Master thesis

Super-time stepping for parabolic PDEs with rough coefficients

Thomas Jacumin

November 22, 2018

Supervisor :	Prof. Dr. Ionut Danaila	University of Rouen
	Prof. Dr. Daniel Peterseim	University of Augsburg
	Dr. Robert Altmann	University of Augsburg
Co-supervisor :	Roland Maier	University of Augsburg

Master of Mathematics Analysis and Modelling

Abstract

We compare the effectiveness of the classical finite element method and of the localized orthogonal decomposition method, introduced in [6], for linear parabolic equation with highly varying diffusion coefficient in space. The finite element method suffers from the pre-asymptotic effect and as a result needs a fine mesh discretization, unlike the localized orthogonal decomposition method. Then, we use the forward Euler method for the temporal discretization. Due to the restrictive stability condition of this method, we introduce the Super-time stepping acceleration, described in [1] for the temporal discretization, which have a relaxed stability condition and which is way faster than the forward Euler method. The convergence rate depends only on the contrast but not on the variations of the diffusion coefficient. Finally, we present numerical experiments, which confirm our theoretical findings, and we compare each method speed.

Acknowledgments

First of all, I would like to express my most sincere gratitude to my supervisors Professor Doctor Daniel Peterseim and Doctor Robert Altmann at the University of Augsburg, Germany, as well as Professor Doctor Ionut Danaila at the University of Rouen, France. I would like to thanks them for providing me the topic that is tightly integrated with their research.

Then, I would like to mention that my master thesis could not be accomplished without the meticulous guidance of Roland Maier. He has largely been responsible for patiently correcting my misconceptions and mistakes.

Finally, my family for their endless support and encouragement. Moreover, I would like to thanks my mother and my cousin, Benoit, for helping me correct language mistakes.

Contents

List of Figures

List of Tables

Introduction

In this thesis we study numerical solution of the linear parabolic equation with a highly varying diffusion coefficient. This equation happens when modeling diffusion problems in micro-heterogeneous media. For example when modelling the thermal conduction for heterogeneous poroelasticity or for thermoelasticity. Such problems are often referred as multiscale problems.

An optimal convergence rate of order one with respect to the mesh size h can be led by using the classical finite elements method. However, this method suffers from the pre-asymptotic effect. This effect appears when the mesh size is greater than ϵ if the diffusion coefficient varies on a scale of ϵ and it implies that the numerical solution will not correspond to the exact solution of the problem.

That is why we use the localized orthogonal decomposition method, introduced in [6], which is a generalized finite element method. This method convergence rate is of order one with respect to the coarse mesh size H and does not have the pre-asymptotic effect.

Concerning the discretization of the temporal domain, we use the forward Euler method. This finite difference method is not always stable. The condition for stability is known as the Courant–Friedrichs– Lewy condition. In order to be stable it must have $\Delta t \lesssim h^2$, where Δt is the time step and h the mesh size. Under this condition, the convergence rate is of order one with respect to the time step Δt. Afterward, we apply the Super-time stepping technique introduced in [1]. This method requires a relaxed stability condition and have a convergence rate of order one with respect to the super-time step ΔT. Moreover, by using this method, we are able to balance the error. Lastly, this method is faster than the Euler one.

Outline of the thesis

Next is a brief outline of contents for each chapter.

- Chapter 1 leads into notations and some useful results.
- Chapter 2 introduces the parabolic problem that we study and the formulation of its variational form.
- Chapter 3 discusses the two discretizations used for the domain of study and time, as well as which properties these discretizations must have.
- Chapter 4 goes into the finite element method, and shows results about it. In this chapter we also study the two different methods for time approximation along with their respective stability and convergence. To finish, it presents an experimental error analysis of the numerical methods.

- Chapter 5 examines the localized orthogonal decomposition method, and shows results about it. After that, like in Chapter 4, it studies the same two methods for time approximation. To finish chapter 5 presents an experimental error analysis of the numerical methods.
- Chapter 6 summarizes the most important results in this thesis, and mentions some possible applications as well as some possible improvement.

Chapter 1

Introduction

1.1 Recall and notations

For the convenience of the reader, notations used in this paper can be found below. In this chapter, U denote a Lipschitz open subset of $\mathbb{R}^n$, $n \in \mathbb{N}^*$.

- $\mathcal{M}_{\text{sym}}(U, \alpha, \beta)$ denote the space of tensor $M \in L^\infty(U, \mathbb{R}^{d\times d})$ such that M is symmetric,

$$0 < \alpha = \text{ess inf}_{x\in U} \inf_{v\in\mathbb{R}^d\setminus\{0\}} \frac{A(x)v\cdot v}{v\cdot v},$$

$$+\infty > \beta = \text{ess sup}_{x\in U} \sup_{v\in\mathbb{R}^d\setminus\{0\}} \frac{A(x)v\cdot v}{v\cdot v}.$$

- $H^1(U)$ denote the classical Sobolev space with norm

$$\|v\|^2_{H^1(U)} = \|v\|^2_{L^2(U)} + \|\nabla v\|^2_{L^2(U)}.$$

- $H^1_0(U)$ denote the space of functions in $H^1(U)$ that vanishes on ∂U. We will write $\|\cdot\|_{H^1_0(U)}$ instead of $\|\cdot\|_{H^1(U)}$.

- $H^{-1}(U) = \big(H^1_0(U)\big)^*$ denote the dual space to $H^1_0(U)$ with norm

$$\|f\|_{H^{-1}(U)} = \sup\big\{ < f, u > \ | \ u \in H^1_0(U) \text{ and } \|u\|_{H^1_0(U)} \le 1\big\}.$$

- $L^p(0,T;X)$, for $(X, \|\cdot\|_X)$ a Banach space, denote the Bochner space with norm

$$\|v\|_{L^p(0,T;X)} = \begin{cases} \Big(\int_0^T \|v\|^p_X dt\Big)^{1/p}, & 1 \le p < +\infty, \\ \text{ess sup}_{0\le t\le T} \|v\|_X, & p = +\infty. \end{cases}$$

- We define the bilinear form

$$a : H^1_0(\Omega)\times H^1_0(\Omega) \to \mathbb{R}, \qquad (u,v) \mapsto \int_\Omega (A\nabla u)\cdot\nabla v \ dx.$$

- We denote the L^2-scalar product between u and v in $L^2(U)$, i.e. $(\int_U uv \ dx)^{1/2}$, by (u,v) and the L^2-norm $\|\cdot\|_{L^2(\Omega)}$ by $\|\cdot\|$.

- Moreover, for any u in $H^1_0(U)$, we define $\|u\|_a := \sqrt{a(u,u)}$.

1.2 Useful results

We start with some unrelated theorems to help us prove results in subsequent chapters.

Theorem 1.2.1. *On the space $H_0^1(U)$, the norm and the seminorm, defined as*

$$\|v\|^2_{H_0^1(U)} = \|v\|^2_{L^2(U)} + \|\nabla v\|^2_{L^2(U)},$$
$$|v|^2_{H_0^1(U)} = \|\nabla v\|^2_{L^2(U)},$$

are equivalent.

Proof. Let v be in $H_0^1(U)$.

$$|v|^2_{H_0^1(U)} = \|\nabla v\|^2_{L^2(U)}$$

Since $\|v\|^2_{L^2(U)}$ is a positive term, we can add it to the left hand side

$$|v|^2_{H_0^1(U)} \leq \|v\|^2_{L^2(U)} + \|\nabla v\|^2_{L^2(U)}.$$

Then, $|v|_{H_0^1(U)} \leq \|v\|_{H_0^1(U)}$. To prove the other inequality, we use Poincaré

$$\begin{aligned}\|v\|^2_{H_0^1(U)} &= \|v\|^2_{L^2(U)} + \|\nabla v\|^2_{L^2(U)} \\ &\leq C_P^2\|\nabla v\|^2_{L^2(U)} + \|\nabla v\|^2_{L^2(U)}.\end{aligned}$$

Thus, $\|v\|_{H_0^1(U)} \leq C|v|_{H_0^1(U)}$. This conclude the proof. □

Theorem 1.2.2 (Gronwall's inequality (differential form))**.** *Let η be a nonnegative, absolutely continuous function on $[0,T]$, which satisfies for a.e. t the differential inequality*

$$\eta'(t) \leq \phi(t)\eta(t) + \psi(t),$$

where ϕ and ψ are nonnegative, summable functions on $[0,T]$. Then

$$\eta(t) \leq e^{\int_0^t \phi(s)\ ds}\Big(\eta(0) + \int_0^t \psi(s)\ ds\Big)$$

for all $0 \leq t \leq T$.

Proof. This proof follows the one in [3], Section B.2.

$$\begin{aligned}\tfrac{d}{ds}\Big(\eta(s)e^{-\int_0^s \phi(r)\ dr}\Big) &= \eta'(s)e^{-\int_0^s \phi(r)\ dr} - \eta(s)\phi(s)e^{-\int_0^s \phi(r)\ dr} \\ &= e^{-\int_0^s \phi(r)\ dr}(\eta'(s) - \phi(s)\eta(s)) \\ &\leq e^{-\int_0^s \phi(r)\ dr}(\phi(s)\eta(s) + \psi(s) - \phi(s)\eta(s)) \\ &= \psi(s)e^{-\int_0^s \phi(r)\ dr},\end{aligned}$$

for a.e. $0 \le s \le T$. Then for all $0 \le t \le T$,

$$\begin{aligned} \eta(t)e^{-\int_0^t \phi(r)\ dr} - \eta(0) &\le \int_0^t \psi(s) \underbrace{e^{-\int_0^s \phi(r)\ dr}}_{\le 1}\ ds \\ &\le \int_0^t \psi(s)\ ds \end{aligned}$$

Thus

$$\eta(t) \le e^{\int_0^t \phi(s)\ ds}\Big(\eta(0) + \int_0^t \psi(s)\ ds\Big).$$

□

Chapter 2

Problem formulation

This chapter introduces the problem studied in this paper. Then, it formulates the variational form of this problem. To finish, it determines in which framework it has a solution.

2.1 Problem

We consider an open bounded polygonal/polyhedral domain $\Omega \subset \mathbb{R}^d$ for $d \leq 3$. Let $A : \Omega \to \mathbb{R}^{d\times d}$, $f : \Omega \times [0,T] \to \mathbb{R}$ and $u_0 : \Omega \to \mathbb{R}$. We consider A as a tensor in the theoretical part, but A will be scalar-valued for the experiments. This two representations of the diffusion coefficient are equivalent. We study the following problem

Problem 2.1.1. *Find $u(x,t)$ and such that*

$$\begin{cases} \frac{\partial u}{\partial t} - \operatorname{div}(A\nabla u) = f, & \text{in } \Omega \times (0,T], \\ u = 0, & \text{on } \partial\Omega \times (0,T], \\ u(\cdot,0) = u_0, & \text{in } \Omega, \end{cases} \tag{2.1}$$

where $T > 0$.

The first equation $\frac{\partial u}{\partial t} - \operatorname{div}(A\nabla u) = f, \text{in } \Omega \times (0,T]$ corresponds to the heat equation. The second one is the Dirichlet boundary condition. And the last one is the initial condition. Together, they are called pure initial value problem (or Cauchy problem) for the heat equation or in more simple terms a parabolic equation. This setting corresponds to the classic framework, or strong formulation, of the parabolic equation. In one dimension and if f is equal to zero, it is possible the compute the exact solution of this problem by using the separation of variables. However in higher dimension or if f is non-zero it is really difficult to find it. That is why we resort to numerical approximation.

2.2 Weak formulation

Now, we want to find a solution of the parabolic equation (2.1) in a bigger space than the one in (2.1). Indeed, we wish to study the problem in which classical derivatives may not exist. In order to do so, we work in a weaker framework. We want to formulate the variational form also called weak formulation of our problem (2.1). If this weak formulation admits a solution and

under some regularity conditions, it is called the weak solution. The variational formulation is the basis for the finite element method and the localized orthogonal decomposition method, which we will study in the next two chapters. Moreover, if the classical framework is regular enough, then this weak solution will correspond to the classical solution. Let $v \in H_0^1(\Omega)$. We start from the heat equation

$$\frac{\partial u}{\partial t} - \operatorname{div}(A\nabla u) = f.$$

Then we multiply both side by v and we integrate over Ω

$$\int_\Omega \frac{\partial u}{\partial t} v \, dx - \int_\Omega \operatorname{div}(A\nabla u) v \, dx = \int_\Omega f v \, dx.$$

We use the Green's formula

$$\int_\Omega \frac{\partial u}{\partial t} v \, dx + \int_\Omega (A\nabla u) \cdot \nabla v \, dx - \int_{\partial\Omega} v(A\nabla u) \cdot n \, ds = \int_\Omega f v \, dx.$$

As v belongs to $H_0^1(\Omega)$, v is equal to zero, in trace sens, on the border of Ω. Thus, we have $\int_{\partial\Omega} v(A\nabla u) \cdot n \, ds = 0$. Finally, by using the notations introduced in Chapter 1, we have the following variational problem

Problem 2.2.1. *For all $t \in (0, T]$, find $u(\cdot, t) \in H_0^1(\Omega)$ such that*

$$\begin{cases} \left(\frac{\partial u}{\partial t}, v\right) + a(u, v) = (f, v), & \forall v \in H_0^1(\Omega), \\ u(\cdot, 0) = u_0 \quad . \end{cases} \tag{2.2}$$

This last problem is the weak formulation of the parabolic problem (2.1).

Definition 2.2.1 (Weak solution)**.** *Let f be in $L^2(0, T; L^2(\Omega))$ and u_0 be in $L^2(\Omega)$. Let A be in $\mathcal{M}_{sym}(U, \alpha, \beta)$, $0 < \alpha \leq \beta$. We say that u is a weak solution to the parabolic problem (2.1) if $u \in L^2(0, T; H_0^1(\Omega))$, $\frac{\partial u}{\partial t} \in L^2(0, T; H^{-1}(\Omega))$ and u satisfies the weak formulation (2.2).*

We proved that the strong formulation implies the weak one. Now we want to demonstrate that the other way around is also true.

Theorem 2.2.1. *Suppose $u(\cdot, t) \in H_0^1(\Omega)$ such that $\frac{\partial u}{\partial t}(\cdot, t) \in C^0(\bar{\Omega})$ and $u(\cdot, t) \in C^2(\Omega)$. If u is a solution of (2.2), then u is a solution of (2.1).*

Proof. We suppose $u(\cdot, t) \in H_0^1(\Omega)$ to be a solution of (2.2). Then for all v in $H_0^1(\Omega)$, we have

$$\left(\frac{\partial u}{\partial t}, v\right) + a(u, v) = (f, v).$$

By using again the Green's formula and the fact that v vanishes on $\partial\Omega$, we have

$$\left(\frac{\partial u}{\partial t} - \operatorname{div}(A\nabla u) - f, v\right) = 0.$$

As the set of bump functions i.e. $C_c^\infty(\Omega)$, is a subset of $H_0^1(\Omega)$, and if we suppose that $\frac{\partial u}{\partial t}(\cdot, t) \in C^0(\bar{\Omega})$ and $u(\cdot, t) \in C^2(\Omega)$, we can apply the fundamental lemma of calculus of variations and conclude that

$$\frac{\partial u}{\partial t} - \operatorname{div}(A\nabla u) - f = 0.$$

□

Under these conditions, the classical problem and the weak formulation are equivalent, in other words, if u is the solution of (2.1) then u is also the solution of (2.2) and vice-versa.

Theorem 2.2.2 (Stability)**.** *Let u be the solution of (2.2). For all $t \in (0, T]$, we have the following results*

$$\|u(t)\| \leq C\Big(\|u_0\| + \int_0^t \|f(s)\|\, ds\Big), \tag{2.3}$$

$$\left\|\frac{\partial u}{\partial t}(t)\right\| \leq C\Big(\|u_0\| + \|f(t)\| + \int_0^t \|f(s)\|\, ds\Big), \tag{2.4}$$

$$\|u(t)\|_a \leq C'\Big(\|u_0\| + \|f(t)\| + \int_0^t \|f(s)\|\, ds\Big). \tag{2.5}$$

Proof. For a more general case, see [3]. We start from the first equation of (2.2) and we put u instead v

$$\left(\frac{\partial u}{\partial t}, u\right) + a(u, u) = (f, u).$$

We have

$$|(f, u)| \leq \frac{1}{2}\|f\|^2 + \frac{1}{2}\|u\|^2,$$

by using Cauchy-Schwarz inequality and the fact that $\frac{1}{2}(a^2 + b^2) \geq ab$, and

$$\left(\frac{\partial u}{\partial t}, u\right) = \frac{\partial}{\partial t}\Big(\frac{1}{2}\|u\|^2\Big).$$

Then

$$\frac{\partial}{\partial t}\Big(\frac{1}{2}\|u\|^2\Big) + a(u, u) \leq \frac{1}{2}\|f\|^2 + \frac{1}{2}\|u\|^2. \tag{2.6}$$

Furthermore, we have that a is coercive

$$a(u, u) \geq C(\alpha)\|u\|^2_{H_0^1(\Omega)},\ C(\alpha) > 0.$$

Then we have

$$\begin{aligned}\frac{\partial}{\partial t}\Big(\|u\|^2\Big) + 2C(\alpha)\|u\|^2_{H_0^1(\Omega)} &\leq \frac{\partial}{\partial t}\Big(\|u\|^2\Big) + 2a(u,u)\\ &= 2(\frac{\partial u}{\partial t}, u) + 2a(u,u)\\ &= 2(f,u)\\ &\leq \|f\|^2 + \|u\|^2.\end{aligned}$$

Thus, because $2C(\alpha)\|u\|^2_{H_0^1(\Omega)} \geq 0$,

$$\frac{\partial}{\partial t}\Big(\|u\|^2\Big) \leq \|f\|^2 + \|u\|^2. \tag{2.7}$$

The Gronwall's inequality implies

$$\|u(t)\|^2 \leq e^t\Big(\|u_0\|^2 + \int_0^t \|f(s)\|^2\ ds\Big)$$

for all $0 \leq t \leq T$. We have the result (2.3). By combining (2.7) and (2.3) we obtain (2.4). Finally, let us come back to (2.6) and since $\frac{\partial}{\partial t}\Big(\frac{1}{2}\|u\|^2\Big) = \frac{1}{2}\|\frac{\partial u}{\partial t}\|^2 \geq 0$, we have

$$a(u,u) \leq \frac{1}{2}\|f\|^2 + \frac{1}{2}\|u\|^2.$$

Again, by combining the inequality above and (2.3), we get (2.5). □

Chapter 3

Discretization in space and in time

Here, we discretize our space of solution. First, we introduce the notion of mesh for our domain Ω. Then, we define the notations we use for the time discretization.

3.1 Finite element meshes

Before discretizing the space, we define the notion of mesh and what kind of properties they should have. The following definition is from [10].

Definition 3.1.1 (Regular simplicial mesh)**.** *A finite subdivision*

$$\tau := \{T_j \mid 1 \leq j \leq N_\tau\}$$

of $\Omega \subset \mathbb{R}^d$ into closed non-empty simplices (denoted elements), i.e.

$$\bar{\Omega} = \bigcup_{j=1}^{N_\tau} T_j$$

is said to be regular if any two elements $T_1, T_2 \in \tau$ are either disjoint or share exactly on vertex or one edge ($d \geq 2$) or one face ($d = 3$).

Notations 1. We denote the size of an element K of a regular simplicial mesh by $h_K := \operatorname{diam}(K)$.

With this notation, it is natural to define a regular simplicial mesh with a specific size for the elements.

Definition 3.1.2 (Regular simplicial mesh of size h)**.** *Let τ be a regular simplicial mesh. It is said of size $h > 0$ if*

$$\max_{K \in \tau} h_K = h.$$

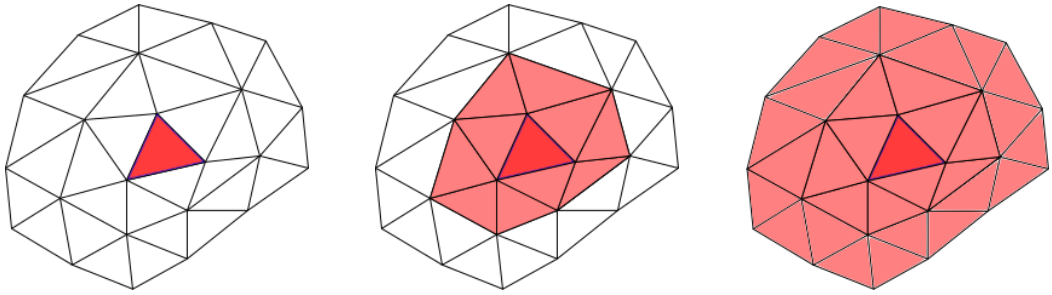
Then we write τ_h.

Now we will define the patches of an element.

Definition 3.1.3 (Patches)**.** *Let τ be a regular simplicial mesh. For all $T \in \tau$ we define $\omega_k(\tau, T)$ to be the patch of size k, where*

$$\begin{cases} \omega_0(\tau, T) := T, \\ \omega_k(\tau, T) := \bigcup \{K \in \tau \mid \bar{K} \cap \omega_{k-1} \neq \emptyset\}, \ k \in \mathbb{N}. \end{cases}$$

Here is a picture (Figure 3.1) to illustrate the definition above for $k = \{0, 1, 2\}$. We fill with red the element T of the definition. The neighbor of this element T are in pink. Together their formed the set $\omega_k(\tau, T)$.



(a) Patches of T for $k = 0$. (b) Patches of T for $k = 1$. (c) Patches of T for $k = 2$.

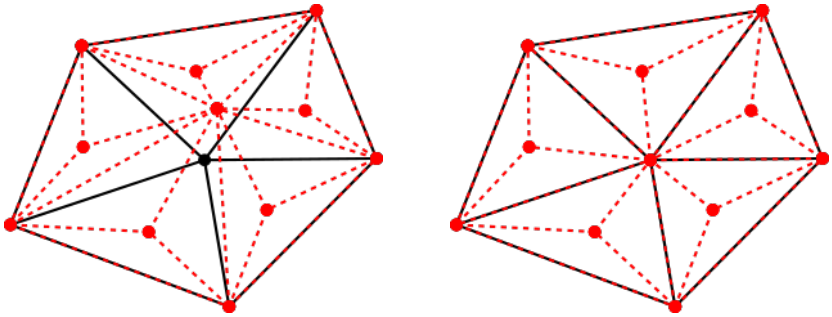
Figure 3.1: Illustration of patches.

To finish, we need one last definition. This definition will be useful for the localized orthogonality decomposition in Chapter 5.

Definition 3.1.4 (Refinement)**.** *Let τ_h and τ_H be two regular meshes of Ω with $0 < h \leq H$. We said τ_h to be a refinement of τ_H if*

$$\forall K \in \tau_H, \ \exists I_K \subset \mathbb{N}^*, \ K = \bigcup_{i \in I_K} k_i, \ k_i \in \tau_h.$$

Again, here is a picture (Figure 3.2) to illustrate the definition above. We are in the case d equal to 2. τ_h is in red and τ_H is in black. On the left picture, the node in the middle of τ_H (in black) does not coincide with any node of τ_h (in red). As a result we cannot decompose an element of τ_H by elements of τ_h. However, on the second picture we clearly see that τ_h is a refinement of τ_H.



(a) τ_h is not a refinement of τ_H. (b) τ_h is a refinement of τ_H.

Figure 3.2: Illustration of refinement.

In this section we saw how to discretize our domain Ω. It is important to notice that for implementations we will use quadrilaterals as elements instead of triangles. In the next section we define the finite element space in which our approximate functions will belong to.

3.2 The piecewise affine finite element spaces

To begin, we define the piecewise affine finite element spaces.

Definition 3.2.1 (The piecewise affine finite element space)**.** *Let τ_h be a regular simplicial mesh of size $h > 0$. We define his associated piecewise affine finite element space by*

$$V_h := \{v \in C^0(\bar{\Omega}) \mid v = 0 \text{ on } \partial\Omega \text{ and } v|_K \in \mathbb{P}^1,\ \forall K \in \tau_h\}.$$

As said before the approximation of our functions will be in these piecewise affine finite element spaces. For quadrilaterals elements instead of triangles, the piecewise affine finite element spaces are referred as $\mathbb{Q}^1$ instead of $\mathbb{P}^1$.

Lemma 3.2.1. *Let τ_h and τ_H be two regular meshes of Ω, $0 < h \leq H$. Let V_h and V_H be their respective piecewise affine finite element spaces. We suppose that τ_h is a refinement of τ_H. Then $V_H \subseteq V_h$.*

Proof. Let v_H be in V_H. We need to show that v_H belongs to V_h. In other words, for any element K_h in τ_h, we must have that $v_H|_{K_h}$ is affine i.e. is in $\mathbb{P}^1$.

Let us take an element K_h in τ_h. We affirm that there exists an unique element $K_H \in \tau_H$ such that $K_h \subset K_H$. We split the proof of the last statement in two parts. First we will prove the existence, and then the uniqueness.

• Existence : We will proceed by contradiction. Suppose that for all $K_H \in \tau_H$, $K_h \nsubseteq K_H$. Let T_H be an element of τ_H such that $T_H \cap K_h \neq \emptyset$. Such a T_H exists. Indeed, $K_h \subset \bigcup_i K_h^i = \bar{\Omega} = \bigcup_j K_H^j$, where K_h^i (resp. K_H^j) are elements of V_h (resp. V_H). However, it may not be unique.

If so, we randomly choose one. Since τ_h is a refinement of τ_H, we know that there exists a set $I \subset \mathbb{N}^*$ such that $T_H = \bigcup_{i\in I} T_h^i$, $T_h^i \in \tau_h$. By definition of T_H, we have

$$T_H \cap K_h \neq \emptyset \Leftrightarrow \Big(\bigcup_{i\in I} T_h^i\Big) \cap K_h \neq \emptyset \Rightarrow \exists i \in I,\ T_h^i = K_h.$$

The last statement is easily understandable since we have regular simplicial mesh i.e. every element of τ_h are two by two disjoint. Thus we have $K_h \subset T_H$. This is in contradiction with our assumption.

• Uniqueness : We showed that there exists $K_H \in \tau_H$ such that $K_h \subset K_H$. Again we will proceed by contradiction. Let us take K_H^1 and K_H^2 in τ_H such that $K_h \subset K_H^1$ and $K_h \subset K_H^2$. Moreover we suppose that $K_H^1 \neq K_H^2$. We have

$$K_h \subseteq K_H^1 \cap K_H^2.$$

This statement is impossible since we have regular simplicial mesh.

We successfully managed to prove that there exists an unique $K_H \in \tau_H$ such that $K_h \subset K_H$. We can come back to the proof of the Lemma now. Let K_H be in τ_H such that $K_h \subset K_H$. As v_H is in V_H, we know that $v_H|_{K_H}$ is an affine function. Then, so is $v_H|_{K_h}$. □

From now, when we write V_h we implicitly declare the mesh τ_h to whom V_h is associated.

3.3 The inverse inequality

Later in this master thesis, we will have to use the inverse inequality below.

Theorem 3.3.1 (Inverse property)**.** *Let V_h be a piecewise affine finite element space. Then V_h satisfies the inverse property*

$$\|\nabla v_h\| \leq C_{inv}\, h^{-1} \|v_h\|,\ \forall v_h \in V_h.$$

Proof. A more general version can be found in Section 8.5 of [9]. Otherwise there is a simpler proof in [2] for the one and two dimensional cases. □

We recall that $\|\nabla \cdot\|$ is also the semi-norm of $H_0^1(\Omega)$, which is $|\cdot|_{H_0^1(\Omega)}$. As a result for any u_h and v_h in V_h we can bound the bilinear form a by the L^2-norm of u_h and v_h i.e. $a(u_h, v_h) \leq C\, h^{-2} \|u_h\|\, \|v_h\|$.

3.4 Time discretization

To discretize in time we introduce the uniform discretization

$$0 = t_0 < t_1 < \cdots < t_N = T$$

and define $\Delta t := t_n - t_{n-1},\ \forall n \in \{1, \ldots, N\}$. This Δt is constant for every time step since the discretization is uniform.

At this point of the document, we have everything we need to start working on computing a solution for our parabolic problem.

Chapter 4

The finite element method

In the previous chapters, we formulated our problem and we discretized our spaces in order to be able to compute the solution. Computing the solution will be achieved by the following to steps. First we approximate the problem in space using the finite element method. Then, we approximate it in time by using the forward Euler method and later by using the Super-time stepping acceleration.

We work on a piecewise affine finite element space V_h with $0 < h$. All along this chapter, $\mathcal{N}_h$ denote the nodes of V_h, $\mathring{\mathcal{N}}_h$ denote the interior nodes of V_h and $(\Phi_i)_{i\in\mathcal{N}_h}$ denote the basis functions of V_h.

4.1 Semi-discrete finite element method

To begin, we approximate our weak formulation in space. As said before, we use the finite element method.

4.1.1 Formulation

First, let us recall the weak formulation of the parabolic problem (2.1)

Problem. *For all $t \in (0,T]$, find $u(\cdot,t) \in H_0^1(\Omega)$ such that*

$$\begin{cases} \left(\frac{\partial u}{\partial t}, v\right) + a(u,v) = (f,v), & \forall v \in H_0^1(\Omega), \\ u(\cdot,0) = u_0 & . \end{cases}$$

We make the approximation in space using the classical finite element method, and we get the following problem

Problem 4.1.1. *For all $t \in (0,T]$, find $u_h(\cdot,t) \in V_h$ such that*

$$\begin{cases} \left(\frac{\partial u_h}{\partial t}, v_h\right) + a(u_h,v_h) = (f,v_h), & \forall v_h \in V_h, \\ u_h(\cdot,0) = P_h \circ u_0 & . \end{cases} \tag{4.1}$$

Where $P_h : H_0^1(\Omega) \to V_h$ is a projection from $H_0^1(\Omega)$ into V_h. Here is a representation in one dimension of some basis functions Φ_i of the finite element space V_h.

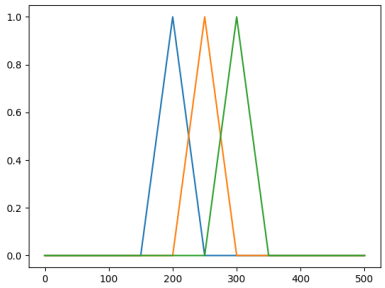


Figure 4.1: Basis functions of the piecewise affine finite element space V_h.

In one dimension, these functions, also called hat functions due to their shape, can be explicitly computed.

4.1.2 Matrix assembly

Now, we write an equivalent matrices formulation of the problem (4.1). We start with the first equation. For any v_h in V_h, we have

$$\left(\frac{\partial u_h}{\partial t}, v_h\right) + a(u_h, v_h) = (f, v_h)$$

As u_h is in V_h, we can decompose u_h in the basis of V_h i.e. $u_h(t) = \sum_{i \in \mathcal{N}_h} u_h|_i(t)\Phi_i$. Then

$$\begin{aligned}
&\left(\frac{\partial u_h}{\partial t}(t), v_h\right) + a(u_h(t), v_h) = (f(t), v_h) \\
\Leftrightarrow \quad &\left(\frac{\partial}{\partial t}\sum_{i \in \mathcal{N}_h} u_h|_i(t)\Phi_i, v_h\right) + a\left(\sum_{i \in \mathcal{N}_h} u_h|_i(t)\Phi_i, v_h\right) = (f(t), v_h) \\
\Leftrightarrow \quad &\sum_{i \in \mathcal{N}_h} \frac{\partial u_h|_i}{\partial t}(t)\big(\Phi_i, v_h\big) + \sum_{i \in \mathcal{N}_h} u_h|_i(t) a(\Phi_i, v_h) = (f(t), v_h).
\end{aligned}$$

We replace the test function v_h by Φ_j, $\forall j \in \mathcal{N}_h$

$$\sum_{i \in \mathcal{N}_h} \frac{\partial u_h|_i}{\partial t}(t)\big(\Phi_i, \Phi_j\big) + \sum_{i \in \mathcal{N}_h} u_h|_i(t) a(\Phi_i, \Phi_j) = (f(t), \Phi_j).$$

Thus, we obtain the weak formulation under matrix form

Problem 4.1.2. *For all $t \in (0, T]$, find $U_h(t)$ such that*

$$\begin{cases} \mathbb{M}\frac{\partial U_h}{\partial t}(t) + \mathbb{S}U_h = \mathbb{F}, \\ U_h(0) = \big((P_h \circ u_0)|_0, \ldots, (P_h \circ u_0)|_N\big)^T. \end{cases} \tag{4.2}$$

where

- $\mathbb{M} = [m_{ij}]_{(i,j)\in\mathcal{N}_h^2}$, $m_{ij} = \int_\Omega \Phi_i\Phi_j \, dx$, is the mass matrix,
- $\mathbb{S} = [s_{ij}]_{(i,j)\in\mathcal{N}_h^2}$, $s_{ij} = \int_\Omega (A\nabla\Phi_i)\cdot\nabla\Phi_j \, dx$, is the stiffness matrix,
- $U_h = (u_h|_0, \dots, u_h|_N)^T$, $N = |\mathcal{N}_h|$,
- $\mathbb{F} = [F_i]_{i\in\mathcal{N}_h}$, $F_i = \int_\Omega f(t)\Phi_i \, dx$.

This formulation allows us to realize the implementation of the finite element approximation of the parabolic problem in Section 4.2.6 and in Section 4.3.4. Furthermore, we will be able to prove the existence and uniqueness of the solution u_h of the finite element problem (4.1).

4.1.3 Well-posedness

Now, we verify that the problem (4.1) is well-posed i.e. we check either the stability, the existence and the uniqueness of the solution.

Theorem 4.1.1 (Stability)**.** *Let u_h be the solution of* (4.1)*. For all $t \in (0,T]$, we have the following results*

$$\|u_h(t)\| \le C\Big(\|P_h \circ u_0\| + \int_0^t \|f(s)\| \, ds\Big), \tag{4.3}$$

$$\left\|\frac{\partial u_h}{\partial t}(t)\right\| \le C\Big(\|P_h \circ u_0\| + \|f(t)\| + \int_0^t \|f(s)\| \, ds\Big), \tag{4.4}$$

$$\|u_h(t)\|_a \le C'\Big(\|P_h \circ u_0\| + \|f(t)\| + \int_0^t \|f(s)\| \, ds\Big). \tag{4.5}$$

Proof. We proceed in the same way as for the stability of u, Theorem 2.2.2. □

We proved the stability of the finite element methods. This stability ensure us that the approximate solution will note growth to infinity. It remains to verify the existence and the uniqueness of the solution in order to prove well-posedness.

Theorem 4.1.2 (Existence and uniqueness)**.** *There exists a unique solution of* (4.1)*.*

Proof. Since the mass matrix $\mathbb{M}$ in invertible, and the system (4.2) is an ordinary differential equations system, there is a unique solution for $t \in [0,T]$. □

4.1.4 Error estimate

In this section, we estimate the error due to the finite element method.

Lemma 4.1.1. *Let us define $\mathcal{E}_h : H_0^1(\Omega) \to V_h$ to be the elliptic projection from $H_0^1(\Omega)$ into V_h i.e. for any $v \in H_0^1(\Omega)$,*

$$a\Big(\mathcal{E}_h v, w_h\Big) = a\Big(v, w_h\Big), \forall w \in V_h.$$

Then, for any v in $H_0^1(\Omega)$, we have

$$\|v - \mathcal{E}_h v\| \leq C(\|A\|_{W^{1,\infty}(\Omega)}, \alpha, \beta) h \|v\|_a.$$

Proof. We consider the following auxiliary problem : find z in $H_0^1(\Omega)$ such that

$$a(z, w) = (v - \mathcal{E}_h v, w), \ \forall w \in H_0^1(\Omega). \tag{4.6}$$

Remplacing w by $v - \mathcal{E}_h v$ leads to

$$\|v - \mathcal{E}_h v\|^2 = a(z, v - \mathcal{E}_h v)$$

Since $\mathcal{E}_h z$ belongs to V_h, we have that $a(\mathcal{E}_h z, v - \mathcal{E}_h v) = 0$ by definition of $\mathcal{E}_h$. Then

$$\begin{aligned} \|v - \mathcal{E}_h v\|^2 &= a(z - \mathcal{E}_h z, v - \mathcal{E}_h v) \\ &\leq \|z - \mathcal{E}_h z\|_a \|v - \mathcal{E}_h v\|_a. \end{aligned}$$

$\|z - \mathcal{E}_h z\|_a$ is the a-norm of the error due to the Galerkin approximation of the elliptic problem (4.6). This error estimate has been computed in [2], Section 4.5. Hence

$$\|z - \mathcal{E}_h z\|_a \leq C(\|A\|_{W^{1,\infty}(\Omega)}, \alpha, \beta) h \|v - \mathcal{E}_h v\|.$$

Then, $\|v - \mathcal{E}_h v\| \leq C(\|A\|_{W^{1,\infty}(\Omega)}, \alpha, \beta) h \|v - \mathcal{E}_h v\|_a$. Finally, we have

$$\begin{aligned} \|v - \mathcal{E}_h v\|_a^2 &= a(v - \mathcal{E}_h v, v - \mathcal{E}_h v) \\ &= a(v - \mathcal{E}_h v, v) - a(v - \mathcal{E}_h v, \mathcal{E}_h v) \end{aligned}$$

Since $\mathcal{E}_h v$ lives in V_h we have $a(v - \mathcal{E}_h v, \mathcal{E}_h v) = 0$. Thus

$$\begin{aligned} \|v - \mathcal{E}_h v\|_a^2 &= a(v - \mathcal{E}_h v, v) \\ &\leq \|v - \mathcal{E}_h v\|_a \|v\|_a. \end{aligned}$$

It yields $\|v - \mathcal{E}_h v\| \leq C(\|A\|_{W^{1,\infty}(\Omega)}, \alpha, \beta) h \|v\|_a$. We have the result.

□

Theorem 4.1.3 (Error due to finite element approximation)**.** *Let u be the solution of* (2.2) *and u_h be the solution of* (4.1)*. Then for all $t \in (0, T]$, we have*

$$\|u(t) - u_h(t)\| \leq C\Big(\|A\|_{W^{1,\infty}(\Omega)}, \alpha, \beta\Big)\Big(\|u_0 - P_h \circ u_0\| + h\|u(t)\|_a\Big).$$

Proof. This proof will follow the one in [5]. Let us define $\mathcal{E}_h : H_0^1(\Omega) \to V_h$ as in Lemma 4.1.1. Thus if $v \in H_0^1(\Omega)$ is the solution of the following elliptic problem

$$a(v, w) = (f, w),\ \forall w \in H_0^1(\Omega)$$

then $\mathcal{E}_h v \in V_h$ is the finite element approximation of v. We fix $t \in [0, T]$. The idea of this proof is to split the error in two parts.

$$\|u(t) - u_h(t)\| \leq \| \underbrace{u(t) - \mathcal{E}_h[u(t)]}_{=:\epsilon_1(t)} \| + \| \underbrace{\mathcal{E}_h[u(t)] - u_h(t)}_{=:\epsilon_2(t)} \|.$$

- ϵ_1 : By using Lemma 4.1.1, we have

$$\|\epsilon_1(t)\| = \|u(t) - \mathcal{E}_h[u(t)]\| \leq C(\|A\|_{W^{1,\infty}(\Omega)}, \alpha, \beta) h \|u(t)\|_a.$$

- ϵ_2 : We put $\epsilon_2(t)$ instead of $u_h(t)$ in Problem 4.1

$$\left(\frac{\partial \epsilon_2(t)}{\partial t}, v_h\right) + a(\epsilon_2(t), v_h) = \left(\frac{\partial \mathcal{E}_h[u(t)]}{\partial t}, v_h\right) + a(\mathcal{E}_h[u(t)], v_h) - \Big(\underbrace{\left(\frac{\partial u_h(t)}{\partial t}, v_h\right) + a(u_h(t), v_h)}_{=(f(t), v_h)} \Big),$$

since u_h solve Problem 4.1. Furthermore $\mathcal{E}_h[u(t)]$ satisfies the Galerkin orthogonality i.e. $a(\mathcal{E}_h[u(t)], w_h) = a(u(t), w_h)$ for any $w \in V_h$

$$\begin{aligned}
\left(\frac{\partial \epsilon_2(t)}{\partial t}, v_h\right) + a(\epsilon_2(t), v_h) &= \left(\frac{\partial \mathcal{E}_h[u(t)]}{\partial t}, v_h\right) + a(u(t), v_h) - (f(t), v_h) \\
&= \left(\frac{\partial \mathcal{E}_h[u(t)]}{\partial t}, v_h\right) - \left(\frac{\partial u(t)}{\partial t}, v_h\right) \\
&= -\left(\frac{\partial \epsilon_1(t)}{\partial t}, v_h\right).
\end{aligned}$$

Thus ϵ_2 solve our parabolic problem 4.1 for $f = -\frac{\partial \epsilon_1}{\partial t}$ and $u_0 = \epsilon_2(0)$. We can apply the stability Theorem 4.1.1 and get the estimate

$$\|\epsilon_2(t)\| \leq C\Big(\|\epsilon_2(0)\| + \int_0^t \left\| \frac{\partial \epsilon_1}{\partial t}(s) \right\| \, ds\Big) = C\Big(\|\epsilon_2(0)\| + \|\epsilon_1(t)\| - \|\epsilon_1(0)\|\Big)$$

Here

$$\|\epsilon_2(0)\| = \|\mathcal{E}_h u_0 - P_h \circ u_0\| \leq \|\mathcal{E}_h u_0 - u_0\| + \|u_0 - P_h \circ u_0\| = \|\epsilon_1(0)\| + \|u_0 - P_h \circ u_0\|.$$

Combining with the estimation of $\|\epsilon_1(t)\|_{L^2(\Omega)}$

$$\begin{aligned}
\|\epsilon_2(t)\| &\leq C\Big(\|\epsilon_2(0)\| + \|\epsilon_1(t)\| - \|\epsilon_1(0)\|\Big) \\
&\leq C\Big(\|\epsilon_1(0)\| + \|u_0 - P_h \circ u_0\| + \|\epsilon_1(t)\| - \|\epsilon_1(0)\|\Big) \\
&\leq C\Big(\|u_0 - P_h \circ u_0\| + \|\epsilon_1(t)\|\Big) \\
&\leq C\Big(\|A\|_{W^{1,\infty}(\Omega)}, \alpha, \beta\Big)\Big(\|u_0 - P_h \circ u_0\| + h\|u(t)\|_a\Big).
\end{aligned}$$

We sum the estimation above with the estimation of $\|\epsilon_1(t)\|$ and we end the proof. □

Note 1. If we use the elliptic projection $\mathcal{E}_h$ instead of P_h, we get

$$\|u(t) - u_h(t)\| \leq C'\Big(\|A\|_{W^{1,\infty}(\Omega)}, \alpha, \beta\Big)h\Big(\|u_0\| + \|u(t)\|_a\Big).$$

Indeed, we have

$$\|u(t) - u_h(t)\| \leq C\Big(\|A\|_{W^{1,\infty}(\Omega)}, \alpha, \beta\Big)\Big(\|u_0 - \mathcal{E}_h \circ u_0\| + h\|u(t)\|_a\Big)$$

By applying Lemma 4.1.1 on $\|u_0 - \mathcal{E}_h \circ u_0\|$,

$$\|u(t) - u_h(t)\| \leq C'\Big(\|A\|_{W^{1,\infty}(\Omega)}, \alpha, \beta\Big)h\Big(\|u_0\|_a + \|u(t)\|_a\Big).$$

4.2 Explicit Euler scheme approximation

In the beginning of the chapter, we said that we will use two different methods to do the approximation in time. In this section we will see the first one, the explicit Euler scheme also called forward Euler scheme. We first write the formulation of the completely discretized method. Then, we move on the stability and the error. And finally, we implement the method in Python and present some numerical results.

4.2.1 Finite differences in time

To the aim of clearness we introduce the following notation.

Notations 2. For a function $v : \Omega \times (0, T] \to \mathbb{R}$, we denote $v(\cdot, n\Delta t)$ by $v|^n(\cdot)$.

Theorem 4.2.1. *Let $u : \Omega \times (0, T] \to \mathbb{R}$. Let $x \in \Omega$. We suppose that $u(x, \cdot)$ is at least C^1 in time. Then we have*

$$\frac{\partial u|^n(x)}{\partial t} = \frac{u|^{n+1}(x) - u|^n(x)}{\Delta t} + \mathcal{O}(\Delta t).$$

Proof. Taylor expansion leads to

$$\begin{aligned} u|^{n+1}(x) &:= u\Big(x, (n+1)\Delta t\Big) \\ &= u(x, n\Delta t) + \Delta t \frac{\partial u(x, n\Delta t)}{\partial t} + \mathcal{O}(\Delta t^2) \\ &= u|^n(x) + \Delta t \frac{\partial u|^n(x)}{\partial t} + \mathcal{O}(\Delta t^2) \end{aligned}$$

Thus

$$\frac{\partial u|^n(x)}{\partial t} = \frac{u|^{n+1}(x) - u|^n(x)}{\Delta t} + \mathcal{O}(\Delta t)$$

□

This theorem will be the keystone of the forward Euler method discussed in the next section.

4.2.2 Formulation

We formulate the completely discretized problem.

Definition 4.2.1. *Let u_h be the solution of (4.1). We denote by $\bar{u_h}$ the approximation of u_h such that*

$$\frac{\partial \bar{u_h}}{\partial t} = \frac{u_h|^{n+1} - u_h|^n}{\Delta t}.$$

Using this definition leads us

$$\begin{aligned}
&\left(\frac{\bar{u_h}|^{n+1} - \bar{u_h}|^n}{\Delta t}, v_h\right) + a(\bar{u_h}|^n, v_h) = (f|^n, v_h),\ \forall v_h \in V_h \\
&\big(\bar{u_h}|^{n+1}, v_h\big) - \big(\bar{u_h}|^n, v_h\big) + \Delta t\ a(\bar{u_h}|^n, v_h) = \Delta t (f|^n, v_h),\ \forall v_h \in V_h \\
&\big(\bar{u_h}|^{n+1}, v_h\big) = \big(\bar{u_h}|^n, v_h\big) + \Delta t\Big((f|^n, v_h) - a(\bar{u_h}|^n, v_h)\Big),\ \forall v_h \in V_h
\end{aligned}$$

This gives us the final problem

Problem 4.2.1. *Knowing $\bar{u_h}|^n$, find $\bar{u_h}|^{n+1} \in V_h$ such that*

$$\begin{cases} \big(\bar{u_h}|^{n+1}, v_h\big) & = \big(\bar{u_h}|^n, v_h\big) + \Delta t\Big((f|^n, v_h) - a(\bar{u_h}|^n, v_h)\Big),\ \forall v_h \in V_h, \\ \bar{u_h}|^0 = P_h \circ u_0 & . \end{cases} \tag{4.7}$$

4.2.3 Stability and the Courant–Friedrichs–Lewy condition

As we use the forward Euler scheme in time, our scheme is not always stable. In this section we will find the condition between the discretization parameters h and Δt for the scheme to be stable. This condition is known as the Courant–Friedrichs–Lewy condition.

Theorem 4.2.2 (Stability)**.** *If we suppose*

$$1 - C_{inv}\beta h^{-2}\frac{\Delta t}{2} \geq 0 \tag{4.8}$$

then the scheme (4.7) is stable i.e. if $\bar{u_h}$ is the solution of (4.7), then for any n we have

$$\|\bar{u_h}|^{n+1}\| \leq \|P_h \circ u_0\| + 2\Delta t \sum_{j=0}^{n} \|f|^j\|.$$

Proof. This proof will follow the idea in [2]. First of all we suppose that (4.8) holds. Then we use $\bar{u_h}|^{n+1}$ as test function in (4.7). It gives

$$\|\bar{u_h}|^{n+1}\|^2 = \big(\bar{u_h}|^n, \bar{u_h}|^{n+1}\big) + \Delta t\Big((f|^n, \bar{u_h}|^{n+1}) - a(\bar{u_h}|^n, \bar{u_h}|^{n+1})\Big).$$

The trick here is to use the two following identities : for any u and v in $H_0^1(\Omega)$, we have

$$(u,v) = -\frac{1}{2}\Big(\|u-v\|^2 - \|u\|^2 - \|v\|^2\Big),$$
$$a(u,v) = \frac{1}{4}\|u+v\|_a^2 - \frac{1}{4}\|u-v\|_a^2.$$

Thus we have

$$\|\bar{u_h}|^{n+1}\|^2 = -\frac{1}{2}\Big(\|\bar{u_h}|^{n+1} - \bar{u_h}|^n\|^2 - \|\bar{u_h}|^{n+1}\|^2 - \|\bar{u_h}|^n\|^2\Big) + \Delta t(f|^n, \bar{u_h}|^{n+1}) - \\ \Delta t\frac{1}{4}\|\bar{u_h}|^{n+1} + \bar{u_h}|^n\|_a^2 + \Delta t\frac{1}{4}\|\bar{u_h}|^{n+1} - \bar{u_h}|^n\|_a^2$$

$$\|\bar{u_h}|^{n+1} - \bar{u_h}|^n\|^2 + \|\bar{u_h}|^{n+1}\|^2 - \|\bar{u_h}|^n\|^2 + \frac{\Delta t}{2}\|\bar{u_h}|^{n+1} + \bar{u_h}|^n\|_a^2 - \\ \frac{\Delta t}{2}\|\bar{u_h}|^{n+1} - \bar{u_h}|^n\|_a^2 = 2\Delta t(f|^n, \bar{u_h}|^{n+1}).$$

Since $\|\bar{u_h}|^{n+1} + \bar{u_h}|^n\|_a^2$ is positive, we have

$$\|\bar{u_h}|^{n+1} - \bar{u_h}|^n\|^2 + \|\bar{u_h}|^{n+1}\|^2 - \|\bar{u_h}|^n\|^2 - \frac{\Delta t}{2}\|\bar{u_h}|^{n+1} - \bar{u_h}|^n\|_a^2 \le 2\Delta t(f|^n, \bar{u_h}|^{n+1}).$$

The troublesome term is $-\frac{\Delta t}{2}\|\bar{u_h}|^{n+1} - \bar{u_h}|^n\|_a^2$ because it is negative. The term $-\|\bar{u_h}|^n\|^2$ is also negative but is not a problem for us since $\|\bar{u_h}|^{n+1} - \bar{u_h}|^n\|^2 \ge \|\bar{u_h}|^n\|^2 - \|\bar{u_h}|^{n+1}\|^2$. Thus, we move the troublesome term to the right hand side and we use the boundedness of a and the inverse inequality

$$\|\bar{u_h}|^{n+1} - \bar{u_h}|^n\|^2 + \|\bar{u_h}|^{n+1}\|^2 - \|\bar{u_h}|^n\|^2 \le 2\Delta t(f|^n, \bar{u_h}|^{n+1}) + C_{\text{inv}}\beta h^{-2}\frac{\Delta t}{2}\|\bar{u_h}|^{n+1} - \bar{u_h}|^n\|^2.$$

Now, we move it to the left hand side

$$\Big(1 - C_{\text{inv}}\beta h^{-2}\frac{\Delta t}{2}\Big)\|\bar{u_h}|^{n+1} - \bar{u_h}|^n\|^2 + \|\bar{u_h}|^{n+1}\|^2 - \|\bar{u_h}|^n\|^2 \le 2\Delta t(f|^n, \bar{u_h}|^{n+1}).$$

As we supposed that $1 - C_{\text{inv}}\beta h^{-2}\frac{\Delta t}{2} \ge 0$, the inequality above and the Cauchy-Schwarz inequality gives

$$\|\bar{u_h}|^{n+1}\|^2 - \|\bar{u_h}|^n\|^2 \le 2\Delta t\|f|^n\|\|\bar{u_h}|^{n+1}\| \\ \le 2\Delta t\|f|^n\|\Big(\|\bar{u_h}|^{n+1}\| + \|\bar{u_h}|^n\|\Big)$$

and finally,

$$\|\bar{u_h}|^{n+1}\| - \|\bar{u_h}|^n\| \leq 2\Delta t \|f|^n\|$$

or

$$\|\bar{u_h}|^{n+1}\| \leq \|\bar{u_h}|^n\| + 2\Delta t \|f|^n\|.$$

By immediate recursivity and that $\bar{u_h}|^0 = P_h \circ u_0$, we obtain the result

$$\|\bar{u_h}|^{n+1}\| \leq \|P_h \circ u_0\| + 2\Delta t \sum_{j=0}^{n} \|f|^j\|.$$

□

Note 2. In this paper, we only proved that this condition is sufficient to have the stability of the forward Euler-Galerkin method. It can be rewritten to emphasizes the link between the mesh size h and the time step Δt as $\Delta t \leq 2C_{\text{inv}}^{-1}\beta^{-1}h^2$. We notice that this condition is very restrictive since we require a very small time steps to ensure the stability. It is even worst for contrasted media i.e. for large β.

4.2.4 Error of the completely discretized method

Let's estimate the error due to the approximations. We first need to find the error due to the forward Euler scheme.

Theorem 4.2.3 (Error due to the forward Euler scheme)**.** *Let u_h be the solution of* (4.1) *and be C^2 in time and $\bar{u_h}$ be the solution of* (4.7)*. If we suppose that u_h is C^2 in time and that the Courant–Friedrichs–Lewy condition* (4.8) *true then we have the following error estimation*

$$\|u_h|^n - \bar{u_h}|^n\| \leq C\Big(n, \Big\|\frac{\partial^2 u_h}{\partial t^2}\Big\|\Big)\Delta t.$$

Proof. Let us define the discretization error by

$$e_n := u_h|^n - \bar{u_h}|^n$$

Taylor's expansion with integral remainder gives us

$$u_h|^{n+1}(x) = u_h|^n(x) + \Delta t \frac{\partial u_h}{\partial t}|^n(x) + R_n(x),$$

where $R_n(x) = \int_{t_n}^{t_{n+1}} (t_{n+1} - t)\frac{\partial^2 u_h(x,t)}{\partial t^2}\, dt$. We have for all v_h in V_h

$$
\begin{aligned}
(e_{n+1}, v_h) &= (u_h|^{n+1}, v_h) - (\bar{u_h}|^{n+1}, v_h) \\
&= (u_h|^n, v_h) + \Delta t \left(\frac{\partial u_h}{\partial t}|^n, v_h \right) + (R_n(x), v_h) - (\bar{u_h}|^{n+1}, v_h) \\
&= (u_h|^n, v_h) + \Delta t \Big((f|^n, v_h) - a(u_h|^n, v_h) \Big) - \big(\bar{u_h}|^n, v_h\big) - \Delta t \Big((f|^n, v_h) - a(\bar{u_h}|^n, v_h) \Big) + (R_n(x), v_h) \\
&= (e_n, v_h) - \Delta t a(e_n, v_h) + (R_n(x), v_h) \\
&= (e_n, v_h) + \Delta t \Big((\frac{1}{\Delta t} R_n(x), v_h) - a(e_n, v_h) \Big)
\end{aligned}
$$

Thus, e_n satisfies the problem (4.7) with $u_0 = e_0$ and $f|^n = \frac{1}{\Delta t} R_n$. Under the Courant–Friedrichs–Lewy condition we can apply the stability theorem 4.2.2 and we get the following estimate

$$
\|e_{n+1}\| \leq \|e_0\| + 2\Delta t \sum_{j=0}^{n} \left\| \frac{1}{\Delta t} R_j \right\| = \|e_0\| + 2\Delta t \sum_{j=0}^{n} \left\| \int_{t_j}^{t_{j+1}} \frac{t_{j+1} - t}{\Delta t} \frac{\partial^2 u_h(\cdot, t)}{\partial t^2} \, dt \right\|.
$$

We have $\|e_0\| = \|u_h|^0 - \bar{u_h}|^0\| = \|P_h \circ u_0 - P_h \circ u_0\| = 0$. Then by switching the integral and the L^2-norm we get

$$
\|e_{n+1}\| \leq 2\Delta t \sum_{j=0}^{n} \int_{t_j}^{t_{j+1}} \frac{t_{j+1} - t}{\Delta t} \left\| \frac{\partial^2 u_h(\cdot, t)}{\partial t^2} \right\| \, dt.
$$

In addition, we have $\frac{t_{j+1} - t}{\Delta t} \leq 1$ for all t between t_j and t_{j+1}. Then

$$
\|e_{n+1}\| \leq 2\Delta t \sum_{j=0}^{n} \int_{t_j}^{t_{j+1}} \left\| \frac{\partial^2 u_h(\cdot, t)}{\partial t^2} \right\| \, dt = 2\Delta t \int_{0}^{t_{n+1}} \left\| \frac{\partial^2 u_h(\cdot, t)}{\partial t^2} \right\| \, dt.
$$

□

Now, we have everything we need to estimate the error of the completely discretized method.

Theorem 4.2.4 (Error of the completely discretized method)**.** *Let u be the solution of* (2.2) *and C^2 in time, u_h be the solution of* (4.1) *and C^2 in time, and $\bar{u}_h$ be the solution of* (4.7)*. Then for all n we have*

$$
\|u|^n - \bar{u}_h|^n\| \leq C\Big(\beta, \alpha^{-1}, n, \|f|^n\|, \left\| \frac{\partial^2 u_h}{\partial t^2}|^n \right\| \Big) \Big(\|u_0 - P_h \circ u_0\| + h + \Delta t \Big).
$$

Proof. Use the triangle inequality, Theorem 4.2.3 and 4.1.3. □

4.2.5 Matrix assembly

In order to implement our approximate scheme with a computer, we have to construct an equivalent problem with matrices. We start from (4.7) and you do the same kind of computation as for the matrix assembly for the semi-discretized finite element method in Section 4.1.2. Thus we obtain

Problem 4.2.2. *Knowing* $\bar{U}_h|^n$*, find* $\bar{U}_h|^{n+1}$ *such that*

$$\begin{cases} \mathbb{M}\bar{U}_h|^{n+1} = \big(\mathbb{M} - \Delta t\mathbb{S}\big)\bar{U}_h|^n + \Delta t\mathbb{F}|^n, \\ \bar{U}_h|^0 = \big((P_h \circ u_0)|_0, \dots, (P_h \circ u_0)|_N\big)^T. \end{cases} \tag{4.9}$$

where

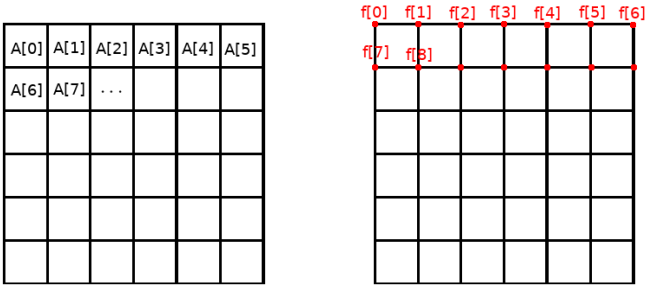
- $\bar{U}_h|^n = (\bar{u}_h|_0^n, \dots, \bar{u}_h|_N^n)^T$, where $N = \mathcal{N}_H$,
- $\mathbb{F}|^n = [F_i]_{i\in\mathcal{N}_h}$, $F_i = \int_\Omega f|^n \Phi_i \, dx$.

4.2.6 Implementation in Python

In order to do the implementation of the finite element method in python, and later the implementation of the localized orthogonal decomposition method, we use the python library gridlod [4]. This library only works on the domain $\Omega = [0,1]^d$ with $d \in \{1,2,3\}$. Moreover, we can only take an approximation of f in V_h. Thus we have $\mathbb{F}|^n = \mathbb{M}F|^n$, where $F|^n = (f|_0^n, \cdots, f|_N^n)^T$ and $\mathbb{M}$. The reader can find all the python code in Appendix. For the finite element method, see Appendix A and Appendix C.1. Now let us explain some important point of the code. First to define a mesh, we use

```
NFine = np.array([Nx, Ny])
```

This allows us to create a two dimensional mesh with Nx elements in the x direction and Ny elements in the y direction. The diffusion coefficient A is defined by a constant on each element (see Figure 4.2). It must be a vector of size $Nx \times Ny$. Unlike the diffusion coefficient, each functions are defined on each node of the mesh (see Figure 4.2). Thus they must be a vector of size $(Nx+1) \times (Ny+1)$.



(a) Representation of the diffusion coefficient A.

(b) Representation of functions, for example f.

Figure 4.2: Data representation in Python.

In order to compute the mass matrix $\mathbb{M}$ and the stiffness matrix $\mathbb{S}$ we use

```
problemFEM.assembleMatrices()
```

Finally, to solve our finite element method with the forward Euler scheme (4.9) we use

```
problemFEM.solveStep(xFullFEM, delta_t)
```

4.2.7 Numerical results

Now we do some experiment. For understanding issue you can refer to the Section 4.2.6 above. First, we will begin with checking the Courant–Friedrichs–Lewy condition and the stability from Theorem 4.2.2. Finally we will presents an experimental error analysis of the numerical method.

Stability

To begin with the experiments, we check the stability of the method discussed in Theorem 4.2.2. We recall the reader that the the Courant–Friedrichs–Lewy condition (4.8) is the following

$$\Delta t \leq C_{\text{CFL}} \beta^{-1} h^2.$$

First of all we wish determine the constant C_{CFL}. We vary the constant while fixing the other parameters and see whether the method is stable or not. The diffusion coefficient have random values between α and β. Here is a table (Table 4.1) which recap the results of this experiment.

C_{cfl}	α	β	h	stable
0.0089	1	1	0.1	Yes
0.009	1	1	0.1	No

Table 4.1: Influence of the constant C_{CFL} on the stability.

Thus, we see that if $C_{\text{CFL}} \leq 0.0089$, the scheme is stable. Nevertheless it is hard to check if this value will ensure the stability over infinite time. To be sure about the stability we can take a bit less by multiplying the C_{CFL} by a constant lower than 1. Now, let's check the influence of β and more especially the influence of the order of β. Because we theoretically found that the order with respect to β is 1, we will first try we an order lower, here $\frac{1}{2}$. Again here is a recap (Table 4.2).

C_{cfl}	α	$\beta^{1/2}$	h	stable
0.0089	1	1	0.1	Yes
0.0089	1	$\sqrt{2}$	0.1	No

Table 4.2: Influence of the diffusion coefficient parameter β with order $\frac{1}{2}$ on the stability.

We can see that the order $\frac{1}{2}$ with respect to β is not sufficient to ensure the stability of the method. Let's try with order 1 (Table 4.3).

C_{cfl}	α	β	h	stable
0.0089	1	1	0.1	Yes
0.0089	1	5	0.1	Yes
0.0089	1	10	0.1	Yes
0.0089	1	20	0.1	Yes
0.0089	1	100	0.1	Yes
0.0089	1	1000	0.1	Yes

Table 4.3: Influence of the diffusion coefficient parameter β with order 1 on the stability.

Now, the scheme seems to stay stable for every β. This result is in accordance to the result we found theoretically earlier. Finally, we wish verify if the Courant–Friedrichs–Lewy condition is independent of α (Table 4.4).

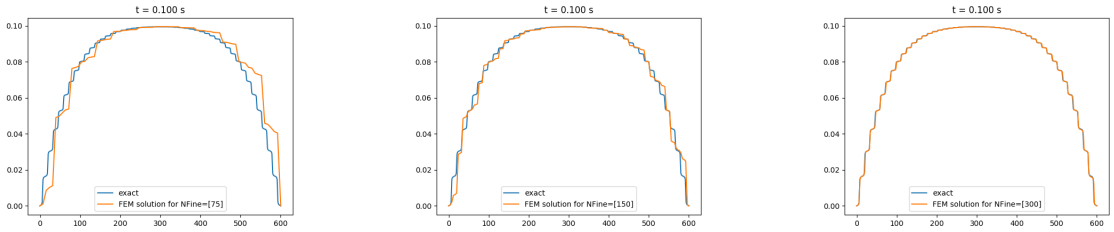
C_{cfl}	α	β	h	stable
0.0089	1	1	0.1	Yes
0.0089	2	1	0.1	Yes
0.0089	10	1	0.1	Yes
0.0089	100	1	0.1	Yes
0.0089	500	1	0.1	Yes
0.0089	1000	1	0.1	Yes

Table 4.4: Influence of the diffusion coefficient parameter α on the stability.

The parameter α seems not to influence the stability. Then the Courant–Friedrichs –Lewy condition found theoretically and the one found experimentally are the same. This is reassuring.

Pre-asymptotic effect

For the sake of clarity, we will do these experiments in one dimension. We want to try the finite element method for a highly oscillating coefficient A. We choose $A(x) = \cos(290 \times x) + 1.01$ and $f = 1$. We have the solutions for different value of h on Figure 4.3.



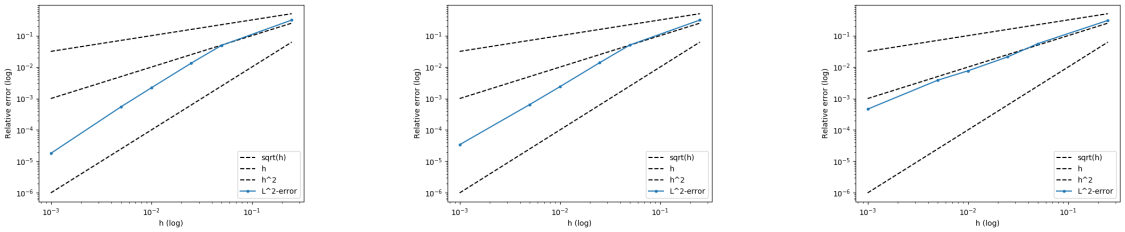
(a) $h = 75^{-1}$. (b) $h = 150^{-1}$. (c) $h = 300^{-1}$.

Figure 4.3: Illustration of the pre-asymptotic effect.

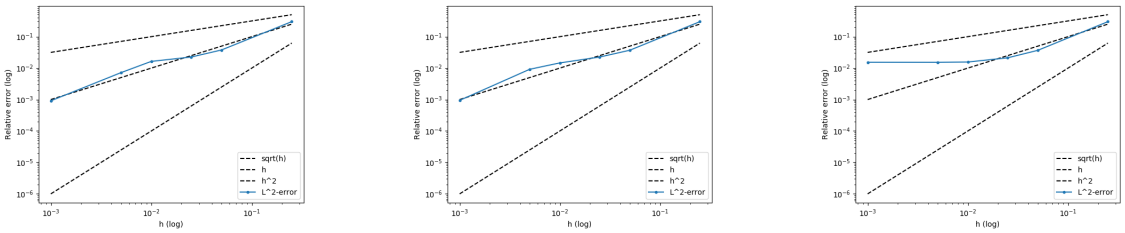
The first two pictures of Figure 4.3 are far from the exact solution, but the third one is not. Hence, it indicates that the solution of the finite element method and the exact solution are close if $h \lesssim \epsilon$, where ϵ is the frequency of oscillations of A. Otherwise, the approximate solution have no sens. This effect is known as the preasymptotic effect.

Error

We check the order of the convergence rate in space and in time. We do these two experiments in one dimension. In both case, we take the diffusion coefficient $A(x) = (2 + \cos(\frac{2\pi}{\epsilon}x))^{-1}$, for $0 < \epsilon \leq 1$. Moreover, we have $u_0 = 0$ and $f = 1$. We begin with the convergence rate in space, Figure 4.4. We draw the relative L^2-error $\|u - u_h\|/\|u\|$ depending on the mesh size h. We fix Δt such that the Courant–Friedrichs–Lewy condition (4.8) is satisfies for every h.



(a) $\varepsilon = 2^0$. (b) $\varepsilon = 2^{-1}$. (c) $\varepsilon = 2^{-2}$.



(d) $\varepsilon = 2^{-3}$. (e) $\varepsilon = 2^{-4}$. (f) $\varepsilon = 2^{-5}$.

Figure 4.4: L^2-error with respect to the mesh size h.

On the picture (a), A have almost no variations. As a result the convergence rate is of order 2 with respect to the mesh size h. We only see that for large mesh size ($\epsilon \lesssim h$), the order is only 1. The is due to the preasymptotic effect. We have the same thing on the picture (b). However from picture (c), the mesh size is too large compared to the variations of A. Indeed we have $\epsilon \lesssim h$. We can conclude that if $h \lesssim \epsilon$ the convergence rate of the finite element method is of order 2 with respect to the mesh size h and if $\epsilon \lesssim h$, the small variations of A are not well-captured and, as a result, we only have an order of 1 with respect to h for the convergence rate.

Now we study the convergence rate in time. We draw on Figure 4.4 the relative L^2-error $\|u - u_h\|/\|u\|$ depending on the time step Δt with $\epsilon = 2^{-4}$. We fix the mesh size h such that the Courant–Friedrichs–Lewy condition (4.8) is satisfies for every Δt.

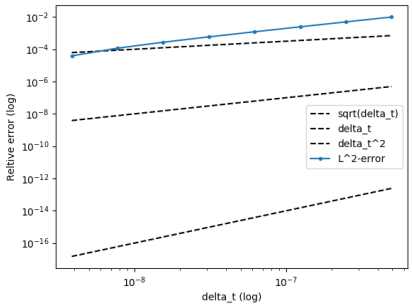


Figure 4.5: L^2-error with respect to the time step Δt.

It seems that, under the Courant–Friedrichs–Lewy condition, the convergence rate of the forward Euler method is of order 1 with respect to the time step Δt. This aligns with our theory.

4.3 Super-time Stepping acceleration

We have successfully computed a good approximation for our solution. However, the Courant–Friedrichs –Lewy condition is an issue for us. Indeed, if we want a very accurate solution, we have to choose a small h and consequently an even smaller Δt. This is not possible if we want to study our problem over a long period. That's why we now introduce an other method : the Super-time Stepping acceleration.

4.3.1 Formulation

The idea behind the super-time-stepping scheme is to solve N times an explicit Euler scheme.

As in [1], let us define a super-step ΔT consisting of N time-steps $\tau_1, \cdots, \tau_N$ i.e. $\Delta T := \sum_{i=1}^{n} \tau_i$. We want to maximize ΔT and to ensure stability over the super-step ΔT instead over each steps. That's why the inner values have no approximating properties and should be considered as intermediate calculations. We have the following problem

Problem 4.3.1. *Knowing $\bar{u_h}|^n$, find $\bar{u_h}|^{n+1} \in V_h$ such that*

$$\begin{cases} \left(\bar{u_h}|^{n+1/N}, v_h\right) &= \left(\bar{u_h}|^{n}, v_h\right) &+\tau_1\Big((f|^n, v_h) - a(\bar{u_h}|^n, v_h)\Big), &\forall v_h \in V_h, \\ \left(\bar{u_h}|^{n+2/N}, v_h\right) &= \left(\bar{u_h}|^{n+1/N}, v_h\right) &+\tau_2\Big((f|^n, v_h) - a(\bar{u_h}|^{n+1/N}, v_h)\Big), &\forall v_h \in V_h, \\ &\vdots & & \\ \left(\bar{u_h}|^{n+1}, v_h\right) &= \left(\bar{u_h}|^{n+(N-1)/N}, v_h\right) &+\tau_N\Big((f|^n, v_h) - a(\bar{u_h}|^{n+(N-1)/N}, v_h)\Big), &\forall v_h \in V_h, \end{cases} \tag{4.10}$$

and such that $\bar{u_h}|^0 = P_h \circ u_0$.

4.3.2 Stability

As we use many forward Euler scheme, the Super-time Stepping is not unconditionally stable. However, unlike the forward Euler scheme, we want to ensure the stability over a super-step ΔT instead of over a single step Δt. It has been proved in [1] that in order to have the stability of the Super-time Stepping, we must have the following inner steps

$$\tau_i = \Delta t_{\text{expl}}\Big((\nu - 1)\cos\big(\frac{2i-1}{N}\frac{\pi}{2}\big) + 1 + \nu\Big)^{-1}, \ \forall i \in \{1, \cdots, N\},$$

where $0 < \nu < \mu_{\min}/\mu_{\max}$, $(\mu_i)_i$ are the eigen values of $\mathbb{M}^{-1}\mathbb{S}$.

4.3.3 Error of the completely discretized method

Theorem 4.3.1 (Error due to the Super-time Stepping method)**.** *The Super-time stepping method is of order one with respect to ΔT.*

Proof. See [1]. □

The theorem above combined with the error due to the semi-discretized finite element method gives us the error of the completely discretized method below.

Theorem 4.3.2 (Error of the completely discretized method)**.** *Let u be the solution of* (2.2) *and $\bar{u}_h$ be the solution of* (4.10) *Then,*

$$\|u|^n - \bar{u}_h|^n\| \leq C\Big(\beta, \alpha^{-1}, \|f(t)\|,\Big)\Big(\|u_0 - P_h \circ u_0\| + h + \Delta T\Big).$$

Proof. Use the triangle inequality, Theorem 4.3.1 and 4.1.3. □

4.3.4 Implementation in Python

Before solving the super-time stepping system (4.10), we have to compute the τ_i. For it we use

```
problemFEM.initSuperStep(N, nu)
```

And then to solve the system (4.10), we invoke

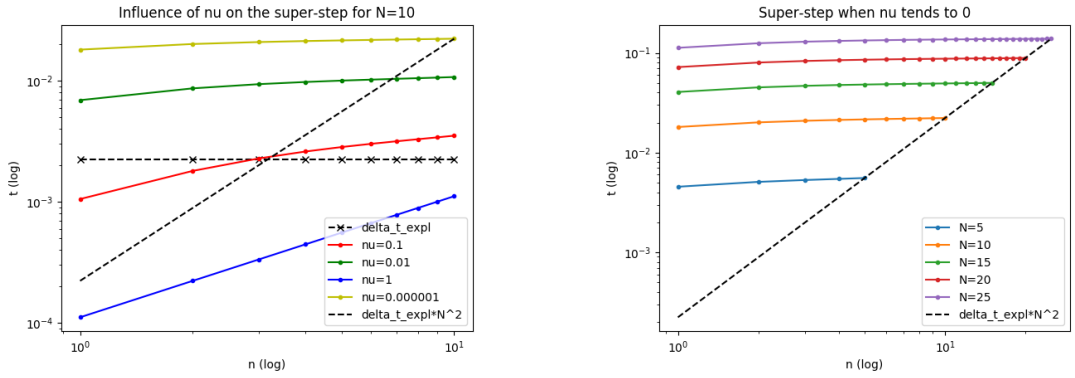
```
problemFEM.solveSuperStep(xFullFEM, N, nu)
```

4.3.5 Numerical results

Like with the forward Euler method, we do experiments to confirm the theory.

Computation of the inner steps

We verify the influence of N and ν on the inner steps τ_i.



(a) Comparison between Δt_{expl} and different super-step ΔT.

(b) Value of ΔT when ν tends to 0.

Figure 4.6: Evolution of the super-step ΔT.

On these graphics (Figure 4.6), we drew in color the following value :

$$\sum_{i=1}^{n} \tau_i(N, \nu), \ \forall n \in \{1, \ldots, N\}.$$

On the first one, we fix N to ten then we vary ν. For large ν, (f.e. ν equal to one), we see that $\Delta T < \Delta t_{\text{expl}}$. In that case, the Super-Time Stepping acceleration is less efficient than the forward Euler. However, for a sufficiently small ν, the Super-Time Stepping acceleration is quicker than the forward Euler. Even more, if $\nu \to 0$, we see that $\Delta T \approx N^2 \Delta t_{\text{expl}}$. This result is confirmed theoretically in [1] and the figure (b) :

$$\lim_{\nu \to 0} \Delta T = N^2 \Delta t_{\text{expl}}.$$

Error

Again, we do the experiments in the one dimensional case. We fix $N = 20$ and we vary ν between 0.001 and 2. We remind the reader that the more ν is close to 0, the more the super-time step ΔT is high. We draw the relative error on Figure 4.7.

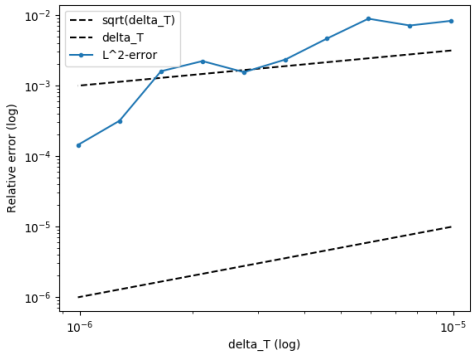
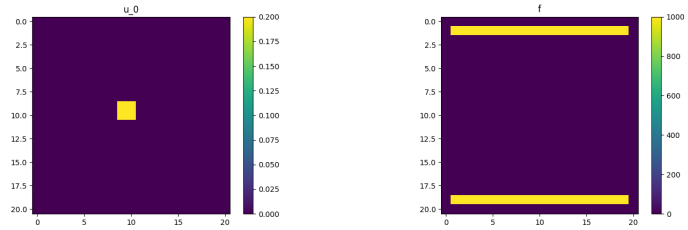


Figure 4.7: L^2-error with respect to the super time step ΔT.

The relative error globally follows the ΔT line. Thus the convergence rate of the super-time stepping is of order 1 with respect to ΔT. The variations can be explained due to the implementation of this experiment. Indeed we fix the same end time T for every computation. However, it happens that this T in not divisible by ΔT. As a result, some computation stop a bit earlier or a bit later than the fixed end time T. Hence, we compare the exact solution with the computed solutions at different time around T.

Solution

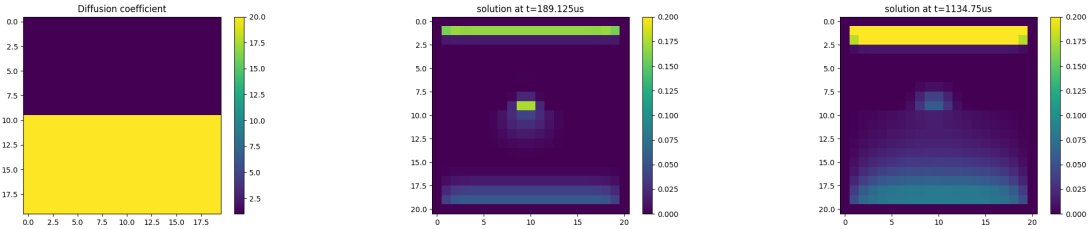
To finish this chapter, here is some drawing, in two dimension, of $\bar{u}_h|^n$ for two kind of diffusion coefficient. The mesh size h is $h = 20^{-1}$. We take u_0 and f like in Figure 4.8.



(a) The initial condition u_0. (b) The forcing term f.

Figure 4.8: Experiment data.

On Figure 4.9, we split the diffusion coefficient A in two parts. On the top we have a non-conductive coefficient ($A_{\text{top}} = 0.01$) and at the bottom a very-conductive coefficient ($A_{\text{bottom}} = 200$). On the top part i.e. the non-conductive part, the heat is very localized and high. On the bottom part i.e. the conductive part, the heat is diffuse and lower than the one in the non-conductive part. This is because at the bottom, the heat transfer is easier than on the top part. As a result, the heat is distributed and is easily absorbed by the boundaries (due to the boundary condition).



(a) Diffusion coefficient A. (b) Solution $\bar{u}_h|^n$. (c) Solution $\bar{u}_h|^n$.

Figure 4.9: Computation with a binary diffusion coefficient.

Chapter 5

The localized orthogonal decomposition

Previously, we managed to compute an approximate solution of the parabolic problem. We saw that the error on this approximation is of order one with respect to the mesh size h. Moreover, we have a pre-asymptotic effect. To avoid it, we have to choose h lower than the variation of the diffusion coefficient A. This condition is not acceptable for a A with fast variations. To solve this problem, we will use one of the multiscale methods.

First, we introduce the interpolation operator that we will use for the method. Then, we present the orthogonal decomposition. After that, we localize this decomposition in order to achieve the localized orthogonal decomposition. To finish, like for the Chapter 4 we use, in a first time, the forward Euler method, then move on the Super-time Stepping.

In this chapter, we consider the piecewise affine finite element spaces V_h and V_H, $h \leq H$. Moreover, we consider that τ_h is a refinement of τ_H. As we saw in Lemma 3.2.1, we have $V_H \subset V_h$. Finally, $\mathcal{N}_h$ (resp. $\mathcal{N}_H$) will denote the interior nodes of V_h (resp. V_H) and Φ_x (resp. φ_x) the corresponding hat functions. Sometimes, τ_h is called fine mesh and τ_H is called coarse mesh.

5.1 Orthogonal decomposition

The idea behind the localized orthogonal decomposition is to split the space of solution in two. The first one will correspond to the interpolation of the solution with a specific interpolation operator. The aim of this first space is to describe the coarse variations of the solution. The second one correspond the fine scale variations of the solution. Then, the fine scale space will be orthogonalized with respect to a specific scalar product in order to have convenient properties.

5.1.1 The interpolation operator

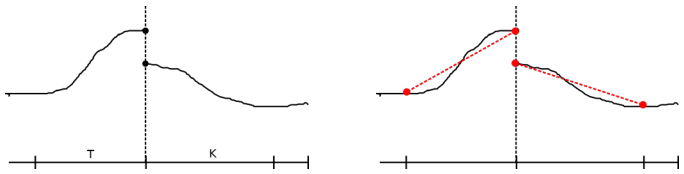
First, we present the interpolation operator that we will use for the coarse variations space.

Definition 5.1.1 (Local L^2-orthogonal projection)**.** *Let V_ξ be piecewise affine finite element spaces, $0 < \xi$. Then $\Pi_\xi|_T : L^2(T) \to \mathbb{P}_1(\tau_\xi)$ is the $L^2(T)$-orthogonal projection onto affine functions on the element T, i.e.,*

$$\int_T \Pi_H|_\xi(v)w \; dx = \int_T vw \; dx$$

for all w in $\mathbb{P}_1(\tau_\xi)$.

Note 3. First, it is important to notice that the result of Π_ξ may be discontinuous (see Figure 5.1). In addition, if v is already an affine function i.e. is in $\mathbb{P}_1(\tau_\xi)$, we have $\Pi_\xi(v) = v$.



(a) A L^2 function v.. (b) $\Pi_H \circ v$ in red.

Figure 5.1: Illustration of the $L^2(T)$-orthogonal projection onto affine functions.

As we saw in the previous note, the result of Π_ξ is possibly discontinuous. For more convenience, we would prefer a continuous interpolation operator. To ensure the continuity of the interpolation operator, we define the averaging operator.

Definition 5.1.2 (Averaging operator)**.** *Let V_ξ be piecewise affine finite element spaces, $0 < \xi$. Let $E_\xi : \mathbb{P}_1(\tau_\xi) \to V_\xi$ be the averaging operator defined by : for any $v \in \mathbb{P}_1(\tau_\xi)$ and for any z an interior node of τ_ξ,*

$$(E_\xi v)(z) := \Big|\{K \in \tau_\xi \,/\, z \in K\}\Big|^{-1} \sum_{T \in \tau_\xi,\, z \in T} v|_T(z)$$

This averaging operator have the following property.

Property 5.1.1. *If $v_\xi \in V_\xi$, then $E_\xi v_\xi = v_\xi$.*

Proof. Let $v_\xi \in V_\xi$ and let z be an interior node of τ_ξ.

$$(E_\xi v_\xi)(z) = \Big|\{K \in \tau_\xi \,/\, z \in K\}\Big|^{-1} \sum_{T \in \tau_\xi,\, z \in T} v_H|_T(z)$$

As v_ξ is in V_ξ, v_ξ is continuous. Then for all $T \in \tau_\xi$ with $z \in T$, $v_\xi|_T(z) = v_\xi(z)$. Thus

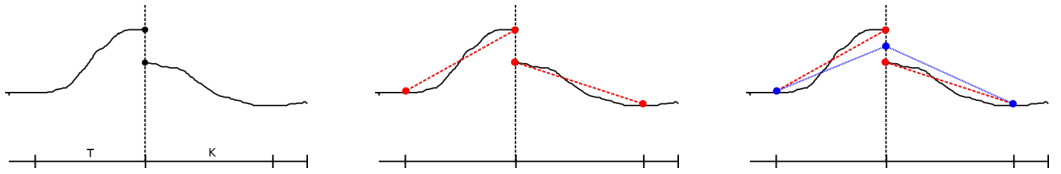
$$\begin{aligned}(E_\xi v_\xi)(z) &= \Big|\{K \in \tau_\xi \,/\, z \in K\}\Big|^{-1} \Big|\{T \in \tau_\xi \,/\, z \in T\}\Big| v_\xi(z) \\ &= v_\xi(z).\end{aligned}$$

□

We can now define the interpolation operator that we will use to create the coarse scale variations space.

Definition 5.1.3 (Quasi-Interpolation operator)**.** *Let V_ξ be piecewise affine finite element spaces, $0 < \xi$. We define the quasi-interpolation operator $\mathcal{I}_\xi : L^2(\Omega) \to V_\xi$ by*

$$\mathcal{I}_\xi = E_\xi \circ \Pi_\xi.$$



(a) A L^2 function v. (b) $\Pi_H \circ v$ in red. (c) $\mathcal{I}_H v = E_H \circ \Pi_H \circ v$ in blue.

Figure 5.2: Illustration of the quasi-interpolation operator.

On the left picture (a) on Figure 5.2, we consider a L^2-function f and two elements of the coarse mesh τ_H : T and K. We apply the local L^2-orthogonal projection Π_H to f to obtain the red function on figure (b). We see that the projection $\Pi_H \circ f$ is piecewise affine and that it coincide on the nodes of the coarse mesh with the original function f. As previously mentioned, the result of the local L^2-orthogonal projection is not always continuous. And it is not the case for this example. That is why we apply the averaging operator, to make the result of Π_H continuous (see figure (c), the blue function). We simply average the value of $\Pi_H \circ f$ on each nodes. This two operators together gives us our quasi-interpolation operator.

Property 5.1.2. *Let V_ξ be piecewise affine finite element spaces, $0 < \xi$. Then the quasi-interpolation operator I_ξ is a projection i.e. if $v_\xi \in V_\xi$, then $I_\xi v_\xi = v_\xi$.*

Proof. Use the fact that Π_ξ is a projection and the property 5.1.1. □

To end this section, we present a useful property of the quasi-interpolation operator to estimate the error.

Property 5.1.3. *Let V_ξ be piecewise affine finite element spaces, $0 < \xi$. For any v in $H^1(\Omega)$, we have*

(a) $\|\nabla \mathcal{I}_\xi v\| + \xi^{-1}\|v - \mathcal{I}_\xi v\| \leq C_I \|\nabla v\|$,

(b) $\|v - \mathcal{I}_\xi v\| \leq C\xi \|v\|_{H^1_0(\Omega)}$.

Proof. Let v be in $H^1(\Omega)$. We start with the following result detailed in [10], lemma 4.1 (b) : for any $T \in \tau_\xi$,

$$\|\nabla \mathcal{I}_\xi v\|_{L^2(T)} + \xi^{-1}\|v - \mathcal{I}_\xi v\|_{L^2(T)} \leq C_I \|\nabla v\|_{L^2(T)},$$

Then since the term $\|\nabla \mathcal{I}_\xi v\| \geq 0$ and by multiplying by ξ, it follows

$$\|v - \mathcal{I}_\xi v\|_{L^2(T)} \leq C_I \xi \|\nabla v\|_{L^2(T)},$$

Now we want an estimation in the norm of $L^2(\Omega)$ instead of $L^2(T)$.

$$\|v - \mathcal{I}_\xi v\|^2 = \sum_{T \in \tau_\xi} \|v - \mathcal{I}_\xi v\|^2_{L^2(T)} \leq C_I^2 \xi^2 \sum_{T \in \tau_\xi} \|\nabla v\|^2_{L^2(T)} = C_I^2 \xi^2 \|v\|^2_{H^1_0(\Omega)}.$$

We have the result. □

5.1.2 Definition of the multiscale space

Now we are able to do the orthogonal decomposition. We begin by giving the definition of the non-orthogonalized fine scale space.

Definition 5.1.4 (Fine scale space)**.** *We define the fine scale space by*

$$V^f := \ker(\mathcal{I}_H) = \{v \in V_h \mid \mathcal{I}_H v = 0\}$$

As said before, this space gives us the fine scale variations of the solution. Now we can split the finite element space.

Property 5.1.4. *We have the following decomposition of V_H :*

$$V_h = V_H \oplus V^f$$

Proof. • Show that $V_h = V_H \oplus V^f$:

We need to show that $V_H \cap V^f = \{0\}$ and that $V_h = V_H + V^f$.

Let $v \in V_H \cap V^f$. We have $v \in V^f$, so

$$\mathcal{I}_H v = 0$$

In addition, v is in V_H, so

$$\mathcal{I}_H v = v$$

Thus

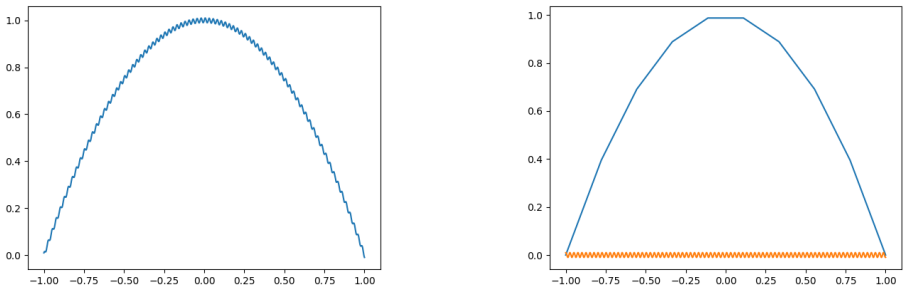
$$v = 0$$

Now let $v_h \in V_h$. We have

$$v_h = \underbrace{\mathcal{I}_H v_h}_{\in V_H} + \underbrace{(1 - \mathcal{I}_H) v_h}_{\in V_f}$$

□

For more clarity, here is a picture to illustrate this decomposition.



(a) (b)

Figure 5.3: Illustration of the non-orthogonalized decomposition.

On the left picture (a) of Figure 5.2, we drew in blue the original function with fine scale variations which lives in V_h. We divide the function into two different ones. The blue one represents the coarse scale variations of the original function and belongs to V_H. The orange one represents the fine scale variations of the original function and belongs to V^{f}.

We managed to decompose the space V_h into two spaces, one for the macro-variations and one for the micro-variations. The orthogonal decomposition is based on the orthogonalization of the decomposition found in the property above with respect the the scalar product a. In order to establish the new decomposition we need first to introduce the corrector associated to our problem. This corrector will allow us to orthogonalize the fine scale space V^{f}.

Definition 5.1.5 (Corrector)**.** *We define $R^f : V_h \to V^f$ as the a-orthogonal projection from V_h into V^f i.e.*

$$a(R^f v, w) = a(v, w), \ \forall v \in V_h, \ \forall w \in V^f.$$

With the corrector we can define the space that will be used later for the orthogonal decomposition. This space is called the multiscale space.

Definition 5.1.6 (Multiscale space)**.** *We define the multiscale space by*

$$V^{ms} := V_H - R^f V_H.$$

This space will encode the "coarse information"of the solution.

Property 5.1.5. *We have the following decomposition of V_h :*

$$V_h = V^{ms} \oplus V^f.$$

Moreover, V^{ms} and V^f are a-orthogonal.

Proof. • Show that $V_h = V^{\mathrm{ms}} \oplus V^f$:

We need to show that $V^{\text{ms}} \cap V^f = \{0\}$ and that $V_h = V^{\text{ms}} + V^f$. Let $v \in V^{\text{ms}} \cap V^f$. Then there exists $v_H \in V_H$, $v = v_H - R^f v_H$.

$$\begin{aligned}\mathcal{I}_H v &= \mathcal{I}_H(v_H - R^f v_H) \\ &= \underbrace{\mathcal{I}_H v_H}_{=v_H} - \mathcal{I}_H R^f v_H \\ &= v_H - \mathcal{I}_H R^f v_H.\end{aligned}$$

However $R^f v_H$ is in V^f i.e. $\mathcal{I}_H R^f v_H = 0$. Then

$$\mathcal{I}_H v = v_H$$

Moreover v is in V^f i.e. $\mathcal{I}_H v = 0$. Thus $v_H = 0$ and so

$$v = 0$$

.

Now let $v_h \in V_h$. We have

$$v_h = \underbrace{\mathcal{I}_H v_h - R^f \mathcal{I}_H v_h}_{\in V^{\text{ms}}} + (1 - \mathcal{I}_H) v_h + R^f \mathcal{I}_H v_h$$

We have that $(1 - \mathcal{I}_H) v_h + R^f \mathcal{I}_H v_h$ is in V^f. Indeed

$$\begin{aligned}\mathcal{I}_H((1 - \mathcal{I}_H) v_h + R^f \mathcal{I}_H v_h) &= 0 + \mathcal{I}_H R^f \mathcal{I}_H v_h \\ &= \mathcal{I}_H \underbrace{R^f \mathcal{I}_H v_h}_{\in V^f} \\ &= 0.\end{aligned}$$

Then

$$v_h = \underbrace{\mathcal{I}_H v_h - R^f \mathcal{I}_H v_h}_{\in V^{\text{ms}}} + \underbrace{(1 - \mathcal{I}_H) v_h + R^f \mathcal{I}_H v_h}_{\in V^f}.$$

- Show that V^{ms} and V^f are a-orthogonal:

Let $v^{\text{ms}} \in V^{\text{ms}}$ and $v^f \in V^f$.

$$\begin{aligned}a(v^{\text{ms}}, v^f) &= a(v_H - R^f v_H, v^f) \\ &= a(v_H, v^f) - \underbrace{a(R^f v_H, v^f)}_{:= a(v_H, v^f)} \\ &= 0.\end{aligned}$$

□

We succeeded in splitting the fine mesh space into two spaces with the property we wanted : the a-orthogonality. Since we have this a-orthogonality it is important to notice that the space V^{ms} will depend on the domain Ω but also on the coefficient A, unlike the finite element space V_h which only depends on the domain. Now we can reformulate the problem.

5.1.3 Formulation

We have the following formulation of our problem

Problem 5.1.1. *For all $t \in (0,T]$, find $u^{ms}(\cdot,t) \in V^{ms}$ such that*

$$\begin{cases} \left(\frac{\partial u^{ms}}{\partial t}, v\right) + a(u^{ms}, v) = (f, v), & \forall v \in V^{ms}(\Omega), \\ u^{ms}(\cdot, 0) = P^{ms} \circ u_0 & . \end{cases} \tag{5.1}$$

Where $P^{\text{ms}} : H_0^1(\Omega) \to V^{\text{ms}}$ is a projection from $H_0^1(\Omega)$ into V^{ms}.

5.1.4 Find a base for the multiscale space

We created our multiscale space. But in practice we need to compute this space. As he have finite dimension, we can compute a finite basis for our space. In this section, we state the relation satisfied by the basis functions. Let $\{\Lambda_x\}_{x \in \mathcal{N}_H}$ be a basis of V^{ms}. Let us define ϕ_x as the projection R^f of the corresponding φ_x i.e.

$$\phi_x := R^f \varphi_x, \ \forall x \in \mathcal{N}_H.$$

We have the following (global) corrector problem

Problem 5.1.2 (Global corrector problem)**.** *For all $x \in \mathcal{N}_H$, find $\phi_x \in V^f$ such that*

$$a(\phi_x, w) = a(\varphi_x, w), \ \forall w \in V^f.$$

and the basis of V^{ms}

$$\{\Lambda_x = \varphi_x - \phi_x \ / \ x \in \mathcal{N}_H\}.$$

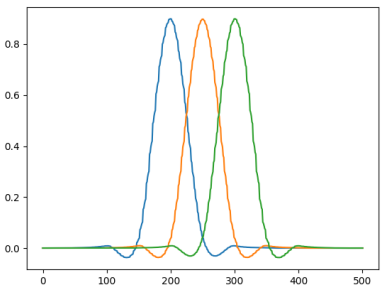


Figure 5.4: Basis functions of the non-localized multiscale space V^{ms}.

The basis function of V^{ms} correspond to the classical basis function of the finite element space φ_x modified by the corrector ϕ_x. The φ_x encodes the coarse variations of the solution and the ϕ_x encodes the fine variations. That is why the space V^{ms} is called multiscale space : it includes both coarse and fine information. The second thing to notice is that the global

corrector problem belongs to the fine scale space V^{f}. Thus the corrector are very expensive to compute. Furthermore, they may have global support. At last, Figure 5.4 gives the Λ for a periodic coefficient A. The Λ are computed in [10] for the one-dimensional case.

5.1.5 Stability

As for the finite element case, we show that the orthogonalized decomposition method is stable.

Theorem 5.1.1 (Stability)**.** *Let u^{ms} be the solution of (5.1). For all $t \in (0,T]$, we have the following results*

$$\|u^{ms}(t)\| \leq C\Big(\|P^{ms} \circ u_0\| + \int_0^t \|f(s)\|\ ds\Big), \tag{5.2}$$

$$\left\|\frac{\partial u^{ms}}{\partial t}(t)\right\| \leq C\Big(\|P^{ms} \circ u_0\| + \|f(t)\| + \int_0^t \|f(s)\|\ ds\Big), \tag{5.3}$$

$$\|u^{ms}(t)\|_a \leq C'\Big(\|P^{ms} \circ u_0\| + \|f(t)\| + \int_0^t \|f(s)\|\ ds\Big). \tag{5.4}$$

Proof. We proceed in the same way as for the stability of u, Theorem 2.2.2. □

5.1.6 Error estimate

Like for the finite element method, we want to estimate the error of the method.

Lemma 5.1.1. *Let us define $\mathcal{E}^{ms} : H_0^1(\Omega) \to V^{ms}$ to be the elliptic projection from $H_0^1(\Omega)$ into V^{ms} i.e. for any $v \in H_0^1(\Omega)$,*

$$a\Big(\mathcal{E}^{ms}v, w^{ms}\Big) = a\Big(v, w^{ms}\Big), \forall w^{ms} \in V^{ms}.$$

Then, for any v in $H_0^1(\Omega)$, we have

$$\|v - \mathcal{E}^{ms}v\| \leq C(\alpha,\beta)H\|v\|_a.$$

Proof. We consider the following auxiliary problem : find z in $H_0^1(\Omega)$ such that

$$a(z,w) = (v - \mathcal{E}^{\mathrm{ms}}v, w),\ \forall w \in H_0^1(\Omega). \tag{5.5}$$

Remplacing w by $v - \mathcal{E}^{\mathrm{ms}}v$ leads to

$$\|v - \mathcal{E}^{\mathrm{ms}}v\|^2 = a(z, v - \mathcal{E}^{\mathrm{ms}}v)$$

Since $\mathcal{E}^{\mathrm{ms}}z$ belongs to V^{ms}, we have that $a(\mathcal{E}^{\mathrm{ms}}z, v - \mathcal{E}^{\mathrm{ms}}v) = 0$ by definition of $\mathcal{E}^{\mathrm{ms}}$. Then

$$\begin{aligned}\|v - \mathcal{E}^{\mathrm{ms}}v\|^2 &= a(z - \mathcal{E}^{\mathrm{ms}}z, v - \mathcal{E}^{\mathrm{ms}}v)\\ &\leq \|z - \mathcal{E}^{\mathrm{ms}}z\|_a\|v - \mathcal{E}^{\mathrm{ms}}v\|_a.\end{aligned}$$

$\|z - \mathcal{E}^{\text{ms}} z\|_a$ is the a-norm of the error due to the orthogonal decomposition approximation of the elliptic problem (5.5). This error estimate has been computed in [8], Lemma 3.1. Hence

$$\|z - \mathcal{E}^{\text{ms}} z\|_a \leq C(\alpha, \beta) H \|v - \mathcal{E}^{\text{ms}} v\|.$$

Then, $\|v - \mathcal{E}^{\text{ms}} v\| \leq C(\alpha, \beta) H \|v - \mathcal{E}^{\text{ms}} v\|_a$. Finally, we have

$$\begin{aligned} \|v - \mathcal{E}^{\text{ms}} v\|_a^2 &= a(v - \mathcal{E}^{\text{ms}} v, v - \mathcal{E}^{\text{ms}} v) \\ &= a(v - \mathcal{E}^{\text{ms}} v, v) - a(v - \mathcal{E}^{\text{ms}} v, \mathcal{E}^{\text{ms}} v) \end{aligned}$$

Since $\mathcal{E}^{\text{ms}} v$ lives in V^{ms} we have $a(v - \mathcal{E}^{\text{ms}} v, \mathcal{E}^{\text{ms}} v) = 0$. Thus

$$\begin{aligned} \|v - \mathcal{E}^{\text{ms}} v\|_a^2 &= a(v - \mathcal{E}^{\text{ms}} v, v) \\ &\leq \|v - \mathcal{E}^{\text{ms}} v\|_a \|v\|_a. \end{aligned}$$

It yields $\|v - \mathcal{E}^{\text{ms}} v\| \leq C(\alpha, \beta) H \|v\|_a$. We have the result.

□

Theorem 5.1.2. *Let u be the solution of (2.2) and u^{ms} be the solution of (5.1). Then for all $t \in (0, T]$, we have*

$$\|u(t) - u^{ms}(t)\| \leq C\Big(\alpha, \beta\Big)\Big(\|u_0 - P^{ms} \circ u_0\| + H\|u(t)\|_a\Big).$$

Proof. The main idea of this proof is the same as for the finite element case. Let us define $\mathcal{E}^{\text{ms}} : H_0^1(\Omega) \to V^{\text{ms}}$ as in Lemma 5.1.1. Thus if $v \in H_0^1(\Omega)$ is the solution of the following elliptic problem

$$a(v, w) = (f, w), \ \forall w \in H_0^1(\Omega)$$

then $\mathcal{E}^{\text{ms}} v \in V_h$ is the solution of

$$a(\mathcal{E}^{\text{ms}} v, w) = (f, w), \ \forall w \in V^{\text{ms}}.$$

We fix $t \in [0, T]$.

$$\|u(t) - u^{\text{ms}}(t)\| \leq \| \underbrace{u(t) - \mathcal{E}^{\text{ms}}[u(t)]}_{=:\epsilon_1(t)} \| + \| \underbrace{\mathcal{E}^{\text{ms}}[u(t)] - u_h(t)}_{=:\epsilon_2(t)} \|.$$

- ϵ_1 : By using Lemma 5.1.1, we have

$$\|\epsilon_1(t)\| = \|u(t) - \mathcal{E}^{\text{ms}}[u(t)]\| \leq C(\alpha, \beta) H \|u(t)\|_a.$$

- ϵ_2 : We put $\epsilon_2(t)$ instead of $u^{\text{ms}}(t)$ in Problem 5.1

$$\left(\frac{\partial \epsilon_2(t)}{\partial t}, v\right) + a(\epsilon_2(t), v) = \left(\frac{\partial \mathcal{E}^{\text{ms}}[u(t)]}{\partial t}, v\right) + a(\mathcal{E}^{\text{ms}}[u(t)], v) - \Bigg(\underbrace{\Big(\frac{\partial u^{\text{ms}}(t)}{\partial t}, v\Big) + a(u^{\text{ms}}(t), v)}_{=(f(t), v)} \Bigg),$$

since u^{ms} solve Problem 5.1. Furthermore $\mathcal{E}^{\text{ms}}[u(t)]$ satisfies the Galerkin orthogonality i.e. $a(\mathcal{E}^{\text{ms}}[u(t)], w) = a(u(t), w)$ for any $w \in V^{\text{ms}}$

$$\begin{aligned}\left(\frac{\partial \epsilon_2(t)}{\partial t}, v\right) + a(\epsilon_2(t), v) &= \left(\frac{\partial \mathcal{E}^{\text{ms}}[u(t)]}{\partial t}, v\right) + a(u(t), v) - (f(t), v) \\ &= \left(\frac{\partial \mathcal{E}^{\text{ms}}[u(t)]}{\partial t}, v\right) - \left(\frac{\partial u(t)}{\partial t}, v\right) \\ &= -\left(\frac{\partial \epsilon_1(t)}{\partial t}, v\right)\end{aligned}$$

Thus ϵ_2 solve our parabolic problem 5.1 for $f = -\frac{\partial \epsilon_1}{\partial t}$ and $u_0 = \epsilon_2(0)$. We can apply the stability Theorem 5.1.1 and get the estimate

$$\|\epsilon_2(t)\| \leq C\Big(\|\epsilon_2(0)\| + \int_0^t \left\|\frac{\partial \epsilon_1}{\partial t}(s)\right\| \, ds\Big) = C\Big(\|\epsilon_2(0)\| + \|\epsilon_1(t)\| - \|\epsilon_1(0)\|\Big).$$

Here

$$\|\epsilon_2(0)\| = \|\mathcal{E}^{\text{ms}} u_0 - P^{\text{ms}} \circ u_0\| \leq \|\mathcal{E}^{\text{ms}} u_0 - u_0\| + \|u_0 - P^{\text{ms}} \circ u_0\| = \|\epsilon_1(0)\| + \|u_0 - P^{\text{ms}} \circ u_0\|.$$

Combining with the estimation of $\|\epsilon_1(t)\|$

$$\begin{aligned}\|\epsilon_2(t)\| &\leq C\Big(\|\epsilon_2(0)\| + \|\epsilon_1(t)\| - \|\epsilon_1(0)\|\Big) \\ &\leq C\Big(\|\epsilon_1(0)\| + \|u_0 - P^{\text{ms}} \circ u_0\| + \|\epsilon_1(t)\| - \|\epsilon_1(0)\|\Big) \\ &\leq C\Big(\|u_0 - P^{\text{ms}} \circ u_0\| + \|\epsilon_1(t)\|\Big) \\ &\leq C\Big(\alpha, \beta\Big)\Big(\|u_0 - P^{\text{ms}} \circ u_0\| + H\|u(t)\|_a\Big).\end{aligned}$$

□

Note 4. Again, if you use the elliptic projection $\mathcal{E}^{\text{ms}}$ instead of P^{ms}, we get

$$\|u(t) - u^{\text{ms}}(t)\| \leq C\Big(\alpha, \beta\Big) H\Big(\|u_0\|_a + \|u(t)\|_a\Big).$$

5.2 Localization

In the last section we split our space of solution into two a-orthogonal spaces. This is the orthogonal decomposition part of the method. However, to compute the basis of the multi-scale space V^{ms}, we need to solve the global corrector problems 5.1.2, which are posed in the fine scale space V^{f}. As a result they are computationally expensive to solve. Also, the correctors ϕ_x generally have global support. Nevertheless, it is proved in [10] that the ϕ_x decays exponentially fast away from x. This last property motivates a localization of the corrector problems. The purpose of the next steps is to truncate every basis functions of the space V^{ms} into a smaller patches of the coarse mesh. Doing this gives us a new multi-scale space, a localized one, and as a result a new approximation of the solution.

5.2.1 Definitions

In order to correctly define the localized multi-scale space, we need to introduce a few notions to define the localized version of the corrector operator R^{f} from the last section.

Definition 5.2.1. *For all $T \in \tau_H$ and for $k \in \mathbb{N}$, we define*

$$V^f\big(\omega_k(T)\big) := \{w \in V^f \mid Supp(w) \subset \omega_k(T)\}.$$

Definition 5.2.2. *For all $T \in \tau_H$, we define the projection $R_T^f : V_h \to V^f$ by*

$$\int_\Omega (A\nabla R_T^f u) \cdot \nabla v \; dx = \int_T (A\nabla u) \cdot \nabla v \; dx, \;\; \forall u \in V_h, \forall v \in V^f.$$

Note 5. Note that $R^f = \sum_{T\in\tau_H} R_T^f$. Indeed, let u in V_h.

$$\begin{aligned} a\Big(R^f u - \sum_{T\in\tau_H} R_T^f u, v\Big) &= a(R^f u, v) - \sum_{T\in\tau_H} a(R_T^f u, v) \\ &= a(u,v) - \underbrace{\sum_{T\in\tau_H} \int_T (A\nabla u) \cdot \nabla v \; dx}_{=a(u,v)} \\ &= 0, \;\; \forall v \in V_h \end{aligned}$$

Now we localize the operator R_T^f.

Definition 5.2.3. *For all $T \in \tau_H$ and for $k \in \mathbb{N}$, we define the projection $R_{T,k}^f : V_h \to V^f\big(\omega_k(T)\big)$ by*

$$\int_{\omega_k(T)} (A\nabla R_{T,k}^f u) \cdot \nabla v \; dx = \int_T (A\nabla u) \cdot \nabla v \; dx, \;\; \forall u \in V_h, \forall v \in V^f\big(\omega_k(T)\big).$$

Now we are able to define our new multi scale space.

Definition 5.2.4 (Localized multiscale space)**.** *For $k \in \mathbb{N}$, we define V_k^{ms} to be the localized multiscale space by*

$$V_k^{ms} := V_H - R_k^f V_H,$$

where $R_k^f := \sum_{T\in\tau_H} R_{T,k}^f$.

Note 6. We have the a-orthogonality between V_k^{ms} and $V^{\mathrm{f}}(\omega_k(\Omega))$. Indeed, let $v \in V_k^{\mathrm{ms}}$ and $w \in V^{\mathrm{f}}(\omega_k(\Omega))$. By definition of V_k^{ms}, there exists $v_H \in V_H$ such that $v = v_H - R_k^{\mathrm{f}} v_H$. Then

$$\begin{aligned} a(v,w) &= a(v_H, w) - a(R_k^{\mathrm{f}} v_H, w) \\ &= a(v_H, w) - \sum_T a(R_{T,k}^{\mathrm{f}} v_H, w) \\ &= a(v_H, w) - \sum_T \int_\Omega (A\nabla R_{T,k}^{\mathrm{f}} v_H) \cdot w \; dx. \end{aligned}$$

Since $\operatorname{Supp} w \subset \omega_k(\Omega)$, we have

$$\begin{aligned} a(v,w) &= a(v_H,w) - \sum_T \int_{\omega_k(\Omega)} (A\nabla R^{\mathrm{f}}_{T,k} v_H)\cdot w \; dx \quad = a(v_H,w) - \sum_T \int_T (A\nabla v_H)\cdot w \; dx \\ &= a(v_H,w) - a(v_H,w) \\ &= 0. \end{aligned}$$

5.2.2 Formulation

With this localized multi-scale space, we obtain the final localized orthogonal decomposition method formulation

Problem 5.2.1. *For a given k and for all $t \in (0,T]$, find $u_k^{ms}(\cdot,t) \in V_k^{ms}$ such that*

$$\begin{cases} \left(\frac{\partial u_k^{ms}}{\partial t}, v\right) + a(u_k^{ms}, v) = (f,v), & \forall v \in V_k^{ms}(\Omega), \\ u_k^{ms}(\cdot,0) = P_k^{ms} \circ u_0 & . \end{cases} \tag{5.6}$$

Where $P_k^{\mathrm{ms}} : H_0^1(\Omega) \to V_k^{\mathrm{ms}}$ is a projection from $H_0^1(\Omega)$ into V_k^{ms}.

5.2.3 Find a basis for the localized multiscale space

Let $\{\Lambda_{k,x}\}_{x\in\mathcal{N}_H}$ be a basis of V_k^{ms}. Let us define $\phi_{k,x}$ as the projection R_k^f of the corresponding φ_x i.e.

$$\phi_{k,x} := R_k^f \varphi_x = \sum_{T\in\tau_H} R_{T,k}^f \varphi_x, \; \forall x \in \mathcal{N}_H.$$

Note that $\{\phi_{k,x}\}_{x\in\mathcal{N}_H}$ is the localized version of $\{\phi_x\}_{x\in\mathcal{N}_H}$. We have the following localized cell problem

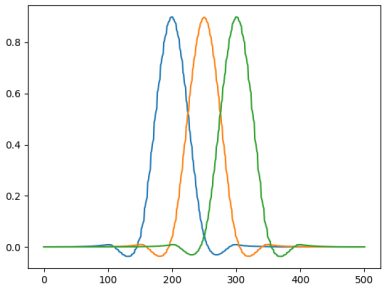
Problem 5.2.2 (Localized cell problem)**.** *For all $x \in \mathcal{N}_H$, find $\phi_{x,T,k} \in V^f\big(\omega_k(T)\big)$ such that*

$$\int_{\omega_k(T)} (A\nabla\phi_{x,T,k})\cdot\nabla v \; dx = \int_T (A\nabla u)\cdot\nabla v \; dx,$$

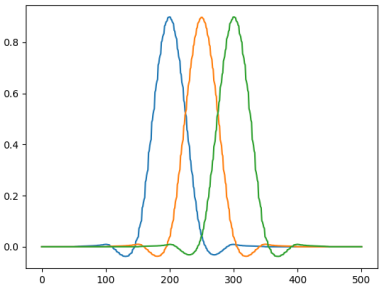
and the basis of V_k^{ms}

$$\{\Lambda_{k,x} = \varphi_x - \phi_{k,x} \;/\; x \in \mathcal{N}_H\},$$

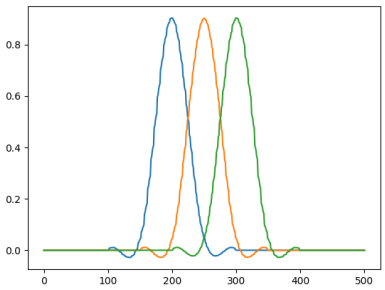
with $\phi_{k,x} := \sum_{T\in\tau_H} \phi_{x,T,k}$.



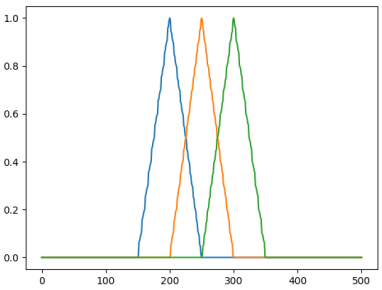
(a) $k = 3$.



(b) $k = 2$.



(c) $k = 1$.



(d) $k = 0$.

Figure 5.5: Influence of the patches size k on the basis functions of the localized multi-scale space V^{ms}.

The basis functions of V_k^{ms} have essentially the same shape as the ones of V^{ms} for a large k. Indeed, we truncate the basis functions φ_x outside a large patch, and so far away from x. As these functions decays exponentially fast away from x, the localization have very few effect on the basis functions (see Figure 5.5 (a) and (b)). It leads to a low error between the solution of the orthogonal decomposition method and the solution of the localized orthogonal method. However, for a small k, the localization will affect the basis functions. In Figure 5.5 (c) and (d), we clearly see this truncation outside the patches of x. Therefore we expect a high error between the solution of the orthogonal decomposition method and the solution of the localized orthogonal decomposition method.

5.2.4 Error estimate

Now, we have to estimate the error due to the localization.

Theorem 5.2.1 (Error due to the localization)**.** *Let u^{ms} be the solution of* (5.1) *and let u_k^{ms} be the solution of* (5.6)*. Then there exists a constant $0 < \mu < 1$ that depends on the contrast β/α such that*

$$\|u^{ms}(t) - u_k^{ms}(t)\| \lesssim k^{d/2}\mu^k + k^d\mu^{2k}H^{-1}.$$

Hence, the choice $k \approx \log H$ recovers the convergence rate of the non localized method.

Proof. We proceed as in [7]. First, we split the error in two parts.

$$\begin{aligned}\|u^{\mathrm{ms}}(t) - u_k^{\mathrm{ms}}(t)\| &= \|(1-R^{\mathrm{f}})v_H^1 - (1-R_k^{\mathrm{f}})v_h^2\| \\ &= \|(1-R^{\mathrm{f}})v_H^1 - (1-R^{\mathrm{f}}+R^{\mathrm{f}}-R_k^{\mathrm{f}})v_h^2\| \\ &\leq \underbrace{\|(1-R^{\mathrm{f}})(v_H^1 - v_H^2)\|}_{=:\epsilon_1(t)} + \underbrace{\|(R^{\mathrm{f}}-R_k^{\mathrm{f}})v_H^2\|}_{=:\epsilon_2(t)}.\end{aligned}$$

- $\epsilon_1(t)$: Let w_H be in V_H. We have

$$\begin{aligned}&\left(\frac{\partial}{\partial t}\big((1-R^{\mathrm{f}})(v_H^1 - v_H^2)\big), (1-R^{\mathrm{f}})w_H\right) + a\big((1-R^{\mathrm{f}})(v_H^1 - v_H^2), (1-R^{\mathrm{f}})w_H\big) \\ &= \left(\frac{\partial}{\partial t}\big((1-R^{\mathrm{f}})v_H^1\big), (1-R^{\mathrm{f}})w_H\right) + a\big((1-R^{\mathrm{f}})v_H^1, (1-R^{\mathrm{f}})w_H\big) - \left(\frac{\partial}{\partial t}\big((1-R^{\mathrm{f}})v_H^2\big), (1-R^{\mathrm{f}})w_H\right) \\ &\quad - a\big((1-R^{\mathrm{f}})v_H^2, (1-R^{\mathrm{f}})w_H\big)\end{aligned}$$

Since $(1-R^{\mathrm{f}})w_H$ is in V^{ms} and since $(1-R^{\mathrm{f}})v_H^1$ satisfies (5.1), we have

$$\begin{aligned}&= \Big(f, (1-R^{\mathrm{f}})w_H\Big) - \left(\frac{\partial}{\partial t}\big((1-R^{\mathrm{f}})v_H^2\big), (1-R^{\mathrm{f}})w_H\right) - a\big((1-R^{\mathrm{f}})v_H^2, (1-R^{\mathrm{f}})w_H\big) \\ &= \Big(f, (1-R^{\mathrm{f}})w_H\Big) - \left(\frac{\partial}{\partial t}\big((1-R_k^{\mathrm{f}})v_H^2\big), (1-R^{\mathrm{f}})w_H\right) - a\big((1-R_k^{\mathrm{f}})v_H^2, (1-R^{\mathrm{f}})w_H\big) \\ &\quad - \left(\frac{\partial}{\partial t}\big((R_k^{\mathrm{f}}-R^{\mathrm{f}})v_H^2\big), (1-R^{\mathrm{f}})w_H\right) - a\big((R_k^{\mathrm{f}}-R^{\mathrm{f}})v_H^2, (1-R^{\mathrm{f}})w_H\big)\end{aligned}$$

$$\begin{aligned}&= \Big(f, (1-R^{\mathrm{f}})w_H\Big) - \left(\frac{\partial}{\partial t}\big((1-R_k^{\mathrm{f}})v_H^2\big), (1-R_k^{\mathrm{f}})w_H\right) \\ &\qquad - \left(\frac{\partial}{\partial t}\big((1-R_k^{\mathrm{f}})v_H^2\big), (R_k^{\mathrm{f}}-R^{\mathrm{f}})w_H\right) - a\big((1-R_k^{\mathrm{f}})v_H^2, (1-R_k^{\mathrm{f}})w_H\big) \\ &\qquad - a\big((1-R_k^{\mathrm{f}})v_H^2, (R_k^{\mathrm{f}}-R^{\mathrm{f}})w_H\big) - \left(\frac{\partial}{\partial t}\big((R_k^{\mathrm{f}}-R^{\mathrm{f}})v_H^2\big), (1-R^{\mathrm{f}})w_H\right) \\ &\qquad\qquad - a\big((R_k^{\mathrm{f}}-R^{\mathrm{f}})v_H^2, (1-R^{\mathrm{f}})w_H\big)\end{aligned}$$

Since $(1-R_k^{\mathrm{f}})w_H^2$ is in V_k^{ms} and since $(1-R_k^{\mathrm{f}})v_H^2$ satisfies (5.6), we have

$$\begin{aligned}&= \Big(-f, (R^{\mathrm{f}}-R_k^{\mathrm{f}})w_H\Big) - \left(\frac{\partial}{\partial t}\big((1-R_k^{\mathrm{f}})v_H^2\big), (R_k^{\mathrm{f}}-R^{\mathrm{f}})w_H\right) \\ &\qquad - \left(\frac{\partial}{\partial t}\big((R_k^{\mathrm{f}}-R^{\mathrm{f}})v_H^2\big), (1-R^{\mathrm{f}})w_H\right) - a\big((1-R_k^{\mathrm{f}})v_H^2, (R_k^{\mathrm{f}}-R^{\mathrm{f}})w_H\big) \\ &\qquad\qquad - a\big((R_k^{\mathrm{f}}-R^{\mathrm{f}})v_H^2, (1-R^{\mathrm{f}})w_H\big)\end{aligned}$$

Moreover,

$$
\begin{aligned}
&a\big((1-R_k^{\mathrm{f}})v_H^2,(R^{\mathrm{f}}-R_k^{\mathrm{f}})w_H\big)+a\big((R^{\mathrm{f}}-R_k^{\mathrm{f}})v_H^2,(1-R^{\mathrm{f}})w_H\big)\\
&=a\big((1-R_k^{\mathrm{f}})v_H^2,(1-R_k^{\mathrm{f}})w_H\big)-a\big((1-R_k^{\mathrm{f}})v_H^2,(1-R^{\mathrm{f}})w_H\big)\\
&\qquad+a\big((1-R_k^{\mathrm{f}})v_H^2,(1-R^{\mathrm{f}})w_H\big)-a\big((1-R^{\mathrm{f}})v_H^2,(1-R^{\mathrm{f}})w_H\big)\\
&=a\big((1-R_k^{\mathrm{f}})v_H^2,(1-R_k^{\mathrm{f}})w_H\big)-a\big((1-R^{\mathrm{f}})v_H^2,(1-R^{\mathrm{f}})w_H\big)\\
&=a\big(v_H^2,w_H\big)-a\big(v_H^2,R_k^{\mathrm{f}}w_H\big)-a\big(R_k^{\mathrm{f}}v_H^2,w_H\big)+a\big(R_k^{\mathrm{f}}v_H^2,R_k^{\mathrm{f}}w_H\big)\\
&\qquad-a\big(v_H^2,w_H\big)+a\big(v_H^2,R^{\mathrm{f}}w_H\big)+a\big(R^{\mathrm{f}}v_H^2,w_H\big)-a\big(R^{\mathrm{f}}v_H^2,R^{\mathrm{f}}w_H\big)
\end{aligned}
$$

Since $a\big(v_H^2,R^{\mathrm{f}}w_H\big)=a\big(R^{\mathrm{f}}v_H^2,R^{\mathrm{f}}w_H\big)$ and $a\big(v_H^2,R_k^{\mathrm{f}}w_H\big)=a\big(R_k^{\mathrm{f}}v_H^2,R_k^{\mathrm{f}}w_H\big)$, we have

$$
\begin{aligned}
&a\big((1-R_k^{\mathrm{f}})v_H^2,(R^{\mathrm{f}}-R_k^{\mathrm{f}})w_H\big)+a\big((R^{\mathrm{f}}-R_k^{\mathrm{f}})v_H^2,(1-R^{\mathrm{f}})w_H\big)\\
&=a\big((R^{\mathrm{f}}-R_k^{\mathrm{f}})v_H^2,w_H\big).
\end{aligned}
$$

Lastly, since $R_k^{\mathrm{f}}v_H^2\in V^{\mathrm{f}}\subset V^{\mathrm{f}}$, we have

$$
a\big((R^{\mathrm{f}}-R_k^{\mathrm{f}})v_H^2,w_H\big)=a\big((R^{\mathrm{f}}-R_k^{\mathrm{f}})v_H^2,R^{\mathrm{f}}w_H\big).
$$

Then, using that $a\big((R^{\mathrm{f}}-R_k^{\mathrm{f}})v_H^2,R_k^{\mathrm{f}}w_H\big)=0$, it follows

$$
a\big((R^{\mathrm{f}}-R_k^{\mathrm{f}})v_H^2,w_H\big)=a\big((R^{\mathrm{f}}-R_k^{\mathrm{f}})v_H^2,(R^{\mathrm{f}}-R_k^{\mathrm{f}})w_H\big).
$$

We set $F((1-R^{\mathrm{f}})w_H):=\big(-f,(R^{\mathrm{f}}-R_k^{\mathrm{f}})w_H\big)+\big(\frac{\partial}{\partial t}\big((1-R_k^{\mathrm{f}})v_H^2\big),(R^{\mathrm{f}}-R_k^{\mathrm{f}})w_H\big)$ $+\big(\frac{\partial}{\partial t}\big((R^{\mathrm{f}}-R_k^{\mathrm{f}})v_H^2\big),(1-R^{\mathrm{f}})w_H\big)+a\big((R^{\mathrm{f}}-R_k^{\mathrm{f}})v_H^2,(R^{\mathrm{f}}-R_k^{\mathrm{f}})w_H\big)$. Thus $(1-R^{\mathrm{f}})(v_H^1-v_H^2)$ satisfies (5.1) with F at the right hand side.

$$
\begin{aligned}
\frac{\|F((1-R^{\mathrm{f}})w_H)\|}{\|(1-R^{\mathrm{f}})w_H\|}\le\|f\|\frac{\|(R^{\mathrm{f}}-R_k^{\mathrm{f}})w_H\|}{\|(1-R^{\mathrm{f}})w_H\|}&+\left\|\frac{\partial}{\partial t}\big((1-R_k^{\mathrm{f}})v_H^2\big)\right\|\frac{\|(R^{\mathrm{f}}-R_k^{\mathrm{f}})w_H\|}{\|(1-R^{\mathrm{f}})w_H\|}\\
&+\left\|\frac{\partial}{\partial t}\big((R^{\mathrm{f}}-R_k^{\mathrm{f}})v_H^2\big)\right\|+\|(R^{\mathrm{f}}-R_k^{\mathrm{f}})v_H^2\|_a\frac{\|(R^{\mathrm{f}}-R_k^{\mathrm{f}})w_H\|_a}{\|(1-R^{\mathrm{f}})w_H\|}.
\end{aligned}
$$

We use the following inequality of which the proof can be found in [6] and the inverse inequality

$$
\|(R^{\mathrm{f}}-R_k^{\mathrm{f}})w_H\|_a\le Ck^{d/2}\mu^k\|w_H\|_a.
$$

This inequality combining with properties of $\mathcal{I}_H$ (Property 5.1.3 (a)) and with the inverse inequality leads to

$$
\|(R^{\mathrm{f}}-R_k^{\mathrm{f}})w_H\|\le CC_Ik^{d/2}\mu^kH\|w_H\|_a\le CC_IC_{\mathrm{inv}}k^{d/2}\mu^k\|w_H\|.
$$

Thus and since $\|w_H\|=\|\mathcal{I}_H((1-R^{\mathrm{f}})w_H)\|\le C_I\|(1-R^{\mathrm{f}})w_H\|$, we have

$$
\frac{\|F((1-R^{\mathrm{f}})w_H)\|}{\|(1-R^{\mathrm{f}})w_H\|}\le Ck^{d/2}\mu^k\left(\|f\|+\left\|\frac{\partial}{\partial t}\big((1-R_k^{\mathrm{f}})v_H^2\big)\right\|+\left\|\frac{\partial}{\partial t}v_H^2\right\|+Ck^{d/2}\mu^kH^{-1}\|v_H^2\|_a\right).
$$

It implies that,

$$\sup_{w^{\mathrm{ms}}\in V^{\mathrm{ms}}} \frac{\|F(w^{\mathrm{ms}})\|}{\|w^{\mathrm{ms}}\|} \lesssim k^{d/2}\mu^k + k^d\mu^{2k}H^{-1},$$

and also F is in $L^2(\Omega)$. We can apply the stability theorem on $(1-R^{\mathrm{f}})(v_H^1 - v_H^2)$ and it leads to $\|(1-R^{\mathrm{f}})(v_H^1 - v_H^2)\| \lesssim k^{d/2}\mu^k + k^d\mu^{2k}H^{-1}$.

- $\epsilon_2(t)$: We use again the following inequality of which the proof can be found in [6]

$$\|(R^{\mathrm{f}} - R_k^{\mathrm{f}})v_H^2\| \leq \alpha^{-1/2}\|(R^{\mathrm{f}} - R_k^{\mathrm{f}})v_H^2\|_a \leq C(\alpha)k^{d/2}\mu^k\|v_H^2\|_a.$$

□

5.3 Explicit Euler scheme approximation

We apply the forward Euler method to (5.6).

Problem 5.3.1. *Knowing $u_k^{\bar{ms}}|^n$, find $u_k^{\bar{ms}}|^{n+1} \in V_k^{ms}$ such that*

$$\begin{cases} \big(u_k^{\bar{ms}}|^{n+1}, v\big) & = \big(u_k^{\bar{ms}}|^n, v\big) + \Delta t\Big((f|^n, v) - a(u_k^{\bar{ms}}|^n, v)\Big),\ \forall v \in V_k^{ms}, \\ u_k^{\bar{ms}}|^0 = P_k^{ms} \circ u_0 & . \end{cases} \tag{5.7}$$

5.3.1 Error of the full method

Theorem 5.3.1. *Let u be the solution of (2.2) and let u_k^{ms} be the solution of (5.7) such that u_k^{ms} is C^2 in time. Then there exists a constant $0 < \mu < 1$ that depends on the contrast β/α such that*

$$\|u|^n - \bar{u}_k^{ms}|^n\| \lesssim \|u_0 - P_k^{ms} \circ u_0\| + k^{d/2}\mu^k + k^d\mu^{2k}H^{-1} + H + \Delta t.$$

Hence, the choice $k \approx \log H$ gives

$$\|u|^n - \bar{u}_k^{ms}|^n\| \lesssim \|u_0 - P_k^{ms} \circ u_0\| + H + \Delta t.$$

Proof. Use the triangle inequality, Theorem 4.2.3, Theorem 5.1.2 and Theorem 5.2.1. □

5.3.2 Matrix assembly

By doing the same kind of computation as in 4.2.5, we obtain

Problem 5.3.2. *Knowing $\bar{U}_k^{ms}|^n$, find $\bar{U}_k^{ms}|^{n+1}$ such that*

$$\begin{cases} \mathbb{M}_k^{ms}\bar{U}_k^{ms}|^{n+1} = (\mathbb{M}_k^{ms} - \Delta t\mathbb{S}_k^{ms})\bar{U}_k^{ms}|^n + \Delta t\mathbb{F}_k|^n, \\ \bar{U}_k^{ms}|^0 = \big((P_k^{ms} \circ u_0)|_0, \ldots, (P_k^{ms} \circ u_0)|_N\big)^T. \end{cases} \tag{5.8}$$

where

- $\mathbb{M}_k^{\mathrm{ms}} = [m_{ij}]_{(i,j)\in\mathcal{N}_H^2}$, $m_{ij} = \int_\Omega \Lambda_{k,i}\Lambda_{k,j}\,dx$, is the multi-scale mass matrix,

- $\mathbb{S}_k^{\text{ms}} = [a_{ij}]_{(i,j)\in\mathcal{N}_H^2}$, $a_{ij} = \int_\Omega (A\nabla\Lambda_{k,i})\cdot\nabla\Lambda_{k,j}\ dx$, is the multi-scale stiffness matrix,
- $\bar{U}_k^{\text{ms}}|^n = (\bar{u}^{\text{ms}}|_{k,0}^n, \dots, \bar{u}^{\text{ms}}|_{k,N}^n)^T$, where $N = |\mathcal{N}_H|$,
- $\mathbb{F}_k|^n = [F_i]_{i\in\mathcal{N}_H}$, $F_i = \int_\Omega f|^n\Lambda_{k,i}\ dx$.

5.3.3 Stability and the Courant–Friedrichs–Lewy condition

Like for the finite element case, using the forward Euler method implies a conditional stability. In this section, we formulate this condition, which is in fact, closed to the one for the finite element method.

Theorem 5.3.2 (Stability)**.** *If we suppose*

$$1 - C_{inv}\beta H^{-2}\frac{\Delta t}{2} \geq 0 \tag{5.9}$$

then the scheme (5.7) *is stable i.e. if* $\bar{u}_k^{ms}$ *is the solution of* (5.7)*, then for any* n *we have*

$$\|\bar{u}_k^{ms}|^{n+1}\| \leq \|P_k^{ms}\circ u_0\| + 2\Delta t\sum_{j=0}^{n}\|f|^j\|.$$

Proof. Similar to the proof for the finite element case, Theorem 4.2.2. □

We see once again that this condition is very restrictive on Δt. As for the finite element case, this condition is even worst for large β. However, the fine mesh have no influence on this condition. Only the coarse mesh is important here. Since $h \ll H$, we can conclude that the complete method using the localized orthogonal decomposition coupled with the forward Euler method is better than the complete method using the finite element method.

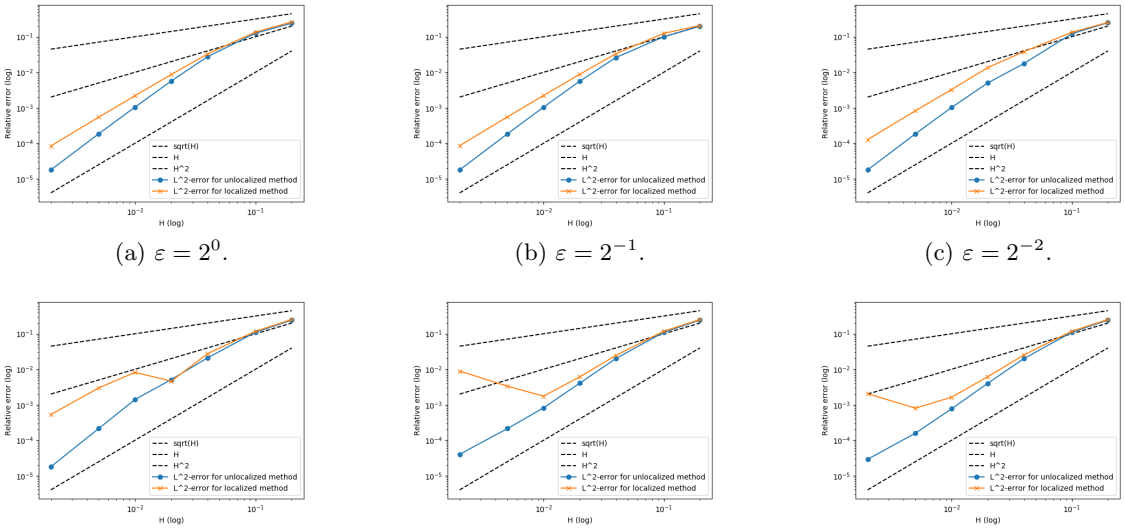
5.3.4 Implementation in Python

See Appendix B and Appendix D.1.

5.3.5 Numerical results

Error

We check the order of the convergence rate in space of the (localized) orthogonal decomposition method. We take the diffusion coefficient $A(x) = (2+\cos(\frac{2\pi}{\epsilon}x))^{-1}$, for $0 < \epsilon \leq 1$. Moreover, we have $u_0 = 0$ and $f = 1$. On Figure 5.6 we draw the relative L^2-error $\|u - u^{\text{ms}}\|/\|u\|$ depending on the coarse mesh size H in orange and the relative L^2-error $\|u - u_k^{\text{ms}}\|/\|u\|$, $k = 0$, depending on the coarse mesh size H in blue. We fix Δt in such way that the condition (5.9) is satisfied for every H.



(a) $\varepsilon = 2^0$. (b) $\varepsilon = 2^{-1}$. (c) $\varepsilon = 2^{-2}$.

(d) $\varepsilon = 2^{-3}$. (e) $\varepsilon = 2^{-4}$. (f) $\varepsilon = 2^{-5}$.

Figure 5.6: L^2-error with respect to the coarse mesh size H

5.4 Super-time Stepping approximation

Here, we apply the Super-time Stepping acceleration to the localized orthogonal decomposition method.

5.4.1 Formulation

With the same reasoning used in the finite element method case, we get the following formulation

Problem 5.4.1. *Knowing $\bar{u}_k^{ms}|^n$, find $\bar{u}_k^{ms}|^{n+1} \in V_k^{ms}$ such that*

$$\begin{cases} \left(\bar{u}_k^{ms}|^{n+1/N}, v_k^{ms}\right) &= \left(\bar{u}_k^{ms}|^{n}, v_k^{ms}\right) &+\tau_1\Big((f|^n, v_k^{ms}) - a(\bar{u}_k^{ms}|^{n}, v_k^{ms})\Big), &\forall v_k^{ms} \in V_k^{ms}, \\ \left(\bar{u}_k^{ms}|^{n+2/N}, v_k^{ms}\right) &= \left(\bar{u}_k^{ms}|^{n+1/N}, v_k^{ms}\right) &+\tau_2\Big((f|^n, v_k^{ms}) - a(\bar{u}_k^{ms}|^{n+1/N}, v_k^{ms})\Big), &\forall v_k^{ms} \in V_k^{ms}, \\ &\vdots & & \\ \left(\bar{u}_k^{ms}|^{n+1}, v_k^{ms}\right) &= \left(\bar{u}_k^{ms}|^{n+(N-1)/N}, v_k^{ms}\right) &+\tau_N\Big((f|^n, v_k^{ms}) - a(\bar{u}_k^{ms}|^{n+(N-1)/N}, v_k^{ms})\Big), &\forall v_k^{ms} \in V_k^{ms}. \end{cases} \tag{5.10}$$

and such that $\bar{u}_k^{ms}|^0 = P_k^{ms} \circ u_0$

5.4.2 Stability

Same as for the finite element method case.

5.4.3 Error of the full method

We saw in Theorem 4.3.1 that the order of convergence is 1 with respect to ΔT. It leads to the following error of the completely discretized method.

Theorem 5.4.1 (Error of the full method)**.** *Let u be the solution of (2.2) and $\bar{u}_k^{ms}$ be the solution of (5.10). Then*

$$\|u|^n - \bar{u}_k^{ms}|^n\| \lesssim \|u_0 - P_k^{ms} \circ u_0\| + k^{d/2}\mu^k + k^d \mu^{2k} H^{-1} + H + \Delta T.$$

Hence, the choice $k \approx \log H$ gives

$$\|u|^n - \bar{u}_k^{ms}|^n\| \lesssim \|u_0 - P_k^{ms} \circ u_0\| + H + \Delta T.$$

Proof. Use the triangle inequality, theorem 4.3.1 and 4.1.3. □

5.4.4 Implementation in Python

See appendix B and appendix D.2.

5.4.5 Numerical results

Lastly, we compare the speed of all the different methods we used so far.

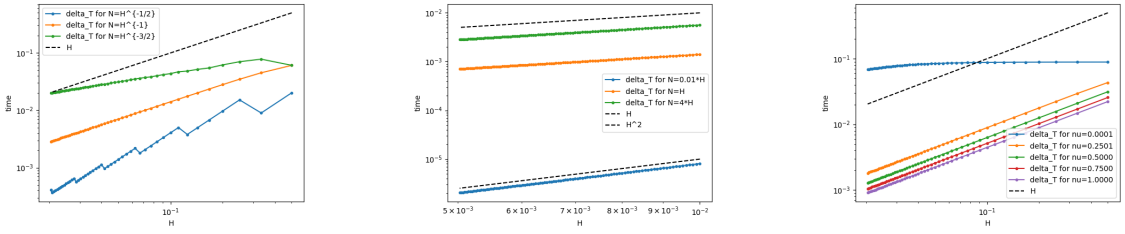
Balance the error convergence rate

The aim of this section is to try to have a convergence rate of the error due to the completly discretized methods of order one with respect to the mesh size (h for the finite element method and H for the localized orthogonal decomposition). First, let us remind the reader that the convergence rate of the error due to the finite element/Euler method is $\mathcal{O}(h + \Delta t)$ and that the convergence rate of the error due to the localized orthogonal decomposition/Euler method is $\mathcal{O}(H + \Delta t)$. Hence, we would like that $\Delta t \leq Ch$ (resp. $\Delta t \leq CH$) with C independent of h (resp. H). However, since the Courant–Friedrichs–Lewy condition have to be statisfied and since the mesh size is lower than 1, the condition $\Delta t \leq Ch$ (resp. $\Delta t \leq CH$) is impossible.

However, by using the Super-time Stepping instead of the forward Euler method, balancing the convergence rate due to the approximation must be possible. For instance, for the localized orthogonal decomposition we have (see [1])

$$\Delta T = \Delta t_{\text{expl}} \frac{N}{2\sqrt{\nu}} \left(\frac{(1+\sqrt{\nu})^{2N} - (1-\sqrt{\nu})^{2N}}{(1+\sqrt{\nu})^{2N} + (1-\sqrt{\nu})^{2N}} \right), \text{ where } \Delta t_{\text{expl}} := 2C_{\text{inv}}^{-1}\beta^{-1}H.$$

We did some experiments to know how to balance the error convergence rate. On Figure 5.7, we drew the Super-time step ΔT with respect to the mesh size H.



(a) (b) (c)

Figure 5.7: Balance the error convergence rate.

On Figure 5.7 (a), we fixed $\nu = 0.1$ and $N = H^a$ with $a \in \{-1/2, -1, -3/2\}$. We see that the orange curve ($N = H^{-1}$) follows the H one. It indicates us that we may have $\Delta T(N, \nu) \leq CH$ if $N = C'H^{-1}$.

On Figure (b), we fixed $\nu = 0.1$ and $N = CH$, for diffents C. It seems that the blue curve ($C = 0.01$) follows the H^2 curve while the others follows the H one. Manifestaly, in order to have $\Delta T \leq CH$, we must have $N \geq C'H^{-1}$.

Finally, on Figure (c), we fixed $N = H^{-1} \geq C'H^{-1}$ and we plot ΔT for different ν. Seemingly, for all ν, we have $\Delta T \leq CH$. However, the constant C must depend on ν.

To conclude, these experiments gives us the following hints : If we have $N \geq C'H^{-1}$, then $\Delta T \leq C(\nu)H$.

Speed comparison

The result of the speed tests is recapitulated in Table 5.1.

NFine	NCoarse	Δt_{expl} (ns)	N	ν	T (ms)	Matrix assembly time (ms)	Simulation time (ms)	Relative L^2-Error
FEM/Euler								
100	-	8900.0	-	-	1.0	5.2	256.6	1.600e-10
200	-	2225.0	-	-	1.0	5.6	1054.2	1.447e-11
500	-	356.0	-	-	1.0	6.4	7083.7	1.050e-12
1000	-	89.0	-	-	1.0	7.7	31961.3	9.709e-14
1500	-	39.6	-	-	1.0	7.6	87790.5	1.050e-14
FEM/STS								
2000	-	22.3	1	1.00	1.0	10.6	351614.2	5.099e-16
2000	-	22.3	1	0.10	1.0	11.0	189142.3	2.824e-16
2000	-	22.3	1	0.01	1.0	9.2	171627.6	1.831e-16
2000	-	22.3	2	1.00	1.0	10.5	257827.8	1.308e-18
2000	-	22.3	2	0.10	1.0	10.2	94199.1	1.517e-15
2000	-	22.3	2	0.01	1.0	9.7	67850.6	4.359e-15
2000	-	22.3	5	1.00	1.0	9.8	207419.3	1.756e-15
2000	-	22.3	5	0.10	1.0	9.4	65897.5	3.185e-14
2000	-	22.3	5	0.01	1.0	10.3	27169.5	1.900e-13
2000	-	22.3	10	1.00	1.0	10.5	190784.6	2.231e-14
2000	-	22.3	10	0.10	1.0	10.7	60923.9	2.227e-13
2000	-	22.3	10	0.01	1.0	10.1	19859.8	3.958e-12
2000	-	22.3	20	1.00	1.0	10.5	184691.3	6.596e-14
2000	-	22.3	20	0.10	1.0	10.0	59115.9	6.525e-13
2000	-	22.3	20	0.01	1.0	10.5	18658.6	1.350e-11
LOD/Euler								
2000	50	35600.0	-	-	1.0	1627.2	59.4	1.478e-09
2000	100	8900.0	-	-	1.0	3141.5	239.9	1.381e-10
2000	200	2225.0	-	-	1.0	6214.9	1003.9	9.331e-12
2000	500	356.0	-	-	1.0	15492.2	6898.8	4.134e-13
2000	1000	89.0	-	-	1.0	31750.2	31696.4	1.650e-14

NFine	NCoarse	Δt_{expl} (ns)	N	ν	T (ms)	Matrix assembly time (ms)	Simulation time (ms)	Relative L^2-Error
LOD/STS								
2000	50	35600.0	1	1.00	1.0	1631.3	120.1	4.916e-10
2000	50	35600.0	1	0.10	1.0	1602.5	65.1	3.605e-09
2000	50	35600.0	1	0.01	1.0	1600.4	60.8	2.253e-09
2000	50	35600.0	2	1.00	1.0	1603.6	75.5	1.492e-09
2000	50	35600.0	2	0.10	1.0	1601.6	26.9	1.448e-08
2000	50	35600.0	2	0.01	1.0	1603.1	18.9	3.227e-08
2000	50	35600.0	5	1.00	1.0	1601.8	46.5	1.143e-08
2000	50	35600.0	5	0.10	1.0	1614.7	8.9	1.810e-07
2000	50	35600.0	5	0.01	1.0	1594.0	0.1	9.559e-07
2000	50	35600.0	10	1.00	1.0	1589.6	30.1	7.801e-08
2000	50	35600.0	10	0.10	1.0	1596.6	0.1	9.559e-07
2000	50	35600.0	10	0.01	1.0	1593.3	0.1	9.559e-07
2000	50	35600.0	20	1.00	1.0	1592.3	14.0	3.927e-07
2000	50	35600.0	20	0.10	1.0	1595.1	0.1	9.559e-07
2000	50	35600.0	20	0.01	1.0	1593.1	0.1	9.559e-07
2000	100	8900.0	1	1.00	1.0	3131.6	487.8	5.747e-11
2000	100	8900.0	1	0.10	1.0	3137.9	267.6	1.570e-10
2000	100	8900.0	1	0.01	1.0	3132.4	246.6	1.606e-10
2000	100	8900.0	2	1.00	1.0	3338.8	321.2	1.390e-10
2000	100	8900.0	2	0.10	1.0	3437.5	113.2	6.797e-10
2000	100	8900.0	2	0.01	1.0	3187.4	78.8	2.344e-09
2000	100	8900.0	5	1.00	1.0	3230.1	198.6	1.746e-09
2000	100	8900.0	5	0.10	1.0	3205.7	59.5	7.191e-09
2000	100	8900.0	5	0.01	1.0	3179.3	18.9	9.627e-08
2000	100	8900.0	10	1.00	1.0	3227.6	158.5	4.005e-09
2000	100	8900.0	10	0.10	1.0	3394.2	45.1	2.262e-08
2000	100	8900.0	10	0.01	1.0	3263.4	7.6	3.066e-07
2000	100	8900.0	20	1.00	1.0	3181.2	132.8	1.131e-08
2000	100	8900.0	20	0.10	1.0	3177.8	27.4	1.798e-07
2000	100	8900.0	20	0.01	1.0	3155.2	0.1	9.559e-07
2000	200	2225.0	1	1.00	1.0	6246.7	2050.7	4.000e-12
2000	200	2225.0	1	0.10	1.0	6250.7	1129.0	7.116e-12
2000	200	2225.0	1	0.01	1.0	6250.2	1033.6	1.657e-11
2000	200	2225.0	2	1.00	1.0	6249.3	1317.3	9.388e-12
2000	200	2225.0	2	0.10	1.0	6247.1	477.1	7.175e-11
2000	200	2225.0	2	0.01	1.0	6222.8	341.1	2.479e-10
2000	200	2225.0	5	1.00	1.0	6225.1	863.3	9.023e-11
2000	200	2225.0	5	0.10	1.0	6242.2	272.8	2.916e-10
2000	200	2225.0	5	0.01	1.0	6228.6	106.4	4.070e-09
2000	200	2225.0	10	1.00	1.0	6280.7	705.7	4.099e-10
2000	200	2225.0	10	0.10	1.0	6288.2	218.1	2.339e-09
2000	200	2225.0	10	0.01	1.0	6295.7	64.8	1.864e-08
2000	200	2225.0	20	1.00	1.0	6251.4	624.5	1.743e-09

NFine	NCoarse	Δt_{expl} (ns)	N	ν	T (ms)	Matrix assembly time (ms)	Simulation time (ms)	Relative L^2-Error
2000	200	2225.0	20	0.10	1.0	6250.0	187.4	6.788e-09
2000	200	2225.0	20	0.01	1.0	6247.8	43.2	1.039e-07
2000	500	356.0	1	1.00	1.0	15493.3	14123.1	9.035e-14
2000	500	356.0	1	0.10	1.0	15478.4	7769.2	3.008e-13
2000	500	356.0	1	0.01	1.0	15517.4	7136.7	1.063e-13
2000	500	356.0	2	1.00	1.0	15490.5	9375.9	4.140e-13
2000	500	356.0	2	0.10	1.0	15501.1	3414.0	2.818e-12
2000	500	356.0	2	0.01	1.0	15609.3	2455.7	1.873e-12
2000	500	356.0	5	1.00	1.0	15560.3	6472.1	1.771e-12
2000	500	356.0	5	0.10	1.0	15591.6	2055.5	1.267e-11
2000	500	356.0	5	0.01	1.0	15605.4	846.4	6.708e-11
2000	500	356.0	10	1.00	1.0	15530.3	5537.6	9.303e-12
2000	500	356.0	10	0.10	1.0	16525.1	1703.0	8.029e-11
2000	500	356.0	10	0.01	1.0	15605.1	552.6	4.237e-10
2000	500	356.0	20	1.00	1.0	15621.2	4933.6	4.211e-11
2000	500	356.0	20	0.10	1.0	16071.9	1577.4	3.934e-10
2000	500	356.0	20	0.01	1.0	15490.6	491.7	1.442e-09
2000	1000	89.0	1	1.00	1.0	31351.2	66856.5	1.751e-15
2000	1000	89.0	1	0.10	1.0	33068.2	36307.1	6.479e-15
2000	1000	89.0	1	0.01	1.0	30994.4	32798.7	5.063e-15
2000	1000	89.0	2	1.00	1.0	31038.5	45100.5	1.631e-14
2000	1000	89.0	2	0.10	1.0	32467.3	16510.8	7.445e-14
2000	1000	89.0	2	0.01	1.0	30969.4	11788.7	1.536e-13
2000	1000	89.0	5	1.00	1.0	30902.9	32948.9	6.586e-14
2000	1000	89.0	5	0.10	1.0	30934.0	10458.0	7.303e-13
2000	1000	89.0	5	0.01	1.0	31572.8	4440.6	2.751e-12
2000	1000	89.0	10	1.00	1.0	32800.8	29524.4	2.224e-13
2000	1000	89.0	10	0.10	1.0	32532.3	9411.3	4.706e-12
2000	1000	89.0	10	0.01	1.0	31308.9	2988.8	6.717e-11
2000	1000	89.0	20	1.00	1.0	32161.7	27138.6	1.772e-12
2000	1000	89.0	20	0.10	1.0	30953.5	8532.3	1.242e-11
2000	1000	89.0	20	0.01	1.0	31143.7	2669.6	1.501e-10

Table 5.1: Speed comparison between the 4 completely discretized methods.

We achieved the experiment in one dimension. We chose $u_0 = 0$ and $f = 1$. Also, the diffusion coefficient $A = (2 + cos(\frac{2\pi}{\epsilon}x)^{-1}$. We performed the test for the four completely discretized methods with the same end time $T = 0.001$. We compare how fast the methods reach this end time for different discretization parameters.

The first thing to notice is that the time to build the mass and the stiffness matrices is much long for the localized orthogonal decomposition method than for the finite element method. Indeed, for the finite element method, the worst case is for a really fine mesh (here 2000) and it takes around 10 ms. For the localized orthogonal decomposition method, the best case is

when the coarse mesh size is large (here 50) and it takes approximately 1600 ms. However, the simulation time is lower for the localized orthogonal decomposition method. Lastly, for both space discretization methods, it is faster to use the Super-time stepping acceleration rather than the forward Euler method.

Chapter 6

Concluding remarks

To conclude, we summarize the main points discussed in this thesis.

6.1 Recap

The main points discussed in this thesis are the following

- The finite element method applied to a problem including fast varying coefficient is not efficient. In addition to a convergence rate of order one with respect to the mesh size, we also have a pre-asymptotic effect where the numeric solution is fully different from the exact solution.
- In order to avoid the pre-asymptotic effect, we constructed the localized orthogonal decomposition method. This method have a better convergence rate than the finite element method one and is less expensive to compute.
- For time discretization we use the forward Euler method. As it is an explicit finite differences method, it has a stability condition : the time step must be lower than the mesh size squared. This condition coupled with the pre-asymptotic effect of the finite element method leads to an unusable completely discretized method. However, coupled with the localized orthogonal decomposition method, it leads to a practicable method.
- We can improve the forward Euler method with the Super-time stepping acceleration. In addition we relax the stability condition of the Euler method to ensure the stability over a super time step.
- For a diffusion coefficient with fast variations, the finite element method mixed with the forward Euler method is the worst. The localized orthogonal decomposition method with super-time stepping acceleration is the best option to solve such problems.

Bibliography

[1] Vasilios Alexiades, Geneviève Amiez, and Pierre-Alain Gremaud. Super-time-stepping acceleration of explicit schemes for parabolic problems. *Communications in Numerical Methods in Engineering*, 12(1):31–42, 1996.

[2] Vassilios A. Dougalis. Finite element methods for the numerical solution of partial differential equations (lecture notes), 2013.

[3] Lawrence C. Evans. *Partial Differential Equations*, volume 19. American Mathematical Society, 2010.

[4] F. Hellman. Gridlod – lod for structured grids, 2017.

[5] Stig Larsson and Vidar Thomee. *Partial Differential Equations with Numerical Methods*, volume 45. Springer-Verlag Berlin Heidelberg, 2003.

[6] A. Målqvist and A. Persson. Multiscale techniques for parabolic equations. *ArXiv e-prints*, April 2015.

[7] R. Maier and D. Peterseim. Explicit Computational Wave Propagation in Micro-Heterogeneous Media. *ArXiv e-prints*, March 2018.

[8] A. Malqvist and D. Peterseim. Localization of Elliptic Multiscale Problems. *ArXiv e-prints*, August 2013.

[9] J. T. Oden. *An Introduction to the Mathematical Theory of Finite Elements*. Dover Publications, 2011.

[10] Daniel Peterseim. Numerical homogenization of partial differential equations (lecture notes), February 2018.

Appendix A

FEM class

```
import numpy as np
import scipy.sparse as sparse
from gridlod import fem, util
from util import computeDeltatExpl

class DiffusionProblemFEM:
        def __init__(self, NPatch):
                self.NPatch = NPatch
                self.diffusion_coeff = np.ones(np.prod(NPatch))
                self.f = np.zeros(np.prod(NPatch+1))

        def generateRandCoeff (self, alpha, beta):
                self.diffusion_coeff = np.random.rand ( np.prod(self.
                   NPatch) )*(beta - alpha) + alpha

        def generateBinaryCoeff (self, alpha, beta):
                self.diffusion_coeff = beta*np.ones ( np.prod(self.
                   NPatch) )
                self.diffusion_coeff[ np.arange(np.prod(self.NPatch)
                   /2) ] = alpha

        def generatePeriodicCoeff(self, epsilon):
                X = np.linspace(0, 1, self.NPatch)
                self.diffusion_coeff = 1./(2+np.cos(2*np.pi*X/epsilon
                   ))

        def assembleMatrices(self):
                KLocFull = fem.localStiffnessMatrix(self.NPatch)
                self.KFull = fem.assemblePatchMatrix(self.NPatch,
                   KLocFull, self.diffusion_coeff)
                MLocFull = fem.localMassMatrix(self.NPatch)
                self.MFull = fem.assemblePatchMatrix(self.NPatch,
                   MLocFull)

```

```
29          def initSuperStep(self, N, nu):
30                  beta = np.max(self.diffusion_coeff)
31                  delta_t_expl, triangle_size = computeDeltatExpl(self.
                        NPatch, beta)
32
33                  self.tau = np.zeros(N)
34                  for i in range(1, N+1):
35                          self.tau[i-1] = delta_t_expl/( (nu-1)*np.cos(
                                np.pi*(2*i-1)/(2*N)) + 1 + nu )
36
37                  return delta_t_expl
38
39          def solveStep(self, xFull_previous, delta_t):
40                  free = util.interiorpIndexMap(self.NPatch)
41
42                  bFull = delta_t*self.MFull*self.f + (self.MFull -
                        delta_t*self.KFull)*xFull_previous
43
44                  MFree = self.MFull[free][:,free]
45                  bFree = bFull[free]
46
47                  xFree = sparse.linalg.spsolve(MFree, bFree)
48
49                  xFull_now = np.zeros(np.prod(self.NPatch+1))
50                  xFull_now[free] = xFree
51                  return xFull_now
52
53          def solveSuperStep(self, xFull_previous, N, nu):
54                  free = util.interiorpIndexMap(self.NPatch)
55                  MFree = self.MFull[free][:,free]
56
57                  for n in range(0, N):
58                          bFull = self.tau[n]*self.MFull*self.f + (self
                                .MFull - self.tau[n]*self.KFull)*
                                xFull_previous
59
60                          bFree = bFull[free]
61                          xFree_now = sparse.linalg.spsolve(MFree,
                                bFree)
62
63                          xFull_now = np.zeros(np.prod(self.NPatch+1))
64                          xFull_now[free] = np.transpose(xFree_now)
65                          xFull_previous = xFull_now
66
67                  return xFull_previous
```

Appendix B

LOD class

```
1  import numpy as np
2  import scipy.sparse as sparse
3  from gridlod import pg, interp, coef, util, fem, world, linalg,
       femsolver, transport
4  from gridlod.world import World
5
6  from util import computeDeltatExpl
7
8  class DiffusionProblemLOD:
9          def __init__(self, NFine, NCoarse):
10                 mod = NFine % NCoarse
11                 if ( len(NFine) == 1 ):
12                         assert ( mod[0] == 0 ), "NCoarse is not a
                               refinement of NFine"
13                 elif ( len(NFine) == 2 ):
14                         assert ( mod[0] == 0 and mod[1] == 0 ), "
                               NCoarse is not a refinement of NFine"
15                 else:
16                         assert ( true == false ), "Dimension not
                               supported"
17
18                 self.NFine = NFine
19                 self.NCoarse = NCoarse
20                 self.diffusion_coeff = np.ones(np.prod(NFine))
21                 self.f = np.zeros(np.prod(NCoarse+1))
22                 NCoarseElement = self.NFine/self.NCoarse
23
24                 if ( len(NFine) == 1 ):
25                         self.boundaryConditions = np.array([[0, 0]])
26                 elif ( len(NFine) == 2 ):
27                         self.boundaryConditions = np.array([[0, 0],
                               [0, 0]])
28                         self.P = interp.L2ProjectionPatchMatrix(np.
                               array([0,0]), self.NCoarse, self.NCoarse,
```

```
                                NCoarseElement, self.boundaryConditions)
29
30          def generateRandCoeff (self, alpha, beta):
31                  self.diffusion_coeff = np.random.rand ( np.prod(self.
                       NFine) )*(beta - alpha) + alpha
32
33          def generateBinaryCoeff (self, alpha, beta):
34                  self.diffusion_coeff = beta*np.ones ( np.prod(self.
                       NFine) )
35                  self.diffusion_coeff[ np.arange(np.prod(self.NFine)
                       /2) ] = alpha
36
37          def generatePeriodicCoeff(self, epsilon):
38                 X = np.linspace(0, 1, self.NFine)
39                  self.diffusion_coeff = 1./(2+np.cos(2*np.pi*X/epsilon
                       ))
40
41          def initSuperStep(self, N, nu):
42                  beta = np.max(self.diffusion_coeff)
43                  delta_t_expl, triangle_size = computeDeltatExpl(self.
                       NCoarse, beta)
44
45                  self.tau = np.zeros(N)
46                  for i in range(1, N+1):
47                          self.tau[i-1] = delta_t_expl/( (nu-1)*np.cos(
                               np.pi*(2*i-1)/(2*N)) + 1 + nu )
48
49                  return delta_t_expl, triangle_size
50
51          def assembleMatrices(self, k=2):
52                  NCoarseElement = self.NFine/self.NCoarse
53                  world = World(self.NCoarse, NCoarseElement, self.
                       boundaryConditions)
54
55                  rCoarse = np.ones(np.prod(self.NCoarse))
56                  aCoef = coef.coefficientCoarseFactor(self.NCoarse,
                       NCoarseElement, self.diffusion_coeff, rCoarse)
57
58                  IPatchGenerator = lambda i, N: interp.
                       L2ProjectionPatchMatrix(i, N, self.NCoarse,
                       NCoarseElement, self.boundaryConditions)
59                  pglod = pg.PetrovGalerkinLOD(world, k,
                       IPatchGenerator, 0)
60                  pglod.updateCorrectors(aCoef, clearFineQuantities=
                       False)
61
62                  self.KFull = pglod.assembleMsStiffnessMatrix() #
```

```
                Striffness Matrix
63              self.MFull = fem.assemblePatchMatrix(self.NCoarse,
                    world.MLocCoarse) # Mass Matrix
64
65              basis = fem.assembleProlongationMatrix(self.NCoarse,
                    NCoarseElement) # VH basis
66              basisCorrectors = pglod.assembleBasisCorrectors() #
                  WH basis
67              modifiedBasis = basis - basisCorrectors # ~VH basis
68
69              return modifiedBasis, basis
70
71      def solveStep(self, xFull_previous, delta_t):
72              NCoarseElement = self.NFine/self.NCoarse
73              free = util.interiorpIndexMap(self.NCoarse)
74
75              bFull = delta_t*self.MFull*self.f + (self.MFull -
                    delta_t*self.KFull)*xFull_previous
76
77              MFree = self.MFull[free][:,free]
78              bFree = bFull[free]
79
80              xFree = sparse.linalg.spsolve(MFree, bFree)
81
82              xFull_now = np.zeros(np.prod(self.NCoarse+1))
83              xFull_now[free] = xFree
84              return xFull_now
85
86      def solveSuperStep(self, xFull_previous, N, nu):
87              beta = np.max(self.diffusion_coeff)
88              delta_t_expl = computeDeltatExpl(self.NCoarse, beta)
89
90              free = util.interiorpIndexMap(self.NCoarse)
91              MFree = self.MFull[free][:,free]
92
93              for n in range(0, N):
94                      bFull = self.tau[n]*self.MFull*self.f + (self
                            .MFull - self.tau[n]*self.KFull)*
                            xFull_previous
95
96                      bFree = bFull[free]
97                      xFree_now = sparse.linalg.spsolve(MFree,
                            bFree)
98
99                      xFull_now = np.zeros(np.prod(self.NCoarse+1))
100                     xFull_now[free] = np.transpose(xFree_now)
101                     xFull_previous = xFull_now
```

```
102
103                 return xFull_previous
```

Appendix C

FEM implementation

C.1 Forward Euler method

```
import numpy as np
import matplotlib.pyplot as plt
from util import computeDeltatExpl

from DiffusionProblemFEM import DiffusionProblemFEM

# Mesh
NFine = np.array([20, 20])
# forcing term
def f(t):
        res = np.zeros(np.prod(NFine+1))
        if t <= 0.002:
                res = np.ones(np.prod(NFine+1))
        return res;
# initial condition
u_0 = lambda X: np.zeros(np.prod(X+1))

# problem setting
alpha = 0.01
beta  = 200.

delta_t_expl, triangle_size = computeDeltatExpl(NFine, beta)
delta_t = delta_t_expl
t_max   = 1.

problemFEM = DiffusionProblemFEM(NFine)
problemFEM.generateRandCoeff(alpha, beta)
problemFEM.assembleMatrices()
problemFEM.f = f(0)

# Initialize the simulation
xFullFEM = u_0(NFine)
```

```
# Simulation's loop
nb_loop = int(t_max / delta_t)
for n in range(1, nb_loop):
        t=n*delta_t
        problemFEM.f = f(t)
        xFullFEM = problemFEM.solveStep(xFullFEM, delta_t)

        # plot solution
        plt.figure(0)
        plt.clf()
        plt.imshow(xFullFEM.reshape(NFine+1) )
        plt.title("solution at t=" + str(t*10**6) + "us")
        plt.colorbar()
        plt.draw()
        plt.pause(0.001)
```

C.2 Super-time Stepping acceleration

```
import numpy as np
import matplotlib.pyplot as plt
from util import computeDeltatExpl

from DiffusionProblemFEM import DiffusionProblemFEM

# Mesh
NFine = np.array([100, 100])
N = 10
nu = 0.1
# forcing term
def f(t):
        res = np.zeros(np.prod(NFine+1))
        if t <= 0.002:
                res = np.ones(np.prod(NFine+1))
        return res;
# initial condition
u_0 = lambda X: np.zeros(np.prod(X+1))

# problem setting
alpha = 0.01
beta  = 200.

t_max   = 1.

problemFEM = DiffusionProblemFEM(NFine)
problemFEM.generateRandCoeff(alpha, beta)
problemFEM.assembleMatrices()
problemFEM.f = f(0)
```

```
30
31 problemFEM.initSuperStep(N,nu)
32 delta_T = np.sum(problemFEM.tau)
33
34 # Initialize the simulation
35 xFullFEM = u_0(NFine)
36 # Simulation's loop
37 nb_loop = int(t_max / delta_T)
38 for n in range(1, nb_loop):
39         t=n*delta_T
40         problemFEM.f = f(t)
41         xFullFEM = problemFEM.solveSuperStep(xFullFEM, N, nu)
42
43         # plot solution
44         plt.figure(0)
45         plt.clf()
46         plt.imshow(xFullFEM.reshape(NFine+1) )
47         plt.title("solution at t=" + str(t*10**6) + "us")
48         plt.colorbar()
49         plt.draw()
50         plt.pause(0.001)
```

Appendix D

LOD implementation

D.1 Forward Euler method

```
import numpy as np
import matplotlib.pyplot as plt

from DiffusionProblemLOD import DiffusionProblemLOD
from util import norm, computeDeltatExpl

# Mesh
NFine = np.array([12, 12])
NCoarse = np.array([3,3])
# forcing term
def f(t):
        res = np.zeros(np.prod(NCoarse+1))
        if t <= 0.002:
                res = np.ones(np.prod(NCoarse+1))
        return res;
# initial condition
u_0 = lambda X: np.zeros(np.prod(X+1))

# problem setting
alpha = 0.01
beta  = 200.

delta_t_expl, triangle_size = computeDeltatExpl(NFine, beta)
delta_t = delta_t_expl
t_max   = 1.

problemLOD = DiffusionProblemLOD(NFine, NCoarse)
problemLOD.generateRandCoeff(alpha, beta)
lod_basis, fem_basis = problemLOD.assembleMatrices()
problemLOD.f = f(0)

# Initialize the simulation
```

```
xFullLOD = u_0(NCoarse)
# Simulation's loop
nb_loop = int(t_max / delta_t)
for n in range(1, nb_loop):
        t=n*delta_t
        problemLOD.f = f(t)
        xFullLOD = problemLOD.solveStep(xFullLOD, delta_t)

        # plot solution
        plt.figure(0)
        plt.clf()
        plt.imshow((lod_basis*xFullLOD).reshape(NFine+1) )
        plt.title("solution at t=" + str(t*10**6) + "us")
        plt.colorbar()
        plt.draw()
        plt.pause(0.001)
```

D.2 Super-time Stepping acceleration

```
import numpy as np
import matplotlib.pyplot as plt

from DiffusionProblemLOD import DiffusionProblemLOD
from util import norm

# Mesh
NFine = np.array([12, 12])
NCoarse = np.array([3,3])
N = 10
nu = 0.1
# forcing term
def f(t):
        res = np.zeros(np.prod(NCoarse+1))
        if t <= 0.002:
                res = np.ones(np.prod(NCoarse+1))
        return res;
# initial condition
u_0 = lambda X: np.zeros(np.prod(X+1))

# problem setting
alpha = 0.01
beta  = 200.

t_max   = 1.

problemLOD = DiffusionProblemLOD(NFine, NCoarse)
problemLOD.generateRandCoeff(alpha, beta)
```

```
29 lod_basis, fem_basis = problemLOD.assembleMatrices()
30 problemLOD.f = f(0)
31
32 problemLOD.initSuperStep(N,nu)
33 delta_T = np.sum(problemLOD.tau)
34
35 # Initialize the simulation
36 xFullLOD = u_0(NCoarse)
37 # Simulation's loop
38 nb_loop = int(t_max / delta_T)
39 for n in range(1, nb_loop):
40         t=n*delta_T
41         problemLOD.f = f(t)
42         xFullLOD = problemLOD.solveSuperStep(xFullLOD, N, nu)
43
44         # plot solution
45         plt.figure(0)
46         plt.clf()
47         plt.imshow((lod_basis*xFullLOD).reshape(NFine+1) )
48         plt.title("solution at t=" + str(t*10**6) + "us")
49         plt.colorbar()
50         plt.draw()
51         plt.pause(0.001)
```