UNIVERSIDAD NACIONAL DE COLOMBIA

Topology Optimization Problems for the Heat Equation

Miguel Andrés Zambrano Garcés

Universidad Nacional de Colombia
Facultad de Ciencias
Bogotá, Colombia

2019

Topology Optimization Problems for the Heat Equation

Miguel Andrés Zambrano Garcés
`mazambranoga@unal.edu.co`

Advisor
Juan Galvis (`jcgalvisa@unal.edu.co`)
Universidad Nacional de Colombia

Co-advisor
Boyan Lazarov (`boyan.lazarov@manchester.ac.uk`)
The University of Manchester

Universidad Nacional de Colombia
Facultad de Ciencias
Bogotá, Colombia
2019

A Sintya y Guambra Pelusa, mi familia.

Agradecimientos

Thanks to Juan and Boyan for their infinite patience, support and dedication in the realization of this project.

Abstract

In this document, we analyze the performance of the Schwarz two-levels preconditioners applied to the topology optimization problem for the heat equation. We discretize the topology optimization problem using the finite element method and apply an optimization method that requires, in each iteration, the solution of the heat equation. For the topology optimization problem, we use the density based formulation which has been actively developed in the last 30 years. Due to the nature of the optimization problem, it yields a high-contrast multiscale coefficient for the heat equation and therefore designing efficient solution methods is a challenging task that we approach by designing and testing several preconditioners. These preconditioners are built using a domain decomposition method and the generalized multiscale finite element method (GMsFEM) recently introduced. It is known that for a good performance of the preconditioner it is important the design of the basis functions. In this document, the calculation of multiscale basis functions uses the solution of carefully selected local eigenvalue problems as usual in the GMsFEM. We also propose the approximation of the local eigenvalues using a randomized algorithm to obtain an overall less expensive methodology. Topology optimization experiments are presented in which the good performance of the proposed methods is verified.

Keywords: heat equation, finite element method, multiscale method, domain decomposition, two-levels Schwarz preconditioner, topology optimization.

Resumen

En este trabajo, analizamos el rendimiento de los precondicionadores de dos niveles de Schwarz aplicados al problema de optimización topológica para la ecuación de calor. Discretizamos el problema de optimización topológica utilizando el método de elementos finitos y aplicamos un método de optimización que requiere, en cada iteración, la solución de la ecuación de calor. En el problema de optimización topológica utilizamos la formulación basada en densidad que se ha desarrollado activamente en los últimos 30 años. Debido a la naturaleza del problema de optimización, este produce un coeficiente multiescala de alto contraste para la ecuación de calor y, por lo tanto, diseñar métodos de solución eficientes es una tarea desafiante que abordamos al diseñar y probar varios precondicionadores. Estos precondicionadores se crean utilizando un método de descomposición de dominios y el método generalizado de elementos finitos multiescala (GMsFEM) recientemente introducido. Se sabe que para el buen desempeño del precondicionador es importante el diseño de las funciones base. En este trabajo, el cálculo de las funciones base multiescala utiliza la solución de problemas de valores propios locales seleccionados cuidadosamente como es habitual en el GMsFEM. También proponemos la aproximación de los valores propios locales utilizando un algoritmo aleatorio para obtener una metodología general menos costosa. Se presentan experimentos de optimización topológica en los que se verifica el buen desempeño de los métodos propuestos.

Palabras clave: ecuación del calor, método de elementos finitos, método multiescala, descomposición de dominios, precondicionador de dos niveles de Schwarz, optimización topológica.

Contents

List of Figures

List of Tables

Introduction

Understanding heat conductivity is very important for many practical applications such as the design of composite materials with specific conducting properties, among many other applications. There are several aspects related to heat conduction that shall be taking into account when dealing with such applications. Roughly speaking, heat flows from hotter to colder parts of a body, which is known as Fourier's law. More precisely, the heat conduction law states that heat flux transfer through a material is proportional to the negative of the temperature gradient. This proportion is given by a coefficient called thermal conductivity that depends on the material. In some applications, it is important to design a material with optimal thermal conductivity. In this case, the optimal design problem can be modeled using optimization methodologies where the cost functional is computed over solutions of the heat equation with the actual proposed design.

In topology optimization problems it is usual to obtain designs with a multiscale structure for the high-conductivity part of the material and also a high-contrast property is present due to the density formulation. These problems make the linear system hard to solve via direct methods, especially if we have large domains, see more in [30].

In this work, we analyze the performance of the two-levels preconditioners applied to the topology optimization problem for the heat equation. This study will lead to an application of heat preconditioners in the more computationally expensive elasticity equation case, see [35] and [36]. We review the mathematical and numerical tools concerning topology optimality problems related to the thermal conductivity design of materials.

We begin with a review of the finite element method including the most relevant aspects. We also present an example of the finite element method applied to a two-dimension problem. Finally, we present a comparison between heat and elasticity equations to see how simpler heat calculations can be applied to elasticity. This is explored in the *sister* master thesis [35] where using the results presented here they design preconditioners for elasticity topology optimization.

The second and third chapters are devoted to the study of the two-levels preconditioner for the heat equation, based on [17, 16, 21]. We give a short review of classical MsFEM and then we focus on GMsFEM, and we make a description of a randomized algorithm to calculate the coarse basis elements as in [8, 25].

Then we review the domain decomposition methods, used to build the two-levels preconditioner. We show that one can obtain preconditioners that yield a contrast-independent condition number, as in [39, 14, 22, 23, 19].

In Chapter 4 we address the main concepts in topology optimization, based in the works of [3, 2, 1, 31]. We present a simple example of the minimum compliance design for the heat equation in two dimensions, which include the FEM discretization on a squared mesh.

In Chapter 5, we give two examples of topology optimization problems with the heat equation. We compare the iteration count of the PCG method when using the preconditioner and when directly solving the system. To reduce computational costs we use a randomized approach to solve the eigenvalue problem as in [8, 25]. To further reduce the computational cost, we also present results of iterations if we reuse the coarse basis for some optimization steps, thus avoiding the calculation of eigenvalue problems in all the optimization iterations.

Chapter 1

The Finite Element Method

The Finite Element Method (FEM) is a procedure that allows us to numerically solve a partial differential equation over a domain, often with complex geometries, by simply solving a matrix equation. In this chapter, we shortly describe the abstract formalism for the method, following [29, 26, 4, 5, 6, 28], and then present an example using the heat equation in two dimensions. The finite element method will be used later to obtain a discrete version of the topology optimization problem for the heat equation that is the main topic of this dissertation.

1.1 FEM Fundamental Concepts

The finite element method is a procedure that allows us to solve boundary value problems numerically on a domain that can have a rather complex geometry. The method allows us to transform the problem defined on function spaces to a matrix problem that is possible to solve using a computing system. Next, we describe in a very brief way the abstract formulation of the method, we follow [29].

Let us define a *trial* function space V and a *test* function space $\widehat{V}$. These spaces are chosen according to a particular problem to be addressed. Let $a : V \times \widehat{V} \to \mathbb{R}$ be a bilinear form and $l : \widehat{V} \to \mathbb{R}$ a bounded linear functional. We consider the variational problem: find $u \in V$ such that

$$a(u, v) = l(v) \quad \text{for all } v \in \widehat{V}.$$

We can discretize this variational problem by searching solutions over a finite-dimensional subspace $V^h \subset V$. Now, the discretized problem is to find $u_h \in V^h$ such that

$$a(u_h, v) = l(v) \quad \text{for all } v \in \widehat{V^h},$$

over an appropriate $\widehat{V^h} \subset \widehat{V}$. Let $\{\phi_i\}_{i=1}^n$ be a basis of V^h and $\{\widehat{\phi}_i\}_{i=1}^n$ a basis of $\widehat{V^h}$, we can write u_h as a linear combination,

$$u_h = \sum_{j=1}^{n} \alpha_j \phi_j,$$

also, lets take $v = \widehat{\phi}_i$ for $i = 1, 2, \ldots, n$. It follows that

$$\sum_{j=1}^{n} \alpha_j a(\phi_j, \widehat{\phi}_i) = l(\widehat{\phi}_i) \quad \text{for } i = 1, 2, \ldots, n.$$

The weights in the finite element solution u_h can be computed by solving the linear system $\mathbf{A}\vec{\alpha} = \vec{b}$, where

$$a_{ij} = a(\phi_j, \widehat{\phi}_i), \quad i, j = 1, 2, \ldots, n \quad \text{and} \quad b_i = l(\widehat{\phi}_i), \quad i = 1, 2, \ldots, n.$$

From here on we adopt the following definitions and conventions, let $(\mathcal{H}, \langle\cdot,\cdot\rangle)$ a Hilbert space over $\mathbb{R}$ with $\|\cdot\|_{\mathcal{H}}$ the associated norm, $f \in \mathcal{H}^*$ and $a : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$ is a continuous and coercive bilinear form, i.e.:

- The form a is continuous if $|a(v, w)| \leq \gamma \|v\| \, \|w\|$ for some $\gamma > 0$ and for all v, w in $\mathcal{H}$.
- The form a is coercive in V or V-elliptic if $a(v, v) \geq \alpha \|v\|^2$ for some $\alpha > 0$ and all v in $\mathcal{H}$.

1.2 Existence and Uniqueness of the Solution

Let us consider the following minimization problem: find $u \in \mathcal{H}$ such that

$$F(u) = \min_{v \in \mathcal{H}} F(v), \tag{1.1}$$

where

$$F(v) = \frac{1}{2} a(v, v) - f(v),$$

with $a : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$ a symmetric bilinear form. Consider also the variational problem: find $u \in \mathcal{H}$ such that

$$a(u, v) = f(v) \qquad \text{for all } v \in \mathcal{H}. \tag{1.2}$$

Now we show that both problems are equivalent, the uniqueness of the solution is verified later using the Lax-Milgram theorem.

Let $v \in \mathcal{H}$ and $\epsilon > 0$, then $(u + \epsilon v) \in \mathcal{H}$ and since u is a minimum of F, we have that for all $\epsilon > 0$

$$F(u) \leq F(u + \epsilon v).$$

Let $g(\epsilon) = F(u + \epsilon v)$, thus

$$g(0) \leq g(\epsilon) \quad \text{ for all } \ \epsilon > 0.$$

Then, g reaches a minimum when $\epsilon = 0$. Now we show that $y'(0)$ exists and it vanishes. Observe that

$$\begin{aligned} g(\epsilon) &= \frac{1}{2} a(u + \epsilon v, u + \epsilon v) - f(u + \epsilon v) \\ &= \frac{1}{2} a(u, u) + \frac{\epsilon}{2} a(u, v) + \frac{\epsilon}{2} a(v, u) + \frac{\epsilon^2}{2} a(v, v) - f(u) - \epsilon f(v), \end{aligned}$$

and since $a(\cdot,\cdot)$ is symmetric we have,

$$= \frac{1}{2} a(u, u) - f(u) + \epsilon a(u, v) - \epsilon f(v) + \frac{\epsilon^2}{2} a(v, v),$$

whence,

$$g'(\epsilon) = a(u, v) - f(v) + \epsilon a(v, v),$$

and since $g'(0) = 0$,

$$0 = g'(0) = a(u, v) = f(v).$$

To prove the other equivalence, we proceed similarly.

As suggested in the previous section, we can discretize the problem (1.2) by looking for u_h in V^h which is a finite dimension subspace of $\mathcal{H}$ such that

$$a(u_h, v_h) = f(v_h) \qquad \text{for all } v_h \in V^h. \tag{1.3}$$

For a Hilbert space $\mathcal{H}$ let $\mathcal{H}^*$ be defined as the linear space of all bounded linear functionals $f : \mathcal{H} \to \mathrm{I\!R}$ with the norm

$$\|f\|_{\mathcal{H}^*} = \sup_{x \neq 0} \frac{|f(x)|}{\|x\|_{\mathcal{H}}}.$$

Note that $\mathcal{H}^*$ is itself a Hilbert space.

Now we need to recall a well-known theorem in the scope of Hilbert spaces.

Theorem 1 (Riesz representation)**.** *Let $\mathcal{H}$ be a Hilbert space over $\mathrm{I\!R}$. The mapping*

$$\begin{aligned} \gamma : \mathcal{H} &\to \mathcal{H}^* \\ \xi &\mapsto \gamma_\xi, \end{aligned}$$

where $\gamma_\xi(\eta) = \langle \xi, \eta \rangle$, for all $\eta \in \mathcal{H}$ is an isometric isomorphism.

By the Riesz representation theorem we have that every $f \in \mathcal{H}^*$ has a unique representation

$$f(v) = \langle u, v \rangle,$$

for some $u \in \mathcal{H}$. Also $\|f\| = \|u\|$.

First, we deal with symmetric variational problems, that is, the case where the bilinear form $a(\cdot, \cdot)$ is symmetric.

Proposition 1. *Let $\mathcal{H}$ be a Hilbert space, and $a(\cdot, \cdot)$ a continuous symmetric bilinear form that is coercive in V, being V a closed subspace of $\mathcal{H}$. Then $(V, a(\cdot, \cdot))$ is a Hilbert space.*

Proof. To verify that $a(\cdot, \cdot)$ defines an inner product, it is enough to see that if $v \in V$ and $a(v, v) = 0$ then $v = 0$. This follows immediately from the fact that $a(\cdot, \cdot)$ is coercive since $0 = a(v, v) \geq \alpha \|v\|^2$, yields $v = 0$.

Now, the norm induced by the bilinear form is $\|v\|_V = \sqrt{a(v, v)}$ and let $\{v_n\}$ a Cauchy sequence in $(V, \|\cdot\|_V)$. Since $a(\cdot, \cdot)$ is coercive

$$\|v_n - v_m\|_{\mathcal{H}}^2 \leq \frac{1}{\alpha} a(v_n - v_m, v_n - v_m) = \|v_n - v_m\|_V^2,$$

where $\alpha > 0$ is the coercivity constant. Then $\{v_n\}$ is a Cauchy succession in $(\mathcal{H}, \|\cdot\|_{\mathcal{H}})$ and since $\mathcal{H}$ is complete, there is $v \in \mathcal{H}$ such that $v_n \longrightarrow v$ in the norm $\|\cdot\|_{\mathcal{H}}$. Since V is closed in $\mathcal{H}$, then $v \in V$. Using that $a(\cdot, \cdot)$ is bounded we have $\|v - v_n\|_V \leq C\|v - v_n\|_{\mathcal{H}}$, from where we deduce that $v_n \longrightarrow v$ in the norm $\|\cdot\|_V$. Therefore $(V, a(\cdot, \cdot))$ is a Hilbert space. ■

If the following conditions are fulfilled:

- The pair $(\mathcal{H}, \langle\cdot,\cdot\rangle)$ is a Hilbert space.
- The subspace V is closed in $\mathcal{H}$.
- The bilinear form $a(\cdot,\cdot)$ is symmetric, bounded and coercive in V.

Now the symmetric variational problem (1.2) is well posed.

Theorem 2. *If the above conditions are fulfilled, then exists a unique solution $u \in V$ that solves problem* (1.2)*.*

Proof. From Proposition 1, we know that $a(u, v) = f(v)$ define an inner product in V and that $(V, a(\cdot,\cdot))$ is a Hilbert space. Then, by Theorem 1 we have that for all $f \in V^*$ exist an unique $u \in V$, that solves (1.2). ■

Definition 1 (Ritz-Galerkin approximation problem)**.** *Given a finite dimensional subspace $V^h \subset V$ y $f \in V^*$, find $u_h \in V^h$ such that $a(u_h, v) = f(v)$ for all $v \in V^h$.*

Theorem 3. *Let us suppose valid the tree conditions from the variational symmetric problem, then there exists a unique u_h that solves Ritz-Galerkin approximation.*

Proof. Since $(V^h, a(\cdot,\cdot))$ is a Hilbert space and $f|_{V^h} \in V^{h*}$, by Theorem 1 follows that exists a unique $u_h \in V^h$ that solves the Riesz-Galerkin approximation. ■

Proposition 2 (Galerkin fundamental orthogonality)**.** *Let u and u_h be solutions of problems* (1.2) *and* (1.3)*, respectively. Then $a(u - u_h, v) = 0$ for all $v \in V^h$.*

Proof. It is obtained simply by subtracting equations (1.2) and (1.3). ■

Corolary 1. *Let u and u_h be as in last proposition, then*

$$\|u - u_h\|_{\mathcal{H}} = \min_{v \in V^h} \|u - v\|_{\mathcal{H}}.$$

We have considered so far that $a(\cdot,\cdot)$ is symmetric, which is not always true. Lets consider the non-symmetric variational problem, for which we assume that the following conditions are valid,

- The pair $(\mathcal{H}, \langle\cdot,\cdot\rangle)$ is a Hilbert space.
- The subspace V is a closed subspace of $\mathcal{H}$.
- The bilinear form $a(\cdot,\cdot)$ is *not necessarily symmetric.*
- The bilinear form $a(\cdot,\cdot)$ is bounded and coercive in V.

Then the possibly non-symetric variational problem yields: given $f \in V^*$, find $u \in V$ such that

$$a(u, v) = f(v) \quad \text{for all } v \in V.$$

The Galerkin approximation is: given a finite dimensional subspace $V^h \in V$ y $f \in V^*$, find $u_h \in V^h$ such that

$$a(u_h, v) = f(v) \quad \text{for all } \ v \in V^h.$$

We want to know if there are solutions for this case, also if they are unique and what is the error estimate for $u - u_h$. Here the Lax-Milgram theorem guarantees the existence and uniqueness of solutions for the non-symmetric variational problem.

Theorem 4 (Lax-Milgram theorem)**.** *Let $a : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$ be a continuous and coercive bilinear form, and $f \in \mathcal{H}^*$. Then exists a unique element $u \in \mathcal{H}$ such that*

$$a(u, v) = \langle f, v\rangle \quad \textit{for all } v \in \mathcal{H}.$$

Proof. If we fix $u \in \mathcal{H}$, the map $v \mapsto a(u, v)$ is a linear functional in $\mathcal{H}$. By Ritz representation theorem exist a unique element $w \in \mathcal{H}$ such that

$$a(u, v) = \langle u, v\rangle, \quad \text{for all } v \in \mathcal{H}. \tag{1.4}$$

We write $Tu = w$ if equation (1.4) is fulfilled. Then

$$a(u, v) = \langle Tu, v\rangle, \quad \text{for } u, v \in \mathcal{H}. \tag{1.5}$$

Lets verify that $T : \mathcal{H} \to \mathcal{H}$ is linear, one-to-one and that $Im(T)$ is a closed subset in $\mathcal{H}$. Let $\lambda \in \mathbb{R}$, u_1, $u_2 \in \mathcal{H}$, for all $v \in \mathcal{H}$

$$\begin{aligned}
\langle T(\lambda u_1 + u_2), v\rangle &= a(\lambda u_1 + u_2, v)\\
&= \lambda a(u_1, v) + a(u_2, v)\\
&= \lambda\langle Tu_1, v\rangle + \langle Tu_2, v\rangle\\
&= \langle \lambda Tu_1 + Tu_2, v\rangle,
\end{aligned}$$

which means that T is linear. Since

$$\|Tu\|^2 = \langle Tu, Tu\rangle = a(u, Tu) \leq \alpha\|u\|\|Tu\|,$$

hence T is bounded. Since $a(\cdot, \cdot)$ is coercive and by Cauchy-Swarz inequality,

$$\gamma\|u\|^2 \leq a(u, u) = \langle Tu, u\rangle \leq \|Tu\|\|u\|,$$

hence T is one-to-one and $Im(T)$ is closed. Lets suppose that $Im(T) \neq \mathcal{H}$, then as $Im(T)$ is closed there exists an element $w \in \mathcal{H}$ non-null such that $w \in Im(T)^{\perp}$. Since $a(\cdot, \cdot)$ is coercive,

$$\gamma\|w\|^2 \leq a(w, w) = \langle Tw, w\rangle = 0,$$

yielding a contradiction. Hence $Im(T) = \mathcal{H}$.

By the Riesz representation theorem we see that for $w \in \mathcal{H}$,

$$\langle f, v\rangle = \langle w, v\rangle, \quad \text{for all } v \in \mathcal{H},$$

nevertheless since the operator T is bijective, we can find $u \in \mathcal{H}$ such that $Tu = w$. Then,

$$a(u, v) = \langle Tu, v\rangle = \langle w, v\rangle = \langle f, v\rangle, \quad \text{for all } v \in \mathcal{H},$$

which completes the proof. ■

The Lax-Milgram theorem assures us that the non-symmetric variational problem has a unique solution. If we apply this theorem to the discrete non-symmetric variational problem, we can conclude that this also has a unique solution in V^h. Note that V^h does not necessarily need to be finite dimensional, it just needs to be closed.

1.3 FEM Error - Lemma of Céa

To estimate the error $\|u - u_h\|$ we need the next lemma:

Lemma 1 (Lemma of Céa)**.** *Let $u \in \mathcal{H}$ be the solution of (1.2) and $u_h \in V^h$ with $V^h \subset \mathcal{H}$. Then*

$$\|u - u_h\|_{\mathcal{H}} \leq \min_{v \in V^h} \frac{\gamma}{\alpha} \|u - v_h\|_{\mathcal{H}},$$

where γ is the continuity constant and α is the coercivity constant of $a(\cdot,\cdot)$ in V.

Proof. We have that $a(u, v) = f(v)$ for all $v \in V$ and $a(u_h, v) = f(v)$ for all $v \in V^h$, if we subtract the two equations get

$$a(u - u_h) = 0 \quad \text{for all } v \in V^h.$$

For all $v \in V^h$, $a(\cdot,\cdot)$ is coercive,

$$\begin{aligned}\alpha\|u - u_h\|_V^2 &\leq a(u - u_h, u - u_h) \\ &= a(u - u_h, u - u_h + v - v) \\ &= a(u - u_h, u - v) + a(u - u_h, v - u_h),\end{aligned}$$

since $v - u_h \in V^h$,

$$= a(u - u_h, u - v),$$

and since $a(\cdot,\cdot)$ is continuous,

$$\leq \gamma\|u - u_h\|_V \|u - v\|_V.$$

Which gives,

$$\|u - u_h\|_V \leq \frac{\gamma}{\alpha}\|u - v\|_V, \qquad \text{for all } v \in V^h.$$

The previous inequality is valid for all $v \in V^h$, and therefore,

$$\|u - u_h\|_V \leq \frac{\gamma}{\alpha} \inf_{v \in V^h} \|u - v\|_V = \frac{\gamma}{\alpha} \min_{v \in V^h} \|u - v\|_V,$$

since V^h is closed. ■

The lemma of Céa gives us a bound for the error of our discretized problem. We can notice that the error is almost optimal, since the error is proportional to the best bound that can be obtained using the subspace V^h. In the case in which the bilinear form is symmetric, it was shown that

$$\|u - u_h\|_V = \min_{v \in V^h} \|u - v\|_V.$$

For instance, let us take the heat equation over $\Omega \subset \mathbb{R}^n$, with Dirichlet boundary conditions,

$$-\Delta u = f \quad \text{in } \Omega, \quad u|_{\partial\Omega} = 0.$$

We can rephrase last equation as a variational problem over $V = \{v \in H^1(\Omega) : \ v|_{\partial\Omega} = 0\}$. The problem is to find $u \in V$ such that

$$\int_\Omega \nabla u \cdot \nabla v \, dx = \int_\Omega fv \, dx, \quad \text{for all } v \in V.$$

Recall that we can get this equivalent problem, know as variational formulation, by multiplying a test function v and integrating, then we use Green identity to reduce derivative order on u. Using the boundary condition we ditch the integral term over $\partial\Omega$.

If we define the bilinear form $a(u, v) = \int_\Omega \nabla u \cdot \nabla v \, dx$ and the functional $f(v) = \int_\Omega fv \, dx$, then we can write the problem in abstract form such as: find $u \in V$ such that

$$a(u, v) = f(v), \quad \text{for all } v \in V. \tag{1.6}$$

We can verify that the bilinear form and the functional both fulfill the conditions from Lax-Milgram theorem, then we know that problem (1.6) has a unique solution called $\tilde{u}$. We can apply the Galerkin method to approximate $\tilde{u}$ and estimate the errors made using the lemma of Céa. To apply the Galerkin method we take the finite dimension subspace $V^h \subset \Omega$, generated by a set of basis functions and we solve then the variational problem of Galerkin: find $u_h \in V^h$ such that

$$a(u_h, v) = f(v), \quad \text{for all } v \in V^h. \tag{1.7}$$

Let $\{\phi_i\}_{i=1}^n$ be a basis of V^h and $\{\widehat{\phi}_i\}_{i=1}^n$ a basis of $\widehat{V^h}$, we can write u_h as a linear combination,

$$u_h = \sum_{j=1}^n \alpha_j \phi_j,$$

also, lets take $v = \widehat{\phi}_i$ for $i = 1, 2, \ldots, n$. It follows that

$$\sum_{j=1}^n \alpha_j a(\phi_j, \widehat{\phi}_i) = f(\widehat{\phi}_i) \quad \text{for } i = 1, 2, \ldots, n.$$

The weights in the finite element solution u_h can be computed by solving the linear system $\mathbf{A}\vec{\alpha} = \vec{b}$, where

$$a_{ij} = a(\phi_j, \widehat{\phi}_i), \quad i, j = 1, 2, \ldots, n \quad \text{and} \quad b_i = f(\widehat{\phi}_i), \quad i = 1, 2, \ldots, n.$$

To get $\mathbf{A}$ and $\vec{b}$ we usually do not compute the exact value of the integrals involved, we instead approximate the value of the integrals using numerical methods such as quadratures. It is here that we begin to generate extra errors that is known as a variational crime.

1.4 The Strang Lemmas

Solving problem (1.7) usually requires the use of some numerical approximation to compute the integrals involved. We are approximating the bilinear form $a(\cdot, \cdot)$ by a discrete version called $a_h(\cdot, \cdot)$. In the same way we define a discrete functional $f_h(\cdot)$ over V^h. Note that here we should also consider precision errors due to the computer calculations, see [27].

On the other hand, if the domain Ω is not polygonal, it cannot be triangulated in an exact way. Then the triangulation of the finite element method is only an approximation of Ω. Here the functions are defined in spaces V^h that are different from the space V and such that $V^h \not\subset V$, we are again generating more approximation errors in these calculations, see [27].

Being the finite element method one of the most popular choices to solve partial differential equations in applied problems, it is common not to take these errors into consideration. That is, not taking care of the difference between using the discrete analogs of the bilinear and functional forms when applying the Galerkin method. Then, the error analysis showed previously for this method is no longer valid.

Now we show the Strang lemmas, in which these errors are taken into account in the calculation of the error bound for the method. The Strang lemmas are a generalization of the lemma of Céa, under the same conditions of the problems (1.2) and (1.3). For the first lemma, it is not necessary for a_h to be defined for all the functions in $\mathcal{H}$, in particular, we can evaluate a_h using quadrature methods for example. In this case, it is still necessary that $V^h \in \mathcal{H}$.

Definition 2 (Uniform V^h-ellipticity). *The sequence $\{a_h(\cdot,\cdot)\}$ with $h \in H$ (H a positive real number sequence that converges to zero) of discrete bilinear forms $a_h(\cdot,\cdot) : V^h \times V^h \to \mathbb{R}$ is called uniformly V^h-elliptic, if there exist a positive constant $\widehat{\alpha}$ such that,*

$$a_h(u_h, u_h) \geq \widehat{\alpha}\|u_h\|^2_{\mathcal{H}} \qquad u_h \in V^h,$$

uniformly for all $h \in H$. See [26].

If we assume the uniform V^h-ellipticity, the variational equation (1.3) admits unique solutions $u - h \in V^h$. The *First lemma of Strang* gives us a bound for the generalized error from the lemma of Céa, see more in [26].

Lemma 2 (First lemma of Strang). *Assume that $\{a_h(\cdot,\cdot)\}$ is a uniformly V^h-elliptic family of bilinear forms. Let $u \in \mathcal{H}$ and $u_h \in V^h$ with $h \in H$ be the unique solutions of (1.2) and (1.3) respectively. Then exists a constant $C \in \mathbb{R}^+$, independent of $h \in H$ such that*

$$\|u - u_h\|_{\mathcal{H}} \leq C \left(\inf_{v_h \in V^h} \left(\|u - v_h\|_{\mathcal{H}} + \sup_{w_h \in V^h} \frac{|a(v_h, w_h) - a_h(v_h, w_h)|}{\|w_h\|_{\mathcal{H}}} \right) + \sup_{w_h \in V^h} \frac{|f(w_h) - f_h(w_h)|}{\|w_h\|_{\mathcal{H}}} \right).$$

Proof. Let $v_h \in V^h$, and take $u_h - v_h = w_h$, given that $a_h(\cdot,\cdot)$ is coercive

$$\begin{aligned}
\alpha\|u_h - v_h\|^2 &\leq a_h(u_h - v_h, u_h - v_h) \\
&= a_h(u_h - v_h, w_h) \\
&= a_h(u_h, w_h) - a_h(v_h, w_h) \\
&= a_h(u_h, w_h) - a_h(v_h, w_h) + a(u - v_h, w_h) - a(u - v_h, w_h) \\
&= a(u - v_h, w_h) + [a(v_h, w_h) - a_h(v_h, w_h)] + [a_h(v_h, w_h) - a(u, w_h)] \\
&= a(u - v_h, w_h) + [a(v_h, w_h) - a_h(v_h, w_h)] + [f_h(w_h) - f(w_h)] \\
&\leq |a(u - v_h, w_h)| + |a(v_h, w_h) - a_h(v_h, w_h)| + |f(w_h) - f_h(w_h)|
\end{aligned}$$

and since $a(\cdot,\cdot)$ is bounded,

$$\leq \gamma\|u - v_h\|\|w_h\| + |a(v_h, w_h) - a_h(v_h, w_h)| + |f(w_h) - f_h(w_h)|,$$

dividing by $\alpha\|u_h - v_h\|$, we get that

$$\|u_h - v_h\| \leq \frac{\gamma}{\alpha}\|u - v_h\| + \frac{1}{\alpha}\frac{|a(v_h, w_h) - a_h(v_h, w_h)|}{\|w_h\|} + \frac{1}{\alpha}\frac{f(w_h) - f_h(w_h)}{\|w_h\|}.$$

Since v_h is an arbitrary element of V^h, and using triangular inequality

$$\|u - u_h\| \leq \|u - v_h\| + \|u_h - v_h\|,$$

it follows that,

$$\|u - u_h\| \leq \left(1 + \frac{\gamma}{\alpha}\right)\|u - v_h\| + \frac{1}{\alpha}\frac{|a(v_h, w_h) - a_h(v_h, w_h)|}{\|w_h\|} + \frac{1}{\alpha}\frac{f(w_h) - f_h(w_h)}{\|w_h\|}.$$

Since v_h is arbitrary, we take each element bound as the worst case scenario, hence,

$$\|u - u_h\|_{\mathcal{H}} \leq C\left(\inf_{v_h\in V^h}\left(\|u - v_h\|_{\mathcal{H}} + \sup_{w_h\in V^h}\frac{|a(v_h, w_h) - a_h(v_h, w_h)|}{\|w_h\|_{\mathcal{H}}}\right) + \sup_{w_h\in V^h}\frac{|f(w_h) - f_h(w_h)|}{\|w_h\|_{\mathcal{H}}}\right).$$

■

The next lemma, often referred to as the second lemma of Strang, gives us a bound for the error if the conditions for the first lemma are fulfilled, but $V^h \not\subset \mathcal{H}$. This is the case of non-conforming finite elements.

Since $V^h \not\subset \mathcal{H}$ we cannot use the induced norm of $\mathcal{H}$ in all the elements of V^h, so it is convenient to use mesh-dependent norms, which are denoted as $\|\cdot\|_h$, see [26].

Definition 3 (Mesh-dependent norm)**.** *Given τ_h a partition of Ω, we define*

$$\|v\|_h = \sqrt{\sum_{Q_j\in\tau_h}\|v\|^2_{Q_j}}.$$

Lemma 3 (Lemma of Berger, Scott and Strang)**.** *Assume that $\{a_h(\cdot,\cdot)\}$ is a family of uniformly V^h-elliptic of bilinear forms. Let $u \in \mathcal{H}$ and $u_h \in V^h$ with $h \in H$ unique solutions from (1.2) and (1.3). There exists a constant $C \in \mathbb{R}^+$ independent of $h \in H$ such that*

$$\|u - u_h\|_h \leq C\left(\inf_{v_h\in V^h}\|u - v_h\|_h + \sup_{w_h\in V^h\setminus\{0\}}\frac{|a_h(u, w_h) - f_h(w_h)|}{\|w_h\|_h}\right).$$

Proof. Following [26], let $v_h \in V^h$,

$$\alpha\|u - u_h\|_h \leq \|u - v_h\|_h + \|v_h - u_h\|_h,$$

since $a_h(\cdot,\cdot)$ is coercive,

$$\begin{aligned}
\alpha\|u-u_h\|_h &\le \|u-v_h\|_h + \frac{1}{\gamma}\sup_{w_h\in V^h\setminus 0}\frac{|a_h(v_h-u_h,w_h)|}{\|w_h\|_h}\\
&= \|u-v_h\|_h + \frac{1}{\gamma}\sup_{w_h\in V^h\setminus\{0\}}\frac{|a_h(v_h-u_h,w_h)|}{\|w_h\|_h}\\
&= \|u-v_h\|_h + \frac{1}{\gamma}\sup_{w_h\in V^h\setminus\{0\}}\frac{|a_h(v_h-u_h,w_h)+a_h(u,w_h)-a_h(u,w_h)|}{\|w\|_h}\\
&= \|u-v_h\|_h + \frac{1}{\gamma}\sup_{w_h\in V^h\setminus\{0\}}\frac{|a_h(v_h,w_h)-a_h(u_h,w_h)+a_h(u,w_h)-a_h(u,w_h)|}{\|w\|_h}\\
&= \|u-v_h\|_h + \frac{1}{\gamma}\sup_{w_h\in V^h\setminus\{0\}}\frac{|a_h(v_h-u,w_h)+a_h(u-u_h,w_h)|}{\|w_h\|_h}\\
&= \|u-v_h\|_h + \frac{1}{\gamma}\sup_{w_h\in V^h\setminus\{0\}}\frac{|a_h(v_h-u,w_h)|}{\|w_h\|_h} + \frac{1}{\gamma}\sup_{w_h\in V^h\setminus\{0\}}\frac{|a_h(u-u_h,w_h)|}{\|w_h\|_h}
\end{aligned}$$

because $a_h(\cdot,\cdot)$ is continuous,

$$\begin{aligned}
\alpha\|u-u_h\|_h &\le \|u-v_h\|_h + \frac{\alpha}{\gamma}\sup_{w_h\in V^h\setminus\{0\}}\frac{\|v_h-u\|_h\|w_h\|_h}{\|w_h\|_h} + \frac{1}{\gamma}\sup_{w_h\in V^h\setminus\{0\}}\frac{|a_h(u-u_h,w_h)|}{\|w_h\|_h}\\
&\le \|u-v_h\|_h + \frac{\alpha}{\gamma}\|v_h-u\|_h + \frac{1}{\gamma}\sup_{w_h\in V^h\setminus\{0\}}\frac{|a_h(u-u_h,w_h)|}{\|w_h\|_h}\\
&= \left(1+\frac{\alpha}{\gamma}\right)\|u-v_h\|_h + \frac{1}{\gamma}\sup_{w_h\in V^h\setminus\{0\}}\frac{|a_h(u-u_h,w_h)|}{\|w_h\|_h}.
\end{aligned}$$

Thus for an arbitrary v_h,

$$\begin{aligned}
\|u-u_h\|_h &\le \left(1+\frac{\alpha}{\gamma}\right)\inf_{v_h\in V^h}\|u-v_h\|_h + \frac{1}{\gamma}\sup_{w_h\in V^h\setminus\{0\}}\frac{|a_h(u-u_h,w_h)|}{\|w_h\|_h}\\
&\le C\left(\inf_{v_h\in V^h}\|u-v_h\|_h + \sup_{w_h\in V^h\setminus\{0\}}\frac{|a_h(u-u_h,w_h)|}{\|w_h\|_h}\right).
\end{aligned}$$

■

Lemma 4. *If the conditions of the previous lemma are fulfilled with dim $V^h<\infty$ and also the bilinear form $a_h(\cdot,\cdot)$ is positive-definite, with*

$$\|v\|_a := \sqrt{a_h(v,v)},$$

then we can improve last bound as,

$$\|u-u_h\|_a \le \inf_{v_h\in V^h}\|u-v_h\|_a + \sup_{w_h\in V^h}\frac{|a_h(u-u_h,w_h)|}{\|w_h\|_a}.$$

In equation (3), the first term is known as *the approximation error*, while the second term is known as *the consistency error*, for more details see [26].

The approximation error comes from the discretization of the function space. This error is the distance between u, which is the exact solution of the original variational problem, to V^h space. This error comes from the construction of the finite dimensional space and measures how well this space captures the solution u. The consistency error comes from the discretization of the variational equation. When working on the space V^h we actually use the discrete analogs of the bilinear form $a(\cdot,\cdot)$ and the functional f. This is why in general the exact solution of the variational equation is not a solution to the discrete variational problem. This error measures how well the solution u satisfies the discrete variational problem, from [40].

1.5 The Heat Equation

Understanding heat conductivity is very important for many practical applications such as the design of composite materials, among many others. There are several aspects related to heat conduction that shall be taking into account when dealing with applications. Roughly speaking, heat flows from hotter to colder parts of a body, which is known as Fourier's law. More precisely, the heat conduction law states that heat flux transfer through a material is proportional to the negative of the temperature gradient. This proportion is given by a coefficient called thermal conductivity that depends on the material. In some applications, it is important to design a material with optimal thermal conductivity. Also important in many applications is the understanding and efficient design of material with optimal elastic properties. In both cases, these optimal design problems can be modeled by using optimization methodologies where the cost functional is computed over solutions of partial differential equations.

In this section, we review the mathematical and numerical tools concerning some optimality problems related to the design of thermal conductivities which are similar to the tools used for the design of elasticity tensors and other related problems in topology optimization.

We now present in some more detail the time-independent heat equation that is needed in order to write the problem for the optimal design of the conductivity coefficient.

Here we work with the flow equation for *stationary heat conduction* over a domain $\Omega \subset \mathbb{R}^2$ with boundary $\partial\Omega$, see [34]. Let $\vec{q} = [q_x, q_y]^\intercal$ denote the heat flux and $f = f(x, y)$ a source term. The energy conservation law states that

$$\operatorname{div}(\vec{q}) = f, \tag{1.8}$$

and the Fourier heat conduction equation says that,

$$\vec{q} = -\mathbf{K}\nabla u, \tag{1.9}$$

where $\mathbf{K}$ is the thermal conductivity matrix and $u = u(x, y)$ is the temperature at the $(x, y) \in \Omega$ point. See Figure 1.1 for an illustration.

Merging equations (1.8) and (1.9), that is, eliminating the flux we get

$$-\operatorname{div}(\mathbf{K}\nabla u) = f. \tag{1.10}$$

This is a second order partial differential equation that describes the distribution of temperature over a domain Ω.

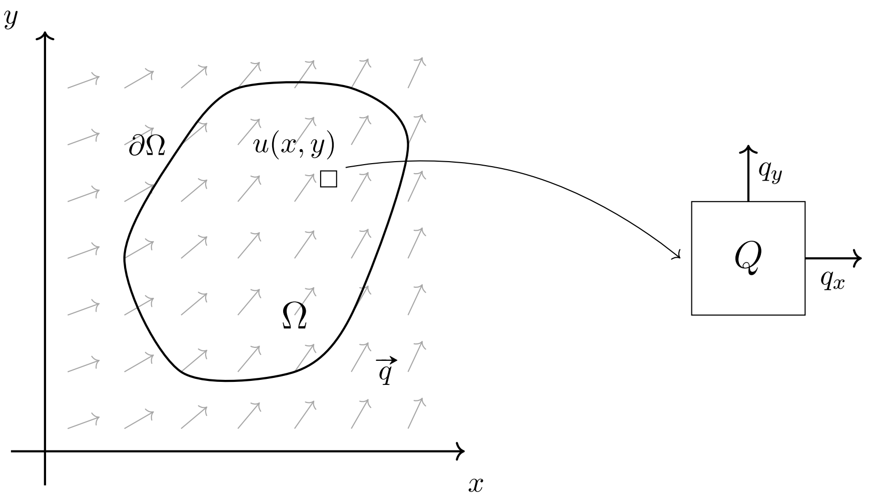

Figure 1.1: Mass conservation law over an infinitesimal element $Q \subset \Omega \subset \mathbb{R}^2$. In the right we have the decomposition of the flux $\vec{q}$ in its vertical and horizontal component. Also note that we have $\operatorname{div}\vec{q} = \partial_x q_x + \partial_y q_y = f$ over Q. Modified from [34].

Depending on the material, the thermal conductivity matrix $\mathbf{K}$ has different properties, see [34]. In the case the material being *isotropic*, that is, when heat distributes evenly over the direction of the plane, e.g., glass, metal, the conductivity matrix is given by a multiple of the identity matrix,

$$\mathbf{K} = \begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix} = k \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

When heat principal distribution directions are in the same directions of the orthogonal axes, we have an *orthotropic* material, e.g., wood, composite materials, etc. In this we have

$$\mathbf{K} = \begin{bmatrix} k_x & 0 \\ 0 & k_y \end{bmatrix}.$$

The most general situation is the case of an *anisotropic* material, when principal heat distribution directions does not coincide with the orthogonal axes,

$$\mathbf{K} = \begin{bmatrix} k_{xx} & k_{xy} \\ k_{yx} & k_{yy} \end{bmatrix}.$$

Now we must set our boundary conditions in order to complete the stationary heat equation formulation. We can consider the following cases[1]:

- When we have a prescribed temperature $u(x,y) = u_*(x,y)$ on the boundary $\partial\Omega$. This boundary condition is known as a *Dirichlet boundary condition.*
- A prescribed heat influx at the boundary $\vec{q}(x,y) = -\vec{q}_*(x,y) \cdot \vec{\eta}$, where $\vec{\eta}$ is the normal direction over $\partial\Omega$ is known as a *Neumann boundary condition.*
- A mixed boundary condition is when we have a Dirichlet boundary condition in part of $\partial\Omega$ and a Neumann boundary condition on the rest of $\partial\Omega$.

[1]There are other types of boundary conditions but we are just interested only in these types.

1.6 FEM for the Heat Equation in two Dimensions

Let us consider a Dirichlet boundary condition with an isotropic material with conductivity $\mathbf{K}$. Then, our formulation is to find $u : \Omega \to \mathbb{R}$ such that $u \in H^1(\Omega)$ we have[2]

$$\begin{cases} -\operatorname{div}(\mathbf{K}\nabla u(x,y)) = f(x,y) & (x,y) \in \Omega, \\ u(x,y) = \beta(x,y) & (x,y) \in \partial\Omega. \end{cases} \tag{1.11}$$

We use a test function $v : \Omega \to \mathbb{R}$ with $v \in H^1(\Omega)$ such that $v(x) = 0$ for all $x \in \partial\Omega$. Then, we multiply by v on both sides of equation (1.11),

$$-\operatorname{div}(\mathbf{K}\nabla u(x,y))v(x,y) = f(x,y)v(x,y).$$

To simplify notation we omit (x,y). Now, we integrate over Ω to get

$$-\int_\Omega \operatorname{div}(\mathbf{K}\nabla u)v = \int_\Omega fv,$$

and by using Green's identity we obtain

$$\int_\Omega (\mathbf{K}\nabla u)\nabla v + \int_{\partial\Omega} (\mathbf{K}\nabla u)v\vec{\eta} = \int_\Omega fv,$$

with $\vec{\eta}$ the normal vector on $\partial\Omega$. Using the fact that $v|_{\partial\Omega} = 0$ we have

$$\int_\Omega (\mathbf{K}\nabla u)\nabla v = \int_\Omega fv.$$

Now the weak formulation of the problem is to find $u : \Omega \to \mathbb{R}$ such that

$$\begin{cases} \int_\Omega (\mathbf{K}\nabla u)\nabla v = \int_\Omega fv & \text{for all} \quad v \in H_0^1(\Omega), \\ u(x,y) = \beta(x,y) & (x,y) \in \partial\Omega. \end{cases} \tag{1.12}$$

Here we have defined $H_0^1(\Omega)$ as the subspace of functions $v \in H^1(\Omega)$ with $v|_{\partial\Omega} = 0$. Note that if we define

$$a(u,v) = \int_\Omega (\mathbf{K}\nabla u)\nabla v, \text{ and } l(v) = \int_\Omega fv,$$

we see that $a(u,v)$ is a bilinear form and $l(v)$ is a linear functional.

To obtain the Galerkin formulation of the problem we need to take $v \in V^h$ where V^h a finite dimensional space. Here $h > 0$ is a discretization parameter. We also need to choose a basis for V^h, say $\{\varphi_1, \varphi_2, \ldots, \varphi_k\}$ with k the dimension of V^h. We can approximate the temperature u with u_h over the domain using a linear combination of basis functions and write $u_h = \sum_{j=1}^k \alpha_j \varphi_j$, with α_j unknown coefficients. Within Galerkin's formulation we should find $u_h \in V^h$ such that

$$a(u_h, v_h) = l(v_h) \qquad \text{for all } v_h \in V^h. \tag{1.13}$$

We get a linear system of size $k \times k$,

$$\mathbf{A}\vec{\alpha} = \vec{b},$$

[2]Here we use the standard definition of Sobolev spaces. In particular $u \in H^1(\Omega)$ means that u and ∇u are square integrable over Ω, for a detailed definition see [18].

where $A_{ij} = a(\varphi_i, \varphi_j)$, with $i, j = 1, \ldots, k$. And $b_j = l(\varphi_j)$ for $j = 1, \ldots, k$, for further details see [19].

For the construction of the matrix system, now suppose that we have a polygonal domain Ω and let τ^h be a polygonal partition of Ω which is made of square elements Q_1, Q_2, …, Q_m with $m \in \mathbb{N}$, see Figure 1.2. Each Q_i is an open square such that

$$\Omega = \bigcup_{i=1}^{m} \overline{Q_i}, \qquad \text{and} \qquad Q_i \bigcap Q_j = \emptyset,$$

to avoid irregularities on mesh squares we need that,

$$\overline{Q_i} \bigcap \overline{Q_j} = \begin{cases} \overline{Q_i} \text{ or } \overline{Q_j} & \text{if} \quad i = j, \\ \text{an entire side of } \overline{Q_i} \text{ and } \overline{Q_j} & \text{if} \quad i \neq j, \\ \text{one vertex of } \overline{Q_i} \text{ and } \overline{Q_j} & \text{if} \quad i \neq j, \\ \text{or } \emptyset & \text{if} \quad i \neq j. \end{cases}$$

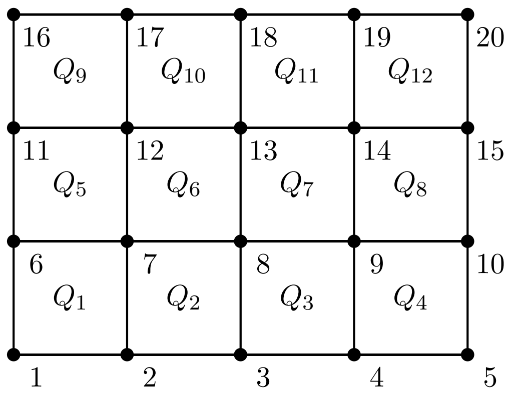

Figure 1.2: Rectangular domain Ω with a partition in square elements Q_i for $i = 1, \ldots, 20$ vertices over the mesh.

Consider the space of linear continuous functions over τ^h,

$$P^1(\tau^h) = \left\{ v : \Omega \to \mathbb{R} \;\middle|\; \begin{array}{ll} v & \text{continuous function in } \Omega, \\ v|_{Q_i} & \text{linear function,} \\ v(x, y) = \beta(x, y) & \text{if } (x, y) \in \partial\Omega, \end{array} \right\}$$

let $I_\Omega = \{i \in \{1, \ldots, k\} | \quad \vec{x}_i \in \Omega\}$ and $I_{\partial\Omega} = \{l \in \{1, \ldots, k\} | \quad \vec{x}_l \in \partial\Omega\}$, be the sets of indexes for the basis functions over Ω and $\partial\Omega$. We are searching for $u_h \in P^1(\tau^h)$ with $u_h = \beta$ in $\partial\Omega$ which can be written as a linear combination of basis functions

$$u = \sum_{i \in I_\Omega} \alpha_i \varphi_i + \sum_{l \in \partial\Omega} \beta_l \varphi_l,$$

with α_i unknown constants for $i \in 1, 2, \ldots, k$. Therefore the gradient for u_h is,

$$\nabla u_h = \nabla \left(\sum_{i \in I_\Omega} \alpha_i \varphi_i + \sum_{l \in \partial\Omega} \beta_l \varphi_l \right) = \sum_{i \in I_\Omega} \alpha_i \nabla \varphi_i + \sum_{l \in \partial\Omega} \beta_l \nabla \varphi_l.$$

Replacing this in equation into (1.12) we obtain

$$\int_{\Omega} K \left(\sum_{i \in I_{\Omega}} \alpha_i \nabla \varphi_i + \sum_{l \in \partial\Omega} \beta_l \nabla \varphi_l \right) \nabla v = \int_{\Omega} f v.$$

Separating the first integral and rearranging terms we get

$$\int_{\Omega} K \left(\sum_{i \in I_{\Omega}} \alpha_i \nabla \varphi_i \right) \nabla v = \int_{\Omega} f v - \int_{\Omega} \sum_{l \in \partial\Omega} K \left(\beta_l \nabla \varphi_l \right) \nabla v.$$

By the linearity of the integral we can write

$$\sum_{i \in I_{\Omega}} \int_{\Omega} K \left(\alpha_i \nabla \varphi_i \right) \nabla v = \int_{\Omega} f v - \sum_{l \in \partial\Omega} \int_{\Omega} K \left(\beta_l \nabla \varphi_l \right) \nabla v.$$

Finally, by taking $v = \varphi_j$ with $j \in I_{\Omega}$ in the last equation, we get the matrix formulation

$$\sum_{i \in I_{\Omega}} \int_{\Omega} K \left(\alpha_i \nabla \varphi_i \right) \nabla \varphi_j = \int_{\Omega} f \varphi_j - \sum_{l \in \partial\Omega} \int_{\Omega} K \left(\beta_l \nabla \varphi_l \right) \nabla \varphi_j.$$

Define the following matrices and vectors,

$$\begin{aligned} a_{ij} &= \int_{\Omega} K \left(\nabla \varphi_i \right) \nabla \varphi_j, \\ f_j &= \int_{\Omega} f \varphi_j, \end{aligned}$$

and

$$b_j = \int_{\Omega} K \left(\beta_l \nabla \varphi_l \right) \nabla \varphi_j.$$

We can write this as a linear system,

$$\mathbf{A} \vec{\alpha} = \vec{f} - \vec{b}, \tag{1.14}$$

that is usually solved using an iterative method due to the size and sparseness of the matrix. We shall extend the study of these methods in the next chapters.

The main issues when solving the linear system (1.14) are the size and condition number of the matrix $\mathbf{A}$, see [22]. In the problems we are interested the coefficient $\mathbf{K}$ has a multiscale structure, i.e. it varies in a significant way across the domain Ω at different scales. The coefficient has also a high-contrast. This is when the contrast $\eta = \mathbf{K}_{\max}/\mathbf{K}_{\min}$ is large when compared to the coarse-grid size, see [22].

The performance of numerical methods for this type of problems depends on η and on the multiscale structure of the coefficient $\mathbf{K}$, see for instance Figure 1.3, for a coefficient with a complicated structure.

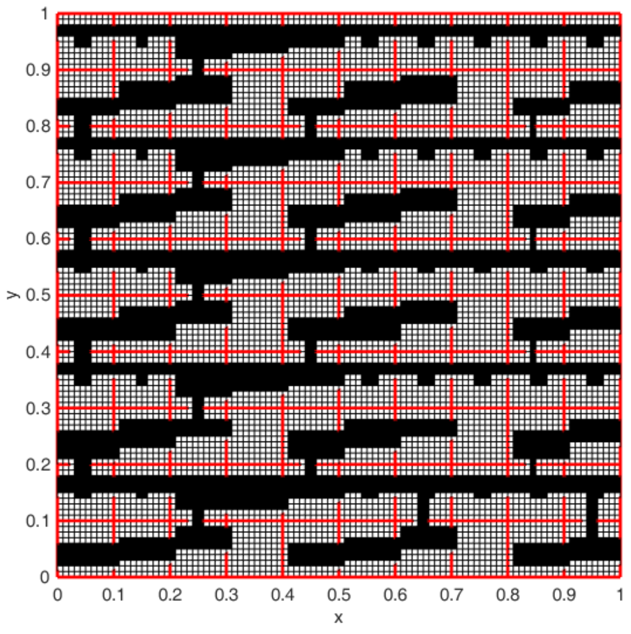

Figure 1.3: Conductivity coefficient, note the high-contrast channels in black that cross coarse elements. The coarse mesh is 10×10 and it is shown in red. The fine mesh is 10×10 inside each coarse element.

We can see in Table 1.1 the results of solving problem (1.11) with $\beta = 0$ and f constant. Using $\mathbf{K}$ as in Figure 1.3 with high-contrast channels in black, in which $\mathbf{K}_{ij} = 1$, and the low conductivity areas in white, where $\mathbf{K}_{ij} = 1 \times 10^{p}$ for $p = 0, -2, -4$ and -6. Table 1.1 shows the number of iterations that the conjugate gradient method needed to converge using a tolerance of 1×10^{-6}. It also shows the spectral condition of the matrix $\mathbf{A}$. Note how iterations and spectral condition increase along with the coefficient contrast η.

Contrast	Iterations	Spectral condition
1	112	2×10^{3}
1×10^{-2}	768	1.7×10^{4}
1×10^{-4}	2961	3.8×10^{5}
1×10^{-6}	7760	3.7×10^{7}

Table 1.1: Heat equation using contrast from Figure 1.3, without preconditioners and with a constant heat source.

The option to avoid these problems is to choose a grid size that is fine enough to capture the multiscale variations of $\mathbf{K}$. Note that the size of $\mathbf{A}$ is proportional to h^{-2} being h the size of the elements and if we choose h to be small we need to solve a large problem. The condition number of $\mathbf{A}$ is proportional to ηh^{-2} meaning that we also have an ill-conditioned system. This is why is necessary to explore other approaches to the problem in order to give a better numerical solution.

Now we make a small comparison of state stationary heat conduction and elasticity in two dimensions. We begin by looking to the variables that have a relation between heat and elasticity formulations.

Let $(x, y) \in \mathbb{R}^2$,

Stationary heat conduction variables:

- $\vec{q}(x, y)$: heat flux,
- $u(x, y) \to \mathbb{R}$: temperature,
- $f(x, y) \to \mathbb{R}$: forcing term,
- $\mathbf{K}$: conductivity matrix, dimension 2×2.

Elasticity variables:

- $\sigma(u)$: stress tensor, dimension 2×2,
- $\vec{u}(x, y)$: displacement,
- $\varepsilon(u)$: strain tensor, dimension 2×2,
- E: stiffness tensor, dimension $2\times2\times2\times2$.

We now state stationary heat conduction and elasticity equations together in Table 1.2 to see its similarities, here we assume a regular Dirichlet boundary condition.

	Heat conduction (2D)	**Elasticity (2D)**
Mass conservation law	$\operatorname{div}(\vec{q}) = f$	$\operatorname{div}(\sigma) = \vec{F}$
Constitutive law	$\vec{q} = -\mathbf{K}\nabla u$	$\sigma = -E\varepsilon(u)$
State equation	$-\operatorname{div}(\mathbf{K}\nabla u) = f$	$-\operatorname{div}(E\,\varepsilon(u)) = \vec{F}$
Weak form	$\int_\Omega (\mathbf{K}\nabla u)\nabla v = \int_\Omega fv$	$\int_\Omega (E\,\varepsilon(\vec{u})) : \epsilon(\vec{v}) = \int_\Omega \vec{F} : \vec{v}$
Galerkin formulation	$\sum_{i\in I_\Omega} \int_\Omega K\,(\alpha_i \nabla\varphi_i)\,\nabla\varphi_j = \int_\Omega f\varphi_j$	$\sum_{i\in I_\Omega} \int_\Omega E\,(\alpha_i \varepsilon(\varphi_i))\,\varepsilon\varphi_j = \int_\Omega \vec{F}\varphi_j$
Matrix formulation	$\mathbf{A}\vec{\alpha} = \vec{b}$	$\mathbf{A_E}\vec{\alpha} = \overrightarrow{b_E}$

Table 1.2: Comparison of stationary heat conduction and elasticity equations in two dimensions.

Note that the variable for which we are solving the equations in heat is temperature, this means that it has just one degree of freedom per node. In elasticity the variable is displacement, which is expressed as a vector, this means that in a two dimensional problem the elasticity formulation has two degrees of freedom per node. The idea is to use the fact that the heat problem has an smaller dimension than elasticity and give a solution to the elasticity equation using a double heat problem. This is the main detail between heat and elasticity problems that makes attractive the idea of using heat problems to try to solve elasticity problems with less cost. In the next chapters we study in a detailed manner this methods for the heat equation in order to apply this ideas in the elasticity equation as it is done in [35] and [36].

Chapter 2

Multiscale Methods for the Heat Equation

In topology optimization, for instance, the linear system obtained from the finite element method is solved on each iteration of the optimization algorithm. Using the density formulation proposed by [3] to solve the topology optimization problem, we see that in each step of the optimization the coefficient $k(x)$ varies in multiple scales. In industrial 3D topology optimization problems, the discretization process often leads to a linear system of equations with several million degrees of freedom, see [30].

Having into account these applications and several others, optimal iterative solvers must be used to obtain the solution of the state equations. In order to study this big and highly heterogeneous problems, a common technique is to use multiscale methods to get a fast approximation of the solution in a coarse grid. This in exchange of losing precision of the fine-scale features of the problem. To account for these problems we use local corrections as in [17]. Also in composite materials applications, low, or high-stiffness regions can have complex geometries at very small scales which leads to a more ill-conditioned linear system, see [10] and [7].

In this chapter, we work for the heat and elasticity equations following the introduction construction in [16] for the MsFEM and [10] for the GMsFEM part.

2.1 Multiscale FEM

Let consider the following linear elliptic equation

$$Lu = f \text{ in } \Omega,$$

where Ω is a polygonal domain in $\mathbb{R}^{\mathrm{d}}$ $(d = 2, 3)$, $Lu := -\operatorname{div}(k(x)\nabla u)$, and $k(x)$ is an heterogeneous field varying in multiple scales. Take a coarse grid partition $\mathcal{T}^H = \{Q\}$ of Ω, where H denotes the size of the coarse-grid partition. Here we remark that the coarse grid does not necessarily resolve the variations of the coefficient $k(x)$, see Figure 1.3. We regard that for the topology optimization case,

$$0 < k_0 \leq k(x) \leq 1, \quad \text{with} \quad k_0 \ll 1.$$

Let x_i be the interior nodes of the coarse mesh $\mathcal{T}^H$ and ϕ_i^H be the nodal basis of the standard finite element space $W_H = \text{span}\{\phi_i^H\}$. We define special basis functions known as multiscale finite element basis functions. Let us consider that W_H consist of piecewise linear functions on a coarse grid. Denote by $S_i = \sup(\phi_i^H)$ and define ϕ_i with support in S_i as follows

$$L\phi_i = 0 \text{ in } Q, \quad \phi_i = \phi_i^H \text{ on } \delta Q, \quad \text{for all } Q \in \mathcal{T}^H,\ Q \subset S_i.$$

The multiscale basis functions coincide with standard finite element basis functions on the boundaries of a coarse-grid block Q, and are oscillatory in the interior of each coarse-grid block, see [16] and Figure 2.1.

The MsFEM consists in projecting the solution on the space $V_{0,ms} = \text{span}\{\phi_i\}$, which is equivalent to solve a coarse linear system, that is, a linear system whose dimension is that of $V_{0,ms}$.

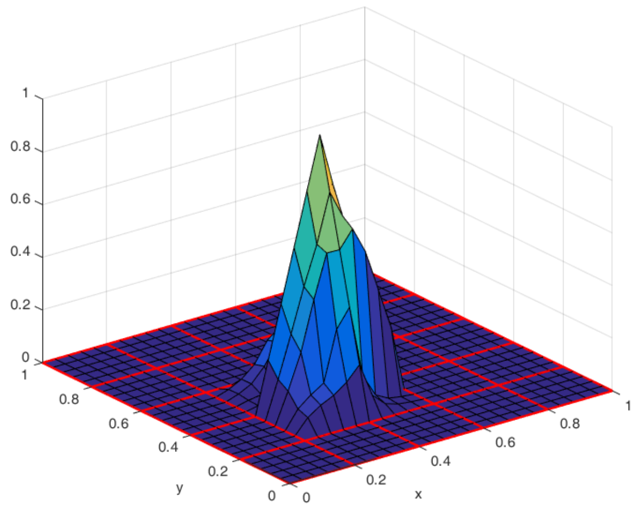

Figure 2.1: Multiscale basis function ϕ for the node $(3, 3)$ in a 5×5 elements coarse mesh, in red. Using coefficient from Figure 1.3 on an smaller 5×5 coarse mesh.

We mention that in topology optimization problems it is usual to obtain designs with a multiscale structure, as in [30]. This, together with the fact that the regularization parameter from the density formulation k_{min} is small, where $k(x) = (k_{min} + (k_0 - k_{min})\rho^p)$, it makes the solution of the linear system of the coefficient proposal quite challenging since performance of classical solvers for the elasticity equation with such material properties is negatively affected by the multiscale structure of the coefficient as well as the high-contrast of the media properties, see [30].

The classical MsFEM can be improved in several ways, see [13]. Special attention has to be taken to choose proper boundary conditions, see [13]. In spite of these special boundary conditions, it was shown in [13, 15] that only one basis function per node is not sufficient to obtain good approximation in the case of a high-contrast multiscale coefficient, where several large high-contrast channels cross multiple coarse block boundaries, see Figure 2.2. The present method is a modification of the MsFEM that is known as the Generalized MsFEM or GMsFEM.

2.2 Generalized Multiscale FEM (GMsFEM)

Now we are concern with domain decomposition preconditioners, specially the two-levels domain decomposition preconditioner. As showed in [13, 15], the classical variants of these preconditioners do not perform well for high-contrast problems such as the ones considered here, the condition number of the preconditioned operator is in general of order η. The construction of the robust preconditioner may depend on the coefficient and it uses a coarse triangulation and local neighborhoods of the domain where the heat equation is posed. We use the Generalized Multiscale Finite Element methods framework introduced in [9, 21, 13] in order to construct robust methods. As in the constructions proposed there, the design of the preconditioner depends on the behavior of the coefficient inside local coarse node neighborhoods. The related work [35] uses our exploration to design efficient preconditioners for the elasticity topology optimization problem.

In this section, we show that these methods can be implemented for the topology optimization that performs, in terms of the number of iterations and condition number estimates, independently of the contrast in the media properties. The important part of the preconditioner in order to obtain the results is well known to be the coarse level. As shown in [10], the coarse level of the preconditioner should, locally, generate the kernel of the operator and also it should generate, locally, the modes whose eigenvalue depend on the contrast of the coefficient. For this, the solution of a properly chosen eigenvalue problem must be computed.

A key part of the construction of robust two-levels domain decomposition methods is the construction of the coarse spaces for the second level of the preconditioner. In particular, two approaches can be identified. For coarse spaces with standard dimension, that is, one basis function per coarse node, it is imperative to have coarse basis functions ϕ such that $\kappa|\epsilon(\phi)|^2$ is bounded independently of the contrast. In some cases where the high-contrast regions do not form long channels that cut several edges of a coarse element this can be achieved by classical multiscale finite element basis functions or by energy minimizing partition of unity basis functions. But this is not possible for high-contrast coefficients with long channels, even for isotropic problems. This is similar to the case of heterogeneous diffusion models on high-contrast media. As mentioned before here we take the approach proposed in [9, 21, 13], where in order to achieve robustness, an enrichment procedure is implemented by adding basis functions that locally generate the small modes of the operator (when considered in local coarse mesh neighborhoods).

Apart from the fact that the coefficient shows high-contrast in the media properties and has multiscale variations, an extra complication comes from the fact that throughout the optimization iteration the coefficient $\mathbf{K}$ (or $\mathbf{C}$ in elasticity) is changing as it approaches the optimal design. In particular, the topology it represents may change as well as its multiscale structure. The subdomain Ω^{mat} is a globally connected domain but, if restricted to coarse neighborhoods, high-contrast channels may break apart or join together from one iteration to the other. Dealing with this extra complication requires re-computation of the preconditioner as the optimization iteration advances towards the optimal design.

Now we focus on high-contrast multiscale problems and summarize a GMsFEM construction, following [33]. For a more detailed description of the development of the GMsFEM methodology, see [13, 15, 14] and references therein.

Let consider the following linear elliptic equation

$$Lu = f \text{ in } \Omega, \tag{2.1}$$

where Ω is a polygonal domain in $\mathbb{R}^{\mathrm{d}}$ $(d = 2, 3)$, $Lu := -\operatorname{div}(k(x)\nabla u)$, and $k(x)$ is an heterogeneous field varying in multiple scales. Recall that for the topology optimization case,

$$0 < k_0 \leq k(x) \leq 1, \quad \text{with} \quad k_0 \ll 1.$$

We can write problem (2.1) in variational formulation as: Find $u \in H_0(\Omega)$ such that

$$a(u, v) = l(v) \qquad \text{for all } v \in H_0(\Omega), \tag{2.2}$$

with

$$a(u, v) = \int_\Omega k(x)\nabla u(x)\nabla v(x)\, dx \qquad \text{for all } u, v \in H_0^1(\Omega),$$

and

$$l(v) = \int_\Omega f(x)v(x)\, dx \qquad \text{for all } v \in H_0^1(\Omega).$$

Take a coarse grid $\mathcal{T}^H$, note that this grid does not necessarily capture the smallest details of the multiscale coefficient, see Figure 2.2. Let $\mathcal{T}^h$ be a refinement of $\mathcal{T}^H$, this fine grid is able to solve the details of the coefficient but the resulting linear system is too large to be solved efficiently by classical numerical methods.

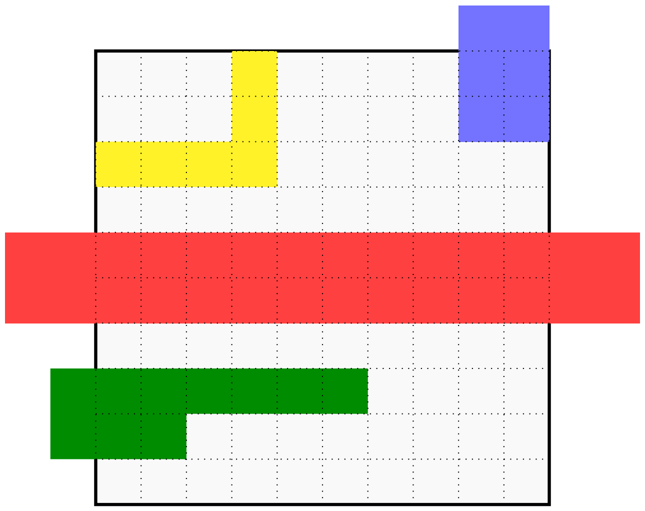

Figure 2.2: High conductivity channels over a coarse element, note how the fine grid capture the smallest details of the multiscale coefficient inside the coarse element.

Denote by $V^h(\Omega)$ the space of piecewise bilinear continuous functions with respect to the fine triangulation $\mathcal{T}^h$, with $V_0^h(\Omega) \subset V^h(\Omega)$ the subspace of piecewise linear continuous functions that vanish on $\delta\Omega$.

The Galerkin formulation of (2.2) is: Find $u_h \in V_0^h(\Omega)$ such that

$$a_h(u_h, v_h) = l_h(v_h) \qquad \text{for all } v_h \in V_0^h(\Omega),$$

which in matrix form yields,

$$\mathbf{A}u_h = b.$$

Let N_c be the number of vertices of the coarse mesh $\mathcal{T}^H$ and $\{y_i\}_{i=1}^{N_c}$ the set of coarse mesh vertices. We define the neighborhood of the node y_i by

$$\omega_i = \bigcup \left\{ Q_j \in \mathcal{T}^H;\ y_i \in \overline{Q_j} \right\},$$

and the neighborhood of the coarse element Q by

$$\omega_Q = \bigcup \left\{ \omega_j \in \mathcal{T}^H;\ y_j \in \overline{Q} \right\}.$$

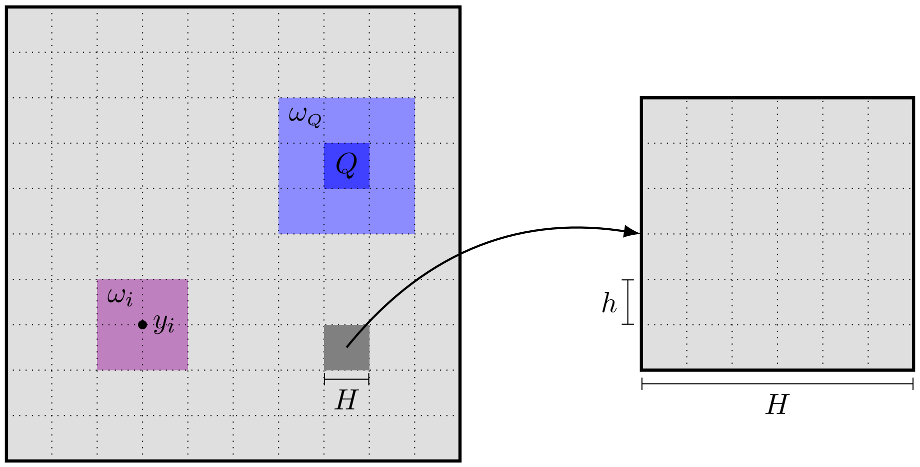

Figure 2.3: In violet, the neighborhood ω_i of the node y_i, in light blue the neighborhood ω_Q of the element Q. Also note how a coarse element in gray, with size H is divided into a fine mesh of size h.

We start by choosing an initial set of basis functions that form a partition of unity. Let $\mathcal{T}^H$ be a partition of Ω into finite elements (triangles, quadrilaterals, etc) a partition of unity is a set R of smooth, non-negative functions χ_i into $[0,1]$ such that for $Q \in \mathcal{T}^H$, $\sup \chi_i \subset Q$, and

$$\sum_{\chi_i \in R} \chi_i(x) = 1.$$

The space generated by this basis functions is enriched using the following local spectral problem, for each coarse node neighborhood ω_i, consider the eigenvalue problem as in [23]

$$-\operatorname{div}(\mathbf{K}\nabla\psi_\ell^{\omega_i}) = \lambda_\ell^{\omega_i}\mathbf{K}\psi_\ell^{\omega_i}, \tag{2.3}$$

with homogeneous Neumann boundary condition on $\partial\omega_i$ and Dirichlet boundary condition on $\partial\omega_i \cap \partial\Omega$ if $\partial\omega_i \cap \partial\Omega$ is not empty.

Here $\lambda_\ell^{\omega_i}$ and $\psi_\ell^{\omega_i}$ are eigenvalues and eigenvectors in ω_i. We use an ascending ordering on the eigenvalues and pick the low energy modes, $\lambda_1^{\omega_i} \leq \lambda_2^{\omega_i} \leq \cdots \leq \lambda_m^{\omega_i}$. To get this eigenvalues we solve an approximation of this eigenvalue problem given by,

$$\mathbf{A}^{\omega_i}\psi^{\omega_i} = \lambda^{\omega_i}\mathbf{M}^{\omega_i}\psi^{\omega_i}, \tag{2.4}$$

with

$$v\mathbf{A}^{\omega_i}w = \int_{\omega_i} k(x)\nabla v\nabla w \, dx \qquad \text{for all } v, w \in V^h(\omega_i),$$

and

$$v\mathbf{M}^{\omega_i}w = \int_{\omega_i} k(x)vw \, dx \qquad \text{for all } v, w \in V^h(\omega_i).$$

Note that the eigenvectors $\psi_\ell^{\omega_i}$ form an orthonormal basis of $V^h(\omega_i)$ with respect to the $\mathbf{M}^{\omega_i}$ inner product, see [15]. Also note that $\lambda_1^{\omega_i} = 0$ if $\partial\omega_i \cap \partial\Omega$ is empty.

As in [33], we construct a set of enriched multiscale basis functions given by $\chi_i\psi_\ell^{\omega_i}$ for the selected eigenvectors $\psi_\ell^{\omega_i}$. Using L_i to denote the number of basis functions from the coarse region ω_i, we then define the coarse GMsFEM space by

$$V_0 = \text{span}\{\Phi_{i,\ell} = \chi_i\psi_\ell^{\omega_i}, \quad i = 1, \dots, N_v, \quad \ell = 1, \dots, L_i\}.$$

For a detailed construction and additional properties of the space V_0 see Figure 2.4 and [13, 14, 15].

2.3 Coarse Multiscale Basis and a Randomized Algorithm

Here we analyze in a detailed manner the construction of the coarse basis using an eigenvector problem, following [23]. We consider a general case in which multiple high-conductivity (high-stiffness for the elasticity case) regions are disconnected from (or do not communicate with) each other, see Figure 2.2. It is shown in [23] that for the case of a singly connected high-conductivity region in a coarse region, only one basis function per node is needed. When there are multiple high-conductivity regions, one needs to enrich the coarse spaces to get robust preconditioners. See [13, 14, 15] and references therein.

This eigenvalue problem is solved in a union of coarse-grid blocks with a common vertex y_i, noted ω_i. It turns out that the problem (2.3) has eigenvalues that scale as the inverse of the high contrast. In particular, the number of eigenvalues that scale as the inverse of high conductivity is the same as the number of connected high-conducting regions.

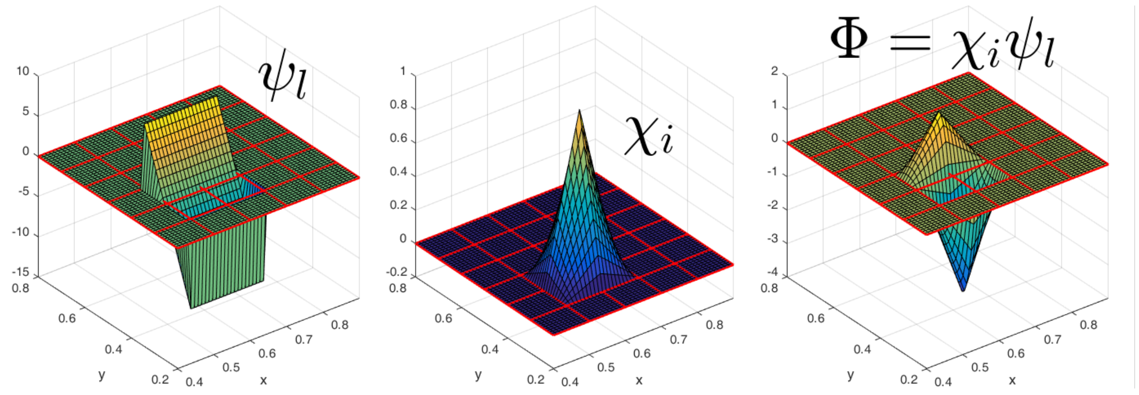

Figure 2.4: Coarse basis construction. Left: selected eigenvalue, center: partition of unity function and right: basis function.

The eigenvalue problem above corresponds to the approximation of the eigenvalue problem(2.3) in Ω with the Neumann boundary condition. In particular, $\psi_l^{\omega_i}$ denotes the l-th eigenvector of the

Neumann matrix associated with the neighborhood of y_i. If there are n inclusions and channels, then one can observe n small, asymptotically vanishing eigenvalues. The eigenvectors corresponding to these eigenvalues are to be used to construct the coarse space V_0. Low energy modes are related to asymptotically vanishing eigenvalues. As we explained before these eigenvalues represent the disconnected high-contrast regions, see [13, 14, 15].

In order to get the coarse basis functions, we need to solve local eigenvalue problems that might be costly to solve. As in [8], we employ randomized singular value decomposition (SVD) techniques in order to reduce the computational cost of local eigenvalue problems. It is shown in [8] that only a few of these randomly generated vectors can approximate the dominant modes of the solution space.

We state the following pseudo-algorithm for the randomized SVD for the eigenvalue problems, from [22],

1. Generate forcing terms $f_1, f_2, \ldots, f_M$ randomly (such that $\int_{\omega_i} f_\ell = 0$);
2. Compute the local solutions $-\operatorname{div}(\kappa \nabla u_\ell) = f_\ell$ with homogeneous Neumann boundary condition;
3. Generate $W_i = \operatorname{span}\{u_\ell\} \cup \{1\}$;
4. Choose the low energy eigenvalues in W_i according to a predefined bound.

More precisely, the idea is to restrict the eigenvalue problem (2.3), and therefore (2.4) to the subspace W_i. This is done as follows: Consider the matrix U_i whose columns generate the subspace $W_i = \operatorname{span}\{u_\ell\} \cup \{1\}$. Then we can introduce the reduced size matrices,

$$\widetilde{\mathbf{A}}^{\omega_i} = U_i^T \mathbf{A}^{\omega_i} U_i,$$

and

$$\widetilde{\mathbf{M}}^{\omega_i} = U_i^T \mathbf{M}^{\omega_i} U_i.$$

Then, instead of (2.4) we can solve the smaller dimension eigenvalue problem

$$\widetilde{\mathbf{A}}^{\omega_i} \widetilde{\psi}^{\omega_i} = \widetilde{\lambda}^{\omega_i} \widetilde{\mathbf{M}}^{\omega_i} \widetilde{\psi}^{\omega_i}. \tag{2.5}$$

As in [22] we then consider the approximations of the eigenvalues as

$$\lambda^{\omega_i} \approx \widetilde{\lambda}^{\omega_i}, \tag{2.6}$$

and the approximation of the eigenvectors as,

$$\psi^{\omega_i} \approx U_i \widetilde{\psi}^{\omega_i}. \tag{2.7}$$

We note that the eigenvalue problem (2.5) is of the size of the dimension of the space W_i (or the number of snapshots, as they are called it in [22, 13]). Therefore the size of the eigenvalue problem (2.5) is much smaller than the size of the full eigenvalue problem (2.4). In the numerical experiments presented later we show that the performance of the preconditioner with the coarse basis functions constructed using the randomized eigenvalue problem (2.5) is similar to the one where the basis functions are constructed using (2.4) when the dimension of the space W_i is large *enough* (or, in the terminology of [22, 13], when the number of snapshots are large enough). For more details on the analysis see [8].

Chapter 3

Preconditioners for the Heat Equation

In this chapter, we describe how the coarse spaces proposed earlier can be used in two-levels domain decomposition preconditioners. In particular, we show that one can obtain preconditioners that yield a contrast-independent condition number, and thus they are optimal in terms of physical parameters. See the details of these results in [20, 11, 12]. An extension of the results to multilevel methods can be found in [42]. See also [35] for an application of our results to the elasticity equation in topology optimization problems.

3.1 Domain Decomposition Methods

Here we do a small review of the main idea of the domain decomposition method, for a detailed explanation see [39, 35], also we present some theorems to apply the decomposition domain method to the heat equation.

Let us begin with a classic example of the Schwarz parallel method in two regions for the Poisson problem (3.1) in a continuous region Ω. Consider the problem,

$$\begin{cases} -\Delta u = 0 & \text{in} \quad \Omega, \\ u = 0 & \text{on} \quad \partial\Omega, \end{cases} \tag{3.1}$$

where the domain Ω can be divided into two subdomains with a simpler geometry than Ω, see Figure 3.1. We divide Ω in two overlapping subdomains Ω_1 and Ω_2 as in Figure 3.1. Then we solve (3.1) in each subdomain using a previous approximation as boundary data,

$$\begin{cases} -\Delta u_1^{(i+1)} = 0 & \text{in} \quad \Omega_1, \\ u_1^{(i+1)} = 0 & \text{on} \quad \partial\Omega_1 \cap \partial\Omega, \\ u_1^{(i+1)} = u^{(i)} & \text{on} \quad \Gamma_1. \end{cases} \tag{3.2}$$

$$\begin{cases} -\Delta u_2^{(i+1)} = 0 & \text{in} \quad \Omega_2, \\ u_2^{(i+1)} = 0 & \text{on} \quad \partial\Omega_2 \cap \partial\Omega, \\ u_2^{(i+1)} = u^{(i)} & \text{on} \quad \Gamma_2, \end{cases} \tag{3.3}$$

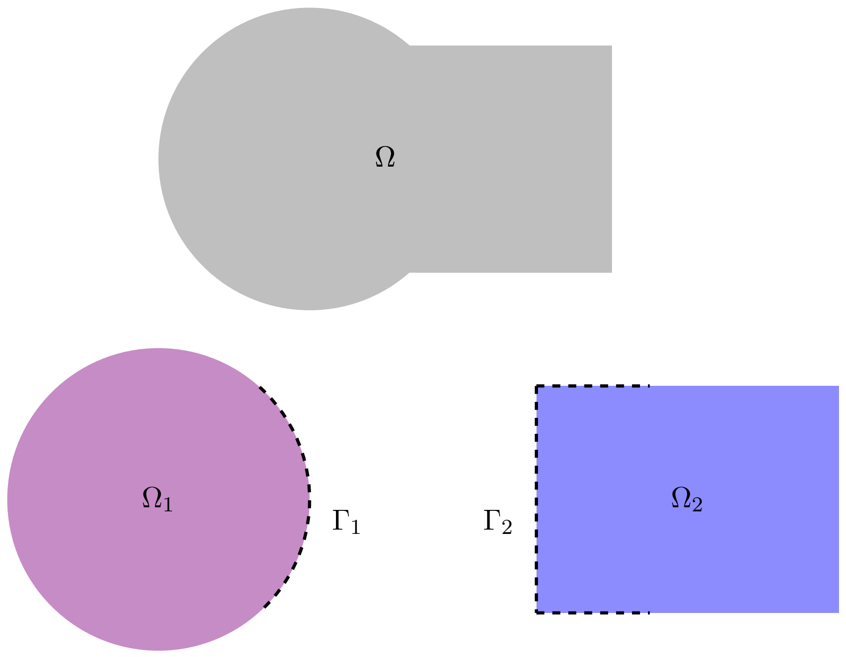

Figure 3.1: Domain Ω decomposed into two simple overlapping subdomains Ω_1 and Ω_2 of the original region $\Omega = \Omega_1 \cup \Omega_2$, see in [39].

We solve the local problems 3.2 and 3.3. These can be solved in parallel which gives a great computational advantage, also the linear problems derived from the FEM method are of a smaller dimension than the original. To get the solution we can add the local solutions via an extension by zero operator $R_j^\top$, other combinations are possible, see [39],

$$u^{(i+1)} = \sum_{j=1}^{2} R_i^\top u_j^{(i+1)}.$$

Then we use $u^{(i+1)}$ as boundary data to compute the next solution approximation.

3.1.1 Schwarz's method

We consider a finite dimensional Hilbert space V, and the symmetric and positive definite bilinear form,

$$\begin{aligned} a(\cdot,\cdot) :& V \times V \to \mathbb{R} \\ & (u,v) \mapsto a(u,v). \end{aligned}$$

Given $f \in V'$ we consider the problem of finding $u \in V$ such that

$$a(u,v) = f(v), \qquad \text{for all } v \in V. \tag{3.4}$$

If the matrix $\mathbf{A}$ is the stiffness matrix associated with the bilinear form $a(\cdot,\cdot)$ in the problem (3.4), and b is the vector associated with the linear form f, we obtain the following linear system,

$$\mathbf{A}u = b,$$

with $\mathbf{A}$ symmetric and positive definite.

We also consider a family of subspaces V_i with $i = 1, 2, \dots, N$ and we assume that there exists extension operators $R_i^\top : V_i \to V$, that extends the elements of subspace V_i into elements of the space V.

Assuming that the space V admits the following decomposition

$$V = R_0^\top V_0 + \sum_{i=1}^{N} R_i^\top V_i, \tag{3.5}$$

see more in [39, 35]. In the subspaces V_i, we introduce the bilinear form associated to the subspaces V_i,

$$a_i(\cdot,\cdot) : V_i \times V_i \to \mathbb{R}, \quad \text{for all i} = 0, 1, \dots, \text{N},$$

defined as,

$$a_i(u_i, v_i) = a\left(R_i^\top u_i, R_i^\top v_i\right), \qquad u_i, v_i \in V_i.$$

The stiffness matrix associated to $a_i(\cdot,\cdot)$ is,

$$A_i = R_i A R_i^\top,$$

and,

$$b_i = R_i^\top b,$$

which yields,

$$A_i u_i = b_i, \qquad u_i, b_i \in V_i,$$

we obtain exact local solutions in the subspace V_i for $i = 1, \dots, N$, with right hand side b_i.

The Schwarz operator P_i, can be written as,

$$P_i = R_i^\top A_i^{-1} R_i A, \qquad 0 \le i \le N,$$

and the additive Schwarz preconditioner is,

$$P_{ad} = A_{ad}^{-1} A,$$

where $A_{ad}^{-1} = \sum_{i=0}^{N} R_i^\top A_i^{-1} R_i$. The additive operator P_{ad} is symmetric and positive definite, see more in [39]. Then we apply the conjugate gradient algorithm to solve the following equation

$$P_{ad} u = A_{ad}^{-1} f.$$

Now we give an estimate for the condition number of P_{ad} with the next assumptions and lemmas. The proofs for the lemmas and theorems can be found in [39] and [35] .

Assumption 1 (Stable Decomposition)**.** *There exists a constant C_0, such that every $u \in V$ admit a decomposition,*

$$u = \sum_{i=0}^{N} R_i u_i, \qquad \{u_i \in V_i, 0 \le i \le N\},$$

that satisfies,

$$\sum_{i=0}^{N} a_i(u_i, u_i) \le C_0^2 a(u, u). \tag{3.6}$$

Assumption 2 (Strengthened Cauchy-Schwarz Inequalities)**.** *There exist constants* $0 \leq \epsilon_{ij} \leq 1$*, for* $1 \leq i, j \leq N$*, such that*

$$|a(R_i^\top u_i, R_j^\top u_j)| \leq \epsilon_{ij} a(R_i^\top u_i, R_j^\top u_j)^{1/2} a(R_i^\top u_i, R_j^\top u_j)^{1/2}$$

for $u_i \in V_i$ *and* $u_j \in V_j$*. We denote the spectral radius of* $\mathcal{E} = \{\epsilon_{ij}\}$ *by* $\rho(\mathcal{E})$*.*

Assumption 3 (Local Stability)**.** *There exist* $\omega > 0$*, such that,*

$$a(R_i^\top u_i, R_i^\top u_i) \leq \omega a_i(u_i, u_i), \qquad u_i \in \mathrm{range}(\tilde{P}_i) \subset V_i, \qquad 0 \leq i \leq N.$$

Lemma 5. *Assuming the stable decomposition. Then,*

$$a(P_{ad}u, u) \geq C_0^{-2} a(u, u), \qquad u \in V, \tag{3.7}$$

and consequently P_{ad} *defined in [39] is invertible. In addition,*

$$a(P_{ad}^{-1}u, u) = \min_{u_i \in V_i \, u = \sum R_i^\top u_i} \sum_{i=0}^{N} a_i(u_i, u_i).$$

Lemma 6. *Assuming the local stability and the inequalities of Cauchy-Schwarz, then for* $i = 0, \ldots, N$ *we have that,*

$$\|P_i\|_a \leq \omega.$$

In addition,

$$a(P_{ad}u, u) \geq \omega(\rho(\mathcal{E}) + 1) a(u, u),$$

where $\rho(\mathcal{E})$ *is the spectral radius.*

Theorem 5 (Additive operator's condition number)**.** *Let assume the stable decomposition, the local stability and the strengthened Cauchy-Schwarz inequalities be satisfied. Then the condition number of the additive Schwarz operator satisfies*

$$\kappa(P_{ad}) \leq C_0^2 \omega(\rho(\mathcal{E}) + 1).$$

This fundamental result is used to obtain estimations on the performance of domain decomposition methods. A related application is the design of robust methods, see [19].

3.2 Two-levels Domain Decomposition Preconditioner

Based on the results above, we extend the ideas of the example at the beginning of this Chapter to more than two local problems. We recall also the generalized multiscale approximation from Chapter 2 and apply these results to build a two-levels preconditioner for the heat equation. We denote by $\{\Omega_i'\}_{i=1}^N$ the overlapping decomposition obtained from the original non-overlapping decomposition $\{\Omega_i\}_{i=1}^N$ by enlarging each subdomain Ω_i to

$$\Omega_i' = \Omega_i \cup \{x \in \Omega, \mathrm{dist}(x, \Omega_i) < \delta_i\}, \quad i = 1, \ldots, N,$$

where dist is some distance function and let $V_0^i(\Omega_i')$ be the set of finite element functions with support in Ω_i'.

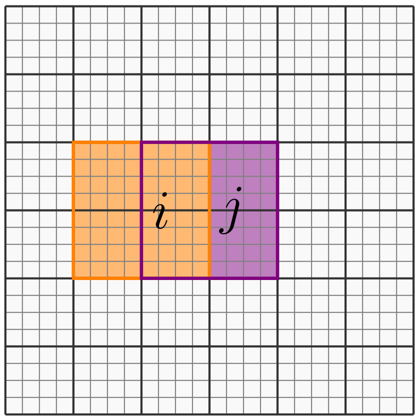

Figure 3.2: Overlap example in a 6×6 coarse mesh. Note the two elements i and j of the overlapping decomposition $\{\Omega_i'\}_{i=1}^N$, and how they overlap.

We also denote by $R_i^\top : V_0^i(\Omega_i') \to V^h$ the extension by the zero operator and $A_i = R_i^\top A R_i$, which yields,

$$M_{Heat,1}^{-1} r = \sum_{i=1}^N R_{Heat,i} A_{Heat,i}^{-1} R_{Heat,i}^\top r,$$

this is the first level preconditioner. We have the following bound from [22],

$$\kappa(M_{Heat,1}^{-1} A) \leq C \left(1 + \frac{1}{\delta H}\right),$$

with,

$$\delta = \max_{1 \leq i \leq N} \delta_i,$$

we assume that the overlapping subdomains $\{\Omega_i'\}$ coincide with the coarse vertex neighborhoods ω_i, which means that $\delta \asymp H$. In high-contrast multiscale problems we have that $C \asymp \eta$, see [22], which yields,

$$\kappa(M_{Heat,1}^{-1} A) \preceq \eta \left(1 + \frac{1}{H^2}\right).$$

Here we use the two-levels domain decomposition preconditioner. The preconditioned operator is $M_{Heat}^{-1} A$ where the preconditioner matrix is defined by,

$$M_{Heat}^{-1} = M_{Heat,1}^{-1} + M_{Heat,2}^{-1},$$

where the part corresponding to the first level is

$$M_{Heat,1}^{-1} r = \sum_{i=1}^N R_{Heat,i} A_{Heat,i}^{-1} R_{Heat,i}^\top r,$$

and the part corresponding to the second (or coarse) level is

$$M_{Heat,2}^{-1} r = R_{Heat,0} A_{Heat,0}^{-1} R_{Heat,0}^\top r,$$

where $A_{Heat,0} = R_{Heat} A_{Heat} R_{Heat}^\top$ and $R_{Heat}^\top$ is a matrix whose columns generate the coarse space V_0.

We can solve the fine-scale linear system iteratively with a numerical method like the preconditioned conjugate gradient (PCG) method. The application of the preconditioner involves solving a coarse-scale system and solving local problems in each iteration. In domain decomposition methods, our main goal is to reduce the number of iterations in the iterative procedure. The appropriate construction of the coarse space V_0 plays a key role in obtaining robust iterative domain decomposition methods.

In the general setting of domain decomposition methods, the overlapping subdomains $\{\Omega_i'\}$ and the coarse triangulation $\mathcal{T}^H$ are not related. The two partitions of unity used here can be chosen independently of each other. Both partitions of unity are needed to construct a contrast-independent domain decomposition method. In the numerical experiments, we assume that the overlapping subdomains $\{\Omega_i'\}$ coincide with the coarse vertex neighborhoods $\{\omega_i\}$ of $\mathcal{T}^H$.

It was proven in [10] that the condition number of the two-levels preconditioner with a GMsFEM second level is

$$\kappa(M_{Heat}^{-1}A) \preceq C\left(1 + \frac{H}{\delta}\right).$$

3.2.1 Experiments for the Heat Preconditioner

In the next experiment we use the coefficient from Figure 1.3 in Chapter 1, which have long high-conductivity channels. We use the conjugate gradient method with preconditioning with a tolerance of 1×10^{-6}. We set a 10×10 coarse mesh and inside each coarse-element we have a 10×10 fine-mesh.

Contrast	Iterations	Spectral condition
1	112	2.0×10^3
1×10^{-2}	768	1.7×10^4
1×10^{-4}	2961	3.8×10^5
1×10^{-6}	7760	3.7×10^7

Table 3.1: Heat equation using coefficient from Figure 1.3. Without preconditioners and with a constant heat source.

For a contrast of 1×10^{-4} the PCG takes roughly 3000 iterations to converge, in the topology optimization, this means expensive computational time which makes this option unreasonable to apply. We can see in Table 3.1, how iterations of the conjugate gradient method without preconditioning does depend on contrast, while in Table 3.2 iterations and spectral condition of $M_{Heat}^{-1}A$ do not depend on contrast. Also, there is a significant improvement in the iteration count using the two-levels preconditioner which means lower computational time in each optimization step. For bounds and theoretical results see [19, 17].

Contrast	Iterations	Spectral condition
1	17	4.8
1×10^{-2}	25	9.5
1×10^{-4}	29	15.0
1×10^{-6}	30	15.0

Table 3.2: Heat equation using coefficient from Figure 1.3. Using a two-levels domain decomposition preconditioner and a constant heat source.

Chapter 4

Topology Optimization

The purpose in topology optimization is to find the best material distribution which optimizes certain objective function. This objective can be for example compliance, displacement or heat distribution. Several approaches can be taken to solve this problem, but our main focus is on *density formulations*. To achieve this optimization, we must minimize or maximize a functional that describes the desired objective subject to a boundary value problem which describes the physical conditions. To solve the boundary value problem the Finite Element Method is applied. For the optimization part, we use mainly two optimization algorithms, the Optimality Criteria (OC) and the Method of Moving Asymptotes (MMA). We present the topology optimization in a more general density formulation as in [31],

$$\begin{cases} \min\limits_{\rho} J(\rho, u), \\ \text{subject to} \\ \qquad r(\rho, u) = 0, \quad u \in \mathcal{U}_{ad}, \\ \qquad g_i(\rho, u) \leq 0, \quad i = 1, \dots, N_g, \\ \qquad \rho \in \mathcal{D}_{ad}, \end{cases}$$

where $J(\rho, u)$ is an objective function, in our case is the compliance, $r(\rho, u) = 0$ represent the residual of the equilibrium equation describing the physical model, and g_i are multiple constraints. The material density field is represented by a bounded function $\rho \in L^\infty(\Omega)$ with $0 \leq \rho_{\min} \leq \rho \leq 1$, see more in [31].

We analyze the minimum compliance design because is a relatively simple topology optimization problem and at the same time carries the most common problems, namely mesh-dependent solutions and checkerboard patterns. The typical topology optimization problem stems from the minimization of the compliance of a linear elastic system. Thus, we seek for the optimal material distribution that minimizes the compliance, which is equivalent to maximizing the stiffness of the design, and fulfill the elasticity equilibrium equation.

Now we pose the topology optimization problem for the heat equation in a similar way as for the elasticity equation. After the FEM discretization the heat equilibrium equation can be written as,

$$\mathbf{K}u = f,$$

with f being the heat forcing term, $\mathbf{K}$ the conductivity matrix and u the temperature. In topology optimization for the heat equation, we seek for the optimal material distribution that produces the least heat for a prescribed material volume, see [24]. We look to minimize the heat potential capacity $C = f^\top u$. We state the discrete minimum compliance design problem as: find $\{\rho_1, \ldots, \rho_n\}$ with $\rho_i = 1$ if the element is solid, and $\rho_i = 0$ if it is void.

$$\begin{cases} \min C = f^\top u, \\ \text{subject to} \\ \qquad \mathbf{K}u = f, \\ \qquad \dfrac{\sum\limits_{i=1}^{N} V_i \rho_i}{\sum\limits_{i=1}^{N} V_i} = V_f, \end{cases}$$

here ρ_i is a binary variable that states is the element i is present or not in the optimized structure. The value v_f is a prescribed volume fraction of the available material and V_i is the volume of the element i.

We change the discrete problem to a continuous one using the modified Simple Isotropic Material with Penalization (SIMP) formulation,

$$\begin{cases} \min C = f^\top u(\rho), \\ \text{subject to} \\ \qquad \mathbf{K}(\widetilde{\rho}(\rho))u = f, \\ \qquad \sum\limits_{i=1}^{N} V_i(\widetilde{\rho}(\rho_i)) - V_f \sum\limits_{i=1}^{N} V_i \leq 0, \\ \qquad 0 \leq \rho \leq 1. \end{cases}$$

With ρ the vector of the design variables and $\widetilde{\rho}$ the vector of filtered densities. The material properties for the element Q in the SIMP are given by

$$K_Q = K_{\min} + \widetilde{\rho}_Q^{\,p} \left(K_{\max} - K_{\min}\right),$$

where the parameter $p = 3$ penalizes the intermediate densities, see more in [30].

For a detailed explanation on density formulations, the Simple Isotropic Material with Penalization (SIMP) and the modified SIMP formulation, see [3, 2, 30, 31, 41].

To explain the multiple options and parameters in topology optimization we set the following heat sink conduction problem, see more in [3] and in [31]. Take a square domain that is uniformly heated, in the left side, on a part of the boundary, we set an homogeneous Dirichlet boundary condition, as in Figure 4.1.

Let Q_x and Q_y be the number of elements in the finite element discretization in x and y respectively, $r_{\min}$ is the filter radius and p is the penalty in the SIMP formulation, see the code from [2].

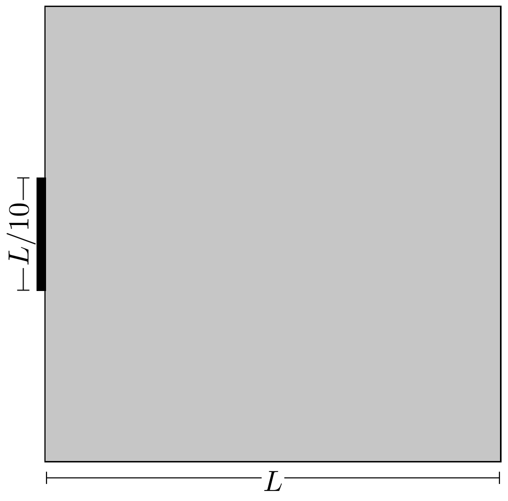

Figure 4.1: Heat sink problem scheme. In the left side on a part of the boundary we have a Dirichlet boundary condition. The rest of the boundary has a free Neumann condition. The plate is uniformly heated.

In the left image we set $Q_x = 50$, $Q_y = 50$, $r_{\min} = 2$ with $r_{\min}$ being 0.04 times the width of the domain, we use the density filter, and we change the penalty to 1 (no penalty applied), this make the optimized design lack of a $0-1$ characteristic, i.e. the optimized design is mainly an intermediate density (gray areas) which in not a manufacturable result.

In the right image we take $Q_x = 50$, $Q_y = 50$, $p = 3$, and $r_{\min} = 1$ which produces a characteristic checkerboard pattern, making the result not feasible. As $r_{\min} = 1$ the filter is not applying any changes on the design field, i.e. it do not regularize the problem.

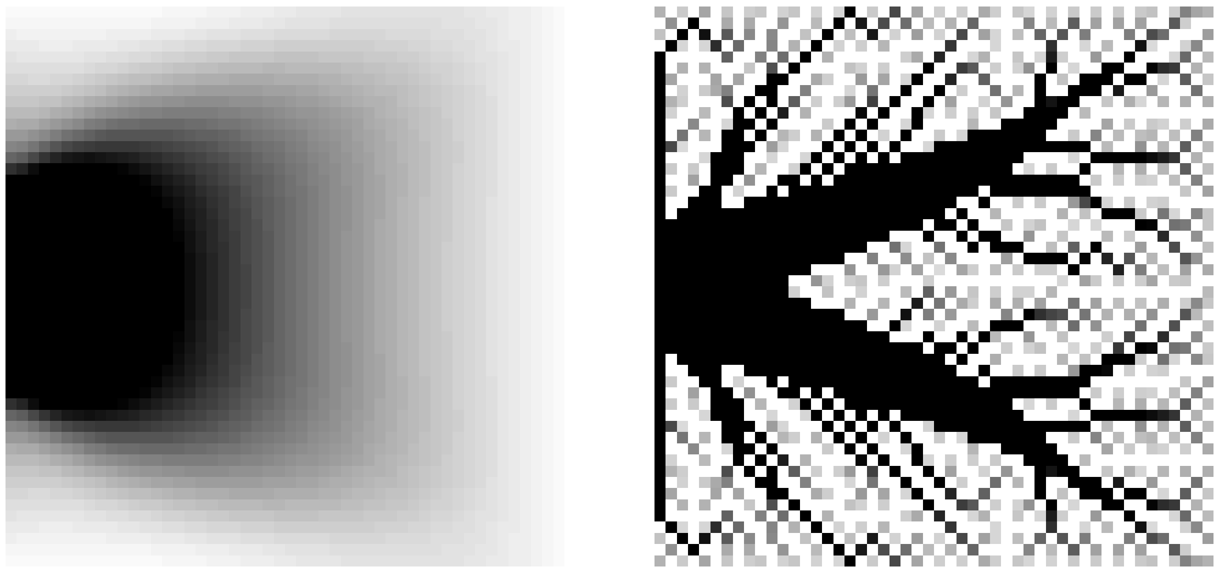

Figure 4.2: Left: without a modified SIMP formulation, which leads to a gray design. Right: without filtering, which leads to a checkerboard pattern.

In the next examples we show how the SIMP formulation is mesh independent. In the left image we have the result of a 50×50 elements topology optimization. In the right image we use a finer mesh of 250×250 elements. We can observe how both yields the same result, with just a more accurate definition in the fine mesh.

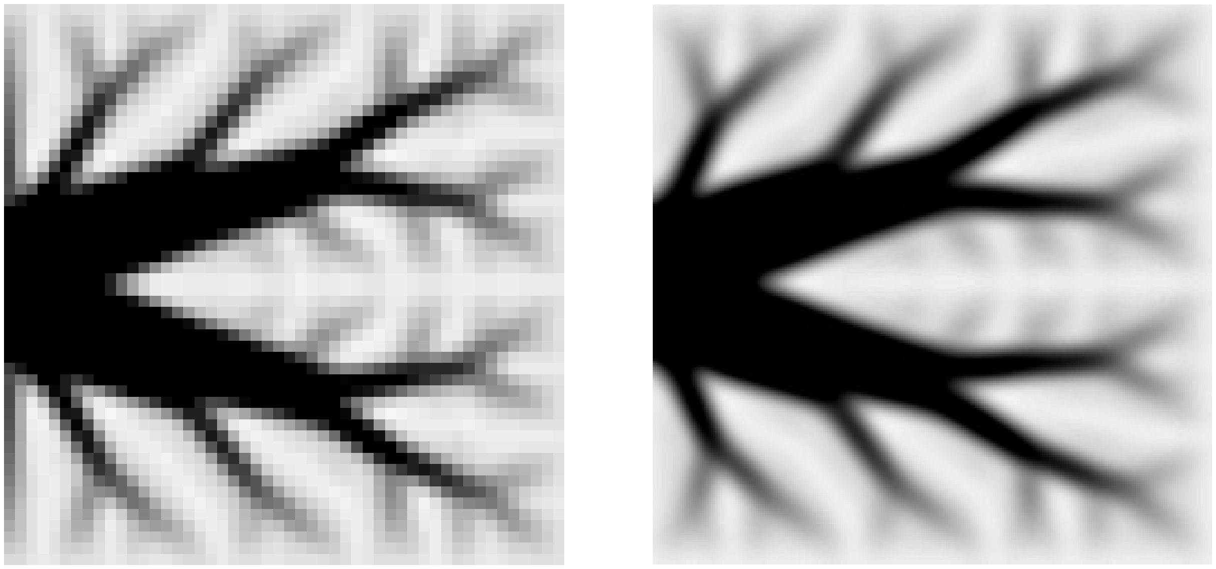

Figure 4.3: Left: with a 50×50 mesh. Right: with a 250×250 mesh. We can see the mesh independence of the formulation.

Sensitivity Filter

The sensitivity filter modifies the sensitivities The oldest filtering technique applied to topology optimization is the so-called sensitivity filter which modifies the sensitivities, $\dfrac{\partial c}{\partial \rho_Q}$ by,

$$\frac{\partial c}{\partial \rho_Q} = \frac{1}{\max(\gamma, r_{\min} - \|x_i - x_Q\|) \sum_{i \in N_Q} \omega_{Q,i}} \sum_{i \in N_Q} \omega_{Q,i} \rho_i \frac{\partial c}{\partial \rho_i}$$

where γ is a small number introduced in order to avoid division by zero, N_Q is the set of element I for which the center-to-center distance $\|x_I - x_Q\|$ to the element Q,

$$\omega(x_I) = r_{\min} - \|x_I - x_Q\|, \tag{4.1}$$

$r_{\min}$ is the radius filter, which usually is equal to 0.04 times the width of the domain. The main issue with the above technique is the lack of consistency, i.e., the actual objective corresponding to the filtered sensitivities is unknown for general optimization problems. The only known case is for linear elasticity case and minimum compliance design, see [37].

Density Filter

Density filtering is the most popular technique used nowadays in topology optimization. For the optimization problem, we apply conditions for the density with the modified SIMP, see more in [3].

Density filtering can be written in form of the convolution integral as:

$$\widetilde{\rho}_Q(x) = \int_\Omega \omega(x-y)\rho_Q(x),$$

where $w(x)$ is a weighting function.

We define the density filter in discrete form as,

$$\widetilde{\rho}_Q = \frac{\displaystyle\sum_{i\in N_Q} \omega(x_I)v_i\rho_I}{\displaystyle\sum_{i\in N_Q} \omega(x_I)v_I},$$

where v_I is the volume of the element I and ω is defined in (4.1). The derivative of the density filter is:

$$\frac{\partial\widetilde{\rho}_Q}{\partial\rho_j} = \frac{w(x_J)v_J}{\displaystyle\sum_{j\in N_Q} w(x_Q)v_Q}.$$

Implementing a filter in the optimization algorithm allows to eliminate the mesh dependency and the checkerboard patterns of the optimized structure, but the filter radius allows gray areas with intermediate density. To eliminate the gray areas and for the optimization process to be a pure black and white design, we are introducing a filter based on projections in the following section.

Heaviside projection filtering

The idea behind the projection filter is to project all density values $\widetilde{\rho}_Q$ above a threshold η to 1 and the values below η to 0. The projected physical density $\bar{\widetilde{\rho}}_Q$ is computed by a smooth function controlled by a projection parameter β and give as:

$$\bar{\widetilde{\rho}}_Q = \begin{cases} \eta[\exp(-\beta(1-\widetilde{\rho}_Q/\eta)) - (1-\widetilde{\rho}_Q/\eta)\exp(-\beta)] & 0 \le \widetilde{\rho}_Q \le \eta, \\ \\ (1-\eta)[1-\exp(-\beta(\widetilde{\rho}_Q-\eta)/(1-\eta)) & \\ +(\widetilde{\rho}_Q-\eta)/(1-\eta)\exp(-\beta)] + \eta & \eta \le \widetilde{\rho}_Q \le 1. \end{cases} \tag{4.2}$$

The expression (4.2) is an approximation of the Heaviside function. Often the above expression is replaced with a simpler one given as:

$$\bar{\widetilde{\rho}}_Q = \frac{\tanh(\beta\eta) + \tanh(\beta(\widetilde{\rho}_Q - \eta))}{\tanh(\beta\eta) + \tanh(\beta(1-\eta))},$$

with $\beta \to \infty$. The physical density $\bar{\widetilde{\rho}}_Q$ is used to compute the stiffness matrix and the sensitivities are:

$$\frac{\partial c}{\partial\rho_Q} = \sum_{I\in N_Q}\sum_{J\in N_I} \frac{\partial c}{\partial\bar{\widetilde{\rho}}_Q}\frac{\partial\bar{\widetilde{\rho}}_Q}{\partial\widetilde{\rho}_Q}\frac{\partial\widetilde{\rho}_Q}{\partial\widetilde{\rho}_Q}.$$

The Heaviside filter produce a sharper result than the density and sensitivity filters, see Figure 4.4, leading to a more $0-1$ design.

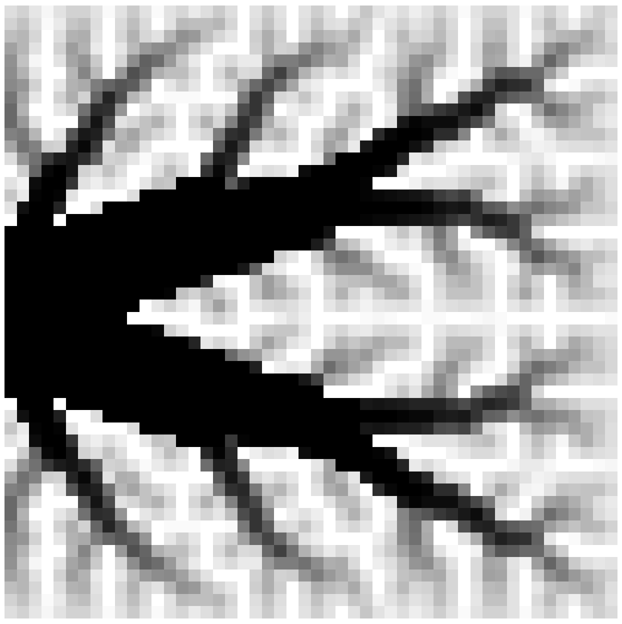

Figure 4.4: Using a projection filter, this leads to a more $0-1$ design.

4.1 Optimization Algorithms

4.1.1 Optimality Criteria (OC)

Here we review the main optimization algorithm to address simple topology optimization problems, following the one developed in [32]. The Optimality Criteria algorithm is the Lagrange Multipliers Method applied to topology optimization problems, mainly to the minimum compliance design.

For the discretized minimum compliance design problem,

$$\begin{cases} \min\limits_{U,\mathbf{K}_e} c(u) = U^\top \mathbf{K} U = \sum\limits_{e=1}^{N_Q} \mathbf{K}_Q(\bar{\tilde{\rho}}_Q) u_Q^\top \mathbf{K}_0 u_Q \\ \text{subject to} \\ \mathbf{KU} = F, \\ \dfrac{V(\rho)}{V_0} - f_\nu(\bar{\tilde{\rho}}_Q) = 0, \\ 0 < \rho_{\min} \leq \rho \leq \rho_{\max}, \end{cases} \tag{4.3}$$

writing the last equation from (4.3) as, $\rho_{\min} - \rho \leq 0$ and $\rho_{\max} - \rho \leq 0$, yields the following problem,

$$\begin{cases} \min\limits_{U,\mathbf{K}_e} c(u) = U^\top \mathbf{K} U = \sum\limits_{e=1}^{N_Q} \mathbf{K}_Q(\bar{\tilde{\rho}}_Q) u_Q^\top \mathbf{K}_0 u_Q \\ \text{subject to} \\ \mathbf{KU} = F, \\ \dfrac{V(\rho)}{V_0} - f_\nu(\bar{\tilde{\rho}}_Q) = 0, \\ \rho_{\min} - \rho \leq 0, \\ \rho_{\max} - \rho \leq 0. \end{cases} \tag{4.4}$$

For the problem (4.4) the Lagrangian can be written as,

$$L = c + \lambda_1(V(\rho) - fV_0) + \lambda_2^\top(Ku - F) + \sum_{e=1}^{N_Q} \lambda_3^e(\rho_{\min} - \rho_Q) + \sum_{e=1}^{N_Q} \lambda_4^e(\rho_Q - \rho_{\max}), \tag{4.5}$$

where λ_i for $i = 1, 2, 3, 4$, are the Lagrange multipliers.

We found the optimum as:

$$\frac{\partial L}{\partial \rho_Q} = 0 \qquad \text{for} \qquad e = 1, \dots, N_Q.$$

using the derivatives in (4.5), we have

$$\frac{\partial L}{\partial \rho_Q} = \frac{\partial c}{\partial \rho_Q} + \lambda_1 \frac{\partial V}{\partial \rho_Q} + \lambda_1^\top \frac{\partial (uK)}{\partial \rho_Q} - \lambda_3^e + \lambda_4^e,$$

and taking into account the constraints $\lambda_3^e = \lambda_4^e = 0$,

$$\frac{\partial L}{\partial \rho_Q} = u^\top K \frac{\partial u}{\partial \rho_Q} + u^\top \frac{\partial K}{\partial \rho_Q} u + u^\top K \frac{\partial u}{\partial \rho_Q} + \lambda_1 \frac{\partial V}{\partial \rho_Q} + \lambda_2^\top u \frac{\partial K}{\partial \rho_Q} - \lambda_2^\top K \frac{\partial u}{\partial \rho_Q},$$

grouping terms we get,

$$\frac{\partial L}{\partial \rho_Q} = u^\top \frac{\partial K}{\partial \rho_Q} u + \lambda_2^\top \frac{\partial K}{\partial \rho_Q} u + \frac{\partial u}{\partial \rho_Q}(2u^\top K + \lambda_2^\top K) + \lambda_1 \frac{\partial V}{\partial \rho_Q},$$

as $\lambda_2^\top$ is arbitrary, we chose $\lambda_2^\top = -2u^\top$, and we get,

$$\frac{\partial L}{\partial \rho_Q} = -u^\top \frac{\partial K}{\partial \rho_Q} u + \lambda_1 \frac{\partial V}{\partial \rho_Q}, \tag{4.6}$$

note that $\partial V / \partial \rho_Q = V_Q$, here we usually have that $V_Q = 1$ because the elements can be taken with unitary volume.

Replacing terms in (4.6),

$$\frac{\partial L}{\partial \rho_Q} = -p(\rho_Q)^{p-1} u_Q^\top \mathbf{K}_0 u_Q + \lambda V_Q = 0,$$

which yields,

$$\frac{-\dfrac{\partial c}{\partial \rho_Q}}{\lambda V_Q} = B_Q = 1,$$

and the update factor is,

$$\rho_Q^{i+1} = B_Q^i \rho_Q^i.$$

Now the heuristic update scheme can be written as in [2],

$$\rho_Q^{i+1} = \begin{cases} \max(0, \rho_Q^i - m) & \text{if } \rho_Q B_Q^\eta \leq \max(0, \rho_Q^i - m), \\ \rho_Q^i B_Q^\eta & \text{if } \max(0, \rho_Q^i - m) < \rho_Q^i B_Q^\eta < \min(1, \rho_Q^i + m), \\ \min(1, \rho_Q^i + m) & \text{if } \rho_Q^i B_Q^\eta \geq \min(1, \rho_Q^i + m), \end{cases}$$

where m is a positive moving limit, usually $m = 0.2$ and $\eta = 1/2$ is a damping coefficient. The value of the Lagrange multiplier λ is obtained using a bisection algorithm.

4.1.2 Method of Moving Asymptotes (MMA)

Now we briefly explain the Method of Moving Asymptotes, from [32] and [38]. The MMA was introduced by Svanberg in [38] as a method for nonlinear programming in general and structural optimization. In each step of the iterative process, a strictly convex approximating subproblem is generated and solved. The generation of these subproblems is controlled by so-called "moving asymptotes", which may both stabilize and speed up the convergence of the general process. The Method of Moving Asymptotes allows us to solve problems with more than one constraint, without significant changes to the code, and its implementation is easy as it requires almost the same parameters as the Optimality Criteria.

Following [32] we state the general optimization problem as,

$$\begin{cases} \min f_0(x), & x \in \mathbb{R}^{\mathrm{n}} \\ \text{subject to} & \\ f_i(x) \leq \bar{f}_i, & i = 1, \dots, m, \\ 0 < x^j_{\min} \leq x^j \leq x^j_{\max}, & j = 1, \dots, n. \end{cases}$$

Where m is the number of constraints and $x = (x_1, \dots, x_n)^\top$ is the vector of design variables, that is bounded by $x_{\min}$ and $x_{\max}$. The objective function and constraints are approximated by $\tilde{f}_0^{(k)}$ and $\tilde{f}_i^{(k)}$ respectively, which are linearizations of these functions, see [32],

$$\tilde{f}_i^{(k)} = \sum_{Q=1}^{n} \left(\frac{p_{iQ}}{U_Q - x_Q} + \frac{q_{iQ}}{x_Q - L_Q} \right) + r_i,$$

where L_Q and U_Q are chosen as

$$\begin{aligned} &\text{if } \frac{\partial f_i}{\partial x_Q}(x^{(k)}) > 0, \quad \text{then } p_{iQ} = (U_Q - x_Q^{(k)})^2 \frac{\partial f_i}{\partial x_Q}(x^{(k)}) \quad \wedge \ q_{iQ} = 0, \\ &\text{if } \frac{\partial f_i}{\partial x_Q}(x^{(k)}) < 0, \quad \text{then } q_{iQ} = -(x_Q^{(k)} - L_Q)^2 \frac{\partial f_i}{\partial x_Q}(x^{(k)}) \ \wedge \ p_{iQ} = 0, \end{aligned}$$

where $L_Q < x_Q^{(k)} < U_Q$ and r_i is chosen such that $\tilde{f}_i^{(k)}(x^{(k)}) = f_i^{(k)}(x^{(k)})$. After every iteration the values of the asymptotic points L_Q and U_Q are updated using a heuristic method.

4.2 Robust Topology Optimization

We apply a modified version of the robust topology optimization from [41] to the minimum compliance design problem. We create a dilated $\bar{\tilde{\rho}}^d$, an intermediate $\bar{\tilde{\rho}}^i$ and an eroded $\bar{\tilde{\rho}}^e$ design using the threshold projection (4.2), as in [41]. Using the threshold parameters η, 0.5 and $1 - \eta$, respectively. The minimum compliance design problem can be formulated as,

$$\begin{cases} \min C = f^\top u(\bar{\tilde{\rho}}^e), \\ \text{subject to} \\ \qquad \mathbf{K}(\bar{\tilde{\rho}}^e) u^e = f, \\ \qquad \sum_{i=1}^{N} V_i(\bar{\tilde{\rho}}_i^d) - V_d^* \sum_{i=1}^{N} V_i \leq 0, \\ \qquad 0 \leq \rho \leq 1, \end{cases}$$

with $\eta \leq 0.5 \leq 1 - \eta$.

We modify the code from [2] to match the modified pseudo-code of robust topology optimization based in [41], we use the Method of Moving Asymptotes as the optimizer.

1. Initialize the design variable ρ. Set up the threshold value η and the maximum design variable change $\Delta\rho_{\max}$, the iteration counter $i = 0$, and $\beta = 2$.

2. For $i \leq i_{\max}$, with $i_{\max} = 200$ or 300, do:

    - $i = i + 1$.
    - Compute $\bar{\tilde{\rho}}^{i}$.
    - Solve the FEM problem $\mathbf{K}(\bar{\tilde{\rho}}^{e})u^e = f$.
    - Compute the sensitivities.
    - Update the design variables.
    - if [$\mathrm{mod}(i, 20) = 1$] update the volume fraction of the dilated structure $V_d^* = \frac{V^*}{V_i} V_d$ such that the volume of the intermediate design V_i becomes equal to V^*.
    - if [$\mathrm{mod}(i, 50) = 1$ or $\|\Delta\rho\|_\infty \leq 0.01$] and [$\beta \leq \beta_{\max}$] then $\beta = 2\beta$.

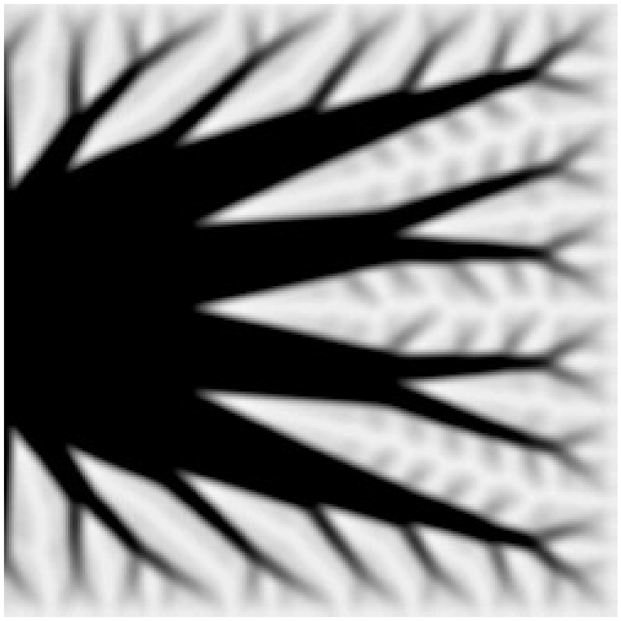

Figure 4.5: Using the robust formulation, with β fixed to 1.

In the next Chapter, we present results of the robust topology optimization algorithm using a two-levels domain decomposition.

4.3 Minimum Compliance Design for the Two Dimensional Heat Equation

In this part we present the minimum compliance design for the stationary heat equation over a domain $\Omega \subset \mathbb{R}^2$, with boundary $\partial\Omega = \Gamma_1 \cup \Gamma_2$, with Dirichlet condition in Γ_1 and Neumann condition in Γ_2. The problem is to find $K_{ad} \subseteq [K \,|\, K : \Omega \to \mathbb{R}\}$ such that

$$J = \min_{K \in K_{ad}} \left(\int_\Omega f\, u_K(x)\, d\,\Omega + \int_{\Gamma_2} \beta(x)\, u_K(x)\, d\,S \right), \tag{4.7}$$

where, given $K \subseteq K_{ad}$, u_K is the solution of

$$\begin{cases} -\operatorname{div}(K \cdot \nabla u_K) = f, & x \in \Omega, \\ K \cdot \nabla u_K = \beta(x), & x \in \Gamma_2, \\ u_K = \alpha(x), & x \in \Gamma_1. \end{cases} \tag{4.8}$$

We need to find the weak form from the strong form of the flow equation for the heat equation (4.8). We multiply (4.8) by a test function $v \in \mathcal{H}^1(\Omega)$ such that $v(x) = 0$ for $x \in \Gamma_1$ and $K \cdot \nabla v = \beta(x)$ for $x \in \Gamma_2$ and integrating over Ω,

$$\int_\Omega -\operatorname{div}(K \cdot \nabla u_K) v\, d\,\Omega = \int_\Omega f v\, d\,\Omega,$$

let η be the normal surface vector and using the Green's identity in the last equation we get

$$\int_\Omega K\, \nabla u_K \nabla v\, d\,\Omega - \int_{\partial\Omega} (K \cdot \nabla u_K)\eta v = \int_\Omega f v\, d\,\Omega.$$

We can divide the boundary in Γ_1 and Γ_2, and apply the boundary conditions

$$\int_\Omega K\, \nabla u_K \nabla v\, d\,\Omega - \int_{\Gamma_1} (K \cdot \nabla u_K)\eta v - \int_{\Gamma_2} (K \cdot \nabla u_K)\eta v = \int_\Omega f v\, d\,\Omega,$$

using the boundary conditions $v(x) = 0$ for $x \in \Gamma_1$ and $K \cdot \nabla v = \beta(x)$ for $x \in \Gamma_2$ we obtain

$$\int_\Omega K\, \nabla u_K \nabla v\, d\,\Omega - \int_{\Gamma_2} K\, \beta\eta = \int_\Omega f v\, d\,\Omega.$$

We take,

$$a_K(u, v) = \int_\Omega K\, \nabla u_K \nabla v\, d\,\Omega,$$

and

$$l(u) = \int_\Omega f u\, d\,\Omega + \int_{\Gamma_2} \beta(x)\eta u\, d\,S$$

The minimum compliance design for the heat equation takes the form

$$\begin{cases} \min\limits_{u \in U, K} l(u) \\ \text{subject to} \\ a_K(u, v) = l(v) \qquad \text{for all } v \in U \\ K \in K_{ad}. \end{cases} \tag{4.9}$$

Now, we apply the finite element method. For this we take a partition τ^h in Ω with square elements $Q_1, Q_2, \ldots, Q_n$, such that $\Omega = \bigcup_{i=1}^{n} \overline{Q_i}$ and $Q_i \cap Q_j = \emptyset$ if $i \neq j$ and,

$$\overline{Q_i} \cap \overline{Q_j} = \begin{cases} \overline{Q_i} \quad \text{if} \quad i = j, \\ \text{one side of } \overline{Q_i} \text{ and } \overline{Q_j}, \\ \text{one vertex of } \overline{Q_i} \text{ and } \overline{Q_j}, \\ \emptyset \quad \text{other cases.} \end{cases}$$

And defining the mesh as in Figure 4.6. And, if we consider

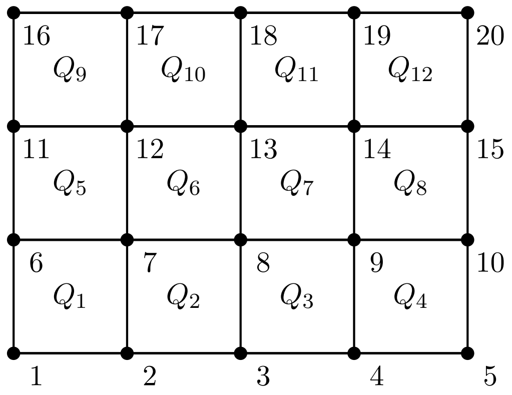

Figure 4.6: Example for a rectangle domain Ω with a partition in square elements Q_i for $i = [1, 2, \ldots, 20\}$ vertices over the mesh.

$$P^1(\tau^h) = \left\{ v : \Omega \to \mathbb{R} \;\middle|\; \begin{array}{ll} v & \text{continuous function in } \Omega, \\ v|_{Q_i} & \text{is a polynomial function of degree one,} \\ & \text{with } v(x) = \alpha \text{ if } x \in \Gamma_1 \text{ and } v(x) = \beta \text{ if } x \in \Gamma_2. \end{array} \right\}$$

If we define the indexes sets,

$$\begin{aligned} I &= [i \in [1, 2, \ldots, n\}/x_i \in \Omega\}, \\ I_1 &= [l \in [1, 2, \ldots, n\}/x_l \in \Gamma_1\}, \\ I_2 &= [k \in [1, 2, \ldots, n\}/x_k \in \Gamma_2\}. \end{aligned}$$

For $u_K(x) \in P^1(\tau^h)$ with $u_K(x) = \alpha$ and for $x \in \Gamma_1$ we want to find the solution in this form,

$$u_K(x) = \sum_{i \in I \cup I_2} \lambda_i \varphi_i(x) + \sum_{l \in \Gamma_1} \alpha \varphi_l(x), \tag{4.10}$$

with λ_i unknown constants for $i \in [1, 2, \ldots, n\}$ and applying the operator ∇ to u_K we have

$$\begin{aligned} \nabla u_E &= \nabla \left(\sum_{i \in I \cup I_2} \lambda_i \varphi_i(x) + \sum_{l \in \Gamma_1} \alpha \varphi_l(x) \right), \\ &= \sum_{i \in I \cup I_2} \lambda_i \nabla \varphi_i(x) + \sum_{l \in \Gamma_1} \alpha \nabla \varphi_l(x). \end{aligned}$$

Now, we replace u_K and $\nabla(u_K)$ in the equation (4.9)

$$\int_\Omega K \left(\sum_{i \in I \cup I_2} \lambda_i \nabla\varphi_i(x) + \sum_{l \in \Gamma_1} \alpha \nabla\varphi_l(x) \right) \nabla v \, d\Omega = \int_\Omega f v \, d\Omega + \int_{\Gamma_2} K \, \beta\eta v,$$

passing the element that do not depend on λ_i for all $i \in [1, 2, \dots, n\}$ to the right side of the equation

$$\int_\Omega K \left(\sum_{i \in I \cup I_2} \lambda_i \nabla\varphi_i(x) \right) \nabla v \, d\Omega = \int_\Omega f v \, d\Omega + \int_{\Gamma_2} K \, \beta\eta v \, dS - \sum_{l \in \Gamma_1} \int_\Omega K \, (\alpha \nabla\varphi_l(x)) \nabla v \, d\Omega,$$

and changing the sum by an integral, we obtain:

$$\sum_{i \in I \cup I_2} \int_\Omega K \, (\lambda_i \nabla\varphi_i(x)) \nabla v \, d\Omega = \int_\Omega f v \, d\Omega + \int_{\Gamma_2} K \, \beta\eta v \, dS - \sum_{l \in \Gamma_1} \int_\Omega K \, (\alpha \nabla\varphi_l(x)) \nabla v \, d\Omega. \quad (4.11)$$

If we take $v(x) = \varphi_j$ with $j \in I$ in (4.11), we obtain the Galerkin formulation for the minimum compliance design,

$$\sum_{i \in I \cup I_2} \int_\Omega K \, (\lambda_i \nabla\varphi_i(x)) \nabla\varphi_j \, d\Omega = \int_\Omega f \varphi_j \, d\Omega + \int_{\Gamma_2} K \, \beta\eta\varphi_j - \sum_{l \in \Gamma_1} \int_\Omega K \, (\alpha \nabla\varphi_l(x)) \nabla\varphi_j \, d\Omega. \quad (4.12)$$

To state the problem in a compact way, we define the following matrix and vectors

$$\begin{aligned} A_{ij} &= \int_\Omega K \, (\lambda_i \nabla\varphi_i(x)) \nabla\varphi_j \, d\Omega, \\ f_j &= \int_\Omega f \varphi_j \, d\Omega, \\ b_j &= \int_{\Gamma_2} K \, \beta\eta\varphi_j, \\ A_{lj} &= \int_\Omega K \, (\alpha \nabla\varphi_l(x)) \nabla\varphi_j \, d\Omega. \end{aligned}$$

The stationary heat equation (4.12) could be written in matrix form like

$$A_{11} \overrightarrow{\lambda} = \overrightarrow{f} + \overrightarrow{b} - A_{12} \overrightarrow{\alpha}. \quad (4.13)$$

And defining the block matrix

$$A = \left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array} \right],$$

where A_{11} contains the $I \cup I_2$ indexes, A_{12} contains I_1 and also writing $\overrightarrow{f}$ and $\overrightarrow{u}$ like block vectors we have

$$F = \left[\begin{array}{c} \overrightarrow{f} + b \\ \hline \overrightarrow{f}_{\Gamma_1} \end{array} \right] \qquad \text{and} \qquad u = \left[\begin{array}{c} \overrightarrow{\lambda} \\ \hline \overrightarrow{\alpha} \end{array} \right].$$

The equation (4.13) can be written in block matrix form,

$$Au = F. \quad (4.14)$$

Now, we replace u_K defined in (4.10) in the first term of the minimum compliance design

$$\begin{aligned}\int_\Omega f\, u_K(x)\, dx &= \int_\Omega f \left(\sum_{i\in I\cup I_2} \lambda_i \varphi_i(x) + \sum_{l\in\Gamma_1} \alpha\varphi_l(x) \right) dx\\ &= \sum_{i\in I\cup I_2} \int_\Omega f\lambda_i\varphi_i(x)\, dx + \sum_{l\in\Gamma_1} \int_\Omega \alpha f\varphi_l(x)\, dx\\ &= \sum_{i\in I\cup I_2} \int_\Omega (f\varphi_i(x))\lambda_i\, dx + \sum_{l\in\Gamma_1} \int_\Omega \alpha(f\varphi_l(x))\, dx\\ &= \overrightarrow{f}^\top \overrightarrow{\lambda} + \overrightarrow{f_{\Gamma_1}}^\top \alpha,\end{aligned}$$

we also replace (4.10) in the second term of (4.7) to get,

$$\begin{aligned}\int_{\Gamma_2} \beta(x) u_K(x)\, dx &= \int_{\Gamma_2} \beta(x) \sum_{k\in I_2} \lambda_k\varphi_k(x)\, dx\\ &= \sum_{k\in I_2} \int_{\Gamma_2} \beta(x)\lambda_k\varphi_k(x)\ dx\\ &= \overrightarrow{b}^\top \overrightarrow{\lambda}_{\Gamma_2},\end{aligned}$$

where $\overrightarrow{\lambda}_{\Gamma_2}$ contains λ_k with $k \in I_2$ indexes.

The minimum compliance design in (4.8) in matrix form is:

$$\begin{cases} \min\limits_{K\in K_{ad}} \left(\overrightarrow{f}^\top \overrightarrow{\lambda^K} + \overrightarrow{f_{\Gamma_1}}^\top \overrightarrow{\alpha} + \overrightarrow{b}^\top \overrightarrow{\lambda}^K_{\Gamma_2} \right),\\ \text{subject to}\\ A_{11}\overrightarrow{\lambda^K} = \overrightarrow{f} + \overrightarrow{b} - A_{12}\overrightarrow{\alpha},\\ K \in K_{ad}.\end{cases}$$

In the next chapter, we show the results of two topology optimization experiments for the minimum compliance design, which due to its formulation has multiscale properties. In each iteration of the optimization, we solve a high-contrast multiscale finite element problem. In order to improve the computation time of the numeric solution we apply the two-levels domain decomposition and compare the iteration count of the PCG method.

Chapter 5

Topology Optimization for the Heat Equation

In this chapter we present the results for the minimum compliance design for the stationary heat equation using the two-levels preconditioner stated in the previous chapters. Let $\Omega \subset \mathbb{R}^2$ a polygonal domain with boundary $\partial\Omega = \Gamma_1 \cup \Gamma_2$. Set a Dirichlet condition in Γ_1 and a Neumann condition in Γ_2. The problem statement is to find $K_{ad} \subseteq \{K \,|\, K : \Omega \to \mathbb{R}\}$ such that,

$$J = \min_{K \in K_{ad}} \left(\int_{\Omega} f\, u_K(x)\, d\,\Omega + \int_{\Gamma_2} \beta(x)\, u_K(x)\, d\,S \right),$$

where, given $K \subseteq K_{ad}$, u_K is the solution of,

$$\begin{cases} -\operatorname{div}(K \cdot \nabla u_K) = f, & x \in \Omega, \\ K \cdot \nabla u_K = \beta(x), & x \in \Gamma_2, \\ u_K = \alpha(x), & x \in \Gamma_1. \end{cases}$$

We begin with a brief reminder of the two-levels domain decomposition preconditioner. The preconditioned operator is $M_{Heat}^{-1}A$ where the preconditioned matrix if defined by,

$$M_{Heat}^{-1} = M_{Heat,1}^{-1} + M_{Heat,2}^{-1}, \tag{5.1}$$

the part corresponding to the first level is

$$M_{Heat,1}^{-1} r = \sum_{i=1}^{N} R_i A_i^{-1} R_i^{\top} r, \tag{5.2}$$

and the part corresponding to the second (or coarse) level is

$$M_{Heat,2}^{-1} r = R_0 A_0^{-1} R_0^{\top} r, \tag{5.3}$$

where $A_0 = RAR^T$. Here the matrix R is a such that each column is a coarse basis function. The basis functions are of the form $\chi_i \psi_\ell$ where χ_i is a partition of unity function and ψ_ℓ are eigenvalues of a generalized eigenvalue problem.

Now we present the performance results of the two-levels preconditioner for two topology optimization problems for the heat equation in two dimensions.

5.1 Uniformly Heated Plane Problem

Here we review the problem of a uniformly heated square plane with homogeneous Dirichlet boundary condition, as in [24]. For the FEM discretization we use a squared mesh with a 10×10 coarse mesh and a 10×10 fine mesh. Note the scheme of the problem in Figure 5.1.

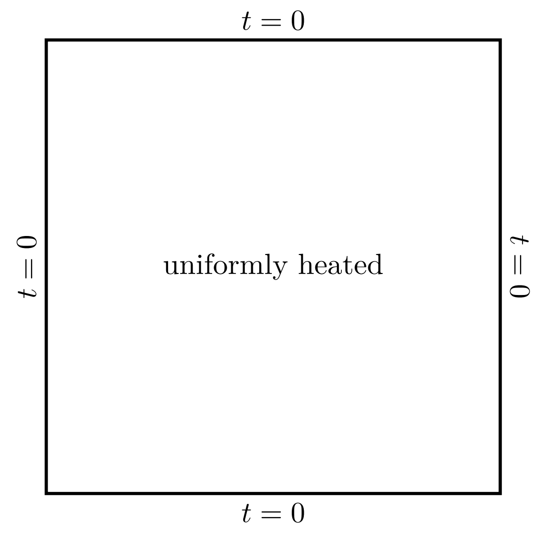

Figure 5.1: Uniformly heated square plane with homogeneous Dirichlet boundary condition.

All of the results below were made using a modified code from [2], including the two-levels preconditioner part. If we just count the time that our algorithm takes to converge, a direct method is faster to achieve the solution. This is because our algorithm is not optimized to perform fast. An optimized version of the two-levels preconditioning will be better in big problems where a direct method is not capable to achieve a solution.

The optimization algorithm is set to stop after 300 iterations. All the tested options achieve the same result in Figure 5.2. Note that the optimized result is a complex design with material in the black part. We show in Table 5.1 the optimized value for the topology optimization and the number of coarse basis calculations. In Figures 5.3 and 5.4 we can see the advantage of the two-levels preconditioner in the optimization.

Method	**Objective value**	**Coarse basis calculations**
No preconditioner	0.1849	—
Two-levels preconditioner	0.1848	300 (on every iteration)

Table 5.1: Results of the topology optimization for the example described in Figure 5.1.

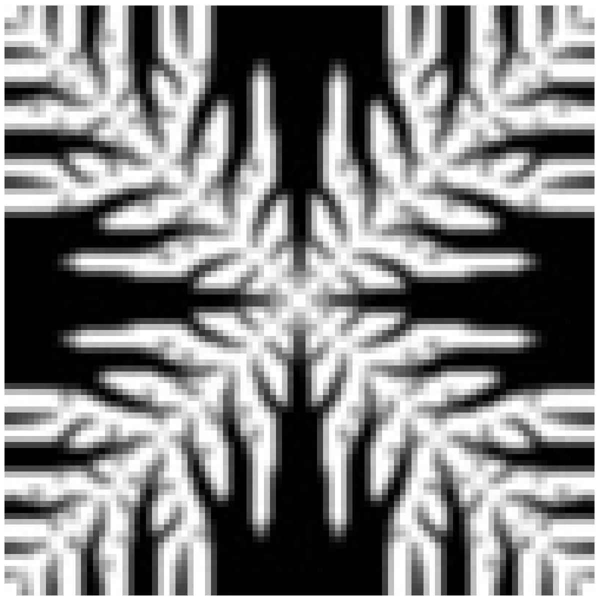

Figure 5.2: Topology optimization density plot for the heat equation with homogeneous Dirichlet boundary condition.

Note that the conjugate gradient method takes an average of 500 iterations to converge, even as this iterations are fairly quick this is a large amount of time to achieve the solution. In Figure 5.4 the PCG iterations are around 20 which is a great improvement.

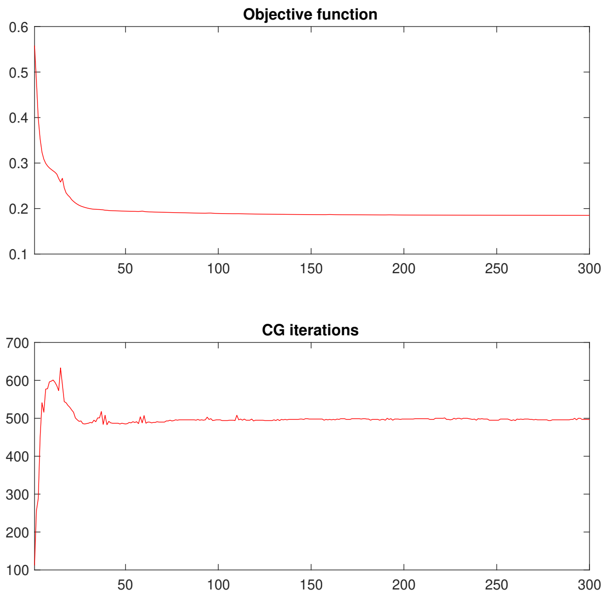

Figure 5.3: Topology optimization evolution of parameters for the heat equation with homogeneous Dirichlet boundary condition. Without preconditioners, note the number of iterations of the conjugate gradient method.

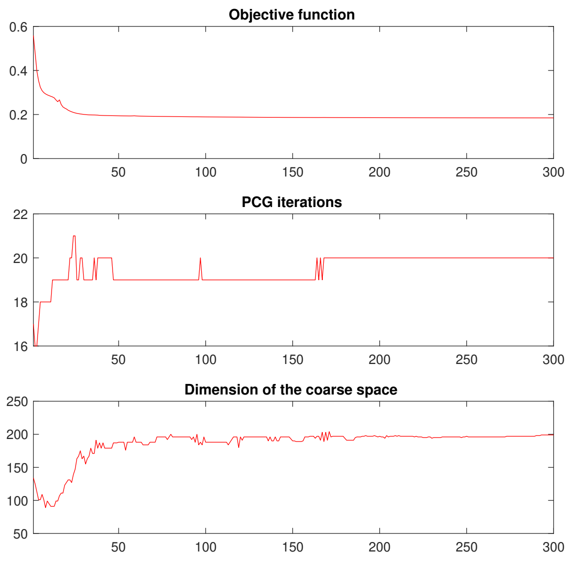

Figure 5.4: Topology optimization evolution of parameters for the heat equation with homogeneous Dirichlet boundary condition. Using the two-levels preconditioner.

5.2 Reusing the Coarse Matrix in the Optimization Iterations

The eigenvectors calculation problem is of order n^3, which is quite expensive. The topology optimization design tends to stabilize across the iterations, i.e. the high conductivity channels are fixed over each ω_i, so we reuse the coarse matrix if the number of iterations of the PCG are below a constraint.

We follow the next rules in order to recalculate the coarse basis:

- For the first two topology optimization iterations, compute the coarse basis.
- If the last PCG iteration count $(i-1)$ is more than 1.2 times the mean of the last five PCG iteration count $(i-6,\ldots,i-2)$, compute the coarse basis.
- Each 25 topology optimization iterations, compute the coarse basis.

The MATLAB code for the coarse matrix recalculation is:

```
if loop == 1 || loop == 2
    meanpcgiter = 1;
elseif loop <= 6
    meanpcgiter = 1.2*mean(pcgiterations(1:loop-2));
else
    meanpcgiter = 1.2*mean(pcgiterations(loop-6:loop-2)); % up to 20%
       increase from the last 5 iterations mean
end

if loop == 1 || pcgiterations(loop-1) >= meanpcgiter || mod(loop,25) =
    = 0
    %calculate coarse basis
end
```

We present two examples using problem from Figure 5.1. In the first example we apply the coarse basis reusing algorithm with the usual eigenvectors calculation. In example two we apply the reusing algorithm in addition to the random eigenvectors calculation.

We can see from the results that the random eigenvectors calculation yields the same results as the usual eigenvectors calculation, with the advantage of a fewer computational cost.

Method	**Objective value**	**Coarse basis calculations**
Two-levels preconditioner	0.1848	14
Two-levels random preconditioner	0.1848	14

Table 5.2: Results of the topology optimization cases for the example described in Figure 5.1, reusing the coarse basis.

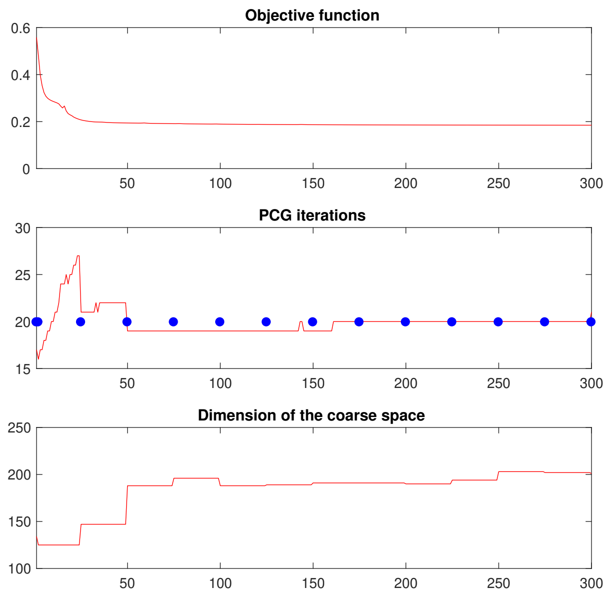

Figure 5.5: Topology optimization evolution of parameters for the heat equation with homogeneous Dirichlet boundary condition. Using the two-levels preconditioner and reusing the coarse basis.

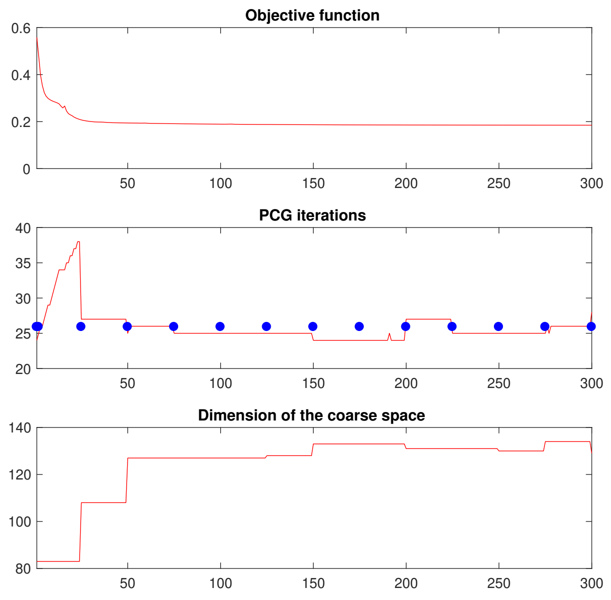

Figure 5.6: Topology optimization evolution of parameters for the heat equation with homogeneous Dirichlet boundary condition. Using the two-levels preconditioner with random basis generation and reusing the coarse basis.

The blue dots in Figure 5.5 and 5.6, represent the iteration in which the coarse basis are calculated. In the PCG iterations line plot we see how the iterations for the random eigenvectors tend to increase faster than in the usual eigenvectors calculation, this is because we are approximating the eigenvectors. Note the lower dimension of the coarse space in the random eigenvectors case, this also means a fewer computational cost in the calculations. The higher number of PCG iterations in the first optimization iterations are due to the fast changing coefficient in these optimization steps.

5.3 Heat Sink

Here we review the results for a uniformly heated square plane with homogeneous Dirichlet boundary condition and a heat sink in the border, as in Figure 5.7. See the optimized result in Figure 5.8.

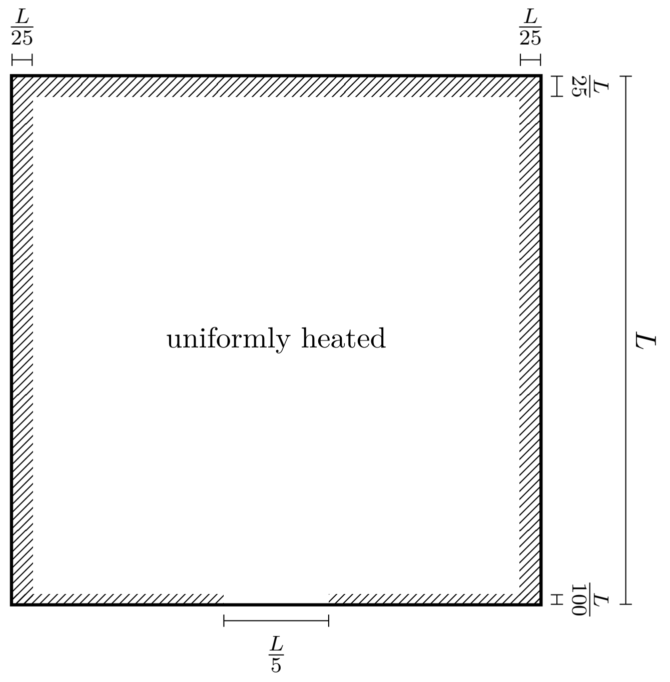

Figure 5.7: Uniformly heated square plane with homogeneous Dirichlet boundary condition and a heat sink in the border.

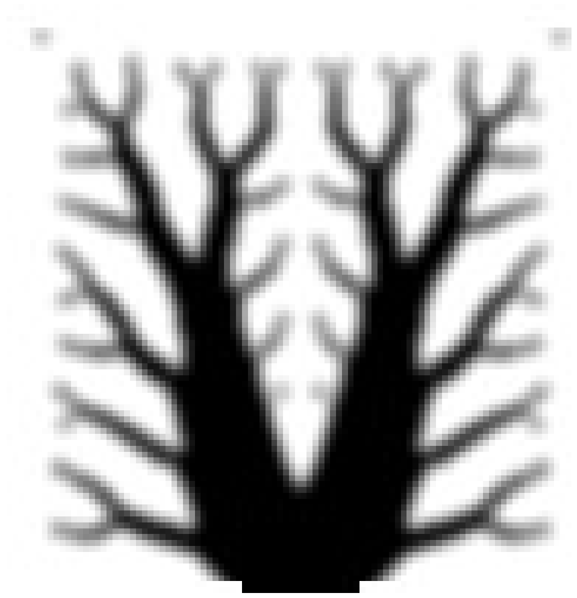

Figure 5.8: Topology optimization density plot for the heat equation with homogeneous Dirichlet boundary condition and a heat sink in the border.

The results for 350 topology optimization iterations are on Table 5.3 and 5.4. In Figure 5.9 we can see how without preconditioning the problem the iterations of the CG method are around 1800, and in Figure 5.10 PCG iterations are around 28 which is a great improve.

Method	**Objective value**	**Coarse basis calculations**
No preconditioner	2.5972	—
Two-levels preconditioner	2.5974	350 (on every iteration)

Table 5.3: Results of the topology optimization for the example described in Figure 5.7.

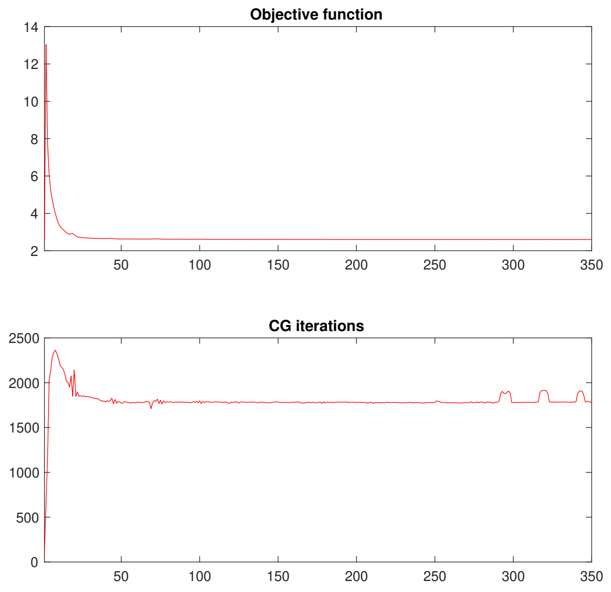

Figure 5.9: Topology optimization evolution of parameters for the heat equation with homogeneous Dirichlet boundary condition, without preconditioners.

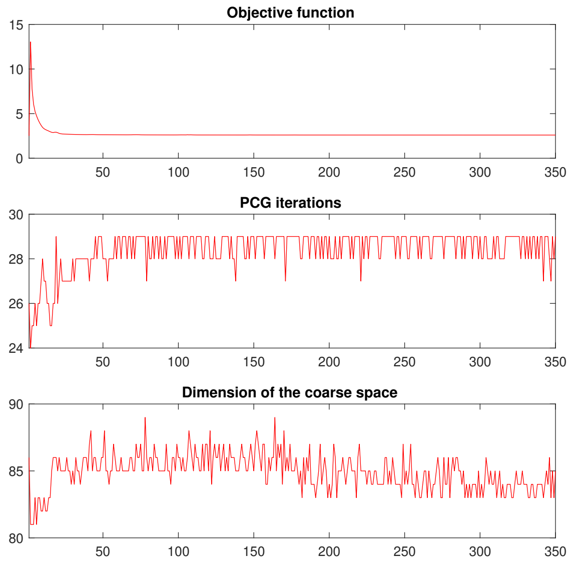

Figure 5.10: Topology optimization evolution of parameters for the heat equation with homogeneous Dirichlet boundary condition. Using the two-levels preconditioner.

In all this examples the maximum number of coarse basis elements is set to 6, and in Figure 5.12 the number of snapshots used is set to 18, this makes the random case very cheap in computational terms.

Method	**Objective value**	**Coarse basis calculations**
Two-levels preconditioner	2.5974	19
Two-levels random preconditioner	2.5972	19

Table 5.4: Results of the topology optimization cases for the example described in Figure 5.7, reusing the coarse basis.

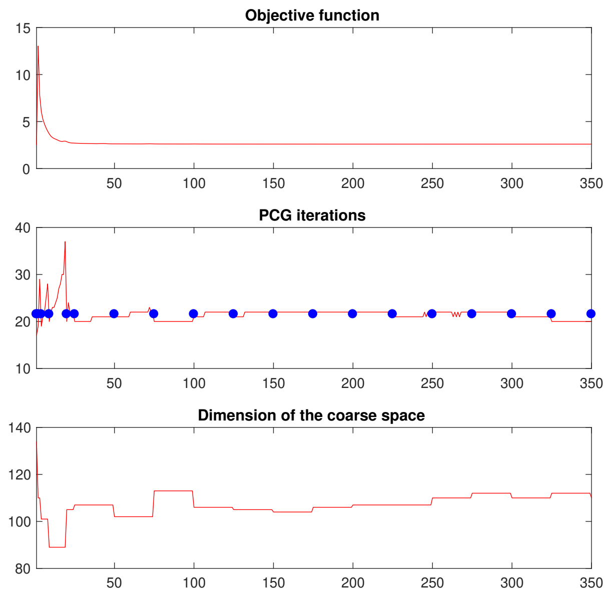

Figure 5.11: Topology optimization evolution of parameters for the heat equation with homogeneous Dirichlet boundary condition. Using the two-levels preconditioner and reusing the coarse basis.

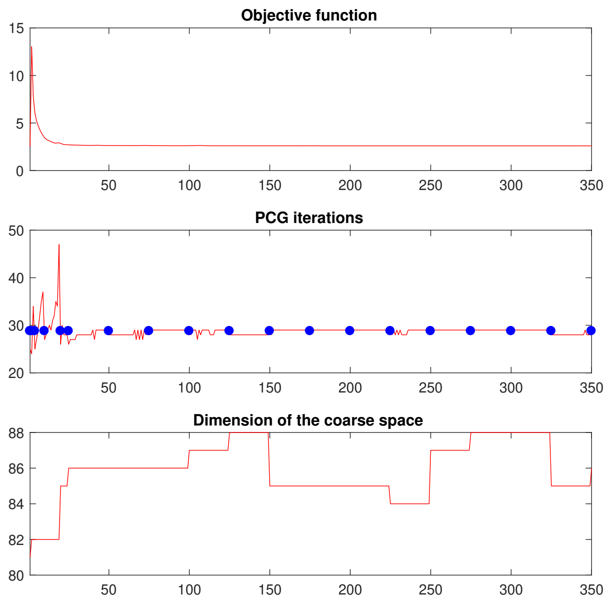

Figure 5.12: Topology optimization evolution of parameters for the heat equation with homogeneous Dirichlet boundary condition. Using the two-levels preconditioner with random basis generation and reusing the coarse basis.

In Figures 5.11 and 5.12 we can see how the PCG method takes more iterations to converge in the first steps of the optimization process. This is due to the fast changing design in the first topology optimization iterations. The PCG iterations tends to stabilize as the design get fixed (with minimum changes).

5.4 Modified Heat Sink

Here we review the results for a uniformly heated square plane with homogeneous Dirichlet boundary condition and two heat sinks in the border, as in Figure 5.13. All of the results below were made using a modified code from [2], including the two-levels preconditioner part.

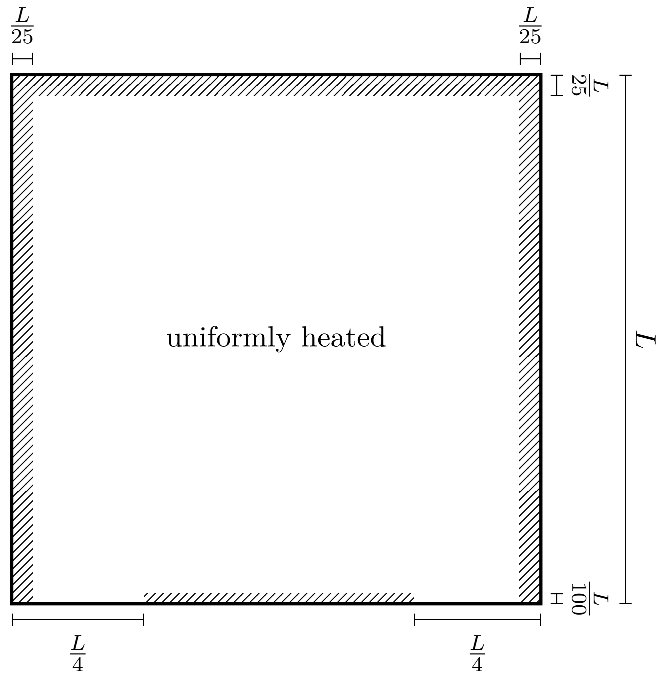

Figure 5.13: Uniformly heated square plane with homogeneous Dirichlet boundary condition and two heat sinks in the border.

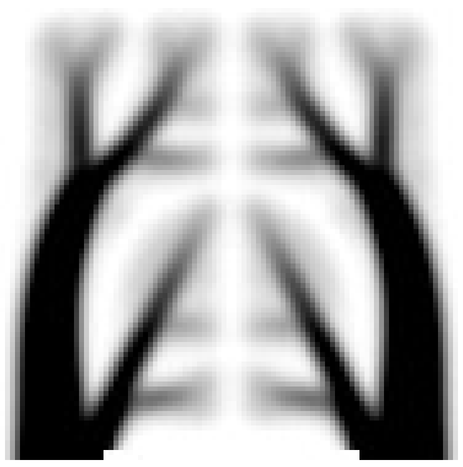

Figure 5.14: Density plot using one eigenvector to generate the coarse basis.

We explore varying the number of eigenvectors used to generate the coarse space. We set in the first example two eigenvectors and four eigenvectors in the second example. We see how we can achieve lower PCG iterations using more eigenvectors. The improvement is not big enough because the structure of this example requires a small number of eigenvectors to perform well.

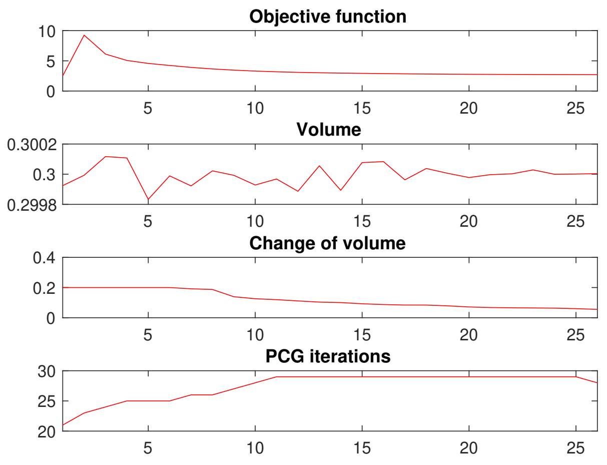

Figure 5.15: Evolution of parameters using one eigenvector to generate the coarse basis.

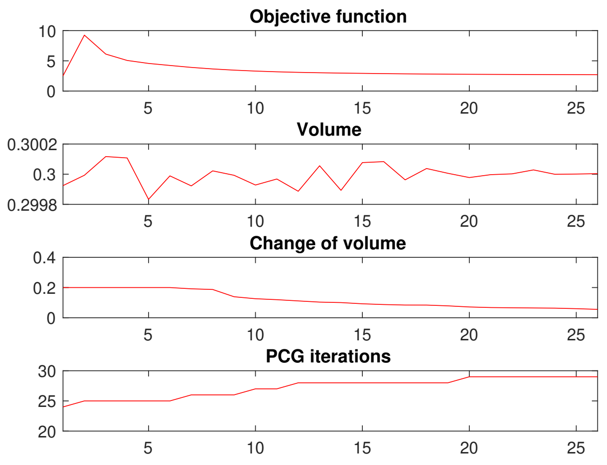

Figure 5.16: Evolution of parameters using a maximum of two eigenvectors to generate the coarse basis.

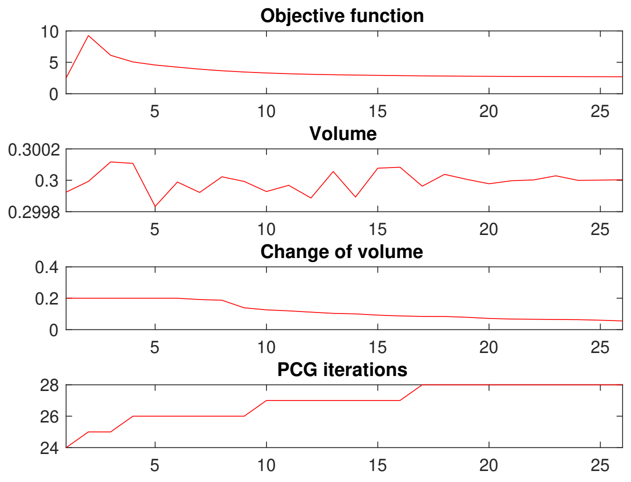

Figure 5.17: Evolution of parameters using a maximum of four eigenvectors to generate the coarse basis.

Chapter 6

Conclusions and Recommendations

6.1 Conclusions

In topology optimization problems it is usual to obtain designs with a multiscale structure for the high-conductivity part of the material and also a high-contrast property is present due to the density formulation. These problems make the linear system hard to solve via direct methods. The use of two-levels preconditioners allows a significant improvement in calculation time for big industrial problems.

It is important to work with the heat problem in topology optimization, since the results can be applied to accelerate the calculations in elasticity problems, as in [35, 36]. The randomized construction proposed in [8] represent a significant reduction in the cost of computing the coarse basis functions that generate the coarse space as this leads to a smaller eigenvalue problem. Using the randomized algorithm does not deteriorate the performance of the method as it was verified here. This is our main contribution on this thesis. We have verified that the two-levels domain decomposition preconditioner with a GMsFEM second level perform well for topology optimization of the heat equation. Specially the randomized algorithm that have not being used before in topology optimization. These results allowed us to design several preconditioners for the similar elasticity problem, see [35] and [36].

The coarse basis calculation problem is of order n^3, which is quite expensive for each iteration. The topology optimization design tends to stabilize across the iterations, i.e. the high conductivity channels are fixed over each ω_i, so we reuse the coarse matrix if the iterations of the PCG are below a constraint allowing a significant reduction of computational cost. This in addition to the randomized construction provides an interesting way to improve performance in the solution calculation for big industrial problems.

6.2 Recommendations

An interesting future work would be to study the selection criteria of the optimum number of the basis of the coarse space in each iteration of the topology optimization. Choosing these basis requires a careful selection of the eigenvalues that might be different between inner and border neighborhoods ω_i.

Choosing the right-hand side in the eigenvalue problem for the heat case is simple while the selection of the right-hand side for the elasticity case requires a deeper analysis.

In order to be competitive in calculation time, it is necessary to implement the method in a high-level programming language such as Fortran or C using optimized libraries. It is also important to take advantage of the possibility of parallelizing the preconditioners and the reuse of certain matrices as the design of the optimization stabilizes.

Bibliography

[1] Joe Alexandersen and Boyan S. Lazarov. Robust topology optimisation of microstructural details without length scale separation - using a spectral coarse basis preconditioner. *CoRR*, abs/1411.3923, 2014.

[2] E. Andreassen, A. Clausen, M. Schevenels, B.S. Lazarov, and O. Sigmund. Efficient topology optimization in MATLAB using 88 lines of code. *Structural and Multidisciplinary Optimization*, 2010.

[3] M.P. Bendsøe and O. Sigmund. *Topology Optimization: Theory, Methods, and Applications*. Springer Berlin Heidelberg, 2003.

[4] Dietrich Braess. *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*. Cambridge University Press, 2007.

[5] S. Brenner and R. Scott. *The Mathematical Theory of Finite Element Methods*. Texts in Applied Mathematics. Springer New York, 2007.

[6] H. Brezis. *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Universitext. Springer New York, 2010.

[7] Marco Buck, Oleg Iliev, and Heiko Andrä. Domain decomposition preconditioners for multiscale problems in linear elasticity. *Numerical Linear Algebra with Applications*, 25(5):e2171, 2018. e2171 nla.2171.

[8] V. Calo, Y. Efendiev, J. Galvis, and G. Li. Randomized oversampling for generalized multiscale finite element methods. *Multiscale Modeling & Simulation*, 14(1):482–501, 2016.

[9] Y. Efendiev and J. Galvis. Domain decomposition preconditioner for multiscale high-contrast problems. In *Proceedings of DD19*, 2009.

[10] Y. Efendiev and J. Galvis. Eigenfunctions and multiscale methods for Darcy problems. Technical report, ISC, Texas A&M University, 2010.

[11] Y. Efendiev and J. Galvis. A domain decomposition preconditioner for multiscale high-contrast problems. In Y. Huang, R. Kornhuber, O. Widlund, and J. Xu, editors, *Domain Decomposition Methods in Science and Engineering XIX*, volume 78 of *Lect. Notes in Comput. Science and Eng.*, pages 189–196. Springer-Verlag, 2011.

[12] Y. Efendiev and J. Galvis. Domain decomposition preconditioner for multiscale high-contrast problems. In Y. Huang, R. Kornhuber, O. Widlund, and J. Xu, editors, *Domain Decomposition Methods in Science and Engineering XIX*, volume 78 of *Lecture Notes in Computational Science and Engineering*, pages 189–196, Berlin, 2011. Springer-Verlag.

[13] Y. Efendiev, J. Galvis, and T. Hou. Generalized multiscale finite element methods. *Journal of Computational Physics*, 251:116–135, 2013.

[14] Y. Efendiev, J. Galvis, G. Li, and M. Presho. Generalized multiscale finite element methods. oversampling strategies. *International Journal for Multiscale Computational Engineering, accepted*, 2013.

[15] Y. Efendiev, J. Galvis, and X. H. Wu. Multiscale finite element methods for high-contrast problems using local spectral basis functions. *Journal of Computational Physics*, 230(4):937–955, 2011.

[16] Y. Efendiev and T.Y. Hou. *Multiscale Finite Element Methods: Theory and Applications*. Surveys and Tutorials in the Applied Mathematical Sciences. Springer New York, 2009.

[17] Yalchin Efendiev, Juan Galvis, and Xiao-Hui Wu. Multiscale finite element methods for high-contrast problems using local spectral basis functions. *Journal of Computational Physics*, 230(4):937 – 955, 2011.

[18] L.C. Evans. *Partial Differential Equations*. Graduate studies in mathematics. American Mathematical Society, 1998.

[19] J. Galvis. *Introdução aos Métodos de Decomposição de Domínio*. IMPA, 2009.

[20] J. Galvis and Y. Efendiev. Domain decomposition preconditioners for multiscale flows in high contrast media. *SIAM J. Multiscale Modeling and Simulation*, 8:1461–1483, 2010.

[21] J. Galvis and Y. Efendiev. Domain decomposition preconditioners for multiscale flows in high contrast media. reduced dimension coarse spaces. *SIAM J. Multiscale Modeling and Simulation*, 8:1621–1644, 2010.

[22] Juan Galvis, Eric Chung, Yalchin Efendiev, and Wing Tat Leung. On overlapping domain decomposition methods for high-contrast multiscale problems, 05 2017.

[23] Juan Galvis and Yalchin Efendiev. Domain decomposition preconditioners for multiscale flows in high-contrast media. *Multiscale Modeling & Simulation*, 8(4):1461–1483, 2010.

[24] T. Gao, W.H. Zhang, J.H. Zhu, Y.J. Xu, and D.H. Bassir. Topology optimization of heat conduction problem involving design-dependent heat load effect. *Finite Elements in Analysis and Design*, 44(14):805 – 813, 2008.

[25] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.*, 53(2):217–288, May 2011.

[26] Ronald Hoppe. Script on finite element methods course. `https://www.math.uh.edu/~rohop/Fall_16/`, 2016.

[27] Ari (https://mathoverflow.net/users/673/ari). What are "variational crimes"; and who coined the term? MathOverflow. URL:https://mathoverflow.net/q/26066 (version: 2017-06-15).

[28] C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite Element Method*. Cambridge University Press, 1987.

[29] Hans Petter Langtangen and Anders Logg. *Solving PDEs in Python*. Springer, 2017.

[30] Boyan S. Lazarov. Topology optimization using multiscale finite element method for high-contrast media. In Ivan Lirkov, Svetozar Margenov, and Jerzy Waśniewski, editors, *Large-Scale Scientific Computing*, pages 339–346, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[31] Boyan S. Lazarov, Fengwen Wang, and Ole Sigmund. Length scale and manufacturability in density-based topology optimization. *Archive of Applied Mechanics*, 86(1):189–218, 01 2016.

[32] M.J.H. Meijboom. *Topology optimization of dynamic problems.* DCT rapporten. Technische Universiteit Eindhoven, 2003. DCT 2003.108.

[33] Michael Presho and Juan Galvis. A mass conservative generalized multiscale finite element method applied to two-phase flow in heterogeneous porous media. *Journal of Computational and Applied Mathematics*, 296:376 – 388, 2016.

[34] A. Samuelsson and N.E. Wiberg. *Finite Element Method: Basics.* Studentlitteratur, 1998.

[35] S. Serrano. *Topology Optimization for the Elasticity Problem.* Masters thesis, Universidad Nacional de Colombia, 2019.

[36] S. Serrano, M. Zambrano, B. Lazarov, and J. Galvis. Fast multiscale contrast independent preconditioner for linear elastic topology optimization problems. In preparation.

[37] Ole Sigmund and Kurt Maute. Sensitivity filtering from a continuum mechanics perspective. *Structural and Multidisciplinary Optimization*, 46(4):471–475, 10 2012.

[38] Krister Svanberg. The method of moving asymptotes—a new method for structural optimization. *International Journal for Numerical Methods in Engineering*, 24(2):359–373, 1987.

[39] Andrea Toselli and Olof B. Widlund. *Domain Decomposition Methods.* Science, 2006.

[40] user53153. Interpret the terms in Strang's second lemma. Mathematics Stack Exchange. URL:https://math.stackexchange.com/q/310714 (version: 2013-02-21).

[41] Fengwen Wang, Boyan Stefanov Lazarov, and Ole Sigmund. On projection methods, convergence and robust formulations in topology optimization. *Structural and Multidisciplinary Optimization*, 43(6):767–784, 06 2011.

[42] J. Galvis Y. Efendiev and P. Vassilevski. Spectral element agglomerate algebraic multigrid methods for elliptic problems with high-contrast coefficients. In Y. Huang, R. Kornhuber, O. Widlund, and J. Xu, editors, *Domain Decomposition Methods in Science and Engineering XIX*, volume 78 of *Lecture Notes in Computational Science and Engineering*, pages 407–414. Springer-Verlag, 2011.

Appendix A

Codes for Topology Optimization

In this appendix we show some of the codes used in this work. We begin with a simple code in FEniCS which we used to test the software abilities to numerically solve PDEs. FEniCS can be used in future work making parts of the programming easier. Then we show an example of the minimum compliance design in one dimension. This code was used to practice the implementation basics in the FEM.

A.1 2D Heat Transfer in FEniCS

Now we present an example of an isotropic body. Take Ω a rectangular plate made of an orthotropic material, with heat conduction coefficient,

$$\mathbf{K}(x,y) = \begin{bmatrix} 0.1y & 0 \\ 0 & x \end{bmatrix},$$

the heat source is the constant function $f(x,y) = 2$. On the right side of the rectangle we set $u = y(y-3)$, and on the other sides we set $u = 0°$. See Figure A.1 for an illustration.

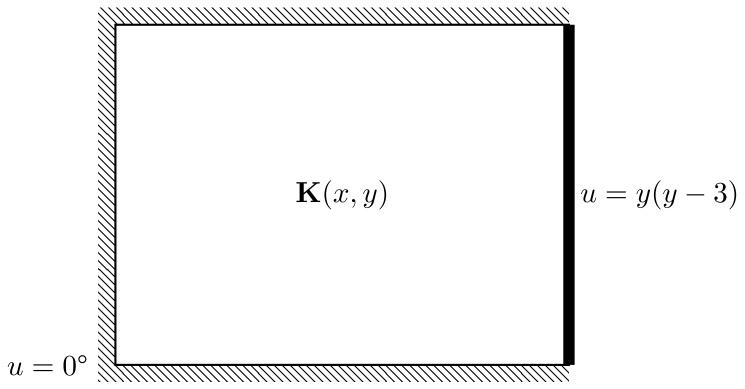

Figure A.1: Problem of a plate made of orthotropic material with heat input f and prescribed temperature u at the border.

Solving the problem using FEniCS we get the heat distribution in Figure A.2. FEniCS allows a simple codding simulation of PDEs due to its advanced commands and the clean programming in Python. Also, Python let us integrate advanced packages easily.

Here we have the code in FEniCS for the heat transfer in 2D. This code is based on the example of the conductivity equation described in [29] with small modifications.

```
from fenics import *

# Create mesh and define function space
mesh = RectangleMesh(Point(0,0), Point(4, 3), 80, 60, "right")
V = FunctionSpace(mesh, 'P', 1)

# Define boundary condition
tol = 1E-14
u_r = Expression('x[1]*(x[1]-3)', degree=2)
def boundary_r(x, on_boundary):
        return on_boundary and not near(x[0], 4, tol)
bc_r = DirichletBC(V, u_r, boundary_r)

u_f = Constant(1)
def boundary_f(x, on_boundary):
    return on_boundary and near(x[0], 4, tol)
bc_f = DirichletBC(V, u_f, boundary_f)

bc = [bc_f, bc_r]

# Define variational problem
u = TrialFunction(V)
v = TestFunction(V)
f = Constant(2.0)
E = as_matrix(((Expression('0.1*x[1]', degree=1), 0),
               (0, Expression('x[0]', degree=1))))
a = inner(E*grad(u),grad(v))*dx
L = f*v*dx

# Compute solution
u = Function(V)
solve(a == L, u, bc)

# Plot solution
parameters["plotting_backend"] = "matplotlib"
from matplotlib import pyplot
pyplot.figure()
ax = plot(u)
pyplot.title("")
pyplot.colorbar(ax)
pyplot.show()
```

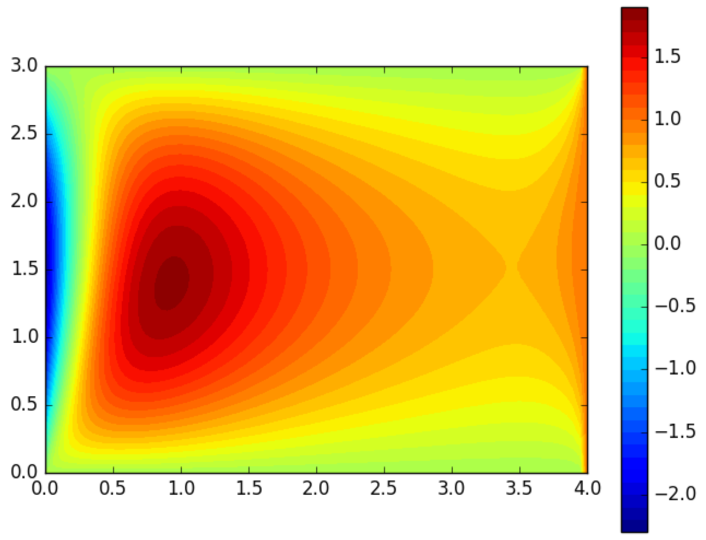

Figure A.2: Distribution of temperature in an orthotropic plate with conduction coefficient **K**, a constant heat source and prescribed temperature at the border. The solution is computed using FEniCS.

A.2 Minimum Compliance Design for the One Dimensional Heat Equation

Given the set $\mathcal{K}_{ad} \subseteq \{K | K : (a,b) \to \mathbb{R}^+\}$ the problem statement is to find the optimal conductivity coefficient K^* that solves

$$J = \min_{K \in \mathcal{K}_{ad}} \left[\int_a^b f u_K + \beta u_K(b) \right], \tag{A.1}$$

where, given $K \in \mathcal{K}_{ad}$, the temperature u_K is the solution of

$$\begin{cases} -\left(K(x)u_K'(x)\right)' = f(x) & x \in (a,b), \\ u_K(a) = \alpha \qquad K(b)u_K'(b) = \beta. \end{cases} \tag{A.2}$$

We note u_K instead of u to emphasize the dependence of u on the conductivity coefficient K. The weak form of (A.2) is obtained by multiplying a test function $v \in C^\infty(0,1)$ with $v(0) = 0$ and then integrating by parts using the boundary condition,

$$-\left(K(x)u_K'(x)\right)' = f(x),$$
$$-\left(K(x)u_K'(x)\right)' v(x) = f(x)v(x),$$

here we drop the x to simplify notation and work on the left side

$$-\int_a^b (Ku_K')'\,v = -\left[K(b)u_K'(b)v(b) - K(a)u_K'(a)v(a)\right] + \int_a^b Ku_K'v'$$
$$= \int_a^b Ku_K'v' - \beta v(b),$$

because $v(a) = 0$ and $K(b)u_K'(b) = \beta$,

$$\int_a^b Ku_K'v' = \int_a^b fv + \beta v(b).$$

Now we define the weak formulation as: given $K \in \mathcal{K}_{ad}$ find u_K as the solution of

$$\begin{cases} u_K : (a,b) \to \mathbb{R}^+ \\ \displaystyle\int_a^b Ku_K'v' = \int_a^b fv + \beta v(b) \quad \text{for all} \quad v \in C^\infty(a,b) \quad \text{with} \quad v(a) = 0 \\ u_K(a) = \alpha. \end{cases} \tag{A.3}$$

We take the interval $[a,b] = [0,1]$, for the finite element discretization and set a triangulation τ of $[0,1]$ with h increments. See Figure A.3 for an illustration of the triangulation.

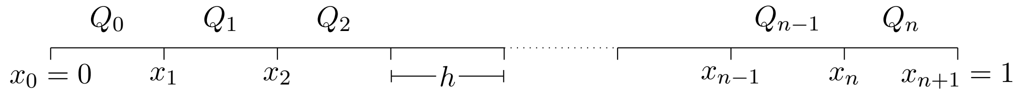

Figure A.3: Elements over the segment $[0,1]$.

Consider the set of piecewise linear functions on the triangulation

$$P_1^h = \{v : [a,b] \to \mathbb{R} \ / \ \mathrm{v}|_{(\mathrm{x_{i-1}},\mathrm{x_i})} \quad \text{is linear and continuous}\},$$

and observe that we can write

$$P_1^h = \operatorname{span}\{\varphi_i\}_{i=0}^{n+1},$$

where

$$\varphi_i(x) = \begin{cases} 1, & \text{for } x = x_i \\ 0, & \text{for } x \neq x_i \\ \text{linear}, & \text{for } (x_{i-1}, x_{i+1}), \quad i = 1, 2, \ldots, n+1. \end{cases} \tag{A.4}$$

The functions φ_i are illustrated in Figure A.4.

Now $v \in P_1^h$ if, and only if, $v = \gamma_0\varphi_0 + \ldots \gamma_{n+1}\varphi_{n+1}$ with $\gamma_i \in \mathbb{R}$, and observe that by equation (A.4) we get $v(x_i) = \gamma_i$. In (A.3), replacing $u_K : (0,1) \to \mathbb{R}^+$ by $u_K = \alpha + \gamma_1\varphi_1 + \ldots + \gamma_{n+1}\varphi_{n+1}$ and $v \in C^\infty(0,1)$ with $v(0) = 0$ by $v = \varphi_i$ for $i = 1, \ldots, n+1$, we get

$$\int_0^1 K\left(\alpha\varphi_0 + \sum_{j=1}^{n+1}\gamma_j\varphi_j\right)'\varphi_i' = \int_0^1 f\varphi_i + \beta\varphi_i(1),$$

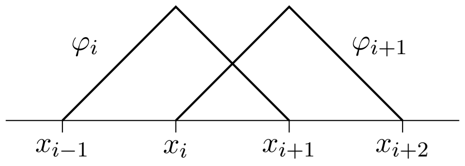

Figure A.4: Illustration of basis functions φ_i and φ_{i+1}, note also the nodes x_{i-1}, x_i, x_{i+1} and x_{i+2}.

working out the first integral we get

$$\sum_{j=1}^{n+1}\left(\int_0^1 K\varphi_j'\varphi_i'\right)\gamma_j = \int_0^1 f\varphi_i + \alpha\int_0^1 E\varphi_0'\varphi_i' + \beta\varphi_i(1),$$

and therefore, each equation of the linear system is

$$\sum_{j=1}^{n} a_{ij}\gamma_j = b_i - \alpha c_i + \beta d_i \quad \text{for } i = 1,\dots,n,$$

which in matrix form is given by

$$\mathbf{A}\vec{\gamma} = \vec{b} - \alpha\vec{c} + \beta\vec{d},$$

or equivalently

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix}\begin{bmatrix}\gamma_1\\ \vdots\\ \gamma_n\end{bmatrix} = \begin{bmatrix}b_1\\ \vdots\\ b_n\end{bmatrix} - \alpha\begin{bmatrix}c_1\\ \vdots\\ c_n\end{bmatrix} + \beta\begin{bmatrix}d_1\\ \vdots\\ d_n\end{bmatrix}.$$

Note that $\mathbf{A}$ depends on K, that is $\mathbf{A} = \mathbf{A}(K)$. Note also that

$$\begin{aligned}\int_0^1 fu_K &= \int_0^1 f(\alpha\varphi_0 + \sum_{j=1}^{N}\gamma_j\varphi_j)\\ &= \alpha\int_0^1 f\varphi_0 + \sum_{j=1}^{N}\left(\int_0^1 f\varphi_j\right)\gamma_j\\ &= \alpha\int_0^1 f\varphi_0 + b^{\intercal}\cdot\gamma.\end{aligned}$$

We conclude that problem (A.1) in matrix form is written as

$$\begin{cases}\min\limits_{K\in\mathcal{K}_{ad}} \alpha b_0 + \vec{b}^{\intercal}\cdot\vec{\gamma}^K + \beta\gamma_{N+1}^K,\\ \text{such that}\\ \mathbf{A}_K\vec{\gamma}^K = \vec{b} - \alpha\vec{c} + \beta\vec{d}.\end{cases} \tag{A.5}$$

With the matrix formulation (A.5) and taking care of the definition of the set $\mathcal{K}_{ad}$ we can compute the solution by using an iterative method to compute the minimum. We added the volume constraint and another constraint to force the coefficient $\mathbf{K}$ to be a continuous by parts function. As stated before, for illustration purposes, we used MATLAB to implement a FEM problem to solve (A.5). The program uses MATLAB's function `fmincon` to perform the minimization, this command calls the FEM script in each step, this is why is important to solve the PDE using FEM in the fastest time possible.

In this work, we seek to explore different methods to numerically solve the partial differential equation. In general, a preconditioned iterative method is used, e.g., PCG (Preconditioned Conjugate Gradient) or a similar method. We remark that due to the fact that (conductivity or elasticity) coefficient K or E is heterogeneous and with large variation across the domain, it is mandatory to use a well-designed preconditioner. In this work, we study the performance of high-contrast domain decomposition preconditioners introduced in [1] and in [17], for the iterative solution of the PDE. These preconditioned iterative methods are constructed by combining local solutions of the partial differential equation with a properly designed coarse-scale solution in order to reduce the number of iterations to convergence and also obtaining a number of iterations independent of the contrast.

Here we present the code and results of the minimum compliance design for the heat equation in 1D. The program was done in MATLAB mainly for its simplicity and the availability of useful commands. The code is organized into four files and it is not thought to have a good readability.

In the file `Jmin`, we use the MATLAB command `fmincon` to minimize the functional that is calculated in `mainopt`. We set a volume constraint of 0.5 and a restriction (see line 7 in `Jmin`) that force the coefficient E to be a continuous by parts function over $[0, 1]$, as we can see in Figure A.5.

file: `Jmin.m`

```
1  %%% Minimization of the functional (J) varying the coefficient (coeff)
2  c_0=zeros(N,1); %lower bound for coeff
3  c_1=ones(N,1); %upper bound for coeff
4  b=0.5; %percentage of material
5  a=1/N*c_1';
6  %obj=@(x)sum(x)-0.5*N;
7  obj_inequal=@(x)-sum(x.^3)+0.75*N;
8  obj_equal=@(x)0;
9  restr=@(x)deal(obj_inequal(x),obj_equal(x));
10 [c_min,fval]=fmincon(@(coeff) mainopt(N,coeff),c_1,a,b,[],[],c_0+eps,
       c_1-eps,restr,optimoptions('fmincon','MaxFunEvals',10000));
```

file: `mainopt.m`

```
function J=mainopt(N,coeff)
%%% Fixing parts of coeff (initial test)
%coeff=[var_coeff(1:5), ones(1,(N)/2), var_coeff(6:10)];

%%% Parameters: N=mesh points, coeff=conductivity coefficient
%%% Quadrature in [-1,1]
%%% Quadrature weights
omega_aux(1)=5/9;
omega_aux(2)=8/9;
omega_aux(3)=5/9;
%%% Quadrature points
```

```
p_aux(1)=-sqrt(15)/5;
p_aux(2)=0;
p_aux(3)=+sqrt(15)/5;
%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%% Elements
for i=1:N
     xini=(i-1)/N;
     xfin=i/N;
    E(i).xini=xini;
    E(i).xfin=xfin;
    omega=0.5*(xfin-xini)*omega_aux;
    x_q=xini+0.5*(1+p_aux)*(xfin-xini);
    % Linear transformation
    E(i).omega=omega;
    E(i).x_q=x_q;
    E(i).b1=(xfin-x_q)./(xfin-xini);     % Phi_i evaluation in x_q
    E(i).b2=(x_q-xini)./(xfin-xini);     % Phi_(i+1) evaluation in x_q
    E(i).b1d=[-1,-1,-1]./(xfin-xini); % Phi_i derived in x_q
    E(i).b2d=[1,1,1]./(xfin-xini); % Phi_i+1 derived in x_q
    E(i).dof=[i,i+1];    % DOF
end

%%%% Local matrices
for i=1:N
    b1=E(i).b1;
    b2=E(i).b2;
    b1d=E(i).b1d;
    b2d=E(i).b2d;
    x_q=E(i).x_q;
    w=E(i).omega;
     coef=1;
    %a11=(coef(1)*w(1)*b1d(1)*b1d(1))+(coef(2)*w(2)*b1d(2)*b1d(2))+(
        coef(3)*w(3)*b1d(3)*b1d(3));
    a11=sum(coef.*w.*b1d.*b1d);
    a12=sum(coef.*w.*b1d.*b2d);
    a21=sum(coef.*w.*b2d.*b1d);
    a22=sum(coef.*w.*b2d.*b2d);
    E(i).Alocal=[a11,a12;a21,a22];
     f=sin(20*x_q); % Forcing term
     f1=sum(w.*f.*b1);
     f2=sum(w.*f.*b2);
    E(i).flocal=[f1,f2];
end

b=zeros(N+1,1);

%%%% Right side coupling
```

```
for k=1:N
    flocal=E(k).flocal;
    dof=E(k).dof;
    b(dof,1)=b(dof,1)+flocal';
end

alpha_value=0;
beta_value=1;

%%%% Energy functional
J=funcionalj(E,N,b,coeff,alpha_value,beta_value);
```

file: `funcionalj.m`

```
function J=funcionalj(E,N,b,coeff,alpha_value,beta_value)

A=matrizglobalopt(E,N,coeff);

%%%% Dirichlet conditions
xd=zeros(N+1,1);
xd(1)=alpha_value;        % c vector
xd(N+1)=beta_value;
%%%% Modification of the global right side
bd=b-A*xd;        % A*xd is the c vector (in A we have E*Phi_0'*Phi_1')
%bd(N+1)=bd(N+1)+beta_value;

%%%% Solution of the linear system
int=2:(N);
xsol=xd; % We fix alpha_value on the solution
Aint=A(int,int);
bint=bd(int);

xaux=Aint\bint;
xsol(int)=xaux;

t=0:1/(N-1):1;
figure(1)
hold off
plot(t,coeff);
axis([0 1 0 1]);
pause(0.005)

%%%% Calculation of the functional J
J=dot(b,xsol)+beta_value*xsol(N+1);

% for k=1:N
%       x=[E(k).xini,E(k).xfin];
%       y=xsol(E(k).dof);
%     plot(x,y,'r-*')
%         hold on
% end
```

file: `matrizglobalopt.m`

```
function A=matrizglobalopt(E,N,coeff)

%%%% Global matrix
A=sparse(N+1,N+1);

for k=1:N
    % F(i) element contribution;
     Alocal=E(k).Alocal;
     dof=E(k).dof;
    A(dof,dof)=A(dof,dof)+coeff(k)*Alocal;
end
```

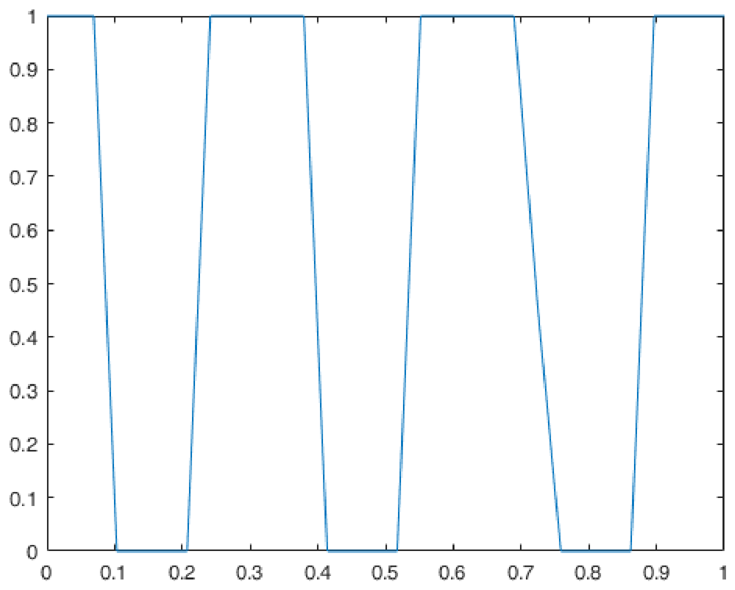

Figure A.5: Coefficient E for the Minimum Compliance Design with $f = \sin(20x)$, note that the function is constant by parts, thus it represents better a black and white coefficient.

A.3 2D Topology Optimization in MATLAB

Here we present our modifications to the 88 lines topology optimization code in MATLAB from [2] used to generate examples in Chapter 5.

A.3.1 Uniformly Heated Plane Problem

```
%%%% AN 88 LINE TOPOLOGY OPTIMIZATION CODE Nov, 2010 %%%%
%%%% MODIFIED HEAT EXAMPLE Jan,2019 %%%%
nelx = 100; nely = 100; volfrac = 0.5; penal = 3; rmin = 2; ft = 2;
beta = 1;
max_it_opt = 299;
%% MATERIAL PROPERTIES
k0 = 1; % Good thermal conductivity
kmin = 1e-3; % Poor thermal conductivity
%% PREPARE FINITE ELEMENT ANALYSIS
KE = 1/12*k0*[4 -1 -2 -1; -1 4 -1 -2; -2 -1 4 -1; -1 -2 -1 4];
nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
edofVec = reshape(nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
edofMat = repmat(edofVec,1,4)+repmat([0 nely+1 nely -1],nelx*nely,1);
iK = reshape(kron(edofMat,ones(4,1))',16*nelx*nely,1);
jK = reshape(kron(edofMat,ones(1,4))',16*nelx*nely,1);
% DEFINE LOADS AND SUPPORTS
F = 0.0001*ones((nely+1)*(nelx+1),1);
U = zeros((nely+1)*(nelx+1),1);
fixeddofs = union([1:(nely+1),1:(nely+1):(nelx+1)*(nely+1),(nely+1):(
    nely+1):(nelx+1)*(nely+1)],(nely+1)*(nelx)+1:(nelx+1)*(nely+1));  %
    Dirichlet boundary example
alldofs = 1:(nely+1)*(nelx+1);
freedofs = setdiff(alldofs,fixeddofs);
%% PREPARE FILTER
iH = ones(nelx*nely*(2*(ceil(rmin)-1)+1)^2,1);
jH = ones(size(iH));
sH = zeros(size(iH));
k = 0;
for i1 = 1:nelx
    for j1 = 1:nely
        e1 = (i1-1)*nely+j1;
        for i2 = max(i1-(ceil(rmin)-1),1):min(i1+(ceil(rmin)-1),nelx)
            for j2 = max(j1-(ceil(rmin)-1),1):min(j1+(ceil(rmin)-1),
                nely)
                e2 = (i2-1)*nely+j2;
                k = k+1;
                iH(k) = e1;
                jH(k) = e2;
                sH(k) = max(0,rmin-sqrt((i1-i2)^2+(j1-j2)^2));
            end
        end
    end
end
H = sparse(iH,jH,sH);
Hs = sum(H,2);
%% ACTIVE AND PASSIVE ELEMENTS
passive = zeros(nely,nelx);
```

```
%% INITIALIZE ITERATION
x = repmat(volfrac,nely,nelx);
loop = 0;
loopbeta = 0;
change = 1;
obj = NaN(max_it_opt,1);
changeplot = NaN(max_it_opt,1);
volume = NaN(max_it_opt,1);
%% INITIALIZE MMA OPTIMIZER
m = 1; % The number of general constraints.
n = nelx*nely;      % The number of design variables x_j.
xmin = zeros(n,1);        % Column vector with the lower bounds for the
    variables x_j.
xmax = ones(n,1); % Column vector with the upper bounds for the
    variables x_j.
xold1 = x(:);      % xval, one iteration ago (provided that iter>1).
xold2 = x(:);      % xval, two iterations ago (provided that iter>2).
low = ones(n,1); % Column vector with the lower asymptotes from the
    previous iteration (provided that iter>1).
upp = ones(n,1); % Column vector with the upper asymptotes from the
    previous iteration (provided that iter>1).
a0 = 1; % The constants a_0 in the term a_0*z.
a = zeros(m,1);        % Column vector with the constants a_i in the
    terms a_i*z.
c_MMA = 1e4*ones(m,1);  % Column vector with the constants c_i in the
    terms c_i*y_i.
d = zeros(m,1);        % Column vector with the constants d_i in the
    terms 0.5*d_i*(y_i)^2.
%% HEAVISIDE FILTER
if ft == 1 || ft == 2
    xPhys = x;
elseif ft == 3
    xTilde = x;
    xPhys = 1-exp(-beta*xTilde)+xTilde*exp(-beta);
end
%% START ITERATION
while change > 0.01 && loop <= max_it_opt
    loop = loop + 1;
    loopbeta = loopbeta + 1;
    %% FE-ANALYSIS
    sK = reshape(KE(:)*(kmin+(1-kmin)*xPhys(:)'.^penal),16*nelx*nely,1)
        ;
    K = sparse(iK,jK,sK); K = (K+K')/2;
    U(freedofs) = K(freedofs,freedofs)\F(freedofs);
    %% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
    ce = reshape(sum((U(edofMat)*KE).*U(edofMat),2),nely,nelx);
    c = sum(sum((kmin+(1-kmin)*xPhys.^penal).*ce));
    dc = -penal*(1-kmin)*xPhys.^(penal-1).*ce;
    dv = ones(nely,nelx);
    obj(loop+1) = c;
    %% FILTERING/MODIFICATION OF SENSITIVITIES
    if ft == 1
        dc(:) = H*(x(:).*dc(:))./Hs./max(1e-3,x(:));
    elseif ft == 2
        dc(:) = H*(dc(:)./Hs);
        dv(:) = H*(dv(:)./Hs);
    elseif ft == 3
        dx = beta*exp(-beta*xTilde)+exp(-beta);
        dc(:) = H*(dc(:).*dx(:)./Hs);
        dv(:) = H*(dv(:).*dx(:)./Hs);
    end
    %% METHOD OF MOVING ASYMPTOTES
    fval  = sum(xPhys(:)) / (volfrac*nelx*nely) - 1;
    dfdx  = dv(:)' / (volfrac*nelx*nely);
    dfdx2 = 0*dv(:)';
    [xmma, ~, ~, ~, ~, ~, ~, ~, ~, low,upp] = ...
        mmasub(m, n, loop, x(:), xmin, xmax, xold1, xold2, c, dc(:), 0*
            dc(:), fval, dfdx, dfdx2, low, upp, a0, a, c_MMA, d);
    % Update MMA Variables
    xnew = reshape(xmma,nely,nelx);
    change = max(abs(xnew(:)-x(:)));
    if ft == 1
        xPhys = xnew;
    elseif ft == 2
        xPhys(:) = (H*xnew(:))./Hs;
    elseif ft == 3
        xTilde(:) = (H*xnew(:))./Hs;
        xPhys = 1-exp(-beta*xTilde)+xTilde*exp(-beta);
    end
    xPhys(passive==1) = 0;
    xPhys(passive==2) = 1;
    xold2 = xold1(:);
    xold1 = x(:);
    x = xnew;
    changeplot(loop+1) = change;
    volume(loop+1) = mean(xPhys(:));
    %% PRINT RESULTS
    fprintf(' It.:%5i Obj.:%11.4f Vol.:%7.3f ch.:%7.3f\n',loop,c, ...
        mean(xPhys(:)),change);
    %% PLOT DENSITIES
    colormap(gray); imagesc(1-xPhys); caxis([0 1]); axis equal; axis
       off; drawnow;
    %% UPDATE HEAVISIDE REGULARIZATION PARAMETER
    if ft == 3 && beta < 512 && (loopbeta >= 50 || change <= 0.01)
        beta = 2*beta;
        loopbeta = 0;
        change = 1;
        fprintf('Parameter beta increased to %g.\n',beta);
    end
end
%% EXTRA PLOTS
obj(isnan(obj)) = [];
changeplot(isnan(changeplot)) = [];
volume(isnan(volume)) = [];
xaxisplot = 1:1:size(obj,1);

figure;
ax1 = subplot(3,1,1);
plot(obj,'r');
title(ax1,'Objective function')

ax2 = subplot(3,1,2);
plot(xaxisplot,volume,'r');
title(ax2,'Volume')

ax3 = subplot(3,1,3);
plot(xaxisplot,changeplot,'r');
title(ax3,'Change of volume')

xlim([ax1 ax2 ax3],[1 size(obj,1)])
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This Matlab code was written by E. Andreassen, A. Clausen, M.
    Schevenels,
% B. S. Lazarov and O. Sigmund,  Department of Solid  Mechanics,
%  Technical University of Denmark,
%  DK-2800 Lyngby, Denmark.
% Please sent your comments to: sigmund@fam.dtu.dk
%
% The code is intended for educational purposes and theoretical details
% are discussed in the paper
% "Efficient topology optimization in MATLAB using 88 lines of code,
% E. Andreassen, A. Clausen, M. Schevenels,
% B. S. Lazarov and O. Sigmund, Struct Multidisc Optim, 2010
% This version is based on earlier 99-line code
% by Ole Sigmund (2001), Structural and Multidisciplinary Optimization,
% Vol 21, pp. 120--127.
%
% The code as well as a postscript version of the paper can be
% downloaded from the web-site: http://www.topopt.dtu.dk
%
% Disclaimer:
% The authors reserves all rights but do not guaranty that the code is
% free from errors. Furthermore, we shall not be liable in any event
% caused by the use of the program.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

A.3.2 Heat Sink

```
%%%% AN 88 LINE TOPOLOGY OPTIMIZATION CODE Nov, 2010 %%%%
%%%% MODIFIED HEAT EXAMPLE Jan,2019 %%%%
nelx = 100; nely = 100; volfrac = 0.3; penal = 3; rmin = 3; ft = 2;
max_it_opt = 349;
beta = 1;
%% MATERIAL PROPERTIES
k0 = 1; % Good thermal conductivity
kmin = 1e-3; % Poor thermal conductivity
%% PREPARE FINITE ELEMENT ANALYSIS
KE = 1/12*k0*[4 -1 -2 -1; -1 4 -1 -2; -2 -1 4 -1; -1 -2 -1 4];
nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
edofVec = reshape(nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
edofMat = repmat(edofVec,1,4)+repmat([0 nely+1 nely -1],nelx*nely,1);
iK = reshape(kron(edofMat,ones(4,1))',16*nelx*nely,1);
jK = reshape(kron(edofMat,ones(1,4))',16*nelx*nely,1);
% DEFINE LOADS AND SUPPORTS
F = 0.0001*ones((nely+1)*(nelx+1),1);
U = zeros((nely+1)*(nelx+1),1);
fixeddofs = union([1:(nely+1),1:(nely+1):(nelx+1)*(nely+1),(nely+1):(
    nely+1):(nelx+1)*(nely+1)],(nely+1)*(nelx)+1:(nelx+1)*(nely+1)); %
    Dirichlet boundary example
alldofs = 1:(nely+1)*(nelx+1);
freedofs = setdiff(alldofs,fixeddofs);
%% PREPARE FILTER
iH = ones(nelx*nely*(2*(ceil(rmin)-1)+1)^2,1);
jH = ones(size(iH));
sH = zeros(size(iH));
k = 0;
for i1 = 1:nelx
    for j1 = 1:nely
        e1 = (i1-1)*nely+j1;
        for i2 = max(i1-(ceil(rmin)-1),1):min(i1+(ceil(rmin)-1),nelx)
            for j2 = max(j1-(ceil(rmin)-1),1):min(j1+(ceil(rmin)-1),
                nely)
                e2 = (i2-1)*nely+j2;
                k = k+1;
                iH(k) = e1;
                jH(k) = e2;
                sH(k) = max(0,rmin-sqrt((i1-i2)^2+(j1-j2)^2));
            end
        end
    end
```

```
end
H = sparse(iH,jH,sH);
Hs = sum(H,2);
%% ACTIVE AND PASSIVE ELEMENTS
passive = zeros(nely,nelx);
passive(1:4,:) = 1;
passive(:,1:4) = 1;
passive(:,97:100) = 1;
passive(99:100,1:40) = 1;
passive(99:100,61:100) = 1;
%% INITIALIZE ITERATION
x = repmat(volfrac,nely,nelx);
loop = 0;
loopbeta = 0;
change = 1;
obj = NaN(max_it_opt,1);
changeplot = NaN(max_it_opt,1);
volume = NaN(max_it_opt,1);
%% INITIALIZE MMA OPTIMIZER
m = 1; % The number of general constraints.
n = nelx*nely;      % The number of design variables x_j.
xmin = zeros(n,1);        % Column vector with the lower bounds for the
    variables x_j.
xmax = ones(n,1); % Column vector with the upper bounds for the
    variables x_j.
xold1 = x(:);      % xval, one iteration ago (provided that iter>1).
xold2 = x(:);      % xval, two iterations ago (provided that iter>2).
low = ones(n,1); % Column vector with the lower asymptotes from the
    previous iteration (provided that iter>1).
upp = ones(n,1); % Column vector with the upper asymptotes from the
    previous iteration (provided that iter>1).
a0 = 1; % The constants a_0 in the term a_0*z.
a = zeros(m,1);        % Column vector with the constants a_i in the
    terms a_i*z.
c_MMA = 1e4*ones(m,1);  % Column vector with the constants c_i in the
    terms c_i*y_i.
d = zeros(m,1);        % Column vector with the constants d_i in the
    terms 0.5*d_i*(y_i)^2.
%% HEAVISIDE FILTER
if ft == 1 || ft == 2
    xPhys = x;
elseif ft == 3
    xTilde = x;
    xPhys = 1-exp(-beta*xTilde)+xTilde*exp(-beta);
end
%% START ITERATION
while change > 0.01 && loop <= max_it_opt
    loop = loop + 1;
    loopbeta = loopbeta + 1;
    %% FE-ANALYSIS
    sK = reshape(KE(:)*(kmin+(1-kmin)*xPhys(:)'.^penal),16*nelx*nely,1)
        ;
    K = sparse(iK,jK,sK); K = (K+K')/2;
    U(freedofs) = K(freedofs,freedofs)\F(freedofs);
    %% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
    ce = reshape(sum((U(edofMat)*KE).*U(edofMat),2),nely,nelx);
    c = sum(sum((kmin+(1-kmin)*xPhys.^penal).*ce));
    dc = -penal*(1-kmin)*xPhys.^(penal-1).*ce;
    dv = ones(nely,nelx);
    obj(loop+1) = c;
    %% FILTERING/MODIFICATION OF SENSITIVITIES
    if ft == 1
        dc(:) = H*(x(:).*dc(:))./Hs./max(1e-3,x(:));
    elseif ft == 2
        dc(:) = H*(dc(:)./Hs);
        dv(:) = H*(dv(:)./Hs);
    elseif ft == 3
        dx = beta*exp(-beta*xTilde)+exp(-beta);
        dc(:) = H*(dc(:).*dx(:)./Hs);
        dv(:) = H*(dv(:).*dx(:)./Hs);
    end
    %% METHOD OF MOVING ASYMPTOTES
    fval  = sum(xPhys(:)) / (volfrac*nelx*nely) - 1;
    dfdx  = dv(:)' / (volfrac*nelx*nely);
    dfdx2 = 0*dv(:)';
    [xmma, ~, ~, ~, ~, ~, ~, ~, ~, low,upp] = ...
        mmasub(m, n, loop, x(:), xmin, xmax, xold1, xold2, c, dc(:), 0*
            dc(:), fval, dfdx, dfdx2, low, upp, a0, a, c_MMA, d);
    % Update MMA Variables
    xnew = reshape(xmma,nely,nelx);
    change = max(abs(xnew(:)-x(:)));
    if ft == 1
        xPhys = xnew;
    elseif ft == 2
        xPhys(:) = (H*xnew(:))./Hs;
    elseif ft == 3
        xTilde(:) = (H*xnew(:))./Hs;
        xPhys = 1-exp(-beta*xTilde)+xTilde*exp(-beta);
    end
    xPhys(passive==1) = 0;
    xPhys(passive==2) = 1;
    xold2 = xold1(:);
    xold1 = x(:);
    x = xnew;
    changeplot(loop+1) = change;
    volume(loop+1) = mean(xPhys(:));
    %% PRINT RESULTS
    fprintf(' It.:%5i Obj.:%11.4f Vol.:%7.3f ch.:%7.3f\n',loop,c, ...
        mean(xPhys(:)),change);
    %% PLOT DENSITIES
    colormap(gray); imagesc(1-xPhys); caxis([0 1]); axis equal; axis
        off; drawnow;
    %% UPDATE HEAVISIDE REGULARIZATION PARAMETER
    if ft == 3 && beta < 512 && (loopbeta >= 50 || change <= 0.01)
        beta = 2*beta;
        loopbeta = 0;
        change = 1;
        fprintf('Parameter beta increased to %g.\n',beta);
    end
end
%% EXTRA PLOTS
obj(isnan(obj)) = [];
changeplot(isnan(changeplot)) = [];
volume(isnan(volume)) = [];
xaxisplot = 1:1:size(obj,1);

figure;
ax1 = subplot(3,1,1);
plot(obj,'r');
title(ax1,'Objective function')

ax2 = subplot(3,1,2);
plot(xaxisplot,volume,'r');
title(ax2,'Volume')

ax3 = subplot(3,1,3);
plot(xaxisplot,changeplot,'r');
title(ax3,'Change of volume')

xlim([ax1 ax2 ax3],[1 size(obj,1)])
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This Matlab code was written by E. Andreassen, A. Clausen, M.
    Schevenels,
% B. S. Lazarov and O. Sigmund,  Department of Solid  Mechanics,
%  Technical University of Denmark,
%  DK-2800 Lyngby, Denmark.
% Please sent your comments to: sigmund@fam.dtu.dk
%
% The code is intended for educational purposes and theoretical details
% are discussed in the paper
% "Efficient topology optimization in MATLAB using 88 lines of code,
% E. Andreassen, A. Clausen, M. Schevenels,
% B. S. Lazarov and O. Sigmund, Struct Multidisc Optim, 2010
% This version is based on earlier 99-line code
% by Ole Sigmund (2001), Structural and Multidisciplinary Optimization,
% Vol 21, pp. 120--127.
%
% The code as well as a postscript version of the paper can be
% downloaded from the web-site: http://www.topopt.dtu.dk
%
% Disclaimer:
% The authors reserves all rights but do not guaranty that the code is
% free from errors. Furthermore, we shall not be liable in any event
% caused by the use of the program.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```