UCLouvain
Institute of Mechanics,
Materials and Civil Engineering

Two-phase flows simulation using the X-Mesh approach

Antoine Quiriny

Thesis typescript submitted in partial fulfilment of the requirements
for the Ph.D. Degree in Engineering Science

Thesis Committee

Advisors	Jean-François Remacle	UCLouvain
	Jean-Philippe Ponthot	ULiège
Jury	Jonathan Lambrechts	UCLouvain
	Nicolas Moës	UCLouvain
	Hadrien Rattez	UCLouvain
	Vincent Moureau	CORIA
Chairperson	Paul Fisette	UCLouvain

Louvain-la-Neuve, November 3, 2025

Remerciements

Toutes les bonnes choses ont une fin, et cette thèse ne fait pas exception à la règle. Après plusieurs années de travail acharné, de découvertes et d'apprentissage, il est temps pour moi de tourner la page sur cette aventure académique. Mais avant de le faire, je tiens à exprimer ma profonde gratitude envers toutes les personnes qui ont contribué à rendre cette expérience inoubliable.

J'aimerais remercier mon promoteur Jean-François, d'abord pour m'avoir donné l'opportunité de réaliser cette thèse, mais surtout pour son encadrement de qualité, ses conseils avisés et son enthousiasme communicatif. Merci également à Jonathan pour son aide précieuse tout au long de cette thèse. J'ai tellement appris à tes côtés, merci pour ta patience, ta disponibilité et ta bonne humeur. Un grand merci aussi à Nicolas, Václav et Vincent pour leur collaboration et leurs conseils tout au long de ces années.

L'environnement de travail est sans doute l'un des aspects les plus importants d'une thèse et sur ce point-là, j'ai été gâté. Le MEMA m'a accueilli à bras ouverts et j'y ai trouvé ma place très rapidement, dans un cadre de travail convivial et stimulant. Merci à tous les membres du MEMA, anciens et actuels, pour tous ces bons moments partagés, les discussions passionnantes et l'entraide constante. Je garde en souvenir tous les fous-rires, les afterworks ensemble, les matchs de la coupe internationale et intercontinentale de foot de l'Euler et les pauses cafés à rallonges. Un tout grand merci à Astrid, P-A, Ange, Insaf, Chiheb, Nathan, Mattéo, Mohamed, Michel, Thomas, Guillaume et Guillaume, Alexandre, Anthony, Simon, Miguel, Illy, Vincent, Christophe, Florian, Gatien, Nathan, Yana, Daoud, Kerem, Alexandre, Félix, Massimo, James et tous ceux que j'oublie. Le bâtiment Euler ne serait pas ce qu'il est sans les INMA avec Yassine, Julien, Philémon, Alexandre et les autres. Merci pour votre bonne humeur et tous les moments passés ensemble.

J'aimerais également remercier mes amis pour leur soutien et leur présence tout au long de ces années. Ils m'ont accompagné dans tellement de moments et ont contribué à façonner la personne que je suis aujourd'hui. Un grand merci à eux !

Je tiens aussi à remercier ma famille pour leur soutien inconditionnel et leur amour. Merci à mes parents pour m'avoir toujours encouragés et pour m'avoir

donné les moyens de réussir. Merci à mes trois frères pour leur complicité et leur soutien constant.
Enfin, un immense merci à Ophélie pour sa patience, son amour et son soutien indéfectible. Tu as été pour moi une source de force et de motivation tout au long de cette aventure. Tu as toujours été là pour moi, dans les bons comme dans les mauvais moments, et je ne saurais jamais assez te remercier pour cela. Je suis extrêmement reconnaissant de t'avoir à mes côtés.

Alors que le monde académique court après les publications et les citations, je suis très fier de pouvoir dire que vous avez été mes véritables références tout au long de mon parcours. Merci à vous toutes et tous du fond du cœur.

Contents

Contents **iii**

1 Introduction **1**
1.1 The X-Mesh approach 2
1.2 An illustrative example 4
1.3 Mesh conditions in the Finite Element Method 15
1.4 A finite element solver for the incompressible Navier-Stokes equations 19
1.5 Objective and methodology 25
1.6 Publications and conferences 27

2 The tempered finite element method for degenerated elements **29**
2.1 Finite elements and meshes: A complex relationship 31
2.2 A simple numerical experiment 34
2.3 Theory behind the TFEM scheme 47
2.4 Extensions 59
2.5 Conclusion 69

3 Interface representation and mesh deformation algorithm **73**
3.1 The level set approach 74
3.2 The front-tracking approach 83

4 Application to two-phase flows **103**
4.1 Solver and coupling algorithm 104
4.2 Surface tension and jump conditions at the interface 105
4.3 Numerical examples - levelset approach 107
4.4 Numerical examples - front-tracking approach 119
4.5 Two-phase jet 129

5 Conclusion **133**
5.1 A summary of the contributions 134
5.2 Perspectives and research opportunities 135
5.3 Final words 140

Bibliography **141**

A Appendix **147**
A.1 Appendix: Study of the FEM stiffness matrix 147
A.2 Appendix: Analysis of tempered FEM on zero-measure elements . . 149

CHAPTER 1

Introduction

Contents

1.1 The X-Mesh approach **2**
1.2 An illustrative example **4**
1.2.1 With a moving interface 10
1.3 Mesh conditions in the Finite Element Method . . **15**
1.4 A finite element solver for the incompressible Navier-Stokes equations **19**
1.4.1 Galerkin finite element formulation 20
1.4.2 Arbitrary Lagrangian-Eulerian formulation 23
1.4.3 Two-phase flow formulation 24
1.5 Objective and methodology **25**
1.5.1 Outline of the thesis 26
1.6 Publications and conferences **27**

When considering a fluid flow, one often overlooks the presence of another ubiquitous component: air. In fact, most flows we encounter are actually two-phase flows. Although, the role of air in something as massive as a tsunami may not be decisive for the devastation it causes onshore, but in many other situations, air changes everything.

A neat experimental demonstration comes from [Xu et al. (2005)], where researchers studied how a water drop behaves when it hits a solid plate. They performed the experiment under normal atmospheric conditions and then repeated it under vacuum. The results were striking: without air, the drop never splashed. Why? At impact, the air trapped beneath the drop is compressed and forced outward, dragging water along with it and generating splashes. No air, no splashing. The takeaway is simple: if air is left out of a simulation, such dynamics will never be captured.

Two-phase flows are everywhere: from the raindrops sliding down a window to the fuel injected into car engines, from the bubbles in the coffee that fueled this very writing to the ink sprayed onto this page. Understanding and predicting the behavior of two-phase flows is crucial for a wide range of applications, including environmental science, engineering, and medicine.

At small scales, surface tension effects become significant, leading to complex phenomena such as droplet formation, coalescence, and breakup. Because surface tension arises from the coexistence of multiple phases, it is natural to focus on two-phase systems when studying small-scale fluid behavior.
In this thesis, we will focus on the development of numerical methods for simulating two-phase flows with sharp interfaces. We will use the finite element method (FEM) as our numerical framework and consider incompressible, immiscible fluids with different densities and viscosities. The main challenge in simulating two-phase flows is to accurately capture the interface between the two fluids, which can undergo large deformations and topological changes. For this, we will use the eXtreme Mesh deformation (X-Mesh) method [Moes et al. (2021); Quiriny et al. (2024)], which allows for large deformations of the interface while maintaining a good representation of the solution.

1.1 The X-Mesh approach

Numerical schemes to address problems with moving interfaces can be classified into two main categories: interface-tracking methods and interface-capturing methods. In the interface-tracking methods, the Arbitrary Lagrangian-Eulerian (ALE) method [Hughes et al. (1981)] is the most well-known and widely used approach. It involves deforming the mesh to follow the moving interface. This

approach allows for a sharp representation of the interface and accurate imposition of boundary conditions, but it can lead to too much mesh distortion and requires remeshing when the interface undergoes large deformations or topological changes. On the other hand, interface-capturing methods, such as the levelset method [Sussman and Fatemi (1999); Marchandise and Remacle (2006); Nagrath et al. (2005)] or the volume-of-fluid (VOF) method [Hirt and Nichols (1981)], use a fixed mesh and represent the interface implicitly through a scalar function. These methods can handle complex interface dynamics and topological changes more easily, but they often suffer from numerical diffusion, leading to a smeared representation of the interface and challenges in accurately imposing boundary conditions.

In this thesis, we propose to use a novel approach called the X-Mesh method [Moes et al. (2021); Quiriny et al. (2024)], which combines the advantages of both interface-tracking and interface-capturing methods. The X-Mesh method started from an observation: none of the methods proposed in the state of the art is entirely satisfactory. For a scheme to be satisfactory, the authors defined four fundamental properties it should fulfill:

- **P1 - Time continuous deforming mesh with fixed topology:** The mesh should deform in a time continuous manner: nodes' trajectories form continuous paths and the mesh adjancies are fixed.

- **P2- Conforming mesh to sharp physical interface.**

- **P3 - Compliant with topological changes:** nucleation, collapse, coalescence and splitting.

- **P4 - Classical finite element approximation:** no enrichment with additional degrees of freedom.

Properties P1, P2, and P4 ensure that the nodal solution evolves continuously in time, thereby improving the scheme's accuracy, while P3 enables the treatment of complex interface dynamics.
No method in the state of the art satisfies all these properties. The levelset and VOF methods satisfy P1, P3 and P4 but not P2 as the interface is inside elements. The extended finite element method (XFEM) [Moës et al. (1999)] satisfies P1, P2 and P3 but not P4 as it requires enrichment of the finite element space with additional degrees of freedom. The ALE method, despite satisfying the P1, P2 and P4 properties, has always been seen as incompatible with topological changes (P3) or large deformations of the interface. It is possible to tackle this issue by remeshing when the mesh is too distorted or when

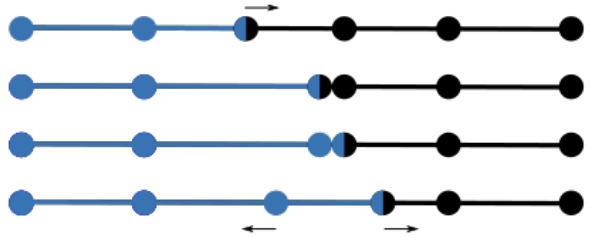
topological changes occur. Unfortunately, remeshing introduces the need to project the solution from the old mesh to the new mesh, which cannot satisfy at the same time the balance laws and the time-continuity of the solution (P1).

In order to satisfy all those properties, the authors of the seminal X-Mesh paper [Moes et al. (2021)] came with a striking conclusion. *The quest for meshes with nice-only elements is counter-productive: bad quality elements or even elements with a zero measure opens a new spectrum of possibility.* The main idea of the X-Mesh method is thus to allow degenerate elements to appear in the mesh and in this way consider *a new paradigm for mesh movements*. In this way the mesh can deform in a time continuous manner (P1) while still conforming to the sharp interface (P2) and allowing topological changes (P3). The X-Mesh method also uses a classical finite element approximation (P4) without any enrichment of the finite element space.

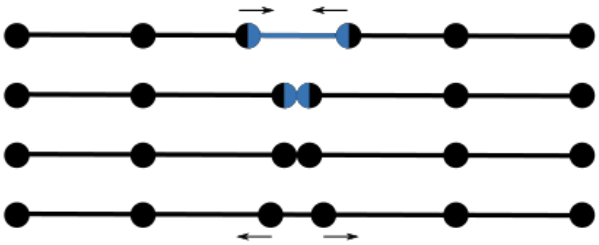
Those degenerate elements allows us to continuously track the interface in situations where the ALE method struggles. To illustrate this, Figure 1.1 represents 3 cases in 1D where those degenerate elements come in handy. The first case is the relaying of the interface position from one node to another. In the ALE method, the interface is represented by a single node that moves with the interface and the whole mesh would be deformed, leading to huge elements if the displacement is large. In the X-Mesh method, the interface is represented by a node that moves with the interface until it reaches a new node of the mesh. At this point, the front is passed to the new node, and the old node is no longer on the interface, slowly moving back to its original position. This is represented in Figure 1.1 where the node carrying the interface is represented by a bicolor dot. This way, the mesh is only locally deformed, preserving an overall good quality of the mesh. The relay of the interface position is made in a time continuous manner which leads to the creation of a degenerate element when the front is passed to a new node. For nucleation and collapse of an interface, the idea is similar. We use a degenerate element to create an appearing phase (nucleation) or to make a phase disappear (collapse), both in a time continuous manner.

1.2 An illustrative example

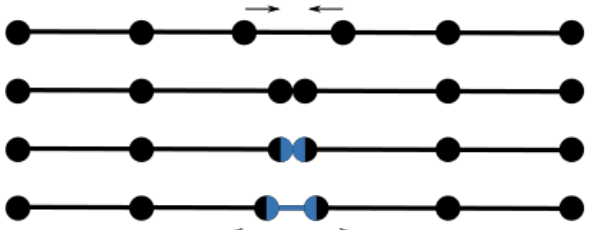
When studying fluid mechanics in class, one of the first types of flow we learn about is the Poiseuille flow. There is a pressure difference between the inlet and the outlet of a channel, and after a certain amount of time, the flow is established, leaving only one velocity in the longitudinal direction of the



(a) Relaying



(b) Collapse



(c) Nucleation

Figure 1.1: Three examples of degenerate elements in 1D that allows to track the interface in a time continuous manner: (a) relaying of the interface, (b) collapse of an interface and (c) nucleation of an interface.

channel. The Navier-Stokes equations can be simplified, leaving only:

$$\mu \frac{\partial^2 u}{\partial y^2} = \frac{\partial p}{\partial x} = -G \tag{1.1}$$

where μ is the dynamic viscosity, u is the velocity in the longitudinal direction and G is the driving pressure gradient.
We have a parabolic velocity profile, with the maximum velocity in the center of the channel, and zero velocity at the walls due to the no-slip boundary

condition:

$$u(y=0)=0$$
$$u(y=H)=0$$

where H is the height of the channel.
Instead of considering only one fluid, we can consider two fluids with different viscosities. The dynamic viscosity is not a constant anymore, but a function of space:

$$\mu(y)=\begin{cases}\mu_1, & \text{if } y<h_\mathrm{i}\\ \mu_2, & \text{if } y\geq h_\mathrm{i}\end{cases}$$

where h_i is the height of the interface between the two fluids.
This problem can be solved analytically by treating each fluid region separately and applying the boundary conditions at the walls and the interface. The velocity profile is continuous at the interface, and the shear stress is also continuous:

$$u_1(y)=\frac{G}{2\mu_1}y^2+A_1y+B_1, \quad y<h_\mathrm{i} \tag{1.2}$$
$$u_2(y)=\frac{G}{2\mu_2}y^2+A_2y+B_2, \quad y\geq h_\mathrm{i} \tag{1.3}$$

$$\begin{aligned}u_1(0)&=0\\ u_2(H)&=0\\ u_1(h_\mathrm{i})&=u_2(h_\mathrm{i})\\ \mu_1\left.\frac{\partial u_1}{\partial y}\right|_{y=h_\mathrm{i}}&=\mu_2\left.\frac{\partial u_2}{\partial y}\right|_{y=h_\mathrm{i}}\end{aligned}$$

where u_1 and u_2 are the velocities in the two fluids. The constants A_1, B_1, A_2 and B_2 can be determined by solving the system of equations given by the boundary conditions.

$$\begin{aligned}A_1&=\frac{0.5\cdot G\left(\frac{h_\mathrm{i}^2-H^2}{\mu_2}-\frac{h_\mathrm{i}^2}{\mu_1}\right)}{h_\mathrm{i}-\frac{\mu_1}{\mu_2}(h_\mathrm{i}-H)}\\ B_1&=0\\ A_2&=\frac{\mu_1}{\mu_2}A_1\\ B_2&=-\frac{G}{2\mu_2}H^2-A_2H\end{aligned}$$

This solution is represented in figure 1.2.
This two-fluid Poiseuille flow can also be solved numerically, and we can compare the numerical solution to the analytical solution to see how well the numerical method performs. As in the rest of this thesis, we will use the finite element method to solve the problem numerically.
Let us denote the 1D domain by Ω and define the space $V = H_0^1(\Omega)$, the corresponding weak form of equation 1.1 is then: Find $u \in V$, such that

$$\int_\Omega \mu(y) \frac{\partial u}{\partial y} \frac{\partial v}{\partial y} \, dy = \int_\Omega G v \, dy, \quad \forall v \in V$$

We will solve this problem on a mesh denoted by $\mathcal{T}_h$, with h the mesh size. The discrete finite element space is then $V_h \subset V$, where we have limited ourselves to piecewise linear functions. The discrete problem is then: Find $u_h \in V_h$, such that

$$\int_\Omega \mu(y) \frac{\partial u_h}{\partial y} \frac{\partial v_h}{\partial y} \, dy = \int_\Omega G v_h \, dy, \quad \forall v_h \in V_h$$

We can then compute the error between the analytical solution u and the numerical solution u_h in the L^2 norm:

$$\|u - u_h\|_{L^2} = \left(\int_\Omega (u - u_h)^2 \, dy \right)^{1/2}$$

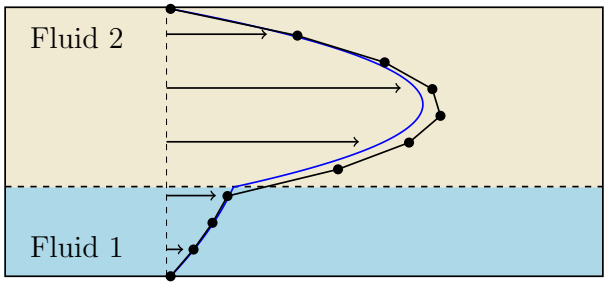


Figure 1.2: Analytical solution (blue) and numerical solution for the sharp levelset approach (black dots) for the two-fluid Poiseuille flow.

In this example, the representation of the interface will be our main focus. We will compare four different methods to represent the interface between the two fluids: a sharp levelset approach, a smooth levelset approach, an Arbitrary Lagrangian-Eulerian (ALE) approach and an X-mesh approach. For the steady-state case, the ALE and X-mesh approaches are equivalent, as the interface does not move.

In the sharp levelset approach, the mesh is fixed and the interface is represented by a levelset function ϕ, which is a signed distance function to the interface. The viscosity is then switched between the two fluids at the interface in a sharp manner but inside an element leading to an error as we can see in the Figure 1.2. In the ALE and X-Mesh case, the interface is represented directly inside the mesh by positioning a node on the interface as we can see in the Figure 1.3. For the smooth levelset approach, the mesh is fixed and the viscosity is smoothed over a certain distance 2ϵ around the interface with a smooth Heaviside function:

$$\begin{aligned}
\mu(y) &= \mu_1 + (\mu_2 - \mu_1) H_\epsilon(\phi(y)) \\
H_\epsilon(\phi) &= 6\left(\frac{\phi+\epsilon}{2\epsilon}\right)^5 - 15\left(\frac{\phi+\epsilon}{2\epsilon}\right)^4 + 10\left(\frac{\phi+\epsilon}{2\epsilon}\right)^3, \quad \text{if } |\phi| < \epsilon \\
H_\epsilon(\phi) &= 1, \quad \text{if } \phi \geq \epsilon \\
H_\epsilon(\phi) &= 0, \quad \text{if } \phi \leq -\epsilon
\end{aligned}$$

where ϵ is the smoothing distance.

A convergence study was performed by refining the mesh and computing the

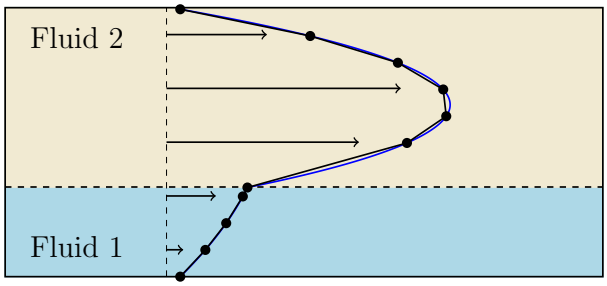


Figure 1.3: Analytical solution (blue) and numerical solution for the ALE/X-Mesh approach (black dots) of the two-fluid Poiseuille flow.

error in the L^2 norm for each method. The results are shown in figure 1.4. We can see that the X-Mesh/ALE method performs the best with a convergence rate of 2. The sharp levelset and smooth levelset methods when $\epsilon = 1.5h$ only reach a convergence rate of 1. The smooth levelset method actually converges as $\mathcal{O}(\epsilon)$, so if we consider a fixed $\epsilon = 0.075$ the error stagnates when refining the mesh. With this simple example, we can already see the advantages of using an X-Mesh/ALE approach to represent the interface between two fluids. The mesh is conform to the interface which allows for a more accurate representation of the discontinuity in viscosity and leads to a better numerical solution. It was shown in [Moës et al. (2003)] that the X-FEM method can also reach optimal convergence rates for such two-phase problems. The method consists in

enriching the finite element space with additional degrees of freedom to capture the discontinuity in the material properties at the interface. This approach, while effective, is quite invasive in FEM codes as it requires modifying the finite element space and adding additional degrees of freedom.
Let's also note that we can use the Ghost Fluid Method (GFM) [Fedkiw et al. (1999)] to solve this problem. The result for this method is also shown in figure 1.4 and we can see that it almost reaches optimal convergence rates while being less invasive than the X-FEM method. Nonetheless, it still requires some modifications to the finite element formulation to account for the discontinuity at the interface. The idea of the GFM is to consider a ghost value u_Γ at the interface and impose the continuity of the velocity and the shear stress at the interface through this ghost value. By imposing those conditions, the ghost value can be expressed as a function of the values of the neighboring nodes and thus disappears from the final formulation. Inside the elements cut by the interface, the solution and integrals are splitted into two parts, one for each fluid. In 1D, the implementation of the GFM is quite straightforward, but in 2D and 3D, it becomes more complex as the interface can cut the elements in various ways.
We considered here a static case, but the real advantage of the X-Mesh approach appears when the interface moves.

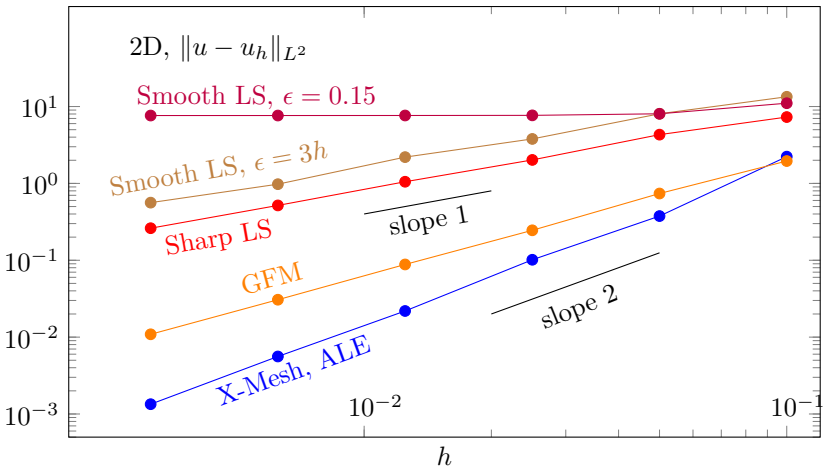


Figure 1.4: Convergence study for the steady-state two-fluid Poiseuille flow.

1.2.1 With a moving interface

In order to demonstrate the advantage of the X-Mesh approach for moving interfaces, we can consider a temporal version of the two-fluid Poiseuille flow. We will consider the same problem as before, but now the interface between the two fluids moves with a constant velocity. We have a manufactured solution for this problem that correspond to the analytical solution of the steady-state case, but where the interface position h_i is now a function of time and where we have introduced a vertical velocity $v(y,t)$:

$$h_\mathrm{i}(t) = 0.8 - 0.6t \tag{1.4}$$

$$v(y,t) = \begin{cases} -0.6\,\frac{y}{h_\mathrm{i}}, & \text{if } y < h_\mathrm{i}(t) \\ -0.6\,\frac{1-y}{1-h_\mathrm{i}}, & \text{if } y \geq h_\mathrm{i}(t) \end{cases} \tag{1.5}$$

Three new terms appear in the equations: a time derivative, a convective term due to the movement of the interface and a forcing term to account for the movement of the interface. The equations are then:

$$\rho\frac{\partial u}{\partial t} + \rho v\frac{\partial u}{\partial y} = G + \mu(y)\frac{\partial^2 u}{\partial y^2} + F_\mathrm{manufactured}$$

where ρ is the density of the fluid and for the sake of simplicity, we will consider that both fluids have the same density $\rho = 1$.
As all the solutions in time have the pressure gradient term G and the viscous term $\mu(y)\frac{\partial^2 u}{\partial y^2}$ that cancel each other out, we can compute the forcing term $F_\mathrm{manufactured}$ by :

$$F_\mathrm{manufactured}(x,t) = \frac{\partial u}{\partial t} + v\frac{\partial u}{\partial y}$$

The solution u is bi-harmonic and is given by equations 1.2 and 1.3 where we use the height and vertical velocity imposed in 1.4 and 1.5.
To solve this problem numerically, we will use the same finite element method as before, but now we will also discretize in time. We will use a backward Euler scheme for the time discretization. The discrete problem at each time step is then: Find $u_h^{n+1} \in V_h$, such that

$$\int_{\Omega^{n+1}} \frac{u_h^{n+1} - u_h^n}{\Delta t} v_h \, dy + \int_{\Omega^{n+1}} (v - v_\mathrm{mesh})\frac{\partial u_h^{n+1}}{\partial y} v_h \, dy$$
$$+ \int_{\Omega^{n+1}} \mu(y)\frac{\partial u_h^{n+1}}{\partial y}\frac{\partial v_h}{\partial y} \, dy = \int_{\Omega^{n+1}} (G + F_\mathrm{manufactured}) v_h \, dy, \quad \forall v_h \in V_h$$

where u_h^n is the solution at the previous time step, Δt is the time step size and v_mesh is the mesh velocity computed as $v_\mathrm{mesh} = \frac{y_{n+1} - y_n}{\Delta t}$.

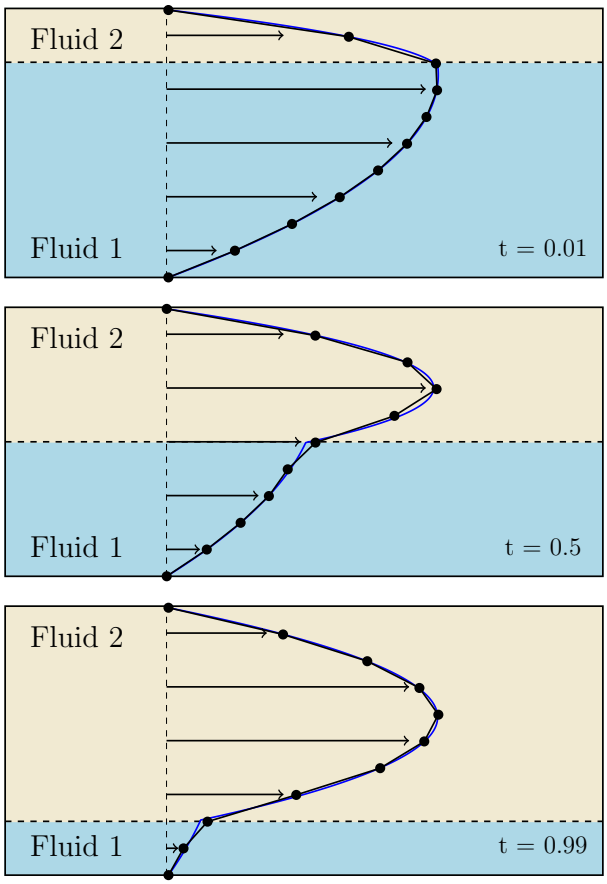


Figure 1.5: Different time snapshots for the sharp levelset approach

The analytical solution along with the numerical solution for the sharp levelset approach is shown in figure 1.5 for different times t. The good distribution of the mesh allows to capture the velocity profile well but does not capture the discontinuity in viscosity at the interface very well.
On the other hand, the ALE approach, for which snapshots are shown in figure 1.6, allows to capture the discontinuity in viscosity at the interface very well but as the interface moves, the mesh becomes distorted, leading to a loss of accuracy. This is where the X-Mesh approach comes in, as it allows to move the mesh nodes in order to keep a good quality of the mesh while still capturing the discontinuity in viscosity at the interface.
Figure 1.7 illustrates the time evolution of the interface relay in the X-Mesh

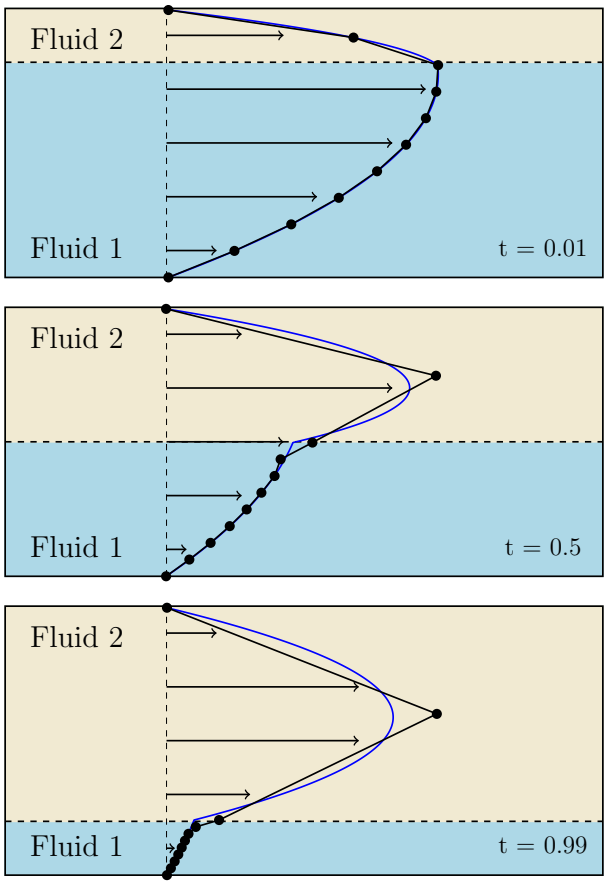


Figure 1.6: Different time snapshots for the ALE approach

approach. We can see that one node is positioned on the interface and moves with it until it reaches a new node of the mesh. At this point, the front is passed to the new node, and the old node is no longer on the interface, slowly moving back to its original position. This way, the mesh quality is preserved while still capturing the discontinuity in viscosity at the interface. Let's note that this relay is made in a continuous manner which leads to the creation of a degenerate element when the front is passed to a new node. This is the key idea of the X-Mesh approach: allowing degenerate elements to appear in order to switch the front nodes in a time continuous manner that doesn't introduce an error.
We can perform a convergence study by refining the mesh and computing the

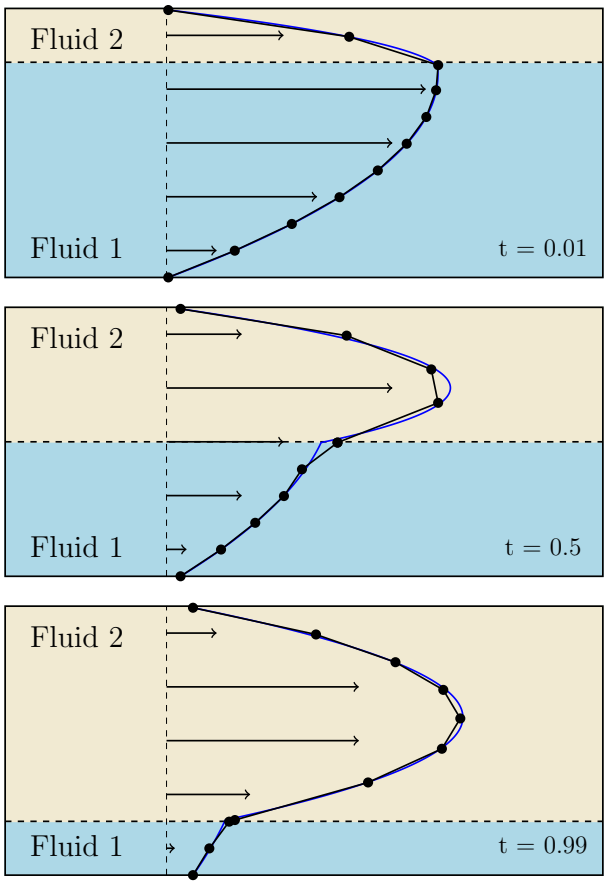


Figure 1.7: Different time snapshots for the X-Mesh approach

error in the L^2 norm for each method at the final time $T = 1$. The results are shown in figure 1.8 where we can observe that the X-Mesh and ALE methods both preserve their convergence rate of 2, but the X-Mesh method has a smaller error constant due to the better node distribution. The sharp levelset and smooth levelset methods when $\epsilon = 1.5h$ again only reach a convergence rate of 1, and the smooth levelset method with a fixed $\epsilon = 0.075$ again stagnates when refining the mesh.

This example shows the advantages of using a conforming mesh to represent the interface between two fluids, as it allows one to capture the discontinuity in viscosity at the interface more accurately, leading to a better numerical solution. Only the ALE and X-Mesh approaches are able to conserve the sec-

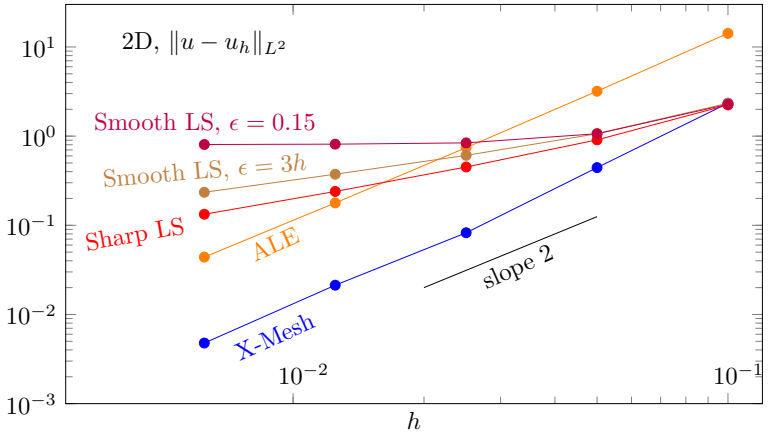


Figure 1.8: Convergence study of the L^2 error at time $T = 1$ for the temporal two-fluid Poiseuille flow.

ond order of convergence in space, but the X-Mesh approach allows to keep a good quality of the mesh during the whole simulation, leading to a smaller error constant. This motivates the use of the X-Mesh method in the context of simulations with moving interfaces and thus in the context of multiphase flows. The ALE approach keep the same convergence rate in this case also because the solution considered is bi-harmonic and the ideal mesh for such a solution is a uniform one. In this example, the ALE deformed mesh keeps a uniformity in both fluid domains, which is favorable to the solution.

Of course, we have chosen an example that is particularly unfavorable to the ALE method, as the interface moves a lot during the simulation. Usually, when a mesh becomes too distorted, a remeshing is performed and with it a projection of the solution from the old mesh to the new mesh. This projection introduces an error that can be significant and that can lead to a loss of accuracy. The X-Mesh approach allows to avoid this remeshing and the associated projection error, leading to a more accurate solution. We also need to note that since this simple example is 1D, the ALE and X-Mesh approaches capture exactly the interface position. This simple example doesn't characterize the error due to a curved interface that is not exactly represented by the mesh.

Since X-Mesh introduces degenerate elements in the mesh, it is crucial to understand how mesh quality affects convergence. The next section provides a brief historical overview of mesh quality criteria in the Finite Element Method, setting the stage for the developments in Chapter 2.

1.3 Mesh conditions in the Finite Element Method

The finite element method (FEM) has become one of the most versatile and widely used numerical techniques for solving partial differential equations arising in physics and engineering. One of its major advantages is its ability to operate on irregular meshes, which makes it particularly attractive for domains with complicated geometries. However, the flexibility of FEM comes with its own set of challenges, particularly regarding mesh quality. Still today, there is no necessary and sufficient condition on the mesh to guarantee convergence of the FEM solution to the true solution as the mesh is refined. To allow degenerate elements in the mesh, as done in the X-Mesh approach, it is crucial to understand how mesh quality affects convergence. This section provides a historical overview of key developments in characterizing mesh quality for FEM, highlighting significant contributions and open questions in the field.
In 2D, triangular elements are the most commonly used in FEM due to their simplicity and flexibility in meshing complex geometries. These elements can become degenerate in 3 different ways as shown in figure 1.9: regular, needle and cap elements. A regular degenerate element is when all the vertices of the triangle collapse into a single point. A needle element is when two vertices collapse, producing an 0 degree angle. A cap element is when one vertex approaches the opposite edge, producing a very flat triangle with a 180 degree angle.

The first condition to guarantee convergence of the FEM solution was the

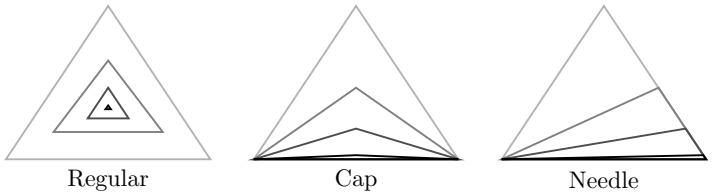


Figure 1.9: Different types of degenerate triangular elements: regular, needle and cap.

minimum angle condition developed independently by Zlámal [Zlámal (1968)] and by Ženíšek [Ženíšek (1969)] in 1968 and 1969 respectively. This condition requires that all angles in a triangulation are bounded away from zero, ensuring that no needle nor cap elements appear in the mesh. This condition is sufficient to guarantee convergence of the FEM solution for linear elements, but it is overly restrictive and not necessary.

The second major development came in the late 1970s, where the minimum an-

gle condition was relaxed independently by [Babuška and Aziz (1976)], [Barnhill and Gregory. (1976)] and [Jamet (1976)]. It is mainly the paper of Babuška and Aziz [Babuška and Aziz (1976)] that popularized this new condition, known as the maximum angle condition. This condition requires that no angle in the triangulation approaches 180 degrees, allowing needle elements but not cap elements. Babuška and Aziz described it as "essential" for convergence, but carefully avoiding the word necessary. This "essential" condition became the standard mesh quality measure for decades and even today, many commercial FEM software still use it as a mesh quality indicator.
More recently, Kobayashi [Kobayashi and Tsuchiya (2015)] established a profound connection between FEM mesh conditions and classical interpolation theory. They noted that the minimum and maximum angle conditions had already been discovered in the 1920s within interpolation theory. The rediscovery in FEM is unsurprising: both rely on constructing an approximant within the finite element space that converges, then applying Céa's lemma to ensure FEM convergence. Babuška and Aziz as well as Zlámal and Ženíšek all constructed their approximants using the Lagrange interpolation of the true solution.

In the interpolation theory setting, a classic counterexample is the Schwarz lantern, in which a mesh refinement approximating a cylinder fails to converge in surface area. This exact example corresponds to the example proposed by Babuška and Aziz of a mesh that violates the maximum angle condition and for which the FEM solution does not converge. This example and the Schwarz lantern are represented in Figure 1.10. Kobayashi's key insight here was to demonstrate that the convergence of the area between two surfaces imply the convergence of the gradients and thus the convergence in the H^1 norm. He then demonstrate that instead of the maximum angle condition, the circumradius condition is sufficient to guarantee convergence. If the maximum circumradius of triangles tends to zero as the mesh is refined, convergence is guaranteed, even if maximum angles approach 180 degrees. This lightly relaxes the maximum angle condition, allowing certain cap elements but does not fundamentally change the situation.

Despite being vastly accepted, the maximum angle condition is not necessary for convergence. Hannukainen, Korotov and Křížek [Hannukainen et al. (2012)] provided a simple counterexample of a mesh that violates the maximum angle condition but for which the FEM solution converges. The idea is to take a regular mesh with all elements that satisfy the maximum angle condition and to divide one element in such a way that the new elements violate the maximum angle condition. The two families of meshes created this way are shown in figure 1.11. The family $\mathcal{F}_1$ satisfies the maximum angle condition and thus the FEM solution converges. The family $\mathcal{F}_2$ violates the maximum angle condition, but the FEM space of the first family is included in the FEM space of

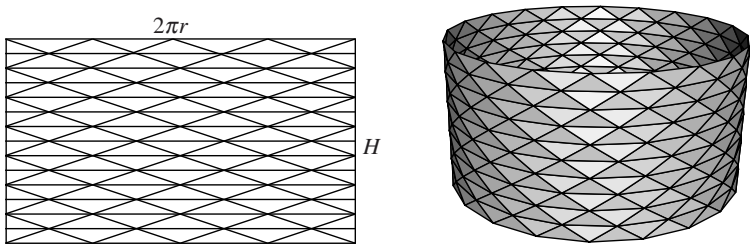


Figure 1.10: Schwarz lantern (right) and its unfolded mesh (left) that correspond to the FEM convergence counter-example of Babuška and Aziz. From [Kobayashi and Tsuchiya (2015)].

the second family. By applying Céa's lemma, the FEM will give us the best solution in the FEM space, thus the FEM solution of the second family will be at least as good as the FEM solution of the first family, which converges. Thus, the FEM solution of the second family also converges, even if it violates the maximum angle condition.

Recently, Václav Kučera [Kučera (2016)] worked on necessary and sufficient

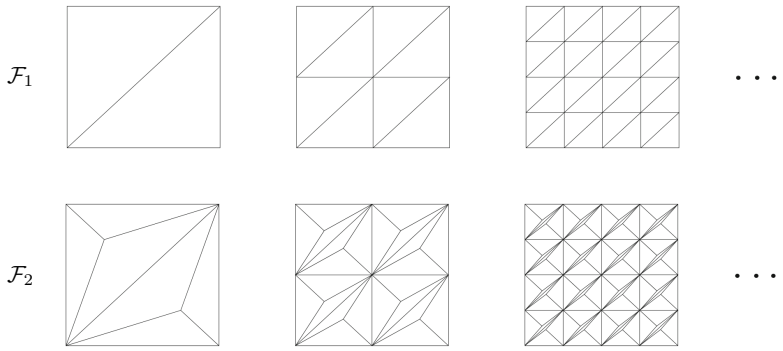


Figure 1.11: Two families of meshes from [Hannukainen et al. (2012)]: $\mathcal{F}_1$ satisfies the maximum angle condition, while $\mathcal{F}_2$ violates it.

conditions (plural) for convergence of the FEM solution. He developed a much more relaxed condition than the maximum angle condition, allowing the presence of cap elements in the mesh as long as they are isolated. Instead of constructing solution in the FEM space that is not the interpolation and that still converges, which can be complex, he used the idea of constructing the solution by taking the interpolation of a modified solution. He then proved that if the

elements that violate the maximum angle condition are isolated, then we can construct a modified solution that converges to the true solution. Figure 1.12 shows an example of a mesh that satisfies Kučera's condition, with isolated cap elements, where the region of the modified solution is shown. Kučera's condition state that cap elements are isolated if there is no nodes from an other cap element in the modified region of the solution.

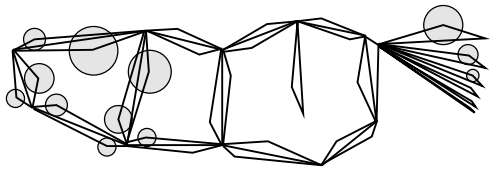


Figure 1.12: Example of a mesh taken from [Kučera (2016)] that satisfies Kučera's sufficient condition, with isolated cap elements and the region of the modified solution (circles).

On the other hand, Kučera also provided a necessary condition for convergence. He demonstrated that if there is a band of caps that are connected together in an alternating pattern, as shown in figure 1.13, then the FEM solution does not converge. This condition demonstrate and characterize the impact of the locking phenomenon on convergence. The locking phenomenon is a well-known issue in FEM, where the solution becomes overly stiff and does not converge to the true solution as the mesh is refined. This phenomenon can occur when there are too many cap elements in the mesh, leading to a loss of accuracy and convergence. Kučera's necessary condition provides a clear characterization of when this phenomenon occurs, allowing for better mesh design and quality control in FEM simulations.

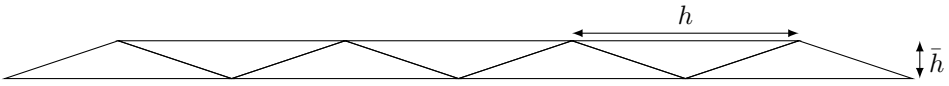


Figure 1.13: Example of a band of cap elements in an alternating pattern that violates Kučera's necessary condition for convergence if the thickness of the band $\bar{h}$ is smaller than $\mathcal{O}(h^2)$.

In Chapter 2, we introduce a new approach, the Tempered Finite Element Method (TFEM), to overcome this limitation.

1.4 A finite element solver for the incompressible Navier-Stokes equations

We consider the flow of two immiscible, incompressible and Newtonian fluids interacting together via an interface Γ. The evolution of the velocity and pressure of each fluid in time is described by the incompressible Navier-Stokes equations:

$$\begin{aligned} \rho(\partial_t \mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u}) &= -\nabla p + \mu\nabla^2\mathbf{u} + \mathbf{f} \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned} \tag{1.6}$$

where $\mathbf{u}$ is the velocity field $(u,v)^T$, p is the pressure, ρ is the density, $\mathbf{f}$ the forces at distance (gravity) and μ the dynamic viscosity.

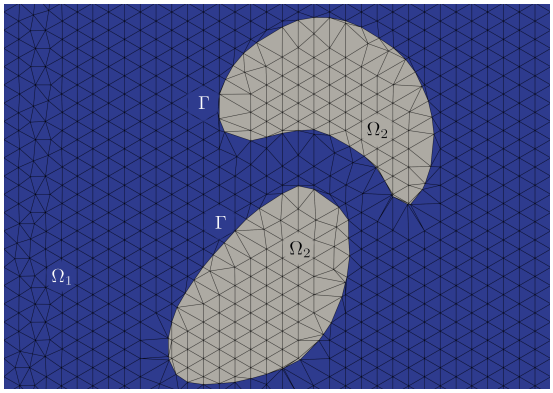


Figure 1.14: Two-phase flow sketch.

The two fluid phase domains are denoted Ω_1 and Ω_2 (see Figure 1.14) and have different dynamic viscosity and density (ρ_1,μ_1) and (ρ_2,μ_2), respectively. These material properties are thus discontinuous across the interface Γ. The Navier-Stokes equations are solved simultaneously on both subdomains and boundary conditions at the interface are necessary. The two equations 1.6 correspond to the conservation of momentum and the conservation of mass. These conservation laws must be respected at the interface between the two fluids. The conservation of mass leads to the condition of continuous velocity in the normal direction at the interface $\mathbf{n}$, thus preventing the transfer of mass between the two phases:

$$[\mathbf{u}]\cdot\mathbf{n} = \mathbf{0}$$

where the brackets $[\cdot]$ refer to the jump at the interface.
In addition to the condition of impermeability of the interface, the conservation

of the momentum in the perpendicular direction to the interface must be respected. To satisfy this condition, the jump in normal stress is counterbalanced by the surface tension

$$\left[-p\mathbf{I}+\mu\,\frac{1}{2}\left(\nabla\mathbf{u}+\nabla\mathbf{u}^T\right)\right]\cdot\mathbf{n}=\sigma\kappa\mathbf{n} \tag{1.7}$$

with $\mathbf{I}$ the identity operator, σ the surface tension coefficient and κ the curvature of the interface.

1.4.1 Galerkin finite element formulation

We first focus on solving the Navier-Stokes equations 1.6 for a single fluid, and we will later see how to take into account the presence of two fluids. We consider a domain Ω with boundary $\partial\Omega=\partial\Omega_D\cup\partial\Omega_N$, where $\partial\Omega_D$ is the part of the boundary where Dirichlet boundary conditions are applied and $\partial\Omega_N$ is the part of the boundary where Neumann boundary conditions are applied.
To solve the Navier-Stokes equations 1.6, we will use the Galerkin finite element method. We consider a 2D mesh $\mathcal{T}_h$ of the domain Ω and we define the finite element trial spaces $(\mathbf{S}^h_{\mathbf{u}},\mathbf{S}^h_p)$ and test spaces $(\mathbf{V}^h_{\mathbf{u}},\mathbf{V}^h_p)$ for the velocity and pressure:

$$\begin{aligned}
\mathbf{S}^h_{\mathbf{u}}&=\{\mathbf{u}_h\in[H^1(\Omega)]^2|\mathbf{u}_h=\mathbf{u}_D \text{ on } \partial\Omega_D\}\\
\mathbf{S}^h_p&=\{q_h\in L^2(\Omega)\}\\
\mathbf{V}^h_{\mathbf{u}}&=\{\mathbf{u}_h\in[H^1(\Omega)]^2\}\\
\mathbf{V}^h_p&=\{q_h\in L^2(\Omega)\}
\end{aligned}$$

Since the finite element method requires solving a linear system, we can consider an implicit time scheme without an extra-cost. An implicit scheme allows to use larger time step than explicit schemes since there is no numerical stability constraints such as the CFL. This allows us to use larger time steps, which is particularly useful for multiphase flows where the time step can be very limited by the CFL. We will use a backward Euler scheme for the time discretization. The discrete form of the Navier-Stokes equations is then: Find $(\mathbf{u}_h,p_h)\in\mathbf{S}^h_{\mathbf{u}}\times$

$\mathbf{S}_p^h$ such that for all $(\mathbf{v}_h, q_h) \in \mathbf{V}_\mathbf{u}^h \times \mathbf{V}_p^h$:

$$\int_\Omega \rho \left(\frac{\mathbf{u}_h^{n+1} - \mathbf{u}_h^n}{\Delta t} + \mathbf{u}_h^{n+1} \cdot \nabla \mathbf{u}_h^{n+1} \right) \cdot \mathbf{v}_h \, d\Omega +$$
$$\int_\Omega \mu \nabla \mathbf{u}_h^{n+1} : \nabla \mathbf{v}_h \, d\Omega - \int_\Omega p_h \nabla \cdot \mathbf{v}_h \, d\Omega = \int_\Omega \mathbf{f}^{n+1} \cdot \mathbf{v}_h \, d\Omega + \int_{\partial\Omega_N} \mathbf{t} \cdot \mathbf{v}_h d\Gamma$$
$$\int_\Omega q_h \nabla \cdot \mathbf{u}_h^{n+1} \, d\Omega = 0$$

There are two main difficulties when solving the Navier-Stokes equations with the finite element method. At high Reynolds numbers, the main numerical obstacle is discretizing the nonlinear, non-symmetric convection term in the momentum equation. In convection-dominated regimes, a standard Galerkin discretization typically generates spurious node-to-node oscillations. One can suppress these artifacts by aggressive mesh and time-step refinement, but the resulting cost and stiffness make this approach impractical.
A more effective strategy is to use a stabilized finite-element scheme, in this thesis we will use the Streamline-Upwind/Petrov–Galerkin (SUPG) [Brooks and Hughes (1982)]. Within the finite-element framework, SUPG emulates the upwinding used in finite-difference and finite-volume methods, damping the oscillations that arise with centered approximations. The trade-off is added artificial dissipation, which must be balanced carefully against accuracy. The SUPG stabilization consists in adding the following term to the weak form of the Navier-Stokes equations:

$$\text{SUPG} = \sum_{e=1}^{N_\text{el}} \int_{\Omega^e} \tau_\text{SUPG} \mathbf{u}_h \cdot \nabla \mathbf{v}_h \cdot \mathcal{R}_{n+1} \, d\Omega$$
$$\text{with } \mathcal{R}_{n+1} = \rho \left(\frac{\mathbf{u}_h^{n+1} - \mathbf{u}_h^n}{\Delta t} + \mathbf{u}_h^{n+1} \cdot \nabla \mathbf{u}_h^{n+1} \right) + \nabla p_h - \mu \nabla^2 \mathbf{u}_h^{n+1} - \mathbf{f}$$

where τ_SUPG is the stabilization parameter and N_el is the number of elements in the mesh.

The second difficulty is the choice of the finite element spaces for the velocity and pressure. The incompressibility constraint leads to a saddle-point problem, and the choice of the finite element spaces must satisfy the inf-sup condition [Babuska (1972/73)], also known as the Babuska-Brezzi compatibility condition. If this condition is not satisfied, the discrete problem may be unstable, leading to spurious pressure modes and poor convergence.
In order to avoid this issue, there are three main strategies: using mixed finite element spaces that satisfy the inf-sup condition, using stabilization techniques or using a fractional step method.

The mixed finite element spaces that satisfy the inf-sup condition have a higher representation for the velocity than the pressure, such as Taylor-Hood elements [Taylor and Hood (1973)] (P2-P1), the MINI element [Arnold et al. (1984)] (P1-bubble-P1) or the Crouzeix-Raviart element [Crouzeix, M. and Raviart, P.-A. (1973)] (P1-nonconforming-P0). The downside of this approach is that it complicates the implementation and increases the computational cost. Indeed, in a 2D triangular mesh there are approximately a 3 to 1 ratio between the number of edges and the number of nodes and a 2 to 1 ratio for the triangular faces versus the nodes. Thus, augmenting the polynomial degree of the velocity space increases significantly the number of degrees of freedom.

The fractional step approach, also known as the projection method [Chorin (1968)], consists in splitting the computation of the velocity and pressure into two steps. In the first step, an intermediate velocity field is computed without considering the incompressibility constraint. In the second step, the pressure is computed by solving a Poisson equation derived from the incompressibility constraint, and the velocity field is then corrected to satisfy the incompressibility constraint. The SIMPLE algorithm [Patankar and Spalding (1972)] is a well-known example of a fractional step method. It is shown in [Marchandise et al. (2006)] that the fractional step methods are equivalent to a stabilized mixed finite element method. The correction step acts as a Laplacian stabilization term for the pressure, for which the stabilization parameter is related to the time step.

In this thesis, we will use a stabilization technique to circumvent the inf-sup condition. The idea is to add a stabilization term to the weak form of the Navier-Stokes equations that penalizes the pressure oscillations. This way, we can use equal-order interpolation for the velocity and pressure, such as P1-P1 elements. Numerous stabilization techniques have been proposed: the pressure-stabilizing/Petrov-Galerkin (PSPG) method [Hughes et al. (1986)], the laplacian-pressure stabilization, the polynomial pressure projection method [Li and Wang (2004)], etc. In this thesis, we will use the PSPG method, which consists in adding the following term to the weak form of the Navier-Stokes equations:

$$\begin{aligned} \text{PSPG} =& \sum_{e=1}^{N_{\text{el}}} \int_{\Omega^e} \tau_{\text{PSPG}} \nabla q_h \cdot \mathcal{R}_{n+1} \, d\Omega \\ \text{with } \mathcal{R}_{n+1} =& \rho \left(\frac{\partial \mathbf{u}_h}{\partial t} + \mathbf{u}_h \cdot \nabla \mathbf{u}_h \right) + \nabla p_h - \mu \nabla^2 \mathbf{u}_h - \mathbf{f} \end{aligned}$$

where τ_{PSPG} is the stabilization parameter and N_{el} is the number of elements in the mesh.

1.4.2 Arbitrary Lagrangian-Eulerian formulation

For now we have only considered a fixed mesh but in the X-Mesh approach, the mesh moves in order to follow the interface between the two fluids. To take into account the movement of the mesh, we will use the Arbitrary Lagrangian-Eulerian (ALE) formulation [Hughes et al. (1981)]. The ALE formulation consists in introducing a mesh velocity $\mathbf{u}_{\text{mesh}}$ that represents the velocity of the mesh nodes. The material derivative is then modified to take into account the movement of the mesh:

$$\frac{D\mathbf{u}}{Dt} = \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} - \mathbf{u}_{\text{mesh}}) \cdot \nabla \mathbf{u}$$

When constructing an ALE formulation, it is important to ensure that the Geometric Conservation Law (GCL) [Farhat and Lesoinne (2000)] is satisfied. The GCL states that a linear field should be preserved when the mesh moves. This is particularly complex when using higher-order time integration schemes, as the GCL must be satisfied at each sub-step of the time integration. In our case we can consider a non-conservative implicit ALE scheme where we consider the integrals over the domain at the new time step Ω^{n+1} and the mesh velocity is computed as $\mathbf{u}_{\text{mesh}} = \frac{\mathbf{x}^{n+1} - \mathbf{x}^n}{\Delta t}$. This scheme is shown to satisfy the GCL in [Formaggia and Nobile (2004)].
We can now write the weak form of the Navier-Stokes equations in the ALE framework: Find $(\mathbf{u}_h, p_h) \in \mathbf{S}_{\mathbf{u}}^h \times \mathbf{S}_p^h$ such that for all $(\mathbf{v}_h, q_h) \in \mathbf{V}_{\mathbf{u}}^h \times \mathbf{V}_p^h$:

$$\begin{aligned}
\int_{\Omega^{n+1}} \rho \left(\frac{\mathbf{u}_h^{n+1} - \mathbf{u}_h^n}{\Delta t} + (\mathbf{u}_h^{n+1} - \mathbf{u}_{\text{mesh}}) \cdot \nabla \mathbf{u}_h^{n+1} \right) \cdot \mathbf{v}_h \, d\Omega + \\
\int_{\Omega^{n+1}} \mu \nabla \mathbf{u}_h^{n+1} : \nabla \mathbf{v}_h \, d\Omega - \int_{\Omega^{n+1}} p_h \nabla \cdot \mathbf{v}_h \, d\Omega \\
= \int_{\Omega^{n+1}} \mathbf{f}^{n+1} \cdot \mathbf{v}_h \, d\Omega + \int_{\partial \Omega_N} \mathbf{t}^{n+1} \cdot \mathbf{v}_h d\Gamma
\end{aligned}$$

$$\int_{\Omega^{n+1}} q_h \nabla \cdot \mathbf{u}_h^{n+1} \, d\Omega = 0$$

We can already observe that the equations are non-linear due to the advection term. We solve these equations with a Newton-Raphson scheme:

$$\begin{aligned}
\frac{\partial R(u_i^{n+1})}{\partial u} \Delta u &= R(u_i^{n+1}) \\
u_{i+1}^{n+1} &= u_i^{n+1} + \Delta u
\end{aligned}$$

where $R(u)$ correspond to the residual of our discretized Navier-Stokes equations.
Let's note that although the ALE formulation is derived here in a variational framework, it admits a natural interpretation in terms of moving control volumes, which is closer to the finite volume viewpoint. The ALE material derivative can be interpreted as the rate of change of a quantity within a control volume that moves with the mesh velocity. The choice of evaluating the integrals at the new time step Ω^{n+1} is referred to a non-conservative ALE scheme and can clash with the finite volume viewpoint where the integrals are evaluated at both time steps Ω^n and Ω^{n+1}. However, as mentioned earlier, this non-conservative ALE scheme satisfies the Geometric Conservation Law (GCL) [Formaggia and Nobile (2004)], which is crucial for preserving the accuracy of the solution when the mesh moves.
As a last ingredient to our FEM solver, we need to take into account the presence of both fluids, which is done in the next section.

1.4.3 Two-phase flow formulation

The presence of two fluids with different material properties introduces discontinuities in the density and viscosity across the interface. To take into account these discontinuities, we will use a one fluid approach [Marchandise and Remacle (2006)]. The idea is to consider the two fluids as a single fluid with variable density and viscosity that depend on the position of the interface. Due to surface tension effects, a jump in normal stress appears at the interface and must be taken into account.
Multiple methods exist to take into account the surface tension effects, such as the Continuum Surface Force (CSF) method [Brackbill et al. (1992)], the Ghost Fluid Method (GFM) [Fedkiw et al. (1999)] or the Immersed Interface Method (IIM) [LeVeque and Li (1997)]. In this thesis, we will use the GFM method, which consists in modifying the pressure gradient operator to take into account the jump in normal stress at the interface. When dealing with surface tension effects, it is important to ensure that the numerical method is well-balanced [Popinet (2018)]. A well-balanced method is a numerical method that preserves the equilibrium state of the system. In the case of two-phase flows, this means that the numerical method should be able to preserve a static circular droplet without introducing spurious currents. This property is more tricky to obtain than one might think and for many years, numerical methods were not able to preserve this simple equilibrium solution. The Ghost Fluid Method is shown to be a naturally well-balanced method in [Popinet (2018)]. When coupled with the X-Mesh approach, the GFM provides an accurate and sharply defined treatment of surface tension effects. This property will be discussed in more details in section 4.2.

1.5 Objective and methodology

The objective of this thesis is to develop and validate the X-Mesh method for the simulation of two-phase flows with sharp interfaces.
In order to achieve this objective, we have identified three main components that need to be developed and validated:

1) Developing robust 2D mesh deformation algorithms
The first component is to develop robust 2D mesh deformation algorithms that allow a fixed topology mesh to exactly match a given interface position at any time. This involves developing algorithms where we immerse an object in a mesh and we adjust only a few elements in order to match its boundary. A first version of this algorithm is presented in [Moës et al. (2023)] and we will build on it for the case of two-phase flows where a carefull treatment of the interface near the domain boundaries is necessary. These algorithms must be robust, efficient, able to handle complex geometries and avoid clusters of cap elements that could lead to a loss of accuracy.

2) Establishing a mathematical theory for zero-measure elements
The second component is to build a robust mathematical theory for zero-measure elements in finite element simulations. This involves studying the impact of degenerate elements on the convergence of the finite element method and developing strategies to mitigate any negative effects. In [Kučera (2016)], Kučera showed that one can create *visually unappealing* yet computationally acceptable meshes containing degenerate elements. In the first objective, we will develop algorithms that create meshes with isolated caps and that are thus computationally acceptable. However, in 3D, the situation is more complex and it is not clear if there exists a mesh deformation algorithms that avoid clusters of locking degenerate tetrahedra. We will thus develop a new approach called the Tempered Finite Element Method (TFEM) that allows to circumvent the locking phenomenon.

3) Applying the X-Mesh method to two-phase flows
The final step is to apply the X-Mesh method to the simulation of two-phase flows with sharp interfaces. This involves implementing the X-Mesh method in a finite element solver for the incompressible Navier-Stokes equations, and validating it against benchmark problems. We also aim to demonstrate the advantages of the X-Mesh method in particular for problems involving surface tension effects, where a sharp representation of the interface is crucial.

1.5.1 Outline of the thesis

The thesis is structured as follows:
In Chapter 1, we presented an introduction to the X-Mesh method along with an illustrative example. We also discussed the challenges associated with mesh quality in the finite element method, highlighting key developments and open questions in the field. Finally, we presented the finite element formulation for solving the incompressible Navier-Stokes equations on a moving mesh we will use throughout this thesis.
In Chapter 2, we will present the Tempered Finite Element Method (TFEM) from a mathematical point of view as well as numerical experiments in 2D and 3D that validate the method. Those numerical experiments will show that TFEM allows to circumvent the necessary condition for convergence and we will address some *extensions* of the method: linear elasticity, mesh mortaring, higher-order elements and advection.
In Chapter 3, we will present robust 2D mesh deformation algorithms that allow to deform a fixed topology mesh to exactly match a given interface position at any time. These algorithms will use two different approaches to represent the interface: a levelset approach and a tracking of marker points approach.
Chapter 4 will present the X-Mesh method for the simulation of two-phase flows with sharp interfaces. We will first discuss the surface tension implementation using the Ghost Fluid Method in the context of X-Mesh and we will then present numerical experiments that validate the method against benchmark problems.

1.6 Publications and conferences

During the course of this thesis, several publications and conference presentations were made to disseminate the research findings. Below is a list of these contributions:

Journal Publications:

Antoine Quiriny, Jonathan Lambrechts, Nicolas Moës and Jean-François Remacle, X-Mesh: A new approach for the simulation of two-phase flow with sharp interface,
Journal of Computational Physics, Volume 501, 2024, 112775, ISSN 0021-9991.

Antoine Quiriny, Václav Kučera, Jonathan Lambrechts, Nicolas Moës and Jean-François Remacle,
The tempered finite element method, 2025, Journal of Computational Physics 549, 114615.

Conference Presentations:

Antoine Quiriny, Jonathan Lambrechts, Nicolas Moës and Jean-François Remacle, The X-Mesh approach applied to fluid-structure interactions.
COUPLED2025, XI International Conference on Coupled Problems in Science and Engineering, 25-28 May 2025, Villasimius, Italy

Antoine Quiriny, Jonathan Lambrechts, Nicolas Moës and Jean-François Remacle, Two-phase flow simulation using the X-Mesh approach.
ECCOMAS2024, 9th European Congress on Computational Methods in Applied Sciences and Engineering, 3-7 June 2024, Lisboa, Portugal

Antoine Quiriny, Jonathan Lambrechts, Václav Kučera, Nicolas Moës and Jean-François Remacle,
X-Mesh: La méthode des éléments finis avec des éléments dégénérés.
CSMA2024, 16eme Colloque National en Calcul des Structures, 13-17 May 2024, Giens, France

Antoine Quiriny, Jonathan Lambrechts, Nicolas Moës and Jean-François Remacle, A new approach for two-phase flows simulation: X-Mesh.
CFC2023, 22nd Computational Fluids Conference, 25-28 April 2023, Cannes, France

Antoine Quiriny, Jonathan Lambrechts, Nicolas Moës and Jean-François Remacle, Multiphase flows: The X-Mesh method.

ACOMEN2022, 8cmth International Conference on Advanced COmputational Methods in ENgineering and Applied Mathematics 31 August-2 September 2022, Liège, Belgium

Antoine Quiriny, Jonathan Lambrechts, Nicolas Moës and Jean-François Remacle, The X-Mesh method applied to Multiphase Flows.
ECCOMAS2022, 8th European Congress on Computational Methods in Applied Sciences and Engineering, 5-9 June 2022, Oslo, Norway

Antoine Quiriny, Jonathan Lambrechts, Nicolas Moës and Jean-François Remacle, X-Mesh appliquée aux écoulements diphasiques.
CSMA2022, 15eme Colloque National en Calcul des Structures, 16-20 May 2022, Giens, France

CHAPTER 2

The tempered finite element method for degenerated elements

Contents

2.1 Finite elements and meshes: A complex relationship 31

2.2 A simple numerical experiment 34

2.2.1 Finite element formulation of the 2D Poisson problem 35

2.2.2 Tempered finite element on a degenerate mesh . . 36

2.2.3 Convergence 39

2.2.4 Error sensitivity and choice of $J_{\min}$ 42

2.2.5 Size effect related to the degenerate band/surface 42

2.2.6 Conditioning 44

2.3 Theory behind the TFEM scheme 47

2.3.1 TFEM for a band of arbitrarily small but non-zero caps 48

2.3.2 Zero-measure elements – TFEM and Mortaring . 55

2.4 Extensions 59

2.4.1 Extension to linear elasticity 59

2.4.2 Mesh mortaring 61

2.4.3 High order elements 64

2.4.4 Advection 65
2.5 Conclusion 69
2.5.1 Discussion on $J_{\min}$ and dimensional analysis . . . 70

This chapter corresponds to the article *The tempered finite element method* by A. Quiriny, V. Kučera, J. Lambrechts, N. Moës and J-F. Remacle [Quiriny et al. (2026)]. A discussion was added at the end of the chapter to present some perspectives for the method.

2.1 Finite elements and meshes: A complex relationship

Since the early 1970s, the finite element method has been used extensively in computational physics and engineering. The fact that there exists a structure of proof for finite element algorithms [Ern and Guermond (2004)] supplemented by the fact that finite elements are able to solve non-linear problems [Wriggers (2008)] on possibly complex geometries using unstructured meshes [Geuzaine and Remacle (2009)] makes finite elements the method that prevails in most branches of manufacturing industry – aerospace, automotive, shipbuilding, electrical machines, ...

Engineering designs are encapsulated in CAD systems. By extension, the word CAD also designates the geometry of the object/system to be analyzed. The analysis process begins with the CAD geometry. Then, the currently prominent method of analysis, finite elements, requires a mesh, which is an alternative (discrete) representation of a geometry. Finite elements (unlike certain classes of finite volumes) are relatively robust when it comes to mesh quality. In the previous sentence, we have deliberately used the word *relatively*. Finite elements are resilient to the poor quality of some meshes, but not all. There exist some specific mesh configurations that make the finite element method become *brittle* i.e. that prevent finite element convergence. In a classical paper [Babuška and Aziz (1976)], Babuška and Aziz claim that *what is essential is the fact that no angle is close to* 180^o. Once again, words are everything: Babuška and Aziz use the word *essential* but not the word *necessary*, which has a more precise mathematical definition. Much later, Hannukainen et al. [Hannukainen et al. (2012)] showed through a very simple geometrical construction presented in Figure 1.11 that the maximum angle condition is not necessary. In [Hannukainen et al. (2012)], the authors do not specify whether there are mesh configurations that render the finite element method inoperative. In [Babuška and Aziz (1976)], the authors already show that degenerate triangles with two angles close to 90^o – which we'll call needles in the following – don't cause any problems. Kučera shows in [Kučera (2016)] that a triangle with an angle close to 180^o – which we'll call a cap in the following – poses no problem *when isolated*. We can explain using basic linear algebra why isolated caps (or needles) causes no problem in the finite element method and why only specific patterns should be avoided. We start by studying the stiffness matrix

related to the Laplace operator on a triangle (see Figure 2.1):

$$K_{ij} = \int_T \nabla\phi_i \cdot \nabla\phi_j \, dx \quad , \quad i,j = 1,2,3.$$

Assume three scalar values u_i, $i = 1,2,3$, associated to nodes located at x_i, $i = 1,2,3$. The stiffness matrix, $[K]$, allows one to write the quadratic expression of the energy:

$$f(u_1,u_2,u_3) = \frac{1}{2}\sum_{i=1}^{3}\sum_{j=1}^{3} K_{ij}u_iu_j. \tag{2.1}$$

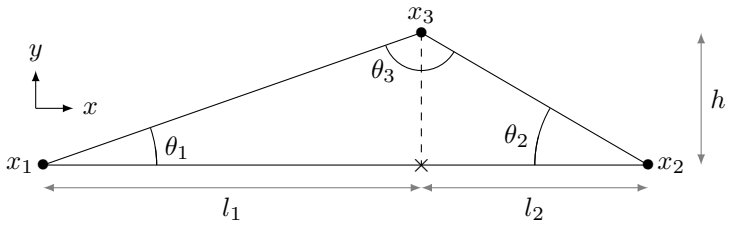


Figure 2.1: A triangle T. The angle θ_i is associated to the vertex x_i.

Without loss of generality, we assume that $\theta_3 \geq \theta_2 \geq \theta_1$. Edge (x_1,x_2) is thus the longest edge of T (see Figure 2.1) with $\|x_2 - x_1\| = l_1 + l_2$. Defining the element flatness f and a symmetry parameter s,

$$f = \frac{h}{l_1+l_2}, \quad s = \frac{l_1}{l_1+l_2}, \quad f \in \left[0, \sqrt{3}/2\right], \quad s \in [0,1]. \tag{2.2}$$

This local matrix has three eigenvalues λ_1, λ_2 and λ_3 and three corresponding eigenvectors $\mathbf{v}_1, \mathbf{v}_2$ and $\mathbf{v}_3$ that relate to different modes of the solution. The computation of the eigenvalues, eigenvectors and energies associated are described in appendix A.1. The first mode is the rigid mode with zero energy. We are interested in the limit case $f \to 0$ for which the element reaches zero measure. It is possible to show that as $f \to 0$, we have

$$\lambda_2 \sim \frac{3}{2}\frac{f}{1-s+s^2} \quad \text{and} \quad \lambda_3 \sim 2\frac{1-s+s^2}{f} \tag{2.3}$$

As $f \to 0$, the energy of the second mode $\mathbf{v}_2$ goes to zero linearly in f and the energy of the third mode $\mathbf{v}_3$ goes to infinity as $1/f$. The energy and therefore the gradients of $u(x,y)$ tend towards infinity if the $\mathbf{v}_3$ mode corresponding to λ_3 is active. It is possible to show that

$$\lim_{f\to 0} \mathbf{v}_2 = (1-s, s, -1)$$

and $\mathbf{v}_2$ is always orthogonal to $\mathbf{v}_3$. Therefore, any u with finite energy can be written as linear combinations of $\mathbf{v}_1$ and $\mathbf{v}_2$:

$$u_3 = (1-s)u_1 + su_2. \tag{2.4}$$

In a degenerated element, values of u_1 and u_2 can be chosen arbitrarily but u_3 **must be the linear interpolation between u_1 and u_2 on edge x_1x_2.**

In the limit $f \to 0$, any u field that does not satisfy (2.4) has an infinite energy. When solving for the global finite element system $Ax = b$, there exists in the limit $f \to 0$ an infinite eigenvalue in A corresponding to an eigenmode $\mathbf{x}_\infty$. Since the solver minimizes the energy, it will choose a solution $\mathbf{x}$ that does not contain $\mathbf{x}_\infty$. The degenerate cap will simply act as a mortar between its three neighbors, imposing condition (2.4).

Now what if we consider a band of caps like the one in Figure 2.2. At the $\bar{h} \to 0$ limit, the solution u_2 at point x_2 is constrained by the solution at points x_1 and x_3. The solution at x_3 is itself constrained by the solution at x_2 and x_4, and so on. So the u solution is forced to be linear along the band. This locking phenomenon leads to the fact that h is no longer the mesh size in the band but L is. And, if L is held fixed, convergence is lost. By making such an argument more rigorous, Kučera [Kučera (2016)] was able to demonstrate that for a band of caps of fixed length L, classical finite elements have $O(h)$-convergence if and only if $\bar{h} \geq Ch^2$ is satisfied, where C a constant independent of the mesh (cf. also Remark 1). Finite elements have in fact a greater tolerance to mesh "quality" than expected.

Note that for a needle we have either $s = 0$ or $s = 1$. The condition for finite energy forces both nodes of the collapsed edge to have the same value. In a needle, the acceptable mode $\mathbf{v}_2$ couples only two nodes, and a band of needles causes no concern.

In the paper [Kučera (2016)], Kučera showed that very strange triangulations allow finite element convergence, stating that *such meshes perhaps do not have any value from the practical point of view.* Recently, we developed a new interface tracking method, called X–Mesh [Moës et al. (2023); Quiriny et al. (2024)], which takes advantage of meshes with zero or close to zero-measure elements. The principle is simple: track the interface by mesh relaying. The interface is handed over from layers of nodes to new layers of nodes across elements which can be rather thin in the direction of the front propagation. These elements may even possibly form bands of caps.

This is the motivation for this chapter: be able from an X–Mesh perspective to compute easily with bands of caps. This requires a more robust finite element approach that we have named the *tempered finite element method.* This name is inspired by an analogy with metallurgy. The problem with finite elements on a band of caps is that the numerical solution undergoes a locking phenomenon which prevents proper functioning of the method and convergence. In a material-science analogy, after discretization the 'material'

becomes extremely hard on a band of caps and the numerical solution can no longer conform to the exact solution. The method we propose alleviates this by locally 'softening' the material on the band. In Poisson's problem, this will correspond to locally decreasing the diffusion parameter in a mesh-dependent way. This allows the numerical solution to conform to the continuous problem. This process is analogous to that of tempering in metallurgy. Steel can be made very hard by heating it to a high temperature and then rapidly cooling it in water or oil. This process is called *quenching* and it makes the steel extremely hard, but also very brittle. *Tempering* is then performed, when the steel is slowly reheated to an intermediate temperature and slowly cooled, allowing the internal stresses of the material to be relieved. This makes the alloy more flexible and ductile, while preserving most of the hardness. This is analogous to our approach in the FEM, where we decrease the diffusion coefficient ('soften') to increase flexibility, while preserving other desired qualities (e.g. good approximation properties). For this reason, we call the method **tempered finite element method (TFEM)**.

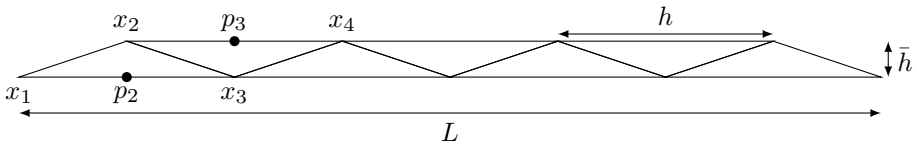


Figure 2.2: Band of caps triangles in 2D.

This chapter is structured as follows. First we perform a simple numerical experiment on a manufactured Laplacian to show that, using a simple numerical trick, we are able to recover finite element convergence. The TFEM solution on a mesh containing a band of caps is compared with the finite element solution on a regular mesh. Section 3 is dedicated to a rigorous mathematical analysis, confirming the numerical experiment. We demonstrate that the TFEM convergence has the same rate as the one of the traditional finite element method on regular meshes. In the next section, we present extensions to the TFEM method and its usefulness in various situations: linear elasticity, non-conformal mesh mortaring, high-order elements, and advection.

2.2 A simple numerical experiment

In this section, we solve a classical finite element problem on a series of meshes exhibiting a band of degenerate caps and observe that a simple *numerical trick* is enough to avoid the locking behavior.

2.2.1 Finite element formulation of the 2D Poisson problem

Let $\Omega \subset \mathbb{R}^2$ be a planar domain with its boundary $\partial\Omega$. We seek u, the solution of:

$$\begin{aligned} -\Delta u &= f \qquad \text{on } \Omega, \\ u &= u_D \qquad \text{on } \partial\Omega. \end{aligned} \tag{2.5}$$

Defining the space $V = \{u \in H^1(\Omega)\ ,\ u = u_D \text{ on } \partial\Omega\}$, the corresponding weak form is to find $u \in V$, such that

$$\int_\Omega \nabla u \cdot \nabla v \,\mathrm{d}x = \int_\Omega f v \,\mathrm{d}x, \qquad \forall v \in V^0, \tag{2.6}$$

where $V^0 = H_0^1(\Omega) = \{v \in H^1(\Omega)\ ,\ v = 0 \text{ on } \partial\Omega\}$. This problem is solved on a mesh denoted $\mathcal{T}_h$ composed of triangular elements of size h. The finite element method constructs a space $X_h \subset H^1(\Omega)$ of continuous and piecewise linear functions over the mesh along with the space $V_h = \{v_h \in X_h\ ,\ v_h = \tilde{u}_D \text{ on } \partial\Omega\}$, where $\tilde{u}_D$ is a piecewise linear interpolation of u_D on $\partial\Omega$. By defining $V_h^0 = X_h \cap V^0$, we obtain the discrete version of (2.6).

Find $u_h \in V_h$ such that

$$\int_\Omega \nabla u_h \cdot \nabla v_h \,\mathrm{d}x = \int_\Omega f v_h \,\mathrm{d}x, \qquad \forall v_h \in V_h^0. \tag{2.7}$$

Following the classical finite element approach, the matrix is obtained as an assembly of local matrices K_{ij} evaluated on a single element E, i and j being two nodes of the element E:

$$K_{ij} = \int_E \nabla\phi_i \cdot \nabla\phi_j \mathrm{d}x.$$

A usual way of obtaining the gradients of the shape functions and integrating them is via a parent element. The coordinates of our element E are expressed in $\mathbf{x} = [x, y]$ while for the parent element they are expressed in $\boldsymbol{\xi} = [\xi, \eta]$. The mapping between both coordinate systems is represented in Figure 2.3.
The gradient of ϕ in the element E can then be obtained via the chain rule $\frac{\partial\phi}{\partial\mathbf{x}} = \frac{\partial\phi}{\partial\boldsymbol{\xi}}\frac{\partial\boldsymbol{\xi}}{\partial\mathbf{x}}$. The calculation of the gradient of the transform $\frac{\partial\boldsymbol{\xi}}{\partial\mathbf{x}}$ is given by:

$$\frac{\partial\boldsymbol{\xi}}{\partial\boldsymbol{x}} = \left(\frac{\partial\boldsymbol{x}}{\partial\boldsymbol{\xi}}\right)^{-1} = \frac{1}{J}\begin{bmatrix} \frac{\partial y}{\partial\eta} & -\frac{\partial x}{\partial\eta} \\ -\frac{\partial y}{\partial\xi} & \frac{\partial x}{\partial\xi} \end{bmatrix}$$

where $J = \frac{\partial x}{\partial\xi}\frac{\partial y}{\partial\eta} - \frac{\partial x}{\partial\eta}\frac{\partial y}{\partial\xi}$ is the mapping determinant. For a triangular element, J is constant and equals to two times the area of the element. J also appears

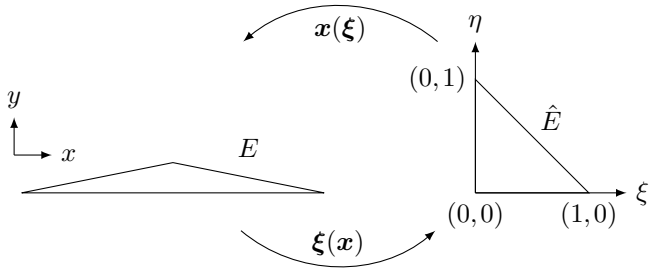


Figure 2.3: Mapping between the physical and reference domains.

in the elementary surface of the integral $\mathrm{d}x = J\mathrm{d}\hat{x}$. Finally, the local matrix K_{ij} can be expressed as:

$$K_{ij} = \int_{\hat{E}} \frac{1}{J} \begin{bmatrix} \frac{\partial \phi_i}{\partial \xi} & \frac{\partial \phi_i}{\partial \eta} \end{bmatrix} \begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial x}{\partial \eta} \\ -\frac{\partial y}{\partial \xi} & \frac{\partial x}{\partial \xi} \end{bmatrix} \begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial y}{\partial \xi} \\ -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} \end{bmatrix} \begin{bmatrix} \frac{\partial \phi_j}{\partial \xi} \\ \frac{\partial \phi_j}{\partial \eta} \end{bmatrix} \mathrm{d}\hat{x}. \tag{2.8}$$

2.2.2 Tempered finite element on a degenerate mesh

The rightmost mesh of Figure 2.4 looks non-conforming with T-junctions. This is in fact not the case. The mesh has been progressively deformed, starting from Figure 2.4 (center), to bring two sub-meshes into contact by "crushing" the intermediate *band of caps* denoted $\mathcal{B}$. As the band reaches a zero thickness, the finite element is forced to be linear (or constant) along the band – this is precisely the locking phenomenon described in the Introduction. In practice however, it is not possible to assemble the stiffness matrix (2.8) for a zero thickness band since it results in a division by zero. A handy way to avoid this division is to modify $K \to \tilde{K}$ by bounding the value of J away from zero with some value denoted $J_{\min}$. Equation (2.8) becomes:

$$\tilde{K}_{ij} = \int_{\hat{E}} \frac{1}{\max(J, J_{\min})} \begin{bmatrix} \frac{\partial \phi_i}{\partial \xi} & \frac{\partial \phi_i}{\partial \eta} \end{bmatrix} \begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial x}{\partial \eta} \\ -\frac{\partial y}{\partial \xi} & \frac{\partial x}{\partial \xi} \end{bmatrix} \begin{bmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial y}{\partial \xi} \\ -\frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} \end{bmatrix} \begin{bmatrix} \frac{\partial \phi_j}{\partial \xi} \\ \frac{\partial \phi_j}{\partial \eta} \end{bmatrix} \mathrm{d}\hat{x}. \tag{2.9}$$

In the rest of this chapter we will note $\tilde{u}_h$ the solution obtained using this modified matrix. Choosing the smallest possible nonzero value for $J_{\min}$ allows one to achieve optimum finite element convergence for isolated caps or needles and even for bands of needles.

In the case of the band of caps, the concern is that we are not interested in finding the zero-measure limit solution because of locking. The question is then: is there a value for $J_{\min}$, possibly dependent on h, which is small enough to allow optimal convergence but large enough to avoid locking? The answer

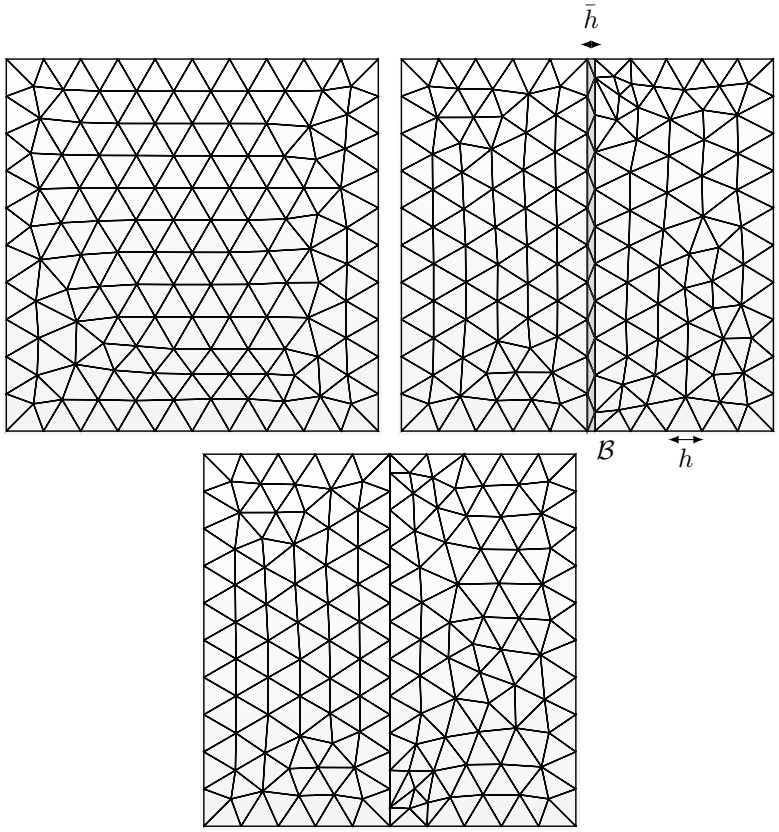


Figure 2.4: Three 2D meshes: regular (top left), with a band of caps (top right) and with a fully degenerated band of caps (bottom).

is yes, and that's what this chapter is all about. We shall call the resulting numerical method corresponding to (2.9) the *Tempered Finite Element Method* (TFEM).

In the case when the band elements are not exactly degenerate ($J > 0$), we can express the modified local matrix from equation (2.9) as

$$\tilde{K}_{ij} = \frac{J}{\max(J, J_{\min})} K_{ij}. \tag{2.10}$$

We call this coefficient $d(x) = \frac{J}{\max(J, J_{\min})}$. This corresponds to modifying the basic FEM scheme (2.7) by introducing a variable diffusion parameter $0 <$

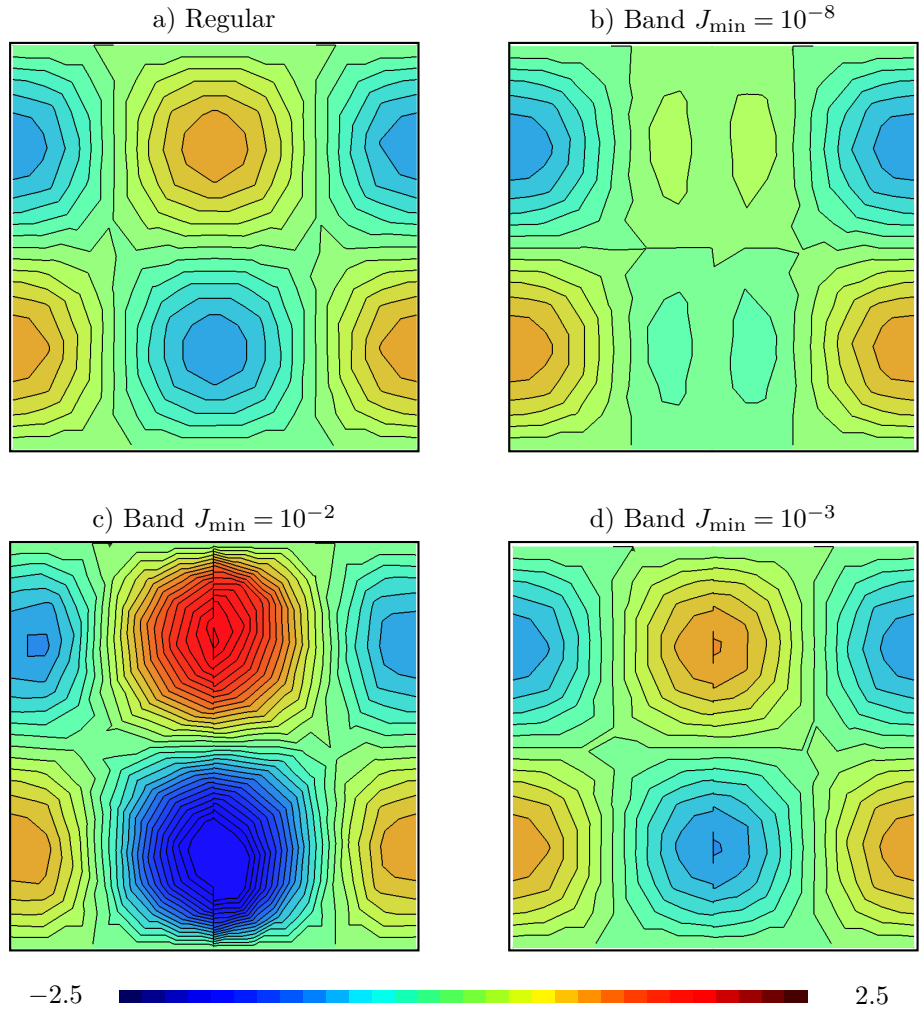


Figure 2.5: Solution of the manufactured solution on the regular mesh and on a mesh with a degenerated band with $h = \frac{1}{10}$ for different values of $J_{\min}$.

$d(x) \leq 1$ which equals a carefully chosen value $0 < D = \frac{J}{J_{\min}} < 1$ on $\mathcal{B}$. Define

$$d(x) = \begin{cases} 1, & x \in \Omega \setminus \mathcal{B}, \\ D, & x \in \mathcal{B}. \end{cases} \tag{2.11}$$

The Tempered FEM problem can then be reformulated as finding $\tilde{u}_h \in V_h$ such

that

$$\int_\Omega d(x)\nabla\tilde{u}_h\cdot\nabla v_h\,\mathrm{d}x = (f, v_h), \quad \forall v_h \in V_h^0. \tag{2.12}$$

Equation (2.12) motivates the name 'tempered' FEM. We effectively lower the diffusion coefficient locally to recover good approximation properties of the original FEM. As explained in more detail in the Introduction, we view this as local 'softening' of the material which we regard as analogous to *tempering* in metallurgy.

In the case when the elements in the band $\mathcal{B}$ are exactly degenerate ($J = 0$), deriving a formulation similar to (2.12) is not as straightforward. One must consider the limit of (2.12) as $\bar{h} \to 0$. This is done in detail in Section 2.3.2. Surprisingly, the resulting scheme (2.43) is a penalization (or mortaring) method.

What remains is to determine an appropriate value of $J_{\min}$. We start with a numerical experiment. Let's consider two manufactured problems, one in 2D and the other one in 3D. The domains Ω are the unit square in 2D and the unit cube in 3D. The exact solutions are respectively $u = \sin(2\pi x)\cos(2\pi y)$ in 2D and $u = \cos(2\pi x)\sin(2\pi y)\cos(2\pi z)$ in 3D. The source term in (2.5) is $f = 8\pi^2\sin(2\pi x)\cos(2\pi y)$ in 2D and $f = 12\pi^2\cos(2\pi x)\sin(2\pi y)\cos(2\pi z)$ in 3D.

As illustrated in Figure 2.5, a small $J_{\min} = 10^{-8}$ which would have worked perfectly for isolated caps or needles leads to locking. On the other hand, a large $J_{\min} = 10^{-2}$ leads to a band that is, say, "too elastic" and solution jumps of too large a magnitude are observed across the band. However, an appropriate choice of $J_{\min} = 10^{-3}$ leads to a solution that is visually close to the reference solution on a standard mesh. The solution with $J_{\min} = 10^{-3}$ still contains jumps but these are limited.

2.2.3 Convergence

An appropriate choice of $J_{\min}$ seems to provide a qualitatively acceptable solution. How does it compare to the solution on a regular mesh? Is it possible to choose $J_{\min}$ so that the convergence rate is the same as the one of the classical finite elements? To answer these questions, we compare the L^2 error and the H^1 error outside the band on two series of meshes: a series of regular successively refined meshes with different element size h and a series of meshes with the same mesh size and a band of cap in the middle. We refine the mesh and observe how $J_{\min}$ must vary in order to make the error converge. We assume that $J_{\min}$ evolves as a power of h such that :

$$J_{\min} = Ch^k,$$

where C is a parameter that will be analyzed later and k a coefficient that must be adequately chosen. Since we have observed that the choice of $J_{\min} = 10^{-3}$

was judicious for a mesh of size $h = \frac{1}{10}$, we choose $C = 10^{k-3}$ so that we get this value for any choice of k. We can see in Figure 2.6 (left) that the curves (except the reference) starts from the same point for this reason. The 2D and 3D convergence studies for the $L^2(\Omega)$ and $H^1(\Omega \setminus \mathcal{B})$ errors are shown in Figures 2.6 and 2.7. These error were computed by comparing to the analytical solution and integrated by a 4th order quadrature rule. Different choices of k are studied. It can be seen that in the 2D case, the choice of $k = 3$ gives the optimal rate of convergence of the error in the L^2 norm. For the H^1 semi-norm, the optimal choice of k lies in the range $[2, 4]$.

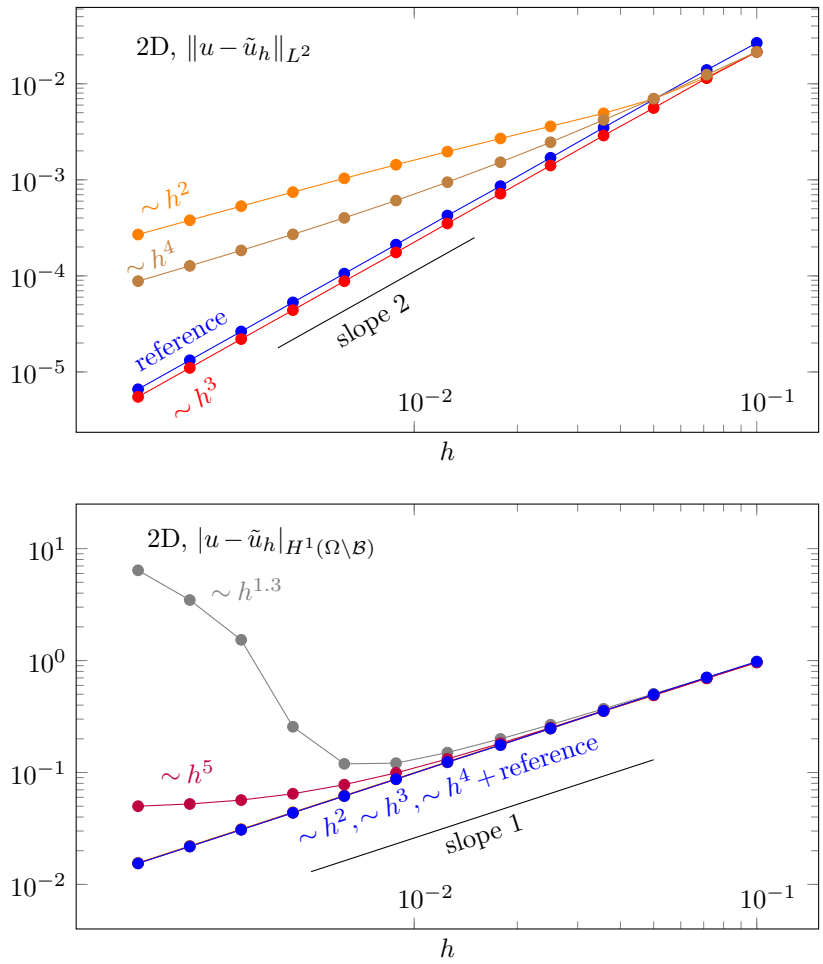


Figure 2.6: Convergence of the error in the L^2 norm (left) and H^1 semi-norm (right) in 2D.

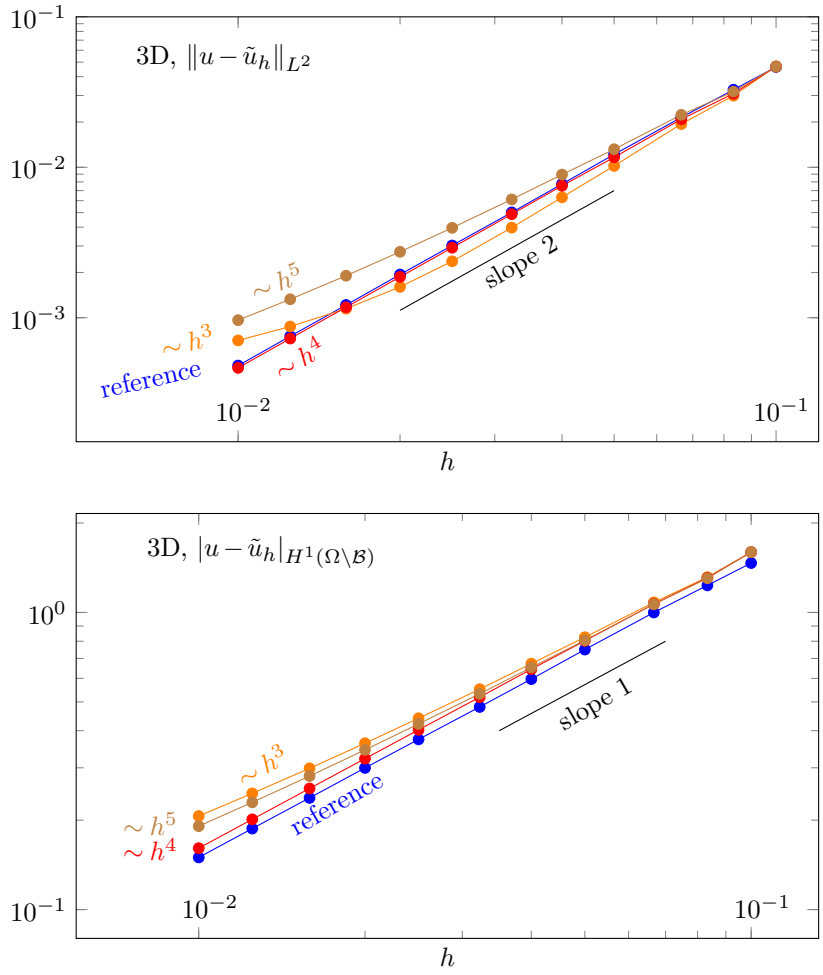


Figure 2.7: Convergence of the error in the L^2 norm (left) and H^1 semi-norm (right) in 3D.

In 3D, where an analogue of a band of caps is a thin slab or 'surface' of degenerate tetrahedra (spades and slivers in our case), the proper rate of convergence is obtained for the L^2 and H^1 error with the choice of $k = 4$. The choice of a higher exponent doesn't bring a too large error but will eventually create a locking effect of the solution as h is reduced. It may come as a surprise that the error of the TFEM method is lower than that of the classical FEM on a regular mesh, while the latter gives the best approximation of the solution

in the discrete space V_h. This is because it is the best approximation in the energy norm of the problem, which is in our case the H^1 semi-norm. In the H^1 semi-norm, the finite element error on a regular mesh is lower, but this is not the case for the L^2 norm as we can see in the Figure 2.6. While the finite element method minimizes the error for the H^1 semi-norm, the tempered FEM cannot offer a better solution than the classical FEM, as observed in Figures 2.6 and 2.7. Nevertheless, in 2D, the errors are almost identical, and in 3D, the TFEM errors are quite close to those of the classical FEM.

2.2.4 Error sensitivity and choice of $J_{\min}$

To assess the sensitivity of the solution to the choice of $J_{\min}$, we have conducted the same test case across a broad spectrum of C values using the formula $J_{\min} = Ch^{d+1}$, $d = 2, 3$ being the problem dimension. The H^1 error is less sensitive than the L^2 error, as illustrated in Figure 2.9 and 2.8. This explains why, in the convergence graphs (Figures 2.6 and 2.7), the convergence rate deteriorates sooner for the L^2 error compared to the H^1 error. The two dashed lines represent the error for the FEM on a regular mesh. For a specific range of C values, the L^2 error of the TFEM is lower than that for the FEM. This range, spanning almost an order of magnitude in 3D, indicates a degree of robustness in the method regarding the choice of C.
The results in 2D and 3D are similar, with the notable exception that the H^1 error is even less sensitive to the choice of $J_{\min}$ as shown in Figure 2.8. Indeed, the error measured for the gradients exhibits a plateau, remaining nearly constant over a wide range of C values. As the mesh is refined, this plateau seems to grow at a rate of $\sim h^{-1}$, explaining the range of exponent values for h that allow gradient convergence. Choosing $J_{\min} = Ch^3$ with a constant C is effectively equivalent to choosing $J_{\min} = Ch^2$ with a C that varies as $\sim h$. The dotted line in Figure 2.8 illustrates this scenario, showing it remains well within the plateau, ensuring the error converges at the expected rate.

2.2.5 Size effect related to the degenerate band/surface

To verify the dependence on L, the length of the band of caps as represented in Figure 2.2, we employed a manufactured solution with several periods represented within the domain. By selecting lengths L that are multiples of this period, we tested the dependence of the optimal choice of C and the error on L without altering the function represented on the band of caps.

This strategy allows one to discriminate effects related to the first and second derivatives of the solution.

As we can see on the Figure 2.10, the length of the band in 2D does not influence the value of the optimal C for both the L^2 and H^1 errors. This

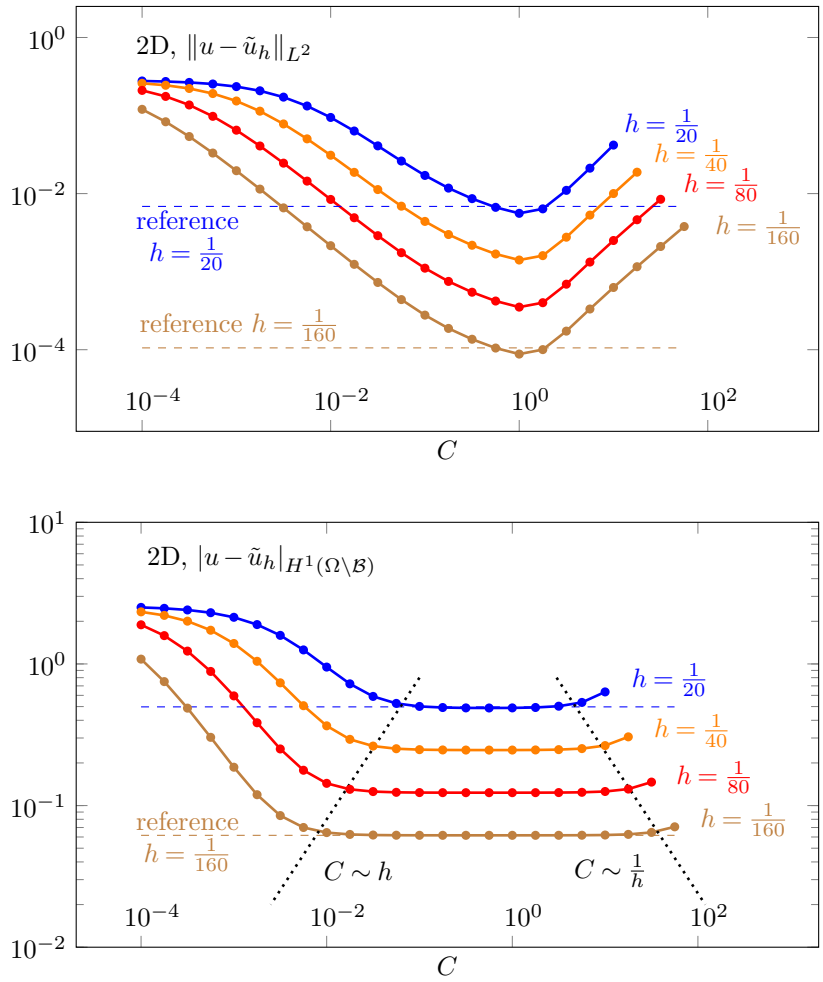


Figure 2.8: Effect of C on the error in the L^2 norm (left) and the H^1 semi-norm (right) in 2D.

is in agreement with Theorem 6 of section 2.3. Additionally, if C is chosen optimally, the error doesn't vary much with the band length. When C is chosen too small and the solution presents locking, we observe that the error varies as $\sim \sqrt{L}$. Moreover, a shorter band results in a flatter profile of the curves around the minimum of the curve, indicating a plateau. Again, this plateau provides greater robustness in the choice of C, allowing for a wider range of acceptable values.

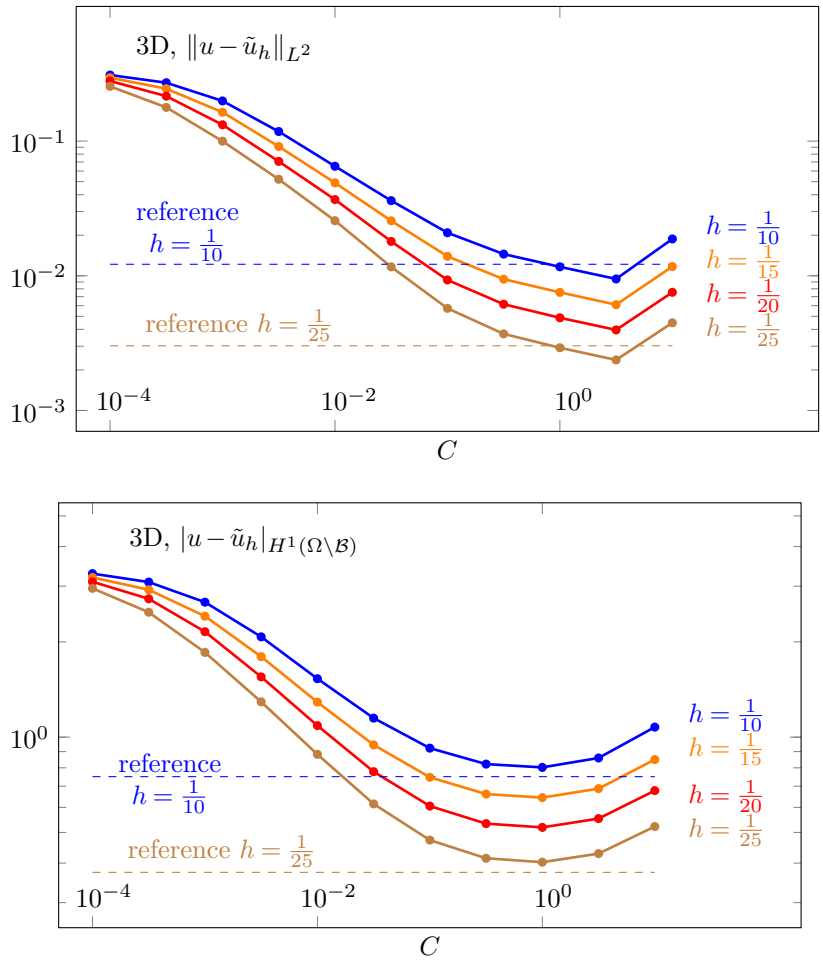


Figure 2.9: Effect of C on the error in the L^2 norm (left) and the H^1 semi-norm (right) in 3D.

In the 3D case, the area S of the degenerate surface affects the choice of the optimal C, as illustrated in Figure 2.11. However, when C is chosen optimally, the error exhibits minimal variation with respect to S.

2.2.6 Conditioning

It is perfectly legitimate to question the possibility of a deterioration in the performance of linear solvers when introducing elements of quasi-zero measure, which necessarily lead to the appearance of very large eigenvalues in K. More

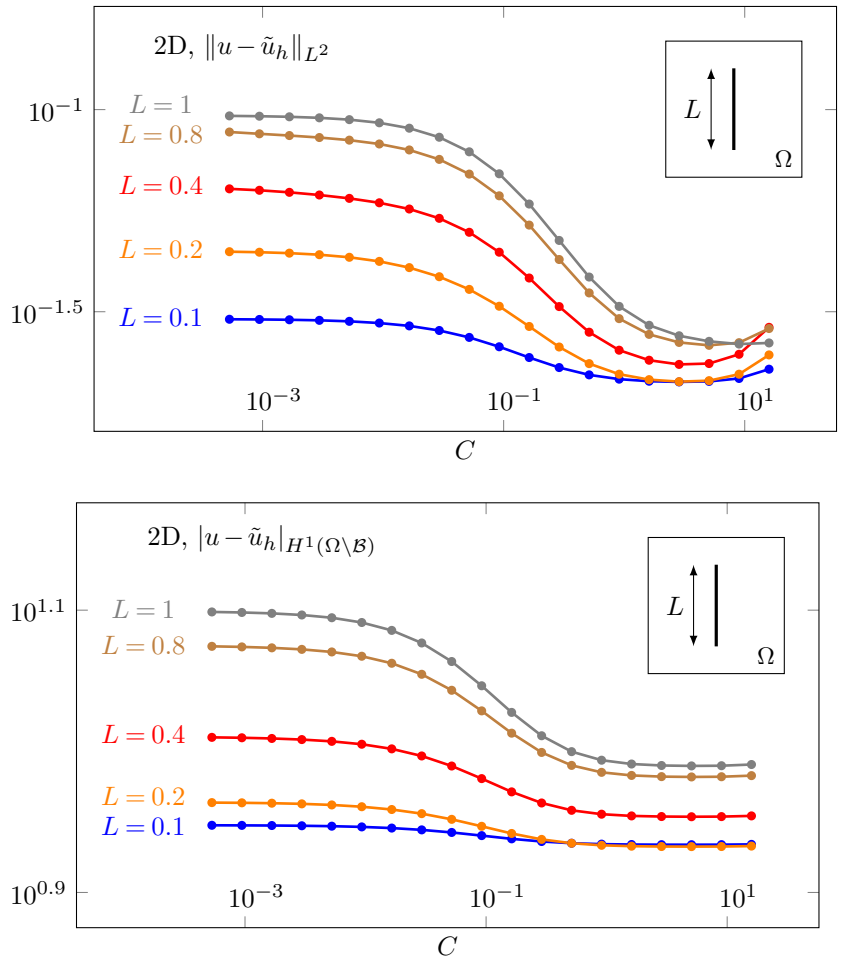


Figure 2.10: Effect of the length L of the band of cap on the error in the L^2 norm (left) and H^1 semi-norm (right) in 2D.

precisely, in the case of a second-order elliptic operator such as the Laplacian, one expects that the largest eigenvalue of K, and thus its conditioning κ, grows as

$$\kappa = \mathcal{O}(1/D),$$

where D denotes the thickness of the band of caps. Figure 2.12 illustrates the evolution of the residual during the iterations of an unpreconditioned Conjugate Gradient applied to the matrix K with TFEM. In this particular case, TFEM is triggered for the two smallest values of D, i.e., $D = 10^{-5}$ and $D = 10^{-6}$.

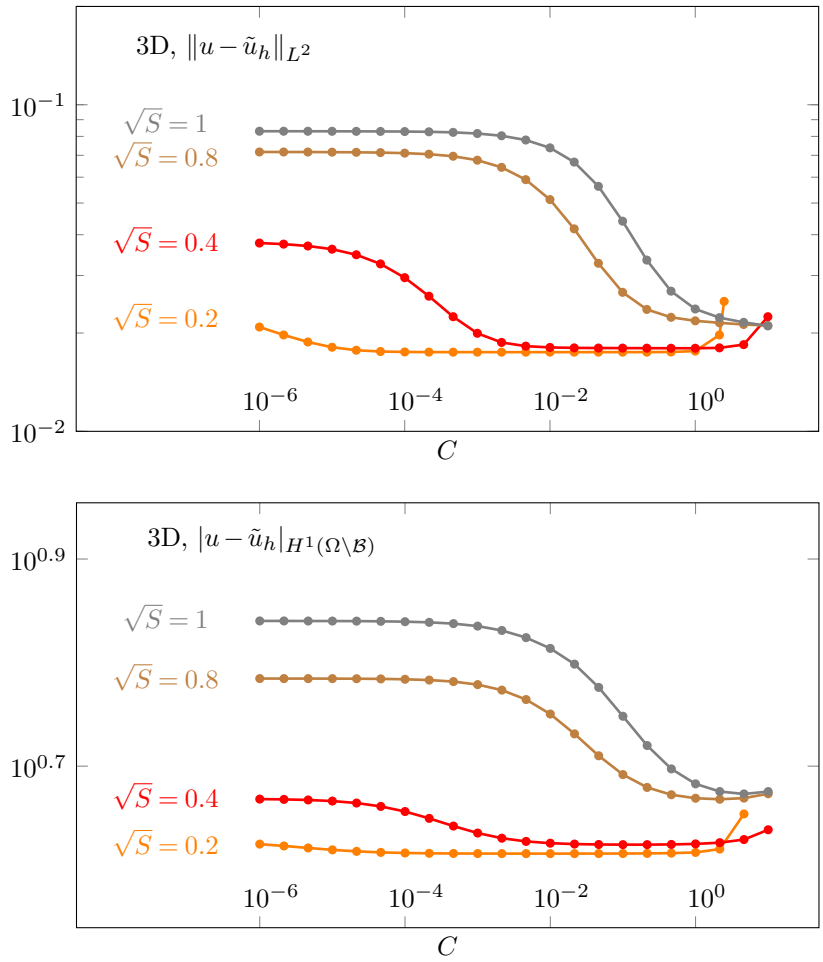
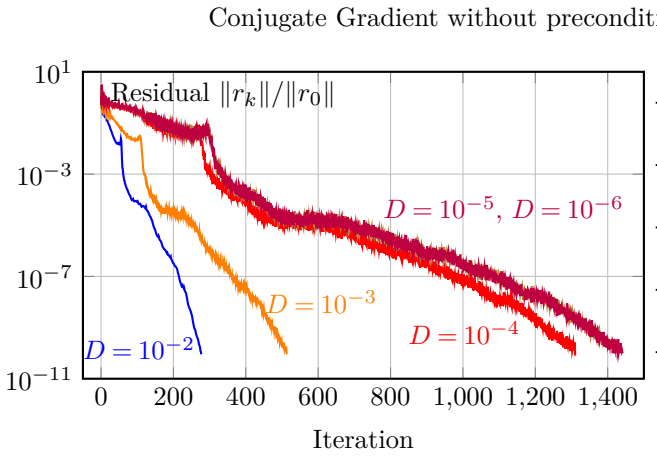


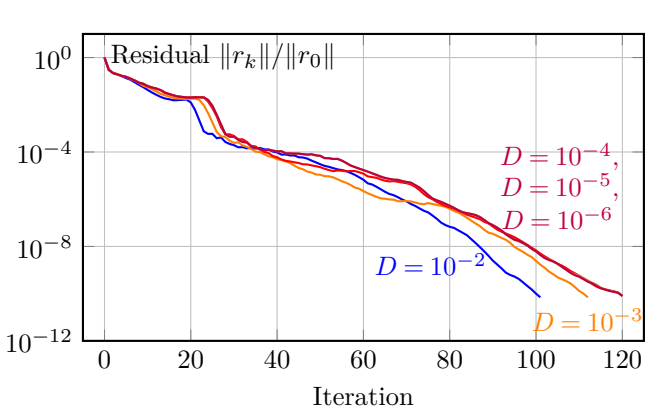
Figure 2.11: Effect of the area of the surface S of the plane of degenerated tetrahedra on the error in the L^2 norm (left) and H^1 semi-norm (right) in 3D.

The number of iterations is approximately multiplied by $\sqrt{10}$ when decreasing D from 10^{-2} to 10^{-3}, and from 10^{-3} to 10^{-4}, which corresponds to the theoretical worst-case scenario. On the other hand, a simple preconditioner of the incomplete Cholesky type, which is commonly used to solve this class of problems, proves to be perfectly sufficient to achieve convergence independent of κ. This is excellent news: the use of TFEM does not introduce any specific difficulty regarding the solution of linear systems.



D	κ	H_1
10^{-2}	3096	12%
10^{-3}	10027	12%
10^{-4}	96255	13%
10^{-5}	120200	14%
10^{-6}	120201	14%

Figure 2.12: This Figure shows iteration history of conjugate gradients to solve a problem with TFEM. The conditioning κ varies as $1/D$ for $10^{-4} \leq D \leq 10^{-2}$ (no TFEM). TFEM is triggered for $D < 10^{-4}$ and conditioning remains stable.



D	κ	H_1
10^{-2}	3096	12%
10^{-3}	10027	12%
10^{-4}	96255	13%
10^{-5}	120200	14%
10^{-6}	120201	14%

Figure 2.13: This Figure shows the same sequence of meshes with a decreasing band of cap width as Figure 2.12 but using preconditioned conjugate gradients. There, TFEM has no influence on the iterative solver's convergence.

2.3 Theory behind the TFEM scheme

In the previous section we showed that a simple numerical trick had allowed us to recover the convergence rate of the finite element method on meshes with a band of caps. In this section we demonstrate this convergence for the

choice of $J_{\min} \sim h^{d+1}$ in $\mathbb{R}^d$. To do so, we need to consider two cases: when the band of caps is not exactly degenerate but tends to a zero width, and the exactly degenerated case. For the first case, we do a classic finite element error analysis where we separate the error made on the band and the rest of the domain and we estimate and balance these carefully. In the second case for exactly degenerate bands of caps, we demonstrate that our scheme corresponds to an under-integrated mortar scheme with penalization of the jump across the band. In order to make this jump converge to zero as the mesh is refined, we obtain an expression the choice of $J_{\min}$.

In order to simplify the analysis, we consider the case of homogeneous Dirichlet boundary conditions ($u_D = 0$). Considering a nonhomogeneous Dirichlet condition would only add a level of technicalities unrelated to the TFEM method itself (Dirichlet lift, error of the interpolation of u_D on the boundary, etc.) which would make the analysis less clear and readable. In the notation of Section 2.2.1 we therefore have $V_h = V_h^0 = X_h \cap H_0^1(\Omega)$ in the following analysis.

2.3.1 TFEM for a band of arbitrarily small but non-zero caps

By applying the circumradius estimate of the Lagrange interpolation error and using the scaling of the band, we are able to bound the error $|u - \tilde{u}_h|_{H^1}$ in the H^1 semi-norm. This requires a careful decomposition and balancing of the error. The main difficulty is that the method (2.12) is not consistent in the sense that we are essentially solving a modified continuous problem (with diffusion coefficient $d(x)$ instead of 1), hence we lack so-called Galerkin orthogonality. Nevertheless this obstacle can be overcome. The analysis requires three main working tools/lemmas.

Lemma 1 (Circumradius estimate [Kobayashi and Tsuchiya (2015))**.** *Let $K \subset \mathbb{R}^2$ be an arbitrary triangle. Let $u \in W^{2,\infty}(K)$ and let $\Pi_K u$ be the linear Lagrange interpolation of u on K. Then there exists a constant C_c independent of u and K such that*

$$|u - \Pi_K u|_{W^{1,\infty}(K)} \leq C_c R_K |u|_{W^{2,\infty}(K)} \leq C_c \frac{h^2}{\bar{h}} |u|_{W^{2,\infty}(K)}. \tag{2.13}$$

where R_K is the circumradius of the K triangle.

Lemma 2 (Maximum-angle condition [Babuška and Aziz (1976))**.** *Let $K \subset \mathbb{R}^2$ be a triangle satisfying the maximum angle condition: $\alpha_K \leq \alpha_0 < \pi$ for some fixed α_0. Let $u \in H^2(K)$ and let $\Pi_K u$ be the linear Lagrange interpolation of u on K. Then there exists a constant C_I independent of u (but depending on the maximum angle α_0) such that*

$$|u - \Pi_K u|_{H^1(K)} \leq C_I h |u|_{H^2(K)}. \tag{2.14}$$

Lemma 3 (Young's inequality)**.** *Let $a, b \in \mathbb{R}$ and $r > 0$. Then*

$$ab \leq \frac{a^2}{2r} + \frac{rb^2}{2}. \tag{2.15}$$

Our main theoretical result is the TFEM error estimate (2.16) with a general value of D. We will then optimize this estimate with respect to D.

Theorem 4. *Let $u \in W^{2,\infty}(\Omega)$. Let $0 < D \leq 1$ in (2.11). Then*

$$\begin{aligned} &|u - \tilde{u}_h|^2_{H^1(\Omega\setminus\mathcal{B})} + D|u - \tilde{u}_h|^2_{H^1(\mathcal{B})} \\ &\quad \leq 4C_I^2 h^2 |u|^2_{H^2(\Omega\setminus\mathcal{B})} + \frac{4}{D}|\mathcal{B}||u|^2_{W^{1,\infty}(\mathcal{B})} + 6D|\mathcal{B}|C_c^2 \frac{h^4}{\bar{h}^2}|u|^2_{W^{2,\infty}(\mathcal{B})}. \end{aligned} \tag{2.16}$$

Proof. We subtract (2.12) and (2.6) with a test function $v = v_h \in V_h$:

$$\int_\Omega \nabla u \cdot \nabla v_h \,\mathrm{d}x - \int_\Omega d(x) \nabla \tilde{u}_h \cdot \nabla v_h \,\mathrm{d}x = 0. \tag{2.17}$$

We define $\xi_h = \Pi_h u - \tilde{u}_h \in V_h$ and set $v_h = \xi_h$ in (2.17):

$$\int_\Omega \Big(\nabla u - d(x)\nabla \Pi_h u + d(x)\nabla \Pi_h u - d(x)\nabla \tilde{u}_h \Big) \cdot \nabla \xi_h \,\mathrm{d}x = 0,$$

where we have introduced the terms $\pm d(x)\nabla\Pi_h u$. Rearranging gives us

$$\int_\Omega d(x)|\nabla \xi_h|^2 \,\mathrm{d}x = -\int_\Omega \Big(\nabla u - d(x)\nabla \Pi_h u \Big) \cdot \nabla \xi_h \,\mathrm{d}x.$$

We split the integrals over $\mathcal{B}$ and $\Omega \setminus \mathcal{B}$ and take (2.11) into account: $d(x) = 1$ over $\Omega \setminus \mathcal{B}$ and $d(x) = D$ over $\mathcal{B}$. This gives us:

$$\begin{aligned} &\int_{\Omega\setminus\mathcal{B}} |\nabla \xi_h|^2 \,\mathrm{d}x + D\int_{\mathcal{B}} |\nabla \xi_h|^2 \,\mathrm{d}x \\ &\quad = -\underbrace{\int_{\Omega\setminus\mathcal{B}} \Big(\nabla u - \nabla \Pi_h u \Big) \cdot \nabla \xi_h \,\mathrm{d}x}_{(\star)} - \underbrace{\int_{\mathcal{B}} \Big(\nabla u - D\nabla \Pi_h u \Big) \cdot \nabla \xi_h \,\mathrm{d}x}_{(\star\star)}. \end{aligned} \tag{2.18}$$

Estimating $(\star)$ is standard, because we have 'nice' elements on $\Omega \setminus \mathcal{B}$:

$$\begin{aligned} \big|(\star)\big| &\leq |u - \Pi_h u|_{H^1(\Omega\setminus\mathcal{B})} |\xi_h|_{H^1(\Omega\setminus\mathcal{B})} \leq \tfrac{1}{2}|u - \Pi_h u|^2_{H^1(\Omega\setminus\mathcal{B})} + \tfrac{1}{2}|\xi_h|^2_{H^1(\Omega\setminus\mathcal{B})} \\ &\leq \tfrac{1}{2}C_I^2 h^2 |u|^2_{H^2(\Omega\setminus\mathcal{B})} + \tfrac{1}{2}|\xi_h|^2_{H^1(\Omega\setminus\mathcal{B})}, \end{aligned} \tag{2.19}$$

where we have used Young's inequality (2.15) with $r = 1$ and then the Maximum Angle Condition from lemma 1. We note that the last term in (2.19) will be 'hidden' under the corresponding left-hand side term of (2.18).

To estimate $(\star\star)$, we proceed more carefully since this integral is evaluated on the band. First, we have

$$(\star\star) = \underbrace{\int_{\mathcal{B}} \Big(\nabla u - D\nabla u\Big)\cdot\nabla\xi_h \,\mathrm{d}x}_{(A)} + \underbrace{\int_{\mathcal{B}} \Big(D\nabla u - D\nabla\Pi_h u\Big)\cdot\nabla\xi_h \,\mathrm{d}x}_{(B)}, \tag{2.20}$$

where we have introduced $\pm D\nabla u$. Since $D<1$, we can estimate (A):

$$\begin{aligned}
\big|(A)\big| &= (1-D)\left|\int_{\mathcal{B}} \nabla u\cdot\nabla\xi_h \,\mathrm{d}x\right| \leq \int_{\mathcal{B}} \big|\nabla u\big|\big|\nabla\xi_h\big| \,\mathrm{d}x \\
&\leq \frac{1}{D}\int_{\mathcal{B}} \big|\nabla u\big|^2 \,\mathrm{d}x + \frac{D}{4}\int_{\mathcal{B}} \big|\nabla\xi_h\big|^2 \,\mathrm{d}x \\
&\leq \frac{1}{D}|\mathcal{B}||u|^2_{W^{1,\infty}(\mathcal{B})} + \frac{D}{4}\int_{\mathcal{B}} \big|\nabla\xi_h\big|^2 \,\mathrm{d}x
\end{aligned} \tag{2.21}$$

where we have used Young's inequality (2.15) with $r=\frac{D}{2}$ and taken the maximum of the first derivative on the band $|u|_{W^{1,\infty}(\mathcal{B})}$ to obtain an upper bound. Again, the last term in (2.21) will be 'hidden' under the corresponding left-hand side term of (2.18).

Finally, it remains to estimate term (B) from (2.20):

$$\begin{aligned}
\big|(B)\big| &\leq D\int_{\mathcal{B}} \big|\nabla u - \nabla\Pi_h u\big|\big|\nabla\xi_h\big| \,\mathrm{d}x \\
&\leq D\int_{\mathcal{B}} \big|\nabla u - \nabla\Pi_h u\big|^2 \,\mathrm{d}x + \frac{D}{4}\int_{\mathcal{B}} \big|\nabla\xi_h\big|^2 \,\mathrm{d}x \\
&\leq D|\mathcal{B}||u-\Pi_h u|^2_{W^{1,\infty}(\mathcal{B})} + \frac{D}{4}|\xi_h|^2_{H^1(\mathcal{B})} \\
&\leq D|\mathcal{B}|C_c^2\left(\frac{h^2}{\bar{h}}\right)^2 |u|^2_{W^{2,\infty}(\mathcal{B})} + \frac{D}{4}|\xi_h|^2_{H^1(\mathcal{B})}
\end{aligned} \tag{2.22}$$

where we have used Young's inequality (2.15) with $r=\frac{1}{2}$ and the circumradius estimate (2.13) in which the maximum of the second derivative on the band, $|u|_{W^{2,\infty}(\mathcal{B})}$, appears. As usual, the last term in (2.22) will be 'hidden' under the left-hand side of (2.18).

Combining estimates (2.18), (2.19), and (2.20)–(2.22) gives us

$$\begin{aligned}
&\frac{1}{2}|\xi_h|^2_{H^1(\Omega\setminus\mathcal{B})} + \frac{D}{2}|\xi_h|^2_{H^1(\mathcal{B})} \\
&\quad\leq \frac{1}{2}C_I^2 h^2 |u|^2_{H^2(\Omega\setminus\mathcal{B})} + \frac{1}{D}|\mathcal{B}||u|^2_{W^{1,\infty}(\mathcal{B})} + D|\mathcal{B}|C_c^2\frac{h^4}{\bar{h}^2}|u|^2_{W^{2,\infty}(\mathcal{B})}.
\end{aligned} \tag{2.23}$$

This gives us an estimate for $\xi_h = \Pi_h u - \tilde{u}_h$. Now we estimate the error $u - \tilde{u}_h$ using the fact that $u - \tilde{u}_h = (u - \Pi_h u) + \xi_h$:

$$\begin{aligned}
&|u-\tilde{u}_h|^2_{H^1(\Omega\setminus\mathcal{B})} + D|u-\tilde{u}_h|^2_{H^1(\mathcal{B})} \\
&\leq \big(|u-\Pi_h u|_{H^1(\Omega\setminus\mathcal{B})} + |\xi_h|_{H^1(\Omega\setminus\mathcal{B})}\big)^2 + D\big(|u-\Pi_h u|_{H^1(\mathcal{B})} + |\xi_h|_{H^1(\mathcal{B})}\big)^2 \\
&\leq 2|u-\Pi_h u|^2_{H^1(\Omega\setminus\mathcal{B})} + 2|\xi_h|^2_{H^1(\Omega\setminus\mathcal{B})} + 2D|u-\Pi_h u|^2_{H^1(\mathcal{B})} + 2D|\xi_h|^2_{H^1(\mathcal{B})},
\end{aligned} \tag{2.24}$$

where we have used the fact that $(a+b)^2 \leq 2a^2 + 2b^2$. The ξ_h-terms in (2.24) are estimated using (2.23). The remaining terms have already been dealt with in (2.19) and (2.22), respectively:

$$|u-\Pi_h u|^2_{H^1(\Omega\setminus\mathcal{B})} \leq C_I^2 h^2 |u|^2_{H^2(\Omega\setminus\mathcal{B})}, \tag{2.25}$$

since $\Omega \setminus \mathcal{B}$ contains only 'nice' elements, and

$$D|u-\Pi_h u|^2_{H^1(\mathcal{B})} \leq D|\mathcal{B}||u-\Pi_h u|^2_{W^{1,\infty}(\mathcal{B})} \leq D|\mathcal{B}|C_c^2 \frac{h^4}{\bar{h}^2}|u|^2_{W^{2,\infty}(\mathcal{B})}, \tag{2.26}$$

where we took the maximum of the gradient of $u - \Pi_h u$ over the band $\mathcal{B}$ and used the circumradius estimate from lemma 1. Using (2.23), (2.25), and (2.26) in (2.24) gives us the error estimate

$$\begin{aligned}
&|u-\tilde{u}_h|^2_{H^1(\Omega\setminus\mathcal{B})} + D|u-\tilde{u}_h|^2_{H^1(\mathcal{B})} \\
&\leq 4C_I^2 h^2 |u|^2_{H^2(\Omega\setminus\mathcal{B})} + \frac{4}{D}|\mathcal{B}||u|^2_{W^{1,\infty}(\mathcal{B})} + 6D|\mathcal{B}|C_c^2 \frac{h^4}{\bar{h}^2}|u|^2_{W^{2,\infty}(\mathcal{B})},
\end{aligned}$$

which is our desired estimate. □

Remark 1. We note that for $D = 1$ (i.e. standard FEM), the second right-hand side term in (2.16) vanishes, since this comes from the consistency term (A) which is zero in this case, cf. (2.21). On the other hand, the last right-hand side term in (2.16) is $O(h^2)$ if and only if $\bar{h} \geq Ch^2$ for some constant C. This is due to the fact that $|\mathcal{B}|\frac{h^4}{\bar{h}^2} = \bar{h}L\frac{h^4}{\bar{h}^2} = L\frac{h^2}{\bar{h}}h^2$. This means that if $\bar{h} \geq Ch^2$ then standard finite elements have optimal $O(h)$ convergence in the $H^1(\Omega)$-seminorm.

Now we will optimize the value of D in the case of a band of caps. The value of D appears both in the numerator and the denominator of the error estimate of Theorem 4. The following lemma optimizes the value of such a function:

Lemma 5. *Let $a, b > 0$ and*

$$g(x) = \frac{a}{x} + bx. \tag{2.27}$$

Then g attains its minimum at a unique $x_{\min}$, where

$$x_{\min} = \sqrt{\frac{a}{b}}, \quad g(x_{\min}) = 2\sqrt{ab}. \tag{2.28}$$

Proof. We have $g'(x) = b - a/x^2$ which is zero for $x = \sqrt{a/b}$. Then $g(\sqrt{a/b}) = a\sqrt{b/a} + b\sqrt{a/b} = 2\sqrt{ab}$. □

We can now apply the result of this lemma to the error estimate of Theorem 4. This gives us an optimal choice for D:

Theorem 6. *Let $u \in W^{2,\infty}(\Omega)$. Let $\mathcal{T}_h$ contain a band of caps of length $L \sim 1$ and width $\bar{h}$. In (2.11), set*

$$D = \min\left(\frac{\bar{h}}{h^2}\frac{|u|_{W^{1,\infty}(\mathcal{B})}}{|u|_{W^{2,\infty}(\mathcal{B})}}\sqrt{\frac{2}{3C_c}},\, 1\right), \tag{2.29}$$

then the tempered FEM solution $\tilde{u}_h$ satisfies

$$|u - \tilde{u}_h|_{H^1(\Omega\setminus\mathcal{B})} + \sqrt{D}|u - \tilde{u}_h|_{H^1(\mathcal{B})} \le C(u, L)h, \tag{2.30}$$

where

$$C(u,L) = 2\sqrt{2}C_I|u|_{H^2(\Omega\setminus\mathcal{B})} + 2\sqrt{2}\sqrt[4]{6}\sqrt{C_c L}\sqrt{|u|_{W^{1,\infty}(\mathcal{B})}|u|_{W^{2,\infty}(\mathcal{B})}} \tag{2.31}$$

is a constant independent of $\bar{h}, h$.

Proof. We optimize the last two terms in (2.16) with respect to D. To this end, we denote them as

$$g(D) = \frac{4}{D}|\mathcal{B}||u|^2_{W^{1,\infty}(\mathcal{B})} + 6D|\mathcal{B}|C_c^2\frac{h^4}{\bar{h}^2}|u|^2_{W^{2,\infty}(\mathcal{B})},$$

which is a function of the form (2.27) with respect to D. The application of Lemma 5 gives us the optimal choice of D:

$$D_{\text{opt}} = \sqrt{\frac{4|\mathcal{B}||u|^2_{W^{1,\infty}(\mathcal{B})}}{6|\mathcal{B}|C_c^2\frac{h^4}{\bar{h}^2}|u|^2_{W^{2,\infty}(\mathcal{B})}}} = \frac{\bar{h}}{h^2}\frac{|u|_{W^{1,\infty}(\mathcal{B})}}{|u|_{W^{2,\infty}(\mathcal{B})}}\sqrt{\frac{2}{3C_c}},$$

which is (2.29). Moreover,

$$\begin{aligned} g(D_{\text{opt}}) &= 2\sqrt{4|\mathcal{B}||u|^2_{W^{1,\infty}(\mathcal{B})}6|\mathcal{B}|C_c^2\frac{h^4}{\bar{h}^2}|u|^2_{W^{2,\infty}(\mathcal{B})}} \\ &= 4\sqrt{6}C_c|\mathcal{B}|\frac{h^2}{\bar{h}}|u|_{W^{1,\infty}(\mathcal{B})}|u|_{W^{2,\infty}(\mathcal{B})} \\ &\le 4\sqrt{6}C_c L h^2|u|_{W^{1,\infty}(\mathcal{B})}|u|_{W^{2,\infty}(\mathcal{B})}, \end{aligned}$$

since $|\mathcal{B}| \leq \bar{h}L$ (inequality here since $\mathcal{B}$ is not a rectangle). With the choice $D = D_{\text{opt}}$, estimate (2.16) therefore becomes

$$\begin{aligned} &|u-\tilde{u}_h|^2_{H^1(\Omega\setminus\mathcal{B})} + D|u-\tilde{u}_h|^2_{H^1(\mathcal{B})} \\ &\quad \leq 4C_I^2 h^2 |u|^2_{H^2(\Omega\setminus\mathcal{B})} + 4\sqrt{6}C_c L h^2 |u|_{W^{1,\infty}(\mathcal{B})}|u|_{W^{2,\infty}(\mathcal{B})}. \end{aligned} \tag{2.32}$$

Now we wish to take the square root of (2.32). This is done by considering the equivalence of norms in $\mathbb{R}^2$, which for $a, b, c, d \geq 0$ gives us

$$a^2 + b^2 \leq c^2 + d^2 \quad \Longrightarrow \quad a + b \leq \sqrt{2}(c+d). \tag{2.33}$$

Applying this to (2.32) gives

$$\begin{aligned} &|u-\tilde{u}_h|_{H^1(\Omega\setminus\mathcal{B})} + \sqrt{D}|u-\tilde{u}_h|_{H^1(\mathcal{B})} \\ &\quad \leq 2\sqrt{2}C_I h|u|_{H^2(\Omega\setminus\mathcal{B})} + 2\sqrt{2}\sqrt[4]{6}\sqrt{C_c L}h\sqrt{|u|_{W^{1,\infty}(\mathcal{B})}|u|_{W^{2,\infty}(\mathcal{B})}}. \end{aligned}$$

This is the desired estimate (2.30) with the constant $C(u, L)$ defined by (2.31). Since D needs to be smaller than 1, we take the minimum between D_{opt} and 1. □

We note that this choice of D_{opt} can be transformed back to a choice of $J_{\min}$ by using (2.10) and the definition of $d(x)$ from (2.11):

$$\begin{aligned} J_{\min,\text{opt}} &= \frac{J}{D_{\text{opt}}} \\ &= \frac{h\bar{h}}{2}\frac{h^2}{\bar{h}}\frac{|u|_{W^{2,\infty}(\mathcal{B})}}{|u|_{W^{1,\infty}(\mathcal{B})}}\sqrt{\frac{3C_c}{2}} \\ &= h^3\frac{|u|_{W^{2,\infty}(\mathcal{B})}}{|u|_{W^{1,\infty}(\mathcal{B})}}\sqrt{\frac{3C_c}{8}} \end{aligned} \tag{2.34}$$

Thus we have proved that the error converges in 2D when we choose $J_{\min} \sim h^3$ as we can see in the convergences plots of the numerical experiment in Figure 2.6.

Contrary to what one might think, the choice of $J_{\min}$ is independent of the length L. This corresponds well to what was observed during the numerical experiments in section 2.2.5. The error estimate provided by the theorem 6 predicts an error that varies as $\sim \sqrt{L}$ when $J_{\min}$ is chosen optimally. In the numerical experiment we observed this dependence in $\sim \sqrt{L}$ when $J_{\min}$ was chosen too small (locking), as we can see in figure 2.10.

Remarks:

- The expression for D in (2.29) guarantees that $D \leq 1$. One might ask what exactly happens when the expression in the first argument of the minimum is greater than 1. In this case $\bar{h} \geq Ch^2$, which means that standard finite elements (i.e. $D = 1$) have optimal $O(h)$ convergence rate (cf. Remark 1), and therefore no tempering is necessary in the finite element method, i.e. $D = 1$ is the best choice in this case. This justifies the fact that the diffusion coefficient is always decreased in TFEM, never increased.

- We note that the very regular structure of elements in the band $\mathcal{B}$ as presented in Figure 2.2 is not needed in the analysis, the band is only assumed to be somehow composed of 'degenerating' elements.

- There need not be a single band of caps, there can be several layers of these bands next to each other or several separate bands of caps in the domain Ω. The only condition for the analysis to remain the same is that their number does not grow as $h \to 0$.

- Poincaré's inequality immediately gives us the estimate

$$|u - \tilde{u}_h|_{L^2(\Omega)} \leq C(u, L)h \tag{2.35}$$

  for some constant $C(u, L)$ independent of h. This can be easily seen by applying Poincaré's inequality to (2.30) on $\Omega \setminus \mathcal{B}$ and a simple continuity argument on $\mathcal{B}$. This L^2-error estimate is suboptimal. However, we were unable to prove optimal L^2-error estimates, as the standard Aubin-Nitsche duality technique runs into technical obstacles. Nevertheless numerical experiments indicate that the method has optimal convergence rates in $L^2(\Omega)$.

- The constant C_c from Lemma 1 is very close to 1. The paper [Kučera (2016)] gives the value $2\sqrt{2}+2$, which can most likely be improved. In the end, the factor $\sqrt{\frac{2}{3C_c}}$ from (2.29) can simply be neglected for simplicity.

Extension to 3D

The results of Theorem 6 can be straightforwardly extended to 3D. In this case, the thin band $\mathcal{B}$ with width $\bar{h}$ and length L becomes a thin slab $\mathcal{S}$ of width $\bar{h}$ and area $\mathcal{A}$ of its base. We assume that the slab is partitioned into degenerating tetrahedra. The situation in 3D is much more complicated than 2D, as there are 9 basic types of degenerate tetrahedra, cf. [Korotov et al. (2022)]. However, what they have in common is that at least one of their four altitudes is always small (similarly to one of the altitudes of a degenerate triangle). We assume the partition of S into tetrahedra is such that the minimal altitude of each tetrahedron is $\bar{h}$ and its longest edge is h. Then the proof of Theorem 6 can be

trivially extended to 3D provided we have an estimate for the error of linear Lagrange interpolation on a tetrahedron T of the same form as in Lemma 1. We then obtain an $O(h)$ estimate of the TFEM error in the $H^1(\Omega \setminus \mathcal{S})$ seminorm provided $D \sim \frac{\bar{h}}{h^2}$, exactly as in Theorem 6. We note that in 3D, this choice corresponds to $J_{\min} \sim h^4$, which agrees with the numerical experiments from Figure 2.7.

The necessary interpolation error estimate analogous to Lemma 1 can be easily obtained from the estimate of Baidakova, [Baidakova (2020)], as in the following lemma.

Lemma 7 (Interpolation error estimate in 3D)**.** *Let $T \subset \mathbb{R}^3$ be an arbitrary tetrahedron with longest edge length h and minimal altitude $\bar{h}$. Let $u \in W^{2,\infty}(T)$ and let $\Pi_T u$ be the linear Lagrange interpolation of u on T. Then there exists a constant C_c independent of u and T such that*

$$|u - \Pi_T u|_{W^{1,\infty}(T)} \leq C_c \frac{h^2}{\bar{h}} |u|_{W^{2,\infty}(T)}. \tag{2.36}$$

Proof. Denote the faces of T as $T_i, i = 1, \ldots, 4$. Let R_i be the radius of the circle circumscribed about T_i, and let φ_{ij} be the angle between faces T_i, T_j. Let $R = \max_i R_i$ and $\sin\varphi = \max_{ij} \sin\varphi_{ij}$. Then in [Baidakova (2020)] it is proven that for some constant C,

$$|u - \Pi_T u|_{W^{1,\infty}(T)} \leq C \frac{R}{\sin\varphi} |u|_{W^{2,\infty}(T)}. \tag{2.37}$$

Now we estimate the factor $\frac{R}{\sin\varphi}$. Let $R = R_i$ for some i. Then the corresponding face T_i is a triangle with vertices A, B, C. Assume that AB is the longest edge. The law of sines then states that $2R_i = |AC||AB|/h_c$, where h_c is the altitude in ABC corresponding to vertex C. Moreover, let $\bar{h}_c$ be the height of the tetrahedron T corresponding to the apex C. Then trivially, $\sin\varphi_{ij} = \bar{h}_c/h_c$, combining these expressions, we get

$$\frac{R}{\sin\varphi} \leq \frac{R}{\sin\varphi_{ij}} = \frac{|AC||AB|/(2h_c)}{\bar{h}_c/h_c} = \frac{|AC||AB|}{2\bar{h}_c} \leq \frac{h^2}{2\bar{h}}, \tag{2.38}$$

since $\bar{h} \leq \bar{h}_c$. Substituting (2.38) into (2.37) gives the desired estimate. □

2.3.2 Zero-measure elements – TFEM and Mortaring

In §2.3.1, we proved a sufficient condition for the convergence of the tempered finite element method for any band of degenerate elements with nonzero width. It is important to realize that this proof is not valid if the band width is exactly zero. It is however valid when the band degenerates to zero width whatever the rate. In order to generalize the proof of section 3.1 to the case where $\bar{h} = 0$

(truly or up to machine precision), we show that our approach is equivalent to a mortar method penalizing the jump of the solution on the band.

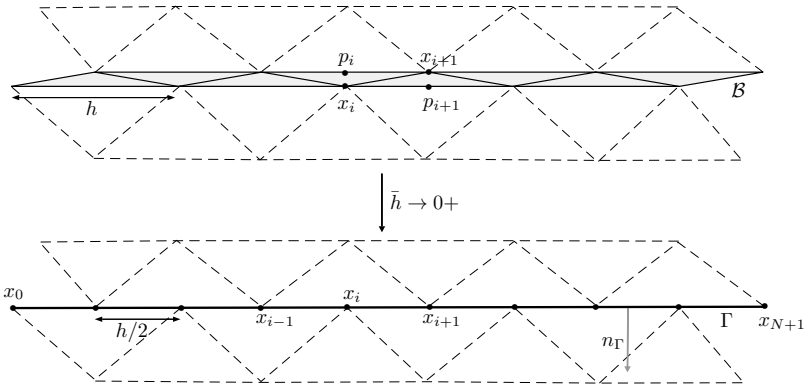


Figure 2.14: Degeneration of a band of caps $\mathcal{B}$ to a line Γ as the width of the band $\bar{h}$ goes to zero.

Again we assume a regular structure of the band of caps $\mathcal{B}$ with length L and width $\bar{h}$. The goal of this section is to formulate the limiting scheme of (2.12) for $\bar{h} = 0$, which we will interpret as the limit for $\bar{h} \to 0^+$.

As $\bar{h} \to 0^+$, the band $\mathcal{B}$ degenerates to a line Γ and the resulting triangulation $\mathcal{T}_h^\Gamma$ is not conforming on Γ, cf. Figure 2.14. We will also call Γ a zero-measure band. We will label the elements in the band as $K_i, i = 1, \ldots, N$, ordered from one end of $\mathcal{B}$ to the other. For an element $K_i \in \mathcal{B}$, let x_i denote its maximum angle vertex and p_i be the foot of the altitude corresponding to x_i, cf. Figure 2.14. We also label the endpoints of Γ as x_0 and x_{N+1},respectively. As we send $\bar{h} \to 0^+$ and $\mathcal{B}$ degenerates to a line Γ, each x_i merges with its corresponding p_i. We note that each x_i is the vertex of a triangle adjacent to $\mathcal{B}$. Let n_Γ be the unit normal to Γ (arbitrarily oriented). We can then define the two one-sided limits of a function $f : \Omega \setminus \Gamma \to \mathbb{R}$ at $x \in \Gamma$:

$$f^+(x) = \lim_{t \to 0^+} f(x + t n_\Gamma), \quad f^-(x) = \lim_{t \to 0^-} f(x + t n_\Gamma) \tag{2.39}$$

and the jump

$$[f]_x = f^-(x) - f^+(x), \tag{2.40}$$

where we will sometimes omit the subscript x for simplicity.

On $\mathcal{B}$ we will need to split the gradient of a function into two parts: $\nabla_\perp u$ will be the component of ∇u in the direction perpendicular to Γ, i.e. $\nabla_\perp u = (\nabla u \cdot n_\Gamma) n_\Gamma$, while $\nabla_\parallel u$ denotes the component of ∇u in the direction parallel

to Γ. Using this notation, we can rewrite the scheme (2.12), i.e.

$$\int_{\Omega\setminus\mathcal{B}} \nabla\tilde{u}_h\cdot\nabla v_h\,\mathrm{d}x + D\int_{\mathcal{B}} \nabla\tilde{u}_h\cdot\nabla v_h\,\mathrm{d}x = (f,v_h),\quad \forall v_h\in V_h. \tag{2.41}$$

On $\mathcal{B}$ we write

$$\nabla\tilde{u}_h\cdot\nabla v_h = (\nabla_\perp\tilde{u}_h+\nabla_\parallel\tilde{u}_h)\cdot(\nabla_\perp v_h+\nabla_\parallel v_h) = \nabla_\perp\tilde{u}_h\cdot\nabla_\perp v_h+\nabla_\parallel\tilde{u}_h\cdot\nabla_\parallel v_h,$$

since $\nabla_\perp\tilde{u}_h\cdot\nabla_\parallel v_h = \nabla_\parallel\tilde{u}_h\cdot\nabla_\perp v_h = 0$ by definition. We can therefore write (2.41) as

$$\int_{\Omega\setminus\mathcal{B}} \nabla\tilde{u}_h\cdot\nabla v_h\,\mathrm{d}x + D\int_{\mathcal{B}} \nabla_\perp\tilde{u}_h\cdot\nabla_\perp v_h+\nabla_\parallel\tilde{u}_h\cdot\nabla_\parallel v_h\,\mathrm{d}x = (f,v_h). \tag{2.42}$$

Now we derive the form of (2.42) for $\bar{h}=0$ by sending $\bar{h}\to 0^+$. Eventually, we will have $D\sim\bar{h}/h^2$, where the numerator $\bar{h}$ will play an important role in the derivation. For this reason, we write $D(\bar{h},h)=\bar{h}\widetilde{D}(h)$ and it will be important to choose $\widetilde{D}$ appropriately in the scheme.

Lemma 8. *Let $D=\bar{h}\widetilde{D}$. Then the tempered FEM scheme* (2.12) *for $\bar{h}=0$ is equivalent to the formulation*

$$\int_{\Omega\setminus\Gamma} \nabla\tilde{u}_h\cdot\nabla v_h\,\mathrm{d}x + \widetilde{D}\frac{h}{2}\sum_{x_i\in\Gamma}[\tilde{u}_h]_{x_i}[v_h]_{x_i} = (f,v_h),\quad \forall v_h\in V_h^\Gamma, \tag{2.43}$$

where

$$V_h^\Gamma = \{v_h\in C(\Omega\setminus\Gamma):\ v_h|_K\in P^1(K)\forall K\in\mathcal{T}_h, v_h|_{\partial\Omega}=0\}$$

is the set of globally continuous piecewise linear functions discontinuous on Γ.

Remark 2. The scheme (2.43) is similar to the nonconsistent penalization method of Babuška, cf. [Babuška (1973)],

$$\int_{\Omega\setminus\Gamma} \nabla u_h\cdot\nabla v_h\,\mathrm{d}x + \widetilde{D}\int_\Gamma [u_h][v_h]\,\mathrm{d}S = (f,v_h),\quad \forall v_h\in V_h^\Gamma, \tag{2.44}$$

due to the fact that

$$\int_\Gamma [u_h][v_h]\,\mathrm{d}S \approx \frac{h}{2}\sum_{x_i\in\Gamma}[u_h]_{x_i}[v_h]_{x_i} \tag{2.45}$$

is a *composite trapezoidal quadrature* approximation on the partition given by the nodes $x_i\in\Gamma$, which has step size $h/2$. We note that the endpoints x_0, x_{N+1} of Γ do not come into play, since $[v_h]_{x_0}=[v_h]_{x_{N+1}}=0$ for all $v_h\in V_h^\Gamma$. Also note that (2.45) is an approximation, not an equality, since $[u_h][v_h]$ is a piecewise quadratic function and composite trapezoidal quadratures integrate exactly only piecewise linear functions.

Proof of Lemma 8. Consider the scheme (2.12) in the form (2.42). Let φ_i, $i = 1, \dots, M$, be the standard 'tent' basis functions of V_h corresponding to individual vertices of $\mathcal{T}_h$. We write $\tilde{u}_h(x) = \sum_{j=1}^{M} U_j \varphi_j(x)$, where $U_j \in \mathbb{R}$, and set $v_h := \varphi_k$. Then, (2.42) becomes

$$\sum_{j=1}^{M} U_j \int_{\Omega \setminus \mathcal{B}} \nabla \varphi_j \cdot \nabla \varphi_k \,\mathrm{d}x + \sum_{j=1}^{M} U_j \bar{h} \widetilde{D} \int_{\mathcal{B}} \nabla_\perp \varphi_j \cdot \nabla_\perp \varphi_k + \nabla_\parallel \varphi_j \cdot \nabla_\parallel \varphi_k \,\mathrm{d}x = (f, \varphi_k). \tag{2.46}$$

Now consider that for all j, it holds $|\nabla_\parallel \varphi_j| \leq 1/h$ on each $K \in \mathcal{B}$ and that $|\mathcal{B}| \leq L\bar{h}$. Therefore, for fixed h,

$$\left| \bar{h} \widetilde{D} \int_{\mathcal{B}} \nabla_\parallel \varphi_j \cdot \nabla_\parallel \varphi_k \,\mathrm{d}x \right| \leq \bar{h}^2 L \widetilde{D} h^{-2} \to 0, \text{ as } \bar{h} \to 0^+,$$

therefore these terms 'disappear' from the limiting system of linear equations (2.46) for $\bar{h} = 0$.

On the other hand, on an element $K_i \in \mathcal{B}$ with apex x_i we have

$$\nabla_\perp \varphi_j = \pm \frac{\varphi_j(x_i) - \varphi_j(p_i)}{\bar{h}} n_\Gamma,$$

for all j (the sign depends on the orientation of K with respect to n_Γ). Therefore (since $|K| = \frac{1}{2}\bar{h}h$ for all $K \in \mathcal{B}$),

$$\begin{aligned} \bar{h} \widetilde{D} \int_{\mathcal{B}} \nabla_\perp \varphi_j \cdot \nabla_\perp \varphi_k \,\mathrm{d}x &= \bar{h} \widetilde{D} \sum_{K_i \in \mathcal{B}} \frac{1}{2} \bar{h} h \frac{\varphi_j(x_i) - \varphi_j(p_i)}{\bar{h}} \frac{\varphi_k(x_i) - \varphi_k(p_i)}{\bar{h}} \\ &= \widetilde{D} \frac{h}{2} \sum_{K_i \in \mathcal{B}} \big(\varphi_j(x_i) - \varphi_j(p_i)\big)\big(\varphi_k(x_i) - \varphi_k(p_i)\big) \longrightarrow \widetilde{D} \frac{h}{2} \sum_{x_i \in \Gamma} [\varphi_j]_{x_i} [\varphi_k]_{x_i}, \end{aligned}$$

for $\bar{h} \to 0^+$.

Altogether, the limiting formulation of (2.46) for $\bar{h} = 0$ is

$$\sum_{j=1}^{M} U_j \int_{\Omega \setminus \Gamma} \nabla \varphi_j \cdot \nabla \varphi_k \,\mathrm{d}x + \sum_{j=1}^{M} U_j \widetilde{D} \frac{h}{2} \sum_{x_i \in \Gamma} [\varphi_j]_{x_i} [\varphi_k]_{x_i} = (f, \varphi_k),$$

which is equivalent to (2.43).

□

Now that we have a limiting formulation for $\bar{h} = 0$, the TFEM scheme can be analyzed. The main result is Theorem 13, which basically states that if $\widetilde{D} \sim \frac{1}{h^2}$, then

$$|u - \tilde{u}_h|_{H^1(\Omega \setminus \Gamma)} + \sqrt{\widetilde{D}} \big\| [u - \tilde{u}_h] \big\|_{L^2(\Gamma)} \leq C(u, L) h, \tag{2.47}$$

as expected. In order not to break the flow of the chapter, we move the theorem and its proof to Appendix A.2. We note that similarly as in (2.34), the choice $\widetilde{D} \sim \frac{1}{h^2}$ corresponds to simply taking $J_{\min} \sim h^3$ in the code.

2.4 Extensions

So far we've worked with a scalar Laplacian operator. In this section we discuss possible extensions. The first one is to consider linear elasticity. The second extension deals with the potential use of the TFEM for the mortaring of non-conformal meshes, even when there is a large difference in mesh size. Third, we show that high-order elements adapt well to our approach. With a slightly modified choice of $J_{\min}$, the TFEM ensures the proper convergence rate of high-order elements. The fourth and last extension studies the TFEM capabilities for advection type equations.

2.4.1 Extension to linear elasticity

The deformation $\mathbf{u}$ of a solid under mechanical loads is defined by the system of partial differential equations:

$$\begin{aligned} \nabla \cdot \sigma(\epsilon(\mathbf{u})) &= \mathbf{f} && \text{in } \Omega, \\ \mathbf{u} &= \mathbf{u}_D && \text{on } \Gamma_D, \\ \sigma \cdot \mathbf{n} &= \mathbf{t}_N && \text{on } \Gamma_N, \end{aligned}$$

where σ is the stress tensor, $\mathbf{f}$ body forces, and $\mathbf{t}_N$ surface traction applied on the part Γ_N of the boundary. The complementary of the boundary Γ_D is where Dirichlet boundary conditions $\mathbf{u}_D$ are imposed. The infinitesimal strain tensor ϵ is given by $\epsilon(\mathbf{u}) = \frac{1}{2}\left(\nabla \mathbf{u} + \nabla \mathbf{u}^T\right)$.

The relationship between the stress tensor and the infinitesimal strain tensor is assumed linear (Hooke's model) :

$$\sigma = \mathbf{C} : \epsilon,$$

where $\mathbf{C}$ is the elasticity tensor for plane strain in our case.

In this section, we simulate the interaction between two different materials with two kinds of meshes: A regular mesh with the interface directly represented in it for the classic FEM and a mesh where the two materials are connected through a band of caps for the TFEM approach. The geometry of the problem is shown in Figure 2.15 as well as the the von Mises stresses $\sigma_{\text{vm}} = \sqrt{\sigma_{xx}^2 + \sigma_{yy}^2 - \sigma_{xx}\sigma_{yy} + 3\sigma_{xy}^2}$. Results for both the classic FEM and the TFEM approach are visually very similar.

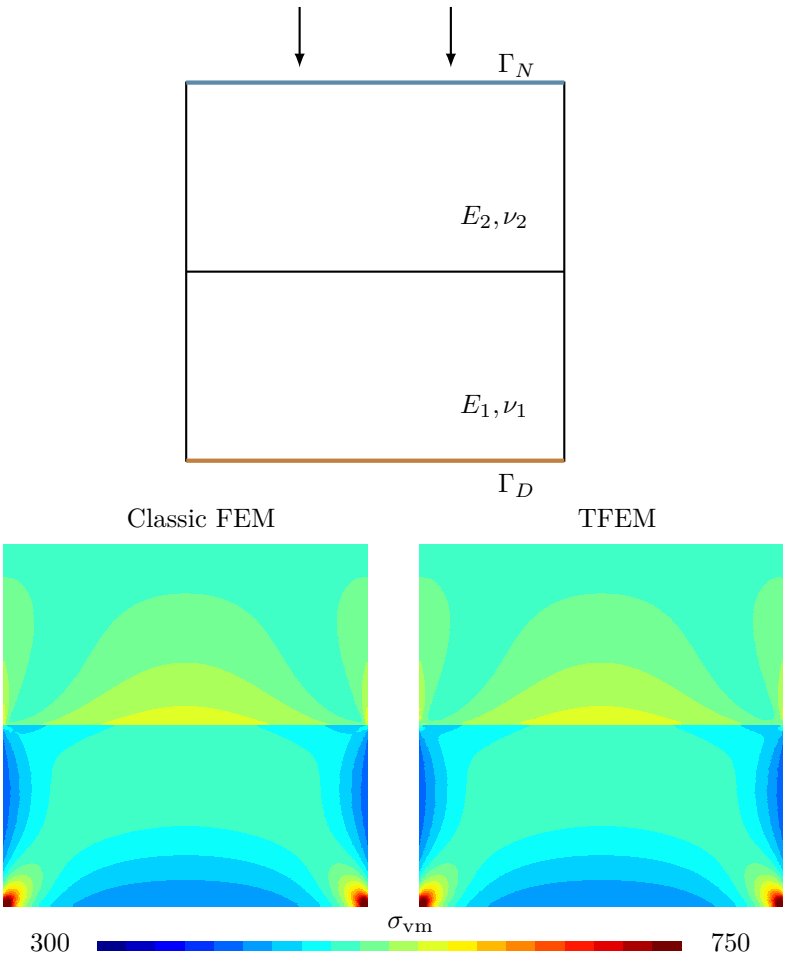


Figure 2.15: Problem configuration (top) and von Mises stresses for the regular mesh (bottom left) and the mesh with a band of caps using TFEM (bottom right).

To validate this observation we consider a problem with a manufactured solution such as the one presented in section 2.2. As shown in the convergence plots at Figure 2.16, the TFEM approach yields results that are comparable to those obtained using traditional finite element methods (FEM). However, the key advantage of TFEM lies in its flexibility: it allows for accurate simulation without requiring the mesh to be of good quality and can thus simplify the meshing of domains with an interface. The convergence graphs illustrate that both the traditional FEM and TFEM achieve similar convergence rates as the mesh is refined, validating the accuracy of the TFEM.

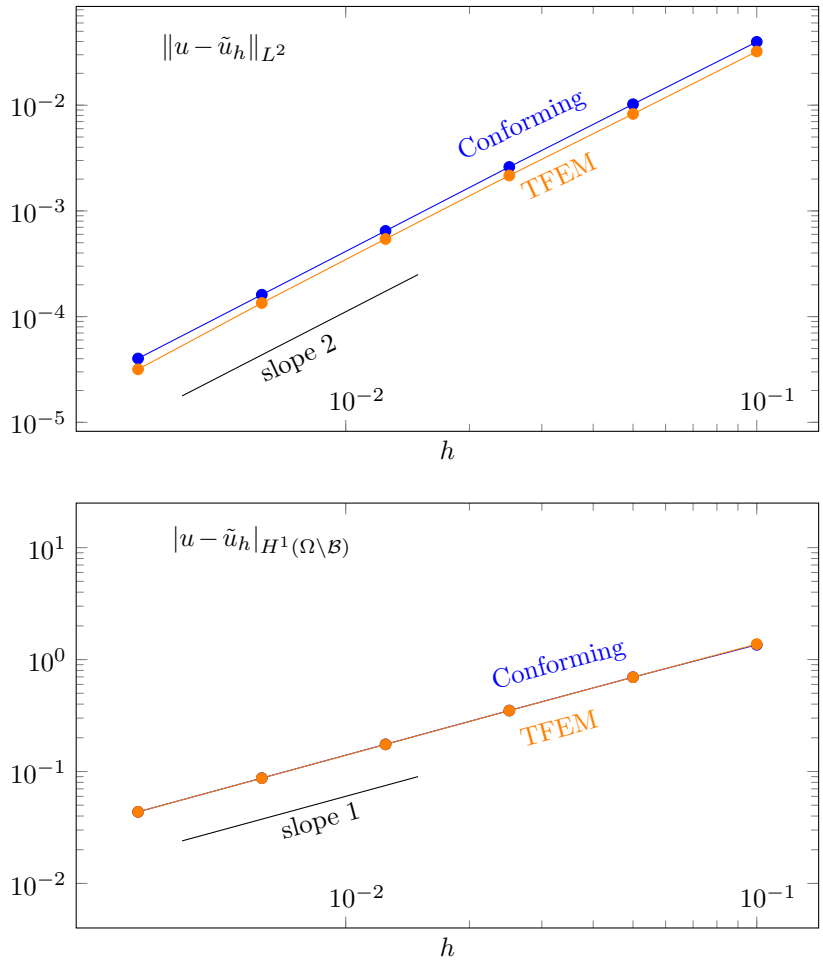


Figure 2.16: Convergence of the error in the L^2 norm (left) and H^1 semi-norm (right) for a manufactured elasticity problem.

2.4.2 Mesh mortaring

The TFEM method is equivalent to a mortaring method in which the solution jump is penalized, cf. Section 2.3.2. Visually, as shown in Figure 2.4, the mesh resembles a non-conforming mesh. This raises the possibility that the method could be valuable for mortaring non-conforming meshes within a standard FEM code, allowing for the simulation of specific regions of the domain with a more

refined mesh without the need to ensure connectivity between different zones during mesh generation. The objective of the following numerical experiment is to compare the results between our approach and a conformal mesh. In this experiment, one half of the domain is meshed five times more finely than the other, as illustrated in Figure 2.17. We compare the TFEM approach to classical FEM, where a mesh with a transitional region is considered.

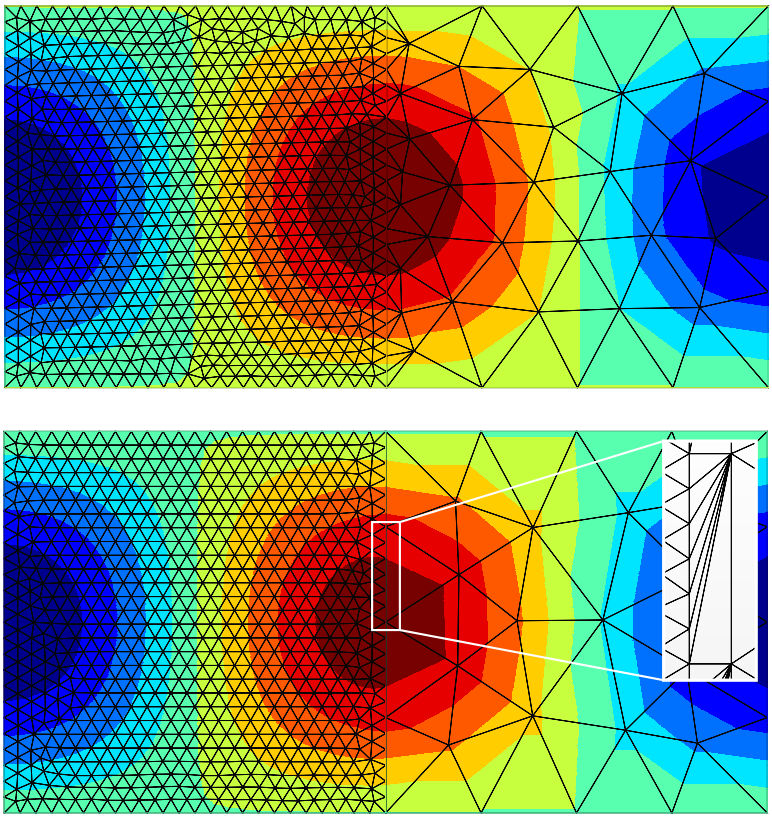


Figure 2.17: Mesh and solution for a classical FEM approach (top) and with the TFEM approach (bottom).

The split view of the mesh in Figure 2.17 (right) demonstrates how the two parts of the mesh are connected using degenerate elements. These are collapsed to have exactly zero width $\bar{h}$ and treated using TFEM. These elements are generated in a straightforward manner, ensuring simplicity of implementation.

A good convergence rate is achieved for both types of meshes, and the error of our approach is very similar to that of a conforming mesh as we can see on Figure 2.18. Unlike some mortar methods, TFEM does not require

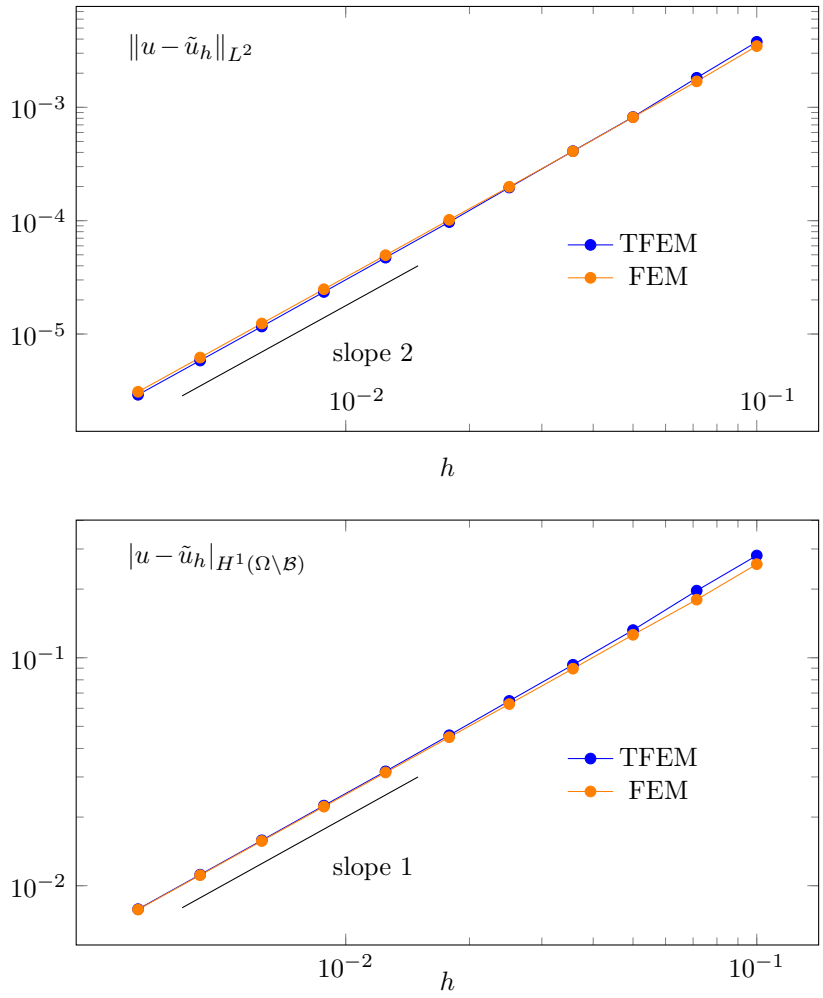


Figure 2.18: Convergence of the error in the L^2 norm (left) and the H^1 semi-norm (right) for the TFEM mortaring of two meshes and the classical FEM with a transitional mesh.

distinguishing between the master and slave sides of the mortaring interfaces, nor does it rely on a specialized integration scheme along those sides. This removes several geometric and algorithmic complications typically associated with mortar formulations. As a result, mortaring with TFEM remains straightforward and robust, even in the presence of intricate or highly irregular domain decompositions.

2.4.3 High order elements

To ensure the method's versatility, it is crucial to determine whether the TFEM is still effective with high-order elements. In this section, we investigate the convergence behavior of high-order elements and examine the impact of the first and second derivatives on the optimal choice of $J_{\min}$.

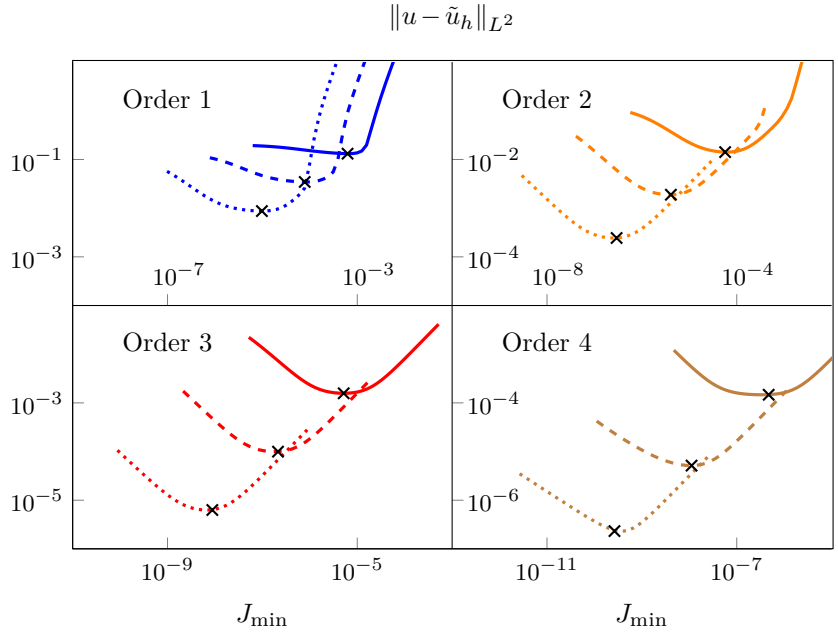


Figure 2.19: Sensitivity analysis of the choice of C for the L^2 semi-norm for high order elements in 2D with $h = \frac{1}{20}$ (solid line), $h = \frac{1}{40}$ (dashed line) and $h = \frac{1}{80}$ (dotted line).

Figures 2.19 and 2.20 illustrates the sensitivity of the L^2 and H^1 errors to the choice of $J_{\min}$ for elements of orders 1 through 4. The error behavior is consistent across different orders, with a notable plateau observed in the H^1 error even for higher-order elements. Black crosses indicate a heuristically fitted choice of $J_{\min} = Ch^{(2.2+0.8\,\text{order})}$. As we can see on these graphs, this formula provides an effective selection for $J_{\min}$.

As illustrated in Figure 2.21, this choice of $J_{\min}$ preserves the good convergence rate of high-order elements in 2D. The solution obtained with our TFEM approach for a degenerate mesh (solid line) is compared to the classical solution on a regular mesh (dashed line). For both the L^2 and H^1 errors, our approach performs comparably to the method on a regular mesh.

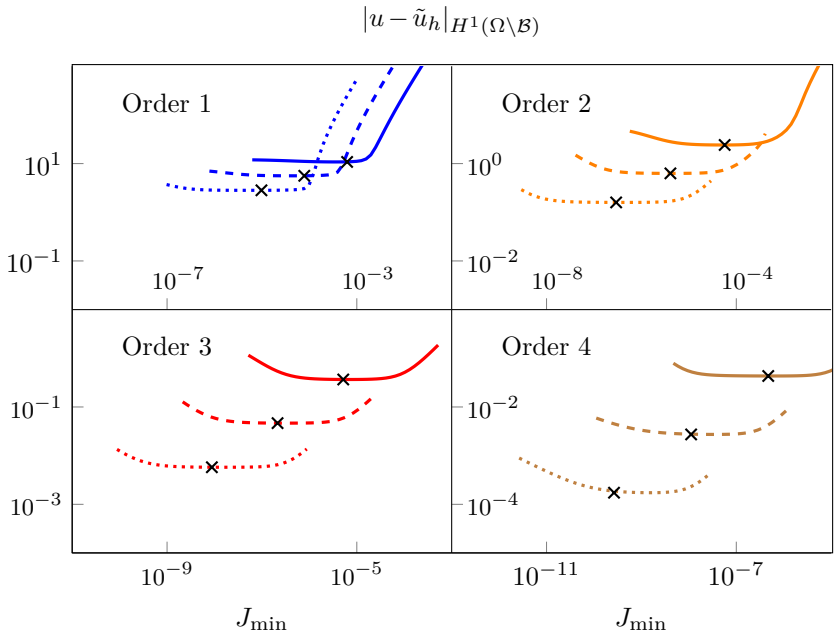


Figure 2.20: Sensitivity analysis of the choice of C for the H^1 semi-norm for high order elements in 2D with $h = \frac{1}{20}$ (solid line), $h = \frac{1}{40}$ (dashed line) and $h = \frac{1}{80}$ (dotted line).

2.4.4 Advection

So far, we were focused on diffusion operators. In this section, we briefly investigate advection. The advection equation is known to pose stability challenges when using continuous finite elements, which necessitates the implementation of stabilization techniques. To address this, we apply the Streamline Upwind/Petrov-Galerkin (SUPG) method. The stabilized advection equation weak form is to find ϕ satisfying (2.48) for all ψ:

$$\int_\Omega \mathbf{v} \cdot \nabla\phi \, (\psi + \tau_{\text{SUPG}} \mathbf{v} \cdot \nabla\psi) \, \mathrm{d}x \tag{2.48}$$

Let us note that for this equation the TFEM approach only needs to be applied on the SUPG stabilization term and not the advection one. Indeed, when the advection local matrix is computed, the determinant J of the transformation between the reference element and the degenerate element appears in the denominator for the computation of the gradient of the shape function and in the numerator for the computation of the integral. They cancel out and no J is left in the final advection matrix.

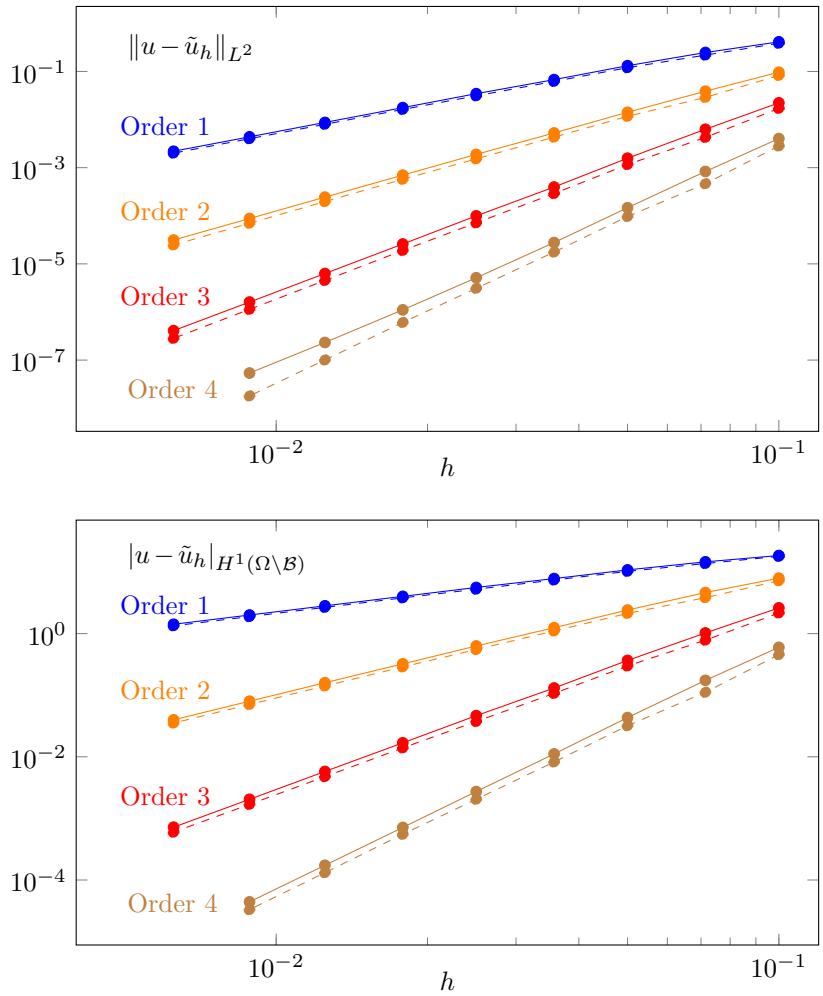


Figure 2.21: Convergence of the error in the L^2 norm (left) and H^1 semi-norm (right) for high-order elements in 2D for the classic FEM on a regular mesh (dashed line) and the TFEM approach with a band of caps (solid line).

We consider a test case with a circular velocity field, denoted as $\mathbf{v}$. The boundary conditions for this case correspond to a Dirichlet condition on the boundary Γ_D defined by a smoothstep function as we can see in Figure 2.22.
We test the advection problem on two types of meshes: a regular reference mesh and a mesh containing a band of caps, where the TFEM approach is applied. The convergence of the error is analyzed by plotting the error as a

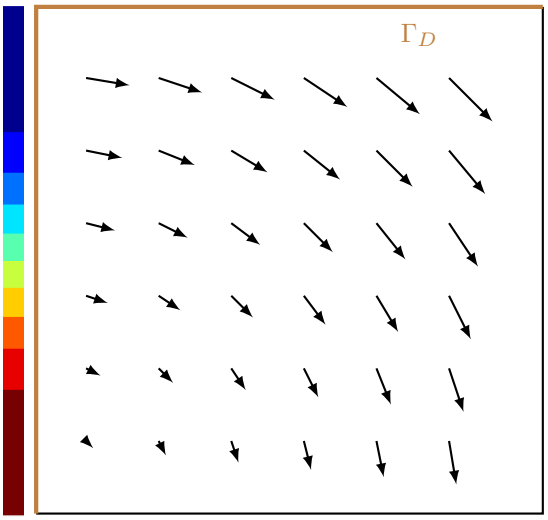


Figure 2.22: Advection problem set up. The arrows represent the advection direction, and the dirichlet boundary Γ_D is highlighted in brown.

function of mesh characteristic length h. The results, shown in the convergence graph of Figure 2.24 (left), indicate that the error converges at the expected rate for both L^2 and H^1 norms, validating the accuracy and robustness of the TFEM approach when applied to advection problems.

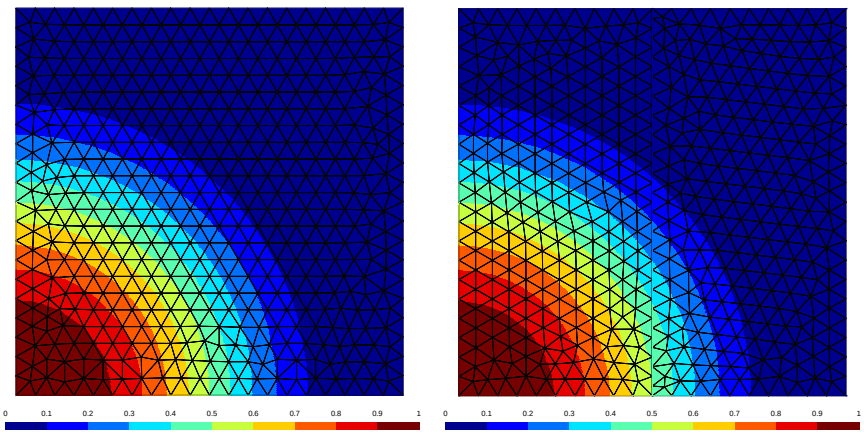


Figure 2.23: Solution to the advection problem on a regular mesh (left) and on a mesh with a band of caps using TFEM (right).

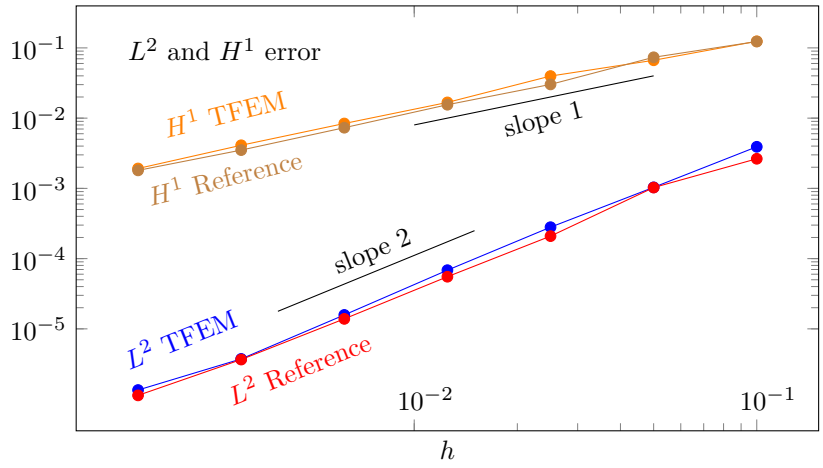


Figure 2.24: Convergence of the error in the L^2 and H^1 semi-norm in 2D.

2.5 Conclusion

In this work we have developed a new method called the Tempered Finite Element Method (TFEM), which increases the flexibility of finite elements with respect to meshing. The method is simple to implement within any finite element code: it simply consists in avoiding division by (almost) zero in the evaluation of FEM matrix entries. We have shown through numerical experiments that on meshes with degenerate elements the method provides an error close to the one of finite elements on regular meshes and that it converges at the same rate, and we have also proven this convergence theoretically for Poisson's problem and linear elements. This method is however not only useful for solving a Laplacian. It works just as well for linear elasticity, advection, and high-order elements and advection. For exactly zero-measure elements, we show that TFEM corresponds to non-conforming mesh mortaring.

A crucial point of the scheme is the choice of a parameter called $J_{\min}$, which is the minimal value of element Jacobians that we are willing to tolerate in denominators when evaluating FEM matrix entries. Our numerical experiments and the mathematical theory have identified the dependence of $J_{\min}$ on h, the mesh size. However, the optimal practical choice of C in the expression $J_{\min} = Ch^{d+1}$ in $\mathbb{R}^d$ remains to be determined. The mathematical study of the scheme provides a theoretically optimal choice of C. However, we have observed numerically that it was not always the best possible in practice. Nevertheless, the sensitivity analysis shows that the method is robust as the choice of C could vary by at least an order of magnitude around the minimum without being too far from the classical finite element error. Although the choice of the optimal C should be the subject of further research, the method can already be used. We show this by performing 100 tests on random functions consisting of a sum of linear, quadratic, sine and cosine functions of random amplitude and frequency, and a Gaussian function of random amplitude and position. The domains considered were rectangles of random sizes containing a band of caps of variable length and we considered the simple choice $J_{\min} = h^3$, i.e. $C = 1$. The relative errors in the L^2 norm and in the H^1 semi-norm are shown in Figure 2.25. It can be seen that, despite the simple choice of $C = 1$, the error of the TFEM method is close to the error of the FEM method on a regular mesh. In the H^1 semi-norm, both errors are almost identical.

The TFEM method could save considerable time and effort in mesh generation by reducing the mesh quality required to perform finite element analysis.

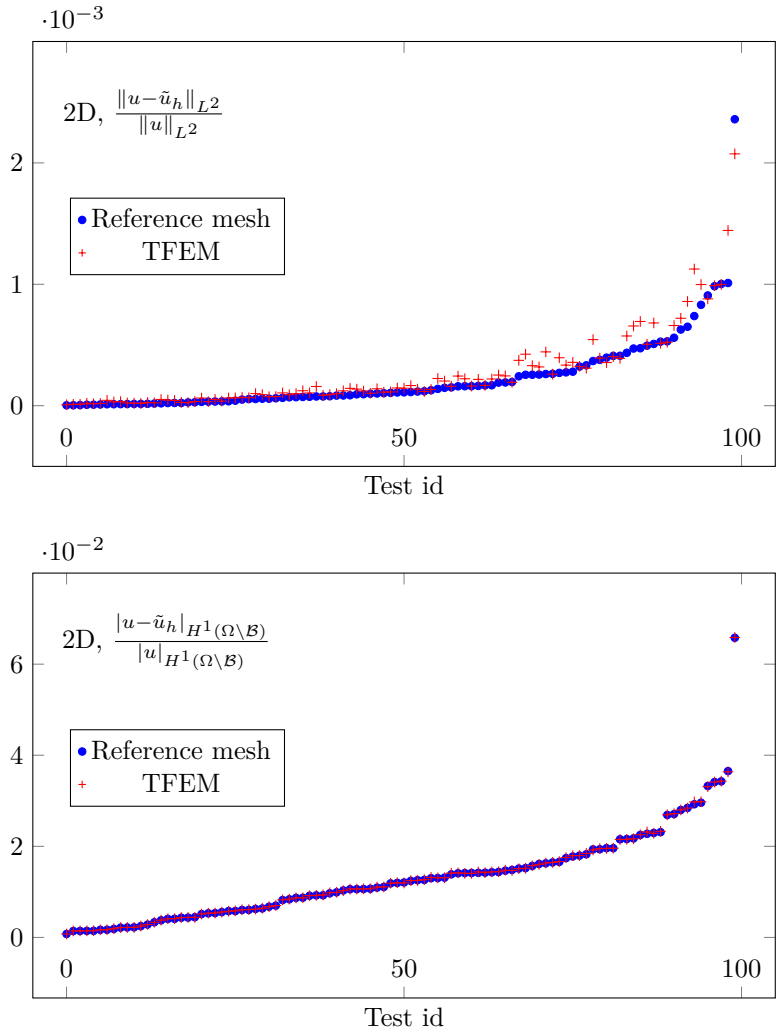


Figure 2.25: Relative L^2 and H^1 error for a reference mesh and a band of caps with the TFEM approach for random tests.

2.5.1 Discussion on $J_{\min}$ and dimensional analysis

We have seen that the choice of $J_{\min}$ is important for the accuracy of the method and even if the method is robust with respect to this choice, it is still a user-defined parameter. At first glance, the choice $J_{\min} = Ch^{d+1}$ in $\mathbb{R}^d$ may seem wrong in terms of dimensions. Indeed, the Jacobian J has the dimension of a volume, i.e. $[L]^d$, while h^{d+1} has the dimension $[L]^{d+1}$. This is due to the

fact that C, despite being mesh independent, is not solution independent and has the dimension $[L]^{-1}$. In the theoretical study of the method, we have seen that C depends on the solution u and its derivatives. An optimal choice of C in 2D is given by (2.34), which we rewrite here:

$$C_{\text{opt}} = \frac{|u|_{W^{2,\infty}(\mathcal{B})}}{|u|_{W^{1,\infty}(\mathcal{B})}} \sqrt{\frac{3C_c}{8}}$$

where C_c is a constant independent of u and the mesh.

As we can see the ratio $\frac{|u|_{W^{2,\infty}(\mathcal{B})}}{|u|_{W^{1,\infty}(\mathcal{B})}}$ has the dimension $[L]^{-1}$, which makes $J_{\min}$ have the correct dimension $[L]^d$. This ratio is also important if we decide to change the units of the problem. For example, if we change the length unit from meters to millimeters, the solution obtained needs to be the same. This the case, if we choose $J_{\min} = Ch^{d+1}$ with C given by the above formula. Indeed, if we change the length unit by a factor α, then h becomes αh, $|u|_{W^{1,\infty}(\mathcal{B})}$ becomes $\alpha^{-1}|u|_{W^{1,\infty}(\mathcal{B})}$ and $|u|_{W^{2,\infty}(\mathcal{B})}$ becomes $\alpha^{-2}|u|_{W^{2,\infty}(\mathcal{B})}$. Therefore, C becomes $\alpha^{-1}C$ and $J_{\min}$ becomes $\alpha^d J_{\min}$, which is the correct scaling for a volume. Figure 2.26 shows the results for our manufactured poisson problem when the length unit is changed from meters to centimeters. We can see that the obtained solution are the same, which confirms that the method is independent of the units of the problem.

Of course, in practice we do not know the solution u and its derivatives, so we cannot use the above formula to compute C directly. However, a user can estimate the ratio $\frac{|u|_{W^{2,\infty}(\mathcal{B})}}{|u|_{W^{1,\infty}(\mathcal{B})}}$ based on the expected behavior of the solution. For example, if we expect the solution to be almost linear, then this ratio will be small and C will be small. If we expect the solution to be very oscillatory, then this ratio will be large and C will be large. The choice of $C = \frac{1}{\lambda}$, where λ is the expected wavelength of the solution, is thus a good starting point. Another possibility is to use the numerical results obtained with a guessed value of C to estimate the ratio $\frac{|u|_{W^{2,\infty}(\mathcal{B})}}{|u|_{W^{1,\infty}(\mathcal{B})}}$ and then update C accordingly. This can be done iteratively until a satisfactory error is obtained. Let's also recall that even-though the mathematical analysis provides a theoretically optimal choice of C, we have seen in practice that it was not always the best possible. This results can be explained by the fact that the mathematical analysis is bounding error terms and thus is not exact. When doing the analysis of the optimal choice of C, we try to minimize the error bound, which is not the same as minimizing the error itself. This is why we have seen that a choice of C slightly different from the theoretical one could provide better results in practice. A deeper analysis

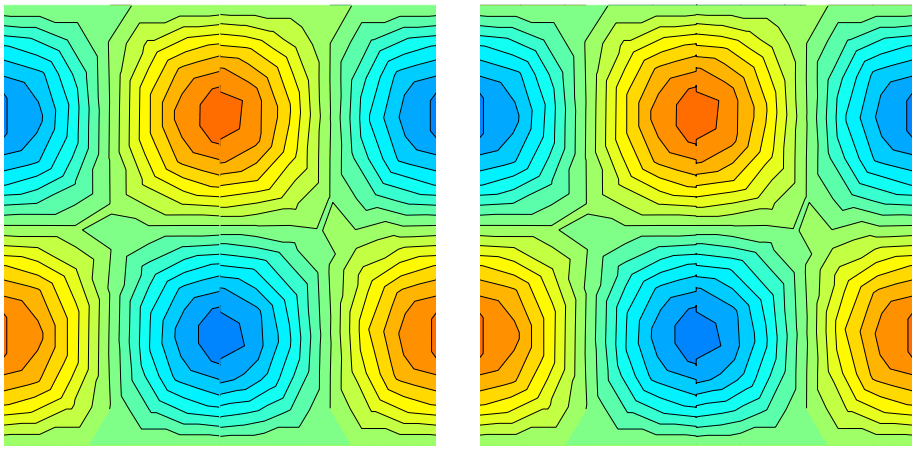


Figure 2.26: Solution of the manufactured poisson problem when the length unit is changed from meters (left) to centimeters (right).

of the optimal choice of C through numerical experiments would be interesting for future work, providing a more practical guide for users of the method.

CHAPTER 3

Interface representation and mesh deformation algorithm

Contents

3.1 The level set approach 74
3.1.1 Interface representation and equations 75
3.1.2 X-Mesh deformation algorithms 77
3.1.3 Front relaying 78
3.2 The front-tracking approach 83
3.2.1 Interface representation and equations 84
3.2.2 Choosing the phase of the triangles 88
3.2.3 Control of the features size 91
3.2.4 X-Mesh deformation algorithms 93
3.2.5 Defining phase of the computational mesh 96
3.2.6 Boundaries management 99
3.2.7 A Winslow energy for mesh displacement in the bulk 100

In this section we present the different 2D mesh deformation algorithms developed for the X-Mesh method in the context of two-phase flows. This corresponds to the first objective of this thesis: to develop a robust and efficient mesh deformation algorithm for the X-Mesh method. This algorithm needs to satisfy several criteria: robustness, efficiency, ability to handle large deformations and topological changes, locality and continuity of the mesh deformation and finally the ability to avoid locking patterns in the mesh.
We first present the level set based algorithm, which is robust and can handle large deformations and topological changes. It is very easy to implement and can be used as a baseline for the X-Mesh method. However, this algorithm does not allow one to control the topology of the interface and topological changes are automatically performed when the interface is one element wide. This algorithm can be sufficient for many applications, but in some cases it is desirable to control the topology of the interface and delay topological changes as much as possible. We thus present a second algorithm based on a front-tracking description of the interface, which allows to control the length scales at which topological changes occur. This algorithm is more complex to implement and requires a good handling of the front-tracking interface, but allows to improve significantly the accuracy of the simulation of two-phase flows with topological changes.
Let us note that both algorithms presented below were designed to avoid the creation of locking patterns in the mesh during deformation. This may seem contradictory to the last chapter, where we showed how the TFEM can handle locking patterns. The main reason is that these deformation algorithms were developed prior to the TFEM, and we therefore sought to avoid locking patterns as much as possible in order to ensure the stability of the simulations. Nevertheless, avoiding locking patterns remains good practice to improve the accuracy of simulations. It would be interesting, in future work, to explore a more Lagrangian deformation algorithm in which locking patterns cannot be avoided, and to examine how the TFEM handles them.

3.1 The level set approach

Accurately capturing the motion of interfaces is a central challenge in computational physics. Among the various approaches, the level set method, introduced by Osher and Sethian [Osher and Sethian (1988)], offers a particularly elegant and robust solution. This method describes the interface as the zero isocontour of a higher-dimensional scalar field, typically a signed distance function. One of its key strengths lies in its implicit treatment of topological changes: phenomena such as interface merging or breakup are naturally handled without additional tracking or mesh reconstruction. In contrast, explicit methods [Tryggvason et al. (2001); Afanador et al. (2021); Prosperetti and Tryggvason

(2009)] often require sophisticated algorithms to detect and resolve such events.

A rich body of literature has developed around applying the level set method to complex two-phase flow scenarios. Early foundational works include Sussman et al. [Sussman and Fatemi (1999)] and Chang et al. [Chang et al. (1996)], who advanced interface capturing techniques by integrating volume-of-fluid formulations to improve mass continuity and geometric fidelity. Olsson and Kreiss (2005) [Olsson and Kreiss (2005)] proposed a conservative level set method designed to better preserve volume during multiphase simulations. More recently, Abu-Al-Saud et al. [Abu-Al-Saud et al. (2017)] developed a sharp-interface level set formulation incorporating moving contact lines, achieving machine-precision force balances and demonstrating convergence properties in capillary-influenced flow regimes

3.1.1 Interface representation and equations

In the level set framework, the interface Γ (a curve in 2D) is represented as the zero level set of a scalar field $\phi(\mathbf{x})$, typically defined as the signed distance to Γ:

$$\phi(\mathbf{x}) = \begin{cases} -d(\mathbf{x},\Gamma), & \mathbf{x} \in \Omega^- \\ d(\mathbf{x},\Gamma), & \mathbf{x} \in \Omega^+ \\ 0, & \mathbf{x} \in \Gamma \end{cases}$$

where $d(\mathbf{x},\Gamma)$ is the Euclidean distance from point $\mathbf{x}$ to the interface. This representation allows for straightforward computation of geometric quantities: the interface normal is given by $\mathbf{n} = \nabla\phi/|\nabla\phi|$, and the mean curvature by $\kappa = \nabla \cdot \mathbf{n}$. When the interface evolves under a velocity field $\mathbf{v}$, its motion is governed by the level set transport equation:

$$\frac{\partial \phi}{\partial t} + \mathbf{v} \cdot \nabla\phi = 0 \tag{3.1}$$

As for the resolution of the Navier-Stokes equations, we stabilize the advection via an SUPG term [Brooks and Hughes (1982)]. The equation is solved on a fixed mesh; we therefore drop the index notation for the computational domain Ω and the time integration is solved by a Crank-Nicolson method. Let $\mathbf{S}_\phi$ be the solution space of ϕ_{n+1} and $w_{n+1} \in \mathbf{V}_\phi$ be the test function associated to ϕ_{n+1}. The discrete formulation can then be written as follows:

Find $\phi_{n+1} \in \mathbf{S}_\phi$ such that for any $\psi \in \mathbf{V}_\phi$:

$$\begin{aligned} &\int_\Omega (\phi_{n+1}\,\psi - \phi_n\,\psi)\,\frac{1}{\Delta t} d\Omega + \frac{1}{2}\int_\Omega (\mathbf{u}\cdot\nabla\phi_n\,\psi + \phi_n \nabla\cdot\mathbf{u}\,\psi)\,d\Omega \\ &+ \frac{1}{2}\int_\Omega (\mathbf{u}\cdot\nabla\phi_{n+1}\,\psi + \phi_{n+1}\nabla\cdot\mathbf{u}\,\psi)\,d\Omega + \mathrm{SUPG} = 0 \end{aligned}$$

The SUPG stabilization term depends on the residual of the advection equation and is given by:

$$\text{SUPG} = \int_{\Omega} \tau_{\text{SUPG}}\, \mathbf{u}\cdot\nabla\psi\, \mathcal{R} d\Omega$$

$$\mathcal{R} = \left((\phi_{n+1} - \phi_n)\frac{1}{\Delta t} + \frac{1}{2}(\mathbf{u}\cdot\nabla\phi_n + \phi_n\nabla\cdot\mathbf{u}) + \frac{1}{2}(\mathbf{u}\cdot\nabla\phi_{n+1} + \phi_{n+1}\nabla\cdot\mathbf{u}) \right)\psi$$

The finite element method then allows us to spatially discretize the domain and we obtain a system of linear equations. Although any Lipschitz-continuous function can serve as a level set function, using the signed distance function improves both numerical stability and the accuracy of curvature and normal vector computations. However, this property is generally lost during evolution. Additionally, because we do not include mesh displacement in the advection solver, after moving the mesh with the deformation algorithm presented below the level set values at the nodes are not correct.

To restore the signed-distance property, we introduce a reinitialization procedure. Sussman et al. [Sussman and Fatemi (1999)] proposed to proceed by solving a reinitialization equation:

$$\frac{\partial\phi}{\partial\tau} + \text{sign}(\phi_0)\,(|\nabla\phi| - 1) = 0 \tag{3.2}$$

where τ is an artificial time, and ϕ_0 is the level set function before the reinitialization procedure. This PDE drives ϕ back toward a signed distance function near the interface.

A highly efficient alternative to PDE-based reinitialization is the Fast Marching Method. Initially developed by James Sethian, this algorithm solves a boundary-value version of the Eikonal equation ($|\nabla u(\mathbf{x})| = 1$) with specified boundary values on the interface, effectively propagating "arrival times" outward to reconstruct the distance field. In the level set context, the fast marching method is used to reset the function ϕ to a signed distance field by marching from the zero-level set outward through the domain. Fast marching offers substantial computational savings over iterative PDE methods and provides sufficiently accurate geometry for curvature and normal computations, making it widely adopted in two-phase flow schemes where interface fidelity and efficiency are both critical.

The Fast Marching algorithm is particularly well suited to the X-Mesh approach because, in X-Mesh, the mesh nodes themselves lie exactly on the interface, meaning the $\phi = 0$ points are already part of the nodal set. This makes it straightforward to define the seed nodes for the Fast Marching computation,

as the initial set of values used for propagation can be taken directly from the interface nodes without interpolation. In contrast, when the zero level set does not coincide with mesh nodes, the seeds must be determined by locating and interpolating along element edges, which complicates the initialization and can introduce additional geometric approximation errors. With X-Mesh, this seeding step is both direct and accurate, allowing the Fast Marching algorithm to be applied efficiently and reliably.

3.1.2 X-Mesh deformation algorithms

The idea behind the X-Mesh approach [Moes et al. (2021)] is to deform a mesh with continuous node movements to continuously match the interfaces of interest, even when the fluid domains undergo topological changes. The mesh however keeps a fixed topology. To achieve this goal, the approach allows elements to become degenerated, meaning that a triangle can deform down to an edge or even a point. This enables the mesh to deform continuously in time and ensure the relay of the front. The interface is transferred from one node to another located at the same position allowing the interface to propagate like a baton in a relay race.

As we have seen in the introduction, there are two ways for a triangle to degenerate into an edge. In the first case, one of the three edges of the triangle collapses to a single point so the opposite angle to this edge degenerates to 0 radians. Such an element is called *a needle*. In the second case, one of the three vertices of the triangle tends to a point that belongs to its opposite edge so the angle associated with this node tends to π radians. Such an element is called *a cap*. The following algorithms are designed to avoid the creation of bands of caps, which can lead to locking patterns in the mesh as explained in Chapter 2.

Our mesh deformation/relaying algorithm has, from the beginning, taken this constraint (avoiding bands of caps) into account. Assume a triangular mesh that exactly represent a front at a given time t (Left images of Figure 3.1). The front moves to the right with a constant speed v during a time Δt and thus we can move front vertices to the right of $v\Delta t$ (Top-center image of Figure 3.1) and then of $v\Delta t$ again (Top-right image of Figure 3.1). At that point, one sees that the relay will essentially occur with triangles that have the form of a cap and that caps are arranged in bands. Such an algorithm clearly leads to locking problems and should be avoided. In general, if we are not careful, a mesh deformation algorithm will naturally tend to create these bands of caps.

What if now mesh vertices move along edges of the current mesh (Bottom images of Figure 3.1). Then, a majority of needles will be created. Isolated caps may be created but their number only depends on the difference between the number of vertices up-stream and downstream the front. Another ingredient of

the algorithm is to move the vertices only upstream of the front. Finally, when a vertex of the front has passed the relay to another vertex, it progressively returns to its initial position which allows to maintain a quality mesh upstream of the front. These nodes are moved by simply following the equation

$$\mathbf{x}_{n+1} = (1-\alpha)\mathbf{x}_n + \alpha\mathbf{x}_0$$

where $\mathbf{x}_n$ is the position of the node at time t_n, $\mathbf{x}_0$ its initial position and $\alpha \in [0,1]$ a parameter that controls the speed at which the node returns to its initial position.

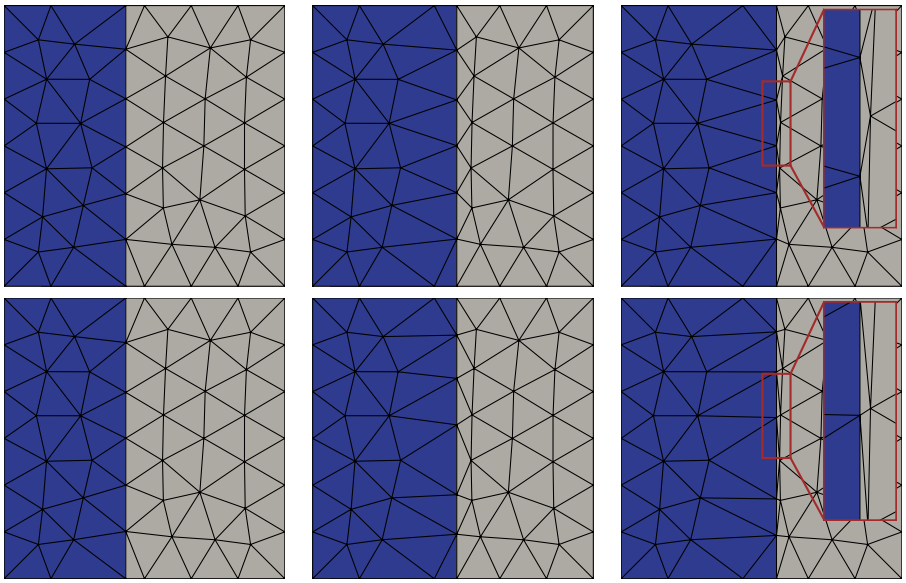
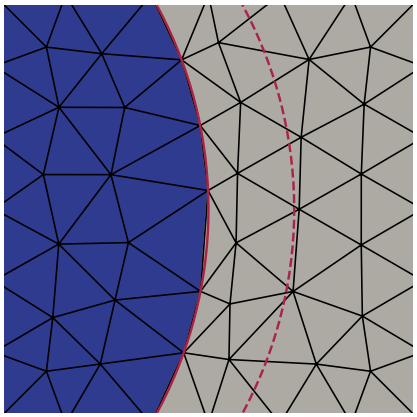


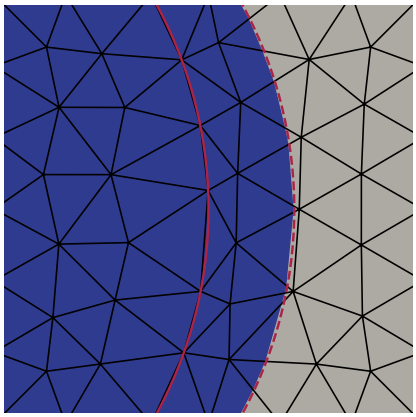
Figure 3.1: Moving the nodes along their velocity creates caps (top) and moving along the edges creates needles and isolate caps (bottom).

3.1.3 Front relaying

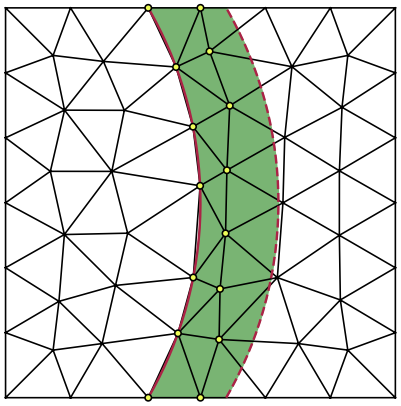
Figure 3.2 shows the algorithm used to deform the mesh between 2 time steps so that it continuously conforms to the new interface. This interface is defined by $\phi = 0$ and the sign of ϕ determines the phase of the fluid. On images (a) and (b) of Figure 3.2, the phases of the fluids for 2 consecutive time steps t and $t+\Delta t$ are represented in blue and grey. The idea of X-Mesh is that the mesh at the two times t and $t+\Delta t$ must conform to the interface. Image (b) shows the phases at time $t+\Delta t$ on the mesh at time t. The new interface at time $t+\Delta t$, shown with dashed lines, is not conforming to the mesh.



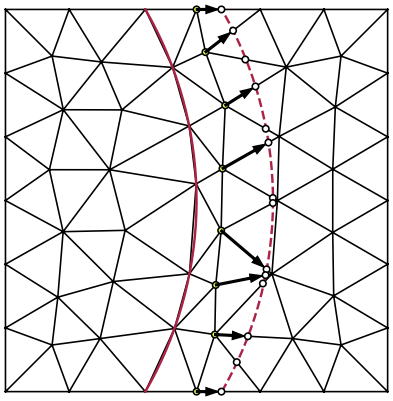
(a) Phases at time t



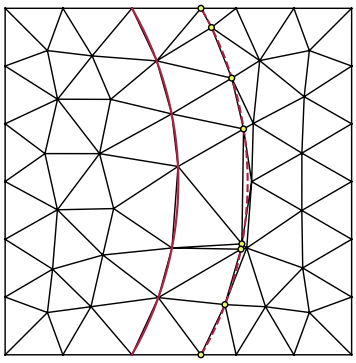
(b) Phases at time $t + \Delta t$



(c) Active vertices



(d) Potential targets and moving vertices



(e) New mesh and new front vertices

Figure 3.2: Mesh deformation algorithm.

The objective of the front relaying algorithm is to deform the existing mesh at time t so that the mesh is conformal to the new interface: we want the interface at time $t+\Delta t$ to be entirely represented by the edges of the mesh. Additionally, the algorithm must limit as much as possible the creation of bands of caps for stability reasons. As mentioned above, moving the vertices along the edges of the mesh allows to avoid the creation of bands of caps. We also want to avoid vertices to accumulate downstream the front. Thus, only nodes upstream of the front should move towards the front. This starting point allows us to develop a robust front relaying algorithm. The region in green of image (c) of Figure 3.2 represent the region where phases have changed between t and $t+\Delta t$. This region is thus upstream of the front and vertices that move should be chosen in this region. We call *active vertices* the vertices for which the sign of the level set ϕ has changed during the time step. Active vertices include vertices of the previous front which had a level set value of 0 and are coloured in yellow in image (c) of Figure 3.2.

An edge (ij) is a cut-edge if, at time step t, its two nodes have a different sign for the level set ϕ at time $t+\Delta t$. There is therefore a potential target on the cut-edge (ij) where the level set is null, i.e. where the new front will be located at $t+\Delta t$. All active vertices that belong to cut-edges will move: we call them moving vertices (see image (d) of Figure 3.2). Now a moving vertex can belong to more than one cut-edges so we have to choose one of the potential targets. The moving vertices at the boundary of the domain (the highest and lowest in Figure (4d)) has to be moved on a potential target located on the boundary to preserve the shape of the computational domain. Note that only polygonal computational domains are considered in this work, non-trivial geometries would require special treatment. For the moving vertices inside the domain, we experimentally observed better mesh quality when we chose the potential target that is the closest to the node (arrows in Figure (4d)). Image (e) of Figure 3.2) shows mesh at time $t+\Delta t$ that is conforming to the new interface.

This mesh deformation method is presented in Algorithm 1. In this algorithm, *level_set* and *previous_level_set* correspond to the new and old values of the level set. A table called *front* is used to store a boolean for each node to indicate whether it was on the previous interface or not. A second table, *on_bnd*, is used to specify if the node is on the boundary of the domain and then requires extra attention.
If the node is not on the boundary, the function *move_bulk* is called, in which we find all the neighbours of node i, through the function *get_neighbours*. We can then compute the position of the closest target and the associated neighbour with the function *determine_closest_target(i, neighbours)*. These targets correspond to the position of the 0-value level set on the edges of the mesh between the node i and its neighbouring nodes.
In the case of the boundary nodes, we call *move_boundary* and we have to

distinguish between the neighbours on the boundary and the neighbours in the bulk. If possible, we place the node i on a target positioned on the boundary to preserve the domain boundaries. If this is not possible, we look at the neighbours inside the bulk and calculate the closest target. The corresponding neighbour is then placed on that target.

Algorithm 1: Move front algorithm

```
/* The function determine_closest_target(i,neighbours) returns
   the position of the closest level set 0-value target on the
   edges between node i and the neighbours.                    */
Function move_front:
    foreach node i at x_i do
        if front[i] is True and sign(previous_level_set[i]) ≠
         sign(level_set[i]) then
            active[i] ← True
    foreach node i at x_i do
        if active[i] is True then
            if on_bnd[i] is True then
                move_boundary (i)
            else
                move_bulk (i)

Function move_bulk (i):
    neighbours ← get_neighbours(i)
    x_closest, neighbour ← determine_closest_target(i,neighbours)
    if x_closest is not Null then
        x_i^new ← x_best
        front^new[i] ← True

Function move_boundary:
    bnd_neighbours ← get_bnd_neighbours(i)
    bulk_neighbours ← get_bulk_neighbours(i)
    x_closest, neighbour ← determine_closest_target(i,bnd_neighbours)
    if x_closest is not Null then
        x_i ← x_closest
        front^new[i] ← True
    else
        x_closest, neighbour ←
         determine_closest_target(i,bulk_neighbours)
        if x_closest is not Null then
            x_neighbour ← x_closest
            front^new[neighbour] ← True
```

The phase of each element can then be determined using the level set func-

tion. Because the mesh is conformal to the boundary, the zero iso-value ($\phi = 0$) of the level set is represented by edges joining front nodes. Thus, no mesh edge can cut the front after mesh deformation. In this way, no elements have vertices with a level set value of opposite sign. The phase of each element can thus be obtained from the level set value of its vertices. If the 3 vertices of a triangle are located on the front (and thus have a level set value of 0), the phase of the element is ambiguous. We call these elements triple-zero elements. This occurs when the curvature of the front is important for the characteristic length of the element or when a topological change occurs. A simple way to deal with such elements is to look at their phase at the previous time step. If the element was already a triple-zero element, we keep the same phase as before, so that these elements don't oscillate from one time step to another. If the element was not a triple-zero before, we change the phase of the element. In this way, the merging of the phases can take place.

3.2 The front-tracking approach

The front tracking method is an explicit interface-tracking approach in which the interface between two phases is represented by a set of marker points or a separate, lower-dimensional mesh that moves through the computational domain according to the local fluid velocity. Unlike implicit methods such as the level set or volume-of-fluid formulations, the front is tracked directly, preserving a sharp and well-defined representation of the interface geometry at all times. This approach has been successfully applied to two-phase flow problems and bubble dynamics, enabling accurate predictions of interface curvature, surface tension forces, and topological evolution in systems such as rising bubbles, droplet collisions, and jet breakup [Tryggvason et al. (2001); Unverdi and Tryggvason (1992); Popinet and Zaleski (1999)].

One of the main advantages of front tracking is its high accuracy in computing interface geometry (normals, curvature) since the interface is explicitly represented, avoiding the numerical diffusion seen in many implicit methods. It also allows for precise application of boundary conditions at the interface, which is particularly valuable for modeling surface tension effects. However, the method also has drawbacks: it requires explicit management of the marker mesh, including re-meshing or redistribution to prevent distortion and handling topological changes such as merging or breakup which is non-trivial. These challenges have limited its widespread use compared to fully Eulerian interface-capturing methods, despite its superior accuracy for smooth, well-resolved interfaces.

3.2.1 Interface representation and equations

In 2D, the interface is represented by a set of marker points $\mathbf{x}_i$ that lie on the interface and linked by edges. The position of these markers evolves according to the local fluid velocity $\mathbf{u}$, which is obtained from the solution of the Navier-Stokes equations. The evolution of the marker positions is given by:

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{u}(\mathbf{x}_i) \tag{3.3}$$

This equation is solved using a simple explicit Euler scheme, where the new position of each marker is computed as:

$$\mathbf{x}_i^{n+1} = \mathbf{x}_i^n + \Delta t \cdot \mathbf{u}(\mathbf{x}_i^n) \tag{3.4}$$

where Δt is the time step size. The markers are typically distributed uniformly along the interface to ensure accurate representation of the curvature and surface tension forces. This distribution is crucial since if there are too few markers, the interface cannot be accurately resolved, leading to a loss of detail and potential errors in curvature and surface tension calculations. Popinet and others [Popinet and Zaleski (1999)] have also noted that having too many markers compared to the mesh size for the fluid simulation can cause oscillations of the tracked interface. To avoid this, the marker distribution must be controlled so that there are neither gaps with too few markers nor regions with too many.
In this redistribution process, we simply remove markers that are too close to each other (replacing them with a single marker at their average position) and add new markers in regions where the distance between markers is too large. The effect of this operation is represented at Figure 3.3. The Figure represents the evolution of the interface of a Rayleigh-Taylor instability. The left image is the initial interface, the middle image corresponds to the interface at time $t = 0.9$ without redistribution and the right image is the interface at time $t = 0.9$ with redistribution. We can see that without redistribution, markers accumulate in some regions and are too sparse in others, leading to a poor representation of the interface. With redistribution, the markers are more evenly spaced, resulting in a more accurate interface description.

Another important aspect of the front tracking method is the treatment of topological changes, such as merging or breakup of the interface. We propose to base the treatment of topological changes on the winding number of the interface, which is a topological invariant that counts how many times the interface winds around a point. There exist multiple algorithms to compute the winding number, the simplest one is to compute the angle between the point and each segment of the interface and sum all these angles. The winding number is then given by the sum divided by 2π. In this thesis, we will use a computationally efficient version of this idea, presented in [Hormann and Agathos (2001)].

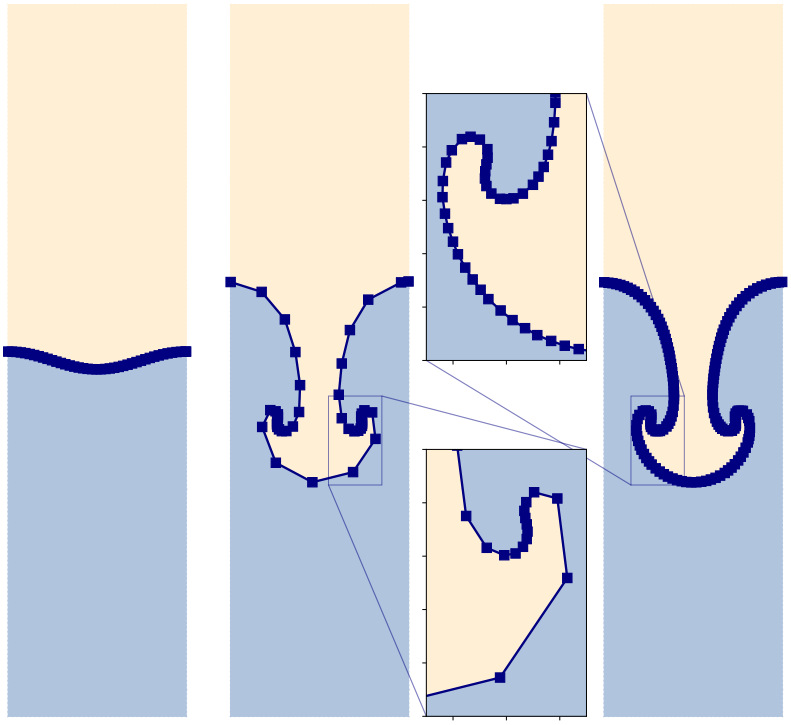


Figure 3.3: Effect of the marker redistribution on the interface representation.Left: initial interface, middle: interface at time $t = 0.9$ without redistribution, right: interface at time $t = 0.9$ with redistribution.

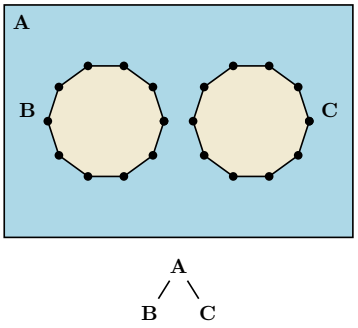
When dealing with multiple interfaces that can be nested or included within one another, the concept of the winding number of a point alone is not sufficient to fully characterize the topology. While the winding number indicates how many times a closed curve wraps around a point, it does not capture hierarchical relationships between interfaces, specifically, which interfaces are contained inside others. To properly represent these nested configurations, it is necessary to build a topological tree structure that encodes the inclusion relationships between interfaces. This tree allows one to identify parent-child relations among interfaces and correctly determine the interior and exterior regions in complex multi-interface domains. Such hierarchical information is essential for accurate numerical treatments of multiphase flows where bubbles or droplets may be enclosed within others.

3.2.1.1 Topological changes

We propose a simple algorithm to detect and handle topological changes in the front tracking method. The algorithm is divided in the 7 following steps.

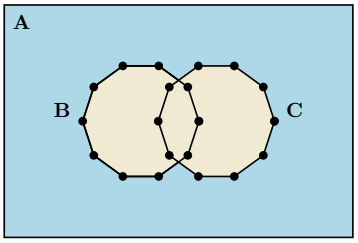
Step 1. Topological tree

We start from a valid (non-self-intersecting) set of interfaces, and construct a topological tree. This tree represents the relationships between the interfaces, where each node corresponds to an interface and edges represent inclusion relationships. The root of the tree is the outermost interface, the domain boundary. This structure allows us to efficiently manage nested interfaces and detect changes in topology.



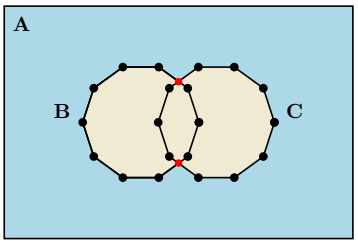
Step 2. Move the markers

We move the markers according to the local fluid velocity, as described in equation 3.4. This step may cause the interfaces to deform and potentially lead to topological changes, such as merging represented on the Figure on the right.

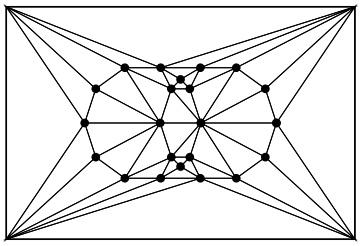


Step 3. Intersect the interfaces

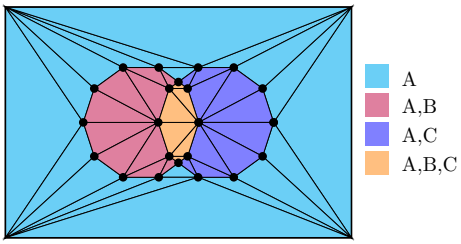
After moving the markers, we check for intersections between the interfaces or auto-intersections within the same interface. This step is performed efficiently using a space-partitioning structure, which allows us to find pairs of segments that are close enough to potentially intersect. Those intersections are added to the set of markers that represent the interface.



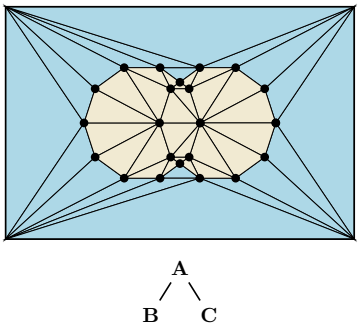
Step 4. Triangulate the markers
In this step we triangulate the marker points while constraining the interface segments to be part of the triangulation. This step is done using previous work developed in Gmsh [Geuzaine and Remacle (2009)].



Step 5. Winding number
For each triangle of the triangulation, we compute the winding number [Hormann and Agathos (2001)] of the center of the triangle with respect to the interfaces. We can color the triangles into different colors as represented in the Figure on the right.

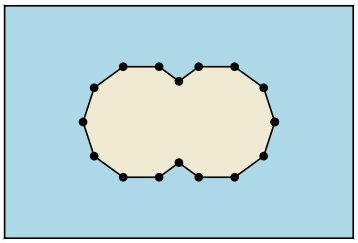


Step 6. Choose phase
We have colored the triangles according to their winding numbers, but we still have to choose which phase corresponds to which color. We propose to use the topological tree to determine the phase of each triangle. We follow the tree from the root to the leaves. If there is no topological change, then we can follow a single path from the root to the last leaf interface. If there is a topological change, this mean that either: 1) the triangle is inside multiple interfaces of the same level or 2) the triangle is inside an interface but not inside his parent interface.



In our simple problem illustrated in the Figures on the right, we are in the case 1) for triangles both in interfaces B and C. In this easy case, the choice is grey since both interface phase is grey. How to choose the phase of the triangles for more complex scenarios are discussed in the next section.

Step 7. Compute new interfaces
Now that each triangle has a phase, we can compute the new interfaces. The new interfaces are defined by the edges of the triangles that have different phases. We can then extract the new interfaces from the triangulation.



3.2.2 Choosing the phase of the triangles

In the previous section, we have described how to manage topological changes in the front tracking method. We also have identified, that topological changes can be detected through the topological tree. However, we have not yet discussed how to choose the phase of the triangles when there is such a change. The choice of the phase is important to ensure that the simulation remains physically consistent and that the interface is correctly represented.

As we said earlier, there are two types of topological changes that can occur. The first type is when two interfaces of the same level merge together as we represented in the figures of the different steps of the algorithm. The second type is when an interface merges with its parent interface. This case is represented in the figures 3.4. For this second type of topological change, we could

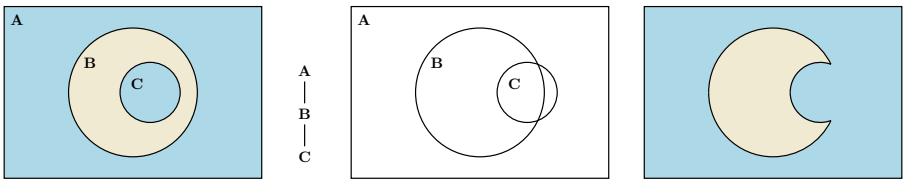


Figure 3.4: Merging of an interface with its parent interface.

also include the case where an interface breaks into two or more interfaces. To simplify the discussion, we will only consider the merging of an interface with its parent interface (and not grand-parent) but the algorithm can be generalized for such case.

Our algorithm to determine the phase of triangle can be summarized in the following 4 steps:

Step 1. We start from our valid topological tree and we will go through the tree from the root to the leaves.

Step 2. For the current level of the tree, we look at the winding number of the triangle for the interfaces of this level. We flag in green the interfaces for which the triangle is inside. An exemple is represented in the Figure 3.5, where an hypothetical triangle is inside the interfaces C1 and C2.

Step 3. We look at the winding number of the triangle for the children interfaces of the non-flagged interfaces. If the triangle is inside one of these children interfaces, we flag it in green. For exemple in Figure 3.5, the considered triangle is not inside the interface C3 but well inside its children C3.1. This situation is demonstrated in Figure 3.6 with the interface and their phases before and after displacement.

Step 4. All the flagged interfaces are considered the new parents of our topological tree and the next level is composed from the children of these flagged interfaces. We then go back to step 2 until we reach the leaves of the tree or until there is no more flagged interfaces. The phase is choosed accordingly to the last parents interface.

We apply those 4 steps iteratively until we reach the leaves of the tree or until there is no more flagged interfaces. For example, in the Figure 3.6, if we apply again the algorithm, we can see that at the next level, the cross is not inside any of the children interfaces. So we stop the algorithm and we choose the phase of the last parents interface, in this case C1, C2 or C3.1. Since C3.1 is the most embedded interface, we choose its phase, which is blue. We can apply this iterative algorithm to all the barycenter of the triangles of the triangulation to determine their phase.



Figure 3.5: Example of a topological tree and the different steps of the algorithm to choose the phase of a triangle.

In some cases the phase of the last parents interfaces are not the same. When

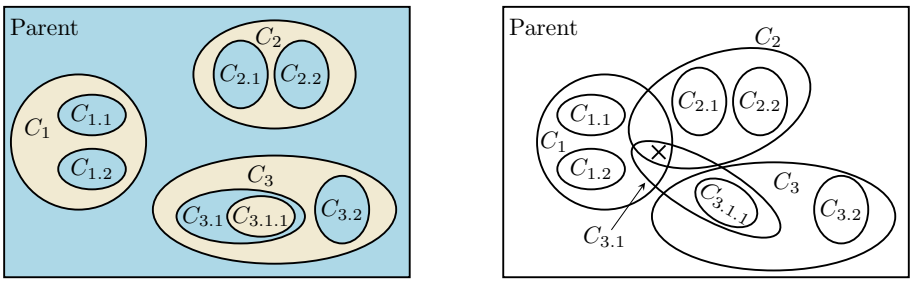


Figure 3.6: Example of interfaces corresponding to the topological tree of Figure 3.5 where we try to determine the phase of the triangle positionned at the black cross.

that happens, we choose the phase of the interface which was the most embedded in the previous topological tree. This is an arbitrary choice and other strategies could be implemented. However, this choice allows to keep small features of the flow that could be lost with other strategies.
The Figure 3.7 shows how a conflict like this can occur. As we can see, after the displacement of the interfaces, if we try to consider the phase at the red dot, we can see that after applying our algorithm the point has the interfaces C and D as last parents. The phase of C is blue while the phase of D is grey. This is a conflict and we need to make a choice. In this case, we choose the phase of C since it was the most embedded in the previous topological tree.
We can apply the same algorithm for complex scenarios with multiple inter-

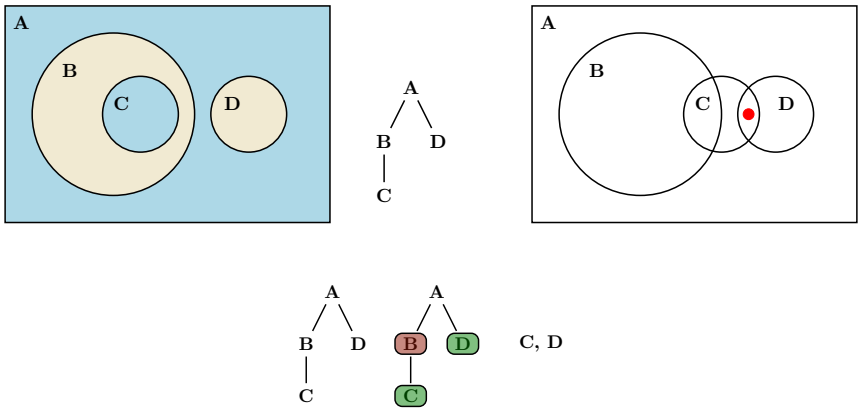


Figure 3.7: Conflict of phase choice.

faces and multiple topological changes. The Figure 3.8 shows such a complex scenario where we can see that the algorithm is able to determine the phase of

the different space regions.

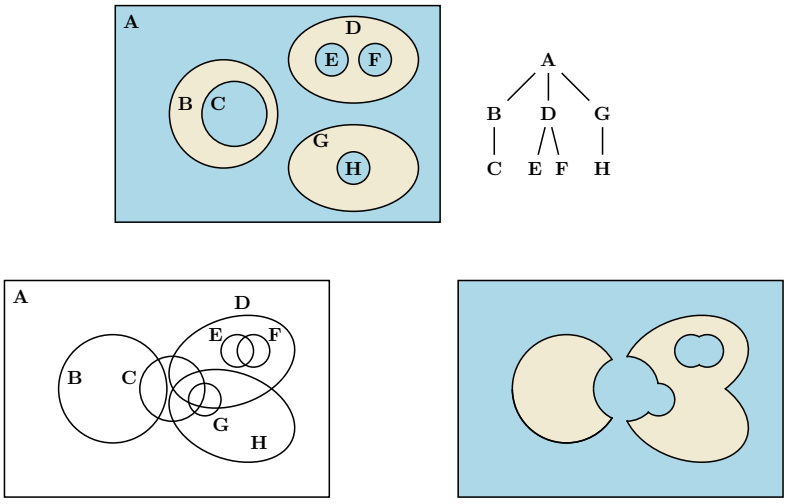


Figure 3.8: Complex scenario with multiple interfaces and multiple topological changes.

3.2.3 Control of the features size

When considering an incompressible flow, topological changes can not occur spontaneously due to the divergence-free condition of the velocity field [Baradat (2019)]. However, experiments and even daily life observations show that topological changes occurs in flows that are in an incompressible regime. Experiments show that before a topological change occurs, the interface develops very thin filaments or films, during an unexpected duration of time before touchdown. Such films are very thin and can be of the order of a few nanometers. This also explains why in some conditions, drops can "bounce off walls" when contact is entirely prevented. An important factor of the breaking of these thin films is the presence of van der Waals forces that become significant at very small scales. These forces can destabilize the thin films, leading to rupture and subsequent topological changes [Chubynsky et al. (2020)].

When considering numerical simulations, the story is different. These van der Waals forces are not taken into account in most of the simulations and the numerical methods have a limited resolution. This means that when a thin film or filament is thinner than the mesh size, capturing interface methods will not

be able to represent it correctly and will create a topological change. Topological changes can also occur for interface tracking approaches but this time this is not due to a lack of resolution but rather due to the accumulation of numerical errors. For example, the PSPG stabilisation method used to solve the Navier-Stokes equations can create small divergence in the velocity field that can accumulate over time or the numerical advection of the markers leads to errors in their position that can induce a topological change.

In function of the physical problem, those topological changes can be either desirable or not. For example, in the case of a droplet impacting a solid surface, it's important to be able to delay the topological change as much as possible since the dynamics of the thin film before touchdown are important. On the other hand, in the case of a dam break, the length scales of the flow are much larger and an accurate representation of thin films is not necessary. In [Afanador et al. (2021)], the authors show that the length scale at which topological changes occur in the context of jet atomisation does not significantly affect some flow quantities such as the phase average and plane average velocities. However, they also show that the impact on the interface position can be significant. In this case, it can be interesting to trigger topological changes earlier to avoid the creation of thin and complex features that could perturb the mesh deformation algorithm presented in the next section.

Since we have developed a robust algorithm to handle topological changes, we will use it to control the size of the features in the interface. To do so, we artificially move the markers a distance δ in the normal direction of the interface, then apply our topological-change algorithm. This operation is done two times (with δ and $-\delta$) and for each phase. This allows to remove features that are smaller than 2δ. The Figure 3.9 shows how this operation works. In this example, we want to remove features smaller than 2δ. We first move the markers of the grey phase in the normal direction of a distance δ and then we apply our topological change algorithm. This allows to remove the blue *filament* between the two grey bubbles. For illustrative purpose the chosen δ is quite large in this example. In practice, we choose δ around a third to a tenth of the mesh size depending on the application.
We then move back the markers to their original position and the new markers created from the intersection are not moved. We apply this step in both normal direction and for both fluids.

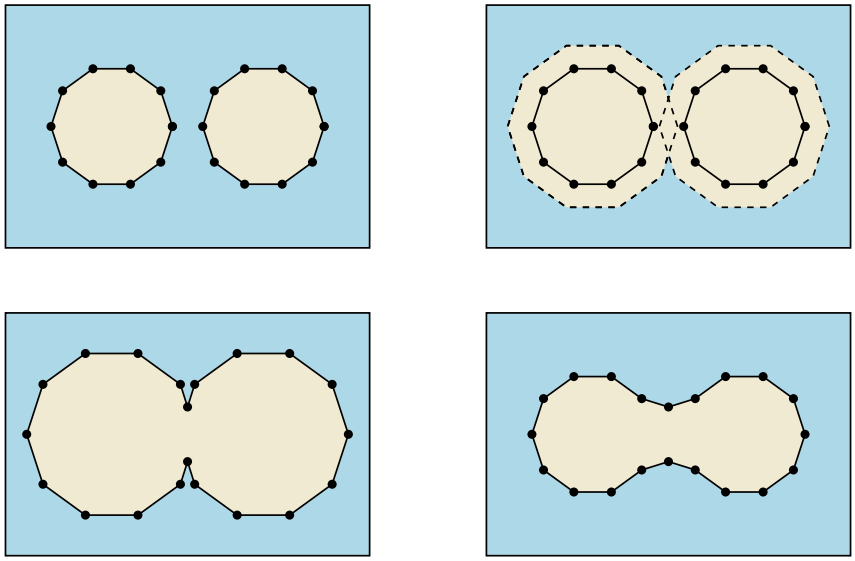


Figure 3.9: Example of feature size control using topological changes. Top left: initial interface with small features, top right: after moving the grey markers by δ, bottom left: after applying topological change algorithm, bottom right: final interface after moving back the markers. The same operation is done for the $-\delta$ direction and for both phases.

Let's note that topological changes in two-phase flows is still an open research topic and there is no consensus on how to best model them numerically. Recently, [Chirco et al. (2022)] has proposed a new approach called the "manifold death" method. This method detect in which regions of the flow (bulk, sheet, ligament, interface, etc.) a certain point is thanks to the signature method. Then, the method initiate the topological change by perforing the interface in regions where thin films or ligaments are detected. The choice of distance at which to perforate the interface is based on physical arguments instead of numerical arguments like the mesh size. This approach helps to obtain a more physically accurate representation of topological changes and improve the overall convergence of the results as the mesh is refined.

3.2.4 X-Mesh deformation algorithms

As we did with the level set method, we can use the front-tracking method and their markers to deform the mesh and try to match this representation of the interface. In the case of the level set method, the interface was represented by the iso-value 0 of the level set directly on the computational mesh. The

size of the features in the interface were thus limited by this computational mesh. This is limiting for capturing smaller features but it also ensures to have a deformed mesh that is exactly conforming to the interface. In the case of the front-tracking method, the interface is represented by a set of markers and those can have features much smaller than the computational mesh. This is interesting when we want to represent small features of the flow, such as filaments, because we can deform the mesh to have near zero-size elements inside those features. However, we cannot ensure that the mesh will exactly match the interface represented by the markers. We can only ensure that the mesh will be close enough to this interface. We will discuss its effect on the simulation in section 3.2.5.

In our mesh deformation algorithm for markers, we start from a mesh and a set of markers representing the interface. An example is represented in Figure 3.10, where we can see a sinusoidal interface. The mesh is not conformal to the interface and the purpose of our algorithm is to deform the mesh locally to be as close as possible to the interface. Note that when deforming the mesh we move only nodes that are allowed to move (e.g., corner nodes remain fixed; boundary nodes move only along the boundary). We also don't allow a movement of nodes that would create inverted elements.

The first step is to deform the mesh to capture 0-dimensional interface fea-

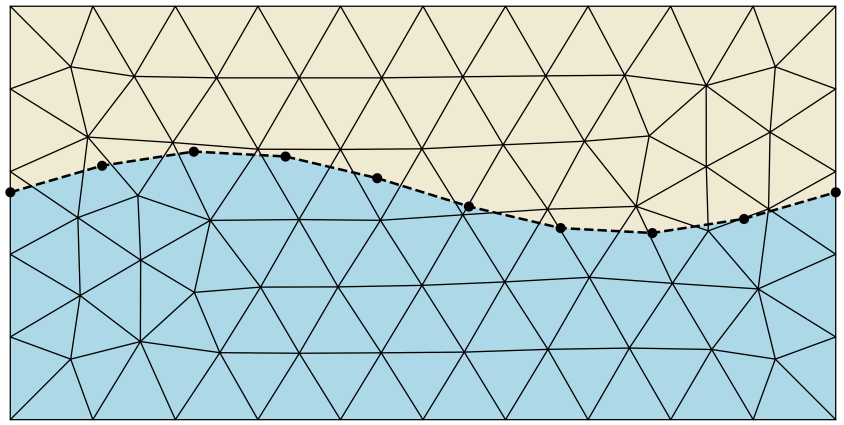


Figure 3.10: Markers representation of a sinusoidal interface and initial mesh.

tures. We locate the triangle containing the feature and move the triangle's closest node onto the feature. In fluid, these 0-dimensional features correspond to the triple-point where three phases meet. In our two-phase flow case, it is the two points where the interface meets the boundary of the domain. This

case is represented in Figure 3.11, where we can see that the two points where the interface meets the boundary are now represented by nodes of the mesh. This allows to have a good representation of these triple-points where important physical phenomena can occur such as wetting and drying.

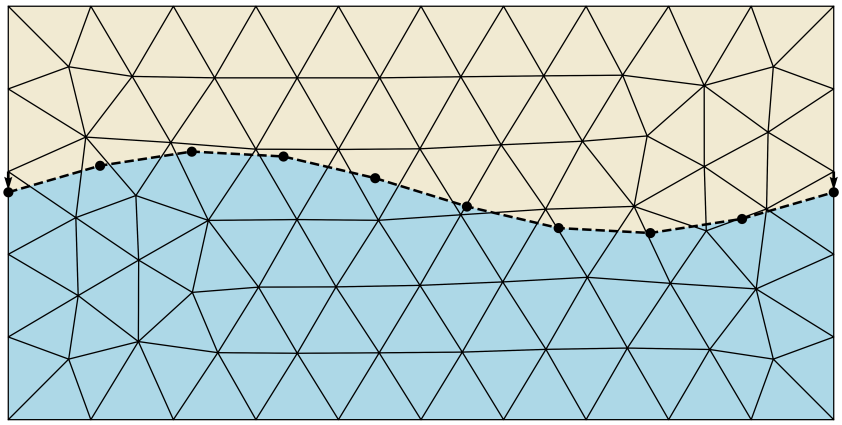


Figure 3.11: Deformation of the mesh to capture 0 dimensional features of the interface represented by markers.

The second step of the algorithm is to compute the intersection between the mesh and the markers. Since we want to move the nodes of the mesh on the interface but we want to avoid creating bands of caps by moving the nodes only along their edges, the potential targets for each node are the intersection points between the edges of the mesh and the segments defined by the markers. This is represented in Figure 3.12, where we can see the intersection points between the mesh and the markers. Each node of the mesh can have multiple potential targets but we only keep the closest one to the node.

The third step consist in moving the nodes of the mesh on one of their potential targets. We order the nodes by their distance to their closest target and we will try to move the nodes from the closest to the farthest. We also only move nodes along an edge that has not been moved yet. This allows to avoid moving multiple nodes on the same intersection and also to add more nodes than necessary on the interface. This step is represented in Figure 3.13, where we can see that the mesh is now conformal to the interface represented by the markers. We iterate between the second and third steps until no more intersection can be found between the mesh and the markers.

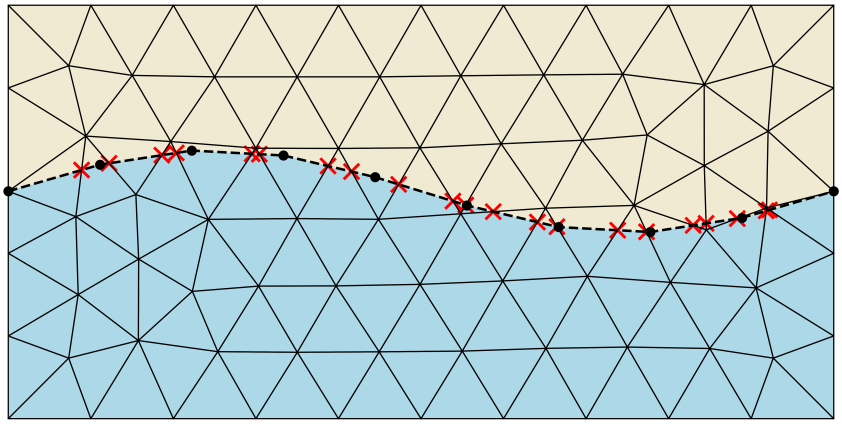


Figure 3.12: Intersection computation between the mesh and the markers.

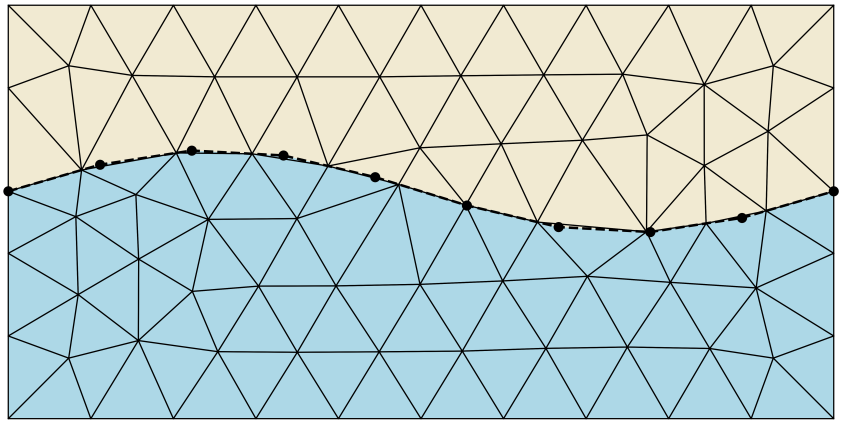


Figure 3.13: Final mesh after deformation to match the markers interface.

3.2.5 Defining phase of the computational mesh

As we have seen for the level set approach, a last ingredient of the mesh deformation algorithm is to define the phase of each element of the computational mesh. In the case of the level set approach, this was done by looking at the sign of the level set at the vertices of each element (and with a special treatment for triple-zero elements). In the case of the front-tracking approach, we can use

the markers to define the phase of each element. We developped two different approaches to do so, each one having its own advantages and drawbacks.

Winding number approach.
The first approach and the most natural one is to use the winding number of each element's centroid with respect to the markers. This approach is very robust and can handle complex scenarios with multiple interfaces and multiple topological changes. However, this approach can be computationally expensive since we need to compute the winding number for each element and for each interface. This could be mitigated by using a space partitioning structure to only compute the winding number for elements that are close to an interface and by using parallel computing.

The Figure 3.14 shows an example of the winding number approach. We can see that the phase of each element is correctly defined according to the markers.

A drawback is that, for high interface curvature, the mesh deformation can *cut* the curvature and a triangle may be considered inside the interface (by the winding number) even though it clearly lies outside. This is represented in Figure 3.15, where we can see that the triangle should be considered outside the interface but the winding number approach considers it inside. This can also appear easily when we have very elongated elements parallel to the interface. This problem can have a non-negligible impact on the simulation when having surface tension effects since the curvature of the interface in the computational mesh will not be correctly represented.

Breadth First Search approach
The second approach is to use a Breadth First Search (BFS) algorithm to propagate the phase from a triangle that is known to be in a certain phase to its neighbours. This allows to avoid the problem of the winding number approach when the mesh deformation algorithm *cuts* the interface presented in Figure 3.15. This approach is thus more adapted when surface tension effects are important. However, in this approach, the mesh deformation algorithm must ensure that the interface is represented in the mesh as a close loop of edges. This means that the mesh deformation algorithm must be robust enough to ensure that no gaps are created in the interface representation. Otherwise, the BFS algorithm will propagate the phase through these gaps and color the whole domain in the same phase. Our mesh deformation algorithm is heuristic and cannot guarantee that no gaps will be created. In practice, this can happen for complex scenarios where the features of the interface are smaller than the mesh size. To mitigate this problem, we can use the winding number approach as a fallback when the BFS algorithm fails. We can also periodically reset the markers by using the nodes of the mesh that are on the interface as new mark-

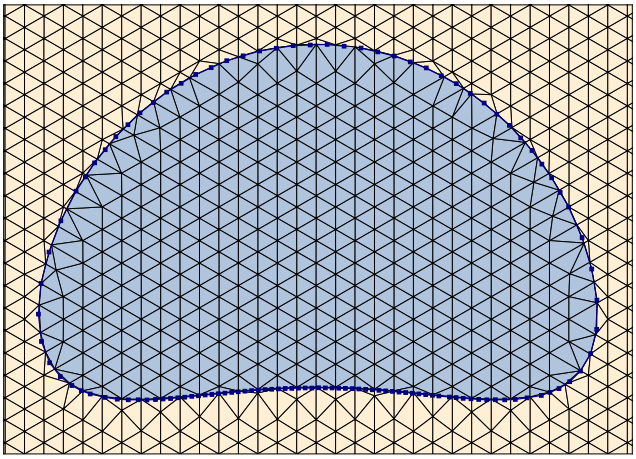


Figure 3.14: Example of the winding number approach to define the phase of each element.

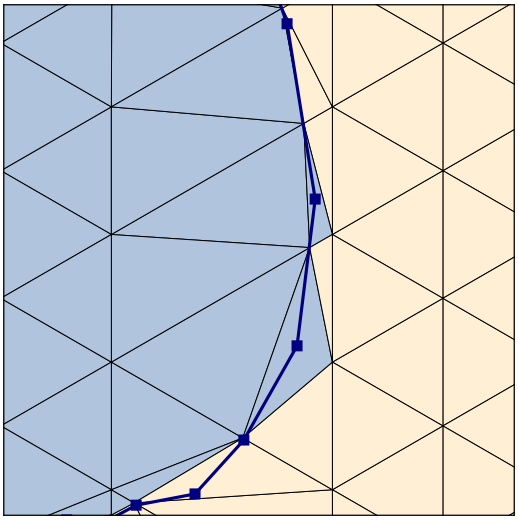


Figure 3.15: Example of the winding number approach failing to define the phase of a triangle.

ers. This prevents the markers to develop too complex features that the mesh cannot represent. An example of this problem occurring is represented in Figure 3.16 where two nodes, represented by red dots, have already been moved on

the interface (blue line with square markers) and thus cannot be moved again to capture the interface. This creates an edge of the mesh that cut the interface.

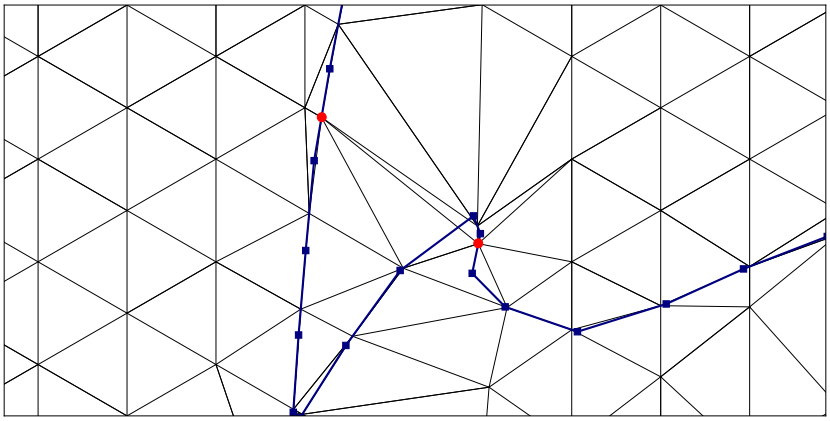


Figure 3.16: Example of the BFS approach failing to define the phase of the triangles due to a gap in the interface representation.

3.2.6 Boundaries management

In our marker framework, the boundaries of the domain are managed as a regular interface, described by a set of markers that are fixed on the boundary. When considering two fluids in a closed domain, the interface will often meet the boundary of the domain. This creates triple-points where three phases meet: the two fluids and the solid boundary. This situation appears in many practical applications, such as the one represented in Figure 3.17, where a dam break occurs in a closed channel. In this case, as the water flows, a triple-point is advancing along the bottom wall and an other triple-point is receding along the top wall. These triple-points are the only markers that can move along the boundary of the domain, while the other markers on the boundary remain fixed.

The dynamics of these triple-points are complex and involve phenomena such as wetting, contact angle hysteresis, and dynamic contact lines. In this work, we do not aim to model these complex phenomena but rather to ensure that our numerical method can handle the presence of triple-points in a robust manner.

A close-up view of the computational mesh and the markers representing the

interface near a triple-point is presented in Figure 3.18. The markers are represented by the blue line with squares and the triple point is represented by the red dot. As we can see, the mesh algorithm ensures that the triple-point is represented by a node of the mesh as explained in the previous section. Despite not being used in this work, having the triple-point represented by a node of the mesh could be useful to apply specific boundary conditions at this point in future work, such as a dynamic contact angle condition.

Advection of the triple point is performed by projecting the local fluid velocity onto the boundary, computing the distance over the time step, and moving the triple point along the boundary. This means that when arriving at a corner of the domain, the triple-point will not stop but will turn around the corner without losing speed. Since the other markers on the boundary are fixed, the triple-points will cross them as they are moving. We simply reorder the markers set to ensure that the markers are always ordered along the interface.

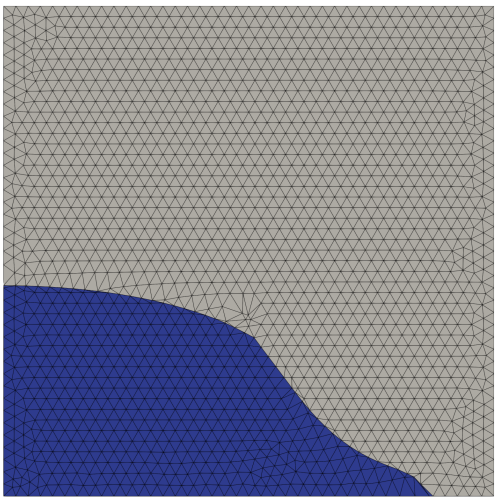


Figure 3.17: Example of a dam break in a closed channel, where two triple-points are present.

3.2.7 A Winslow energy for mesh displacement in the bulk

In addition to moving nodes close to the interface to match it, we also move nodes in the bulk. This allows to improve the quality of the mesh but also to bring more nodes close to the interface to be able to capture small features of

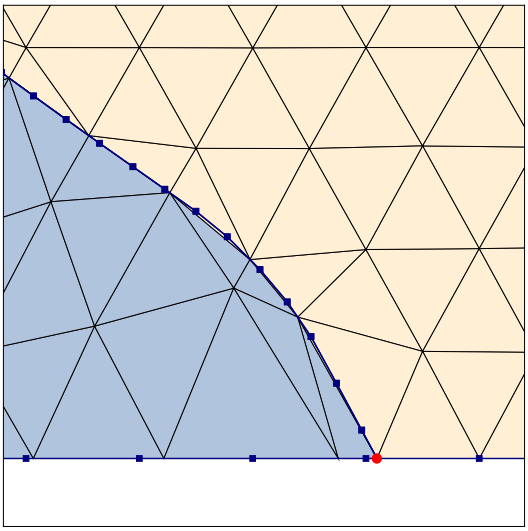


Figure 3.18: Close-up view of the computational mesh and the markers representing the interface near a triple-point.

the flow. To do so, we used the approach developed in [Garanzha et al. (2021)] which consists in minimizing a regularized version of the Winslow energy of the mesh. We used the implementation developed by Maxence Reberol in Gmsh [Geuzaine and Remacle (2009)].
This energy comprises two competing terms, balanced by a parameter λ. The first term corresponds to the area of the elements and the second term corresponds to the angles of the elements. In the approach developed in [Garanzha et al. (2021)], the authors consider ideal equilateral elements of a constant size. They then compute a regularized version of the Winslow energy and minimize it in order to obtain a mesh with elements as close as possible to the ideal ones. In order to control the size of the elements in the domain, we adapted this approach to consider ideal equilateral elements of a size that can vary in space. We give a field corresponding to the desired size of the elements in the domain and we also lock the position of the boundary nodes and the nodes that have been moved to match the interface. To create the size field we assign a small target element size close to the interface and a larger size far from it, with a specified transition distance between the two regions. This is represented in Figure 3.19, where we can see the impact on the mesh of this operation. By using the Winslow energy minimization, we are able to bring more nodes close to the interface and thus to capture smaller features of the flow that occurs at this place such as boundary layers or thin films.

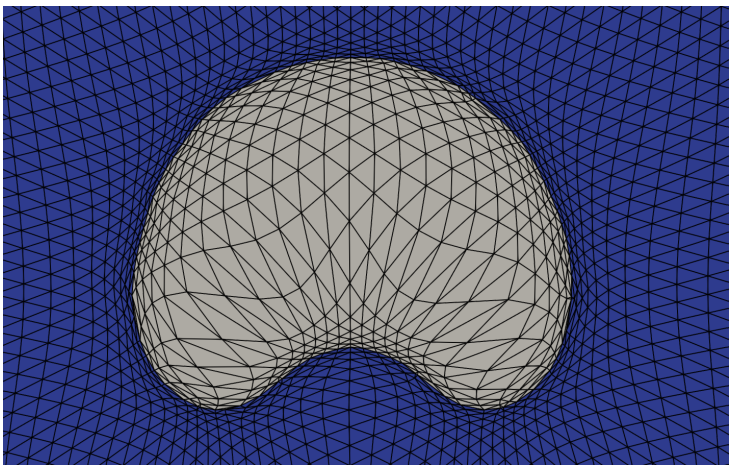


Figure 3.19: Example of mesh obtained after minimizing the Winslow energy for a bubble simulation.

CHAPTER 4

Application to two-phase flows

Contents

4.1 Solver and coupling algorithm 104
4.2 Surface tension and jump conditions at the interface 105
4.3 Numerical examples - levelset approach 107
4.3.1 Verification of the solver – sloshing 108
4.3.2 Dambreak 109
4.3.3 Rayleigh-Taylor instability 110
4.3.4 Single Bubble rising 112
4.3.5 Bubble merging 117
4.4 Numerical examples - front-tracking approach . . 119
4.4.1 Vortex in a box 119
4.4.2 Rayleigh-Taylor instability 119
4.4.3 Ellipsoidal bubble relaxation 120
4.4.4 Single bubble rising - Case 1 122
4.4.5 Single bubble rising - Case 2 127
4.5 Two-phase jet 129

This chapter applies the X-Mesh and deformation algorithms from Chapter 3 to simulate two-phase flows using the TFEM from Chapter 2. Before validating the method on complex two-phase flow problems, we first describe the solver and coupling algorithm used to solve the Navier-Stokes equations. Then we present how surface tension effects are taken into account and how we take advantage of the sharp interface representation to impose exactly the jump conditions at the interface. Finally, we present several numerical examples to demonstrate the capabilities of the X-Mesh method for simulating two-phase flows with sharp interfaces both with the levelset-based and the front-tracking-based mesh deformation algorithms.

4.1 Solver and coupling algorithm

In the introduction of this thesis, we progressively built a numerical solver for two-phase flows using tools already available in the literature. We started from the Navier-Stokes equations, developed the weak formulation and discretized it in space using the finite element method. We pointed out the need for stabilization techniques to handle high Reynolds number flows and the inf-sup condition for the pressure-velocity coupling. We chose to circumvent those two problems using the SUPG and PSPG stabilization techniques respectively. We then presented the time discretization using a simple implicit Euler scheme. Finally, since the mesh is moving, we need to take that into account in the equations and we use the Arbitrary Lagrangian-Eulerian (ALE) framework to do so. The obtained weak formulation is the one we will use in this chapter to solve the two-phase flow problem without surface tension. We recall it here for convenience:

Find $(\mathbf{u}_h, p_h) \in \mathbf{S}_{\mathbf{u}}^h \times \mathbf{S}_p^h$ such that for all $(\mathbf{v}_h, q_h) \in \mathbf{V}_{\mathbf{u}}^h \times \mathbf{V}_p^h$:

$$\int_{\Omega^{n+1}} \rho \left(\frac{\mathbf{u}_h^{n+1} - \mathbf{u}_h^n}{\Delta t} + (\mathbf{u}_h^{n+1} - \mathbf{u}_{\text{mesh}}) \cdot \nabla \mathbf{u}_h^{n+1} \right) \cdot \mathbf{v}_h \, d\Omega + \int_{\Omega^{n+1}} \mu \nabla \mathbf{u}_h^{n+1} : \nabla \mathbf{v}_h \, d\Omega - \int_{\Omega^{n+1}} p_h \nabla \cdot \mathbf{v}_h \, d\Omega + \text{SUPG} = \int_{\Omega^{n+1}} \mathbf{f}^{n+1} \cdot \mathbf{v}_h \, d\Omega + \int_{\partial\Omega_N} \mathbf{t}^{n+1} \cdot \mathbf{v}_h d\Gamma$$

$$\int_{\Omega^{n+1}} q_h \nabla \cdot \mathbf{u}_h^{n+1} \, d\Omega + \text{PSPG} = 0$$

with $\mathbf{S}$ and $\mathbf{V}$ the appropriate functional spaces for the velocity and pressure, $\mathbf{u}_{\text{mesh}}$ the mesh velocity, $\mathbf{f}$ the body forces (gravity) and $\mathbf{t}$ the surface traction on the Neumann boundary $\partial\Omega_N$.

We choose a sequential coupling of the different parts presented previously: the resolution of the Navier-Stokes equations, the displacement of the interface (level set or markers) and the deformation of the mesh with the algorithm. All the results have been obtained with a sequential algorithm. The simulation of a time step consists in the successive resolution of these different sub-steps. We first do an explicit prediction of the interface position by solving the advection equation for the levelset approach or by moving the markers and resolving the potential topology changes. We then deform the mesh using the algorithms presented in sections 3.1 and 3.2. We compute the mesh velocity $\mathbf{u}_{\text{mesh}}$ as the difference between the current and previous position of the mesh divided by the time step Δt. Then we have an intermediate step to ensure the quality of interface representation. For the levelset-based algorithm, the level set values at the nodes are no longer correct after the mesh deformation. We thus reinitialize the level set using the fast-marching method [Sethian (1996)] with the nodes of the front as seeds. For the front-tracking-based algorithm, the evolution of the interface can lead to some markers being too close or too far from each other. We thus add or remove markers to ensure a good quality of the interface representation. We then assign a phase to each triangle of the mesh based on the interface position. Finally, we solve the Navier-Stokes equations on the new mesh using the ALE formulation presented above. Let's note that there is no stability constraint on the time step due to the small elements since we are using an implicit scheme. The algorithm is summarized in Algorithm 2 for the levelset-based approach and in Algorithm 3 for the front-tracking-based approach. Both algorithms are very similar, the only difference being the way the interface is represented and advected and the associated mesh deformation algorithm.

Algorithm 2: Sequential coupling with the levelset approach

1 ϕ_{adv} = level set $(\mathbf{u}_n, \phi_n)$
2 $\mathbf{x}_{n+1}$, front_{n+1} = X-Mesh $(\mathbf{x}_n, \phi_{\text{adv}}, \phi_n, \text{front}_n)$
3 ϕ_{n+1} = Fastmarching $(\mathbf{x}_{n+1}, \text{front}_{n+1})$ $\mathbf{u}_{\text{mesh}} = (\mathbf{x}_{n+1} - \mathbf{x}_n)\frac{1}{\Delta t}$
4 $\mathbf{u}_{n+1}$ = Navier-Stokes $(\mathbf{u}_n, \phi_{n+1}, \mathbf{u}_{\text{mesh}})$
5 $t = t + \Delta t$

4.2 Surface tension and jump conditions at the interface

When developing the Navier-Stokes equations in Chapter 1, we have seen that the presence of surface tension at the interface between two fluids leads to jump conditions in the normal stress across the interface. Those jump conditions need to be taken into account in the weak formulation of the Navier-Stokes

Algorithm 3: Sequential coupling with the front-tracking approach

1 markers_{n+1} = AdvectMarkers ($\mathbf{u}_n, \text{markers}_n$)
2 $\mathbf{x}_{n+1}$ = X-Mesh ($\mathbf{x}_n, \text{markers}_{n+1}$)
3 markers_{n+1} = AdaptMarkers (markers_{n+1})
4 $\mathbf{u}_{\text{mesh}} = (\mathbf{x}_{n+1} - \mathbf{x}_n)\,\frac{1}{\Delta t}$
5 $\mathbf{u}_{n+1}$ = Navier-Stokes ($\mathbf{u}_n, \phi_{n+1}, \mathbf{u}_{\text{mesh}}$)
6 $t = t + \Delta t$

equations. Multiple approaches exist and we reviewed them shortly in the introduction of this thesis. We employ the Ghost Fluid Method (GFM) [Fedkiw et al. (1999); Kang et al. (2000)] to impose the jump conditions exactly. The GFM consists in modifying the gradient of the pressure operator in the elements cut by the interface to take into account the jump conditions. In our approach, the interface is exactly represented by the mesh so we can add a pressure field that represent the jump conditions and that is discontinuous at the mesh edges of the interface. This is represented in 1D in Figure 4.1 where we can see the continuous pressure p and the discontinuous pressure field p_{jump} that is added to take into account the jump conditions. The sum of those two fields is the total pressure field p_{total} that corresponds to the physical pressure. The discontinuous field p_{jump} is computed using the Young-Laplace equation that relates the jump in pressure to the surface tension coefficient σ and the curvature κ of the interface:

$$[p] = p_{\text{jump}} = \sigma\kappa$$

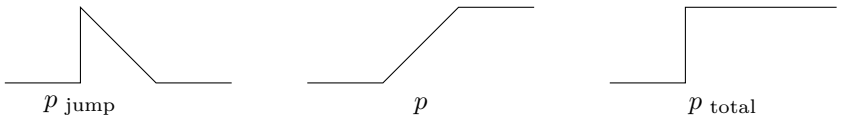


Figure 4.1: 1D representation of the pressure field in the presence of surface tension using the Ghost Fluid Method.

The advantage of using X-Mesh with the GFM is that the interface is exactly represented by the mesh so we can impose exactly the jump conditions. This is not possible with a fixed mesh where the interface is represented inside the elements. The Figure 4.2 shows in 1D what happens for the GFM on a fixed mesh when the interface is slightly moved. The pressure field is not modified at all since the interface is still in the same element. So even if the interface is moved, the surface tension effects are not modified at all. The approach has

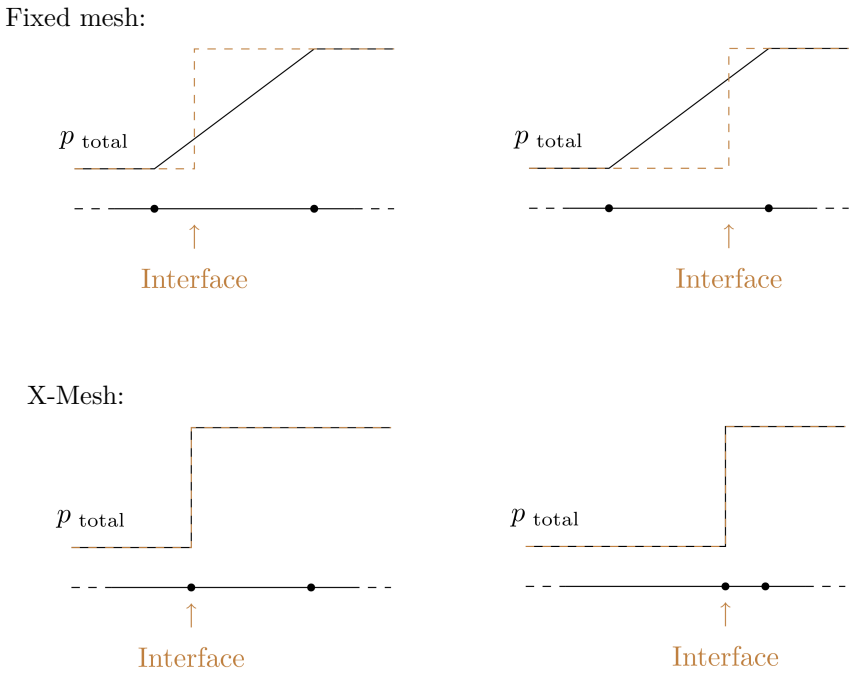


Figure 4.2: 1D representation of the pressure field in the presence of surface tension using the Ghost Fluid Method on a fixed mesh versus with X-Mesh.

thus a certain *thickness* over which the interface can be moved without modifying the pressure [Popinet (2018)]. This is not the case with X-Mesh where the interface is always represented by the mesh and thus any modification of its position will lead to a modification of the pressure field as we can see in Figure 4.2.

The curvature κ is computed using the mesh nodes that represent the interface. In 2D, the curvature is simply the inverse of the radius of the osculating circle at each node of the interface. As we did in 1D, we show in Figure 4.3 the pressure jump at the interface, the continuous pressure field and the total pressure field for the simulation of a Laplace static bubble.

4.3 Numerical examples - levelset approach

In this section, we present several numerical examples to demonstrate the capabilities of the X-Mesh method for simulating two-phase flows with sharp interfaces using the levelset-based mesh deformation algorithm. We first present a sloshing problem with an analytical solution (in small perturbation regime)

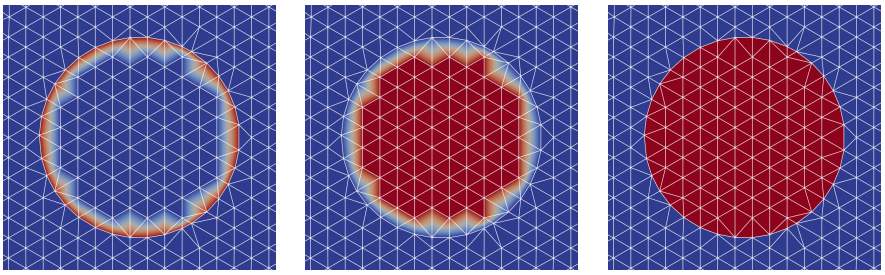


Figure 4.3: Visualization of the 2D pressure field in the presence of surface tension using the Ghost Fluid Method.

to verify that our solver solves the right equations. We then present three classical benchmarks for two-phase flow solvers: the dambreak, the Rayleigh-Taylor instability and bubbles simulations. These results have been presented in [Quiriny et al. (2024)].

4.3.1 Verification of the solver – sloshing

The *sloshing problem* consists in computing the free oscillations of a liquid in a tank. This problem essentially allows to do a verification of the solver because it is one of the few cases where an analytical solution exists in the case of small perturbations of a planar interface [Wu et al. (2001)]. In the analytical setting, the interface is sufficiently simple so that it can be represented by a height function $\eta(x,t)$ that is initialized as

$$\eta(x,0) = d + \eta_0 \sin k\,(0.5 - x)$$

where $d = 1.0$ is the mean elevation, $\eta_0 = 0.01$ is the initial oscillation amplitude and k is the wave number.

The analytical evolution of the maximum height of the free surface η depends on $\nu = \frac{\mu}{\rho}$ the kinematic viscosity and the dimensionless parameter $\lambda = \frac{g}{\nu^2 k^3}$:

$$\frac{\eta(t)}{\eta_0} = 1 - \lambda e^{-\nu k^2 t} f(\nu k^2 t)$$

where $f(t)$ is the inverse Laplace transform of the function

$$F(s) = \frac{1}{(s-1)\left((s+1)^2 - 4s^{\frac{1}{2}} + \lambda\right)}.$$

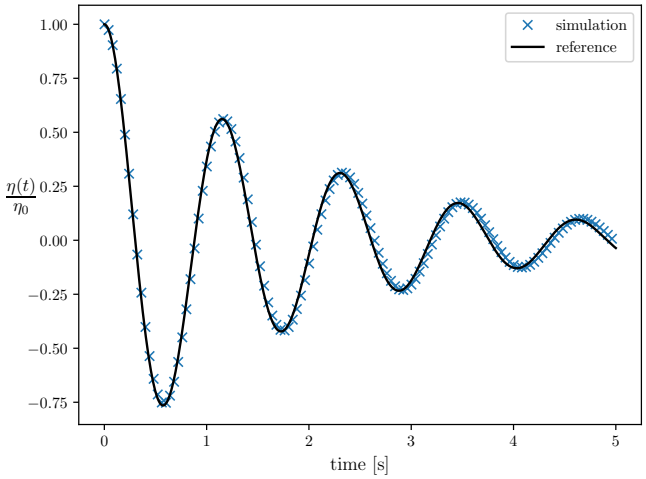


Figure 4.4: Wave elevation evolution in a sloshing problem.

The computational domain considered is a rectangle of dimensions $[0, d] \times [0, 1.3d]$ and the four wall have free-slip condition.

Figure 4.4 shows the evolution of the perturbation η over time. We can observe that the simulation is in good agreement with the analytical solution. Since this analytical solution is obtained from the linearized Navier-Stokes equations, the initial perturbation $\eta_0 = 10^{-2}$ cannot be too large with respect to $d = 0.5$.

For the reference solution of the linearized Navier-Stokes equations to be valid, the amplitude of the sloshing must be small enough. However, the mesh used for this simulation is quite coarse with 3900 vertices. At the beginning of the simulation the characteristic size of the elements is smaller than the amplitude of the sloshing and thus the front is relayed between vertices. However, quite quickly the amplitude of the sloshing decreases and the simulation is close to a classical ALE simulation without relaying of the interface

4.3.2 Dambreak

The dambreak test case consists of the collapse of a fluid column in a container. It corresponds to the sudden rupture of a dam and the flow of the impounded water. The evolution of the interface in this problem is complex and involves multiple topology changes. Its complexity has made it a reference test case to validate two-phase flow simulations. The considered column has height $H = 0.4\,[m]$ and width $L = 0.4\,[m]$. It collapses in a calculation domain of

size $1.4 \times 1.1\,[m^2]$. Boundary conditions of free-slip are applied to the 4 walls. The two fluids used in this test case can be described by their density ρ and kinematic viscosity ν:

$$\rho_0 = 10 \qquad \rho_1 = 1000 \qquad \nu_0 = \nu_1 = 10^{-3}$$

The mesh used has $75,950$ vertices. Figure 4.5 represents the evolution of the free surface at the different adimensional times $t = 0.5, 1.75, 3, 5.5, 6.5$ and 22.5, where the reference time considered is $t_{ref} = \sqrt{h/g}$. Despite the numerous topological changes present during a dambreak, the method is robust enough to simulate such complex flows.

4.3.3 Rayleigh-Taylor instability

The Rayleigh-Taylor instability consists in placing a heavier fluid above a lighter one. This position is an unstable equilibrium and is sensitive to any perturbation. A classical benchmark for the validation of two-phase-flow solver is to trigger a single-mode perturbation in the initialization. The instability develops under the influence of the gravity field and is commonly occurring in a wide range of physical phenomena. A disturbance is initialized in the interface position at $t = 0$ and it's initial position is given by:

$$y = 2.0 + 0.05\cos(2\pi x)$$

Initially, the velocity field is set to 0 and the pressure is hydrostatic. The evolution in time of the 2 fluids is determined by 2 adimensional numbers, the Atwood (At) and Reynolds (Re) numbers defined in our case by:

$$At = \frac{\rho_1 - \rho_2}{\rho_1 + \rho_2}$$
$$Re = \frac{\sqrt{W g} W}{\nu}$$

with W the width of the canal.

Figure 4.6 shows the evolution of the two phases for the case At = 0.5 and Re = 256. We used three meshes, (a) 123 k, (b) 31 k, and (c) 8 k nodes, to study mesh sensitivity. We compared our results with those of He et al [He et al. (1999)] for a mesh of approximately 262,000 elements using the lattice Boltzmann method. The dashed white line in Figure 1 correspond to the interface they obtained. The sharp interface of the method presented, as well as the fine mesh (a) used, allows the features of the flow to be preserved for a long time. As explained in section 3.1, if the 3 nodes of an element are positioned on the interface, the phase of this element is ambiguous. This happens every time a phase merges or splits because the level set naturally handles topological changes at the scale of an element. The process of merging is thus mesh

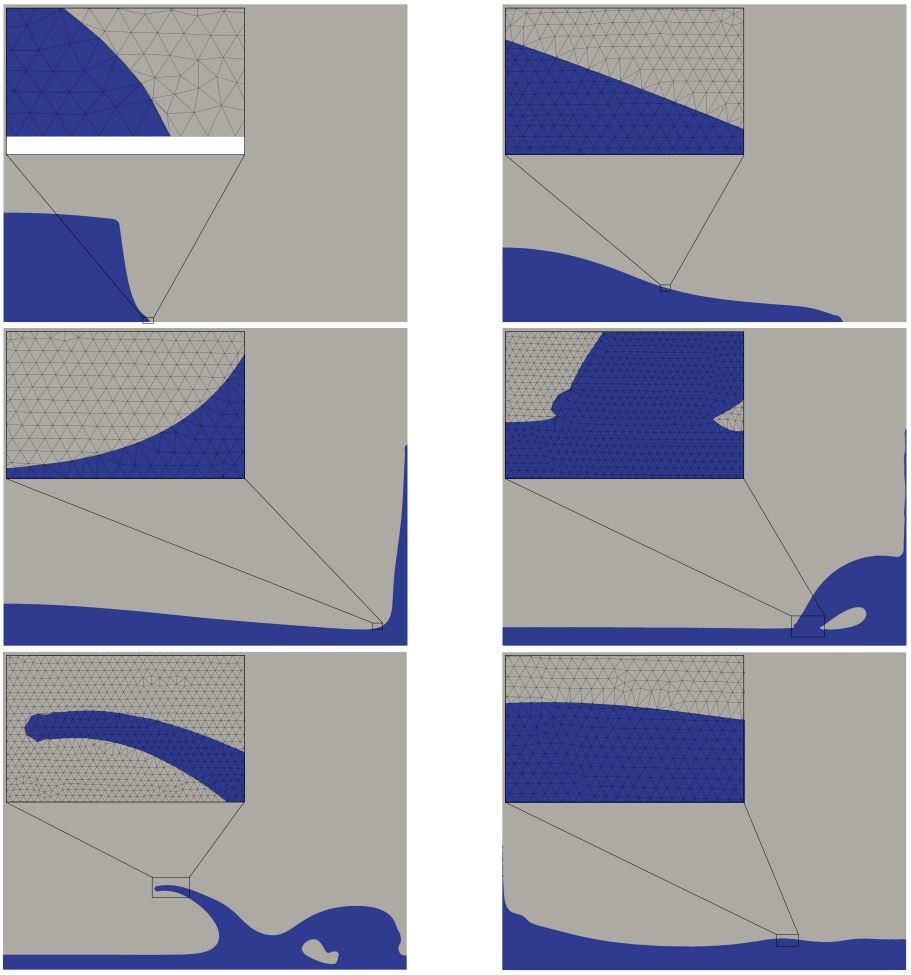


Figure 4.5: Dambreak simulation: Free surface position for the adimensional times $t = 0.5$, 1.75, 3, 5.5, 6.5 and 22.5.

dependent and the asymmetry of the mesh generates a small asymmetry in the flow which propagates until it generates a completely asymmetric flow, as we can see in Figure 4.7 (left). The effect of the characteristic length of the mesh is shown in Figure 4.6 (bottom), where the same Rayleigh-Taylor instability is simulated but with the 4 times coarser mesh b. An analytical solution for the linearized equations exist and is valid for the linear phase of the instability

development [Drazin and Reid (2004)]:

$$h = h_0 e^{\hat{\alpha} t}$$

with h the perturbation size, h_0 the perturbation at $t = 0$ and $\hat{\alpha}$ the growth rate. For the parameter of the simulation the analytical grow rate is $\hat{\alpha}_a = 9.3$ and the observed one is $\hat{\alpha}_o = 9.2$.

It is also interesting to compare the mass conservation for different mesh refinements. The evolution of the relative mass of the grey fluid during the Rayleigh-Taylor simulation is shown in Figure 4.7. We can see that the mass variation is limited to 1% and is more chaotic for coarser meshes. The dots on the graph represent the moment when the flow is no longer completely captured by the mesh. As we can see the mass start to vary significantly once that point is reached. By placing the nodes of the mesh on the boundary, we naturally *cut off the corners* of the boundary, resulting in a loss of mass for fluid 1 or fluid 2, depending on whether the boundary is convex or concave. The Rayleigh-Taylor instability has an interface that is convex in some places and concave in others. This leads to opposite effects of mass loss, which explain the ascending and descending aspects of the curves in Figure 4.7.

4.3.4 Single Bubble rising

Bubble dynamics are frequently encountered in a variety of industrial processes or natural flows. The correct representation of the interface is essential for this type of problem because the surface tension force is proportional to the curvature of the interface. Grace et al [Grace et al. (1976)] have characterized in 1976 the final shape obtained during single bubble rise experiment for different fluids. Figure 4.9 summarizes their observations. To validate our method for bubble dynamics we try to reproduce these results. The y axis of Figure 4.9 corresponds to the Reynolds number and its x axis to the Bond (or Eötvos) number. These numbers can be computed for the case of bubbles by

$$Re = \frac{\rho_1 \sqrt{g} D^{3/2}}{\mu_1}$$
$$Bo = \frac{\rho_1 g D^2}{\sigma}$$

where D is the diameter of the bubble and the subscripts 1 corresponds to the heavier fluid.

We consider three cases with different pairs of Reynolds and Bond numbers represented in Figure 4.8 and on Grace's diagram at Figure 4.9. The density and viscosity ratios considered are $\frac{\rho_1}{\rho 2} = 1000$ and $\frac{\mu_1}{\mu_2} = 100$. The diameter of the bubble is fixed to $D = 1\,[m]$ and to avoid any impact of the lateral wall on the bubble dynamic the computational domain considered is $W = 6\,[m]$ width

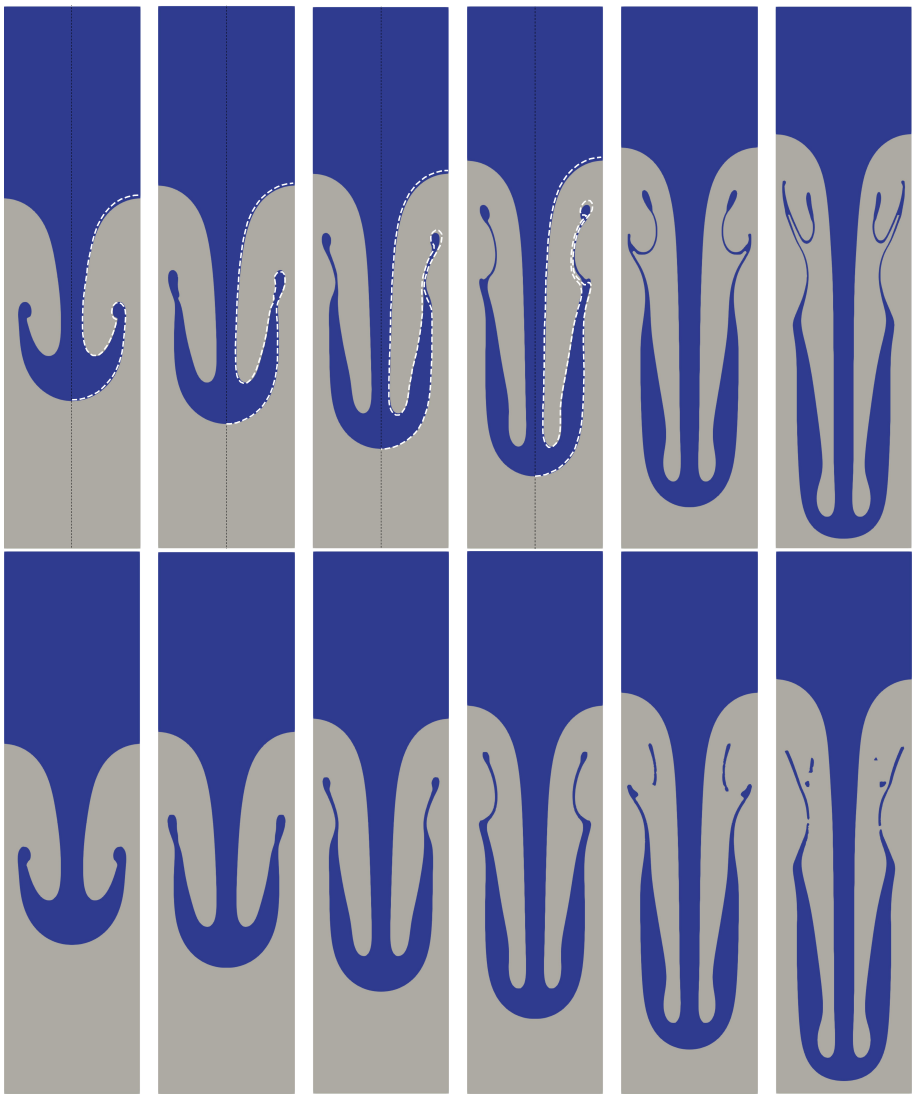


Figure 4.6: Rayleigh-Taylor instability for $Re = 256$, $At = 0.5$ at times $t = 3, 3.5, 4, 4.5, 5$ and 5.5 with a fine mesh (top) and a coarser mesh (bottom).

and $H = 12\,[m]$ high. Surface tension is applied explicitly, imposing a time-step constraint:

$$\Delta t < \sqrt{\frac{(\rho_1 + \rho_2)\, h^3}{4\pi\sigma}}$$

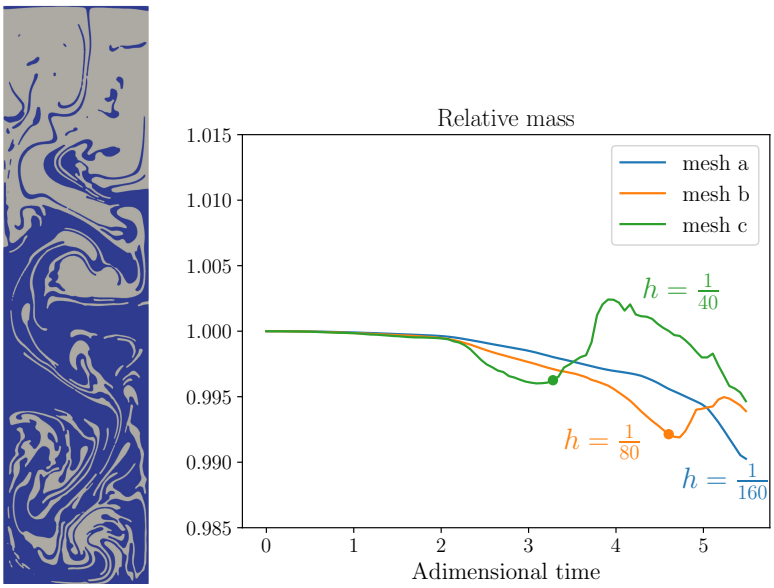


Figure 4.7: (left) Rayleigh-Taylor instability at time $t = 15$ with the mesh (a). (right) Relative mass of the grey fluid during the Rayleigh-Taylor simulation for the 3 different meshes

where h is the characteristic size of the mesh.

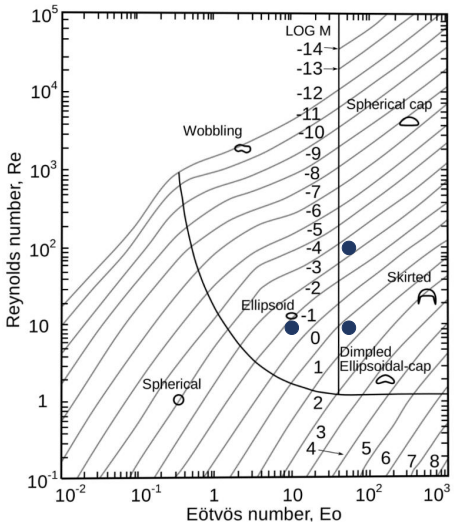
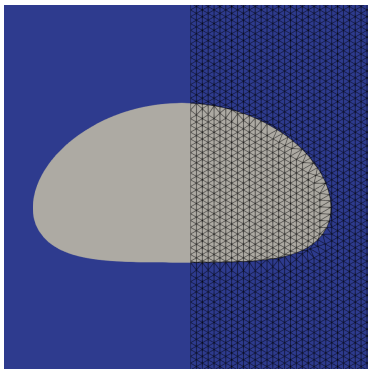
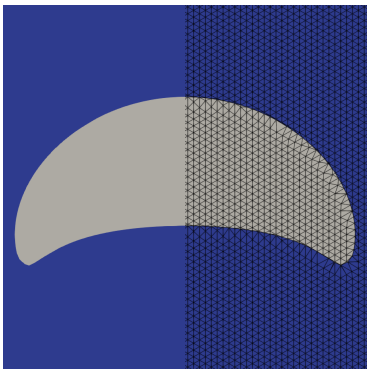


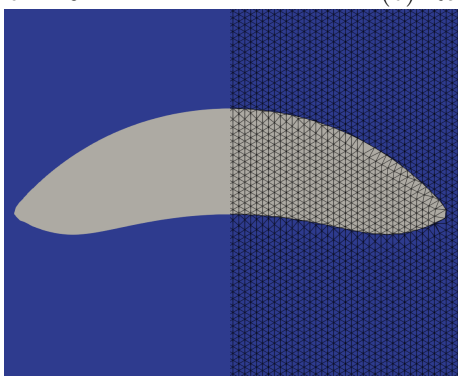
Figure 4.9: Grace's diagram [Grace et al. (1976)].



(a) $Re = 10,\ Bo = 10$



(b) $Re = 10,\ Bo = 50$



(c) $Re = 100,\ Bo = 50$

Figure 4.8: Final form of free rise bubbles

When we look at the final shape of the bubbles and compare it with Grace's experimental diagram or with Hua and Lou [Hua and Lou (2007)], we see that the simulations follow the same trends. A good way of validating our results is to compare them using the first case of the 2D benchmark proposed by Hysing et al. [Hysing et al. (2009)]. This is a free rising bubble with the following properties for the 2 fluids:

$$\rho_1 = 1000 \qquad \rho_2 = 100 \qquad \mu_1 = 10 \qquad \mu_2 = 1 \qquad g = 0.98 \qquad \sigma = 24.5$$

with units in $[kg/m^3]$, $[kg/(m.s)]$, $[m/s^2]$ and $[N/m]$ for the densities, the dynamic viscosities, the gravity coefficient and the surface tension respectively. In addition to comparing the final shape of the bubbles, the paper proposes to compare several benchmark quantities: the centre of mass of the bubble, the mean velocity of the bubble and the circularity. The latter is defined as the ratio between the perimeter of a circle of the same area as the bubble

and the perimeter of the bubble. We will also add the mass (area) of the bubble. To observe the convergence of the method, this simulation is carried out with different levels of mesh refinement: $h = \frac{1}{40}, \frac{1}{80}$ and $\frac{1}{160}$ where h is the characteristic size of an element.

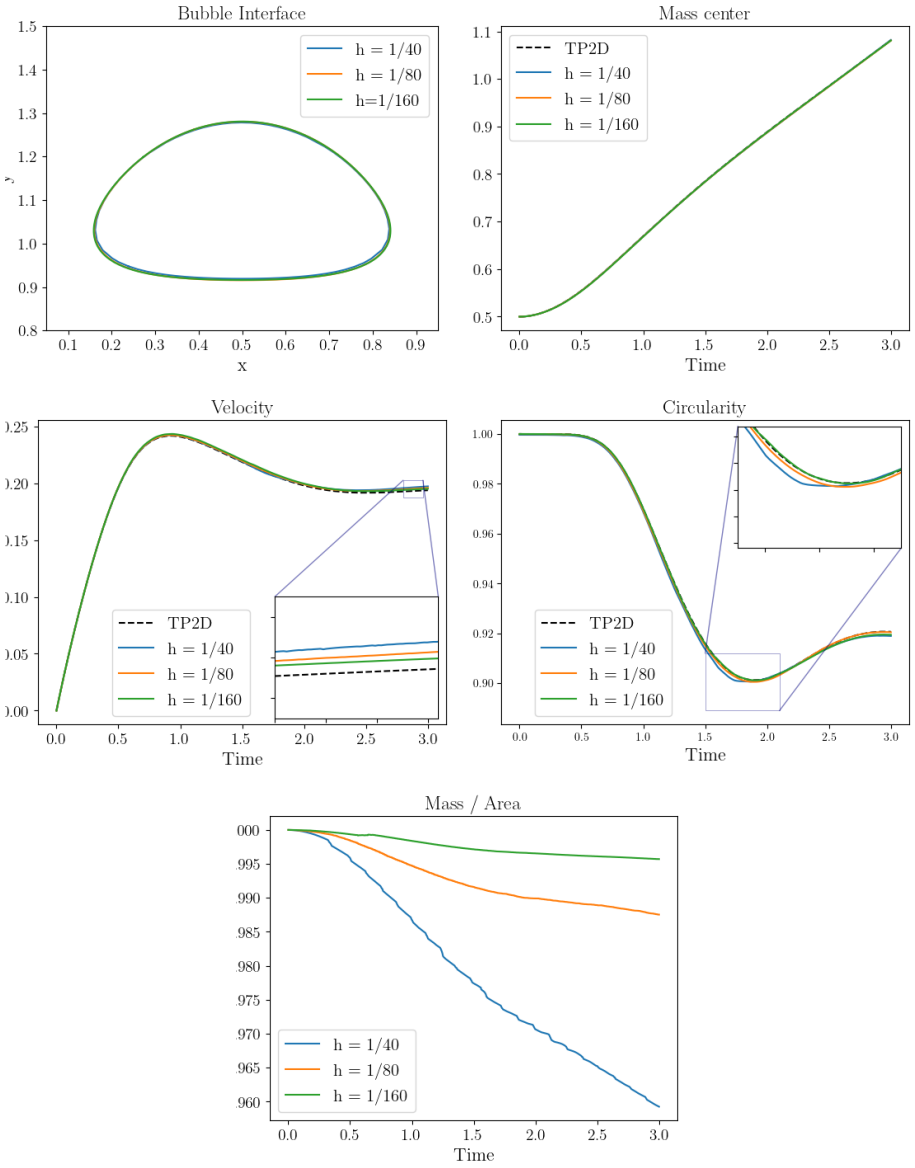


Figure 4.10: Results of the benchmark of Hysing et al. for a free rising bubble

A reference solution is also displayed, the solution obtained by the TP2D solver [Osher and Sethian (1988); Turek (1997)] for a mesh size $h = \frac{1}{320}$. TP2D, which is short for Transport Phenomena in 2D, is a code developed by TU Dortmund and is an extension of the Featflow incompressible flow solver to treat immiscible fluids with the level set method. As we can see on Figure 4.10, the bubble interface converges rapidly as the interface position for $h = \frac{1}{80}$ and $h = \frac{1}{160}$ are difficult to distinguish. The mass center vertical position of the bubble as well as the velocity and the circularity of the bubble can be compared with the reference solution. The position of the mass center is in good agreement with the reference and the solution obtained with our method is close to the reference solution even for a coarse mesh. The velocity of the bubble, measured as the integral of the velocity inside the bubble, does not seem to converge but the solution is already almost on the reference solution. The circularity is a more complex benchmark quantity to obtain in the simulations. We observe in Figure 4.10 that our method converges to the reference solution though a noticeable difference remains. Finally we can compute the relative mass of the bubble evolving during the simulation. We can clearly see the impact of the mesh on the mass conservation with a variation going from approximately 4 % for a characteristic mesh size $h = \frac{1}{40}$ to 0.25 % for $h = \frac{1}{160}$.

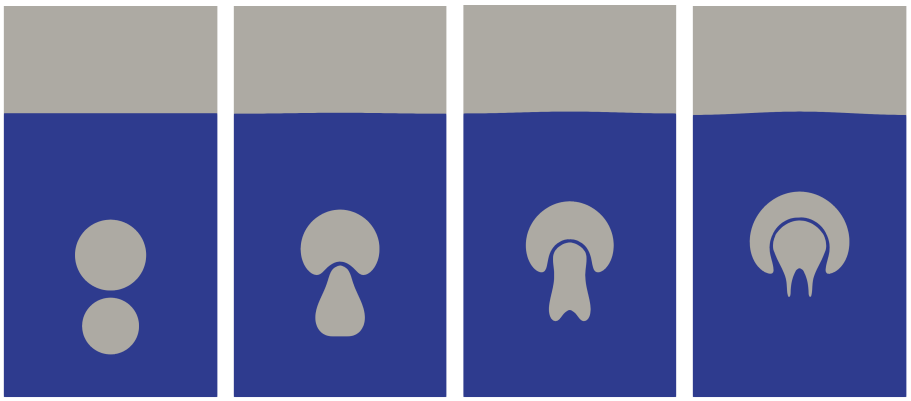
4.3.5 Bubble merging

To verify that the approach used for the surface tension works during topological changes of the fluid phases, we are interested in a test case of two bubbles placed one above the other in a column of heavy fluid with a free surface. The computational domain is a 6 by 3 $[m]$ square, the two bubbles are centered horizontally, their centers are positioned at height $y = 1\,[m]$ and $y = 2\,[m]$ with radius of 0.4 and 0.5 meters, respectively. The parameters of the two fluids are chosen such that:

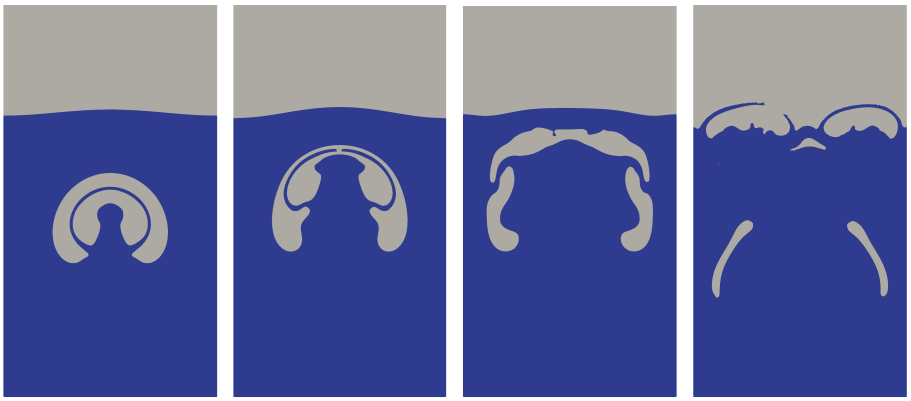
$$\frac{\rho_1}{\rho_0} = 1000 \qquad \frac{\mu_1}{\mu_0} = 2 \qquad Re = 104 \qquad Bo = 313$$

with the Re and Bo numbers defined with the diameter of the greater bubble.

The lower bubble is sucked by the drag of the upper bubble such that at $t = 0.4\,[s]$ it is engulfed by the latter. The two bubbles rise together until at $t = 0.95\,[s]$, the stronger currents in the center of the column push them to collapse. This leads to multiple changes in topology, resulting in two bubbles that rise to the free surface and two elongated bubbles that are carried to the bottom by downward currents on the sides.



(a) $t = 0$ (b) $t = 0.2$ (c) $t = 0.3$ (d) $t = 0.4$



(e) $t = 0.6$ (f) $t = 0.95$ (g) $t = 2.2$ (h) $t = 3.85$

4.4 Numerical examples - front-tracking approach

In this section, we present several numerical examples to demonstrate the capabilities of the X-Mesh method for simulating two-phase flows with sharp interfaces using the front-tracking-based mesh deformation algorithm. We consider the same Rayleigh-Taylor instability and bubble rising benchmarks as in the previous section to compare the two approaches. As we will see, the front-tracking approach allows a representation of the interface decoupled from the mesh size, allowing us to use coarser meshes while maintaining a good representation of the interface.

4.4.1 Vortex in a box

In order to demonstrate the ability of the front-tracking approach to represent a complex interface, we consider the classical test case of a vortex in a box. In this testcase, a circular interface is deformed by an analytical velocity field given by:

$$u(x,y) = \sin(2\pi y)\sin^2(\pi x)$$
$$v(x,y) = -\sin(2\pi x)\sin^2(\pi y)$$

The initial interface is a circle of radius 0.15 centered at (0.5, 0.75) inside a domain $\Omega = [0,1] \times [0,1]$ and we consider a mesh of characteristic length $\frac{1}{40}$. Front-tracking approaches are particularly well suited for this type of problem but here we want to verify that the mesh deformation algorithm is able to follow such a complex interface. The solution at time $t = 5$ is shown in Figure 4.12 and we can see that the interface is well represented despite its complexity. The same figure but with the mesh is shown in Figure 4.13 and we can see that the mesh is able to deform and become almost degenerate inside the filaments of the interface. In this simulation, we applied the Winslow approach presented in section 3.2.7 to move the nodes of the bulk of the mesh. It is important to note that what is shown on both those figures is the phase of the computational mesh and not the position of the markers. The markers are advected by the velocity field which is analytical, so the only error on the interface position comes from the time integration scheme. By showing the phase of the computational mesh, we can also verify that the mesh deformation algorithm is able to follow the complex dynamics of the interface.

4.4.2 Rayleigh-Taylor instability

We consider the same Rayleigh-Taylor instability as in the section 4.3.3 with $At = 0.5$ and $Re = 256$. We considered a much coarser mesh than in the previous section with only 7665 vertices. Results are shown in Figure 4.14 and are in good agreement with the both the reference solution from He et al [He



Figure 4.12: Vortex in a box at time $t = 5$.

et al. (1999)] and the results obtained with the levelset approach on a much finer mesh. Figure 4.15 shows a zoomed view of the interface at time $t = 5$ with the computational mesh. We can see that despite having very fine filaments in the flow, we are still able to represent them correctly with the front-tracking approach. In this simulation, we have used the Winslow energy approach presented in section 3.2.7 to deform the mesh in the bulk. This allows us to *attract* the mesh nodes towards the interface and thus be able to capture smaller details of the flow. We can notice the effect of this approach on the mesh in Figure 4.15 where the elements are smaller near the interface and larger far from it.

4.4.3 Ellipsoidal bubble relaxation

To further demonstrate the ability of the front-tracking approach to represent complex interface dynamics, we consider the relaxation of an ellipsoidal bubble. The initial shape of the bubble is an ellipse with a major axis of 3 and a minor axis of 2, centered in a computational domain of size 10 by 10. The parameters of the two fluids are chosen such that:

$\rho_1 = 0.01$ $\rho_2 = 1$ $\mu_1 = 5 \cdot 10^{-5}$ $\mu_2 = 0.01$ $\sigma = 1$

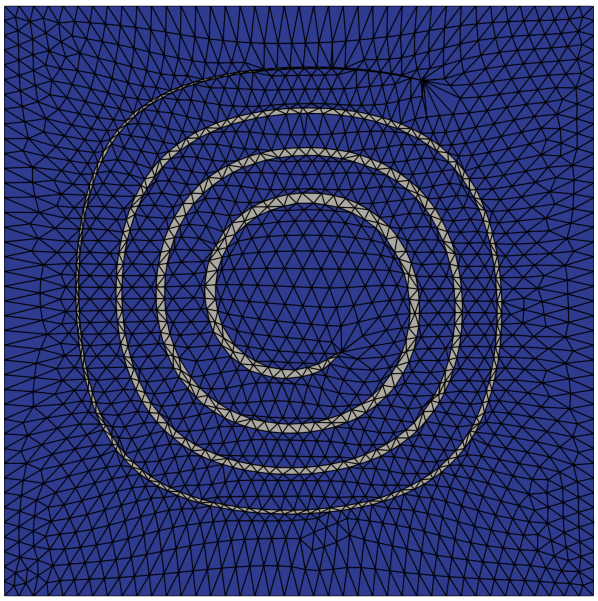


Figure 4.13: Vortex in a box at time $t = 5$ with the computational mesh.

with units in $[kg/m^3]$, $[kg/(m.s)]$ and $[N/m]$ for the densities, the dynamic viscosities and the surface tension respectively. The bubble relaxes to a circular shape due to surface tension forces and oscillates around this shape due to inertia effects. Figure 4.16 shows the evolution of the bubble shape over time. We can see that the front-tracking approach is able to represent the interface dynamics in a sharp manner without any diffusion of the interface. This test-case is particularly interesting when considering energy conservation since the initial energy is purely surface energy and as the bubble relaxes, this surface energy is converted into kinetic energy of the fluids. Figure 4.17 shows the evolution of the different forms of energy during the simulation. The potential energy of surface tension can be computed as:

$$E_\sigma = \sigma|\Gamma|$$

with $|\Gamma|$ the perimeter of the bubble. The kinetic energy is computed as:

$$E_k = \frac{1}{2}\int_\Omega \rho||\mathbf{u}||^2 d\Omega$$

and the total energy is the sum of both. Let's also note that the system naturally dissipates energy due to viscous effects. This energy can be computed

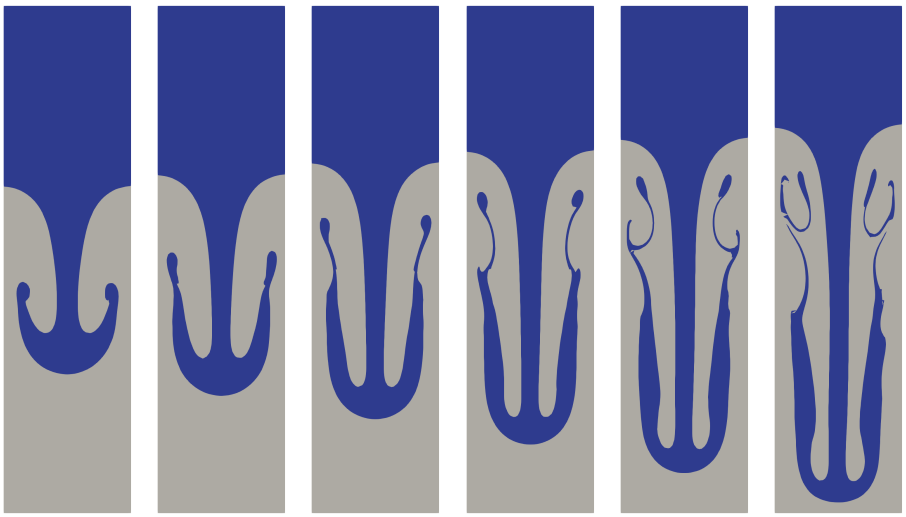


Figure 4.14: Rayleigh-Taylor instability for $Re = 256$, $At = 0.5$ at times $t = 3, 3.5, 4, 4.5, 5$ and 5.5 with the front-tracking approach on a coarse mesh.

as:

$$E_\mu = \int_0^t \int_\Omega 2\mu \mathbf{D} : \mathbf{D} d\Omega dt$$

with $\mathbf{D}$ the rate of deformation tensor defined as $\mathbf{D} = \frac{1}{2}\left(\nabla \mathbf{u} + \nabla \mathbf{u}^T\right)$. We can see in Figure 4.17 that the total energy decreases over time due to viscous dissipation, but the sum of the kinetic energy, the surface energy and the dissipated energy grows over time. This is due to numerical errors in our scheme and could be improved by considering a more accurate time integration scheme for the marker position or the incorporation of the surface tension. For now, we considered an explicit treatment of those two steps, leading to an energy that does not decreases fast enough.

4.4.4 Single bubble rising - Case 1

We consider the benchmark of Hysing et al. [Hysing et al. (2009)] for a free rising bubble in 2D with the same parameters as in the section 4.3.4. The benchmark is composed of four quantities to compare: the center of mass of the bubble, the mean velocity of the bubble, the circularity and the mass (area) of the bubble. To observe the convergence of the method, this simulation is carried out with different levels of mesh refinement: $h = \frac{1}{40}, \frac{1}{80}$ and $\frac{1}{160}$ where h is the characteristic size of an element. A reference solution is also displayed,

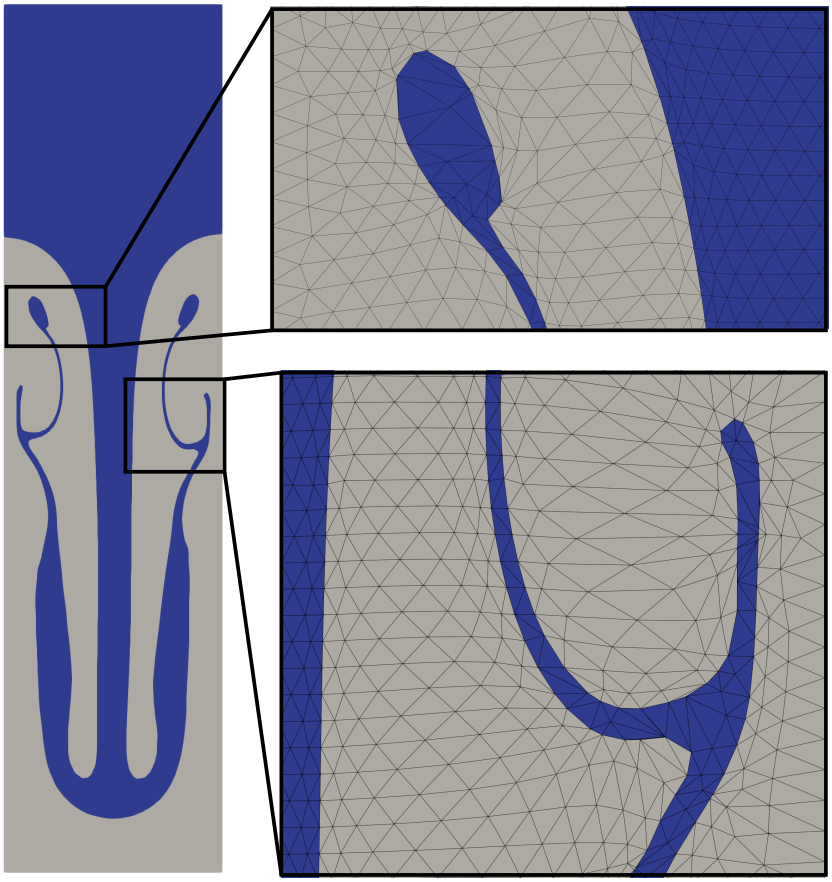
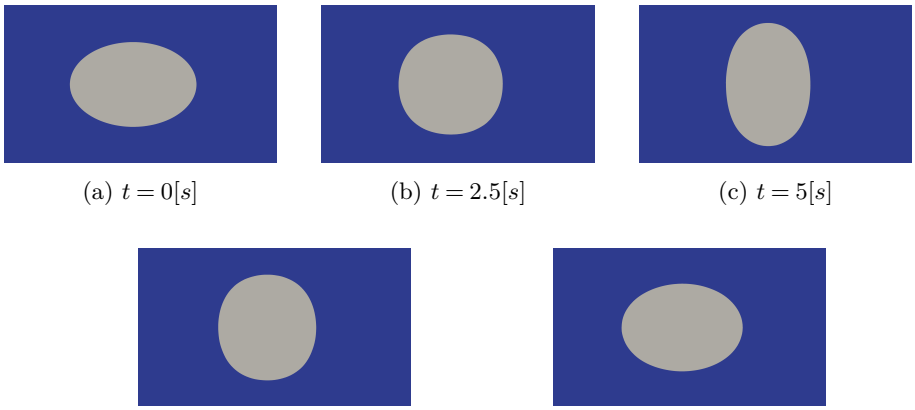


Figure 4.15: Zoomed view of the interface at time $t = 5.5$ with the computational mesh.

the solution obtained by the TP2D solver [Hysing et al. (2009),Turek (1997)] for a mesh size $h = \frac{1}{320}$. In comparison to the results obtained with the levelset approach, we can see that the mass conservation (Figure 4.19) is significantly improved with a variation of only 0.5 % for the coarsest mesh for a variation of 4 % with the levelset approach on the same mesh. It is also very important to note that in the case of front-tracking, the total mass of the bubble is oscillating around the initial mass, whereas in the levelset approach, the mass is continuously decreasing. In the front-tracking approach, we deform the mesh to follow the interface represented by markers and thus obtain a P1



(a) $t = 0[s]$ (b) $t = 2.5[s]$ (c) $t = 5[s]$

(d) $t = 7.5[s]$ (e) $t = 10[s]$

Figure 4.16: Evolution of an ellipsoidal bubble relaxing under the effect of surface tension at different times.

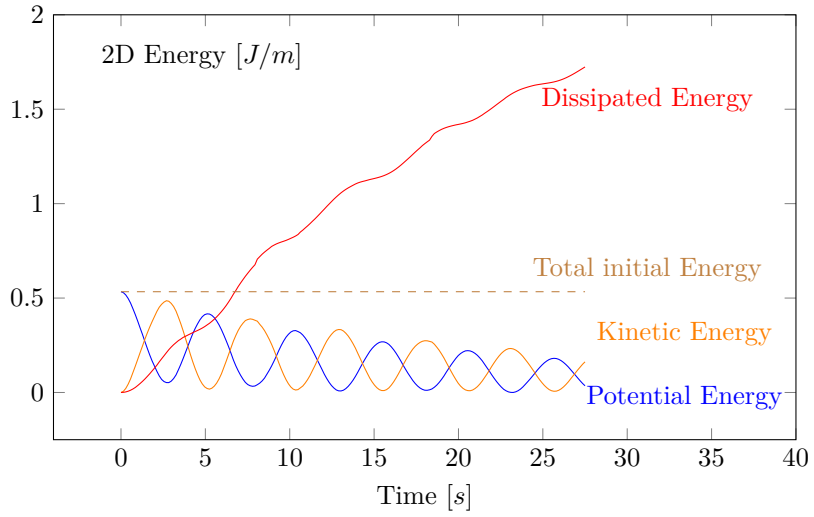


Figure 4.17: Evolution of the different energy contributions for an ellipsoidal oscillating interface in 2D using markers.

interpolation of the interface. As the simulation progresses, this interpolation vary which leads to gain or loss of mass. However, this variation is limited and does not accumulate over time since the markers are not affected by this phenomenon. In the levelset approach, when we do the redistancing step, we modify the levelset function and thus the position of the interface. Since the

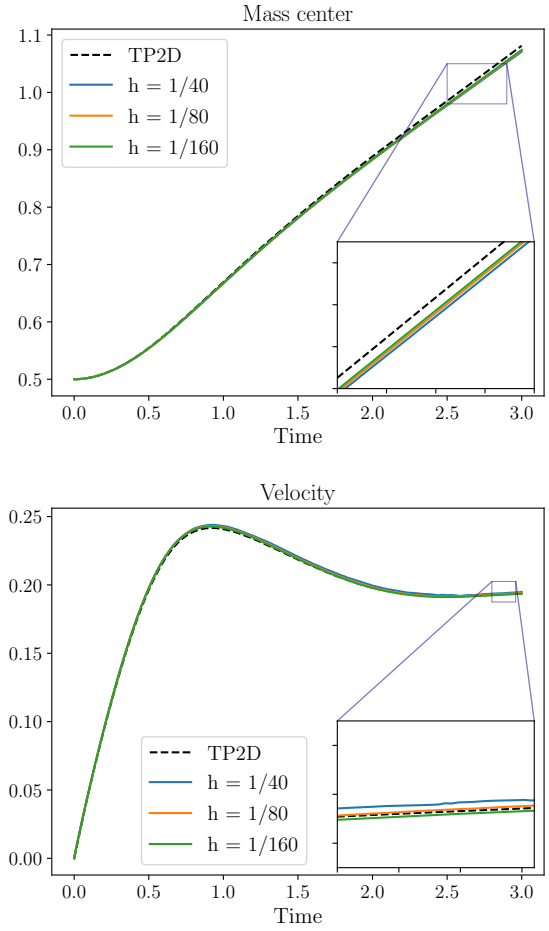


Figure 4.18: Results of the benchmark of Hysing et al. for a free rising bubble - case 1, part 1.

Hysing bubble is convex, the redistancing step systematically leads to a loss of mass which accumulates over time.

The other quantities are in good agreement with the reference solution and the results obtained with the levelset approach as we can see it in Figures 4.18 and 4.19. But we can see that the markers approach underestimates both the center of mass position and the circularity of the bubble. Multiple reasons could be the cause of this discrepancy. The first one is that we considered a first-order time integration scheme for the marker position (Explicit-Euler)

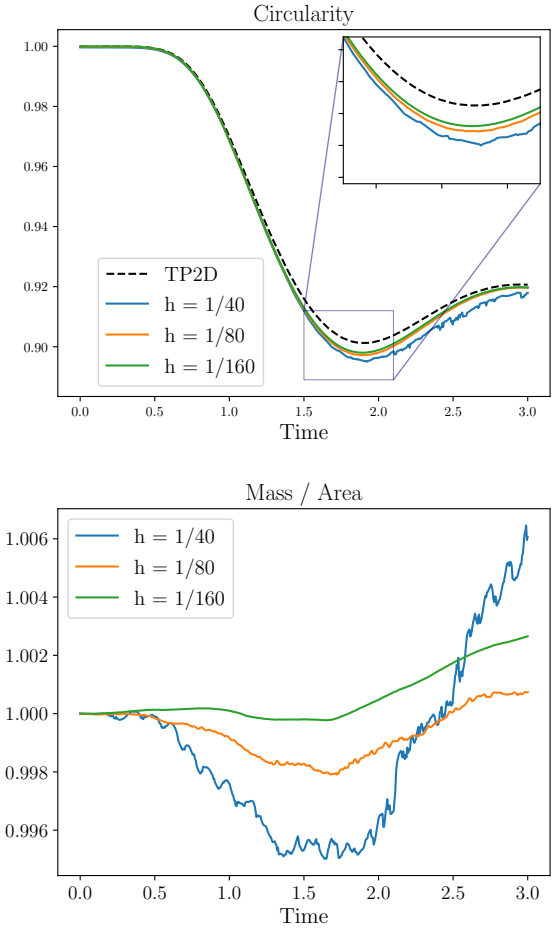


Figure 4.19: Results of the benchmark of Hysing et al. for a free rising bubble - case1, part 2.

which could lead to an accumulating error on the marker position. Another reason could be that we considered an interpolation of the perfect circle as the markers initial position. This interpolation lead to a initial mass of the bubble that is lower than the perfect circle.

4.4.5 Single bubble rising - Case 2

This test case corresponds to the second case of the benchmark of Hysing et al. for a free rising bubble in 2D with the following parameters:

$$\rho_1 = 1000 \qquad \rho_2 = 1 \qquad \mu_1 = 10 \qquad \mu_2 = 0.1 \qquad g = 0.98 \qquad \sigma = 1.96$$

We consider the same mesh refinements as in the previous section: $h = \frac{1}{40}$, $\frac{1}{80}$ and $\frac{1}{160}$ where h is the characteristic size of an element. A reference solution is also displayed, the solution obtained by the TP2D solver [Hysing et al. (2009)] for a mesh size $h = \frac{1}{320}$. The higher density and viscosity ratios make this case more challenging than the previous one. The bubble undergoes larger deformations and the interface is subject to stronger shear forces. The results for the four benchmark quantities are shown in Figure 4.20 and 4.21. The evolution of the interface for different times is shown in Figure 4.22 and a comparison of the interface at time $t = 3$ for the different mesh sizes is shown in Figure 4.23. We can see that despite having a coarse mesh, the benchmark quantities are in good agreement with the reference solution. The mass conservation is also good with a variation of approximately 1 % in the worst case. The circularity of the bubble is highly dependent on the splitting of the bubble. Indeed, on our two coarser meshes, the bubble splits around time $t = 2.25$ creating two smaller *satellite* bubbles. But for the finest mesh, the filaments created during the stretching of the bubble are conserved much longer and the bubble splits later at time $t = 2.6$. This delay in the splitting of the bubble explains the difference in circularity between the different meshes. We can observe the effect of this delay in the interface position at time $t = 3$ shown in Figure 4.23.

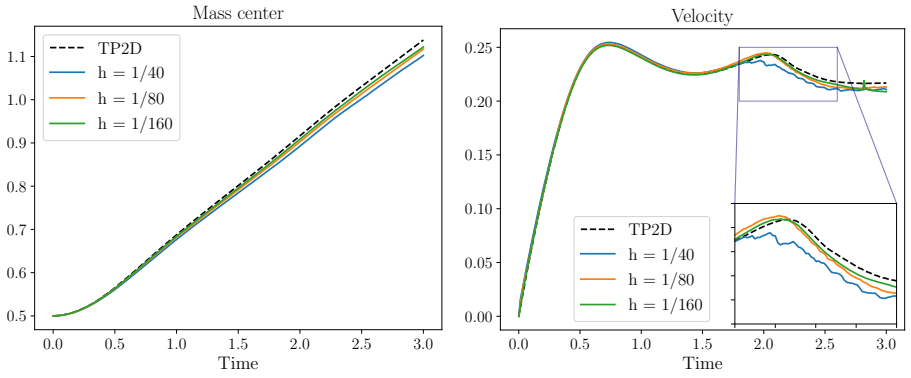


Figure 4.20: Results of the benchmark of Hysing et al. for a free rising bubble - case2, part 1.

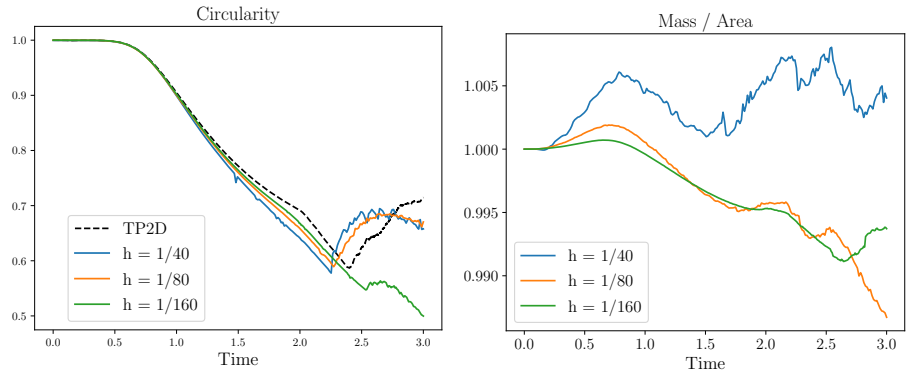


Figure 4.21: Results of the benchmark of Hysing et al. for a free rising bubble - case2, part 2.

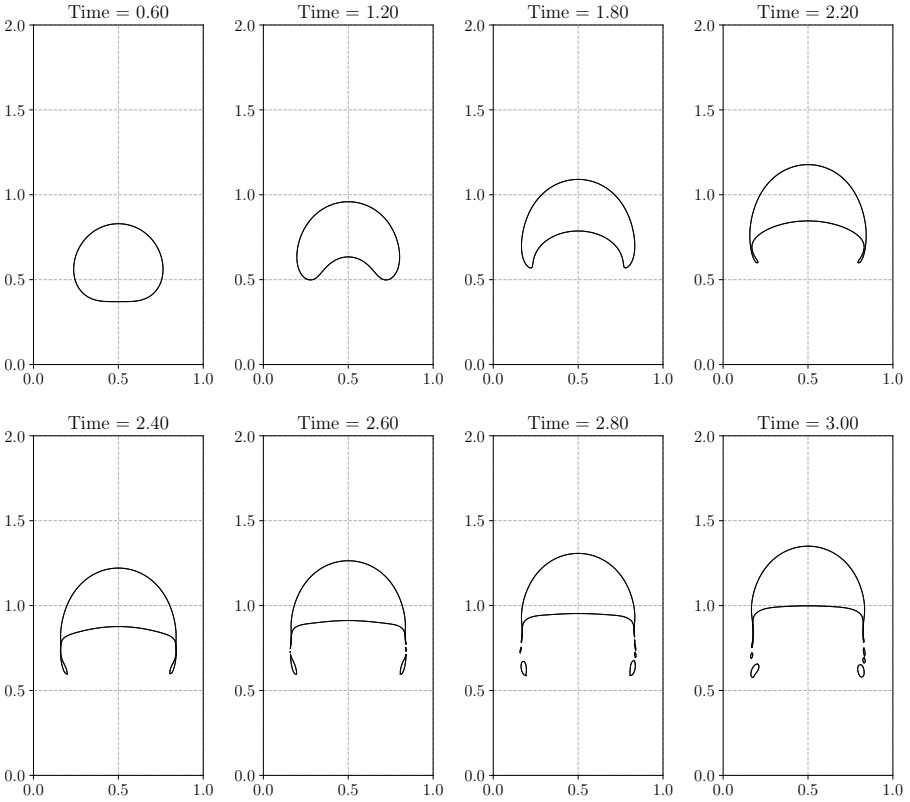


Figure 4.22: Evolution of the interface for the second case of the benchmark of Hysing et al. for a free rising bubble with a mesh size of $h = \frac{1}{160}$.

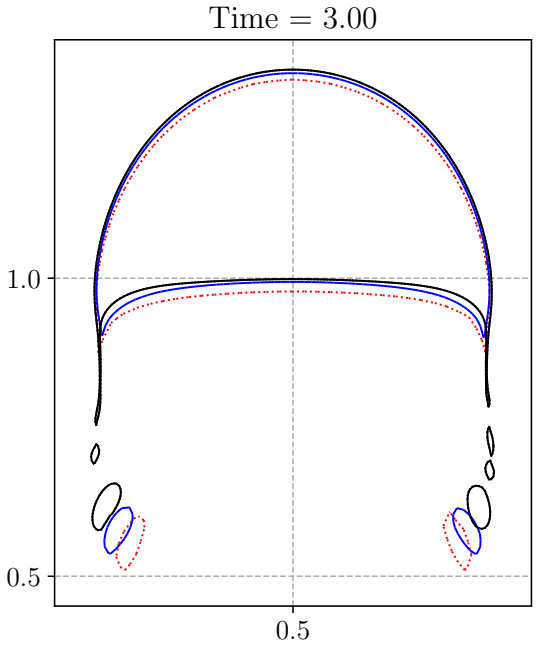


Figure 4.23: Comparison of the interface at time $t = 3$ for the different mesh sizes: $h = \frac{1}{40}$ (red dotted line).

4.5 Two-phase jet

To demonstrate the capabilities of the X-Mesh method for simulating complex two-phase flows with many topological changes, we consider the case of a two-phase jet in the transition to turbulence regime. This test case involves the injection of a fluid into another immiscible fluid at high velocity, leading to the formation of complex interfacial structures. The many topological changes occurring during the simulation make this test case particularly appropriate for evaluating the robustness of our handling of topological changes presented in section 3.2.3.

The computational domain is a 2D rectangle of size $W = 1\,[m]$ width and $H = 0.25\,[m]$ high with an inlet on the left side of size $0.05\,[m]$ where the fluid 1 is injected at a velocity of $U = 0.1\,[m/s]$. The fluid 2 fills the rest of the domain and is initially at rest and we consider wall boundary conditions on the top and bottom of the domain and an outlet condition on the right side. This domain is meshed with an unstructured triangular mesh of size $[200 \times 50]$ elements. The fluid 1 is water with properties $\rho_1 = 1000\,[kg/m^3]$ and $\mu_1 = 0.001\,[kg/(m.s)]$

and the dimensionless numbers characterizing the flow are:

$$Re = \frac{\rho_1 U D}{\mu_1} = 5000 \qquad \frac{\rho_1}{\rho_2} = 2 \qquad \frac{\mu_1}{\mu_2} = 2$$

Results are shown in Figure 4.24 where we can see the complex interfacial structures formed during the simulation. The front-tracking approach allows us to accurately capture the interface dynamics and the many topological changes occurring during the simulation. The topological changes are allowed when the distance between two parts of the interface is less than $h/10$ with $h = 5 \cdot 10^{-3}$ the mesh size. Let's note that Kelvin-Helmholtz instabilities develop at the interface between the two fluids due to the velocity difference, leading to the formation of vortices and bubbles of fluid 2 within fluid 1. Since we considered a 2D simulation, we don't apply surface tension forces at the interface as they would tend to overly stabilize the interface and prevent the development of these instabilities.

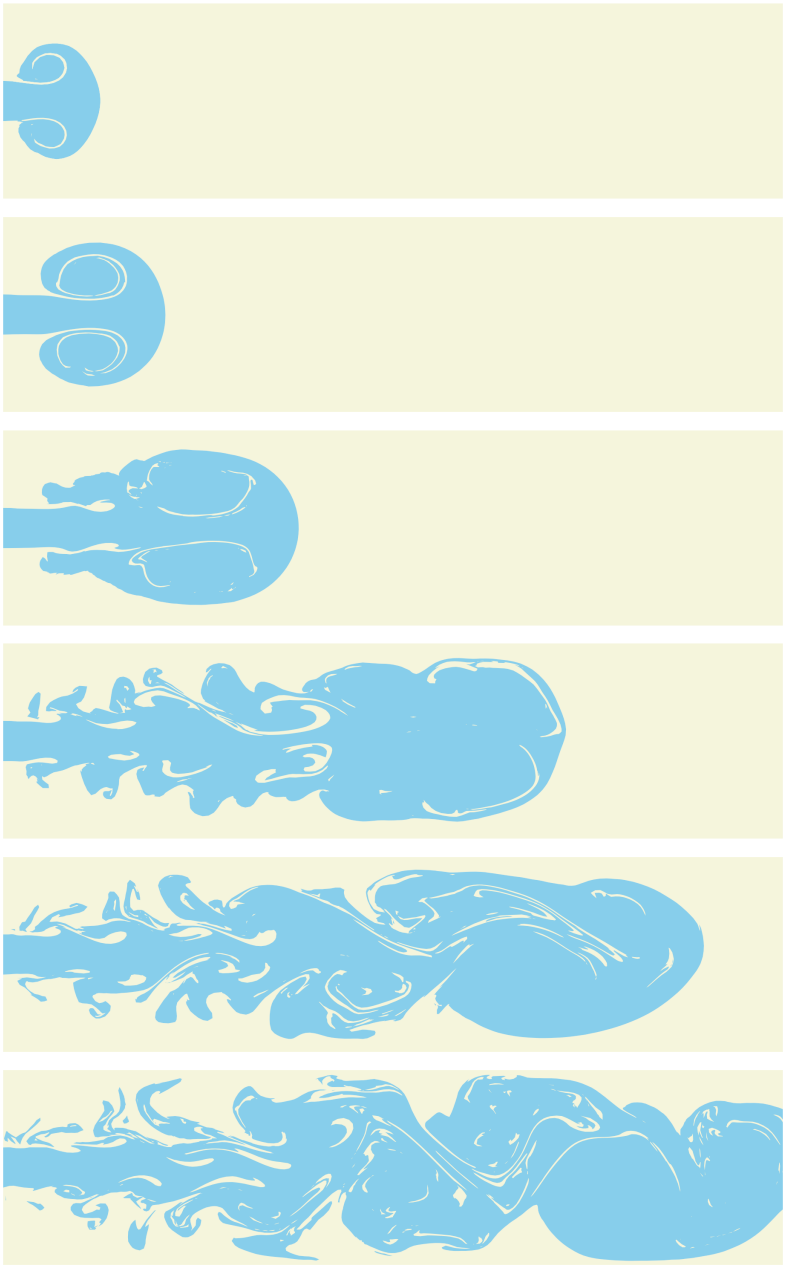


Figure 4.24: Turbulent two-phase jet at times $t = 0.1, 0.2, 0.4, 0.8, 1.0$ and 1.2 seconds.

CHAPTER 5

Conclusion

Contents

5.1 A summary of the contributions 134
5.2 Perspectives and research opportunities 135
5.3 Final words 140

Throughout this work, we have used the eXtreme Mesh deformation (X-Mesh) method to simulate two-phase flows with sharp interfaces. The key idea of this method is to allow degenerate elements in the mesh, which enables large deformations of the interface and topological changes without the need for remeshing. The method is very new and there are still many open questions regarding its theoretical properties and practical implementation. In this section we will summarize the contributions of this thesis and list some of the perspectives and potential research opportunities.

5.1 A summary of the contributions

In the first part of this thesis, we have presented the Tempered Finite Element Method (TFEM) for handling degenerate elements in the mesh. We have shown that the TFEM can be used to solve FEM problems with optimal convergence rates, even in the presence of band of caps in 2D or surface of slivers and other locking degenerate tetrahedra in 3D. This approach is easy to implement in existing FEM codes as it only requires a one-line modification of the computation of elements determinants. A limitation of TFEM is that it only works for clusters of locking degenerate elements of 1 dimension lower than the problem dimension. We are thus not able to handle situation like the Babuška and Aziz example from Figure 1.10 where the whole domain is locking. However, this limitation is not a problem for the X-Mesh method as we only need to handle degenerate elements that are created to be conforming to the interface, which are always of (at least) one dimension lower than the problem dimension. Thanks to TFEM, we greatly relax the constraints on the mesh deformation algorithm, especially in 3D where an algorithm that guarantees to avoid locking patterns would be very complex to find. The extension of TFEM to linear elasticity, mortaring of non-matching meshes, advection problems and higher order elements shows its versatility and robustness. The method is not limited to be used in the context of the X-Mesh method and could be used in other applications where degenerate elements appear.

In the second part of this thesis, we have presented two mesh deformation algorithms for the X-Mesh method. The first one is based on a levelset and is similar to what was presented in the first X-Mesh paper [Moës et al. (2023)]. The main difference appears in the careful treatment of the mesh deformation close to the boundary. This algorithm is robust and can handle large deformations of the interface as well as topological changes. The levelset description of the interface allows to easily manage topological changes. However, those changes are not handled in a very accurate way as they immediately occur when the phase is one element wide. The second algorithm was developed to improve the accuracy of the topological changes. It is based on a front-

tracking description of the interface and allows to control the size at which the topology changes of the interface occur. Since the X-Mesh method can represent arbitrary thin features thanks to the degenerate elements, we can delay the topological changes until the feature is very thin. This algorithm is more complex to implement and requires a careful management of the front-tracking interface. However, it allows to improve significantly the accuracy of the simulation of two-phase flows with topological changes.

Finally, we have presented several numerical examples in 2D to demonstrate the capabilities of the X-Mesh method. We have shown that the method can handle large deformations of the interface, topological changes and surface tension effects. The results are promising and show that the X-Mesh method is a viable alternative to other methods for simulating two-phase flows with sharp interfaces. Thanks to the Ghost Fluid Method, we can accurately impose the jump conditions at the interface and obtain an exactly sharp interface, which is not possible for a fixed mesh. We have also shown that the X-Mesh method can be used to simulate more complex two-phase flows, such as the rising bubble benchmark and Rayleigh-Taylor instabilities. The results are in good agreement with reference solutions and show the potential of the X-Mesh method for simulating complex two-phase flows.

5.2 Perspectives and research opportunities

The X-Mesh method is a very new method and there are still many open questions that merit further investigation. In two-phase flows, there is still much room for improvement and the method could be extended in several directions. Here are some of the perspectives and potential research opportunities:

Extension to 3D. The extension of the X-Mesh method to three-dimensional two-phase flows represents a natural next step. The primary challenge lies in developing a robust and efficient 3D mesh deformation algorithm that can prevent the formation of inverted elements. Although the 2D algorithms presented in this thesis could, in principle, be generalized to 3D, new difficulties specific to three-dimensional geometries are expected to emerge. In particular, it remains uncertain whether the proposed algorithms would effectively avoid inverted elements in 3D, especially in the case of the front-tracking-based approach. Moreover, managing the front-tracking interface in 3D introduces additional complexity, as it requires highly robust intersection algorithms. Within the X-Mesh framework, this challenge is further amplified by the presence of degenerate elements, which such intersection algorithms must handle precisely.

Fortunately, this issue has been addressed in the work of Bruno Lévy [Levy (2025)], offering a promising foundation for future developments.

Mass conservation. Except for the VOF method, most methods for simulating two-phase flows with sharp interfaces suffer from mass conservation issues. Our approach is not an exception and we have observed some mass loss in our simulations, especially in the presence of topological changes for the levelset-based mesh deformation algorithm. The front-tracking-based algorithm seems to be more robust in this regard, but still suffers from some mass loss. A potential solution to improve mass conservation would be to consider the advection of the markers as a minimisation problem under the constraint of mass conservation. Figure 5.1 presents preliminary results of this approach for the vortex-in-a-box test case. The markers are advected using a simple explicit Euler scheme (blue dashed line) and with a mass conservation constraint (black solid line). The explicit Euler scheme exhibits a mass loss of 39% at $t = 5s$, whereas the mass-conserving approach preserves mass exactly up to machine precision. The main challenge of this method lies in properly accounting for boundary conditions and possible topological changes of the interface. While this would be relatively straightforward for simple configurations, such as the rising bubble benchmark, it becomes more complex when the interface undergoes topological transitions or when it must interact with curved boundaries along which it must slide.

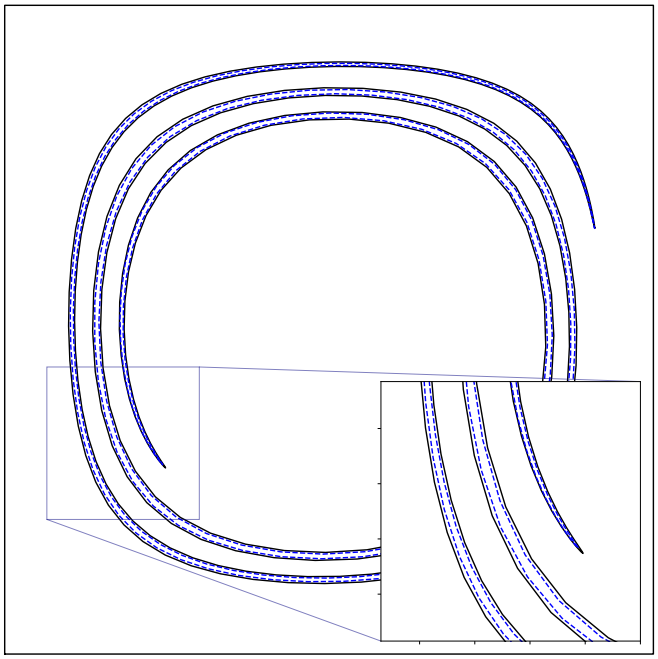


Figure 5.1: Interface at time $t = 5[s]$ for the vortex-in-a-box test case. In blue dashed line, the result of a simple Euler explicit advection of the markers. In black solid line, the result of the advection of the markers with a mass conservation constraint.

Fluid-structure interaction. Another natural extension of this thesis would be to consider fluid-structure interaction problems. The X-Mesh method is well suited for this type of problems as it can handle large displacements of the structure without the need for remeshing. The mesh deformation algorithm based on the markers was even developed with this type of problems in mind. Indeed, the ability to keep 0D features of the interface allows to represent structures with non-smooth geometries, such as sharp corners. The main challenge would be to take into account the phase change of elements transitioning from the fluid to the structure and vice versa as we relay the interface through the mesh. Preliminary results of a free falling ellipse in a viscous fluid are presented in Figure 5.2. We applied the mesh deformation algorithm based on the markers to the interface of the ellipse and used the equations

of a rigid body to compute its motion. The results are promising and show that the X-Mesh method can be used to simulate fluid-structure interaction problems. However, a lot of work remains to be done to accurately take into account the switching of elements from the fluid to the structure and vice versa.

Inverted elements. With the Tempered Finite Element Method, we can handle exactly degenerate elements so the natural next step would be to consider inverted elements. Since a 0^+ determinant or a 0^- determinant are the same from a numerical point of view, the question of handling inverted elements naturally arises. This would allow to relax even more the constraints on the mesh deformation algorithm and potentially improve the robustness of the method. Even outside of the context of the X-Mesh method, this could help in a lot of applications where a single inverted element can make the whole simulation fail. Preliminary results not presented in this thesis show that for a manufactured Laplacian problem, we can handle small inverted elements with optimal convergence rates and even regular size elements as long as the inverted domain converges in $\mathcal{O}(h)$.

Contact line dynamics. The simulation of contact line dynamics is a challenging problem in the context of two-phase flows. The X-Mesh method could be well suited for this type of problems as it can capture in 2D the triple point where the two fluids and the solid boundary meet. The implementation of contact line dynamics would require to take into account the contact angle condition at the triple point and its dynamics in the case of advancing or receding contact lines.

Contact simulations. Since the TFEM approach can be interpreted as a mortaring method, it could be interesting to consider contact simulations with the X-Mesh method. The main idea would be to add degenerate elements at the interface between two objects to allow for non-matching meshes and to *activate* them based on the contact conditions. Preliminary results in static small strain elasticity contact problems have been obtained and show the potential of this approach. However, a lot of work remains to be done to develop a robust and efficient algorithm for contact simulations with the X-Mesh method, especially in the case of dynamic contact.

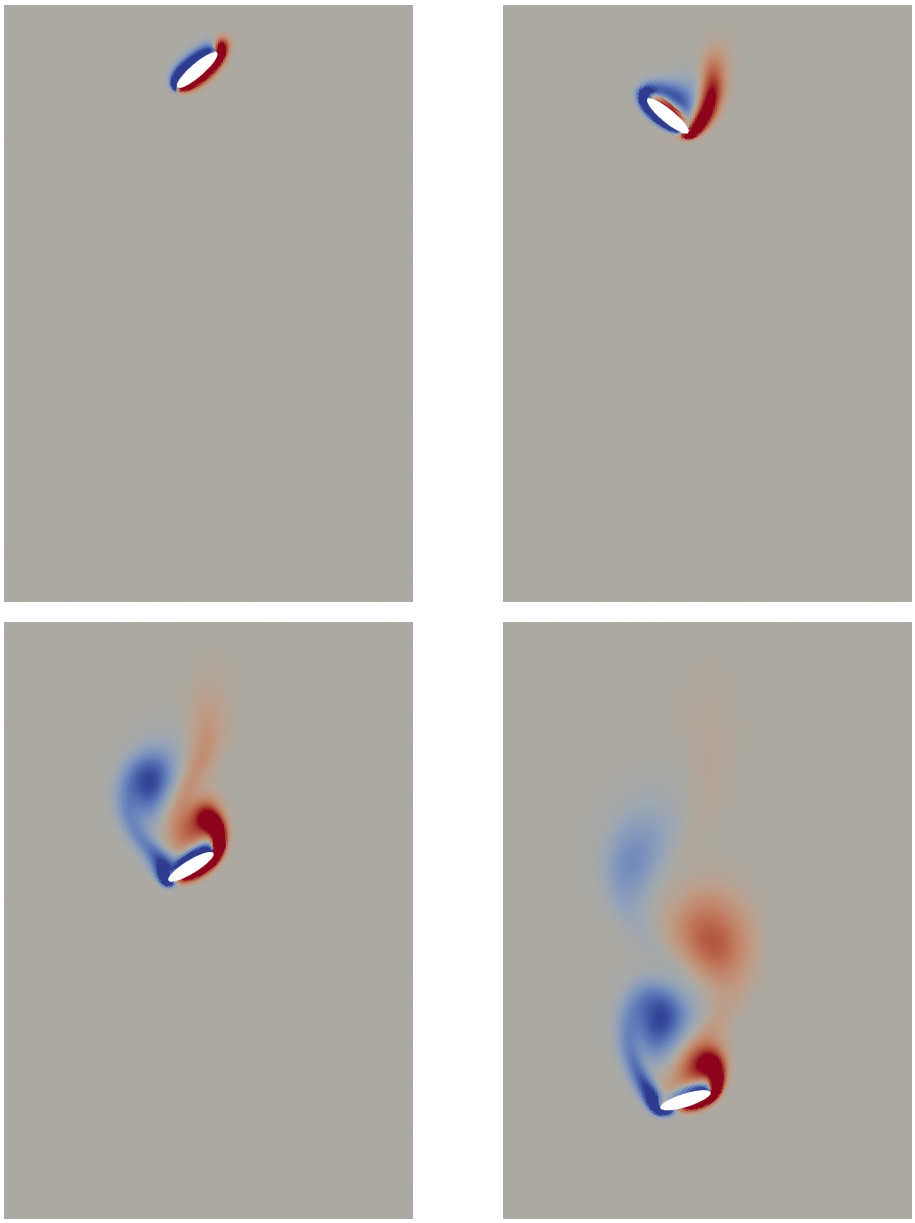


Figure 5.2: Vorticity field for the free falling of a solid ellipse in a steady regime at times $t = 0.1$, 0.33, 0.8 and $1.6[s]$.

5.3 Final words

With so many perspectives and open questions, one thing is certain: the X-Mesh method is still in its infancy, and much remains to be done to fully uncover its potential. One might even wonder whether this thesis provides more answers than it raises. Such is the nature of research, as the saying goes, *you don't truly finish a thesis; you simply stop working on it.*

Bibliography

Abu-Al-Saud, M.O., Soulaine, C., Riaz, A., Tchelepi, H.A., 2017. Level-set method for accurate modeling of two-phase immiscible flow with moving contact lines. arXiv: Fluid Dynamics .

Afanador, A.R., Zaleski, S., Tryggvason, G., Lu, J., 2021. Effect of topology changes on the breakup of a periodic liquid jet. Computers and Fluids 228, 105059.

Arnold, D., Brezzi, F., Fortin, M., 1984. A stable finite element for the stokes equations. Calcolo 21, 337–344.

Babuska, I., 1972/73. The finite element method with lagrangian multipliers. Numerische Mathematik 20, 179–192.

Babuška, I., 1973. The finite element method with penalty. J. Math. Comput. 27, 221–228.

Babuška, I., Aziz, A.K., 1976. On the angle condition in the finite element method. SIAM Journal on numerical analysis 13, 214–226.

Baidakova, N., 2020. Linear interpolation on a tetrahedron. Proc. Steklov Inst. Math. 308, S31–S34.

Baradat, A., 2019. Incompressible optimal transport : dependence to the data and entropic regularization. Theses. Université Paris Saclay (COmUE).

Barnhill, R.E., Gregory., J.A., 1976. Sard kernel theorems on triangular domains with applications to finite element error bounds. Numer. Math. 25, 215–229.

Brackbill, J., Kothe, D., Zemach, C., 1992. A continuum method for modeling surface tension. Journal of Computational Physics 100, 335–354.

Brooks, A.N., Hughes, T.J., 1982. Streamline upwind/petrov-galerkin formulations for convection dominated flows with particular emphasis on the incompressible navier-stokes equations. Computer methods in applied mechanics and engineering 32, 199–259.

Chang, Y., Hou, T., Merriman, B., Osher, S., 1996. A level set formulation of eulerian interface capturing methods for incompressible fluid flows. Journal of Computational Physics 124, 449–464.

Chirco, L., Maarek, J., Popinet, S., Zaleski, S., 2022. Manifold death: A volume of fluid implementation of controlled topological changes in thin sheets by the signature method. Journal of Computational Physics 467, 111468.

Chorin, A.J., 1968. Numerical solution of the navier-stokes equations. Mathematics of Computation 22, 745–762.

Chubynsky, M.V., Belousov, K.I., Lockerby, D.A., Sprittles, J.E., 2020. Bouncing off the walls: The influence of gas-kinetic and van der waals effects in drop impact. Phys. Rev. Lett. 124, 084501.

Crouzeix, M., Raviart, P.-A., 1973. Conforming and nonconforming finite element methods for solving the stationary stokes equations i. R.A.I.R.O. 7, 33–75.

Davis, P.J., 1975. Interpolation and Approximation. Dover, New York.

Drazin, P.G., Reid, W.H., 2004. Hydrodynamic stability. Cambridge university press.

Ern, A., Guermond, J.L., 2004. Theory and practice of finite elements. volume 159. Springer.

Farhat, C., Lesoinne, M., 2000. Two efficient staggered algorithms for the serial and parallel solution of three-dimensional nonlinear transient aeroelastic problems. Computer Methods in Applied Mechanics and Engineering 182, 499–515.

Fedkiw, R.P., Aslam, T., Merriman, B., Osher, S., 1999. A non-oscillatory eulerian approach to interfaces in multimaterial flows (the ghost fluid method). Journal of Computational Physics 152, 457–492.

Formaggia, L., Nobile, F., 2004. Stability analysis of second-order time accurate schemes for ale–fem. Computer Methods in Applied Mechanics and Engineering 193, 4097–4116. The Arbitrary Lagrangian-Eulerian Formulation.

Garanzha, V., Kaporin, I., Kudryavtseva, L., Protais, F., Ray, N., Sokolov, D., 2021. Foldover-free maps in 50 lines of code. ACM Transactions on Graphics 40, 1–16.

Geuzaine, C., Remacle, J.F., 2009. Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities. International journal for numerical methods in engineering 79, 1309–1331.

Grace, J., Wairegi, T., Nguyen, T., 1976. Shapes and velocities of single drops and bubbles moving freely through immiscible liquids. Trans Inst Chem Eng 54, 167–173.

Hannukainen, A., Korotov, S., Křížek, M., 2012. The maximum angle condition is not necessary for convergence of the finite element method. Numerische mathematik 120, 79–88.

He, X., Chen, S., Zhang, R., 1999. A lattice boltzmann scheme for incompressible multiphase flow and its application in simulation of rayleigh–taylor instability. Journal of computational physics 152, 642–663.

Hirt, C., Nichols, B., 1981. Volume of fluid (vof) method for the dynamics of free boundaries. Journal of Computational Physics 39, 201–225.

Hormann, K., Agathos, A., 2001. The point in polygon problem for arbitrary polygons. Computational Geometry 20, 131–144.

Hua, J., Lou, J., 2007. Numerical simulation of bubble rising in viscous liquid. Journal of Computational Physics 222, 769–795.

Hughes, T.J., Franca, L.P., Balestra, M., 1986. A new finite element formulation for computational fluid dynamics: V. circumventing the babuška-brezzi condition: A stable petrov-galerkin formulation of the stokes problem accommodating equal-order interpolations. Computer Methods in Applied Mechanics and Engineering 59, 85–99.

Hughes, T.J., Liu, W.K., Zimmermann, T.K., 1981. Lagrangian-eulerian finite element formulation for incompressible viscous flows. Computer Methods in Applied Mechanics and Engineering 29, 329–349.

Hysing, S., Turek, S., Kuzmin, D., Parolini, N., Burman, E., Ganesan, S., Tobiska, L., 2009. Quantitative benchmark computations of two-dimensional bubble dynamics. International Journal for Numerical Methods in Fluids 60, 1259 – 1288.

Jamet, P., 1976. Estimation de l'erreur pour des éléments finis droits presque dégénérés. Anal. Numer. 10, 43–60.

Kang, M., Fedkiw, R., Liu, X., 2000. A boundary condition capturing method for multiphase incompressible flow. Journal of Scientific Computing 15, 323–360.

Kobayashi, K., Tsuchiya, T., 2015. On the circumradius condition for piecewise linear triangular elements. Japan Journal of Industrial and Applied Mathematics 32, 65–76.

Korotov, S., Křížek, M., Kučera, V., 2022. On degenerating finite element tetrahedral partitions. Numer. Math. 152, 307–329.

Kučera, V., 2016. On necessary and sufficient conditions for finite element convergence. arXiv preprint arXiv:1601.02942 .

Kučera, V., 2016. Several notes on the circumradius condition. Appl. Math. 61, 287–298.

LeVeque, R.J., Li, Z., 1997. Immersed interface methods for stokes flow with elastic boundaries or surface tension. SIAM Journal on Scientific Computing 18, 709–735. arXiv:https://doi.org/10.1137/S1064827595282532.

Levy, B., 2025. Exact predicates, exact constructions and combinatorics for mesh csg. ACM Trans. Graph. 44.

Li, C.W., Wang, L.L., 2004. An immersed boundary finite difference method for les of flow around bluff shapes. International Journal for Numerical Methods in Fluids 46, 85–107. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/fld.749.

Marchandise, E., Remacle, J.F., 2006. A stabilized finite element method using a discontinuous level set approach for solving two phase incompressible flows. Journal of Computational Physics 219, 780–800.

Marchandise, E., et al., 2006. Simulation of three-dimensional two-phase flows: coupling of a stabilized finite element method with a discontinuous level set approach. Ph.D. thesis.

Moes, N., Remacle, J., Lambrechts, J., Le, B., 2021. The extreme mesh deformation approach (X-MESH) for the stefan phase-change model. CoRR abs/2111.04179. arXiv:2111.04179.

Moës, N., Remacle, J.F., Lambrechts, J., Lé, B., Chevaugeon, N., 2023. The extreme mesh deformation approach (x-mesh) for the stefan phase change model. Journal of Computational Physics 477, 111878.

Moës, N., Cloirec, M., Cartraud, P., Remacle, J.F., 2003. A computational approach to handle complex microstructure geometries. Computer Methods in Applied Mechanics and Engineering 192, 3163–3177. Multiscale Computational Mechanics for Materials and Structures.

Moës, N., Dolbow, J., Belytschko, T., 1999. A finite element method for crack growth without remeshing. International Journal for Numerical Methods in Engineering 46, 131–150.

Nagrath, S., Jansen, K.E., Lahey, R.T., 2005. Computation of incompressible bubble dynamics with a stabilized finite element level set method. Computer Methods in Applied Mechanics and Engineering 194, 4565–4587.

Olsson, E., Kreiss, G., 2005. A conservative level set method for two phase flow. Journal of Computational Physics 210, 225–246.

Osher, S., Sethian, J.A., 1988. Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations. Journal of Computational Physics 79, 12–49.

Patankar, S., Spalding, D., 1972. A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. International Journal of Heat and Mass Transfer 15, 1787–1806.

Popinet, S., 2018. Numerical models of surface tension. Annual Review of Fluid Mechanics 50, 49–75.

Popinet, S., Zaleski, S., 1999. A front-tracking algorithm for accurate representation of surface tension. International Journal for Numerical Methods in Fluids 30, 775–793.

Prosperetti, A., Tryggvason, G., 2009. Computational methods for multiphase flow. Cambridge university press.

Quiriny, A., Kučera, V., Lambrechts, J., Moës, N., Remacle, J.F., 2026. The tempered finite element method. Journal of Computational Physics 549, 114615.

Quiriny, A., Lambrechts, J., Moës, N., Remacle, J.F., 2024. X-mesh: A new approach for the simulation of two-phase flow with sharp interface. Journal of Computational Physics 501, 112775.

Sethian, J.A., 1996. A fast marching level set method for monotonically advancing fronts. Proceedings of the National Academy of Sciences 93, 1591–1595.

Sussman, M., Fatemi, E., 1999. An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow. SIAM Journal on Scientific Computing 20, 1165–1191. arXiv:https://doi.org/10.1137/S1064827596298245.

Taylor, C., Hood, P., 1973. A numerical solution of the navier-stokes equations using the finite element technique. Computers and Fluids 1, 73–100.

Tryggvason, G., Bunner, B., Esmaeeli, A., Juric, D., Al-Rawahi, N., Tauber, W., Han, J., Nas, S., Jan, Y.J., 2001. A front-tracking method for the computations of multiphase flow. Journal of Computational Physics 169, 708–759.

Turek, S., 1997. On discrete projection methods for the incompressible navier-stokes equations: An algorithmical approach. Computer Methods in Applied Mechanics and Engineering 143, 271–288.

Unverdi, S.O., Tryggvason, G., 1992. A front-tracking method for viscous, incompressible, multi-fluid flows. Journal of Computational Physics 100, 25–37.

Ženíšek, A., 1969. The convergence of the finite element method for boundary value problems of a system of elliptic equations. Apl. Mat. 14, 355–377.

Wriggers, P., 2008. Nonlinear finite element methods. Springer Science & Business Media.

Wu, G., Eatock Taylor, R., Greaves, D., 2001. The effect of viscosity on the transient free-surface waves in a two-dimensional tank. Journal of Engineering Mathematics 40, 77–90.

Xu, L., Zhang, W.W., Nagel, S.R., 2005. Drop splashing on a dry smooth surface. Phys. Rev. Lett. 94, 184505.

Zlámal, M., 1968. On the finite element method. Numer. Math. 12, 394–409.



Appendix

A.1 Appendix: Study of the FEM stiffness matrix

Let's study the stiffness matrix related to the Laplace operator on a triangle (see Figure A.1):

$$K_{ij} = \int_T \nabla\phi_i \cdot \nabla\phi_j \, dx \quad , \quad i,j = 1,2,3.$$

Assume three scalar values u_i, $i = 1,2,3$, associated to nodes located at x_i, $i = 1,2,3$. The stiffness matrix, $[K]$, allows one to write the quadratic expression of the energy:

$$f(u_1,u_2,u_3) = \frac{1}{2}\sum_{i=1}^{3}\sum_{j=1}^{3} K_{ij}u_iu_j. \tag{A.1}$$

It turns out that $[K]$ only depends on the triangle angles denoted θ_i's:

$$[K] = \begin{bmatrix} \cot(\theta_2)+\cot(\theta_3) & -\cot(\theta_3) & -\cot(\theta_2) \\ -\cot(\theta_3) & \cot(\theta_1)+\cot(\theta_3) & -\cot(\theta_1) \\ -\cot(\theta_2) & -\cot(\theta_1) & \cot(\theta_1)+\cot(\theta_2) \end{bmatrix} = \begin{bmatrix} c_2+c_3 & -c_3 & -c_2 \\ -c_3 & c_1+c_3 & -c_1 \\ -c_2 & -c_1 & c_1+c_2 \end{bmatrix} \tag{A.2}$$

Without loss of generality, we assume that $\theta_3 \geq \theta_2 \geq \theta_1$. Edge (x_1,x_2) is thus the longest edge of T (see Figure A.1) with $\|x_2 - x_1\| = l_1 + l_2$. We define

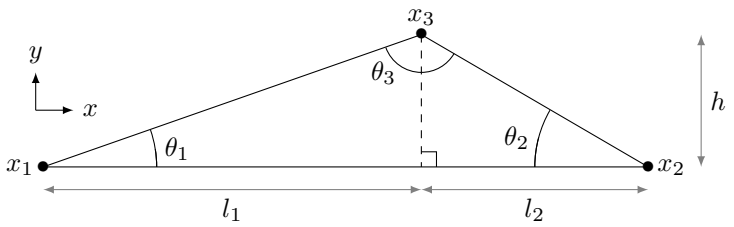


Figure A.1: A triangle T. The angle θ_i is associated to the vertex x_i.

the element flatness f and its symmetry s with the following expression

$$f = \frac{h}{l_1 + l_2}, \quad s = \frac{l_1}{l_1 + l_2}, \quad f \in \left[0, \sqrt{3}/2\right], \quad s \in [0,1]. \tag{A.3}$$

The cotangents may be expressed as a function of f and s:

$$c_1 = \frac{s}{f}, \quad c_2 = \frac{1-s}{f}, \quad c_3 = f - \frac{s(1-s)}{f}.$$

Eigenvalues $\lambda_1 \leq \lambda_2 \leq \lambda_3$ of $[K]$ can be expressed as functions of $c = c_1 + c_2 + c_3 = f + \frac{1-s+s^2}{f}$ as:

$$\lambda_1 = 0, \quad \lambda_2 = c - \sqrt{c^2 - 3}, \quad \lambda_3 = c + \sqrt{c^2 - 3}.$$

whereas the corresponding eigenvectors are

$$\begin{aligned}
&\mathbf{v}_1 = [1,1,1],\\
&\mathbf{v}_2 = [c_3 - c_1 - \sqrt{c^2-3}, -c_3 + c_2 + \sqrt{c^2-3}, c_1 - c_2],\\
&\mathbf{v}_3 = [1-\alpha, \alpha, -1], \quad \alpha = \frac{2c_1 - c_2 - c_3 + \sqrt{c^2-3}}{c_1 + c_2 - 2c_3 + 2\sqrt{c^2-3}} \in [0,1].
\end{aligned}$$

Finally, the energy e_i associated to each mode is given by $e_i = \lambda_i \|\mathbf{v}_i\|^2/2$.

$$\begin{aligned}
e_1 &= 0,\\
e_2 &= \frac{(c-\beta)}{2}\left[(c_1 - c_2)^2 + (-c_1 + c_3 - \beta)^2 + (c_2 - c_3 + \beta)^2\right],\\
e_3 &= \frac{(c-\beta)}{2}\left[\left((1 - \frac{2c_1 - c_2 - c_3 + \beta}{c_1 + c_2 - 2c_3 + 2\beta}\right)^2 + 1 + \left(\frac{2c_1 - c_2 - c_3 + \beta}{c_1 + c_2 - 2c_3 + 2\beta}\right)^2\right],
\end{aligned}$$

with $\beta = \sqrt{c^2 - 3}$.

We note that the first mode is the rigid mode with zero energy. We are interested in the limit case $f \to 0$ for which the element reaches zero measure. It is possible to show using a Taylor expansion that as $f \to 0$ we have

$$\lambda_2 \sim \frac{3}{2}\frac{f}{1-s+s^2} \quad \text{and} \quad \lambda_3 \sim 2\frac{1-s+s^2}{f}. \tag{A.4}$$

A.2 Appendix: Analysis of tempered FEM on zero-measure elements

Now we analyze the limiting scheme (2.43) and prove its convergence. There will be two technical issues with this analysis:

- The scheme is not consistent and we must deal with a consistency error.
- As noted in Remark 2, the second term in (2.43) is essentially a quadrature approximation of an integral, while the non-consistency term will be in integral form and this discrepancy must be dealt with.

First, we detail the equation that the exact solution u satisfies on $\Omega \setminus \Gamma$, if we use $v_h \in V_h^\Gamma$ as a test function.

Lemma 9. *The exact solution u of (2.5) satisfies*

$$\int_{\Omega\setminus\Gamma} \nabla u \cdot \nabla v_h \,\mathrm{d}x - \int_\Gamma \nabla u \cdot n_\Gamma [v_h] \,\mathrm{d}S + \widetilde{D}\frac{h}{2}\sum_{x_i\in\Gamma} [u]_{x_i}[v_h]_{x_i} = (f, v_h), \tag{A.5}$$

for all $v_h \in V_h^\Gamma$.

Proof. We multiply problem (2.5) by v_h, integrate over $\Omega \setminus \Gamma$ and apply Green's theorem. Since v_h is in general discontinuous on Γ, we obtain

$$\int_{\Omega\setminus\Gamma} \nabla u \cdot \nabla v_h \,\mathrm{d}x - \int_\Gamma \nabla u \cdot n_\Gamma [v_h] \,\mathrm{d}S = (f, v_h). \tag{A.6}$$

Since $[u] = 0$ on Γ, we can add the last left-hand side term in (A.5) to the formulation as it is equal to zero. □

Now we address the second problem.

Lemma 10. *Let $f : \Gamma \to \mathbb{R}$ be a piecewise linear function on the partition $x_i, i = 0, \ldots, N+1$, such that $f(x_0) = f(x_{N+1}) = 0$. Then*

$$\int_\Gamma f^2 \,\mathrm{d}S \le \frac{h}{2}\sum_{x_i\in\Gamma} f^2(x_i). \tag{A.7}$$

Proof. For simplicity, we can consider Γ to be the one-dimensional interval $[0, L]$ with corresponding nodes x_i. We consider two neighboring points x_i, x_{i+1} of the partition and estimate the integral over this segment of Γ, which has length $h/2$. Since f is linear on this segment, we can write $f(x) = f(x_i)\phi_i(x) + f(x_{i+1})\phi_{i+1}(x)$, where $\phi_i(x) = \frac{2}{h}(x_{i+1} - x)$ and $\phi_{i+1}(x) = \frac{2}{h}(x - x_i)$ are the standard Lagrange basis functions in 1D, for which the necessary integrals can be easily evaluated. Therefore

$$\begin{aligned}
&\int_{x_i}^{x_{i+1}} f^2(x)\,\mathrm{d}x \\
&= \int_{x_i}^{x_{i+1}} f^2(x_i)\phi_i^2(x) + 2f(x_i)f(x_{i+1})\phi_i(x)\phi_{i+1}(x) + f^2(x_{i+1})\phi_{i+1}^2(x)\,\mathrm{d}x \\
&= \frac{h}{6}f^2(x_i) + 2\frac{h}{12}f(x_i)f(x_{i+1}) + \frac{h}{6}f^2(x_{i+1}) \leq \frac{h}{4}f^2(x_i) + \frac{h}{4}f^2(x_{i+1}),
\end{aligned} \tag{A.8}$$

since $2ab \leq a^2 + b^2$. Summing (A.8) over all $i = 0, \ldots, N$ and taking into account $f(x_0) = f(x_{N+1}) = 0$ gives us (A.7). □

We will eventually apply Lemma 10 to the function $f = [\xi_h]$. For this it is important to realize that $[\xi_h]$ is a piecewise linear function on the partition given by the nodes x_i. Indeed, $[\xi_h] = \xi_h^- - \xi_h^+$, and ξ_h^- is linear on the segments between x_i, x_{i+2} for some i, while ξ_h^+ is linear on the segment between x_{i-1}, x_{i+1}, cf. Figure 2.14. Therefore, their difference $[\xi_h]$ is linear on the intersection, i.e. on the segment between x_i, x_{i+1}. Furthermore, $[\xi_h]_{x_0} = [\xi_h]_{x_{N+1}} = 0$, since ξ_h is continuous at the endpoints of Γ.

Next, we need an estimate of the interpolation error on Γ.

Lemma 11. *It holds that*

$$\frac{h}{2}\sum_{x_i \in \Gamma} [u - \Pi_h u]_{x_i}^2 \leq \frac{1}{64} L h^4 \|\nabla_\parallel^2 u\|_{L^\infty(\Gamma)}^2, \tag{A.9}$$

$$\int_\Gamma [u - \Pi_h u]^2\,\mathrm{d}S \leq \frac{1}{16} L h^4 \|\nabla_\parallel^2 u\|_{L^\infty(\Gamma)}^2, \tag{A.10}$$

where $\nabla_\parallel^2 u$ is the second derivative of u in the direction parallel to Γ.

Proof. First estimate: Since each x_i is the vertex of an element neighboring Γ, and $\Pi_h u$ interpolates u at vertices, we have either $(u - \Pi_h u)^-(x_i) = 0$ or $(u - \Pi_h u)^+(x_i) = 0$. Let e.g. the latter case hold. The point x_i is the midpoint of the edge of an adjacent element K^- and $(u - \Pi_h u)^-(x_i)$ is simply the interpolation error of the 1D linear Lagrange interpolation of u on the edge of K^- with endpoints x_{i-1}, x_{i+1}. Cauchy's theorem for the remainder of linear Lagrange

interpolation on an interval can be applied, cf. Davis (1975), which guarantees the existence of a point ζ on the edge, such that

$$\big|(u-\Pi_h u)^-(x_i)\big| = \frac{1}{2}\big|\nabla_\parallel^2 u(\zeta)(x_i-x_{i-1})(x_i-x_{i+1})\big| \le \frac{1}{2}\|\nabla_\parallel^2 u\|_{L^\infty(\Gamma)}\frac{h}{2}\frac{h}{2}. \tag{A.11}$$

Using this estimate, we get

$$\begin{aligned}\frac{h}{2}\sum_{x_i\in\Gamma}[u-\Pi_h u]_{x_i}^2 &\le \frac{h}{2}\sum_{x_i\in\Gamma}\big(\tfrac{1}{8}h^2\|\nabla_\parallel^2 u\|_{L^\infty(\Omega)}\big)^2\\ &= \frac{1}{64}h^4\|\nabla_\parallel^2 u\|_{L^\infty(\Omega)}^2\sum_{x_i\in\Gamma}\frac{h}{2} = \frac{1}{64}Lh^4\|\nabla_\parallel^2 u\|_{L^\infty(\Omega)}^2,\end{aligned} \tag{A.12}$$

since $\sum_{x_i\in\Gamma}\frac{h}{2} = L$. This gives us (A.9).

Second estimate: We have

$$\begin{aligned}\int_\Gamma [u-\Pi_h u]^2\,\mathrm{d}S &= \int_\Gamma\big((u-\Pi_h u)^- - (u-\Pi_h u)^+\big)^2\,\mathrm{d}S\\ &\le 2\int_\Gamma\big((u-\Pi_h u)^-\big)^2\,\mathrm{d}S + 2\int_\Gamma\big((u-\Pi_h u)^+\big)^2\,\mathrm{d}S.\end{aligned} \tag{A.13}$$

The right-hand side integrals can be written as the sum over edges of neighboring elements that lie on Γ. On these edges, $u-\Pi_h u$ is simply the error of the 1D linear Lagrange interpolation of u on the edge. Similarly as in the proof of Lemma 10, we can consider Γ to be the one-dimensional interval $[0,L]$ with corresponding nodes x_i. An edge of an element neighboring Γ is an interval $[x_i,x_{i+2}]$ of length h, and Cauchy's theorem for the remainder of linear Lagrange interpolation gives for each x a ζ_x such that

$$\begin{aligned}\int_{x_i}^{x_{i+2}}(u-\Pi_h u)^2\,\mathrm{d}S &= \int_{x_i}^{x_{i+2}}\left(\frac{1}{2}\nabla_\parallel^2 u(\zeta_x)(x-x_i)(x-x_{i+2})\right)^2\mathrm{d}S\\ &\le \frac{1}{4}\|\nabla_\parallel^2 u\|_{L^\infty(\Gamma)}^2\left(\frac{h}{2}\right)^4\int_{x_i}^{x_{i+2}}\mathrm{d}S,\end{aligned} \tag{A.14}$$

since $\big((x-x_i)(x-x_{i+2})\big)^2$ attains its maximum at $x=x_{i+1}$ with a value of $(h/2)^4$. When we sum (A.14) over all elements neighboring Γ, the last integral in (A.14) sums up to L. Using this estimate in (A.13) gives us (A.10). □

We note that the constant in (A.10) can be improved, e.g. by evaluating the integral of $\big((x-x_i)(x-x_{i+2})\big)^2$ in (A.14) exactly, instead of estimating it. This gives an improved constant of 1/30 in (A.10). This is purely cosmetic, and only leads to 'uglier' fractions in Theorems (12) and (13).

Theorem 12. *Let $u \in W^{2,\infty}(\Omega)$ and $\widetilde{D} > 0$. Then the solution of (2.43) satisfies*

$$\begin{aligned} &|u-\tilde{u}_h|^2_{H^1(\Omega\setminus\Gamma)} + \widetilde{D}\int_\Gamma [u-\tilde{u}_h]^2 \,\mathrm{d}S \\ &\quad \leq 4C_I^2 h^2 |u|^2_{H^2(\Omega\setminus\Gamma)} + \frac{4}{\widetilde{D}} L |u|^2_{W^{1,\infty}(\Gamma)} + \frac{3}{16}\widetilde{D} L h^4 \|\nabla^2_\parallel u\|^2_{L^\infty(\Gamma)}. \end{aligned} \tag{A.15}$$

Proof. We subtract (A.5) and (2.43) with a test function $v_h \in V_h^\Gamma$:

$$\int_{\Omega\setminus\Gamma} (\nabla u - \nabla\tilde{u}_h)\cdot\nabla v_h \,\mathrm{d}x - \int_\Gamma \nabla u\cdot n_\Gamma [v_h]\,\mathrm{d}S + \widetilde{D}\frac{h}{2}\sum_{x_i\in\Gamma} [u-\tilde{u}_h]_{x_i}[v_h]_{x_i} = 0. \tag{A.16}$$

We define $\xi_h = \Pi_h u - \tilde{u}_h \in V_h$ and set $v_h = \xi_h$ in (A.16). Rearranging gives us

$$\begin{aligned} &\int_{\Omega\setminus\Gamma} |\nabla\xi_h|^2 \,\mathrm{d}x + \widetilde{D}\frac{h}{2}\sum_{x_i\in\Gamma}[\xi_h]^2_{x_i} \\ &= \underbrace{\int_\Gamma \nabla u\cdot n_\Gamma [\xi_h]\,\mathrm{d}S}_{(a)} - \underbrace{\int_{\Omega\setminus\Gamma} (\nabla u - \nabla\Pi_h u)\cdot\nabla\xi_h \,\mathrm{d}x}_{(b)} - \underbrace{\widetilde{D}\frac{h}{2}\sum_{x_i\in\Gamma}[u-\Pi_h u]_{x_i}[\xi_h]_{x_i}}_{(c)}. \end{aligned} \tag{A.17}$$

Now we estimate the individual right-hand side terms.

Term (a): Young's inequality (2.15) with $r = \widetilde{D}/2$ gives us

$$(a) \leq \frac{1}{\widetilde{D}}\int_\Gamma |\nabla u|^2 \,\mathrm{d}S + \frac{\widetilde{D}}{4}\int_\Gamma [\xi_h]^2 \,\mathrm{d}S \leq \frac{1}{\widetilde{D}} L \|\nabla u\|^2_{L^\infty(\Gamma)} + \widetilde{D}\frac{h}{8}\sum_{x_i\in\Gamma}[\xi_h]^2_{x_i}, \tag{A.18}$$

where we have used Lemma 10 with $f = [\xi_h]$.

Term (b): Young's inequality (2.15) with $r = 1$ and Lemma 2 give us

$$\begin{aligned} (b) &\leq \frac{1}{2}\int_{\Omega\setminus\Gamma} |\nabla u - \nabla\Pi_h u|^2 \,\mathrm{d}x + \frac{1}{2}\int_{\Omega\setminus\Gamma} |\nabla\xi_h|^2 \,\mathrm{d}x \\ &\leq \frac{1}{2} C_I^2 h^2 |u|^2_{H^2(\Omega\setminus\Gamma)} + \frac{1}{2}|\xi_h|^2_{H^1(\Omega\setminus\Gamma)}. \end{aligned} \tag{A.19}$$

Term (c): Young's inequality (2.15) with $r = 1/2$ along with Lemma 11 give us

$$\begin{aligned} (c) &\leq \widetilde{D}\frac{h}{2}\sum_{x_i\in\Gamma}[u-\Pi_h u]^2_{x_i} + \widetilde{D}\frac{h}{8}\sum_{x_i\in\Gamma}[\xi_h]^2_{x_i} \\ &\leq \widetilde{D}\frac{1}{64} L h^4 \|\nabla^2_\parallel u\|^2_{L^\infty(\Gamma)} + \widetilde{D}\frac{h}{8}\sum_{x_i\in\Gamma}[\xi_h]^2_{x_i}. \end{aligned} \tag{A.20}$$

Applying the last three estimates to (A.17) gives us

$$\begin{aligned}&\frac{1}{2}|\xi_h|^2_{H^1(\Omega\setminus\Gamma)}+\widetilde{D}\frac{h}{4}\sum_{x_i\in\Gamma}[\xi_h]^2_{x_i}\\&\le\frac{1}{\widetilde{D}}L\|\nabla u\|^2_{L^\infty(\Gamma)}+\frac{1}{2}C_I^2h^2|u|^2_{H^2(\Omega\setminus\Gamma)}+\widetilde{D}\frac{1}{64}Lh^4\|\nabla^2_\parallel u\|^2_{L^\infty(\Gamma)}.\end{aligned}\tag{A.21}$$

Now we estimate the error $u-\tilde{u}_h$ similarly as in (2.24) using the facts that $u-\tilde{u}_h=(u-\Pi_h u)+\xi_h$ and $(a+b)^2\le 2a^2+2b^2$:

$$\begin{aligned}&|u-\tilde{u}_h|^2_{H^1(\Omega\setminus\Gamma)}+\widetilde{D}\int_\Gamma[u-\tilde{u}_h]^2\,\mathrm{d}S=|u-\tilde{u}_h|^2_{H^1(\Omega\setminus\Gamma)}+\widetilde{D}\|[u-\tilde{u}_h]\|^2_{L^2(\Gamma)}\\&\le\big(|u-\Pi_h u|_{H^1(\Omega\setminus\Gamma)}+|\xi_h|_{H^1(\Omega\setminus\Gamma)}\big)^2+\widetilde{D}\big(\|[u-\Pi_h u]\|_{L^2(\Gamma)}+\|[\xi_h]\|_{L^2(\Gamma)}\big)^2\\&\le 2|u-\Pi_h u|^2_{H^1(\Omega\setminus\Gamma)}+2|\xi_h|^2_{H^1(\Omega\setminus\Gamma)}+2\widetilde{D}\|[u-\Pi_h u]\|^2_{L^2(\Gamma)}+\widetilde{D}h\sum_{x_i\in\Gamma}[\xi_h]^2_{x_i},\end{aligned}\tag{A.22}$$

where we have used inequality (A.7) with $f=[\xi_h]$. The ξ_h-terms in (A.22) are estimated using (A.21). The remaining terms are standard:

$$|u-\Pi_h u|^2_{H^1(\Omega\setminus\Gamma)}\le C_I^2h^2|u|^2_{H^2(\Omega\setminus\Gamma)},\tag{A.23}$$

and for the third right-hand side term in (A.22), we use (A.10). Altogether, we get from (A.22)

$$\begin{aligned}&|u-\tilde{u}_h|^2_{H^1(\Omega\setminus\Gamma)}+\widetilde{D}\frac{h}{2}\sum_{x_i\in\Gamma}[u-\tilde{u}_h]^2_{x_i}\\&\le 4C_I^2h^2|u|^2_{H^2(\Omega\setminus\Gamma)}+\frac{4}{\widetilde{D}}L\|\nabla u\|^2_{L^\infty(\Gamma)}+\widetilde{D}\frac{3}{16}Lh^4\|\nabla^2_\parallel u\|^2_{L^\infty(\Gamma)},\end{aligned}\tag{A.24}$$

which is our desired estimate. □

Similarly as in Theorem 6, we can now optimize the value of $\widetilde{D}$ in order to minimize the right-hand side of (A.15).

Theorem 13. *Let $u\in W^{2,\infty}(\Omega)$. Let $\mathcal{T}_h$ contain a zero-measure band Γ of length L. In (2.43) set*

$$\widetilde{D}=\frac{1}{h^2}\frac{8}{\sqrt{3}}\frac{|u|_{W^{1,\infty}(\Gamma)}}{\|\nabla^2_\parallel u\|_{L^\infty(\Gamma)}},\tag{A.25}$$

then the tempered FEM solution $\tilde{u}_h$ satisfies

$$|u-\tilde{u}_h|_{H^1(\Omega\setminus\Gamma)}+\sqrt{\widetilde{D}}\big\|[u-\tilde{u}_h]\big\|_{L^2(\Gamma)}\le C(u,L)h,\tag{A.26}$$

where

$$C(u,L) = 2\sqrt{2}C_I|u|_{H^2(\Omega\setminus\mathcal{B})} + \sqrt{2}\sqrt[4]{3}\sqrt{L}\sqrt{|u|_{W^{1,\infty}(\Gamma)}\|\nabla_\parallel^2 u\|_{L^\infty(\Gamma)}} \tag{A.27}$$

is a constant independent of h.

Proof. We optimize the last two terms in (A.15) with respect to $\widetilde{D}$. To this end, we denote them as

$$g(\widetilde{D}) = \frac{4}{\widetilde{D}}L|u|^2_{W^{1,\infty}(\Gamma)} + \frac{3}{16}\widetilde{D}Lh^4\|\nabla_\parallel^2 u\|^2_{L^\infty(\Gamma)}. \tag{A.28}$$

The application of Lemma 5 gives us the optimal choice of $\widetilde{D}$:

$$\widetilde{D}_{\text{opt}} = \sqrt{\frac{4L|u|^2_{W^{1,\infty}(\Gamma)}}{\frac{3}{16}Lh^4\|\nabla_\parallel^2 u\|^2_{L^\infty(\Gamma)}}} = \frac{8}{\sqrt{3}}\frac{1}{h^2}\frac{|u|_{W^{1,\infty}(\Gamma)}}{\|\nabla_\parallel^2 u\|_{L^\infty(\Gamma)}}, \tag{A.29}$$

which is (A.25). Moreover,

$$\begin{aligned} g(\widetilde{D}_{\text{opt}}) &= 2\sqrt{4L|u|^2_{W^{1,\infty}(\Gamma)}\tfrac{3}{16}Lh^4\|\nabla_\parallel^2 u\|^2_{L^\infty(\Gamma)}} \\ &= \sqrt{3}Lh^2|u|_{W^{1,\infty}(\Gamma)}\|\nabla_\parallel^2 u\|_{L^\infty(\Gamma)}. \end{aligned} \tag{A.30}$$

With the choice $\widetilde{D} = \widetilde{D}_{\text{opt}}$, estimate (A.15) therefore becomes

$$\begin{aligned} &|u-\tilde{u}_h|^2_{H^1(\Omega\setminus\Gamma)} + \widetilde{D}\|[u-\tilde{u}_h]\|^2_{L^2(\Gamma)} \\ &\quad\le 4C_I^2h^2|u|^2_{H^2(\Omega\setminus\Gamma)} + \sqrt{3}Lh^2|u|_{W^{1,\infty}(\Gamma)}\|\nabla_\parallel^2 u\|_{L^\infty(\Gamma)}. \end{aligned} \tag{A.31}$$

We take the square root of (A.31) by using (2.33):

$$\begin{aligned} &|u-\tilde{u}_h|_{H^1(\Omega\setminus\Gamma)} + \sqrt{\widetilde{D}}\big\|[u-\tilde{u}_h]\big\|_{L^2(\Gamma)} \\ &\quad\le 2\sqrt{2}C_Ih|u|_{H^2(\Omega\setminus\Gamma)} + \sqrt{2}\sqrt[4]{3}\sqrt{L}h\sqrt{|u|_{W^{1,\infty}(\Gamma)}\|\nabla_\parallel^2 u\|_{L^\infty(\Gamma)}}. \end{aligned} \tag{A.32}$$

This is the desired estimate (A.26) with the constant $C(u,L)$ defined by (A.27). □

We could also state Theorem 13 in the 'quadrature form':

Theorem 14. *Let $u \in W^{2,\infty}(\Omega)$. Let $\mathcal{T}_h$ contain a zero-measure band Γ of length L. In (2.43) set*

$$\widetilde{D} = \frac{1}{h^2}\frac{|u|_{W^{1,\infty}(\Gamma)}}{\|\nabla_\parallel^2 u\|_{L^\infty(\Gamma)}}8\sqrt{\frac{2}{3}} \tag{A.33}$$

then the tempered FEM solution $\tilde{u}_h$ satisfies

$$|u-\tilde{u}_h|_{H^1(\Omega\setminus\Gamma)}+\left(\widetilde{D}\frac{h}{2}\sum_{x_i\in\Gamma}[u-\tilde{u}_h]^2_{x_i}\right)^{1/2}\leq C(u,L)h, \tag{A.34}$$

where

$$C(u,L)=2\sqrt{2}C_I|u|_{H^2(\Omega\setminus\mathcal{B})}+\sqrt[4]{6}\sqrt{L}\sqrt{|u|_{W^{1,\infty}(\Gamma)}\|\nabla^2_{\|}u\|_{L^\infty(\Gamma)}} \tag{A.35}$$

is a constant independent of h.

Proof. The proof follows exactly the proofs of Theorems 12 and 13, the only difference being the application of (A.9) instead of (A.10) in the corresponding version of (A.22). □