# MULTISCALE MODEL REDUCTION AND DEEP REINFORCEMENT LEARNING FOR THE SELECTED APPLICATIONS

A Dissertation

by

ZECHENG ZHANG

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

| | |
|---|---|
| Chair of Committee, | Yalchin Efendiev |
| Co-Chair of Committee, | Tsz Shun Eric Chung |
| Committee Members, | Eduardo Gildin |
| | Raytcho Lazarov |
| Head of Department, | Sarah Witherspoon |

May 2021

Major Subject: Mathematics

ABSTRACT

In many applications arising from geosciences, one needs to solve problems with multiple scales. For examples, simulating the complex flow, porous media applications and so on. To capture the multiscale features of a problem with heterogeneous property by the traditional finite element method, one need to use the fine mesh. Consequently the computation is accurate but the efficiency is compromised because of the large number of the degree of freedoms used. One idea is to improve the computation efficiency while preserve the accuracy is to design basis which can resolve multiscale features of the problem on coarse mesh. A class of methods have been proposed based on this methodology and in this work, I am going to develop the multiscale methods on solving some important problems. The first two applications of the multiscale modeling in this work are based on the Quasi-gas-dynamic (QGD) model. In particular, we study QGD model in a multiscale environment. This is not only because of the challenges in solving and analyzing the equation; but also because of its wide applications in solving other types of equations, for example, paraxial wave equations.

It should be noted that the key step in the multiscale methods is to find the multiscale basis defined on the coarse mesh. This is the most time consuming step which usually involves constructing snapshot or auxiliary space, and solving spectral problems. In particular, there are heavy computations when one is dealing with nonlinear or time dependent problems. Besides basis are problems dependent, i.e., given a new heterogeneous input, one need to evaluate a new set of basis; hence it is not flexible to apply the methods on solving stochastic problems .

Deep learning is a branch in the machine learning and started to show its power in computer science since 2010's. It is accurate meanwhile very efficient in many computer vision and language processing applications. Deep learning usually consists of two steps: training and testing. Given a model which is well trained by a data set, one can evaluate the new example which has common features as the training data set very efficiently and accurately. This motivates us solving the multiscale problems with the deep learning approach. In the second part of this work, we will

show some problems which are solved by combining the multiscale methods and deep learning. The deep learning approach indeed improves the efficiency of the traditional method.

# ACKNOWLEDGMENTS

I want to express my thanks to Professor Yalchin Efendiev and Professor Eric Chung for their supervision and help. They introduced me to the world of the multiscale finite element methods and modeling. They made great and important contributions to the area and showed me their deep understandings of the theory which is extraordinary helpful.

What is important, Yalchin and Eric introduced me with some interesting, important and unsolved problems in the area. Some problems have been solved with the supervision of two professors and I will summarize these works in this paper. Professors also gave me the freedom to study what I was interested and provided me with the resources to research in the area.

Professors always give me their suggestions on my academic career. Professor Efendiev introduced me with the future job opportunity and I really appreciate their help. In all, I am grateful to Professor Yalchin Efendiev and Professor Eric Chung for their help throughout the years.

I would also like to thank Raytcho Lazarov and Professor Eduardo Gildin for serving as my committee members, guiding my research and providing me with suggestions in the past years.

My Ph. D. study was also enriched by my collaborators and friends. I want to thank you for all the discussions, your help, your suggestions and supports.

Finally and most importantly, my gratitude goes to my family. The love, caring and support from my family give me the courage and anything I need to complete the dissertation.

# CONTRIBUTORS AND FUNDING SOURCES

## Contributors

This work was supervised by a dissertation committee consisting of Professor Yalchin Efendiev and Professor Eric T. Chung of the Department of Mathematics.

All other work conducted for the dissertation was completed by the student independently.

## Funding Sources

Graduate study was supported by a fellowship from Texas A&M University.

# NOMENCLATURE

| | |
|---|---|
| $\Omega$ | Spatial domain |
| $\kappa$ | Permeability |
| $\mathcal{T}^h$ | Fine-scale partition |
| $\mathcal{T}^H$ | Coarse-scale partition |
| $h$ | Fine mesh size |
| $H$ | Coarse mesh size |
| $K$ | Coarse grid element |
| $E$ | Coarse grid edge |
| $\omega$ | Coarse neighborhood |
| $\chi$ | Partition of unity |
| $K_{i,m}$ | Coarse oversampled region |
| $\phi$ | Spectral basis function |
| $\psi$ | Multiscale basis function |

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

# 1. INTRODUCTION

Problems with musicale features widely exist in the real world. To capture the features of all scales of the multiscale problems, fine scale discretization should be used; however, it is computationally expensive to capture heterogeneous properties using very fine mesh. For this reason, we consider the coarse mesh methods. The coarse mesh methods require less number of degree of freedoms (basis) and hence are computational friendly. To resolve the multiscale features and preserve the accuracy, coarse mesh basis which are defined on either coarse elements or the coarse neighborhoods should be well designed. Many methods basing the coarse mesh principle have been proposed and these methods include homogenization-based approaches [1, 2, 3, 4, 5, 6, 7], multiscale finite element methods [8, 9, 10, 11], generalized multiscale finite element methods (GMsFEM) [12, 13, 14, 15, 16, 17, 18], constraint energy minimizing GMsFEM (CEM-GMsFEM) [19, 20], nonlocal multi-continua approaches (NLMC) [21], metric-based upscaling [22], heterogeneous multiscale method [23, 24], localized orthogonal decomposition (LOD) [25, 26], equation free approaches [27, 28, 29], computational continua [30, 31, 32], hierarchical multiscale method [33, 34, 35], and so on. Some of these approaches, such as homogenization-based approaches, are designed for problems with scale separation. In porous media applications, the spatial heterogeneities are complex and do not have scale separation. In addition, they contain large jumps in the coefficients. As a result, the coarse grid does not resolve scales and contrast. For these purposes, we have introduced a more general concept: GMsFEM and CEM-GMSFEM, where multiple basis functions are designed to solve problems on a coarse grid [20, 21]. In this work, I will first review and then apply the GMsFEM and the CEM-GMsFEM approaches to solve a branches important problems.

Coarse mesh basis evaluation plays a key role in the multiscale finite element methods and it requires heavy computation, in particular when one is solving nonlinear or time dependent problems. This step usually involves constructing the snapshot or auxiliary space, and solving a spectral problem. Besides this is a problems dependent step, i.e., given a different heterogeneous input, people

need to design a different set of basis. For problems with stochastic natural, the multiscale method is no longer efficient since people need to solve a large number of multiscale problems.

Deep learning method was first studied in 1980's and researchers have realized its power in solving many computer science problems. It is accurate and reset the benchmarks in many computer vision applications like the recognition, classification, segmentation, super resolution; and it also introduces some new and active research topics like the images generation and captioning. Besides the success in the computer vision, deep learning also makes breakthroughs in other area like the language processing. While the methods create the new benchmarks, the computation speed and memory in the real-time applications are not compromised. Deep learning methods consist of training and testing. People need to first train a model by using a large number training samples. The training step is slow and is usually realized by performing a variant of gradient descent method; however, given a well trained model, people can evaluate the new testing sample which shares some common features as training data set very efficiently and accurately.

Deep learning can also be taken as a model reduction technique. People have established the relation between the neural network layers and image features in the computer vision applications [36, 37, 38]. This motivate us designing networks which play the role of the model reduction. We hence propose to introduce the deep learning in solving problems with multiscale natural. In the rest of this section, I am first going to review the GMsFEM and CEM methods. Next, I will give a brief introduction of the deep learning method. Finally, I am going to summarize the applications of the multiscale methods and the deep learning approach in this dissertation.

### 1.1 Generalized multiscale finite element methods (GMsFEM)

In this section, I will introduce the GMsFEM. For more details about this method, please refer to [12, 13, 14, 15, 16, 17, 18].

Consider the second order elliptic equation $Lu = f$ in $D$ with proper boundary conditions;

denote the the elliptic operator as:

$$L(u) = -\frac{\partial}{\partial x_i}(k_{ij}(x)\frac{\partial}{\partial x_j}u). \tag{1.1}$$

Special case of the elliptic operator is $L(u) = -\nabla \cdot (\kappa \nabla u)$, where $\kappa(x)$ is the permeability field. Let the spatial domain $D$ be partitioned by a coarse grid $\mathcal{T}^H$; this does not resolve the multiscale features. Let us denote $K$ as one cell in $\mathcal{T}^H$ and refine $K$ to obtain the fine grid partition $\mathcal{T}^h$ (blue box in Figure 1.1). We assume the fine grid is a conforming refinement of the coarse grid. See Figure 1.1 for details.

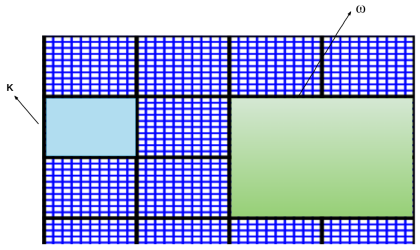


Figure 1.1: Domain Partition $\mathcal{T}^H$. Reprinted with permission from "Computational multiscale methods for quasi-gas dynamic equations" by Boris Chetverushkin, Eric Chung, Yalchin Efendiev, Sai-Mang Pun and Zecheng Zhang. To be published in Journal of Computational Physics.

For the $i$-th coarse grid node, let $\omega_i$ be the set of all coarse elements having the vertex $i$ (green region in Figure 1.1). We will solve local problems in each coarse neighborhood to obtain set of multiscale offline basis functions $\{\phi_i^{\omega_i}\}$; and finally solve the problem using the offline basis.

### 1.1.1 Snapshot space

We need to first construct the snapshot space. There are several options and we will introduce the harmonic extension of the fine grid functions defined on the boundary of $\omega_i$. Let us denote $\delta_l^h(x)$ as fine grid delta function, which is defined as $\delta_l^h(x_k) = \delta_{l,k}$ for $x_k \in J_h(\omega_i)$ where $J_h(\omega_i)$ denotes the boundary nodes of $\omega_i$. The snapshot function $\psi_l^{\omega_i}$ is then calculated by solving local

problem in $\omega_i$:

$$L(\psi_l^{\omega_i}) = 0 \tag{1.2}$$

subject to the boundary condition $\psi_l^{\omega_i} = \delta_l^h(x)$. The snapshot space $V_{snap}^{\omega_i}$ is then constructed as the span of all snapshot functions.

### 1.1.2 Offline spaces

The offline space $V_{off}^{\omega_i}$ is derived from the snapshot space and is used for computing the solution of the problem. We need to solve for a spectral problem and this can be summarized as finding $\lambda$ and $v \in V_{snap}^{\omega_i}$ such that:

$$a_{\omega_i}(v, w) = \lambda s_{\omega_i}(v, w), \forall w \in V_{snap}^{\omega_i}, \tag{1.3}$$

where $a_{\omega_i}$ is symmetric non-negative definite bilinear form and $s_{\omega_i}$ is symmetric positive definite bilinear form. When $L(u) = -\nabla \cdot (\kappa \nabla u)$, by convergence analysis, they are given by

$$a_{\omega_i}(v, w) = \int_{\omega_i} \kappa \nabla v \cdot \nabla w, \tag{1.4}$$

$$s_{\omega_i}(v, w) = \int_{\omega_i} \tilde{\kappa} v \cdot w. \tag{1.5}$$

In the above definition of $s_{\omega_i}$, the function $\tilde{\kappa} = \kappa \sum |\nabla \chi_j|^2$ where $\{\chi_j\}$ is a set of partition of unity functions corresponding to the coarse grid partition of the domain $D$ and the summation is taken over all the functions in this set. The offline space is then constructed by spanning the first $L_i$ eigen-functions corresponding to eigenvalues. Finally we are trying to find $u_{off} \in V_{off}$ such that

$$a(u_{off}, v) = \int_D fv, \forall v \in V_{off} \tag{1.6}$$

where $a(u, v) = \int_D \kappa \nabla u \cdot \nabla v$, if $L(u) = -\nabla \cdot (\kappa \nabla u)$.

## 1.2 Constraint energy minimization GMsFEM (CEM)

The CEM method [19, 20] is also used to capture the multiscale features on the coarse mesh. One of the advantages of the CEM method is that one can prove the first order coarse mesh convergence, from which one can establish many theoretical results by applying the CEM methods on solving other problems. The CEM method is motivated by the the global basis convergence property; however the global basis which are derived by solving the energy minimization problems on the whole domain cannot be used in practice due to the global support of the basis. Researchers hence proposed to build the local basis, which is inspired by the exponential decay property of the global basis. By the estimate of the difference between the global and local basis, one can prove the convergence of the local basis. The multiscale method consists of two steps. First, form an auxiliary space by spectral decomposition. Second, construct multiscale space.

### 1.2.1 The spectral decomposition

We consider solving the same problem as before. Define $V(K_i)$ as the restriction of the abstract space $V$ on the coarse element $K_i$. We consider a local spectral problem: Find $\lambda_j^{(i)} \in \mathbb{R}$ and $\phi_j^{(i)} \in V(K_i)$ such that

$$a_i(\phi_j^{(i)}, v) = \lambda_j^{(i)} s_i(\phi_j^{(i)}, v) \quad \text{for all } v \in V(K_i). \tag{1.7}$$

Here, $a_i : V(K_i) \times V(K_i)$ and $s_i : V(K_i) \times V(K_i)$ can be chosen as:

$$\begin{aligned} a_i(v, w) &= \int_{K_i} \kappa \nabla v \cdot \nabla w, \\ s_i(v, w) &= \int_{K_i} \tilde{\kappa} v \cdot w. \end{aligned}$$

where $\tilde{\kappa} := \sum_{j=1}^{N_c} \kappa |\nabla \chi_j^{\text{ms}}|^2$. The functions $\{\chi_j^{\text{ms}}\}_{j=1}^{N_c}$ are the standard multiscale finite element basis functions which satisfy the partition of unity property. More precisely, $\chi_j^{\text{ms}}$ is the solution of

the following system:

$$\begin{aligned} \nabla \cdot (\kappa \nabla \chi_j^{\text{ms}}) &= 0 \qquad \text{in each } K \subset \omega_j, \\ \chi_j^{\text{ms}} &= g_j \qquad \text{on } \partial K \setminus \partial \omega_j, \\ \chi_j^{\text{ms}} &= 0 \qquad \text{on } \partial \omega_j. \end{aligned}$$

Here, we define $\omega_j := \bigcup \{K : x_j \in \overline{K}\}$ the coarse neighborhood corresponding to the coarse node $x_j$. The function $g_j$ is continuous and linear along the boundary of the coarse element. We assume that the eigenvalues $\lambda_j^{(i)}$ are arranged in ascending order and we pick $\ell_i \in \mathbb{N}^+$ corresponding eigenfunctions to construct the local auxiliary space $V_{\text{aux}}^{(i)} := \text{span}\{\phi_j^{(i)} : j = 1, \cdots, \ell_i\}$. We assume the normalization $s_i\left(\phi_j^{(i)}, \phi_j^{(i)}\right) = 1$. After that, we define the global auxiliary multiscale space $V_{\text{aux}} := \bigoplus_{i=1}^N V_{\text{aux}}^{(i)}$. We remark that the global auxiliary multiscale space is used to construct multiscale basis functions that are orthogonal to the auxiliary space with respect to the weighted $L^2$ inner product $s(\cdot, \cdot)$.

Note that the bilinear form $s_i(\cdot, \cdot)$ defines an inner product with norm $\|\cdot\|_{s(K_i)} := \sqrt{s(\cdot, \cdot)}$ in the local auxiliary space $V_{\text{aux}}^{(i)}$. Based on these local inner products and norms, one can naturally define a new inner product and norm for the global auxiliary space $V_{\text{aux}}$ as follows: for all $v, w \in V_{\text{aux}}$,

$$s(v, w) := \sum_{i=1}^N s_i(v, w) \quad \text{and} \quad \|v\|_s := \sqrt{s(v, v)}. \tag{1.8}$$

The inner product and norm defined above can be extended for the abstract space $V$. In addition, we define $\pi : L^2(\Omega) \to V_{\text{aux}}$ as the projection with respect to the inner product $s(\cdot, \cdot)$ such that

$$\pi u = \pi(u) := \sum_{i=1}^N \sum_{j=1}^{\ell_i} s_i(u, \phi_j^{(i)}) \phi_j^{(i)} \quad \text{for all } u \in L^2(\Omega).$$

### 1.2.2 The construction of the multiscale basis functions

In this section, we present the construction of the multiscale basis functions. First, we define an oversampling region for each coarse element. Specifically, given a non-negative integer $m \in \mathbb{N}$ and a (closed) coarse element $K_i$, we define the oversampling region $K_{i,m} \subset \Omega$ such that

$$K_{i,m} := \begin{cases} K_i & \text{if } m = 0, \\ \bigcup\{K : K_{i,m-1} \cap K \neq \emptyset\} & \text{if } m \geq 1. \end{cases}$$

See Figure 1.2 for an illustration of oversampling region. For simplicity, we denote $K_i^+$ the oversampled region $K_{i,m}$ for some nonnegative integer $m$.

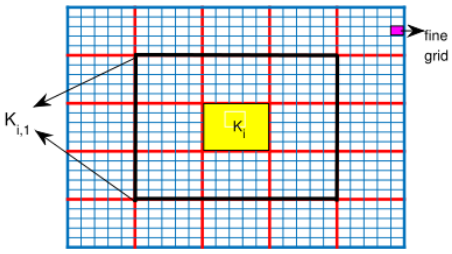


Figure 1.2: Oversampling region with $m = 1$. Reprinted with permission from "Computational multiscale methods for quasi-gas dynamic equations" by Boris Chetverushkin, Eric Chung, Yalchin Efendiev, Sai-Mang Pun and Zecheng Zhang. To be published in Journal of Computational Physics.

Let $V_0(K_i^+)$ be the subspace of $V(K_i^+)$ with zero trace on the boundary $\partial K_i^+$. For each eigenfunction $\phi_j^{(i)} \in V_{\text{aux}}$, we define the multiscale basis $\psi_{j,\text{ms}}^{(i)} \in V_0(K_i^+)$ to be the solution of the equation:

$$a(\psi_{j,\text{ms}}^{(i)}, v) + s\left(\pi(\psi_{j,\text{ms}}^{(i)}), \pi(v)\right) = s(\phi_j^{(i)}, v) \quad \text{for all } v \in V_0(K_i^+), \tag{1.9}$$

where $s(\cdot, \cdot)$ is defined as in (1.8).

Then, the multiscale space is defined as $V_{\text{ms}} := \text{span}\left\{\psi_{j,\text{ms}}^{(i)} : i = 1, \cdots, N,\ j = 1, \cdots, \ell_i\right\}$. By construction, we have $\dim(V_{\text{ms}}) = \dim(V_{\text{aux}})$. Finally, we seek a solution in the $V_{ms}$ space by standard finite element approximation.

### 1.3 Deep and reinforcement learning

Deep learning [39] was a branch of the machine learning and it was first studied in late 1980's. The success of the Alex Net [40] in early 2010's draw people's attention on the deep learning research. Since then, the deep learning became very successful in many research areas including computer vision, language processing and so on.

Most of the deep learning algorithms contain two steps: training and testing. In the training stage, people are given samples and then people train a network using the samples by gradient descent algorithm. In the testing stage, people fit the trained network with a testing sample and make the prediction, which is computational efficiently.

Because of the efficiency in the testing and accuracy of the algorithm, deep learning method has been used in the computational multiscale heterogeneous problems [41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 41].

In the computer vision applications, people have built the relation between the network layers and the image features. Also, people can prove the equivalence of the linear network and the principle component analysis (PCA) which can be taken a a simple dimension reduction technique [39]. This motivate us using the network as a method of model reduction. We hence proposed several deep learning based algorithms which solve the traditional multiscale problems in an efficient way.

### 1.4 Organization of the dissertation

In this work, I will solve and analyze problems with multiscale features. We apply GMsFEM and CEM in the spatial discretization and derive the mathematical analysis. I will present four works. The first two works are based on the quasi gas dynamic models (QGD). In the first work which is detailed in chapter 2, I describe the QGD model and illustrate the discretization techniques

of solving the equations. Stability and convergence analysis will also be shown.

The second project (see chapter 3) is based on the analysis of the QGD model which is presented in the first section. In this part, we consider solving a full wave equation using the paraxial approximation. I will not focus on the approximation theory; instead, I will study the resulting equations of the approximation. These equations are of QGD type and we can apply the analysis derived before on solving the problem; however, in practice, the dimension of the result system is high and it is not practical to solve the entire system. The idea to solve this issue is to apply the proper orthogonal decomposition (POD). I will present the details of the POD and give convergence analysis of the method.

The next two topics are more computational related which rely on the deep learning method. In the chapter 4 of the dissertation , I introduce an unsupervised deep clustering algorithm which is used to coarsen the uncertain space. We will compare the deep learning algorithm with the traditional algorithms and our algorithm can also be applied to solve other problems with the uncertainty.

The last chapter (5) is also deep learning based project; however, deep learning is used as a tool in the reinforcement learning setup. This work provides a deep reinforcement learning (RL) alternative to accelerating the Monte Carlo Markov Chain (MCMC). I will review the RL background and present the proposed method. To verify our method, we solve an inverse problem whose underlying model has multiscale property. The GMsFEM method comes in this project as we design the forward solver using the multiscale methods.

# 2. COMPUTATIONAL MULTISCALE METHODS FOR QUASI-GAS DYNAMIC EQUATIONS *

## 2.1 Introduction

The simulations of complex flows play an important role in many applications, such as porous media, aerodynamics, and so on. There are various model equations used for simulation purposes, which vary from kinetic to continuum models, such as the Navier-Stokes equations. There are several intermediate-scale models that are successfully used in the literature, which includes the quasi-gas dynamic (QGD) system of equations. The QGD model has shown to be effective for various applications. The QGD model equations are derived from kinetic equations under the assumption that the distribution function is similar to a locally Maxwellian representation. The QGD model has an advantage that it guarantees the smoothing of the solution at the free path distance. The QGD equations are extensively described in the literature [54, 55, 56, 57, 58, 59].

In the chapter, we consider a simplified QGD system involving second derivatives with respect to the time, in addition to spatial diffusion. In literature, this model has also been used to regularize purely parabolic equations by adding a hyperbolicity. This regularization has been employed in designing efficient time stepping algorithms [55, 57, 58].

We consider the QGD model in a multiscale environment. More precisely, we consider a simplified QGD model (see (2.1)) and introduce multiscale coefficients. These coefficients represent the media properties and spatially vary. The applications of these equations can be considered in porous media for compressible flows. The heterogeneities of the coefficients represent the media properties, which can have large variations. Our objective in this work is to make some first steps in understanding multiscale systems in these hyperbolic quasi-dynamic systems.

In the chapter, we would like to solve the QGD model equations on a coarse grid that is much larger compared to spatial heterogeneities. There are many methods for coarse-grid approximation

*Reprinted with permission from "Computational multiscale methods for quasi-gas dynamic equations" by Boris Chetverushkin, Eric Chung, Yalchin Efendiev, Sai-Mang Pun and Zecheng Zhang. To be published in Journal of Computational Physics by Elsevier.

as we have discu Our goal is to design an approach for hyperbolic quasi-dynamic systems.

For spatial discretization, we adopt the idea of CEM-GMsFEM presented in [19] and construct a specific multiscale space for approximating the solution. Starting with a well-designed auxiliary space, we construct multiscale basis functions (supported in some oversampling regions) which are minimizers of a class of constraint energy minimization problems. One of the theoretical benefits of the CEM-GMsFEM is that the convergence of the method can be shown to be independent of the contrast from the heterogeneities; and the error linearly decreases with respect to coarse mesh size if the oversampling parameter is appropriately chosen. Our analysis indicates that a moderate number of oversampling layers, depending logarithmically on the contrast, seems sufficient to archive accurate approximation. The present CEM-GMsFEM setting allows flexibly adding additional basis functions based on spectral properties of the differential operators. This enhances the accuracy of the method in the presence of high contrast in the media. It is shown that if enough basis functions are selected in each local patch, the convergence of the method can be shown independently of the contrast.

For temporal discretization, we use a central finite difference scheme to discretize the first and second order time derivatives in the equation. We show that the corresponding fully-discretized scheme is stable under a relaxed version of the CFL condition. In order to prove the stability and convergence of the full discretization, we first establish an inverse inequality in the multiscale finite element space. This result relies on the localized estimate between the global and local multiscale basis functions [19]. A complete convergence analysis is presented in this work. In particular, the error estimate of semi-discretization is shown in Theorem 2.4.3. For the complete analysis of the fully-discretized numerical scheme, the main result is summarized in Theorem 2.4.8. Throughout the part of analysis, we need proper regularity assumptions on the source term and initial conditions. Numerical results are provided to illustrate the efficiency of the proposed method and it confirms our theoretical findings.

The remainder of the chapter is organized as follows. We provide in Section 2.2 the background knowledge of the problem. Next, we introduce the multiscale method and the discretization in Sec-

tion 2.3. In Section 2.4, we provide the stability estimate of the method and prove the convergence of the proposed method. We present the numerical results in Section 2.5. Finally, we give concluding remarks in Section 2.6.

## 2.2 Preliminaries

Consider the quasi-gas dynamics (QGD) model in a polygonal domain $\Omega \subset \mathbb{R}^d$ ($d = 2, 3$):

$$\begin{aligned} u_t + \alpha u_{tt} - \nabla \cdot (\kappa \nabla u) &= f \qquad \text{in } (0, T] \times \Omega, \\ u|_{t=0} &= u_0 \qquad \text{in } \Omega, \\ u_t|_{t=0} &= v_0 \qquad \text{in } \Omega, \\ u &= 0 \qquad \text{on } \partial\Omega. \end{aligned} \tag{2.1}$$

Here, $u_t$ denotes the time derivative of the function $u$, $\alpha$ is a constant, $\kappa : \Omega \to \mathbb{R}$ is a time-independent high-contrast permeability field such that $0 < \gamma \leq \kappa(x) \leq \beta$ for almost every $x \in \Omega$, $f$ is a source term with suitable regularity, and $T > 0$ is the terminal time. Further, we assume that the initial conditions $u_0 \in H_0^1(\Omega)$ and $v_0 \in L^2(\Omega)$.

We clarify the notation used throughout the work. We write $(\cdot, \cdot)$ to denote the inner product in $L^2(D)$ and $\|\cdot\|$ for the corresponding norm. Let $H_0^1(\Omega)$ be the subspace of $H^1(\Omega)$ with functions having a vanishing trace and the corresponding dual space is denoted by $H^{-1}(\Omega)$. Moreover, we write $L^p(0, T; X)$ for the Bochner space with the norm

$$\|v\|_{L^p(0,T;X)} := \left( \int_0^T \|v\|_X^p \, dt \right)^{1/p} \qquad 1 \leq p < \infty,$$

where $X$ is a Banach space equipped with the norm $\|\cdot\|_X$.

Instead of the original PDE formulation, we consider the variational formulation corresponding to (2.1): Find $u \in L^2(0, T; H_0^1(\Omega))$ with $u_t \in L^2(0, T; L^2(\Omega))$ and $u_{tt} \in L^2(0, T; L^2(\Omega))$ such that

$$(u_t, v) + \alpha \, (u_{tt}, v) + a(u, v) = (f, v) \tag{2.2}$$

for all $v \in V := H_0^1(\Omega)$. Here, we define $a(u,v) := \int_\Omega \kappa \nabla u \cdot \nabla v \, dx$ for all $u, v \in V$. Employing Galerkin's method and the method of energy estimate, one can show the well-posedness of the variational formulation (2.2). See [60, Chapter 7.2] for more details.

In this research, we apply the constraint energy minimizing generalized multiscale finite element method (CEM-GMsFEM) to approximate the solution of the above QGD model. First, we introduce fine and coarse grids for the computational domain. Let $\mathcal{T}^H = \{K_i\}_{i=1}^N$ be a conforming partition of the domain $\Omega$ with mesh size $H > 0$ defined by

$$H := \max_{K \in \mathcal{T}^H} \Big( \max_{x,y \in K} |x - y| \Big).$$

We refer to this partition as the coarse grid. We denote the total number of coarse elements as $N \in \mathbb{N}^+$. Subordinate to the coarse grid, we define the fine grid partition $\mathcal{T}^h$ (with mesh size $h \ll H$) by refining each coarse element $K \in \mathcal{T}^H$ into a connected union of finer elements. We assume that the refinement above is performed such that $\mathcal{T}^h$ is also a conforming partition of the domain $\Omega$. Denote $N_c$ the number of interior coarse grid nodes of $\mathcal{T}^H$ and we denote $\{x_i\}_{i=1}^{N_c}$ the collection of interior coarse nodes in the coarse grid.

### 2.3 Multiscale method

We have reviewed the CEM-GMsFEM in the introduction. For the QGD equation, it has the similar formulation. We want to recall the localization property of the local multiscale basis.

**Remark 2.3.1.** *The local construction of multiscale basis function $\psi_{j,ms}^{(i)}$ supported in $K_i^+$ is motivated by the following global construction: Find $\psi_j^{(i)} \in V$ such that*

$$a(\psi_j^{(i)}, v) + s\left(\pi(\psi_j^{(i)}), \pi(v)\right) = s(\phi_j^{(i)}, v) \quad \textit{for all } v \in V. \tag{2.3}$$

*We then define $V_{glo} :=$ span $\Big\{\psi_j^{(i)} : i = 1, \cdots, N,\ j = 1, \cdots, \ell_i\Big\}$. It has been shown in [19] that the decomposition $V = V_{glo} \oplus$ Ker$(\pi)$ holds and this decomposition is orthogonal with respect to the energy bilinear form $a(\cdot,\cdot)$. We will use this property to prove the inverse inequality (Lemma*

*2.4.5) below.*

*Using the result of [19, Lemma 5], we have the error estimate of localization: For any multi-scale function* $v_{ms} = \sum_{i=1}^{N} \sum_{j=1}^{\ell_i} \alpha_j^{(i)} \psi_{j,ms}^{(i)} \in V_{ms}$*, there exists a function* $v_{glo} = \sum_{i=1}^{N} \sum_{j=1}^{\ell_i} \alpha_j^{(i)} \psi_j^{(i)} \in V_{glo}$ *such that*

$$\|v_{glo} - v_{ms}\|_a^2 \lesssim (m+1)^d E \sum_{i=1}^{N} \sum_{j=1}^{\ell_i} \left(\alpha_j^{(i)}\right)^2. \tag{2.4}$$

*Here,* $m$ *is the number of oversampling,* $E := 3(1+\Lambda^{-1})\left(1+(2(1+\Lambda^{-1/2}))\right)^{1-m}$ *is the factor of exponential decay, and* $\Lambda := \min_{1\le i\le N} \lambda_{\ell_i+1}^{(i)}$ *with* $\left\{\lambda_j^{(i)}\right\}$ *being obtained from* (1.7)*.*

### 2.3.1 The method and discretization

In this section, we discuss the discretizations of the equation (2.2). Let $u_{\text{ms}} \in V_{\text{ms}}$ be the multiscale approximation to the exact solution $u$. In particular, the function $u_{\text{ms}}$ solves

$$((u_{\text{ms}})_t, v) + \alpha\left((u_{\text{ms}})_{tt}, v\right) + a(u_{\text{ms}}, v) = (f, v) \quad \text{for all } v \in V_{\text{ms}}. \tag{2.5}$$

For time discretization, we first partition the temporal domain $(0, T)$ into equally $N_T$ pieces with time step size $\Delta t$. For any function $v = v(t)$, we use the following finite differences to approximate time derivatives appearing in the QGD model:

$$v_t \approx \frac{v(t_{n+1}) - v(t_{n-1})}{2\Delta t} =: D_t v^n \quad \text{and} \quad v_{tt} \approx \frac{v(t_{n+1}) - 2v(t_n) + v(t_{n-1})}{(\Delta t)^2} =: D_{tt} v^n.$$

The fully discretization of the equation (2.2) reads: Find $\mathbf{u}_H^T := (u_H^n)_{n=0}^{N_T}$ with $u_H^n \in V_{\text{ms}}$ such that for any $n = 1, \cdots, N_T - 1$,

$$(D_t u_H^n + \alpha D_{tt} u_H^n, v) + a(u_H^n, v) = (f^n, v) \quad \text{for all } v \in V_{\text{ms}}, \tag{2.6}$$

where $f^n := f(t_n)$.

## 2.4 Convergence analysis

In this section, we analyze the convergence of the multiscale method. Throughout the work, we denote $a \lesssim b$ if there is a generic constant $C > 0$ such that $a \leq Cb$. We write $a \lesssim_T b$ if there is a constant $C_T$ depending on $T$ such that $a \leq C_T b$. We denote $\|\cdot\| = \|\cdot\|_{L^2(\Omega)}$ and $\|\cdot\|_a := \sqrt{a(\cdot,\cdot)}$.

### 2.4.1 Semi-discretized scheme

We first consider the stability and error estimate in semi-discretization. The following results give a stability estimate for the scheme (2.5).

**Lemma 2.4.1.** *Let $u_{ms} \in V_{ms}$ be the solution of the equation* (2.5)*. Then,*

$$\alpha \|(u_{ms})_t(T)\|^2 + \|(u_{ms})(T)\|_a^2 \lesssim \alpha \|v_0\|^2 + \|u_0\|_a^2 + \|f\|^2_{L^2(0,T;L^2(\Omega))}. \tag{2.7}$$

*Proof.* Let $v = (u_{\text{ms}})_t$ in (2.5). We have

$$\|(u_{\text{ms}})_t\|^2 + \frac{1}{2}\frac{d}{dt}\left(\alpha \|(u_{\text{ms}})_t\|^2 + \|u_{\text{ms}}\|_a^2\right) = (f, (u_{\text{ms}})_t) \leq \|f\| \cdot \|(u_{\text{ms}})_t\|.$$

We remark that if $f \equiv 0$, the scheme is of energy conservation. Integrating over $(0, T)$ leads to

$$\begin{aligned} 2\int_0^T \|(u_{\text{ms}})_t\|^2 \, dt + \alpha \|(u_{\text{ms}})_t(T)\|^2 + \|u_{\text{ms}}(T)\|_a^2 &\leq \alpha \|v_0\|^2 + \|u_0\|_a^2 + 2\int_0^T \frac{1}{\sqrt{2}} \|f\| \cdot \sqrt{2} \|(u_{\text{ms}})_t\| \, dt \\ &\leq \alpha \|v_0\|^2 + \|u_0\|_a^2 + \frac{1}{2}\int_0^T \|f\|^2 \, dt + 2\int_0^T \|(u_{\text{ms}})_t\|^2 \, dt \end{aligned}$$

using Cauchy-Schwarz inequality. This completes the proof. □

To estimate the error bound for semi-discretization scheme, we introduce the definition of elliptic projection.

**Definition 2.4.2.** *For any function $v \in V$, we define the elliptic projection $\widehat{v} \in V_{ms}$ of the function $v$ such that*

$$a(v - \widehat{v}, w) = 0 \quad \textit{for all } w \in V_{ms}. \tag{2.8}$$

Next, we analyze the convergence of the proposed multiscale method. For any function $v \in V$, we define the *energy functional* $\mathcal{E} : V \to \mathbb{R}$ such that $\mathcal{E}(v) := \sqrt{\alpha}\,\|v_t\| + \|v\|_a$. It is not difficult to verify that

$$\mathcal{E}(v+w) = \sqrt{\alpha}\,\|v_t + w_t\| + \|v+w\|_a \le \sqrt{\alpha}\,(\|v_t\| + \|w_t\|) + \|v\|_a + \|w\|_a = \mathcal{E}(v) + \mathcal{E}(w)$$

for any $v, w \in V$. That is, the triangle inequality holds for the energy functional. Note that for any $v \in V$, we have

$$(\mathcal{E}(v))^2 = \left(\sqrt{\alpha}\,\|v_t\| + \|v\|_a\right)^2 \lesssim \alpha\,\|v_t\|^2 + \|v\|_a^2\,.$$

We have the following error estimate for the semi-discretization of the QGD model.

**Theorem 2.4.3.** *Let $u \in V$ be the solution to* (2.2) *and $u_{ms} \in V_{ms}$ be the multiscale solution to* (2.5)*. Assume that the number of oversampling layers $m = O(\log(\beta\gamma^{-1}H^{-1}))$ and $\{\chi_j^{ms}\}_{j=1}^{N_c}$ are bilinear partition of unity. Then, for any $t \in (0, T]$, the following error estimate holds*

$$\|u(t) - u_{ms}(t)\|_a \lesssim_T H\Lambda^{-1/2}, \tag{2.9}$$

*where $\Lambda = \min\limits_{1\le i\le N} \lambda_{\ell_i+1}^{(i)}$ and $\{\lambda_j^{(i)}\}$ are the eigenvalues obtained by solving* (1.7)*.*

*Proof.* Denote $\widehat{u}$ the elliptic projection of the exact solution $u$. We write

$$e := u - u_{\text{ms}} = \underbrace{u - \widehat{u}}_{=:\rho} + \underbrace{\widehat{u} - u_{\text{ms}}}_{=:\theta} = \rho + \theta.$$

Denote $\mathcal{F} := f - u_t - \alpha u_{tt}$. Note that the function $\widehat{u}$ satisfies the equation:

$$a(\widehat{u}, v) = (\mathcal{F}, v) \quad \text{for all } v \in V_{\text{ms}}.$$

Using the result of [19, Lemma 1], we obtain that

$$\|\rho\|_a = \|u - \widehat{u}\|_a \lesssim H\Lambda^{-1/2}\left\|\kappa^{-1/2}\mathcal{F}\right\| \quad \text{and} \quad \|\rho_t\| = \|(u-\widehat{u})_t\| \lesssim H^2\Lambda^{-1}\left\|\kappa^{-1/2}\mathcal{F}_t\right\|.$$

Therefore, we have

$$\mathcal{E}(\rho) \lesssim \sqrt{\alpha} H^2 \Lambda^{-1} \left\| \kappa^{-1/2} \mathcal{F}_t \right\| + H \Lambda^{-1/2} \left\| \kappa^{-1/2} \mathcal{F} \right\| \lesssim H \Lambda^{-1/2}.$$

Next, we analyze the term $\mathcal{E}(\theta)$. Subtracting (2.5) from (2.2), we obtain

$$(e_t, v) + \alpha \left(e_{tt}, v\right) + a(e, v) = 0 \quad \text{for all } v \in V_{\text{ms}}.$$

Note that, by the property of elliptic projection, we have $a(\rho, v) = 0$ for all $v \in V_{\text{ms}}$. That is, we have

$$(\theta_t, v) + \alpha \left(\theta_{tt}, v\right) + a(\theta, v) = ((\widehat{u} - u)_t + \alpha(\widehat{u} - u)_{tt}, v)$$

for all $v \in V_{\text{ms}}$. Denote $\mathcal{G} := (\widehat{u} - u)_t + \alpha(\widehat{u} - u)_{tt}$. Let $v = \theta_t \in V_{\text{ms}}$ and use the same technique for proving the stability result (2.7), one can show that

$$\left(\mathcal{E}(\theta)\right)^2 \lesssim \alpha \left\|\theta_t(0)\right\|^2 + \left\|\theta(0)\right\|_a^2 + \left\|\mathcal{G}\right\|^2_{L^2(0,T;L^2(\Omega))}.$$

Note that $\theta_t(0)$ and $\theta(0)$ are given by the initial conditions of quasi gas-dynamics equation. If we choose $u_{\text{ms}}(0)$ be such that

$$a(u_{\text{ms}}(0), v) = a(u_0, v) \quad \text{for all } v \in V_{\text{ms}},$$

then $\theta_t(0) = \theta(0) = 0$ because of the property of elliptic projection. Therefore, we have

$$\mathcal{E}(\theta) \lesssim \left\|\mathcal{G}\right\|_{L^2(0,T;L^2(\Omega))} \lesssim \left\|\rho_t\right\|_{L^2(0,T;L^2(\Omega))} + \alpha \left\|\rho_{tt}\right\|_{L^2(0,T;L^2(\Omega))} \lesssim_T H^2 \Lambda^{-1}.$$

To conclude, we show that

$$\mathcal{E}(u - u_{\text{ms}}) \leq \mathcal{E}(\rho) + \mathcal{E}(\theta) \lesssim_T H \Lambda^{-1/2}. \tag{2.10}$$

This completes the proof. □

### 2.4.2 Fully discretization

In this section, we analyze the method in fully discretization. First, we define $\sigma_{\text{aux}} := \max_{1\le i\le N}\left(\max_{1\le j\le \ell_i}\lambda_j^{(i)}\right)$. We observe that the inverse inequality (in the multiscale space) holds. To prove the inverse inequality in $V_{\text{ms}}$, we first prove the following lemma.

**Lemma 2.4.4.** *For any $v_{ms}=\sum_{i=1}^{N}\sum_{j=1}^{\ell_i}\alpha_j^{(i)}\psi_{j,ms}^{(i)}\in V_{ms}$, the following estimation holds*

$$\sum_{i=1}^{N}\sum_{j=1}^{\ell_i}(\alpha_j^{(i)})^2\le(1+D)\left\|v_{ms}\right\|_s^2, \tag{2.11}$$

*where $D$ is a generic constant depending on the value of $\sigma_{aux}$.*

*Proof.* Let $v_{\text{ms}}=\sum_{i=1}^{N}\sum_{j=1}^{\ell_i}\alpha_j^{(i)}\psi_{j,\text{ms}}^{(i)}\in V_{\text{ms}}$. By the variational formulation (1.9), for any $\phi_k^{(l)}\in V_{\text{aux}}$, we have

$$s(\pi v_{\text{ms}},\phi_k^{(l)})=\sum_{i=1}^{N}\sum_{j=1}^{\ell_i}\alpha_j^{(i)}s(\pi\psi_{j,\text{ms}}^{(i)},\phi_k^{(l)})=\sum_{i=1}^{N}\sum_{j=1}^{\ell_i}\alpha_j^{(i)}\left(s(\pi\psi_{j,\text{ms}}^{(i)},\pi\psi_{k,\text{ms}}^{(l)})+a(\psi_{j,\text{ms}}^{(i)},\psi_{k,\text{ms}}^{(l)})\right).$$

Denote $b_{lk}=s(\pi v_{\text{ms}},\phi_k^{(l)})$ and $\mathbf{b}=(b_{lk})$, we have

$$\|\mathbf{c}\|_2\le\|A^{-1}\|_2\cdot\|\mathbf{b}\|_2,$$

where $A\in\mathbb{R}^{p\times p}$ is the matrix representation of the bilinear form

$$s(\pi\psi_{j,\text{ms}}^{(i)},\pi\psi_{k,\text{ms}}^{(l)})+a(\psi_{j,\text{ms}}^{(i)},\psi_{k,\text{ms}}^{(l)})$$

with $p=\sum_{i=1}^{N}\ell_i$ and $\mathbf{c}=\left(\alpha_j^{(i)}\right)\in\mathbb{R}^p$. We then estimate the largest eigenvalue of $A^{-1}$. Define an auxiliary function $\phi:=\sum_{i=1}^{N}\sum_{j=1}^{\ell_i}\alpha_j^{(i)}\phi_j^{(i)}\in V_{\text{aux}}$ and $\psi_{\text{ms}}\in V_{\text{ms}}$ to be the solution of the

following equation:

$$a(\psi_{\text{ms}}, \omega) + s(\pi\psi_{\text{ms}}, \pi\omega) = s(\phi, \pi\omega) \quad \text{for all } \omega \in V_{\text{ms}}. \tag{2.12}$$

On the other hand, by [19, Lemma 2], there is a function $z \in V$ such that

$$\pi z = \phi \quad \text{and} \quad \|z\|_a^2 \leq D\|\phi\|_s^2.$$

Here, $D$ is a generic constant depending on the value of $\sigma_{\text{aux}}$ (cf. [19, Lemma 2]). Taking $\omega = z$ in (2.12) and using the fact that $s(\phi, \phi) = \|\mathbf{c}\|_2^2$, we have

$$\begin{aligned}\|\mathbf{c}\|_2^2 &= a(\psi_{\text{ms}}, z) + s(\pi\psi_{\text{ms}}, \phi) \leq \|\psi_{\text{ms}}\|_a \|z\|_a + \|\pi\psi_{\text{ms}}\|_s \|\phi\|_s \\ &\leq (1+D)^{\frac{1}{2}} \|\phi\|_s \left(\|\psi_{\text{ms}}\|_a^2 + \|\pi\psi_{\text{ms}}\|_s^2\right)^{\frac{1}{2}}.\end{aligned}$$

Denote $\Psi \in \mathbb{R}^p$ the vector representation of $\psi_{\text{ms}}$ and $(\cdot,\cdot)_2$ the $\ell_2$ Euclidean inner product on $\mathbb{R}^p$. This implies that

$$\frac{\|\mathbf{c}\|_2}{(A\Psi, \Psi)_2} \leq (1+D)^{\frac{1}{2}}.$$

This implies that $\|A^{-1}\|_2 \leq (1+D)^{\frac{1}{2}}$. It follows that $\|\mathbf{c}\|_2^2 \leq (1+D)\|\mathbf{b}\|_2^2 \leq (1+D)\|v_{\text{ms}}\|_s^2$. □

**Lemma 2.4.5** (Inverse Inequality)**.** *Assume that $\{\chi_j^{ms}\}_{j=1}^{N_c}$ is a set of bilinear partition of unity. For any $v_{ms} \in V_{ms}$, there is a constant $C_{inv} > 0$ such that*

$$\|\nabla v_{ms}\|_a \leq C_{inv} H^{-1} \beta^{1/2} \|v_{ms}\|. \tag{2.13}$$

*Proof.* Let $v \in V_{\text{glo}}$. Applying the orthogonality of $V_{\text{glo}}$, we get

$$\|v\|_a^2 = a(v,v) = a(v, \pi v) \leq \|v\|_a \|\pi v\|_a \leq \|v\|_a \sigma_{\text{aux}}^{1/2} \|\pi v\|_s,$$

which implies that $\|v\|_a \leq \sigma_{\text{aux}}^{1/2}\|\pi v\|_s$.

Next, for any $v_{\text{ms}} = \sum_{i=1}^{N}\sum_{j=1}^{\ell_i}\alpha_j^{(i)}\psi_{j,\text{ms}}^{(i)} \in V_{\text{ms}}$, let $v = \sum_{i=1}^{N}\sum_{j=1}^{\ell_i}\alpha_j^{(i)}\psi_j^{(i)} \in V_{\text{glo}}$. We claim that $\|\pi v\|_s \leq \sum_{i=1}^{N}\sum_{j=1}^{\ell_i}(\alpha_j^{(i)})^2$. Notice that by (2.3), we have

$$\begin{aligned}\|\pi v\|_s^2 = s(\pi v, \pi v) &= \sum_{i=1}^{N}\sum_{j=1}^{\ell_i}\alpha_j^{(i)} s(\pi\psi_j^{(i)}, \pi v) = \sum_{i=1}^{N}\sum_{j=1}^{\ell_i}\alpha_j^{(i)}\left(s(\phi_j^{(i)}, \pi v) - a(\psi_j^{(i)}, v)\right)\\ &= s(\phi, \pi v) - a(v, v) = s(\phi, \pi v) - \|v\|_a^2\end{aligned}$$

with $\phi := \sum_{i=1}^{N}\sum_{j=1}^{\ell_i}\alpha_j^{(i)}\phi_j^{(i)}$. This implies that

$$\|\pi v\|_s^2 \leq s(\phi, \pi v) \leq \|\phi\|_s \|\pi v\|_s \implies \|\pi v\|_s^2 \leq \|\phi\|_s^2 = \sum_{i=1}^{N}\sum_{j=1}^{\ell_i}\left(\alpha_j^{(i)}\right)^2$$

using the orthogonality of the auxiliary basis functions. By the inequalities (2.4) and (2.11), we have

$$\begin{aligned}\|v_{\text{ms}}\|_a^2 &\leq \|(v - v_{\text{ms}})\|_a^2 + \|v\|_a^2\\ &\lesssim (m+1)^d E\sum_{i=1}^{N}\sum_{j=1}^{\ell_i}\left(\alpha_j^{(i)}\right)^2 + \sigma_{\text{aux}}\|\pi v\|_s^2\\ &\lesssim \left((m+1)^d E + \sigma_{\text{aux}}\right)\sum_{i=1}^{N}\sum_{j=1}^{\ell_i}\left(\alpha_j^{(i)}\right)^2\\ &\lesssim \left((m+1)^d E + \sigma_{\text{aux}}\right)(1+D)\|v_{\text{ms}}\|_s^2.\end{aligned}$$

Using the definition of $s$-norm, this gives that $\|v_{\text{ms}}\|_a \leq C_{\text{inv}} H^{-1}\beta^{1/2}\|v_{\text{ms}}\|$ holds for any $v_{\text{ms}} \in V_{\text{ms}}$ with $C_{\text{inv}}^2 = (1+D)\left((m+1)^d E + \sigma_{\text{aux}}\right)$.

□

Recall that $\mathbf{u}_H^T := (u_H^n)_{n=0}^{N_T}$ with $u_H^n \in V_{\text{ms}}$ is the solution to (2.6). The following result gives the stability estimate of the fully discretization.

**Lemma 2.4.6** (Stability of the method)**.** *Assume that the CFL condition*

$$\alpha - \frac{1}{2} C_{inv}^2 \beta H^{-2} (\Delta t)^2 \geq \delta \tag{2.14}$$

*holds for some constant $\delta > 0$. Then, the fully discretization method* (2.6) *is stable; that is,*

$$\alpha \left\| \frac{u_H^n - u_H^{n-1}}{\Delta t} \right\| + \|u_H^n\|_a \lesssim \left( \Delta t \sum_{k=1}^{n} \left\| f^k \right\| + \alpha \left\| \frac{u_H^1 - u_H^0}{\Delta t} \right\| + \left\| u_H^1 \right\|_a + \left\| u_H^0 \right\|_a \right). \tag{2.15}$$

*Proof.* Let $v = u_H^{n+1} - u_H^{n-1}$ in (2.6). We have

$$\frac{1}{2\Delta t} \left\| u_H^{n+1} - u_H^{n-1} \right\|^2 + \frac{\alpha}{(\Delta t)^2} \left( u_H^{n+1} - u_H^n - (u_H^n - u_H^{n-1}), u_H^{n+1} - u_H^n + u_H^n - u_H^{n-1} \right)$$
$$+ a(u_H^n, u_H^{n+1} - u_H^{n-1}) = \Delta t \left( f^n, \frac{u_H^{n+1} - u_H^{n-1}}{\Delta t} \right).$$

Define $\mathcal{E}_{n,H} := \frac{1}{2} \left( \alpha \left\| \frac{u_H^n - u_H^{n-1}}{\Delta t} \right\|^2 + a(u_H^{n-1}, u_H^n) \right)$. It implies that

$$\alpha \left( \left\| \frac{u_H^{n+1} - u_H^n}{\Delta t} \right\|^2 - \left\| \frac{u_H^n - u_H^{n-1}}{\Delta t} \right\|^2 \right) + a(u_H^n, u_H^{n+1}) - a(u_H^{n-1}, u_H^n) \leq (f^n, u_H^{n+1} - u_H^{n-1})$$

$$\implies \mathcal{E}_{n+1,H} \leq \mathcal{E}_{n,H} + \frac{1}{2} (f^n, u_H^{n+1} - u_H^{n-1}).$$

Note that

$$\begin{aligned}
\mathcal{E}_{n,H} &= \frac{1}{2}\left(\alpha\left\|\frac{u_H^n-u_H^{n-1}}{\Delta t}\right\|^2 + a(u_H^n,u_H^{n-1})\right)\\
&= \frac{\alpha}{2}\left\|\frac{u_H^n-u_H^{n-1}}{\Delta t}\right\|^2 + \frac{1}{4}a(u_H^n,u_H^n) + \frac{1}{4}a(u_H^{n-1},u_H^{n-1}) - \frac{1}{4}a(u_H^n-u_H^{n-1},u_H^n-u_H^{n-1})\\
&\geq \frac{\alpha}{2}\left\|\frac{u_H^n-u_H^{n-1}}{\Delta t}\right\|^2 + \frac{1}{4}a(u_H^n,u_H^n) + \frac{1}{4}a(u_H^{n-1},u_H^{n-1}) - \frac{1}{4}\left\|(u_H^n-u_H^{n-1})\right\|_a^2\\
&\geq \frac{\alpha}{2}\left\|\frac{u_H^n-u_H^{n-1}}{\Delta t}\right\|^2 + \frac{1}{4}a(u_H^n,u_H^n) + \frac{1}{4}a(u_H^{n-1},u_H^{n-1}) - \frac{1}{4}C_{\text{inv}}^2H^{-2}\beta(\Delta t)^2\left\|\frac{u_H^n-u_H^{n-1}}{\Delta t}\right\|^2\\
&= \frac{1}{2}\left(\alpha - \frac{1}{2}C_{\text{inv}}^2H^{-2}\beta(\Delta t)^2\right)\left\|\frac{u_H^n-u_H^{n-1}}{\Delta t}\right\|^2 + \frac{1}{4}\left(\|u_H^n\|_a^2 + \left\|u_H^{n-1}\right\|_a^2\right).
\end{aligned}$$

Then, we have

$$\begin{aligned}
\mathcal{E}_{n+1,H}-\mathcal{E}_{n,H} &\leq \frac{1}{2}(f^n,u_H^{n+1}-u_H^{n-1}) \leq \frac{1}{2}\Delta t\left\|f^n\right\|\left(\left\|\frac{u_H^{n+1}-u_H^n}{\Delta t}\right\| + \left\|\frac{u_H^n-u_H^{n-1}}{\Delta t}\right\|\right)\\
&\leq \frac{1}{2}\Delta t\left\|f^n\right\|\cdot\sqrt{\frac{2}{\delta}}\left(\sqrt{\mathcal{E}_{n+1,H}}+\sqrt{\mathcal{E}_{n,H}}\right),\\
\sqrt{\mathcal{E}_{n+1,H}}-\sqrt{\mathcal{E}_{n,H}} \leq \frac{1}{\sqrt{2\delta}}\Delta t\left\|f^n\right\| &\implies \sqrt{\mathcal{E}_{n,H}} \leq \sqrt{\mathcal{E}_{0,H}} + \frac{\Delta t}{\sqrt{2\delta}}\sum_{k=1}^{n}\left\|f^k\right\|.
\end{aligned}$$

This implies that

$$\alpha\left\|\frac{u_H^n-u_H^{n-1}}{\Delta t}\right\| + \|u_H^n\|_a \lesssim \left(\Delta t\sum_{k=1}^{n}\left\|f^k\right\| + \alpha\left\|\frac{u_H^1-u_H^0}{\Delta t}\right\| + \left\|u_H^1\right\|_a + \left\|u_H^0\right\|_a\right).$$

This completes the proof. □

**Remark 2.4.7.** *According to the CFL condition* (2.15)*, the fully discretization* (2.6) *is stable if and only if the temporal step size* $\Delta t > 0$ *satisfies the following inequality: for some constant* $\delta \in (0,\alpha)$*,*

$$\Delta t \leq \frac{H\sqrt{2(\alpha-\delta)}}{\beta^{1/2}C_{inv}}.$$

*The above condition for the temporal step size is relaxed since the coarse mesh size* $H$ *is much larger than that of the fine mesh. In other words, with the proposed multiscale method, one may*

*use a larger temporal step size (comparing to that of any fine-scale methods) in simulations while maintaining the stability of the explicit discretization; however, this is still a restrictive time step and we plan to study implicit or implicit-explicit schemes in future.*

Recall that $u \in V$ is the solution of (2.2). The total error between $\mathbf{u} := (u(t_n))_{n=0}^{N_T}$ and $\mathbf{u}_H^T$ can be split into two parts: the spatial discretization error $u(t_n) - u_{\text{ms}}(t_n)$ and the time discretization error $u_{\text{ms}}(t_n) - u_H^n$. Here, $u_{\text{ms}} \in V_{\text{ms}}$ is the solution of (2.5). Using the result of (2.10), we have

$$\|u(t_n) - u_{\text{ms}}(t_n)\|_a \lesssim_T H\Lambda^{-1/2}.$$

Next, we estimate the time discretization error. Let $\widetilde{e}_n := u_{\text{ms}}^n - u_H^n$ with $u_{\text{ms}}^n := u_{\text{ms}}(t_n)$. Subtracting (2.5) from (2.6), we obtain

$$\left(\frac{\widetilde{e}_{n+1} - \widetilde{e}_{n-1}}{2\Delta t}, v\right) + \alpha\left(\frac{\widetilde{e}_{n+1} - 2\widetilde{e}_n + \widetilde{e}_{n-1}}{(\Delta t)^2}, v\right) + a(\widetilde{e}_n, v) = (\mathcal{H}^n, v) \quad \text{for all } v \in V_{\text{ms}},$$

where

$$\mathcal{H}^n := (u_{\text{ms}})_t + \alpha(u_{\text{ms}})_{tt} - \frac{u_{\text{ms}}^{n+1} - u_{\text{ms}}^{n-1}}{2\Delta t} - \alpha\frac{u_{\text{ms}}^{n+1} - 2u_{\text{ms}}^n + u_{\text{ms}}^{n-1}}{(\Delta t)^2}.$$

Using the result of (2.15), one can obtain

$$\begin{aligned}\alpha\left\|\frac{\widetilde{e}_{n+1} - \widetilde{e}_n}{\Delta t}\right\| + \|\widetilde{e}_n\|_a \lesssim \alpha\left\|\frac{\widetilde{e}_1 - \widetilde{e}_0}{\Delta t}\right\| + \|\widetilde{e}_1\|_a + \Delta t\sum_{k=1}^{n}\Bigg\{\left\|(u_{\text{ms}})_t - \frac{u_{\text{ms}}^{k+1} - u_{\text{ms}}^{k-1}}{2\Delta t}\right\| \\ +\alpha\left\|(u_{\text{ms}})_{tt} - \frac{u_{\text{ms}}^{k+1} - 2u_{\text{ms}}^k + u_{\text{ms}}^{k-1}}{(\Delta t)^2}\right\|\Bigg\}.\end{aligned} \tag{2.16}$$

Under the assumption of some additional regularity and appropriate initial conditions, the right-hand side of (2.16) scales like $H + (\Delta t)^2$.

Finally, we have the error estimate for the fully discretization scheme.

**Theorem 2.4.8.** *Assume that $u$, $u_{ms}$, and $f$ are smooth enough with respect to the variable $t$. Let*

*$\widetilde{u}_H(t)$ be the piecewise linear function that interpolates $\mathbf{u}_H^T$ in time. Then*

$$\|u - \widetilde{u}_{H,ms}\|_{L^2(0,T;a)} \lesssim_T H + (\Delta t)^2, \quad \textit{where } \|\cdot\|_{L^2(0,T;a)} := \left(\int_0^T \|\cdot\|_a^2 \, dt\right)^{1/2}.$$

### 2.5 Numerical experiments

In this section, we present several numerical experiments to demonstrate the efficiency of the proposed method. We set the computational domain $\Omega = (0,1)^2$. We partition the domain into $100 \times 100$ rectangular elements and refer it as a fine mesh $\mathcal{T}^h$ with mesh size $h = \sqrt{2}/100$.

In the example below, we solve the QGD model (2.2) with $f(x_1, x_2) = \sin(\pi x_1)\sin(\pi x_2)$. Terminal time $T = 4.0$ is set and step size $\Delta t$ is chosen subjected to the CFL condition. The initial conditions are $u_0 = v_0 = 0$. Practical experiments showed that $\Delta t = 10^{-5}$ provides a sufficient and rather sharp choice for the stability with small value of $\alpha$ and high value of contrast. To implement the scheme, we set $u_H^0 = u_H^1 = 0$. We use the permeability field $\kappa$ with contrast $10^3$ (see Figure 2.1).

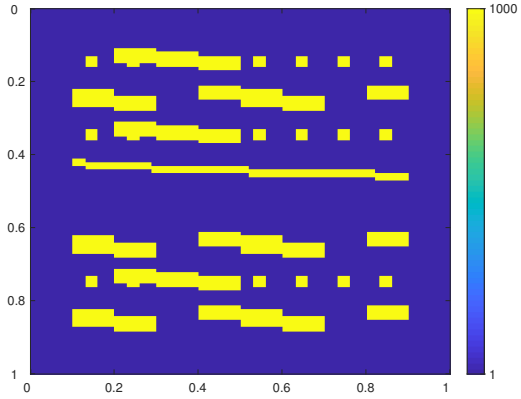


Figure 2.1: Permeability field $\kappa$ with contrast values $10^3$. Reprinted with permission from "Computational multiscale methods for quasi-gas dynamic equations" by Boris Chetverushkin, Eric Chung, Yalchin Efendiev, Sai-Mang Pun and Zecheng Zhang. To be published in Journal of Computational Physics by Elsevier.

We solve the fully discretization (2.6) and seek $u_H^n \in V_{\text{ms}}$. We define the corresponding relative $L^2$ and energy errors between the multiscale solution and the exact solution (up to a fine-scale) as follows:

$$e_{L^2} := \frac{\|u(T) - u_H^{N_T}\|_s}{\|u(T)\|_s} \quad \text{and} \quad e_a := \frac{\|u(T) - u_H^{N_T}\|_a}{\|u(T)\|_a},$$

where $\|\cdot\|_a = \sqrt{a(\cdot,\cdot)}$ and $\|\cdot\|_s = \sqrt{s(\cdot,\cdot)}$.

We present the convergence history in the energy and $L^2$ norms when the coarse mesh size is $H = \sqrt{2}/5$, $\sqrt{2}/10$, and $\sqrt{2}/20$, respectively. The number of oversampling layers $m$ is set to be $3, 4$, and $6$ in all experiments. The number of multiscale basis functions is $\ell_i = 3$ in each local coarse element $K_i$. We test with different values of $\alpha = 0.001, 0.005, 0.1, 0.5, 1, 5$, and $10$. The results of $e_{L^2}$ and $e_a$ are shown in Tables 2.1 and 2.2, respectively. A first-order convergence in energy norm and second-order convergence in $L^2$ norm have been observed as expected; see Figure 2.2 for illustration.

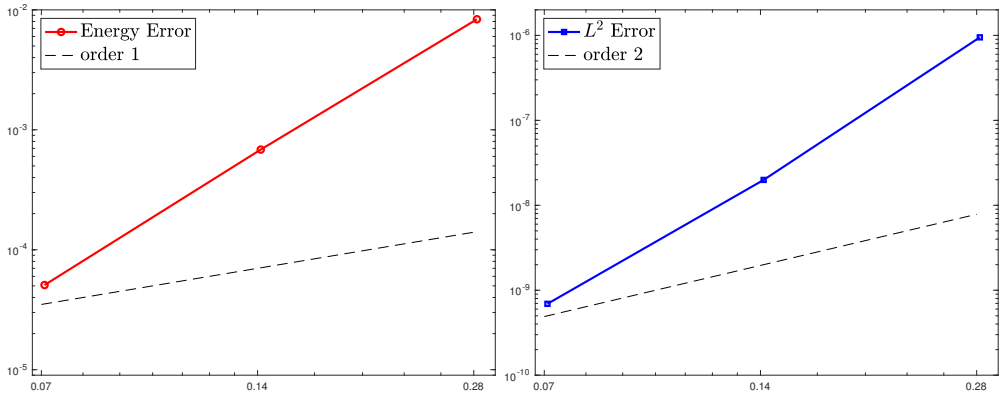


Figure 2.2: Convergence history in $e_a$ (left) and $e_{L^2}$ (right) with $\alpha = 0.1$. Reprinted with permission from "Computational multiscale methods for quasi-gas dynamic equations" by Boris Chetverushkin, Eric Chung, Yalchin Efendiev, Sai-Mang Pun and Zecheng Zhang. To be published in Journal of Computational Physics by Elsevier.

| $H$ | $m$ | $\alpha = 10$ | $\alpha = 5$ | $\alpha = 1$ | $\alpha = 0.5$ | $\alpha = 0.1$ | $\alpha = 0.05$ | $\alpha = 0.01$ |
|---|---|---|---|---|---|---|---|---|
| $\sqrt{2}/5$ | 3 | 2.07e-03 | 4.85e-05 | 2.09e-05 | 5.40e-06 | 9.49e-07 | 9.49e-07 | 9.49e-07 |
| $\sqrt{2}/10$ | 4 | 9.39e-06 | 2.12e-07 | 1.60e-07 | 4.17e-08 | 1.99e-08 | 1.99e-08 | 1.99e-08 |
| $\sqrt{2}/20$ | 6 | 1.95e-07 | 5.38e-09 | 2.45e-09 | 6.92e-10 | 6.92e-10 | 6.92e-10 | 6.92e-10 |

Table 2.1: Convergence in relative $L^2$ norm for different $\alpha$. Reprinted with permission from "Computational multiscale methods for quasi-gas dynamic equations" by Boris Chetverushkin, Eric Chung, Yalchin Efendiev, Sai-Mang Pun and Zecheng Zhang. To be published in Journal of Computational Physics by Elsevier.

| $H$ | $m$ | $\alpha = 10$ | $\alpha = 5$ | $\alpha = 1$ | $\alpha = 0.5$ | $\alpha = 0.1$ | $\alpha = 0.05$ | $\alpha = 0.01$ |
|---|---|---|---|---|---|---|---|---|
| $\sqrt{2}/5$ | 3 | 2.08e-02 | 8.76e-03 | 8.54e-03 | 8.55e-03 | 8.53e-03 | 8.53e-03 | 8.53e-03 |
| $\sqrt{2}/10$ | 4 | 1.75e-03 | 6.28e-04 | 6.74e-04 | 6.86e-04 | 6.84e-04 | 6.84e-04 | 6.84e-04 |
| $\sqrt{2}/20$ | 6 | 1.89e-04 | 5.19e-05 | 5.11e-05 | 5.09e-05 | 5.08e-05 | 5.08e-05 | 5.08e-05 |

Table 2.2: Convergence in relative energy norm for different $\alpha$. Reprinted with permission from "Computational multiscale methods for quasi-gas dynamic equations" by Boris Chetverushkin, Eric Chung, Yalchin Efendiev, Sai-Mang Pun and Zecheng Zhang. To be published in Journal of Computational Physics by Elsevier.

We also test our algorithm on a problem with time dependent source. In this example, we set $f(x_1, x_2, t) = \sin(\pi t)\sin(\pi x_1)\sin(\pi x_2)$. All the other settings are same with the first example. The convergence in $L_2$ and energy norm are presented in Tables 2.3 and 2.4. Convergence rate in both norms are observed, please check Figure (2.3) for the illustration.

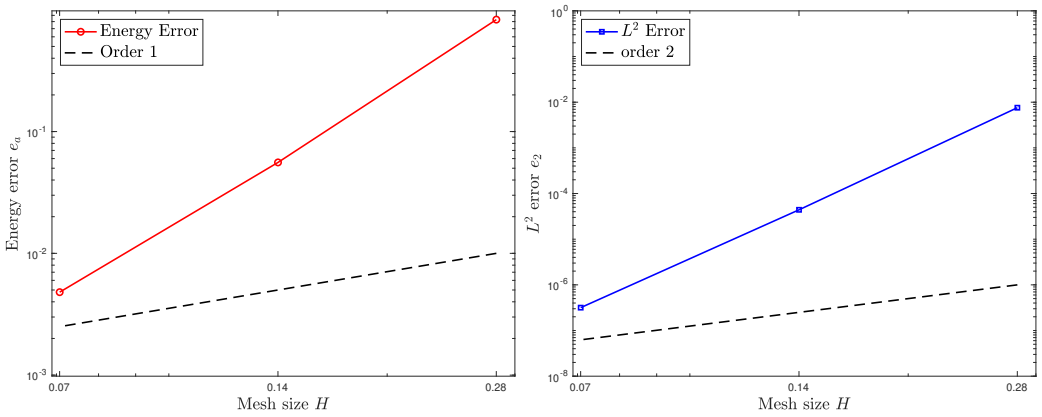


Figure 2.3: Convergence history (time dependent source) in $e_a$ (left) and $e_{L^2}$ (right) with $\alpha = 0.1$. Reprinted with permission from "Computational multiscale methods for quasi-gas dynamic equations" by Boris Chetverushkin, Eric Chung, Yalchin Efendiev, Sai-Mang Pun and Zecheng Zhang. To be published in Journal of Computational Physics by Elsevier.

| $H$ | $m$ | $\alpha = 10$ | $\alpha = 5$ | $\alpha = 1$ | $\alpha = 0.5$ | $\alpha = 0.1$ | $\alpha = 0.05$ | $\alpha = 0.01$ |
|---|---|---|---|---|---|---|---|---|
| $\sqrt{2}/5$ | 3 | 3.00e-01 | 7.83e-01 | 1.89e-01 | 7.41e-03 | 7.57e-03 | 7.11e-03 | 6.76e-03 |
| $\sqrt{2}/10$ | 4 | 1.07e-03 | 3.41e-03 | 8.82e-04 | 4.80e-05 | 4.40e-05 | 4.16e-05 | 3.98e-05 |
| $\sqrt{2}/20$ | 6 | 1.03e-05 | 2.70e-05 | 6.52e-06 | 3.46e-07 | 3.17e-07 | 3.00e-07 | 2.87e-07 |

Table 2.3: Convergence (time dependent source) in relative $L_2$ norm for different $\alpha$. Reprinted with permission from "Computational multiscale methods for quasi-gas dynamic equations" by Boris Chetverushkin, Eric Chung, Yalchin Efendiev, Sai-Mang Pun and Zecheng Zhang. To be published in Journal of Computational Physics by Elsevier.

| $H$ | $m$ | $\alpha = 10$ | $\alpha = 5$ | $\alpha = 1$ | $\alpha = 0.5$ | $\alpha = 0.1$ | $\alpha = 0.05$ | $\alpha = 0.01$ |
|---|---|---|---|---|---|---|---|---|
| $\sqrt{2}/5$ | 3 | 2.0198 | 1.5981 | 1.0128 | 0.8306 | 0.8304 | 0.8299 | 0.8295 |
| $\sqrt{2}/10$ | 4 | 0.0656 | 0.0589 | 0.0565 | 0.0557 | 0.0558 | 0.0558 | 0.0558 |
| $\sqrt{2}/20$ | 6 | 0.0072 | 0.0048 | 0.0048 | 0.0048 | 0.0048 | 0.0048 | 0.0048 |

Table 2.4: Convergence (time dependent source) in relative energy norm for different $\alpha$. Reprinted with permission from "Computational multiscale methods for quasi-gas dynamic equations" by Boris Chetverushkin, Eric Chung, Yalchin Efendiev, Sai-Mang Pun and Zecheng Zhang. To be published in Journal of Computational Physics by Elsevier.

## 2.6 Concluding remarks

In this work, we have proposed a novel computational multiscale method based on the idea of constraint energy minimization for solving the problem of quasi-gas-dynamics. The spatial discretization is based on CEM-GMsFEM which provides a framework to systematically construct multiscale basis functions for approximating the solution of the model. The multiscale basis functions with locally minimal energy are constructed by employing the techniques of oversampling, which leads to an improved accuracy in the simulations. Combined with the central difference scheme for the time discretization, we have shown that the fully discrete method is stable under a relaxed version of CFL condition and has optimal convergence rates despite the heterogeneities of the media. Numerical results have been presented to illustrate the performance of the proposed method.

# 3. COMPUTATIONAL MULTISCALE METHODS FOR PARAXIAL WAVE APPROXIMATIONS IN HETEROGENEOUS MEDIA

## 3.1 Introduction

Paraxial wave approximation have been used to approximate wave equations with a preferred direction [61, 62]. To be more specific, we study the approximation of the full wave equation:

$$\rho \partial_{tt} u - \nabla \cdot (\mu \nabla u) = 0. \tag{3.1}$$

Here, $\nabla$ denotes the gradient operator in $\mathbb{R}^3$; $\rho$ and $\mu$ are positive functions in $\mathbb{R}^3$. We then consider the paraxial approximation of (3.1) in $D \times \Omega \subseteq \mathbb{R}^3$: for $(z, x) \in D \times \Omega$ with the boundary $\Gamma := \partial(D \times \Omega)$, find $v = \sigma^{1/2} u$ such that

$$\begin{aligned} c^{-1}\partial_{tt} v + \partial_t(\partial_z v) - \frac{1}{2}\nabla_x \cdot (c \nabla_x v) = 0 \qquad & \text{in } D \times \Omega \times (0, T], \\ v(z, x, 0) = v_0(z, x) \qquad & \text{in } D \times \Omega, \\ \partial_t v(z, x, 0) = v_1(z, x) \qquad & \text{in } D \times \Omega, \\ v(z, x, t) = g(z, x, t) \qquad & \text{in } \Gamma \times (0, T]. \end{aligned} \tag{3.2}$$

Here, $\nabla_x$ denotes the gradient operator defined in the bounded domain $\Omega \subset \mathbb{R}^2$; $D = [d_1, d_2] \subset \mathbb{R}$ is a bounded domain; $v_0$ and $v_1$ are initial conditions; $g$ is boundary condition and $T > 0$ is a given terminal time.

Wave equations of this type have a propagation direction $z$ which plays a role of time [63]. Many real-world problems can be solved by the paraxial approximation; in particular in the areas of geology [64], under water acoustics [65, 66, 67, 68] and optics [69, 70, 71]. There are two benefits for using the paraxial approximation instead of the full wave equation [61, 62]: (i) it is easier to be realized and computationally more efficient; and (ii) it makes it possible to use the approximation as an evolution equation in $z$ direction. The second property makes it possible to

solve the problem as an equation evolving in $z$ direction; we hence can discretize $(\partial_z v)$ by applying classical difference scheme. Consequently, we obtain a quasi-gas-dynamic (QGD) equation which have been thoroughly studied in literature [72, 55, 57].

It is very common in the geological problems [64] that the media is variable and heterogeneous; that is, $c$ is non-homogeneous given a cross section in $z$ and is changing in $z$ direction. This brings in two difficulties. The first to mention is the multiscale property brought by the heterogeneous field. The second difficulty is the intense work in solving a QGD model given a $z$ cross-section in the $z$ evolution. Throughout the work, we are going to study (3.2) with variable and heterogeneous media. We will derive scheme in discretizing the time and $z$ evolution and also provide our approach to solve the problems mentioned before.

As we have discussed, the model has heterogeneous in $z$ direction. Directly solving this problem on fine mesh can capture the multiscale features; however, this is computationally intense and this issue becomes exacerbate when people are solving time-dependent problems. Therefore, many methods which solve the multiscale problems on coarser mesh have been proposed. These include homogenization-based approaches [1, 2, 3, 4, 5, 6, 7], multiscale finite element methods [8, 9, 10, 11], generalized multiscale finite element methods (GMsFEM) [18, 12, 13, 14, 15, 16, 17], constraint energy minimizing GMsFEM (CEM-GMsFEM) [19, 20], nonlocal multi-continua (NLMC) approaches [21], metric-based upscaling [22], heterogeneous multiscale method [23, 24], localized orthogonal decomposition (LOD) [25, 26], equation free approaches [27, 29, 28], computational continua [30, 31, 32], hierarchical multiscale method [33, 34, 35], and so on. Some of these approaches, such as homogenization-based approaches, are designed for problems with scale separation. In this work, we apply the CEM-GMsFEM [19, 20] and provide the convergence analysis of our proposed scheme based on the coarse mesh convergence results of the CEM-GMsFEM.

The second difficulty of the problem is the variable media. If we discretize $z$ evolution using some classical difference scheme, each $z$ level is a QGD model with heterogeneous. This model can be solved in the framework of CEM-GMsFEM [72]; however the coarse scale basis evaluation is time consuming; in particular, this process will be repeated for each level of $z$. We hence

proposed the proper orthogonal decomposition (POD) technique [73] which is target to find low dimensional subspace such that the error of the orthogonal projection is minimized in the sense of the norm induced by the inner product of the original space. To be more specific, the proposed method can be summarized as follows:

We proposed a method to solve a paraxial wave model with variable and heterogeneous media. We first apply the Backward-Euler scheme in the discretization of the $\partial_z v$. This leads to a set of two dimensional (space) QGD models (3.3); we call this level of discretization the quasi-time scheme and an unconditional stability result is established. To further discretize the problem, we apply the central scheme to deal with the time derivative $v_{tt}$, $v_t$; and then use CEM-GMsFEM in space on coarse scale to capture the heterogeneous brought by the media. We then prove that the full discretization scheme is stable in an energy norm under some CFL condition by using an inverse inequality in the multiscale space.

The key of the CEM-GMsFEM method is to construct the CEM basis. The standard procedure is first to build the auxiliary multiscale basis by solving local spectral problems in coarse mesh; we then can construct the CEM basis by evaluating a set of energy minimization problems. Due to the variable media, we need to construct a set of basis for each QGD model and the solution of the full discretized scheme is in the space of all multiscale basis. This is time consuming; and we hence can apply the POD technique to find the best set of orthogonal basis in the sense of $L_2$ minimization; that is, the projection error of the full discretized solution onto the POD basis is optimal in the norm induced by the original space. In practice, we collect CEM basis for some QGD models and then construct POD basis of the space spanned by all CEM basis; the POD models will finally be solved by using POD basis. A convergence analysis of the POD approximation is established and the numerical results prove the algorithm is successful.

The remainder of the chapter is organized as follows. In Section 3.2, we present some preliminaries of the model problem and briefly overview the framework of proper orthogonal decomposition. Section 3.3 is devoted to the multiscale methods and we will briefly overview the construction of multiscale basis function within the framework of the CEM-GMsFEM. In Section

3.4, we present a complete analysis of the proposed computational multiscale method. We then present some numerical results to demonstrate the efficiency of the proposed method in Section 3.5. Concluding remarks are drawn in Section 2.6.

### 3.2 Preliminaries

In this section, we present some preliminaries of the model problem. For simplicity, we assume the homogeneous boundary condition $g = 0$ is equipped in the model problem (3.2). The extension of inhomogeneous case is straightforward. The system (3.2) is the one we shall consider throughout the remainder of this chapter and we shall develop computational multiscale methods in order to solve this problem efficiently. We remark that under appropriate regularity assumptions on initial and boundary conditions, the problem (3.2) has a unique solution such that

$$t \to v(z, x, t) \in W^{1,\infty}(0, T; L^2(D \times \Omega)) \cap L^\infty(0, T; L^2(D \times \Omega)),$$

$$z \to v(z, x, t) \in L^\infty(D, H^1(0, T; L^2(\Omega))).$$

The last property enables us to consider $z$ as an evolution direction. The result is an application of the semigroup theory and the Hille-Yoshida theorem. See [62, Section 4] for more details.

Instead of the PDE formulation (3.2), we consider its corresponding variational formulation. In the following, we treat $z$ as an evolution direction and use backward Euler method to discretize the term $\partial_z v$. We divide the domain $D = [d_1, d_2]$ along $z$-direction into $K$ pieces. We write, for $k = 0, 1, \cdots, K$,

$$v_k = v(z_k) \quad \text{with} \quad z_k = d_1 + k\Delta z \quad \text{and} \quad d_2 = d_1 + K\Delta z.$$

The quasi-time discretization of (3.2) reads: find $\theta_k \in V$ for $k = 1, \cdots, K-1$ such that

$$\left(\ddot{\theta}_k, w; z_k\right)_c + \frac{1}{\Delta z}\left(\dot{\theta}_k - \dot{\theta}_{k-1}, w\right) + \frac{1}{2}a(\theta_k, w; z_k) = 0 \quad \text{for all } w \in V := H_0^1(\Omega) \tag{3.3}$$

or equivalently

$$\Delta z \left(\dot{\theta}_k, w; z_k\right)_c + \left(\dot{\theta}_k, w\right) + \frac{\Delta z}{2} a(\theta_k, w; z_k) = \left(\dot{\theta}_{k-1}, w\right) \quad \text{for all } w \in V,$$

where $(v, w; z_k)_c := \int_\Omega c^{-1}(z_k) vw \, dx$, $(v, w) := \int_\Omega vw \, dx$, and $a(v, w; z_k) := \int_\Omega c(z_k) \nabla v \cdot \nabla w \, dx$. Here, we denote $\dot{v} := \partial_t v$ and $\ddot{v} := \partial_{tt} v$. We assume that $c \in L^\infty(D \times \Omega)$ and we denote $c_{\max} := \|c\|_{L^\infty(D\times\Omega)}$. Employing Galerkin's method and the method of energy estimate, one can show the well-posedness of the variational formulation (3.3). Denote $\|v\| := \sqrt{(v, v)}$, $\|v\|_{c(z_k)} := \sqrt{(v, v; z_k)_c}$, and $\|v\|_{a(z_k)} := \sqrt{a(v, v; z_k)}$. We establish the following stability estimate for the quasi-time discretization (3.3).

**Lemma 3.2.1.** *Let $\{v_k\}_{k=0}^K \subseteq V$ solve the equation* (3.3)*. Then, the following stability estimate holds:*

$$\|\dot{v_K}\|^2_{L^2(0,T;L^2(\Omega))} + \Delta z \sum_{k=1}^K \mathcal{E}_k(v_k(T)) \lesssim \|\dot{v_0}\|^2_{L^2(0,T;L^2(\Omega))} + \Delta z \sum_{k=1}^K \mathcal{E}_k(v_k(0)),$$

*where we denote $\mathcal{E}_k(v) := \|\dot{v}\|^2_{c(z_k)} + \frac{1}{2}\|v\|^2_{a(z_k)}$.*

**Remark 3.2.2.** *The term $\mathcal{E}_k(v_k(0)) = \|v_1(z_k)\|^2_{c(z_k)} + \frac{1}{2}\|v_0(z_k)\|^2_{a(z_k)}$ on the right-hand side of the inequality above is determined by the initial conditions.*

*Proof of Lemma 3.2.1.* Taking $w = \dot{v_k}$, we have

$$\begin{aligned} &\frac{\Delta z}{2}\frac{d}{dt}\left(\|\dot{v}_k\|^2_{c(z_k)} + \frac{1}{2}\|v_k\|^2_{a(z_k)}\right) + \|\dot{v}_k\|^2 - (\dot{v}_{k-1}, \dot{v}_k) \\ &= \Delta z \left(\ddot{v}_k, \dot{v}_k; z_k\right)_c + (\dot{v}_k - \dot{v}_{k-1}, \dot{v}_k) + \frac{\Delta z}{2} a(v_k, \dot{v_k}; z_k) = 0. \end{aligned}$$

Thus, we have

$$\frac{\Delta z}{2}\frac{d}{dt}\left(\|\dot{v}_k\|^2_{c(z_k)} + \frac{1}{2}\|v_k\|^2_{a(z_k)}\right) + \|\dot{v}_k\|^2 = (\dot{v}_k, \dot{v}_{k-1}) \le \|\dot{v}_k\| \, \|\dot{v}_{k-1}\| .$$

Multiplying by 2 and integrating over $(0, T]$, we have

$$\begin{aligned}
&\Delta z \left( \|\dot{v}_k(T)\|^2_{c(z_k)} + \frac{1}{2} \|v_k(T)\|^2_{a(z_k)} \right) + 2 \int_0^T \|\dot{v}_k\|^2 \, dt \\
&\leq 2 \int_0^T \|\dot{v}_k\| \, \|\dot{v}_{k-1}\| \, dt + \Delta z \left( \|\dot{v}_k(0)\|^2_{c(z_k)} + \frac{1}{2} \|v_k(0)\|^2_{a(z_k)} \right) \\
&\leq \int_0^T \|\dot{v}_k\|^2 \, dt + \int_0^T \|\dot{v}_{k-1}\|^2 \, dt + \Delta z \left( \|\dot{v}_k(0)\|^2_{c(z_k)} + \frac{1}{2} \|v_k(0)\|^2_{a(z_k)} \right).
\end{aligned}$$

Therefore, we have

$$\begin{aligned}
\|\dot{v}_k\|^2_{L^2(0,T;L^2(\Omega))} &+ \Delta z \left( \|\dot{v}_k(T)\|^2_{c(z_k)} + \frac{1}{2} \|v_k(T)\|^2_{a(z_k)} \right) \\
&\leq \|\dot{v}_{k-1}\|^2_{L^2(0,T;L^2(\Omega))} + \Delta z \left( \|\dot{v}_k(0)\|^2_{c(z_k)} + \frac{1}{2} \|v_k(0)\|^2_{a(z_k)} \right).
\end{aligned}$$

Summing over $k = 1, \cdots, K$, we obtain the desired result and this completes the proof. □

### 3.2.1 The Proper orthogonal decomposition

In this section, we briefly introduce the proper orthogonal decomposition (POD) method. This method aims to generate optimally ordered orthogonal basis functions in the least squares sense for a given set of theoretical, experimental, or computational data. Reduced-order models or surrogate models are then obtained by truncating this set of optimal basis functions, providing considerable computational savings over the original high-dimensional problems.

Let $X$ be a real Hilbert space endowed with inner product $(\cdot, \cdot)_X$ and norm $\|\cdot\|_X$. We set $\mathcal{V} := \text{span}\{y_1, y_2, \cdots, y_n\}$ with each $y_i \in X$ for $i \in \{1, \cdots, n\}$. We refer to $\mathcal{V}$ as ensemble consisting of the snapshots $\{y_i\}_{i=1}^n$, at least one of which is assumed to be non-zero. Let $\{\psi_k\}_{k=1}^{\mathcal{N}}$ denote a set of orthonormal basis functions of $\mathcal{V}$ with $\mathcal{N} := \dim(\mathcal{V}) \leq n$. Then, each member of the ensemble can be expressed as

$$y_j = \sum_{k=1}^{\mathcal{N}} (y_j, \psi_k)_X \psi_k$$

for each $j \in \{1, \cdots, n\}$. The POD method consists in choosing the orthonormal basis functions such that for every $\ell \in \{1, \cdots, \mathcal{N}\}$ the mean square error between the elements $y_j$ (for any $j \in$

$\{1, \cdots, n\}$), and the corresponding $\ell$-th partial sum is minimized on average:

$$\min_{\{\psi_k\}_{k=1}^{\ell}} \frac{1}{n} \sum_{j=1}^{n} \left\| y_j - \sum_{k=1}^{\ell} (y_j, \psi_k)_X \psi_k \right\|_X^2 \tag{3.4}$$
$$\text{subject to} \quad (\psi_k, \psi_t) = \delta_{kt} \quad \text{for any } k, t \in \{1, 2, \cdots, \ell\}.$$

Here, $\delta_{kt}$ denotes the Kronecker-delta function. A solution $\{\psi_k\}_{k=1}^{\ell}$ to (3.4) is called a POD-basis of rank $\ell$. We introduce the correlation matrix

$$K = \left( \frac{1}{n} (y_j, y_i)_X \right) \in \mathbb{R}^{n \times n}$$

corresponding to the snapshots $\{y_j\}_{j=1}^{n}$. The matrix $K$ is positive semi-definite and has rank $\mathcal{N}$. The minimization problem (3.4) can be reduced to an eigenvalue problem

$$Kv = \lambda v. \tag{3.5}$$

We sort all the positive eigenvalues in a decreasing order as $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{\mathcal{N}} > 0$ and the associated eigenvectors are denoted by $v_k$ with $k = 1, \cdots, \mathcal{N}$. It can be shown that the POD-basis of rank $\ell \in \mathbb{N}^+$ with $\ell \leq \mathcal{N}$ is formed by

$$\varphi_k = \frac{1}{\sqrt{\lambda_k}} \sum_{j=1}^{n} (v_k)_j y_j \quad \text{for } k = 1, \cdots, \ell. \tag{3.6}$$

Here, $(v_k)_j$ is the $j$-th component of the eigenvector $v_k$. The basis functions $\{\varphi_k\}_{k=1}^{\ell}$ form a POD-basis of rank $\ell$ and we have the following error formula.

**Proposition 3.2.3.** *Let $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{\mathcal{N}} > 0$ be the positive eigenvalues of $K$ in* (3.5) *and $v_1, \cdots, v_{\mathcal{N}} \in \mathbb{R}^n$ be the associated eigenvectors. Then, $\{\varphi_k\}_{k=1}^{\ell}$ given by* (3.6) *forms a set of*

*POD-basis of rank $\ell$ with $\ell \leq \mathcal{N}$. Moreover, we have the error formula*

$$\frac{1}{n}\sum_{j=1}^{n}\left\|y_j - \sum_{k=1}^{\ell}(y_j,\varphi_k)_X\varphi_k\right\|_X^2 = \sum_{k=\ell+1}^{\mathcal{N}}\lambda_k.$$

In practice, we shall make use of the decay property of eigenvalues in $\lambda_k$ and choose the first $\ell$ dominant eigenvalues such that the ratio $\zeta := \frac{\sum_{k=\ell+1}^{\mathcal{N}}\lambda_k}{\sum_{k=1}^{\mathcal{N}}\lambda_k}$ is small enough to achieve an expected accuracy, for instance $\zeta = 1\%$. One would prefer the eigenvalues decays as fast as possible so that one can ensure high accuracy with a few POD basis functions.

### 3.3 Multiscale method

In this section, we develop the computational multiscale method in order to solve the paraxial approximation. For spatial discretization, we will apply the CEM-GMsFEM. In particular, for each node $z_i \in D$ along the $z$-direction, we will construct a set of multiscale basis functions in the spirit of CEM-GMsFEM. To further reduce the dimension of the multiscale space, we will perform POD procedure related to these CEM basis functions. Once the multiscale space is constructed, one can use leapfrog scheme to discretize time derivatives and solve the resulting fully-discretized problem.

#### 3.3.1 Spatial discretization: CEM-GMsFEM

We are going to apply the CEM-GMsFEM framework to discretize the space. Then, we perform the POD procedure on the snapshot space $\mathcal{V}$ and we denote $\{\psi_k\}_{k=1}^{\mathcal{N}}$ the corresponding POD basis functions with cardinality $\mathcal{N} \in \mathbb{N}^+$. We then define $V_{\text{POD}}^{\ell} := \text{span}\{\psi_k\}_{k=1}^{\ell}$ for a given positive number $\ell \leq \mathcal{N}$.

### 3.3.2 Construction of multiscale reduced basis functions using POD

In this section, we present the construction of multiscale reduced basis functions using the POD technique. First, we define $\{v_{\text{CEM},k}\}_{k=0}^{K} \subset V_{\text{CEM}}$ to be the solution of the following equation:

$$\left(\ddot{v}_{\text{CEM},k}, w; z_k\right)_c + \frac{1}{\Delta z}\left(\dot{v}_{\text{CEM},k} - \dot{v}_{\text{CEM},k-1}, w\right) + \frac{1}{2}a(v_{\text{CEM},k}, w; z_k) = 0 \quad \text{for all } w \in V_{\text{CEM}}. \tag{3.7}$$

Space $X = H^1$ and inner product associated with $X$ is defined as $(u, v)_{H^1} = (\nabla u, \nabla v) + (u, v)$ for $u, v \in X$; the corresponding norm will then be defined as $\|.\|_{H^1} := \sqrt{(.,.)_{H^1}}$. Next, we define the snapshot space $\mathcal{V}$ as the collection of:

$$y_j := v_{\text{CEM}}(t, z_{j-1}), \quad y_{j+K+1} := \ddot{v}_{\text{CEM}}(t, z_{j-1}), \quad j = 1, \cdots, K+1,$$

and

$$y_{j+2K+2} = \tilde{\partial}\dot{v}_{\text{CEM}}(t, z_j) := \frac{\dot{v}_{\text{CEM}}(t, z_j) - \dot{v}_{\text{CEM}}(t, z_{j-1})}{\Delta z}, \quad j = 1, \cdots, K.$$

Then, we perform the POD procedure on the snapshot space $\mathcal{V}$ as described in section (3.2.1). It should be noted that the correlation matrix is defined as $K_{ij} := \frac{1}{3K+2}(y_j, y_i)_{H^1}$. We denote $\{\psi_k\}_{k=1}^{\mathcal{N}}$ the corresponding POD basis functions with cardinality $\mathcal{N} \in \mathbb{N}^+$. We then define $V_{\text{POD}}^{\ell} := \text{span}\{\psi_k\}_{k=1}^{\ell}$ for a given positive number $\ell \leq \mathcal{N}$.

### 3.3.3 Fully discretization

In this section, we present the fully discretization for the problem (3.3). We can further consider the discretization in time. We divide $(0, T]$ into $N$ pieces and write

$$v^n = v(t^n) \quad \text{with} \quad t^n = n\Delta t \quad \text{and} \quad \Delta t = \frac{T}{N}.$$

Specifically, we use first- and second-order central difference schemes. We define $\tau := \Delta t \Delta z$. The fully discretization reads: for $k = 1, \cdots, K-1$ and $n = 1, \cdots, N-1$, find $\{v_k^n\} \subset V_{\text{POD}}^{\ell}$

such that

$$\left(\frac{v_k^{n+1}-2v_k^n+v_k^{n-1}}{(\Delta t)^2},w;z_k\right)_c+\left(\frac{v_k^{n+1}-v_k^{n-1}}{2\tau},w\right)+\frac{1}{2}a(v_k^n,w;z_k)=\left(\frac{v_{k-1}^{n+1}-v_{k-1}^{n-1}}{2\tau},w\right) \quad (3.8)$$

for all $w \in V_{\text{POD}}^{\ell}$. In short, we discretize implicitly in $z$ and explicitly in time. We show that the fully discretization is stable in time. To this aim, we first recall the inverse inequality for the multiscale space, which is proved in [72].

**Proposition 3.3.1** (Lemma 4.5 in [72]). *Assume that $\{\chi_{j,k}^{ms}\}_{j=1}^{N_c}$ is a set of bilinear partition of unity. For any $v \in V_{CEM}$, there is a constant $C_{inv} > 0$ such that*

$$\|\nabla v\|_a \leq C_{inv} H^{-1} \beta^{1/2} \|v\| .$$

Then, we have the following stability result for (3.8).

**Lemma 3.3.2.** *Suppose that the following CFL condition*

$$\frac{1}{c_{\max}}-\frac{1}{4}C_{inv}^2 H^{-2}\beta(\Delta t)^2 \geq \delta^2,$$

*holds for some $\delta > 0$. Then, we have the following stability estimate:*

$$\frac{1}{4\tau}\sum_{n=1}^{N}\left\|v_K^{n+1}-v_K^{n-1}\right\|^2+\sum_{k=1}^{K}\mathcal{E}_{N,k} \leq \frac{1}{4\tau}\sum_{n=1}^{N}\left\|v_0^{n+1}-v_0^{n-1}\right\|^2+\sum_{k=1}^{K}\mathcal{E}_{0,k}.$$

*Proof.* Define the energy as follows:

$$\mathcal{E}_{n,k} := \left\|\frac{v_k^{n+1}-v_k^n}{\Delta t}\right\|_c^2+\frac{1}{2}a(v_k^{n+1},v_k^n).$$

Using the inverse inequality for the multiscale functions, one can show that

$$
\begin{aligned}
\mathcal{E}_{n,k} &= \left\| \frac{v_k^{n+1} - v_k^n}{\Delta t} \right\|_c^2 + \frac{1}{2} a(v_k^{n+1}, v_k^n) \\
&= \left\| \frac{v_k^{n+1} - v_k^n}{\Delta t} \right\|_c^2 + \frac{1}{4} a(v_k^{n+1}, v_k^{n+1}) + \frac{1}{4} a(v_k^n, v_k^n) - \frac{1}{4} a(v_k^{n+1} - v_k^n, v_k^{n+1} - v_k^n) \\
&\geq \left\| \frac{v_k^{n+1} - v_k^n}{\Delta t} \right\|_c^2 + \frac{1}{4} \left( \left\| v_k^{n+1} \right\|_a^2 + \| v_k^n \|_a^2 \right) - \frac{1}{4} C_{\text{inv}}^2 H^{-2} \beta (\Delta t)^2 \left\| \frac{v_k^{n+1} - v_k^n}{\Delta t} \right\|^2 \\
&\geq \left( \frac{1}{c_{\max}} - \frac{1}{4} C_{\text{inv}}^2 H^{-2} \beta (\Delta t)^2 \right) \left\| \frac{v_k^{n+1} - v_k^n}{\Delta t} \right\|^2 + \frac{1}{4} \left( \left\| v_k^{n+1} \right\|_a^2 + \| v_k^n \|_a^2 \right) \geq 0.
\end{aligned}
$$

Next, taking $w = v_k^{n+1} - v_k^{n-1}$ in (3.8) and denoting $f_{k-1}^n = v_{k-1}^{n+1} - v_{k-1}^{n-1}$, we have

$$
\frac{1}{2\tau} \left\| v_k^{n+1} - v_k^{n-1} \right\|^2 + \left\| \frac{v_k^{n+1} - v_k^n}{\Delta t} \right\|_c^2 - \left\| \frac{v_k^n - v_k^{n-1}}{\Delta t} \right\|_c^2 + \frac{1}{2} a(v_k^n, v_k^{n+1} - v_k^{n-1}) = \frac{1}{2\tau} \left( f_{k-1}^n, v_k^{n+1} - v_k^{n-1} \right).
$$

Then, it implies that

$$
\begin{aligned}
\frac{1}{2\tau} \left\| v_k^{n+1} - v_k^{n-1} \right\|^2 + \mathcal{E}_{n,k} - \mathcal{E}_{n-1,k} &= \frac{1}{2\tau} \left( f_k^n, v_k^{n+1} - v_k^{n-1} \right) \\
&\leq \frac{1}{2\tau} \left\| v_{k-1}^{n+1} - v_{k-1}^{n-1} \right\| \left\| v_k^{n+1} - v_k^{n-1} \right\| \\
&\leq \frac{1}{4\tau} \left( \left\| v_{k-1}^{n+1} - v_{k-1}^{n-1} \right\|^2 + \left\| v_k^{n+1} - v_k^{n-1} \right\|^2 \right) \\
\implies \frac{1}{4\tau} \left\| v_k^{n+1} - v_k^{n-1} \right\|^2 + \mathcal{E}_{n,k} &\leq \frac{1}{4\tau} \left\| v_{k-1}^{n+1} - v_{k-1}^{n-1} \right\|^2 + \mathcal{E}_{n-1,k}.
\end{aligned}
$$

Summing over $n = 1, \cdots, N$, we have

$$
\frac{1}{4\tau} \sum_{n=1}^{N} \left\| v_k^{n+1} - v_k^{n-1} \right\|^2 + \mathcal{E}_{N,k} \leq \frac{1}{4\tau} \sum_{n=1}^{N} \left\| v_{k-1}^{n+1} - v_{k-1}^{n-1} \right\|^2 + \mathcal{E}_{0,k}.
$$

Therefore, summing over $k = 1, \cdots, K$, we have

$$
\frac{1}{4\tau} \sum_{n=1}^{N} \left( \sum_{k=1}^{K} \left\| v_k^{n+1} - v_k^{n-1} \right\|^2 - \sum_{k=0}^{K-1} \left\| v_k^{n+1} - v_k^{n-1} \right\|^2 \right) + \sum_{k=1}^{K} \mathcal{E}_{N,k} \leq \sum_{k=1}^{K} \mathcal{E}_{0,k}
$$

and

$$\frac{1}{4\tau}\sum_{n=1}^{N}\left\|v_K^{n+1}-v_K^{n-1}\right\|^2+\sum_{k=1}^{K}\mathcal{E}_{N,k}\leq\frac{1}{4\tau}\sum_{n=1}^{N}\left\|v_0^{n+1}-v_0^{n-1}\right\|^2+\sum_{k=1}^{K}\mathcal{E}_{0,k}.$$

This completes the proof. □

**Remark 3.3.3.** *It should be noted that the CFL condition depends on the* $c_{\max}:=\|c\|_{L^\infty(D\times\Omega)}$. *It also relies on* $C_{inv}$ *which comes from the inverse inequality (Lemma 2.4.5). From the proof of the lemma,* $C_{inv}$ *depends on the highest permeability of the medium. We are using the CEM method which is a coarse mesh method, hence the coarse mesh size* $H$ *will make the CFL much less restrictive. The idea to improve the CFL is to use the time implicit scheme; more precisely, use* $a(v_k^{n+1},w;z_k)$ *in (3.8); however, the stability is hardly to be proved in this case. This will become our future work.*

### 3.4 Convergence analysis

In this section, we present the convergence analysis of the proposed POD-based multiscale method for the paraxial wave equation. Let $v_{\text{CEM}}\in V_{\text{CEM}}$ be the semi-discretized solution which solves the problem (3.7).

Throughout this section, we assume that the coercivity and boundedness of this bilinear form hold. Specifically, there exist two constants $\gamma$ and $\beta$, independent of any $z_k\in D$, such that

$$\gamma\|u\|^2_{H_1(\Omega)}\leq a(u,u;z_k)\quad\text{and}\quad a(u,w;z_k)\leq\beta\|u\|_{H_1(\Omega)}\|w\|_{H_1(\Omega)}\tag{3.9}$$

for any $u,w\in V$. We denote $(\cdot,\cdot)_{H^1}$ the $H^1$-norm such that $(u,v)_{H^1}=(\nabla u,\nabla v)+(u,v)$ for any $u,v\in V$ with its associated norm being $\|\cdot\|_{H^1}:=\sqrt{(\cdot,\cdot)_{H^1}}$.

We have the following lemma for the proper orthogonal decomposition with the Hilbert space $X=V_{\text{CEM}}$ and $\|v\|_X:=\|v\|_{L^2(0,T;H^1(\Omega))}$.

**Lemma 3.4.1** (cf. Proposition 1 and Equation (7) in [73])**.** *Let* $v_{CEM}\in V_{CEM}$ *be the semi-discretized solution which solves the problem* (3.7) *and* $K_{ij}:=\frac{1}{3K+2}(y_j,y_i)_{H^1}$ *be the correlation matrix and* $\lambda_k$ *be the corresponding eigenvalues sorted ascending. For any integer* $\ell$ *with* $0<\ell\leq\mathcal{N}$, *the*

*following error formula holds:*

$$\frac{1}{3K+2}\left[\int_0^T \sum_{k=1}^{K+1}\left\|v_{CEM}(t,z_k)-\sum_{j=1}^{\ell}\left(v_{CEM}(t,z_k),\psi_j\right)_{H^1}\psi_j\right\|_{H^1}^2\right.$$
$$+\sum_{k=1}^{K}\left\|\tilde{\partial}\dot{v}_{CEM}(t,z_k)-\sum_{j=1}^{\ell}\left(\tilde{\partial}\dot{v}_{CEM}(t,z_k),\psi_j\right)_{H^1}\psi_j\right\|_{H^1}^2 \tag{3.10}$$
$$\left.+\sum_{k=1}^{K+1}\left\|\ddot{v}_{CEM}(t,z_k)-\sum_{j=1}^{\ell}\left(\ddot{v}_{CEM}(t,z_k),\psi_j\right)_{H^1}\psi_j\right\|_{H^1}^2 dt\right]=\sum_{k=\ell+1}^{\mathcal{N}}\lambda_k.$$

Next, we define the Ritz-projection $P^\ell : V \to V_{\text{POD}}^\ell$ by:

$$(P^\ell v,\psi)_{H^1}=(v,\psi)_{H^1} \quad \text{for all } \psi\in V_{\text{POD}}^\ell, \tag{3.11}$$

for any $v \in V$. We have the following estimate for the projection operator.

**Lemma 3.4.2.** *Let $\{z_k\}_{k=1}^{K+1} \subset D$. For any $\ell \in \{1,2,...,\mathcal{N}\}$ and $v \in V$, the projection operator $P^\ell$ satisfies the following estimate:*

$$\frac{1}{K+1}\sum_{k=1}^{K+1}\int_0^T \left\|v_{CEM}(z_k)-P^\ell v_{CEM}(z_k)\right\|_{H^1}^2 \, dt \lesssim \sum_{k=\ell+1}^{\mathcal{N}}\lambda_k. \tag{3.12}$$

*Proof.* Let $v \in V$ be arbitrary. By the definition of the projection $P^\ell$, we have

$$\left\|v-P^\ell v\right\|_{H^1}^2=(v-P^\ell v,v-P^\ell v)_{H^1}=(v-P^l v,v-\psi)_{H^1}\le\left\|v-P^\ell v\right\|_{H^1}\|v-\psi\|_{H^1}$$

for all $\psi \in V_{\text{POD}}^\ell$. This implies that,

$$\left\|v-P^\ell v\right\|_{H^1}\lesssim\|v-\psi\|_{H^1} \quad \text{for all } \psi\in V_{\text{POD}}^\ell. \tag{3.13}$$

The lemma follows immediately by Lemma 3.4.1 and (3.13).

□

Clearly, from the proof of Lemma 3.4.2, the corollary below follows immediately.

**Corollary 3.4.3.** *Let $v_{CEM} \in V_{CEM}$ be the semi-discretized solution which solves* (3.7)*. Then, the following estimates hold:*

$$\frac{1}{K}\sum_{k=1}^{K}\int_0^T \left\|\tilde{\partial}\dot{v}_{CEM}(z_k) - P^\ell\tilde{\partial}\dot{v}_{CEM}(z_k)\right\|_{H^1}^2 dt \lesssim \sum_{i=\ell+1}^{\mathcal{N}}\lambda_i \tag{3.14}$$

*and*

$$\frac{1}{K+1}\sum_{k=0}^{K}\int_0^T \left\|\ddot{v}_{CEM}(z_k) - P^\ell\ddot{v}_{CEM}(z_k)\right\|_{H^1}^2 dt \lesssim \sum_{i=\ell+1}^{\mathcal{N}}\lambda_i. \tag{3.15}$$

Let $\{U_k\}_{k=0}^K \subset V_{\text{POD}}^\ell$ be the semi-discretized solution satisfying the following equation

$$(\ddot{U}_k, \phi; z_k)_c + (\tilde{\partial}\dot{U}_k, \phi) + \frac{1}{2}a(U_k, \phi; z_k) = 0 \tag{3.16}$$

for all $\phi \in V_{\text{POD}}^\ell$ with appropriate initial condition $U_0$. Here, we denote $\tilde{\partial}\dot{U}_k := \frac{\dot{U}_k - \dot{U}_{k-1}}{\Delta z}$.

We are now able to analyze the error. Assume that $\{v_k^n\} \subset V_{\text{POD}}^\ell$ solves (3.8), $\{U_k\}_{k=0}^K \subset V_{\text{POD}}^\ell$ solves (3.16), and the function $v_{\text{CEM}} \in V_{\text{CEM}}$ solves (3.7). We decompose the error into three parts:

$$v_k^n - v_{\text{CEM}} = \underbrace{v_k^n - U_k(t_n)}_{=:\mu_k} + \underbrace{U_k(t_n) - P^\ell v_{\text{CEM}}}_{=:\nu_k} + \underbrace{P^\ell v_{\text{CEM}} - v_{\text{CEM}}}_{=:\rho_k}.$$

Due to Lemma 3.4.2, we have

$$\frac{1}{K}\sum_{k=1}^{K}\|\rho_k\|_{L^2(0,T;H^1(\Omega))}^2 \lesssim \sum_{k=\ell+1}^{\mathcal{N}}\lambda_k.$$

It should be noted that $\sum_{k=\ell+1}^{\mathcal{N}} \lambda_k$ comes from the POD and is the typical error in the POD analysis. The error decay to $0$ very fast; and by the theory, this error is optimal [73] (please check Proposition (3.2.3)) since the POD solves a minimization problem (3.4). Next, using the notation $\tilde{\partial}\nu_k = \frac{\nu_k - \nu_{k-1}}{\Delta z}$ for all $k = 1, ..., K$ and the equation (3.7), we obtain

$$\begin{aligned}(\ddot{\nu}_k, \psi; z_k)_c + (\tilde{\partial}\dot{\nu}_k, \psi) + \frac{1}{2}a(\nu_k, \psi; z_k) &= -(P^\ell \ddot{v}_{\text{CEM}}, \psi; z_k)_c - (\tilde{\partial} P^\ell \dot{v}_{\text{CEM}}, \psi) - \frac{1}{2}a(v_{\text{CEM}}, \psi; z_k) \\ &= (\ddot{v}_{\text{CEM}} - P^\ell \ddot{v}_{\text{CEM}}, \psi; z_k)_c + (\tilde{\partial}\dot{v}_{\text{CEM}} - \tilde{\partial} P^\ell \dot{v}_{\text{CEM}}, \psi)\end{aligned}$$

for any $\psi \in V_{\text{POD}}^\ell$. Let us denote

$$h_k := \tilde{\partial}\dot{v}_{\text{CEM}} - \tilde{\partial} P^\ell \dot{v}_{\text{CEM}} \quad \text{and} \quad w_k := \ddot{v}_{\text{CEM}} - P^\ell \ddot{v}_{\text{CEM}}.$$

By Corollary 3.4.3, it follows that

$$\frac{1}{K}\sum_{k=1}^{K} \|h_k\|^2_{L^2(0,T;H^1(\Omega))} + \frac{1}{K+1}\sum_{k=0}^{K} \|w_k\|^2_{L^2(0,T;H^1(\Omega))} \lesssim \sum_{i=\ell+1}^{\mathcal{N}} \lambda_i.$$

Using the same technique of showing Lemma 3.2.1, one can obtain an estimate for $\nu_k$:

$$\|\dot{\nu}_k\|^2_{L^2(0,T;L^2(\Omega))} + \Delta z \sum_{k=1}^{K} \mathcal{E}_k(\nu_k(T)) \lesssim \Delta z \sum_{k=1}^{K} \|h_k\|^2_{L^2(0,T;H^1(\Omega))} + \|w_k\|^2_{L^2(0,T;H^1(\Omega))} \lesssim \sum_{i=\ell+1}^{\mathcal{N}} \lambda_i.$$

It remains to estimate the term $\mu_k$. This error comes from the temporal discretization and it follows that

$$\|\mu_k\|^2_{L^2(0,T;H^1(\Omega))} = \int_0^T \|\mu_k\|^2_{H^1}\, dt = \int_0^T O(\tau^2) dt = O(\tau).$$

Denote $v_k(t)$ the piecewise linear function that interpolates $\{v_k^n\}$ in time. As a result, we have the following error estimate

$$\frac{1}{K}\sum_{k=1}^{K} \|v_k^n - v_{\text{CEM}}\|^2_{L^2(0,T;H^1(\Omega))} \lesssim O(\tau) + \sum_{i=\ell+1}^{\mathcal{N}} \lambda_i. \tag{3.17}$$

We remark that the error between the solution $\theta$ (that solves (3.3)) and the solution $v_{\text{CEM}}$ has the relation: $\|\theta - v_{\text{CEM}}\|_{L^2(0,T;H^1(\Omega))} = O(H)$. Therefore, the whole error estimate reads as follows:

$$\frac{1}{K}\sum_{k=1}^{K}\|\theta - v_k^n\|^2_{L^2(0,T;H^1(\Omega))} \lesssim O(H^2) + O(\tau) + \sum_{i=\ell+1}^{\mathcal{N}}\lambda_i. \tag{3.18}$$

### 3.5 Numerical experiments

In this section, we present some numerical results to demonstrate the efficiency of the proposed computational multiscale method. In $(x_1, x_3)$ dimension, we discretize the domain using the POD-based CEM-GMsFEM. The fine mesh size $h = \frac{1}{100}$ and the coarse mesh is $H = 10h$. We compute the numerical solutions $v_k^N$ for all $k = 1, \cdots, K$ and compare it with the reference solution $v_f(z, x, T)$ for all $z \in D$ which are computed by using the underlying fine mesh. We will measure the relative error, which is defined as follows:

$$\text{Relative Error} = \frac{\|v_k^N - v_f(z_k, x, T)\|_{a(z_k)}}{\|v_f(z_k, x, T)\|_{a(z_k)}}.$$

#### 3.5.1 The first experiment

In the experiment below, we set the terminal time $T = 0.05$ and $\Omega = (0, 1)^2$. We set the initial condition to be $v(0, x, t) = \sin(\pi x_1)\sin(\pi x_3)\sin(t)$ where $x = (x_1, x_3) \in \Omega$; also we set homogeneous initial condition for all $z$, i.e., $v(z, x, 0) = 0$ for all $(z, x) \in D \times \Omega$. The temporal direction and $z$-direction are discretized as in the scheme (2.6). We choose the time step to be $\Delta t = 10^{-5}$ and the step size in $z$-direction is $\Delta z = 10^{-4}$. We set $K = 30$, i.e., the $z$ direction is partitioned into 30 levels and $D = [0, 30 \cdot \Delta z]$.

In this experiment, the pattern of the permeability field $c(x)$ is given in Figure 3.1 (Left). In $x_1$- and $x_3$-directions, the permeability will follow the same pattern. However, along the $z$-direction, the permeability field $c(x)$ has different values of contrast. See Figure 3.1 (Right) for an illustration for this layer-structured heterogeneous media. In particular, we partition the permeability field along the $z$-direction into 3 different layers; the value of contrast in the yellow region within each

layer are equal to 10, 15, and 20, respectively. We remark that the patterns keep the same as demonstrated in Figure 3.1 (Left).

To get the POD basis, we first calculate the CEM basis for each layer and then obtain the POD basis by performing the SVD on the combined basis space. We choose 100 POD basis from the total 300 CEM basis.

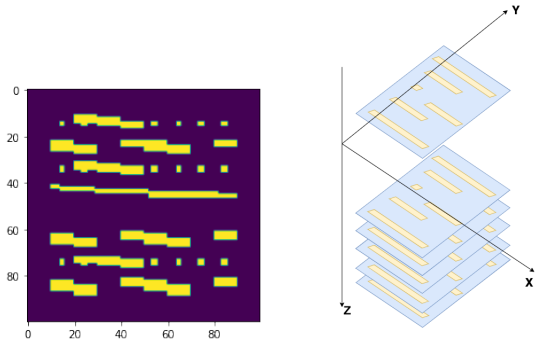


Figure 3.1: Left: The pattern of permeability field $c$ in $\Omega$. Right: Permeability field for all different $z$ have the same pattern but different contrasts ranges from 10 to 20).

The record of relative error for all $z_k$ at the terminal time is plotted in Figure 3.2. One can observe from the graph that the scheme is stable as expected.

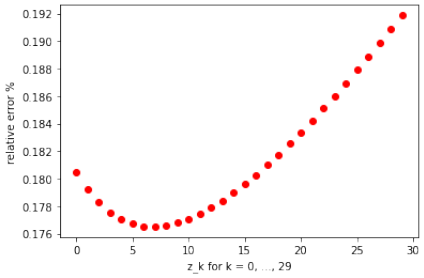


Figure 3.2: The relative error at terminal time in $z$-direction. $Y$ axis is the relative error. $X$ axis is the $z$ direction. We measure the relative error at the terminal time of each $z$ grid points

### 3.5.2 The second example

In the second experiment, we consider the Marmousi examples. More precisely, given $z_k \in D$, $c(z_k)$ is a Marmousi permeability field as shown in Figure (3.3).

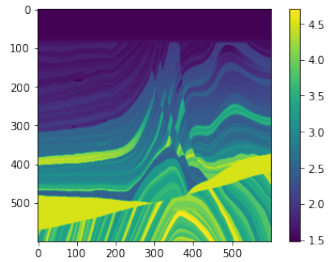


Figure 3.3: Marmousi permeability field. Each $c(z_k)$ is obtained by taking a fine element in each coarse element of size $6 \times 6$ in this fine grid field of size $600 \times 600$

In this example, initial and boundary data are set the same as before. We set $T = 0.005$ and $dt = 0.000005$. $\Delta z = 10^{-4}$ and we consider 10 z levels, i.e., $K = 10$ and $D = [0, 10 \cdot \Delta z]$. Each $c(z_k)$ which is a Marmousi permeability is obtained by taking data from a very fine Marmousi field of size $600 \times 600$. To be more specific, we partition the fine field ($600 \times 600$) into a coarse grid mesh of size $100 \times 100$ whose coarse element size is $6 \times 6$; then in each coarse element, we random pick up a fine element and formulate our computational fine grid ($100 \times 100$) as combining all selected elements accordingly.

To get the POD basis, we first calculate the CEM basis 3 layers ($k = 0, 4, 8$) and then obtain the POD basis by performing the SVD on the combined basis space. We choose $50$ POD basis from the total $300$ CEM basis. The record of relative error for all $z_k$ at the terminal time is plotted in Figure 3.4. One can observe from the graph that the scheme is accurate.

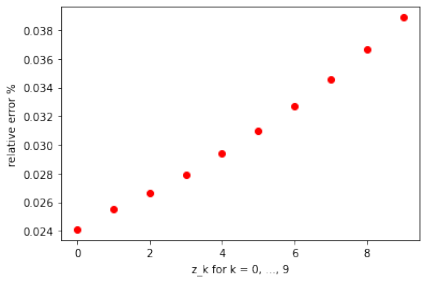


Figure 3.4: Marmousi Example. The relative error at terminal time in $z$-direction. $Y$ axis is the relative error. $X$ axis is the $z$ direction. We measure the relative error at the terminal time of each $z$ grid points

# 4. LEARNING ALGORITHMS FOR COARSENING UNCERTAINTY SPACE AND APPLICATIONS TO MULTISCALE SIMULATIONS *

## 4.1 Introduction

Many problems are multiscale with uncertainties. Examples include problems in porous media, material sciences, biological sciences, and so on. For example, in porous media applications, engineers can obtain fine-scale data about pore geometries or subsurface properties at very fine resolutions. These data are obtained in some spatial locations and then generalized to the entire reservoir domain. As a result, one uses geostatistical or other statistical tools to populate the media properties in space. The resulting porous media properties are stochastic and one needs to deal with many porous media realizations, where each realization is multiscale and varies at very fine scales. There are other realistic problems which have multiscale properties with uncertainties such as the multiscale public safety systems, [74], multiscale social networks [75]; these problems usually have more data.

Simulating each realization can be computationally expensive because of the media's multiscale nature. Our objective is to simulate many of these realizations. To address the issues associated with spatial and temporal scales, many multiscale methods have been developed [76, 77, 78, 79, 80, 81, 82, 83, 84, 85]. These methods perform simulations on the coarse grid by developing reduced-order models. However, developing reduced-order models requires local computations, which can be expensive when one deals with many realizations. For this reason, some type of coarsening of the uncertainty space is needed [86]. In this paper, we consider some novel approaches for developing coarsening of uncertainty space as discussed below.

To coarsen the uncertainty space, clustering algorithms are often used; but a proper distance function should be designed in order to make the clusters have physical sense and achieve a reduction in the uncertainty space. The paper [86] proposed a method that uses the distance between

*Reprinted with permission from "Learning Algorithms for Coarsening Uncertainty Space and Applications to Multiscale Simulations" by Zecheng Zhang, Eric T. Chung, Yalchin Efendiev and Wing Tat Leung, 2020. Mathematics, 8(5):720, 2020, Copyright [2020] by MDPI.

local solutions. The motivation is that the local problems with random boundary conditions can represent the main models with all boundary conditions. Due to a high dimension of the uncertainty space, the authors in [86] proposed to compute the local solutions of only several realizations and then use the Karhunen–Loeve expansion [87] to approximate the solutions of all the other realizations. The distance function is then defined to be the distance between solutions and the standard K-means [88] algorithm is used to cluster the uncertainty space.

The issue with this method is computing the local solutions in the local neighborhoods. It is computationally expensive to compute the local solutions; although the KL expansion can save time to approximate the solutions of other realizations, one still needs to decide how many selected realizations we need to represent all the other solutions. In this paper, we propose the use of deep learning methodology and avoid explicit clustering as in earlier works. We remark that the development of deep learning techniques for multiscale simulations are recently reported in [52, 50, 89, 90, 91].

In this work, to coarsen the uncertainty space, we propose a deep learning algorithm which will learn the clusters for each local neighborhood. Due the nature of the permeability fields, we can use the transfer learning which uses the parameters of one local neighborhood to initialize the learning of all the other neighborhoods. This saves significantly computational time.

The auto encoder structure [39] has been widely used in improving the K-mean clustering algorithm [92, 93, 94]. The idea is to use the encoder to extract features and reduce the dimension; the encoding process can also be taken as a kernel method [95] which maps the data to a space which is easier to be separated. The decoder is used to upsample the latent space (reduced dimension feature space) back to the input space. The clustering algorithm is then used to cluster the latent space, which will save time due to the low dimension of the latent space and also preserve the accuracy due to the features extracted by the encoder.

Traditionally, the learning process is only involved in reconstructing the input space. Such kind of methods ignore the features extracted by latent space; so, it is not clear if the latent space is good enough to represent the input space and is easily clustered by the K-means method. In [94],

the authors proposed a new loss which includes the reconstruction loss meanwhile the loss results from the clustering. The authors claimed that the new loss improves the clustering results.

We will apply the auto encoder structure and the multiple loss function; however, we will design the auto encoder as a generative network, i.e., the input and output space are different. More precisely, the input is the uncertain space (permeability fields) and the output will be the multiscale functions co-responding to the uncertain space. Intuitively, we want to use the multiscale basis to supervise the learning of the clusters so that the clusters will inherit the property of the solution. The motivation is the multiscale basis can somehow represent the real solutions and permeability fields; hence, the latent space is no longer good for clustering the input space but will be suitable for representing the multiscale basis function space.

To define the reconstructing loss, the common idea is the mean square error (MSE); but many works [96, 97, 98, 37] have shown that the MSE tends to produce the average effect. In fact, in the area of image super-resolution [96, 97, 98, 37, 99, 100, 101, 102, 103, 104, 105] and other low level computer vision tasks, the generated images are usually over-smooth if trained using MSE. The theory is the MSE will capture the low frequency features like the background which is relatively steady; but for images with high contrast, the MSE will usually try to blur the images and the resulting images will lose the colorfulness and become less vivid [96]. Our problem has multiscale nature and we want to capture the dominant modes and multiscale features, hence a single MSE is clearly not enough.

Following the idea from [37, 97], we consider adding an adversary net [106]. The motivation is the fact that different layers of fully convolutional network extract different features [36, 37, 38]. Deep fully convolutional neural networks (FCN) [107, 108, 109, 110, 111, 112] have demonstrated its power in almost all computer vision tasks. Convolution operation is a local operation and the network with full convolutions are independent with the input size. People now are clear about the functioning of the different layers of the FCN. In computer vision task, the lower layers (layers near input) tend to generate sharing features of all objects like edges and curves while the higher layers (near output) are more object oriented. If we train the network using the loss from the

lower layers, the texture and details are persevered, while the higher layers will keep the general spatial structure.

This motivates us using the losses from different layers of the fully convolutional layers. Multiple layers will give us a multilevel capture of the basis features and hence measure the basis in a more complete way. To implement the idea, we will pretrain an adversary net; and input the multiscale basis of the generative net and the real basis. The losses then come from some selected layers of the adversary net. Although it is still not clear the speciality of each layer, if we consider the multiscale physical problem, the experiments show that the accuracy is improved and, amazingly, the training becomes easier when compared to the MSE of the basis directly.

The uncertain space coarsening (cluster) is performed using the deep learning idea described above. Due to the space dimension, we will perform the clustering algorithm locally in space; that is, we first need a spatial coarsening. Due to the multiscale natural of the problem, this motivates us using the generalized multiscale finite element methods (GMsFEM) which derive the multiscale basis of a coarse neighborhood by solving the local problem. GMeFEM was first proposed in [113] and further studied in [76, 77, 78, 79, 80, 81, 82, 83]. This method is a generalization of the multiscale finite element method [114, 115]. The work starts from constructing the snapshot space for each local neighborhood. The snapshot space is constructed by solving local problems and several methods including harmonic extension, random boundary condition [116] have been proposed. Once we have the snapshot space, the offline space which will be used as computing the solution are constructed by using spectral decomposition.

Our method is designed for solving PDEs with heterogeneous properties and uncertainty. The heterogeneity and uncertainty in our models come from the permeability $\kappa(x, s)$. To verify our method, we numerically simulate around 240,000 local spatial fields which contain complex information such as the moving channels. Our model is then trained and tested based on the generated spatial fields. It should be noted that our method could be applied to some other realistic problems which contain large-scale data such as detecting extreme values with order statistics in samples from continuous distributions [117], as well as to some other subjects, e.g., multiscale

social networks [75] and the multiscale public safety systems [74]. These topic will be studied in the future.

The rest of the work is organized as follow: in Section 4.2, we consider the problem setup and introduce both uncertain space and spatial coarsening. In Section 4.3, we introduce the structure of the network and the training algorithm. In Section 4.4, we will present the numerical results. The paper ends with conclusions.

## 4.2 Problem settings

In this section, we will present some basic ideas involving the use of the generalized multiscale finite element method (GMsFEM) for parameter-dependent problems. Let $D$ be a bounded domain in $\mathbb{R}^2$ and $\Omega$ be the parameter space in $\mathbb{R}^N$. We consider the following parameter-dependent elliptic problem:

$$-\nabla \cdot (\kappa(x,s)\nabla u(x,s)) = f(x,s), (x,s) \in D \times \Omega, \tag{4.1}$$

$$u(x,s) = 0, (x,s) \in \partial D \times \Omega, \tag{4.2}$$

where $\kappa(x,s)$ is a heterogeneous coefficient depending on both the spatial variable $x$ and the parameter $s$, and $f \in L^2(D)$ is a given source. We remark that the differential operators in Equation (4.1) are defined with respect to the spatial variable $x$. This is the case for the rest of the paper.

### 4.2.1 The coarsening of the parameter space. the main idea

The parameter space $\Omega$ is assumed to be of very high dimension (i.e., large $N$) and consists of very large number of realizations. For a given realization, the idea is to find its representation in the coarse space and use the coarse space to perform the computation. We will use the deep cluster learning algorithm to perform the coarsening. Due to the heterogeneous properties of the proposed problem, fine mesh is used; this will bring difficulties in coarsening the parameter space and in computation of the solution. We hence perform the parameter coarsening locally in the space $D$, i.e., we also coarsen the spatial domain. To coarsen the spatial domain, we use coarse grids and

consider the GMsFEM.

In Figure 4.1, we present an illustration of the proposed coarsening technique. On the left figure, the coarse grid blocks in the space are shown. Each coarse grid has a different cluster in the uncertainty space $\Omega$, which corresponds to the coarsening of the uncertainty space. The main objective in multiscale methods is efficiently finding the clustering of the uncertainty space, which is our main goal.

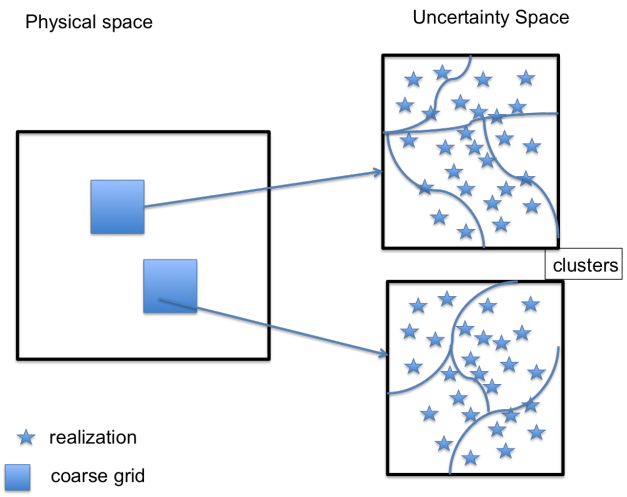


Figure 4.1: Illustration of coarsening of space and uncertainties. Different clusters for different coarse blocks. On the left plot, two coarse blocks are shown. On the right plot, clusters are illustrated. Reprinted with permission from "Learning Algorithms for Coarsening Uncertainty Space and Applications to Multiscale Simulations" by Zecheng Zhang, Eric T. Chung, Yalchin Efendiev and Wing Tat Leung, 2020. Mathematics, 8(5):720, 2020, Copyright [2020] by MDPI.

### 4.2.2 Space coarsening — generalized multiscale finite element methods

It is computationally expensive to capture heterogeneous properties using very fine mesh. For this reason, we use GMsFEM to coarsen the spatial representation of the solution. The coarsening of the parameter space will be performed in each local spatial neighborhood. We will achieve this goal by the GMsFEM, which has been discussed in the introduction of this paper.

### 4.2.3 The idea of the proposed method

We present the general methodology in this section. The target is to save the time in computing the GMsFEM basis $\phi_k^{\omega_i}$ for all $\omega_i$ and for all uncertain space parameters. We propose the clustering algorithm to coarsen the uncertain space in each local neighborhood. The key to the success of the clustering is that: the cluster should inherit the property of the solution, that is, the local heterogeneous fields $\kappa(x, s)$ clustered into the same group should have similar solution properties. When the cluster is learned by the some learning algorithm, the only computation involved is to fit the local neighborhood of the given testing heterogeneous field into some cluster. This is a feed forward process including several convolution operations and matrix multiplications and compared to the direct computing, we save a lot of time in computing the spectral problem in Equation (1.3) and the inverse of a matrix Equation (1.6). The detailed process is illustrated in the following chart (Figure 4.2):

1. (Training) For a given input local neighborhood $\omega_j$, we train the cluster (which will be detailed in next section) of the parameter space $\Omega$ and get the clusters $S_1^j, ..., S_n^j$, where $n$ is the number of clusters and is uniform for all $j$. Please note that we may have different cluster assignments in different local neighborhoods.

2. (Training) For each local neighborhood $\omega_j$ and cluster $S_i^j$, define the average $\bar{\kappa}_{ij}$ and compute generalized multiscale basis for $\bar{\kappa}_{ij}$.

3. (Testing) Given a new $\kappa(x, s)$ and for each local neighborhood $\omega_j$, fit $\kappa(x, s)$ into a $S_i^j$ by the trained network (step 1) and use the pre-computed GMsFEM basis (step 2) to find the solution.

It should be noted that we perform clustering using the heterogeneous fields; however, the cluster should inherit the property of the solution corresponding to the heterogeneous fields. This makes the clustering challenging. The performance of the standard K-means algorithm relies on the initialization and the distance metric. We may initialize the algorithm based on the clustering of

the heterogeneous fields but we need to design a good metric. In the next section, we are going to introduce a learning algorithm which uses an auto-encoder structure and multiple losses to achieve the required clustering task.

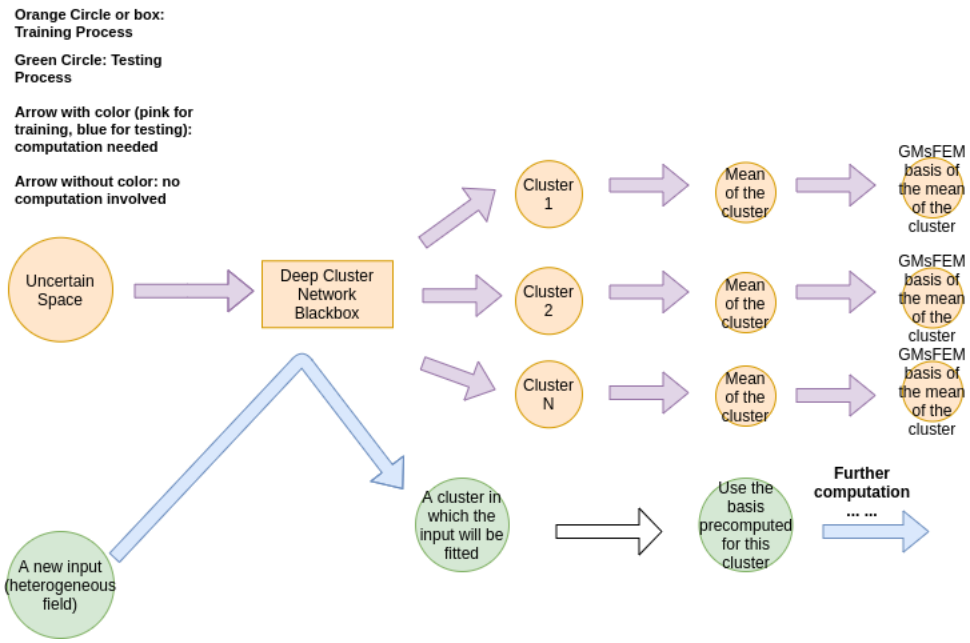


Figure 4.2: Work flow of the proposed method. Reprinted with permission from "Learning Algorithms for Coarsening Uncertainty Space and Applications to Multiscale Simulations" by Zecheng Zhang, Eric T. Chung, Yalchin Efendiev and Wing Tat Leung, 2020. Mathematics, 8(5):720, 2020, Copyright [2020] by MDPI.

## 4.3 Deep learning

The network is consisted of two sub networks. The first one is targeted to performing the clustering and the second one, which is the adversary net, will serve as the reconstruction of loss function.

### 4.3.1 Clustering net

The cluster net is aimed for clustering the heterogeneous fields $\kappa(x, s)$; but the resulting clusters should inherit the properties of the solution corresponding to the $\kappa(x, s)$, i.e., the heterogeneous

fields grouped in the same cluster should have similar corresponding solution properties. This similarity will be measured by the adversary net which will be introduced in Section 4.3.3. We hence design the network demonstrated in Figure 4.3.

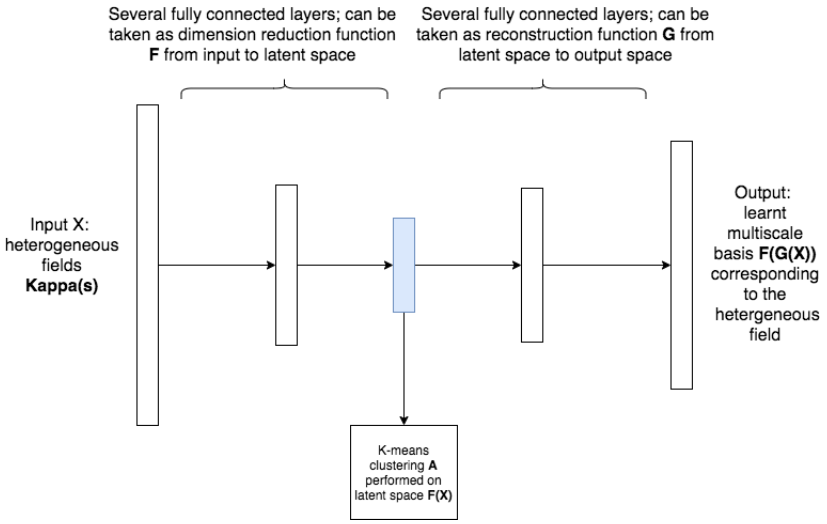


Figure 4.3: Cluster network. Reprinted with permission from "Learning Algorithms for Coarsening Uncertainty Space and Applications to Multiscale Simulations" by Zecheng Zhang, Eric T. Chung, Yalchin Efendiev and Wing Tat Leung, 2020. Mathematics, 8(5):720, 2020, Copyright [2020] by MDPI.

The input $X \in \mathbb{R}^{m,d}$, where $m$ is the number of samples and $d$ is the dimension of one local heterogeneous field, of the network is the local heterogeneous fields which are parametrized by the random variable $s \in \Omega$. The output of the network is the multiscale basis (first GMsFEM basis) which somehow represents the solution corresponding to the coefficient $\kappa(x, s)$. This is a generative network which has an auto encoder structure. The dimension reduction function $F(X)$ can be interpreted as some kind of kernel method which maps the input data to a new space which is easier to be separated. This can also be interpreted as the learning of a good metric function for the later performed K-mean clustering. We will perform K-means clustering algorithm in latent space $F(X)$. $G(\cdot)$ will then transfer the latent space data to the space of multiscale basis function. This process can be taken as a generative process and we reconstruct the basis from the extracted

features. The detailed algorithm is as follow (see Figure 4.4 for an illustration):

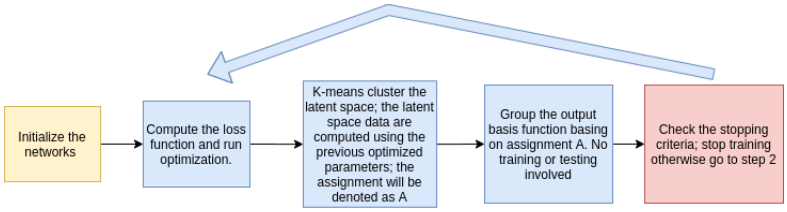


Figure 4.4: Deep learning algorithm. Reprinted with permission from "Learning Algorithms for Coarsening Uncertainty Space and Applications to Multiscale Simulations" by Zecheng Zhang, Eric T. Chung, Yalchin Efendiev and Wing Tat Leung, 2020. Mathematics, 8(5):720, 2020, Copyright [2020] by MDPI.

Steps illustrated in Figure 4.4:

1. Initialize the networks and clustering the output basis function.

2. Compute the loss function $L$ (defined later) and run optimization.

3. Cluster the latent space by K-means algorithm (reduced dimension space, which is a middle layer of the cluster network); the latent space data are computed using the previous optimized parameters; the assignment will be denoted as $A$.

4. Basis functions whose corresponding inputs are in the same cluster (basing on assignment A) will be grouped together. No training or fitting-in involved in this step.

5. Repeat step $2$ to step $4$ until the stopping criteria is met.

### 4.3.2 Loss functions

Loss function is the key to the deep learning. Our loss function is consisted of cluster loss and the reconstruction loss.

1. Clustering loss $C(\theta_F, \theta_G)$: this is the mean standard deviation of all clusters of the learned basis and $\theta$ is the parameters we need to optimize. It should be noted that the loss here is

computed using the learned basis instead of the input of the network. This loss controls the clustering process, i.e., the smaller the loss, the better the clustering in the sense of clustering the multiscale basis. Let us denote $\kappa_{ij}$ as $j$th realization in $i$th cluster; $G(F(\kappa_{ij})) \in \mathbb{R}^d$ will then be $j$th learned basis in cluster $i$ and let $\theta_G$ and $\theta_F$ be the parameters associated with $G$ and $F$, the loss is then defined as follow,

$$C(\theta_F, \theta_G) = \frac{1}{|A|} \sum_{i}^{|A|} \sum_{j}^{A_i} \frac{1}{A_i} \|G(F(\kappa_{ij})) - \bar{\phi}_i\|_2^2, \tag{4.3}$$

where $|A|$ is the number of clusters which is a hyper parameter and $A_i$ denotes the number of elements in cluster $i$; $\bar{\phi}_i \in \mathbb{R}^d$ is the mean of cluster $i$. This loss clearly serves the purpose of clustering the solution instead of the input heterogeneous fields; however, in order to guarantee the learned basis are closed to the pre-computed multiscale basis, we need to define the reconstruction loss which measures the difference between the learned basis and the pre-computed basis.

2. Reconstruction loss $R(\theta_F, \theta_G)$: this is the mean square error of multiscale basis $Y \in \mathbb{R}^{m,d}$, where $m$ is the number of samples. This loss controls the construction process, i.e., if the loss is small, the learned basis are close to the real multiscale basis. This loss will supervise the learning of the cluster. It is defined as follow:

$$R(\theta_F, \theta_G) = \frac{1}{m} \sum_{i}^{m} \|G(F(\kappa_i)) - \phi_i\|_2^2, \tag{4.4}$$

where $G(F(\kappa_i)) \in \mathbb{R}^d$ and $\phi_i \in \mathbb{R}^d$ are learned and pre-computed multiscale basis of $i$th sample $\kappa_i$ separately.

The entire loss function is then defined as $L(\theta_F, \theta_G) = \lambda_1 C + \lambda_2 R$, where $\lambda_1, \lambda_2$ are predefined weights. We are going to solve the following optimization problem:

$$\min_{\theta_G, \theta_F} L(\theta_F, \theta_G) \tag{4.5}$$

for the required training process.

### 4.3.3 Adversary network severing as an additional loss

We have introduced the reconstruction loss which measures the similarity between the learned basis and the pre-computed basis in the previous section. It is the mean square error (MSE) of the learned and pre-computed basis. MSE is a smooth loss and easy to train but there is a well known fact about MSE that this loss will blur the image. In the area of image super-resolution and other low level computer vision tasks, the loss is not friendly to inputs with high contrast and the resulting generated images are usually over smooth. Our problem has multiscale nature and is similar with the low level computer vision task, i.e., this is a generative task; hence blurring and over smoothing should happen if the model is trained by MSE. To define a great reconstruction loss is important.

Motivated by some works about the successful application of deep fully convolutional network (FCN) in computer vision, we design a perceptual loss to measure the error. It is now clear that the lower layers in the FCN usually will extract some general features shared by all objects like the horizontal (vertical) curves, while the higher layers are usually more objects oriented. This gives people the insight to train the network using different layers. Johnson then proposed the perceptual loss [37] which is the combination of the MSE of selected layers of the VGG model [118]. The authors claim in their paper that the early layers tends to produce images that are visually indistinguishable from the input; however if reconstruct from higher layers, image content and overall spatial structure are preserved but color, texture, and exact shape are not.

We will adopt the perceptual loss idea and design an adversary network to compute an additional reconstruction loss. The network structure can be seen in Figure 4.5.

The adversary net is fully convolutional with input and output both pre-computed multiscale basis. The network has an auto encoder structure and is pre-trained; i.e., we are going to solve the

following minimization problem:

$$\min_{\theta_A} \frac{1}{m} \sum_i \|f(\phi_i) - \phi_i)\|_2^2, \tag{4.6}$$

where $\phi_i$ is the multiscale basis and $f$ is the adversary net associated with trainable parameter $\theta_A$. Denote $f_j(\cdot)$ as the output of layer $j$ of the adversary network. The additional reconstruction loss is then redefined as:

$$A(\theta_F, \theta_G) = \frac{1}{m} \sum_{i=1}^{m} \sum_{j \in I} \|f_j(G(F(\kappa_i))) - f_j(\phi_i)\|_2^2, \tag{4.7}$$

where $I$ is the index set which contains some layers of the adversary net. The complete optimization problem can be now formulated as:

$$\min_{\theta_G, \theta_F} \lambda_1 C + \lambda_2 R + \lambda_3 A. \tag{4.8}$$

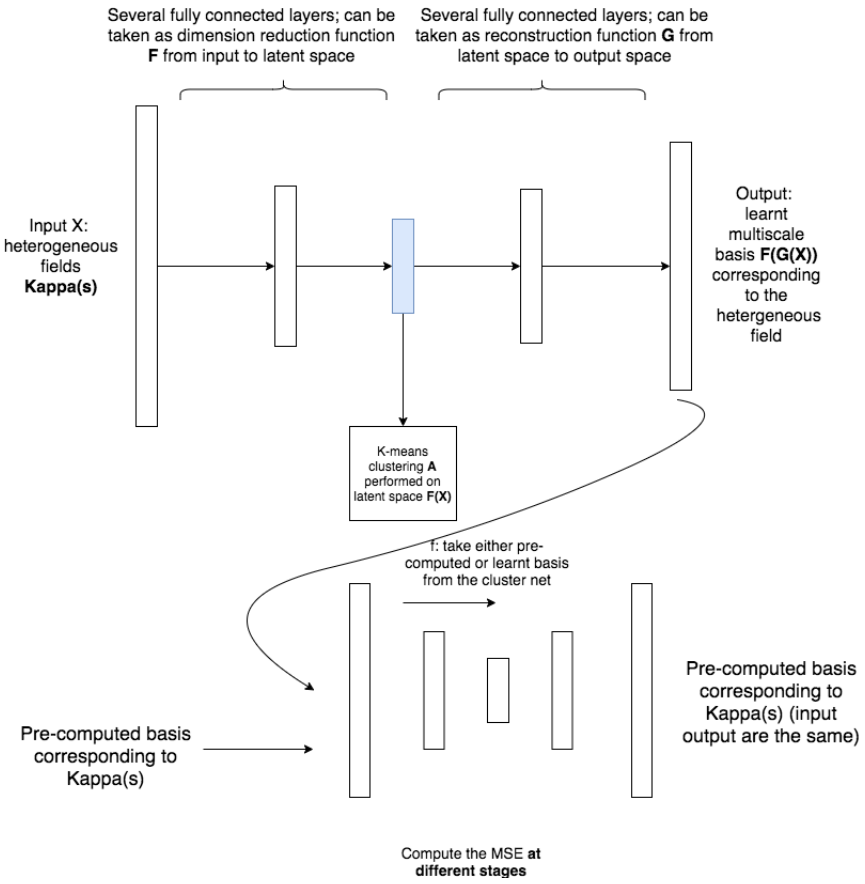


Figure 4.5: The complete network. Reprinted with permission from "Learning Algorithms for Coarsening Uncertainty Space and Applications to Multiscale Simulations" by Zecheng Zhang, Eric T. Chung, Yalchin Efendiev and Wing Tat Leung, 2020. Mathematics, 8(5):720, 2020, Copyright [2020] by MDPI.

## 4.4 Numerical experiments

In this section, we will demonstrate a series of experiments. We are going to apply our method on problems with high contrast including moving background and moving channels. he experiments are related to subsurface simulations. The moving background and moving channels simulate some important characteristics in the field. We numerically generate heterogeneous fields which contain moving channels and varying well rates. In Section 4.4.1, we first demonstrate a set of simulated heterogeneous oil fields to be used to train and solve the PDE modeling the reservoirs simulation; the deep learning model settings are also detailed in this section. In Section 4.4.2,

we simulate some other more complicated heterogeneous fields and conduct the experiments to demonstrate the power of clustering algorithm. This experiments can show that our method is robust to handle complicated problems. In the last section, we will solve the PDE using the proposed method based on the heterogeneous field proposed in Section 4.4.1 and compute the relative error to demonstrate the accuracy of our method.

### 4.4.1 High contrast heterogeneous fields with moving channels

We consider solving Equations (4.1)–(4.2) for a heterogeneous field with moving channels and changing background. Let us denote the heterogeneous field as $\kappa(x)$, where $x \in [0,1]^2$, then $\kappa(x) = 1000$ if $x$ is in some channels which will be illustrated later and otherwise,

$$\kappa(x) = e^{\eta \cdot sin(7\pi x) sin(8\pi y) + sin(10\pi x) sin(12\pi y)},$$

where $\eta$ follows discrete uniform distribution in $[0,1]$. The channels are moving and we include cases of the intersection of two channels and formation and dissipation of the channels in the fields. These simulate the realistic petroleum oil fields. In Figure 4.6, we demonstrate $20$ heterogeneous fields.

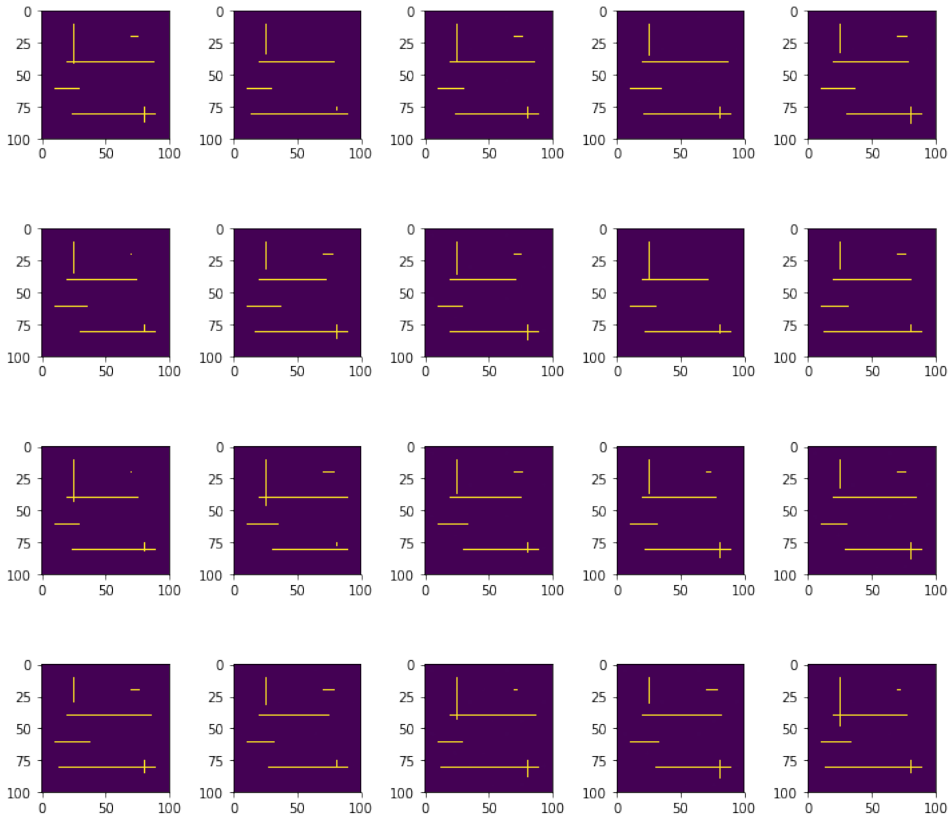

Figure 4.6: Heterogeneous fields, the yellow strips are the channels. Reprinted with permission from "Learning Algorithms for Coarsening Uncertainty Space and Applications to Multiscale Simulations" by Zecheng Zhang, Eric T. Chung, Yalchin Efendiev and Wing Tat Leung, 2020. Mathematics, 8(5):720, 2020, Copyright [2020] by MDPI.

It can be observed from the images that, vertical channel (at around $x = 30$) (not always) intersects with horizontal channels (at around $y = 40$); and the channel at $x = 75, y = 25$ demonstrates the case of generation and degeneration of a channel.

We train the network using 600 samples using the Adam gradient descent. We find that the cluster assignment of $600$ realizations in uncertain space is stable(fixed) when the gradient descent epoch reaches a certain number, so we set the stopping criteria to be: the assignment does not change for $100$ iteration epochs; and the maximum number of iteration epochs is set to be $1000$. We also find that the coefficients in Equation (4.8) can affect the training result. We set $\lambda_1 = \lambda_2 = \lambda_3 = 1$.

It should be noted that we train the network locally in each coarse neighborhood. The fine mesh element has size $1/100$ and $5$ fine elements are merged into one coarse element.

The dimension reduction network $F$ contains $4$ fully connected layers to reduce the size of local coarse elements from $100$ to $60, 40, 30, 20$ gradually. The K-means clustering is conducted in space $F(x)$ of dimension $20$; the reconstruction net $G$ is designed symmetrically with the reduction network $F$. The adversary net is fully convolutional. All convolution layers except the last layer have kernels of size $3$ by $3$ with stride $1$; we use $1$ by $1$ convolution in the last layer to reduce the number of channels to $1$. The number of channels is doubled if the spatial dimension is reduced and half-ed if the spatial dimension is increased. Max pooling of size $2$ by $2$ is used to reduce the spatial dimension in the encoder; and to increase the dimension in the decoder, we perform the nearest neighbor resize followed by convolution [119].

### 4.4.2 Results

We will present the numerical results of the proposed method in this section. We are going to show the cluster assignment experiment first, followed by two other experiments which demonstrate the error of the method.

#### *4.4.2.1 Cluster Assignment in a Local Coarse Element*

Before diving into the error analysis, we will show some of the cluster results in a local neighborhood. In this neighborhood, we manually created the cases such as: the extraction of a channel (longer), the expansion of a channel(wider), the discontinuity of a channel, the diagonal channels, the intersection of channels, and so on. In Figure 4.7, the number on top of each image is the cluster assignment ID number.

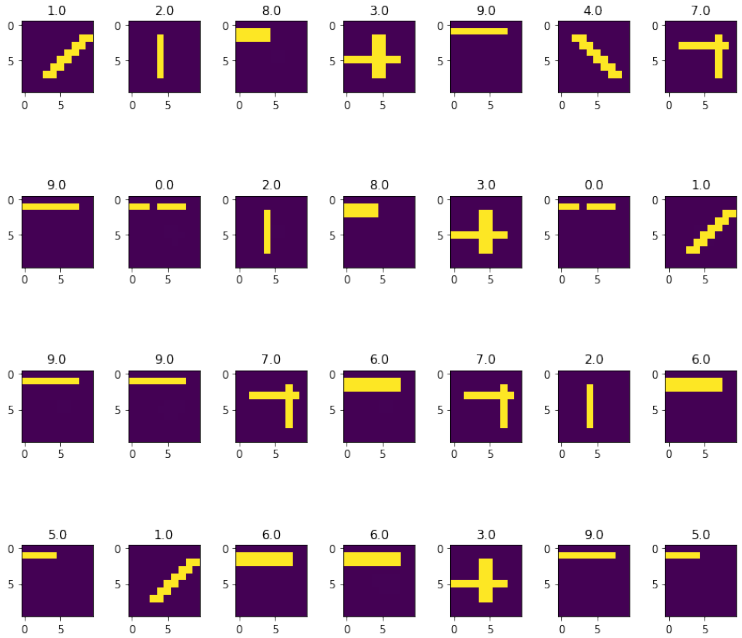


Figure 4.7: Cluster results of 28 samples, images shown are heterogeneous fields, the number on top of each image is the cluster assignment ID number. Reprinted with permission from "Learning Algorithms for Coarsening Uncertainty Space and Applications to Multiscale Simulations" by Zecheng Zhang, Eric T. Chung, Yalchin Efendiev and Wing Tat Leung, 2020. Mathematics, 8(5):720, 2020, Copyright [2020] by MDPI.

We also demonstrate the clustering result in Figure 4.8 of another neighborhood which is around $(25, 45)$ in Figure 4.6. From the results in both Figures 4.7 and 4.8, we observe that our proposed clustering algorithm based on deep learning is able create a good clustering of the parameter space. That is, heterogeneous coefficients with similar spatial structures are grouped in the same cluster.

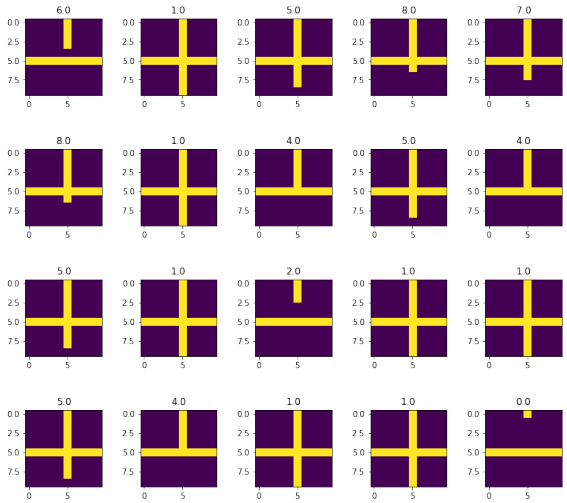


Figure 4.8: Cluster results of 20 samples, images shown are heterogeneous fields, the number on top of each image is the cluster assignment ID number. Reprinted with permission from "Learning Algorithms for Coarsening Uncertainty Space and Applications to Multiscale Simulations" by Zecheng Zhang, Eric T. Chung, Yalchin Efendiev and Wing Tat Leung, 2020. Mathematics, 8(5):720, 2020, Copyright [2020] by MDPI.

#### *4.4.2.2 Relation of error and the number of clusters*

In this section, we will demonstrate the error change when the hyperparamter—the number of clusters—increases. Given a new realization $\kappa(x, \hat{s})$ where $\hat{s}$ denotes the parameter and a fixed neighborhood, suppose the neighborhood of this realization will be fitted into cluster $S_i$ by the model trained. We compute $\bar{\kappa}_i = \frac{1}{|S_i|}\sum_{j=1}^{|S_i|}\kappa_{ij}$ where $|S_i|$ is the number of points in this cluster $S_i$. The GMsFEM basis of this neighborhood can then be derived using $\bar{\kappa}_i$. We finally construct the solution using the GMsFEM basis pre-computed in all neighborhoods. We define the $l_2$ relative error as :

$$ratio = \frac{\int_D (u - u_H)^2 dx}{\int_D u^2 dx}, \tag{4.9}$$

where $u$ is the exact solution computed by finite element method with fine enough mesh and $u_H$ is the solution of the proposed method. We test the network on newly generated $300$ samples and take the average of the errors.

In this experiment, we calculate the $l_2$ relative error with the number of clusters increases.

The number of clusters ranges from $5$ to $11$; and for each case, we will train the model and compute the $l_2$ relative error. The result can be seen in Figure 4.9 and it can be observed from the picture that, the error is decreasing with the number of cluster increases.

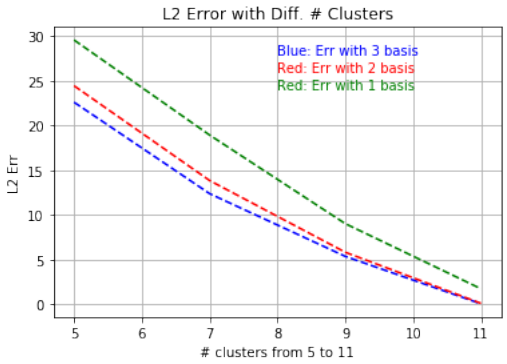


Figure 4.9: The $l_2$ error when the number of clusters changes, colors represent the number of GMsFEM basis. Reprinted with permission from "Learning Algorithms for Coarsening Uncertainty Space and Applications to Multiscale Simulations" by Zecheng Zhang, Eric T. Chung, Yalchin Efendiev and Wing Tat Leung, 2020. Mathematics, 8(5):720, 2020, Copyright [2020] by MDPI.

#### *4.4.2.3 Comparison of cluster-based method with tradition method*

In the second experiments, we first compute the $l_2$ relative error (defined in Equation (4.9) with $u_H$ denoting the GMsFEM solution) of traditional GMsFEM method with given $\kappa(x, \hat{s})$. This means that the construct multiscale basis functions using the particular realization $\kappa(x, \hat{s})$. We then compare this error with the cluster method proposed (11 clusters). The comparison can be seen in Figure 4.10.

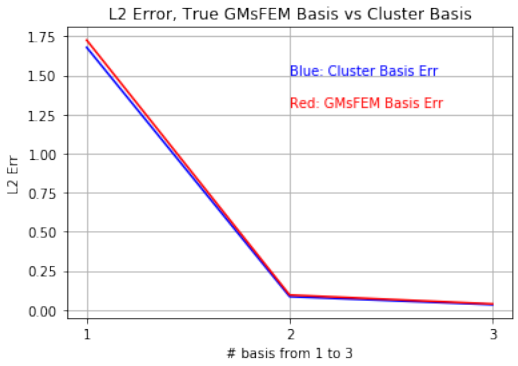


Figure 4.10: The $l_2$ error cluster solution (11 clusters) vs. solution by real $\kappa(x, \hat{s})$. Color represents number of basis. Reprinted with permission from "Learning Algorithms for Coarsening Uncertainty Space and Applications to Multiscale Simulations" by Zecheng Zhang, Eric T. Chung, Yalchin Efendiev and Wing Tat Leung, 2020. Mathematics, 8(5):720, 2020, Copyright [2020] by MDPI.

It can be seen that the difference is negligible when the number of clusters reaches 11. We can then benefit from the deep learning; i.e., the fitting of $\kappa(x, \hat{s})$ into a cluster is fast; and since we will use the pre-computed basis, we also save time on computing the GMsFEM basis.

### 4.4.3 Effect of the adversary net

The target of this task is not the learning of multiscale basis; the multiscale basis in this work is just a supervision of learning the cluster. However, to demonstrate the effectiveness of the adversary network, we also test the the effect of the adversary net. There are many hyper-parameters like the number of clusters and coefficients of the loss function which can affect the result; so to reduce the influence from the clustering, we remove the clustering loss from the training, so this is a generative task which will generate the multiscale basis from the output of the first network in Figure 4.5. The loss function now can be defined as:

$$\min_{\theta_G, \theta_F} \lambda_1 R + \lambda_2 A, \tag{4.10}$$

where $R$ and $A$ are defined in Equations (4.4) and (4.7), separately; and $\lambda_1$ and $\lambda_1$ are both set to be 1. We compute the relative error with Equation (4.9) first by using the learned multiscale basis

which is trained by Equation (4.10); and second by using the multiscale basis trained without the adversary loss Equation (4.7), i.e.,

$$\min_{\theta_G,\theta_F} A. \tag{4.11}$$

The $l_2$ relative error improves from $41.120439$ to $36.760918$ if we add one middle layer from the adversary net.

We also calculate the MSE difference of two learned basis (by loss Equation (4.10) and Equation (4.11), separately) and real multiscale basis, i.e., we calculate $\|B_{\text{learned basis}} - B_{\text{real basis}}\|_{MSE}$, where $B_{\text{learned basis}}$ refers to two basis trained with Equation (4.10) and Equation (4.11), separately and $B_{\text{real basis}}$ is the real multiscale basis formed using the input heterogeneous field. The MSE amazingly decreases from $0.9073400$ to $0.748312$ if we use basis trained with the adversary loss Equation (4.10). This can show the benefit from the adversary net.

# 5. MULTIAGENT REINFORCEMENT LEARNING ACCELERATED MCMC ON MULTISCALE INVERSION PROBLEM

## 5.1 Introduction

The method of Monte Carlo Markov Chain (MCMC) has been widely used in solving inverse problems [120, 121, 122, 123]. There are two issues which limit the computation speed of the MCMC algorithm. The first to mention is the evaluation of the forward solver. To compute the posterior distributions, computationally demanding simulations are needed for evaluating acceptance probabilities driven by forward problems. This issue can be partially solved by applying the multi-level algorithms proposed in [122]. Instead of using one solver, the multi-level algorithms apply different solvers and perform the multi-level acceptances/rejections. If the coarser solver rejects the proposal, then a new proposal is proposed. Since coarse solvers are easy to be evaluated, the rejection process can be accelerated.

Multi-level algorithms can naturally be incorporated with the multiscale inverse problems. This is due to the nature of the multiscale finite element methods [124, 125, 19, 72]. Multiscale basis functions defined on a coarse grid of the computational domain are calculated by solving some local cell problems. Then, we can select different basis functions and use them in coarse forward solvers. In this work, we will use the generalized multiscale finite element methods (GMsFEM) as a forward solver.

Another issue associated with the MCMC is the proposal generator. The earliest methods developed by Metropolis [126] use the random walk to generate the proposal. Many other methods [127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139] including using kernel adaptation, Hamiltonian dynamics, Langevin dynamics, parallel marginalization, kernel coupling and multiple chain simulation, parallelization have been proposed to improve the efficiency of MCMC methods. Deep learning method has been widely applied to solve the problem with multiscale features [50, 41, 52]. In this work, we will recourse to the reinforcement learning approach and propose to use

the reinforcement learning to accelerate the MCMC sampling.

Reinforcement learning (RL) has been studied since the last century. With the development of deep learning, deep reinforcement learning has been applied to solve various problems. Model-free RL algorithm is one of the two most common RL algorithms. It usually includes Q-learning and policy iteration methods (actor critic) [140, 141, 142, 143, 144, 145, 145, 146, 143, 147, 148, 148, 149, 150, 151, 152, 153, 154]. In this work, we are going to study the combination of the policy iteration method with the MCMC algorithms. The target of the policy iteration RL is to learn an acceptable policy such that the expected future reward is maximized. Most of the models have a single agent that is in charge of predicting the behavior; however, there are problems which require the interplay (cooperation, competition) among multiple agents. Many multi-agent algorithms then have been proposed [155, 156, 157, 158, 159]. One of the essential principles of multi-agent design is to centralized the critics but decentralized the actors. This means that the actors make decisions using local information while the critic is centralized learnt using global information.

In this work, we proposed an RL alternative to accelerate the MCMC. The main idea is to use the RL policy as the proposal generator in the MCMC process. More precisely, the RL agent is learning a distribution of actions, and we can make a new proposal by sampling an action from this distribution; that is, the action sampled will modify the current state and we hence obtain the new proposal. The MCMC then comes and either rejects or accepts this proposal as standard. Since the proposal generated may be rejected by MCMC which involves heavy computation, it is natural to use the off-policy algorithm; that is, instead of generating an entire trajectory which is used to learn the critic in the RL, we only run the policy for one time and get a state action state transaction. This saves much time, however, brings in the exploration issue. We hence proposed to apply the $\epsilon$-greedy strategy to overcome this issue. To be more specific, we use the RL policy as the proposal with a certain probability $p$ and use the random walk, which is guaranteed to convergence with probability $1 - p$.

We will verify our method by solving a time-dependent flow inverse problem. Inversely locating the high permeability channels given the measurements is a challenging problem. Our goal is to

identify the heterogeneous permeability field such that the corresponding solution is closed to the measurement (observation). To be more specific, the underlying model of the inversion problem is a parabolic equation as follows:

$$\begin{aligned} u_t - \nabla \cdot (\kappa \nabla u) &= f(x,t) && \text{in } D \times [0,T], \\ u(x,0) &= g(x) && x \in D, \\ \frac{\partial u}{\partial \mathbf{n}} &= h(x,t) && t \in [0,T] \text{ and } x \in \partial D, \end{aligned} \tag{5.1}$$

where $D \subset \mathbb{R}^d$ (with $d \in \{2,3\}$); $\partial D$ denotes the boundary of the domain and $\mathbf{n}$ is the unit outward normal vector; $\kappa(x)$ is a high-contrast permeability field and contains some channelized features. The measurement is the solution of the model at given region and the target is to find the permeability $\kappa(x)$ such that $\|F - \mathcal{F}(\kappa(x))\|$ is small, where $F$ is the observation and $\mathcal{F}(\cdot)$ is the forward solver by which we use to calculate the solution of the input permeability field.

We achieve our goal by sampling a sequence of permeability fields which are characterized as the state in the reinforcement learning framework. The sampling is supervised by the RL agent, and the new sampling will be either rejected or accepted by the MCMC algorithm. We solve a challenging problem with multi-channels and use a multi-agents RL algorithm. The basic idea is to use agents to capture the channels. Each agent is decentralized to learn a policy which will locate one channel. All agents work cooperatively, and the critic is centralized learnt. The more concrete setting will be presented in Section 5.4. To summarize our contribution, we proposed an algorithm which uses multi-agent RL to accelerate the MCMC sampling. We verify our method by solving two challenging inverse problems with the multiscale property.

The rest of the work is organized as follow. In Section 5.2, we will introduce the preliminaries of the works. These include the generalized multiscale finite element method, reinforcement learning and the probabilistic MCMC formulation. Our proposed method is presented in Section 5.3. We conduct several numerical experiments in Section 5.4 to verify our idea.

## 5.2 Background

### 5.2.1 Actor-Critic reinforcement learning

We consider the multi-agent extension of the Markov decision processes (MDPs). A Markov game with $N$ agents is defined by a set of state $\mathcal{S}$ and a set of actions $\mathcal{A}_1, ..., \mathcal{A}_N$ and a set of observations $\mathcal{O}_1, ..., \mathcal{O}_N$ for each agent $i$. To choose an action $a_i$, each agent will follow the policy $\pi_{\theta_i}(a_i|s_i) : \mathcal{O}_i \times \mathcal{A}_i \to [0, 1]$, which is a distribution of the actions given current observation $s_i$; the policy $\pi_{\theta_i}(.)$ depends on the parameter $\theta_i$ and can be formulated as a network. The agents will then move to the next state $s' = \{s'_1, ..., s'_N\}$ by sampling from the transition probability distribution $T(s'|s, a) \in [0, 1]$, where $s$ is the set of current observations and $a = \{a_1, ..., a_N\}$ is the set of actions of each individual agent. Each agent $i$ obtains a reward $r_i(s_i, a_i) : \mathcal{O}_i \times \mathcal{A}_i \to \mathbb{R}$ given the current observation $s_i$ and the action $a_i$ determined by the policy; the reward of all agents can be denoted as $r(s, a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ where $\mathcal{A} = \mathcal{A}_1 \times ... \times \mathcal{A}_N$ is the set of all actions.

We denote the trajectory of length $T$ formed in this Markov process as $\tau : \mathcal{S}\times\mathcal{A}\times\mathcal{S}\times\mathcal{A}...\mathcal{S}\times\mathcal{A}$, the associated probability will then be denoted as $p_\theta(\tau)$. The target of the reinforcement learning is to maximize the expected return:

$$J(\theta) = \mathbb{E}_\tau[R],$$

where $R$ is the reward of the trajectory and $\theta = \{\theta_i, ..., \theta_N\}$ are the parameters associated with the policy of each agent; to be more specific,

$$\max_\theta J(\theta) = \max_\theta E_{\tau\sim p_\theta(\tau)} \left[ \sum_t r(s^t, a^t) \right],$$

where $(s^t, a^t)$ is the state-action pair at step $t$ in the trajectory $\tau$. This optimization process can be implemented by moving in the direction $\nabla J(\theta)$ which is called policy gradient.

There are several approaches to estimate the expected return. The most simple choice is the REINFORCE algorithm proposed by Sutton [153], that is, $R^t = \sum_{j=t}^{T} \gamma^{j-t} r^j$, where $\gamma$ is the

discount factor which determines how much the policy favors the immediate reward over the long-term gain; to be more specific,

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[ \left( \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a^t | s^t) \right) \left( \sum_{t'=t}^{T} \gamma^{t'-t} r(s^{t'}, a^{t'}) \right) \right], \tag{5.2}$$

The REINFORCE algorithm usually leads to high variance and this issue can be ameliorated by approximating the expected return. One choice is to approximate the state value function $Q(s, a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ which is defined recursively [154] as:

$$Q(s, a) = \mathbb{E}_{s'}[r(s, a) + \gamma \mathbb{E}_{a' \sim \pi_\theta}(Q(s', a'))]. \tag{5.3}$$

The policy gradient will then become:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[ \left( \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a^t | s^t) \right) Q(s^t, a^t) \right]. \tag{5.4}$$

By varying the advantage function $A$ (state value function $Q$ and REINFORCE are two special cases), many practical algorithms have been proposed [151]. In this work, we will consider the advantage function defined as:

$$A(s, a) = Q(s, a) - V(s), \tag{5.5}$$

where $V(s) : \mathcal{S} \to \mathbb{R}$ is the state value function which is the expectation of $Q$,

$$V(s) = \mathbb{E}_{a \sim \pi} Q(s, a). \tag{5.6}$$

The approximation of the advantage function have been thoroughly studied in [151]. In this work, we will use:

$$r(s, a) + \gamma V_\omega(s') - V_\omega(s), \tag{5.7}$$

where $V_\omega$ is the approximation of the value function and can be formulated as a network with parameter $\omega$. This approximation is a special case of the $\gamma-$just (unbiased) operator [151] when $V_\omega = V$ and typically has lower variance. We now introduce the basic actor-critic algorithm here:

1. Generate trajectories by policy $\pi_\theta$.

2. Get the updated advantage function $A_\omega$ by learning $V_\omega(.)$ with the trajectories.

3. Update the policy $\pi_\theta$ by optimize the objective $J(\theta)$:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim p_\theta(\tau)}\left[\left(\sum_{t=1}^{T}\nabla_\theta \log \pi_\theta(a^t|s^t)\right)A(s^t,a^t)\right].$$

It should be noted that we use the off-policy algorithm in practice. That is, in step 1 above, we run the policy for one time and get a state action state transaction. This transaction will be saved into reply buffer and we train the value function in step 2 by sampling some transactions from the reply buffer. The off-policy algorithm is fast but may result in the local convergence. We will discuss this issue in detail in the numerical example sections. In this work, we use the actor-critic (AC) algorithm incorporated with the MCMC algorithm. The proposed algorithm will be detailed in later sections.

### 5.2.2 Probabilistic MCMC

This problem can be summarized as sampling the permeability field which has high contrast channels given pressure data with some measurement error. Denote $P(\kappa|F)$ as the conditional probability of the permeability field $\kappa$ given the observation $F$, we then will sample permeability from the $P(\kappa|F)$. By the Baysian' formula, it follows that:

$$p(\kappa|F) \propto P(F|\kappa)\cdot P(\kappa),$$

where $P(\kappa)$ is prior, $P(F|\kappa)$ is the likelihood and $\Pi(\kappa) = p(\kappa|F)$ is the posterior. There are some errors associated with the likelihood function. The first to mention is the error of the observation of

$F$. The second error comes from the forward solver denoted as $\mathcal{F}$ which solves the model problem given the sample permeability $\kappa$. We will assume the total errors follows the normal distribution with standard deviation $\sigma_f$; this gives us that,

$$P(F|\kappa) \propto exp\{-\frac{\|F - \mathcal{F}(\kappa)\|_2^2}{\sigma_f^2}\} \tag{5.8}$$

Sampling from $\Pi(\kappa)$ can be accomplished by the MCMC methods. In this work, we will build the algorithm basing on the Metropolis-Hasting algorithm [127]. It should be noted that in the computation of the posterior $\Pi(\kappa)$, intense computation will be involved in evaluating the forward solver $\mathcal{F}$. To improve the efficiency of computing the acceptance rate and give fast rejection, we adopted the multi-level MCMC algorithm basing on the GMsFEM.

We use GMsFEM coarse solvers $\mathcal{F}_1, ..., \mathcal{F}_L$ defined in Section (1.1); then the corresponding posterior will be denoted as:

$$\Pi_l \propto exp\{-\frac{\|F - \mathcal{F}_l(\kappa)\|_2^2}{\sigma_f^2}\}P(\kappa). \tag{5.9}$$

We use $q(y|x)$ as the proposal generator in which $x$ is the current sampling and $y$ is the proposal. The multi-level MCMC algorithm is detailed at the end of this chapter (5.2.3).

### 5.2.3 Multi-level MCMC algorithm

**Algorithm 1:** MLMCMC

1 Set the total number of levels $L$;

2 Given input $s^m$, sample a new proposal $c$ from the proposal generator $q(.|s^m)$;

3 Compute the acceptance probability:

$$\rho_0(c, s^m) = \min\left\{1, \frac{\Pi_0(c)q(s^m|c)}{\Pi_0(s^m)q(c|s^m)}\right\}$$

4 Set $c_0 = c$ with the probability $\rho_0(c, s^m)$ and $c_0 = s^m$ with the probability $1 - \rho_0(c, s^m)$;

5 **for** $l = 1, ..., L$ **do**

6 Compute the acceptance probability by,

$$\rho(c_{l-1}, s^m) = \min\left(1, \frac{\Pi_{l-1}(s^m)\Pi_l(c_{l-1})}{\Pi_{l-1}(c_{l-1})\Pi_l(s^m)}\right)$$

Set $c_l = c_{l-1}$ with the probability $\rho(c_{l-1}, s^m)$ and $c_l = s^m$ with the probability $1 - \rho(c_{l-1}, s^m)$;

7 **end**

8 Set $s^{m+1} = c_L$ and return $s^{m+1}$;

**Remark 5.2.1.** *It should be noted that once the proposal is rejected at some level, the following levels computation can be avoided since the accept probability is equal to* $1$ *and* $c_l = s^m$ *for all following* $l$*. This means that the proposal will be rejected by coarser solver which requires less computation time.*

## 5.3 Proposed approach

The traditional MCMC algorithm can be accelerated by multi-level MCMC since the multi-level MCMC uses coarse solver which gives faster rejection computation; however MCMC can be further improved by modifying the proposal generator. We hence proposed the multi-agent reinforcement approach.

### 5.3.1 Explanations of the algorithm

Each channel in the target permeability field can be parametrized and taken as observations ($s_i$) in the RL algorithm; and each agent is in charge of chasing for one target channel of quadrilateral shape. Actions ($\mathcal{A}_i$) are defined so that each rectangle can be reshaped or moved; and action distribution of each agent is determined by the policy $\pi_{\theta_i}(.|s_i)$. Detailed setup can be seen in Section 5.4.1.

To be more clear, each agent is like a robot; at the beginning of the simulation, each robot stands at some position in the field, then they can transform and move individually so that the output of the forward solver at the end of simulation is closed to the real observation. The pseudo algorithm has 3 iterative steps and please check Appendix 5.3.3 for the details of the algorithm:

1. Each RL agent return a new local proposal for its own channel according to its own policy; make a new state by combining all proposals.

2. Apply standard multilevel MCMC to accept or reject the new state. If the new state is rejected, go back to step 1; otherwise proceed to the step 3.

3. Update the RL agents (policy and critic) using the past trajectory offline and then go back to step 1.

The first step above is the sampling process in MCMC algorithm and we replace the random proposal $q(y|x)$ by the reinforce. More precisely, the sampling (for a single agent) is achieved by following three steps below:

1. Given current state $s_i$.

2. Sample an action $a_i$ from the policy $\pi_{\theta_i}(.|s)$.

3. Execute action $a_i$ to get the next state $s_i'$.

That is, $q(s_i'|s_i) = \pi_{\theta_i}(a_i|s_i)$, such that $s_i'$ is acquired by taking action $a_i$ at state $s_i$.

### 5.3.2 Discussions of the algorithm

There are some key notes of the algorithm.

1. Due to the evaluation of the forward solvers in the MCMC step, it is not realistic to run the policy multiple times and get a long enough trajectory to update the RL agents (step 3). We hence apply the off-policy algorithm as discussed before. This may result in the local convergence since the RL agent may never explore some regions in the solution space and we are learning the value function greedily. This exploitation (or the under exploration) issue will result in the early stopping in our problem. More precisely, proposals with large probabilities suggested by the RL policy will be very likely rejected by MCMC; meanwhile, the low probabilities actions are easily to be accepted by MCMC but are hard to be sampled. Consequently, the sampling process becomes slow and we call this phenomenon "early stopping". We will demonstrate this phenomenon in details later on the numerical example sections; but to solve this issue, we consider the $\epsilon$-greedy strategy. That is, we use the RL policy as the proposal with a certain probability $p$ and use the random walk proposal which is guaranteed to convergence with probability $1 - p$.

2. We use the the decentralized actor and centralized critic principle. To be more specific, the policy $\pi_{\theta_i}(.|s_i)$ of each agent $i$ predict the action distribution depending only on the local observation $s_i$. Once agent $i$ moves to new observation $s_i'$ and obtains the new state $\hat{s}' = \{s_1', ..., s_i', s_{i+1}, s_N\}$, we can evaluate the reward $r$ using the global information $\hat{s}'$; the critic value function $V_i(.|\omega_i)$ will then be updated accordingly. Intuitively, the centralized critic is able to enhance the cooperation among all agents. The expected value reward ($V_i(|\omega_i)$) for one observation-action pair (of one agent) relies on the global state information; hence the agent will not make decision greedily only improving its own reward, instead, individual move will be beneficial to the global reward. In our problem, it is natural to evaluate the forward solver using global information, that is, we need all channels in order to compute the reward; hence the centralized critic should be a good choice.

3. (Intuition of the idea) Instead of moving to a state greedily, the RL algorithm is supposed to move to a state which has better expected reward to go. This is accomplished by the critic learning in the RL iteration. Hence, the proposal generated by the RL agent should give faster convergence than the random sampling.

### 5.3.3 Detailed algorithm

We first initialize the algorithm. Please check algorithm in Algorithm (2).

1. Set the max length of the trajectory $T$
2. Initialize the critic net $V_l(.|\omega_l)$. the policy network $\pi_{\theta_l}(,|s)$ for all agent $l = 1, ..., C$
3. Initialize the target net $\hat{V}_l(.|\omega_l')$ by setting $\omega_l' \leftarrow \omega_l$
4. Initialize the reply buffer $\mathcal{D}$
5. Initialize the starting state $s^t = (s_1^t, ..., s_C^t)$

**Remark 5.3.1.** *We have several remarks regarding the algorithm. We need to call MLMCMC algorithm to generate samples; however, in the implementation of the MLMCMC algorithm, the proposal generation is completed in* 2 *steps. First, we can sample an action from* $\pi_{\theta_l}(.|s_l^t)$*; and then we execute the action to get the proposal.*

*To initialize the reply buffer, we randomly generate samples* $(s^t, r, s^{t+1})$*; to get the reward of the transaction, we use the most coarse forward solver to evaluate the solution. This saves a lot of time in generating samples and the performance is not compromised.*

## 5.4 Numerical examples

In this section, we are going to demonstrate 3 sets of the numerical experiments. The first experiment (Section (5.4.2), (5.4.3), (5.4.4), (5.4.6) ) and the third experiments (Section (5.4.7)) are aimed at showing the efficiency of the proposed method; the second experiment in section (5.4.5) is used to explain the early stopping of the RL method. The RL setup is presented in Section (5.4.1).

**Algorithm 2:** Replay Buffer A2C with MCMC

**for** $t = 0, ..., T$ **do**
  **for** $l = 0, ..., C$ **do**
    **while** $s_l^t == s_l^{t+1}$ **do**
      Run $c = MLMCMC(s_l^t)$ with the proposal generator $q = \pi_{\theta_l}$;
      Set $s_l^{t+1} = c$;
    **end**
    Random select $S$ samples $\{(\mathbf{s}_j, r_j, \mathbf{s}'_j)\}_{j=1}^S$ from the reply buffer $\mathcal{D}$; the sample of agent $l$ will be denoted as $(s_{l,j}, s'_{l,j})$;
    Calculate the target for each sample by $V_{tar,l}(s_{l,j}|\omega'_l) = r(\mathbf{s}_j, \mathbf{s}'_j) + \gamma \hat{V}_l(s'_{l,j}|\omega'_l)$ using the target network and the selected samples;
    Update the critic network $V_l(\omega_l)$ (over the batch) by optimizing,

$$L_{V_l}(\omega_l) = \frac{1}{S}\sum_{j=1}^{S}(V_l(s_{l,j}|\omega_l) - V_{tar,l}(s_{l,j}|\omega'_l))^2$$

    Calculate the advancement $A_l(s_{l,j}, s'_{l,j}) = r(\mathbf{s}_j, \mathbf{s}'_j) + \gamma V_l(s'_{l,j}|\omega_l) - V_l(s_{l,j}|\omega_l)$ based on the critic network for all samples in the batch;
    Update the policy $\pi_{\theta_l}$ by optimizing

$$L_{\pi_l}(\theta_l) = \frac{1}{S}\sum_{j=1}^{S}(-A_l(s_{l,j}, s'_{l,j}) \log \pi_{\theta_l}(s'_{l,j}|s_{l,j}))$$

    Update the target network by:

$$\omega'_l \leftarrow \epsilon\omega'_l + (1-\epsilon)\omega_l$$

  **end**
  Set $s^{t+1} = (s_1^{t+1}, ..., s_C^{t+1})$ and store $(s^t, r(s^t, s^{t+1}), s^{t+1})$ in the buffer;
**end**

### 5.4.1 Reinforcement learning setup

In this section, we will model our problem under the reinforcement learning framework. There are 3 basic elements that we need to define to model our problem: $\mathcal{S}$, $\{\mathcal{O}\}_{i=1}^{N}$, $\{\mathcal{A}\}_{i=1}^{N}$ and the reward $r(s, a)$.

The target of the inversion problem is to find the high contrast channel such that the observation loss is minimized. In the framework of the RL, each agent is in charge of one channel and moves to the target channels step by step. To simplify the setting of the problem, we assume all channels have quadrilateral shape and we hence characterize a channel by a tuple of 4 parameters $(x, y, w, d)$, where $(x, y)$ is the coordinate of the lower left vertex of the channel while $(w, d)$ stands for the width and height. The observation of each agent is then defined as:

$$\mathcal{O}_i = \{(x, y, w, d)\}, i = 1, ..., N \tag{5.10}$$

and the state is consisted of all observations and is then defined as:

$$\mathcal{S} = \{(x_1, y_1, w_1, d_1), ..., (x_N, y_N, w_N, d_N)\}, \tag{5.11}$$

where $N$ is the number of channels. Since we are using the decentralized actor, each agent will modify its prediction basing on the local information; we hence define the 8 actions for each agent. Also, we assume the transition is deterministic and then the state and action can be summarized as

follow:

$$
\begin{aligned}
&T\big((x-h,y,w,d)|(x,y,w,d),a\big)=1, a=\text{shift to the left}\\
&T\big((x+h,y,w,d)|(x,y,w,d),a\big)=1, a=\text{shift to the right}\\
&T\big((x,y-h,w,d)|(x,y,w,d),a\big)=1, a=\text{shift downwards}\\
&T\big((x,y+h,w,d)|(x,y,w,d),a\big)=1, a=\text{shift upwards}\\
&T\big((x,y,w-h,d)|(x,y,w,d),a\big)=1, a=\text{squeeze horizontally}\\
&T\big((x,y,w+h,d)|(x,y,w,d),a\big)=1, a=\text{stretch horizontally}\\
&T\big((x,y,w,d-h)|(x,y,w,d),a\big)=1, a=\text{squeeze vertically}\\
&T\big((x,y,w,d+h)|(x,y,w,d),a\big)=1, a=\text{stretch vertically},
\end{aligned}
$$

where $h$ is the size of one fine element. We assume the deterministic transition, hence define the reward as

$$r(s,a)=r(s,s')=\|\mathcal{F}-\mathcal{F}_L(s)\|-c_1\|\mathcal{F}-\mathcal{F}_L(s')\|+c_2,$$

where $(s,s')$ are the current and proceeding states respectively; $c_1$ and $c_2$ are two hyper-parameters to set; $\mathcal{F}_L$ is the coarse scale forward solver and $\mathcal{F}$ is the observation. Throughout all the experiments, the observation is the pressure at wells.

We random choose the starting position for each channel and run the algorithm. At each step, we compute $\|\mathcal{F}-\mathcal{F}_L(s)\|$. We will demonstrate the convergence of the multilevel MCMC (MCMC), MCMC improved by RL (RLMCMC) and an $\epsilon-greedy$ update version of the RLMCMC algorithm (eRLMCMC).

### 5.4.2 The first experiment setup

The target permeability field with two high contrast channels are shown in Figure (5.1). The source with 2 injection wells and 2 production wells is defined as following (see Figure (5.1) for

the illustration):

$$f(x) = \begin{cases} 20, x \in [0.1, 0.2] \times [0.1, 0.2], \\ -5, x \in [0.8, 0.9] \times [0.1, 0.2], \\ 20, x \in [0.2, 0.3] \times [0.8, 0.9], \\ -5, x \in [0.75, 0.85] \times [0.55, 0.65] \end{cases} \tag{5.12}$$

The solution of the system is shown in Figure (5.1) and we obtain the solution using the fine mesh ($h = 1/100$). It should be noted that we only use the data at the given wells as the observation data.

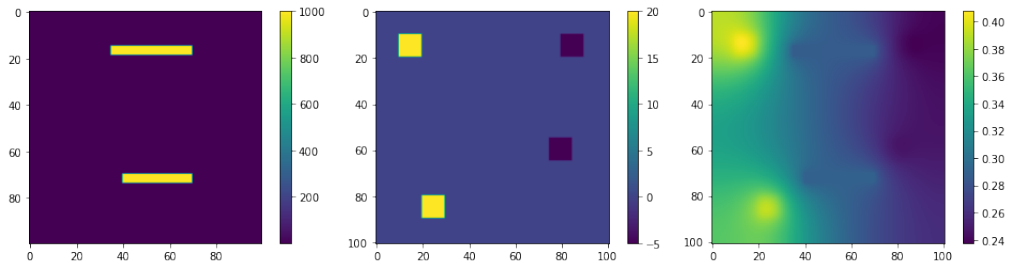

Figure 5.1: Left: the target permeability field. The permeability of the field at the channels is equal to 1000 and is equal to 1 otherwise. Middle: source of the system. Right: corresponding solution.

### 5.4.3 Multilevel MCMC

We will first demonstrate that the conventional multilevel MCMC algorithm (MCMC). It should be noted that we do not use prior in the algorithm, that is, $P(\kappa) = 1$ in the (5.9). Also, we use the uniform proposal. Please see Figure (5.2) for the convergence of the MCMC method. In Figure 5.2, the $x$-axis is the sampling steps and please note that, only the accepted samplings are recorded; while the $y$-axis represents the difference between the measurement and the solution.

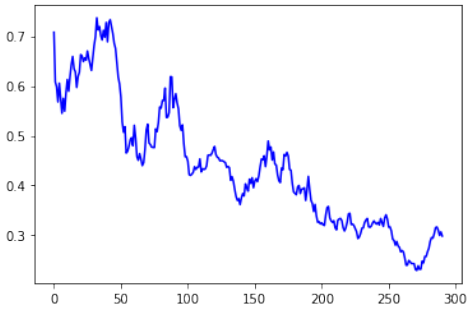


Figure 5.2: Convergence of the Multilevel MCMC. The Y axis is: $\|\mathcal{F} - \mathcal{F}_L(s)\|$ and X axis is the steps of the sampling.

Our large scale experiments show that the convergence speed of the traditional method is slow when compared to the proposed method which is shown in the next section.

### 5.4.4 RL accelerated MCMC (RLMCMC)

In this experiment, we accelerate the conventional MCMC by the reinforcement learning. To be more specific, the proposal is replaced by the RL agent policy. All the other hyper-parameters will be kept the same as the previous experiment. Please check Figure (5.3, left) for the convergence of the RL-MLMCMC; the comparison of two method can be seen in Figure (5.3, right).

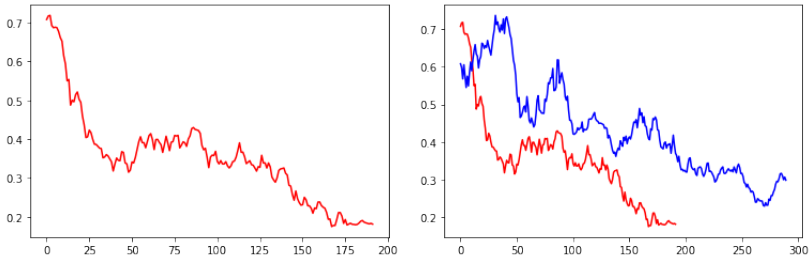


Figure 5.3: Left: the convergence of the the RL-MCMC method. Right, the comparison of the MCMC method and the proposed method. RED: Proposed, BLUE: MCMC.

The CPU time for the computation can be seen in Table (5.1). We can see the faster convergence of the RLMCMC method; however, we also observe the early stopping of the RLMCMC, that is,

almost all new proposals proposed by RL agents will be rejected by the MCMC algorithm. In this experiment, this phenomenon happens at around $100$ steps. This will slow down the convergence of the method. We will explain the reason of this in the next section and give an update of the method in section (5.4.6).

### 5.4.5 Early stopping

We observed the early stopping phenomenon of the RLMCMC; that is, almost all new proposals suggested by the actors are rejected by the MCMC algorithm. The good news is, our large scale experiments show that the stopping predictions have satisfying results and this stopping can be controlled by the adjusting the learning rate. Please check the Figure (5.3, left).

The direct reason of the early stopping is: the probabilities for two actions which are in a pair (eg, move to the left and move the right) have multi scales as RL learning goes. By the formula of the calculating the acceptance rate (1), the acceptance rate becomes extremely low and hence the new proposal which is sampled with large probability suggested by the RL agent is rejected. This causes the slow sampling process.

The fundamental reason of the multi-scales probabilities is the RL exploration problem. We use the off-policy RL, that is, the value function is updated from samples in the reply buffer. The benefits of the off-policy strategy is: we do not need to run the policy for multiple times and get a trajectory, which is super time consuming. That means if the agent never explores a region, it has no information about that region and hence makes decision only depending on the region it has explored. To explain this issue better, let us consider a simpler model.

The target permeability has only one channel (Figure (5.4, left)); and we have 1 injection well

and 3 production wells (Figure (5.4), middle), that is, source is defined as:

$$f(x) = \begin{cases} -5, x \in [0.45, 0.55] \times [0.2, 0.3], \\ -5, x \in [0.45, 0.55] \times [0.7, 0.8], \\ 20, x \in [0.1, 0.2] \times [0.45, 0.55], \\ -5, x \in [0.8, 0.9] \times [0.45, 0.55] \end{cases} \tag{5.13}$$

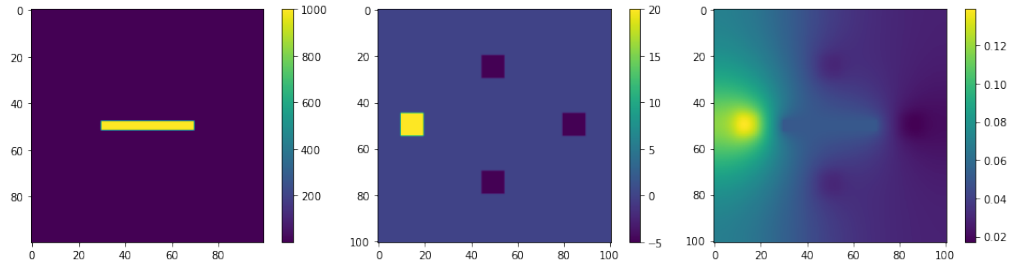

Figure 5.4: One channel example. Left: the target permeability field. The permeability of the field at the channels is equal to 1000 and is equal to 1 otherwise. Middle: source of the system. Right: corresponding solution.

In this example, we assume we know $(y, w, d)$ and only $x$ is unknown, that means the agent only needs to move to the right or left. We use one agent. State (observations) will be formulated same as before; however, the action space contains two actions: move to the left and move to the right. To better demonstrate the probability evolution and avoid the eraly stopping caused by the MCMC rejection, we use only RL to make the sampling and there is no MCMC involved.

The guess starting position will be to the right of the target place. This means that the agent only has information of moving to the left; hence the probability of moving to the left increases drastically before the agent passes over the target position; the multiscale in probabilities happens

then. In the formula of the acceptance rate, $q(s^m|c)$ is the probability of moving to the right which is small compared to $q(c|s^m)$ which is moving to the left. This results in the low acceptance rate. Please check Figure (5.5) for the illustration of the evolution of the probability distribution.

Our experiments show that the early stopping prediction has been good enough when compared to the traditional MCMC algorithm; however, this brings the under-exploration issue to the RL algorithm. To solve this issue, we employ the $\epsilon - greedy$ method to upgrade the RLMCMC algorithm.

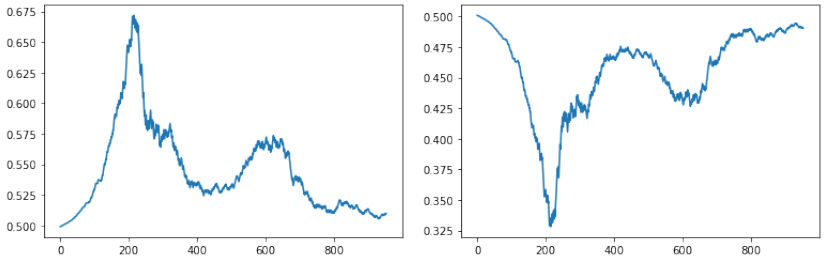


Figure 5.5: Left: Probability of going left. Right: Probability of going right. In both graphs, the y axis is the probability and x axis is the training step.

### 5.4.6 $\epsilon - greedy$ RL-MCMC (eRLMCMC)

To solve the early stopping issue the RL-MCMC method, we proposed the $\epsilon - greedy$ RL-MCMC (eRLMCMC) method. The idea is to apply the $\epsilon - greedy$ strategy and mix the RL-MCMC and MCMC method. To be more specific, in each step, we use RL policy as proposal with the probability $\epsilon$ and use uniform policy with probability $1 - \epsilon$.

Please check Figure (5.6, Left) for the convergence of eRLMCMC; and the computation time is shown in Table (5.1).

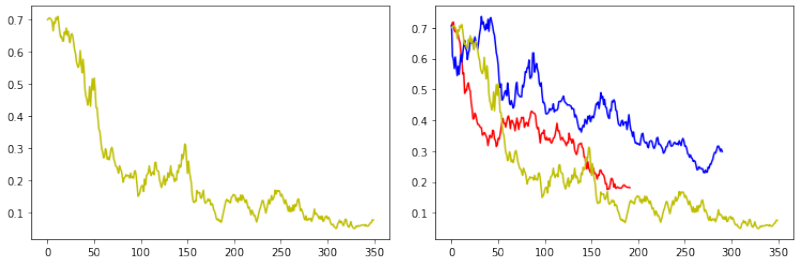


Figure 5.6: Left: the convergence of the the eRLMCMC method. Right, the comparison of all 3 methods. RED: RLMCMC, BLUE: MCMC, YELLOW: eRLMCMC

| Method | Number of Steps | Time |
|---|---|---|
| MCMC | 289 | 8060.6s |
| RLMCMC | 191 | 7274.6s |
| eRLMCMC | 349 | 9680.4s |

Table 5.1: Computation time of the first example

In this table, we compared the computational time for all three methods. It should be noted that the average step time of the RLMCMC method (proposed method) is larger than the that of the traditional method. This is because of the computations needed in training the RL agent. However, we can observe that the RLMCMC method has shorter total computation time and better result (see Figure 5.6 (Right)); this shows that the proposals suggested by the RL agent are meaningful and can lead to the convergence in a more efficient way. Similar results can be seen in the second examples Table 5.2. Our method hence works.

### 5.4.7 Test on diagonal channels

In the first set of experiments, we assume the underlying channels are horizontal or vertical; however there are cases in the real applications that the channels are diagonal. In this set of

experiments, we will test more challenging example with diagonal channels; however, the state and action formulations are kept the same as before, that is, we still use the rectangles which are parallel with the domain boundaries as the predicting states (observations).

We use the same source term but the channels are diagonal. Please check Figure (5.7) for the details.

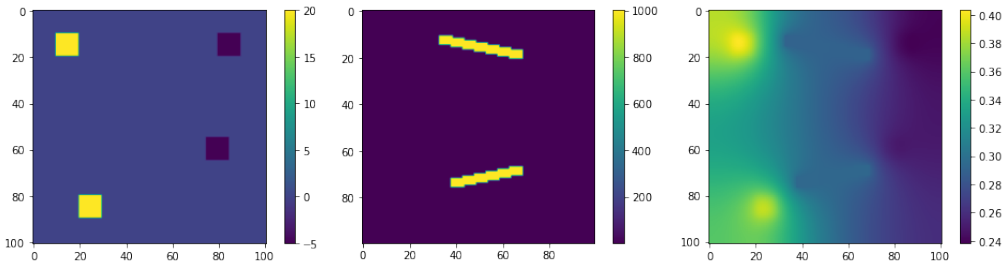

Figure 5.7: Problem with the diagonal channels. Left: the target permeability field. The permeability of the field at the channels is equal to 1000 and is equal to 1 otherwise. Middle: source of the system. Right: corresponding solution.

Same as before we will compute $\|\mathcal{F} - \mathcal{F}_L(s)\|$ and demonstrate the results of applying MCMC, RLMCMC and eRLMCMC. Please check Figure (5.8) for the individual results of three methods.

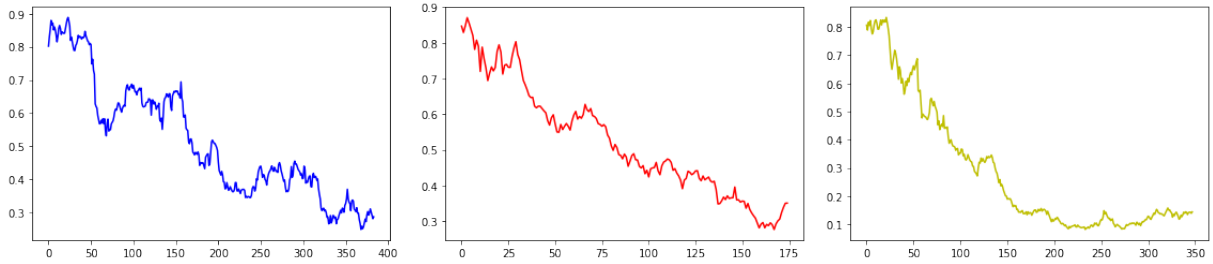

Figure 5.8: Left: MCMC result. Middle: MCMC improved by RL (RLMCMC). Left: $\epsilon - greedy$ strategy RLMCMC

We also observe the early stopping of the RLMCMC method and hence use the $\epsilon - greedy$ strategy to improve the RL algorithm. The comparison of three methods are shown in Figure (5.9).

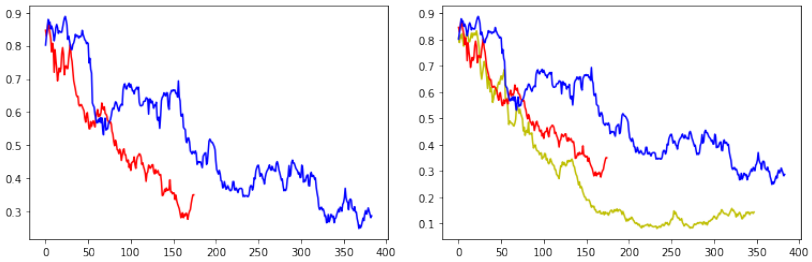

Figure 5.9: Left: MCMC vs RLMCMC; blue curve is the MCMC and red curve is the RLMCMC. Left: MCMC vs RLMCMC vs eRLMCMC; blue curve is the MCMC, red curve is the RLMCMC and yellow is the eRLMCMC.

We can observe from Figure (5.9) that the RLMCMC does improve the convergence speed of the MCMC method. The $\epsilon - greedy$ strategy further improves the result by extending the sampling process. The comparison of the computational time is shown is in Table (5.2).

| Method | Number of Steps | Time |
|---|---|---|
| MCMC | 383 | 8048s |
| RLMCMC | 174 | 6062.4.6s |
| eRLMCMC | 347 | 13472.4.4s |

Table 5.2: Computation time of the third example (Diagonal channels)

From Table 5.2, we can see that our methods still work. One may notice the long total computation time of the eRLMCMC method. This happens because of the rejections. We apply the $\epsilon$-greedy strategy; however, setting $\epsilon$ is tricky. In this experiment, we set a relative large $\epsilon$. This means more rejections will be suggested by the MCMC and hence results in the long computation time; but this strategy indeed extends the trajectory and gives us a better convergence result.

# 6. SUMMARY AND CONCLUSIONS

Let us now conclude the dissertation. We first discuss the quasi gas dynamic (QGD) (in chapter 2) in the multiscale setting. This model is later used in the paraxial approximation project.

In this work, we have proposed a novel computational multiscale method based on the idea of constraint energy minimization for solving the problem of quasi-gas-dynamics. The spatial discretization is based on constrain energy minimization generalized multiscale finite element methods (CEM-GMsFEM) which provides a framework to systematically construct multiscale basis functions for approximating the solution of the model. The multiscale basis functions with locally minimal energy are constructed by employing the techniques of oversampling, which leads to an improved accuracy in the simulations. Combined with the central difference scheme for the time discretization, we have shown that the fully discrete method is stable under a relaxed version of CFL condition and has optimal convergence rates despite the heterogeneities of the media. Numerical results have been presented to illustrate the performance of the proposed method.

Secondly, in chapter 3, we discuss a new approach to solve a full equation in the multiscale environment. Instead of solving the full wave equation, we apply the paraxial approximation which results in a QGD equation give a certain spatial level. For the spatial discretization, we employed the CEM-GMsFEM methods, which is proved to be efficient to reduce the dimension of the model in space. We then combined the technique of POD to further reduce the dimension along the quasi time direction. A complete analysis of the proposed algorithm has been provided. Numerical results are provided to demonstrate the effectiveness and efficiency of the proposed method.

Deep learning has been showing its power in solving the computer vision, language processing and many other problems since 2010. In this dissertation, we also present two works in which we use the deep learning techniques to solve the traditional multiscale problems.

In chapter 4, we propose a deep learning clustering technique within generalized multiscale finite element methods (GMsFEM) to solve flows in heterogeneous media. The main idea is to cluster the uncertainty space such that we can reduce the number of multiscale basis functions for

each coarse block across the uncertainty space. We propose the adversary loss motivated by the perceptual loss in the computer vision task. We use convolutional neural networks combined with some techniques in adversary neural networks, where the loss function is composed of several parts that includes terms related to clusters and reconstruction of basis functions. We present numerical results for channelized permeability fields in the examples of flows in porous media.

Reinforcement learning was firstly studied in the late 1980‘s. With the development of the deep learning, deep reinforcement learning becomes a powerful tool in the control theory and computer science applications. We use the deep reinforcement learning in the last project presented in this dissertation.

In chapter 5, we proposed to use the RL agent as a proposal generator in the MCMC algorithm. We invented a multi-agents actor critic algorithm which will accelerate the MCMC sampling. We verify our idea in solving an inverse problem with multiscale. As far as we know, this is the first time, RL is used in the combination with the MCMC sampling. Many experiments are provided and showed the success of our method.

# REFERENCES


[1] E. Cances, V. Ehrlacher, F. Legoll, and B. Stamm, "An embedded corrector problem to approximate the homogenized coefficients of an elliptic equation," *Comptes Rendus Mathematique*, vol. 353, no. 9, pp. 801–806, 2015.

[2] J. Chen, S. Sun, and Z. He, "Homogenize coupled stokes–cahn–hilliard system to darcy's law for two-phase fluid flow in porous medium by volume averaging," *Journal of Porous Media*, vol. 22, no. 1, 2019.

[3] J. Chen, S. Sun, and X. Wang, "Homogenization of two-phase fluid flow in porous media via volume averaging," *Journal of Computational and Applied Mathematics*, vol. 353, pp. 265–282, 2019.

[4] S. Fu, E. Chung, and G. Li, "Edge multiscale methods for elliptic problems with heterogeneous coefficients," *Journal of Computational Physics*, 2019.

[5] C. Le Bris, F. Legoll, and A. Lozinski, "An MsFEM type approach for perforated domains," *Multiscale Modeling & Simulation*, vol. 12, no. 3, pp. 1046–1077, 2014.

[6] C. Le Bris, F. Legoll, and F. Thomines, "Multiscale finite element approach for weakly random problems and related issues," *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 48, no. 3, pp. 815–858, 2014.

[7] A. Salama, S. Sun, M. F. El Amin, Y. Wang, and K. Kumar, "Flow and Transport in Porous Media: A Multiscale Focus," *Geofluids*, vol. 2017, 2017.

[8] H. Hajibeygi, D. Kavounis, and P. Jenny, "A hierarchical fracture model for the iterative multiscale finite volume method," *Journal of Computational Physics*, vol. 230, no. 4, pp. 8729–8743, 2011.

[9] T. Hou and X. Wu, "A multiscale finite element method for elliptic problems in composite materials and porous media," *J. Comput. Phys.*, vol. 134, pp. 169–189, 1997.

[10] P. Jenny, S. Lee, and H. Tchelepi, "Multi-scale finite volume method for elliptic problems in subsurface flow simulation," *J. Comput. Phys.*, vol. 187, pp. 47–67, 2003.

[11] P. Jenny, S. Lee, and H. Tchelepi, "Adaptive multi-scale finite volume method for multi-phase flow and transport in porous media," *SIAM J. Multiscale Modeling and Simulation*, vol. 3, pp. 30–64, 2004.

[12] E. Chung, Y. Efendiev, and C. Lee, "Mixed generalized multiscale finite element methods and applications," *SIAM Multicale Model. Simul.*, vol. 13, pp. 338–366, 2014.

[13] E. T. Chung, Y. Efendiev, and W. T. Leung, "Generalized multiscale finite element methods for wave propagation in heterogeneous media," *Multiscale Modeling & Simulation*, vol. 12, no. 4, pp. 1691–1721, 2014.

[14] E. T. Chung, Y. Efendiev, and W. T. Leung, "Fast online generalized multiscale finite element method using constraint energy minimization," *Journal of Computational Physics*, vol. 355, pp. 450–463, 2018.

[15] E. T. Chung, W. T. Leung, and S. Pollock, "Goal-oriented adaptivity for GMsFEM," *Journal of Computational and Applied Mathematics*, pp. 625–637, 2015.

[16] Y. Efendiev, J. Galvis, and T. Hou, "Generalized multiscale finite element methods (GMsFEM)," *Journal of Computational Physics*, vol. 251, pp. 116–135, 2013.

[17] K. Gao, S. Fu, R. L. Gibson Jr, E. T. Chung, and Y. Efendiev, "Generalized multiscale finite-element method (GMsFEM) for elastic wave propagation in heterogeneous, anisotropic media," *Journal of Computational Physics*, vol. 295, pp. 161–188, 2015.

[18] E. Chung, Y. Efendiev, and T. Y. Hou, "Adaptive multiscale model reduction with generalized multiscale finite element methods," *Journal of Computational Physics*, vol. 320, pp. 69–95, 2016.

[19] E. Chung, Y. Efendiev, and W.-T. Leung, "Constraint energy minimizing generalized multiscale finite element method," *Computer Methods in Applied Mechanics and Engineering*, vol. 339, pp. 298–319, 2018.

[20] E. Chung, Y. Efendiev, and W. T. Leung, "Constraint energy minimizing generalized multiscale finite element method in the mixed formulation," *Computational Geosciences*, vol. 22, no. 3, pp. 677–693, 2018.

[21] E. T. Chung, Y. Efendiev, W. T. Leung, M. Vasilyeva, and Y. Wang, "Non-local multi-continua upscaling for flows in heterogeneous fractured media," *Journal of Computational Physics*, vol. 372, pp. 22–34, 2018.

[22] H. Owhadi and L. Zhang, "Metric-based upscaling," *Comm. Pure. Appl. Math.*, vol. 60, pp. 675–723, 2007.

[23] A. Abdulle and B. Engquist, "Finite element heterogeneous multiscale methods with near optimal computational complexity," *SIAM J. Multiscale Modeling and Simulation*, vol. 6, no. 4, pp. 1059–1084, 2007.

[24] W. E and B. Engquist, "Heterogeneous multiscale methods," *Comm. Math. Sci.*, vol. 1, no. 1, pp. 87–132, 2003.

[25] P. Henning, A. Målqvist, and D. Peterseim, "A localized orthogonal decomposition method for semi-linear elliptic problems," *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 48, no. 5, pp. 1331–1349, 2014.

[26] A. Målqvist and D. Peterseim, "Localization of elliptic multiscale problems," *Mathematics of Computation*, vol. 83, no. 290, pp. 2583–2603, 2014.

[27] A. Roberts and I. Kevrekidis, "General tooth boundary conditions for equation free modeling," *SIAM J. Sci. Comput.*, vol. 29, no. 4, pp. 1495–1510, 2007.

[28] G. Samaey, D. Roose, and I. Kevrekidis, "The gap-tooth scheme for homogenization problems," *SIAM J. Multiscale Modeling and Simulation*, vol. 4, no. 1, pp. 278–306, 2005.

[29] G. Samaey, I. Kevrekidis, and D. Roose, "Patch dynamics with buffers for homogenization problems," *J. Comput. Phys.*, vol. 213, no. 1, pp. 264–287, 2006.

[30] D. Fafalis and J. Fish, "Computational continua for linear elastic heterogeneous solids on unstructured finite element meshes," *International Journal for Numerical Methods in Engineering*, vol. 115, no. 4, pp. 501–530, 2018.

[31] J. Fish and S. Kuznetsov, "Computational continua," *International Journal for Numerical Methods in Engineering*, vol. 84, no. 7, pp. 774–802, 2010.

[32] J. Fish and Z. Yuan, "Multiscale enrichment based on partition of unity," *International Journal for Numerical Methods in Engineering*, vol. 62, no. 10, pp. 1341–1359, 2005.

[33] D. L. Brown, Y. Efendiev, and V. H. Hoang, "An efficient hierarchical multiscale finite element method for stokes equations in slowly varying media," *Multiscale Modeling & Simulation*, vol. 11, no. 1, pp. 30–58, 2013.

[34] V. Hoang and C. Schwab, "High dimensional finite elements for elliptic problems with multiple scales," *SIAM Multiscale Modeling and Simulation*, vol. 3, pp. 168–194, 2004.

[35] W. C. Tan and V. H. Hoang, "High dimensional finite element method for multiscale nonlinear monotone parabolic equations," *Journal of Computational and Applied Mathematics*, vol. 345, pp. 471–500, 2019.

[36] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

[37] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European conference on computer vision*, pp. 694–711, Springer, 2016.

[38] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," *arXiv preprint arXiv:1805.08318*, 2018.

[39] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[41] Z. Zhang, E. T. Chung, Y. Efendiev, and W. T. Leung, "Learning algorithms for coarsening uncertainty space and applications to multiscale simulations," *Mathematics*, vol. 8, no. 5, p. 720, 2020.

[42] E. Chung, W. T. Leung, S.-M. Pun, and Z. Zhang, "A multi-stage deep learning based algorithm for multiscale modelreduction," *arXiv preprint arXiv:2009.11341*, 2020.

[43] S. L. Brunton and J. N. Kutz, "Methods for data-driven multiscale model discovery for materials," *Journal of Physics: Materials*, vol. 2, no. 4, p. 044002, 2019.

[44] S. W. Cheung, E. T. Chung, Y. Efendiev, E. Gildin, Y. Wang, and J. Zhang, "Deep global model reduction learning in porous media flow simulation," *Computational Geosciences*, vol. 24, no. 1, pp. 261–274, 2020.

[45] A. Heinlein, A. Klawonn, M. Lanser, and J. Weber, "Machine learning in adaptive domain decomposition methods—predicting the geometric location of constraints," *SIAM Journal on Scientific Computing*, vol. 41, no. 6, pp. A3887–A3912, 2019.

[46] J. Magiera, D. Ray, J. S. Hesthaven, and C. Rohde, "Constraint-aware neural networks for Riemann problems," *Journal of Computational Physics*, vol. 409, p. 109345, 2020.

[47] F. Regazzoni, L. Dede, and A. Quarteroni, "Machine learning of multiscale active force generation models for the efficient simulation of cardiac electromechanics," *Computer Methods in Applied Mechanics and Engineering*, vol. 370, p. 113268, 2020.

[48] M. Vasilyeva, W. T. Leung, E. T. Chung, Y. Efendiev, and M. Wheeler, "Learning macroscopic parameters in nonlinear multiscale simulations using nonlocal multicontinua upscaling techniques," *Journal of Computational Physics*, p. 109323, 2020.

[49] K. Wang and W. Sun, "A multiscale multi-permeability poroplasticity model linked by recursive homogenizations and deep learning," *Computer Methods in Applied Mechanics and Engineering*, vol. 334, pp. 337–380, 2018.

[50] M. Wang, S. W. Cheung, W. T. Leung, E. T. Chung, Y. Efendiev, and M. Wheeler, "Reduced-order deep learning for flow dynamics. the interplay between deep learning and model reduction," *Journal of Computational Physics*, vol. 401, p. 108939, 2020.

[51] Q. Wang, N. Ripamonti, and J. S. Hesthaven, "Recurrent neural network closure of parametric POD-Galerkin reduced-order models based on the Mori-Zwanzig formalism," *Journal of Computational Physics*, p. 109402, 2020.

[52] Y. Wang, S. W. Cheung, E. T. Chung, Y. Efendiev, and M. Wang, "Deep multiscale model learning," *Journal of Computational Physics*, vol. 406, p. 109071, 2020.

[53] T. S. A. Yeung, E. T. Chung, and S. See, "A deep learning based nonlinear upscaling method for transport equations," *arXiv preprint arXiv:2007.03432*, 2020.

[54] B. Chetverushkin, "Kinetic schemes and quasi-gas-dynamic system of equations," *Russian Journal of Numerical Analysis and Mathematical Modelling*, vol. 20, no. 4, pp. 337–351, 2005.

[55] B. Chetverushkin, A. Saveliev, and V. Saveliev, "Compact quasi-gasdynamic system for high-performance computations," *Computational Mathematics and Mathematical Physics*, vol. 59, no. 3, pp. 493–500, 2019.

[56] B. N. Chetverushkin, N. G. Churbanova, A. A. Kuleshov, A. A. Lyupa, and M. A. Trapeznikova, "Application of kinetic approach to porous medium flow simulation in environmental hydrology problems on high-performance computing systems," *Russian Journal of Numerical Analysis and Mathematical Modelling*, vol. 31, no. 4, pp. 187–196, 2016.

[57] B. N. Chetverushkin, N. D'Ascenzo, A. V. Saveliev, and V. Saveliev, "Kinetic model and magnetogasdynamics equations," *Computational Mathematics and Mathematical Physics*, vol. 58, no. 5, pp. 691–699, 2018.

[58] B. N. Chetverushkin and A. A. Zlotnik, "On a hyperbolic perturbation of a parabolic initial–boundary value problem," *Applied Mathematics Letters*, vol. 83, pp. 116–122, 2018.

[59] A. Lutskii and B. Chetverushkin, "Compact version of the quasi-gasdynamic system for modeling a viscous compressible gas," *Differential Equations*, vol. 55, no. 4, pp. 575–580, 2019.

[60] L. C. Evans, *Partial differential equations*, vol. 19. American Mathematical Society, 2010.

[61] A. Bamberger, B. Engquist, L. Halpern, and P. Joly, "Higher order paraxial wave equation approximations in heterogeneous media," *SIAM Journal on Applied Mathematics*, vol. 48, no. 1, pp. 129–154, 1988.

[62] A. Bamberger, B. Engquist, L. Halpern, and P. Joly, "Parabolic wave equation approximations in heterogenous media," *SIAM Journal on Applied Mathematics*, vol. 48, no. 1, pp. 99–128, 1988.

[63] A. Bamberger, B. Euggnist, L. Helpern, and P. Joly, "The paraxial approximation for the wave equation: some new results," *Adv. Comput. Methods Partial Differential Equations*, pp. 340–344, 1984.

[64] J. F. Claerbout, *Fundamentals of geophysical data processing*, vol. 274. Citeseer, 1976.

[65] P. Blomgren, G. Papanicolaou, and H. Zhao, "Super-resolution in time-reversal acoustics," *The Journal of the Acoustical Society of America*, vol. 111, no. 1, pp. 230–248, 2002.

[66] H. Brock, R. Buchal, and C. Spofford, "Modifying the sound-speed profile to improve the accuracy of the parabolic-equation technique," *The Journal of the Acoustical Society of America*, vol. 62, no. 3, pp. 543–552, 1977.

[67] S. T. McDaniel, "Parabolic approximations for underwater sound propagation," *The Journal of the Acoustical Society of America*, vol. 58, no. 6, pp. 1178–1185, 1975.

[68] S. T. McDaniel, "Propagation of normal mode in the parabolic approximation," *The Journal of the Acoustical Society of America*, vol. 57, no. 2, pp. 307–311, 1975.

[69] A. Hasegawa and F. Tappert, "Transmission of stationary nonlinear optical pulses in dispersive dielectric fibers. i. anomalous dispersion," *Applied Physics Letters*, vol. 23, no. 3, pp. 142–144, 1973.

[70] J. D. Cole, "Modern developments in transonic flow," *SIAM Journal on Applied Mathematics*, vol. 29, no. 4, pp. 763–787, 1975.

[71] J. Hudson, "A parabolic approximation for elastic waves," *Wave Motion*, vol. 2, no. 3, pp. 207–214, 1980.

[72] B. Chetverushkin, E. Chung, Y. Efendiev, S.-M. Pun, and Z. Zhang, "Computational multiscale methods for quasi-gas dynamic equations," *arXiv preprint arXiv:2009.00068*, 2020.

[73] K. Kunisch and S. Volkwein, "Galerkin proper orthogonal decomposition methods for parabolic problems," *Numerische mathematik*, vol. 90, no. 1, pp. 117–148, 2001.

[74] E. Tsiropoulou, K. Koukas, and S. Papavassiliou, "A socio-physical and mobility-aware coalition formation mechanism in public safety networks," *EAI Endorsed Trans. Future Internet*, vol. 4, p. 154176, 2018.

[75] M. T. Thai, W. Wu, and H. Xiong, *Big Data in Complex and Social Networks*. CRC Press, 2016.

[76] V. M. Calo, Y. Efendiev, J. Galvis, and M. Ghommem, "Multiscale empirical interpolation for solving nonlinear pdes," *Journal of Computational Physics*, vol. 278, pp. 204–220, 2014.

[77] E. T. Chung, Y. Efendiev, and W. T. Leung, "Residual-driven online generalized multiscale finite element methods," *Journal of Computational Physics*, vol. 302, pp. 176–190, 2015.

[78] E. T. Chung, Y. Efendiev, and W. T. Leung, "An online generalized multiscale discontinuous galerkin method (gmsdgm) for flows in heterogeneous media," *Communications in Computational Physics*, vol. 21, no. 2, pp. 401–422, 2017.

[79] E. T. Chung, Y. Efendiev, and G. Li, "An adaptive gmsfem for high-contrast flow problems," *Journal of Computational Physics*, vol. 273, pp. 54–76, 2014.

[80] E. T. Chung, Y. Efendiev, G. Li, and M. Vasilyeva, "Generalized multiscale finite element methods for problems in perforated heterogeneous domains," *Applicable Analysis*, vol. 95, no. 10, pp. 2254–2279, 2016.

[81] Y. Efendiev, J. Galvis, R. Lazarov, M. Moon, and M. Sarkis, "Generalized multiscale finite element method. symmetric interior penalty coupling," *Journal of Computational Physics*, vol. 255, pp. 1–15, 2013.

[82] Y. Efendiev, J. Galvis, G. Li, and M. Presho, "Generalized multiscale finite element methods: Oversampling strategies," *International Journal for Multiscale Computational Engineering*, vol. 12, no. 6, 2014.

[83] E. Chung, Y. Efendiev, and T. Y. Hou, "Adaptive multiscale model reduction with generalized multiscale finite element methods," *Journal of Computational Physics*, vol. 320, pp. 69–95, 2016.

[84] E. Chung, Y. Efendiev, and S. Fu, "Generalized multiscale finite element method for elasticity equations," *International Journal on Geomathematics*, vol. 5(2), pp. 225–254, 2014.

[85] E. Chung, M. Vasilyeva, and Y. Wang, "A conservative local multiscale model reduction technique for stokes flows in heterogeneous perforated domains," *Journal of Computational and Applied Mathematics*, vol. 321, pp. 389–405, 2017.

[86] E. T. Chung, Y. Efendiev, W. T. Leung, and Z. Zhang, "Cluster-based generalized multiscale finite element method for elliptic pdes with random coefficients," *Journal of Computational Physics*, vol. 371, pp. 606–617, 2018.

[87] K. Karhunen, *Über lineare Methoden in der Wahrscheinlichkeitsrechnung*, vol. 37. Sana, 1947.

[88] S. Lloyd, "Least squares quantization in pcm," *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.

[89] M. Vasilyeva, W. T. Leung, E. T. Chung, Y. Efendiev, and M. Wheeler, "Learning macroscopic parameters in nonlinear multiscale simulations using nonlocal multicontinua upscaling techniques," *arXiv preprint arXiv:1907.02921*, 2019.

[90] S. W. Cheung, E. T. Chung, Y. Efendiev, E. Gildin, Y. Wang, and J. Zhang, "Deep global model reduction learning in porous media flow simulation," *Computational Geosciences*, pp. 1–14, 2019.

[91] M. Wang, S. W. Cheung, E. T. Chung, Y. Efendiev, W. T. Leung, and Y. Wang, "Prediction of discretization of gmsfem using deep learning," *Mathematics*, vol. 7, no. 5, p. 412, 2019.

[92] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 132–149, 2018.

[93] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *International conference on machine learning*, pp. 478–487, 2016.

[94] B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong, "Towards k-means-friendly spaces: Simultaneous deep learning and clustering," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3861–3870, JMLR. org, 2017.

[95] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[96] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.

[97] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681–4690, 2017.

[98] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Deep laplacian pyramid networks for fast and accurate super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 624–632, 2017.

[99] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2015.

[100] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1646–1654, 2016.

[101] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 286–301, 2018.

[102] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2472–2481, 2018.

[103] Y. Tai, J. Yang, and X. Liu, "Image super-resolution via deep recursive residual network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3147–3155, 2017.

[104] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, "Enhanced deep residual networks for single image super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 136–144, 2017.

[105] Y. Tai, J. Yang, X. Liu, and C. Xu, "Memnet: A persistent memory network for image restoration," in *Proceedings of the IEEE international conference on computer vision*, pp. 4539–4547, 2017.

[106] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[107] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.

[108] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-first AAAI conference on artificial intelligence*, 2017.

[109] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

[110] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[111] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proceedings of the IEEE international conference on computer vision*, pp. 1520–1528, 2015.

[112] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

[113] Y. Efendiev, J. Galvis, and T. Y. Hou, "Generalized multiscale finite element methods (gmsfem)," *Journal of Computational Physics*, vol. 251, pp. 116–135, 2013.

[114] 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.

[115] P. Jenny, S. Lee, and H. A. Tchelepi, "Multi-scale finite-volume method for elliptic problems in subsurface flow simulation," *Journal of Computational Physics*, vol. 187, no. 1, pp. 47–67, 2003.

[116] V. M. Calo, Y. Efendiev, J. Galvis, and G. Li, "Randomized oversampling for generalized multiscale finite element methods," *Multiscale Modeling & Simulation*, vol. 14, no. 1, pp. 482–501, 2016.

[117] A. Vidal-Jordana and X. Montalban, "Multiple sclerosis: epidemiologic, clinical, and therapeutic aspects," *Neuroimaging Clinics*, vol. 27, no. 2, pp. 195–204, 2017.

[118] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[119] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, vol. 1, no. 10, p. e3, 2016.

[120] X. Tan*, R. L. Gibson Jr, W. T. Leung, and Y. Efendiev, "Adaptive multiscale mcmc algorithm for uncertainty quantification in seismic parameter estimation," in *SEG Technical Program Expanded Abstracts 2014*, pp. 4665–4669, Society of Exploration Geophysicists, 2014.

[121] A. Mondal, B. Mallick, Y. Efendiev, and A. Datta-Gupta, "Bayesian uncertainty quantification for subsurface inversion using a multiscale hierarchical model," *Technometrics*, vol. 56, no. 3, pp. 381–392, 2014.

[122] Y. Efendiev, T. Hou, and W. Luo, "Preconditioning markov chain monte carlo simulations using coarse-scale models," *SIAM Journal on Scientific Computing*, vol. 28, no. 2, pp. 776–803, 2006.

[123] E. T. Chung, Y. Efendiev, B. Jin, W. T. Leung, and M. Vasilyeva, "Generalized multiscale inversion for heterogeneous problems," *arXiv preprint arXiv:1707.08194*, 2017.

[124] E. T. Chung, Y. Efendiev, W. T. Leung, M. Vasilyeva, and Y. Wang, "Non-local multi-continua upscaling for flows in heterogeneous fractured media," *Journal of Computational Physics*, vol. 372, pp. 22–34, 2018.

[125] E. Chung, Y. Efendiev, and T. Y. Hou, "Adaptive multiscale model reduction with generalized multiscale finite element methods," *Journal of Computational Physics*, vol. 320, pp. 69–95, 2016.

[126] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The journal of chemical physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[127] W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," 1970.

[128] R. V. Craiu, J. Rosenthal, and C. Yang, "Learn from thy neighbor: Parallel-chain and regional adaptive mcmc," *Journal of the American Statistical Association*, vol. 104, no. 488, pp. 1454–1466, 2009.

[129] W. R. Gilks, G. O. Roberts, and E. I. George, "Adaptive direction sampling," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 43, no. 1, pp. 179–189, 1994.

[130] E. Laloy and J. A. Vrugt, "High-dimensional posterior exploration of hydrologic models using multiple-try dream (zs) and high-performance computing," *Water Resources Research*, vol. 48, no. 1, 2012.

[131] M. D. Hoffman and A. Gelman, "The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo.," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1593–1623, 2014.

[132] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth, "Hybrid monte carlo," *Physics letters B*, vol. 195, no. 2, pp. 216–222, 1987.

[133] G. O. Roberts and J. S. Rosenthal, "Optimal scaling of discrete approximations to langevin diffusions," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 60, no. 1, pp. 255–268, 1998.

[134] M. Girolami and B. Calderhead, "Riemann manifold langevin and hamiltonian monte carlo methods," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 73, no. 2, pp. 123–214, 2011.

[135] B. Calderhead, "A general construction for parallelizing metropolis- hastings algorithms," *Proceedings of the National Academy of Sciences*, vol. 111, no. 49, pp. 17408–17413, 2014.

[136] S. P. Brooks and A. Gelman, "General methods for monitoring convergence of iterative simulations," *Journal of computational and graphical statistics*, vol. 7, no. 4, pp. 434–455, 1998.

[137] E. Laloy, B. Rogiers, J. A. Vrugt, D. Mallants, and D. Jacques, "Efficient posterior exploration of a high-dimensional groundwater model from two-stage markov chain monte carlo simulation and polynomial chaos expansion," *Water Resources Research*, vol. 49, no. 5, pp. 2664–2682, 2013.

[138] A. E. Brockwell, "Parallel markov chain monte carlo simulation by pre-fetching," *Journal of Computational and Graphical Statistics*, vol. 15, no. 1, pp. 246–261, 2006.

[139] J. Zhang, J. A. Vrugt, X. Shi, G. Lin, L. Wu, and L. Zeng, "Improving simulation efficiency of mcmc for inverse modeling of hydrologic systems with a kalman-inspired proposal distribution," *Water Resources Research*, vol. 56, no. 3, p. e2019WR025474, 2020.

[140] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum, "Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation," in *Advances in neural information processing systems*, pp. 3675–3683, 2016.

[141] T. Haarnoja, H. Tang, P. Abbeel, and S. Levine, "Reinforcement learning with deep energy-based policies," *arXiv preprint arXiv:1702.08165*, 2017.

[142] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[143] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *International conference on machine learning*, pp. 1995–2003, 2016.

[144] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[145] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[146] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.

[147] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, pp. 1928–1937, 2016.

[148] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[149] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *International Conference on Machine Learning*, pp. 2052–2062, 2019.

[150] S. Fujimoto, H. Van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," *arXiv preprint arXiv:1802.09477*, 2018.

[151] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[152] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[153] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, pp. 229–256, 1992.

[154] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, second ed., 2018.

[155] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *International Conference on Machine Learning*, pp. 2961–2970, PMLR, 2019.

[156] T. Zhao, Y. Xu, M. Monfort, W. Choi, C. Baker, Y. Zhao, Y. Wang, and Y. N. Wu, "Multi-agent tensor fusion for contextual trajectory prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 12126–12134, 2019.

[157] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in neural information processing systems*, pp. 6379–6390, 2017.

[158] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," *arXiv preprint arXiv:1705.08926*, 2017.

[159] J. Foerster, N. Nardelli, G. Farquhar, T. Afouras, P. H. Torr, P. Kohli, and S. Whiteson, "Stabilising experience replay for deep multi-agent reinforcement learning," *arXiv preprint arXiv:1702.08887*, 2017.