Master's Programme in Mechanical Engineering

A multiscale finite element framework for additive manufacturing process modeling

Ivan Yashchuk

Aalto University
School of Engineering

MASTER'S
THESIS

Aalto University
MASTER'S THESIS 2018

A multiscale finite element framework for additive manufacturing process modeling

Ivan Yashchuk

Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Technology.
Otaniemi, April 3, 2018

Supervisor: Prof. Antti Hannukainen

Aalto University
School of Engineering
Master's Programme in Mechanical Engineering

Author
Ivan Yashchuk

Title of thesis
A multiscale finite element framework for additive manufacturing process modeling

Master's programme Mechanical Engineering

Thesis supervisor Prof. Antti Hannukainen

Thesis advisor Anssi Laukkanen

Date April 3, 2018 **Number of pages** 5+51 **Language** English

Abstract

This thesis describes a finite element framework for solving partial differential equations with highly varying spatial coefficients. The goal is to model the heat transfer in a heterogeneous powder medium of the selective laser melting process. An operator based framework is developed and the implementation details are discussed. The main idea of the work is based on the two level domain decomposition and construction of special operators to transfer the system between the coarse and fine levels. The system of equations is solved on a coarse level and the solution is transferred to the fine level. The operators are computed using Localized Orthogonal Decomposition (LOD) method. The method is applied to several numerical experiments and an optimal convergence rates in the H^1 and L^2 norms are observed. The computational efficiency of LOD is studied and its limitations are discussed.

Keywords multiscale method, localized orthogonal decomposition, heat equation

Preface

This master's thesis has been carried out in the material modeling group at VTT Technical Research Centre of Finland.

I would like to thank and give appreciation to my supervisor Antti Hannukainen for his excellent support, invaluable guidance and patience.

I would also like to give a special thanks to my parents and my wonderful Viktoriya, who supported me during the studies with love and encouragement.

Otaniemi, April 3, 2018,

Ivan Yashchuk

Contents

1. Introduction

Additive manufacturing (AM) processes for metallic materials open up possibilities for creating functional components that are fabricated directly from a digital model. Even though the technology is over twenty years old, it started to gain popularity in commercial manufacturing quite recently. Manufacturing of parts is done layer-by-layer allowing one to create complex shapes and to control the microstructure of the final part.

During fabrication, metallic AM parts experience a repeated melting and solidification, resulting in the complex thermal history. These factors introduce complexities to the analysis of microstructural evolution. Models are needed at multiple length scales to account for the structural details and to understand the basic physical processes that are active in the performance response of these materials. Component performance simulations require development of models at multiple length scales. Multiscale model and algorithm that will incorporate knowledge of the microstructure within the macro-scale continuum representation is needed [1].

1.1 Challenges for Selective Laser Melting process simulation

Powder scale models for Selective Laser Melting (SLM) process capture the details of laser interaction with the powder. These models give temperature and time histories of the melt pool, with applications to real-time process diagnostics and microstructure development.

SLM simulation has heterogeneities on different length scales resulting in a requirement of a fine mesh in order to get accurate results. Traditional homogenization techniques, or upscaling techniques, aim at simplifying properties of heterogeneous materials, replacing by the 'homogenized' one, with fictive material properties, homogenized material should be a good approximation of the original heterogeneous material. Having homogenized material description the problem can be relatively accurately solved on a coarse grid. In the realistic case, many problems have high contrast and non-periodicity in material coefficients and upscaling methods do not provide accurate and robust solutions to these problems.

1.2 Multiscale methods

Multiscale methods have been developed in attempt to overcome the drawbacks of traditional homogenization techniques. These methods rely on a coarse representation of the underlying finescale domain to create a reduced problem. They are capable of producing an approximation of the solution on the fine scale defined with the exact heterogenous properties opposed to the upscaling, which provides solutions only for a coarse scale.

The idea behind the multiscale methods is to use basis functions that restrict the equations from the fine grid to a coarser grid having fewer unknowns. At the coarse grid the equations can be solved more efficiently. New basis functions are used to prolong the coarse scale solution to the original fine scale. The basis functions are constructed so that fine scale heterogeneities are preserved.

Unfortunately, the total computational cost of getting the solution by a multiscale method is about the same as for solving the same problem discretized on the fine scale. However, multiscale methods can potentially give savings in the computational cost. In this thesis, the heat equation is used to model the temperature distribution and evolution in SLM process. The heat equation is time-dependent and when discretized in time needs to be solved repeatedly. Since the temporal changes in varying coefficient are usually moderate compared to the spatial variability, it is seldom necessary to perform the local problem computations for every iteration in time. The re-computation will be needed only in regions undergoing the phase change. The more solutions to the local problem are reused the more efficient the method should be. It needs to mention that the local problems are usually formulated in such a way that they are independent and can be easily parallelized.

There is a variety of different methods to treat multiscale problems numerically. One example for such a method is the Heterogenous Multiscale Method (HMM), which was introduced in [2]. The fine-scale behavior is reconstructed only in a few cells around quadrature points and to transfer the gained information to a macroscopic equation, which uses a local average of this information. Another example is the Multiscale Finite Element Method (MsFEM) developed by Hou and Wu [3]. A set of multiscale basis functions is constructed by adding fine-scale features to the original basis functions. Then, the original problem is posed and solved in the low-dimensional space that is spanned by the multiscale basis. Main focus in this work is Localized Orthogonal Decomposition (LOD) [4], which is the modification of the Variational Multiscale Method (VMM). VMM was introduced in [5] and is based on a splitting of the original solution space into a direct sum of a coarse space and a fine space. This results in a coupled coarsescale and finescale equations.

1.3 Outline

The present contribution focuses on the development of a multiscale solver for problems with highly varying spatial coefficients, with application to modeling the SLM process. Some parts of the physical model are simplified and do not represent all physical phenomena of the process, e.g. the mechanical response is not simulated, heat convection and evaporation as well as shrinkage effects are neglected. Therefore, also no attempt is made to compare the result in detail with experimental data. However, the implementation is validated against known analytical exact solutions. The computed solutions of the multiscale solver are compared to the standard formulation of finescale problems.

The thesis is organized as follows: in the next chapter the thermal model is introduced, with description of the laser and powder models. In chapter 3 the temporal and spatial discretizations of the problem are specified. Then the solution method for multiscale problems, introduced in chapter 3, is presented in chapter 4. Chapter 5 is devoted to discussing implementation of the solver. Chapter 6 discusses numerical examples and compares the efficiency of the standard and multiscale solvers.

2. Selective Laser Melting process model

2.1 Physical phenomena

In its initial state, the selective melting process consists of a powder bed with a certain porosity, deposited on top of consolidated layers acting as a substrate. When a laser passes over the powder, the laser beam melts the powder particles forming a melt pool. To achieve full melting, enough heat should by supplied by the laser. The melt pool also penetrates to the substrate below, in order to have the melted powder and the previous layers melted together. When the laser has finished passing over a layer, a new layer of powder is deposited, and the process is repeated layer-by-layer.

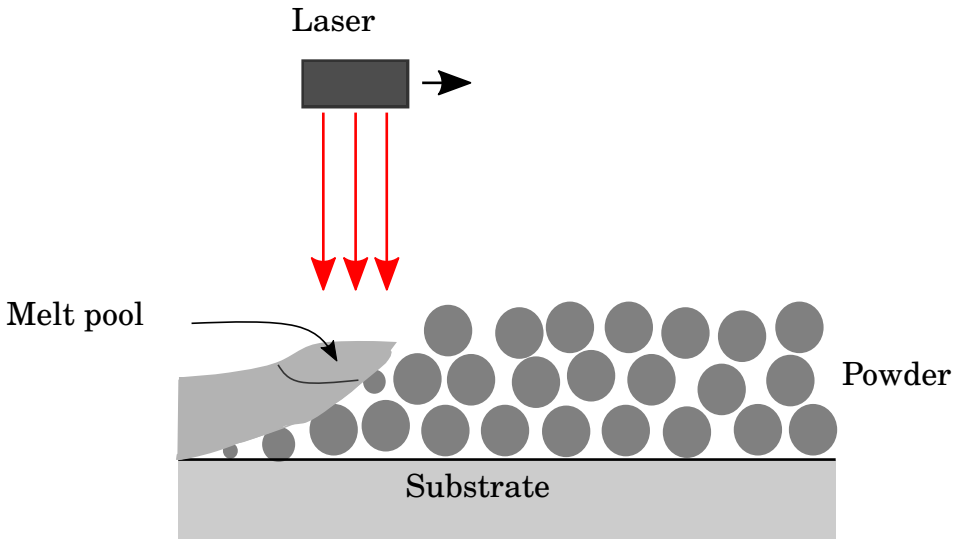


Figure 2.1. Schematic representation of SLM process

Figure 2.1 presents an illustration of a selective laser melting process for a single layer of powder. The relevant physical phenomena that need to be considered in the model of the SLM process are described in [6]. Main phenomena that are included in the present thermal model are:

- Laser-powder interaction

  The absorption coefficient of a material determines the amount of energy supplied by the laser that is absorbed and the amount re-

flected. Several laser irradiation models exist, one common approach is modelling the heat input as a surface Gaussian heat flux. The laser absorption within the powder bed is governed by multiple reflections of incident laser rays within the open-pore system of the powder bed, each with partial absorption of the incident radiation. Therefore, the net absorptivity of powder beds is considerably higher than the value known for flat surfaces and, moreover, the laser beam energy source should be modelled as a volumetric heat source distributed over the powder bed thickness, as opposed to a surface heat source.

- Phase change

  During the SLM process the deposited powder melts due to the passing laser and solidifies afterwards. The equations that govern the energy balance of phase transformation are commonly referred to as the Stefan-Neumann problem. Evaporation and condensation can also be present in the process, however, these effects are not added to the model. Shrinkage due to melting is also not considered in the present study.

- Conduction

  Heat conduction to consolidated material is modelled using Fourier's law $q = -\kappa(T)\nabla T$, where q is the heat flux, $\kappa(T)$ is the temperature-dependent conductivity and ∇T is the temperature gradient. The amount of heat transferred via conduction to the powder depends on the porosity of the powder and the conductivity of the gas filling the pores. The thermal conductivity of powder is much smaller than the conductivity in the solidified phase [7]. One common approach is to rely on a model for an effective, homogenized powder bed conductivity. However, in the characterization of SLM processes, one is often interested in length scales comparable to the powder layer thickness or laser beam spot size. Because powder bed mesoscopic heterogeneities in form of individual particles are on the same length scale, the application of continuum models can often only be considered as a rough estimate of the underlying heat transfer processes. Typically, powder particle sizes in the range of 10µm-50µm, layer thicknesses in the range of 20µm-100µm and laser beam spot sizes in the range of 20µm-200µm are employed. In the present study, powder medium is modelled to consist of individual powder particles with a gas between them. This is the main difference from most other models presented in the literature which use homogenized continuum for the powder region.

2.2 Governing equations

There are several ways of modeling a phase change, which form two main groups: fixed grid methods and variable grid (front tracking) methods. The variable grid methods treat the moving front explicitly and aim at aligning the grid nodes with the position of the interface. An overview of different methods is presented in [8]. The fixed grid method is used in this work because of its simplicity of formulation. The energy balance at the moving interface is implicitly accounted in the fixed grid methods and the interface position can be obtained a posteriori from the temperature field.

In a numerical context, one of the major problems in modeling the solidification phase change is dealing with latent heat evolution [9].

Let $\Omega \subset \mathbb{R}^d$ be a non-empty bounded domain with boundary $\partial\Omega$, where d is the number of space dimensions. Let us consider the governing nonlinear transient heat conduction equation with phase changes in the form given in [10]

$$\frac{\partial H}{\partial t} - \nabla \cdot (\kappa \nabla T) = r \quad \text{in } \Omega, \tag{2.1}$$

written in terms of volumetric enthalpy $H = H(T)$ and temperature $T = T(\mathbf{x}), \mathbf{x} \in \Omega$, where t is the time, κ is the thermal conductivity and r is the laser heat source.

The governing heat equation (2.1) is subject to the boundary conditions

$$\begin{aligned} T &= T_w \quad \text{on } \Gamma_D, \\ (\kappa \nabla T) \cdot \mathbf{n} &= q \quad \text{on } \Gamma_N, \end{aligned} \tag{2.2}$$

where Γ_D and Γ_N are non-overlapping portions of $\partial\Omega$, T_w is the prescribed temperature on Γ_D and q is the prescribed heat flux Γ_N. The initial condition is set as

$$T(\mathbf{x}, t = 0) = T_0(\mathbf{x}) \quad \text{in } \Omega. \tag{2.3}$$

The equation (2.1) must hold throughout the system irrespective of phase. When the domain splits in multiple phases additional boundary conditions (the so-called Stefan conditions) should be satisfied at the interface(s) Γ. In the case of isothermal phase-change, the domain separates into a solid (Ω_s) and liquid (Ω_l) regions such that $\Omega = \Omega_s \cup \Omega_l$, and the following conditions must be satisfied at the phase front Γ:

$$\begin{aligned} T_s = T_l &= T_m \quad \text{on } \Gamma, \\ (\kappa_s \nabla T_s) \cdot \mathbf{n}_s + (\kappa_l \nabla T_l) \cdot \mathbf{n}_l - \rho L \mathbf{s} \cdot \mathbf{n}_l &= 0 \quad \text{on } \Gamma, \end{aligned} \tag{2.4}$$

where κ, T, and $\mathbf{n}$ are the thermal conductivity, temperature, and the unit vector normal to Γ with the subscripts s and l refer to solid and liquid phases, respectively, T_m is the melting temperature, ρL is the latent heat and $\mathbf{s}$ is the interface velocity.

In the case of non-isothermal phase-change, the Stefan conditions (2.4)

must be satisfied at each interface between different phases to enforce local energy balance. However, the formulation of the problem (2.1) implicitly includes the heat balance conditions (2.4) as shown in [11]. In the numerical implementation of isothermal phase-change, an artificial mushy region is introduced to the system to take care of the discontinuity between phases. Therefore, in the numerical context, we always treat the problem at hand as non-isothermal phase-change. This is a reasonable assumption, if the phase change range is kept small.

Sensible heat is the thermal energy exchanged by a thermodynamic system, which directly affects the temperature of the system. Latent heat is the released or absorbed thermal energy associated with the phase change. The volumetric enthalpy function can be expressed as a sum of sensible and latent heat:

$$H(T) = H_{\text{sensible}} + H_{\text{latent}} = \int_{T_{ref}}^{T} \rho c(T) dT + \rho L g(T), \tag{2.5}$$

where T_{ref} is an arbitrary reference temperature, $\rho c(T)$ is the material heat capacity, L is the latent heat and $g(T)$ is the local liquid volume fraction. The liquid phase fraction for the isothermal phase-change case is defined by a Heavyside step function:

$$\tilde{g}(T) = \begin{cases} 0 & T \leq T_m \\ 1 & T > T_m \end{cases}. \tag{2.6}$$

The step function $\tilde{g}(T)$ is regularized by a continuous and differentiable hyperbolic-tangent function with two parameters, defined for all T:

$$g(T; T_c, R_s) = \frac{1}{2}\left[1 + \tanh\left(\frac{T_c - T}{R_s}\right)\right], \tag{2.7}$$

where T_c is the central value (around which regularization is happening), and R_s is the smoothing radius. This regularized phase fraction function is also used to regularize discontinuous functions representing the variation of material functions (conductivity, heat capacity, latent heat) across the phase interfaces as

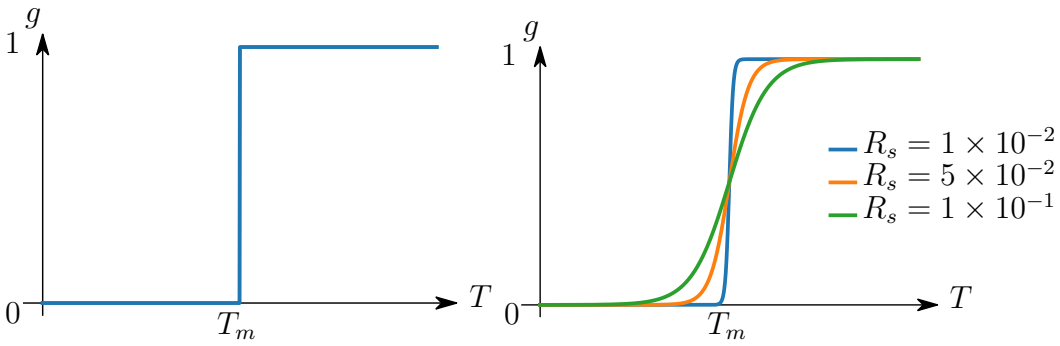
$$f(T; g) = f_s(T) + g(f_l(T) - f_s(T)), \tag{2.8}$$

where f_l, f_s are the imposed values in the liquid and solid phases.

With enthalpy definition (2.5) rewriting of equation (2.1) leads to the temperature based phase-change model:

$$\rho c \frac{\partial T}{\partial t} - \nabla \cdot (\kappa \nabla T) = r - \rho L \frac{\partial g}{\partial t} \quad \text{in } \Omega. \tag{2.9}$$

When the latent heat term is removed from equation (2.9), classical transient heat conduction equation without phase-change is recovered. Latent



(a) Heavyside step phase function **(b)** Regularized phase function

Figure 2.2. Liquid phase fraction function g

heat effects are the primary sources of non-linearities.

2.3 Laser model

Heating due to the laser is represented by the r term in Equation (2.9).

There exist several possibilities for the heating model. The one that is used in this work was created specifically for the case of laser-powder interaction. The laser model described in [12] was derived using the radiation transfer equation. The following description of the model is adapted from [13].

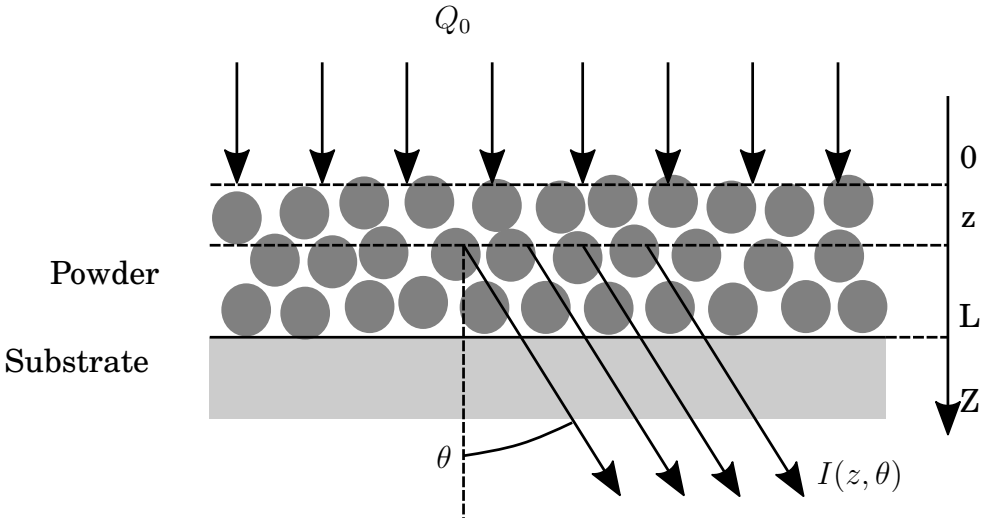


Figure 2.3. Laser radiation transfer in a powder layer on a substrate. Adapted from [12].

Gusarov et al. assume that the powder can be considered as absorbing scattering medium. Therefore, the radiation transfer equation (RTE) can be used to describe the laser energy transfer. Radiative transfer is the physical phenomenon of energy transfer in the form of electromagnetic radiation. The RTE says that as a beam of laser travels, it loses energy to absorption, gains energy by emission, and redistributes energy by scattering [14].

A powder with layer thickness L_p is subject to heating with a laser in the normal direction to the powder surface. Nominal power density of a laser is denoted as Q_0. The radiation in the powder bed is gathered with

the intensity $I(z', \theta)$. Prime notation of the coordinates indicates that it corresponds to the powder surface coordinates before melting. The RTE takes the form

$$\mu \frac{\partial I(z', \mu)}{\partial z'} = -\beta_h I(z', \mu) + \beta_h \frac{\omega}{2} \int_{-1}^{1} I(z', \tilde{\mu}) P(\tilde{\mu}, \mu)\, d\tilde{\mu}, \tag{2.10}$$

where $\beta_h I(z', \mu)$ is the extinction of the radiation, and the second term is a scattering effect in the z' direction. $\mu = \cos\theta$, β_h is the extinction coefficient, ω is the albedo (a measure for reflectance or optical brightness of a surface), and $P(\tilde{\mu}, \mu)$ is the scattering phase function.

After applying boundary conditions and and integration of Equation (2.10) gives the definition for the normalized power density q as

$$\begin{aligned} q = \frac{Q}{Q_0} = \frac{\rho_h a}{(4\rho_h - 3)D} \Big(& (1-\rho_h^2) e^{-\lambda} ((1-a) e^{-2a\xi'} + (1+a) e^{2a\xi'}) \\ & -(3 + \rho_h e^{-2\lambda})((1 + a - \rho_h(1-a)) e^{2a(\lambda - \xi')} + (1 - a - \rho_h(1+a)) e^{2a(\xi' - \lambda)}) \Big) \\ & - \frac{3(1-\rho_h)(e^{-\xi} - \rho_h e^{\xi' - 2\lambda})}{4\rho_h - 3}, \end{aligned} \tag{2.11}$$

where $a = \sqrt{1 - \rho_h}$, the optical thickness $\lambda = \beta_h L_p$, $\xi' = \beta_h z'$ is a dimensionless coordinate in the z-direction of the powder, and D is

$$D = (1-a)(1 - a - \rho_h(1+a)) e^{-2a\lambda} - (1+a)(1 + a - \rho_h(1-a)) e^{2a\lambda}.$$

In the end, the laser heating r is defined as

$$r(x', y', z') = -\beta_h Q_0 \frac{\partial q}{\partial \xi'}, \tag{2.12}$$

where x', y' are the coordinates in the tangential direction of the powder surface, Q_0 is defined as

$$Q_0 = Q_m \left(1 - \frac{r_h}{R}\right)^2 \left(1 + \frac{r_h}{R}\right)^2 \quad \text{for } 0 < r_h < R, \tag{2.13}$$

where $r_h^2 = x'^2 + y'^2$ is the radial distance of a given point from the axis projection point, Q_m come from the definition of the effective total laser power W_e as

$$W_e = 2\pi \int_0^R Q_0(r) r\, dr = \frac{\pi}{3} R^2 Q_m. \tag{2.14}$$

The parameters needed for the Gusarov laser model are as follows: the effective total power of the laser W_e, the laser beam radius R, the extinction coefficient β_h, and the hemispherical reflectivity of the powder ρ_h. Full derivation of the model is presented in [12].

The depth profiles of the normalized power density q and volumetric heat source r in the powder layer versus dimensionless depth ξ for various values of optical depth λ at the reflectivity of dense material $\rho_h = 0.7$ are

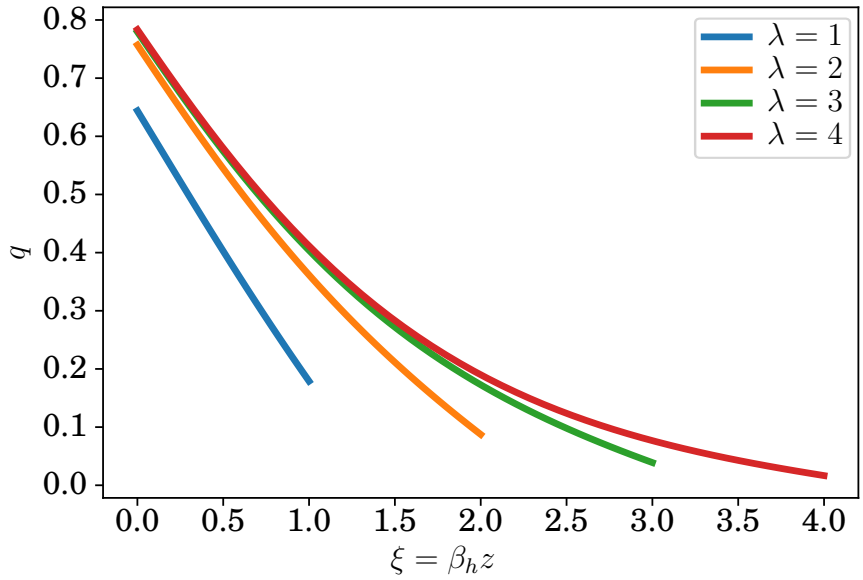


Figure 2.4. Realization of dimensionless radiative flux $q = Q/Q_0$ in the powder layer

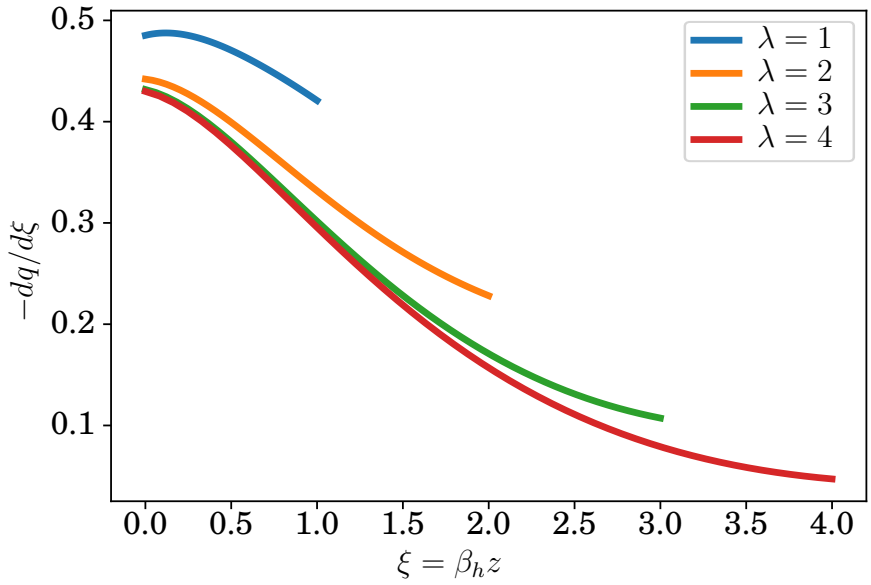


Figure 2.5. Realization of volumetric heat source $r/(\beta_h Q_0)$ in the powder layer

presented in Figures 2.4, 2.5.

2.4 Powder model

Powder bed consists of metal particles and gas between them. Thermal conductivity of gas is almost zero, thus thermal conductivity field for the powder bed varies rapidly and non-periodically. To perform simulation of the SLM process we need a model for the powder bed which mimics the real properties, like powder size distribution, packing density. For simplicity, it is assumed that the powder bed consists of spherical particles

not necessary of the same size. For particle size characterization log-normal distribution is commonly used [15]. To generate the powder bed a Monte-Carlo simulation model described in [16] is adopted. It is an iterative algorithm to simulate random packings of log-normal distributed particles. The algorithm consists of the following steps:

1. Generate n particles with the desired size distribution. The probability density function of particle r is:

$$z(r) = \frac{1}{\sqrt{2\pi}\sigma r} e^{-\frac{(\ln r - \mu)^2}{2\sigma^2}}, \tag{2.15}$$

   where μ is the mean size and σ is the standard deviation.

2. Calculate the size of the bounding box:

$$L_0 = \left(\frac{1}{\phi_0} \sum_{i=1}^{n} \frac{4}{3} \pi r_i^3 \right)^{1/3}, \tag{2.16}$$

   where ϕ_0 is the target packing density of the powder bed, r_i is the radius of the ith particle and L_0 is the side length of the bed.

3. Generated particles are uniformly distributed within the bounding box.

4. Overlap o_{ij} is calculated for every particle pair using the formula:

$$o_{ij} = \frac{r_i + r_j - d_{ij}}{r_i + r_j}, \tag{2.17}$$

   where d_{ij} is the distance between ith and jth particles.

5. For each particle that overlaps other the new position is calculated as

$$\vec{R}_i^{new} = \frac{1}{n_i} \sum_{j=1}^{n_i} \left(\vec{R}_j + (\vec{R}_i - \vec{R}_j) \frac{r_i + r_j}{d_{ij}} \right), \tag{2.18}$$

   where $\vec{R}_i^{new}$ is the new location for the ith particle, n_i is the number of particles overlapping particle i. If a particle neither overlaps nor contacts others, it is moved to contact its nearest neighbor.

6. Repeat last two step until overlap rate drops below a specified threshold or maximum iteration number is reached.

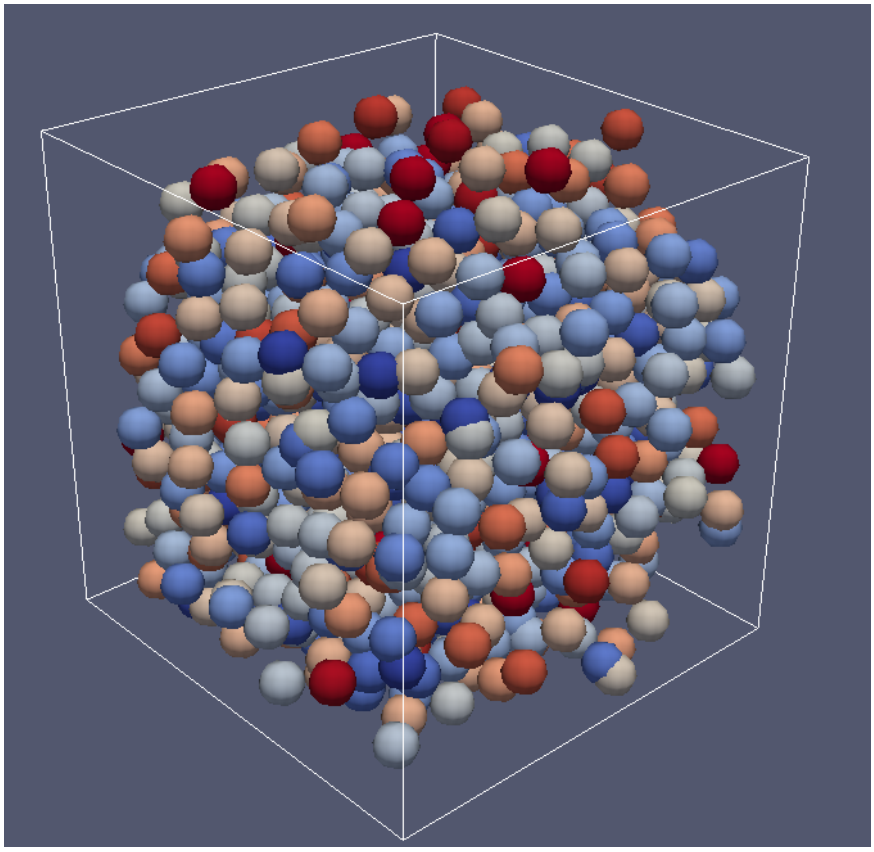


Figure 2.6. Simulated powder packing.

3. Time and space discretization

3.1 The finite element method

Abstract variational formulation

Let V be some Hilbert space, $a : V \times V \to \mathbb{R}$ be a continuous and V-elliptic (coercive) bilinear form and $L : V \to \mathbb{R}$ be a continuous linear form. The abstract variational problem written in the canonical form is: Find $u \in V$ such that:

$$a(u, v) = L(v), \quad \forall v \in V. \tag{3.1}$$

The problem (3.1) has a unique solution according to Lax-Milgram Theorem [17].

The finite element method (FEM) is developed to approximate the solution of PDEs in the variational formulation. The main idea of the FEM is to replace the Hilbert space V in which the variational formulation is posed by a finite-dimensional subspace V_h. Then the discrete variational problem can be written as a linear system, which can be solved using a computer.

The problem (3.1) is replaced by the following discrete problem: Find $u_h \in V_h \subset V$ such that:

$$a(u_h, v_h) = L(v_h), \quad \forall v_h \in V_h. \tag{3.2}$$

To solve problem (3.2) a basis $\{\phi_i\}_{i=1}^N$ of V_h with $N = \dim(V_h)$ is defined. The discrete solution u_h can be represented in the basis of V_h as $u_h = \sum_{i=1}^N u_i \phi_i$. This leads to rewriting the problem (3.2) as following:

$$\sum_{j=1}^{N} u_j a(\phi_j, \phi_i) = L(\phi_i), \quad \text{for } 1 \leq i \leq N, \tag{3.3}$$

which is a system of N linear equations with N unknowns. In matrix form, the linear system (3.3) can be represented as:

$$\mathbf{A}\mathbf{u} = \mathbf{b}, \tag{3.4}$$

where $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{N,N}$ is the matrix with elements $a_{ij} = a(\phi_j, \phi_i)$, for all

$1 \leq i, j \leq N$, $\mathbf{u} = (u_i)_{1 \leq i \leq N}$ is the solution vector and $\mathbf{b} = (L(\phi_i))_{1 \leq i \leq N}$ is the load vector.

Due to the V-ellipticity of the bilinear form $a(\cdot, \cdot)$, the matrix $\mathbf{A}$ is positive definite and system (3.4) has a unique solution.

Construction of the finite element function space

The finite-dimensional space V_h is usually called the finite element space. Let $\mathcal{T}$ be a family of shape regular, conforming partitions of the domain Ω such that

$$\bigcup_{K \in \mathcal{T}} K = \Omega.$$

A family of partitions $\mathcal{T}$ is called shape regular provided that there exists a number $\tau > 0$ such that every K in $\mathcal{T}$ contains a circle of radius ρ_K with $\rho_K \geq \frac{h_K}{\tau}$, where h_K is the diameter of K [18]. In a conforming partition, edges and faces of neighboring cells match exactly. A partition K is called cell and may typically have a shape of intervals, triangles, quadrilaterals, tetrahedrons or hexahedrons. These cells form a mesh of the domain Ω. The maximum diameter of an element in $\mathcal{T}$ is denoted with h and called the mesh size.

Ciarlet defines the finite element by a triple $(K, \mathcal{V}, \mathcal{L})$ [19]:

- the cell K is a bounded, closed subset of $\mathbb{R}^d$ (for $d = 1, 2, 3, \dots$) with nonempty interior and piecewise smooth boundary;
- the space $\mathcal{V} = \mathcal{V}(K)$ is a finite-dimensional function space on K of dimension n;
- the set of degrees of freedom (nodes) $\mathcal{L} = \{\ell_1, \ell_2, \dots, \ell_n\}$ is a basis for the dual space $\mathcal{V}'$, that is, the space of bounded linear functionals on $\mathcal{V}$.

Typically, a nodal basis $\{\phi_i^K\}_{i=1}^{n_K}$ satisfying $\ell_i(\phi_j^K) = \delta_{ij}$ for $1 \leq i, j \leq n$ is used for $\mathcal{V}$ [20].

In this thesis the most common finite element is used, which is the $\mathcal{P}_1$ Lagrange element. The corresponding finite element space V_h is defined as

$$V_h(\mathcal{T}) := \{v \in C^0(\Omega) \,|\, v_{|K} \text{ is a polynomial of degree 1, for every } K \in \mathcal{T}\}.$$

Due to the local support of the nodal basis functions, the resulting matrix $\mathbf{A}$ has a sparse structure, i.e. it has many zero entries.

In this thesis, all of the FEM associated routines: construction of finite element spaces, assembly of matrices and vectors, solving linear systems, etc. are done using open-source library called FEniCS Project [20].

Finite element error estimation

A computed finite element solution u_h is an approximation to the exact solution u and the discretization error is $e = u - u_h$. In order to measure efficiency of the given method we need error estimates which tell us how fast the error decreases as we decrease the mesh size. *A priori error*

estimates show the convergence rate of a finite element method. The finite element error is expressed as $\|u - u_h\|$ in some norm $\|\cdot\|$.

Consider the linear variational problem (3.2). The bilinear form a and the linear form L are continuous, this is, there exists a constant $C > 0$ such that

$$a(v, w) \leq C \|v\|_V \|w\|_V, \quad \forall v, w \in V, \tag{3.5}$$

$$L(v) \leq C \|v\|_V, \quad \forall v \in V. \tag{3.6}$$

Furthermore, since the bilinear form a is V-elliptic, there exist a constant $\alpha > 0$ such that

$$a(v, v) \geq \alpha \|v\|_V^2 \quad \forall v \in V. \tag{3.7}$$

Since all $v_h in V_h$ are also in V, we can choose $v = v_h$ in problem (3.1). After subtracting equation (3.2) from equation (3.1) we get

$$a(u - u_h, v_h) = a(u, v_h) - a(u_h, v_h) = L(v_h) - L(v_h) = 0, \quad \forall v_h \in V_h \subset V. \tag{3.8}$$

Equation (3.8) is known as Galerkin orthogonality and tells that the error e is orthogonal to the space V_h. Using coercivity and continuity of the bilinear form a we have

$$\begin{aligned} \alpha \|u - u_h\|_V^2 \leq a(u - u_h, u - u_h) = \\ a(u - u_h, u - v_h) + a(u - u_h, v_h - u_h) = \\ a(u - u_h, u - v_h) \leq C \|u - u_h\|_V \|u - v_h\|_V, \quad \forall v_h \in V_h. \end{aligned} \tag{3.9}$$

Then the following error estimate, also called Cea's lemma, is obtained

$$\|u - u_h\|_V \leq \frac{C}{\alpha} \|u - v_h\|_V, \quad \forall v_h \in V_h. \tag{3.10}$$

Note that the bilinear form a is also an inner product, because it is symmetric. We may use the energy norm defined as $\|\cdot\|_E = a(\cdot, \cdot)^{1/2}$, then u_h is the a-projection onto V_h and Cea's lemma states that

$$\|u - u_h\|_E = \inf_{v_h \in V_h} \|u - v_h\|_E, \quad \forall v_h \in V_h, \tag{3.11}$$

in other words u_h minimizes the error in the energy norm over all function in V_h, alternatively u_h is the best possible approximation of u in the subspace V_h. Figure 3.1 illustrates a-projection of $u \in V$ onto the subspace V_h as well as the error $u - u_h$.

Let $\Pi_h u$ be the piecewise linear interpolant of u, then Cea's lemma together with the choice $v = \Pi_h u$ yields an a priori error estimate for u_h: there exist an interpolation constant C_i independent of h such that:

$$\|u - u_h\|_E \leq \|u - \Pi_h u\|_E \leq C_i h \left\|D^2 u\right\|_{L^2}. \tag{3.12}$$

That means that in the case when $\mathcal{P}_1$ Lagrange elements are used and the

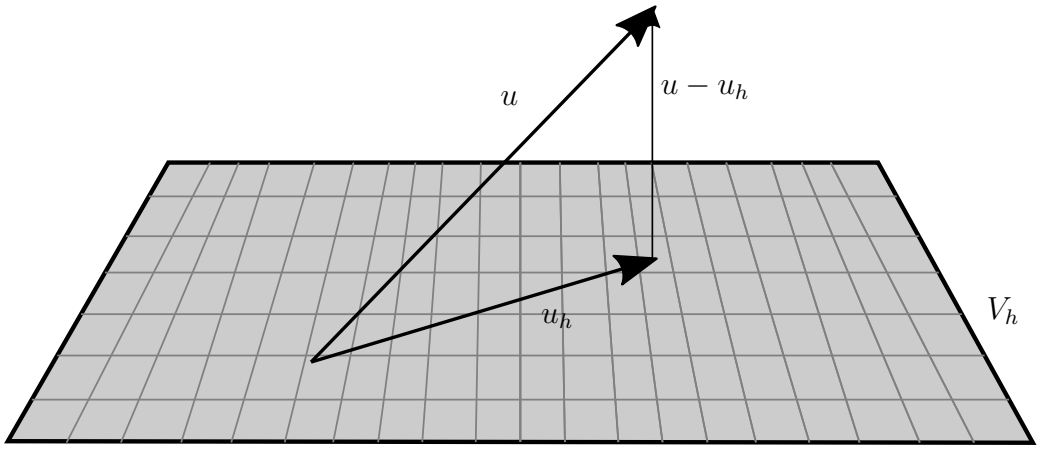


Figure 3.1. a-projection of $u \in V$ onto the subspace V_h.

solution u is smooth the finite element method has first order convergence rate in the energy norm.

3.2 Discretization of the heat equation

A common way of solving time-dependent PDEs, such as problem (2.9), by the finite element method is to first approximate the time derivative by a finite difference. It produces a sequence of stationary problems, which are transformed into a weak formulation.

Here and after the solution to PDE is denoted as u, and f governs all source terms of Equation (2.9). Let Δt denote a constant time-step, then a quantity ϕ^n is an approximation of $\phi(t)$ at time $t = t_n = n\Delta t$, where n is an integer counting time intervals. A finite difference discretization in time begins with taking a sample of PDE at some time interval, t_n:

$$\left(\frac{\partial u}{\partial t}\right)^n = \nabla \cdot (\kappa \nabla u^n) + f^n. \tag{3.13}$$

A backward Euler discretization of the time derivative is chosen because this discretization is unconditionally stable:

$$\left(\frac{\partial u}{\partial t}\right)^n \approx \frac{u^n - u^{n-1}}{\Delta t}. \tag{3.14}$$

Substituting expression (3.14) into Equation (3.13) gives

$$\frac{u^n - u^{n-1}}{\Delta t} = \nabla \cdot (\kappa \nabla u^n) + f^n. \tag{3.15}$$

Equation (3.15) is the time-discrete version of the heat equation. The result is a sequence of steady-state problems for u^n, assuming u^{n-1} is known from the previous time step:

$$u^n - \Delta t \nabla \cdot (\kappa \nabla u^n) = u^{n-1} + \Delta t f^n, \quad n = 0, 1, 2, \ldots \tag{3.16}$$

A finite element method is used to solve Equation (3.16), it requires transforming the equation into weak form. The equation is multiplied by an arbitrary test function v and Green's formula is applied. The arising weak form can be written as:

$$a(u, v) = L(v), \quad \forall v \in V, \tag{3.17}$$

where

$$a(u^n, v) := \int_\Omega (u^n v + \Delta t \kappa \nabla u^n \cdot \nabla v)\, dx, \tag{3.18}$$

$$L(v) := \int_\Omega \left(u^{n-1} v + \Delta t f^n v\right)\, dx. \tag{3.19}$$

The heat transfer problem (2.9) includes nonlinear source term and material parameters c and κ may be defined to be nonlinear as well. Therefore, Newton's method is applied to linearize the problem.

Let $u^{n,k}$ be an approximation to the unknown u^n. We seek a correction δu such that

$$u^{n,k+1} = u^{n,k} + \delta u \tag{3.20}$$

fulfills the nonlinear problem. The idea of Newton's method is to insert $u^{n,k+1}$ in the PDE, apply Taylor expansion to the nonlinearities and keep only terms that are linear in δu and then iteratively update $u^{n,k+1}$ with solutions δu until convergence.

In order to apply the Newton's method to the weak formulation, the Equation 3.17 is regarded as $F(u; v) = 0$, where a nonlinear form is defined as

$$F(u; v) := L(v) - a(u, v). \tag{3.21}$$

Then, starting from an initial guess $u^{n,0} = u^{n-1}$, the sequence $u^{n,k}$ is constructed by solving for each inner iteration k:

$$DF(u^{n,k}; \delta u, v) = -F(u^{n,k}; v), \quad \forall v \in V, \tag{3.22}$$

where $DF(u^{n,k}; \delta u, v)$ is the Gateaux derivative of $F(u^{n,k}; v)$ at $u^{n,k}$ in the direction of δu and defined as

$$DF(u^{n,k}; \delta u, v) := \lim_{\tau \to 0} \frac{d}{d\tau} F(u^{n,k} + \tau \delta u; v). \tag{3.23}$$

In the numerical context this Gateaux derivative is called Jacobian. FEniCS can differentiate F automatically to obtain the Jacobian.

3.3 Intoduction to multiscale PDEs

At each time step we solve the Equation (3.16) and we obtain a sequence of finite element solutions to the heat equation. This sequence of problems is similar to the sequence of steady-state heat equations. Therefore, in

order to simplify presentation of the multiscale methods we further discuss steady-state heat equation, which is Poisson equation.

To demonstrate the critical effects that motivate this work we consider the following *model problem*: Find $u \in C^2(\Omega) \cap C(\bar{\Omega})$ such that

$$\begin{cases} -\nabla \cdot \kappa \nabla u = f & \text{in } \Omega \\ u = 0 & \text{on } \partial\Omega \end{cases} \tag{3.24}$$

where $\Omega \subset \mathbb{R}^d \, (d = 1, 2, 3)$ is a non-empty bounded domain, $\kappa \in L^\infty(\Omega)$ is the variable coefficient, $\kappa(x) \geq \alpha$ a.e. in Ω for some $\alpha > 0, \alpha \in \mathbb{R}$, and $f \in L^2(\Omega)$ is a given function.

For the finite element method, the problem (3.24) is reformulated as a variational problem: Find $u \in V := H_0^1(\Omega)$ such that

$$a(u, v) = L(v) \quad \forall v \in V \tag{3.25}$$

where $a : H_0^1(\Omega) \times H_0^1(\Omega) \to \mathbb{R}$ is the bilinear form given by

$$a(u, v) := \int_\Omega (\kappa \nabla u) \cdot \nabla v \quad \forall u, v \in V$$

and $L : H_0^1(\Omega) \to \mathbb{R}$ is the linear functional defined by

$$L(v) := \int_\Omega f v \quad \forall v \in V.$$

In FEM solution to Problem (3.25) is approximated by: Find $u_h \in V_h$ such that

$$a(u_h, v_h) = L(v_h) \quad \forall v_h \in V_h \tag{3.26}$$

Here V_h is a finite-dimensional subspace of V which is composed of $\mathcal{P}_1$ Lagrange elements.

Recall *a priori* error estimate (3.12). Then, assuming Ω is a convex polygon, the exact solution $u \in H^2(\Omega)$ and the standard FEM approximation u_h satisfy [21]

$$\|u - u_h\|_E \leq C_i h \, \|\Delta u\|_{L^2(\Omega)} \, . \tag{3.27}$$

It might occur that the solution of the Poisson problem is not smooth and has rapid variations, which are induced by high variation in the coefficient κ. Such oscillations in the solution with at least a frequency of ϵ^{-1} cause $\|\Delta u\|_{L^2(\Omega)}$ to produce a factor in the estimate which is approximately of the same size as frequency. For the given model problem by a few manipulations and using Cauchy–Schwarz inequality an estimate for $\|\Delta u\|_{L^2(\Omega)}$ is obtained

$$\|\Delta u\|_{L^2(\Omega)} \leq \left\| \frac{f}{\kappa} \right\|_{L^2(\Omega)} + \left\| \frac{\nabla \kappa}{\kappa} \right\|_{L^\infty(\Omega)} \|\nabla u\|_{L^2(\Omega)}$$

where the term $\left\| \frac{\nabla \kappa}{\kappa} \right\|_{L^\infty(\Omega)}$ is of order $O(\epsilon^{-1})$. Therefore, first order finite

element approximation does not converge to u until $h \ll \epsilon$. When $h \ll \epsilon$ the first order finite element method obtains quadratic convergence in L^2-norm and linear convergence in the energy norm. If ϵ is small the condition $h \ll \epsilon$ requires one to use a very fine mesh. The problems that exhibit similar behavior are called multiscale problems.

Example

We demonstrate a failure of the standard finite element method by considering Problem (3.24) with parameters: $f = 1$ and $\kappa = \left(2 + \cos\left(\frac{2\pi x}{\epsilon}\right)\right)^{-1}$.

The exact solution to the problem with given parameters was introduced in [22]

$$u_\epsilon(x) = x - x^2 + \epsilon\left(\frac{1}{4\pi}\sin\left(2\pi\frac{x}{\epsilon}\right) - \frac{1}{2\pi}x\sin\left(2\pi\frac{x}{\epsilon}\right) - \frac{\epsilon}{4\pi^2}\cos\left(2\pi\frac{x}{\epsilon}\right) + \frac{\epsilon}{4\pi^2}\right).$$

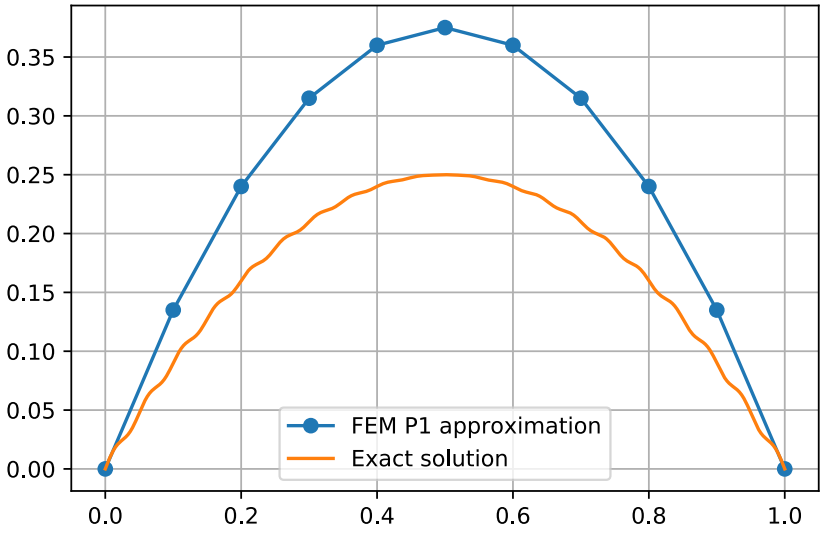


Figure 3.2. P1 FEM does not approximate the solution correctly in multiscale problems.

Figure 3.2 shows an oscillating exact solution of the example problem and the FEM approximation. Computed solution is quite different from the exact one. The computation is done for different values of the parameter ϵ and the convergence behavior is represented in Figures 3.3 and 3.4. Results clearly show that the FEM solution does not converge to the exact solution when size of the mesh elements is bigger than the size of the oscillations. In the next chapter we present the method to overcome this issue allowing solution of multiscale problem using coarse mesh that does not satisfy the requirement $h \ll \epsilon$.

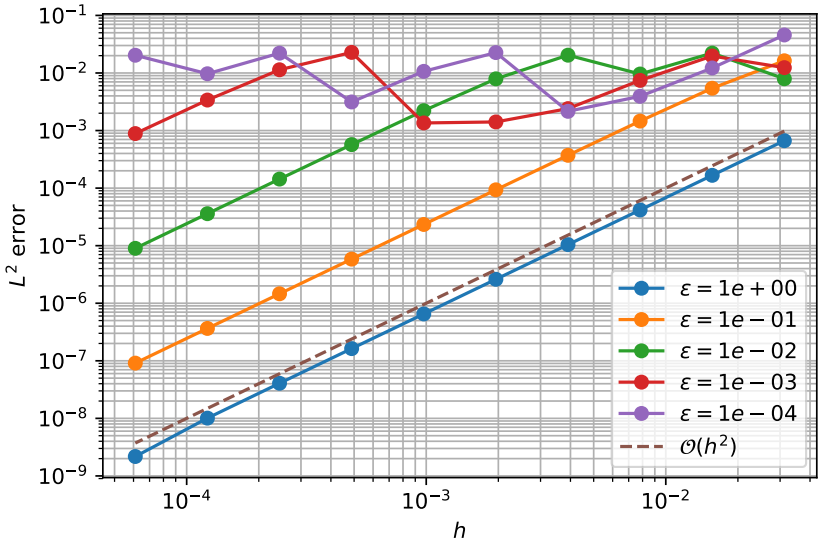


Figure 3.3. L^2 norms of the error of standard FEM with P1 elements for different ϵ.

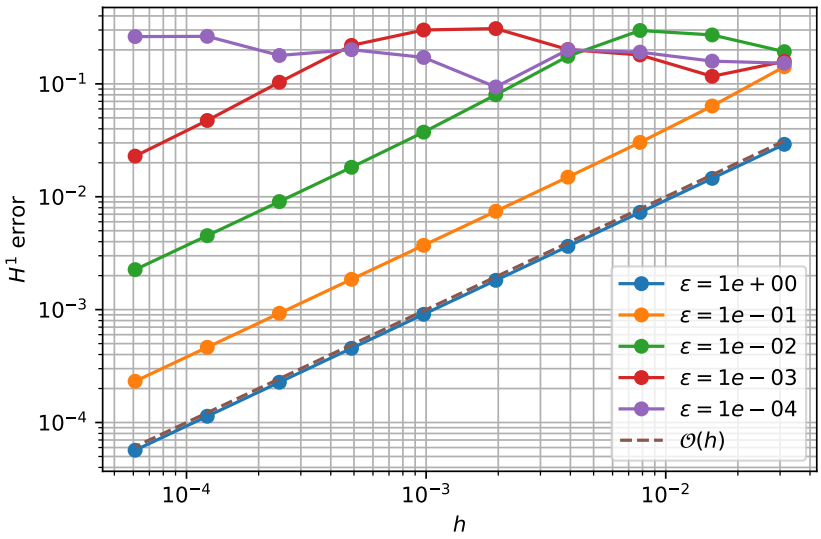


Figure 3.4. H^1 norms of the error of standard FEM with P1 elements for different ϵ.

4. Localized orthogonal decomposition

4.1 Multiscale solvers

Let $\mathcal{T}_c$ be a family of coarse meshes and $\mathcal{T}_f$ be a family of fine meshes assumed to consist of conforming and shape regular simplicial cells. The meshes $\mathcal{T}_c$ and $\mathcal{T}_f$ are alligned such that each cell in $\mathcal{T}_f$ belongs to exactly one cell in $\mathcal{T}_c$. We denote number of vertices as N_c and N_f for the coarse and fine scale systems.

Let $\mathcal{T}_{ms} := (\mathcal{T}_c, \mathcal{T}_f)$ be a two level mesh partitioning of domain Ω. Figure 4.1 is a simple illustration of multiscale mesh in two-dimensional space for a unit square domain. The fine discretization is assumed to be fine enough to capture all microscopic features.

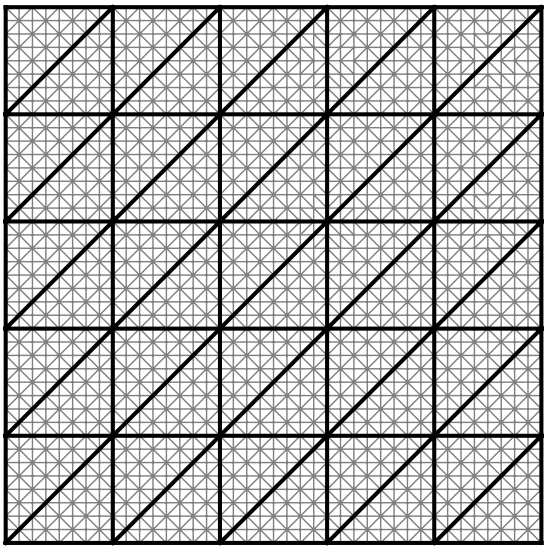


Figure 4.1. 2D multiscale mesh ($\mathcal{T}_{ms}$): the black lines indicate the coarse mesh ($\mathcal{T}_c$) and the grey lines indicate the fine mesh ($\mathcal{T}_f$).

The fine-scale finite element discretization of the Equation (3.25) written in the form of the linear system of equations is

$$\mathbf{A}_f \mathbf{u}_f = \mathbf{b}_f, \tag{4.1}$$

where $\mathbf{u}_f$ is the solution vector in the space associated with the fine mesh

$\mathcal{T}_f$, which is of dimension N_f. The direct solution of the system (4.1) can be computationally expensive. The computational cost can be reduced by reducing the dimensionality of the problem.

Let $\mathbf{u}_c$ be the coarse solution defined on $\mathcal{T}_c$ and $\mathcal{P}$ be a prolongation operator, that maps coarse-scale entities to the fine scale. In addition, we define $\mathcal{R}$ to be a restriction operator, which is an inverse mapping from the fine scale system to the coarse scale. Algebraically these operators are characterized as matrices $\mathbf{P}$ of size $N_f \times N_c$ and $\mathbf{R}$ of size $N_c \times N_f$. Then we can define the fine scale solution $\mathbf{u}_f$ on $\mathcal{T}_f$ as

$$\mathbf{u}_f = \mathbf{P}\mathbf{u}_c. \tag{4.2}$$

Inserting the new representation of $\mathbf{u}_f$ into the system (4.1) results in reducing number of degrees of freedom,

$$\mathbf{A}_f(\mathbf{P}\mathbf{u}_c) = \mathbf{b}. \tag{4.3}$$

Then we left multiply this system with the restriction operator in order to reduce number of equations,

$$\mathbf{R}(\mathbf{A}_f(\mathbf{P}\mathbf{u}_c)) = \mathbf{R}\mathbf{b}_f, \tag{4.4}$$

$$(\mathbf{R}\mathbf{A}_f(\mathbf{P})\mathbf{u}_c = \mathbf{b}_c, \tag{4.5}$$

$$\mathbf{A_c}\mathbf{u}_c = \mathbf{b}_c. \tag{4.6}$$

Solution to the coarse system (4.6) is obtained by

$$\mathbf{u}_c = (\mathbf{A_c})^{-1}\mathbf{b}_c. \tag{4.7}$$

Then the multiscale solution is computed by

$$\mathbf{u}^{ms} = \mathbf{P}(\mathbf{A_c})^{-1}\mathbf{b}_c \equiv \mathbf{P}(\mathbf{R}\mathbf{A_f}\mathbf{P})^{-1}\mathbf{R}\mathbf{b_f}. \tag{4.8}$$

Note that the solution $\mathbf{u}^{ms}$ is equal to $\mathbf{u}_f$ exactly when a prolongation operator can be defined such that Equation (4.2) is satisfied exactly, in most cases $\mathbf{u}^{ms}$ is an approximation to $\mathbf{u}_f$. Therefore, the quality of the multiscale solution depends on the choice of the prolongation and restriction operators.

Multiscale methods differ from each other by the choice how the prolongation operator $\mathcal{P}$ and the restriction operator $\mathcal{R}$ are constructed. For the restriction operator we choose to use a Galerkin restriction operator, which is defined as

$$\mathcal{R} = \mathcal{P}^T. \tag{4.9}$$

The prolongation operator represents the multiscale basis functions. These basis functions are usually obtained by solving some local problem for each coarse cell in some local domain for the given cell. We choose to use Localized Orthogonal Decomposition (LOD) method [4] for constructing the prolongation operator.

4.2 Orthogonal decomposition of the solution space

The idea of the LOD method is to construct a finite-dimensional space $V_H^{ms} \subset V := H_0^1(\Omega)$ in which convergence happens before $h < \epsilon$. Therefore, accurate results can be obtained by using low dimensional subspaces of the solution space V.

We define finite element space associated with the coarse mesh as

$$V_H := \{v_H \in V \,|\, \forall T \in \mathcal{T}_c : v_H|_T \in P_1\}$$

In order to characterize the functions of the solution space V that are not captured by V_H we introduce the space of finescale functions

$$W := \{w \in V \,|\, I_H w = 0\} = \mathbf{ker}(I_H),$$

where I_H is the (quasi)-interpolation operator $I_H : V \to V_H$.

Any function $v \in V$ can be decomposed as

$$v = I_H v + (1 - I_H)v = v_H + w$$

where $v_H = I_H v \in V_H$ is the nodal interpolation of v at the vertices x_j and $w = (1 - I_H)v \in W$ is the error of interpolation. Therefore the solution space V can be decomposed as

$$V = V_H \oplus W.$$

Keeping W fixed, define a new low-dimensional (associated with $\mathcal{T}_c$) space $V_H^{ms} \subset V$ as the subspace that satisfies

$$V = V_H^{ms} \oplus W \quad \text{and} \quad a(V_H^{ms}, W) = 0,$$

i.e.,

$$V_H^{ms} := \{v_H^{ms} \in V \,|\, \forall w \in W \,:\, a(v_H^{ms}, w) = 0\}$$

The Galerkin method with subspace V_H^{ms} applied to model problem seeks $u_H^{ms} \in V_H^{ms}$ such that

$$a(u_H^{ms}, v_H^{ms}) = L(v_H^{ms}) \quad \forall v_H^{ms} \in V_H^{ms}. \tag{4.10}$$

Define correction operator $-\mathcal{Q} : V \to W$ to be the a-orthogonal projection onto the subspace $W \subset V$. Its complementary projection $(1 - (-\mathcal{Q}))$ maps V_H onto V_H^{ms}. Now reformulating Equation (4.10) as finite element method with modified bilinear form: find $u_H \in V_H$ such that

$$a(u_H + \mathcal{Q}u_H, v_H + \mathcal{Q}v_H) = L(v_H + \mathcal{Q}v_H) \quad \forall v_H \in V_H. \tag{4.11}$$

Theorem 4.2.1 (A priori error estimate of the ideal method)**.** *The problem (4.11) admits a unique solution* $u_H \in V_H$ *for any* $L \in H^{-1}(\Omega)$ *and the error*

is bounded by

$$\|u - u_H\|_{H^1(\Omega)} \leq C h_c \|f\|_{L^2(\Omega)}.$$

Proof. See Lemma 3.1 in [4]. □

By Theorem 4.2.1 it is guaranteed that the presented LOD method possesses first order convergence for the error in the H^1-norm independent of variations in the problem coefficient.

4.3 Characterization of the correction operator $\mathcal{Q}$

We introduce a decomposition of the corrector operator $\mathcal{Q}$,

$$\mathcal{Q}v := \sum_{K \in \mathcal{T}_c} \mathcal{Q}^K v, \quad \forall v \in V,$$

where $\mathcal{Q}^K v \in W$ are obtained by solving

$$a(\mathcal{Q}^K v, w)_\Omega = -a(v, w)_K \quad \forall w \in W, \tag{4.12}$$

where

$$a(v, w)_D := \int_D \kappa \nabla u \cdot v, \text{ for an arbitrary set } D \subseteq \Omega.$$

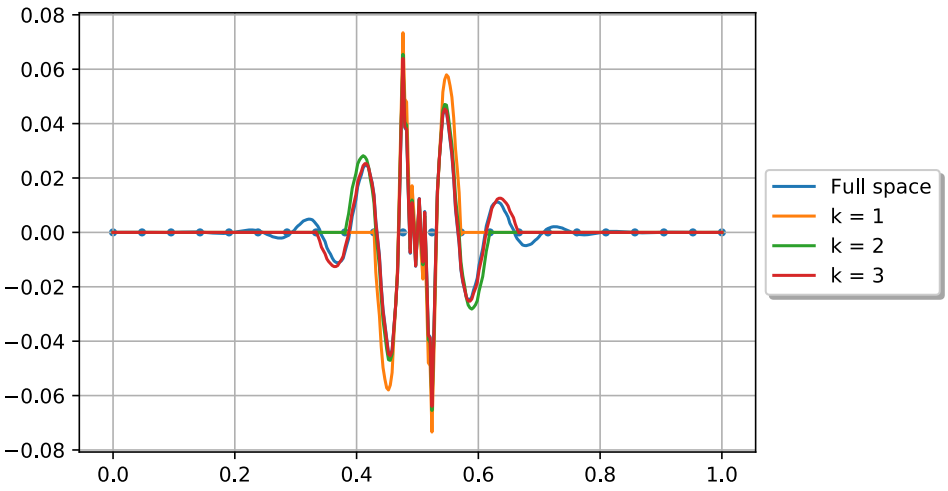


Figure 4.2. Solutions for the local problem in full W space and in patches of different size.

It is proved in [4] that $\mathcal{Q}^K v$ decays exponentially, meaning quickly goes to small practically zero values outside the cell K, therefore solving of local problems can be restricted to local patches of neighboring elements around K. Figure 4.2 shows that with increasing the size of the local patch the solution is becoming closer to the full space solution.

Let $U_k(K)$ define a patch consisting of K and k-surrounding layers of

elements. Then restriction of W to a patch $U(K)$ is defined by

$$W(U(K)) := \{w \in W \,|\, w = 0 \in \Omega \setminus U(K)\}$$

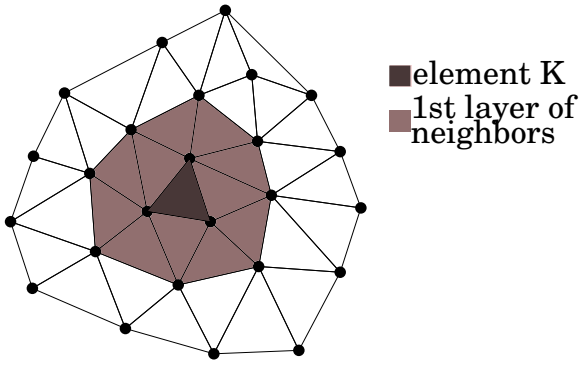


Figure 4.3. Element K with neighbors

The essence of the LOD method is to use the localized corrector operator $\mathcal{Q}_k := \sum_{K \in \mathcal{T}_c} \mathcal{Q}_k^K v$, obtained by solving the localized version of problem (4.12),

$$a(\mathcal{Q}_k^K v, w)_{U_k(K)} = -a(v, w)_K \quad \forall w \in W. \tag{4.13}$$

The localized problems are cheaper to solve than the full problems and they are independent of each other. The following theorem suggests that the choice $k \approx |\log h_c|$ recovers the convergence rate of the ideal method.

Theorem 4.3.1 (A priori error estimate of the localized method)**.** *The localized problem (4.11) admits a unique solution $u_H \in V_H$ and the error is bounded by*

$$\|u - u_H\|_{H^1(\Omega)} \leq C(h_c + k^d e^k) \|f\|_{L^2(\Omega)}.$$

Proof. See Theorem 5.2 in [23]. □

4.4 Characterization of the interpolation operator I_H

Interpolation operator $I_H : V \to V_H$ is important part in defining local problems. It has the form of

$$I_H v = \sum_{x \in \mathcal{N}} (I_H v)(x) \Phi_x,$$

Interpolation should satisfy the following assumptions from [24]:

(i) $I_H : V \to V_H$ is linear and continuous,

(ii) the restriction on V_H is an isomorphism,

(iii) the stability estimate

$$H_K^{-1} \|v - I_H v\|_{L^2(K)} + \|\nabla I_H v\|_{L^2(K)} \leq C_{I_H} \|\nabla v\|_{L^2(U(K))}, \tag{4.14}$$

for every $v \in V$, $K \in \mathcal{T}_c$, with $C_{I_H} > 0$,

(iv) there exist C'_{I_H}, which does not depend on the local mesh size h, such that, for all $v_H \in V_H$, there exist $v \in V$ with properties

$$I_H v = v_H,$$

$$\|\nabla v\|_{L^2(\Omega)} \leq C'_{I_H} \|\nabla v_H\|_{L^2(\Omega)},$$

$$\text{supp } v \subset \text{supp } v_H$$

Assumptions (ii) and (iv) are satisfied if I_H is a projection. L^2-projection is one possible candidate. For L^2-projection assumption (i) holds by definition and assumption (iii) has been proven in [24].

5. Implementation

5.1 FEniCS Project

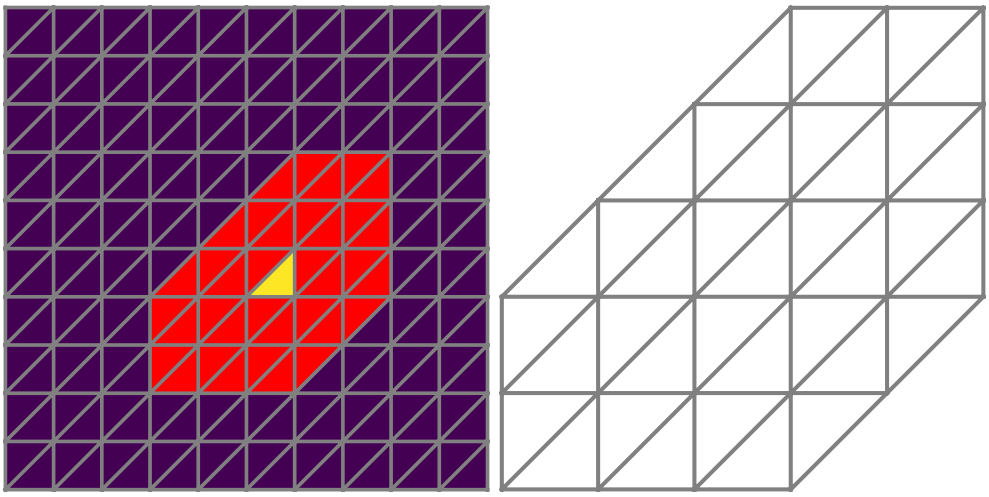
FEniCS Project is a parallel software library for solving partial differential equations by the finite element method [25]. The core code DOLFIN is written in C++ and also has a Python interface [26]. One of the high-level features is the use of a Unified Form Language (UFL) which mimics the mathematical language used for weak formulations of partial differential equations [27]. FEniCS supports a wide variety of finite elements defined on triangles, quadrilaterals, tetrahedrons, or hexahedrons. In this work the Python interface to FEniCS is used together with PETSc [28] as a linear algebra backend.

5.2 Domain discretization

The first step in the solution process of PDE is discretization of the domain. The two level mesh partitioning of the domain Ω, namely $\mathcal{T}_{ms}$, is needed for LOD method. In the developed code, this is realized with the new class `MultiscaleMesh`, which takes as input `Mesh` object representing the coarse mesh ($\mathcal{T}_c$) and provides methods for getting the fine mesh ($\mathcal{T}_f$) from the coarse mesh together with "coarse_cell-to-fine_cells" connectivity information. In the current implementation the fine mesh is obtained by refinement of the coarse mesh. The refinement does not need to be uniform, in fact adaptive refinement based on some error markers can be used, but every coarse cell has to be refined at least once.

"Cell-to-cell" connectivity of the coarse mesh together with "coarse_cell-to-fine_cells" connectivity is used to construct a set of localized domains $\{U_k(K^i)\}$ for each coarse cell K. FEniCS Project has possibility of creating `SubMesh` objects. These objects are created based on the connectivity markers and represent a separate mesh on which localized function space can be defined. Figure 5.1 shows an example of such submesh.

Once the mesh is created, the finite element space can be defined. The finite dimensional subspaces of V and W associated with the fine mesh $\mathcal{T}_f$



(a) Example of `MeshFunction` for a patch with 2 layers of neighboring elements

(b) Example of `SubMesh`

Figure 5.1. Domain localization $U_2(K)$

are denoted as V_h and W_h be and defined as

$$V_h := \{v_h \in V \,|\, \forall T \in \mathcal{T}_f : v_h|_T \in P_1\},$$

$$W_h := W \cap V_h.$$

In the code, the finite element space is obtained with `FunctionSpace`.

5.3 Computing of the correction operator $\mathcal{Q}_k$

The discretized version of the correction operator $\mathcal{Q}_k$ is obtained by solving the local problems (4.12) for every $v_H \in V_H$. It is difficult to define the space W_h explicitly. In order to solve numerically the local problem (4.12) the test function w_h and the trial function $\mathcal{Q}_k^K v_H$ need to be in V_h instead of W_h. Let $I_H^h : V_h \to V_H$ be L^2-projection. A function $v_h \in V_h$ is in the kernel of the L^2-projection if it holds

$$(v_h, \Phi_i)_{L^2} = 0 \quad \text{for dofs in } V_H, \tag{5.1}$$

where Φ_i is the i-th basis function of V_H. Now the local problem (4.12) can be solved in $V_h(U_k(K))$ space with the constraint from Equation (5.1). The constraint is realized using Lagrange multipliers.

Define bilinear forms related to the Lagrange multipliers of the local problem:

$$b(u, d) = \sum_{j=0}^{N_c} d_j(\Phi_j, u)_{L^2(U_k(K))} \quad \forall d \in \mathbb{R}^{N_c}$$

$$b(c, v) = \sum_{j=0}^{N_c} c_j (\Phi_j, v)_{L^2(U_k(K))} \quad \forall v \in V_h$$

Then the local problem is redefined as: Find $(u, c) \in V_h \times \mathbb{R}^{N_c}$ such that

$$a((u, c), (v, d)) = L((v, d)) \quad \forall (v, d) \in V_h \times \mathbb{R}^{N_c}, \tag{5.2}$$

where

$$\begin{aligned} a((u, c), (v, d)) &:= (\kappa \nabla u, \nabla v)_{L^2(U_k(K))} \\ &\quad + \sum_{j=0}^{N_c} d_j (\Phi_j, u)_{L^2(U_k(K))} + \sum_{j=0}^{N_c} c_j (\Phi_j, v)_{L^2(U_k(K))}, \\ L((v, d)) &:= -(\kappa \nabla \Phi_H, \nabla v)_{L^2(K)}, \\ u &:= \mathcal{Q}_k^K v_H. \end{aligned}$$

Source Code 5.1. FEniCS routines to compute the correction operator Q_k

```
from dolfin import *
import scipy.sparse

# Given an instance of "MultiscaleMesh" class, varying coefficient "kappa"

def compute_correction_operator(multiscale_mesh, kappa):
    """
    Return correction operator in the matrix form with shape V_f.dim() x V_c.dim().
    """
    # Initialize sparse matrix with given shape to store the corrector
    Q = scipy.sparse.csr_matrix((V_f.dim(), V_c.dim()))
    for cell_id in range(multiscale_mesh.coarse_mesh.num_cells()):
        Q += local_solve(multiscale_mesh, cell_id, kappa)
    return Q

def local_solve(multiscale_mesh, cell_id, kappa):
    """
    Solve local problem for a given cell.
    Return contribution of the given cell to the global corrector matrix.
    """
    # Create local solution domain
    local_patch_coarse = SubMesh(multiscale_mesh.coarse_mesh,
                                 multiscale_mesh.coarse_submesh[cell_id], 1)
    local_patch_fine = SubMesh(multiscale_mesh.fine_mesh,
                               multiscale_mesh.fine_submesh[cell_id], 1)
    # RHS has to be integrated over given cell (K_local)
    # Lines for K_local_f and K_local_c, which are MeshFunction's, are omitted here.
    # The two functions are the markers for the given coarse cell and its fine cells
    # Define local function spaces
    V_c = FunctionSpace(local_patch_coarse, 'P', 1)
    V_f = FunctionSpace(local_patch_fine, 'P', 1)
    # Create local coarse-to-fine transfer (prolongation) matrix
```

```
    P_H = PETScDMCollection.create_transfer_matrix(V_c, V_f)
    # Create P1 element for V_h
    P1 = FiniteElement('P', local_patch_fine.ufl_cell(), 1)
    # Create R^N_c element
    R = VectorElement('Real', local_patch_fine.ufl_cell(), 0, V_c.dim())
    # Define mixed function space V_h x R^N_c
    W = FunctionSpace(local_patch_fine, MixedElement([P1, R]))
    # Solution should vanish on boundary and outside the patch
    bc = DirichletBC(W.sub(0), Constant(0.0), "on_boundary")
    # Define variational problem
    u, c = TrialFunctions(W)
    v, d = TestFunctions(W)
    # LHS of the local problem
    a = inner(kappa*grad(u), grad(v))*dx
    # Construct a basis for the nullspace and RHS
    basis_functions = []
    P1_function = Function(V_f)
    for j in range(V_c.dim()):
        basis_functions.append(P1_function.copy(deepcopy=True))
        #"P_H[:, j]"is the projection of the coarse lagrange basis functions to fine space
        basis_functions[j].vector()[:] = P_H.mat().getColumnVector(j).getArray()
    # Wrap "basis_functions" object as a vector
    bf = as_tensor(basis_functions)
    # Add lagrange multiplier contributions to the bilinear form
    a += inner(bf, d)*u*dx + inner(bf, c)*v*dx
    # RHS of the local problem. Here RHS integral is over K cell as required
    phi_H = Function(V_f)
    L = - inner(kappa*grad(phi_H), grad(v))*dx(subdomain_data=K_local_f, subdomain_id=1)
    # Assemble A matrix and apply b.c.
    A = assemble(a)
    bc.apply(A)
    # Obtain dofs of the given coarse cell
    K_local_cell_dofs = V_c.dofmap().cell_dofs(int(numpy.nonzero(K_local_c.array())[0]))
    for i in K_local_cell_dofs:
        phi_H.assign(basis_functions[i])
        b = assemble(L)
        bc.apply(b)
        # Solve the problem
        h = Function(W)
        solve(A, h.vector(), b)
        # "h" is a mixed function containing solutions u and c. Extract u solution
        # and save to "w"
        usol = h.split(True)[0]
        w.append(usol.vector().get_local())
    # Then save solutions to full corrector matrix Q using correct indices
    # that map local solution to global one
    return Q
```

The implementation of the local problem solver is presented in Listing (5.1). The code itself is not more complicated than the program for solving

standard Poisson problem would be. It follows the standard procedure: first the domain and function spaces are defined on lines 21-39, then lines 42-60 correspond to the definition of bilinear and linear forms of the problem (5.2) in the UFL syntax, and finally the solutions are obtained using direct method and saved into sparse matrix $\mathbf{Q} \in \mathbb{R}^{N_f \times N_c}$. Here the assembly and solving the linear system is taken care of by FEniCS. The other approach would be to use FEniCS only for assembling separate stiffness matrix, constraint matrix and RHS vector. Then manipulate the matrices explicitly to form the block matrix and solve the system using some linear algebra library. The presented approach in the code is preferred in this work as it resembles the mathematical language of the local problem (5.2).

5.4 Implementation of the multiscale solver

Listing (5.2) shows how model problem can be solved using FEniCS library. The similar program is used to assemble the fine-scale system, then the fine-scale system is transformed to the coarse-scale LOD system to obtain the multiscale solution.

Source Code 5.2. A FEniCS program to solve model problem using standard FEM

```
1  from dolfin import *
2  # Define kappa-coefficient dependent on epsilon
3  epsilon = Constant(1e-5)
4  kappa = Expression('1.0/(2.0 + cos( (2.0 * pi * x[0]) / epsilon))',
5                  degree = 1, epsilon = epsilon)
6  # Create mesh for the problem
7  mesh = UnitSquareMesh(16, 16)
8  # Create P1 function space
9  V = FunctionSpace(mesh, 'P', 1)
10 # Define boundary conditions
11 def boundary(x):
12     return x[0] < DOLFIN_EPS or x[0] > 1 - DOLFIN_EPS
13 bc = DirichletBC(V, Constant(0.0), boundary)
14 # Define variational problem
15 u = TrialFunction(V)
16 v = TestFunction(V)
17 f = Constant(1.0)
18 a = inner(kappa*grad(u), grad(v))*dx
19 L = f*v*dx
20 # Assemble the stiffness matrix and right hand side vector
21 u = Function(V)
22 problem = LinearVariationalProblem(a, L, u, bc)
23 solver = LinearVariationalSolver(problem)
24 solver.solve()
```

In the context of LOD the multiscale prolongation operator $\mathcal{P}$ is defined as a sum of the normal (in the sense of standard multigrid methods)

prolongation operator $\mathcal{P}^H$ and correction operator $\mathcal{Q}_k$,

$$\mathcal{P} = \mathcal{P}^H + \mathcal{Q}_k,$$

where $\mathcal{P}^H$ represents the basis functions of V_H in terms of the basis functions of V_h. $\mathcal{P}^H$ is computed by evaluating coarse basis functions at dofs of V_h, it can be done effectively using in-built FEniCS function `create_transfer_matrix(V_H, V_h)`. Practically the coarse-scale matrix $\mathbf{A}_H^{LOD}$ and the load vector $\mathbf{b}_H$ arising by assembling bilinear and linear forms of Equation (4.11) are constructed by Galerkin projection using prolongation operator $\mathcal{P}$. The coarse-scale matrix is obtained by

$$\mathbf{A}_H^{LOD} = (\mathbf{P}_H + \mathbf{Q})^T \mathbf{A}_h (\mathbf{P}_H + \mathbf{Q}),$$

similarly, the load vector is obtained

$$\mathbf{b}_H = (\mathbf{P}_H + \mathbf{Q})^T \mathbf{b}_h,$$

where $\mathbf{A}_h$ and $\mathbf{f}_h$ are the stiffness matrix and load vector assembled in V_h. Then the following linear system is solved to get the coarse-scale solution $\mathbf{u}_H^{LOD} \in \mathbb{R}^{N_c}$:

$$\mathbf{A}_H^{LOD} \mathbf{u}_H^{LOD} = \mathbf{b}_H.$$

Finally, the fine-scale solution $\mathbf{u}_h^{LOD} \in \mathbb{R}^{N_f}$ is obtained by

$$\mathbf{u}_h^{LOD} = (\mathbf{P}_H + \mathbf{Q}) \mathbf{u}_H^{LOD}.$$

Described steps of matrix transformations are wrapped in classes `LinearMultiscaleProblem` and `NonlinearMultiscaleProblem` allowing simple interfacing with FEniCS solvers. To apply LOD method a few lines of code corresponding need to be added to the existing program (5.2):

Source Code 5.3. A modified FEniCS program to solve model problem using LOD.

```
# Given an instance of "MultiscaleMesh" class, varying coefficient "kappa"
...
V_c = FunctionSpace(multiscale_mesh.coarse_mesh, 'P', 1)
V_f = FunctionSpace(multiscale_mesh.fine_mesh, 'P', 1)
...
# Compute prolongation matrix P
Q = compute_correction_matrix(multiscale_mesh, kappa)
P_H = compute_transfer_matrix(V_c, V_f)
P = P_H - Q
u_f = Function(V_f)
u_c = Function(V_c)
multiscale_problem = LinearMultiscaleProblem(a, L, u_f, u_c, bcs)
solver = LinearSolver(problem)
# Compute solution
solver.solve()
```

6. Numerical results

This chapter contains the description and results of the simulation of three test problems. The first test problem has a known analytical solution and is used to verify the implementation and to check the efficiency. The second problem is a modification of classic Stefan problem with powder-like material used to prove that the multiscale methods can be applied to nonlinear problems. The third problem is a one laser pass over the powder bed to demonstrate a starting point for additive manufacturing process simulation.

6.1 The analytical problem

In Section 3.3, an example on the effect of oscillations in the diffusion coefficient was presented. It was shown that as long as the mesh size is larger than the frequency of oscillations the approximate solution is inaccurate. The same example problem is used to study the convergence and the accuracy of the implemented multiscale solver.

Simulations are performed on the unit square domain $\Omega = (0,1)^2$. Diffusion coefficient is

$$\kappa = \left(2 + \cos\left(\frac{2\pi x}{\epsilon}\right)\right)^{-1},$$

where $\epsilon = 0.05$. The right hand side of the problem is $f = 1$. Dirichlet boundary conditions $u = 0$ are set on the left and right boundaries. The coarse mesh is obtained by dividing the unit square domain into n squares and then dividing each square into two triangles. The underlying fine mesh is obtained by applying three uniform refinements to the coarse mesh. The sequence of meshes is constructed by creating a sequence of n evenly spaced from 2^3 to 2^5 on a log scale. Information about the created multiscale meshes is gathered in the Table 6.1.

Figure 6.1 shows convergence of the H^1- and L^2- error norms. Results tell that with the ideal multiscale method the convergence is achieved immediately irrespective of the size of the coarse mesh, which is in accordance with Theorem 4.2.1. Moreover, solutions obtained by ideal multiscale method are exactly equal (to the machine precision) to the solutions obtained by

Table 6.1. Information about the meshes.

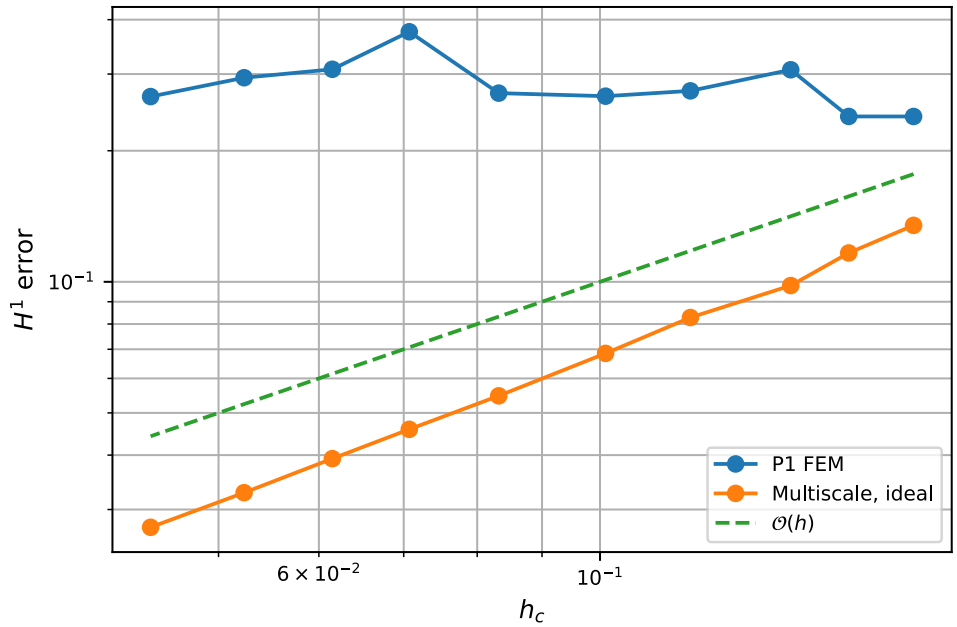
Mesh	Number of cells		Number of vertices	
	Coarse	Fine	Coarse	Fine
8x8	128	8192	81	4225
9x9	162	10368	100	5329
10x10	200	12800	121	6561
12x12	288	18432	169	9409
14x14	392	25088	225	12769
17x17	578	36992	324	18769
20x20	800	51200	441	25921
23x23	1058	67712	576	34225
27x27	1458	93312	784	47089
32x32	2048	131072	1089	66049

P1 FEM on the underlying fine mesh. Figure 6.2 displays convergence for different sizes k of local patches. The accuracy of the method increases with larger local patches as expected by Theorem 4.3.1. In the case $k = 1$ the convergence is far from being optimal and at some point the rate of convergence goes to zero, i.e. the norm of the error does not decrease anymore. Other cases are much closer to the ideal case, however the memory consumption and computational cost increases with increasing k. Table 6.2 gives information about the median size of the local problems needed to solve for each coarse cell and each degree of freedom in the cell. Local systems have the size of coarse+fine number of vertices. From tables 6.1, 6.2 one can deduce the amount of work needed to obtain the corrector matrix Q.

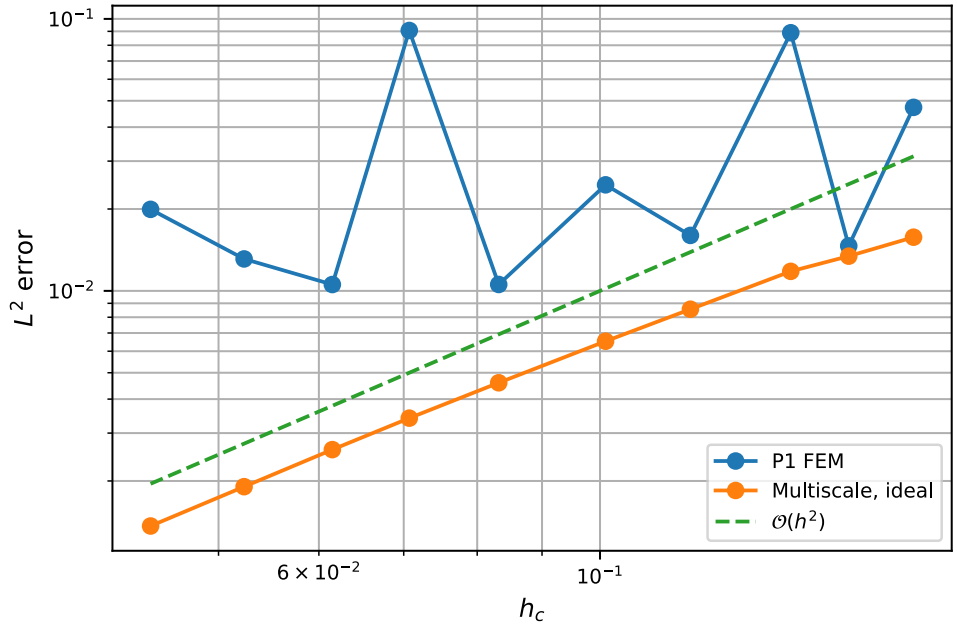
Convergence plots prove that the implementation of the multiscale method is correct, but more important is to study the computational efficiency of the method compared to the standard P1 FEM. Time spent on computing the corrector matrix Q, assembly of $\mathbf{A}_c$ and $\mathbf{b}_c$, and solving the linear system $\mathbf{A}_c\mathbf{u}_c = \mathbf{b}_c$ was measured. For the standard method, time spent on assembly operation and solving the system was measured. All computations were

Table 6.2. Information about the size of local patches.

Mesh	Median number of vertices in the local patch							
	$k = 1$		$k = 2$		$k = 3$		$k = 4$	
	Coarse	Fine	Coarse	Fine	Coarse	Fine	Coarse	Fine
8x8	12	453	22	967	33	1545	46	2247
9x9	12	453	23	1017	35	1673	49	2429
10x10	12	453	23	1017	37	1773	50	2521
12x12	12	453	24	1081	39	1901	54	2735
14x14	12	453	27	1245	40	1965	57	2913
17x17	12	453	27	1245	43	2129	62	3205
20x20	12	453	27	1245	44	2193	64	3333
23x23	12	453	27	1245	48	2421	69	3625
27x27	12	453	27	1245	48	2421	70	3689
32x32	12	453	27	1245	48	2421	75	3981



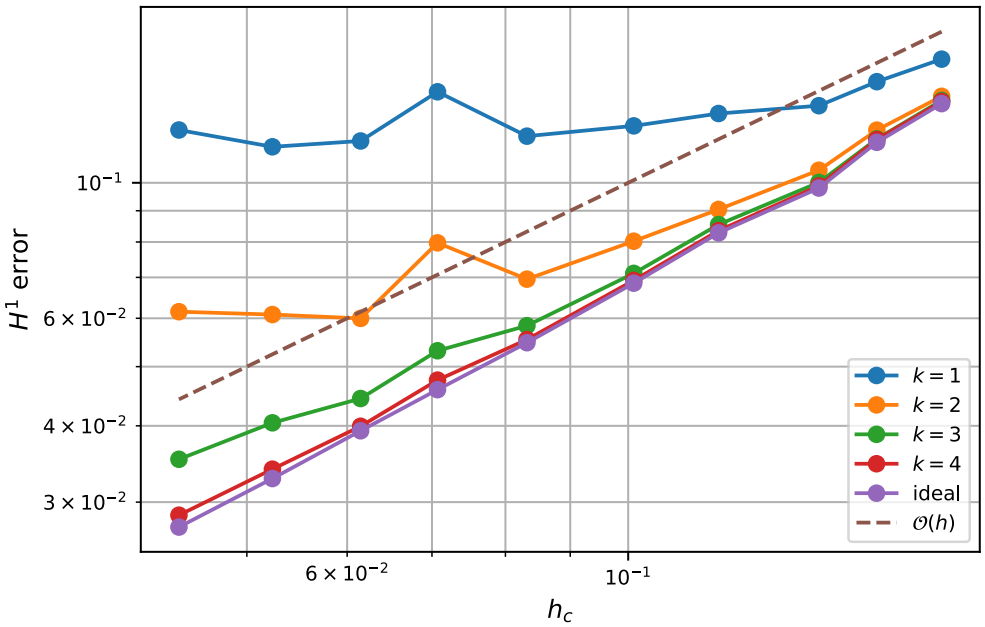
(a) H^1 norm of the error.



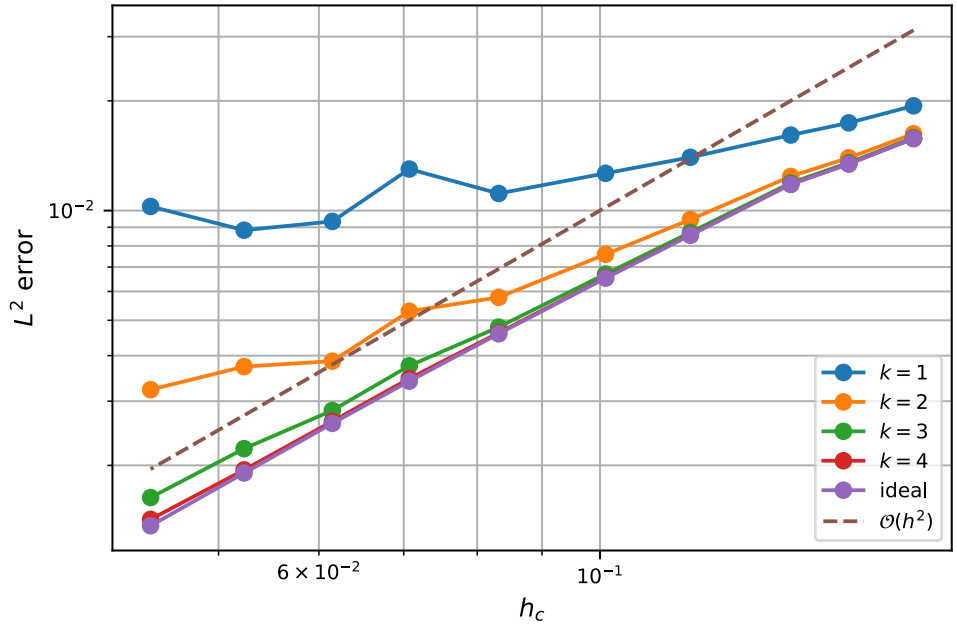
(b) L^2 norm of the error.

Figure 6.1. Comparison of P1 FEM and ideal multiscale method.

performed on Intel® Xeon® E5-2670. The correctors were computed in parallel using `pathos` module [29] and employing the pool of 30 processes. Other operations were performed in serial. LU decomposition was used for solving the linear systems. From figures 6.1, 6.2 it is seen that the multiscale solutions obtained with $k = 4$ are the closest to the ideal case, but the amount of time spent to get the same solution is approximately three orders



(a) H^1 norm of the error.



(b) L^2 norm of the error.

Figure 6.2. Norms of the error for different size k of local patches.

of magnitude larger (Figure 6.3). The reason is being that there are a lot of additional computations involved in the multiscale method compared to the standard one. Figure 6.4 provides timings for the multiscale method with different k. The rate of increase of the spent time seems to be larger than $\mathcal{O}(N)$. Each additional layer in the local patch results in the increase of the spent time by the order of magnitude approximately.

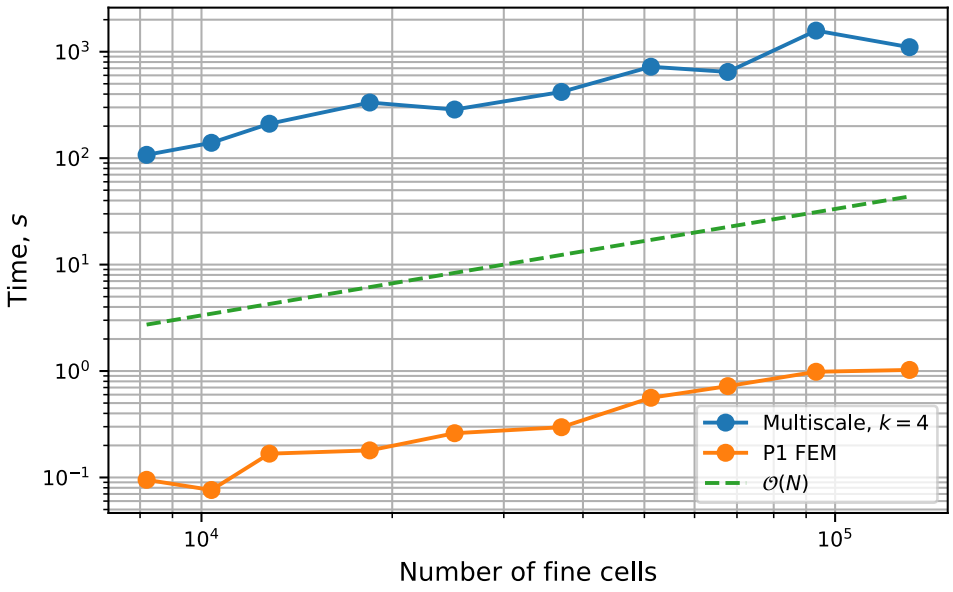


Figure 6.3. Total time spent for computing multiscale and its corresponding exact finescale solutions.

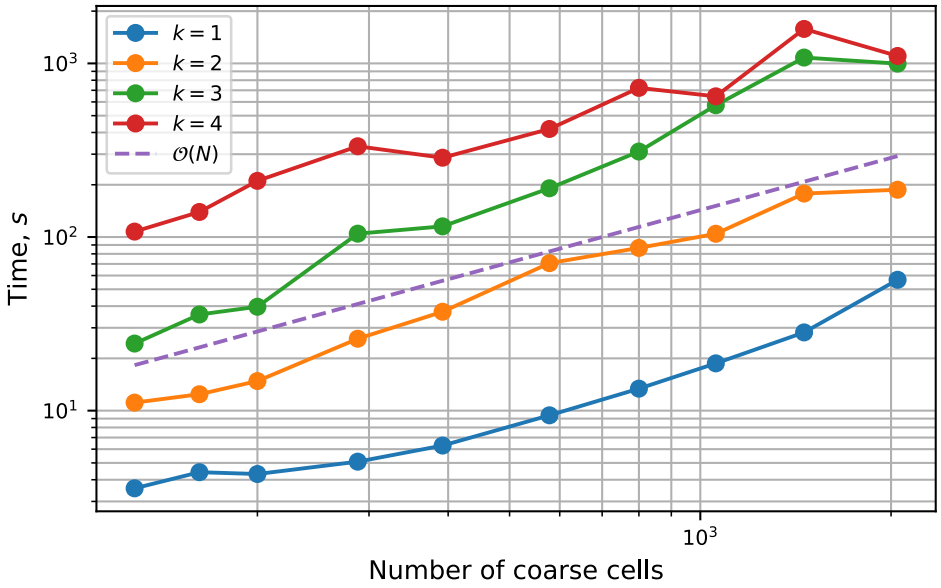


Figure 6.4. Total time spent for computing multiscale solution for different size k of local patches.

The timings show clearly that the method cannot be applied efficiently to elliptic linear PDEs, where the assembly and solving happens once. In all cases P1 FEM for the corresponding fine problem is much faster. Huge computational cost of obtaining the multiscale basis makes the method worthless to use, even though the resulting multiscale system is cheaper to solve than the corresponding finescale system. This is because computing of the multiscale basis functions takes 99% of the total time and they are used only once. Therefore, in order to take advantage of the method the computed corrector has to be reused. In the case of parabolic PDEs, the

sequence of problems is solved at different time points. Thus, potentially the multiscale method can outperform standard P1 FEM after some number of iterations.

First consider the case where the system has to be reassembled at each different iteration, for example when using Newton's method for a non-linear problem, but diffusion coefficient remains constant throughout the simulation. Figure 6.5 shows time spent on assembly and solve operations for the multiscale method and P1 FEM. The multiscale method outperforms the standard one only in the cases when $k = 1, 2$. For $k = 3, 4$ the corrector matrix is less sparse, therefore cost of assembly of the coarse system increases. Time spent on solving the resulting linear system does not differ much between methods with different k (Figure 6.6). For linear solve timings multiscale method outperforms P1 FEM in all cases.

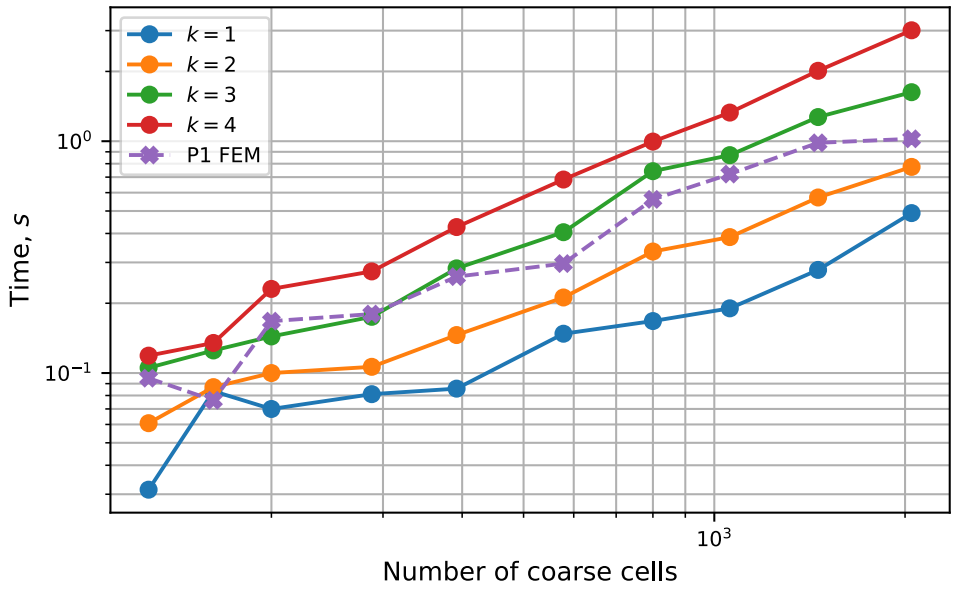


Figure 6.5. Time spent on assembly and solving the system for different size k of local patches and P1 FEM.

These timings suggest that there should be an optimal number showing how many times at minimum corrector matrix has to be reused in order to outperform the standard method. In the case where reassembly is needed at each iteration this number is computed by t_{corr}/t_{as}, where t_{corr} is the time spent to compute the corrector and t_{as} is the time spent on assembling and solving. And if the assembly is done only once then the optimal number of iterations is obtained by $(t_{corr} + t_a)/t_s$, where t_a and t_s are the time spent on assembling and solving the system correspondingly. Approximate findings are presented in the Table 6.3.

The other way of gaining efficiency from using the multiscale method is decreasing the total time spent on computing Q. Naive parallelization can be applied, since local problems that are needed to solve are completely independent of each other. For example, Figure 6.7 shows strong scaling of computing the corrector matrix for 14x14 coarse mesh and $k = 2$.

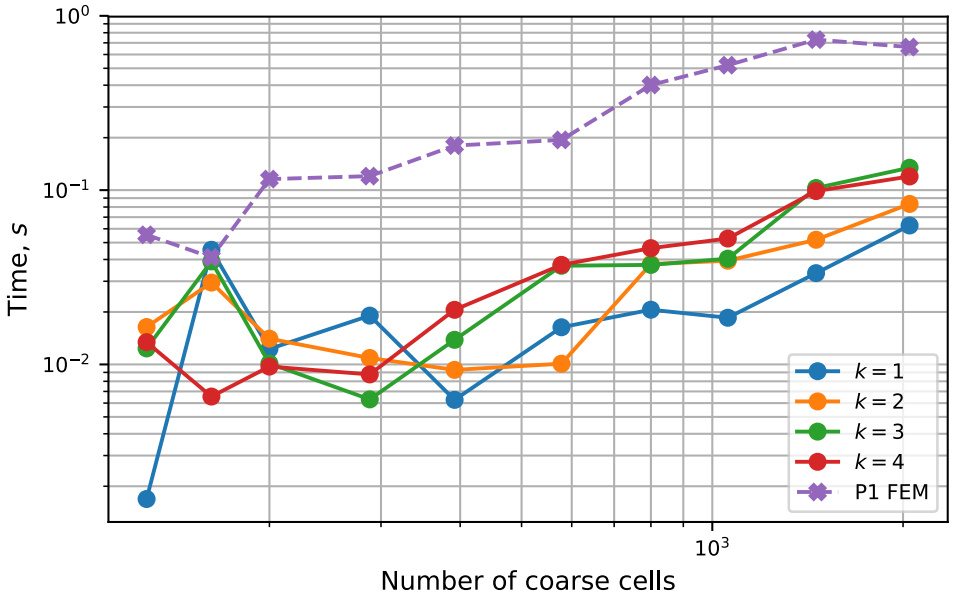


Figure 6.6. Time spent on solving the system for different size k of local patches and P1 FEM.

Table 6.3. Minimum number of iterations with reusing the Q in order to outperform P1 FEM.

	Assembly+Solve	Solve
k=1	115	2111
k=2	333	7004
k=3	impossible	16593
k=4	impossible	38017

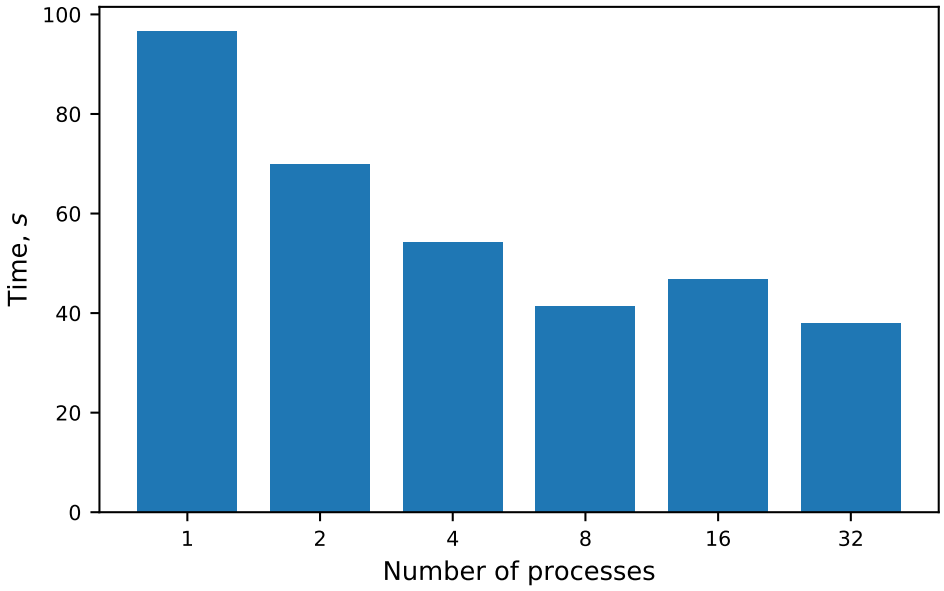


Figure 6.7. Strong scaling for computing Q.

6.2 The heterogeneous Stefan problem

The following numerical experiment demonstrates capability of the implemented solver to approximate the solution of nonlinear multiscale problem with rough and non-periodic coefficient. Heat transfer problem of a powder packing subject to phase changes is considered. Let computational domain Ω be the unit square filled with some gas and randomly distributed spherical particles. Conductivity of the gas and the particles is denoted as κ_g and κ_p correspondingly. Let c_g and c_p be the specific heat of the gas and the powder particles. Initial temperature T_0 is set below the melting point. On the left boundary temperature is $T(x_0 = 0.0) = T_h$, and on the right boundary it is prescribed to be $T(x_0 = 1.0) = T_c$. There is no external forcing, $f = 0$, the heat flow is driven by the temperature difference between the boundaries. The powder material heats up progressively starting from the left boundary, changing its state from "solid" to "liquid", phase change process is accompanied with absorption of energy. Parameters used for this simulation case are presented in the Table 6.4. For simplicity, identical material parameters are set for "solid" and "liquid" states of the powder and gaseous part of the problem does not undergo any phase changes.

Table 6.4. Parameters for heterogeneous Stefan problem test case.

T_h	$1.0\,^{\circ}\mathrm{C}$
T_c	$-0.01\,^{\circ}\mathrm{C}$
c_g	$1.0\,\mathrm{J/kg\,K}$
c_p	$1.0\,\mathrm{J/kg\,K}$
κ_g	$0.01\,\mathrm{W/m\,K}$
κ_p	$1.0\,\mathrm{W/m\,K}$

The phase change case is controlled by the Stefan number parameter

$$Ste = \frac{c\Delta T}{L},$$

with c being the specific heat (of solid for freezing case or of liquid for melting case), ΔT is the temperature difference between phases, L is the latent heat of melting/solidification. The phase function is regularized using $r_{reg} = 0.005$. In this case, Stefan number is set to $Ste = 0.009$. Simulation runs until $t = 1.0$ with constant intervals $dt = 0.1$.

Conductivity field κ for the given problem is presented in Figure 6.8, which also shows that variations in the conductivity do not lay on the scale of the mesh size. Thermal conductivity κ is assumed to be constant in time, thus there is no need to recalculate the correction operator.

As a result of the simulation coarse and reconstructed fine scale solutions are obtained (Figure 6.9). Reconstructed solution reveals the effect of heterogeneous conductivity field, while coarse scale solution is smooth yet accurate representation of the heterogeneous temperature field.

However, comparing the multiscale method results to P1 FEM solution

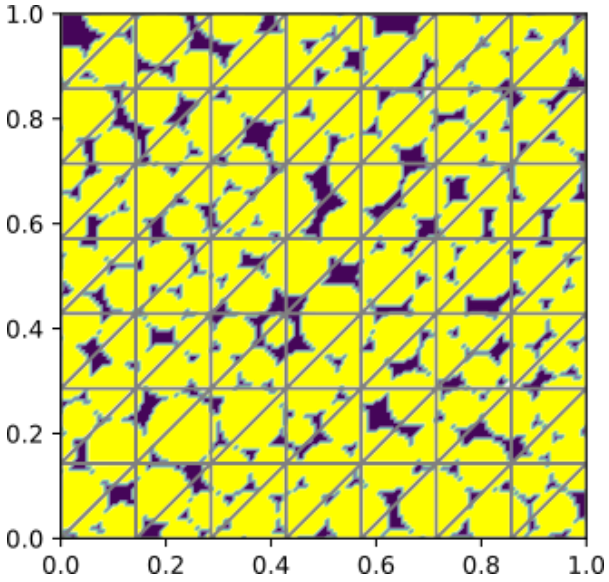
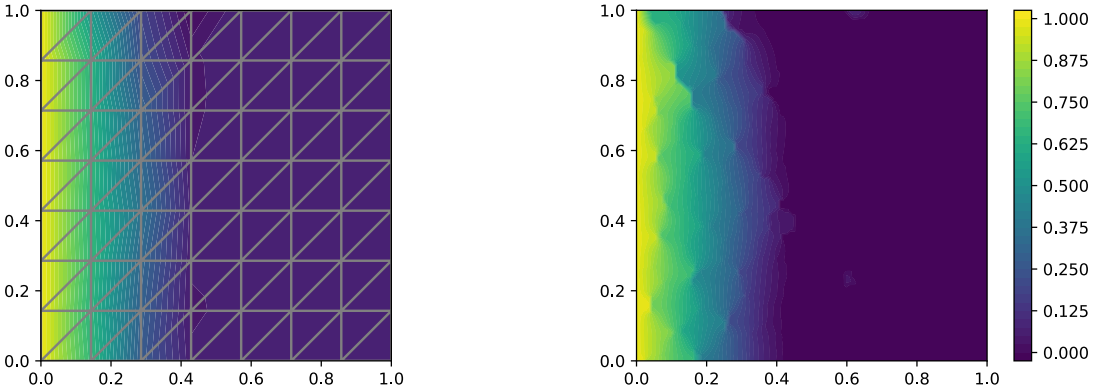


Figure 6.8. Conductivity field κ together with the coarse mesh. κ_p is represented by yellow color, κ_g is in the background.



(a) Coarse scale solution. **(b)** Reconstructed fine scale solution.

Figure 6.9. Results for the heterogeneous Stefan problem at the final time.

suggests that LOD method is not well suited for capturing phase interface position correctly (Figure 6.10).

6.3 Selective laser melting process simulation

The last test case shows the possibility to use the developed solver for the simulation of selective laser melting process. The problem description and parameters are adapted from [12] and [13]. In both of these studies, the powder layer is considered to be a homogeneous material with low thermal conductivity compared to the consolidated solid material. In the given numerical experiment, the powder layer is represented by a heterogeneous media consisting of metal spherical particles and gas.

Implemented solver works in 3D as well, but 3D domain is quite demanding computationally, therefore the test case is restricted to two dimensions. Size of the computational domain is specified to be $0.6\,\mathrm{mm} \times 0.2\,\mathrm{mm}$. There are two physical domains: the consolidated substrate material with a pow-

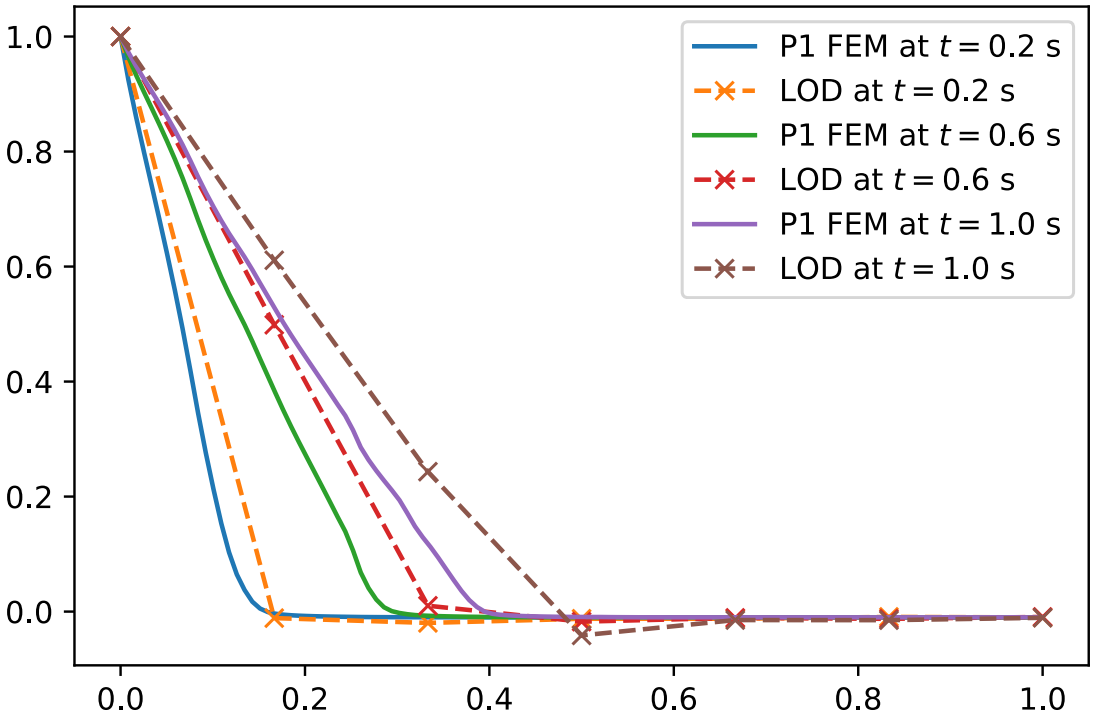


Figure 6.10. Comparison of multiscale method and P1 FEM solutions to the Stefan problem.

der layer on top of it with thickness $0.050\,\mathrm{mm}$. The laser moves along the x direction, with speed $v_{source} = 120\,\mathrm{mm/s}$ over the range $0 \leq x \leq 0.5\,\mathrm{mm}$, heating the powder and transforming it into the consolidated material. The effective laser power is taken to be $W_e = 30\,\mathrm{W}$. The radius of the laser beam is $R = 0.060\,\mathrm{mm}$. One Dirichlet boundary condition is set $T(x = 0.6\,\mathrm{mm}) = 303\,\mathrm{K}$.

Table 6.5. Parameter values for the SLM process simulation.

	Parameter	Value
$(\rho c)_s$	specific heat, solid steel	$4.25\,\mathrm{MJ/m^3\,K}$
$(\rho c)_l$	specific heat, liquid steel	$5.95\,\mathrm{MJ/m^3\,K}$
$(\rho c)_g$	specific heat, gas	$1.0\,\mathrm{kJ/m^3\,K}$
κ_s	thermal conductivity, solid steel	$20.0\,\mathrm{W/m\,K}$
κ_l	thermal conductivity, liquid steel	$20.0\,\mathrm{W/m\,K}$
κ_g	thermal conductivity, gas	$0.03\,\mathrm{W/m\,K}$
ρ_h	hemispherical reflectivity	0.7
β_h	extinction coefficient	$60.000\,1/\mathrm{m}$
T_m	melting point	$1700\,\mathrm{K}$
r_{reg}	regularization radius	25.0
L	latent heat of melting	$2.18\,\mathrm{GJ/m^3}$

The material considered is 316L stainless steel. Material properties used in the simulation are given in Table 6.5. The values taken here are not exactly the same as presented by Gusarov, et al. in [12], since the goal is to show the possibility to use heterogeneous material parameters on the coarse mesh, and not to reproduce the results exactly. For simplicity of implementation the material parameters are assumed to be constant in each individual state. The powder is modelled as a random packing of

spheres with log-normal distributed radius with mean $r_{mean} = 1.0 \times 10^{-5}$ and standard deviation $\sigma_p = 3.5 \times 10^{-6}$. Material properties for the powder domain include some gas and steel either in liquid or solid phase. The maximum temperature is tracked throughout the simulation. If maximum temperature at the node ever exceeds the melting temperature then the material is considered consolidated and this portion of the domain can be represented now only by consolidated liquid or solid steel, geometry of metal spheres is then ignored, it is assumed to be melted together with the surrounding material and solidified. Gas is assumed to escape the system, and no gas entrapment is considered here. Also, no fluid convection nor mechanical material response is modelled.

The coarse mesh is constructed by dividing the domain into 24×8 squares and then each square into two triangles. The region of powder layer is refined four times to get the underlying fine mesh for `MultiscaleMesh` construction. Time step is taken to be $dt = 1 \times 10^{-5}$. The corrector matrix is computed only for the powder region. The columns that correspond to the nodes that went through the phase change are removed at each time iteration. It reduces the accuracy to P1 FEM on the coarse scale, but gives the possibility to calculate the corrector matrix only once and reuse it. Since it is assumed that the powder melts completely and forms a homogeneous material, this is a reasonable technique.

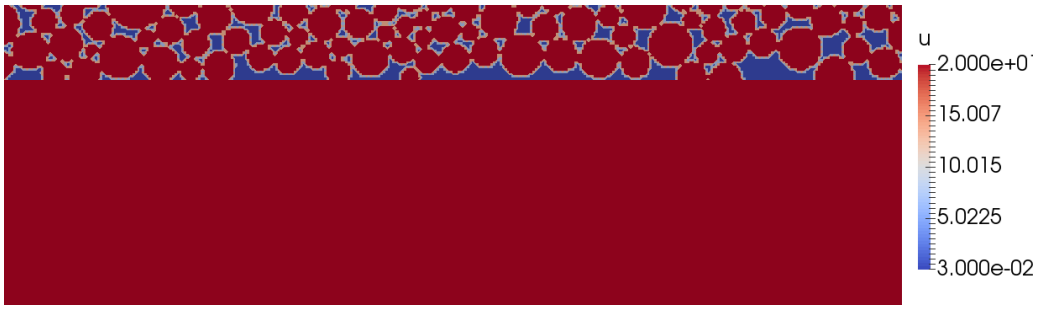
The results are displayed in Figures 6.11 and 6.12. Variations in the temperature profiles are visible in front of the laser path, that are due to the variations in the conductivity field. The thermal conductivity profiles showcase the expected behavior: once the laser passes over the powder material, it transforms into the consolidated material.

(a) Temperature profile at 100th time interval.

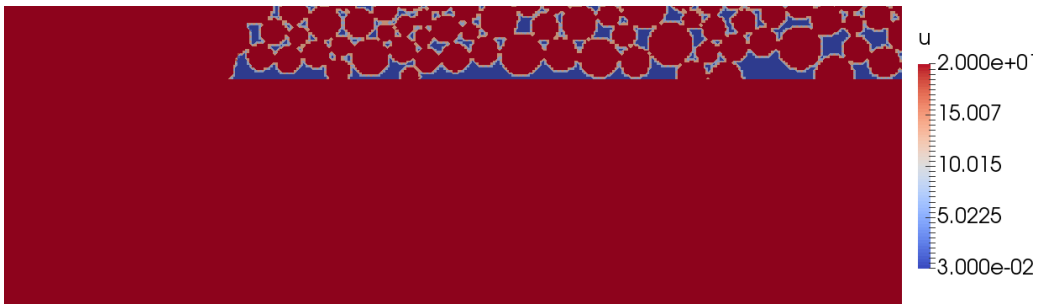
(b) Temperature profile at 300th time interval.

(c) Temperature profile at the final time.

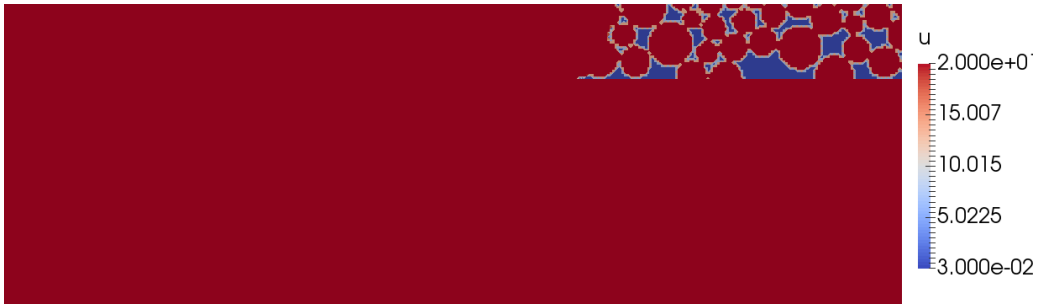
Figure 6.11. Temperature profiles for the simulation of laser pass over heterogeneous layer of powder.



(a) Thermal conductivity field at initial time.



(b) Thermal conductivity field at 100th time interval.



(c) Thermal conductivity field at 300th time interval.

Figure 6.12. Conductivity field for the simulation of laser pass over heterogeneous layer of powder.

7. Summary

7.1 Conclusions

In this thesis, the multiscale finite element framework for solving steady-state and transient heat equation with highly heterogeneous coefficient was successfully developed. Convergence of the method was verified. The linear convergence in H^1 and quadratic convergence in L^2 was obtained, if local patch size k was big enough. It was shown that the multiscale solution is equal to the solution computed with standard methods on the same fine mesh.

The main idea of the work was based on the two level domain decomposition and proper prolongation and restriction operators to move between the coarse and fine levels. Multiscale basis functions that are used for construction of the operators were computed using Localized Orthogonal Decomposition. The method requires local problems to be solved for each cell in the coarse mesh. These local problems were reformulated as saddle point problems with Lagrange multipliers imposing the needed constraint. Operator based implementation is rather general and allows switching between different methods for construction of multiscale basis with little effort.

Each local problem is independent of others, therefore this computation can be naively parallelized. This was done using simple method of spawning a new process for each individual local problem and summing up the result. The performance of the parallelization was demonstrated.

While the method has a good accuracy, it was also shown that it brings significant additional computational cost. In order to hide the cost, the computed corrector matrix has to be reused, for example in time-stepping or newton iterations. Timings for the corrector computation, assembly and solve operations were presented and analyzed. Results suggest that the assembly operation is dominating over solving the linear system. Original wrong assumption was that the linear solve operation is more time consuming, and it motivated the construction of the equivalent reduced dimension system. However, in some cases there is still a possibility to gain

computational efficiency. An example estimates were given on minimum number of iterations needed to outperform P1 FEM.

Two numerical examples of the simplified heat transfer in heterogeneous media with phase changes were presented. Even though the solver is capable of solving nonlinear problems, it was not able to capture the phase interface position correctly. That is important as it introduces additional errors in the simulation that accumulate over time.

In light of the above, it is challenging to apply LOD technique in realistic additive manufacturing process simulations in the current state. The implementation should be further improved or a different faster method for constructing multiscale basis functions shall be formulated.

7.2 Future work

This thesis has demonstrated the potential and some drawbacks of the LOD method. Future work for extending the scope of this thesis concerns different directions.

- Optimizing individual local problem

  The main bottleneck of the LOD method is calculating the local problems. Even slight improvement in the efficiency of individual local problems will result in significant overall speedup for large problems.

- Better parallel implementation

  Even though the solver was implemented to work in parallel using MPI, applying LOD method for 3D selective laser melting simulation is not feasible currently. The construction of corrector matrix is limited to the serial run. Some amount of work needs to be done to implement the correct distribution of the matrix over the processes. Computation of the local problems is naively parallelized using simple tools of spawning new processes and gathering the results. The efficiency needs to be investigated and possibly better solution using MPI should be used. The computers that were used had at most only 16 cores, and therefore no more than 32 threads were created. Moving to the computational cluster with many distributed computers is the way to go.

- Automatic solution to multiscale PDEs

  The goal of FEniCS Project is to automate solution of PDEs utilizing the abstract variational formulation. In this work, computation of the corrector matrix was implemented specifically for the poisson equation. In the future, first step towards automatic solutions of multiscale PDEs would be to implement the local problem computation for a general bilinear form. The presented orthogonal decomposition method can be possibly applied not only to simple spaces of piecewise

polynomials, but for other spaces as well, like Raviart-Thomas, mixed spaces, etc., opening the possibility to have automated formulation of multiscale problems for a wide range of PDEs.

There exist several different techniques for constructing the multiscale basis function, their performance and accuracy should be investigated and compared to determine the optimal choice.

Bibliography

[1] M. Francois, A. Sun, W. King, N. Henson, D. Tourret, C. Bronkhorst, N. Carlson, C. Newman, T. Haut, J. Bakosi, J. Gibbs, V. Livescu, S. Vander Wiel, A. Clarke, M. Schraad, T. Blacker, H. Lim, T. Rodgers, S. Owen, F. Abdeljawad, J. Madison, A. Anderson, J.-L. Fattebert, R. Ferencz, N. Hodge, S. Khairallah, and O. Walton, "Modeling of additive manufacturing processes for metals: Challenges and opportunities", *Current Opinion in Solid State and Materials Science*, vol. 21, no. 4, pp. 198–206, Aug. 2017. DOI: 10.1016/j.cossms.2016.12.001 (cit. on p. 1).

[2] W. E and B. Engquist, "The heterognous multiscale methods", *Commun. Math. Sci.*, vol. 1, no. 1, pp. 87–132, Mar. 2003 (cit. on p. 2).

[3] T. Y. Hou and X.-H. Wu, "A multiscale finite element method for elliptic problems in composite materials and porous media", *Journal of Computational Physics*, vol. 134, no. 1, pp. 169–189, 1997 (cit. on p. 2).

[4] A. Malqvist and D. Peterseim, "Localization of Elliptic Multiscale Problems", Oct. 2011. arXiv: 1110.0692 (cit. on pp. 2, 22, 24).

[5] T. J. Hughes, G. R. Feijóo, L. Mazzei, and J.-B. Quincy, "The variational multiscale method—a paradigm for computational mechanics", *Computer Methods in Applied Mechanics and Engineering*, vol. 166, no. 1, pp. 3–24, 1998, Advances in Stabilized Methods in Computational Mechanics (cit. on p. 2).

[6] D. D. Gu, W. Meiners, K. Wissenbach, and R. Poprawe, "Laser additive manufacturing of metallic components: materials, processes and mechanisms", *International Materials Reviews*, vol. 57, no. 3, pp. 133–164, May 2012. DOI: 10.1179/1743280411Y.0000000014 (cit. on p. 4).

[7] M. Rombouts, L. Froyen, A. V. Gusarov, E. H. Bentefour, and C. Glorieux, "Photopyroelectric measurement of thermal conductivity of metallic powders", DOI: 10.1063/1.1832740 (cit. on p. 5).

[8] V. Voller, "An overview of numerical methods for solving phase change problems", *Advances in numerical heat transfer*, vol. 1, no. 9, pp. 341–380, 1997 (cit. on p. 6).

[9] C. R. Swaminathan and V. R. Voller, "A general enthalpy method for modeling solidification processes", *Metallurgical Transactions B*, vol. 23, no. 5, pp. 651–664, Oct. 1992. DOI: 10.1007/BF02649725 (cit. on p. 6).

[10] D. Celentano, E. Oñate, and S. Oller, “A temperature-based formulation for finite element analysis of generalized phase-change problems”, *International Journal for Numerical Methods in Engineering*, vol. 37, no. 20, pp. 3441–3465, Oct. 1994. DOI: 10.1002/nme.1620372004 (cit. on p. 6).

[11] N. Shamsundar and E. M. Sparrow, “Analysis of Multidimensional Conduction Phase Change Via the Enthalpy Model”, *Journal of Heat Transfer*, vol. 97, no. 3, p. 333, Aug. 1975. DOI: 10.1115/1.3450375 (cit. on p. 7).

[12] A. V. Gusarov, I. Yadroitsev, P. Bertrand, and I. Smurov, “Model of Radiation and Heat Transfer in Laser-Powder Interaction Zone at Selective Laser Melting”, *Journal of Heat Transfer*, vol. 131, no. 7, p. 072 101, Jul. 2009. DOI: 10.1115/1.3109245 (cit. on pp. 8, 9, 41, 42).

[13] N. E. Hodge, R. M. Ferencz, and J. M. Solberg, “Implementation of a thermomechanical model for the simulation of selective laser melting”, *Computational Mechanics*, vol. 54, no. 1, pp. 33–51, Jul. 2014. DOI: 10.1007/s00466-014-1024-2 (cit. on pp. 8, 41).

[14] S. Chandrasekhar, *Radiative Transfer*, ser. Dover Books on Intermediate and Advanced Mathematics. Dover Publications, 1960 (cit. on p. 8).

[15] W. E. King, A. T. Anderson, R. M. Ferencz, N. E. Hodge, C. Kamath, S. A. Khairallah, and A. M. Rubenchik, “Laser powder bed fusion additive manufacturing of metals; physics, computational, and materials challenges”, *Applied Physics Reviews*, vol. 2, no. 4, p. 041 304, Dec. 2015. DOI: 10.1063/1.4937809 (cit. on p. 11).

[16] D. He, N. Ekere, and L. Cai, “Computer simulation of random packing of unequal particles”, *Physical review E*, vol. 60, no. 6, p. 7098, 1999 (cit. on p. 11).

[17] P. Lax and A. Milgram, “Parabolic Equations”, *Annals of Mathematics Studies*, vol. 33, pp. 167–190, 1954 (cit. on p. 13).

[18] D. Braess, *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*, 3rd ed. Cambridge University Press, 2007. DOI: 10.1017/CBO9780511618635 (cit. on p. 14).

[19] P. Ciarlet, *Numerical analysis of the finite element method*. Quebec: Les Presses de L’Université de Montréal, 1976 (cit. on p. 14).

[20] A. Logg, K.-A. Mardal, G. N. Wells, *et al.*, *Automated Solution of Differential Equations by the Finite Element Method*, A. Logg, K.-A. Mardal, and G. N. Wells, Eds. Springer, 2012. DOI: 10.1007/978-3-642-23099-8 (cit. on p. 14).

[21] S. Brenner and L. Scott, *The Mathematical Theory of Finite Element Methods*, ser. Texts in Applied Mathematics. Springer New York, 2013 (cit. on p. 18).

[22] D. Peterseim, "Variational Multiscale Stabilization and the Exponential Decay of Fine-scale Correctors", May 2015. arXiv: 1505.07611 (cit. on p. 19).

[23] ——, *Lecture notes in numerical homogenization of partial differential equations*, Jan. 2017 (cit. on p. 25).

[24] D. Peterseim and R. Scheichl, "Robust Numerical Upscaling of Elliptic Multiscale Problems at High Contrast", *Computational Methods in Applied Mathematics*, vol. 16, no. 4, pp. 579–603, Jan. 2016. DOI: 10.1515/cmam-2016-0022 (cit. on pp. 25, 26).

[25] M. S. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M. E. Rognes, and G. N. Wells, "The fenics project version 1.5", *Archive of Numerical Software*, vol. 3, no. 100, 2015. DOI: 10.11588/ans.2015.100.20553 (cit. on p. 27).

[26] A. Logg and G. N. Wells, "Dolfin: Automated finite element computing", *ACM Transactions on Mathematical Software*, vol. 37, no. 2, 2010. DOI: 10.1145/1731022.1731030 (cit. on p. 27).

[27] M. S. Alnæs, A. Logg, K. B. Ølgaard, M. E. Rognes, and G. N. Wells, "Unified form language: A domain-specific language for weak formulations of partial differential equations", *ACM Transactions on Mathematical Software*, vol. 40, no. 2, 2014. DOI: 10.1145/2566630 (cit. on p. 27).

[28] S. Balay, S. Abhyankar, M. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. Gropp, D. Kaushik, M. Knepley, D. May, L. C. McInnes, K. Rupp, P. Sanan, B. Smith, S. Zampini, H. Zhang, and H. Zhang, "PETSc users manual", Argonne National Laboratory, Tech. Rep. ANL-95/11 - Revision 3.8, 2017 (cit. on p. 27).

[29] M. M. Mckerns, L. Strand, T. Sullivan, A. Fang, and M. A. G. Aivazis, "Building a Framework for Predictive Science", *Proc. of the 10th Python in Science conference*, 2011 (cit. on p. 35).