial y})^2+1}\approx 1$, which dimensionally translates into $(\frac{\delta_\infty}{\lambda_\infty})^2\approx\epsilon^2$, where ϵ is defined as the ratio between the characteristic wavelength and the typical thickness, which is a small quantity.
A final assumption regards the velocity vector $\mathbf{v}=(u,v,w)$, where the vertical component at infinity and the ones tangential to the wall will be considered of different orders of magnitude, respectively w_∞ and u_∞.

3.2.1 Asymptotic Continuity Equation

After setting a characteristical length for all the dimensional variables, we can start the analysis from the mean continuity equation in differential form for the generic layer between δ_i and δ_j, thus we can write:

$$\begin{aligned} \frac{\rho_\infty\delta_\infty}{t_\infty}\frac{\partial\rho\delta_{ij}}{\partial t}+\frac{\rho_\infty u_\infty\delta_\infty}{\lambda_\infty}\frac{\partial(\rho\bar{u}_j\delta_{ij})}{\partial x}+\frac{\rho_\infty u_\infty\delta_\infty}{\lambda_\infty}\frac{\partial(\rho\bar{v}_j\delta_{ij})}{\partial y}+\rho_\infty u_\infty\frac{\delta_\infty}{\lambda_\infty}\rho_j(u_{\delta_j}n_x-u_{s_j}n_x)+ \\ +\rho_\infty u_\infty\frac{\delta_\infty}{\lambda_\infty}\rho_j(v_{\delta_j}n_y-v_{s_j}n_y)+\rho_\infty w_\infty\rho_j(w_{\delta_j}-w_{s_j})+\mathbf{a}_i\rho_i(\mathbf{v}_{s_i}-\mathbf{v}_{\delta_i})\cdot\mathbf{n}_i=0, \end{aligned} \tag{3.8}$$

where $\mathbf{a}_i$ contains compressed in its value the dimensional constants of the scalar product between the velocity and the layer surface normal vector. Note that the quantities under

partial derivatives are adimensional, but, for readability reasons, the same symbols are used. Simplifying the latter equation we get:

$$\frac{\lambda_\infty}{u_\infty}\frac{1}{t_\infty}\frac{\partial \rho\delta_{ij}}{\partial t}+\frac{\partial(\rho\bar{u}_j\delta_{ij})}{\partial x}+\frac{\partial(\rho\bar{v}_j\delta_{ij})}{\partial y}+\rho_j(u_{\delta_j}n_x-u_{s_j}n_x)+\rho_j(v_{\delta_j}n_y-v_{s_j}n_y)$$
$$+\frac{\lambda_\infty w_\infty}{\delta_\infty u_\infty}\rho_j(w_{\delta_j}-w_{s_j})+\mathbf{a}_i^*\rho_i(\mathbf{v}_{s_i}-\mathbf{v}_{\delta_i})\cdot\mathbf{n}_i=0. \tag{3.9}$$

From this form of the equation we obtain some useful considerations, because, in order to respect mass balance, all term must be of the same order of magnitude. First of all we notice that the temporal derivative is multiplied by the inverse of the Strouhal number, defined as $St=\frac{fL}{U}$, where f is the frequency of vortex shedding, while U and L have the same definition introduced for Reynolds number. In order not to fall back in a stationary case, we consider its value equal to 1, which implies that the characteristic time is governed by the tangential variables $t_\infty\approx\frac{\lambda_\infty}{u_\infty}$.
Moreover, and this information turns out to be essential, we can affirm that $\frac{\lambda_\infty w_\infty}{\delta_\infty u_\infty}\approx 1$ and subsequently recover the following relation between the components of the velocity $\frac{w_\infty}{u_\infty}\approx\epsilon$, which can let us state the different order of magnitude of the vertical and tangential speed, i.e. $w_\infty\ll u_\infty$. Finally, we may also notice that the previous relation can be rewritten to highlight that the ratio $\frac{w_\infty}{\delta_\infty}\approx\frac{u_\infty}{\lambda_\infty}$ and obviously also $\frac{w_\infty}{u_\infty}\approx\frac{\delta_\infty}{\lambda_\infty}$ result constant and, more importantly, of the same order.

3.2.2 Asymptotic Momentum Equation

Afterwards, we perform the same analysis for the conservation of the momentum. Our aim is to simplify the complete equation with term of order $\frac{\delta_\infty}{\lambda_\infty}$ or smaller. Since we identified a significant difference in w, the component of the velocity orthogonal to the wall, the analysis will focus only in the x and z direction, with a symmetric extension to y. Because the dynamic viscosity μ is an intrinsic property of the fluid, the discriminant for our analysis will be Reynolds number. We can distinguish two types of this adimensional quantity according to the referment length considered and so we can explicit the following relationship:

$$Re_\lambda=\frac{u_\infty\lambda_\infty\rho_\infty}{\mu}=\frac{u_\infty\delta_\infty\rho_\infty}{\mu}\frac{\lambda_\infty}{\delta_\infty}=\frac{Re_\delta}{\epsilon}, \tag{3.10}$$

and the obvious inverse $Re_\delta=Re_\lambda\epsilon$.
A last assumption regards the source term f, which can be considered without loss of generality as the gravity force $\rho\mathbf{g}$, as happens in the film falling down an inclined plane. Thus, we can start analizing one of the component parallel to the wall and , after some computation similar to the adimensionalization of Navier-Stokes, Reynolds constant pops out in front of viscous terms.

$$\frac{\partial(\rho\bar{u}_j\delta_{ij})}{\partial t}+\frac{\partial(\rho\bar{u}_j^2\delta_{ij})}{\partial x}+\frac{\partial(\rho\bar{u}_j\bar{v}_j\delta_{ij})}{\partial y}+\frac{p_\infty}{\rho_\infty u_\infty^2}\frac{\partial(\bar{p}_j\delta_{ij})}{\partial x}=\frac{1}{Re_\lambda}\frac{\partial(\bar{\tau}^j_{xx}\delta_{ij})}{\partial x}+\frac{1}{Re_\lambda}\frac{\partial(\bar{\tau}^j_{xy}\delta_{ij})}{\partial y}$$
$$+\rho g_x\delta_{ij}-\frac{p_\infty}{\rho_\infty u_\infty^2}p_{\delta_j}n_x+\frac{1}{Re_\lambda}(\tau_{xx}|_{\delta_j}n_x+\tau_{xy}|_{\delta_j}n_y)+\frac{1}{Re_\lambda\epsilon^2}\frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_j}+\frac{1}{Re_\lambda}\frac{1}{2}\mu\frac{\partial w}{\partial x}|_{\delta_j}$$
$$+\frac{p_\infty}{\rho_\infty u_\infty^2}p_{\delta_i}n_x-\frac{1}{Re_\lambda}(\tau_{xx}|_{\delta_i}n_x+\tau_{xy}|_{\delta_i}n_y)-\frac{1}{Re_\lambda\epsilon^2}\frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_i}-\frac{1}{Re_\lambda}\frac{1}{2}\mu\frac{\partial w}{\partial x}|_{\delta_i}. \tag{3.11}$$

In the thin film setting, an assumption such as $Re_\lambda\approx 1$ is not feasible. The range of value taken by Reynolds number is wide because we are dealing with a boundary layer like region,

but we can summirize it in the interval between ϵ^{-1} and ϵ^{-2}. If it is near the upper extreme, almost any viscosity contribute disappear due to a minor influence, except the derivative of the tangential velocity with respect to the height. Decreasing towards the lower bound, the terms previously kept still have an higher order and, not to fall in an incorrect Euler model, we can let them survive also in this case.
Another crucial discussion must be faced when talking about the pressure. Considering that the effects of the pressure are more appreciable across the surface represented by the bottom rather than its change along the thin film thickness, plus what suggested by the asymptotic analysis, we affirm that $p_\infty \approx u_\infty^2 \rho_\infty$. Hence, the pressure gradient in x and y direction cannot therefore be forgotten. Finally, we remark that all the dimensional quantities multiplying the gravitational source are semplified, making this contribution not neglectable. To end the study of the momentum equation in tangential direction with the adoption of the asymptotic technique, we present the result in the following formula, where all the terms multiplied by ϵ^2 or smaller quantities have been annihilated:

$$\frac{\partial(\rho\bar{u}_j\delta_{ij})}{\partial t}+\frac{\partial(\rho\bar{u}_j^2\delta_{ij})}{\partial x}+\frac{\partial(\rho\bar{u}_j\bar{v}_j\delta_{ij})}{\partial y}+\frac{\partial(\bar{p}_j\delta_{ij})}{\partial x}=\rho g_x\delta_{ij}-p_{\delta_j}n_{j,x}+p_{\delta_i}n_{i,x}+\frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_j}-\frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_i}. \tag{3.12}$$

Let's now tackle the problem of the momentum conservation in normal direction, where the analysis is completely analogous, but some facts need to be highlighted. As reference, we write the law where we already applied the asymptotic technique:

$$\begin{aligned}\frac{\partial(\rho\bar{w}_j\delta_{ij})}{\partial t}+\frac{\partial(\rho\bar{u}_j\bar{w}_j\delta_{ij})}{\partial x}+\frac{\partial(\rho\bar{v}_j\bar{u}_j\delta_{ij})}{\partial y}&=-\frac{1}{Fr^2\epsilon}\rho g_z\delta_{ij}+\frac{1}{Re_\lambda\epsilon^2}\frac{\partial(\bar{\tau}_{zx}^j\delta_{ij})}{\partial x}\\&+\frac{1}{Re_\lambda\epsilon^2}\frac{\partial(\bar{\tau}_{zy}^j\delta_{ij})}{\partial y}-\frac{1}{\epsilon^2}(p_{\delta_j}-p_{\delta_i})+\frac{1}{Re_\lambda\epsilon^2}[\underline{\underline{\tau}}_{\delta_j}\cdot\mathbf{n}_j]_z-\frac{1}{Re_\lambda\epsilon^2}[\underline{\underline{\tau}}_{\delta_i}\cdot\mathbf{n}_i]_z.\end{aligned} \tag{3.13}$$

Concerning the pressure gradient, the situation is completely opposite, since this term is of order $\frac{\lambda_\infty}{\delta_\infty}\frac{u_\infty}{w_\infty}=\frac{1}{\epsilon^2}$ which is much greater than one. This means that, with respect to the vertical change of pressure, all other factors can be considered negligible, which result in the hydrostatic hypothesis $\frac{\partial p}{\partial z}+\rho g_z=0$, the only factor surviving momentum equation in normal direction. This assumption is known in literature, referincing for example [10], as primitive equations, thus, from now on, we refer to this approximation as PE. In addition, to simplify the notation, in the following paragraph, we suppose the case of pure vertical gravity, meaning $g_z=g$.
From the PE hypothesis, the explicit expression for the pressure can be retrieved simply by integrating between the generic vertical point and the whole thickness δ obtaining

$$\int_\delta^z\frac{\partial p}{\partial z}dz=\int_\delta^z-\rho g dz, \tag{3.14}$$

which, supposing that the density is constant across the depth, results in

$$p(x,y,z)-p(x,y,\delta)=-\rho g(z-\delta).$$

Another useful consideration is that $p(x,y,\delta)$ is the pressure at the free surface and it is independent of z, but on the tangential components, hence, in normal direction, we can adopt the semplification to consider $p(x,y,\delta)$ as an arbitrary constant and therefore put it equal zero, considering as final expression for the pressure

$$p(x,y,z)=-\rho g(z-\delta). \tag{3.15}$$

However, during the asymptotic analysis, if Reynolds number is close to ϵ^{-1}, some derivative

of the viscous stress tensor have order much greater than one. They can be therefore included in the new z-momentum equation, giving birth to what Lions, Temam and Wang in [10] call the primitive equations with vertical viscosity (PEV^2), i.e.

$$\frac{\partial p}{\partial z} + \rho g = \frac{\partial \tau_{zx}}{\partial x} + \frac{\partial \tau_{zy}}{\partial y} + \frac{\partial \tau_{zz}}{\partial z}. \tag{3.16}$$

If we go even more in the specification of mean stress, we can see that the only surviving pieces, beyond $\frac{\partial \tau_{zz}}{\partial z}$, would be $\frac{\partial}{\partial x}(\frac{1}{2}\mu\frac{\partial u_j}{\partial z})$ and $\frac{\partial}{\partial y}(\frac{1}{2}\mu\frac{\partial v_j}{\partial z})$, because the asymptotic analysis suggests that the other cross entry is of order $\frac{1}{Re_\lambda \epsilon^2}$, and therefore neglectable.
Since the intent of the model taken in consideration is to be fairly general, and the impact of the hypothesis chosen does not provoke almost any change on the nature of the equation, we decide to continue the analysis mainly with PE, studying them and implementing a numerical scheme, while presenting only the most relevant theoretical aspects of PEV^2. Assuming hydrostatic pressure, as we will observe in the following sections, the normal component of the velocity no longer appears into the new set of equations.

3.3 PE Asymptotic Multilayer System

To compact the equations, in shallow water literature it is acclaimed to perform a substition of PE hypothesis, integrating them in vertical direction. The integration is made in parallel with the averaging of Navier-Stokes in order to retrieve mean pressure and to substitute it in the system. It relys on the mean value theorem applied to the pressure

$$\bar{p}_k \delta_{k-1k} = \int_{\delta_{k-1}}^{\delta_k} p dz, \tag{3.17}$$

in which it is substituted the chosen hypothesis for the pressure, in this case PE to simplify the calculation,

$$\bar{p}_k \delta_{k-1k} = \int_{\delta_{k-1}}^{\delta_k} -\rho g(z-\delta) dz. \tag{3.18}$$

Performing the integration, the formula, which will be substitued in our set of equations, is

$$\bar{p}_k \delta_{k-1k} = -\rho g \frac{(\delta_k^2 - \delta_{k-1}^2)}{2} + \rho g \delta \delta_{k-1k}, \tag{3.19}$$

that can be condensed, via decomposition of difference of squares $(\delta_k^2 - \delta_{k-1}^2) = (\delta_k - \delta_{k-1})(\delta_k + \delta_{k-1}) = \delta_{k-1k}(\delta_k + \delta_{k-1})$, leading to the following equality

$$\bar{p}_k = -\frac{1}{2}\rho g(\delta_k + \delta_{k-1} - 2\delta). \tag{3.20}$$

However, one must note that the quantities δ and δ_k hide the conserved variable δ_{k-1k}, which is useful to explicit, in presvision of the analysis of the nature of the equations. Indeed

$$\begin{aligned} \delta_{k-1} &= \sum_{i=1}^{k-1} \delta_{i-1i}, \\ \delta_k &= \sum_{i=1}^{k} \delta_{i-1i}, \\ \delta &= \sum_{i=1}^{N} \delta_{i-1i}. \end{aligned} \tag{3.21}$$

Expanding all those terms in the hydrostatic hypothesis, the result obtained is

$$\bar{p}_k = -\frac{1}{2}\rho g(2\sum_{i=1}^{k-1}\delta_{i-1i} + \delta_{kk-1} - 2\sum_{i=1}^{N}\delta_{i-1i}), \tag{3.22}$$

which, solving the summation, can be rewritten in

$$\bar{p}_k = -\frac{1}{2}\rho g(-\delta_{k-1k} - 2\sum_{i=k+1}^{N}\delta_{i-1i}) = \frac{1}{2}\rho g(\delta_{k-1k} + 2\sum_{i=k+1}^{N}\delta_{i-1i}), \tag{3.23}$$

that is composed by a part present in the classic hydrostatic theory, i.e. $\frac{1}{2}\rho g\delta_{kk-1}$, and an interaction term coming from the multilayer method, i.e. $\rho g\sum_{i=k+1}^{N}\delta_{i-1i}$. A precisation, that will become useful later, is that the summation $\sum_{i=k+1}^{N}\delta_{i-1i}$ is equal to the difference $\delta_N - \delta_k$, which unfortunately hides implicitly the conserved variable.
Grouping the results obtained in the previous pages, we will have to solve the following systems for each layer.

- Bottom layer

$$\begin{cases} \frac{\partial(\delta_{01}\rho)}{\partial t} + \frac{\partial(\delta_{01}\rho\bar{u}_1)}{\partial x} + \frac{\partial(\delta_{01}\rho\bar{v}_1)}{\partial y} - \rho_0 w_0 = 0 \\ \frac{\partial(\rho\bar{u}_1\delta_{01})}{\partial t} + \frac{\partial(\rho\bar{u}_1^2\delta_{01} + \frac{1}{2}\rho g\delta_{01}(\delta_{01}+2\sum_{i=2}^{N}\delta_{i-1i}))}{\partial x} + \frac{\partial(\rho\bar{u}_1\bar{v}_1\delta_{01})}{\partial y} = \\ \qquad [p_{\delta_1}\frac{\partial\delta_1}{\partial x} + \frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_1}] - \tau^0_{xz} + \rho_0 u_0 w_0 \\ \frac{\partial(\rho\bar{v}_1\delta_{01})}{\partial t} + \frac{\partial(\rho\bar{u}_1\bar{v}_1\delta_{01})}{\partial x} + \frac{\partial(\rho\bar{v}_1^2\delta_{01} + \frac{1}{2}\rho g\delta_{01}(\delta_{01}+2\sum_{i=2}^{N}\delta_{i-1i}))}{\partial y} = \\ \qquad [p_{\delta_1}\frac{\partial\delta_1}{\partial y} + \frac{1}{2}\mu\frac{\partial v}{\partial z}|_{\delta_1}] - \tau^0_{yz} + \rho_0 v_0 w_0 \\ \bar{p}_1\delta_{01} = \frac{1}{2}\rho g\delta_{01}(\delta_{01} + 2\sum_{i=2}^{N}\delta_{i-1i}) \end{cases}$$

- K-th layer

$$\begin{cases} \frac{\partial(\delta_{k-1k}\rho)}{\partial t} + \frac{\partial(\delta_{k-1k}\rho\bar{u}_k)}{\partial x} + \frac{\partial(\delta_{k-1k}\rho\bar{v}_k)}{\partial y} = 0 \\ \frac{\partial(\rho\bar{u}_k\delta_{k-1k})}{\partial t} + \frac{\partial(\rho\bar{u}_k^2\delta_{k-1k} + \frac{1}{2}\rho g\delta_{k-1k}(\delta_{k-1k}+2\sum_{i=k+1}^{N}\delta_{i-1i}))}{\partial x} + \frac{\partial(\rho\bar{u}_k\bar{v}_k\delta_{k-1k})}{\partial y} = \\ \qquad [p_{\delta_k}\frac{\partial\delta_k}{\partial x} + \frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_k}] - [p_{\delta_{k-1}}\frac{\partial\delta_{k-1}}{\partial x} + \frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_{k-1}}] \\ \frac{\partial(\rho\bar{v}_k\delta_{k-1k})}{\partial t} + \frac{\partial(\rho\bar{u}_k\bar{v}_k\delta_{k-1k})}{\partial x} + \frac{\partial(\rho\bar{v}_k^2\delta_{k-1k} + \frac{1}{2}\rho g\delta_{k-1k}(\delta_{k-1k}+2\sum_{i=k+1}^{N}\delta_{i-1i}))}{\partial y} = \\ \qquad [p_{\delta_k}\frac{\partial\delta_k}{\partial y} + \frac{1}{2}\mu\frac{\partial v}{\partial z}|_{\delta_k}] - [p_{\delta_{k-1}}\frac{\partial\delta_{k-1}}{\partial y} + \frac{1}{2}\mu\frac{\partial v}{\partial z}|_{\delta_{k-1}}] \\ \bar{p}_k\delta_{k-1k} = \frac{1}{2}\rho g\delta_{k-1k}(\delta_{k-1k} + 2\sum_{i=k+1}^{N}\delta_{i-1i}) \end{cases}$$

- Top layer

$$\begin{cases} \frac{\partial(\delta_{N-1N}\rho)}{\partial t} + \frac{\partial(\delta_{N-1N}\rho\bar{u}_N)}{\partial x} + \frac{\partial(\delta_{N-1N}\rho\bar{v}_N)}{\partial y} + \rho_N(\mathbf{v}_{\delta_N} - \mathbf{v}_s)\cdot\mathbf{n}_N = 0 \\ \frac{\partial(\rho\bar{u}_N\delta_{N-1N})}{\partial t} + \frac{\partial(\rho\bar{u}_N^2\delta_{N-1N} + \frac{1}{2}\rho g\delta_{N-1N}^2)}{\partial x} + \frac{\partial(\rho\bar{u}_N\bar{v}_N\delta_{N-1N})}{\partial y} + \rho_N u_{\delta_N}(\mathbf{v}_{\delta_N} - \mathbf{v}_s)\cdot\mathbf{n}_N = \\ \qquad [p_{\delta_N}\frac{\partial\delta_N}{\partial x} + \frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_N}] - [p_{\delta_{N-1}}\frac{\partial\delta_{N-1}}{\partial x} + \frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_{N-1}}] \\ \frac{\partial(\rho\bar{v}_N\delta_{N-1N})}{\partial t} + \frac{\partial(\rho\bar{u}_N\bar{v}_N\delta_{N-1N})}{\partial x} + \frac{\partial(\rho\bar{v}_N^2\delta_{N-1N} + \frac{1}{2}\rho g\delta_{N-1N}^2)}{\partial y} + \rho_N v_{\delta_N}(\mathbf{v}_{\delta_N} - \mathbf{v}_s)\cdot\mathbf{n}_N = \\ \qquad [p_{\delta_N}\frac{\partial\delta_N}{\partial y} + \frac{1}{2}\mu\frac{\partial v}{\partial z}|_{\delta_N}] - [p_{\delta_{N-1}}\frac{\partial\delta_{N-1}}{\partial y} + \frac{1}{2}\mu\frac{\partial v}{\partial z}|_{\delta_{N-1}}] \\ \bar{p}_N\delta_{N-1N} = \frac{1}{2}\rho g\delta_{N-1N}^2 \end{cases} \tag{3.24}$$

Notice that in top layer, because $\delta = \delta_N$, the average pressure contribution to the convection is $-\frac{1}{2}\rho g(\delta_N + \delta_{N-1} - 2\delta_N)\delta_{N-1N} = -\frac{1}{2}\rho g(\delta_{N-1} - \delta_N)\delta_{N-1N} = \frac{1}{2}\rho g\delta_{N-1N}^2$, which is a term

usually existing in shallow water theory, see for example Kurganov [26]. Moreover, since we assumed that our model should contain no phase transition and no impacting drops, the difference $\mathbf{v}_{\delta_N} - \mathbf{v}_s$ between the top layer velocity vector and the one of the world outside the film can be set to zero.
We finally obtained a closed system which counts 4 unknowns and 4 equations for each layer. However, it still requires some minor treatments to retrieve the optimal shape for an efficient resolution.

3.3.1 Reformulation of the PEs

As anticipated before, now that the system is in closed form, the intent is to collocate it in a well known theoretical framework, in order to decide which numerical method to implement. However, previa doing that, a couple of remark and semplification will result very handful. We want to manipulate in the momentum conservation laws, part of system (3.24), the variables under a spatial derivative at the left-hand side and compare them with the source contributions, especially of the pressure, at the right-hand side of the equations.
First of all, starting from the left-hand side of the momentum equation, for simplicity illustrated only in x-direction, the flux term can be split, via linearity of the derivative, in

$$\frac{\partial(\rho\bar{u}_k^2\delta_{k-1k} + \frac{1}{2}\rho g\delta_{k-1k}^2)}{\partial x} + \frac{\partial\left(\rho g\delta_{k-1k}\sum_{i=k+1}^{N}\delta_{i-1i}\right)}{\partial x}, \tag{3.25}$$

that with suitable assumptions on the density and on the volumetric force, which are constant along the single layer, and reminding that $\sum_{i=k+1}^{N}\delta_{i-1i} = \delta_N - \delta_k$, can be rewritten as

$$\frac{\partial(\rho\bar{u}_k^2\delta_{k-1k} + \frac{1}{2}\rho g\delta_{k-1k}^2)}{\partial x} + \rho g(\delta_N - \delta_k)\frac{\partial\delta_{k-1k}}{\partial x} + \rho g\delta_{k-1k}\frac{\partial(\delta_N - \delta_k)}{\partial x}. \tag{3.26}$$

The sum retrieved can now be indentified as the well known term in the shallow water literature plus an interaction part. The goal is to transform this extra contribution into a source influence. Focusing now the attention at the right-hand side, precisely on the pressure expression

$$p_k\frac{\partial\delta_k}{\partial x} - p_{k-1}\frac{\partial\delta_{k-1}}{\partial x}, \tag{3.27}$$

we add and subtract the mixed term $p_k\frac{\partial\delta_{k-1}}{\partial x}$ obtaining

$$p_k\frac{\partial(\delta_k - \delta_{k-1})}{\partial x} + (p_k - p_{k-1})\frac{\partial\delta_{k-1}}{\partial x}. \tag{3.28}$$

Now the PE hypothesis (3.15) can be explicitly exploited in the following way

$$\rho g(\delta_N - \delta_k)\frac{\partial(\delta_k - \delta_{k-1})}{\partial x} + \rho g(\delta_N - \delta_k - \delta_N + \delta_{k-1})\frac{\partial\delta_{k-1}}{\partial x}, \tag{3.29}$$

that can be finally reformulated in

$$\rho g(\delta_N - \delta_k)\frac{\partial\delta_{k-1k}}{\partial x} - \rho g\delta_{k-1k}\frac{\partial\delta_{k-1}}{\partial x}. \tag{3.30}$$

At this point, one can note that the first term of (3.30) is equal and contrary, because at opposite side, to the interaction part of (3.26), therefore they cancel out, leaving only $-\rho g\delta_{k-1k}\frac{\partial(\delta_N-\delta_k)}{\partial x} -$

$\rho g \delta_{k-1k} \frac{\partial \delta_{k-1}}{\partial x}$. One can note that the derived terms do not depend on any conserved quantities, since the difference $(\delta_N - \delta_k)$ cuts out the dependency from δ_{k-1k}, but it only includes an interexchange between layers.
In addition, note that if one would use PEV^2 when specifing p_k and p_{k-1}, we will regain the same exact expression and the same cancellation, but many other normal stress contribution will pop out.
Hence, we can rewrite the system in this new simplified form, for the bottom layer

$$\begin{cases} \frac{\partial(\delta_{01}\rho)}{\partial t} + \frac{\partial(\delta_{01}\rho\bar{u}_1)}{\partial x} + \frac{\partial(\delta_{01}\rho\bar{v}_1)}{\partial y} - \rho_0 w_0 = 0 \\ \frac{\partial(\rho\bar{u}_1\delta_{01})}{\partial t} + \frac{\partial(\rho\bar{u}_1^2\delta_{01} + \frac{1}{2}\rho g\delta_{01}^2)}{\partial x} + \frac{\partial(\rho\bar{u}_1\bar{v}_1\delta_{01})}{\partial y} = -\rho g\delta_{01}\frac{\partial(\delta_N - \delta_1)}{\partial x} + \frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_1} - \tau^0_{xz} + \rho_0 u_0 w_0 \\ \frac{\partial(\rho\bar{v}_1\delta_{01})}{\partial t} + \frac{\partial(\rho\bar{u}_1\bar{v}_1\delta_{01})}{\partial x} + \frac{\partial(\rho\bar{v}_1^2\delta_{01} + \frac{1}{2}\rho g\delta_{01}^2)}{\partial y} = -\rho g\delta_{01}\frac{\partial(\delta_N - \delta_1)}{\partial y} + \frac{1}{2}\mu\frac{\partial v}{\partial z}|_{\delta_1} - \tau^0_{yz} + \rho_0 v_0 w_0 \\ \bar{p}_1\delta_{01} = \frac{1}{2}\rho g\delta_{01}(\delta_{01} + 2\sum_{i=2}^{N}\delta_{i-1i}) \end{cases},$$

remarking that, since the bottom doesn't vary, $\frac{\partial \delta_0}{\partial x} = \frac{\partial \delta_0}{\partial y} = 0$, then for the k-th layer

$$\begin{cases} \frac{\partial(\delta_{k-1k}\rho)}{\partial t} + \frac{\partial(\delta_{k-1k}\rho\bar{u}_k)}{\partial x} + \frac{\partial(\delta_{k-1k}\rho\bar{v}_k)}{\partial y} = 0 \\ \frac{\partial(\rho\bar{u}_k\delta_{k-1k})}{\partial t} + \frac{\partial(\rho\bar{u}_k^2\delta_{k-1k} + \frac{1}{2}\rho g\delta_{k-1k}^2)}{\partial x} + \frac{\partial(\rho\bar{u}_k\bar{v}_k\delta_{k-1k})}{\partial y} = \\ \qquad -\rho g\delta_{k-1k}\frac{\partial(\delta_N - \delta_k)}{\partial x} - \rho g\delta_{k-1k}\frac{\partial\delta_{k-1}}{\partial x} + [\frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_k} - \frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_{k-1}}] \\ \frac{\partial(\rho\bar{v}_k\delta_{k-1k})}{\partial t} + \frac{\partial(\rho\bar{u}_k\bar{v}_k\delta_{k-1k})}{\partial x} + \frac{\partial(\rho\bar{v}_k^2\delta_{k-1k} + \frac{1}{2}\rho g\delta_{k-1k}^2)}{\partial y} = \\ \qquad -\rho g\delta_{k-1k}\frac{\partial(\delta_N - \delta_k)}{\partial y} - \rho g\delta_{k-1k}\frac{\partial\delta_{k-1}}{\partial y} + [\frac{1}{2}\mu\frac{\partial v}{\partial z}|_{\delta_k} - \frac{1}{2}\mu\frac{\partial v}{\partial z}|_{\delta_{k-1}}] \\ \bar{p}_k\delta_{k-1k} = \frac{1}{2}\rho g\delta_{k-1k}(\delta_{k-1k} + 2\sum_{i=k+1}^{N}\delta_{i-1i}) \end{cases}$$

and finally for the top layer

$$\begin{cases} \frac{\partial(\delta_{N-1N}\rho)}{\partial t} + \frac{\partial(\delta_{N-1N}\rho\bar{u}_N)}{\partial x} + \frac{\partial(\delta_{N-1N}\rho\bar{v}_N)}{\partial y} + \rho_N(\mathbf{v}_{\delta_N} - \mathbf{v}_s)\cdot\mathbf{n}_N = 0 \\ \frac{\partial(\rho\bar{u}_N\delta_{N-1N})}{\partial t} + \frac{\partial(\rho\bar{u}_N^2\delta_{N-1N} + \frac{1}{2}\rho g\delta_{N-1N}^2)}{\partial x} + \frac{\partial(\rho\bar{u}_N\bar{v}_N\delta_{N-1N})}{\partial y} + \rho u_{\delta_N}(\mathbf{v}_{\delta_N} - \mathbf{v}_s)\cdot\mathbf{n}_N = \\ \qquad -\rho g\delta_{N-1N}\frac{\partial\delta_{N-1}}{\partial x} + [\frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_N} - \frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_{N-1}}] \\ \frac{\partial(\rho\bar{v}_N\delta_{N-1N})}{\partial t} + \frac{\partial(\rho\bar{u}_N\bar{v}_N\delta_{N-1N})}{\partial x} + \frac{\partial(\rho\bar{v}_N^2\delta_{N-1N} + \frac{1}{2}\rho g\delta_{N-1N}^2)}{\partial y} + \rho v_{\delta_N}(\mathbf{v}_{\delta_N} - \mathbf{v}_s)\cdot\mathbf{n}_N = \\ \qquad -\rho g\delta_{N-1N}\frac{\partial\delta_{N-1}}{\partial y} + [\frac{1}{2}\mu\frac{\partial v}{\partial z}|_{\delta_N} - \frac{1}{2}\mu\frac{\partial v}{\partial z}|_{\delta_{N-1}}] \\ \bar{p}_N\delta_{N-1N} = \frac{1}{2}\rho g\delta_{N-1N}^2 \end{cases} \tag{3.31}$$

Notice that for the surface layer, most of the terms, such as $(\delta_N - \delta_k)$, simplify naturally when k = N.
Now that we have a final expression of the set of equations, we can proceed to study the nature of the system and choose an appropriate numerical resolution technique.

3.3.2 Hyperbolicity of the Equations

A useful step needed before choosing which numerical method to implement, is the study of the nature of the equations, meaning in which framework they can be classified. To begin, the discussion focuses on the generic stratum k and its associated set of equations, then we treat the boundary terms which may couple the layers. We do not focus on the complete set of equations, because the advected variables belong exclusively to each k-th system, granting a block diagonal structure of the Jacobians, which uncouple the different layers. This analysis can be carried out with both PE and PEV^2 hypothesis, since the second one does not add convective terms, but it contributes only diffusively. The k-th system appears very similar to a quasilinear system of conservation laws, therefore we will have to individuate the vector of

conserved quantities U and to compute the Jacobian with respect to U of the flux function. Since, due to integration, the setting we are dealing with is bidimensional, there exist two flux functions, which will be called $f(U)$ and $g(U)$, and the investigation on the eigenvalues will involve their Jacobians both separately and linearlly combined.
The physical evolution is led by the variables under temporal derivative, so that the conserved vector results in $U = (\rho\delta_{k-1k}, \rho\delta_{k-1k}\bar{u}_k, \rho\delta_{k-1k}\bar{v}_k)^T$. For readability reasons, we adopt the notation introduced by Kurganov in [11], defining the thickness $h = \rho\delta_{k-1k}$ and the discharge in the tangential directions $q^x = \bar{u}_k h$, $q^y = \bar{v}_k h$, which result in $U = (h, q^x, q^y)$. Regarding the flux functions, the one derived in the x direction is

$$F(U) = \begin{bmatrix} \rho\delta_{k-1k}\bar{u}_k \\ \rho\delta_{k-1k}\bar{u}_k^2 + \frac{1}{2}\rho g\delta_{k-1k}^2 \\ \rho\delta_{k-1k}\bar{u}_k\bar{v}_k \end{bmatrix}, \quad G(U) = \begin{bmatrix} \rho\delta_{k-1k}\bar{v}_k \\ \rho\delta_{k-1k}\bar{u}_k\bar{v}_k \\ \rho\delta_{k-1k}\bar{v}_k^2 + \frac{1}{2}\rho g\delta_{k-1k}^2 \end{bmatrix}, \tag{3.32}$$

or with the simplified notation

$$F(U) = \begin{bmatrix} q^x \\ \frac{(q^x)^2}{h} + \frac{1}{2}\frac{g}{\rho}h^2 \\ \frac{q^x q^y}{h} \end{bmatrix}, \quad G(U) = \begin{bmatrix} q^y \\ \frac{q^x q^y}{h} \\ \frac{(q^y)^2}{h} + \frac{1}{2}\frac{g}{\rho}h^2 \end{bmatrix}. \tag{3.33}$$

Afterwards are showed the Jacobian matrices with respect to U, expliciting the original variables:

$$F'(U) = \begin{bmatrix} 0 & 1 & 0 \\ -\bar{u}_k^2 + \frac{g}{\rho}h & 2\bar{u}_k & 0 \\ -\bar{u}_k\bar{v}_k & \bar{v}_k & \bar{u}_k \end{bmatrix}, \quad G'(U) = \begin{bmatrix} 0 & 0 & 1 \\ -\bar{u}_k\bar{v}_k & \bar{v}_k & \bar{u}_k \\ -\bar{v}_k^2 + \frac{g}{\rho}h & 0 & 2\bar{v}_k \end{bmatrix}, \tag{3.34}$$

and it is not difficult to see that they have the same structure, hence the calculation of the eigenvalue are presented completely in one case. The characteristic polynomial associated to the first matrix is the following

$$(\bar{u}_k - \lambda)\left(2\bar{u}_k\lambda - \lambda^2 - \bar{u}_k^2 + \frac{g}{\rho}h\right), \tag{3.35}$$

which can be easily solved as

$$\lambda_1 = \bar{u}_k, \quad \lambda_{2,3} = \bar{u}_k \pm \sqrt{\frac{g}{\rho}h}. \tag{3.36}$$

The eigenvalues of the Jacobian $G'(U)$ are obtained in a perfectly analogous way and the result is almost identical, but have $\bar{v}_k$ instead of $\bar{u}_k$. The problem to be faced now is to determine whether those eigenvalues are real, which is quite obvious since the do thickness cannot physically be negative, but it is always positive or at most null. Finally, one can observe that in the case of one layer, the result $\lambda_{2,3} = \bar{u}_k \pm \sqrt{\frac{g}{\rho}h} = \bar{u} \pm \sqrt{g\delta}$ is identical to the one obtained in the literature of shalow water.
The aforementioned check can be executed, with a series of long calculation that we will omit, also for any linear combination of the Jacobian matrices. For the sake of completeness, we just report the resulting eigenvalues obtained by Marques in [12], which are

$$\lambda_1 = \tilde{\alpha}\bar{u}_k + \tilde{\beta}\bar{v}_k, \quad \lambda_{2,3} = \lambda_1 \pm \sqrt{\frac{g}{\rho}h}, \tag{3.37}$$

being $\tilde{\alpha}$ and $\tilde{\beta}$ the generic real coefficients of the linear combination.
Having checked that the eigenvalues are real, we can affirm in first instance that our system is hyperbolic.

3.3.3 Boundary Terms Influence

Boundary conditions involve the interaction with other layers, hence their action may act as a mechanism to couple two different strata, modifying the block diagonal structure of the Jacobians of the flux functions. Fortunately, most of the boundary terms affected by derivation are not conserved quantities. Indeed, for the generic layer k the result of $(\delta_N - \delta_k)$ does not depend on the unknown δ_{k-1k}, therefore they should not alter the outcome on the nature of the system. Furthermore, derivation in normal component may not be accounted as a convective derivative, since the problem is now bidimensional, and therefore some quantities may be left out of the advection.
Nevertheless, we cannot say a priori that the source thickness terms, e.g. $\rho g \delta_{k-1k} \frac{\partial(\delta_N - \delta_k)}{\partial x}$ does not affect the flux functions, with possible modification of their Jacobians and consequently of theirs eigenvalues. The problem is widely treated by Audusse in [20], in which the author reaches the conclusion that, for suitable water height data, the non conservative terms in the right-hand side are neglectable, mantainig unaltered the hyperbolic nature of the equations.

3.3.4 System in Conservative Form

To conclude this chapter, we present a the final summarized set of equations for the generic k-th layer with notation used when studing the hyperbolicity of the system, which is

$$\mathbf{U}_k = \begin{bmatrix} \rho\delta_{k-1k} \\ \rho\delta_{k-1k}\bar{u}_k \\ \rho\delta_{k-1k}\bar{v}_k \end{bmatrix} = \begin{bmatrix} h_k \\ q_k^x \\ q_k^y \end{bmatrix} = \begin{bmatrix} U_{1,k} \\ U_{2,k} \\ U_{3,k} \end{bmatrix}. \tag{3.38}$$

The expression for the bottom or top layer are simply particular case of the following general system and therefore omitted. To obtain the most simple and clean expression possible, we also suppose the absence of viscous stresses depending on the vertical velocity.

$$\begin{cases} \frac{\partial U_{1,k}}{\partial t} + \frac{\partial U_{2,k}}{\partial x} + \frac{\partial U_{3,k}}{\partial y} = 0 \\ \frac{\partial U_{2,k}}{\partial t} + \frac{\partial [\bar{u}_k U_{2,k} + \frac{1}{2}\frac{g}{\rho}U_{1,k}^2]}{\partial x} + \frac{\partial \bar{v}_k U_{2,k}}{\partial y} = -gU_{1,k}\frac{\partial(\delta_N - \delta_k)}{\partial x} - gU_{1,k}\frac{\partial \delta_{k-1}}{\partial x} + [\frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_k} - \frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_{k-1}}] \\ \frac{\partial U_{3,k}}{\partial t} + \frac{\partial \bar{u}_k U_{3,k}}{\partial x} + \frac{\partial [\bar{v}_k U_{3,k} + \frac{1}{2}\frac{g}{\rho}U_{1,k}^2]}{\partial y} = -gU_{1,k}\frac{\partial(\delta_N - \delta_k)}{\partial y} - gU_{1,k}\frac{\partial \delta_{k-1}}{\partial y} + [\frac{1}{2}\mu\frac{\partial v}{\partial z}|_{\delta_k} - \frac{1}{2}\mu\frac{\partial v}{\partial z}|_{\delta_{k-1}}] \\ \bar{p}_k = \frac{1}{2}\rho g(\delta_{k-1k} + 2\sum_{i=k+1}^{N}\delta_{i-1i}) \end{cases} \tag{3.39}$$

Presenting the system in this form, it can be clearly seen the parallelism of the left-hand side with Saint-Venant shallow water equations. For a more accurate comparison, one can check Castro and Macias [19] or Kurganov [26] papers, where, before defining a multilayer resolutive numerical method, they regain the average equations. Therefore, at least for the convective part, we are legitimated to exploit techniques implemented for this category of problems.

4 Numerical Method

In this chapter, after a brief recall to the finite volume theory, will be presented Kurganov method for the upwind discretization of the convective term of the thin film equation. Successively, an ad hoc method for the source contribution is treated vastly when the problem is solved assuming hydrostatic pressure (PE). The discussion on the additional endowment played by vertical viscosity is introduced only theoretically, but it has not been tested or implemented. Those techniques are applied identically to each separate layer, since their equations have the same structure, except form the addition of bottom or interface boundary conditions, which, however, are known retrievable values.

4.1 Finite Volume

Finite volume methods is well known and vastly used for the discretization of differential problems in conservative form, which can be summarized in the following expression

$$\frac{\partial U}{\partial t} + \bigtriangledown \cdot (\mathbf{F}(U)) = s(U), \quad \mathbf{x} \in \Omega,\ t > 0, \tag{4.1}$$

where the conserved quantity is $U : (\mathbf{x}, t) \to \mathbb{R}$, $\mathbf{x} \in \Omega \subset \mathbb{R}^d$ for d = 2,3 and $\mathbf{F}$ is the so-called flux function, which can be either linear or not. The key idea is to partition the whole domain into time independent control volumes Ω_i for i=1,2,...,M such that $\bigcup_i^M \bar{\Omega}_i = \bar{\Omega}$, or at least it provides an acceptable approximation.
There are various possible choices for the control volumes, or cells if we are not in a threeedimensional environment. The most frequently used are the cell-centered and vertex-centered methods, which generally have as common starting point, the triangulation $\mathcal{T}_h$ of the domain in elements of the same shape, typically triangles or squares in 2D and tetrahedral or hexahedral in 3D. The difference regards the storage of the conservative variable, which for cell-centered approach is in fact in the center of the triangulation elements, as it can be seen in Fig. 4.1 a). Instead, in vertex-centered method, the control volumes are built using the middle point of edges connecting the vertices of the grid elements and their baricenter, in such a way that the variables will be more naturally stored in the triangulation nodes. The software used in this work relies on the second technique described, building internally a dual grid.
Hence, acting on the integral form of a general conservative equation and taking separately the contribution of each control volume, the integral form results in

$$\frac{\partial}{\partial t} \int_{\Omega_i} U dV + \int_{\partial \Omega_i} \mathbf{F}(U) \mathbf{n}_i d\sigma = \int_{\Omega_i} s(U) dV, \tag{4.2}$$

where we exchanged integral and time derivative due to the fact that the control volume does not vary over time and the divergence theorem has been applied. Additionally, we can further

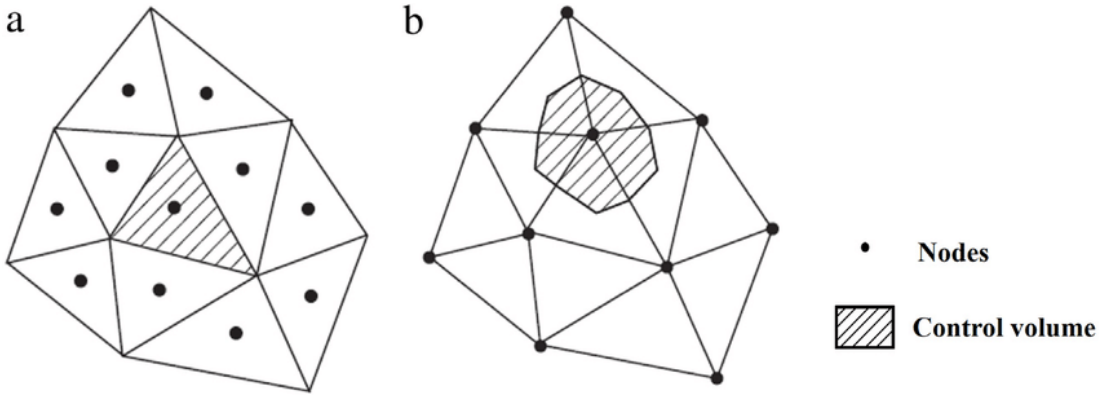

Figure 4.1: Cell and Vertex centered control volumes.

simplify the boundary integral indicating the normal to each side of the element as $\mathbf{n}_{ij}$, for i,j = 1,...,L, being L the total number of faces of the control volume. An ulterior step is to consider the conserved quantities averaged in space, indicating it with an overlaying bar, in order to write:

$$\frac{d}{dt}\bar{U}_i + \frac{1}{V_i}\sum_{j=1}^{L}\int_{A_{ij}} \mathbf{F}(U)\mathbf{n}_{ij}d\sigma = \frac{1}{V_i}\int_{\Omega_i} s(U)dV, \tag{4.3}$$

naming $V_i = \| \Omega_i \|$ the volume of Ω_i and A_{ij} its boundary faces.
The next step required is the discretization of the flux function with the numerical flux, which consist in the substitution of the integral on the border with the sum of approximated contributions of the fluid mass entering or exiting the control volume. The definition and the specification of the formula for the flux determines the typology of method used. The specific one, used for this work, is described by Kurganov in its paper for Acta Numerica [11], which is the development of its previous work [23]. The symbol that will be utilized for the numerical flux on the face A_{ij} of the control volume Ω_i will be $\mathcal{F}_{ij} = \int_{A_{ij}} \mathbf{F}(U)\mathbf{n}_{ij}d\sigma$.
Regarding the source term, it needs to have an ad hoc discretization, which in our problem, will consist in classical techniques with the addition of a polynomial reconstruction of the profile via Hermite polynomials. More precise details will be discussed in the dedicated section.
The final choice that needs to be applied is the time discretization strategy, which can either be explicit or implicit. Before describing the possibilities, it may be useful to express the system of equations in a more practical form, i.e. its residual counterpart:

$$\frac{d}{dt}\bar{U}_i + \frac{1}{V_i}R_i(U) = 0 \quad for\ i = 1, ..., M, \tag{4.4}$$

where $R_i(U) = \sum_{j=1}^{L}\mathcal{F}_{ij} - \int_{\Omega_i} s(U)dV$ is the convective contribution plus eventual shear stresses, to which the integrated source is subtracted. Finally, we present the most common, and therefore the ones that have been implemented, techniques for the time evolution tracking, which for simplicity are only explicit.
Supposing to divide the time domain from $t_0 = 0$ to $t_N = T$ in instants of constant difference $\triangle t$, the easiest method is the **Euler Explicit** time integration, which reads as

$$\frac{\bar{U}_i^{n+1} - \bar{U}_i^n}{\triangle t} + \frac{1}{V_i}R_i(U^n) = 0, \quad for\ n = 0, ..., N-1. \tag{4.5}$$

In this case, as well as in the other explicit discretizations, the solution update follows immediately because the values at the previous time step are all known and are substituted directly

in the residual expression, obtaining

$$\triangle U_i^n = -\frac{\triangle t}{V_i} R_i(U^n), \tag{4.6}$$

indicating the difference between conserved quantities as $\triangle U_i^n = \bar{U}_i^{n+1} - \bar{U}_i^n$.
Another algorithm implemented is the classical 4 steps **Runge-Kutta**, which starts exactly as Explicit Euler, but it requires more than one iteration to retrieve a solution, reason why it belongs to the category of Multi-Steps methods. In its most general form, a Runge-Kutta procedure of order S can be written in the form

$$U_{n+1} = U_n + hH(t_n, U_n, h; R) \quad n \geq 0, \tag{4.7}$$

being R the residual of an arbitrary control volume and H the increment function, defined as

$$\begin{aligned} H(t_n, U_n, h; R) &= \sum_{s=1}^{S} b_s K_s \\ K_s &= -R(t_n + c_s h, U_n + h \sum_{q=1}^{S} a_{sq} K_q) \quad for\ s = 1, 2, ..., S, \end{aligned} \tag{4.8}$$

where the coefficients a_{sq}, b_s and c_s are what really defines the typology of Runge-Kutta, aside from h, which is the difference between the two consequent time instants. Indeed, if the desired integration is explicit, all the a_{sq} with $q \geq s$ should be null, in order to use only the K_s computed at previous steps. Moreover, to obtain consistency, it has been verified that the condition $\sum_{s=1}^{S} b_s = 1$ must hold. Having done this preliminary considerations, we simply report the classical and most diffused explicit Runge-Kutta, which implies the iteration of 4 steps, i.e.

$$\begin{aligned} U_{n+1} &= U_n + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ K_1 &= -R(t_n, U_n), \\ K_2 &= -R(t_n + \frac{h}{2}, U_n + \frac{h}{2} K_1), \\ K_3 &= -R(t_n + \frac{h}{2}, U_n + \frac{h}{2} K_2), \\ K_4 &= -R(t_{n+1}, U_n + hK_3). \end{aligned} \tag{4.9}$$

A final remark which underlines the powerfullness of this method, is that it achieves the 4^{th} order of convergence in time when solving an Ordinary Differential Equation.
For the sake of completeness, we report also what would happen using an implicit technique, in the specific case **Euler Implicit** time discretization. Unlike the explicit case, an implicit approach requires the solution of the linear system derived from the expansion of the residual $R_i(U^{n+1})$, namely

$$R_i(U^{n+1}) = R_i(U^n) + \frac{\partial R_i(U^n)}{\partial t} \triangle t + \mathcal{O}(\triangle t^2) = R_i(U^n) + \sum_{j=1}^{N_f} \frac{\partial R_i(U^n)}{\partial U_j^n} \triangle U_i^n + \mathcal{O}(\triangle t^2) \tag{4.10}$$

calling N_f the total number of faces of the control volume taken in consideration.
The resulting expression for the implicit system linearized is

$$\left(\frac{V_i}{\triangle t} \delta ij + \frac{\partial R_i(U^n)}{\partial U_j^n} \right) \triangle U_i^n = -R_i(U^n) \quad for\ j = 1, ..., N_f, \tag{4.11}$$

where δij is the Kronecker's delta.

4.2 Kurganov Central Upwind Scheme

In order to complete the numeric scheme, we need to discretize the residual function for each control volume i and at each time instant n. For the model provided in chapter 3, assuming the PE hypothesis, we observe that $R_i(U^n)$ is given by the sum of a convective and a source contribution. For the advection part, we rely on the numerous recent work carried on by Kurganov regarding shallow water, which, given the similarities with the equation already explained, can be translated and adapted to the thin film world. For the source part, since we want to avoid the addition of unknown terms to estimate normal stresses, a polynomial interpolation strategy will be presented.

4.2.1 One- and Two-Dimensional Central-Upwind Scheme

Since the modeling equations are averaged, the approximating scheme needed for discretizing the flux can be presented only for the mono and bidimensional case. To present the 1-D approach, which dates back to the original Godunov scheme, the control volume selected is the Cartesian product of the two simpliest space-time intervals $[x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}] \times [t^n, t^{n+1}]$, which reduces the general complexity of the exposition. Furthermore, as a semplification of the specific multilayer technique, the method is described for a single layer, allowing the omission of the subscript k. In order to evaluate the numerical flux, i.e. the advetcion part of the subsequent equation, with no source,

$$\frac{d\bar{\mathbf{U}}}{dt} = -\frac{1}{\triangle x}[\mathcal{F}_{j+\frac{1}{2}}(t) - \mathcal{F}_{j-\frac{1}{2}}(t)], \tag{4.12}$$

one has to approximately solve the following Riemann problem with the initial data prescribed at time $t = t^n$ such that

$$\frac{d\mathbf{U}}{dt} + \frac{dF(\mathbf{U})}{dx} = 0 \quad t \in (t, t+\tau] \quad \text{with}$$

$$\mathbf{U}(x,t) = \begin{cases} \mathbf{U}^-_{j+\frac{1}{2}}(t) & \text{if } x < x_{j+\frac{1}{2}} \\ \mathbf{U}^+_{j+\frac{1}{2}}(t) & \text{if } x > x_{j+\frac{1}{2}} \end{cases}, \tag{4.13}$$

where τ is a small positive number and $\mathbf{U}^+_{j+\frac{1}{2}}(t)$ and $\mathbf{U}^-_{j+\frac{1}{2}}(t)$ are the right and left side values of the piece-wise polynomial interpolant, namely

$$\mathbf{U}^-_{j+\frac{1}{2}}(t) = \mathcal{P}_j(x_{j+\frac{1}{2}}; t) \text{ and } \mathbf{U}^+_{j+\frac{1}{2}}(t) = \mathcal{P}_{j+1}(x_{j+\frac{1}{2}}; t), \tag{4.14}$$

where $\mathcal{P}_j$ and $\mathcal{P}_{j+1}$ belong to the space of discontinuous interpolant polynomial. As a reference for the polynomial reconstruction of first order, we present it in the form where the interface quantities are deduced with the aid of the derivative, i.e.

$$\mathbf{U}^-_{j+\frac{1}{2}}(t) = \bar{\mathbf{U}}_j + \frac{\triangle x}{2}\left(\frac{d\mathbf{U}}{dx}\right)_j \text{ and } \mathbf{U}^+_{j+\frac{1}{2}}(t) = \bar{\mathbf{U}}_{j+1} - \frac{\triangle x}{2}\left(\frac{d\mathbf{U}}{dx}\right)_{j+1}, \tag{4.15}$$

The reconstruction is performed at the interface of the control volume to achieve an higher accuracy for the numerical method.
Once computed the conserved values at the interface $x_{j+\frac{1}{2}}$, in its works Kurganov proposes a central-upwind scheme to solve the problem mentioned above, which takes advantage of the direction of the fluxes. Indeed, the one-sided wave speed propagation is computed at the cell interface and can be estimated by the upper and lower eigenvalues of the flux function's

Jacobian, which are respectively $\lambda_n = \bar{u} + \sqrt{g\delta_{k-1k}}$ and $\lambda_1 = \bar{u} - \sqrt{g\delta_{k-1k}}$ computed in (3.36). In most cases, reliable estimates are given by

$$a^+_{j+\frac{1}{2}}(t) = max\left\{\lambda_n\left(\frac{\partial F}{\partial U}(U^-_{j+\frac{1}{2}}(t))\right), \lambda_n\left(\frac{\partial F}{\partial U}(U^+_{j+\frac{1}{2}}(t))\right), 0\right\}, \tag{4.16a}$$

$$a^-_{j+\frac{1}{2}}(t) = min\left\{\lambda_1\left(\frac{\partial F}{\partial U}(U^-_{j+\frac{1}{2}}(t))\right), \lambda_1\left(\frac{\partial F}{\partial U}(U^+_{j+\frac{1}{2}}(t))\right), 0\right\}. \tag{4.16b}$$

Finally, we present the formula for numerical flux at the interface

$$\mathcal{F}_{j+\frac{1}{2}} = \frac{a^+_{j+\frac{1}{2}}F(U^-_{j+\frac{1}{2}}) - a^-_{j+\frac{1}{2}}F(U^+_{j+\frac{1}{2}})}{a^+_{j+\frac{1}{2}} - a^-_{j+\frac{1}{2}}} + \frac{a^+_{j+\frac{1}{2}}a^-_{j+\frac{1}{2}}}{a^+_{j+\frac{1}{2}} - a^-_{j+\frac{1}{2}}}[U^+_{j+\frac{1}{2}} - U^-_{j+\frac{1}{2}} - \delta U_{j+\frac{1}{2}}], \tag{4.17}$$

where $\delta U_{j+\frac{1}{2}}$ is the so-called built-in anti-diffusion term, whose purpose is to prevent the arise of numerical diffusion, which adds up to the natural fluid viscosity slowing down the flow. It can be computed as

$$\delta U_{j+\frac{1}{2}} = minmod\left(U^+_{j+\frac{1}{2}} - U^*_{j+\frac{1}{2}}, U^*_{j+\frac{1}{2}} - U^-_{j+\frac{1}{2}}\right), \tag{4.18}$$

using the minmod function

$$minmod(z_1, z_2, ...) = \begin{cases} min(z_1, z_2, ...) & if\ z_i > 0\ for\ all\ i, \\ max(z_1, z_2, ...) & if\ z_i < 0\ for\ all\ i, \\ 0 & otherwise. \end{cases} \tag{4.19}$$

The term specified with a star represents an expression of the intermediate values, which takes into account the estimate of the wave propagation speed and it can be found by

$$U^*_{j+\frac{1}{2}} = \frac{a^+_{j+\frac{1}{2}}U^+_{j+\frac{1}{2}} - a^-_{j+\frac{1}{2}}U^-_{j+\frac{1}{2}} - [F(U^+_{j+\frac{1}{2}}) - F(U^-_{j+\frac{1}{2}})]}{a^+_{j+\frac{1}{2}} - a^-_{j+\frac{1}{2}}}. \tag{4.20}$$

Again, for the sake of simplicity of the 2-D setting, we describe the central-upwind scheme on a bidimensional Cartesian grid $C_{j,i} := [x_{j-\frac{1}{2},i}, x_{j+\frac{1}{2},i}] \times [y_{j,i-\frac{1}{2}}, y_{j,i+\frac{1}{2}}]$, in order to have the unknown coordinates centered in the grid vertex.
The discretization is applied to the two numerical fluxes, which are

$$F(U) = \begin{bmatrix} U_2 \\ \bar{u}U_2 + \frac{1}{2}\frac{1}{\rho}gU_1^2 \\ \bar{u}U_3 \end{bmatrix} \quad G(U) = \begin{bmatrix} U_3 \\ \bar{v}U_2 \\ \bar{v}U_3 + \frac{1}{2}\frac{1}{\rho}gU_1^2 \end{bmatrix}, \tag{4.21}$$

whose Jacobians have eigenvalues respectively $\lambda_1 = \bar{u} - \sqrt{g\delta_{k-1k}}$, $\lambda_2 = \bar{u}$, $\lambda_3 = \bar{u} + \sqrt{g\delta_{k-1k}}$ and $\lambda_1 = \bar{v} - \sqrt{g\delta_{k-1k}}$, $\lambda_2 = \bar{v}$, $\lambda_3 = \bar{v} + \sqrt{g\delta_{k-1k}}$.
The mean vector of conserved quantities $\bar{U}$ is needed to evaluate the slopes at the boundaries of the Cartesian control volume in the following way

$$U^W_{j,i}(t) = \bar{U}_{j,i} - \frac{\triangle x}{2}\partial_x U_{j,i}, \quad U^E_{j,i}(t) = \bar{U}_{j,i} + \frac{\triangle x}{2}\partial_x U_{j,i}, \tag{4.22}$$

$$U^S_{j,i}(t) = \bar{U}_{j,i} - \frac{\triangle y}{2}\partial_y U_{j,i}, \quad U^N_{j,i}(t) = \bar{U}_{j,i} + \frac{\triangle y}{2}\partial_y U_{j,i}. \tag{4.23}$$

Due to the upwind nature of the scheme, the estimation of the propagation speed is retrieved

by the maximum and minimum eigenvalues of the Jacobians, resulting in

$$a^+_{j+\frac{1}{2},i}(t) = max\left\{\lambda_n\left(\frac{\partial F}{\partial U}(U^E_{j,i}(t))\right), \lambda_n\left(\frac{\partial F}{\partial U}(U^W_{j+1,i}(t))\right), 0\right\} \tag{4.24a}$$

$$a^-_{j+\frac{1}{2},i}(t) = min\left\{\lambda_1\left(\frac{\partial F}{\partial U}(U^E_{j,i}(t))\right), \lambda_1\left(\frac{\partial F}{\partial U}(U^W_{j+1,i}(t))\right), 0\right\} \tag{4.24b}$$

$$b^+_{j,i+\frac{1}{2}}(t) = max\left\{\lambda_n\left(\frac{\partial G}{\partial U}(U^N_{j,i}(t))\right), \lambda_n\left(\frac{\partial G}{\partial U}(U^S_{j,i+1}(t))\right), 0\right\} \tag{4.24c}$$

$$b^-_{j,i+\frac{1}{2}}(t) = min\left\{\lambda_1\left(\frac{\partial G}{\partial U}(U^N_{j,i}(t))\right), \lambda_1\left(\frac{\partial G}{\partial U}(U^S_{j,i+1}(t))\right), 0\right\} \tag{4.24d}$$

where (4.24a) and (4.24b) are evaluated in x direction, while the remaining along y.
The central-upwind schemes may be significantly simplified if one passes to a semi-discrete limit by taking $max_n \triangle t^n \to 0$, leading to the complex expression for the numerical fluxes:

$$\mathcal{F}_{j+\frac{1}{2},i} = \frac{a^+_{j+\frac{1}{2},i}F(U^E_{j,i}) - a^-_{j+\frac{1}{2},i}F(U^W_{j+1,i})}{a^+_{j+\frac{1}{2},i} - a^-_{j+\frac{1}{2},i}} + \frac{a^+_{j+\frac{1}{2},i}a^-_{j+\frac{1}{2},i}}{a^+_{j+\frac{1}{2},i} - a^-_{j+\frac{1}{2},i}}[U^W_{j+1,i} - U^E_{j,i} - \delta U_{j+\frac{1}{2},i}], \tag{4.25}$$

$$\mathcal{G}_{j,i+\frac{1}{2}} = \frac{b^+_{j,i+\frac{1}{2}}G(U^N_{j,i}) - b^-_{j,i+\frac{1}{2}}G(U^S_{j,i+1})}{b^+_{j,i+\frac{1}{2}} - b^-_{j,i+\frac{1}{2}}} + \frac{b^+_{j,i+\frac{1}{2}}b^-_{j,i+\frac{1}{2}}}{b^+_{j,i+\frac{1}{2}} - b^-_{j,i+\frac{1}{2}}}[U^S_{j,i+1} - U^N_{j,i} - \delta U_{j,i+\frac{1}{2}}]. \tag{4.26}$$

In the previous formula for the flux $\mathcal{F}$, the term $\delta U_{j+\frac{1}{2},i}$ is the anti-diffusion term, which reads as

$$\delta U_{j+\frac{1}{2},i} = minmod\left(U^W_{j+1,i} - U^*_{j+\frac{1}{2},i}, U^*_{j+\frac{1}{2},i} - U^E_{j,i}\right), \tag{4.27}$$

The estimation of the intermediate value of the conserved quantities is

$$U^*_{j+\frac{1}{2},i} = \frac{a^+_{j+\frac{1}{2},i}U^W_{j+1,i} - a^-_{j+\frac{1}{2},i}U^E_{j,i} - [F(U^W_{j+1,j}) - F(U^E_{j,i})]}{a^+_{j+\frac{1}{2},i} - a^-_{j+\frac{1}{2},i}}. \tag{4.28}$$

For its counterpart $\mathcal{G}$ in the other direction, the analogous expressions are

$$\delta U_{j,i+\frac{1}{2}} = minmod\left(U^S_{j,i+1} - U^*_{j,i+\frac{1}{2}}, U^*_{j,i+\frac{1}{2}} - U^N_{j,i}\right), \tag{4.29}$$

$$U^*_{j,i+\frac{1}{2}} = \frac{b^+_{j,i+\frac{1}{2}}U^S_{j,i+1} - b^-_{j,i+\frac{1}{2}}U^N_{j,i} - [G(U^S_{j,i+1}) - G(U^N_{j,i})]}{b^+_{j,i+\frac{1}{2}} - b^-_{j,i+\frac{1}{2}}}. \tag{4.30}$$

All this formula can be identically translated into the fluxes entering from the other side of the control volume simply substituting the subscript $j+\frac{1}{2},i$ with $j-\frac{1}{2},i$ or $j,i+\frac{1}{2}$ with $j,i-\frac{1}{2}$.
Substituting the numerical fluxes, we obtain the following ODE

$$\frac{d}{dt}\bar{U}_{j,i} = -\frac{\mathcal{F}_{j+\frac{1}{2},i} - \mathcal{F}_{j-\frac{1}{2},i}}{\triangle x} - \frac{\mathcal{G}_{j,i+\frac{1}{2}} - \mathcal{G}_{j,i-\frac{1}{2}}}{\triangle y} + S(U_{j,i}), \tag{4.31}$$

whose terms on the right-hand side are all known after the consideration made in the next section.
As final considerations, two important properties of the scheme must be enounced. The first is the so called 'lake at rest' condition, which states that, if the incoming flux is null and the thickness is constant, i.e. $q^x = q^y = 0$ in the whole fluid and $\delta_N = const.$, the convection is annihilated by the source terms. Therefore, in this designed situation, the numerical flux should be equal the to real flux. This fact can be easily verified since if those circumstances are met,

then the conserved variables are equal in all the control volume cells, implying that also their value at the interface are identical, e.g. in one dimension $U^+_{j+\frac{1}{2}} = U^-_{j+\frac{1}{2}} = U_{j+\frac{1}{2}}$. Hence, it can be observed that, setting the horizontal velocity equal to zero, the propagation speed result opposite in sing, resulting in $a^+_{j+\frac{1}{2},i} = -a^-_{j+\frac{1}{2},i}$. Moreover, with simple calculation, it can be computed that the built-in anti-diffusion term $\delta U_{j+\frac{1}{2}}$ is null, since the intermediate expression $U^*_{j,i+\frac{1}{2}}$ has the same value of $U^+_{j+\frac{1}{2}}$ and $U^-_{j+\frac{1}{2}}$. Finally, substituting all the aforementioned consideration in the numerical flux formula, we obtain that

$$\mathcal{F}_{j+\frac{1}{2}} = \frac{a^+_{j+\frac{1}{2}} F(U_{j+\frac{1}{2}}) + a^+_{j+\frac{1}{2}} F(U_{j+\frac{1}{2}})}{2a^+_{j+\frac{1}{2}}} = F(U_{j+\frac{1}{2}}) = \frac{\partial\left(\frac{1}{2}\frac{1}{\rho} g U_1^2\right)}{\partial x}, \tag{4.32}$$

if we consider the momentum equation in x direction and remind that $q^x = 0$. Computing effectively this partial derivative we obtain

$$\frac{1}{2}\frac{1}{\rho} g \frac{\partial U_1^2}{\partial x} = \rho g \delta_{k-1k} \frac{\partial \delta_{k-1k}}{\partial x}. \tag{4.33}$$

The last step to perform, is to manipulate the source term, consisting only of $-\rho g \delta_{k-1k} \frac{\partial(\delta_N - \delta_k)}{\partial x} - \rho g \delta_{k-1k} \frac{\partial \delta_{k-1}}{\partial x}$, because the velocity and consequently the stresses are null in the whole fluid, and highlight the matching with (4.33). This can be done exploiting the linearity of the derivation, showing that

$$-\rho g \delta_{k-1k} \frac{\partial(\delta_N - \delta_k + \delta_{k-1})}{\partial x} = -\rho g \delta_{k-1k} \frac{\partial \delta_N}{\partial x} + \rho g \delta_{k-1k} \frac{\partial \delta_{k-1k}}{\partial x} = \rho g \delta_{k-1k} \frac{\partial \delta_{k-1k}}{\partial x} \tag{4.34}$$

since, for hypothesis, $\delta_N = const$. The equality between (4.33) and (4.34) means that the 'lake at rest' condition is satisfied.
The second property of the Kurganov discretization is that his scheme is positivity-preserving, which ensures that the reconstructed thickness is non-negative and that it does not surpass the prescribed bottom surface. The conclusion is achieved by two elements that are added and linked to the discretization, which are a linear piece-wise approximation of the layers, in order for the slope not to cross the fundal, and a desingularization of the velocity, meaning that if the thickness decreases to zero, the velocity takes the value of the inferior layer.

4.2.2 Source Contribution Discretization

Source contribution can be summarized, recalling for example (3.39), in three contributions: a first part encloses the interaction between the layer heights, i.e. the sum of the thickness $\rho g \delta_{k-1k}$ multiplied by the derivative of the difference between the top and the k-th stratum $\frac{\partial(\delta_N - \delta_k)}{\partial x}$ and by the derivative of the inferior layer cumulative height $\frac{\partial \delta_{k-1}}{\partial x}$, a second piece constituted by the difference between vertical stress evalued exactly at the layer height $\frac{1}{2}\mu\left(\frac{\partial u}{\partial z}|_{\delta_k} - \frac{\partial u}{\partial z}|_{\delta_{k-1}}\right)$ and eventually a volumetric force, usually gravity $\rho g_x \delta_{k-1k}$.
The most straight forward way to discretize the thickness interaction terms is to treat it explicitly, like usually done with the gravitational force, and considering the conserved quantity constant inside the control volume, which is true if the area of Ω_i tends to zero, or equivalently when the mesh is very fine. Moreover, we suppose the same happens with the derivative, even if this assumption introduces an approximation error, because the control volumes are time invariant, but can change spatially throughout the domain. Adopting such treatment, the

expression obtained, for example presented in x direction, is

$$-\int_{\Omega_j} \rho g \delta_{k-1k} \frac{\partial(\delta_N - \delta_k)}{\partial x} dV = -\rho g A_j \bar{\delta}_{k-1k,j} \frac{\partial(\delta_N - \delta_k)_j}{\partial x},$$
$$-\int_{\Omega_j} \rho g \delta_{k-1k} \frac{\partial \delta_{k-1}}{\partial x} dV = -\rho g A_j \delta_{k-1k,j} \frac{\partial(\delta_{k-1})_j}{\partial x} \tag{4.35}$$

where $A_j = \| \Omega_j \|$, i.e. A_j is the area of the control volume Ω_j, and the pedix $_j$ represent the constant value computed exactly in the primary grid node $\mathbf{x}_j$.
The second contribution is given by the difference between vertical stress, therefore the decision adopted , unlike the classical literature in which the speed is expanded locally, is to reconstruct the profile of the velocity, using the mean values that can be regained by the conserved variables. To reconstruct the profile of a variable at a generic point knowing its mean value at the previous iteration, the technique applied is the polynomial interpolation via Hermite polynomial in integral form. Choosing Hermite is not random, indeed we can rely on the property $H_n'(z) = 2nH_{n-1}(z)$ to regain easily the normal stress from the velocity profile. The process used to recover the interpolant polynomial is different whether the analysis is carried for the exact value or the integrated one.
If we decided to approximate the mean velocity at the mid point between two adjacent layers, first of all, we approximate the real function with the sum of coefficients multiplied by the base polynomial up to the max degree M, i.e. $u(z) = \sum_{i=0}^{M} \alpha_i H_i(z)$ and then estimate the speed in the middle of the strata, i.e. $u(\frac{\delta_0+\delta_1}{2}), u(\frac{\delta_{k-1}+\delta_k}{2}), ..., u(\frac{\delta_{N-1}+\delta_N}{2})$. Hence, we can set up the following linear system

$$\sum_{i=0}^{M} \alpha_i H_i \left(\frac{\delta_{j-1} + \delta_j}{2} \right) dz = u \left(\frac{\delta_{j-1} + \delta_j}{2} \right) \quad for\ j = 1, ..., N, \tag{4.36}$$

solved for the unknown vector of coefficients α.
In the other possibility, the one implemented in the code, the approximation is performed in an integral form. The choice fell back to this alternative, because it is more consistent and does not require the strong assumption that the mean conserved variable is exactly evaluated in the middle point of two layers. Therefore, after writing $u(z)$ as the sum of basis function $H_i(z)$, the polynomial expression of the speed is integrated along the whole thickness, calling its extrema δ_0 and δ_N, where N is the total number of layers, resulting in $\int_{\delta_0}^{\delta_N} u(z)dz = \sum_{i=0}^{M} \int_{\delta_0}^{\delta_N} \alpha_i H_i(z)dz$. We can successively split the integral for each layer interval and carry out the coefficients, which do not depend on the normal coordinate, obtaining

$$\sum_{i=0}^{M} \alpha_i \sum_{k=1}^{N} \int_{\delta_{k-1}}^{\delta_k} H_i(z)dz = \sum_{k=1}^{N} \bar{u}_k \delta_{k-1k}, \tag{4.37}$$

that is a linear system having as unknowns the coefficients of the polynomial α_i for $i = 0, 1, ..., M$ and the integral of the basis Hermite polynomial stored in the matrix $\mathcal{H}$. Contrary to the previous case, the value of the velocity is not the punctual evaluation at a specific coordinate, but, accordingly to the mean value theorem, at a point called $\eta \in [\delta_{k-1}; \delta_k]$. Another useful precisation, is that we are taking as correct the hypothesis that the derivative of function is equal to the derivative of its polynomial reconstruction.
For consistency reasons, we also add to the system one equation for the value at the bottom, known by boundary conditions, i.e. the velocity evaluated exactly at δ_0, achieving a square system when the max order is $M = N + 1$. The identical procedure can be repeated for $\bar{v}$, the velocity in y direction.
The full-fledged discretization of the expanded and rearranged interaction terms, i.e. $\int_{\Omega_j} [\frac{1}{2}\mu \frac{\partial u}{\partial z}|_{\delta_k} -$

$\frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_{k-1}}]dV$ where j is now the index of the control volume associated with the j-th grid point, can be summarized in the following declaration

$$A_j\left[\frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_k}-\frac{1}{2}\mu\frac{\partial u}{\partial z}|_{\delta_{k-1}}\right]=A_j\left[\frac{1}{2}\mu\sum_{i=1}^{M}2i\alpha_i H_{i-1}(\delta_k)-\frac{1}{2}\mu\sum_{i=1}^{M}2i\alpha_i H_{i-1}(\delta_{k-1})\right], \quad (4.38)$$

reminding that α is the resulting vector of coefficient associated to the Hermite polynomials for the velocity, in this example in x direction.
For the sake of completeness, since the source contribution is being described, the explicit expression of the discretization of a possible volumetric force is presented in the form of a gravitational acceleration, for example in x direction, which is the most common bequest in thin film theory and reads as

$$\int_{\Omega_j}\rho g_x\delta_{k-1k}dV=A_j\rho g_x\delta_{k-1k,j}. \quad (4.39)$$

4.3 Primitive Equations with Vertical Viscosity

When considering the vertical viscosity contribution, the ulterior term which concours to oppose or favor the fluid motion is the summation of normal stress, which can be seen in as the expression of the expanded hypothesis (3.16). Therefore, in order to follow the guidelines of this work, i.e. trying to model the variables using the informations provided by the solution on the layers instead of expanding the unknowns and without adding extra constraints, we need a way to treat the z-component of the stress tensor $\underline{\underline{\tau}}$.
The simpliest solution is the application of Chezy formula for the bottom stress, as reported in [27], which states:

$$\tau_{xz}=\frac{g}{C^2}u(u^2+v^2)^{\frac{1}{2}}, \quad \tau_{yz}=\frac{g}{C^2}v(u^2+v^2)^{\frac{1}{2}}, \quad (4.40)$$

giving an evaluation using only the known components of the velocity. For what concerns the value of the so-called Chezy constant C, whose physical dimension is $[\frac{m^{\frac{1}{2}}}{s}]$, it can be retrieved in two different ways. The former estimate is given by the equality $u_\infty=C\sqrt{Ri}$, where u_∞ is the mean reference velocity, used also in the calculation of Reynolds number, R is the hydraulic radius, given by the cross-sectional area of flow divided by the wetted perimeter, and i is the hydraulic gradient, which for simple backdrop is equal to the bottom slope. The latter method is the application of another formula, which requires the knowledge of the material composing the bottom, because it requires Manning's roughness coefficient $\frac{1}{n}$, dimensionally equal to $[\frac{m^{\frac{1}{3}}}{s}]$, and it can be formulated as $C=\frac{1}{n}R^{\frac{1}{6}}$, again being R the hydraulic radius. Exploiting one of this relations, we can obtain the extra diagonal terms of the tensor and finally having all the information needed to move to the numerical discretization, which, for example, can use values at the previous time step for an explicit treatment.
For the normal stress, the scenario to be faced is a little more complex because the quantity to estimate at each layer interface is the second derivative of the normal velocity, which is the unknown variable neglected by the average process. Reconstructing its profile is often useful, and in this case necessary, although the procedure may be complicated if the desired accuracy is high. For the sake of completeness, we present and summarize the key principles of two methods described by Muccino *et al.* in [28]. Before starting, we recall that the vertical velocity reconstruction should be performed after the resolution of the multilayer system (3.24) at each time step.
Knowing that the traditional method consists in solving the full continuity equation neglecting

one boundary condition, the first improvement covered by the paper requires the derivation with respect to the z direction of the continuity equation, which results in

$$\frac{\partial^2 w}{\partial z^2} = -\frac{\partial}{\partial z} \bigtriangledown \cdot(\mathbf{v}), \tag{4.41}$$

where now $\mathbf{v}$ obviously is a 2 dimensional vector containing the tangential components of the speed (u, v). Since the obtained equation is second-order, it can be closed with the imposition of both boundary conditions.
A criticism of the approach just explained is that, having neglected the normal component, the solving algorithm can only rely on the normal coordinate in which a layer is instantiated, which can be an exiguous number, since the layers are few with respect to a complete discretization. Hence, the second method we are going to introduce tries to exploit the stratification of the domain imposed by the multilayer technique. The Least Square method starts from the original continuity equation taken under integration between adjacent nodes in the vertical direction z, obtaining the problem

$$w_{i+1} - w_i = -\int_{z_i}^{z_{i+1}} \bigtriangledown \cdot (\mathbf{v}) dz \quad \text{for i=1,2,...,N-1,} \tag{4.42}$$

having, just like in the previous case, the possibility to include both boundary conditions for $i = 0$ and $i = N$. Although the overdetermined system of equations cannot be solved uniquely, the Eulerian norm of the residual may be minimized according to the theory of least squares.
Having calculated those extra source terms, one can finally solve the multilayer thin film problem with the addition of vertical viscosity. As a final remark, we want to point out that both the application of Chezy formula and the Least Square reconstruction introduce approximation errors, that should be ensured to be neglectable.

5 | SU2 Code

SU2 is an open-source software, written in C++, developed at Stanford University by Dr. Francisco Palacios and Dr. Thomas D. Economon, downloadable from the official website [a]. It can perform many tasks depending on the C++ executable chosen, which, for the purpose of this work, is exclusively SU2_CFD. SU2_CFD primary applications are computational fluid dynamics and aerodynamic shape optimization, but has been extended to treat more general equations such as electrodynamics and chemically reacting flows. It can be run efficently in parallel and it supports the resolution of multiphysics problems, which is the case of thin film. The list of other components, such as SU2_DOT, which computes the partial derivative of a functional, or SU2_DEF, which retrieve the geometrical deformation of an aerodynamic surface and the surrounding volumetric grid, can be found in the site [a] under the voice *Installation/Software Components*.

5.1 General Structure and Prerequisites

To run a CFD simulation, only two files are needed: the first defines the mesh on the physical domain, which can be in the native '.su2' or in '.cgns' format, while the second is called configuration file, which is essentially a collection of all the option specific to the problem. The possible choices avaible in the '.cfg' file include the set of equation solved, physical parameters, the type of discretization applied to the convective and viscous part, informations on the linear system solver and many other features. There exists also the possibility to define a multiphysics problem, i.e. a scenario where two or more fluids are interacting, which is the case of thin film, or a multizone domain, i.e. same fluid on a partitioned geometry, that must be provided either two configuration files with their respective meshes or, enabling the option MULTIZONE_MESH, a single document which includes the number of zones and both tasselation, preceded by the indicator IZONE.
To introduce new features, one must update or create the enumeration corresponding to the desired alternative in the file option_structure.hpp, which will be introduced in a SU2-defined map containing the string of the option and its possible values. The settings added to include thin film problem are:

- FILM_SOLVER= (MULTI_LAYER_ASYMP, NAVIER_STOKES_FILM) select which resolutive method to use.
- NLAYER sets the number of layer, one by default to prevent improper use in other problems.
- FILM_HP= (PE, PEV2) switches the hypothesis for the pressure.
- BOTTOM_TOPOGRAPHY= (UNIFORM, STRAIGHT, PLANE, REAL_DATA) let

the user choose between the shape of the bottom, from simple known topography to real data which must be provided through an additional file.

- BOTTOM_VALUE= (q, α, β) sets respectively the normal intercept and the angles, measured in degrees, formed anti-clockwise with the positive portion of the axis.

Those choices are stored into newly created variables, contained in the class CConfig, which is passed as parameter to many of the functions invoked during the computations, granting the adoption of the right methodology chosen by the user.
After reading the configuration options, a driver, which is the leading object for the whole simulation, is initialized. The class CDriver, or one its children, contains all the necessary objects, clearly illustrated with the dotted lines in Fig. 5.1, to perform task such as run each time iteration, integrate the equation in space and time or storage of the solution. The continuous lines, instead, represent inheritance.

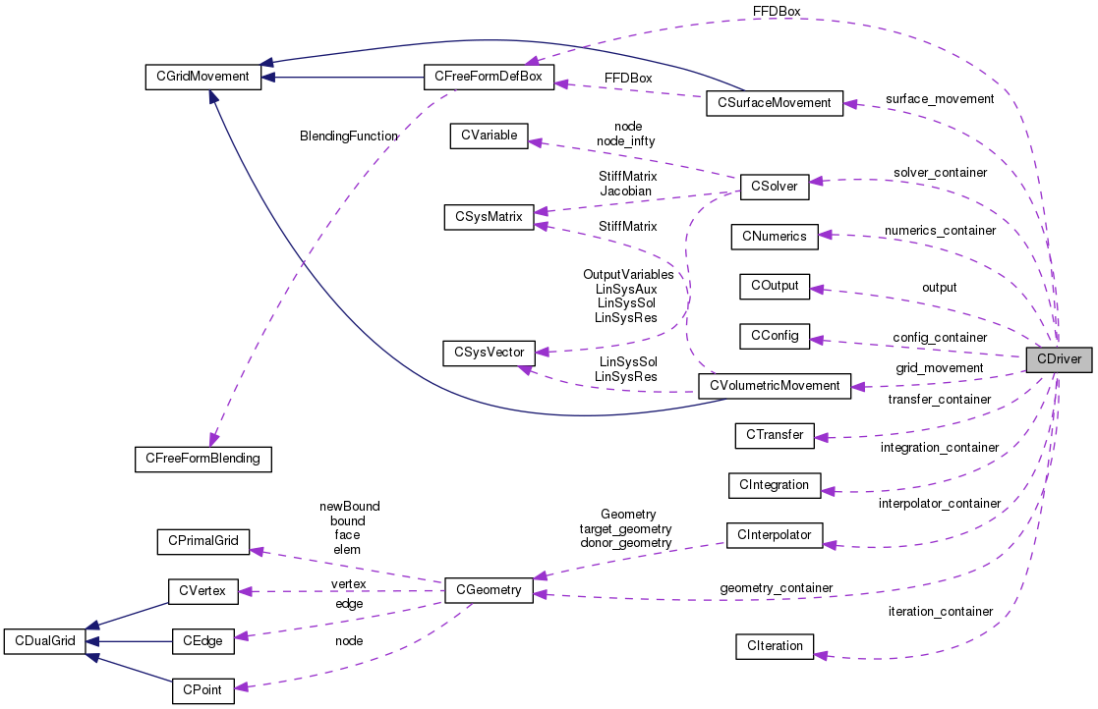

Figure 5.1: CDriver members.

Due to the vaste typology of physical dynamics the software can face, SU2 strongly relies on the run-time polymorphism, activated during the instantiation of the driver. To enable the resolution of a new type of problem, three specific class will be specialized, namely CSolver, CVariable and CNumerics.
However, before describing the implementation for the thin film setting, we must take into consideration a preliminary step concerning the geometry. Indeed, due to the average nature of the method and since SU2 allows only bidimensional and threedimensional domains, we must make sure that the integration of the film equation, which is devoid from the normal direction, occurs correctly.

5.1.1 CGeometry

Since SU2 does not support monodimensional problems, implementing an average method is not straight forward. To bypass this limitation, the idea is to maintain in the mesh file still the higher dimension, but adding into the geometry class a fictitious volumetric element for the line and a new vertex boundary, different from the current CVertexMPI used for parallelization. Following this guideline, we added the possibility to read vertices as boundary units. Moreover, when dealing with 3D domains, the reading mechanism of the bidimensional case is activated by a boolean, checking whether we are solving the thin film multilayer equations, allowing the initialization of linear control volume in the frontier.
Going slightly into the details, choosing a finite volume approach, the CPhysicalGeometry class stores both the primal grid, read directly from the input file, and the dual grid, composed by the points where the unknowns are calculated, edges of the control volumes and vertex, which are boundary edges. This structure is clearly illustrated in Fig. 5.2. The new classes defined,

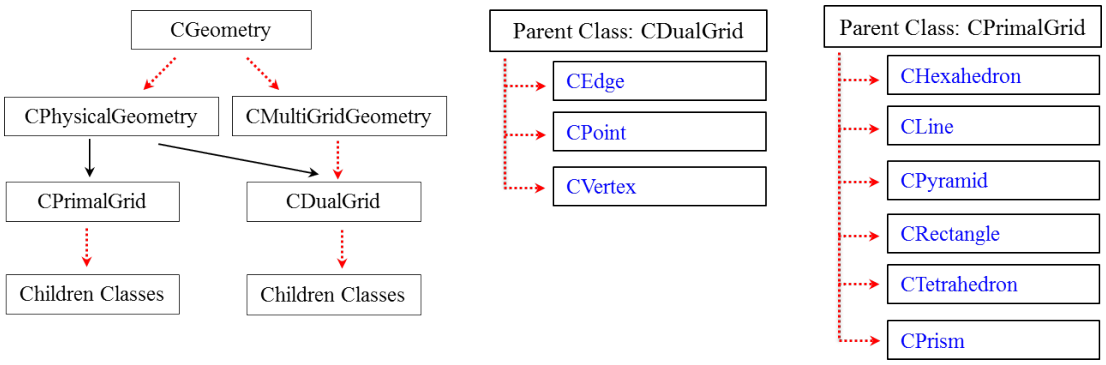

Figure 5.2: Original CGeometry structure.

children of CPrimalGrid, are CLineVol and CVeretxBound and their use is to be searched in the function which parses the connectivity, called for all the possible typology of element shape, i.e. DistributeVolumeConnectivity(config, geometry, LINE) and DistributeSurfaceConnectivity(config, geometry, VERTEX).

5.1.2 Mesh Extractor

Since the creation of a proper geometry requires some care, due to the integration of a domain deprived of one dimension, and, most importantly, the film domain comes from a boundary of the volume occupied by the other fluid, the decison made is to extract, and write in a separate su2 file, the new geometry directly from the mesh of the surrounding zone. This process is performed by a new program called mesh_extractor, whose compilation happens together with the installation of SU2. Doing so, we ensure that the points where the unknowns are computed, are exactly the same, allowing a faster communication without interpolation at the interface.
First of all we briefly need to describe how a .su2 file is organized. The first line simply asserts the number of dimensions of the problem. Then, both the element and the points are archived with the same organization, i.e. it is initially stated their numerosity, immediately followed by their corresponding list. In the case of an element, the line starts with a vtk identifier (for example 3=line, 5=triangle, 13=prism), which defines its shape, followed by all the nodes that compose its vertices, while for the point we only have the list of coordinates. Both of the discussed components are accompanied by an number, indicating the global index, that ends the line. The last piece of information regards the frontier, again preceded by the number of

different marker, then by the name of the boundary and an elencation of its elements, equipped with the vtk identifier but without any ordering index.
To setup the extraction, we need to write a small configuration indicating the input and output file name, the tag of the marker which defines the free surface. The key idea is straightforward and consist in looping on the target boundary storing the elements and the points which compose them, for a successive renumeration while using the old indices to choose the correct corrispective point. An additional complexity has to be faced when writing the boundary of the newly retrieved mesh, obviously not in the monodimensional case, where we simply take the maximum and the minimum x-coordinate. When extracting a surface from a 3D domain, the decision made is to keep as boundary the intersection of the interface with the other markers already present in the volume. For this reason, the CMesh_Extractor member function set_bc_2D creates an entire threedimensional geometry and, looping on its frontier, chooses and stores the point common to the interface and inherits the same name for the new boundary. As a consequence of the CGeometry object instantiation, in the configuration file all the marker type must be specified either with their real condition or with a placeholder one, while for a bidimensional input mesh we need to state only the interface tag. For more informations, examples and tests are stocked up in the subdirectory SU2/Mesh_Extractor/TestExtractor. Here it is reported the first piece of the code executing the process just explained.

```
 CConfig *config = new CConfig(config_file_name, SU2_MSH, false);
 CGeometry *geometry;
 CGeometry *geometry_aux = NULL;
 /*--- Definition of the geometry class to store the primal grid in the
       partitioning process. ---*/
 geometry_aux = new CPhysicalGeometry(config, ZONE_0, 1);
 geometry = new CPhysicalGeometry(geometry_aux, config);
...
 unsigned long  iPoint, jPoint, aux_Point, iElem;
 size_t *VecSize;
 unsigned short iMarker, nMarker, nCommonMarker, interface_Marker = -1;
 unsigned short iNode, iNode_Conn;
 bool *isCommonMarker;
 std::string vtk_line = "3";
 std::vector<std::string> marker_names;
 std::map<unsigned short,std::vector<unsigned long>> Marker_common_points;
 std::vector<std::pair<unsigned long, unsigned long>> *boundary_lines;

 nMarker = config->GetnMarker_All();
 nCommonMarker = 1; // surely FLUID_INTERFACE is common

...

 marker_names.resize(nMarker);
 isCommonMarker = new bool[nMarker];
 VecSize = new size_t[nMarker];
 for(iMarker = 0; iMarker < nMarker; iMarker++){
  marker_names[iMarker] = config->GetMarker_All_TagBound(iMarker);
  isCommonMarker[iMarker] = false;
 }

 /*--- Loop on all the boundaries ---*/
 for(iMarker = 0; iMarker < nMarker; iMarker++) {
  if( iMarker != interface_Marker ){
  /*--- Loop on the marker's elements ---*/
   for(iElem = 0; iElem < geometry->nElem_Bound[iMarker]; iElem++){
    /*--- Loop on the nodes composing each boundary element ---*/
    for(iNode = 0; iNode < geometry->bound[iMarker][iElem]->GetnNodes(); iNode
    ++){
     iPoint = geometry->bound[iMarker][iElem]->GetNode(iNode);
     /*--- Points reindexing through mapping ---*/
     if(old_to_new_idx.find(iPoint) != old_to_new_idx.end() ){
```

```
      if(map_idx_point.find(old_to_new_idx.at(iPoint)) != map_idx_point.end())
       Marker_common_points[iMarker].push_back(iPoint);
     }
    }
   }
   if(!Marker_common_points[iMarker].empty()){
    nCommonMarker++;
    isCommonMarker[iMarker] = true;
   }
  } // end iMarker != interface_Marker
} // end for iMarker
```

Afterwards, we need to write in the new mesh file for the film the boundary of the surface, which will unequivocally be lines. Hence, after we remove duplicate common points, the line elements are inserted with the tag they had in the complete domain, but with the new enumeration of the points. The conclusive part of the code is here detailed.

```
boundary_lines = new std::vector<std::pair<unsigned long, unsigned long>>[
    nMarker];

for(iMarker = 0; iMarker < nMarker; iMarker++) {
 if(iMarker != interface_Marker){
  for(size_t ii_vec = 0; ii_vec < VecSize[iMarker]; ii_vec++){
   iPoint = Marker_common_points[iMarker][ii_vec];
   for(size_t jj_vec = ii_vec; jj_vec < VecSize[iMarker]; jj_vec++) { //
   forward check to avoid duplicate couples
    jPoint = Marker_common_points[iMarker][jj_vec];
    if(iPoint != jPoint){
     for(iElem = 0; iElem < geometry->nElem_Bound[iMarker]; iElem++){
      for(iNode = 0; iNode < geometry->bound[iMarker][iElem]->GetnNodes();
   iNode++){
       if(iPoint == geometry->bound[iMarker][iElem]->GetNode(iNode) ){
        /*--- Loop on all the nodes connected to iNode ---*/
        for(iNode_Conn = 0; iNode_Conn < geometry->bound[iMarker][iElem]->
   GetnNeighbor_Nodes(iNode); iNode_Conn++){
         aux_Point = geometry->bound[iMarker][iElem]->GetNeighbor_Nodes(iNode,
   iNode_Conn);
         if(jPoint == geometry->bound[iMarker][iElem]->GetNode(aux_Point) )
          boundary_lines[iMarker].push_back(std::pair<unsigned long,unsigned
   long>(iPoint, jPoint));
        }
       }
      }
     } // end iElem
    } // end iPoint != jPoint
   } // end for jj_vec
  } // end for ii_vec
 } // end if iMarker != interface_marker
} // end for iMarker

/*--- Opening output mesh file ---*/
std::ofstream surface_mesh;
surface_mesh.open(out_mesh_name, std::ios_base::out | std::ios_base::app);
surface_mesh << "NMARK= " << nCommonMarker << endl;
surface_mesh.close();
write_fluid_interface();
surface_mesh.open(out_mesh_name, std::ios_base::out | std::ios_base::app);

/*--- Writing marker's name and nr. of elements---*/
for(iMarker = 0; iMarker < nMarker; iMarker++) {
 if(isCommonMarker[iMarker]){
  surface_mesh << "MARKER_TAG= " << marker_names[iMarker] << endl;
```

```
40      surface_mesh << "MARKER_ELEMS= " << boundary_lines[iMarker].size() << endl;
41      for(auto it = boundary_lines[iMarker].begin(); it != boundary_lines[iMarker].
        end(); it++){
42        /*--- Writing all elements and their shape (vtk identifier) ---*/
43        surface_mesh << vtk_line << " " << old_to_new_idx.at(it->first)<< " "<<
        old_to_new_idx.at(it->second) <<endl;
44      }
45    } // end if isCommonMarker
46  } // end for iMarker
```

5.2 Problem-specific Classes

As mentioned before, in the driver are polymorphically initialized the classes needed to define a problem, such as

- CSolver: that builds and solves the residual system,
- CVariable: which is a simple container of the most relevant physical variables,
- CNumerics: where the specific numerics method is implemented.

Each variable of the listed class types is stored in an array, because the usage of this containers allows to enlarge the case histories of resolutive methods, as will be described in the following example. Taking as exemplification variable solver_container, it is declared as a quintuple pointer to the base class CSolver and successively allocated throught the ripartion nZone, nInts, Multigrid_Level, MAX_SOLS. The first 3 position are common to many of the aforementioned objects because they concern generic extensions, which respectively are the number of zones in which the domain is divided, each with his own mesh and which may represent different fluids, like in the thin film case, the number of instances of the problem and the level on which the grid must be refined. The fourth array spot is proper of the solver and it indicates whether the contribution is to the laminar or turbulent flow solution or also its adjoint counterpart. The last pointer is obviously necessary to activate polymorphism.
All the previously explained constitutents are activated during the various CDriver preprocessing members, such as Geometrical_Preprocessing, Solver_Preprocessing and Numerics_Preprocessing, in which we added the specialization for the film problem. Therefore, in the following subsections, the final blocks of this tree of pointers will be presented, highlighting their principal changes specific to the thin film framework.

5.2.1 CFilmSolver

When the solver object is instantiated, the first instruction run by the constructor are the initialization of variables regarding the geometry and the physics of the film and, most importantly for this work, the activation of the boolean variable multilayer, which states whether the solver takes into consideration coherently the number of layers, by default set to 1, or adjust the amount of layer to a default of 3. In conclusion, it defines the number of averaged dimension nADim and the number of primitive variables nPrimVar, which differs from nVar, which considers only the conserved quantities.

```
/*--- Define geometry constants in the solver structure ---*/
nDim  = geometry->GetnDim();
nADim = nDim  - 1;
nMarker       = config->GetnMarker_All();
nPoint        = geometry->GetnPoint();
nPointDomain  = geometry->GetnPointDomain();
nEdge         = geometry->GetnEdge();
nLayer        = config->GetnLayer();

/*-- Unknowns (p, h, q_x, q_y) -- h=rho*delta -- q_x=u*h -- q_y=v*h --*/
nVar  = nADim + 2;
/*--- Primitive variables (p, vx, vy, rho, delta) ---*/
nPrimVar      = nADim+3;
nPrimVarGrad  = nADim;

/*--- Enabling multilayer and fallback if wrong input ---*/
if(nLayer == 0)
 SU2_MPI::Error(string("Invalid value for NLAYER, it must be at least 1."),
    CURRENT_FUNCTION);
if( (config->GetKind_Film_Solver() == MULTI_LAYER_ASYMP) && (nLayer > 1) )
 multilayer = true;
else if((config->GetKind_Film_Solver() == MULTI_LAYER_ASYMP) && (nLayer == 1)){
 multilayer = true;
 nLayer = 3;
 config->SetnLayer(nLayer);
}

...

/*--- Initialization of the system of residuals ---*/
unsigned long nlayernpoint    = nLayer*nPoint;
unsigned long nlayernpointdom = nLayer*nPointDomain;
Layer_SysRes.Initialize(nlayernpoint, nlayernpointdom, nVar, 0.0);

...

/*--- Get physical quantities ---*/
su2double *Density_Inf_Layer    = NULL;
su2double **Thickness_Inf_Layer = NULL;
if(multilayer) {
 Density_Inf_Layer   = new su2double [nLayer];
 Thickness_Inf_Layer = new su2double*[nLayer];
 for(iLayer = 0; iLayer < nLayer; iLayer++) {
 Density_Inf_Layer[iLayer]   = Density_Inf;
 Thickness_Inf_Layer[iLayer] = new su2double[nPoint];
 for(iPoint = 0; iPoint < nPoint; iPoint++)
  Thickness_Inf_Layer[iLayer][iPoint] = total_thickness[iLayer][iPoint]/
    su2double(nLayer);
 }
 /*--- Initialization of CVariable inherited class ---*/
 nodes = new CMultilayerFilmVariable(Density_Inf_Layer,Thickness_Inf_Layer,
    Velocity_Inf,layer_toll,nPoint,nDim,nVar,config);
} else {
 nodes = new CFilmVariable(Density_Inf,Thickness_Inf,Velocity_Inf,layer_toll,
    nPoint,nDim,nVar,config,1);
}
```

Finally, it is also initialized the variable nodes, which contains the physical informations for each point, coinciding in this case with the mesh node because the dual grid technique is applied. The other imperative subroutines are the ones which calculate the residual, divided specifically for the convective, viscous and source part. We report the typical loop present in Upwind_Residual function, where, for each control volume edge of the dual grid, we compute the

flux calling the members of CNumerics and the value is thereafter added and subtracted to the two points connected by that edge, corresponding to the flux entering and exiting the two elements interface. It is worth noticing how all the obtained residual are stored in the same container variable Layer_SysRes.

```
/*--- Loop on all the dual grid edges ---*/
for (iEdge = 0; iEdge < geometry->GetnEdge(); iEdge++) {
 /*--- Loop on all layers ---*/
 for(iLayer = 0; iLayer < (nLayer-1); iLayer++){

  iPoint = layer_edge[iLayer][iEdge]->GetNode(0);
  jPoint = layer_edge[iLayer][iEdge]->GetNode(1);

  /*--- Setting geometrical quantities to CNumerics variable  ---*/
  numerics->SetCoord(layer_node[iLayer][iPoint]->GetCoord(), layer_node[iLayer][
    jPoint]->GetCoord());
  numerics->SetNormal(layer_edge[iLayer][iEdge]->GetNormal());

  /*--- Grid movement ---*/
  if (dynamic_grid)
   numerics->SetGridVel(layer_node[iLayer][iPoint]->GetGridVel(), layer_node[
    iLayer][jPoint]->GetGridVel());

  /*--- Getting primitive variables from CVariable  ---*/
  V_i_Layer[iLayer] = nodes->GetLayerPrimitive(iPoint,iLayer);
  V_j_Layer[iLayer] = nodes->GetLayerPrimitive(jPoint,iLayer);

  numerics->SetPrimitive(V_i_Layer[iLayer], V_j_Layer[iLayer]);

  /*--- Computing and storing convective residuals ---*/
  numerics->ComputeResidual(Res_Conv_Layer[iLayer],config);
  Layer_SysRes.AddBlock(iPoint + iLayer*nPoint, Res_Conv_Layer[iLayer]);
  Layer_SysRes.SubtractBlock(jPoint + iLayer*nPoint, Res_Conv_Layer[iLayer]);
 }
}
```

The profile reconstruction executed by ComputeProfile, called in Source_Residual, is done through an auxiliary class, named CBlasStructure, which solves general linear systems for the polynomial coefficients exploiting, if the option is activated at compile time, blas and lapack routines, as shown in the following lines of code. The source uses two numerical objects, because the second takes care exclusively of the viscosity interaction between layers, which is the one with most available treatment, to which this work contributes defining a new possible recipe. The first numeric deals with all the terms related to the thickness, as long as, if present, with the volumetric force.

```
/*--- Loop on primal grid nodes ---*/
 for(iPoint = 0; iPoint < nPointDomain; iPoint++){

 /*--- Setting Thickness derivative ---*/
 ComputeDThickness(geometry, dThick_dx, iPoint);
 /*--- Compute the coefficient of the profile reconstruction ---*/
 for(iDim = 0; iDim < nADim; iDim++)
  ComputeProfile(config, V_Coeff[iDim], iPoint, iDim+1);

 /*--- Loop on all layers ---*/
 for(iLayer = 0; iLayer < nLayer; iLayer++){

 /*--- Setting primitive variables to CNumerics ---*/
  numerics->SetPrimitive(nodes->GetLayerPrimitive(iPoint, iLayer),
                         nodes->GetLayerPrimitive(iPoint, iLayer));
```

```
 second_numerics->SetPrimitive(nodes->GetLayerPrimitive(iPoint, iLayer),
                               nodes->GetLayerPrimitive(iPoint, iLayer));

...

/*--- Setting thickness derivatives ---*/
 for(iDim = 0; iDim < nADim; iDim++){
  dThickNmK_dx = (dThick_dx[nLayer-1][iDim]-dThick_dx[iLayer][iDim]);
  if(iLayer == 0) {
   numerics->SetdThick(0.0, dThickNmK_dx, iDim);
  } else if(iLayer != 0 && iLayer < (nLayer-1)) {
   numerics->SetdThick(dThick_dx[iLayer-1][iDim], dThickNmK_dx, iDim);
  } else {
   numerics->SetdThick(dThick_dx[iLayer-1][iDim], 0.0, iDim);
  }
 } // end iDim

/*--- Compute thickness source residual ---*/
 numerics->ComputeResidual(Res_Sour, config);
/*--- Add the source residual to the system ---*/
 Layer_SysRes.AddBlock(iPoint + iLayer*nPoint, Res_Sour);

/*--- Set extradiagonal stress profiles ---*/
if(iLayer == 0){
 second_numerics->SetStressBC(V_Coeff, 0.0, total_thickness[iLayer][iPoint]);
 } else {
  second_numerics->SetStressBC(V_Coeff, total_thickness[iLayer-1][iPoint],
   total_thickness[iLayer][iPoint]);
 }

 /*--- Reintialization of the residual to zero ---*/
 for(iVar = 0; iVar < nVar; iVar++) Res_Sour[iVar] = 0.0;

/*--- Compute viscous source residual ---*/
 second_numerics->ComputeResidual(Res_Sour, config);
/*--- Add the source residual to the system ---*/
 Layer_SysRes.AddBlock(iPoint + iLayer*nPoint, Res_Sour);
```

A last consideration on the source contribution is that, unlike the convection, the loop is performed directly on the primary mesh nodes, which are the points in which the solution variable are stored and consequently added to the residual system. Therefore, any computation which requires the calculus of a quantity only in one point are perfomed by other subroutines, such as ComputeDThickness and ComputeProfile. The former member function computes the derivative of the thickness exploiting a finite difference method for a bidimensional geometry or a more accurate central difference for the monodimensional case, applicable because the points are connected consequently and it is not necessary a connection matrix. The latter routine starts assembling the matrix of integrated Hermite coefficients, utilizing Cavalieri-Simpson technique, and retrieves the primitive mean values from the container CVariable, or one of its children.
Finally, for time integration, we just want to remark that only explicit methods are implemented in the solver class, which means explicit Euler, classical 4-step Runge-Kutta and generic Runge-Kutta, that must be accompanied by the option RK_ALPHA_COEFF in the configuration file to indicate all multi-steps coefficients.

5.2.2 CBlasStructure

Since profile reconstruction is performed for every point and for each single iteration, an efficient routine to solve the system which finds the Hermite polynomial coefficients. In SU2 it already exists a class named CBlasStructure deputed to linear algebra operations, but unfortunately

member functions for the solution of linear systems are not implemented. To be fair, SU2 provides a way of constructing matrices, which, howerver, is strongly connected to the whole triangulation grid and consequently cannot be exploited for our purposes.
Therefore, the problem is finding an efficient strategy to solve every possible linear system, or at least giving a reasonable approximation. The members added to CBlasStructure work in this direction, indeed, after a preliminary call to dgetrs(), the matrix is first of all factorized through LU with permutation technique, which does not require particular hypothesis. Thanks to this step, we can verify that the system is not singular, performing a check on the diagonal values of U, which coincides with the eigenvalues of A. If the singularity condition is not met, we can take advantage of the already computed factorization and solve the lower and upper triangular systems with respectively forward and backward substiution. In the opposite case, i.e. one or more eigenvalues are equal to zero, we must rely on a different and always feasible techniques, such as Least Squares. The minimization of $\|\mathcal{H}x - b\|$ is done, as before, through a first step of factorization and the subsequent resolution of two linear systems. The QR factorization is therefore implemented, relying on the Grant-Schmidt algorithm to compute the orthogonal matrix.
The described techniques are achieved both leaning on the external open linear algebra library openblas, which SU2 already includes in its compilation options, and manually, if the aforementioned alternative is not avaible. The code is able to switch between those options thanks to the $\#if$ ... $\#endif$ conditional. The references to all the library functions can be found at the official site [c].

5.2.3 CFilmVariable

As previously said, the general CVariable is nothing more than a container for the physical informations of each node, therefore it contains mostly setter and getter methods for the primitive variables and for the conserved quantities.
Since in the multilayer case the number of layers can vary and it is useful, due to the possible influence of specific interaction term, to know which are the bottom or top stratum, the decision adopted is to create the child class storing the necessary members for a generic layer, then derive from it in the class CMultilayerFilmVariable and add an array of CFilmVariable for all the inferior layers, while the surface physical properties are saved in the main class members. The diagram, presented in Fig. 5.3, shows this organization, as well as some important member functions. In the constructor, we define a container for the primitive physical quantities of the fluid and one for the conserved quantities in the variable Solution, which are successively returned by the function GetLayerPrimitive.

Because the nature of the equation becomes singular when the thickness of a layer degenerates to zero, we must ensure that, while setting the primitive variable, it does not decrease too much, which means that δ_{k-1k} doesn't go below the 1% of the initial value. This tolerance threshold is chosen in the solver constructor and it is taken as the smallest value between all mesh points. Whenever the singularity condition is verified, the conserved velocity flux is not divided by the layer thickness, but the speed is imposed directly as the value of the underlying layer or with the bottom boundary conditions, if the thickness that is becoming zero is the one corresponding to the stratum most near to the backdrop of the domain.
Another possibility, may be the quadratic reconstruction, presented by Kurganov [11] at pag. 313, in the form for the horizontal velocity in a generic layer:

$$u = \frac{2qh}{h^2 + max(h^2, toll^2)}, \tag{5.1}$$

where *toll* is again the threshold set in CFilmSolver. If this technique is preferred, on should change the implementation in the member function CFilmVariable::ReconstructVelocity.

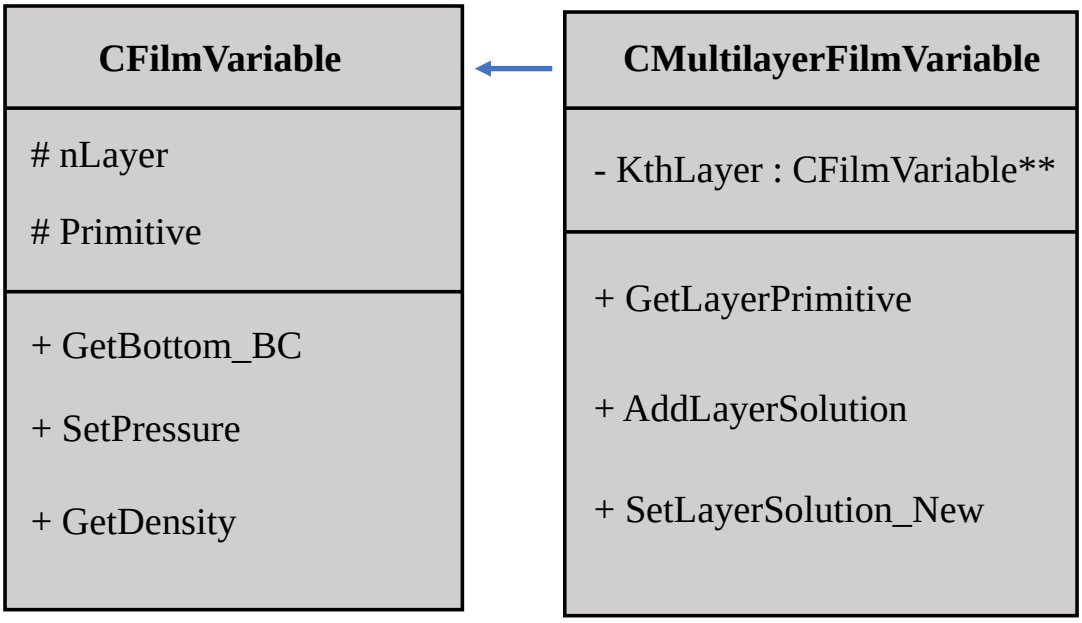

Figure 5.3: CFilmVariable inheritance diagram.

5.2.4 CFilmNumerics

The last piece needed to tie up the integration of a new solving method, is a class which effectively computes the residuals between two arbitrary points connected by an edge, by a vertex, in the case of boundary conditions, or in a single point, for source contribution. For every problem, more than one child must be defined to implement ad hoc techniques for the specific contribution, which are again stored in the driver array numeric_container. The classes inheriting from CNumerics work essentially in two steps: at the generic iteration in the loop on the edges, geometrical and physical quantities of the two connected nodes are set through the functions SetNormal, SetCoord and SetPrimVar, followed by ComputeResiduals, invoked to perform the computation of the respective residual.

Therefore, we specialized the base class in order to develop each separate numerical techniques, as presented in Fig. 5.4 and described afterwards.

The children class which handles the convective part is CCentUpw_SW, implementing the central upwind scheme for shallow water problems introduced by Kurganov in [11] and discussed in Chapter 4. To ensure that an upwind approach is used, one should set the option CONV_NUM_METHOD_FLOW equal to ROE. Its most interesting member variables are the functor SW_Flux, that contains the expression of the real flux vector, e.g. $(q_x,\ q_x^2 + \frac{1}{2}\frac{1}{\rho}gh^2,\ q_xq_y/h)^T$ for x component, deduced in the derivation of the model, and all the ones corresponding to the contributes to the numerical flux, such as δU, U_card, i.e. U_E, U_W, U_S, U_N, U^* and the collection of advection coefficients stored in adv_coeff. Hence, as previously explained, the key routine ComputeResidual executes the following instructions:

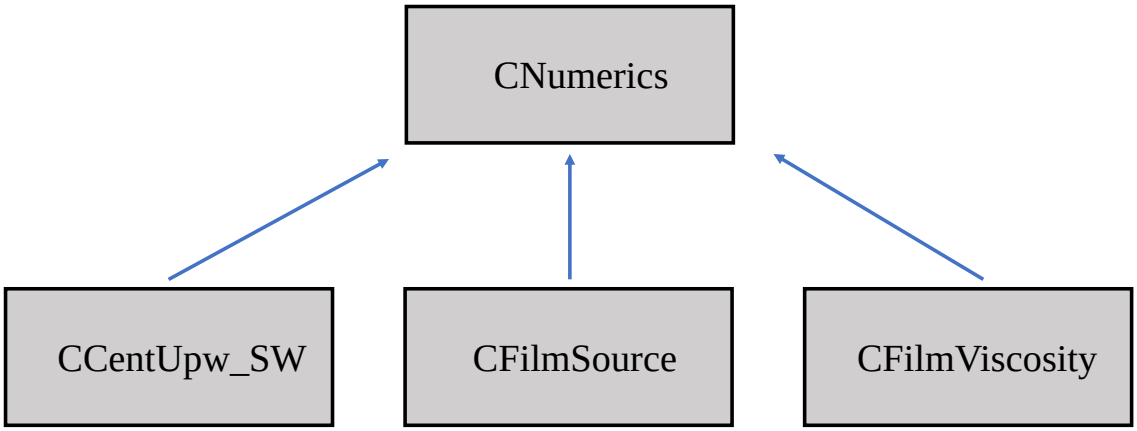

Figure 5.4: CNumerics derived classes.

```
/*--- Primitive variables at point i ---*/
 Pressure_i   = V_i[0];
 for (iDim = 0; iDim < nADim; iDim++)
  Velocity_i[iDim] = V_i[iDim+1];
 Density_i    = V_i[nADim+1];
 DeltaThick_i = V_i[nADim+2];

/*--- Primitive variables at point j ---*/
 Pressure_j   = V_j[0];
 for (iDim = 0; iDim < nADim; iDim++)
  Velocity_j[iDim] = V_j[iDim+1];
 Density_j    = V_j[nADim+1];
 DeltaThick_j = V_j[nADim+2];

 real_flux.SetDensity(Density_i);

/*--- Setting conservative variables ---*/
 Conservatives_i[0] = Density_i*DeltaThick_i;
 Conservatives_j[0] = Density_j*DeltaThick_j;

 for (iDim = 0; iDim < nADim; iDim++) {
  Conservatives_i[iDim+1] = Density_i*DeltaThick_i*Velocity_i[iDim];
  Conservatives_j[iDim+1] = Density_j*DeltaThick_j*Velocity_j[iDim];
 }

/*--- Setting cardinals conservative variables, order important ---*/
 SetCardinals();
 Compute_adv_coeff(U_card);
 Compute_U_star();
 Compute_delta_U();

/*--- Computing numerical flux ---*/
 Compute_numerical_flux();

 for(iVar = 0; iVar < nVar; iVar++)
  val_residual[iVar] = 0.0;

```

```
/*--- Continuity & Momentum  eq. contribution ---*/
 for(iVar = 1; iVar < nVar; iVar++){
  for(iDim = 0; iDim < nADim; iDim++){
   val_residual[iVar] += numerical_flux[iVar-1][iDim]*UnitNormal[iDim];
  }
 }
```

For completeness, we report also the calculation of numerical_flux

```
/*--- Compute numerical fluxes ---*/
for(iEqs = 0; iEqs < nEqs; iEqs++){
 for(iDim = 0; iDim < nADim; iDim++){
  adv_diff = adv_coeff[iDim] - adv_coeff[iDim + nADim];
  adv_prod = adv_coeff[iDim] * adv_coeff[iDim + nADim];
  numerical_flux[iEqs][iDim]= (adv_coeff[iDim]*real_flux(U_card[iDim],iEqs,iDim)
     - adv_coeff[iDim+nADim]*real_flux(U_card[iDim+nADim],iEqs,iDim) );
 /*--- Anti-diffusive term ---*/
  numerical_flux[iEqs][iDim] += adv_prod*( U_card[iDim+nADim][iEqs]
                                - U_card[iDim][iEqs] - delta_U[iEqs][iDim] );
  if(adv_diff != 0.0){
   numerical_flux[iEqs][iDim] /= adv_diff;
  } else {
  numerical_flux[iEqs][iDim] = 0.0;
  }
 }
}
```

which retrace the formula (4.25)

$$\mathcal{F}_{j+\frac{1}{2},i} = \frac{a^+_{j+\frac{1}{2},i} f(U^E_{j,i}) - a^-_{j+\frac{1}{2},i} f(U^W_{j+1,i})}{a^+_{j+\frac{1}{2},i} - a^-_{j+\frac{1}{2},i}} + \frac{a^+_{j+\frac{1}{2},i} a^-_{j+\frac{1}{2},i}}{a^+_{j+\frac{1}{2},i} - a^-_{j+\frac{1}{2},i}} [U^W_{j+1,i} - U^E_{j,i} - \delta U_{j+\frac{1}{2},i}].$$

We can note a couple of interesting details, such as the use of nEqs, which is equal to nVar - 1 because the first conserved variable is the pressure, retrieved directly from the PE or PEV^2 hypothesis, making unecessary a residual addition.
Regarding the contribution of interaction between layers, the classes CFilmSource and CFilmViscosity deal respectively with pure source terms and with the reconstruction of the stress profile. When called into the function Source_Residual, the numerics' variables retrieve the profile using the function SetStressBC, which takes in input the coefficients of the Hermite polynomial and the heigth of the two layers which are interacting and then it simply sums the evaluation of the polynomial weighted with the coefficient, paying attention for to use the Hermite property $H'_n(x) = 2nH_{n-1}(x)$. Finally, the residual is simply the difference between the quantities evaluated exactly at adjacent layers. In addition to the thickness source discretization, CFilmSource verify if any volumetric force is acting and consequently computes its effect.
For demonstrative purpouses, we report the essential codelines of CFilmSource::ComputeResidual

```
/*--- Source contribution to the PE hypothesis ---*/
 val_residual[0] = 0.0;

/*--- Source contribution to the continuity equation ---*/
 val_residual[1] = 0.0;

/*--- Source contribution to the momentum equations ---*/
/*--- NB: opposite signes because source brought to the LHS ---*/
 for(iDim = 0; iDim < nADim; iDim++){
  val_residual[iDim+2]  = 0.0:
  val_residual[iDim+2] += Volume*Density_i*g_const*Thickness_i*dThick_Nmk[iDim];
  val_residual[iDim+2] += Volume*Density_i*g_const*Thickness_i*dThick_km1[iDim];
  if(gravity_force){
   val_residual[iDim+2] -= Volume*Density_i*Thickness_i*body_force_vector[iDim];
```

```
  }
 }
```

and for its viscous counterpart CFilmViscosity::ComputeResidual

```
/*--- Source contribution to the PE hypothesis ---*/
 val_residual[0] = 0.0;
 /*--- Source contribution to the continuity equation ---*/
 val_residual[1] = 0.0;

/*--- Source contribution to the momentum equations ---*/
 for(iDim = 0; iDim < nADim; iDim++){
  val_residual[iDim+2] = -Volume*(0.5*Laminar_Viscosity_i*BC_Stress_k[iDim] -
    0.5*Laminar_Viscosity_i*BC_Stress_km1[iDim]);
 }
```

5.2.5 CFilmOutput

Normally, SU2 returns an output containing, beyond the physical variables required in the configuration file, the primary grid, complete with the typology of elements and their connectivity. This ordering is not entirely feaseble with the geometry defined by the multilayer framework, since we can say each layer owns a proper mesh, which is not connected with the adjacents. Therefore, the decision taken is to write a new kind of output file, which does not expect SU2 structure, but allows immediate readability and easy postprocess operations. Hence, the most natural choice is to organize each output variable as list of their value for all point of a single layer, then start a new line for the successive inferior layer.

5.3 SU2 Setup and Execution

The additional features for solving the thin film set of equation have been introduced in SU2 via the least invasive and most cooperanting way possible. Indeed, the installation procedure is kept equal to the standard one introduced for version 7.0 of the software, which has become faster with respect to the past thanks to the employment of the Meson build system. Meson is an open source build system meant to support the compilation process for several programming languages, including C++. To give a basic idea of its functioning, we report the most simple commands which allow the compilation of a program, which are

```
// A variable containing all source files
 src = files(['source1.cpp', 'source2.cpp'])
// A variable containing include directories
 inc_dir = include_directories('../path/to/inc')
// The instruction to create the executable main
 exc = executable('main', src,
                  install : true,
                  include_directories : inc_dir,
                  cpp_args:  ['-Wall'] + [cpp_args])
```

where cpp_args are the compiler arguments. The only requirement needed both for Meson and SU2 is having installed at least version 3.0 of Python, which, if connected with super-user privileges, will require the entry of the password.
Therefore the user should just type the following commands to achieve a full installation:

```
 ./meson.py build -Denable-option=true
 ./ninja -C build install
```

where Ninja is another build system with a focus on speed. Both Meson and Ninja are downloaded automatically alongside the set-up of SU2. The complete list of options can be consulted on the official website of SU2, but for the optimization of this work, the only relevant utility to activate is -Denable-openblas.
Once SU2 installation is completed, the paths SU2_HOME="/PersonalPath/SU2" and SU2_RUN="/PersonalPath/SU2/bin" must be exported to the environmental variables and consequently included SU2_RUN into the existing variable PATH.
Completed this step, the user can finally run any fluid dynamics simulation, in particular the ones regarding thin film, including the correct option in the configuration file. For what concerns this typology of problem, the add on proposed by this project can work either as a stand-alone, solving only the film dynamic, or as a multizone computation, able to connect for example a liquid sheared by a gas.
However, in both of the aforementioned cases, the second document needed, i.e. the mesh file, may require a preliminary step. Indeed, the geometry based on this input must follow two precautions, which can be taken care manually by the user or through the program mesh_extractor, created by the author of this work. The former provision is the addition of the boundary marker representing the free surface, which is exactly equal to the primary grid, while the latter is the congruence of that mesh with the bounday communicating with the world outside the film. Those two issues are automatically resolved by the call to mesh_extractor, which is used to extract form a pre existing su2 mesh, proper to the fluid paired with the film, the boundary on which the free surface is located. In order to perform this operation, we need a brief input file, which can again be saved with the extension .cfg, containing the name of the su2 grid file, the tag with which the interfaced is referred to and the denomination of the output mesh. This procedure grants a sure integration with the code extension for problem studied in this work.
Finally, one can launch the simulation with the command SU2_CFD accompanied by the configuration file with the desired options for the thin film, described at the start of the previous section.

6 Test Cases

The final purpose of this work is to obtain a code capable to extend SU2 in order to simulate real physical problem catalogable in the thin film setting. To validate both the mathematical model, the numerical discretization and its implementation in SU2, we rely on the test cases most widely used in the literature of thin films, but also in shallow water theory. We will treat situations where the solution reaches a steady state, a parabolic speed front and a particualr shape of the thickness. At last, it will be shown the behaviour of the free surface if a perturabtion affects the inlet and it will be studied its propagation and repercussions. Each case will be analized in a separate section, illustrating its principal specific characteristics and the essential instructions to reproduce the result using SU2 software.

6.1 Lake at Rest

The starting point of every analysis, which is a common benchmark to many typologies of problem, is to see if the numerical scheme reaches a steady state starting from a an initial condition where the fluid is still. As stated in chapter 4, the discretization of the convective flux satisfies the 'lake at rest' condition, therefore we set up a configuration file for a motionless thin stratum of liquid, resting on a bottom inclined by 30°. Since we are taking only a part of the fluid, we suppose that simmetry boundary condition at the side of the domain taken into consideration. Since the example is rather manageable, the method is applied considering only 3 layers.
The result, aside from some numerical inaccuracy at most proportional to 10^{-9}, matches the theoretical prediction, as it is shown in the following contour plot of the velocity.
After performing this simple, but necessary, check, more interesting cases are studied in the next sections.

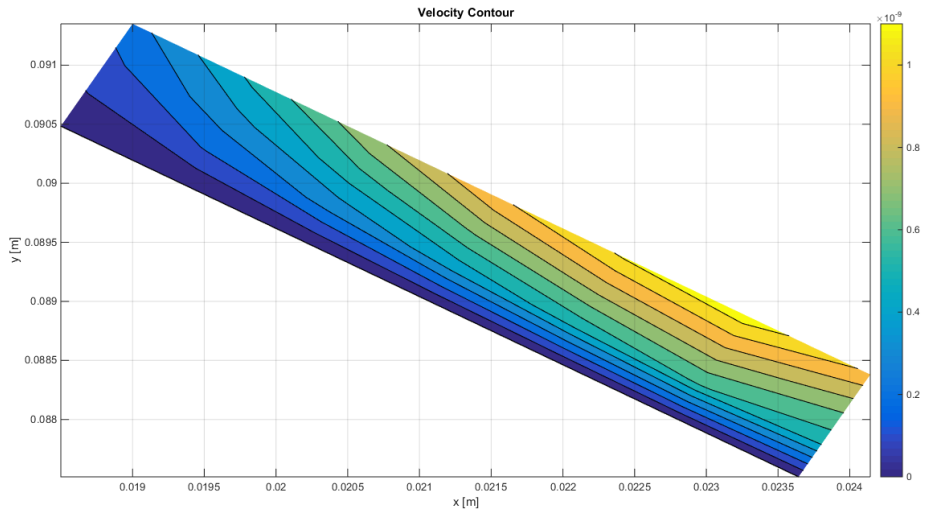

Figure 6.1: Velocity contour plot of the lake at rest solution.

6.2 Falling Film

Thin films flowing down inclined surface exhibits a rich phenomenology and offer the best testing ground for the study of this field. The geometry taken into consideration is mono-dimensional and the flat bottom is inclined with respect to the abscissa by an angle α included between $0°$ and $90°$, while, in the configuration file the input angle is β, defined as the anti clock-wise angle starting from the positive x axis, for which holds the relation $\alpha = \pi - \beta$.
A trivial solution to the flow equations is easily found in the form of a steady uniform parallel flow with parabolic velocity profile, often called Nusselt's solution, where the work done by gravity is exactly consumed by viscous dissipation. The explicit expression of the Nusselt profile is

$$u(y) = \frac{1}{2\mu}\rho g \sin(\alpha)(2\delta_N - y)y, \tag{6.1}$$

which is really useful to have for a direct comparison with our numerical result. Morever, also the analytically computed can be illustrated as

$$p(y) = p_0(x) + \rho g \cos(\alpha)(\delta_N - y), \tag{6.2}$$

where $p_0(x)$ represents the imposed pressure at the free surface.
For this specific test, the fluid simulated is water, imagined at a temperature of $25°C$, having the following physical properties: a constant density $\rho = 998\frac{kg}{m^3}$ and a dynamic viscosity equal to $\mu = 8.9 \times 10^{-4} Pa\ s$. From expression (6.1), considering a maximum film thickness $\delta_N = 1mm$, we can compute the typical velocity of order $u_\infty \approx 0.1\frac{m}{s}$, making therefore easy to retrieve Reynolds number, whose value is $Re = 112$.
The simulation is initially carried out dividing uniformly the water in 5 layers, with a parabolic inlet entering from the left side and a plane inclined by $30°$, until the computation reaches the maximum number of iterations or the final time of half a second, chosen to have the possibility to fully propagate the fluid and to arrive at a steady state solution. The choice of this final instant t_F can be justified by the next formula, which is regained by the double integration of

the differential equation $\frac{\partial^2 x}{\partial t^2} = g\sin(\alpha)$, and reads as

$$t_F = \frac{-u_\infty + \sqrt{u_\infty^2 - 2g\sin(\alpha)(L_0 - L_F)}}{g\sin(\alpha)}, \tag{6.3}$$

where $(L_0 - L_F)$ is the total length of the channel.
If someone would like to change the number of layers through the SU2 option NLAYER, also the inlet option should be substituted, because the last value of this option is the uniform thickness between two adjacient layers, which obviously decreases as the fliud is divided more densely. In addition, if the inlet profile is known, activating the option SPECIFIED_INLET= YES, an additional file should be given as input and must be organized in the following way: the first lines indicate the number of marker in which the inlet boundary condition is acting, their name tag and the number of point proper to that frontier, expressed in the option NROW. The next line, NCOL, should be equal to the formula nDim + nLayer×(nADim+2), because all the successive rows are organized in border coordinates, followed by the temperature, the direction of the velocity and its magnitude, repeated for the number of layers.
First of all, it is interesting to see how the thickness of each layer deacreases or increases along the domain and, as we can see in Fig 6.2, where the blue lines are the fluid strata, while the black one represents the bottom, it seems that their height remains constant. However, the total depth reduces very slightly form 1 mm to 0.965 mm, phenomenon due to the gravity accelaration, which increases the velocity and consequently diminish the thickness to respect the continuity equation.

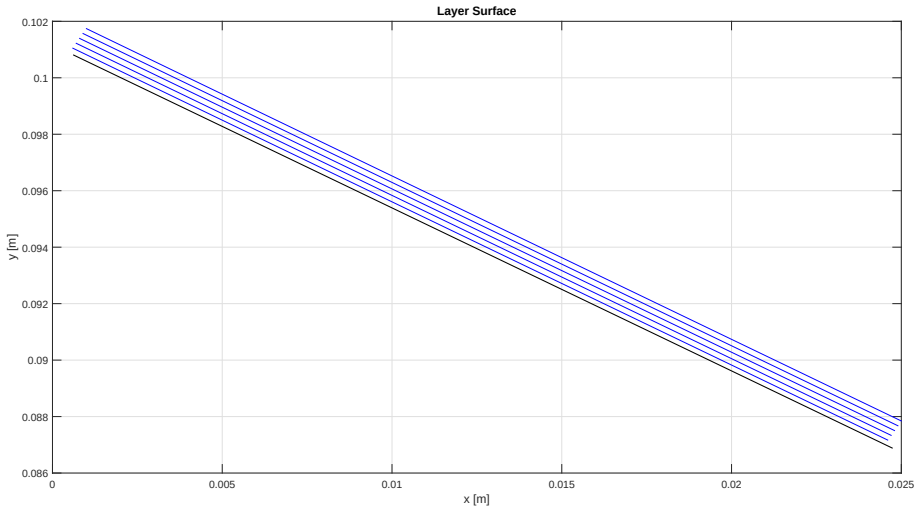

Figure 6.2: Falling film multilayer geometry.

The analysis continues comparing the profile of the horizontal velocity and of the pressure, respectively presented in Fig 6.3 and Fig 6.4, with the aforementioned solutions regained by Nusselt. The first picture is the photography of the shape at different distances from the inlet, to ensure that the stationary state is reached. Regarding the pressure instead, only the comparison with the exact solution is shown, since the result is not derived by the system resolution, but it is directly imposed by the hypothesis adopted, the hydrostatic one in this example.

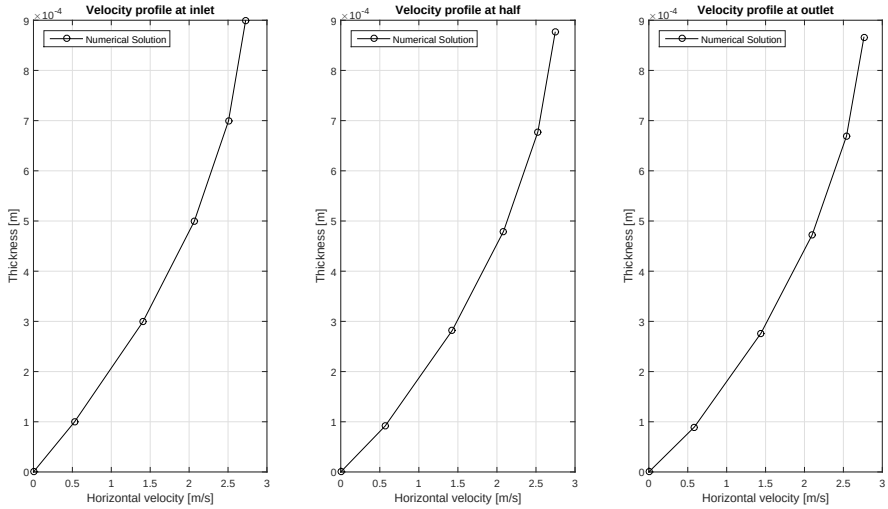

Figure 6.3: Horizontal velocity profile at different distances from the inlet.

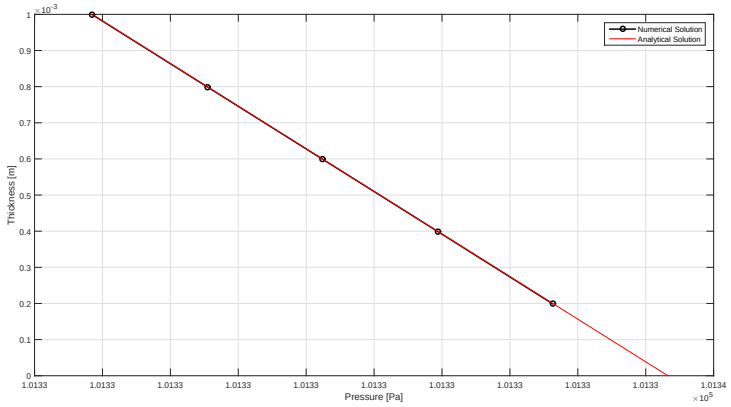

Figure 6.4: Pressure profile compared to Nusselt solution.

As stated before, for completeness purposes, the following image (Fig 6.5) verifies the mass flow conservation, displaying the product between the thickness and the velocity across the axis tangent to the bottom. The inferior layers have obviously lower values, because the fluid is

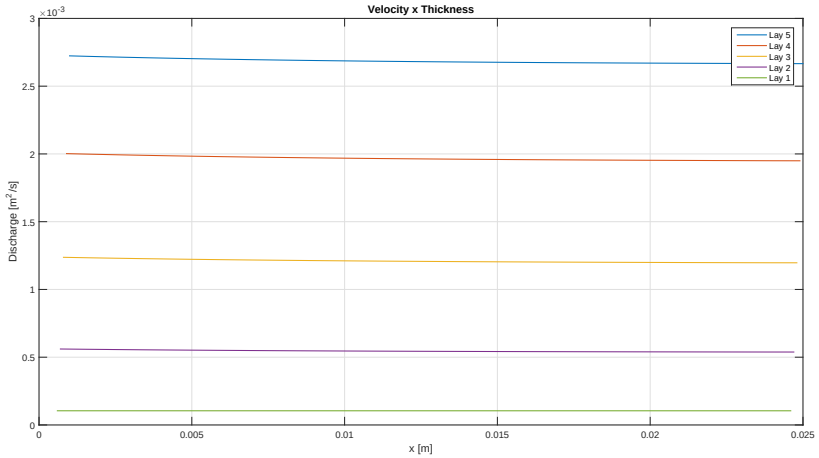

Figure 6.5: Velocity flow rate across the channel.

slower, while, in general, the flow rate slightly decreases, until a value equal to the 0.9174% of its maximum, which is neglectable.
An interesting question is to see how the solution improves increasing the number of layers and whether this addition, in the face of a computational load growth, achieves more accurate results. The deduced considerations are not of general character, because of the typology of problem and the solution doesn't have particualrities that necessite specific distribution of the thickness.
The decision adopted to test the quality of the results is the juxtaposition of the obtained values for the velocity with the known expression regained by Nusselt. The same confrontation done with the pressure would be useless, since fruit of an imposed formula attained by the assumption adopted. Therefore, the following graphic is the confirm that adding layers increase the accuracy of the result, but not eccesively, hence it may not always be worth demanding more computational resources to obtain similar results.
Although more precision is retrieved increasing the number of layers, it may not always be worth adopting this decision if the ultimate purpose is to obtain a fast result. Moreover, as proved by Fig. 6.7, which for each time step pictures the logarithm of the root-mean square error of the top layer velocity, the convergence to the exact solution may be faster in the initial iterations, but the it is evident that after an higher number of layers solws down the attainment of the steady state solution.
The phenomen of the slower convergence rate can have two possible explanations. It may be a drawback of the polynomial reconstruction, because a wrong velocity profile implies imprecise stresses, which may influence negatively the viscous contrast to the convection. Another plausible justification, is that, since we are using an explicit time discretization, increasing the number of layers, but mantainig the same temporal step $\triangle t$, we need more instants to reach convergence.
After discussing of the most relevant characteristics of this test case, important for the reconstruction of the velocity profile, we can pass to a new one, which instead focuses on the thickness computation.

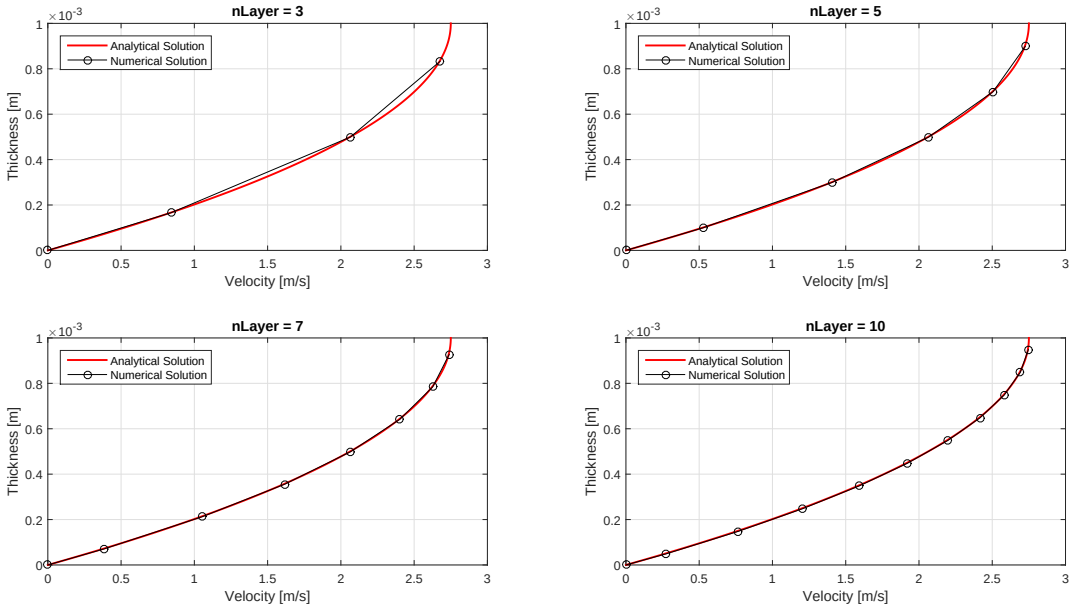

Figure 6.6: Comparison of the velocity profile with Nusselt's solution with different number of layers.

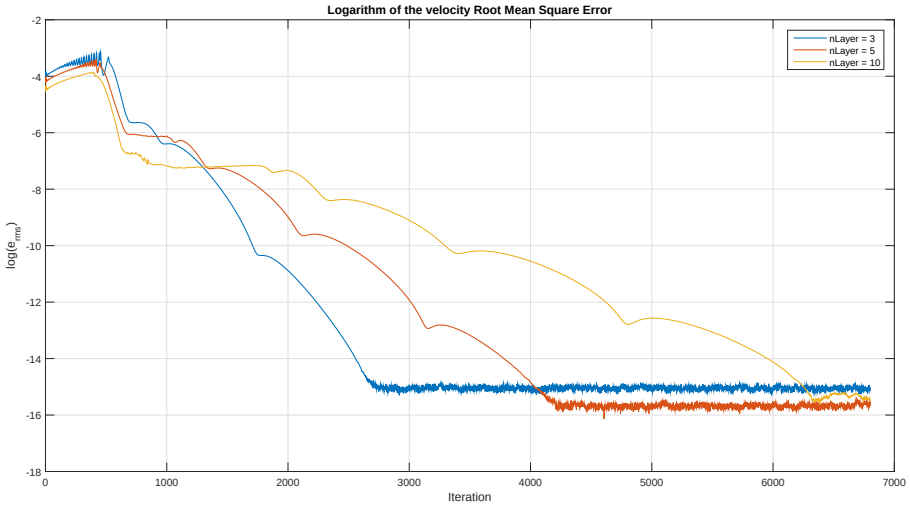

Figure 6.7: Comparison of the error convergence with different number of layers.

6.3 Dam Break

The analysis of dam break flow is part of dam design and safety analysis: dam breaks can release an enormous amount of water in a short time. This could be a threat to human life and to the infrastructures. To quantify the associated risk, a detailed description of the dam break flood wave is required. Research on dam break started more than a century ago, therefore plenty of studies have been discussed and are available as benchmarks. Usually, both analytical solutions and experimental data are presented in the literature. Hence, to test our code, we focused our attention on the wet bed dam break problem and relied on the exact expression exposed in [31] by Delestre *et al.* and the results obtained in laboratory by Wang in [32], which will be treated in the following sections.

6.3.1 Dam Break: Comparison with Inviscid Exact Solution

The analytical solution to the dam break problem was first introduced by Stroker [33] in 1957 and since then it has been object of numerous studies and papers. The hypothesis proposed is to idealize the problem, considering an instantaneous break of the dam, which pours out on a wet bed leaning on a flat bottom. Furthermore, friction effects are ignored. Although, there isn't a direct option to omit the viscous effect, it can be achieved selecting MU_CONST= 0.0 in the configuration file. Doing so, stresses are still computed, but their contribution is not added.
Therefore, before the start of the event, the height of the fluid follows the initial condition of the Riemann problem, i.e.

$$\delta(x) = \begin{cases} h_l & for\ 0\ m < x < x_0, \\ h_r & for\ x_0 < x < L, \end{cases} \tag{6.4}$$

where L is the length of the domain, x_0 is the position of the dam and $h_l \geq h_r$.
Starting from a steady state of the fluid, at time $t \geq 0$ we have a left-going rarefaction wave (or a part of parabola between $x_A(t)$ and $x_B(t)$) that reduces the initial depth h_l into h_m, half of the beginning left height, and a right-going shock, located in $x_C(t)$, that increases the initial height h_r into h_m. The evolution of the waves can be tracked because the points advance at a know expression in time, accordingly to the following formulas: $x_A(t) = x_0 - t\sqrt{gh_l}$, $x_B(t) = x_0 + t(2\sqrt{gh_l} - 3c_m)$, $x_C(t) = x_0 + t\frac{c_m^2(\sqrt{gh_l}-c_m)}{c_m^2 - gh_r}$, where c_m is the propagation speed defined as $\sqrt{gh_m}$. Given those coordinate one can state also the exact piece-wise formulation for the thickness, which is

$$\delta(t,x) = \begin{cases} h_l & for\ 0\ m < x < x_A(t), \\ \frac{4}{9g}(\sqrt{gh_l} - \frac{x-x_0}{2t})^2 & for\ x_A(t) < x < x_B(t), \\ \frac{c_m^2}{g} & for\ x_B(t) < x < x_C(t), \\ h_r & for\ x_C(t) < x < L. \end{cases} \tag{6.5}$$

In addition, the horizontal velocity can be regained explicitly in the form

$$u(t,x) = \begin{cases} 0\frac{m}{s} & for\ 0\ m < x < x_A(t), \\ \frac{2}{3}(\frac{x-x_0}{t} + \sqrt{gh_l}) & for\ x_A(t) < x < x_B(t), \\ 2(\sqrt{gh_l} - c_m) & for\ x_B(t) < x < x_C(t), \\ 0\frac{m}{s} & for\ x_C(t) < x < L. \end{cases} \tag{6.6}$$

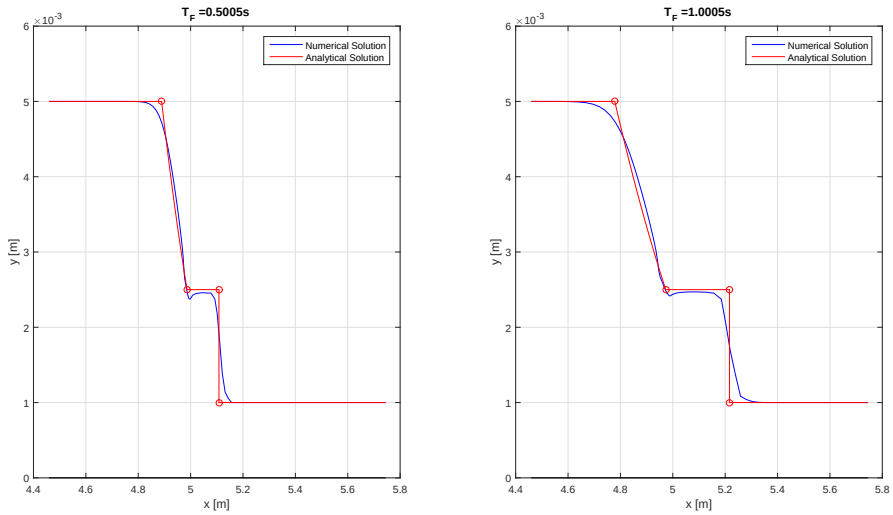

Figure 6.8: Comparison between analytical Stroke thickness solution and numerical result.

The aforementioned analytical solutions are tested against the numerical result obtained in SU2, first of all confronting the free surface profile evolution, as presented in Fig. 6.8.
An advantage of the multilayer technique adopted is that, beyond the surface communicating with the outside world, also the behaviour of the inferior strata can be observed, which differs from the shallow water integrated method, that focuses its attention on the water surface. In the following image, we see that there is no substantial difference, due to the fact that the viscous interaction between layers is neglected by hypothesis.

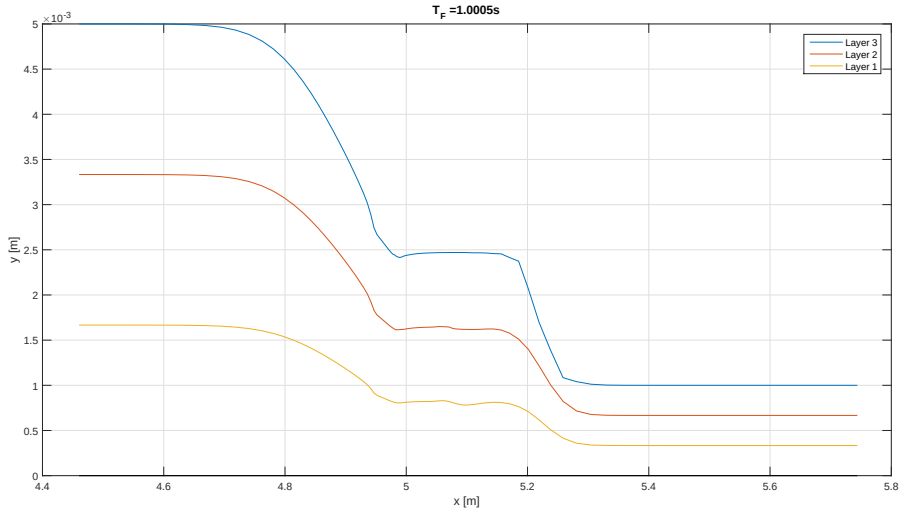

Figure 6.9: Layer thickness near the dam break.

To end the analysis, we check the similarities among the exact velocity and the numerically computed one, where unfortunately the values calculated by SU2 present some oscillatory instabilities.

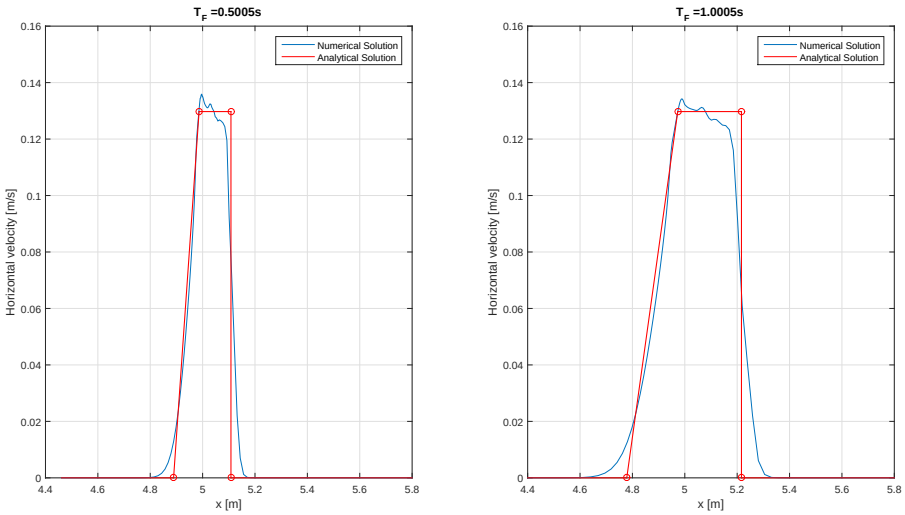

Figure 6.10: Comparison between analytical Stroke velocity solution and numerical result.

6.3.2 Dam Break: Comparison with Experimental Data

In its work [32] Wang *et al.* reproduced experimentally the evolution of a dam break, investigating primarly the similarities among different initial condition, for example the ratio between the water depth downstream h_d and upstream h_u with respect to the dam, which takes the name α. The setup consists of a rectangular cross section channel , which is divided into two parts, i.e. reservoir and downstream flooded area, by a 15 mm thick fiberglass, while the whole structure lies on a flat bottom. After lifting the gate, fluid height data are measured by eight CCD cameras arranged to capture the evolution of the flow. To retrieve the values from the paper, we used Engauge Digitizer, a tool which allows the user to obtain the coordinate of graphics pictured in documents. For this reason the experimental solution used as comparison are a bit rough, although their main purpose is the confront the shape and the evolution of the wave front.
Since the authors' objective is to analyze similarities of the water surface behaviour, the relevant quantities are adimensionalized taking as references the following values

$$T = \frac{t}{\sqrt{\frac{h_u}{g}}}, \quad H = \frac{h}{h_u}, \quad U = \frac{u}{\sqrt{g h_u}}. \tag{6.7}$$

The test case we executed, fixes the ratio of up and downstream height, called α, at 0.2. The first confrontation performed is obviously the one with the free surface of the fluid, which is pretty consistent, as one can observe in Fig. 6.11. In the downstream channel, the water levels at the measurement sections rise sharply due to the arrivals of dam-break shock waves, while upstream the fuild reduces its height with a simil parabolic shape.
Another correspondence may be found when looking at the velocity, which is null at the boundaries because the front wave has not reached them yet, and it increases in the middle, where the break occurs.

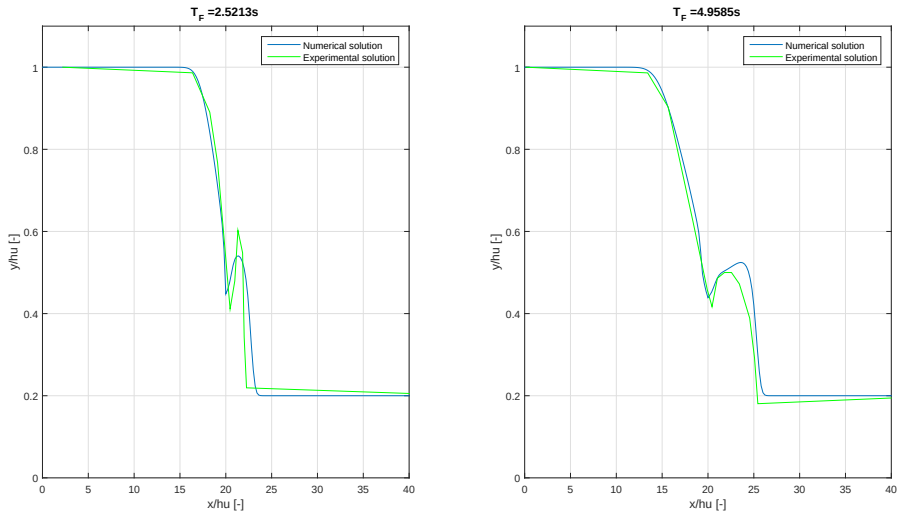

Figure 6.11: Comparison between experimental thickness and numerical result.

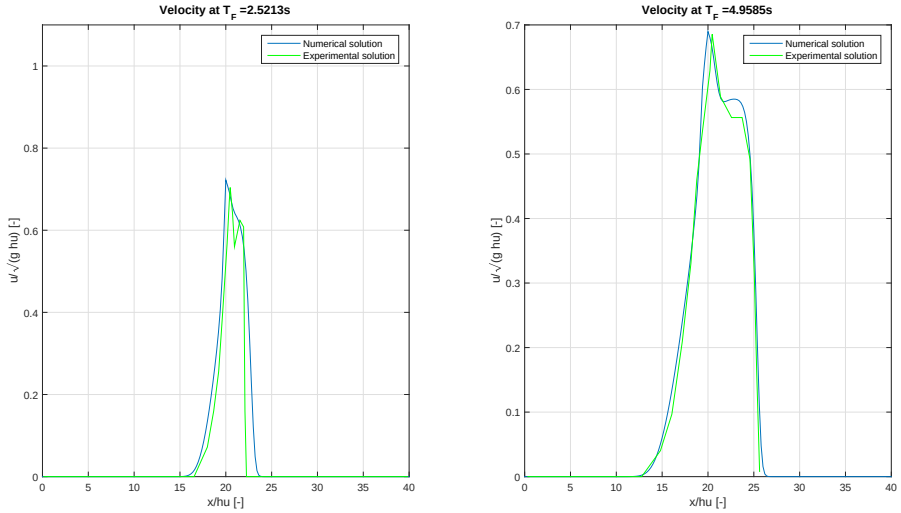

Figure 6.12: Comparison between experimental velocity and numerical result.

6.4 Laminar Roll-Waves on a Falling Film

In this last section, we examine the behaviour of the film if the inlet velocity is perturbed, which arise roll wave instabilities in the transitional laminar flow. This case models a more realistic situation, where, due to external factors, the boundary conditions are not always uniform or stable during time, therefore its importance is crucial.
Taking as a reference the work of Fiorot, Maciel *et al.* [34], we reproduced in a simulation their experimental setup, which will be briefly described afterwards. The domain is the classical channel inclined by angle θ, which, in order to differentiate the resulting surface wave from the gravitational one, is taken small, precisely in our simulation $\theta = 8°$.
In addition, a minimum length of the channel must be granted with the intent to reassure that the available duct size is enough for roll waves development. The formula to compute the minimal stabilizing distance presented in the work is the following

$$L_{min} = -ln\ \eta \left(\frac{u_0^2}{g \sin\theta} \right) \left(\frac{2(Fr+1)}{Fr \cos\theta (Fr-2)} \right), \tag{6.8}$$

where are combined the gravity g, the Froude number $Fr = \frac{u_0}{\sqrt{g \sin\theta h_0}}$, the typical velocity u_0, plus an additional constant $\eta = 10^{-4}$. Regarding Froude number, a point that must be taken care of, according to the shallow water theory, is that Fr should be greater than 0.5774, for the purpose of maintaining a laminar flow. Indeed, a last geometrical consideration is that, for the data acquisition system employed, that will be discussed in a following paragraph, a systematic error was found for values of the thickness smaller than 5mm, therefore the paper focuses its attention on height larger than 9 mm and, consequently, the simulations are reproduced with the initial thickness $h_0 = 9.83$ mm and $h_0 = 10.73$ mm, which moreover satisfy the condition for Froude number.
The liquid used during the experiments is glycerin, whose physical property at $25°C$ is a high dynamic viscosity, $\mu \sim 900$ mPa s, and a density equal to 1273 $\frac{kg}{m^3}$, similar to the one of the water. The reason behind this choice of this Newtonian fluid, is that it allows for the flow conditions to be maintained in the laminar regime with a smooth free surface, not surpassing the critical Reynolds number, hence ensuring that the thickness changement is mainly caused by the perturbation.
To introduce the aforementioned disturb at the inlet, even though maintaining a constant incoming discharge rate Q, the experimental setup consists in a bass-speaker coupled to a function generator, able to control the frequency, which were located at the start of the channel. However, the measurement system control, was installed at a certain distance from the disturbance apparatus, to remove external interferences. Going slightly into the details, the data acquisition technique relys on the combination of a photodetector and a laser. The laser, which is installed over the channel, emits a beam that is perpendicular to the bottom of the flow, crosses the dyed fluid in motion, and reaches the photodetector, which is installed directly beneath the channel bed. Then, the height is regained from the difference of light intensity before and after penetrating the fluid, according to Beer–Lambert law. As done for the viscous dam break problem, we extracted manually the data reported in the paper using the tool Engauge Digitizer.
However, before running a simulation to compare our numerical result with the experimental one, SU2 boundary condition set must be expanded, with objective to include a disturbed inlet input. As done for the configuration option addition, we enlarged the possible typology of constrains imposed at the frontier, retracing the option for the normal inlet structure also for the perturbed inlet. The name given to this alternative is MARKER_PERTURBED_INLET and the associated parameters are (marker name, temperature, velocity magnitude, flow_direction_x, flow_direction_y, thickness, amplitude, frequency), where we note that the flow in normal direction z can be omitted in favor of the total thickness. To end the list of inputs, we jointed the quantities proper to the perturbations, i.e. the amplitude in $\frac{m}{s}$ and the frequency in Hz of the

sinusoidal disturb applied to the velocity. We successively implemented the CSolver function BC_Perturbed_Inlet and added it to the integration class in the routine Space_Integration, in particular in the switch that lists all the boundary conditions. For the moment, the code is only able to handle a disturbance to the velocity, which is however sufficient for our work, where the sinusoidal perturbation equal to the 5% of u_0, repeated with a frequency of $3Hz$.
Now that we have all the necessary components, the simulation can be run, starting from an initial solution equal to the Nusselt profile, which solves the unperturbed problem, equivalent to the falling film introduced in Section 6.2. Beginning from the steady solution, avoids variations in the thickness caused by the convergence from the numerical solution to the analytic one. Another technique could be to let the computation reach the steady state and then add the perturbation, but this would require more changement to the software that we consider superfluous.
The solution is taken every 3 iteration. The result are presented in the following images Fig.6.13 and 6.14, where the thickness is scaled by its initial value h_0 for a better visualization, and they show that SU2 solution follow the same pattern, although the numerical result are smoother, phenomenon that may be due to the lack of precision from the function generator employed at low frequencies input.

Test 1: initial film thickness $h_0 = 9.83$ mm

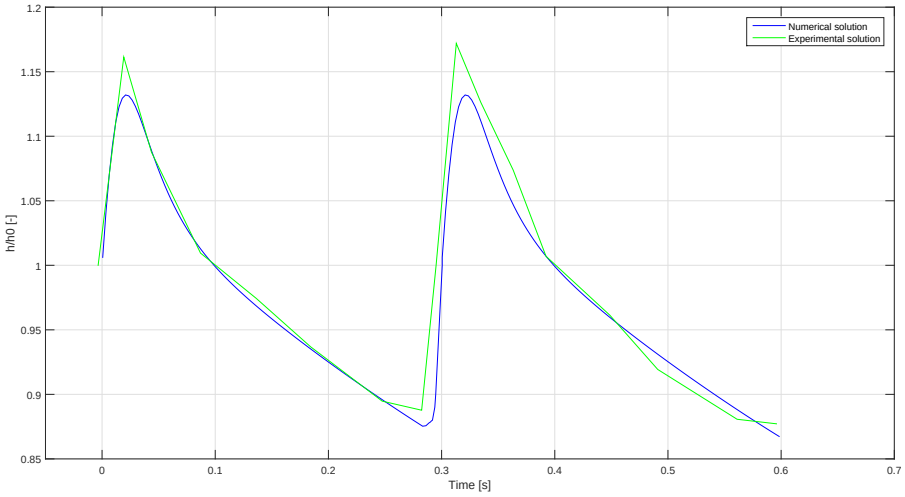

Figure 6.13: Comparison between experimental perturbed thickness and numerical result with mean height equal to 9.83 mm.

Moreover, we can see that the wave amplitude increases with Froude number, i.e. when the initial velocity u_0 is higher, because the other parameters are fixed, since the inclination θ and the channel length are immutable.
Tackling this test case, emphasizes that the code is capable of solving also perturbed problems which admit a limited solution.

Test 2: initial film thickness $h_0 = 10.73$ mm

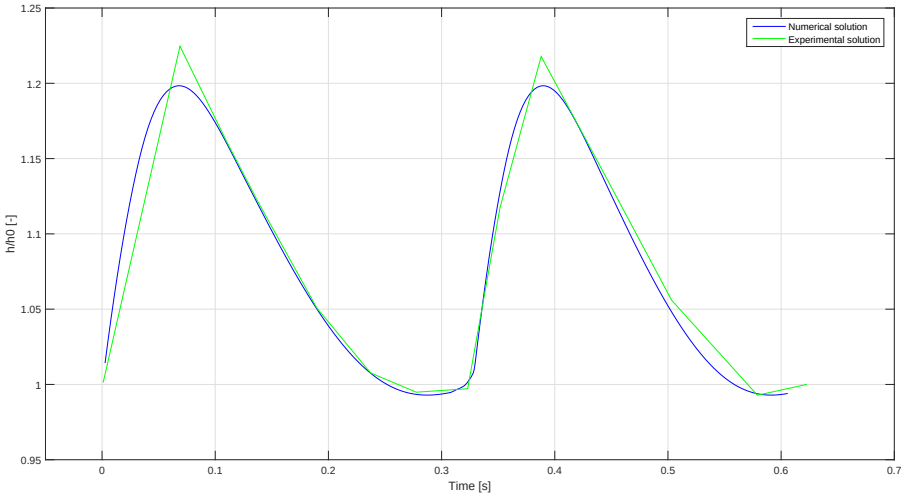

Figure 6.14: Comparison between experimental perturbed thickness and numerical result with mean height equal to 10.73 mm.

7 Conclusions

Before concluding the work, all the main objective are recapped and then verified their accomplishment. First of all, we state the profound motivation of this thesis, which is to find an alternative and less mathematically complicated way to close the averaged equations modeling the thin film, which is instead of using the Taylor expansion to adjust the non-linear terms, as done by most author in literature, the reconstruction the velocity profile.
Therefore, the first step was to regain the governing equations through the integration along the neglectable direction and then to elide smaller order terms, pointing out the similarities with the shallow water theory. In virtue of this analogy, the multilayer technique, which divides the geometry in an arbitrary number of stratum, on which it lives a similar set of equations, was explained and implemented. This method facilitates the profile reconstruction, because it can be regained using the mean value at each layer as interpolation knot. Furthermore, we specified the two possible hypothesis on the pressure that come from the asymptotic analysis, preferring the more straight forward hydrostatic assumption, which leads to what is known as primitive equations (PE). As a final step for the model construction, we verified the hyperbolic nature of the governing system, which restricts the fan of modus operandi numerically available.
Proceeding, the influence of shallow water philosophy extends also to the most recent numerical technique, like the one described in Kurganov's works. Therefore, planning to develop the resolution as an extension of the software SU2, we performed a discretization following the finite volume approach, which is the one already used by the code. After a general recall to this theory, we discussed the numerical approximation employed with the various terms that appear in the governing equations. In this specific case, as mentioned before, we utilized Kurganov central upwind approach for the convective part, while, for the source injection, the method are fully customed by the creator of this project for the determined system of equations.
Now that all the pieces are ready, we passed to the C++ implementation as an add-on to the SU2 free source software, which mainly consists in the specification of the principal elements as derived inheriting from the base classes in order to enable the run-time polymorphism, which widely characterizes SU2. Going briefly into the details, we wrote a simple program which extracts the film tasselation from the native su2 format and enables the integration of averaged geometries. Then, we designed a child for the solver class, which leads the residual system construction, the variable, container for the conserved quantities proper to the problem, and the numerical techniques between control volume nodes. Moreover, we added a lapack routine to solve the profile reconstruction problem and a particular output structure, to ease the result consultation for the film variables.
Finally, all the work is validated challenging the most common test case provided by the literature. In particular, the solutions produced by SU2 are confronted with both analytical expressions and experimental data to investigate the accuracy of the principal conserved variables, i.e. the velocity and the thickness. The film falling down an inclined plane serves as a benchmark for the parabolic profile of the velocity, while the dam break studies the evolution of the film height across time. In addition, those simple tests make the user acquainted with

the most frequently options necessary to set up a simulation with SU2.
A last test, more elaborate and realistic than the previous, replicates the film falling down an inclined plane with a sinusoidal perturbation affecting the inlet velocity. This result in a periodic variation of the film thickness, creating roll wave due exclusively to the disturb introduced upstream and not related with gravitational effects.
In conclusion, starting from the similarities with the shallow water and taking advantage of the tools already implemented for this typology of problems, we found a way to solve the thin film equations, without having to expand the non-linear terms, hence diminishing the complexity of the expression, but maintaining a certain accuracy also for the phenomenology under the water surface.
Further development of this work may be the study of the primitive equations with vertical viscosity (PEV^2 hypothesis) and a suitable numerical discretization. Another addendum, one could try to extend properly the resolution to the multizone case, which, although already possible, has not been tested and may require some smaller precautions. Finally, all the changement brought to the code can be considered as a starting point to look for brand new formulations and discretizations to solve the thin film problem with many other techniques.

Bibliography

[1] Craster R.V., Matar O.K. *Dynamics and stability of thin liquid films,* Reviews of Modern Physics, Volume 81, 5 August 2009.

[2] C. Schoof, , R. Hindmarsh, *Thin-Film Flows with Wall Slip: An Asymptotic Analysis of Higher Order Glacier Flow Models*, The Quarterly Journal of Mechanics and Applied Mathematics, January 2010

[3] C. Yanxiang, L. Qiang, Y. Yang, *Numerical Evaluation of The Effect of Icing Accretion on An Airship*, AIAA SPACE 2016.

[4] Stanton D.W., Rutland C.J., *Modeling Fuel Film Formation and Wall Interaction in Diesel Engines ,* SAE 960628, 1996.

[5] Stanton D.W., Rutland C.J., *Multi-Dimensional Modeling of Heat and Mass Transfer of Fuel Films Resulting from Impinging Sprays,* SAE 980132, 1998.

[6] Lavalle G., *Integral modeling of liquid films sheared by a gas flow,* Fluids mechanics [physics.class-ph]. ISAE - Institut Supérieur de l'Aéronautique et de l'Espace, 2014.

[7] D. J. Benney, *Long waves on liquid films*, J. Math. Phys., 1966, 150–155.

[8] Oron A., Davis S.H., Bankoff S.G., *Long-scale evolution of thin film,* Reviews of Modern Physics, Vol. 69, No. 3, July 1997.

[9] Macías J., Pares C., Castro M. J., *Improvement and generalization of a finite element shallow-water solver to multi-layer systems,* Int. J. Numer. Meth. Fluids 31. 1999.

[10] Lions J.L., Temam R., Wang S., *On the equations of the large-scale ocean,* Nonlinearity, Semptember 1992, pp 1007-1053.

[11] Kurganov A., *Finite-volume schemes for shallow-water equations,* Acta Numerica, 2018, pp. 289-351.

[12] Marqués J. M. F., *Introduction to the Finite Volumes Method. Application to the Shallow Water Equations.*

[13] Forte C., *Sviluppo e validazione di un modello di film fluido per simulazioni CFD di motori a combustione interna,* PhD Thesis Università di Bologna, 2008.

[14] Park, C. D., Nosoko, T., *Three-dimensional wave dynamics on a falling film and associated mass transfer,* AIChE Journal 49, 11 (Nov. 2003), 2715-2727.

[15] Nosoko T., Miyara, A., *The evolution and subsequent dynamics of waves on a vertically falling liquid film,* Physics of Fluids 16, 4 (2004), 1118-1126.

[16] Nusselt W., *Die Oberflachenkondensation des Wasserdampfes,* VDI-Zeitschrift 60 (1916), 541-546.

[17] Davis S. H., *Thermocapillarity instabilities,* Annu. Rev. Fluid Mech. 19 (1987), 403-435.

[18] Trujillo M. F., Lee C. F., *Modeling Film Dynamics in Spray Impingement,* Journal of Fluids Engineering, January 2003, Vol. 125 .

[19] Macías J., Pares C., Castro M. J.*A Multilayer Shallow Water Model. Application to the Strait of Gibraltar and the Alboran Sea,* NATO ASI Series, Vol 148, 1997.

[20] E. Audusse emphA Multilayer Saint-Venant Model: Derivation and Numerical Validation, Discrete and Continous Dynamical Systems, Volume 5, Number 2, May 2005.

[21] Quarteroni A. *Modellistica numerica per problemi differenziali,* Springer, 6a edizione.

[22] LeVeque R. *Finite-Volume Methods for Hyperbolic Problems,* Cambridge Texts in Applied Mathematics, 2002.

[23] Bryson S., Epshteyn Y., Kurganov A., *Well-Balanced Positivity Preserving Central-Upwind Scheme on Triangular Grids for the Saint-Venant System,* ESAIM Mathematical Modelling and Numerical Analysis May 2011.

[24] Quarteroni A., Sacco R., Saleri F., *Matematica Numerica,* Springer, 2a edizione.

[25] Gottlieb S., Shu Chi-Wang, *Strong Stability Preserving High Order Time Discretization Methods,* SIAM Review, May 2001.

[26] Kurganov A., *Well-Balanced Positivity Preserving Central-Upwind Scheme for the Shallow Water System with Friction Term,* Tulane University.

[27] Pacheco J. R., Pacheco-Vega A., *Analysis of Thin Film Flows Using a Flux Vector Splitting,* Journal of Fluids Engineering, March 2003.

[28] Muccino J. C., Gray W. G., Foreman M. G. G., *Calculation of vertical velocity in three-dimensional, shallow water equation, finite elements model,* International Journal for Numerical Methods in Fluids, Vol 25 (1997).

[29] Economon T. D., Palacios F., *SU2: An Open-Source Suite for Multiphysics Simulation and Design,* AIAA JOURNAL Vol. 54, March 2016.

[30] Orlando G., *Development of a laminar numerical solver for reacting flows based on the SU2 CFD code,* PACS Report A.Y. 2018-2019.

[31] Delestre O., Laguerre C., Lucas C., Ksinant P.-A., *SWASHES: a compilation of Shallow Water Analytic Solutions for Hydraulic and Environmental Studies,* International Journal for Numerical Methods in Fluids, January 22, 2016.

[32] Wang B., Liu W., Wang W., Zhang J., *Experimental and numerical investigations of similarity for dam-break flows on wet bed,* Journal of Hydrology 582 (2020).

[33] J. J. Stoker, *Water Waves: The Mathematical Theory with Applications,* volume 4 of Pure and Applied Mathematics, Interscience Publishers, New York, USA, 1957.

[34] Fiorot G.H., Maciel G.F., Cunha E.F., Kitano C., *Experimental setup for measuring roll waves on laminar open channel flows,* Flow Measurement and Instrumentation 41 (2015).

Sitography

[a] SU2 - https://su2code.github.io/docs_v7/home/

[b] Meson - https://mesonbuild.com/

[c] Lapack - http://www.netlib.org/lapack/explore-html/

[d] Engauge Digitizer - http://markummitchell.github.io/engauge-digitizer/