École Doctorale 146

THÈSE
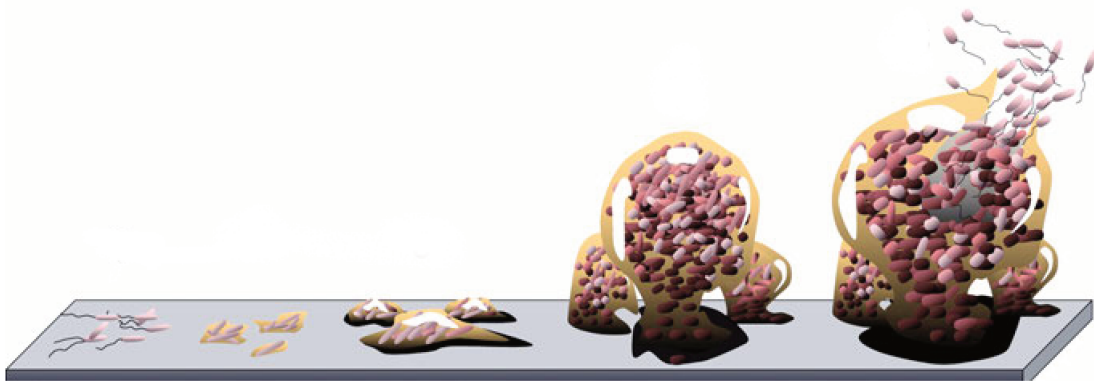
pour obtenir le grade de docteur délivré par

Université Paris 13

Spécialité doctorale "*Mathématiques appliquées*"

Méthodes d'éléments finis pour des systèmes d'EDP non linéaires avec interface. Application à un modèle de croissance de biofilm

présentée publiquement par **DINH Anh-Thi**

JURY

Jean-Stéphane Dhersin	Université Paris 13	Directeur de thèse
Linda El Alaoui	Université Paris 13	Co-encadrant de thèse
Adel Blouza	Université de Rouen	Co-encadrant de thèse
Pascal Frey	Université Pierre-et-Marie-Curie	Rapporteur
Yves Renard	INSA de Lyon	Rapporteur
Nicolas Vauchelet	Université Paris 13	Examinateur
Sébastien Martin	Université Paris Descartes	Examinateur
Luís Neves de Almeida	CNRS & U. Pierre-et-Marie-Curie	Invité

20-12-2018
Villetaneuse, France

Finite Element Methods for nonlinear interface problems. Application to a biofilm growth model

presented by **DINH Anh-Thi**
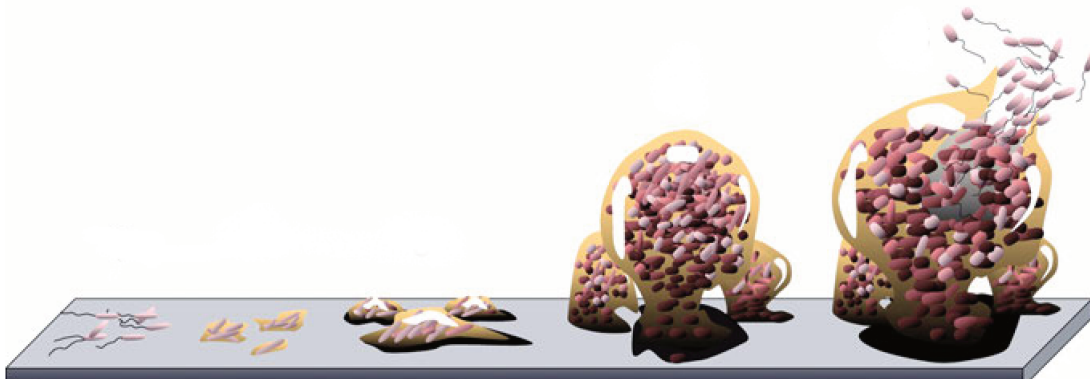
JURY

Jean-Stéphane Dhersin	Paris 13 University	Supervisor
Linda El Alaoui	Paris 13 University	Co-Supervisor
Adel Blouza	Rouen University	Co-Supervisor
Pascal Frey	Pierre and Marie Curie University	Reviewer
Yves Renard	Lyon INSA	Reviewer
Nicolas Vauchelet	Paris 13 University	Examiner
Sébastien Martin	Paris Descartes University	Examiner
Luís Neves de Almeida	CNRS & Pierre and Marie Curie U.	Invited member

20-12-2018
Villetaneuse, France

Résumé

Un biofilm est un ensemble de micro-organismes tels que les bactéries, les champignons ou encore les algues qui vivent en communauté. Les biofilms ont la capacité d'être présents en tout lieu. Ils sont observés dans les milieux aqueux ou humides. Ils peuvent se développer sur n'importe quel type de surface naturelle ou artificielle, qu'elle soit minérale (roche, interfaces air-liquide...) ou organique (peau, tube digestif, racines et feuilles des plantes), industrielle (canalisations, coques des navires) ou médicale comme les prothèses et les cathéters. Cette ubiquité est à l'origine de nombreuses infections bactériennes. Les infections nosocomiales contractées dans les hôpitaux sont un exemple majeur. Certaines de ces infections pouvant être mortelles. Le traitement médical des biofilms est souvent inefficace pour lutter contre ce type d'infection. Il est donc important de comprendre les mécanismes de croissance d'un biofilm. Telle est la motivation de la présente thèse.

Afin de réaliser des simulations numériques d'un modèle décrivant la croissance d'un biofilm, nous combinons différentes méthodes de calcul basées sur la méthode Nitsche-Extended Finite Element Method (NXFEM) ainsi que sur la méthode des lignes de niveau. Ces méthodes nous permettent d'étudier des modèles complexes dans lesquels l'interface entre le biofilm et son environnement est capable de se déformer tout en dépendant du temps. Ceci permet de considérer une discrétisation à l'aide d'un maillage ne coïncidant pas avec l'interface biofilm/environnement. Nous présentons également une technique de découplage d'un système d'équations aux dérivées partielles semi-linéaires et la façon dont nous appliquons la méthode NXFEM pour résoudre un tel problème. Ce système est en relation avec le modèle de croissance du biofilm qui est traité dans cette thèse

Pour l'implémentation, une boîte à outils NXFEM, développée en Matlab, a été entièrement conçue pour résoudre un tel problème. Nous donnons dans ce document

les détails des algorithmes et techniques numériques utilisés afin que chacun puisse utiliser cette boîte à outils pour ses propres projets.

Abstract

A biofilm is a collective of living, reproducing microorganisms, such as bacteria, that stick together as a colony or community. They appear everywhere in human life and have impacts on our environment. Biofilm modeling, together with laboratory experiments, has risen to produce quantitative tools for scientists to better understand the biofilm's growth. This thesis is motivated to research on this subject.

A combination of computational methods which are based on *Nitsche-Extended Finite Element Method* (NXFEM), *Level Set Method* and some other stabilized techniques is used to solve and simulate a biofilm growth model. These methods allow us to work with a complex scheme in which the interface between the biofilm and its environment may change with time and on an unfitted mesh. We also present a technique of decoupling a system of semilinear differential equations and how we apply the NXFEM method to solve such a problem. This system has a relation to a model of biofilm's growth which will be examined carefully in the work.

For the implementations, *NXFEM toolbox* which is a Matlab based toolbox is built for solving such a problem. We also give the details of all algorithms and numerical techniques so that everyone can use this toolbox for their own projects.

Keywords: *NXFEM, Nitsche-Extended Finite Element Method, interface problem, level set method, biofilm, unfitted mesh.*

Acknowledgements

say about all things I would like to thank him.

I would like to thank my friends in University Paris 13: Tarek Ghoudi, Oussama Landoulsi, Mohammed Zitouni, and Antoine Kaszczyc. With their help, I can improve my French day by day more comfortable and confident. They are very friendly, kind and fun.

Next and finally, I express my friendly thanks to my Vietnamese friends and my family. I write in Vietnamese for them.

Em cảm ơn anh Việt đã giúp em rất nhiều khi em bị bệnh mắt và cần người phiên dịch. Cảm ơn anh Hoàng và Ky đã cho em/anh ở tạm một khoảng thời gian dài để hoàn thành luận án. Cảm ơn Quang, Trâm, Thiện, Thủy, Nhật, chị Hòa, bé Thảo, Long đã giúp anh rất nhiều trong việc chuẩn bị tiệc sau bảo vệ.

Em cảm ơn anh chị Dũng Trang đã giúp và xem em như một người em trong gia đình. Em cảm ơn chị Kiều đã và đang có những lời khuyên quý giá giúp em trong việc định hình hướng đi tiếp theo trong sự nghiệp. Em cảm ơn thầy Ngô Văn Thiện đã luôn đồng hành và động viên em như một người bạn chân tình. Em cảm ơn chị Đoan, chị Tú và gia đình đã cho em những giây phút ấm áp cùng gia đình các anh chị và giúp đỡ em nhiều khi sống ở bên Pháp.

Em/Anh xin cảm ơn đến gia đình anh chị Hoàn Huyền, anh Điệp, Hoàng Gia, Trọng, chị Nga, chị Thu, anh Chiến Bùi, anh Chiến Thái, anh Ân, anh chị Bảo Cúc, chị Dung và rất nhiều anh, chị, em, bạn bè khác ở Pháp đã cho em/anh những phút giây thật vui vẻ và sảng khoái như được có thêm một gia đình nữa bên Pháp.

Cảm ơn người bạn đặc biệt giấu tên của anh, người đã cùng anh chia sẻ một trong những khoảnh khắc đẹp nhất cuộc đời anh trong giai đoạn anh làm luận án.

Em muốn dành lời cảm ơn đặc biệt đến chị Phượng, người chị đã hết lòng động viên và giúp đỡ em không ngừng khi em gặp những khoảng thời gian khó khăn nhất khi ở Pháp. Em cũng muốn gởi lời cảm ơn đến anh Lộc khi đã tạo điều kiện cho chúng em đến chơi nhà anh chị và vui đùa thỏa thích để quên đi cảm giác xa quê.

Cuối cùng, con cảm ơn cha mẹ, anh cảm ơn Huy, thằng em trai nhỏ, những người đã luôn ủng hộ, tin tưởng và động viên con trong suốt khoảng thời gian qua. Con muốn dành kết quả này như một món quà đặc biệt biểu tặng cha mẹ.

From a friend, a brother, a son and a Vietamese guy.

Contents

I   nxfem method

II   nxfem with biofilms

appendix

B Biofilm growth   121

references

Glossaries, notations and symbols

In this section, I list all of the acronyms, operators and notations I will use throughout this thesis. There may be ones that are only defined here and used later without recall the definition at the place they appear.

Acronyms

Initials	Meaning
1D	Dimension 1 or 1 dimensional
2D	Dimension 2 or 2 dimensional
DDM	Domain Decomposition Method
EPS	Extracellular Polymeric Substances
FEM	Finite Element Method
IIM	Immersed Interface Method
ips	intersection points
LHS	Left Hand Side
LSM	Level Set Method
NFEM	Nitsche Finite Element Method

Initials	Meaning
NXFEM	Nitsche-Extended Finite Element Method
RHS	Right Hand Side
SUPG	Streamline Upwinding Petrov-Galerkin method
w.r.t	with respect to
XFEM	Extended Finite Element Method

Notations

Notations	Meaning	Page
G_h	set of all cells that are interseted by interface Γ.	17
$H^k(\Omega), H_0^k(\Omega)$	are $W^{k,2}(\Omega), W_0^{k,2}(\Omega)$ respectively.	xv
$H^k(\Omega_{12})$	$H^k(\Omega_1 \cup \Omega_2) := \{v \in L^2(\Omega) : v_i \in H^k(\Omega_i) \text{ for } i = 1,2\}$ for $k = 0,1,2$ where $v_i = v\|_{\Omega_i}$.	21
H_Γ	a heaviside function w.r.t each subdomain Ω_i.	30
I_{SD}, I_{SD}	Interpolation operator from V_h^Γ to V_h and vice versa.	118
$K, \partial K$	element K of triangulation $\mathscr{T}_h$ and its sides ∂K.	17
$L^p(\Omega)$	functions whose p-th power is Lebesgue integrable on Ω.	xv
$L^p(]0, T[; V)$	V-valued functions whose norm in V is in $L^p(]0, T[)$.	xv
$N_r(\hat{x}, \hat{y})$	Local shape functions defined on reference triangle $\hat{K}$.	111
$N_{V_h}, N_{\mathscr{I}}, N_{\text{new}}$	number of standard basis functions in V_h, number of "new" basis functions and number of degree of freedom in V_h^Γ respectively.	45
$Oxy, O\hat{x}\hat{y}$	Coordinate systems Oxy and its reference $O\hat{x}\hat{y}$.	109
V, V_0	$V = \{v \in H^1(\Omega) : v = 0 \text{ on } \Gamma\}, V_0 = \{v \in V : v = 0 \text{ on } \partial\Omega\}$.	xv
$V_A^h, V_{h,i}^\Gamma$	$= V_h \oplus V_{h,1}^\Gamma \oplus V_{h,2}^\Gamma$. NXFEM space defined by Reusken et al. [50], cf. (2.33, 2.41).	28, 31

Notations	Meaning	Page
V_h	standard finite element method, cf. (2.4).	16
V_σ, V_σ^0	Symbolic spaces defined in Definition 4.3.	65
$W^{k,p}(\Omega), W_0^{k,p}(\Omega)$	$W^{k,p}(\Omega)$ contains functions whose derivatives up to order k are in $L^p(\Omega)$, $W_0^{k,p}(\Omega)$ contain functions in $W^{k,p}(\Omega)$ and they are zero on $\partial\Omega$.	xv
X_a, X_b	two endpoints of $\Gamma_{K,h}$.	47
$\Gamma, \Gamma_K, \Gamma_{K,h}, \Gamma_h$	the interface Γ and its part Γ_K on the triangle K and $\Gamma_{K,h}$ the segment connecting the intersection points between Γ and boundary ∂K of element K. $\Gamma_h = \cup\Gamma_{K,h}$	16–18
Ω_{12}	$\Omega_1 \cup \Omega_2$.	xv
$\Omega_{\mathscr{T}}$	fictitious domain of physical domain Ω.	26
V_h^Γ, V_h^0	finite element spaces considered in NXFEM method, cf. (2.14), (2.15).	21, 30
$\hat{K}$	Reference triangle $\hat{K}$ in $O\hat{x}\hat{y}$ which is corresponding to a triangle K in Oxy under trasform mapping $\mathbf{P}$ (Figure A.1). Note that, the notation $\hat{\cdot}$ is used to indicate the *reference* of some components in $O\hat{x}\hat{y}$.	108
$\mathbb{P}_k$	vector space of polynomials in the variables $x_1, \ldots, x_d$ of global degree at most k in $\mathbb{R}^d$.	16
$\mathscr{D}(\Omega)$	infinitely differentiable functions compactly supported in Ω.	xv
$\mathscr{E}_G^i$	set of edges used to compute the ghost penalty terms in each subdomain Ω_i, cf. (2.32).	28
$\mathscr{I}, \mathscr{I}_\Gamma, \mathscr{I}_i^\Gamma$	set of indexes numbering the nodes associated to V_h and its subset which neighbour the interface, cf. (2.34, 2.35).	28, 29, 119
$\mathscr{I}_1, \mathscr{I}_2, \mathscr{I}_0$	set of indexes numbering the nodes associated to V_h in each subdomain Ω_i and the nodes on interface, cf. Section A.7.1.	119
$\mathscr{T}_h, \mathscr{T}_h^i$	conforming triangulation of domain Ω and it subset which covers subdomain Ω_i.	16, 17

Operators

Operators	Meaning	Page
$\{\!\{u\}\!\}$	average condition on an interface or on an edge of the mesh's element, cf. Definition 2.1.	18
$\mathbf{P}$	a mapping which transforms each vertex of $\hat{K}$ in $O\hat{x}\hat{y}$ to corresponding vertex of K in Oxy, $\mathbf{P}$ is given in (A.5).	109
$\mathbf{Q}$	a mapping which transforms each vertex of $\tilde{K}$ in $O\tilde{x}\tilde{y}$ to corresponding vertex of $\hat{K}$ in $O\hat{x}\hat{y}$, $\mathbf{Q}$ is given in (3.14).	51
$[\![u]\!]$	jumb condition on an interface or on an edge of the mesh's element, cf. Definition 2.1.	18
$|K|$	measure of element K.	xv
$\|\|\!|\cdot\|\!|\|_1$	norm defined in (4.21).	62
$\|\|\!|\cdot\|\!|\|_2$	norm defined in (4.24).	63
$\|\|\!|\cdot\|\!|\|_H$	norm proposed by Hansbo [31], cf. (2.21).	23
$\|\|\!|\cdot\|\!|\|_N$	norm in original Nitsche method, cf. (2.5).	16
$\|\|\!|\cdot\|\!|\|$	Symbolic norm defined in (4.27).	65

1. Introduction

Contents

In this chapter, we take you from the start of biofilm problem to the methods and the motivation of this thesis. We also present a general idea about biofilm and methods people have used for recent years to solve interface problems which are the focus in a biofilm model.

1.1 Biofilm

1.1.1 What is biofilm?

Definition. The term "biofilm" may be strange but it is something we may contact to every day. The plaque that forms on teeth and causes tooth decay and periodontal

disease is a type of biofilm (Figure 1.1.a.). Biofilm is defined as a community of micro-organisms (bacteria, fungi, algae and protozoa) attached to a surface. In [71], biofilm is fully called as "*a layer of prokaryotic and eukaryotic cells anchored to a substratum surface and embedded in an organic matrix of biological origin*". Biofilm may take the most responsibility for most infections [49] and bacteria inside biofilm are more resistance to the antibiotic than planktonic cells of the same type [1].

Living environment. Biofilm is, generally, observed in aqueous media or in a medium exposed to moisture. They can grow on any natural or artificial surface. This surface may be mineral (rock interfaces, air-liquid,...) or organic (skin, plants,...), industrial (pipes, oil, waste-water,...) or medical (prosthesis, catheters,...),...

a. On human body[1].

b. In nature[2].

c. In food[3].
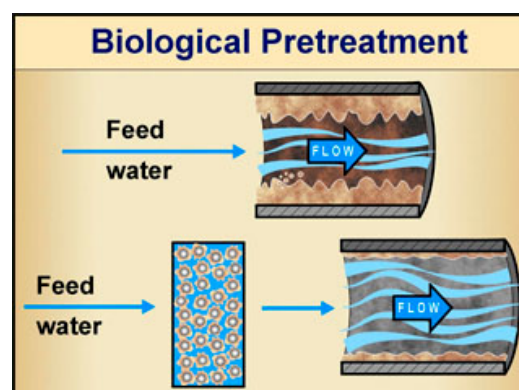
d. Walter filter[4].

Figure 1.1. Biofilm appears everywhere in human life.

Biofilm's life circle. The life circle of a biofilm form can be described via three main stages: *attachment*, *growth* and *dispersal* [62, 44] (Figure 1.2). In the first stage, free-floating, or isolated planktonic-state microbes encounter a submerged surface and within minutes can become attached to this surface. They start to produce viscous *extracellular polymeric substances* (EPS) and to colonize the surface. In the second stage, EPS production allows the emerging biofilm community to develop a

---

[1]Credit: Modification of work by Klaus D. Peter (ear) and bacteriality.com (teeth).

[2]Credit: skfaquatics.com.

[3]Credit: *The strange case of a biofilm-forming strain of Pichia fermentans, which controls Monilinia brown rot on apple but is pathogenic on peach fruit* - Sara Giobbe *et al*.

[4]Credit: P. Dirckx, Center for Biofilm Engineering, Montana State University, Bozeman.

complex, three-dimensional structure that is influenced by a variety of environmental factors. Biofilm communities can develop within hours. Biofilm can growth under everywhere and conditions, things they need to growth may be mentioned as microorganisms, moisture, nutrients and surfaces [62]. Finally, the fully mature biofilm reaches a quasi-steady state where growth is balanced by loss through erosion and detachment due to mechanical stress. They can propagate through detachment of small or large clumps of cells, or by "seeding dispersal" that releases individual cells. Either type of detachment allows bacteria to attach to a surface or to a biofilm downstream of the original community.

Figure 1.2. Stages of the biofilm life cycle[5].

Good or bad? They are in both ways. We have talked about tooth decay above and there is also a link between biofilms with ulcerative colitis and several hospital acquired infections. To dentists and doctors, biofilms are more than an eyesore. They are expensive, destructive and sometimes deadly [64, 71],...Nevertheless, there are some good news in that not all biofilms are bad. Some industries use them in water-filter, production of medicines, food additives and even as cleaning agents [62, 71].

In this thesis, we focus on the "growth" stage of biofilm and study the change of its form under some environment conditions.

1.1.2 Modeling of biofilm

A *mathematical model* is a description of a system of natural phenomena using mathematical concepts and language. The process of developing a mathematical model is termed *mathematical modelling*. In this work, we develop a mathematical model describing the growth of a biofilm model under some affects. A good understanding

---

[5]Credit: Courtesy of the Montana State University Center for Biofilm Engineering, P. Dirckx.

of this phenomenon is in accordance with a good model and inversely. Using and creating a mathematical model require six steps [71]:

1. *The important variables and processes acting in the system must be identified.* In our problem, necessary components are substrates, bacteria, fluid flows, biomass flows and some antimicrobial.

2. *Performing processes as mathematical expressions.* In our case, this is the system of two partial differential equations describing the evolution of substrates and bacteria. They can be represented by Monod's law (Section 1.1.3) and diffused by Fick's law.

3. *The mathematical expressions are combined appropriately in equations.* We will discuss the meaning of a biofilm model in Chapter 6.

4. *The parameters involved in the mathematical expressions are given values.* In our model, parameters come from real experiments and from some others' works.

5. *Approximate the solution of the system by numerical methods.* If a problem has a solution, then we approximate it by a numerical scheme. We based on the biofilm model to work with an interface problem. From that, we choose the NXFEM method to treat a such problem. See more detail in Part I.

6. *The properties of the system are explored via the solution of the model.* After doing maths on the biofilm models, we give comments on the results and play with them. See more in Part II.

The presence of biofilms in a wide range of situations has led researchers to study their properties for their both good and bad aspect. It turns out that mathematical modeling can be an efficient and essential tool for the study of biofilms because it's difficult to study biofilm in a laboratory with their thickness is on the order of 10 to 100 microns [42] (a micron is about 10^{-6} meters).

Modeling is a powerful tool for studying biofilm processes as well as for understanding how to encourage good biofilms or discourage bad biofilms. A mathematical model may be a good way to connect the different processes to each other and to measure their relative contributions [71].

A biofilm model should be *realistic* in the sense that it does not necessarily contain all phenomena within a biofilm, but it should be able to tell accurately the properties of biofilm on which we are working.

Because we only focus on the growth of a biofilm, we see its form as *a layer* (without considering its thickness) covering around the bacteria (Figure 1.3). We will work on this layer which is *an interface* in a mathematical model. We will discuss this in more detail in Section 1.2.

1.1.3   Monod's law

In biofilm modeling, people have mainly worked on the relationship between nutrients (substrates) and biomass (bacteria). This relation follows the Monod's kinetic [41] which is named after a French biologist Jacques Monod (1910-1976) who first introduced it. This relation is usually presented by a nonlinear function given in (1.1)
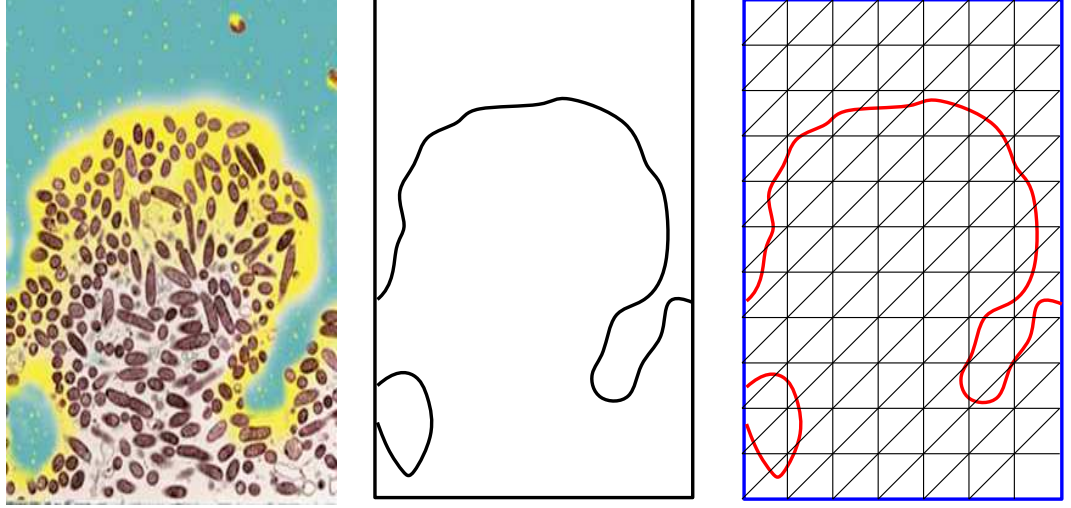
Figure 1.3. An idea of translating a real biofilm form to a theoretical model.

$$g_M(S) = \mu_{\max}\frac{S}{K_S+S},\qquad(1.1)$$

in which we use S for the substrate, $g_M(S)$ for the specific growth rate of a microorganism, $\mu_{\max}$ for the maximum specific growth rate and K_S for the half-velocity constant.

The relation (1.1) tells the truth that bacteria cannot grow up perpetually if there is an infinitely concentration of nutrients. There is always a *maximum specific growth rate* $\mu_{\max}$ which is the maximum number of microorganisms could be (Figure 1.4).

Monod had empirically found this kinetic, and it's very interesting in the sense that

Figure 1.4. Monod's relation.

- If there is no food, there are no bacteria either.
- The relation is smooth from one state to another one.
- If there is an enormous amount of food, the rate approaches the maximum growth.

Sometimes, g_M is also presented in simpler forms as in (1.2) if $S \ll K_S$ (where $k_1 = \frac{\mu_{\max}}{K_S}$) or in (1.3) if $S \gg K_S$ (where $k_0 = \mu_{\max}$). If there are over one substrate S in the environment necessary for the bacteria's growth, multiple terms of the form $\frac{S}{K_S+S}$ are multiplied together. However, we don't consider this complicated cases and we leave them for future works.

$$g_M(S) = k_1 S,\qquad(1.2)$$
$$g_M(S) = k_0.\qquad(1.3)$$

Because of relation (1.1), in our main biofilm model, we will work on a semilinear equation. It makes more difficult to solve the problem, we will discuss more on Section 1.2.

1.1.4 Some former biofilm models & works

Categorization. Mathematical models for biofilm processes have been proposed since the early of the 80s in which scientists worked on simple ordinary or partial differential equations. For recent decades, there are many researches on modeling biofilm with new technologies and interests. Szomolay, in his PhD thesis [74], had classified the biofilm researches into 4 categories,

1. *Resistance*: an ability of a microorganism in which they can grow despite the presence of antimicrobial in the environment.
2. *Phenotypic tolerance*: one of the hypotheses which explain the failure of antimicrobial treatments is an ability of bacteria to evade the bactericidal activity of antimicrobial.
3. *Dosing*: study about biofilm dosing strategies for controlling the growth of biofilm. By adjusting the time of dosing or the concentration at each time, ones can reduce or sometimes increase the influence of the biofilms under the presence of antimicrobial.
4. *Detachment*: these are researches on the third stage of the biofilm (Figure 1.2) under some impacts like shear stress, nutrient supply or biofilm thickness.

In this thesis, we are interested in some typical biofilm models which have relation to our purpose of biofilm growth. For this section, we list some general form of models. Concerning the methods that are used corresponding to them, we will discuss more on Section 1.3.

Picioreanu. Recall the work of Picioreanu *et al* [68, 46, 47], they worked on several structures of biofilm and they wanted to see the biofilm growth under the effects of *diffusive and convection substrate transports*. The system (1.4) gives a basic idea about the model they use.

$$\begin{cases} \dfrac{\mathrm{d}B}{\mathrm{d}t} + \alpha B - \beta \dfrac{\gamma BS}{K_S + S} = 0, \\ -D_S \Delta S + \mathbf{u} \cdot \nabla S - \dfrac{\gamma BS}{K_S + S} = 0, \end{cases} \tag{1.4}$$

where B, S are respectively bacteria and substrate in the context and $\mathbf{u}$ is the flow of fluid in the environment. In this model, the biofilm growth is regulated by the concentration of substrate under the influence of fluid flow. On the other hand, the flow shears the biofilm surface and erodes the biofilm structural protuberances. They considered two cases of limitation, one for *transport limited* and one for *microbial growth limited*, then they get some nice results [47] on the dynamic structure of biofilm evolving in *non-steady-state conditions*.

Chopp. Picioreanu mentioned about shear stresses [68] (which has firstly introduced

by Rittman [52]) and their effects on the biofilm structure but they didn't consider biofilm detachments in detail. It has been done better in the later work of Duddu, Chopp *et al.* [17, 26]. Their model is introduced formally in (1.5),

$$\begin{cases} -D_S\Delta S - \alpha \dfrac{BS}{K_S+S} = 0, \text{ in } \Omega_b, \\ -D_S\Delta S + \mathbf{v}\cdot\nabla S = 0, \text{ in } \Omega_f, \\ \dfrac{\mathrm{d}B}{\mathrm{d}t} + \nabla\cdot(\mathbf{u}B) - \beta \dfrac{BS}{K_S+S} = 0, \text{ in } \Omega_b, \end{cases} \tag{1.5}$$

where equations are separated in subdomains Ω_f (fluid region) and Ω_b (biofilm region). It looks like in the model of Picioreanu (1.4) with S, B are respectively the substrate and bacteria, there are differences in the presence of fluid outside the biomass region $\mathbf{v}$ and the biomass flow $\mathbf{u}$. Actually, Chopp and his coauthors didn't work directly on the equation of bacteria B. They worked on an intermediary component called *potential* Φ which represents the advection velocity of biomass $\mathbf{u}$. This $\mathbf{u}$ has a close relation with the concentration of bacteria B. We will rely much on this idea to work on our biofilm model.

The group of Chopp worked most on the techniques which help simulate the biofilm growth and verify again some already done results of other scientists like the group of Picioreanu. They didn't give in details the mathematical analysis of the model and methods they proposed. This is one of our differences in comparison with their work. I will recall it in Section 1.3.

Beside the techniques, the group of Chopp also worked much on *quorum sensing* [15, 16] which is a mechanism in which bacteria have the ability to monitor their own population density and modulate gene expression. This is not what we concern in this thesis.

Cogan. We are also interested in the work of Cogan [19, 20, 21, 22]. He worked on the *physiological resistance mechanism* of biofilm and dosing strategies. In this mechanism, there is the existence of novel cells namely *persisters* which are extremely tolerant of antibiotics. Persister cells, together with *susceptible cells*, give a very interesting relation with the presence of the antibiotic in the environment. If an antibiotic is absent, the susceptible bacteria consume substrates and reproduce. Otherwise, a part of susceptible cells are killed and the other one convert to persister cells. Persister cells cannot be killed by the antibiotic nor grow. They just revert to the susceptible cells if there is no antibiotic again. He suggested a biofilm model given in (1.6).

$$\begin{cases} \dfrac{dB_s}{dt} = \underbrace{g(B_s,S)}_{\text{Growth}} - \underbrace{d(B_s,S,A)}_{\text{Disinfection}} - \underbrace{l(B_s,S,A)}_{\text{Loss}} + \underbrace{r(B_p,S,A)}_{\text{Reversion}}, \\ \dfrac{dB_p}{dt} = \underbrace{l(B_s,S,A)}_{\text{Gain}} - \underbrace{r(B_p,S,A)}_{\text{Reversion}}, \end{cases} \tag{1.6}$$

where B_s, B_p, S, A are susceptible cells, persister cells, substrates and antibiotics

respectively. Here, the loss of B_s is also the gain of B_p and the reversion is the same for both cases. The signs in the model show the increasing or decreasing of concentration of biomass. All functions g, d, l, r follow the Modod kinetic (1.1).

Cogan *et al* [20] also worked with a mathematics system of substrate S, bacteria B, antimicrobial A and a neutralizing agent N. Antimicrobial fails to penetrate biofilm because there is neutralizing reaction between it and some components in biofilm region. The relation is given in system (1.7).

$$\begin{cases} \partial_t S + U \cdot \nabla S = \nabla \cdot (D_s \nabla S) - \mu_S \dfrac{S}{K_s + S} B, \\ \partial_t A + U \cdot \nabla A = \nabla \cdot (D_a \nabla A) - k_r AN, \\ \partial_t N = -k_r Y_n AN, \\ \partial_t B = -p_1(A, S)B. \end{cases} \tag{1.7}$$

Others. As an additional reference, one can check the work of Patricio Cumsille *et al.* [23]. They work on a biofilm growth model by using the hybrid Immersed Interface Method coupling with the Level Set Method. Another one is the work of XianLong Zhang *et al.* [57]. They work on a biofilm model in which biofilm grows on an agar substrate and they use the Extended Finite Element Method to research the problem. In this thesis, we choose different methods which are given in the next section in comparison with theirs.

1.2   Motivation & objective of thesis

Motivation & objective. In this thesis we are interested in the growth of a bacterial biofilm. To this end, we consider a model in which the two main components are bacteria and substrates. The bacteria present only inside the biofilm region whereas substrate is both inside and outside biofilm region. There are flows of fluid and biomass on both sides of biofilm like the model of Chopp (1.5). We want to see the dependence of bacteria on the concentration of substrates, the rate of flows and also the presence of antibiotics. The rate of diffusion of substrate and biomass is also considered in this case.

There are not so much differences between our model with the ones of Chopp, Picioreanu or Cogan described above. The main difference of our work in comparison with the others' is *the method* we will use to simulate the growth of biofilm and also the dependence of the model on parameters.

As mentioned in Section 1.1.2, we focus on the *form* of the biofilm, this form's shape is based on the interface separating biofilm with the environment. This leads to work on *an interface problem*. Moreover, because of the Monod relation (1.1), this problem is not linear anymore, we have to consider *a semilinear problem* in the company of *interface problem*.

The big new contribution of our works in comparison with former ones is to consider carefully the mathematical analysis of the problem. We also propose *a self-building*
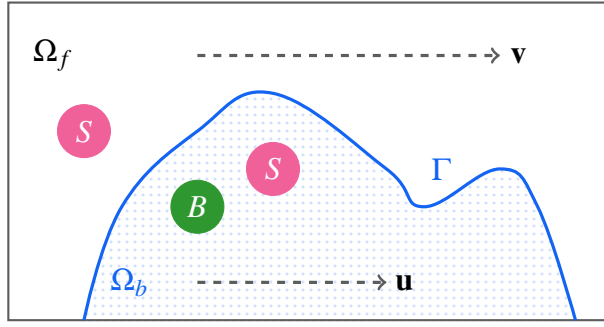
Figure 1.5. An illustration of computational domain Ω. Ω_f stands for fluid region, Ω_b stands for biofilm region.

matlab toolbox to simulate the model (cf. Chapter 3).

We are considering *an evolution problem* in which all components are conditional upon a time. The interface will also change its coordinate every time step. If we use a numerical method to solve the problem, it's necessary to solve it on a mesh. If the mesh depends on the position of the interface, it will be re-generated many times. It costs too much if we are solving on a very smooth mesh or on a long time scale problem. That's why we choose a method with which we can only use one mesh for the whole process. In other words, the mesh does not depend on the interface's change. We will discuss more about this method in Section 1.3 and Part I.

General model. With all above purposes, let us consider a convex polygonal, Lipschitz and bounded domain Ω in $\mathbb{R}^2$ containing two subdomains Ω_f, Ω_b which satisfies $\overline{\Omega} = \overline{\Omega_f} \cup \overline{\Omega_b}$. They present for fluid region Ω_f and biofilm region Ω_b. This is an environment where the biomass B lives. In this environment, there is some substrate S (eg. oxygen) where the biofilm can find the nutrient for their growth. There is also the fluid $\mathbf{v}$ flowing outside the biofilm region and the flow of biomass $\mathbf{u}$ locating inside the biofilm region. The governing equation for the fluid flow at steady state is given by the incompressible Navier-Stokes equation (1.8)

$$\begin{cases} \mathbf{v} \cdot \nabla \mathbf{v} + \nabla p + \frac{1}{Re}\Delta \mathbf{v} = 0 & \text{in } \Omega_f, \\ \nabla \cdot \mathbf{v} = 0 & \text{in } \Omega_f, \end{cases} \tag{1.8}$$

where Re is a Reynolds number, p is the hydrostatic pressure.

The biomass advection velocity, $\mathbf{u}$, is associated with the volumetric expansion or the contraction of the biofilm. Accordingly, we assume that the biofilm growth is irrotational, that is, $\nabla \times \mathbf{u} = 0$. Thus, $\mathbf{u}$ can be derived from a potential function Φ defined in Ω_b such that $\mathbf{u} = \nabla \Phi$. The governing system of equations is given in (1.9)

$$\begin{cases} \partial_t S - \nabla \cdot (D_S^b \nabla S) + \mathbf{u} \cdot \nabla S + \alpha B g(S) = 0 & \text{in } (0,T) \times \Omega_b, \\ \partial_t S - \nabla \cdot (D_S^f \nabla S) + \mathbf{v} \cdot \nabla S = 0 & \text{in } (0,T) \times \Omega_f, \\ \partial_t B - \nabla \cdot (D_B \nabla B) + \mathbf{u} \cdot \nabla B - \beta B g(S) = 0 & \text{in } (0,T) \times \Omega_b, \\ -\Delta \Phi + \beta g(S) = 0 & \text{in } \Omega_b. \end{cases} \tag{1.9}$$

Here $g(S)$ follows the Monod's law (1.1) under the environment of biomass flow $\mathbf{u}$. We consider here the continuity of substrates and their flow through the interface by

taking an interface condition for S given in (1.10). It also goes with some appropriate boundary conditions for S, B and Φ.

$$\llbracket S \rrbracket = \llbracket D_S \nabla S \cdot \mathbf{n} \rrbracket = 0. \tag{1.10}$$

> **Remark 1.1** For the physical meaning of signs and terms in the model of biofilm, we refer to Section 6.2.

1.3 Methods to use

Methods to simulate biofilm model. To work with a biofilm model, we have to deal with the fact that the interface is not steady. In [27], before describing the method being used, Duddu *et al* had listed briefly and clearly the state of the art of numerical methods to simulate a biofilm model. According to them, there are three main branches of biofilm simulating models: *cellular automata*, *individual-based models* and *continuum models*. The first two types based on a combination of deterministic and stochastic rules on lattice or non-lattice grids. These methods are interesting, but they are not somewhat under-control and difficult to be analyzed mathematically. The works of Picioreanu and his group [47, 48] are examples of these types. The continuum models do not rely on ad-hoc rules but depend much on the conditions on the biofilm's surface. In this thesis, we work with a continuous model where there are continuous conditions (1.10) on the interface.

Work with fixed meshes. Naturally, a biofilm growth simulating has a close relation to *an interface problem*. There are two main approaches we can use to deal with such a problem, *fitted* and *unfitted* meshes (Figure 1.6). Sometimes, we also use names *Lagrangian representation* or *interface tracking* for the former and *Eulerian representation* of *interface capturing* for the latter. With a *fitted mesh*, the mesh grid is fitted to the interface, the standard finite element method (FEM) seems to work well [8, 14]. However, working on an unsteady state problem where the interface changes its position requires the mesh has to update frequently, an *unfitted mesh* where we work with only one mesh from the beginning is a better choice in this case. A comparison of using these two ideas with finite element methods for elliptic equations with smooth interfaces was given in [6].

There are many *unfitted mesh* based methods which have been proposed for recent years, one of them is *Immersed Boundary Method* (IIM) which is particularly designed for interface problems. The IIM is a sharp interface method based on Cartesian grids [38]. The work of Li [39] gave an overview of this method and some of its applications. However, when we work with discontinuous coefficients and singular sources, especially with large ones, the accuracy obtained by using IIM is not so good. Other interesting approaches are using *Extended Finite Element Method* (XFEM) [7, 40] and *Nitsche based Finite Element Method* (NFEM) [31]. XFEM is an extension of the standard finite element method in which arbitrary discontinuous

a. Fitted mesh.   b. Unfitted mesh.
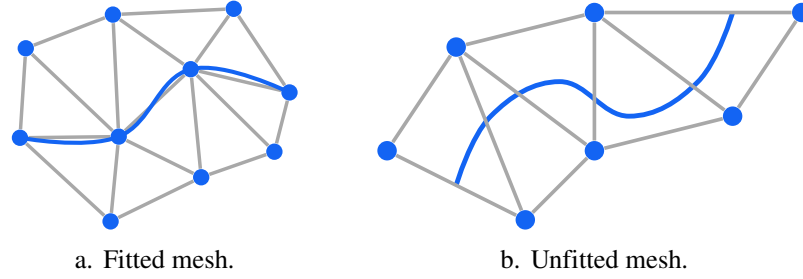
Figure 1.6. Fitted vs unfitted meshes.

functions and derivatives are added to the standard finite element approximation. Arnold Reusken and his coworkers had many contributions when using XFEM for two-phase incompressible flows problems [66, 36, 73, 50, 51]. NFEM uses the idea of Nitsche [43] to enforce weakly the interface condition in the weak forms. Note that, there is also relation between XFEM and NFEM which is commented in [3]. In this thesis, I will use NFEM and its innovated versions to work with the purpose of unfitted mesh. I use the name *NXFEM* to allude the relation of XFEM and NFEM. NXFEM is also called "unfitted FEM" or "CutFEM" in some literatures.

Describe interface & its motion. With NXFEM, we do not care too much for the position of the interface on a fixed mesh. However, the interface is not fixed at all, it changes in evolution problems. That leads to the need of describing numerically the interface at each time step. Thanks to *Level Set Method* (LSM) [45], this difficulty can be overcome. In Section 5.1, details about LSM, also the *Streamline Upwinding Petrov-Galerkin Method* (SUPG), will be presented. In this section, just to know that the interface can be described via a *level set function* ϕ as a zero-solution of a Cauchy problem (1.11).

$$\begin{cases} \dfrac{\partial \phi}{\partial t}(\mathbf{x},t) + \nabla \Phi(\mathbf{x},t) \cdot \nabla \phi(\mathbf{x},t) = 0, & \forall (\mathbf{x},t) \in \Omega \times (0,T), \\ \phi(\mathbf{x},0) = \phi^0(\mathbf{x}), & \forall \mathbf{x} \in \Omega. \end{cases} \tag{1.11}$$

Notice that, the level set function ϕ depends on the potential function Φ given in (1.9). Because of this relation, we can update the interface at each time step and use the new interface to seek the concentration of substrates and bacteria. Details of the idea on coupling NXFEM and LVS are given in Chapter 6.

Stabilization & Preconditioning. NXFEM is very powerful on tracking the interface but it also leads to a big problem on the stability of the scheme. The methods suffer from ill-conditioning if the interface intersects elements close to the nodes of the mesh and leaving a very small part of the element [9, 10, 11, 50] (Figure 2.6 gives an illustration about this). The *condition number* will be very large in this situation. Additionally, the problem is worse if we work on a big contrast problem where there is very difference between diffusion coefficients. Two possible options to be chosed to overcome these difficulties are *preconditioning* and *stabilization*. The former is given in [12, 37, 50, 58]. In these works, the authors proposed some optimal subspace preconditioners in which the condition number is independent of the mesh size and

the interface position. These techniques also take effect on treating the contrast problem. The second option, stabilization, is given in a series of works [9, 11, 13, 18]. In these contributions, the authors introduced an additional term called *Ghost Penalty* which is added to the variational formulation over the band of elements that are cut by the interface. This term helps the scheme be more stable. In this thesis, I choose this technique to deal with the small cut and large contrast problems. More details about ghost penalty method are given in Section 2.4.

1.4   Outline of the thesis

The thesis is divided into two main parts. The first one is for the NXFEM method and the second one contains the way we apply this method to researching a biofilm growth model. The manuscript is organnized as follows:

Chapter 2 In this chapter, I am going to recall the main idea of NXFEM and some principle results which first proposed by A. Hansbo & P. Hansbo [31] and later developed by some other authors. I start with the idea of Nitsche method on a non interface problem and then give details of NXFEM on an interface problem which borrows its idea.

Chapter 3 In this chapter, I will give in details about algorithms and guide to use `NXFEM toolbox` developed by myself. I build it based on the idea of space proposed by Hansbo in Section 2.5, coming from the idea of implementing Standard FEM in Matlab. In other point of view, this chapter is also used to implement NXFEM in other programming language instead of being used only in Matlab.

Chapter 4 Under the motivation of modeling a biofilm model, we introduce a system of semilinear unsteady interface problem. We also propose a technique of decoupling a semilinear problem and apply the NXFEM method to prove the existence and uniqueness of solutions and their convergent properties. In order to prove the convergence of NXFEM discrete solutions to the continuous ones, we apply the idea of proofs in Discontinuous Galerkin Method proposed by Ern & Di Pietro [25]. Their work relied on techniques inspired by the Finite Volume literature given in the work of Eymard et al. [29]. Noting that, Ern & Di Pietro worked on the discontinuity on each side of element mesh whereas we only work on the discontinuity of functions on the interface.

Chapter 5 As mentioned in Section 1.3, we need to track the interface's position on a fixed mesh from time to time. The *Level Set Method* helps us to do that. In this chapter, I present a general idea of LSM, as well as its advantages, its inherent drawback and the way we couple it with NXFEM in solving an evolution problem. Some numerical test cases are also given.

Chapter 6 This last chapter provides a way we use the NXFEM method and the toolbox NXFEM in solving biofilm growth models. The models we use are introduced in literatures, but the methods are different. We also have comments on the dependence of the model on parameters.

nxfem method

2. Original NXFEM

Contents

In this chapter, I am going to recall the main idea of NXFEM and some principle results which first proposed by A. Hansbo & P. Hansbo [31] and later developed by some other authors. I start with the idea of Nitsche method on a non interface problem and then give details of NXFEM on an interface problem which borrows its idea.

2.1 Nitsche's method

Nitsche didn't present his work with a weak formulation at all. He worked on an equivalent minimization problem in which the essential boundary conditions are imposed in a weak sense so that we can solve a Dirichlet problem without enforcing the boundary condition in the definition of the finite element space. The idea of

applying this technique to a weak formulation is given in later works, e.g. [30]. To describe the method, we consider a simple Poisson's problem (2.1).

$$\begin{cases} -\Delta u = f & \text{on } \Omega, \\ u = g & \text{on } \partial\Omega. \end{cases} \tag{2.1}$$

We would like to find a weak formulation that: (i) is satisfied by a (weak) solution $u \in H^1(\Omega)$, i.e. *consistency*; (ii) makes the associated bilinear form be *symmetric* and *coercive*. Firstly, multiplying both sides of (2.1) by a test function $v \in H^1(\Omega)$ and using Green formula, we obtain

$$\begin{aligned} \langle f, v \rangle_\Omega = \langle -\Delta u, v \rangle_\Omega &= \langle \nabla u, \nabla v \rangle_\Omega - \langle \nabla_\mathbf{n} u, v \rangle_{\partial\Omega} \\ &= \langle \nabla u, \nabla v \rangle_\Omega - \langle \nabla_\mathbf{n} u, v \rangle_{\partial\Omega} - \langle u - g, \nabla_\mathbf{n} v \rangle_{\partial\Omega}, \end{aligned} \tag{2.2}$$

where in the last equality, we have added the condition $u = g$ on the boundary $\partial\Omega$ to get a symmetric form. Here, $\mathbf{n}$ denotes *the unit normal vector* at a given point on $\partial\Omega$ pointing ouside the domain, $\langle u, v \rangle_K := \int_K uv$ in a convex set or on a line. When $u \in H^1(\Omega)$, it's obviously a solution of (2.2). We also need an additional term $\lambda \langle u - g, v \rangle_{\partial\Omega}$ where λ is large enough to guarantee the coercivity. Finally, we consider the binlinear form (2.3)

$$\langle \nabla u, \nabla v \rangle_\Omega \underbrace{- \langle \nabla_\mathbf{n} u, v \rangle_{\partial\Omega}}_{\text{consistency}} \underbrace{- \langle \nabla_\mathbf{n} v, u \rangle_{\partial\Omega}}_{\text{symmetry}} + \underbrace{\lambda \langle u, v \rangle_{\partial\Omega}}_{\text{stabilization}}$$
$$= \langle f, v \rangle_\Omega - \langle g, \nabla_\mathbf{n} v \rangle_{\partial\Omega} + \lambda \langle g, v \rangle_{\partial\Omega}, \quad \forall v \in H^1(\Omega). \tag{2.3}$$

Put in a context where we consider a discrete solution u_h in a finite element space,

$$V_h := \{ v_h \in H^1(\Omega) : v_h|_K \in \mathbb{P}^k(K), \forall K \in \mathscr{T}_h \}, \tag{2.4}$$

where $\mathscr{T}_h$ is a conforming triangulation of domain Ω. Nitsche showed that if λ is chosen in a form of ch^{-1} for $c > 0$ sufficiently large and h is the mesh size, the discrete problem is stable in $\|\|\cdot\|\|_N$ norm which is defined in (2.5).

$$\|\|v\|\|_N := \|v\|^2_{H^1(\Omega)} + \sum_{E \in G_h} h_E^{-1} v^2 \, dx, \tag{2.5}$$

for a subset $G_h \subset \partial\Omega$.

2.2 Original NXFEM

Model. Consider problem (2.6) on a bounded domain $\Omega \subset \mathbb{R}^2$ with a convex polynomial boundary $\partial\Omega$ and a smooth interface Γ dividing Ω into two open sets Ω_1, Ω_2.

$$\begin{cases} -\nabla \cdot (k\nabla u) = f & \text{on } \Omega_1 \cup \Omega_2, \\ \llbracket u \rrbracket = 0 & \text{on } \Gamma, \\ \llbracket k\nabla_{\mathbf{n}} u \rrbracket = g & \text{on } \Gamma, \\ u = 0 & \text{on } \partial\Omega, \end{cases} \tag{2.6}$$

where *the jumb operator* $\llbracket \cdot \rrbracket$ is defined in Definition 2.1.

Assumption 2.1 We assume that $f \in L^2(\Omega), g \in H^{1/2}(\Gamma)$ and k is constant in Ω_i with $\alpha_i > 0$ for $i = 1, 2$.

Weak form. The weak formulation of problem (2.6) is as follows: Find $u \in H_0^1(\Omega)$ such that

$$\langle k\nabla u, \nabla v \rangle_\Omega = \langle f, v \rangle_\Omega + \langle g, v \rangle_\Gamma, \forall v \in H_0^1(\Omega). \tag{2.7}$$

Triangulation. Let h_K be the diameter of each element K in $\mathscr{T}_h$ and $h := \max_{K \in \mathscr{T}_h} h_K$. Based on the location of interface Γ, we also define the set of triangles covering each subdomain Ω_i by $\mathscr{T}_h^i := \{K \in \mathscr{T}_h : K \cap \Omega_i \neq \emptyset\}$ and $G_h := \{K \in \mathscr{T}_h : K \cap \Gamma \neq \emptyset\}$ is the set of all cells that are intersected by Γ. For each $K \in G_h$, let $\Gamma_K := \Gamma \cap K$ be the part of Γ in K. For each $K \in \mathscr{T}_h$, let $K_i := K \cap \Omega_i$ be the part of K in Ω_i. In this session, we will use the same assumptions stated in [31] for the triangulation.

a. $\mathscr{T}_h$.   b. G_h.

c. $\mathscr{T}_h^1$.   d. $\mathscr{T}_h^2$.
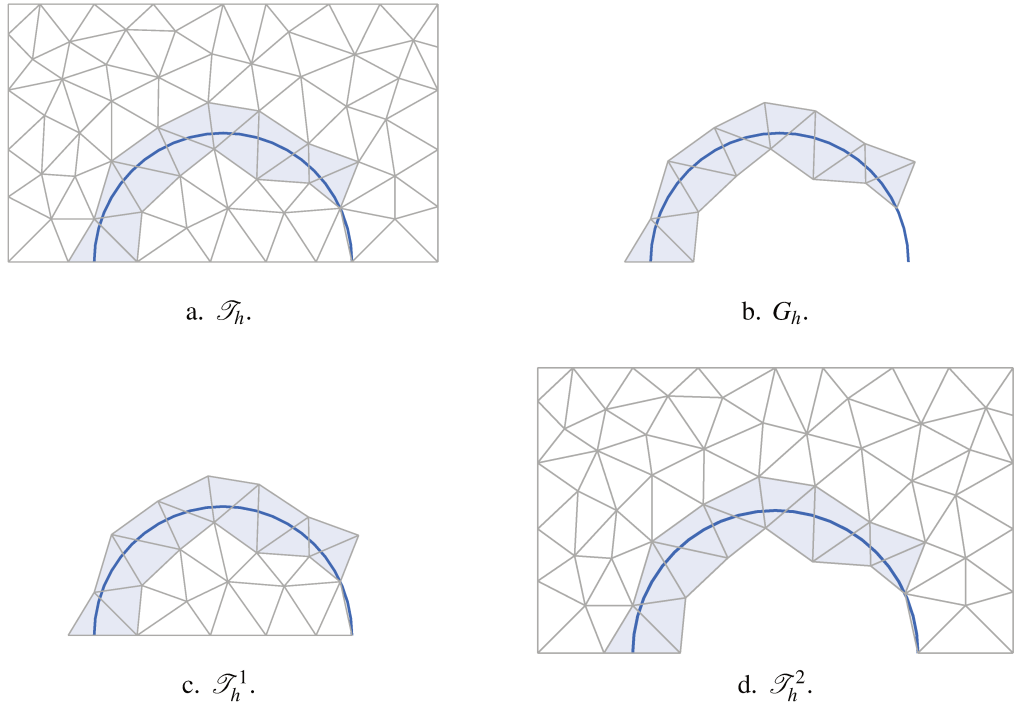
Figure 2.1. Illustration of the triangulation.

Assumption 2.2 The triangulation is non-degenerate, i.e. $\frac{h_K}{\rho_K} \leq C, \forall K \in \mathscr{T}_h$ for some

$C > 0$ where h_K and ρ_K are the diameter of K and the diameter of the largest ball in K respectively.

Assumption 2.3 For $K \in G_h$, Γ cuts each element boundary ∂K exactly twice and each open edge at most once.

Assumption 2.4 Let $\Gamma_{K,h}$ be the segment connecting the intersection points between Γ and ∂K, we assume that $\Gamma_{K,h}$ is the function of length on Γ_K in the local coordinates (2.8).

$$\begin{aligned}\Gamma_{K,h} &= \{(x,y) : 0 < x < |\Gamma_{K,h}|, y = 0\},\\ \Gamma_K &= \{(x,y) : 0 < x < |\Gamma_{K,h}|, y = \delta(x)\}.\end{aligned} \tag{2.8}$$

> **Remark 2.1 — Cut triangle.** A triangle is called a *cut triangle* if the Assumption 2.3 is satisfied and at least one cut point is not the vertex of this triangle. Some special cases of cut or not-cut triangles are given in Figure 2.2. Note that, we are considering a $\mathbb{P}^1$ - finite element space so Γ on each triangle is actually a line segment, that why we see all three cases a, b and c in Figure 2.2 are the same. That is the reason why in the triangulation $\mathcal{T}_h$, there is some triangle we see that it is cut by interface but it is still considered as a not-cut triangle (Figure 2.3).

a. It doesn't satisfy Assumption 2.3. b. Not-cut triangle. c. It doesn't satisfy Assumption 2.3.

d. Cut two edges. e. Go through one vertex and cut one edge.
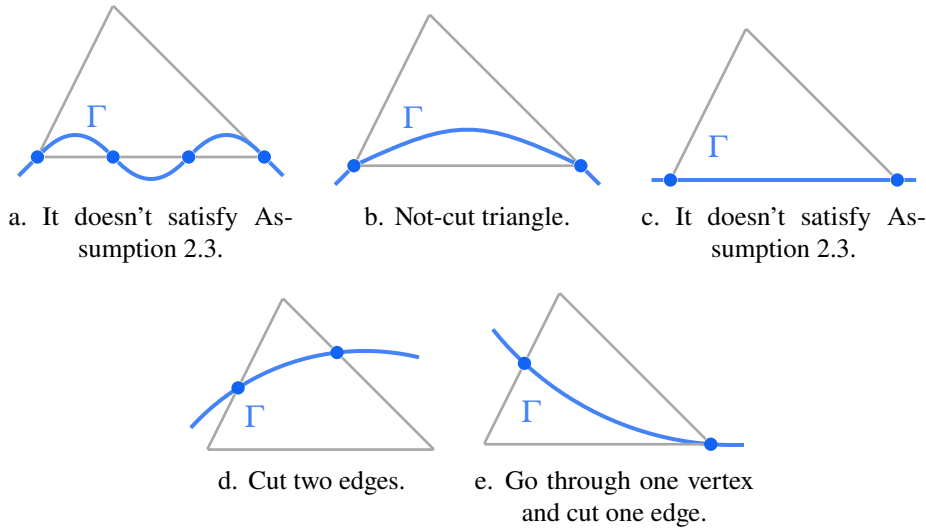
Figure 2.2. Some special cases of cut triangles (d,e) or not-cut triangles (a,b,c).

> **Definition 2.1 — Jump operator & average operator .**
>
> Before defining these two operators, we need to define a restriction of u on:
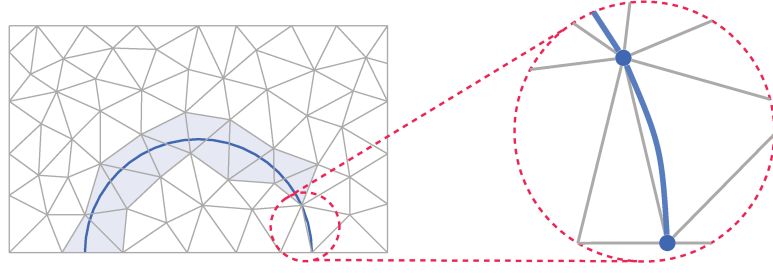
Figure 2.3. Zoom-in a special triangle of $\mathscr{T}_h$.

- each subdomain Ω_i of Ω for $i = 1, 2$ at a point x on Γ as in (2.9) where **n** pointing from Ω_1 to Ω_2.
- two adjacent triangles at a point x on $e = K_1 \cap K_2$ as in (2.9) where **n** pointing from K_1 to K_2.

$$
\begin{aligned}
u_1(x) &:= \lim_{\varepsilon \to 0} u(x - \varepsilon \mathbf{n}), \\
u_2(x) &:= \lim_{\varepsilon \to 0} u(x + \varepsilon \mathbf{n}),
\end{aligned}
\tag{2.9}
$$

Then,

- **Jump** is defined in (2.10),

$$
\begin{aligned}
[\![u]\!] &:= u_1|_\Gamma - u_2|_\Gamma, \\
[\![u]\!]_e &:= u_1|_e - u_2|_e.
\end{aligned}
\tag{2.10}
$$

  where e is a triangle's edge or a part of the interface.
- **Average** is defined in (2.11),

$$
\begin{aligned}
\{\!\{u\}\!\} &:= \kappa_1 u_1|_\Gamma + \kappa_2 u_2|_\Gamma, \\
\{\!\{u\}\!\}_e &:= \kappa_1 u_1|_e + \kappa_2 u_2|_e,
\end{aligned}
\tag{2.11}
$$

  where weight parameters κ_1, κ_2 satisfies $\sum \kappa_i = 1, \kappa_i > 0$.

An 1D illustration is given in Figure 2.4.
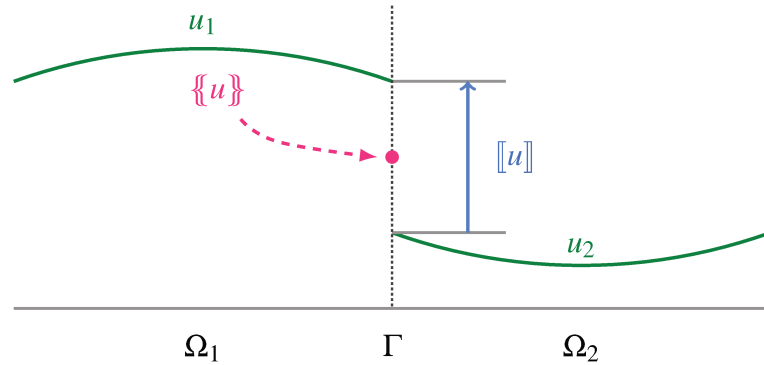
Figure 2.4. 1D example of jump and average operators in the case of $\kappa_1 = \kappa_2 = \frac{1}{2}$.

> **Proposition 2.1** With the jump and average operators defined in Definition 2.1 and u, v two discontinuous functions across Γ, we have,
>
> $$[\![uv]\!] = [\![u]\!]\{\!\{v\}\!\} + \{\!\{u\}\!\}[\![v]\!] + (\kappa_2 - \kappa_1)[\![u]\!][\![v]\!] \text{ and}$$
> $$\{\!\{uv\}\!\} = \{\!\{u\}\!\}\{\!\{v\}\!\} + \kappa_1\kappa_2[\![u]\!][\![v]\!].$$

Proof. Denote $u_i = u|_{\Omega_i}, v_i = v|_{\Omega_i}$ for $i = 1, 2$,

$$
\begin{aligned}
[\![uv]\!] = u_1 v_1 - u_2 v_2 &= \sum \kappa_i u_1 v_1 + \sum \kappa_i u_2 v_2 \\
&= 2\kappa_1 u_1 v_1 - 2\kappa_2 u_2 v_2 + (\kappa_2 - \kappa_1)(u_1 v_1 + u_2 v_2) \\
&= 2\kappa_1 u_1 v_1 - 2\kappa_2 u_2 v_2 + (\kappa_2 - \kappa_1)(u_1 v_2 + u_2 v_1) + (\kappa_2 - \kappa_1)[\![u]\!][\![v]\!] \\
&= \kappa_1 u_1 v_1 + \kappa_2 u_1 v_2 - \kappa_1 u_2 v_1 - \kappa_2 u_2 v_2 \\
&\quad + \kappa_1 u_1 v_1 - \kappa_1 u_1 v_2 + \kappa_2 u_2 v_1 - \kappa_2 u_2 v_2 + (\kappa_2 - \kappa_1)[\![u]\!][\![v]\!] \\
&= [\![u]\!]\{\!\{v\}\!\} + \{\!\{u\}\!\}[\![v]\!] + (\kappa_2 - \kappa_1)[\![u]\!][\![v]\!].
\end{aligned}
$$

We also have

$$
\begin{aligned}
\{\!\{u\}\!\}\{\!\{v\}\!\} + \kappa_1\kappa_2[\![u]\!][\![v]\!] &= (\kappa_1 u_1 + \kappa_2 u_2)(\kappa_1 v_1 + \kappa_2 v_2) + \kappa_1\kappa_2(u_1 - u_2)(v_1 - v_2) \\
&= \kappa_1^2 u_1 v_1 + \kappa_2^2 u_2 v_2 + \kappa_1\kappa_2 u_1 v_1 + \kappa_1\kappa_2 u_2 v_2 \\
&= \{\!\{uv\}\!\}.
\end{aligned}
$$

■

> **Remark 2.2** For simplicity, there are many works in that authors choose $\kappa_1 = \kappa_2 = \frac{1}{2}$. There are also many choices of forms of κ_i which are discussed in Section 2.3, Hansbo & Hansbo [31] used (2.12) where $|K|$ is the measure of K.
>
> $$\kappa_i|_K := \frac{|K_i|}{|K|}. \tag{2.12}$$

> **Remark 2.3** In [5, 51], authors used a different definition of average operator,
>
> $$\{\!\{u\}\!\}_\kappa := \kappa_1 u_1 + \kappa_2 u_2,$$
> $$\{\!\{u\}\!\}_{\hat{\kappa}} := \kappa_1 u_1 + \kappa_2 u_2. \tag{2.13}$$
>
> which leads to a different relation
>
> $$[\![uv]\!] = [\![u]\!]\{\!\{v\}\!\}_\kappa + \{\!\{u\}\!\}_{\hat{\kappa}}[\![v]\!].$$

In the case $\kappa_1 = \kappa_2 = \frac{1}{2}$, both of two definitions (2.11) and (2.13) of average operator give us the same relation

$$[\![uv]\!] = [\![u]\!]\{\!\{v\}\!\} + \{\!\{u\}\!\}[\![v]\!].$$

Definition 2.2   We use notation of function space $H^k(\Omega_{12})$ as follows,

$$H^k(\Omega_1 \cup \Omega_2) = \{v \in L^2(\Omega) : v_i \in H^k(\Omega_i) \text{ for } i = 1, 2\},$$

for $k = 0, 1, 2$ where $v_i = v|_{\Omega_i}$ and H^0 stands for L^2.

We equip space $H^k(\Omega_{12})$ by the norm:

$$\|u\|_{H^k(\Omega_{12})}^2 := \|u\|_{H^k(\Omega_1)}^2 + \|u\|_{H^k(\Omega_2)}^2,$$

Discrete form. Before stepping to the discrete problem, we define finite element spaces $V_h^\Gamma = V_h^1 \times V_h^2$ and discrete gradient associated to them as in (2.14), (2.15) and Definition 2.3.

$$V_h^i := \{v_h \in L^2(\Omega) : v_h|_{\Omega_i} \in H^1(\Omega_i) \text{ and } v_h|_K \in \mathbb{P}^1(K), \forall K \in \cup_i \mathscr{T}_h^i\}. \quad (2.14)$$

For functions taking zero values on $\partial\Omega$, we use space V_h^0,

$$V_h^0 := \{v_h \in V_h^\Gamma : v_h = 0 \text{ on } \partial\Omega\}. \quad (2.15)$$

Remark 2.4   Note that, $V_h^\Gamma \subset H^1(\Omega_{12})$ and the functions in V_h^Γ are no need to be continuous across the interface.

Definition 2.3 — Discrete gradient.   For all $v_h \in V_h^\Gamma$, we define ∇v_h as the piecewise gradient of v_h, such that

$$\nabla v_h|_{\Omega_i} = \nabla(v_h|_{\Omega_i}), \text{ for } i = 1, 2.$$

It's known in [63, Lemma 1.22] that the discrete gradient defined in Definition 2.3

is consistent corresponding to the the space $H^1(\Omega)$. There is a little difference, in this work, we are working on a discrete space in that the functions may be only discontinuous at the interface.

Multiply both sides of (2.6) by a test function $v \in H^1(\Omega_{12})$ and use Green formula again, but this time, we consider on each subdomain,

$$
\begin{aligned}
\langle f, v \rangle_\Omega &= \langle -\nabla \cdot (k\nabla u), v \rangle_{\Omega_1} + \langle -\nabla \cdot (k\nabla u), v \rangle_{\Omega_2} \\
&= \langle k\nabla u, \nabla v \rangle_{\Omega_{12}} - \langle k\nabla_{\mathbf{n}} u, v \rangle_{\partial\Omega_1} - \langle k\nabla_{\mathbf{n}} u, v \rangle_{\partial\Omega_2} \\
&= \langle k\nabla u, \nabla v \rangle_{\Omega_{12}} - \langle k\nabla_{\mathbf{n}} u, v \rangle_{\partial\Omega} - \int_\Gamma [\![k\nabla_{\mathbf{n}} uv]\!] \, \mathrm{d}s,
\end{aligned}
\tag{2.16}
$$

where $\mathbf{n}$ points from Ω_1 to Ω_2. Apply Propotition 2.1, (2.16) becomes

$$
\begin{aligned}
\langle f, v \rangle_\Omega ={}& \langle k\nabla u, \nabla v \rangle_{\Omega_{12}} - \langle k\nabla_{\mathbf{n}} u, v \rangle_{\partial\Omega} \\
&- \langle [\![k\nabla_{\mathbf{n}} u]\!], \{\!\{ v \}\!\} \rangle_\Gamma - \langle \{\!\{ k\nabla_{\mathbf{n}} u \}\!\}, [\![v]\!] \rangle_\Gamma + (\kappa_2 - \kappa_1) \langle [\![k\nabla_{\mathbf{n}} u]\!], [\![v]\!] \rangle_\Gamma.
\end{aligned}
\tag{2.17}
$$

> **Remark 2.5** We don't consider a test function in $H_0^1(\Omega)$ in the weak form (2.16) because we want to apply later a test function in a finite element space of which functions are not continuous at the interface. That's why we take $v \in H^1(\Omega_{12})$ instead.

In order to obtain the consistency, solution of problem (2.6) must satisfy (2.17). It means that the interface conditions and boundary condition in (2.6) take action and (2.17) becomes

$$
\langle k\nabla u, \nabla v \rangle_{\Omega_{12}} - \langle \{\!\{ k\nabla_{\mathbf{n}} u \}\!\}, [\![v]\!] \rangle_\Gamma = \langle f, v \rangle_\Omega - \langle g, \{\!\{ v \}\!\} \rangle_\Gamma.
\tag{2.18}
$$

With the same idea in Nitsche's method, we add two more terms in the discrete form to guarantee the symmetry and the coercivity. That leads us to consider a discrete problem of finding $u_h \in V_h^\Gamma$ such that,

$$
\boxed{a_h(u_h, v_h) = L_h(v_h), \forall v_h \in V_h^\Gamma,}
\tag{2.19}
$$

where[1],

$$
\begin{aligned}
a_h(u_h, v_h) :={}& \langle k\nabla u_h, \nabla v_h \rangle_{\Omega_{12}} \\
&\underbrace{- \langle \{\!\{ k\nabla_{\mathbf{n}} u_h \}\!\}, [\![v_h]\!] \rangle_\Gamma}_{\text{consistency}} \underbrace{- \langle [\![u_h]\!], \{\!\{ k\nabla_{\mathbf{n}} v_h \}\!\} \rangle_\Gamma}_{\text{symmetry}} \underbrace{+ \lambda \langle [\![u_h]\!], [\![v_h]\!] \rangle_\Gamma}_{\text{stabilization}}, \\
L_h(v_h) :={}& \langle f, v \rangle_\Omega - \langle g, \{\!\{ v \}\!\} \rangle_\Gamma,
\end{aligned}
\tag{2.20}
$$

---

[1] Compare a_h in this case with the one in (2.3) to see the similarity.

with λ sufficiently large.

Norms. Hansbo & Hansbo have used following norms for their results in [31],

$$\|v\|_{1/2}^2 := \sum_{K \in G_h} h_K^{-1} \|v\|_{L^2(\Gamma_K)}^2, \quad \|v\|_{-1/2}^2 := \sum_{K \in G_h} h_K \|v\|_{L^2(\Gamma_K)}^2,$$
$$\|\|v\|\|_H^2 := \|\nabla v\|_{L^2(\Omega_{12})}^2 + \|\{\nabla_{\mathbf{n}} v\}\|_{-1/2}^2 + \|[\![v]\!]\|_{1/2}^2. \tag{2.21}$$

Remark 2.6 It depends on the problem we are working on, choices of norm $\|\|\cdot\|\|_H$ are different. For example,

- In [5], Barrau considered diffusion coefficient inside the norm of gradient, i.e. $\|k \nabla v\|_{L^2(\Omega_{12})}$, because this coefficient is discontinous on the interface,
- In [51], Hieu Nguyen considered the interface condition $[\![\beta u]\!] = 0$ instead of the continuity $[\![u]\!] = 0$, he thus used $\|[\![\beta v]\!]\|_{1/2}$.

Definition 2.4 — Interpolant. Consider an operator $E_i : H^2(\Omega_i) \to H^2(\Omega)$ such that $(E_i w)|_{\Omega_i} = w$ and

$$\|E_i w\|_{s,\Omega} \leq C \|w\|_{s,\Omega_i}, \quad \forall w \in H^s(\Omega_i), s = 0, 1, 2.$$

Let I_h be the standard interpolation operator and define

$$I_h^* v := (I_{h,1}^* v_1, I_{h,2}^* v_2) \text{ where } I_{h,i}^* v_i := (I_h E_i v_i)|_{\Omega_i}. \tag{2.22}$$

We recall following results and useful remarks in the proofs. For more details in the arguments, please see [31].

Proposition 2.2 For $u \in V_h^\Gamma$, $\|\cdot\|_{-1/2}$ defined in (2.21) and $\kappa_i = \frac{|K_i|}{|K|}$, the following inverse inequality holds,

$$\|\nabla_{\mathbf{n}} u\|_{-1/2}^2 \leq C \|\nabla u\|_{L^2(\Omega_{12})}^2. \tag{2.23}$$

Proof. Cf. Lemma 4 in [31]. Note that, the validity of the inequality depends on the choice of κ_i. In this case, Hansbo & Hansbo have used $\kappa_i|_K = \frac{|K_i|}{|K|}$. If we want to use other definitions, e.g. Section 2.3, we have to verify this property again. ∎

With the interpolation defined in Definition 2.4, we have a very importance inter-

polant estimate which are used later in the proof of next results.

> **Theorem 2.1** Let I_h^* be the interpolation operator defined in (2.22), then
>
> $$\|u - I_h^* u\|_H \leq Ch\|u\|_{2,\Omega_{12}}, \forall u \in H_0^1(\Omega) \cap H^2(\Omega_{12}).$$

Proof. A detailed proof is given in [31, Theorem 2]. Note that, the estimate depends much on the definition of norm $\|\cdot\|_H$. In following chapters, we will use different norms and thus, we need to prove again this estimate. ∎

Hansbo & Hansbo have also proved the *consistency* [31, Lemma 1], *existence* and *uniqueness* of the problem (2.19) by following proposition.

> **Proposition 2.3** The discrete form a_h in (2.20) is coercive on V_h^Γ, i.e.,
>
> $$a_h(v_h, v_h) \geq C\|v_h\|_H^2, \forall v_h \in V_h^\Gamma,$$
>
> provided λ is sufficiently large. It's also continuous, i.e.,
>
> $$a_h(u_h, v_h) \leq C\|u_h\|_H\|v_h\|_H, \forall v_h \in V_h^\Gamma.$$

Proof. Refer to [31, Lemma 5]. In this case, they choose λ such that $\lambda > 4C \max_\Omega k$ where C is the coefficient obtained from inequality (2.23) and k is diffusion coefficients in problem (2.6). ∎

They also gave some results about *a priori* error estimates in Theorem 2.2, for the *a posteriori* error estimates, cf. [31, Theorem 7].

> **Theorem 2.2 — *a priori* error estimates.** Under Assumptions 2.2–2.4 and for u_h solving (2.19), u solving (2.6), the following estimates hold,
>
> $$\|u_h - u\|_H \leq Ch\|u\|_{H^2(\Omega_{12})}, \tag{2.24}$$
>
> $$\|u_h - u\|_{L^2(\Omega_{12})} \leq Ch^2\|u\|_{H^2(\Omega_{12})}. \tag{2.25}$$

Proof. Cf. [31, Theorem 6]. ∎

2.3 The choice of parameters

The roles of parameters κ_1, κ_2 in Definition 2.1 and penalty parameter λ in (2.19) are very important because they impact on the stability of the method. As mentioned

in the proof of Propotition 2.2, Hansbo & Hanbso chose very simple ones as in (2.26),

$$\kappa_1 = \frac{|K_1|}{|K|}, \; \kappa_2 = \frac{|K_2|}{|K|}, \; \lambda = \hat{\lambda} \max \left\{ k_1 \frac{|K_1|}{|K|}, k_2 \frac{|K_2|}{|K|} \right\}, \tag{2.26}$$

for some large enough positive constant $\hat{\lambda}$.

In [5], the authors showed that, the choice (2.26) makes the stability depend *only* on the cut geometry (where the interface cuts triangles) whereas the difference between diffusion coefficients k_i is also very important for a robust method. Therefore, they proposed an update choice (2.27) which gives robustness regarding both the geometry and the coefficients.

$$\kappa_1 = \frac{k_2|K_1|}{k_1|K_2| + k_2|K_1|}, \; \kappa_2 = \frac{k_1|K_2|}{k_1|K_2| + k_2|K_1|}, \; \lambda = \hat{\lambda} \frac{k_1 k_2 |\Gamma_K|}{k_1|K_2| + k_2|K_1|}. \tag{2.27}$$

Another option for λ, corresponding to weights κ_i in (2.27), is[2],

$$\lambda = \hat{\lambda} h_K \frac{k_1 k_2 |\Gamma_K|}{k_1|K_2| + k_2|K_1|}. \tag{2.28}$$

Remark 2.7 For simplicity, we use notations κ_i and λ but they actually are defined on each cut triangle, i.e., $\kappa_i|_K$ and $\lambda|_K$.

All above researches give advice in which we need to choose a sufficiently large $\hat{\lambda}$. However, the benefit of an increasing λ is two-faced. It helps to stabilize the discretization and to make a small error in the interface condition. However, it leads an increase in the *condition number* which is due to the fashion of which the triangles are cut by the interface.

Remark 2.8 A stabilization technique is necessary to control the possible bad quality of how interface intersects the computational mesh at tiny different parts. In this thesis, we use a method named *Ghost penalty* [9, 13] in which some additional "ghost penalty" terms, acting on the jumps of the gradients over element faces in the interface zone, are added. Notice that, there is also another approach [32] which is inspired by the XFEM in the adaptation of a stabilisation technique presented by Barbosa and Huges [4]. This method has an advantage of not fitting the parameters like Ghost Penalty. However, it is necessary to consider mixed formulations with Lagrange multipliers for which

Figure 2.5. An illustration of a fictitious domain $\Omega_{\mathscr{T}}$ of a physical domain Ω.
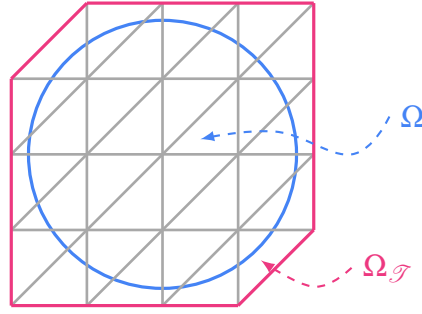
it is not always easy to establish inf-sup conditions in appropriate spaces. For simplicity of not changing our variational forms, also let it be a future research, we consider using Ghost Penalty method under the possibility of choosing good values for the parameters.

$$\kappa_1 = \frac{k_2}{k_1 + k_2}, \; \kappa_2 = \frac{k_1}{k_1 + k_2}, \; \lambda = 4\hat{\lambda} \frac{k_1 k_2}{k_1 + k_2}, \tag{2.29}$$

where $\hat{\lambda}$ which is independent of the mesh intersection but depends on the interface is given in [13].

Remark 2.9 In practice, we need to choose empirically the value of $\hat{\lambda}$. There are also some big relations of the parameters used in the method. The above authors and works didn't mention about it too much. In this thesis, I will give a more clarity about the choice of parameters regarding some validation test cases. Please step to Section 2.6 for more information.

2.4 Ghost penalty & Stability property

Fictitious domain method. Burman *et al.* [10, 11] have worked with *fictitious domain methods* because these methods are effective for complex-shape domains and also free-interface ones. The basic idea of fictitious domain methods is instead of considering the physical domain Ω, we consider a larger domain $\Omega_{\mathscr{T}}$ which has simpler geometry (Figure 2.5). The restriction to $\Omega_{\mathscr{T}}$ of the solution coincides with the original problem's solution which is on Ω. However, people who work with these methods have to integrate on the fictitious domain $\Omega_{\mathscr{T}}$ or on the physical domain Ω. The former gives better stabilization but lack of consistency and accuracy. The latter gives accuracy, but the condition number is sensitive to the way domain boundary cuts the mesh.

Ghost penalty. Put in the context of interface problem, the classical XFEM is not good because of the ill-conditioned problem when the interface divides a triangle into very different size parts [18] (Figure 2.6). This difficulty, coupling with a *large*
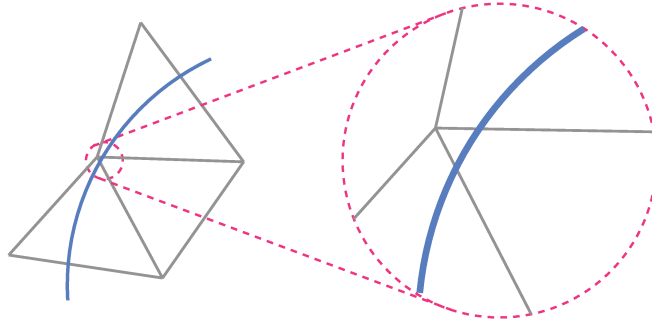
Figure 2.6. Interface cuts triangle at positions which close to its vertex.

a. Set $\mathscr{E}_G^1$ of edges in $\mathscr{T}_h^1$ for $j_1(u_h, v_h)$. In this case, $\Omega_{\mathscr{T}}^1$ considered as a fictitious domain applied to Ω_1.

b. Set $\mathscr{E}_G^2$ of edges in $\mathscr{T}_h^2$ for $j_2(u_h, v_h)$. In this case, $\Omega_{\mathscr{T}}^2$ considered as a fictitious domain applied to Ω_2.
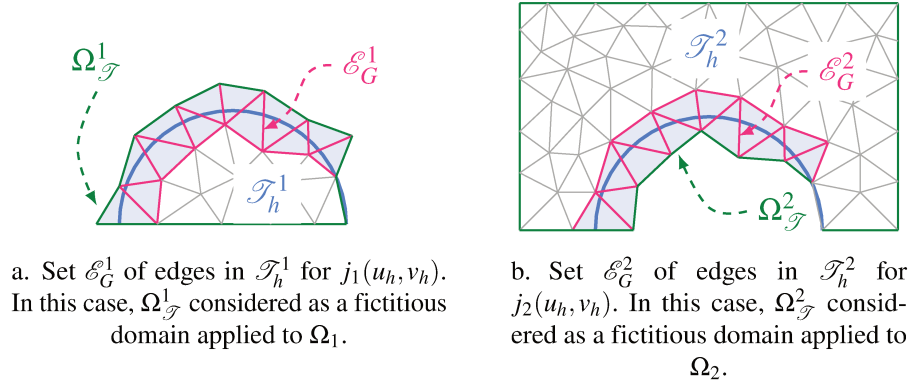
Figure 2.7. Considered edges for ghost penalty terms.

contrast problem on the diffusion coefficients [12], leads the interface problem to be more difficult to stabilize. With the help of ghost penalty method [9], Burman *et al.*, they are able to handle these two problems at the same time by adding ghost-penalty terms to the variational formulation over the band of elements cut by the interface. The conditioning number of the matrix is independent of the way interface cuts the mesh [11].

Instead of working on a fictitious domain $\Omega_{\mathscr{T}}$, Burman *et al.* see each subdomain Ω_i in an union with area around the interface G_h as a new "separated fictitious domain" $\Omega_{\mathscr{T}}^i := \Omega_i \cup G_h$, (Figure 2.7). Note that, corresponding to each $\Omega_{\mathscr{T}}^i$ is the sub-triangulation $\mathscr{T}_h^i$.

The method states that in place of working with the discrete problem (2.19), we work with

$$a_h(u_h, v_h) + k_1 j_1(u_h, v_h) + k_2 j_2(u_h, v_h) = L_h(v_h), \forall v_h \in V_h^\Gamma, \tag{2.30}$$

where $a_h(\cdot, \cdot)$ and $L_h(\cdot)$ are given in (2.20) and

$$j_r(u_h, v_h) = \sum_{e \in \mathscr{E}_G^r} \langle \gamma_r h [\![\nabla_{\mathbf{n}_e} u_h]\!]_e, [\![\nabla_{\mathbf{n}_e} v_h]\!]_e \rangle_e, \quad r = 1, 2. \tag{2.31}$$

In this case, we add two more ghost penalty terms $j_i(\cdot,\cdot)$ to the original bilinear form $a_h(\cdot,\cdot)$ in order to control the method's stabilization. Segments $\mathscr{E}_G^i$ which corresponds to each subdomain Ω_i is defined in (2.32) (Figure 2.7) and the jump operator on an edge e is defined in (2.10).

$$\forall e \in \mathscr{E}_G^i, \exists K, K' \in \mathscr{T}_h^i : e = K \cap K', K \in G_h \text{ or } K' \in G_h. \tag{2.32}$$

We can understand the definition of $\mathscr{E}_G^i$ as follows: for each edge $e \in \mathscr{E}_G^i$, there exists two elements K and K' in $\mathscr{T}_h^i$ such that e is their common edge and at least one of them is the member of G_h. This means, in particular, the boundary edges of $\mathscr{T}_h^i$ are excluded from $\mathscr{E}_G^i$.

> **Remark 2.10** Remind that if we want to use the ghost penalty method, it's necessary to take parameters defined in (2.29)[13].

An idea of the implementation ghost penalty is described in Section 3.4.3.

2.5 The implementation issue

Representation of NXFEM space. In the original work [31], the authors have used NXFEM space defined in (2.14). This style of NXFEM space is a type of *domain decomposition method* (DDM) where we seek a solution in each subdomain separately.

Another interesting representation of V_h^Γ which is proposed in [50] is in the style of the XFEM method. It means that the NXFEM space is a direct decomposition of spaces $V_h, V_{h,1}^\Gamma, V_{h,2}^\Gamma$:

$$V_h^\Gamma = V_h \oplus V_{h,1}^\Gamma \oplus V_{h,2}^\Gamma. \tag{2.33}$$

where V_h is a standard $\mathbb{P}_1$ finite element space defined in (2.4) and $V_{h,i}^\Gamma$ is defined as follows (an illustration in 1D is given in Figure 2.8).

We start by defining an operator

$$R_i : L^2(\Omega) \to L^2(\Omega), \quad R_i v := \begin{cases} v|_{\Omega_i} & \text{in } \Omega_i, \\ 0 & \text{in } \Omega \setminus \Omega_i. \end{cases}$$

Let $\mathscr{I}$ be a set of indexes numbering the nodes associated with V_h and let $\{x_k\}_{x \in \mathscr{I}}$ be the corresponding set of points on Ω. We define subsets of $\mathscr{I}$ which neighbor the interface as
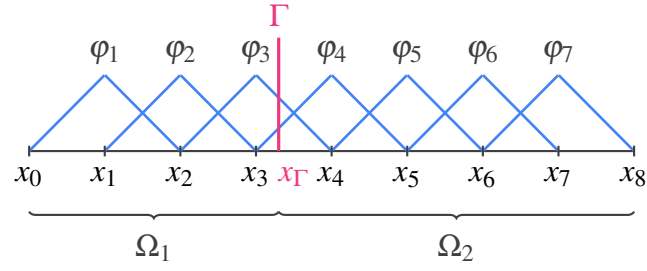
a. Satndard FE space $V_h = \mathrm{span}\{\varphi_1, \ldots, \varphi_7\}$ and indices of nodes $\mathscr{I} = \{x_1, \ldots, x_7\}$, $\mathscr{I}_1^\Gamma = \{x_4\}$, $\mathscr{I}_2^\Gamma = \{x_3\}$.
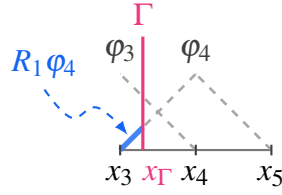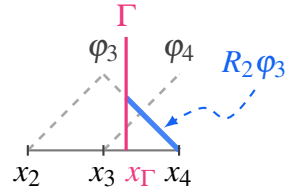
b. $V_{h,1}^\Gamma = \mathrm{span}\{R_1 \varphi_4\}$.   c. $V_{h,2}^\Gamma = \mathrm{span}\{R_2 \varphi_3\}$.

Figure 2.8. An example of V_h^Γ in 1D. $V_h^\Gamma = \mathrm{span}\{\varphi_1, \ldots, \varphi_7\} \oplus \mathrm{span}\{R_1 \varphi_4\} \oplus \mathrm{span}\{R_2 \varphi_3\}$.

$$\mathscr{I}_i^\Gamma := \{k \in \mathscr{I} : x_k \in \Omega_j, \mathrm{supp}(\varphi_k) \cap G_h \neq \emptyset\}, \forall i, j = 1, 2 \; j \neq i, \qquad (2.34)$$

where φ_k are the standard basis functions associated with the node x_k. Then we have the definition of spaces $V_{h,i}^\Gamma$

$$V_{h,i}^\Gamma := \mathrm{span}\{R_i \varphi_k : k \in \mathscr{I}_i^\Gamma\}.$$

It's known in [50, Theorem 2] that $V_h^\Gamma = V_A^h$. Because of this relation, there are many ways to characterize technically an NXFEM space. Below are two typical choices given by Hansbo *et al.* and Reusken *et al.* In this thesis, I use the former.

The choice of Hansbo. In [31, Section 7], the authors gave a technical description of space V_H^h. From the standard basis functions in V_h, we replace ones living on the elements intersected by the interface by two new basis functions, one is restricted to Ω_1, the other is restricted to Ω_2. More detailed, let $\{\varphi_i\}_i$ be a standard basis of V_h. Let $\mathscr{I}_\Gamma$ be a set of indexes numbering the nodes associated with G_h (a set of elements intersected by the interface). In other words,

$$\mathscr{I}_\Gamma := \{i \in \mathscr{I} : |\Gamma \cap \mathrm{supp}(\varphi_i)| > 0\}. \qquad (2.35)$$

For each $i \in \mathscr{I}_\Gamma$, we define two new basis functions $\varphi_i^{(1)}, \varphi_i^{(2)}$ from the standard basis function φ_i such that

$$\boxed{\varphi_i^{(1)} := \varphi_i|_{\Omega_1}, \quad \varphi_i^{(2)} := \varphi_i|_{\Omega_2},} \qquad (2.36)$$

and then an NXFEM space is defined as

$$V_h^\Gamma := \mathrm{span}\{\varphi_i | i \in \mathscr{I} \setminus \mathscr{I}_\Gamma\} \oplus \mathrm{span}\{\varphi_i^{(1)} | i \in \mathscr{I}_\Gamma\} \oplus \mathrm{span}\{\varphi_i^{(2)} | i \in \mathscr{I}_\Gamma\}. \quad (2.37)$$

These new basis functions are discontinuous at the interface and take the same values as the original standard ones, namely,

$$\begin{cases} \varphi_i^{(1)}|_{\Omega_1}(x_\Gamma) &= \lim_{x|_{\Omega_1} \to x_\Gamma} \varphi_i(x) = \varphi_i(x_\Gamma), \\ \varphi_i^{(2)}|_{\Omega_1}(x_\Gamma) &= \lim_{x|_{\Omega_1} \to x_\Gamma} \varphi_i^{(2)}(x) = 0, \\ \varphi_i^{(1)}|_{\Omega_2}(x_\Gamma) &= \lim_{x|_{\Omega_2} \to x_\Gamma} \varphi_i(x) = 0, \\ \varphi_i^{(2)}|_{\Omega_2}(x_\Gamma) &= \lim_{x|_{\Omega_2} \to x_\Gamma} \varphi_i^{(2)}(x) = \varphi_i^{(2)}(x_\Gamma) = \varphi_i(x_\Gamma). \end{cases} \quad (2.38)$$
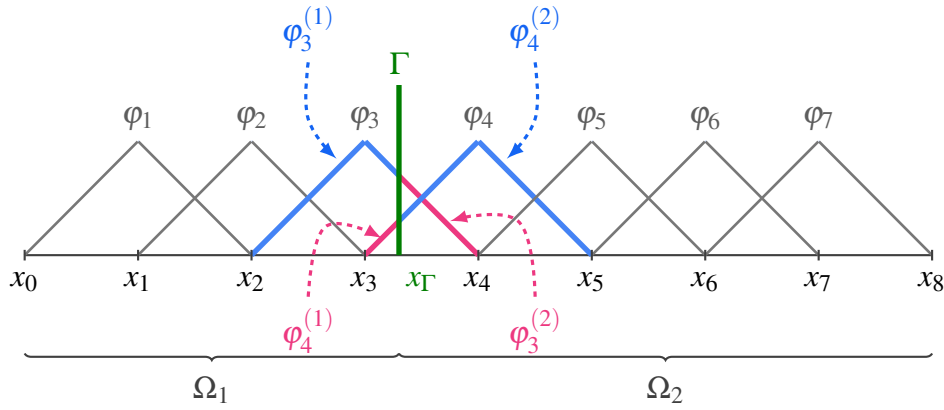
Figure 2.9. 1D example of basis proposed by Hansbo & Hansbo.
$$V_H^h = \mathrm{span}\{\varphi_1, \varphi_2, \varphi_5, \varphi_6, \varphi_7\} \oplus \mathrm{span}\{\varphi_3^{(1)}, \varphi_3^{(2)}, \varphi_4^{(1)}, \varphi_4^{(2)}\}.$$

An illustration in 1D of basis functions described in (2.38) is given in Figure 2.9.

> **Remark 2.11** The choice of Hansbo doesn't use all standard basis functions. We will keep functions whose support are not cut by the interface while the ones cut by the interface will be "doubled" into two new ones. We use the same index numbering for functions whose support on Ω_1, i.e. $\varphi_i = \varphi_i^{(1)}$, for an easier implementation in Matlab (cf. Chapter 3).

The choice of Reusken. In [66, Section 7.9.2], the author gave a detailed description of V_A^h in which the standard finite element space is enriched by certain functions which is discontinuous across the interface. He also gave a stability property of such basis functions in the L^2 norm [50] and it was updated later in the H^1 norm by Zunino *et al.* [58].

Using a *Heaviside function* defined as follow

$$H_\Gamma(x) = \begin{cases} 0, & \text{for } x \in \Omega_1, \\ 1, & \text{for } x \in \Omega_2, \end{cases} \tag{2.39}$$

we introduce basis functions

$$\boxed{\varphi_i^\Gamma(x) := \varphi_i(x)(H_\Gamma(x) - H_\Gamma(x_i)), \text{ for } i \in \mathscr{I}_\Gamma,} \tag{2.40}$$

and then an NXFEM space is defined as

$$\boxed{V_A^h := V_h \oplus \text{span}\{\varphi_i^\Gamma | i \in \mathscr{I}_\Gamma\}.} \tag{2.41}$$
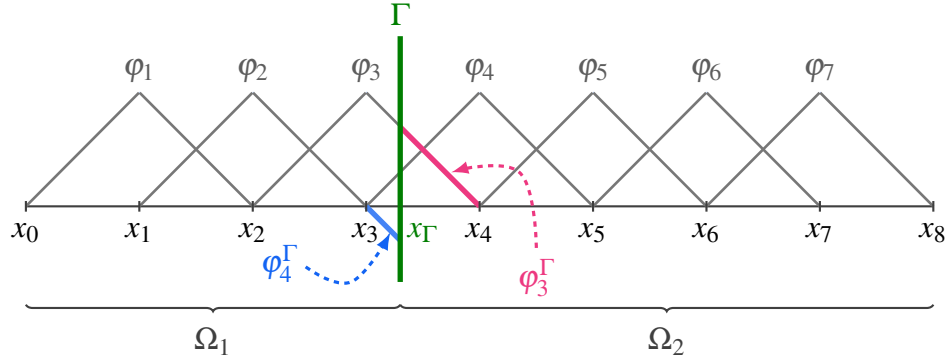
Figure 2.10. 1D example of basis proposed by Reusken.
$V_A^h = \text{span}\{\varphi_i\}_{i=1,7} \oplus \text{span}\{\varphi_3^\Gamma, \varphi_4^\Gamma\}$.

An illustration in 1D of basis functions described in (2.40) is given in Figure 2.10.

Remark 2.12 The choice of Reusken uses all standard basis functions and just add more enrichment functions defined on basis ones whose support cut by the interface. There is also little difference between the two choices given in Figure 2.10 and Figure 2.8.

Remark 2.13 The basis functions defined by Hansbo& Hansbo spaned the same finite element space as the ones defined by Reusken.

Indeed, recall Heaviside function H_Γ defined in (2.39), for $i \in \mathscr{I}_\Gamma$, Hansbo's basis functions (2.36) can be rewritten as follows

$$\begin{cases} \varphi_i^{(1)}(x) & = \varphi_i(x)H_\Gamma(x), \\ \varphi_i^{(2)}(x) & = \varphi_i(x)(1 - H_\Gamma(x)). \end{cases}$$

Using notations $H_\Gamma, H_\Gamma^{(i)}$ for $H_\Gamma(x)$ and $H_\Gamma(x_i)$ respectively, we take $u \in V_A^h$ to get

$$
\begin{aligned}
u &= \sum_{i \in \mathscr{I} \setminus \mathscr{I}_\Gamma} u_i \varphi_i + \sum_{i \in \mathscr{I}_\Gamma} u_i^{(1)} \varphi_i^{(1)} + \sum_{i \in \mathscr{I}_\Gamma} u_i^{(2)} \varphi_i^{(2)} \\
&= \sum_{i \in \mathscr{I} \setminus \mathscr{I}_\Gamma} u_i \varphi_i + \sum_{i \in \mathscr{I}_\Gamma} u_i^{(1)} \varphi_i H_\Gamma + \sum_{i \in \mathscr{I}_\Gamma} u_i^{(2)} \varphi_i (H_\Gamma - H_\Gamma^{(i)}) \\
&= \sum_{i \in \mathscr{I} \setminus \mathscr{I}_\Gamma} u_i \varphi_i + \sum_{i \in \mathscr{I}} (u_i^{(2)} + (u_i^{(1)} - u_i^{(2)}) H_\Gamma^{(i)}) \varphi_i + \sum_{i \in \mathscr{I}} (u_i^{(1)} - u_i^{(2)}) \varphi_i (H_\Gamma - H_\Gamma^{(i)}) \\
&= \sum_{i \in \mathscr{I} \setminus \mathscr{I}_\Gamma} u_i \varphi_i + \sum_{i \in \mathscr{I}} (u_i^{(2)} + (u_i^{(1)} - u_i^{(2)}) H_\Gamma^{(i)}) \varphi_i + \sum_{i \in \mathscr{I}} (u_i^{(1)} - u_i^{(2)}) \varphi_i^\Gamma \\
&= \sum_{i \in \mathscr{I}} u_i \varphi_i + \sum_{i \in \mathscr{I}} u_i^\Gamma \varphi_i^\Gamma,
\end{aligned}
$$

in which, $u_i = u_i$ for $i \in \mathscr{I} \setminus \mathscr{I}_\Gamma$, $u_i = u_i^{(2)} + (u_i^{(1)} - u_i^{(2)}) H_\Gamma^{(i)}$ for $i \in \mathscr{I}_\Gamma$ and $u_i^\Gamma = u_i^{(1)} - u_i^{(2)}$ for $i \in \mathscr{I}_\Gamma$. Therefore, $u \in V_A^h$. Thus $v \in V_H^h$.

Conversely, if $u \in V_A^h$, we have

$$
\begin{aligned}
u &= \sum_{i \in \mathscr{I}} u_i \varphi_i + \sum_{i \in \mathscr{I}} u_i^\Gamma \varphi_i^\Gamma \\
&= \sum_{i \in \mathscr{I}} u_i \varphi_i + \sum_{i \in \mathscr{I}} u_i^\Gamma \varphi_i (H_\Gamma - H_\Gamma^{(i)}) \\
&= \sum_{i \in \mathscr{I} \setminus \mathscr{I}_\Gamma} u_i \varphi_i + \sum_{i \in \mathscr{I}} (u_i^\Gamma + u_i - u_i^\Gamma H_\Gamma^{(i)}) \varphi_i H_\Gamma + \sum_{i \in \mathscr{I}} (u_i - u_i^\Gamma H_\Gamma^{(i)}) \varphi_i (1 - H_\Gamma) \\
&= \sum_{i \in \mathscr{I} \setminus \mathscr{I}_\Gamma} u_i \varphi_i + \sum_{i \in \mathscr{I}} u_i^{(1)} \varphi_i^{(1)} + \sum_{i \in \mathscr{I}} u_i^{(2)} \varphi_i^{(2)},
\end{aligned}
$$

where $u_i^{(1)} = u_i^\Gamma + u_i - u_i^\Gamma H_\Gamma^{(i)}$ and $u_i^{(2)} = u_i - u_i^\Gamma H_\Gamma^{(i)}$ for $i \in \mathscr{I}_\Gamma$.
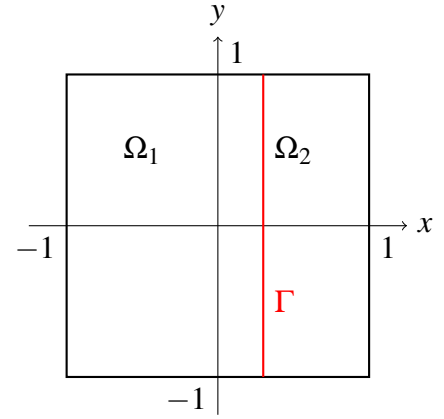
2.6 Numerical test cases

In this section, we present some validation test cases introduced to illustrate the method.

2.6.1 Barrau's test case

The model of the first test is given in [5]. We consider a domain $\Omega = [-1, 1] \times [-1, 1]$ where the interface Γ is a straight line $\{x_0\} \times [-1, 1]$. We will use NXFEM method to find a solution of (2.43),

$$\begin{cases} -\nabla \cdot (k\nabla u) = f & \text{in } \Omega, \\ [\![u]\!] = [\![k\nabla u]\!] = 0 & \text{on } \Gamma, \\ u = u_{\text{ex}} & \text{on } \partial\Omega. \end{cases} \qquad (2.43)$$

The exact solution is already known as,

$$u_{\text{ex}} = \begin{cases} \dfrac{x^2}{k_1} & (x \le x_0), \\ \dfrac{x^2 - x_0^2}{k_2} + \dfrac{x_0^2}{k_1} & (x > x_0), \end{cases}$$

where $k_i = k|_{\Omega_i}$ is a constant in Ω_i. With this u_{ex}, the RHS f is fixed at the value of -4. The exact solution in this case depends much on the values of diffusion coefficients. We will apply a very different k_1, k_2 to check the method ($k_1 = 1, k_2 = 100$).

With the `NXFEM toolbox` which is described in Chapter 3, we can find a solution u_h of (2.43) as in Figure 2.11.

a. Exact solution u.

b. Numerical solution u_h.
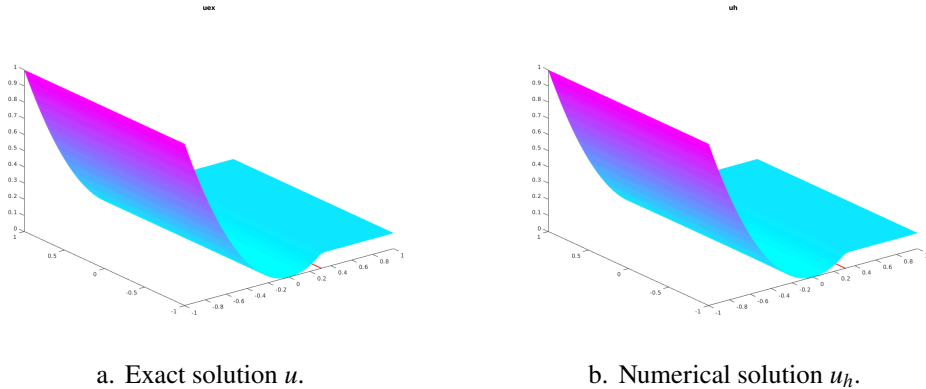
Figure 2.11. Exact soltion and numerical solution in the Barrau's test case with a fine mesh.

The corresponding L^2 and $\|\!|\cdot|\!\|_H$ norm and convergence rate are given in Table 2.1 and Figure 2.12. This is consistent with the result in Theorem 2.2.

2.6.2 Sinha's test case

A similar test is given in [54]. In this test, we consider $\Omega = [0,2] \times [0,1]$ with the interface is $\{1\} \times [0,2]$. The exact solution and the right hand side f are given in (2.44).

h	$\|u_h - u_{\mathrm{ex}}\|_{L^2}$	order	$\|\|u_h - u_{\mathrm{ex}}\|\|_H$	order
9.52×10^{-2}	7.3×10^{-3}		0.51	
4.88×10^{-2}	1.7×10^{-3}	1.82	0.32	0.7
2.47×10^{-2}	3.91×10^{-4}	2.01	0.17	0.9
1.24×10^{-2}	9.97×10^{-5}	1.93	0.08	1.1

Table 2.1. $L^2, \|\cdot\|_H$ norm errors of the solutions with different mesh sizes in Barrau's test case.
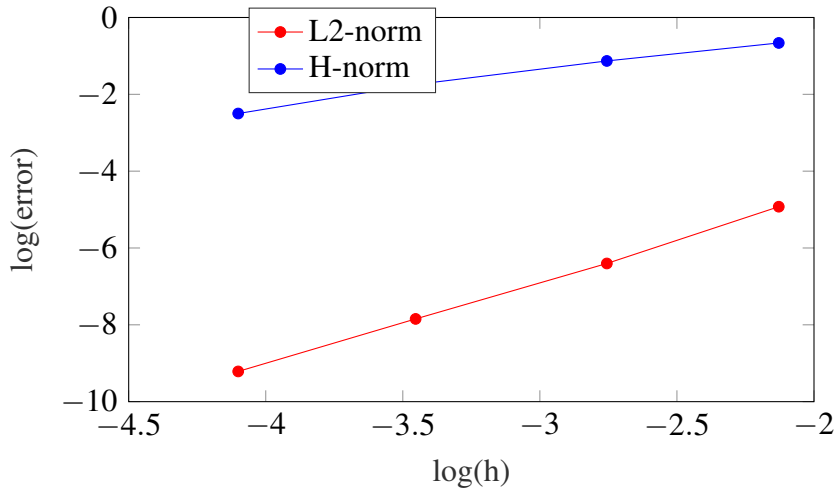
Figure 2.12. The convergence in $L^2, \|\cdot\|_H$ norms of the solution in Barrau's test case using NXFEM method.

$$u_{\mathrm{ex}} = \begin{cases} \sin(\pi x)\sin(\pi y) \text{ in } \Omega_1, \\ -\sin(2\pi x)\sin(\pi y) \text{ in } \Omega_2. \end{cases} \qquad f = \begin{cases} 2k_1\pi^2\sin(\pi x)\sin(\pi y), \\ -5k_2\pi^2\sin(2\pi x)\sin(\pi y). \end{cases} \qquad (2.44)$$

Remark 2.14 Different from Barrau's test case, the interface in this test is fixed at $x = 1$, the diffusion coefficients must follow a rule that $k_1 = 2k_2$. The solution in this test doesn't depend on the diffusion coefficients like the one in the Barrau's test. In contrast, these coefficients change the RHS f.

Using NXFEM toolbox, we can obtain an approximated solution u_h of (2.44) as in Figure 2.13

The same results as in Barrau's test for the errors and convergence rates in $L^2, \|\cdot\|_H$ norms are also obtained. We don't present them here. The reason we consider this test is to check the effect of the Ghost Penalty term (GP) in the next section.

a. Exact solution u.   b. Numerical solution u_h.

Figure 2.13. Exact solution and numerical solution in the Sinha's test case with a fine mesh.

2.6.3 The choice of parameters

As mentioned in Section 2.3, the choice of parameters is very important. With two previous test cases, we give some interesting results about the choice of values for:

- The penalty coefficient λ and the value of $\hat{\lambda}$ which are given in Section 2.3.
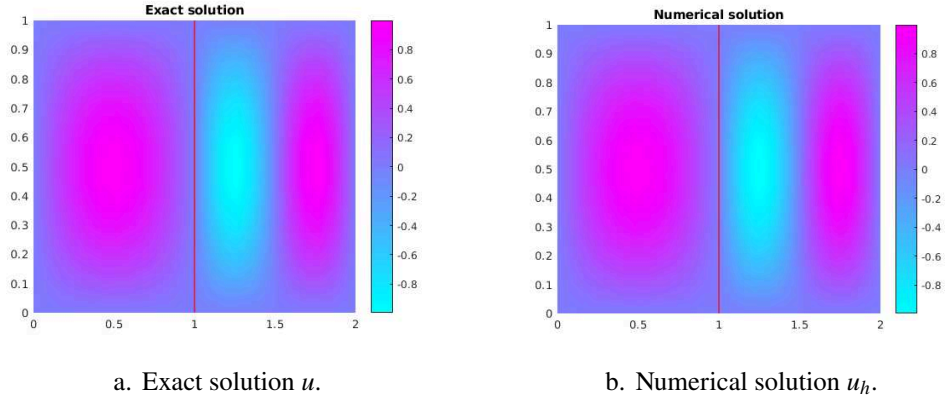- The values of γ_1, γ_2 when we work with the Ghost Penalty mentioned in Section 2.4.
- The choice of using or not using Ghost Penalty.
- The *very contrast* problem where the diffusion coefficients take very different values in each subdomain Ω_i.

The results we showed in the test case of Barrau's and Sinha's are satisfied conditions:

- λ is chosen like a form of (2.29) in which $\hat{\lambda}$ is around 10^5.
- We use the ghost penalty with γ_1, γ_2 not being too big for both of them (around 10^{-3}).

Remark 2.15 However, when we change a little in the above choices, we get some interesting things. Some of them are not mentioned in any work before. All of below results are evaluated under the view of convergence rate in L^2 norm. They will be good if the rate is approximately 2 and bad otherwise.

1. We cannot choose either large $\hat{\lambda}$ or small it, both will give a bad result in the convergence rate (*CR*). When we take $\hat{\lambda} = 10^{10}$, $CR = 0.9$ and $CR = 1.53$ for the case $\hat{\lambda} = 1$.
2. The ghost penalty term seems to be useless in the Barrau's test case. We can choose either small or 0 of $\hat{\lambda}$ to get the same result. However, if we take very big γ_1, γ_2, GP will affect badly the final result.
3. In contrast, GP only shows it power in the Sinha's test case. If we don't use GP, $CR = 1.6$ while $CR = 2.1$ when GP comes.
4. There is not any problem if $\frac{k_1}{k_2} < 10^3$, we get $CR = 1.97$ for those cases. However, when we take $k_1 = 1, k_2 = 10^4$, CR downs to 1.47. This is for

the case we use λ like in (2.26). Only when we apply the Barrau's form (2.27), *CR* gets back to the stability 1.93.

3. Implementation NXFEM with Matlab

Contents

In this chapter, I will give in details about algorithms and guide to use `NXFEM toolbox` developed by myself. I build it based on the idea of space proposed by Hansbo in Section 2.5, coming from the idea of implementing Standard FEM in Matlab. In other point of view, this chapter is also used to implement NXFEM in other programming language instead of being used only in Matlab.

For convenience, I put all technical description in this chapter including implementation of decoupling system, level set method, biofilm models which are given later in following chapters.

To access the toolbox, please email to dinhanhthimail@gmail.com.

> **Remark 3.1** Notice that, if I mention the name of a certain function, it also alludes to the file with the same name. For instance, function `getTriangles` is

corresponding to a file named *getTriangles.m* in the toolbox.

Without loss of generality, I sometimes use x in replace of (x, y) in $\Omega \subset \mathbb{R}^2$.

3.1 Model

This section recalls a general interface model and the discrete forms of the problem which are given specifically in Section 2.2. We need them to describe techniques to be used in this chapter.

Consider a problem (3.1) on a bounded domain $\Omega \subset \mathbb{R}^2$ with a convex polynomial boundary $\partial\Omega$ and a smooth interface Γ dividing Ω into two open sets Ω_1, Ω_2.

$$\begin{cases} -\nabla \cdot (k\nabla u) = f & \text{on } \Omega_1 \cup \Omega_2, \\ [\![u]\!] = 0 & \text{on } \Gamma, \\ [\![k\nabla_{\mathbf{n}} u]\!] = 0 & \text{on } \Gamma, \\ u = 0 & \text{on } \partial\Omega. \end{cases} \tag{3.1}$$

A weak form of (3.1) is given in (2.7), a discrete form which is recalled from (2.19): Seek $u_h \in V_h^\Gamma$ such that,

$$a_h(u_h, v_h) = L_h(v_h), \ \forall v_h \in V_h^\Gamma,$$

where

$$a_h(u_h, v_h) = \langle k\nabla u_h, \nabla v_h \rangle_{\Omega_{12}}$$
$$- \langle \{\!\{k\nabla_{\mathbf{n}} u_h\}\!\}, [\![v_h]\!] \rangle_\Gamma - \langle [\![u_h]\!], \{\!\{k\nabla_{\mathbf{n}} v_h\}\!\} \rangle_\Gamma + \lambda \langle [\![u_h]\!], [\![v_h]\!] \rangle_\Gamma, \tag{3.2}$$
$$L_h(v_h) = \langle f, v \rangle_\Omega.$$

Remark 3.2 In comparison with the problem (2.6), here we consider $g = 0$, *i.e.*, $[\![\nabla_{\mathbf{n}} u]\!] = 0$.

3.2 Quadrature

Principle. In order to find an approximation of an integral, we need to use quadrature rules. For more information about the theory and the implementation of the quadrature, I suggest referring to [65, Chapter 8]. In Section A.1, I recall some of its principal properties. Understanding this well will help us much on building a toolbox for the NXFEM method. Most of the idea in the implementation has relations to the quadrature's rules.

3.3 Getting information about the mesh and some principle components

3.3.1 Mesh generation

Thanks to *Matlab PDE Toolbox*, we can generate a triangle-mesh with an important triplet `[p,e,t]` containing three factors: coordinate of *points* `p` (nodes), edges `e` on the boundary of the mesh and the triangle's information `t`. There are two choices of the generating: an irregular mesh or a regular mesh (Figure 3.1).

```
% irregular mesh
[points,edges,triangles] = initmesh(GeoDom,'hmax',hEdgeMax);
% regular mesh
[points,edges,triangles] = poimesh(GeoDom,numSeg,numSeg);
```


a. Irregular mesh.   b. Regular mesh.
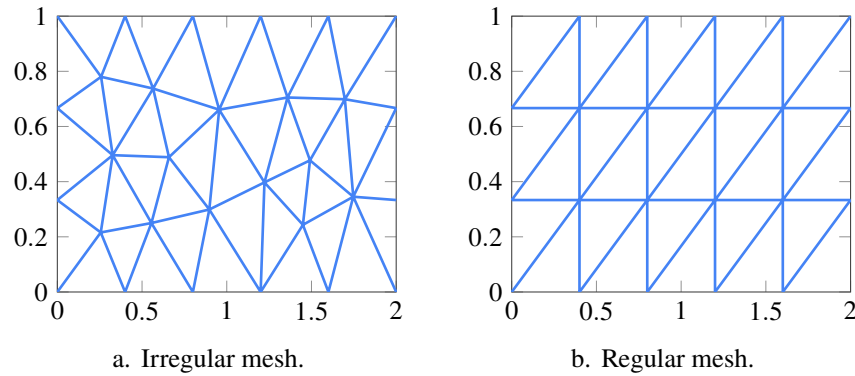
Figure 3.1. Meshes generated by *Matlab PDE Toolbox*.

> **Remark 3.3** `hEdgeMax` in the code of generating the mesh is the maximum length of segments on the boundary of the mesh. It's not the maximum length of all mesh edges. `numSeg` is the number of segments on each side of the rectangle mesh.

> **Remark 3.4** The vertices of each triangle are stored *counter-clockwise* in `t`. For instance, the triangle `1-2-3` in Figure 3.3 has 3 vertices stored in `t` as $[1, 2, 3]$ or $[2, 3, 1]$ or $[3, 1, 2]$. An example of triangles and their presentation in `t` is given in Table A.5.

A connectivity among components in `[p,e,t]` is given in Section A.2.

Figure 3.2. Level set function ϕ corresponding to an approximated interface Γ_h in the computation domain Ω.
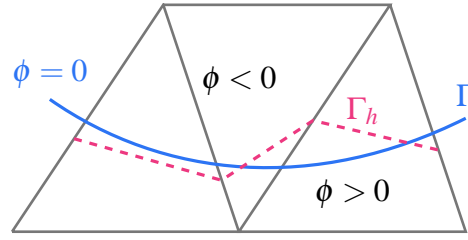
3.3.2 Describe the interface

An approximation Γ_h of the interface Γ is determined via a so-called *level set function* ϕ (cf. Figure 3.2). Definition of a level set function is given in Section 5.1. For this section, ϕ is considered as a signed distance function on Ω which measures the distance between a point in the domain and the interface. In the implementation, we determine the interface based on the values of ϕ at nodes.

Notation 3.1 Corresponding to the domain Ω given in Section 3.1, we assume that

$$
\begin{aligned}
\Omega_1 &= \{x \in \Omega : \phi(x) < 0\}, \\
\Omega_2 &= \{x \in \Omega : \phi(x) > 0\}, \\
\Gamma &= \{x \in \Omega : \phi(x) = 0\}.
\end{aligned}
$$
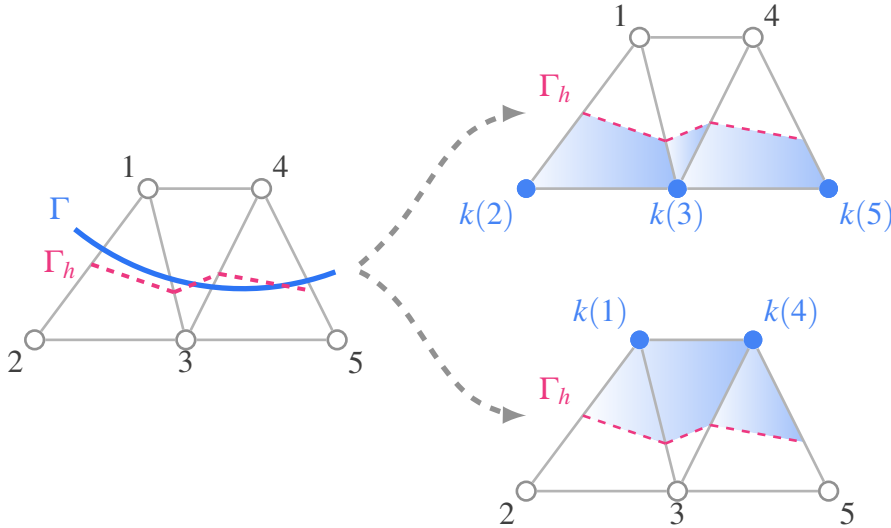
Figure 3.3. An idea of doubling nodes around the interface. We keep the numbering of node i where the support of a basis function locating on it belongs to Ω_1 and we renumber it to $k(i)$ for otherwise (blue).

Notation 3.1 is useful to be used in the numbering of *additional nodes* in the NXFEM method. We will build new Hansbo-type basis functions described in Section 2.5. Let us consider all basis functions whose support are cut by the interface. We keep the indexes of ones whose support lies in Ω_1 and define new indexes of ones whose support lies in Ω_2 (cf. Figure 3.3). Other words, the basis defined in (2.36) will be

numbered in the code like (3.3). The subscript indicates the index of node in the mesh.

$$\varphi_i := \varphi_i|_{\Omega_1}, \quad \varphi_{k(i)} := \varphi_i|_{\Omega_2}. \tag{3.3}$$

In order to get $k(i)$ nodes, we use the function `getNewNodes` in the toolbox. The numbering of these new nodes will be continued after the number of nodes of the mesh.

3.3.3 Get triangles

As mentioned in (A.6), we need to work on the triangles of the mesh $\mathcal{T}_h$. Because of special basis functions locating at nodes around the interface, we have to distinguish the triangles based on their position regarding to the position of the interface (Figure 3.4). For each triangle, we consider the value of level set function ϕ (`phi`) at the vertices of this triangle ($\phi_i = \phi(\mathrm{x}_i)$) and subject to the type of triangle, we will use different choices. In the toolbox, we use the function `getTriangles` to find such triangles.
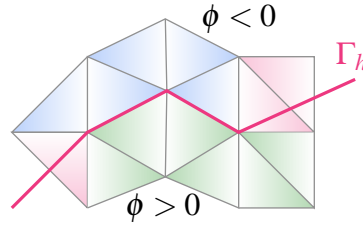
Figure 3.4. Three types of triangles are classified: `tris.NCTs1` (blue), `tris.NCTs2` (green), `tris.CTs` (red).

Not-cut triangles `tris.NCTsi` contains triangles locating totally inside Ω_i ($i = 1, 2$). They have no common part with the interface, `tris.CTs` $= \{K \in \mathcal{T}_h : K \cap \Gamma \neq \emptyset\}$. If a triangle on which there are $\max_i \phi_i < 0$ for all $i \in \{1, 2, 3\}$, it is in `NCTs1` because its all three vertices are located in Ω_1. Conversely, if $\min_i \phi_i > 0$, it is in `NCTs2` because three vertices are located in Ω_2.

Cut triangles `tris.CTs` contains triangles which are cut by the interface (their union is denoted as G_h), `tris.CTs` $= \{K \in \mathcal{T}_h : K \cap \Gamma \neq \emptyset\}$. If a triangle on which, there are $\max_i \phi_i \times \min_i \phi_i < 0$ for all $i \in \{1, 2, 3\}$, it is in `tris.CTs` because it has two vertices locating on two sides of the interface.

If a triangle $K \in \mathcal{T}_h$ has three vertices who has coordinates $(x_i, y_i), i \in \{1, 2, 3\}$, its area can be calculated as in (3.4). This formula will be used in some functions in the toolbox.

$$|K| = \frac{1}{2} \det \begin{bmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{bmatrix}. \tag{3.4}$$

3.3.4 Cut triangles

The main difference of NXFEM in comparison with FEM is when we want to implement it on cut triangles (cf. Remark 2.1 to clarify what a cut triangle is). This leads us to find all necessary information of this type of triangles. This work is done with the help of the function `getInfoCTs` in the toolbox.

Type of cut triangles. The interface may cut a triangle in 3 different ways. It divides the triangle into two different parts in shapes - a triangular shape and a quadrilateral shape or two triangular shapes (Figure 3.5). Because we want to use quadrature rules *on a triangle* to find the integrals in 2D (cf. (A.4)), we prefer to work *only on the triangle part* of a cut triangle. That's why we need to know exactly which kind of triangle we are working on. Thanks to `CT.type`, the information about the type of all cut triangles is stored in this variable.

a. Type 2: two vertices are located in Ω_1, one vertex is located in Ω_2.

b. Type 0: one vertex is located on interface.

c. Type 4: two vertices are located in Ω_2, one vertex is located in Ω_1.
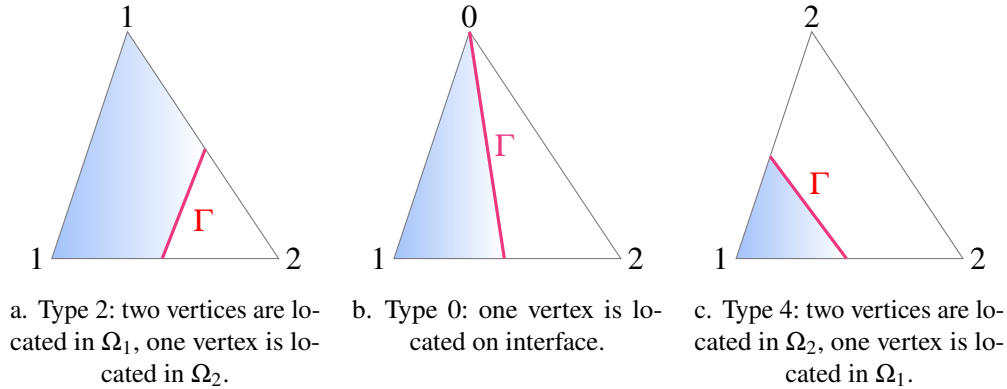
Figure 3.5. Three types of a cut triangle: a vertex is numbered "1" if it is in Ω_1, "2" in Ω_2 and "0" on Γ_h. The blue area is in Ω_1.

If vertices are located in Ω_1, we denote them as "1"; "2" for ones locating in Ω_2 and "0" if they are located on the interface (Figure 3.5). Based on these numbers, I classify all cut triangles into three types: 0, 2 and 4 (These numbers are the product of all three numbers on three vertices.).

Intersection points. Look back to the bilinear form a_h in (3.2), there are some terms computed on the interface. In the manner of quadrature point of view, we use the formula (A.2) to find approximate values of these forms. For this reason, we need to know the intersection between the interface and the edges of cut triangles.

On each triangle so-called `i-j-k` (notation of three vertices in this orientation, cf. Remark 3.4), we recall the value of ϕ on each vertex as ϕ_i, ϕ_j, ϕ_k which are

corresponding to coordinates of three vertices. The function `getiPs`[1] which follows Algorithm 3.1 results all necessary intersections whereas the function `getiPsonE`[2] which follows[3] (3.5) helps us find intersection (x_0, y_0) between the interface and the edge `i-k`.

$$
\begin{aligned}
x_0 &= \frac{x_i}{2}\left(1 - \frac{\phi_i + \phi_k}{\phi_i - \phi_k}\right) + \frac{x_k}{2}\left(1 + \frac{\phi_i + \phi_k}{\phi_i - \phi_k}\right) \\
y_0 &= \frac{y_i}{2}\left(1 - \frac{\phi_i + \phi_k}{\phi_i - \phi_k}\right) + \frac{y_k}{2}\left(1 + \frac{\phi_i + \phi_k}{\phi_i - \phi_k}\right)
\end{aligned}
\tag{3.5}
$$

We store intersections in `CT.iPs`. The way we store them is also very important because the order of two intersections in `iPs` will be used to find the unit normal vector on each segment of the interface (cf. the function `getUNCT`, Section 3.3.5).

- If the type of a cut triangle is 2 or 4 (the interface doesn't pass any triangle's vertex), the order of two intersections depends on the order of vertices of the triangle ($i \to j \to k$).
- If the type of a cut triangle is 0 (the interface passes one vertex), the order of two intersections depends on which vertex is locating on the interface. If that vertex is `i` or `k`, the order is `[intersection,vertex]`. Otherwise, the order is `[vertex,intersection]`.

Algorithm 3.1. Determine intersection points on a cut triangle (`getiPs`).

Input : ϕ_i for $i = 1, 3$ at three vertices x_i of cut triangle.
Output `CT.iPs` of size 2 coordinates $\times 2$ intersections $\times N_{CTs}$.
:

for $1 \le \tau \le N_{CTs}$ **do**
    $r = 0$;
    if $\phi_1 \times \phi_2 < 0$ **then**
        `CT.iPs`$(:, r, \tau) = $ `getiPsonE`$(x_1, x_2, \phi_1, \phi_2)$;
        $r = r + 1$;
    if $\phi_2 \times \phi_3 \le 0$ **then**
        `CT.iPs`$(:, r, \tau) = $ `getiPsonE`$(x_2, x_3, \phi_2, \phi_3)$;
        $r = r + 1$;
    if $\phi_3 \times \phi_1 \le 0$ and $r < 2$ **then**
        `CT.iPs`$(:, r, \tau) = $ `getiPsonE`$(x_3, x_1, \phi_3, \phi_1)$;
        $r = r + 1$;

An example describing the idea of Algorithm 3.1 together with Algorithm 3.2 is

---

[1] `getiPs` means "get intersection points"
[2] `getiPsonE` means "get intersection points on edge"
[3] cf. a proof in Section A.3

given in Appendix A.5.

3.3.5 Unit normal vector

A unit normal vector *on the interface* Γ_h is defined from Ω_1 to Ω_2 and it can be found by the function `getUNCT`. From the intersection points obtained in `getiPs`, they are ordered in a such special way, e.g. `[A,B]`, the algorithm to find a such unit normal vector is given in Algorithm 3.2.

Algorithm 3.2. Determine a unit normal vector on the interface Γ_h (`getUNCT`).

Input : ϕ_i for $i = 1, 3$ at three vertices x_i of cut triangle;
  Intersection points `CT.iPs`;
  Type of cut triangles `CT.type`.
Output `CT.uN` of size 2 coordinates $\times N_{CTs}$.
:

for $1 \leq \tau \leq N_{CTs}$ **do**
  $A = $ `CT.iPs`$(:, 1, \tau)$;
  $B = $ `CT.iPs`$(:, 2, \tau)$;
  if `CT.type` $== 0$ **then**
    if $\phi_1 < 0$ or $\phi_3 > 0$ **then**
      `CT.uN`$(:, \tau) = $ `getUnitNV`(B, A);
    else
      `CT.uN`$(:, \tau) = $ `getUnitNV`(A, B);
  else if $\phi_1 < 0$ **then**
    `CT.uN`$(:, \tau) = $ `getUnitNV`(B, A);
  else
    `CT.uN`$(:, \tau) = $ `getUnitNV`(A, B);

When we have a segment whose two endpoints is in order $\overrightarrow{AB}$, we will use the function `getUnitNV` to find its normal vector. A normal vector of AB always points to *the left hand side* when we go from A to B.

An example describing the idea of Algorithm 3.1 together with Algorithm 3.2 is given in Appendix A.5.

3.3.6 Other components

Consider a triangle $K \in \mathscr{T}_h$ with three vertices i, j, k. φ_i is a shape function located at node i. The gradient of the basis function $\nabla \varphi_i$ can be found by

$$\nabla \varphi_i(x, y) = \frac{1}{2|K|} \mathbf{n}_i = \frac{1}{2|K|} \begin{bmatrix} y_j - y_k \\ x_k - x_j \end{bmatrix}, \tag{3.6}$$

in which $|K|$ denotes the area of K. The function `getGradPhi` gives us the gradients of three shape functions regarding three vertices of each triangle in a group of input triangles.

3.4 Assembling

3.4.1 Assembling of the stiffness matrix

In this section, I present a way to construct a stiffness matrix which corresponds to the bilinear form a_h in (3.2). The key idea is to construct a triplet `[i,j,k]` which describes a *sparse matrix A*. Two components `[i,j]` give the numbering of a column and a row of A respectively in which the value `v` of A at this position is not zero.

We observe that if $u_h \in V_h^\Gamma = \mathrm{span}\{\varphi_i\}_{i=1,\ldots,N_{\mathrm{new}}}$ satisfies (3.2),

$$a_h(u_h, \varphi_j) = L_h(\varphi_j), \quad j = 1, \ldots, N_{\mathrm{new}}, \tag{3.7}$$

where N_{new} is the number of degrees of freedom in V_h^Γ, *i.e.*, $N_{\mathrm{new}} = N_{V_h} + N_{\mathscr{I}}$ in which N_{V_h} is the number of standard basis functions, $N_{\mathscr{I}}$ is the number of "new" basis ones. Because $u_h = \Sigma_{i=1}^{N_{\mathrm{new}}} u_i \varphi_i$, we can write (3.7) as

$$\sum_{i=1}^{N_{\mathrm{new}}} u_i a_h(\varphi_i, \varphi_j) = L_h(\varphi_j), \quad j = 1, \ldots, N_{\mathrm{new}},$$

which can also be written as

$$AU = F, \tag{3.8}$$

where $A_{ji} = a_h(\varphi_i, \varphi_j), F_j = L_h(\varphi_j)$ for $i, j = 1, \ldots, N_{\mathrm{new}}$. More specifically,

$$A_{ji} = \sum_{K \in \mathscr{T}_h} a_K^\Omega(\varphi_i, \varphi_j) + \sum_{K \in \mathscr{T}_h} a_K^{\Gamma_G}(\varphi_i, \varphi_j) + \sum_{K \in \mathscr{T}_h} a_K^{\Gamma_P}(\varphi_i, \varphi_j), \tag{3.9}$$

where

$$a_K^\Omega(\varphi_i, \varphi_j) = \int_K \nabla \varphi_i \cdot \nabla \varphi_j \, \mathrm{dx},$$
$$a_K^{\Gamma_G}(\varphi_i, \varphi_j) = -\langle \{\!\{k\nabla_{\mathbf{n}}\varphi_i\}\!\}, [\![\varphi_j]\!]\rangle_{\Gamma_K} - \langle [\![\varphi_i]\!], \{\!\{k\nabla_{\mathbf{n}}\varphi_j\}\!\}\rangle_{\Gamma_K},$$
$$a_K^{\Gamma_P}(\varphi_i, \varphi_j) = \lambda \langle [\![\varphi_i]\!], [\![\varphi_j]\!]\rangle_{\Gamma_K}.$$

We see that, each term in (3.9) can be computed on two types of triangles, non-cut ones and cut ones. It's described in Figure 3.6.
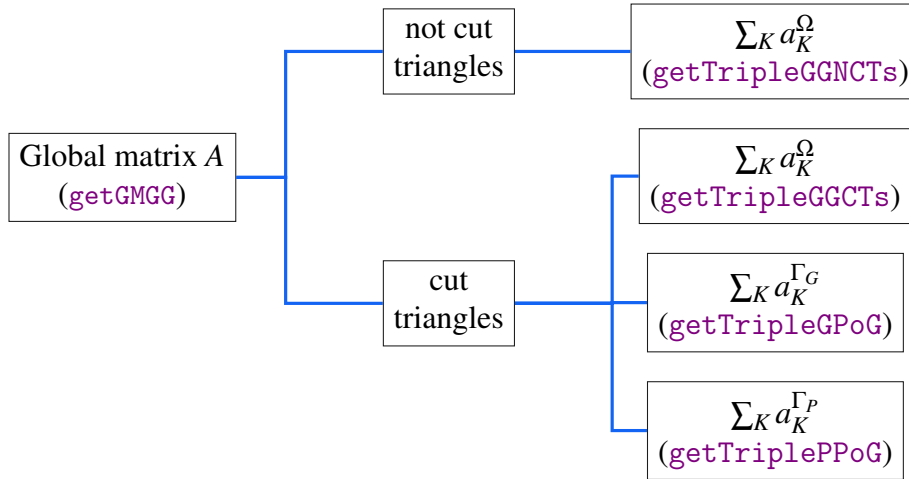
Figure 3.6. The idea of assembling the global stiffness matrix A.

On not-cut triangles. Because not-cut triangles are not cut by the interface, we don't need to compute a_K^Γ on these such triangles. That's why there is no terms a_K^Γ on not-cut triangles.

The term a_K^Ω contains only gradients in $\mathbb{P}^1$ which are constants, we don't need to use quadratures in this case. If K which is a not-cut triangle has three vertices i, j, k,

$$
a_K^\Omega(\varphi_i, \varphi_j) = \int_K \nabla \varphi_i \cdot \nabla \varphi_j \, dx = \frac{1}{4|K|} (y_k - y_j, x_j - x_k) \begin{bmatrix} y_k - y_j \\ x_j - x_k \end{bmatrix},
$$

where $\nabla \varphi_i$ can be computed by (3.6) and $|K|$ can be computed by (3.4).

This process is corresponding with the function `getTripleGGNCTs`[4] in the toolbox.

On cut triangles. Different from the case of not-cut triangles, the basis functions of V_h^Γ locating on cut triangles are divided into two types, namely ones restricted on Ω_1 and ones restricted on Ω_2. It is already described in Section 3.3.2 and (2.36).

For the term a_K^Ω where K is a cut triangle which has three vertices i, j, k, we have

$$
a_K^\Omega(\varphi_i^{(l)}, \varphi_j^{(l)}) = \int_{K_l} \nabla \varphi_i^{(l)} \cdot \nabla \varphi_j^{(l)} \, dx = \frac{|K_l|}{|K|^2} (y_k - y_j, x_j - x_k) \begin{bmatrix} y_k - y_j \\ x_j - x_k \end{bmatrix},
$$

where $K_l = K \cap \Omega_l$ for $l = 1, 2$. Note that, we don't have the case of $a_K^\Omega(\varphi_i^{(l)}, \varphi_j^{(r)})$ where $l \neq r$ because the intersection of these basis' supports is empty. This can be found in the toolbox by the function `getTripleGGCTs`[5].

---

[4] `getTripleGGNCTs` means "get triple gradient-gradient on not-cut triangles".
[5] `getTripleGGCTs` means "get triple gradient-gradient on cut triangles".

For terms $a_K^{\Gamma_G}, a_K^{\Gamma_P}$, we have to consider four cases which depend on the subdomain the basis functions located on. First,

$$
\begin{aligned}
a_K^{\Gamma_G}(\varphi_i^{(1)}, \varphi_j^{(1)}) &= -\langle \{\!\{k\nabla_{\mathbf{n}}\varphi_i^{(1)}\}\!\}, [\![\varphi_j^{(1)}]\!]\rangle_{\Gamma_K} - \langle [\![\varphi_i^{(1)}]\!], \{\!\{k\nabla_{\mathbf{n}}\varphi_j^{(1)}\}\!\}\rangle_{\Gamma_K} \\
&= -\langle \kappa_1 k_1 \nabla_{\mathbf{n}}\varphi_i, \varphi_j\rangle_{\Gamma_K} - \langle \varphi_i, \kappa_1 k_1 \nabla_{\mathbf{n}}\varphi_i\rangle_{\Gamma_K}, \\
a_K^{\Gamma_P}(\varphi_i^{(1)}, \varphi_j^{(1)}) &= \lambda \langle [\![\varphi_i^{(1)}]\!], [\![\varphi_j^{(1)}]\!]\rangle_{\Gamma_K} \\
&= \lambda \langle \varphi_i, \varphi_j\rangle_{\Gamma_K},
\end{aligned}
\tag{3.10}
$$

where we have applied Definition 2.1 for $[\![\cdot]\!], \{\!\{\cdot\}\!\}$ and a restriction of basis functions on interface (2.38). Similarly,

$$
\begin{aligned}
a_K^{\Gamma_G}(\varphi_i^{(1)}, \varphi_j^{(2)}) &= \langle \kappa_1 k_1 \nabla_{\mathbf{n}}\varphi_i, \varphi_j\rangle_{\Gamma_K} - \langle \varphi_i, \kappa_2 k_2 \nabla_{\mathbf{n}}\varphi_j\rangle_{\Gamma_K}, \\
a_K^{\Gamma_G}(\varphi_i^{(2)}, \varphi_j^{(1)}) &= -\langle \kappa_2 k_2 \nabla_{\mathbf{n}}\varphi_i, \varphi_j\rangle_{\Gamma_K} + \langle \varphi_i, \kappa_1 k_1 \nabla_{\mathbf{n}}\varphi_j\rangle_{\Gamma_K}, \\
a_K^{\Gamma_G}(\varphi_i^{(2)}, \varphi_j^{(2)}) &= \langle \kappa_2 k_2 \nabla_{\mathbf{n}}\varphi_i, \varphi_j\rangle_{\Gamma_K} + \langle \varphi_i, \kappa_2 k_2 \nabla_{\mathbf{n}}\varphi_i\rangle_{\Gamma_K}, \\
a_K^{\Gamma_P}(\varphi_i^{(1)}, \varphi_j^{(2)}) &= -\lambda \langle \varphi_i, \varphi_j\rangle_{\Gamma_K}, \\
a_K^{\Gamma_P}(\varphi_i^{(2)}, \varphi_j^{(1)}) &= -\lambda \langle \varphi_i, \varphi_j\rangle_{\Gamma_K}, \\
a_K^{\Gamma_P}(\varphi_i^{(2)}, \varphi_j^{(2)}) &= \lambda \langle \varphi_i, \varphi_j\rangle_{\Gamma_K}.
\end{aligned}
\tag{3.11}
$$

We see that, in (3.10) and (3.11), there are only two principle terms $\langle \nabla_{\mathbf{n}}\varphi_i, \varphi_j\rangle_{\Gamma_K}$ (together with its symmetry) and $\langle \varphi_i, \varphi_j\rangle_{\Gamma_K}$. They are corresponding to $a_K^{\Gamma_G}$ and $a_K^{\Gamma_P}$ respectively. I name these functions based on their form, *i.e.*, "`GPoG`" means "gradient and phi on Gamma" and "`PPoG`" means "phi and phi on Gamma".

Term $\langle \nabla_{\mathbf{n}}\varphi_i, \varphi_j\rangle_{\Gamma_K}$ is computed by the function `intGradnPhi` which is based on

$$
\begin{aligned}
\int_{\Gamma_K} \nabla_{\mathbf{n}}\varphi_i(\mathbf{x})\varphi_j(\mathbf{x})\,\mathrm{d}s &= \int_{X_a}^{X_b} \nabla_{\mathbf{n}}\varphi_i(\mathbf{x})\varphi_j(\mathbf{x})\,\mathrm{d}s = \frac{|X_a X_b|}{|\hat{X}_a \hat{X}_b|}\nabla_{\mathbf{n}}\varphi_i \int_{\hat{X}_a}^{\hat{X}_b} N_j(\hat{\mathbf{x}})\,\mathrm{d}\hat{s} \\
&= \nabla_{\mathbf{n}}\varphi_i |X_a X_b| \int_0^1 N_j(\hat{\mathbf{x}}(t))\,\mathrm{d}t \\
&= \frac{1}{2}\nabla_{\mathbf{n}}\varphi_i |X_a X_b| \sum_{q=1}^{l_q} \omega_q N_j\left(\hat{\mathbf{x}}\left(\frac{1+\xi_q}{2} \right) \right) \\
&= \frac{1}{2}\nabla_{\mathbf{n}}\varphi_i |X_a X_b| \sum_{q=1}^{l_q} \omega_q N_j\left(\hat{\mathbf{x}}_a + \frac{\hat{\mathbf{x}}_b - \hat{\mathbf{x}}_a}{2}(1+\xi_q) \right),
\end{aligned}
$$

where X_a, X_b are two endpoints of Γ_K and $\hat{X}_a, \hat{X}_b$ are their corresponding coordinates in $O\hat{x}\hat{y}$. Here, we applied the quadrature rule (A.2) and $\hat{\mathbf{x}}(t) = (1-t)\hat{\mathbf{x}}_a + t\hat{\mathbf{x}}_b$.

> **Remark 3.5** Notice that, for simplicity, I have used notation N_i which is corre-

sponding to the basis function φ_i on the triangle K. This notation is not theoretically true because the basis function φ_i has $i \in \{1, \ldots, N_{\text{new}}\}$ while the shape function N_i has $i \in \{1, 2, 3\}$. However, when some basis function φ_i restricts on triangle K, it will coincide with the shape function on the corresponding vertex. In the code, thanks to the connectivity of the triplet `[p,e,t]`, they are best agreed together.

Term $\langle \varphi_i, \varphi_j \rangle_{\Gamma_K}$ is computed via the function `intPhiPhi` which is based on

$$
\int_{\Gamma_K} \varphi_i(x) \varphi_j(x) \, ds
$$
$$
= \int_{X_a}^{X_b} \varphi_i(x) \varphi_j(x) \, ds = \frac{|X_a X_b|}{|\hat{X}_a \hat{X}_b|} \int_{\hat{X}_a}^{\hat{X}_b} N_i(\hat{x}) N_j(\hat{x}) \, d\hat{s}
$$
$$
= |X_a X_b| \int_0^1 N_i(\hat{x}(t)) N_j(\hat{x}(t)) \, dt
$$
$$
= \frac{1}{2} |X_a X_b| \sum_{q=1}^{l_q} \omega_q N_i \left(\hat{x} \left(\frac{1 + \xi_q}{2} \right) \right) N_j \left(\hat{x} \left(\frac{1 + \xi_q}{2} \right) \right)
$$
$$
= \frac{1}{2} |X_a X_b| \sum_{q=1}^{l_q} \omega_q N_i \left(\hat{x}_a + \frac{\hat{x}_b - \hat{x}_a}{2} (1 + \xi_q) \right) N_j \left(\hat{x}_a + \frac{\hat{x}_b - \hat{x}_a}{2} (1 + \xi_q) \right).
$$

Algorithm 3.3 gives a way to compute $a_K^{\Gamma_G}$ while Algorithm 3.4 is used to compute $a_K^{\Gamma_P}$.

Algorithm 3.3. Determing a triple vector $\mathbf{i}, \mathbf{j}, \mathbf{v}$ used to find the global stiffness matrix A for the term $a_K^{\Gamma_G}$ (`getTripleGPoG`).

Input : Cut triangles `tris.CTs`, intersections `CT.iPs`, unit noral vectors `CT.uN`.

Output : Triple vectors $\mathbf{i}, \mathbf{j}, \mathbf{v}_{ij}$ where $\mathbf{v}_{ij}$ is corresponding to cases φ_i, φ_j.

$r = 1$;
for $1 \leq \tau \leq N_{CTs}$ **do**
     $s_\tau = $ `CT.iPs`$(:,:,\tau)$;
     $\mathbf{n}_\tau = $ `CT.uN`$(:,\tau)$;
     for $1 < i < 3$ **do**
         for $1 < j < 3$ **do**
             $\mathbf{i}(r) = $ `tris.CTs`(i, τ); $\mathbf{j}(r) = $ `tris.CTs`(j, τ);
             $c_{ji} = $ `intgradnPhi`$(j, i, s_\tau, \mathbf{n}_\tau)$; $c_{ij} = $ `intgradnPhi`$(i, j, s_\tau, \mathbf{n}_\tau)$;
             $\mathbf{v}_{1,1}(r) = -c_{ji} - c_{ij}$; $\mathbf{v}_{1,2}(r) = c_{ji} - c_{ij}$;
             $\mathbf{v}_{2,1}(r) = -c_{ji} + c_{ij}$; $\mathbf{v}_{2,2}(r) = c_{ji} + c_{ij}$;
             $r = r + 1$;

Algorithm 3.4. Determing a triple vector $\mathbf{i}, \mathbf{j}, \mathbf{v}$ used to find the global stiffness matrix A for the term $a_K^{\Gamma_P}$ (`getTriplePPoG`).

Input : Cut triangles `tris.CTs`, intersections `CT.iPs`.
Output : Triple vectors $\mathbf{i}, \mathbf{j}, \mathbf{v}_{ij}$ where $\mathbf{v}_{ij}$ is corresponding to cases φ_i, φ_j.

$r = 1$;
for $1 \leq \tau \leq N_{CTs}$ **do**
  $s_\tau = $ `CT.iPs`$(:,:,\tau)$;
  for $1 < i < 3$ **do**
    for $1 < j < 3$ **do**
      $\mathbf{i}(r) = $ `tris.CTs`(i,τ); $\mathbf{j}(r) = $ `tris.CTs`(j,τ);
      $\mathbf{v}_{1,1}(r) = $ `intPhiPhi`(j,i,s_τ,τ);
      $\mathbf{v}_{1,2}(r) = $ `intPhiPhi`(j,i,s_τ,τ);
      $\mathbf{v}_{2,1}(r) = -$ `intPhiPhi`(j,i,s_τ,τ);
      $\mathbf{v}_{2,2}(r) = -$ `intPhiPhi`(j,i,s_τ,τ);
      $r = r + 1$;

Remark 3.6 In the code, we notice on the order of $\mathbf{i}, \mathbf{j}$ which follows the rule $A_{ij} = a_h(\varphi_j, \varphi_i)$.

3.4.2 Assembling of right hand side

From (3.8), we have

$$F_j = L_h(\varphi_j) = \sum_{K \in \mathscr{T}_h} L_K(\varphi_j) := \sum_{K \in \mathscr{T}_h} \int_K f \varphi_j \, \mathrm{d}x. \tag{3.12}$$

Similar to the case of assembling the stiffness matrix, we also need to consider (3.12) on not-cut and cut triangles. The process is described in Figure 3.7.
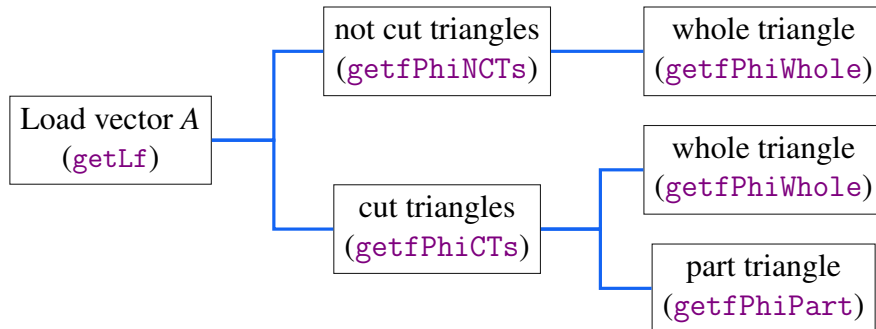

Figure 3.7. The idea of assembling the right hand side F.

On not-cut triangles. On not-cut triangles, the integral is computed on the whole triangle K, thanks to quadrature formula (A.4), we have the following expression (3.13) which is corresponding to the function `getfPhiWhole` in the toolbox.

$$
\begin{aligned}
L_K(\varphi_j) = \int_K f(\mathbf{x})\varphi_j(\mathbf{x})\,d\mathbf{x} &= 2|K| \int_{\hat{K}} f(\mathbf{P}(\hat{\mathbf{x}}))N_j(\hat{\mathbf{x}})\,d\hat{\mathbf{x}} \\
&= |K| \sum_{q=1}^{l_q} \omega_q f(\mathbf{P}(\hat{\mathbf{x}}_q))N_j(\hat{\mathbf{x}}_q).
\end{aligned}
\tag{3.13}
$$

Algorithm 3.5. Determing couple vector $\mathbf{i}, \mathbf{f}$ used to find load vector F for term L_K on not-cut triangles (`getfPhiNCTs`).

Input : Not-cut triangles `tris.NCTsi` and function f.
Output Couple vectors $\mathbf{j}, \mathbf{f}$
:

$r = 1$;
for $1 \leq \tau \leq N_{NCTs}$ **do**
 for $1 < i < 3$ **do**
  $\mathbf{j}(r) = $ `tris.NCTsi`(i, τ);
  $\mathbf{f}(r) = $ `getfPhiWhole`(i, τ, f);
  $r = r + 1$;

On cut triangles. This case is very different from the one for not-cut triangles because the supports of basis functions locating on this type of triangle is only a part of triangle. That's the reason why we need to modify a little bit on the quadrature rule. We need more than one reference triangle, specifically two. An idea is illustrated in Figure 3.8.
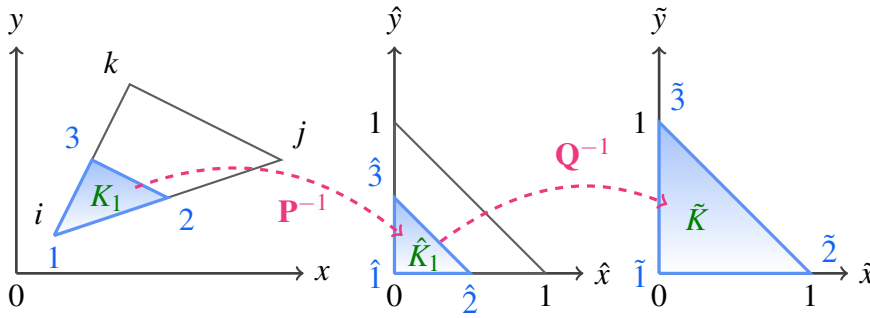

Figure 3.8. An idea to get the quadrature when we want to integrate on a part of triangle.

This process contains two steps,

- **Step 1**: $K_1 \rightarrow \hat{K}_1$ based on $K \rightarrow \hat{K}$,

$$
\begin{pmatrix} x \\ y \end{pmatrix} = \mathbf{P} \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} x_i(1-\hat{x}-\hat{y}) + x_j\hat{x} + x_k\hat{y} \\ y_i(1-\hat{x}-\hat{y}) + y_j\hat{x} + y_k\hat{y} \end{pmatrix},
$$

$$dx\,dy = 2|K|\,d\hat{x}\,d\hat{y}.$$

- **Step 2**: $\hat{K}_1 \to \tilde{K}$,

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \mathbf{Q}\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} = \begin{pmatrix} x_{\hat{1}}(1-\tilde{x}-\tilde{y}) + x_{\hat{2}}\tilde{x} + x_{\hat{3}}\tilde{y} \\ y_{\hat{1}}(1-\tilde{x}-\tilde{y}) + y_{\hat{2}}\tilde{x} + y_{\hat{3}}\tilde{y} \end{pmatrix}, \tag{3.14}$$

$$d\hat{x}\,d\hat{y} = 2|K_1|\,d\tilde{x}\,d\tilde{y}. \tag{3.15}$$

Then we have (3.16) which is corresponding to the function `getfPhiPart` in the toolbox. This function will find the integral on a part of triangle.

$$\begin{aligned} L_K(\varphi_j^{(1)}) &= \int_K f(x,y)\varphi_j^{(1)}(x,y)\,dx\,dy = \int_{K_1} f_1(x,y)\varphi_j(x,y)\,dx\,dy \\ &= 2|K| \int_{\hat{K}_1} f_1(\mathbf{P}(\hat{x},\hat{y}))N_j(\hat{x},\hat{y})\,d\hat{x}\,d\hat{y} \\ &= 4|K||K_1| \int_{\tilde{K}} f_1(\mathbf{P}(\mathbf{Q}(\tilde{x},\tilde{y})))N_j(\mathbf{Q}(\tilde{x},\tilde{y}))\,d\tilde{x}\,d\tilde{y} \\ &= 2|K||K_1| \sum_{q=1}^{l_q} \omega_q f_1(\mathbf{P}(\mathbf{Q}(\tilde{x}_q,\tilde{y}_q)))N_j(\mathbf{Q}(\tilde{x}_q,\tilde{y}_q)), \end{aligned} \tag{3.16}$$

where $f_i = f|_{\Omega_i}$.

> **Remark 3.7** Note that, (3.16) is only used for the basis function whose support is the *triangular-part* of a cut triangle. If one wants to find L_K for some basis function whose support is the *quadrangular-part* of a cut triangle, we will find on the whole triangle first and take the substraction from the triangular part. More specifically, suppose that $\varphi_j^{(2)}$ has support K_2 which is a quadrangular-part of triangle K, then
>
> $$\begin{aligned} L_K(\varphi_j^{(2)}) &= \int_K f\varphi_j^{(2)}\,dx = \int_{K_2} f_2\varphi_j\,dx \\ &= \int_K f_2\varphi_j\,dx - \int_{K_1} f_2\varphi_j\,dx, \\ &= \mathtt{getfPhiWhole}(j,K,f_2) - \mathtt{getfPhiPart}(j,K,1,f_2), \end{aligned}$$
>
> where K_1 is the triangular part of K.

A full algorithm to find the load vector on cut triangles is given in Algorithm 3.6.

3.4.3 Implementing the Ghost Penalty

An idea of the Ghost penalty is given in Section 2.4. Recall that, we need to add two following terms into the bilinear form a_h,

---

Algorithm 3.6. Determing couple vector $\mathbf{i}, \mathbf{f}$ used to find load vector F for term L_K on cut triangles (`getfPhiCTs`).

Input : Cut triangles `tris.CTs` and function $f_i = f|_{\Omega_i}$ for $i = 1, 2$.

Output Couple vectors $\mathbf{j}, \mathbf{f}_1, \mathbf{f}_2$ where $\mathbf{f}_1, \mathbf{f}_2$ are corresponding to $L_K(\varphi_j^{(1)})$ and
:
$\qquad L_K(\varphi_j^{(2)})$ respectively.

$r = 1$;
for $1 \leq \tau \leq N_{CTs}$ **do**
$\quad$ **for** $1 < i < 3$ **do**
$\quad\quad$ $\mathbf{j}(r) = \text{tris.CTs}(i, \tau)$;
$\quad\quad$ **if** `CT.type` $== 2$ **then**
$\quad\quad\quad$ $\mathbf{f}_1(r) = \text{getfPhiWhole}(i, \tau, f_1) - \text{getfPhiPart}(i, \tau, 2, f_1)$;
$\quad\quad\quad$ $\mathbf{f}_2(r) = \text{getfPhiPart}(i, \tau, 2, f_2)$;
$\quad\quad$ **else**
$\quad\quad\quad$ $\mathbf{f}_2(r) = \text{getfPhiWhole}(i, \tau, f_2) - \text{getfPhiPart}(i, \tau, 1, f_2)$;
$\quad\quad\quad$ $\mathbf{f}_1(r) = \text{getfPhiPart}(i, \tau, 1, f_1)$;
$\quad\quad$ r=r+1;

$$j_r(u_h, v_h) = \sum_{e \in \mathscr{E}_G^r} \langle \gamma_r h [\![\nabla_{\mathbf{n}_e} u_h]\!]_e, [\![\nabla_{\mathbf{n}_e} v_h]\!]_e \rangle_e, \; r = 1, 2,$$

where $\mathscr{E}_G^i$ is defined in (2.32).

Ghost penalty edges. Because ghost penalty terms are defined on special edges of cut triangles depending on the "fictitious domain" $\Omega_{\mathscr{T}}^2$ (j_1 uses different edges in comparison with j_2, cf. Figure 2.7), this leads us to first find the "ghost penalty edges" which are the ones to be used in computing each ghost penalty term.

In addition, we are computing the jump on edges $[\![\nabla u_h \cdot \mathbf{n}_e]\!]_e$ which is defined as

$$[\![\nabla u_h \cdot \mathbf{n}_e]\!]_e = (\nabla u_h)|_K \cdot \mathbf{n}_e - (\nabla u_h)|_{K'} \cdot \mathbf{n}_e,$$

where $e = K \cap K'$. That means, for each considered edge, we need to determine two adjacent triangles to which this edge belongs. Another problem comes with determining a unit normal vector on edge $\mathbf{n}_e$, *i.e.*, we need to know the order of two endpoints of each considered edge. Thanks to the function `getGPEdges` in the toolbox (Algorithm 3.7), it gives us a vector `idxNBTris` and a $9 \times N_{\text{edges}}$ matrix `eGP`. The former contains the indices of cut triangles and their neighbors while the latter contains the following information.

- **Rows** $1, 2$: contains the information of two endpoints of GP edges.
- **Rows** $3, 4$: contains the information of two adjacent triangles that these edges belongs to.

- **Row 5**: contains the type of GP edges (1 if this edge is entirely in Ω_1, 2 in Ω_2 and 3 if it cuts the interface).
- **Rows 6, 7**: the order of two endpoints in the triangle is indicated in line 3 (1, 2 or 3).
- **Rows 8, 9**: the order of two endpoints in the triangle is indicated in line 4 (1, 2 or 3).

Why we need rows $6-9$? It's because we need to know about ∇u_h on each triangle K, K'. Thanks to the function `getGradPhi` (cf. (3.6)), we are able to know ∇u_h on each vertex of some triangle but in the fixed order. When we consider some edge, for example, e_{ij}, we don't know its endpoints i, j are corresponding to which vertex of the triangle K and K' (*i.e.*, $K \cap K' = e_{ij}$), lines $6-9$ help us do that.

A full algorithm and an example describing the way `getGPEdges` works are given in Algorithm 3.7.

Algorithm 3.7. Determine the ghost penalty edges (`getGPEdges`).

Input : Triangles `tris` of the mesh.
Output $9 \times N_{\text{edges}}$ matrix `eGP` contains ghost penalty edges.
:

1. Find all neighbors of cut triangles (`neighborTris` which contains indexes in `msh.t`).
2. Collect all edges of `neighborTris` and store them to two first lines of `eGP` (`eGP(1:2,:)` contains two endpoints of edges).
3. Collect all triangles that contain the edges in step 2 and store them to the third line of `eGP` (`eGP(3,:)` contains the indexes in `neighborTris`).
4. Remember the position of vertices of edges in step 2 in the triangles `eGP(3,:)` and store them to next two lines (`eGP(6:7,:)`).
5. Find the other triangles which are adjacent to the triangles found in step 3 and also contain the edges in step 2. Then we store them to the fourth line of `eGP` (`eGP(4,:)` contains the indexes in `neighborTris`).
6. Remember the position of vertices found in step 2 in the triangles `eGP(4,:)` and store them to `eGP(8:9,:)`.
7. Filter out the wrong edges (to be sure that we are only considering the edges of cut triangles).
8. Mark the type of edges and store them to `eGP(5,:)`.

Ghost penalty terms. Recall (2.30), we look for

$$
\begin{aligned}
A_{ji}^G &= a_h(\varphi_i, \varphi_j) + k_1 j_1(\varphi_i, \varphi_j) + k_2 j_2(\varphi_i, \varphi_j) \\
&= A_{ij} + k_1 j_1(\varphi_i, \varphi_j) + k_2 j_2(\varphi_i, \varphi_j), \quad \text{for } i, j = 1, \dots, N_{\text{new}},
\end{aligned}
$$

where A_{ij} is computed in (3.9).

Suppose that nodes i and j are located on the edge e_{ij} which is the common part of two adjacent triangles K and K'. The edge e_{ij} falls into one of two cases, either cut

or not-cut edge.

$$
\begin{aligned}
j_r(\varphi_i^{(r)}, \varphi_j^{(r)}) &= \int_{e_{ij}} \gamma_r h [\![\nabla_{\mathbf{n}_e} \varphi_i^{(r)}]\!][\![\nabla_{\mathbf{n}_e} \varphi_j^{(r)}]\!]\,\mathrm{d}s = \int_{e_{ij}^{(r)}} \gamma_r h [\![\nabla_{\mathbf{n}_e} \varphi_i]\!][\![\nabla_{\mathbf{n}_e} \varphi_j]\!]\,\mathrm{d}s \\
&= \gamma_r h [\![\nabla_{\mathbf{n}_e} \varphi_i]\!][\![\nabla_{\mathbf{n}_e} \varphi_j]\!]|e_{ij}^{(r)}|.
\end{aligned}
\tag{3.17}
$$

where $\mathbf{n}_e := \mathbf{n}_{e_{ij}}, e_{ij}^{(r)} = e_{ij} \cap \Omega_r$ which is either cut or not-cut edge, for $r = 1, 2$.

> **Remark 3.8** We don't have the cases $j_r(\varphi_i^{(l)}, \varphi_j^{(l)})$ where $r \neq l$ because we only consider j_r which is corresponding to basis functions whose supports are located in Ω_r. We don't have either the cases $j_r(\varphi_i^{(1)}, \varphi_j^{(2)})$ or $j_r(\varphi_i^{(2)}, \varphi_j^{(1)})$ for $r = 1, 2$ because their supports have no common on the edges of cut triangles.

A full algorithm to find $\Sigma_{r=1}^2 k_r j_r$ is given in Algorithm 3.8.

Algorithm 3.8. Determine the triple vectors $\mathbf{i}, \mathbf{j}, \mathbf{v}$ which are corresponding to the ghost penalty terms.

Input : Cut triangles `tris.CTs`.
Output Triple vectors $\mathbf{i}, \mathbf{j}, \mathbf{v}$.
:

    1. Get ghost penalty edges: `eGP = getGPEdges()`.
    2. Classify edges into there groups depending on the subdomain: e_{j_1} (the not-cut edges for the term j_1), e_{j_2} (the not-cut edges for the term j_2), e_{j_c} (the cut edge for both terms).
    3. Get the triplet on each group based on (3.17).

3.4.4   Implementation issue of norms

With the same manner as in the section of assembling (cf. Sections 3.4.1, 3.4.2), we can calculate the norms in the standard FE space or in the NXFEM space of discrete or continuous functions. You can find more details in Section A.6.

3.5   Numerical examples

Some numerical validation test cases and comments are already given in Section 2.6.

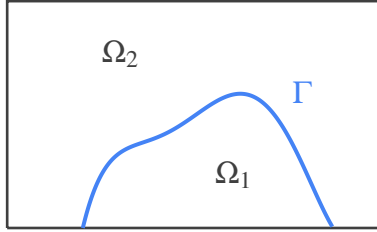# 4. Resolution of semilinear-interface system by NXFEM

Contents

Under the motivation of modeling a biofilm model, we introduce a system of semilinear interface problem. A technique of decoupling such a problem into three new separated ones in which we can handle each of them more easily is also presented. The equivalence between the main system and the new one has been clarified afterward.

For the three separated problems, we only need to work with two of them, the remaining will be solved based on the others. There is one simple linear problem and one nonlinear problem we need to work with. For the former, we use the standard NXFEM method introduced in Chapter 2 to find a solution. For the latter which has a nonlinear form, we use the fixed point theorem to demonstrate the existence of its solution. After that, a result of the convergence of discrete solutions to the solution of the weak problem will be showed thanks to the idea of techniques in the Discontinuous Galerkin Method proposed by Ern & Di Pietro [25]. Their work

actually relied on techniques inspired by the Finite Volume literature given in the work of Eymard *et al.*. [29]. Note that, Ern & Di Pietro worked on the discontinuity on each side of the mesh's elements while we only work on the discontinuity of functions on the interface.

4.1 Model

Main model. Let us consider a convex polygonal, Lipschitz and bounded domain Ω in $\mathbb{R}^2$ such that $\overline{\Omega} = \overline{\Omega}_1 \cup \overline{\Omega}_2$. These two regions are separated by a sufficiently smooth interface Γ. We consider the following stationary problem (4.2) which is a system of semilinear equations.

$$\begin{cases} -\nabla \cdot (\alpha \nabla u) + vg(u) = f_u & \text{in } \Omega, \\ -\nabla \cdot (\beta \nabla v) - \lambda vg(u) = f_v & \text{in } \Omega, \\ [\![u]\!] = [\![\alpha \nabla_{\mathbf{n}} u]\!] = 0 & \text{on } \Gamma, \\ v = \nabla_{\mathbf{n}} v = 0 & \text{on } \Gamma, \\ u = \bar{u} & \text{on } \partial\Omega, \\ v = \bar{v} & \text{on } \partial\Omega. \end{cases} \quad (4.2)$$

Here, $\mathbf{n}$ denotes the unit normal at a given point on Γ pointing from Ω_1 to Ω_2. We also take the following special form of $g, f_u, f_v, \bar{v}$ which are agreed to a biofilm's model, i.e. there is no bacteria outside the biomass region Ω_1.

$$g(u) = \begin{cases} g_1(u) \text{ in } \Omega_1, \\ 0 \text{ in } \Omega_2. \end{cases} \quad f_u = \begin{cases} f_{u_1} \text{ in } \Omega_1, \\ f_{u_2} \text{ in } \Omega_2. \end{cases} \quad f_v = \begin{cases} f_{v_1} \text{ in } \Omega_1, \\ 0 \text{ in } \Omega_2. \end{cases} \quad \bar{v} = \begin{cases} \bar{v}_1 \text{ on } \partial\Omega_1 \backslash \Gamma, \\ 0 \text{ on } \partial\Omega_2 \backslash \Gamma. \end{cases}$$

Assumption 4.1 We suppose that $\bar{u}, \lambda > 0, \bar{v}_1 \geq 0$ and two diffusion coefficients α, β are assumed to be piecewise constants $\alpha = \alpha_i > 0, \beta = \beta_i > 0$ in Ω_i for $i = 1, 2$. In general, we have $\alpha_1 \neq \alpha_2, \beta_1 \neq \beta_2$. We also assume that functions g, f_u, f_v satisfies (4.3).

$$\begin{cases} f_u, f_v \in L^2(\Omega), \\ g \text{ measurable with respect to } \mathrm{x} \in \Omega \text{ and } 0 \leq \dfrac{\partial g(\mathrm{x}, u)}{\partial u} \leq \xi(\mathrm{x}) \in L^1(\Omega). \end{cases} \quad (4.3)$$

Recall from Definition 2.2 the space $H^1(\Omega_{12})$ and its norm. We also need to use notation of spaces

$$\begin{array}{|ll|} \hline V := \{v \in H^1(\Omega) : v = 0 \text{ on } \Gamma\}, & \\ V_0 = \{v \in V : v = 0 \text{ on } \partial\Omega\}. & \\ \hline \end{array} \quad (4.4)$$

They are Hilbert spaces with the usual norms in $H^1(\Omega)$.

4.2 Decoupling the system of equations

By putting $w = u + \frac{\beta}{\alpha\lambda}v$, we are able to decouple the system (4.2) into three separated problems (4.5), (4.6) and (4.7).

$$
\begin{cases}
-\nabla \cdot (\alpha\nabla w) = f_w := f_u + \frac{1}{\lambda}f_v & \text{in } \Omega_i, \, i = 1,2, \\
[\![w]\!] = [\![\alpha\nabla_{\mathbf{n}}w]\!] = 0 & \text{on } \Gamma, \\
w = \bar{w} := \bar{u} + \frac{\beta}{\alpha\lambda}\bar{v} & \text{on } \partial\Omega.
\end{cases}
\tag{4.5}
$$

$$
\begin{cases}
-\nabla \cdot (\beta\nabla v) - \lambda v g(w - \frac{\beta}{\alpha\lambda}v) = f_v & \text{in } \Omega_i, \, i = 1,2, \\
v = \nabla_{\mathbf{n}}v = 0 & \text{on } \Gamma, \\
v = \bar{v} & \text{on } \partial\Omega.
\end{cases}
\tag{4.6}
$$

$$
u = w - \frac{\beta}{\alpha\lambda}v. \tag{4.7}
$$

The idea of this decoupling is that instead of working directly on (4.2), we can work on an equivalent system (4.5, 4.6, 4.7). The advantage is to change from working on a system of equations to working on separated equations which are more easily to be handled.

> **Proposition 4.1** The system (4.2) is equivalent to decoupled problems (4.5), (4.6), (4.7). In other words, u, v is a solution of (4.2) if and only if w, v and u are solutions of (4.5), (4.6) and (4.7).

Proof. It's easy to get the result from definition of w, $w = u + \frac{\beta}{\alpha\lambda}v$. Note that, because of the difference in the interface condition between u (with jump) and v (without jump), we have to seek w, v before seeking u. ∎

4.2.1 Weak formulations

Multiply both sides of (4.5) by a test function $\varphi \in H_0^1(\Omega)$ and apply Green formula, we have

$$
\begin{aligned}
\langle f_w, \varphi \rangle_\Omega &= -\langle \nabla \cdot (\alpha\nabla w), \varphi \rangle_\Omega = -\langle \nabla \cdot (\alpha\nabla w), \varphi \rangle_{\Omega_1} - \langle \nabla \cdot (\alpha\nabla w), \varphi \rangle_{\Omega_2} \\
&= \langle \alpha\nabla w, \nabla\varphi \rangle_{\Omega_1} - \langle \alpha\nabla_{\mathbf{n}}w, \varphi \rangle_{\partial\Omega_1} \\
&\quad + \langle \alpha\nabla w, \nabla\varphi \rangle_{\Omega_2} - \langle \alpha\nabla_{\mathbf{n}}w, \varphi \rangle_{\partial\Omega_2} \\
&= \langle \alpha\nabla w, \nabla\varphi \rangle_{\Omega_{12}} - \langle \alpha\nabla_{\mathbf{n}}w, \varphi \rangle_{\partial\Omega} - \int_\Gamma [\![\alpha\nabla_{\mathbf{n}}w\varphi]\!] \, \mathrm{d}s \\
&= \langle \alpha\nabla w, \nabla\varphi \rangle_{\Omega_{12}} - \langle [\![\alpha\nabla_{\mathbf{n}}w]\!], \{\!\{\varphi\}\!\} \rangle_\Gamma - \langle \{\!\{\alpha\nabla_{\mathbf{n}}w\}\!\}, [\![\varphi]\!] \rangle_\Gamma \\
&\quad + (\kappa_2 - \kappa_1)\langle [\![\alpha\nabla_{\mathbf{n}}w]\!], [\![\varphi]\!] \rangle_\Gamma,
\end{aligned}
\tag{4.8}
$$

Because $[\![w]\!] = [\![\alpha\nabla_{\mathbf{n}}w]\!] = 0$ on Γ and $\varphi \in H_0^1$, we will get

$$\langle \alpha\nabla w, \nabla\varphi \rangle_\Omega = \langle f_w, \varphi \rangle_\Omega, \quad \forall\varphi \in H_0^1(\Omega). \qquad (4.9)$$

A weak form of (4.5) is to find $w \in H^1(\Omega)$ such that $w = \bar{w}$ on $\partial\Omega$ and w satisfies (4.9).

Similarly, multiply both sides of (4.6) by a test function $\varphi \in V_0$ and apply Green formula on each subdomain (note that, we apply the Nitsche's technique on each subdomain to enforce weakly the jump on the interface, it will be helpful and be recalled in the discrete section)

$$\langle \beta\nabla v, \nabla\varphi \rangle_{\Omega_1} - \langle \beta\nabla_{\mathbf{n}}v, \varphi \rangle_{\partial\Omega_1} - \langle q(v), \varphi \rangle_{\Omega_1} + \langle v, \theta\kappa_1\varphi - \beta\nabla_{\mathbf{n}}\varphi \rangle_\Gamma = \langle f_v, \varphi \rangle_{\Omega_1},$$
$$\langle \beta\nabla v, \nabla\varphi \rangle_{\Omega_1} + \langle \beta\nabla_{\mathbf{n}}v, \varphi \rangle_{\partial\Omega_2} - \langle q(v), \varphi \rangle_{\Omega_2} + \langle v, \theta\kappa_1\varphi + \beta\nabla_{\mathbf{n}}\varphi \rangle_\Gamma = \langle f_v, \varphi \rangle_{\Omega_2},$$

with θ sufficiently large enough and

$$q(v) := \lambda v g(w - \frac{\beta}{\alpha\lambda}v).$$

Then we have

$$\langle f_v, \varphi \rangle_\Omega = \langle \beta\nabla v, \nabla\varphi \rangle_{\Omega_{12}} - \langle \beta\nabla_{\mathbf{n}}v, \varphi \rangle_{\partial\Omega} - \langle q(v), \varphi \rangle_\Omega$$
$$- \int_\Gamma [\![\beta\nabla_{\mathbf{n}}v\varphi]\!]\,\mathrm{d}s + \theta\int_\Gamma \{\!\{v\varphi\}\!\}\,\mathrm{d}s - \int_\Gamma [\![v\beta\nabla_{\mathbf{n}}\varphi]\!]\,\mathrm{d}s.$$

Recall (2.11), Propotition 2.1 and expand the expression,

$$\begin{aligned}
\langle f_v, \varphi \rangle_\Omega = {} & \langle \beta\nabla v, \nabla\varphi \rangle_{\Omega_{12}} - \langle q(v), \varphi \rangle_\Omega \\
& - \langle [\![\beta\nabla_{\mathbf{n}}v]\!], \{\!\{\varphi\}\!\} \rangle_\Gamma - \langle \{\!\{\beta\nabla_{\mathbf{n}}v\}\!\}, [\![\varphi]\!] \rangle_\Gamma \\
& - (\kappa_2 - \kappa_1)\langle [\![\beta\nabla_{\mathbf{n}}v]\!], [\![\varphi]\!] \rangle_\Gamma + \theta\langle \{\!\{v\}\!\}, \{\!\{\varphi\}\!\} \rangle_\Gamma \\
& + \theta\kappa_1\kappa_2\langle [\![v]\!], [\![\varphi]\!] \rangle_\Gamma - \langle [\![v]\!], \{\!\{\beta\nabla_{\mathbf{n}}\varphi\}\!\} \rangle_\Gamma \\
& - \langle \{\!\{v\}\!\}, [\![\beta\nabla_{\mathbf{n}}\varphi]\!] \rangle_\Gamma - (\kappa_2 - \kappa_1)\langle [\![v]\!], [\![\beta\nabla_{\mathbf{n}}\varphi]\!] \rangle_\Gamma.
\end{aligned} \qquad (4.10)$$

Because $v = \nabla_{\mathbf{n}}v = 0$ on Γ and $\varphi \in V_0$,

$$\langle \beta\nabla v, \nabla\varphi \rangle_\Omega - \langle q(v), \varphi \rangle_\Omega = \langle f_v, \varphi \rangle_\Omega, \quad \forall\varphi \in V_0. \qquad (4.11)$$

A weak form of (4.6) is to find $v \in V$ such that $v = \bar{v}$ on $\partial\Omega$ and v satisfies (4.11).

Next, we need to verify the equivalence between the weak problems (4.5), (4.6) and the decoupled system (4.9), (4.11). Indeed,

> **Proposition 4.2** If (w, v) is a solution of (4.5, 4.6) then it is also a solution of (4.9, 4.11). We have also the converse if the weak solution (w, v) of (4.9, 4.11) belongs to $H^2(\Omega)$.

Proof. The first statement can be obtained easily from the construction of weak formulations. We now prove that if (w, v) solves (4.9, 4.11) and $w, v \in H^2(\Omega)$ then it also solves (4.5, 4.6).

First, consider problem (4.9) and $w \in H^2(\Omega)$ is a solution of it, we have $w = \bar{w}$ on $\partial \Omega$ and

$$\langle \alpha \nabla w, \nabla \varphi \rangle_\Omega = \langle f_w, \varphi \rangle_\Omega, \quad \forall \varphi \in H_0^1(\Omega).$$

Performing integration by parts on the $\langle \alpha \nabla w, \nabla \varphi \rangle_{\Omega_{12}}$ backwards on each subdomain Ω_i, we have

$$-\langle \nabla \cdot (\alpha \nabla w), \varphi \rangle_{\Omega_{12}} + \langle [\![\alpha \nabla_\mathbf{n} w]\!], \{\!\{\varphi\}\!\} \rangle_\Gamma = \langle f_w, \varphi \rangle_\Omega, \quad \forall \varphi \in H_0^1(\Omega). \qquad (4.12)$$

We choose $\varphi = 0$ on Γ, (4.12) becomes

$$-\langle \nabla \cdot (\alpha \nabla w), \varphi \rangle_{\Omega_{12}} = \langle f_w, \varphi \rangle_\Omega, \quad \forall \varphi \in H_0^1(\Omega) \cap \{\varphi = 0 \text{ on } \Gamma\}.$$

We could argue in each subdomain Ω_i,

$$-\nabla \cdot (\alpha \nabla w) = f_w \text{ a.e. on } \Omega_i.$$

Now back to (4.12) we have

$$\langle [\![\alpha \nabla_\mathbf{n} w]\!], \{\!\{\varphi\}\!\} \rangle_\Gamma = 0, \quad \forall \varphi \in H_0^1(\Omega) \cap \{\varphi \neq 0 \text{ on } \Gamma\}.$$

This implies, $[\![\alpha \nabla_\mathbf{n} w]\!] = 0$ on Γ. To sum up, we have shown that w solves (4.9) and w also satisfies all conditions of problem (4.5).

With the same technique, we can easily obtain the same result for problem (4.6) and (4.7). Indeed, from (4.11) and for any $\varphi \in H_0^1(\Omega)$, we have

$$-\langle \nabla \cdot (\beta \nabla v), \varphi \rangle_{\Omega_{12}} + \langle [\![\beta \nabla_\mathbf{n} v]\!], \{\!\{\varphi\}\!\} \rangle_\Gamma + \langle [\![\varphi]\!], \{\!\{\beta \nabla_\mathbf{n} v\}\!\} \rangle_\Gamma - \langle q(v), \varphi \rangle_\Omega = \langle f_v, \varphi \rangle_\Omega.$$

Choose $\varphi = 0$ on Γ then

$$-\langle \nabla \cdot (\beta \nabla v), \varphi \rangle_{\Omega_{12}} - \langle q(v), \varphi \rangle_\Omega = \langle f_v, \varphi \rangle_\Omega, \quad \forall \varphi \in \{\psi \in H_0^1(\Omega), \psi = 0 \text{ on } \Gamma\}.$$

This implies $-\nabla \cdot (\beta \nabla v) - q(v) = f_v$. Thus,

$$\langle [\![\beta \nabla_{\mathbf{n}} v]\!], \{\!\{ \varphi \}\!\} \rangle_\Gamma + \langle [\![\varphi]\!], \{\!\{ \beta \nabla_{\mathbf{n}} v \}\!\} \rangle_\Gamma = 0, \quad \forall \varphi \in \{ \psi \in H_0^1(\Omega), \psi \neq 0 \text{ on } \Gamma \}. \tag{4.13}$$

Don't forget that $\varphi \in H_0^1(\Omega)$ or we have $[\![\varphi]\!] = 0$, this leads to

$$\langle [\![\beta \nabla_{\mathbf{n}} v]\!], \{\!\{ \varphi \}\!\} \rangle_\Gamma = 0, \quad \forall \varphi \in \{ \psi \in H_0^1(\Omega), \psi \neq 0 \text{ on } \Gamma \},$$

or we have

$$[\![\beta \nabla_{\mathbf{n}} v]\!] = 0, \text{ on } \Gamma. \tag{4.14}$$

Replace (4.14) in (4.13), we get

$$\langle [\![\varphi]\!], \{\!\{ \beta \nabla_{\mathbf{n}} v \}\!\} \rangle_\Gamma = 0, \quad \forall \varphi \in \{ \psi \in H_0^1(\Omega), \psi \neq 0 \text{ on } \Gamma, [\![\psi]\!] \neq 0 \text{ on } \Gamma \}.$$

or,

$$\{\!\{ \beta \nabla_{\mathbf{n}} v \}\!\} = 0, \text{ on } \Gamma. \tag{4.15}$$

Coupling (4.14) and (4.15), we have $\nabla_{\mathbf{n}} v = 0$ on Γ. To sum up, we have shown that v solves (4.11) and v also satisfies all conditions of problem (4.6). ∎

Proposition 4.3 With g, f_u, f_v satisfying Assumption 4.1, the problem (4.9, 4.11) has a unique solution.

Proof. If $f_w \in L^2(\Omega)$ and thanks to [14], problem (4.9) has unique solution in H^2 on each subdomain and further,

$$\|w\|_{H^1(\Omega_{12})} + \|w\|_{H^2(\Omega_{12})} \leq C\|f_w\|_{L^2(\Omega_{12})}.$$

It's also known in [35, Theorem 2.1] that the semilinear problem (4.11) has unique solution in V if $f_v \in L^2(\Omega)$ and $g(x, u(x))$ satisfies

$$g \text{ measurable w.r.t } x \in \Omega \text{ and } 0 \leq \frac{\partial g(x, u)}{\partial u} \leq \xi(x) \in L^1(\Omega). \tag{4.16}$$

Clearly, the choice of g in (1.1) satisfies the condition (4.16) because

$$0 \leq \frac{\partial g}{\partial u} = \mu_s \frac{K_s}{(K_s + u)^2} \leq \mu_s \in L^1(\Omega).$$

∎

4.2.2 Discrete formulations

Applying the same arguments as in Section 2.2, a discrete form of problem (4.5) is as follows: Seek a solution $w_h \in V_h^\Gamma$ such that $w_h = \bar{w}$ on $\partial\Omega$ and

$$a_{wh}(w_h, \varphi_h) = K_{wh}(\varphi_h), \quad \forall \varphi_h \in V_h^0, \tag{4.17}$$

where V_h^0 defined in (2.15) and

$$a_{wh}(w_h, \varphi_h) := \langle \alpha \nabla w_h, \nabla \varphi_h \rangle_{\Omega_{12}}$$
$$- \langle [\![w_h]\!], \{\!\{\alpha \nabla_{\mathbf{n}} \varphi_h\}\!\} \rangle_\Gamma - \langle \{\!\{\alpha \nabla_{\mathbf{n}} w_h\}\!\}, [\![\varphi_h]\!] \rangle_\Gamma + \zeta \langle [\![w_h]\!], [\![\varphi_h]\!] \rangle_\Gamma,$$
$$K_{wh}(\varphi_h) := \langle f_w, \varphi_h \rangle_\Omega.$$

Similarly, look back to (4.10), if we take $\varphi_h \in V_h^0$ instead of $H_0^1(\Omega)$, we don't have $[\![\varphi_h]\!] = 0$ on Γ and the terms corresponding to this will remain in the formulation. Moreover, because of the condition $v = \nabla_{\mathbf{n}} v = 0$ on Γ instead of jump condition like in the case of w, we need not only $[\![v]\!] = 0$ but also $\{\!\{v\}\!\} = 0$ on Γ so that we can imply $v = 0$ on Γ. That's why when we apply the Nistche's idea in (4.10), we choose a little different coefficients. Note that, one can choose different forms of a_{vh} based on (4.10) as long as terms contains jump and average of v will be kept. We propose a choice of a_{vh} given in (4.19) with which, corresponding to problem (4.6), we consider a discrete problem: Seek a solution $v_h \in V_h^\Gamma$ such that $v_h = \bar{v}$ on $\partial\Omega$ and

$$a_{vh}(v_h, \varphi_h) - \langle q(v_h), \varphi_h \rangle_\Omega = K_{vh}(\varphi_h), \quad \forall \varphi_h \in V_h^0, \tag{4.18}$$

where,

$$a_{vh}(v_h, \varphi_h) := \langle \beta \nabla v_h, \nabla \varphi_h \rangle_{\Omega_{12}} - \langle [\![v_h]\!], \{\!\{\beta \nabla_{\mathbf{n}} \varphi_h\}\!\} \rangle_\Gamma - \langle \{\!\{\beta \nabla_{\mathbf{n}} v_h\}\!\}, [\![\varphi_h]\!] \rangle_\Gamma$$
$$+ \theta \langle \{\!\{v_h\}\!\}, \{\!\{\varphi_h\}\!\} \rangle_\Gamma + \theta \kappa_1 \kappa_2 \langle [\![v_h]\!], [\![\varphi_h]\!] \rangle_\Gamma, \tag{4.19}$$
$$K_{vh}(\varphi_h) := \langle f_v, \varphi_h \rangle_\Omega.$$

4.3 Analysis

Proposition 4.4 For $u, v \in V_h^\Gamma$ and the norms $\|\cdot\|_{\frac{1}{2}}, \|\cdot\|_{-\frac{1}{2}}$ defined as in (2.21), we have

$$\langle u, v \rangle_\Gamma \leq \|u\|_{\frac{1}{2}} \|v\|_{-\frac{1}{2}}. \tag{4.20}$$

Proof. Thanks to Hölder's inequality and Schwarz's inequality, we have

$$\langle u,v \rangle_\Gamma = \sum_{T\in G_h} \langle h_T^{-\frac{1}{2}}u, h_T^{\frac{1}{2}}v \rangle_{\Gamma_T} \le \sum_{T\in G_h} \langle \|h_T^{-\frac{1}{2}}u\|_{L^2(\Gamma_T)}, \|h_T^{\frac{1}{2}}v\|_{L^2(\Gamma_T)} \rangle_{\Gamma_T}$$

$$\le \left(\Sigma_{T\in G_h} h_T^{-1} \|u\|^2_{L^2(\Gamma_T)} \right)^{\frac{1}{2}} \left(\Sigma_{T\in G_h} h_T \|v\|^2_{L^2(\Gamma_T)} \right)^{\frac{1}{2}} = \|u\|_{\frac{1}{2}} \|v\|_{-\frac{1}{2}}.$$

∎

> **Proposition 4.5 — Consistency.** If w, v solve the continuous problems (4.5), (4.6) respectively then w, v also solve the discrete problems (4.17), (4.18) respectively.

Proof. For w, using the same technique given in [31, Lemma 1], we are able to get

$$a_{wh}(w, \varphi_h) = K_{wh}(\varphi_h), \quad \forall \varphi_h \in V_\Gamma^0.$$

For v solving (4.18), it's even easier because of the condition of v on Γ ($v = \nabla_{\mathbf{n}} v = 0$). After put v into a_{vh}, all terms on the interface will disappear and we get instantly the result. ∎

> **Proposition 4.6** The following estimation hold
>
> (i) $a_{wh}(w_h, \varphi_h) \le C \|\|w_h\|\|_1 \|\|\varphi_h\|\|_1$, for all $w_h \in V_h^\Gamma$ and $\varphi_h \in V_h^0$,
> (ii) $a_{wh}(w_h, w_h) \ge C \|\|w_h\|\|_1^2$, $\forall v \in V_h^\Gamma$ for all $w_h \in V_h^\Gamma$,
>
> where $\|\|\cdot\|\|_1$ defined in (4.21) and α is the diffusion coefficient in the problem (4.5).
>
> $$\|\|v_h\|\|_1^2 := \|\nabla v_h\|^2_{L^2(\Omega_{12})} + \|\{\alpha \nabla_{\mathbf{n}} v_h\}\|^2_{-1/2} + \|[\![v_h]\!]\|^2_{1/2}. \qquad (4.21)$$
>
> Thus, the discrete problem (4.17) has unique solution in V_h^Γ.

Proof. The estimations (i) and (ii) can be obtained from [31, Lemma 5]. The existence of unique solution can be deduced using Lax-Milgram theorem. ∎

> **Remark 4.1** There is slightly difference between $\|\|\cdot\|\|_1$ and $\|\|\cdot\|\|_H$ (defined in (2.21)) but it does not effect to the demonstrations.

> **Proposition 4.7** The discrete problem (4.18) has a solution $v_h \in V_h^\Gamma$.

Proof. Given $\tilde{v}_h \in V_h^\Gamma$, consider the problem : Find $v_h \in V_h^\Gamma$ such that

$$a_{vh}(v_h, \varphi_h) - \langle q(\tilde{v}_h), \varphi_h \rangle_\Omega = \langle f_v, \varphi_h \rangle_\Omega, \quad \forall \varphi_h \in V_h^0, \tag{4.22}$$

We can obtain the continuity of a_{vh} thanks to Hölder's inequality and (4.20),

$$\begin{aligned}
|a_{vh}(v_h, \varphi_h)| &\leq |\langle \beta \nabla v_h, \nabla \varphi_h \rangle_{\Omega_{12}}| + |\langle [\![v_h]\!], \{\!\{\beta \nabla_{\mathbf{n}} \varphi_h\}\!\} \rangle_\Gamma| + |\langle \{\!\{\beta \nabla_{\mathbf{n}} v_h\}\!\}, [\![\varphi_h]\!] \rangle_\Gamma| \\
&\quad + |\theta \langle \{\!\{v_h\}\!\}, \{\!\{\varphi_h\}\!\} \rangle_\Gamma| + |\theta \kappa_1 \kappa_2 \langle [\![v_h]\!], [\![\varphi_h]\!] \rangle_\Gamma| \\
&\leq C \|\nabla v_h\|_{L^2(\Omega_{12})} \|\nabla \varphi_h\|_{L^2(\Omega_{12})} + C \|[\![v_h]\!]\|_{\frac{1}{2}} \|\{\!\{\beta \nabla_{\mathbf{n}} \varphi_h\}\!\}\|_{-\frac{1}{2}} \\
&\quad + C \|\{\!\{\beta \nabla_{\mathbf{n}} v_h\}\!\}\|_{-\frac{1}{2}} \|[\![\varphi_h]\!]\|_{\frac{1}{2}} + C \|\{\!\{v_h\}\!\}\|_{L^2(\Gamma)} \|\{\!\{\varphi_h\}\!\}\|_{L^2(\Gamma)} \\
&\quad + C \|[\![v_h]\!]\|_{\frac{1}{2}} \|[\![\varphi_h]\!]\|_{\frac{1}{2}} \\
&\leq C \|\!|\!| v_h \|\!|\!|_2 \|\!|\!| \varphi_h \|\!|\!|_2.
\end{aligned}$$

Using the same technique as in the proof of [31, Lemma 5] with any $\xi > 0$, we have

$$\begin{aligned}
a_{vh}(v_h, v_h) &\geq \frac{1}{2} \|\beta^{1/2} \nabla v_h\|_{L^2(\Omega_{12})}^2 + C \|v_h\|_{L^2(\partial\Omega_1 \setminus \Gamma)}^2 \\
&\quad + \left(\frac{1}{2} - \frac{2C_I \beta_{\max}}{\xi} \right) \|\beta^{1/2} \nabla v_h\|_{L^2(\Omega_{12})}^2 + \frac{1}{\xi} \|\{\!\{\beta \nabla_{\mathbf{n}} v_h\}\!\}\|_{-\frac{1}{2}}^2 \\
&\quad + \sum_{T \in G_h} \left(\theta \kappa_1 \kappa_2 - \frac{\xi}{h_T} \right) \|[\![v_h]\!]\|_{L^2(\Gamma_T)}^2 + \theta \|\{\!\{v_h\}\!\}\|_{L^2(\Gamma)}^2.
\end{aligned}$$

By choosing $\xi = 4C_I \beta_{\max}$ (C_I is the coefficient in the inverse inequality (2.23)) and $\theta > \frac{\xi}{h_T \kappa_1 \kappa_2}$, we have

$$a_{vh}(v_h, v_h) \geq C \|\!|\!| v_h \|\!|\!|_2^2, \tag{4.23}$$

where $\|\!|\!| \cdot \|\!|\!|_2$ defined in (4.24) (β is the diffusion coefficient in problem (4.6)).

$$\boxed{\|\!|\!| v_h \|\!|\!|_2^2 := \|\nabla v_h\|_{L^2(\Omega_{12})}^2 + \|\{\!\{\beta \nabla_{\mathbf{n}} v_h\}\!\}\|_{-1/2}^2 + \|[\![v_h]\!]\|_{1/2}^2 + \|\{\!\{v_h\}\!\}\|_{L^2(\Gamma)}^2} \tag{4.24}$$

Thanks to Lax-Milgram theorem, easily to check that (4.22) has unique solution in V_h^Γ. From this, we define operator T by

$$\begin{aligned}
T \quad : \quad V_h^\Gamma &\longrightarrow \quad V_h^\Gamma \\
\tilde{v}_h &\longmapsto \quad T(\tilde{v}_h) = v_h \text{ solving (4.22)}
\end{aligned}$$

Owing to (4.23),

$$a_{vh}(v_h, v_h) \geq C \|\!|\!| v_h \|\!|\!|_2^2 \geq C \|\nabla v_h\|_{L^2(\Omega_{12})}^2.$$

Besides that,

$$a_{vh}(v_h, v_h) = \langle q(\tilde{v}_h), v_h \rangle_{\Omega_{12}} + \langle f_v, v_h \rangle_{\Omega} \leq |\langle q(\tilde{v}_h), v_h \rangle_{\Omega_{12}} + \langle f_v, v_h \rangle_{\Omega}|$$
$$\leq \|q\|_{L^2(\Omega)} \|v_h\|_{L^2(\Omega)} + \|f_v\|_{L^2(\Omega)} \|v_h\|_{L^2(\Omega)} \leq C \|\nabla v_h\|_{L^2(\Omega_{12})}.$$

Therefore $\|\nabla v_h\|_{L^2(\Omega_{12})} \leq C$ (thanks to Poincaré's inequality), then T is thus bounded in V_h^{Γ}. Take a R "large enough", T will map $B_R = \{v_h \in V_h^{\Gamma} : \|v_h\|_{L^2(\Omega_{12})} \leq R\}$ to B_R.

It's enough to prove that T is continuous to conclude by the Brouwer fixed point theorem (cf. [61]). Indeed, let $\{\tilde{v}_h^n\}_n$ be a sequence in V_h^{Γ} such that $\tilde{v}_h^n \to \tilde{v}_h$. Putting $v_h^n = T(\tilde{v}_h^n)$. Because T is bounded in V_h^{Γ}, thanks to Bolzano-Weierstrass theorem, there exists a subsequence, also denoted v_h^n, converges to a quantity so-called $z_h \in V_h^{\Gamma}$. What we need to do now is to prove that z_h is a solution of (4.22). Take $\varphi_h \in V_h^0$, we have

$$a_{vh}(v_h^n, \varphi_h) - \langle q(\tilde{v}_h^n), \varphi_h \rangle_{\Omega_{12}} = \langle f_v, \varphi_h \rangle_{\Omega}.$$

With assumptions like in Propotition 4.3, $\langle q(\tilde{v}_h^n), \varphi_h \rangle_{\Omega_{12}} \to \langle q(\tilde{v}_h), \varphi_h \rangle_{\Omega_{12}}$. Moreover, $v_h^n \to z_h$ implies

$$\langle \beta \nabla v_h^n, \nabla \varphi_h \rangle_{\Omega_{12}} \to \langle \beta \nabla z_h, \nabla \varphi_h \rangle_{\Omega_{12}},$$

$$\|[\![v_h^n]\!] - [\![z_h]\!]\|_{L^2(\Gamma)} = \|[\![v_h^n - z_h]\!]\|_{L^2(\Gamma)} \sum_{i=1}^{2} \|(v_h^n - z_h)|_{\Omega_i}\|_{L^2(\Gamma)}$$

$$\leq C \sum_{i=1}^{2} \|(v_h^n - z_h)|_{\Omega_i}\|_{L^2(\Omega_i)} \to 0,$$

$$\|\{\!\{v_h^n\}\!\} - \{\!\{z_h\}\!\}\|_{L^2(\Gamma)} = \|\sum_{i=1}^{2} \kappa_i (v_h^n - z_h)|_{\Omega_i}\|_{L^2(\Gamma)} \leq C \sum_{i=1}^{2} \|(v_h^n - z_h)|_{\Omega_i}\|_{L^2(\Omega_i)} \to 0.$$

$$\|\{\!\{\beta \nabla_{\mathbf{n}} v_h^n\}\!\} - \{\!\{\beta \nabla_{\mathbf{n}} z_h\}\!\}\|_{L^2(\Gamma)}^2$$

$$\leq C \sum_{i=1}^{2} \|\kappa_i \nabla_{\mathbf{n}} (v_h^n - z_h)|_{\Omega_i}\|_{L^2(\Gamma)}^2 \leq C \sum_{i=1}^{2} \sum_{K} \|\kappa_i \nabla_{\mathbf{n}} (v_h^n - z_h)|_{\Omega_i}\|_{L^2(\Gamma_K)}^2$$

$$\leq C \sum_{i=1}^{2} \sum_{K} \kappa_i^2 |\Gamma_K| |\nabla_{\mathbf{n}} (v_h^n - z_h)|_{\Omega_i}|^2 = C \sum_{i=1}^{2} \sum_{K} \frac{|K_i||\Gamma_K|}{|K|^2} \|\nabla_{\mathbf{n}} (v_h^n - z_h)|_{\Omega_i}\|_{L^2(K_i)}^2$$

$$\leq C \sum_{i=1}^{2} \sum_{K} \|\nabla_{\mathbf{n}} (v_h^n - z_h)|_{\Omega_i}\|_{L^2(K_i)}^2 = C \sum_{i=1}^{2} \|\nabla_{\mathbf{n}} (v_h^n - z_h)|_{\Omega_i}\|_{L^2(\Omega_i)}^2 \to 0.$$

In the estimate $\|\{\!\{\beta \nabla_{\mathbf{n}} v_h^n\}\!\} - \{\!\{\beta \nabla_{\mathbf{n}} z_h\}\!\}\|_{L^2(\Gamma)}^2$, we have used the fact that $|\Gamma_K| \leq h_K, |K_i| \leq h_K^2, |K| \geq C h_K^2$. And therefore, $a_{vh}(v_h^n, \varphi_h) \to a_{vh}(z_h, \varphi_h)$. Finally, we will get,

$$a_{vh}(z_h, \varphi_h) - \langle q(\tilde{v}_h), \varphi_h \rangle_{\Omega_{12}} = \langle f_v, \varphi_h \rangle_{\Omega},$$

or z_h solves (4.22). Sum up, we conclude that T admits a fixed point $z_h = T(z_h)$ which solves the problem (4.18). ∎

4.4 The convergence

In this section, we are going to show that the solutions of discrete problems obtained from (4.17) and (4.18) will converge to solutions of weak problems (4.9) and (4.11) respectively. In order to do that, we need to define some operators dertermined on the interface and also their convergence result in V_h^Γ. After that, we split the forms of (4.17) (4.18) into separated terms. On each term, we will prove the convergence to the corresponding one in the form of (4.9), (4.11). These results are obtained thanks to the operators we define earlier.

Definition 4.1 — Lifting operator. For $v_h \in V_h^\Gamma$, let $\mathscr{L}_h : L^2(\Gamma) \to [V_h^\Gamma]^2$ such that

$$\forall \varphi \in [V_h^0]^2, \quad \langle \mathscr{L}_h(\llbracket v_h \rrbracket), \varphi \rangle_{\Omega_{12}} := \langle \{\!\{\varphi\}\!\} \cdot \mathbf{n}, \llbracket v_h \rrbracket \rangle_\Gamma. \qquad (4.25)$$

We observe that the support of $\mathscr{L}_h$ consists of two subdomains of which Γ is part of the boundary $\partial \Omega_i$, or

$$\mathrm{supp}(\mathscr{L}_h) = \overline{\Omega}_1 \cup \overline{\Omega}_2 = \overline{\Omega}.$$

The following discrete gradient operators will play an important role in the analysis.

Definition 4.2 — Discrete gradient operators. For all $v_h \in V_h^\Gamma$, let $\mathscr{G}_h : V_h^\Gamma \to [V_h^\Gamma]^2$ such that,

$$\mathscr{G}_h(v_h) := \nabla v_h - \mathscr{L}_h(\llbracket v_h \rrbracket). \qquad (4.26)$$

Definition 4.3 — Symbolic notations. It's necessary to define symbolic spaces V_σ, V_σ^0 as follows: $V_\sigma := H^1(\Omega), V_\sigma^0 := H_0^1(\Omega)$ if $\sigma = \alpha$ and $V_\sigma := V, V_\sigma^0 := V_0$ if $\sigma = \beta$. We also define a symbolic norm $\|\!|\cdot|\!\|$ which stands for $\|\!|\cdot|\!\|_i, i = 1,2$ given as below

$$\|\!|z|\!\|^2 = \|\nabla z\|_{L^2(\Omega_{12})}^2 + \|\llbracket z \rrbracket\|_{\frac{1}{2}}^2 + \|\{\!\{\sigma \nabla_\mathbf{n} z\}\!\}\|_{-\frac{1}{2}}^2 + \mu \|\{\!\{z\}\!\}\|_{L^2(\Gamma)}^2, \qquad (4.27)$$

where $\sigma = \alpha, \sigma = \beta$ are corresponding to $\|\!|\cdot|\!\|_1, \|\!|\cdot|\!\|_2$ respectively and $\mu = 0, \mu = 1$ are corresponding to $\|\!|\cdot|\!\|_1, \|\!|\cdot|\!\|_2$ respectively.

Theorem 4.1 Let $I_h^* : V_\sigma \cap H^2(\Omega_{12}) \to V_h^\Gamma$ be the interpolation operator defined in Definition 2.4, then

$$\|\|v - I_h^* v\|\| \leq Ch\|v\|_{L^2(\Omega_{12})}, \quad \forall v \in V_\sigma \cap H^2(\Omega_{12}). \tag{4.28}$$

Proof. Look back to definition of norm $\|\|\cdot\|\|$, there are 4 terms. For the first 3 terms, using the result given from the proof of [31, Theorem 2], we have

$$\|\nabla z\|^2_{L^2(\Omega_{12})} + \|[\![z]\!]\|^2_{\frac{1}{2}} + \|\{\!\!\{\sigma \nabla_{\mathbf{n}} z\}\!\!\}\|^2_{-\frac{1}{2}} \leq Ch^2\|v\|^2_{L^2(\Omega_{12})}, \quad \forall v \in V_\sigma \cap H^2(\Omega_{12}),$$

where $z = v - I_h^* v$. For the last term,

$$\|\{\!\!\{z\}\!\!\}\|_{L^2(\Gamma)} = \|\sum_{i=1}^2 \kappa_i z_i\|_{L^2(\Gamma)} \leq C\sum_{i=1}^2 \|z_i\|_{L^2(\Gamma)} \leq C\sum_{i=1}^2 \|z_i\|_{L^2(\partial\Omega_i)}$$

$$\leq C\sum_{i=1}^2 \|\nabla z_i\|_{L^2(\Omega_i)} = C\|\nabla z\|_{L^2(\Omega_{12})} \leq Ch\|v\|_{L^2(\Omega_{12})}.$$

∎

> **Proposition 4.8** Let $\|\cdot\|_{-\frac{1}{2}}$ and $\mathscr{L}_h$ be defined in (2.21) and (4.25) respectively, we have the boundedness for the lifting operator $\mathscr{L}_h$ as following
>
> $$\|\mathscr{L}_h([\![v_h]\!])\|_{[L^2(\Omega_{12})]^2} \leq C\|[\![v_h]\!]\|_{\frac{1}{2}}, \quad \forall v_h \in V_h^\Gamma. \tag{4.29}$$

Proof. Coming from the left hand side of (4.29), we have

$$\|\mathscr{L}_h([\![v_h]\!])\|^2_{[L^2(\Omega_{12})]^2} = \langle \{\!\!\{\mathscr{L}_h([\![v_h]\!])\}\!\!\} \cdot \mathbf{n}, [\![v_h]\!]\rangle_\Gamma \leq C\|\{\!\!\{\mathscr{L}_h([\![v_h]\!])\}\!\!\}\|_{[L^2(\Gamma)]^2}\|[\![v_h]\!]\|_{L^2(\Gamma)}$$

$$\leq C\left(\kappa_1\|\mathscr{L}_h([\![v_h]\!])|_{\Omega_1}\|_{[L^2(\Gamma)]^2} + \kappa_2\|\mathscr{L}_h([\![v_h]\!])|_{\Omega_2}\|_{[L^2(\Gamma)]^2}\right)\|[\![v_h]\!]\|_{L^2(\Gamma)}$$

$$\leq C\left(\kappa_1\|\mathscr{L}_h([\![v_h]\!])|_{\Omega_1}\|_{[L^2(\partial\Omega_1)]^2} + \kappa_2\|\mathscr{L}_h([\![v_h]\!])|_{\Omega_2}\|_{[L^2(\partial\Omega_2)]^2}\right)\|[\![v_h]\!]\|_{L^2(\Gamma)}$$

$$\overset{(\mathscr{A})}{\leq} C\left(\kappa_1 h^{-\frac{1}{2}}\|\mathscr{L}_h([\![v_h]\!])|_{\Omega_1}\|_{[L^2(\Omega_1)]^2} + \kappa_2 h^{-\frac{1}{2}}\|\mathscr{L}_h([\![v_h]\!])|_{\Omega_2}\|_{[L^2(\Omega_2)]^2}\right)\|[\![v_h]\!]\|_{L^2(\Gamma)}$$

$$\leq C\|\mathscr{L}_h([\![v_h]\!])\|_{[L^2(\Omega_{12})]^2} h^{-\frac{1}{2}}\|[\![v_h]\!]\|_{L^2(\Gamma)}$$

$$\leq C\|\mathscr{L}_h([\![v_h]\!])\|_{[L^2(\Omega_{12})]^2}\|[\![v_h]\!]\|_{\frac{1}{2}}.$$

In above estimate, the reason $(\mathscr{A})$ comes from following estimate owing to the trace theorem and [59, Theorem 1.3],

$$\|v_h\|_{L^2(\partial\Omega_i)} \leq C\|\nabla v_h\|_{L^2(\Omega_i)} \leq Cd_i^{-1}\|v_h\|_{L^2(\Omega_i)} \leq Ch^{-\frac{1}{2}}\|v_h\|_{L^2(\Omega_i)}, \tag{4.30}$$

where $d_i := \sup_{x,y \in \Omega_i}\|x - y\|$ which is the diameter of the domain Ω_i satisfies an assumption on the domain that there exists $C_i > 0$ such that $d_i \geq C_i$. ∎

Proposition 4.9 For the discrete gradient operator $\mathscr{G}_h$ defined in (4.26),

    a) $\forall v_h \in V_h^\Gamma, \quad \|\mathscr{G}_h(v_h)\|_{[L^2(\Omega_{12})]^2} \leq C\||v_h\||.$
    b) $\mathscr{G}_h(I_h^*\varphi) \to \nabla\varphi$ strongly in $[L^2(\Omega_{12})]^2$ for all $\varphi \in V_\sigma^0$

Proof. a) Using the definition (4.26) of $\mathscr{G}_h$ and the triangle inequality coupling with the boundedness (4.29) of $\mathscr{L}_h$, we have

$$
\begin{aligned}
\|\mathscr{G}_h(v_h)\|_{[L^2(\Omega_{12})]^2} &= \|\nabla v_h - \mathscr{L}_h(\llbracket v_h \rrbracket)\|_{[L^2(\Omega_{12})]^2} \\
&\leq C\|\nabla v_h\|_{L^2(\Omega_{12})} + C\|\mathscr{L}_h(\llbracket v_h \rrbracket)\|_{[L^2(\Omega_{12})]^2} \\
&\leq C\|\nabla v_h\|_{L^2(\Omega_{12})} + C\|\llbracket v_h \rrbracket\|_{\frac{1}{2}} \leq C\||v_h\||.
\end{aligned}
$$

b) For all $\varphi \in V_\sigma^0$,

$$
\begin{aligned}
\|\mathscr{G}_h(I_h^*\varphi) - \nabla\varphi\|_{[L^2(\Omega_{12})]^2} &\leq \|\nabla(I_h^*\varphi) - \mathscr{L}_h(\llbracket I_h^*\varphi \rrbracket) - \nabla\varphi\|_{[L^2(\Omega_{12})]^2} \\
&= \|\nabla(I_h^*\varphi - \varphi) - \mathscr{L}_h(\llbracket(I_h^*\varphi - \varphi)\rrbracket)\|_{[L^2(\Omega_{12})]^2} \\
&\leq C\|\nabla(I_h^*\varphi - \varphi)\|_{L^2(\Omega_{12})} + C\|\mathscr{L}_h(\llbracket(I_h^*\varphi - \varphi)\rrbracket)\|_{[L^2(\Omega)]^2} \\
&\leq C\||I_h^*\varphi - \varphi\|| + C\|\llbracket(I_h^*\varphi - \varphi)\rrbracket\|_{\frac{1}{2}} \\
&\leq C\||I_h^*\varphi - \varphi\|| \leq Ch\|\varphi\|_{L^2(\Omega_{12})}.
\end{aligned}
$$

∎

Proposition 4.10 Let $\{v_h\}_h$ be a sequence in V_h^Γ and assume that this sequence is bounded in the $\||\cdot\||$-norm. Then, the family $\{v_h\}_h$ is relatively compact in $L^2(\Omega)$.

Proof. We will borrow the idea of proofs in the work of [25] and [28]. While the authors of [25] work on Discontinuous Galerkin Method in which they consider the discontinuity throughout faces of all elements of the mesh and the authors of [28] work on Finite Volume Method, our work will focus only on the zone around the interface.

For $v \in L^1(\mathbb{R}^2)$, define a space $\mathrm{BV} := \{v \in L^1(\mathbb{R}^2) : \|v\|_{\mathrm{BV}} < +\infty\}$ where

$$
\|v\|_{\mathrm{BV}} := \sum_{i=1}^{2} \sup\left\{ \int_{\mathbb{R}^2} v\partial_i\varphi \, dx; \varphi \in C_c^\infty(\mathbb{R}^2), \|\varphi\|_{L^\infty(\mathbb{R}^2)} \leq 1 \right\}.
$$

Extending the functions v_h by zero outside Ω and for all $\varphi \in C_c^\infty(\mathbb{R}^2)$ with $\|\varphi\|_{L^\infty(\mathbb{R}^2)} \leq 1$, integrating by parts gives us

$$
\int_{\mathbb{R}^2} (v_h)\partial_i\varphi \, dx = \int_\Omega (v_h)\partial_i\varphi \, dx = \int_{\Omega_1} (v_h)\partial_i\varphi \, dx + \int_{\Omega_2} (v_h)\partial_i\varphi \, dx
$$

$$= -\int_{\Omega_{12}} (e_i \cdot \nabla(v_h)) \varphi \, dx + \sum_{T \in G_h} \int_{\Gamma_T} (e_i \cdot \mathbf{n}) [\![v_h]\!] \varphi \, ds.$$

Hölder's inequality and the fact that $\|\varphi\|_{L^\infty(\mathbb{R}^2)} \leq 1$ will give us

$$-\int_{\Omega_{12}} (e_i \cdot \nabla(v_h)) \varphi \, dx \leq \|\nabla(v_h)\|_{L^1(\Omega_{12})} \|\varphi\|_{L^\infty(\Omega_{12})} \leq \|\nabla(v_h)\|_{L^1(\Omega_{12})},$$

$$\sum_{T \in G_h} \int_{\Gamma_T} (e_i \cdot \mathbf{n}) [\![v_h]\!] \varphi \, ds \leq \sum_{T \in G_h} \|[\![v_h]\!]\|_{L^1(\Gamma_T)} \|\varphi\|_{L^\infty(\Gamma_T)} \leq \sum_{T \in G_h} \|[\![v_h]\!]\|_{L^1(\Gamma_T)}.$$

Applying Hölder's inequality again,

$$\|\nabla v_h\|_{L^1(\Omega_{12})} \leq \|1\|_{L^2(\Omega_{12})} \|\nabla v_h\|_{L^2(\Omega_{12})} \leq C \|\nabla v_h\|_{L^2(\Omega_{12})}$$

$$\sum_{T \in G_h} \|[\![v_h]\!]\|_{L^1(\Gamma_T)} = \sum_{T \in G_h} \|h_T^{\frac{1}{2}} h_T^{-\frac{1}{2}} [\![v_h]\!]\|_{L^1(\Gamma_T)}$$

$$\leq \left(\sum_{T \in G_h} h_T \|1\|_{L^2(\Gamma_T)}^2 \right)^{\frac{1}{2}} \left(\sum_{T \in G_h} h_T^{-1} \|[\![v_h]\!]\|_{L^2(\Gamma_T)}^2 \right)^{\frac{1}{2}}$$

$$\leq C \|[\![v_h]\!]\|_{\frac{1}{2}},$$

in which we have used that $|\Gamma_T| \leq h_T$ and the non-degenerate property of the mesh, $h_T^2 \leq C|T|$. Therefore, $\int_{\mathbb{R}^2} (v_h) \partial_i \varphi \, dx \leq C \||v_h\||$ or we have

$$\|\sigma v_h\|_{\mathrm{BV}} \leq C \||v_h\|| \leq C.$$

From [28], for all $y \in \mathbb{R}^2$,

$$C \|v_h(\cdot + y) - v_h\|_{L^1(\mathbb{R}^2)} \leq |y| \|\sigma v_h\|_{\mathrm{BV}} \leq C|y|,$$

where $|y|$ is the Euclidean norm of y. From this and thanks to Kolmogorov's Compactness Criterion, we have that $\{v_h\}_h$ is relatively compact in $L^1(\mathbb{R}^2)$. Besides that, Poincaré's inequality helps us

$$\|v_h\|_{L^2(\mathbb{R}^2)} = \|v_h\|_{L^2(\Omega)} = \|v_h\|_{L^2(\Omega_{12})} \leq C \|\nabla v_h\|_{L^2(\Omega_{12})} \leq C \||v_h\|| \leq C,$$

or $\{v_h\}_h$ is also bounded in $L^2(\mathbb{R}^2)$, hence it is also relatively compact in $L^2(\mathbb{R}^2)$. Finally, we have $\{v_h\}_h$ is relatively compact in $L^2(\Omega)$ because v_h has been extended by zero outside Ω. ∎

Theorem 4.2 Let $\{v_h\}_h$ be a sequence in V_h^Γ. Assume that this sequence is bounded in $\||\cdot\||$-norm. There exists a function $v \in V_\sigma$ such that as $h \to 0$, up to a subsequence, $v_h \to v$ strongly in $L^2(\Omega)$ and $\mathscr{G}_h(v_h) \rightharpoonup \nabla v$ weakly in $[L^2(\Omega)]^2$.

Proof. Thanks to Propotition 4.10 and Rellich's theorem, there exists a function $v \in L^2(\Omega)$ and a subsequence, also denoted by v_h, such that $v_h \to v$ strongly in $L^2(\Omega)$. Moreover, Propotition 4.9 gives us the boundedness of $\mathscr{G}_h$ in $[L^2(\Omega)]^2$, thus there exists a new subsequence, again denoted as v_h, and $\mathbf{w} \in [L^2(\Omega)]^2$ such that $\mathscr{G}_h(v_h) \rightharpoonup \mathbf{w}$ weakly in $[L^2(\Omega)]^2$. What we need to do is to prove that $\mathbf{w} = \nabla v$. Indeed, for all $\varphi \in [C_c^\infty(\Omega)]^2$ (note that, $[\![\varphi]\!] = 0$ on Γ and $\varphi = 0$ on $\partial\Omega$),

$$
\begin{aligned}
\langle \mathscr{G}_h(v_h), \varphi \rangle_{\Omega_{12}} &= \langle \nabla v_h - \mathscr{L}_h([\![v_h]\!]), \varphi \rangle_{\Omega_{12}} \\
&= \langle \nabla v_h, \varphi \rangle_\Omega - \langle \mathscr{L}_h([\![v_h]\!]), \varphi \rangle_{\Omega_{12}} \\
&= -\langle v_h, \nabla \cdot \varphi \rangle_\Omega + \int_\Gamma [\![v_h(\varphi \cdot \mathbf{n})]\!] \, ds - \langle \mathscr{L}_h([\![v_h]\!]), \varphi \rangle_{\Omega_{12}} \\
&= -\langle v_h, \nabla \cdot \varphi \rangle_\Omega + \langle [\![v_h]\!], \{\!\!\{\varphi\}\!\!\} \cdot \mathbf{n} \rangle_\Gamma - \langle \mathscr{L}_h([\![v_h]\!]), \varphi \rangle_{\Omega_{12}} \\
&= -\langle v_h, \nabla \cdot \varphi \rangle_\Omega.
\end{aligned}
$$

Observe that when $h \to 0$, $\langle v_h, \nabla \cdot \varphi \rangle_\Omega \to \langle v, \nabla \cdot \varphi \rangle_\Omega$ because of the strong convergence of v_h in $L^2(\Omega)$. As a result,

$$
\langle \mathbf{w}, \varphi \rangle_\Omega = \lim_{h \to 0} \langle \mathscr{G}_h(v_h), \varphi \rangle_{\Omega_{12}} = -\langle v, \nabla \cdot \varphi \rangle_\Omega, \quad \forall \varphi \in [C_c^\infty(\Omega)]^2.
$$

If we can prove that $[\![v]\!] = 0$, we can obtain $\mathbf{w} = \nabla v$, hence $v \in V_\sigma$. Indeed, considering the relation,

$$
\begin{aligned}
\big| \|[\![v_h]\!]\|_{L^2(\Gamma)} - \|[\![v]\!]\|_{L^2(\Gamma)} \big| &\leq \|[\![v_h - v]\!]\|_{L^2(\Gamma)} \\
&\leq \|(v_h - v)|_{\Omega_1}\|_{L^2(\Gamma)} + \|(v_h - v)|_{\Omega_2}\|_{L^2(\Gamma)}
\end{aligned}
$$

Using the relation (4.30), we have

$$
\begin{aligned}
\|(v_h - v)|_{\Omega_i}\|_{L^2(\Gamma)} &\leq \|(v_h - v)|_{\Omega_i}\|_{L^2(\partial\Omega_i)} \\
&\leq Cd_i^{-1} \|(v_h - v)|_{\Omega_i}\|_{L^2(\Omega_i)} \leq C \|(v_h - v)|_{\Omega_i}\|_{L^2(\Omega_i)}
\end{aligned}
$$

Thus,

$$
\big| \|[\![v_h]\!]\|_{L^2(\Gamma)} - \|[\![v]\!]\|_{L^2(\Gamma)} \big| \leq C \sum_i \|(v_h - v)|_{\Omega_i}\|_{L^2(\Omega_i)},
$$

which tends to zero because $v_h \to v$ in $L^2(\Omega)$. This implies

$$
\|[\![v_h]\!]\|_{L^2(\Gamma)} \to \|[\![v]\!]\|_{L^2(\Gamma)}. \tag{4.31}
$$

Besides that, v_h is bounded in $\|\!|\cdot|\!\|$-norm,

$$
h^{-1}\|[\![v_h]\!]\|_{L^2(\Gamma)}^2 = \sum_{T \in G_h} h^{-1}\|[\![v_h]\!]\|_{L^2(\Gamma_T)}^2 \leq \sum_{T \in G_h} h_T^{-1}\|[\![v_h]\!]\|_{L^2(\Gamma_T)}^2
$$

$$= \|[\![v_h]\!]\|_{\frac{1}{2}}^2 \leq \||v_h\||^2 \leq C,$$

which yields that as $h \to 0$,

$$\|[\![v_h]\!]\|_{L^2(\Gamma)} \to 0. \tag{4.32}$$

From (4.31), (4.32) we have $\|[\![v]\!]\|_{L^2(\Gamma)} = 0$. This yields $[\![v]\!] = 0$.   ■

Now, we have all needed tools to consider the convergence of the solutions of discrete problems to a solution of the weak problems.

> **Theorem 4.3** For $\{w_h\}_h, \{v_h\}_h$ be the sequence of discrete solutions generated by solving discrete problems (4.17),(4.18) respectively, there exist solutions w, v solving (4.9),(4.11) respectively such that as $h \to 0$, $(w_h, v_h) \to (w, v)$ strongly in $L^2(\Omega)$.

Proof. Using Propotition 4.6, (4.23), Hölder's inequality and Poincaré's inequality, it is inferred that,

$$\||w_h\||_1^2 \leq C a_{wh}(w_h, w_h) = K_{wh}(w_h) \leq C\|w_h\|_{L^2(\Omega)} \leq C\|\nabla w_h\|_{L^2(\Omega)} \leq C\||w_h\||,$$

$$\begin{aligned} \||v_h\||_2^2 &\leq C a_{vh}(v_h, v_h) = C\langle q(v_h), v_h \rangle_\Omega + C\langle f_v, v_h \rangle_\Omega \\ &\leq \|q\|_{L^2(\Omega)}\|v_h\|_{L^2(\Omega)} + \|f_v\|_{L^2(\Omega)}\|v_h\|_{L^2(\Omega)} \\ &\leq C\|k\|_{L^\infty(\Omega)}(\mathrm{mesh}(\Omega))^{1/2}\|\nabla v_h\|_{L^2(\Omega)} + C\|\nabla v_h\|_{L^2(\Omega)} \leq C\||v_h\||_2. \end{aligned}$$

Hence, $\{w_h\}_h, \{v_h\}_h$ are bounded in $\||\cdot\||_1$-norm and $\||\cdot\||_2$-norm respectively. Thanks to Theorem 4.2, there exist $w^* \in H^1(\Omega)$ and $v^* \in V$ such that, as $h \to 0$, up to a subsequence, $w_h \to w^*, v_h \to v^*$ strongly in $L^2(\Omega)$ and $\mathscr{G}_h(w_h) \rightharpoonup \nabla w^*, \mathscr{G}_h(v_h) \rightharpoonup \nabla v^*$ weakly in $[L^2(\Omega)]^2$.

We want to prove that w^* and v^* are solutions of problems (4.9),(4.11) respectively. In deed,

(i) For all $\varphi \in H_0^1(\Omega)$,

$$a_{wh}(w_h, I_h^* \varphi) = \langle \alpha^{\frac{1}{2}} \mathscr{G}_h(w_h), \alpha^{\frac{1}{2}} \mathscr{G}_h(I_h^* \varphi) \rangle_{\Omega_{12}} + j_{wh}(w_h, I_h^* \varphi) = F_1 + F_2,$$

where

$$\begin{aligned} F_1 &= \langle \alpha^{\frac{1}{2}} \mathscr{G}_h(w_h), \alpha^{\frac{1}{2}} \mathscr{G}_h(I_h^* \varphi) \rangle_{\Omega_{12}}, \\ F_2 &= j_{wh}(w_h, I_h^* \varphi) = \zeta \langle [\![w_h]\!], [\![I_h^* \varphi]\!] \rangle_\Gamma - \langle \alpha^{\frac{1}{2}} \mathscr{L}_h([\![w_h]\!]), \alpha^{\frac{1}{2}} \mathscr{L}_h([\![I_h^* \varphi]\!]) \rangle_{\Omega_{12}}. \end{aligned}$$

From the weak convergence of $\mathscr{G}_h(w_h)$ to ∇w^* and the strong convergence of $\mathscr{G}_h(I_h^* \varphi)$ to $\nabla \varphi$ (cf. Lemma 4.9), as $h \to 0$, we have $F_1 \to \langle \alpha \nabla w^*, \nabla \varphi \rangle_{\Omega_{12}}$. We show that $F_2 \to 0$ also. Indeed,

$$|\zeta\langle[\![w_h]\!],[\![I_h^*\varphi]\!]\rangle_\Gamma| \le C\|[\![w_h]\!]\|_{\frac{1}{2}}\|[\![I_h^*\varphi]\!]\|_{\frac{1}{2}} \le C\|\!|w_h\|\!|_1\|[\![I_h^*\varphi-\varphi]\!]\|_{\frac{1}{2}}$$
$$\le C\|\!|w_h\|\!|_1\|\!|I_h^*\varphi-\varphi\|\!|_1 \to 0,$$
$$|\langle\alpha^{\frac{1}{2}}\mathscr{L}_h([\![w_h]\!]),\alpha^{\frac{1}{2}}\mathscr{L}_h([\![I_h^*\varphi]\!])\rangle_\Gamma| \le C\|\mathscr{L}_h([\![w_h]\!])\|_{[L^2(\Omega_{12})]^2}\|\mathscr{L}_h([\![I_h^*\varphi]\!])\|_{[L^2(\Omega_{12})]^2}$$
$$\le C\|[\![w_h]\!]\|_{\frac{1}{2}}\|[\![I_h^*\varphi]\!]\|_{\frac{1}{2}} \to 0.$$

because w_h is bounded in $\|\!|\cdot\|\!|_1$ and owing to Theorem 4.1 and Propotition 4.8 .

Besides that, we also have $K_{wh}(I_h^*\varphi) \to K_{wh}(\varphi)$. It's because

$$|K_{wh}(I_h^*\varphi) - K_{wh}(\varphi)| = |K_{wh}(I_h^*\varphi-\varphi)| \le \|f_w\|_{1,\infty,\Omega}\|I_h^*\varphi-\varphi\|_{L^2(\Omega)}$$
$$\le C\|\nabla(I_h^*\varphi-\varphi)\|_{L^2(\Omega)} \le C\|\!|I_h^*\varphi-\varphi\|\!|_1 \to 0.$$

In short, for all $\varphi \in H_0^1(\Omega)$,

$$a_w(w^*,\varphi) \leftarrow a_{wh}(w_h,I_h^*\varphi) = K_w(I_h^*\varphi) \to K_w(\varphi). \tag{4.33}$$

The remaining thing to be verified is the boundary condition $w^* = \bar{w}$ on $\partial\Omega$. It's easy to obtained thanks to the strong convergence of w_h to w in $L^2(\Omega)$ and the trace theorem.

In one word, w^* is a solution of discrete problem (4.17). Since the solution of this problem is unique (cf. Propotition 4.3), we also have that the whole sequence $\{w_h\}_h$ strongly converges to w^* in $L^2(\Omega)$.

(ii) Similarly, for all $\varphi \in V_0$,

$$a_{vh}(v_h,I_h^*\varphi) = \langle\beta^{\frac{1}{2}}\mathscr{G}_h(v_h),\beta^{\frac{1}{2}}\mathscr{G}_h(I_h^*\varphi)\rangle_{\Omega_{12}} + j_{vh}(v_h,I_h^*\varphi),$$

where

$$j_{vh}(v_h,I_h^*\varphi) = \theta\kappa_1\kappa_2\langle[\![v_h]\!],[\![I_h^*\varphi]\!]\rangle_\Gamma + \theta\langle\{\!\{v_h\}\!\},\{\!\{I_h^*\varphi\}\!\}\rangle_\Gamma$$
$$- \langle\beta^{\frac{1}{2}}\mathscr{L}_h([\![v_h]\!]),\beta^{\frac{1}{2}}\mathscr{L}_h([\![I_h^*\varphi]\!])\rangle_{\Omega_{12}}.$$

With the same technique as in a_{wh} in a notice that,

$$|\langle\{\!\{v_h\}\!\},\{\!\{I_h^*\varphi\}\!\}\rangle_\Gamma| = |\langle\{\!\{v_h\}\!\},\{\!\{\varphi-I_h^*\varphi\}\!\}\rangle_\Gamma| \quad (\{\!\{\varphi\}\!\} = \varphi = 0 \text{ on } \Gamma)$$
$$\le C\|\!|v_h\|\!|_2\|\!|\varphi-I_h^*\varphi\|\!|_2 \to 0,$$

we have

$$a_{vh}(v_h,I_h^*\varphi) \to \langle\beta\nabla v^*,\nabla\varphi\rangle_{\Omega_{12}}.$$

We want also that $\langle q(v_h),I_h^*\varphi\rangle_\Omega \to \langle q(v^*),\varphi\rangle_\Omega$. Indeed,

$$|\langle q(v_h), I_h^* \varphi \rangle_\Omega - \langle q(v^*), \varphi \rangle_\Omega|$$
$$= |\langle q(v_h), I_h^* \varphi \rangle_\Omega - \langle q(v^*), I_h^* \varphi \rangle_\Omega + \langle q(v^*), I_h^* \varphi \rangle_\Omega - \langle q(v^*), \varphi \rangle_\Omega|$$
$$\leq |\langle q(v_h) - q(v^*), I_h^* \varphi \rangle_\Omega| + |\langle q(v^*), I_h^* \varphi - \varphi \rangle_\Omega|.$$

With the same assumptions for q as in Propotition 4.3, $q(x, v_h(x)) \to q(x, v^*(x))$, a.e. in $L^2(\Omega)$ and we will get $\langle q(x, v_h(x)), I_h^* \varphi(x) \rangle_\Omega \to \langle q(x, v^*(x)), I_h^* \varphi(x) \rangle_\Omega$ thanks to convergence dominated theorem. Moreover,

$$|\langle q(v^*), I_h^* \varphi - \varphi \rangle_\Omega| \leq \|q\|_{L^2(\Omega)} \|I_h^* \varphi - \varphi\|_{L^2(\Omega)} \leq \|k\|_{L^\infty} (\text{mes}(\Omega))^{1/2} \|I_h^* \varphi - \varphi\|_{L^2(\Omega)}$$
$$\leq C\|\nabla(I_h^* \varphi - \varphi)\|_{L^2(\Omega)} \leq C\|I_h^* \varphi - \varphi\|_2 \to 0.$$

Again, we do similarly get $K_{vh}(I_h^* \varphi) \to K_v(\varphi)$ and $v^* = \bar{v}$ on $\partial\Omega$. We get finally that v^* satisfies

$$\langle \beta \nabla v^*, \nabla \varphi \rangle_{\Omega_{12}} - \langle q(v^*), \varphi \rangle_\Omega \leftarrow a_{vh}(v_h, I_h^* \varphi) - \langle q(v_h), I_h^* \varphi \rangle_\Omega = K_{v_h}(I_h^* \varphi) \to K_v(\varphi),$$

or v^* is a solution of discrete problem (4.18). Since the solution of (4.18) is unique (cf. Propotition 4.3), we also have the strong convergence of v_h to v^* in $L^2(\Omega)$. ∎

4.5 A numerical test case

We consider the problem (4.2) in which a domain $\Omega = [0,1] \times [0,1]$ with an interface is a circle centered at the origin with a radius r_0. The boundary condition and the source term f_u, f_v are determined from the exact solutions

$$u(x,y) = \begin{cases} \dfrac{r^2}{\alpha_1} & \text{if } r \leq r_0, \\ \dfrac{r^2 - r_0^2}{\alpha_2} + \dfrac{r_0^2}{\alpha_1} & \text{otherwise,} \end{cases} \qquad v(x,y) = \begin{cases} \dfrac{(r^2 - r_0^2)^2}{\beta_1} & \text{if } r \leq r_0, \\ 0 & \text{otherwise,} \end{cases}$$

where $r = \sqrt{x^2 + y^2}$, $r_0 = 0.6$. Notice that the exact solutions satisfy interface conditions in equation (4.2). The coefficients of this test are taken as $\alpha_1 = 1, \alpha_2 = 100, \beta_1 = 0.5$.

Numerical solutions u_h, v_h in comparison with the exact solutions u, v are given Figure 4.1 (here we are showing the solutions in 3D view with the z-index is the value of solutions at points on Oxy). You can see, with a smooth mesh, we obtain almost the same results for both exact and numerical solutions.

For a reference, we also plot the two solutions of $w = u + \frac{\beta}{\alpha\lambda} v$ as in Figure 4.2.

The corresponding L^2 norm errors and convergence rates of $w - w_h$ and $v - v_h$ are given in Table 4.1 and Figure 4.3.
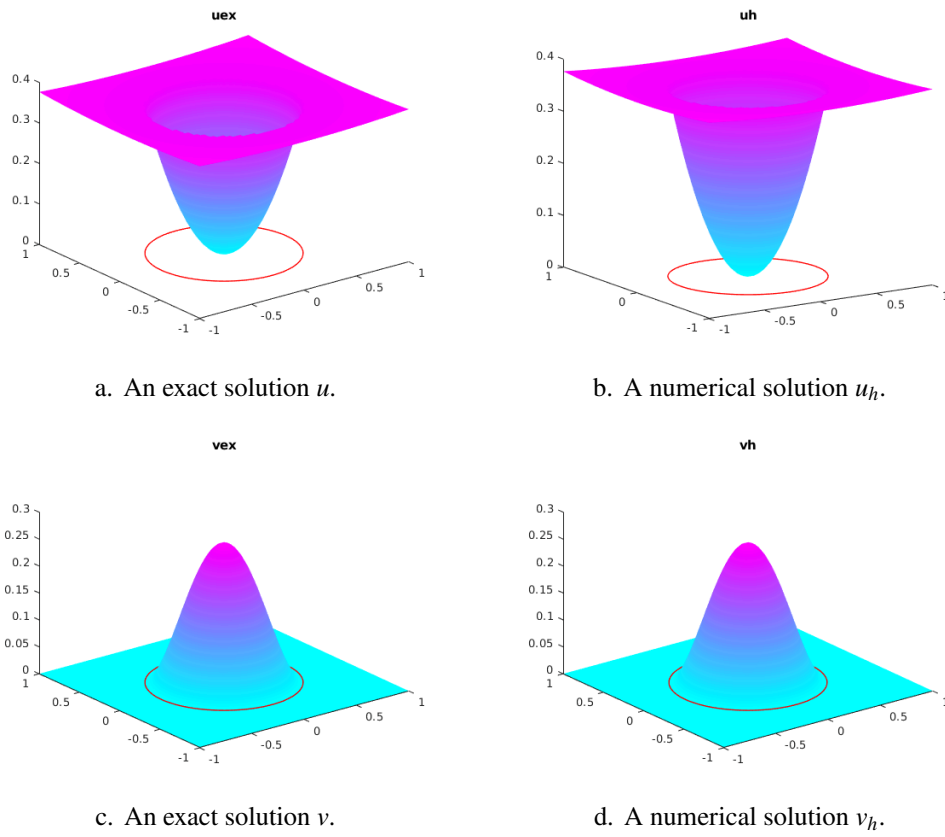
a. An exact solution u.

b. A numerical solution u_h.

c. An exact solution v.

d. A numerical solution v_h.

Figure 4.1. An exact solution and a numerical solution of u, v in a fine mesh.

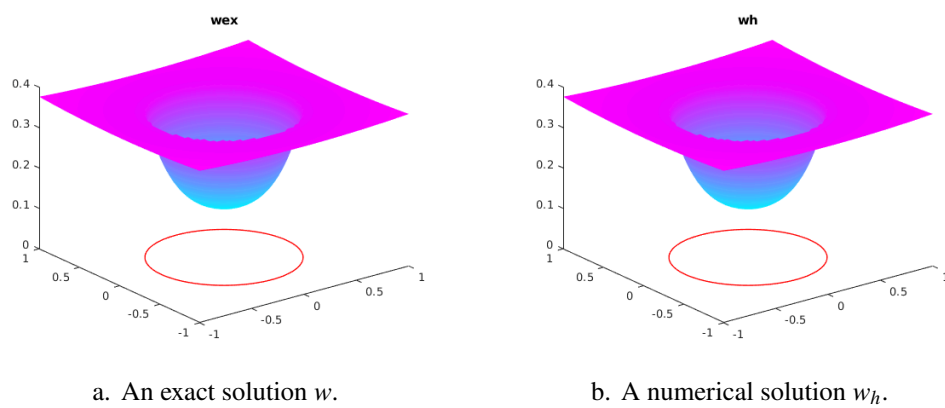

a. An exact solution w.

b. A numerical solution w_h.

Figure 4.2. An exact solution and a numerical solution of w in a fine mesh.

h	$\|w - w_h\|_{L^2}$	order	$\|v - v_h\|_{L^2}$	order
1.34×10^{-1}	7×10^{-3}		2.5×10^{-3}	
6.9×10^{-2}	2.1×10^{-3}	1.82	5.94×10^{-4}	2.14
3.49×10^{-2}	5.33×10^{-4}	2.01	1.16×10^{-4}	2.40
1.76×10^{-2}	1.38×10^{-4}	1.93	2.57×10^{-5}	2.18

Table 4.1. L^2 norm errors of the solutions with different mesh sizes.
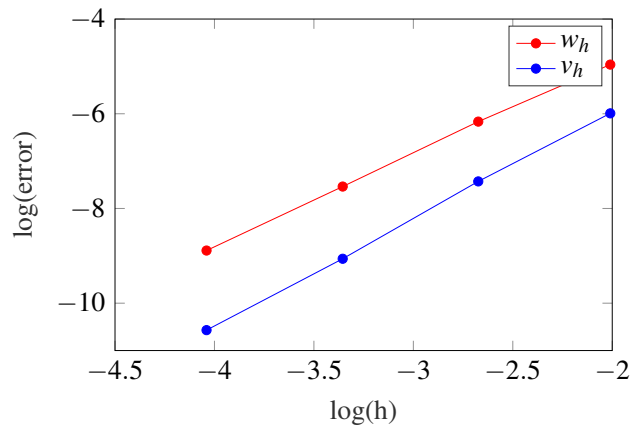
Figure 4.3. The convergence of numerical solutions to exact solutions of the system.

II

nxfem with biofilms

5. Level Set Method

Contents

As mentioned in Section 1.3, we need to track the interface's position on a fixed mesh from time to time. The *Level Set Method* (LSM) which is first introduced by Sethian and Osher in 1987 [45] helps us do that. In this chapter, I present a general idea of LSM, as well as its advantages, its inherent drawback and a way we couple it with NXFEM in solving an evolution problem. Some numerical test cases are also given.

5.1 Recall the method

The LSM comes with an idea of describing *implicitly* the interface Γ by a zero level set $\phi(x,t) = 0$, where Γ is the intersection between a plane $\phi = 0$ and a surface ϕ. The surface ϕ depends on a time t, i.e. its change leads to the change of the shape of the interface Γ (cf. Figure 5.1 and (5.1)).

$$\Gamma(t) = \{x \in \Omega : \phi(x(t), t) = 0\}. \tag{5.1}$$
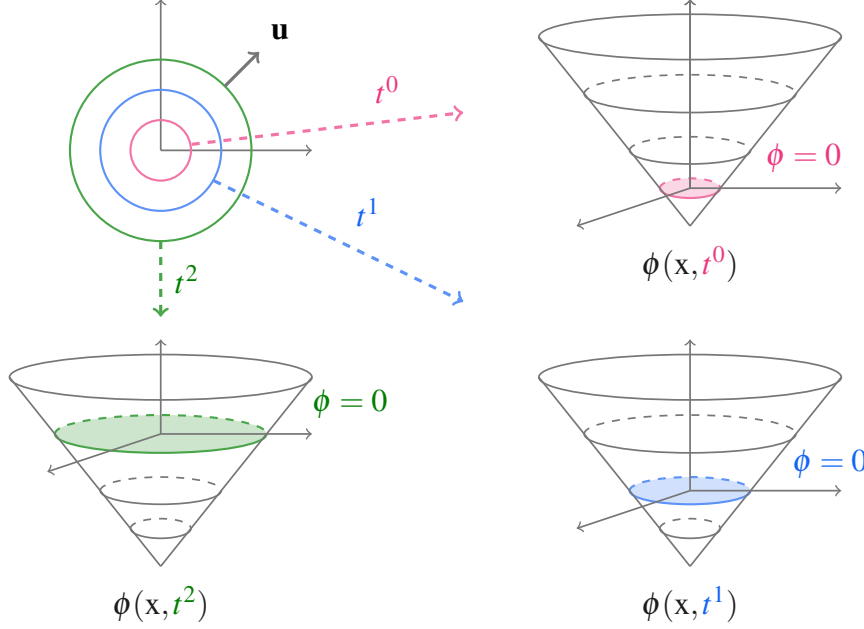
Figure 5.1. An illustration of the level set method. The interface at each time step t^n is determined by a *zero-level* set function $\phi = 0$.

The main purpose of LSM is to determine a function ϕ such that its zero level coincides to the interface Γ for all time. The change of Γ is controlled by a supposed velocity $\mathbf{u}$ given at every point $x(t) \in \Gamma(t)$. Then a movement of Γ can be described by a Lagrangian equation [67, Section 3.1],

$$\frac{dx}{dt}(t) = \mathbf{u}(x(t), t). \tag{5.2}$$

Because we want to find $\phi(x, t)$, or more precisely, we want to find a zero level of ϕ at the time t. This zero level set describes the interface Γ at this time as explained before. Given an initial value of ϕ at $t = 0$ (initial interface $\Gamma^0 := \Gamma(0)$), it's possible to find ϕ at all time steps with the help of a motion equation,

$$\frac{d}{dt}\phi(x(t), t) = 0. \tag{5.3}$$

The chain rule gives us,

$$\frac{\partial \phi}{\partial t}(x(t), t) + \frac{dx}{dt}(t) \cdot \nabla \phi(x(t), t) = 0.$$

Under the effect of the velocity $\mathbf{u}$ in (5.2) along the interface, we have a problem of finding ϕ such that,

$$\begin{cases} \dfrac{\partial \phi}{\partial t}(x,t) + \mathbf{u}(x,t) \cdot \nabla \phi(x,t) = 0, & \forall (x,t) \in \Omega \times \mathbb{R}^+, \\ \phi(x,t) = \phi_{in}(x,t), & \forall (x,t) \in \partial \Omega_{in} \times \mathbb{R}^+, \\ \phi(x,0) = \phi^0(x), & \forall x \in \Omega. \end{cases} \qquad (5.4)$$

In (5.4), $\partial \Omega_{in} := \{x \in \partial \Omega : \mathbf{u}(x,t) \cdot \mathbf{n}_\Omega < 0\}$ is the inflow boundary where $\mathbf{n}_\Omega$ is a outward unit normal vector on $\partial \Omega$.

> **Remark 5.1** The inflow Dirichlet boundary condition which is important for a well-posed problem is required to be satified that $\phi_{in}(x,0) = \phi^0(x)$ for all $x \in \partial \Omega_{in}$. However, a level set function obtained from (5.4) by using a time discretization method is used in a very short time interval, hence the boundary condition can be ignored in some cases.

The initial value ϕ^0 can be any function as long as its zero level set matches the initial interface Γ^0. One option is to choose ϕ^0 as a *signed distance function* defined in (5.5),

$$\phi^0(x) = \begin{cases} -d(x) & \text{if } x \in \Omega_1, \\ 0 & \text{if } x \in \Gamma, \\ d(x) & \text{if } x \in \Omega_2, \end{cases} \qquad (5.5)$$

where $d(x)$ is a distance function defined in (5.6), *i.e.* d measures the shortest distance between a point x in domain to the interface.

$$d(x) = \min_{x_\Gamma \in \Gamma} \|x - x_\Gamma\|, \forall x \in \Omega, \qquad (5.6)$$

Recall that Ω_1, Ω_2 are denoted as in (5.7) based on the value of ϕ (We already used these notations in Notation 3.1).

$$\begin{aligned} \Omega_1 &:= \{x \in \Omega : \phi(x) < 0\}, \\ \Omega_2 &:= \{x \in \Omega : \phi(x) > 0\}. \end{aligned} \qquad (5.7)$$

We can further compute a normal vector $\mathbf{n}_\Gamma$ and a mean curvature κ on the interface by a level set function in (5.8) [67].

$$\mathbf{n}_\Gamma = \frac{\nabla \phi(x)}{\|\nabla \phi(x)\|}, \quad \kappa(x) = \nabla \cdot \left(\frac{\nabla \phi(x)}{\|\nabla \phi(x)\|} \right), \quad x \in \Gamma. \qquad (5.8)$$

The signed distance function is a good candidate for ϕ^0 because it has a very useful

property in that $\|\nabla \phi^0\| = 1$ almost everywhere. The gradient of ϕ which is close to zero makes the numerical solution of (5.4) more difficult to find [69]. Meanwhile, its large norm decreases the numerical stability when we need to determine Γ from ϕ because it influences significantly to the transport of the interface. Moreover, the value of $\mathbf{n}_\Gamma$ and κ in (5.8) are more reliable if the gradient of ϕ is closer to 1.

Unfortunately, the property of signed distance function is generally lost during advection process [56]. In practice, there are two properties we have to ensure for the level set function at each time step $t = t^n$: (i) the zero level of ϕ^n describes the interface $\Gamma(t^n)$ and (ii) this ϕ^n satisfies the signed distance function's property, i.e. $\|\nabla \phi^n\| = 1$. These two properties lead us to two main difficulties when we work on a level set problem. First, the level set equation (5.4) is a hyperbolic equation which cannot be solved by a standard finite element method. One solution is to combine FEM with a stabilization technique. Following an option given in [66, Section 7.2], we will use the *streamline-diffusion finite element method* (SDFEM, which is also known as the *streamline upwinding Petrov-Galerkin method*) [70]. The second difficulty is to reestablish the signed distance property of the level set function from time to time. A re-initialization technique can be used and we choose the *Fast Marching Method* (FMM) [53] for our problems.

For more information, there is another common way to reinitialize ϕ [56], that is to solve the first order partial differential equation for $\psi = \psi(\mathrm{x}, \tau)$ such that,

$$\begin{cases} \dfrac{\partial \psi}{\partial \tau} = S_\alpha(\phi)(1 - \|\nabla \psi\|), \ \tau \geq 0, \mathrm{x} \in \Omega, \\ \psi(\mathrm{x}, 0) = \phi, \end{cases} \tag{5.9}$$

where $S_\alpha = \dfrac{\zeta}{\sqrt{\zeta^2 + \alpha^2}}, \zeta \in \mathbb{R}, 0 < \alpha \leq 1$. In practice, we need to use some time discretization method to find ψ numreically with a big enough time interval $(0, \tau_f)$ for τ. However, it's difficult to choose either a good τ_f or a best α in this case. Moreover, (5.9) is a nonlinear hyperbolic problem which is more difficult to find a solution. That's why we prefer to choose more technical method, that is Fast Marching Method for our problem.

> **Remark 5.2** Notice that, we talk too much about a continuous function ϕ but in practice, we work on an approximate ϕ_h of ϕ instead. Thus, an approximate signed distance is constructed too. We keep all ideas mentioned above to this discrete function including the reinitialization.

> **Remark 5.3** An important issue when we work with LSM is that the conservation of mass is generally lost during a temporal and spatial discretization of the level set equation. Some techniques are introduced in [66, Section 7.4.2] to overcome this disadvantage. In this thesis, I will use one of them sometimes.

5.2   The SUPG method with a Crank-Nicolson scheme

In this section, we will use the same techniques proposed in [66] for a discretization of level set equation (5.4) with the notice that we ignore the boundary condition for simplicity.

Recall that V_h is a finite element space defined on a standard mesh $\mathscr{T}_h$ given in (2.4), using test functions of a form $v_h + \delta_K \mathbf{u} \cdot \nabla v_h$ for some positive number δ_K defined in (5.11) and $v_h \in V_h$, a *Streamline diffusion finite element discretization* of the level set function (5.4) is as follows: Find $\phi_h(t) \in V_h$ such that,

$$\sum_{K \in \mathscr{T}_h} \langle \frac{\partial \phi_h}{\partial t}(t) + \mathbf{u}(t) \cdot \nabla \phi_h(t), v_h + \delta_K \mathbf{u}(t) \cdot \nabla v_h \rangle_K = 0, \qquad (5.10)$$

for all $v_h \in V_h$, $t \in [0, T_{\max}]$ and

$$\delta_K = C \frac{h_K}{\max\{\varepsilon_0, \|\mathbf{u}\|_{\infty, K}\}}, \qquad (5.11)$$

with a given small $\varepsilon_0 > 0$ and some positive constant C and $K \in \mathscr{T}_h$.

Let $\{\varphi_i\}_{i=1}^{N_{V_h}}$ be a basis of V_h, we introduce matrices $\mathbf{E}(\mathbf{u}) \in \mathbb{R}^{N_{V_h} \times N_{V_h}}$ and $\mathbf{H}(\mathbf{u}) \in \mathbb{R}^{N_{V_h} \times N_{V_h}}$ such that

$$
\begin{aligned}
\mathbf{E}_{ij}(\mathbf{u}(t)) &:= \sum_{K \in \mathscr{T}_h} \langle \varphi_j, \varphi_i + \delta_K \mathbf{u}(t) \cdot \nabla \varphi_i \rangle_K, \\
\mathbf{H}_{ij}(\mathbf{u}(t)) &:= \sum_{K \in \mathscr{T}_h} \langle \mathbf{u}(t) \cdot \nabla \varphi_j, \varphi_i + \delta_K \mathbf{u}(t) \cdot \nabla \varphi_i \rangle_K,
\end{aligned}
$$

where $1 \le i, j \le N_{V_h}$. Thus, using

$$\phi_h(t) = \sum_{i=1}^{N_{V_h}} \phi_i(t) \varphi_i, \quad \vec{\phi}(t) := (\phi_1(t), \dots, \phi_{N_{V_h}}(t)),$$

and $\vec{\phi}^0$ represents the initial value of $\vec{\phi}$, we can rewrite (5.10) in a matrix form: Find $\vec{\phi}(t) \in \mathbb{R}^{N_{V_h}}$ such that $\vec{\phi}(0) = \vec{\phi}^0$ and

$$\mathbf{E}(\mathbf{u}(t)) \frac{d\vec{\phi}}{dt}(t) + \mathbf{H}(\mathbf{u}(t)) \vec{\phi}(t) = 0, \text{ for all } t \in [0, T_{\max}]. \qquad (5.12)$$

We suppose that $[0, T_{\max}]$ is divided into a number of equal intervals τ and we denote that $t^n = n\tau$, $\vec{\phi}^n \simeq \vec{\phi}(t^n)$, a θ-schema applied to (5.12) results in

$$\frac{\vec{\phi}^{n+1} - \vec{\phi}^n}{\tau} + \theta \mathbf{E}^{-1}(\mathbf{u}^{n+1}) \mathbf{H}(\mathbf{u}^{n+1}) \vec{\phi}^{n+1} + \bar{\theta} \mathbf{E}^{-1}(\mathbf{u}^n) \mathbf{H}(\mathbf{u}^n) \vec{\phi}^n = 0, \qquad (5.13)$$

with $\bar{\theta} = 1 - \theta$ for $\theta \in [0, 1]$. In the case that $\mathbf{u}$ does not depend on time, (5.13) has a form,

$$(\mathbf{E}(\mathbf{u}) + \tau\theta\mathbf{H}(\mathbf{u}))\vec{\phi}^{n+1} = (\mathbf{E}(\mathbf{u}) - \tau\bar{\theta}\mathbf{H}(\mathbf{u}))\vec{\phi}^{n}. \qquad (5.14)$$

We don't forget to take $\theta = \frac{1}{2}$ in corresponding to the Crank-Nicolson method.

> **Remark 5.4** In practice, the velocity $\mathbf{u}$ will be replaced by a finite element approximation $\mathbf{u}_h$ and a Dirichlet boundary condition at the inlet $\partial\Omega_{in}$ will be applied in some cases.

Implementation issue. In the `NXFEM toolbox`, we use either the form (5.15) or the form (5.14) in the cases if $\mathbf{u}$ does or does not depend on time respectively. Functions `getMEls`, `getMHls` are used to compute the matrices $\mathbf{E}$ and $\mathbf{H}$ respectively.

$$(\mathbf{I} + \tau\theta\mathbf{K}(\mathbf{u}^{n+1}))\vec{\phi}^{n+1} = (\mathbf{I} - \tau\bar{\theta}\mathbf{K}(\mathbf{u}^{n}))\vec{\phi}^{n}, \qquad (5.15)$$

where $\mathbf{K}(u) = \mathbf{E}^{-1}(u)\mathbf{H}(u)$ and $\mathbf{I}$ is an identity matrix. The Algorithm 5.1 illustrates the basic steps to get the level set function at each time step.

Algorithm 5.1. Get the level set function ϕ at each time step.

Input : $\phi_h^0, T_{\max}, \mathbf{u}_h^0, \theta, \delta_T, \tau$.
Output ϕ at each time step.
:
$t = 0$;
$\phi_h^{\text{old}} = \phi_h^0$;
while $t < T_{\max}$ **do**

$\quad$ $t = t + \tau$;
$\quad$ $\mathbf{u}_h^{\text{new}} = \mathbf{u}_h(t)$;

$\quad$ $\mathbf{E}^{\text{new}} = \texttt{getMEls}(\mathbf{u}_h^{\text{new}}, \theta = \frac{1}{2}, \delta_T)$; $\quad$ $\mathbf{E}^{\text{old}} = \texttt{getMEls}(\mathbf{u}_h^{\text{old}}, \theta = \frac{1}{2}, \delta_T)$;
$\quad$ $\mathbf{H}^{\text{new}} = \texttt{getMHls}(\mathbf{u}_h^{\text{new}}, \theta = \frac{1}{2}, \delta_T)$; $\quad$ $\mathbf{H}^{\text{old}} = \texttt{getMHls}(\mathbf{u}_h^{\text{old}}, \theta = \frac{1}{2}, \delta_T)$;

$\quad$ Finding ϕ_h^{new} from (5.14) or (5.15);

$\quad$ $\phi_h^{\text{old}} = \phi_h^{\text{new}}$;

5.3 Reinitialization - Fast Marching Method

As mentioned before, it's necessary to use some method of reparametrization in order to make the level set function be a signed distance function again. We will not focus too much on this step and we use the toolbox `mshdist` from Pascal Frey

and Charles Dapogny [24] instead. Toolbox `mshdist` helps us compute a signed distance function to a discrete domain on an arbitrary triangular background mesh. In our case, it helps us reinitialize the level set function when it is far from the signed distance function.

From the manual of this toolbox, we use the following command to reparameter the level set function `phi` and couple Algorithm 5.2 and Algorithm 5.1 together. Note that, there must be two files `phi.mesh` and `phi.col`.

```
mshdist phi.mesh
```

When the level set function is reinitialized, the interface may move a little bit. This is due to the fact that the invariance of the zero level only applies to the continuous case, but it does not hold true for a discrete solution. That's why we cannot apply the reinitialization at all time steps, it is only applied when necessary. In `NXFEM toolbox`, I propose that we only apply the `mshdist` to reinitialize ϕ_h^n when $\|\nabla\phi_h^n\|$ is far a way from 1, i.e.,

$$|\|\nabla\phi^n\| - 1| > \varepsilon, \tag{5.16}$$

for some $\varepsilon > 0$.

Algorithm 5.2. Apply reinitialization.

Input : $\varepsilon, \tau, \phi_h^0, T_{\max}$.
Output ϕ at each time step.
:
$t = 0$;
$\phi_h^{\text{old}} = \phi_h^0$;
while $t < T_{\max}$ **do**
  $\quad t = t + \tau$;
  $\quad$ Solving ϕ_h^{new} as in Algorithm 5.1;
  $\quad$ **if** $|\|\nabla\phi_h^{\text{new}}\| - 1| > \varepsilon$ **then**
  $\quad\quad$ Write ϕ_h^{new} to `phi.sol`;
  $\quad\quad$ Write `msh` to `phi.mesh`;
  $\quad\quad$ Apply `mshdist` and update ϕ_h^{new};
  $\quad \phi_h^{\text{old}} = \phi_h^{\text{new}}$;

5.4 A numerical test case

In this section, I will use a test case proposed in [34] to illustrate the level set method and the arguments I mentioned in the previous sections.

We will solve a level set problem (5.4) on a domain $\Omega = [0,1] \times [0,1]$ with homogeneous Newmann boundary conditions. A velovity field $\mathbf{u}$ is given in (5.17).

$$\mathbf{u} = \begin{bmatrix} 2\sin(2\pi y)\sin^2(\pi x)\cos(\pi t) \\ -2\sin(2\pi x)\sin^2(\pi y)\cos(\pi t) \end{bmatrix}. \tag{5.17}$$

The initial condition ϕ^0 is a circle with the radius 0.15 and the center $(0.5, 0.75)$. It is given in a form of the equation (5.18).

$$\phi^0 = \sqrt{(x-0.5)^2 + (y-0.75)^2} - 0.15. \tag{5.18}$$

We investigate the change of the level set ϕ (a vortex) from $t = 0$ to $t = T_{\max} = 1$. This vortex's direction changes at $t = 0.5$ and it goes back to the old path (cf. Figure 5.2). This means that when $t = T_{\max}$, ϕ must be the same as the initial ϕ^0 (cf. Figure 5.3). This special property gives us a chance to examine the quality of our implementation.

Remark 5.5 In general, we need an additional step called "*initialisation*" which makes the level set function become a signed distance function. With `mshdist`, we are also able to do this. The function ϕ^0 defined in (5.18) is already a signed distance function, we don't need an initialisation step for this test case.

a. For $0 \leq t \leq \frac{1}{2}$.   b. For $\frac{1}{2} < t \leq 1$.
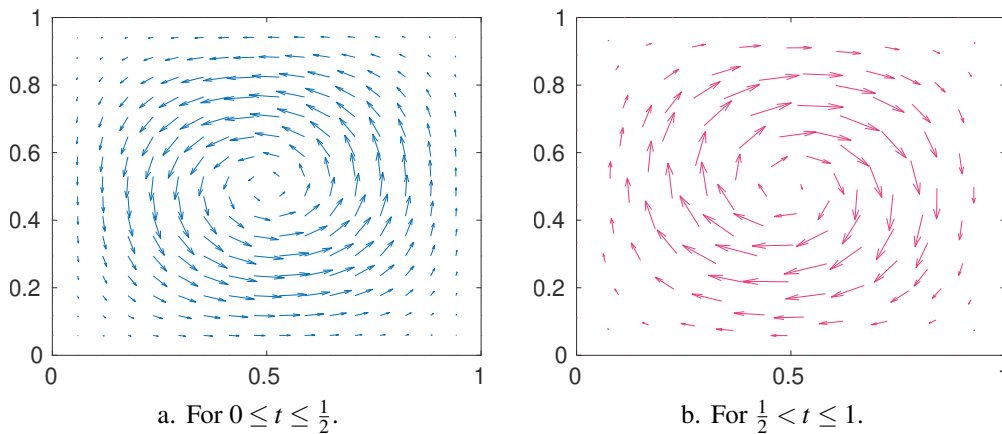
Figure 5.2. Vortex test case: Direction of velocity $\mathbf{u}$ at different time.

In order to estimate the quality of the method, we need below estimations in comparison with the original signed distance function ϕ,

a. $t = 0$.   b. $t = 0.24$.   c. $t = 0.48$.

d. $t = 0.52$.   e. $t = 0.76$.   f. $t = 1$.
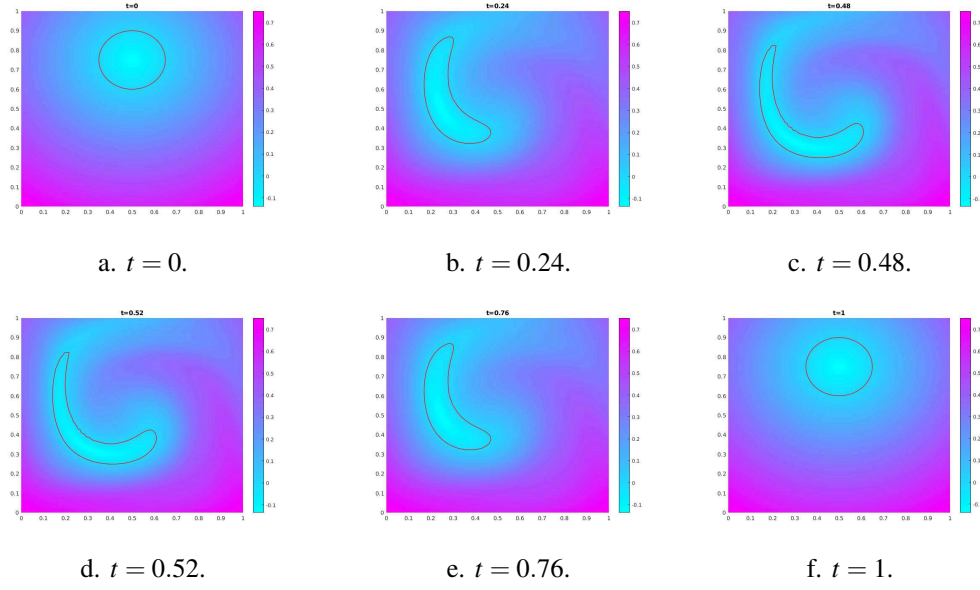
Figure 5.3. Vortex test case: Computed interface at different time.

$$\|\phi\|_{L^2(\Gamma_h^{t_n})} = \left(\sum_{K \in G_h^{t_n}} \int_{\Gamma_{K,h}} \phi(\mathrm{x})^2 \, \mathrm{dx} \right)^{\frac{1}{2}},$$

$$\|\phi_h^{t_n} - \phi\|_{L^2(\Omega)} = \left(\sum_{T \in \mathscr{T}_h} \int_T |\phi_h^{t_n}(\mathrm{x}) - \phi(\mathrm{x})|^2 \right)^{\frac{1}{2}}, \tag{5.19}$$

in which G_h is recalled to be a set of all triangles cut by the interface, $\Gamma_{K,h}$ denotes the segment connecting the intersection points between Γ and ∂K. $\Gamma_h^{t_n}, \phi_h^{t_n}$ is the approximate of Γ and ϕ respectively at time $t = t_n$. Ones can find, in the NXFEM toolbox, the corresponding functions which are called getNormL2foGh and getNormL2stdfhf. An implementation issue of these functions are given in Appendix A.6.1.

For example, we consider at the beginning time when $t_n = 0$, the estimates in (5.19) are actually the interpolation errors between ϕ and ϕ_h. By computing independently on NXFEM toolbox and on FreeFem++[33], we have almost the same results which are given in Table 5.1.

If we don't use either SUPG or FMM, *i.e.* we just use the standard FEM, a strange result which is very nice is obtained in Table 5.2 and Figure 5.4 (it goes back exactly to the beginning position no matter how fine the mesh is). This "strange" is also mentioned in [66] and it's still under study.

For a hyperbolic equation like the level set equation, we should (and must) use a stabilized method like SUPG method. Table 5.3 gives a result of using SUPG (without using FMM). It's really better than the result shown in Table 5.2 in which

$h_{\max}$	using FreeFem++		using NXFEM toolbox	
	order of $\\|\phi\\|_{L^2(\Gamma_h^0)}$	order of $\\|\phi_h^0 - \phi\\|_{L^2(\Omega)}$	oder of $\\|\phi\\|_{L^2(\Gamma_h^0)}$	oder of $\\|\phi_h^0 - \phi\\|_{L^2(\Omega)}$
19.64×10^{-2}				
9.70×10^{-2}	2.06	2.38	2.06	2.39
4.78×10^{-2}	1.92	1.64	1.92	1.62
2.45×10^{-2}	2.10	2.15	2.10	2.16
1.28×10^{-2}	2.14	2.03	2.14	2.04

Table 5.1. Vortex test case: Approximation errors computed by NXFEM toolbox and FreeFem++ for different mesh sizes.

$h_{\max}$	$\\|\phi\\|_{L^2(\Gamma_h^{T\max})}$	order	$\\|\phi_h^{T\max} - \phi\\|_{L^2(\Omega)}$	order
19.64×10^{-2}	87.40×10^{-4}		71.92×10^{-4}	
9.70×10^{-2}	20.37×10^{-4}	2.09	13.30×10^{-4}	2.42
4.78×10^{-2}	5.23×10^{-4}	1.92	4.21×10^{-4}	1.62
2.45×10^{-2}	1.28×10^{-4}	2.58	0.99×10^{-4}	2.65

Table 5.2. Vortex test case: Approximation errors for different mesh sizes in the case: **without** SUPG and **without** FMM.

a. $h = 9.7 \times 10^{-2}, t = 0.$

b. $h = 9.7 \times 10^{-2}, t = 1.$

c. $h = 2.45 \times 10^{-2}, t = 0.$

d. $h = 2.45 \times 10^{-2}, t = 1.$
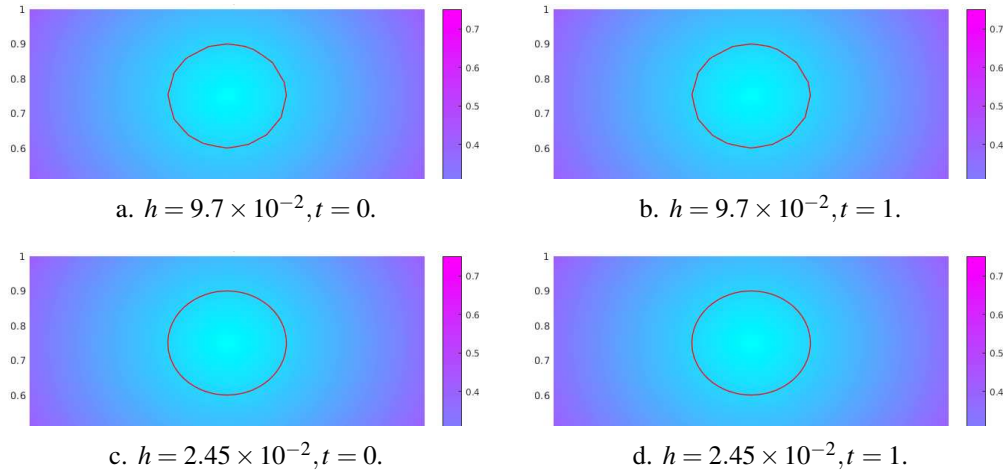
Figure 5.4. Vortex test case: Interface's position before and after the process in the case: **without** SUPG and **without** FMM.

we don't use SUPG method. Besides that, when we consider an interface's position, it doesn't go back to the beginning position like in the case of the standard FEM for less number of elements. However, when we increase the number of elements of the mesh, the result is better, cf. Figure 5.5.

nSeg	$h_{\max}$	$\|\phi\|_{L^2(\Gamma_h^{T_{\max}})}$	order	$\|\phi_h^{T_{\max}} - \phi\|_{L^2(\Omega)}$	order
11	13.82×10^{-2}	32.57×10^{-3}		19.90×10^{-3}	
22	6.92×10^{-2}	8.39×10^{-3}	1.96	6.06×10^{-3}	1.72
44	4.07×10^{-2}	2.81×10^{-3}	2.06	2.25×10^{-3}	1.87
88	2.10×10^{-2}	0.74×10^{-3}	2.02	0.47×10^{-3}	2.35

Table 5.3. Vortex test case: Approximation errors for different mesh sizes in the case: **with** SUPG and **without** FMM.

a. $\mathrm{nSeg} = 2^4, t = 0$.   b. $\mathrm{nSeg} = 2^4, t = 1$.

c. $\mathrm{nSeg} = 2^6, t = 0$.   d. $\mathrm{nSeg} = 2^6, t = 1$.
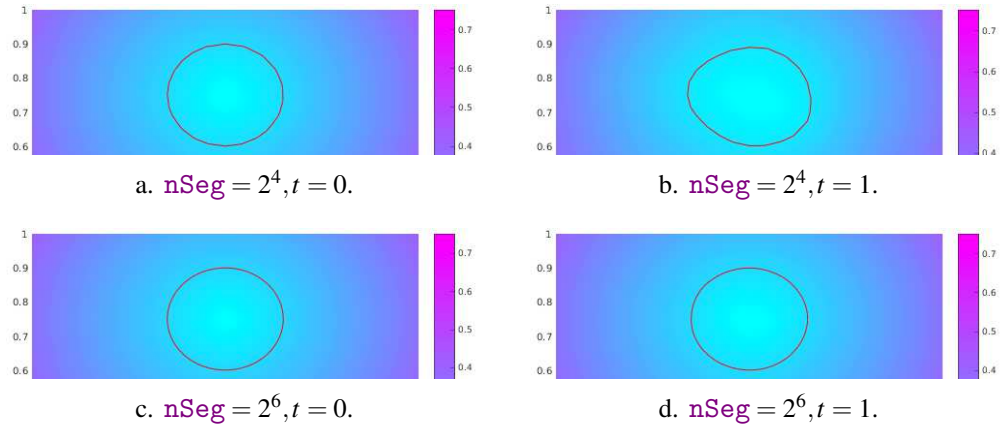
Figure 5.5. Vortex test case: Interface's position before and after the process in the case: **with** SUPG and **without** FMM.

Despite of above good results, the property in which $\|\nabla \phi_h^{t_n}\| \simeq 1$ cannot be guaranteed. More specifically, if we don't use FMM, in the total of 100 time steps, there are up to 16 times in which $\|\|\nabla \phi_h^{t_n}\| - 1| > \varepsilon$ where $\varepsilon = 0.1$. If we take $\varepsilon = 0.5$, there will be up to 54 times. It's obviously not good as it is if we use this level set function as an interface in our biofilm problem.

In backwards, if we use FMM in many time steps, the zero-level property cannot be obtained because FMM makes the interface move too much. You can see in Table 5.4, the convergence orders are bad for the case of $\|\phi_h^{T_{\max}} - \phi\|_{L^2(\Omega)}$. A reason for that is because the zero-level of $\phi_h^{T_{\max}}$ and ϕ are different (maybe not much) but their signed-distance values *on the whole domain* are very different. For this reason, we cannot use FMM how many times we want, we need to use an estimate like (5.16). For a comparison, we limit the number of times we use the FMM and get a better result shown in Figure 5.6.

As mentioned before, a difficulty when working with FMM is the way we choose the parameter ε in (5.16). It's normal in practice and also indicated in [24, 34, 72]. This work is empirically done regarding specific problems. I don't focus too much time on this test case and let the work of choosing ε for our main problem in the next chapter.

nSeg	$h_{\max}$	$\|\phi\|_{L^2(\Gamma_h^{T_{\max}})}$	order	$\|\phi_h^{T_{\max}} - \phi\|_{L^2(\Omega)}$	order
37	6.60×10^{-2}	10.03×10^{-3}		6.34×10^{-2}	
57	4.36×10^{-2}	5.58×10^{-3}	1.41	4.95×10^{-2}	0.59
77	3.45×10^{-2}	3.45×10^{-3}	2.06	5.03×10^{-2}	-0.07
101	2.59×10^{-2}	1.93×10^{-3}	2.02	5.11×10^{-2}	-0.05

Table 5.4. Vortex test case: Approximation errors for different mesh sizes in the case of using both SUPG method and FMM.

a. $\texttt{nSeg} = 2^6, t = 0$. **Unlimited** number of times using the FMM.

b. $\texttt{nSeg} = 2^6, t = 1$. **Unlimited** number of times using the FMM.

c. $\texttt{nSeg} = 2^6, t = 0$. **Limited** number of times using the FMM.

d. $\texttt{nSeg} = 2^6, t = 1$. **Limited** number of times using the FMM.
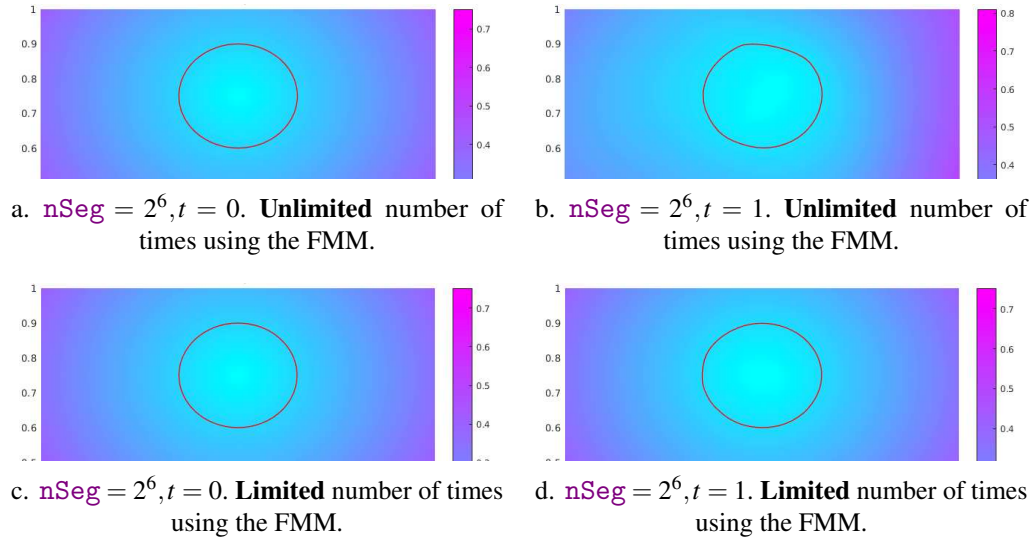
Figure 5.6. Vortex test case: Interface's position before and after the process in the case of using FMM in 2 ways: limited and unlimited number of uses.

Remark 5.6 Summing up, we remarks some notable points,

    i) Sometimes, we need to use (5.19) to estimate the quality of the level set method. However, (5.19) shows only the quality on the whole domain, not the position of the interface. What we need is the position of the interface, not its values on the whole domain.

    ii) A "beautiful but strange" result is obtained if we don't use either SUPG or FMM. However, the property $\nabla\phi \simeq 1$ is not guaranteed during the process.

    iii) A better result is obtained if we use the SUPG method for such hyperbolic problem. However, the interface's position is worse with less fine mesh and it's better otherwise.

    iv) We cannot use FMM how many times we want because it makes the interface move. It's necessary to use estimate like in (5.16).

    v) A good choice of parameters depends on the problem we are working on and they are chosen empirically.

6. Application to a biofilm growth model

Contents

6.1 Coupling NXFEM with Level Set Method

Look back to the equation (1.9) of Substrate S and Biomass B,

$$
\begin{cases}
\partial_t S - \nabla \cdot (D_S^b \nabla S) + \mathbf{u} \cdot \nabla S + \alpha B g(S) = 0 & \text{in } (0,T) \times \Omega_b, \\
\partial_t S - \nabla \cdot (D_S^f \nabla S) + \mathbf{v} \cdot \nabla S = 0 & \text{in } (0,T) \times \Omega_f, \\
\partial_t B - \nabla \cdot (D_B \nabla B) + \mathbf{u} \cdot \nabla B - \beta B g(S) = 0 & \text{in } (0,T) \times \Omega_b, \\
-\Delta \Phi + \beta g(S) = 0 & \text{in } \Omega_b.
\end{cases}
\qquad \text{(1.9 revisited)}
$$

In that equation, the velocity field $\mathbf{u}$ is assumed to be irrotational, i.e. $\nabla \times \mathbf{u} = 0$. Thus, it can be extracted from the equation of Φ thanks to $\mathbf{u} = \nabla \Phi$ and we can use Φ to track the interface ϕ via an equation level set like (1.11).

$$
\begin{cases}
\dfrac{\partial \phi}{\partial t}(\mathbf{x},t) + \nabla \Phi(\mathbf{x},t) \cdot \nabla \phi(\mathbf{x},t) = 0, & \forall (\mathbf{x},t) \in \Omega \times (0,T), \\
\phi(\mathbf{x},0) = \phi^0(\mathbf{x}), & \forall \mathbf{x} \in \Omega.
\end{cases}
\qquad \text{(1.11 revisited)}
$$

By using an appropriate discretization method, we start with initial condition of interface ϕ_h^0. At each time step, we seek the values of S_h and B_h based on the NXFEM method described in Chapter 2. After that, we find a new interface for the next time step by solving a level set equation (1.11) with the help of techniques described in previous sections in this chapter. With the new interface, we continue to work on a new system of S_h and B_h and keep going. An idea of this coupling method is illustrated in Figure 6.1 and Algorithm 6.1.
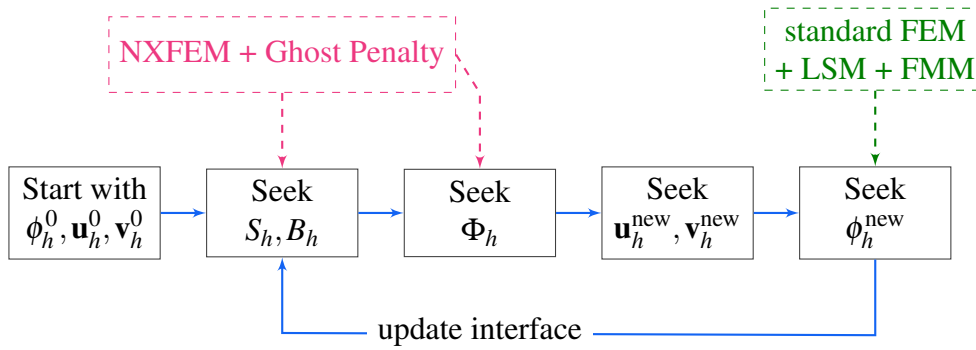
Figure 6.1. An idea of coupling NXFEM with Level Set Method.

Remark 6.1 Because we solve the equation of ϕ on a standard finite element space V_h based on Φ which is solved in NXFEM space V_h^Γ, we have to interpolate Φ to V_h. One more thing, if we want to perform the plot, we need to interpolate the solution to V_h. For the interpolation between V_h and V_h^Γ, please see Section A.7. We use functions `interNX2STD` and `interSTD2NX` in `NXFEM toolbox` to perform these tasks (cf. Section A.7.1).

For an example of this coupling method, please read Section 6.2.

6.2 Some biofilm growth models

In this section, we examine a kinetic model suggested in [27]. In this work, Chopp *et al.* introduced a strategy to couple the XFEM and LSM. They used XFEM engine on a triangle Finite Element mesh to solve equations of substrate and biomass. The Level Set engine is used on a square finite difference grid to update the change of interface. There are bilinear interpolations between these two engines of the substrate and the biomass solution.

In contrast, we use only one grid for both engines. We use SUPG and FMM on a triangle grid described in Chapter 5 for the level set equation. For equations of the

---

Algorithm 6.1. Coupling NXFEM with Level Set Method.

Input : $\phi_h^0, \mathbf{u}_h^0, \mathbf{v}_h^0, g, \tau, T_{\max}$, parameters
Output S_h, B_h, ϕ_h
:
$t = 0; \phi^{\mathrm{old}} = \phi_h^0; \mathbf{u}_h^{\mathrm{old}} = \mathbf{u}_h^0; \mathbf{v}_h^{\mathrm{old}} = \mathbf{v}_h^0;$
while $t < T_{\max}$ **do**

> $t = t + \tau;$
> Solve equation of S_h, B_h;
> Solve equation of Φ_h;
> Solve level set equation to get ϕ_h^{new};
> Update interface Γ_h ;
>
> $\phi_h^{\mathrm{old}} = \phi_{\mathrm{new}}, \mathbf{u}_h^{\mathrm{old}} = \mathbf{u}_h^{\mathrm{new}}, \mathbf{v}_h^{\mathrm{old}} = \mathbf{v}_h^{\mathrm{new}};$

---

substrate and the biomass, we use NXFEM to find solutions in a discretization of space and time. All of these will be implemented with the help of `NXFEM toolbox`.

> **Remark 6.2** In spite of using the same models and some parameters' value given in the work of Chopp *et al.*, we focus on the methods we have used and we give some new behaviors of the growth of the biofilm.

6.2.1 Models

Let us consider a convex polygonal, Lip-schitz and bounded domain Ω in $\mathbb{R}^2$ such that $\overline{\Omega} = \overline{\Omega}_f \cup \overline{\Omega}_b$ where Ω_b denotes the biofilm region. These two regions are separated by a sufficiently smooth interface Γ. The concentration of the substrate S is presented both in Ω_f and Ω_b and it diffuses in the domain in different rates D_S^b and D_S^f which are such that $D_S^b = \alpha_D D_S^f$ where $0 < \alpha_D < 1$. This means that we consider S diffusing in Ω_b with a lower rate because of the obstruction of the biomass in the biofilm region. Within the biofilm, S is also affected by an advection velocity $\mathbf{U}$. This velocity will reduce an amount of S inside the biofilm. Of course, S is the main nutrient of the biofilm, it will be also diminished if there are (much) bacteria. Outside the biofilm, there is only the diffusive phenomenon which has an effect on S.

We derive an equation of S which describes above relations via (6.1). The negative sign $(-)$ indicates a reduction of the concentration of S whereas the positive one $(+)$ shows the increasement.
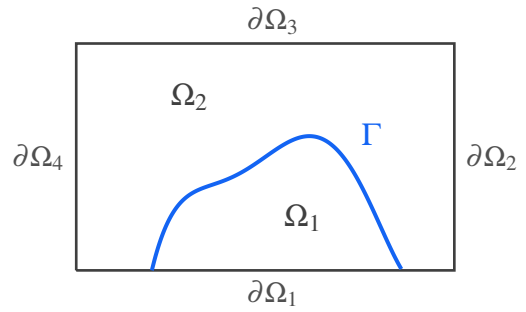
$$\begin{cases} \partial_t S = D_S^b \nabla^2 S - \nabla \cdot (S\mathbf{U}) + f\mu_S(S) & \text{in } \Omega_b, \\ \partial_t S = D_S^f \nabla^2 S & \text{in } \Omega_f, \end{cases} \tag{6.1}$$

where f is the active biomass volume fraction [16], μ_S is the specific substrate consumption rate which is given in (6.2).

$$\mu_S(S) = \bar{\mu}_S \frac{S}{K_0 + S}. \tag{6.2}$$

The velocity $\mathbf{U}$ which depends on the growth of biofilm is assumed to be irrotational, i.e., $\nabla \times \mathbf{U} = 0$. This leads us to find an alternative presentation Φ of $\mathbf{U}$ instead of itself (in that, $\mathbf{U} = \nabla\Phi$). We call Φ a *potential*. The mass balance of Φ can be derived in terms of the rates of components inside the biofilm as in (6.3).

$$\begin{cases} \nabla^2\Phi = f\mu_\Phi(S) & \text{in } \Omega_b, \\ \nabla^2\Phi = 0 & \text{in } \Omega_f, \end{cases} \tag{6.3}$$

where $\mu_\Phi = \mu_x + \mu_w$, μ_x and μ_w are the net rate of active biomass production and the net rate of EPS production respectively. They follow (6.4). Because there is no bacteria outside the biofilm region, it's reasonable if Φ is 0 in Ω_f.

$$\mu_x(S) = \bar{\mu}_x \frac{S}{K_S + S}, \quad \mu_w(S) = \bar{\mu}_w \frac{S}{K_S + S}. \tag{6.4}$$

For more details about the parameters $\bar{\mu}_S, \bar{\mu}_x, \bar{\mu}_w$ and their values in experiments, please reference to [26, Table A.1].

Both substrates and bacteria are continuous through the interface Γ. That why we need to use jumb conditions for S and Dirichlet conditions for Φ (becase Φ is zero outside Ω_b). We want the flows of S and Φ to be smooth across Γ also, that why we need flux conditions ($\nabla_\mathbf{n}$) for each of them. Summing up, we need the conditions given in (6.5).

$$\begin{cases} [\![S]\!] = [\![\nabla_\mathbf{n}S]\!] = 0, \\ \Phi = \nabla_\mathbf{n}\Phi = 0. \end{cases} \tag{6.5}$$

For the test cases, we apply some Dirichlet boundary conditions for both of S and Φ on the top of the domain.

Remark 6.3 Because of the values of S, Φ from the results in [15, 16], we can ignore, sometimes, the time derivatives and advection terms in (6.1). It's because the change of the concentration S is very slightly different in time scale and the velocity $\mathbf{U}$ doesn't have a large impact on the change.

Notation 6.1 From this to the end of the chapter, for simplicity, we replace the

notations $\Omega_b, \Omega_f, S, \Phi$ by Ω_1, Ω_2, u, v respectively.

With Remark 6.3 and Notation 6.1, (6.1) will be rewritten in (6.6) for the full equation of substrate.

$$
\begin{cases}
-\nabla \cdot (k_S \nabla u) + f\mu_S(u) = 0, & \text{in } \Omega, \\
\llbracket u \rrbracket = \llbracket k_S \nabla_{\mathbf{n}} u \rrbracket = 0, & \text{on } \Gamma, \\
u = \bar{S}, & \text{on } \partial\Omega_3, \\
\nabla_{\mathbf{n}} u = 0 & \text{on } \partial\Omega \backslash \partial\Omega_3.
\end{cases}
\tag{6.6}
$$

The same for (6.3) with (6.7).

$$
\begin{cases}
-\nabla \cdot (\nabla v) = -f\mu_\Phi(u), & \text{in } \Omega, \\
v = \nabla_{\mathbf{n}} v = 0, & \text{on } \Gamma, \\
v = 0, & \text{on } \partial\Omega_3, \\
\nabla_{\mathbf{n}} v = 0 & \text{on } \partial\Omega \backslash \partial\Omega_3,
\end{cases}
\tag{6.7}
$$

where,

$$
k_S := \begin{cases} D_S^b, \text{ in } \Omega_1, \\ D_S^f, \text{ in } \Omega_2. \end{cases}
\quad
\bar{\mu}_S := \begin{cases} \bar{\mu}_S, \text{ in } \Omega_1, \\ 0, \text{ in } \Omega_2. \end{cases}
\quad
\bar{\mu}_\Phi := \begin{cases} \bar{\mu}_x + \bar{\mu}_w, \text{ in } \Omega_1, \\ 0, \text{ in } \Omega_2. \end{cases}
$$

Remark 6.4 Different from the work of Chopp *et al.*in which the equation of potential Φ was solved by the *penalty method* (introduced in [60]), we use here a method which is already described in Section 4.2.2 for the equation of v. Moreover, we use Newton method coupling with NXFEM to solve the nonlinear equation of u.

6.2.2 Weak forms

With the same arguments and notations in Chapter 4, a weak form of (6.6) is to seek $u \in H^1(\Omega)$ such that $u = \bar{S}$ on $\partial\Omega_3$ and u satisfies (6.8)[1].

$$
\langle k_S \nabla u, \nabla \varphi \rangle_\Omega + \langle f\mu_S(u), \varphi \rangle_\Omega = 0, \quad \forall \varphi \in H_0^1(\Omega).
\tag{6.8}
$$

Similarly, a weak form of (6.7) is to seek $v \in V$ such that $v = 0$ on $\partial\Omega_3$ and v satisfies

---

[1]For the definition of $H^1(\Omega), H_0^1(\Omega)$, cf. Definition 2.2.

$(6.9)^2$.

$$\langle \nabla v, \nabla \varphi \rangle_\Omega = \langle -f \mu_\Phi(u), \varphi \rangle_\Omega, \quad \forall \varphi \in V_0. \tag{6.9}$$

6.2.3 Discretization and iteration

Again, with the same manner as in Section 2.2, a discrete form of the problem (6.8) is as follows: Seek a solution $u_h \in V_h^\Gamma$ such that $u_h = \bar{S}$ on $\partial \Omega_3$ and u satisfies (6.10).

$$\boxed{a_{uh}(u_h, \varphi_h) = 0, \quad \forall \varphi_h \in V_h^0,} \tag{6.10}$$

where V_h^Γ, V_h^0 are defined in (2.14), (2.15) and

$$a_{uh}(u_h, \varphi_h) := \langle k_S \nabla u_h, \varphi_h \rangle_{\Omega_{12}} - \langle [\![u_h]\!], \{\!\{k_S \nabla_\mathbf{n} \varphi_h\}\!\} \rangle_\Gamma$$
$$- \langle \{\!\{k_S \nabla_\mathbf{n} u_h\}\!\}, [\![\varphi_h]\!] \rangle_\Gamma + \lambda_u \langle [\![u_h]\!], [\![\varphi_h]\!] \rangle_\Gamma + \langle f \mu_S(u_h), \varphi_h \rangle_\Omega.$$

A weak form (6.9) of v_h is discretized: Seek a solution $v_h \in V_h^\Gamma$ such that $v_h = 0$ on $\partial \Omega_3$ and v_h satisfies (6.11).

$$\boxed{a_{vh}(v_h, \varphi_h) = K_{vh}(u_h, \varphi_h), \quad \forall v_h \in V_h^0,} \tag{6.11}$$

in that,

$$a_{vh}(v_h, \varphi_h) := \langle \nabla v_h, \varphi_h \rangle_{\Omega_{12}} - \langle [\![v_h]\!], \{\!\{\nabla_\mathbf{n} \varphi_h\}\!\} \rangle_\Gamma - \langle \{\!\{\nabla_\mathbf{n} v_h\}\!\}, [\![\varphi_h]\!] \rangle_\Gamma$$
$$+ \langle \{\!\{v_h\}\!\}, \{\!\{\varphi_h\}\!\} \rangle_\Gamma + \lambda_v \kappa_1 \kappa_2 \langle [\![v_h]\!], [\![\varphi_h]\!] \rangle_\Gamma,$$
$$K_{vh}(u_h, \varphi_h) := \langle -f \mu_\Phi(u_h), \varphi_h \rangle_\Omega$$

> **Remark 6.5** The main different point in our work is in the last two terms of a_{vh}.

For the nonlinear problem (6.10), we use the Newton method coupling with NXFEM to find a solution. Let $F(u_h)$ be the LHS of (6.10) and $\delta_h \in V_h^\Gamma$, we have

$$F(u_h + \delta_h) - F(u_h) = \langle k_S \nabla \delta_h, \varphi_h \rangle_{\Omega_{12}} - \langle [\![\delta_h]\!], \{\!\{k_S \nabla_\mathbf{n} \varphi_h\}\!\} \rangle_\Gamma - \langle \{\!\{k_S \nabla_\mathbf{n} \delta_h\}\!\}, [\![\varphi_h]\!] \rangle_\Gamma$$
$$+ \lambda_u \langle [\![\delta_h]\!], [\![\varphi_h]\!] \rangle_\Gamma + \langle f \mu_S(u_h + \delta_h) - f \mu_S(u_h), \varphi_h \rangle_\Omega.$$

From definition (6.2) of μ_S,

---

[2]For the definition of V, V_0, cf. (4.4).

$$\mu_S(u_h + \delta_h) - \mu_S(u_h) = \frac{\partial \mu_S}{\partial u_h}(u_h)\delta + o(\delta_h) = \bar{\mu}_S \frac{K_S}{(K_S + u_h)^2}\delta_h + o(\delta_h),$$

where $o(\delta_h) \to 0$ when $h \to 0$. Therefore, we have a differential $DF(u_h)$ of F at u_h such that,

$$F(u_h + \delta_h) = F(u_h) + DF(u_h)\delta_h + o(\delta_h),$$

where,

$$DF(u_h) = \langle k_S \nabla \delta_h, \varphi_h \rangle_{\Omega_{12}} - \langle [\![\delta_h]\!], \{\!\{k_S \nabla_{\mathbf{n}} \varphi_h\}\!\} \rangle_\Gamma - \langle \{\!\{k_S \nabla_{\mathbf{n}} \delta_h\}\!\}, [\![\varphi_h]\!] \rangle_\Gamma$$
$$+ \lambda_u \langle [\![\delta_h]\!], [\![\varphi_h]\!] \rangle_\Gamma + \langle f\mu_S \frac{K_S \delta_h}{(K_S + u_h)^2}, \varphi_h \rangle_\Omega.$$

$DF(u_h)$ is a linear application which is easier to handle in NXFEM space. We go from seeking a solution u_h of (6.10) to seeking a solution $\delta_h \in V_h^\Gamma$ of problem (6.12) w.r.t. Algorithm 6.2.

$$DF(u_h)\delta_h = F(u_h). \tag{6.12}$$

Algorithm 6.2. Newton method for finding solution u_h.

Input : An initial value of u_h and a stopping criteria ε.
Output u_h
:
while $\dfrac{\|\delta_h\|}{\|u_h\|} > \varepsilon$ **do**

> Solve (6.12) for δ_h;
> Update u_h by $u_h - \delta_h$;

Remark 6.3 reminds us not to forget the main biofilm model is an evolution problem. It means that we are looking for solutions for each time step and from that, we can know the growth of biofilm with time. Refer to Section 6.1, we investigate a more detailed strategy to solve this problem.

We start with an initial interface Γ^0 (at time step t^0) which stands for a layer covering biofilm region. With this Γ^0, we find the substrate u_h^1 from (6.6) by solving problem (6.10) thanks to Algorithm 6.2. With this substrate u_h^1, we have enough information to find the potential v_h^1 from (6.7) by solving problem (6.11). Because the RHS of (6.11) is actually a constant with already known u_h^1, we use the standard method of NXFEM stated in previous chapter to find v_h^1. After solving v_h^1, we calculate its gradient to find the new interface Γ^1 thanks to level set equation (6.13). Keep going until we understand the behavior of interface (biofilm) after a maximum time. The whole process is redescribed in Algorithm 6.3.

$$\begin{cases} \dfrac{\partial \phi}{\partial t}(\mathrm{x},t) + \nabla v(\mathrm{x},t) \cdot \nabla \phi(\mathrm{x},t) = 0, & \forall (\mathrm{x},t) \in \Omega \times (0,T), \\ \phi(\mathrm{x},0) = \phi^0(\mathrm{x}), & \forall \mathrm{x} \in \Omega. \end{cases} \tag{6.13}$$

Algorithm 6.3. Solving problem of biofilm.

Input : ϕ^0, $T_{\max}$
Output u_h^n, v_h^n, ϕ^n at time step t^n
:

$t = 0$; $\phi^{\mathrm{old}} = \phi_h^0$;
while $t < T_{\max}$ **do**

> Solve (6.10) for u_h based on Algorithm 6.2;
> Solve (6.11) for v_h;
> $\delta_t = \mathrm{CFL}\dfrac{h}{\|v_h\|}$;
> $t := t + \delta_t$;
> Solve (6.13) for ϕ_h at time t based on v_h and the method introduced in
>   Chapter 5;
> Update the interface with this new ϕ_h;

6.3 Some numerical test cases

6.3.1 Linear model

We consider a simpler version of (6.6) and (6.7) which is introduced in [55]. In this version, the reaction term will be streamlined to a linear function of u. The parameters are chosen to approximate the behavior of the nonlinear problem (6.6).

With the same interface condition and boundary condition as in (6.6) and (6.7), we consider a problem of u, v in $\Omega = [0,0.5] \times [0,0.5]$ (of size mm) such that,

$$\begin{cases} -\nabla \cdot (k_S \nabla u) & = -\alpha u \text{ in } \Omega, \\ -\nabla \cdot (\nabla v) & = -\beta u \text{ in } \Omega. \end{cases} \tag{6.14}$$

The parameters are chosen as follows,

$$k_S = \begin{cases} 120 \text{ in } \Omega_1, \\ 150 \text{ in } \Omega_2. \end{cases} \quad \alpha = \begin{cases} 3.6 \times 10^6 \text{ in } \Omega_1, \\ 0 \text{ in } \Omega_2. \end{cases} \quad \beta = \begin{cases} 10^6 \text{ in } \Omega_1, \\ 0 \text{ in } \Omega_2. \end{cases}$$

The Dirichlet condition at the top of domain: $\bar{S} = 10^{-5} \frac{\mathrm{mgO_2}}{\mathrm{mm^3}}$. The initial interface is chosen as a semi-circle $(x - 0.25)^2 + y^2 = r_0^2$.

The importance of penalty terms. As mentioned in Section 2.3, the choice of λ_u, λ_v in (6.10) and (6.11) are very important. Figure 6.2 shows a very bad result if we use a smaller value of $\hat{\lambda}_u$ and $\hat{\lambda}_v$ (10^2 and 10^4 respectively) while Figure 6.3 gives a more stable solution if we use a bigger of them (10^6 and 10^8 respectively).

a. $t = 0$ (day)   b. $t = 1.52$ (day)   c. $t = 3.22$ (day)
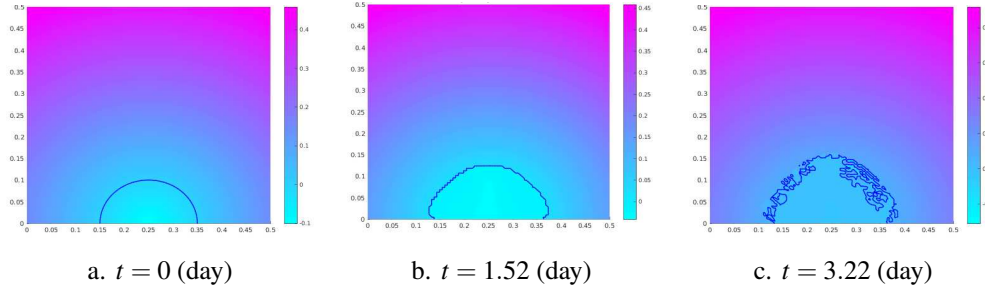
Figure 6.2. The interface and the value of ϕ at different time steps (days) when we use low values of $\hat{\lambda}_u, \hat{\lambda}_v$.

a. $t = 0$ (day)   b. $t = 19.91$ (day)   c. $t = 39.35$ (day)
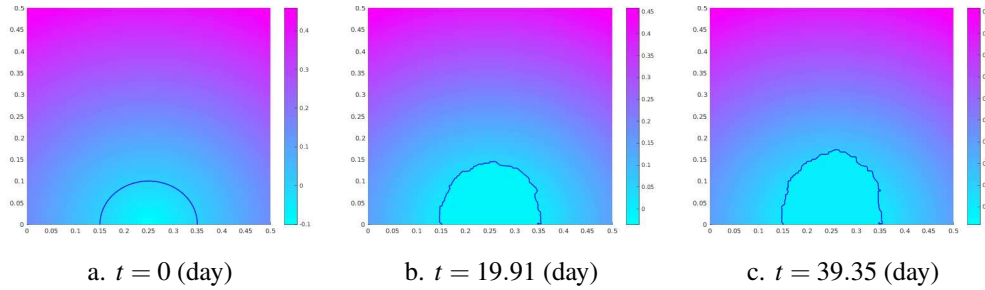
Figure 6.3. The interface and the value of level set function ϕ at different time steps (days) when we use high values of $\hat{\lambda}_u, \hat{\lambda}_v$.

The speed of biofilm's growth. Figure 6.4 suggests that the growths of biofilm is not equal at all time steps. For the first few days of growing, it grows very fast. After that, the rate is slower with time. It can be explained by an affect of constant value of substrate at the top of the domain and the height of the biofilm changes. Other words, the distance between the top of the biofilm and the top of the domain is reduced and it seems that bacteria need more time to feed themselves when their community is bigger.

Another aspect of their growth is when we change the value of β (from 10^6 to 10^8) which stands for the net rate of the active biomass and EPS production. Figure 6.5 illustrates this point clearly. To get the same height of the biofilm as in Figure 6.4, it takes only 5 *hours* (in comparison with *days* in the other test). We can understand this case easily. When we increase an amount of the active biomass and the rate of growth, bacteria is more active to find and eat the nutrient. The same phenomenon occurs if we increase an amount of the substrate on the top of the domain (from 10^{-5} $\frac{mgO_2}{mm^3}$ to 10^{-3} $\frac{mgO_2}{mm^3}$), bacteria have a chance to eat more nutrient than usual so they grow faster.

The importance of Ghost Penalty. In the test of without using ghost penalty terms,

a. $t = 0$ (day)   b. $t = 3.45$ (day)   c. $t = 8.52$ (day)

d. $t = 10.26$ (day)   e. $t = 30.01$ (day)   f. $t = 39.35$ (day)
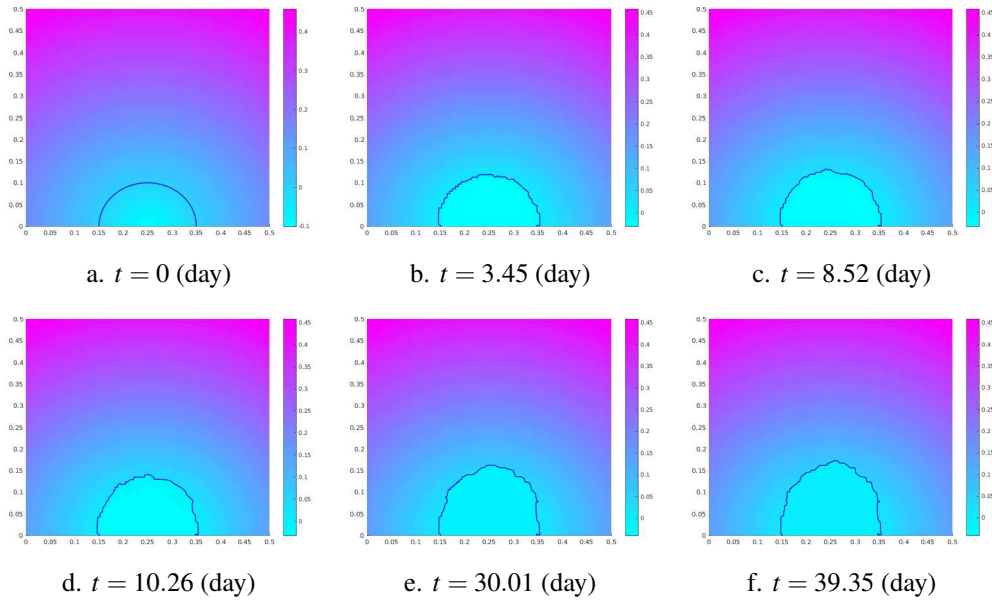
Figure 6.4. The interface and the value of the level set function ϕ at different time steps (days) when we test a regular of speed of growth.

although we keep all other parameters as in the test of Figure 6.4, we cannot obtain good results because it takes up to 36 days for the first little change.

a. $t = 0$ (hour)   b. $t = 1.68$ (hour)   c. $t = 5.04$ (hour)
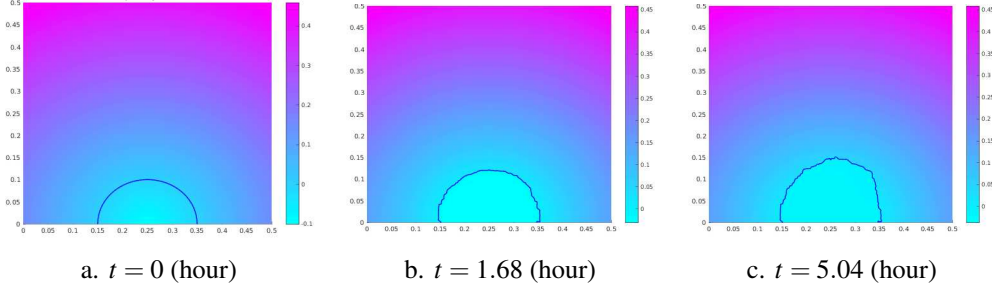
Figure 6.5. The interface and the value of the level set function ϕ at different time steps (hours) when we use bigger values of β or with more substrates.

6.3.2   Nonlinear model

Go back to the problem (6.6) and (6.7), using real parameters given in literature. These experimental parameters are given in Section B.1, they are used to compute $k_S, \bar{\mu}_S, \bar{\mu}_\Phi$ in a compound form. In this thesis, I use already-computed values of $k_S, \bar{\mu}_S, \bar{\mu}_\Phi$ which are given as below,

$$k_S = \begin{cases} 146.88 \text{ in } \Omega_1, \\ 183.6 \text{ in } \Omega_2, \end{cases} \quad \bar{\mu}_S = \begin{cases} 8.54932 \text{ in } \Omega_1, \\ 0 \text{ in } \Omega_2, \end{cases} \quad \bar{\mu}_\Phi = \begin{cases} 8.28785 \text{ in } \Omega_1, \\ 0 \text{ in } \Omega_2, \end{cases}$$

and the top Dirichlet condition $\bar{S} = 8.3 \times 10^{-6} \frac{\text{mgO}_2}{\text{mm}^3}$.

Note that, we sometimes change the values of parameters to see the relation of components in the problem.

Dome-like structure. In this test, the interface is chosen as in the linear case (a semi-circle $(x-0.25)^2 + y^2 = r_0^2$) in the same domain $\Omega = [0,0.5] \times [0,0.5]$ (size of mm). In order to make sure that there is always enough substrate in the test case (and also that we don't want the biofilm reaches the source of nutrient too fast if we reduce the height of domain), we apply a Dirichlet boundary condition on an imagine top boundary Γ_S of the domain. Γ_S which depends on time t is a horizontal line measured from the top most point of the biofilm, denoted by L_S. The quantity L_S is assumed to be a constant at all times. An illustration of $\Gamma_S(t)$ is given in Figure 6.6. In the following test cases, we mainly take $L_S = 0.1$ mm.

These settings coupling with more complicated terms in the equations (6.6) and (6.7) in comparison with equations (6.14), we obtain a different behavior. In the linear case (cf. Figure 6.5), the colony grow into a *finger-like* structure which is toward to the subtrate (on top). In the nonlinear case, biofilm develops into a *dome-like* structure which is spreading in the horizontal direction (cf. Figure 6.8).
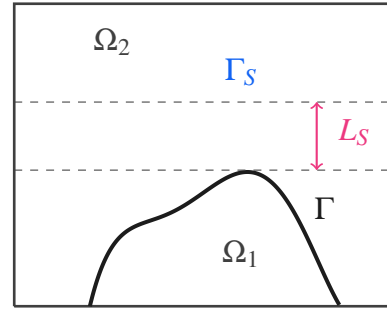
Figure 6.6. An illustration of dynamic domain.

We obtain the same behavior with the linear case when we increase the amount of substrate on Γ_S. More specifically, if we apply $\bar{S} = 1 \times 10^{-4} \frac{\text{mgO}_2}{\text{mm}^3}$ which is bigger than the case in Figure 6.8 where $\bar{S} = 8.3 \times 10^{-6} \frac{\text{mgO}_2}{\text{mm}^3}$, the biofilm grows faster (0.42 days in comparison with 3.22 days).

Interface with noise. In the next test case, we consider an interface which contains a noise $\hat{n}$ and this interface is height $\hat{h}$ mm from the bottom. With these noise and height, we can check the behavior of the biofilm's growth if it has a complicated shape (there are some *valleys* in it form, for example).

$$\phi(x,y) = y - \hat{h} + \hat{n} \times \cos(8\pi x). \tag{6.15}$$

> **Remark 6.6** When we say *"bigger/smaller" noise*, it means that we talk about the absolute value of $\hat{n}$ (i.e. $|\hat{n}|$). In (6.15), $|\hat{n}|$ decides the depth of the *valley* in the biofilm's shape. The sign of $\hat{n}$ indicates the location of the biofilm on the left of the domain.

First, we consider $\hat{h} = 0.1$ and $\hat{n} = 0.1$, the growth of biofilm is given in Figure 6.9. In this case, we take $\bar{S} = 1 \times 10^{-3}$ (there is much substrate), the growth is thus quick.

a. Substrate u at $t = 0$ (day)

b. Substrate u at $t = 3.22$ (day)

c. Potential velocity v at $t = 0$ (day)

d. Potential velocity v at $t = 3.22$ (day)
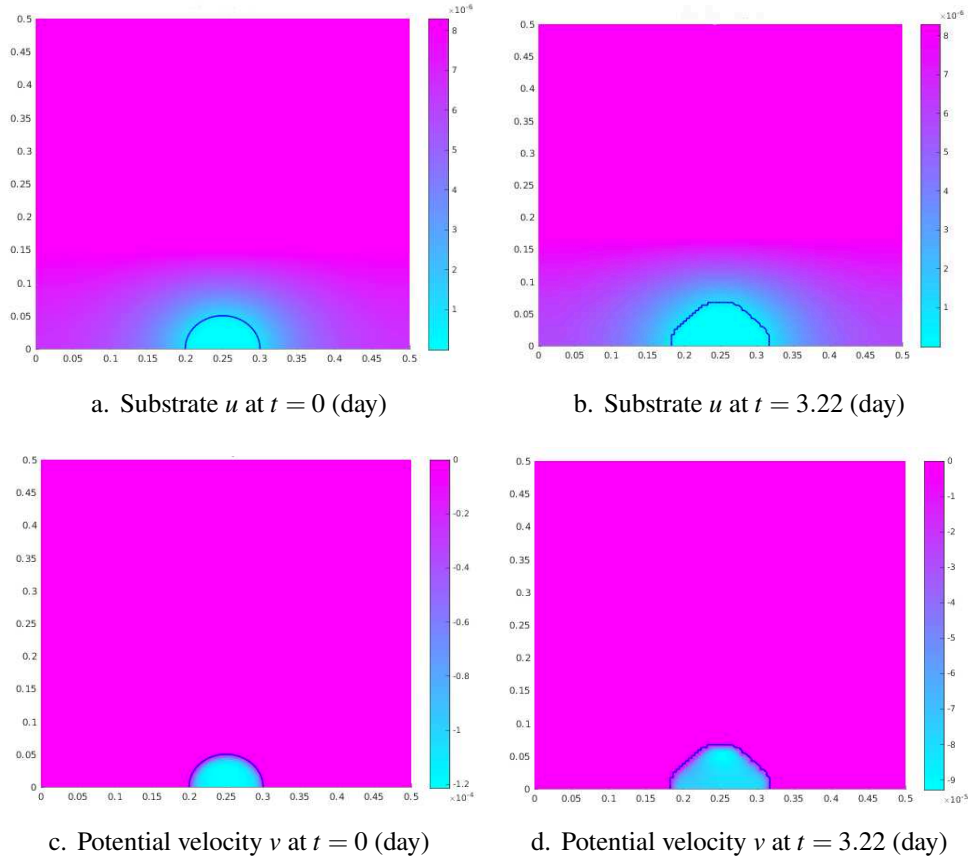
Figure 6.7. Dome-like structure of a biofilm growth when $\bar{S} = 8.3 \times 10^{-6} \frac{\text{mgO}_2}{\text{mm}^3}$.

a. Substrate u at $t = 0$ (day)

b. Substrate u at $t = 0.42$ (day)
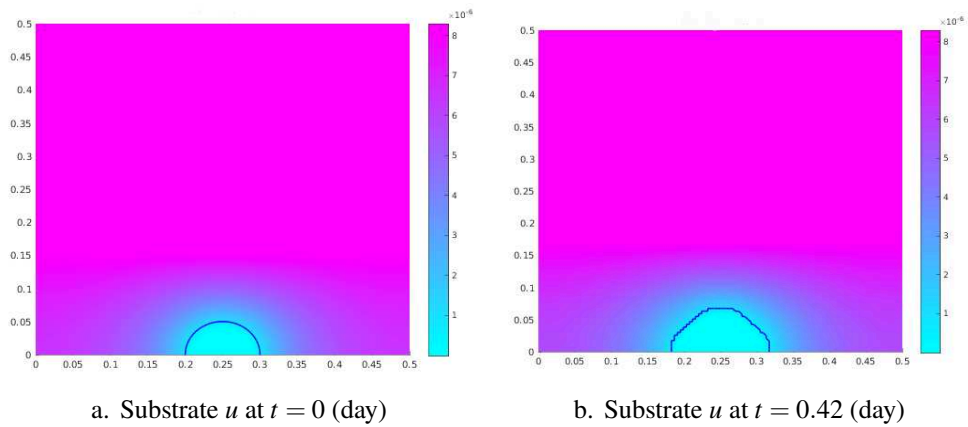
Figure 6.8. Dome-like structure of a biofilm growth when $\bar{S} = 1 \times 10^{-4} \frac{\text{mgO}_2}{\text{mm}^3}$.

One can see that, with a low of noise, biofilm develops equally at different location of its shape.

a. $t = 0$ (hour)   b. $t = 2.64$ (hour)   c. $t = 4.8$ (hour)
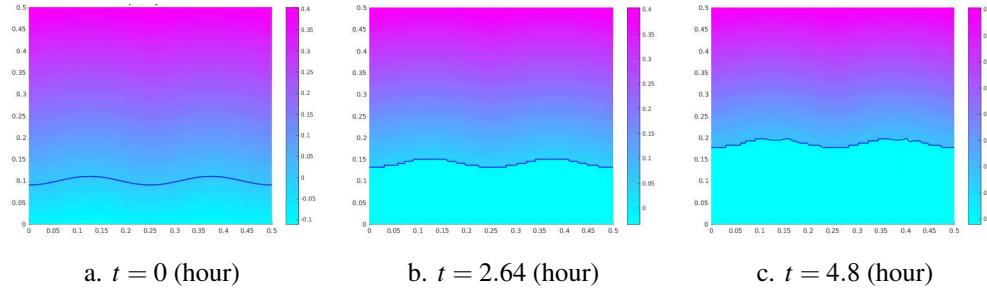
Figure 6.9. The height of biofilm at different times (hour).

However, if we take different levels of noise, we will see the differences in Figure 6.10. With more noise (the right figure), the growth of biofilm at the top of *moutain* is faster than others in the *valley*. You can check the color in the plot, the *brighter color* inside the biofilm region indicates the *bigger potential v* velocity. In contrast, if the noise is trivial (the left figure), the growth of biofilm seems to be equally at all locations. We can check that there is no much brighter places then the others in the biofilm region.

a. $\hat{n} = -0.1$ b. $\hat{n} = -0.3$
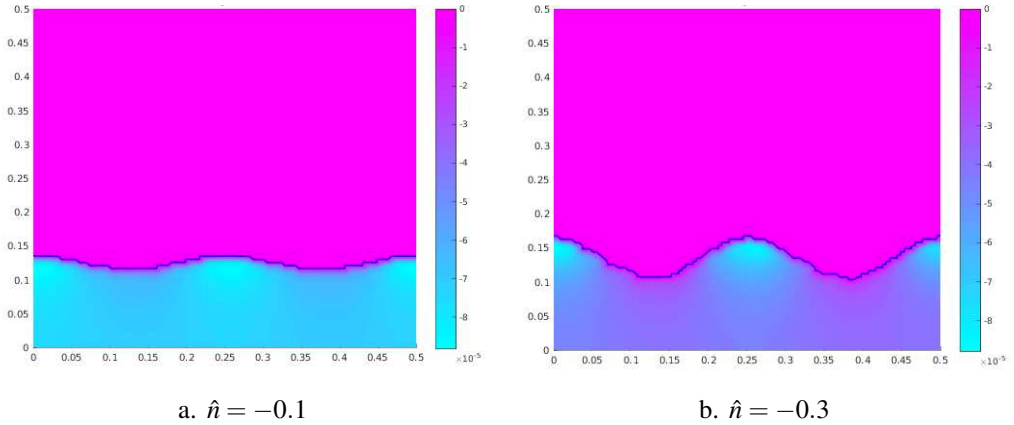
Figure 6.10. Value of potential velocity v for different values of noise at time around $t = 4$ (day).

For the last test, we consider an interesting example in which we change the place the substrate occurs. Look at Figure 6.11, if the substrate comes from the top of the domain, it takes longer for the biofilm to reach to the height of 0.15 mm (1.21 days) whereas, it takes only 0.24 days for it to reach to the same height if substrate occurs at the bottom. We can understand this easily. Biofilm is nearer the source of nutrient in the case of bottom than the case of top, its faster growth is reasonable. Note that, in Figure 6.11, the plot of potential velocity also gives us a remark. In the case of top, the highest place of biofilm grows more quickly than other locations. In the case of bottom, the growth of biofilm seems equally for all locations on it.

a. From top of domain at 1.21 (day).   b. From bottom of domain at 0.24 (day).
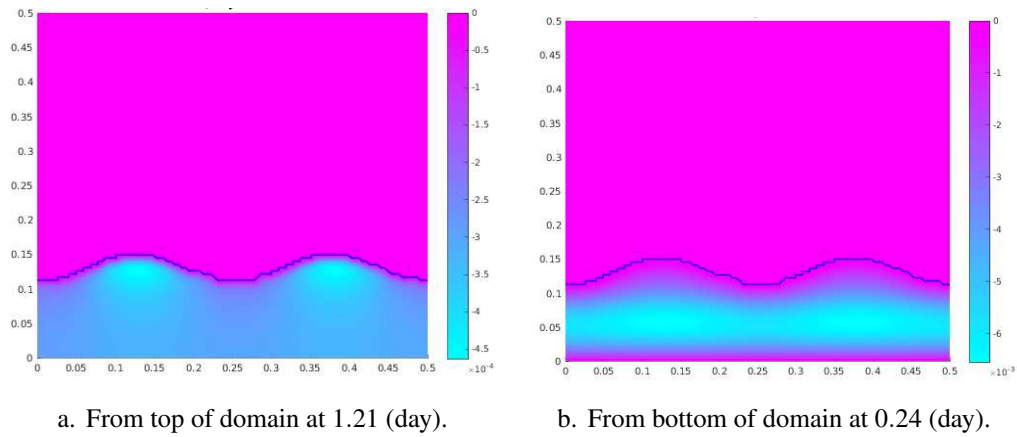
Figure 6.11. Value of potential velocity v_h corresponding to different locations of substrate's occurance.

7. Conclusion

Contents

7.1   The methods

For such a problem like the problem biofilm growth, we need to combine many different methods. We use NXFEM method to solve an interface problem with an unfitted mesh. With this method, we don't have to generate a new mesh many times, just use one mesh for the whole process. However, the NXFEM method is not so stable as it has to be, especially when the interface cuts the mesh at very different parts. We need (and have to) apply some stabilized methods such as Ghost Penalty to our scheme. The choice of parameters is also very sensitive and important if we want to get a good result. There is not any clear standard in which we can choose good values for them but we can do it empirically and get the right ones depending on a specific model.

For capturing the change of an interface (e.g. biofilm's shape), we need to use the Level Set Method. This one is very modern and a good choice for this purpose. However, working with it on a triangle mesh is another problem we have to handle.

Thanks to the Streamline Upwinding Petrov-Galerkin method coupling with the Fast Marching Method and Crank-Nicolson scheme, we can comfortably work with LSM. Notice that, although FMM helps us make the level set function be a signed distance function as it has to be, we cannot apply this kind of method every time we want. The number of times we apply FMM in solving a level set equation is another difficulty and it's again chosen empirically.

In practice, working with nonlinear equations on each iteration or each time step, we sometimes use the Newton method to get a discrete solution. Generally, we don't have too many troubles with this method when working with a biofilm's model.

7.2 The NXFEM toolbox

`NXFEM toolbox` is built for the implementation of the NXFEM method and the Level Set Method. All of algorithms are explained carefully to those who want to use it in the future. For the current version, it works well only on a single core at the time of executing. This is a disadvantage and it takes a little more time to get the results for tests in this thesis. Of course, the toolbox is under the way of updating to a parallel version and open to everyone to contribute.

7.3 For the future

In this work, we have only worked on problems in which the diffusion coefficients are constant in each subdomain. It's intersting to approach the cases with nonlinear diffusive terms.

The growth of biofilm is also affected by the presence of antibiotics. There are some new resistance mechanisms which only appear if there is an antibiotics in the environment. Continue to the models in this thesis, we can add more terms relating to this resistance and research the growth of biofilm on these cases. Of course, we will examine the phenomena with the NXFEM method and `NXFEM toolbox`.

Remark 6.3 mentioned about the removing of time dependent term in the model. It's because the ontained values in the cases we were working on change very little in each time iteration. However, for a more realistic phenomenon with more relations in the model, it's neccessary to work with a full evolution system.

A. Implementation and NXFEM toolbox

Contents

A.1 Some principles of quadrature

Definition A.1 — Quadrature. [65] Let D be a non-empty, Lipschitz, compact, connected subset of $\mathbb{R}^n$. Let l_q be an integer. A quadrature is an approximation of the definite integral of a function. It's usually stated as a weighted sum of function values at specific points within domain of integration. So, a quadrature on D with l_q points consists of

(i) A set of l_q real numbers $\{\omega_1,\ldots,\omega_q\}$ called *quadrature weights*.

(ii) A set of l_q points $\{\xi_1, \ldots, \xi_q\}$ in D called *Gaussian points* or *quadrature nodes*.

The largest integer k such that

$$\forall g \in \mathbb{P}_k, \int_D g(x)\,\mathrm{d}x = \sum_{q=1}^{l_q} \omega_q g(\xi_q),$$

is called the *quadrature order* and is denoted by k_q.

In this thesis, I will use an *n*-point *Gaussian quadrature rule* which is a quadrature rule constructed to yield *an exact* result for polynomials of degree $2n - 1$ or less by using suitable couples $\{\omega_q, \xi_q\}$ for $q = 1, \ldots, n$. More specifically, I apply Gaussian quadrature only for type of domain which is a segment (in dimension 1) or a triangle (in dimension 2). Note that, *n*-point Gaussian quadrature is corresponding to quadrature order $k_q = 2n + 1$ (see the proof in [65, Proposition 8.2]).

1D Case. On the *reference interval* $[-1, 1]$, we use formula (A.1), on a general interval $[a, b]$, we use formula (A.2).

$$\int_{-1}^{1} g(x)\,\mathrm{d}x \simeq \sum_{q=1}^{l_q} \omega_q g(\xi_q). \tag{A.1}$$

$$\int_a^b g(x)\,\mathrm{d}x \simeq \frac{b-a}{2} \sum_{q=1}^{l_q} \omega_q g\left(a + \frac{b-a}{2}(1 + \xi_q)\right). \tag{A.2}$$

Remark A.1 Because we mainly consider the $\mathbb{P}_1$-FE space (or $\mathbb{P}_2$-FE in some cases), we thus don't need too many number of Gaussian points for the quadrature rules. It depends on the degree of polynomial function g which we are going to find the integration on D. A table of Gaussian points and weights in 1D for formula (A.1) is given in Table A.1. It's corresponding to function `getGaussQuad` in the toolbox (there are more points and weights in this function).

2D Case. On a *reference triangle* $\hat{K}$, we use formula (A.3). On a general triangle K, we use formula (A.4).

l_q	k_q	ξ_q	Aprx	ω_q	Aprx
1	1	0	0	2	2
2	3	$\pm\frac{1}{\sqrt{3}}$	± 0.57735	1	1
3	5	0	0	$\frac{8}{9}$	0.888889
		$\pm\sqrt{\frac{3}{5}}$	± 0.774597	$\frac{5}{9}$	0.555556
4	7	$\pm\sqrt{\frac{3}{7}-\frac{2}{7}\sqrt{\frac{6}{5}}}$	± 0.339981	$\frac{18+\sqrt{30}}{36}$	0.652145
		$\pm\sqrt{\frac{3}{7}+\frac{2}{7}\sqrt{\frac{6}{5}}}$	± 0.861136	$\frac{18-\sqrt{30}}{36}$	0.347855

Table A.1. Some Gauss–Legendre quadrature couples $\{\omega_q, \xi_q\}$ on the reference interval $[-1,1]$.

$$\int_{\hat{K}} g(\hat{x})\, d\hat{x} \simeq \frac{1}{2} \sum_{q=1}^{l_q} \omega_q g(\hat{x}_q). \tag{A.3}$$

$$\int_{K} g(x)\, dx\, dy = 2|K| \int_{\hat{K}} g(\mathbf{P}(\hat{x}))\, d\hat{x} \simeq |K| \sum_{q=1}^{l_q} \omega_q g(\mathbf{P}(\hat{x}_q)), \tag{A.4}$$

where $|K|$ is the area of K and $\mathbf{P}$ is a mapping which transforms each vertex of $\hat{K}$ in $O\hat{x}\hat{y}$ to corresponding vertex of K in Oxy, cf. Figure A.1. In $\mathbb{P}^1$-FE, $\mathbf{P}$ which is given in (A.5) is corresponding to function `getCoorSTD` in the toolbox.
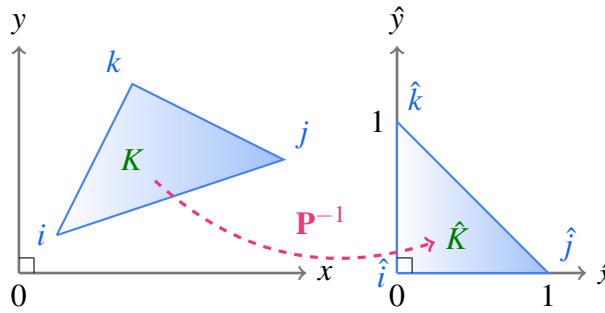
Figure A.1. Quadrature in 2D between a general triangle K in Oxy and its reference triangle $\hat{K}$ in $O\hat{x}\hat{y}$.

$$\begin{pmatrix} x \\ y \end{pmatrix} = \mathbf{P} \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} x_i(1-\hat{x}-\hat{y}) + x_j\hat{x} + x_k\hat{y} \\ y_i(1-\hat{x}-\hat{y}) + y_j\hat{x} + y_k\hat{y} \end{pmatrix}, \tag{A.5}$$

where $\{(x_i, y_i), (x_j, y_j), (x_k, y_k)\}$ is the coordinate of triangle K in Oxy. These vertices are numbered in the counter-clockwise direction. Note that, we also have $dx\,dy = 2|K|\,d\hat{x}\,d\hat{y}$.

> **Remark A.2** We also have the inverse of $\mathbf{P}$ in the case of $\mathbb{P}^1$-FE as follow
>
> $$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \mathbf{P}^{-1} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \frac{1}{2|K|}\left((x-x_i)(y_k-y_i)-(y-y_i)(x_k-x_i)\right) \\ \frac{1}{2|K|}\left(-(x-x_i)(y_j-y_i)+(y-y_i)(x_j-x_i)\right) \end{pmatrix}.$$
>
> It's corresponding to function `getCoorRef` in the toolbox. A table of some Gaussian points and weights in 2D for formula (A.3) is given in Table A.2 (there are more points and weights in the function `getGaussQuad`).

l_q	k_q	$\{\hat{x}_q; \omega_q\}$
1	1	$\left\{\left(\frac{1}{3}\,\frac{1}{3}\right); 1\right\}$
3	2	$\left\{\left(\frac{1}{2}, \frac{1}{2}\right); \frac{1}{3}\right\}, \left\{\left(\frac{1}{2}, 0\right); \frac{1}{3}\right\}, \left\{\left(0, \frac{1}{2}\right); \frac{1}{3}\right\}$
4	3	$\left\{\left(\frac{1}{3}, \frac{1}{3}\right); -\frac{27}{48}\right\}, \left\{\left(\frac{1}{5}, \frac{1}{5}\right); \frac{25}{48}\right\}, \left\{\left(\frac{1}{5}, \frac{3}{5}\right); \frac{25}{48}\right\}, \left\{\left(\frac{3}{5}, \frac{1}{5}\right); \frac{25}{48}\right\}$

Table A.2. Some Gauss–Legendre quadrature couples $\{\hat{x}_q, \omega_q\}$ on the reference triangle $\hat{K}$.

Quadrature in FEM. Consider the integral $\int_\Omega g(\mathbf{x})\,d\mathbf{x}$ where g is a smooth function. Using the decomposition,

$$\int_\Omega g(u(\mathbf{x}))\,d\mathbf{x} = \sum_{K \in \mathcal{T}_h} \int_K g(u(\mathbf{x}))\,d\mathbf{x},$$

we can change the computation from integrating on the whole domain Ω to integrating on each triangle K. Thanks to quadrature rules (A.4), we again change to integrating on a reference triangle $\hat{K}$ which is much easier.

$$\begin{aligned}
\int_\Omega g(u(\mathbf{x}))\,d\mathbf{x} &= \sum_{K \in \mathcal{T}_h} \int_K g(u(\mathbf{x}))\,d\mathbf{x} = \sum_{K \in \mathcal{T}_h} 2|K| \int_{\hat{K}} g(\mathbf{P}(u(\hat{x})))\,d\hat{x} \\
&\simeq \sum_{K \in \mathcal{T}_h} \left(2|K| \sum_{q=1}^{l_q} \omega_q g(\mathbf{P}(u(\hat{x}_q))) \right).
\end{aligned} \tag{A.6}$$

In NXFEM or in general FEM, we are working on basic functions $\{\varphi_i\}_i$ in $\mathbb{P}^1$-FE space or $\mathbb{P}^2$-FE space. We cannot compute exactly the value of integrals with the

types of these functions because they are very complicated. The job is much easier if we use quadrature rules and change to work on the local shape functions on reference triangles which are determined in Table A.3.
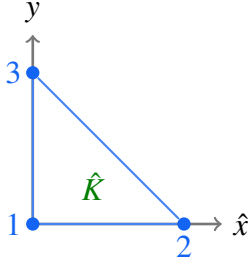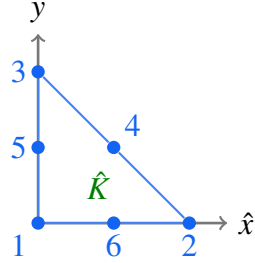
$\mathbb{P}^1$-FE	$\mathbb{P}^2$-FE
$N_1(\hat{x},\hat{y}) = 1 - \hat{x} - \hat{y},$ $N_2(\hat{x},\hat{y}) = \hat{x},$ $N_3(\hat{x},\hat{y}) = \hat{y}.$	$N_1(\hat{x},\hat{y}) = 1 - 3\hat{x} - 3\hat{y} + 2\hat{x}^2 + 4\hat{x}\hat{y} + 2\hat{y}^2,$ $N_2(\hat{x},\hat{y}) = 2\hat{x}^2 - \hat{x},$ $N_3(\hat{x},\hat{y}) = 2\hat{y}^2 - \hat{y},$ $N_4(\hat{x},\hat{y}) = 4\hat{x}\hat{y},$ $N_5(\hat{x},\hat{y}) = 4\hat{y} - 4\hat{x}\hat{y} - 4\hat{y}^2,$ $N_6(\hat{x},\hat{y}) = 4\hat{x} - 4\hat{x}\hat{y} - 4\hat{x}^2.$
function `getP1shapes` in the toolbox	function `getP2shapes` in the toolbox

Table A.3. Local shape functions defined on reference triangle $\hat{K}$ where $N_i(x_j) = \delta_{ij}$.

In this chapter, I build all stiffness matrices and load vectors based on the idea of Propotition A.1 given as below.

Proposition A.1 Let φ_i be a basis function defined in $\mathbb{P}^k$-FE and $\tilde{\varphi}_{ir}$ is its corresponding shape function determined on element $K \in \mathscr{T}_h$ at vertex r ($r = 1, 2, 3$). Let N_r be the corresponding shape function w.r.t $\tilde{\varphi}_r$ on the reference triangle $\hat{K}$. Then we have the relations

$$\tilde{\varphi}_{ir}(x,y) = N_{ir}(\hat{x},\hat{y}), \ \forall (x,y) = \mathbf{P}(\hat{x},\hat{y}), r = 1, 2, 3.$$
$$\int_K g(\tilde{\varphi}_{ir}(x,y)) \, dx \, dy = 2|K| \int_{\hat{K}} g(N_{ir}(\hat{x},\hat{y})) \, d\hat{x} \, d\hat{y}.$$

where g is a smooth function well-defined on K and $\hat{K}$.

A.2 Connectivity of triplet `[p,e,t]`

An example of mesh data `[p,e,t]` given in Matlab PDE toolbox is illustrated by Figure A.2 and following description.
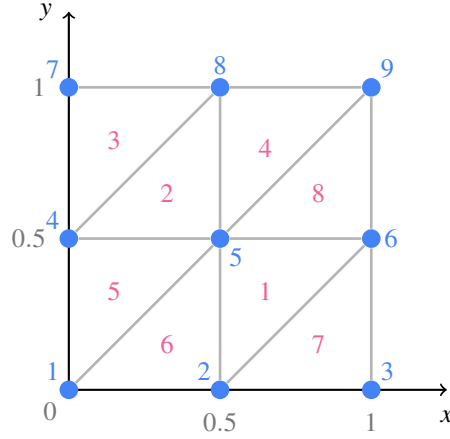
`p` is a $2 \times$ *number of nodes* matrix (cf. (A.7)). Each column of `p` represents a node of the mesh (blue number in Figure A.2). Its first row contains the *x*-coordinates of these nodes and the second row for *y*. For example, if we want to consider the *y*-coordinate of the 5th node, we take `p(5,2)`.

$$p = \begin{bmatrix} 0.0 & 0.5 & 1.0 & 0.0 & 0.5 & 1.0 & 0.0 & 0.5 & 1.0 \\ 0.0 & 0.0 & 0.0 & 0.5 & 0.5 & 0.5 & 1.0 & 1.0 & 1.0 \end{bmatrix}, \tag{A.7}$$

`t` is a $4 \times$ *number of elements* matrix (cf. (A.8)). Each column of `t` represents an element of the mesh (red number in Figure A.2). Each element have three vertices which are presented as three first numbers on each column. Each number is the index of node in `p`. For example, `t(3,2) = 8` is the 8-th point of the mesh, i.e. `p(:,8)`.

$$t = \begin{bmatrix} 2 & 4 & 4 & 5 & 1 & 1 & 2 & 5 \\ 6 & 5 & 8 & 9 & 5 & 2 & 3 & 6 \\ 5 & 8 & 7 & 8 & 4 & 5 & 6 & 9 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{A.8}$$

`e` is a $7 \times$ *number of edges on the boundary* matrix (cf. (A.9)). We focus on two first rows of this matrix. It contains the indices of two endpoints in `p` of these edges. For example, the bottom boundary in Figure A.2 contains two edges. The first one is the connection between node 1 and 2, that is `e(:,1)`. The second is $2 - 3$ which is `e(:,2)`.

$$
\mathbf{e} =
\begin{bmatrix}
1 & 2 & 3 & 6 & 9 & 8 & 7 & 4 \\
2 & 3 & 6 & 9 & 8 & 7 & 4 & 1 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots
\end{bmatrix},
\tag{A.9}
$$

Notice that, the elements connectivities are all ordered in a counter-clockwise fashion, for example, 5-th element has three vertices $1 \rightarrow 5 \rightarrow 4$.

A.3 Proof of formula used in finding intersection points

We will prove (3.5). Indeed, because $\phi \in V_h$, ϕ has a form of $ax + by + c$ where $a, b, c \in \mathbb{R}$. We also recall that $\phi_i = \phi(x_i, y_i), \phi_k = \phi(x_k, y_k)$, then

$$
\begin{aligned}
\phi(x_0, y_0) &= a\frac{x_i}{2}\left(1 - \frac{\phi_i + \phi_k}{\phi_i - \phi_k}\right) + a\frac{x_k}{2}\left(1 + \frac{\phi_i + \phi_k}{\phi_i - \phi_k}\right) \\
&\quad + b\frac{y_i}{2}\left(1 - \frac{\phi_i + \phi_k}{\phi_i - \phi_k}\right) + b\frac{y_k}{2}\left(1 + \frac{\phi_i + \phi_k}{\phi_i - \phi_k}\right) + c \\
&= \frac{\phi_i}{2}\left(1 - \frac{\phi_i + \phi_k}{\phi_i - \phi_k}\right) + \frac{\phi_k}{2}\left(1 + \frac{\phi_i + \phi_k}{\phi_i - \phi_k}\right) = 0.
\end{aligned}
$$

A.4 Example of finding intersections

Let us consider a cut triangle, namely `i-j-k`, where `i`, `j`, `k` are its three vertices in that order in the matrix of triangles `t`. We will list all possible cases in Table A.4 Example of determining intersections between cut triangle and interface.

A.5 Example of finding unit normal vector

Algorithm 3.1 in Section 3.3.4 describes how to get intersections between interface and cut triangles. These intersections are later applied to find unit normal vector lying interface segment Γ_h. In this section, I will give a detailed example explaining them.

An example group of cut triangles which contains intersections used to find unit normal vectors are clarified in Table A.5.

A.6 Implementation issue of some norms in NXFEM toolbox

A.6.1 Norms in standard finite element space V_h

In `NXFEM toolbox`, the norms given in (5.19) for finding errors of level set functions are computed via following expressions.
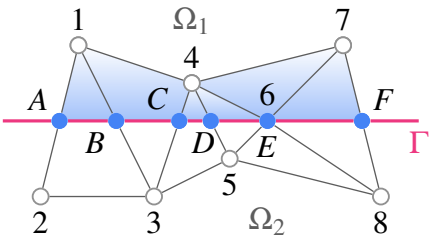
Table A.4. Example of determining intersections between cut triangle `i-j-k` (in that order) and interface Γ_h. They are all stored in `CT.iPs` in order of $[A, B]$.

- For all $\phi \in L^2(\Omega)$,

$$\|\phi(\mathrm{x})\|^2_{L^2(\Gamma_h)} = \sum_{K \in G_h} \int_{\Gamma_{K,h}} \phi^2(\mathrm{x}) \, ds = \sum_K \int_{X_a}^{X_b} \phi^2(\mathrm{x}) \, ds$$

$$= \sum_K \frac{|X_a X_b|}{|\hat{X}_a \hat{X}_b|} \int_{\hat{X}_a}^{\hat{X}_b} \phi^2(\mathbf{P}(\hat{\mathrm{x}})) \, d\hat{s} = \sum_K |X_a X_b| \int_0^1 \phi^2(\mathbf{P}(\hat{\mathrm{x}}(t))) \, dt$$

$$= \sum_K \frac{1}{2} |X_a X_b| \sum_{q=1}^{l_q} \omega_q \phi^2 \left(\mathbf{P} \left(\hat{\mathrm{x}} \left(\frac{1 + \xi_q}{2} \right) \right) \right)$$

$$= \sum_K \frac{1}{2} |X_a X_b| \sum_{q=1}^{l_q} \omega_q \phi^2 \left(\mathbf{P} \left(\hat{\mathrm{x}}_a + \frac{\hat{\mathrm{x}}_b - \hat{\mathrm{x}}_a}{2} (1 + \xi_q) \right) \right)$$

$$=: \texttt{getNormL2foGh}^2(\phi),$$

where $X_a(\mathrm{x}_a), X_b(\mathrm{x}_b)$ are two endpoints of $\Gamma_{K,h}$, $\mathbf{P}$ is a mapping which transforms

figure			cut triangles				
		i	1	3	3	6	6
		j	2	4	5	4	8
		k	3	1	4	5	7
		CT.type	4	2	4	0	0
		CT.iPs	$[A,B]$	$[C,B]$	$[D,C]$	$[D,E]$	$[F,E]$
		CT.uN$^\perp$	$[B,A]$	$[C,B]$	$[D,C]$	$[E,D]$	$[F,E]$

Table A.5. Example of determining unit normal vectors `CT.uN` based on intersections `CT.iPs`. This result follows Algorithm 3.2. Note that, `CT.uN`$^\perp$ is an orthogonal vector of `CT.uN`.

each vertex of $\hat{K}$ in $O\hat{x}\hat{y}$ to vertices of K in Oxy and (ω_q, ξ_q) is a quadrature couple given in Table A.1.

- For $\phi_h \in V_h, \phi \in L^2(\Omega)$,

$$\|\phi_h - \phi\|^2_{L^2(\Omega)} = \int_\Omega (\phi_h - \phi)^2 \, dx = \int_\Omega (\phi_h)^2 \, dx + \int_\Omega \phi^2 \, dx - 2 \int_\Omega \phi_h \phi \, dx$$
$$= \|\phi_h\|^2_{L^2(\Omega)} + \|\phi\|^2_{L^2(\Omega)} - 2 \int_\Omega \phi_h \phi \, dx \tag{A.10}$$
$$=: \texttt{getNormL2fhfSTD}^2(\phi_h, \phi),$$

where $\|\phi_h\|_{L^2(\Omega)}$, $\|\phi\|_{L^2(\Omega)}$ and $\int_\Omega \phi_h \phi \, dx$ are constructed based on (A.11), (A.13) and (A.14) respectively.

- For all $e_h = \Sigma_i e_i \varphi_i \in V_h$,

$$\|e_h\|^2_{L^2(\Omega)} = \int_\Omega e_h e_h \, dx = \int_\Omega (\Sigma_i e_i \varphi_i)(\Sigma_j e_j \varphi_j) \, dx = \sum_i \sum_j e_i \int_\Omega \varphi_i \varphi_j \, dx \, e_j$$
$$= E^T M E =: \texttt{getNormL2fhSTD}^2(e_h), \tag{A.11}$$

where $E = (e_i)_i$ and

$$M_{ij} = \int_\Omega \varphi_j \varphi_i \, dx = \sum_{K \in \mathscr{T}_h} \int_K \varphi_j(x) \varphi_i(x) \, dx = \sum_K 2|K| \int_{\hat{K}} N_j(\hat{x}) N_i(\hat{x}) \, d\hat{x}$$
$$= \sum_K |K| \sum_{q=1}^{l_q} \omega_q N_j(\hat{x}_q) N_i(\hat{x}_q) \tag{A.12}$$
$$= \sum_K \texttt{getTriplePPWhole}(K, j, i)$$
$$=: \texttt{getTriplePPNCTs}.$$

- For all function $f \in L^2(\Omega)$,

$$
\begin{aligned}
\|f\|_{L^2(\Omega)}^2 &= \sum_{K \in \mathscr{T}_h} \int_K f^2(\mathbf{x})\,\mathrm{d}\mathbf{x} = \sum_K 2|K| \int_{\hat{K}} f^2(\mathbf{P}(\hat{\mathbf{x}}))\,\mathrm{d}\hat{\mathbf{x}} \\
&= \sum_K |K| \sum_{q=1}^{l_q} \omega_q f^2(\mathbf{P}(\hat{\mathbf{x}}_q)) \\
&=: \texttt{getNormL2fSTD}^2(f).
\end{aligned}
\tag{A.13}
$$

> **Remark A.3** Because f doesn't depend on the finite element space V_h^{Γ}, we don't need to construct a function like `getNormL2fNX` for the case of NXFEM space. The function `getNormL2fSTD` is enough for both cases.

- For all $f_h = \Sigma_i f_i \varphi_i \in V_h$, $f \in L^2(\Omega)$,

$$
\int_{\Omega} f_h f\,\mathrm{d}\mathbf{x} = \sum_i f_i \int_{\Omega} f(\mathbf{x}) \varphi_i(\mathbf{x})\,\mathrm{d}\mathbf{x} = F_h F,
\tag{A.14}
$$

where $F_h = (f_i)_i$ and $F_i = \int_{\Omega} f(\mathbf{x}) \varphi_i(\mathbf{x})\,\mathrm{d}\mathbf{x}$ can be found via function `getfPhiNCTs` like in Algorithm 3.5.

- For all $f \in H^1(\Omega)$,

$$
\begin{aligned}
\|\nabla f\|_{L^2(\Omega)}^2 &= \int_{\Omega} (\partial_x f)^2 + (\partial_y f)^2\,\mathrm{d}\mathbf{x} \\
&= \texttt{getNormL2fSTD}^2(\partial_x f) + \texttt{getNormL2fSTD}^2(\partial_y f) \\
&=: \texttt{getNormL2GfSTD}^2(f).
\end{aligned}
\tag{A.15}
$$

A.6.2 Norms in NXFEM space V_h^{Γ}

In order to calculate the norm on NXFEM space, we follow the same technique as in section of assembling a stiffness matrix (cf. (3.9)), the difference in this case is on the form we are considering.

- For $e \in V_h^{\Gamma}$,

$$
\begin{aligned}
\|e\|_{L^2(\Omega)}^2 &= \int_{\Omega} |e|^2\,\mathrm{d}\mathbf{x} = \int_{\Omega} |\Sigma_{i=1}^{N_{\mathrm{new}}} e_i \varphi_i|\,\mathrm{d}\mathbf{x} \\
&= \langle \Sigma e_i \varphi_i, \Sigma e_j \varphi_j \rangle_{\Omega} = \Sigma_i \Sigma_j e_i \langle \varphi_i, \varphi_j \rangle e_j \\
&= E A_{L^2} E^T,
\end{aligned}
\tag{A.16}
$$

where $E = (e_i)_{i=1,\ldots,N_{\mathrm{new}}}$ and matrix A_{L^2} is computed as in the illustration of Figure A.3. In diagram A.3, function `getTriplePPNCTs` can be constructed as in (A.12)

and function `getTriplePPPart` can be constructed as in (A.17) (the same idea with `getfPhiPart` illustrated in Figure 3.8).
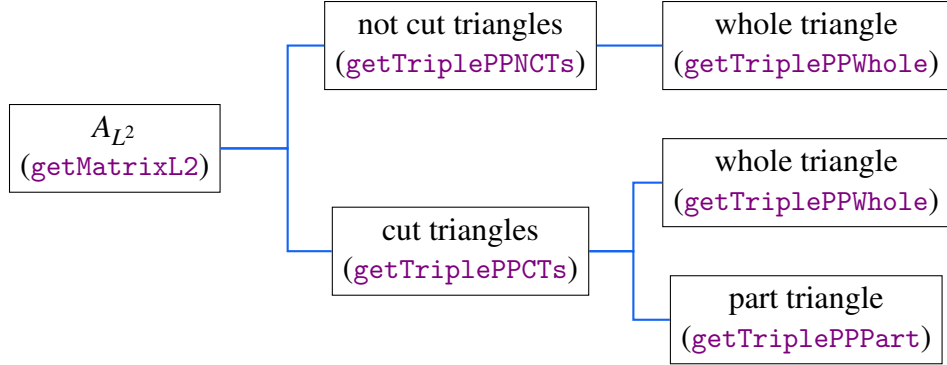
Figure A.3. The idea of assembling the matrix A_{L^2} for L^2 norm in NXFEM.

$$
\begin{aligned}
A_{L^2(K)}(\varphi_i^1, \varphi_j^1) &= \int_K \varphi_j^1(x)\varphi_i^1(x)\,dx = \int_{K_1} \varphi_j(x)\varphi_i(x)\,dx \\
&= 2|K| \int_{\hat{K}_1} N_j(\hat{x})N_i(\hat{x})\,d\hat{x} \\
&= 4|K||K_1| \int_{\tilde{K}} N_j(\mathbf{Q}(\tilde{x}))N_i(\mathbf{Q}(\tilde{x}))\,d\tilde{x} \\
&= 2|K||K_1| \sum_{q=1}^{l_q} w_q N_j(\mathbf{Q}(\tilde{x}_q))N_i(\mathbf{Q}(\tilde{x}_q)).
\end{aligned}
\tag{A.17}
$$

> **Remark A.4** Note that, the most difference between norms in NXFEM and norms in standard FEM is the terms computed on cut triangles. Thus, we can use the same function in `NXFEM toolbox` on not-cut triangles for both cases, i.e. `getTriplePPNCTs`, `getTriplePPWhole` can be used both in NXFEM and standard FEM.

• For H^1 norm in V_h^Γ, we apply the norm L^2 and functions `getTripleGGNCTs`, `getTripleGGCTs` like in the way we compute the stiffness matrix A (cf. Section 3.4.1).

• For all $f \in V_h^\Gamma, f \in L^2(\Omega)$,

$$
\begin{aligned}
\|f_h - f\|_{L^2(\Omega)}^2 &= \int_\Omega (f_h - f)^2\,dx = \int_\Omega f_h^2\,dx + \int_\Omega f^2\,dx - 2\int_\Omega f_h f\,dx \\
&= \|f_h\|_{L^2(\Omega)}^2 + \|f\|_{L^2(\Omega)}^2 - 2\int_\Omega f_h f\,dx \\
&=: \texttt{getNormL2fhfNX}^2(fh, f),
\end{aligned}
\tag{A.18}
$$

where $\|\phi_h\|_{L^2(\Omega)}$, $\|\phi\|_{L^2(\Omega)}$ and $\int_\Omega \phi_h \phi\,dx$ are constructed based on (A.16), (A.13) and (A.19) respectively.

- For all $f_h = \Sigma_i f_i \varphi_i \in V_h^\Gamma, f \in L^2(\Omega)$,

$$\int_\Omega fhf\,\mathrm{dx} = \sum_{i=1}^{N_{\text{new}}} f_i \int_\Omega f(\mathrm{x})\varphi_i\,\mathrm{dx} = F_h F, \qquad (A.19)$$

where $F_h = (f_i)_i$ and $F_i = L_h(\varphi_i)$ can be computed as in (3.12).

> **Remark A.5** There are differences between `getNormL2fhfSTD` in (A.10) and `getNormL2fhfNX` in (A.19). The former is computed on standard FE space V_h whereas the latter is computed on NXFEM space V_h^Γ.

- For $e_h \in V_h^\Gamma$,

$$\begin{aligned}
\|\nabla e\|_{L^2(\Omega)}^2 = \int_\Omega |\nabla e|^2\,\mathrm{dx} &= \int_\Omega |\Sigma_{i=1}^{N_{\text{new}}} e_i \varphi_i| \\
&= \Sigma_i \Sigma_j e_i \langle \varphi_i, \varphi_j \rangle e_j = E G_{L^2} E^T \\
&= \texttt{getNormL2GfhNX}^2(e),
\end{aligned} \qquad (A.20)$$

where $E = (e_i)_{i=1,\dots,N_{\text{new}}}$ and matrix G_{L^2} is computed as in (3.9) thanks to functions `getTripleGGNCTs` and `getTripleGGCTs` in `NXFEM toolbox`.

A.7 Interpolation between V_h and V_h^Γ

A.7.1 From V_h^Γ to V_h

We use functions `pdemesh`, `pdeplot` and `pdesurf` from the PDE toolbox to illustrate the mesh and the solution. However, these functions require the solution to be in the standard FEM V_h. That's why we need to interpolate the obtained solution in V_h^Γ to V_h.

In the `NXFEM toolbox`, one can use `interNX2STD` to perform this job. This function follows the operator I_{SD}:

$$\begin{array}{rlcl}
I_{SD} & : \quad V_h^\Gamma & \longrightarrow & V_h \\
& \qquad u & \longmapsto & \tilde{u}
\end{array},$$

where $u = \Sigma_{i \in \mathscr{I} \setminus \mathscr{I}_\Gamma} u_i \varphi_i + \Sigma_{k=1}^2 \Sigma_{i \in \mathscr{I}_\Gamma} u_i^{(k)} \varphi_i^{(k)} \in V_h^\Gamma$ and

$$\tilde{u}(\mathrm{x}_i) = \begin{cases}
u_i & \text{for } i \in \mathscr{I} \setminus \mathscr{I}_\Gamma, \\
u_i^{(1)} & \text{for } i \in \mathscr{I}_2^\Gamma \setminus \mathscr{I}_0, \\
u_i^{(2)} & \text{for } i \in \mathscr{I}_1^\Gamma \setminus \mathscr{I}_0, \\
u_i^{(1)} + u_i^{(2)} & \text{for } i \in \mathscr{I}_\Gamma \cap \mathscr{I}_0.
\end{cases}$$

Recall that, $\mathscr{I}$ is the set of numbering of nodes associated to V_h and $\mathscr{I}_1, \mathscr{I}_2, \mathscr{I}_0$ are its subset in each subdomain and on the interface respectively. We also use again the notations of $\mathscr{I}_\Gamma, \mathscr{I}_i^\Gamma$ defined as in (2.34) for the nodes around the interface. For the notation of solution u, cf. Section 2.5.

A.7.2 From V_h to V_h^Γ

In many cases, we need an interpolation from V_h to V_h^Γ so that the solution can be used as a component in the computations of NXFEM space. We achieve this thanks to function `interSTD2NX` in `NXFEM toolbox`. This function follows the operator I_{NX} defined as

$$I_{NX} \quad : \quad \begin{array}{ccc} V_h & \longrightarrow & V_h^\Gamma \\ \tilde{u} & \longmapsto & u \end{array} \quad ,$$

where $\tilde{u} = \Sigma_i \tilde{u}_i \varphi_i \in V_h, \tilde{u}_i = \tilde{u}(x_i)$ and

$$\begin{cases} u & = \Sigma_{i \in \mathscr{I} \setminus \mathscr{I}_\Gamma} u_i \varphi_i + \Sigma_{k=1}^2 \Sigma_{i \in \mathscr{I}_\Gamma} u_i^{(k)} \varphi_i^{(k)}, \\ u_i & = \tilde{u}_i \text{ for } i \in \mathscr{I} \setminus \mathscr{I}_\Gamma, \\ u_i^{(1)} & = u_i^{(2)} = \tilde{u}_i, \text{ for } i \in \mathscr{I}_\Gamma. \end{cases}$$

B. Biofilm growth

Contents

B.1 Biofilm's experimental parameters

The specific substrate consumption rate, $\bar{\mu}_S$, is given by Michealis-Menten kinetics as in (B.1),

$$\bar{\mu}_S = \rho_x(\hat{q}_0 + \gamma f_D b).\tag{B.1}$$

The net rate of active biomass production, $\bar{\mu}_x$, and net rate of EPS production, $\bar{\mu}_w$, are given by (B.2),

$$\begin{aligned}\bar{\mu}_x &= Y_{x/0}\hat{q}_0 - b,\\ \bar{\mu}_w &= (1 - f_D)b + Y_{w/0}\hat{q}_0.\end{aligned}\tag{B.2}$$

All specific values of parameters are cited in [26], for a reference of sources, please check this article. I give in Table B.1 their values only.

Name	Description	Value
ρ_x	Biomass concentration	1.0250 mg VS/mm^3
ρ_w	Inactive material concentration	1.0125 mg VS/mm^3
$Y_{x/0}$	Yield of active biomass due to substrate consumption	0.583 mg VS/mgO$_2$
$Y_{w/0}$	Yield of EPS due to substrate consumption	0.477 mg VS/mgO$_2$
$\hat{q}_0$	Maximum specific substrate utilization rate	8 mgO$_2$/mg VS day
K_0	Half-maximum-rate concentration for utilization of substrate	5×10^{-7} mgO$_2$/mm^3
b	Endogenous decay rate coefficient	0.3/day
D_S^b	Substrate diffusion coefficient in the biofilm	146.88 mm^2/day
$S_{\max}$	Substrate concentration in bulk liquid	8.3×10^{-6} mgO$_2$/mm^3
f_D	Biodegradable fraction of active biomass	0.8
γ	Chemical oxygen demand of VS	1.42 mgO$_2$/mg VS

Table B.1. Table of experimental parameters.

references

List of Figures

List of Tables

List of Algorithms

Bibliography

Articles

[1] DG Allison and P Gilbert. "Modification by surface association of antimicrobial susceptibility of bacterial populations". In: *Journal of industrial microbiology* 15.4 (1995), pages 311–317 (*cited on page 2*).

[2] Chandrasekhar Annavarapu, Martin Hautefeuille, and John E Dolbow. "A robust Nitsche's formulation for interface problems". In: *Computer Methods in Applied Mechanics and Engineering* 225 (2012), pages 44–54 (*cited on page 25*).

[3] Pedro MA Areias and Ted Belytschko. "A comment on the article "A finite element method for simulation of strong and weak discontinuities in solid mechanics" by A. Hansbo and P. Hansbo [Comput. Methods Appl. Mech. Engrg. 193 (2004) 3523–3540]". In: *Computer methods in applied mechanics and engineering* 195.9-12 (2006), pages 1275–1276 (*cited on page 11*).

[4] Helio JC Barbosa and Thomas JR Hughes. "The finite element method with Lagrange multipliers on the boundary: circumventing the Babuška-Brezzi condition". In: *Computer Methods in Applied Mechanics and Engineering* 85.1 (1991), pages 109–128 (*cited on page 25*).

[5] Nelly Barrau et al. "A robust variant of NXFEM for the interface problem". In: *Comptes Rendus Mathematique* 350.15-16 (Aug. 2012), pages 789–792 (*cited on pages 20, 23, 25, 32*).

[6] John W Barrett and Charles M Elliott. "Fitted and unfitted finite-element methods for elliptic equations with smooth interfaces". In: *IMA journal of numerical analysis* 7.3 (1987), pages 283–300 (*cited on page 10*).

[7]   Ted Belytschko and Tom Black. "Elastic crack growth in finite elements with minimal remeshing". In: *International journal for numerical methods in engineering* 45.5 (1999), pages 601–620 (*cited on page 10*).

[8]   JH Bramble and JT King. "A finite element method for interface problems in domains with smooth boundaries and interfaces". In: *Advances in Computational Mathematics* 05 (1996), page 33 (*cited on page 10*).

[9]   Erik Burman. "Ghost penalty". In: *Comptes Rendus Mathematique* 348.21-22 (2010), pages 1217–1220 (*cited on pages 11, 12, 25, 27*).

[10] Erik Burman and Peter Hansbo. "Fictitious domain finite element methods using cut elements: I. A stabilized Lagrange multiplier method". In: *Computer Methods in Applied Mechanics and Engineering* 199.41-44 (2010), pages 2680–2686 (*cited on pages 11, 26*).

[11] Erik Burman and Peter Hansbo. "Fictitious domain finite element methods using cut elements: II. A stabilized Nitsche method". In: *Applied Numerical Mathematics* 62.4 (2012), pages 328–341 (*cited on pages 11, 12, 26, 27*).

[12] Erik Burman and Paolo Zunino. "Numerical approximation of large contrast problems with the unfitted Nitsche method". In: (2011), pages 227–282 (*cited on pages 11, 27*).

[13] Erik Burman et al. "CutFEM: Discretizing geometry and partial differential equations". In: *International Journal for Numerical Methods in Engineering* 104.7 (2015), pages 472–501 (*cited on pages 12, 25, 26, 28*).

[14] Zhiming Chen and Jun Zou. "Finite element methods and their convergence for elliptic and parabolic interface problems". In: *Numerische Mathematik* (1998), pages 1–23 (*cited on pages 10, 60*).

[15] David L Chopp et al. "A mathematical model of quorum sensing in a growing bacterial biofilm.". In: *Journal of industrial microbiology & biotechnology* 29.6 (Dec. 2002), pages 339–46 (*cited on pages 7, 92*).

[16] David L Chopp et al. "The dependence of quorum sensing on the depth of a growing biofilm.". In: *Bulletin of mathematical biology* 65.6 (Nov. 2003), pages 1053–79 (*cited on pages 7, 92*).

[17] DavidL. Chopp. "Simulating Bacterial Biofilms". In: Topics in Biomedical Engineering. International Book Series (2007), pages 1–31 (*cited on page 7*).

[18] Susanne Claus, Erik Burman, and Andre Massing. "CutFEM: a stabilised Nitsche XFEM method for multi-physics problems". In: (2015), pages 171–174 (*cited on pages 12, 26*).

[19] N G Cogan. "Effects of persister formation on bacterial response to dosing.". In: *Journal of theoretical biology* 238.3 (Feb. 2006), pages 694–703 (*cited on page 7*).

[20] N G Cogan, Ricardo Cortez, and Lisa Fauci. "Modeling physiological resistance in bacterial biofilms.". In: *Bulletin of mathematical biology* 67.4 (July 2005), pages 831–53 (*cited on pages 7, 8*).

[21] NG Cogan. "Incorporating toxin hypothesis into a mathematical model of persister formation and dynamics". In: *Journal of theoretical biology* 248.2 (2007), pages 340–349 (*cited on page 7*).

[22] NG Cogan. "Two-fluid model of biofilm disinfection". In: *Bulletin of mathematical biology* 70.3 (2008), pages 800–819 (*cited on page 7*).

[23] Patricio Cumsille, Juan A Asenjo, and Carlos Conca. "A novel model for biofilm growth and its resolution by using the hybrid immersed interface-level set method". In: *Computers & Mathematics with Applications* 67.1 (2014), pages 34–51 (*cited on page 8*).

[24] Charles Dapogny and Pascal Frey. "Computation of the signed distance function to a discrete contour on adapted triangulation". In: *Calcolo* 49.3 (2012), pages 193–219 (*cited on pages 83, 87*).

[25] Daniele A. Di Pietro and Alexandre Ern. "Discrete functional analysis tools for Discontinuous Galerkin methods with application to the incompressible Navier–Stokes equations". In: *Mathematics of Computation* 79.271 (2010), pages 1303–1330 (*cited on pages 12, 55, 67*).

[26] Ravindra Duddu, David L Chopp, and Brian Moran. "A two-dimensional continuum model of biofilm growth incorporating fluid flow and shear stress based detachment.". In: *Biotechnology and bioengineering* 103.1 (May 2008), pages 92–104 (*cited on pages 7, 92, 121*).

[27] Ravindra Duddu et al. "A combined extended finite element and level set method for biofilm growth". In: *International journal for numerical method in engineering int.* 2.847 (2006), pages 1–33 (*cited on pages 10, 90*).

[28] R. Eymard, T. Gallouët, and R. Herbin. "Discretization of heterogeneous and anisotropic diffusion problems on general nonconforming meshes". In: *IMA Journal of Numerical Analysis* 30.4 (2010), pages 1009–1043 (*cited on pages 67, 68*).

[29] Robert Eymard, Thierry Gallouët, and Raphaele Herbin. "Discretization schemes for heterogeneous and anisotropic diffusion problems on general nonconforming meshes". In: *Available online as HAL report* 203269 (2008) (*cited on pages 12, 56*).

[30] Jouni Freund and Rolf Stenberg. "On weakly imposed boundary conditions for second order problems". In: (1995), pages 327–336 (*cited on page 16*).

[31] Anita Hansbo and Peter Hansbo. "An unfitted finite element method, based on Nitsche's method, for elliptic interface problems". In: *Computer methods in applied mechanics and engineering* 191 (2002), pages 5537–5552 (*cited on pages xv, 10, 12, 15, 17, 20, 23, 24, 28, 29, 62, 63, 66*).

[32] Jaroslav Haslinger and Yves Renard. "A new fictitious domain approach inspired by the extended finite element method". In: *SIAM Journal on Numerical Analysis* 47.2 (2009), pages 1474–1499 (*cited on page 25*).

[33] F. Hecht. "New development in FreeFem++". In: *J. Numer. Math.* 20.3-4 (2012), pages 251–265 (*cited on page 85*).

[34] Niklas Johansson. "Implementation of a standard level set method for incompressible two-phase flow simulations". In: (2011) (*cited on pages 83, 87*).

[35] János Karátson and Sergey Korotov. "Discrete maximum principles for FEM solutions of some nonlinear elliptic interface problems". In: *International Journal of Numerical Analysis and Modeling* 6.1 (2009), pages 1–16 (*cited on page 60*).

[36] Christoph Lehrenfeld and Arnold Reusken. "Nitsche-XFEM with Streamline Diffusion Stabilization for a Two-Phase Mass Transport Problem". In: *SIAM*

Journal on Scientific Computing 34.5 (Jan. 2012), A2740–A2759 (*cited on page 11*).

[37] Christoph Lehrenfeld and Arnold Reusken. "Optimal preconditioners for Nitsche-XFEM discretizations of interface problems". In: *Numerische Mathematik* 135.2 (2017), pages 313–332 (*cited on page 11*).

[38] Randall J Leveque and Zhilin Li. "The immersed interface method for elliptic equations with discontinuous coefficients and singular sources". In: *SIAM Journal on Numerical Analysis* 31.4 (1994), pages 1019–1044 (*cited on page 10*).

[39] Zhilin Li. "An overview of the immersed interface method and its applications". In: *Taiwanese journal of mathematics* 7.1 (2003), pages 1–49 (*cited on page 10*).

[40] Nicolas Moës, John Dolbow, and Ted Belytschko. "A finite element method for crack growth without remeshing". In: *International journal for numerical methods in engineering* 46.1 (1999), pages 131–150 (*cited on page 10*).

[41] Jacques Monod. "The Growth of Bacterial Cultures". In: *Annual Review of Microbiology* 3.1 (1949), pages 371–394 (*cited on page 4*).

[42] Ricardo Murga, Philip S Stewart, and Don Daly. "Quantitative analysis of biofilm thickness variability". In: *Biotechnology and bioengineering* 45.6 (1995), pages 503–510 (*cited on page 4*).

[43] J. Nitsche. "Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind". In: *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg* 36.1 (1971), pages 9–15 (*cited on page 11*).

[44] George O'Toole, Heidi B. Kaplan, and Roberto Kolter. "Biofilm formation as microbial development". In: *Annual Review of Microbiology* 54 (2000), pages 49–79 (*cited on page 2*).

[45] Stanley Osher and James A Sethian. "Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations". In: *Journal of Computational Physics* 79.1 (Nov. 1988), pages 12–49 (*cited on pages 11, 77*).

[46] Cristian Picioreanu, Van Loosdrecht, and J J M Cheijnen. "Discrete-differential modelling of biofilm structure". In: *Water Science and Technology* 39.7 (1999), pages 115–122 (*cited on page 6*).

[47] Cristian Picioreanu, M C Van Loosdrecht, and J J Heijnen. "Effect of diffusive and convective substrate transport on biofilm structure formation: a two-dimensional modeling study.". In: *Biotechnology and bioengineering* 69.5 (Sept. 2000), pages 504–15 (*cited on pages 6, 10*).

[48] Cristian Picioreanu, Mark CM Van Loosdrecht, Joseph J Heijnen, et al. "Two-dimensional model of biofilm detachment caused by internal stress from liquid flow". In: *Biotechnology & Bioengineering* 72.2 (2001), pages 205–218 (*cited on page 10*).

[49] Carol Potera. "Forging a link between biofilms and disease". In: (1999) (*cited on page 2*).

[50]   Arnold Reusken. "Analysis of an extended pressure finite element space for two-phase incompressible flows". In: *Computing and Visualization in Science* 11.4-6 (Apr. 2008), pages 293–305 (*cited on pages xii, 11, 28–30*).

[51]   Arnold Reusken and Trung Hieu Nguyen. " Nitsche's method for a transport problem in two phase incompressible flow ". In: *Journal of Fourier Analysis and Applications* 15.5 (Aug. 2009), pages 663–683 (*cited on pages 11, 20, 23*).

[52]   B E Rittman. "The Effect of Shear Stress on Bioflm Loss Rate". In: *Biotechnology and bioengineering* 24.2 (1982), pages 501–506 (*cited on page 7*).

[53]   James A Sethian. "A fast marching level set method for monotonically advancing fronts". In: *Proceedings of the National Academy of Sciences* 93.4 (1996), pages 1591–1595 (*cited on page 80*).

[54]   Rajen Kumar Sinha and Bhupen Deka. "An unfitted finite-element method for elliptic and parabolic interface problems". eng. In: *IMA journal of numerical analysis* 27.3 (2005), pages 529–549 (*cited on page 33*).

[55]   Bryan G Smith, Benjamin L Vaughan Jr, and David L Chopp. "The extended finite element method for boundary layer problems in biofilm grown". In: *Communications in Applied Mathematics and Computational Science* 2.1 (2007), pages 35–56 (*cited on page 96*).

[56]   Mark Sussman, Peter Smereka, and Stanley Osher. "A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow". In: 114.1 (1994), pages 146–159. eprint: 1994 (*cited on page 80*).

[57]   Xianlong Zhang, Xiaoling Wang, and Qingping Sun. "Modeling of Biofilm Growth on Ager Substrate Using the Extended Finite Element Method". In: *Procedia IUTAM* 23 (2017), pages 33–41 (*cited on page 8*).

[58]   Paolo Zunino, Laura Cattaneo, and Claudia Maria Colciago. "An unfitted interface penalty method for the numerical approximation of contrast problems". In: *Applied Numerical Mathematics* 61.10 (2011), pages 1059–1076 (*cited on pages 11, 30*).

Books

[59]   Mark Ainsworth and J.Tinsley T Oden. *A posteriori error estimation in finite element analysis*. Edited by Peter Lax Myron B. Allen III, David A. Cox. John Wiley and Sons, 1997 (*cited on page 66*).

[60]   Ted Belytschko et al. *Nonlinear finite elements for continua and structures*. John wiley & sons, 2013 (*cited on page 93*).

[61]   Haim Brezis. *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Springer, 2010 (*cited on page 64*).

[62]   Alfred B. Cunningham, John E. Lennox, and Rockford J. Ross. *Biofilms: The hypertextbook*. 2010 (*cited on pages 2, 3*).

[63]   Daniele Antonio Di Pietro and Alexandre Ern. *Mathematical aspects of discontinuous Galerkin methods*. Volume 69. Springer Science & Business Media, 2011 (*cited on page 21*).

[64]   S. Dürr and J.C. Thomason. *Biofouling*. Wiley, 2009 (*cited on page 3*).

[65] Alexandre Ern and Jean-Luc Guermond. *Theory and practice of finite elements*. Volume 159. Springer Science & Business Media, 2013 (*cited on pages 38, 107, 108*).

[66] S Gross and Arnold Reusken. *Numerical methods for two-phase incompressible flows*. Volume 40. Springer Series in Computational Mathematics. 2011, page 482 (*cited on pages 11, 30, 80, 81, 85*).

[67] Stanley Osher and Ronald Fedkiw. *Level set methods and dynamic implicit surfaces*. Volume 153. Springer Science & Business Media, 2006 (*cited on pages 78, 79*).

[68] Cristian Picioreanu, M van Loosdrecht, and J Heijnen. *Multidimensional modeling of biofilm structure*. Delft University of Technology, Faculty of Applied Sciences, 1999 (*cited on page 6*).

[69] Alfio Quarteroni and Alberto Valli. *Numerical Approximation of Partial Differential Equations*. Springer, 1994 (*cited on page 80*).

[70] Hans-Görg Roos, Martin Stynes, and Lutz Tobiska. *Robust numerical methods for singularly perturbed differential equations: convection-diffusion-reaction and flow problems*. Volume 24. Springer Science & Business Media, 2008 (*cited on page 80*).

[71] O. Wanner. *Mathematical Modeling of Biofilms*. London: IWA Publishing, 2006 (*cited on pages 2–4*).

Thesis

[72] Eva Loch. "The level set method for capturing interfaces with applications in two-phase flow problems". 2013, page 174 (*cited on page 87*).

[73] Trung Hieu Nguyen and Arnold Reusken. "Numerical methods for mass transport equations in two-phase incompressible flows". Lehrstuhl für Numerische Mathematik, 2009 (*cited on page 11*).

[74] Barbara Szomolay et al. "Analysis and control of a biofilm disinfection model". Montana State University-Bozeman, College of Letters & Science, 2006 (*cited on page 6*).

Index

Titre: **Méthodes d'éléments finis pour des systèmes d'EDP non linéaires avec interface. Application à un modèle de croissance de biofilm**.

Mots clés: *NXFEM, Nitsche-Extended Finite Element Method, problème d'interface, méthode de lignes de niveau, biofilm.*

Résumé: Un biofilm est un ensemble de micro-organismes tels que les bactéries, les champignons ou encore les algues qui vivent en communauté. Les biofilms ont la capacité d'être présents en tout lieu. Ils sont observés dans les milieux aqueux ou humides. Ils peuvent se développer sur n'importe quel type de surface naturelle ou artificielle, qu'elle soit minérale (roche, interfaces air-liquide...) ou organique (peau, tube digestif, racines et feuilles des plantes), industrielle (canalisations, coques des navires) ou médicale comme les prothèses et les cathéters. Cette ubiquité est à l'origine de nombreuses infections bactériennes. Les infections nosocomiales contractées dans les hôpitaux sont un exemple majeur. Certaines de ces infections pouvant être mortelles. Le traitement médical des biofilms est souvent inefficace pour lutter contre ce type d'infection. Il est donc important de comprendre les mécanismes de croissance d'un biofilm. Telle est la motivation de la présente thèse.

Afin de réaliser des simulations numériques d'un modèle décrivant la croissance d'un biofilm, nous combinons différentes méthodes de calcul basées sur la méthode Nitsche-Extended Finite Element Method (NXFEM) ainsi que sur la méthode des lignes de niveau. Ces méthodes nous permettent d'étudier des modèles complexes dans lesquels l'interface entre le biofilm et son environnement est capable de se déformer tout en dépendant du temps. Ceci permet de considérer une discrétisation à l'aide d'un maillage ne coïncidant pas avec l'interface biofilm/environnement. Nous présentons également une technique de découplage d'un système d'équations aux dérivées partielles semi-linéaires et la façon dont nous appliquons la méthode NXFEM pour résoudre un tel problème. Ce système est en relation avec le modèle de croissance du biofilm qui est traité dans cette thèse.

Pour l'implémentation, une boîte à outils NXFEM, développée en Matlab, a été entièrement conçue pour résoudre un tel problème. Nous donnons dans ce document les détails des algorithmes et techniques numériques utilisés afin que chacun puisse utiliser cette boîte à outils pour ses propres projets.

Title: **Finite Element Methods for nonlinear interface problems. Application to a biofilm growth model**.

Keywords: *NXFEM, Nitsche-Extended Finite Element Method, interface problem, level set method, biofilm, unfitted mesh.*

Abstract: A biofilm is a collective of living, reproducing microorganisms, such as bacteria, that stick together as a colony, or community. They appear everywhere in human life and have some impacts on our environment. Biofilm modeling, together with laboratory experiments, has risen as a means of producing quantitative tools for scientists to better understand the biofilm's growth. This thesis is motivated to research on this subject.

A combination of computational methods which are based on *Nitsche-Extended Finite Element Method* (NXFEM), *Level Set Method* and some other stabilized techniques are used to solve and simulate a biofilm growth model. These methods allow us to work with a complex scheme in which the interface between the biofilm and its environment is allowed to change with time and on an unfitted mesh. We also present a technique of decoupling a system of semilinear differential equations and how we apply the NXFEM method to solve such a problem. This system has a relation to a model of biofilm's growth which will be examined carefully in the work.

For the implementations, *NXFEM toolbox* which is a Matlab based toolbox is built for solving such a problem. We give also the details of all algorithms and numerical techniques so that everyone can use this toolbox for their own projects.