POD-DEIM GLOBAL-LOCAL MODEL REDUCTION FOR MULTI-PHASE FLOWS IN HETEROGENEOUS POROUS MEDIA

A Dissertation

by

YANFANG YANG

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
Committee Members,	Eduardo Gildin
	Raytcho Lazarov
	Jianxian Zhou
Head of Department,	Emil Straube

December 2016

Major Subject: Mathematics

ABSTRACT

Many applications such as production optimization and reservoir management are computationally demanding due to a large number of forward simulations. Typically, each forward simulation involves multiple scales and is computationally expensive. The main objective of this dissertation is to develop and apply both local and global model-order reduction techniques to facilitate subsurface flow modeling.

We develop a POD-DEIM global model reduction method for multi-phase flow simulation. The approach entails the use of Proper Orthogonal Decomposition (POD)- Galerkin projection, and Discrete Empirical Interpolation Method (DEIM). POD technique constructs a small POD subspace spanned by a set of global basis that can approximate the solution space. The reduced system is set up by projecting the full-order system onto the POD subspace. Discrete Empirical Interpolation Method (DEIM) is used to reduce the nonlinear terms in the system. DEIM overcomes the shortcomings of POD in the case of nonlinear PDEs by retaining nonlinearities in a lower dimensional space. The POD-DEIM global reduction method enjoys the merit of significant complexity reduction.

We also propose an online adaptive global-local POD-DEIM model reduction method. This unique global-local online combination allows (1) developing local indicators that are used for both local and global updates; (2) computing global online modes via local multiscale basis functions. The multiscale basis functions consist of offline and some online local basis functions. The main contribution of the method is that the criteria for adaptivity and the construction of the global online modes are based on local error indicators and local multiscale basis functions which can be cheaply computed. The approach is particularly useful for situations where one needs to solve the reduced system for inputs or

controls that result in a solution outside the span of the snapshots generated in the offline stage.

Another aspect of my dissertation is the development of a local model reduction method for multiscale problems. We use global coupling in the coarse grid level via the mortar framework to link the sub-grid variations of neighboring coarse regions. The mortar framework offers some advantages, such as the flexibility in the constructions of the coarse grid and sub-grid capturing tools. By following the framework of the Generalized Multiscale Finite Element Method (GMsFEM), we design an enriched multiscale mortar space. Using the proposed multiscale mortar space, we (1) construct a multiscale finite element method to solve the flow problem on a coarse grid; (2) design two-level preconditioners as exact solver for the flow problem.

ACKNOWLEDGEMENTS

First of all, I would like to express my deepest gratitude to my advisor Dr. Yalchin Efendiev. Dr. Efendiev has been of great help for my whole PhD study. I really appreciate his insightful advice, constant encouragement and patient guidance, which made this work possible. I am also thankful for his generous support of my study, and for the opportunity to attend conferences supported by him. It has been a privilege and pleasure to work with him.

I would like to thank Dr. Eduardo Gildin, for serving as my thesis committee, providing me with information from the petroleum engineering related to my work, and writing recommendation letters.

I am grateful to my thesis committee, Dr. Raytcho Lazarov and Dr. Jianxin Zhou for their patience and time. I would like to thank Dr. Howard for for substituting in my preliminary exam. Sincere thanks to the Department of Mathematics, Texas A&M University, for providing me the great opportunity to pursue my doctorate degree here. I also would like to acknowledge the faculty, staff and students in the the Department of Mathematics for their help during these years. I am thankful to our group members, for the informative discussions, happy conversations, and relaxing dinner gatherings.

Last but not least, I would like to thank my parents and brother for their encouragement and support during my time at Texas A&M University. I am indebted to my husband, Bingqi, for his company and love throughout this journey.

Yanfang Yang

November 15, 2016

TABLE OF CONTENTS

Page

ABSTRACT ii

ACKNOWLEDGEMENTS iv

TABLE OF CONTENTS v

LIST OF FIGURES vii

LIST OF TABLES ix

1. INTRODUCTION 1

1.1 Motivation and background 1
1.1.1 Global model reduction 1
1.1.2 Local model reduction 5
1.2 Dissertation outline 7

2. PRELIMINARIES 10

2.1 Model problem 10
2.2 Global model reduction method using POD-DEIM 13
2.2.1 Proper Orthogonal Decomposition (POD) 15
2.2.2 Discrete Empirical Interpolation Method (DEIM) 16
2.3 Coarse and fine grids 19
2.4 Local model reduction method via GMsFEM 22

3. POD-DEIM MODEL REDUCTION FOR MULTI-PHASE FLOWS 26

3.1 Introduction 26
3.2 Discretization 27
3.3 POD-DEIM model reduction 30
3.4 Numerical examples 32
3.4.1 Mass conservation in POD with finite volume method 32
3.4.2 POD-DEIM model reduction 34
3.4.3 Global-Local model reduction 38
3.4.4 Case study with viscous only 39
3.4.5 Case study with gravity 44

4. ONLINE ADAPTIVE GLOBAL-LOCAL MODEL REDUCTION FOR MULTI-PHASE FLOWS IN HETEROGENEOUS MEDIA 50

4.1 Introduction 50
4.2 Online adaptive global-local POD-projection 52
4.3 Online adaptive DEIM 57
4.4 Numerical examples 61
4.4.1 Single-phase flow 61
4.4.2 An incompressible two-phase flow model 65

5. AN ENRICHED MULTISCALE MORTAR SPACE FOR HIGH CONTRAST FLOW PROBLEMS 77

5.1 Introduction 77
5.2 Preliminaries 78
5.2.1 Variational form 79
5.2.2 The finite element approximation 79
5.2.3 Interface problem 81
5.3 Multiscale mortar space 83
5.3.1 Construction of local snapshot space 84
5.3.2 Construction of multiscale mortar space 85
5.3.3 Oversampling and randomized snapshot 86
5.4 Two-level domain decomposition preconditioners 86
5.4.1 Global coarse preconditioners 87
5.4.2 Local preconditioners 87
5.4.3 Two-level preconditioners 88
5.5 Numerical examples 89
5.5.1 Coarse grid multiscale solution 91
5.5.2 Preconditioner 96

6. CONCLUSIONS 104

REFERENCES 106

LIST OF FIGURES

FIGURE Page

2.1 First 30 singular values and first 20 DEIM points. 20

2.2 Comparison of the nonlinear function of dimension 441 with the POD and DEIM approximation of dimension 11 at $\mu = (-0.02, -0.5)$. 20

2.3 Illustration of coarse and fine grids. 21

3.1 Training and test (perturbed) bottom hole pressure of the injector. 35

3.2 POD model reduction on finite volume formulation. 35

3.3 Evaluation of mass conservation at final time. 36

3.4 POD model reduction on mixed finite element formulation. 36

3.5 Logarithm of permeability field. 39

3.6 SPE10 - 5 layers 40

3.7 Training schedule. 41

3.8 Singular values of snapshot matrix. 42

3.9 Test schedule ($\pm 20\%$ random variation in training) 42

3.10 Results for training schedule. 45

3.11 Final relative error with the training schedule. 45

3.12 Results for test schedule. 46

3.13 Final relative error with the test schedule. 46

3.14 Saturation errors advancing with respect to time. 49

4.1 Flowchart of the online adaptive global-local POD method. 53

4.2 Schematic description of the online adaptive global-local POD method. . 54

4.3 Grid information and q. 63

4.4 Injection rate schedules for training and test models. 64

4.5 Relative errors advancing in time for the adaptive POD. 64

4.6 Injection rate schedules for training and test models. 67

4.7 Relative saturation errors advancing in time: POD for the pressure equation. 68

4.8 Water-cut of online adptive POD for the pressure equation. 69

4.9 Relative saturation errors advancing in time: POD for the saturation equation. 71

4.10 Water-cut of online adptive POD for the saturation equation. 72

4.11 Water-cut of online adaptive POD-DEIM for the saturation equation. . . . 73

4.12 Relative saturation errors advancing in time: POD-DEIM for the saturation equation. 74

4.13 Relative saturation errors advancing in time: POD-DEIM for the coupled system. 75

4.14 Water-cut of online adaptive POD-DEIM for the coupled system. 76

5.1 Illustration of a fine edge on a coarse edge. 85

5.2 Permeability fields. 90

5.3 Illustration of an oversampled neighborhood. 91

5.4 Comparison of the coarse-scale solutions with the reference (fine-scale) solution, $N_H = 10$, $\eta = 10^4$, model 1. 97

5.5 Comparison of the coarse-scale solutions with the reference (fine-scale) solution, $N_H = 10$, $\eta = 10^4$, model 2. 98

5.6 Relative error for p (left), $\mathbf{u}$ (right) with contrast order $\eta = 10^4$ and $\eta = 10^6$ for model 2, basis generation case 2. 99

5.7 Relative error for p (left), $\mathbf{u}$ (right) with contrast order $\eta = 10^4$ and $\eta = 10^6$ for model 1, basis generation case 2. 100

LIST OF TABLES

TABLE Page

3.1 Relative saturation error, Global (POD-DEIM). 37

3.2 Relative saturation error as defined in (3.17), Global-Local. 38

3.3 Overhead time for calculating the basis. 43

3.4 Compare fine and reduced scale model. 43

3.5 Number of DEIM points corresponding to different nonlinear functionals defined in 3.8 using Algorithm 2. d refers to the derivative. Number of basis for states: $N_v = 20, N_p = 5, N_s = 25$ 48

4.1 Adaptive-POD-1 with 5 initial global POD basis. 65

4.2 Average POD errors. 65

4.3 Number of POD basis for saturation after each update of POD subspace for Equation (2.6). 70

4.4 Number of POD basis for saturation after each update of POD subspace for the coupled system. 75

5.1 Relative error between multiscale solution and fine scale solution with different types of basis for model 1, $N_H = 5$, $\eta = 10^4$. "Nb" represent the number of basis per coarse edge. 94

5.2 Relative error between multiscale solution and fine scale solution with different types of basis for model 1, $N_H = 10$, $\eta = 10^4$. "Nb" represent the number of basis per coarse edge. 95

5.3 Relative error between multiscale solution and fine scale solution with different types of basis for model 2, $N_H = 5$, $\eta = 10^4$. "Nb" represent the number of basis per coarse edge. 95

5.4 Relative error between multiscale solution and fine scale solution with different types of basis for model 2, $N_H = 10$, $\eta = 10^4$. "Nb" represent the number of basis per coarse edge. 96

5.5 PCG iteration number with different types of coarse space, 2 basis is used for coarse space, $N_H = 5$. 102

5.6 GMRES(2) iteration number with different types of preconditioners and different local preconditioner, 2 basis is used for coarse space with multi-scale basis case 3, $N_H = 5$. 102

5.7 GMRES(2) iteration number with different types of preconditioners and different local preconditioner, 2 basis is used for coarse space with multi-scale basis case 4, $N_H = 5$. 103

1. INTRODUCTION

1.1 Motivation and background

1.1.1 Global model reduction

Optimization and uncertainty quantification are essential components of reservoir management. The computation for such problems is usually prohibitively time-consuming due to the the large number of forward flow simulations that need to be performed and the typically large dimension of the simulation models. Reduced order modeling techniques (ROMs) are powerful tools for such time-critical applications. ROMs seek to replace large-scale computational models with smaller approximations of these models that are capable of faithfully reproducing their essential features at reduced computational cost.

Many local, global, and global-local model reduction techniques have been developed. Next we give a brief review of these techniques. Most of the global reduction methods are based on the technique of projecting the full-order states onto a low dimension representations of global basis functions. The approaches to construct these basis functions include the Krylov subspace projection methods [63], the truncated-balance reduction (T-BR) method [64], and the proper orthogonal decomposition (POD) method [85]. The first two are equation driven strategies, which are primarily used for linear time invariant models (LTI). POD is empirical data driven, and is widely used for both linear and nonlinear models. Next, we give more detailed description of these methods.

Krylov subspace based techniques have emerged as one of the most powerful tools for reduced-order modeling of large-scale LTI systems. The reduced bases are generated by approximating eigenvectors corresponding to the largest eigenvalues. For Krylov subspace based techniques, they have the advantages of low computational cost, and the ease of generating the projection basis. However, lack of provable error bounds, and inability to

preserve the original system's stability and passivity are their limitations.

TBR constructs the basis matrix by exploiting the structure of the system of equations. This method has global error bounds on the accuracy of the resulting reduced model. However, its computational cost is $O(N^3)$, N is the number of unknowns of the full-order system. In the context of fluid flow simulation, TBR has been applied to single-phase flow by Zandvliet [118], and two-phase flow by Heijn et al. [70]. In [102], Rowley applied TBR to fluid dynamics problems.

POD was first introduced by Pearson [93] in 1901 as an approach to deal with coherent structure in dynamic systems. POD was successfully applied to a number of areas [66, 80, 103, 21, 18, 88, 100, 111, 110, 70, 23, 59]. For ROM procedures based on POD projection, the process typically consists of an offline stage and an online stage. In the offline stage, snapshots are collected from experimental data or from forward simulations, then POD constructs basis functions, known as spatial modes, from the snapshots by a singular value decomposition (SVD). By projection onto the subspace spanned by the basis functions, the reduced model for the problem of interest is constructed. In the online stage, the reduced model is solved. Computational complexity is reduced since only relatively few basis functions must be retained.

However, for nonlinear problems, POD projection alone is not sufficient. The reason is that the cost to evaluate the projected nonlinear function is as expensive as evaluating the original system. To lower the computational cost of dealing with nonlinearities, Astrid et al. [15] proposed Missing Point Estimation (MPE), Cardoso et al. [22] proposed Trajectory Piecewise Linearization (TPWL) to linearize the nonlinear functions around several known states; Chaturantabut et al. [25] proposed the Discrete Empirical Interpolation Method (DEIM) to approximate a nonlinear function by combining projection with interpolation. Next we briefly discuss these three approaches.

To improve the computational efficiency of POD, Astrid et al. [15] proposed MPE for

large scale linear time-varying (LTV) and nonlinear models. In principle, MPE constructs POD basis from a subset of the spatial domain instead of the whole spatial domain. A restricted POD basis is obtained by extracting some rows of the standard POD basis vectors corresponding to the selected grid blocks. The reduced order system is constructed by projecting the corresponding subset of the governing equations onto the subspace spanned by the restricted POD basis. Astrid et al. [15] introduced two algorithms to select the subset of the grid blocks. Both algorithms seek to limit the growth of the condition number of the matrix formed by the restricted POD basis. It is shown in [15], by applying MPE with POD to a problem of temperature distribution in a glass melt feeder, a speed-up of 5.3 has been achieved compared with the POD reduced order model. Cardoso et al. [23] applied MPE to multiphase flow problems.

TPWL was introduced by Rewie$\acute{n}$ski [101]. The main idea of TPWL is to represent the nonlinear model as a weighted combination of the piecewise linear models at selected points along its trajectory. It has been implemented in conjunction with other reduced order modeling methods such as Krylov subspace [61, 23], and POD [69]. TPWL works well for problems with weak nonlinearity. However, for problems with strong nonlinearity, it is difficult to accurately approximate nonlinear terms by representation of a relatively small number of linearized models. Another limitation of TPWL is that the selection of the training trajectories and linearization points are an ad-hoc process [61].

Another alternative to approximate the nonlinear terms is by representation of their basis vectors at a set of interpolation points. Methods such as Empirical Interpolation Method (EIM) [87] and Best Point Interpolation Method (BPIM) [90] fall into this category. In these methods, basis vectors for the nonlinear terms are constructed from snapshots of nonlinear terms generated during the training simulations. A small number of interpolation points are used, such that the nonlinear terms are evaluated only over a subset of the spatial grid blocks instead of the entire spatial domain. EIM selects the interpolation

points in a "greedy" way that the $n-$ th interpolation point is placed at the spatial location where the difference between the $n-$ th basis vector and its approximation using a linear combination of the first $n-1$ basis vectors at the $n-1$ interpolation points is greatest. BPIM chooses the set of interpolation points by solving an n dimensional optimization problem to minimize the least square error between each snapshot and its approximation using n interpolation points. The cost of solving the constrained optimization problem for the optimal set of interpolation points is expensive in comparison with the sub-optimal interpolation points selected by EIM. Both methods have been used in several application such as simulation of second order wave propagation [82], inverse parameter estimation [54], convection-diffusion reaction problem [74], and nonaffine parametrized problems [62].

Chaturantabut et al. [25] developed a discrete variant of EIM, the Discrete Empirical Interpolation Method (DEIM), that can be easily implemented on semi-discrete systems. DEIM focuses on approximating each nonlinear function such that a certain coefficient matrix can be precomputed. Therefore, the complexity in evaluating the nonlinear term becomes proportional to the small number of selected interpolation points. DEIM has been applied to neuron modeling [75], shallow water model [108, 16] and multiphase flow in porous media [107, 4].

Adaptivity in the context of model reduction has attracted much attention. Offline adaptive methods [65, 26] seeks to provide a better reduce solution space while the reduced system is constructed in the offline stage. However, once the reduced system is derived, it stays fixed and is kept unchanged in the online stage. Therefore, the online solution relies only on the precomputed information from the offline stage. Online adaptive methods modify the reduced system during the computations in the online stage. Most of the existing online adaptive methods [94, 86, 6, 7], however, rely only on precomputed quantities from the offline stage. These methods work fine when the offline data contains

representative information of the solution to the problem of interest. However, in many applications, such as inverse problem and optimization, the final solution path may be difficult to predict before the problem is solved. Accurate reduction methods for inputs or controls that result in a solution outside the span of the snapshots generated in the offline stage are desired. We are motivated to consider online adaptive method by incorporating new data that becomes available online. Therefore, our method can deal with cases that the solution of the problem at hand lies out of the span of the snapshots generated in the offline stage.

1.1.2 Local model reduction

Thanks to the development of reservoir characterization methods and geostatistical modeling techniques, the description of the reservoir properties can be detailed at multiple scales. In cases where we care about the details of the fluid flow, such as the presence of fracture, resolving all the scales with direct simulations are prohibitively expensive. Moreover, in the context of the aforementioned global reduction method, forward flow problems with multiple scale and high contrast need to be solved repeatedly. In this respect, efficient methods have been proposed to reduce the dimension of the models while preserving a certain prescribed accuracy. These methods include numerical upscaling [115, 41, 1], variational multiscale methods [71, 72], multiscale finite element methods [47, 49, 10, 29, 53], mixed multiscale finite element methods [31, 27], the multiscale finite volume method [73], mortar multiscale finite element methods [114, 113, 12, 13], multiscale hybrid-mixed finite element methods [9, 68], generalized multiscale finite element methods [44, 30, 37, 58, 34], and weak Galerkin generalized multiscale finite element method [89].

Generally speaking, multiscale methods [44, 31, 45] construct an approximation of the solution on a coarse grid for arbitrary coarse-level inputs. Multiscale techniques provide

substantial computational savings when forward problems are solved many times because the same multiscale basis functions (or coarse spaces) can be utilized for all forward simulations. The proposed approaches solve these forward problems on a coarse grid multiple times and thus can provide a substantial speed-up.

The aforementioned local methods typically use some type of global couplings on the coarse-grid level to link the sub-grid variations of neighboring coarse regions. In my dissertation, we will also study a local model reduction technique using the mortar framework. The mortar framework offers some advantages, such as the flexibility in the constructions of the coarse grid and sub-grid capturing tools. The framework also gives a smaller dimensional global system since the degrees of freedom are reduced to coarse region boundaries.

By following the framework of the GMsFEM, we design an enriched multiscale mortar space. Using the proposed multiscale mortar space, we (1) construct a multiscale finite element method to solve the flow problem on a coarse grid. The method shares some common elements with hybridized multiscale methods [50, 51]; (2) design two-level preconditioners as exact solver for the flow problem. It is well known that for high contrast heterogeneous media, if the coarse problem is not suitably chosen, the performance of the preconditioner may deteriorate. To deal with this problem, many researchers designed different types of robust two-level preconditioners with nonstandard coarse problems in the past several decades. For example, in [105] the authors proposed a nonstandard coarse space for the elliptic problems with discontinuous coefficients. The idea of using single multiscale basis to form the coarse space is reported in [60, 91], this method is robust if the high conductivity does not cross the coarse grid. Using spectral functions to enrich the coarse space turns out to be very efficient and robust for problems with almost any types of media [55, 56, 46, 39, 77, 106, 78, 76, 79, 92]. We use the multiscale mortar space as coarse space for the preconditioners.

1.2 Dissertation outline

We start in Section 2 to present preliminary background materials. We first introduce the subsurface flow and transport equations in porous media that will be studied throughout the dissertation. We will consider these equations for both single- and two-phase flows. We then briefly review the framework of POD-DEIM global model reduction method. Coarse and fine grids concept is introduced and basic coarse-grid solution techniques based on GMsFEM to solve porous media flow equations are also briefly revisited.

In Section 3, we present a global-local POD-DEIM model reduction for fast multi-scale reservoir simulations in highly heterogeneous porous media. Our approach identifies a low dimensional structure in the solution space via POD. We introduce an auxiliary variable (the velocity field) in our model reduction that achieves a high compression of the model. This compression is achieved because the velocity field is conservative for any low-order reduced model in our framework, while a typical global model reduction based on POD Galerkin projection can not guarantee local mass conservation. The lack of mass conservation can be observed in numerical simulations that use finite volume based approaches. DEIM approximates fine-grid nonlinear functions in Newton iterations. This approach delivers an online computational cost that is independent of the fine grid dimension. POD snapshots are inexpensively computed using local model reduction techniques based on the GMsFEM which provides (1) a hierarchical approximation of the snapshot vectors, (2) adaptive computations by using coarse grids, (3) inexpensive global POD operations in small dimensional spaces on a coarse grid. By balancing the errors of the global and local reduced-order models, our new methodology provides an error bound in simulations. Our numerical results, utilizing a two-phase immiscible flow, show a substantial speed-up and we compare our results with the standard POD-DEIM in a finite volume setup.

In Section 4, we develop an online adaptive global-local POD-DEIM model reduction

method. The main idea of the proposed method is to use local online indicators to decide on the global update, which is performed via reduced cost local multiscale basis functions. This unique global-local online combination allows (1) developing local indicators that are used for both local and global updates (2) computing global online modes via local multiscale basis functions. The multiscale basis functions consist of offline and some online local basis functions. The online adaption is performed by incorporating new data, which become available at the online stage. Once the criterion for updates is satisfied, we adapt the reduced system online by changing the POD subspace and the DEIM approximation of the nonlinear functions. The main contribution of the method is that the criterion for adaption and the construction of the global online modes are based on local error indicators and local multiscale basis function which can be cheaply computed. Since the adaption is performed infrequently, the new methodology does not add significant computational overhead associated with when and how to adapt the global basis. The approach is particularly useful for situations where it is desired to solve the reduced system for inputs or controls that result in a solution outside the span of the snapshots generated in the offline stage. Our method also offers an alternative of constructing a robust reduced system even if a potential initial poor choice of snapshots is used. Applications to single-phase and two-phase flow problems demonstrate the efficiency of our method.

In Section 5, we propose a local model reduction method for multicale problems. We use a global coupling on the coarse-grid level via the mortar framework to link the sub-grid variations of neighboring coarse regions. The mortar framework offers some advantages, such as the flexibility in the constructions of the coarse grid and sub-grid capturing tools. By following the framework of the Generalized Multiscale Finite Element Method (GMsFEM), we design an enriched multiscale mortar space. In particular, we first construct a local snapshot space. Then we select the dominated modes within the snapshot space using the POD technique. Using the proposed multiscale mortar space, we will construct

a multiscale finite element method to solve the flow problem on a coarse grid and a preconditioning technique for the fine scale discretization of the flow problem. In particular, we develop a multiscale mortar mixed finite element method using the mortar space. In addition, we design a two-level *additive* preconditioner and a two-level *hybrid* preconditioner based on the proposed mortar space for the iterative method applied to the fine scale discretization of the flow problem. We present several numerical examples to demonstrate the efficiency and robustness of our proposed mortar space with respect to both the coarse multiscale solver and the preconditioners.

In Section 6, we summarize the contributions made throughout this dissertation.

2. PRELIMINARIES

In this section, we introduce preliminary background materials for the rest of the sections, including the description of the flow-transport model problem, the global POD-DEIM model reduction framework, coarse and fine grids, and coarse-grid local model order reduction techniques. We start with the description of the mathematical model problem.

2.1 Model problem

In the dissertation, we consider a coupled system of flow and transport equations in heterogeneous porous media. In particular, we study both single- and two-phase flows, where single-phase flow can be considered as a special case of the two-phase flow systems. First, we briefly describe the two-phase flow equations. We consider nonlinear immiscible, incompressible, two-phase (water-oil) flow in heterogeneous porous media in a subsurface formation (denoted by Ω). The continuity equation, also known as the mass conservation law for each phase, designated α (where $\alpha = w$ for water, and $\alpha = o$ for oil), is given by

$$\phi \frac{\partial s_\alpha}{\partial t} + \nabla \cdot \mathbf{u}_\alpha = q_\alpha \quad \alpha = w, o, \tag{2.1}$$

where ϕ is the porosity of the medium, which is considered to be constant in this dissertation, $\mathbf{u}_\alpha$ is the phase Darcy velocity, s_α is the saturation, and q_α is the source term. The phase velocity is related with pressure through the Darcy's law:

$$\mathbf{u}_\alpha = -\kappa \frac{k_{r\alpha}(s_\alpha)}{\mu_\alpha} (\nabla p_\alpha + \rho_\alpha g \nabla z), \tag{2.2}$$

where κ is the absolute permeability tensor, $k_{r\alpha}$ is the relative permeability for phase α, ρ_α is the density, g is the magnitude of the gravitational force, z is the depth, p_α is the pressure, and μ_α is the viscosity. We define the phase relative mobility as $\lambda_\alpha = \frac{k_{r\alpha}(s_\alpha)}{\mu_\alpha}$.

Substituting the phase velocities (2.2) into (2.1), we derive two mass-balance equations with four unknowns: s_w, s_o, p_w, p_o. To close the system, we add the saturation constraint, and the capillary-pressure relation, which are respectively expressed as:

$$s_w + s_o = 1, \tag{2.3}$$

$$p_o - p_w = p_c, \tag{2.4}$$

where the capillary pressure $p_c = p_c(s_w)$ is a nonlinear function of the water saturation.

We derive the flow-transport system for immiscible, incompressible two-phase flow:

$$\nabla \cdot \mathbf{u} = q_t \quad \text{in} \quad \Omega, \tag{2.5}$$

$$\phi \frac{\partial s_w}{\partial t} + \nabla \cdot \mathbf{u_w} = q_w \quad \text{in} \quad \Omega, \tag{2.6}$$

where $\mathbf{u}$ is the total velocity, $\mathbf{u_w}$ is the water phase Darcy velocity expressed in terms of the total velocity $\mathbf{u}$, which are respectively written as:

$$\mathbf{u} = -\kappa\lambda_t(s_w)\nabla p_w - \kappa g\Big(\lambda_w(s_w)\rho_w + \lambda_o(1-s_w)\rho_o\Big)\nabla z - \kappa\lambda_o(1-s_w)\nabla p_c, \tag{2.7}$$

$$\mathbf{u_w} = f_w\Big(\mathbf{u} - \kappa g\lambda_o(\rho_w - \rho_o)\nabla z + \kappa\lambda_o\nabla p_c\Big).$$

Here $\lambda_t(s) = \lambda_w(s) + \lambda_o(s)$ is the total mobility, $q_t = q_w + q_o$ is the total flow rate and

$$f_w(s) = \frac{\lambda_w(s)}{\lambda_t(s)} = \frac{k_{rw}(s)}{k_{rw}(s) + \frac{\mu_w}{\mu_o}k_{ro}(s)} \tag{2.8}$$

is the water fractional flow. The term $f_w\mathbf{u}$ represents viscous forces, the term $f_w\kappa g\lambda_o(\rho_w-\rho_o)\nabla z$ represents gravitational forces, and the term $f_w\kappa\lambda_o\nabla p_c$ represents the capillary forces.

If we ignore the capillary pressure, then $p_w = p_o$. Denote $p = p_w$, and there are two unknowns (p, s_w). Once these primary unknowns are computed, s_o can be determined by the saturation constraint $s_w + s_o = 1$. We denote $s = s_w$ for the sake of brevity. Without capillary effects, the total velocity becomes

$$\mathbf{u} \;=\; -\kappa\lambda_t\nabla p - \kappa g(\lambda_w\rho_w + \lambda_o\rho_o)\nabla z. \tag{2.9}$$

Put (2.9) into (2.5), we obtain the pressure equation as:

$$\nabla\cdot\Big(-\kappa\lambda_t\nabla p - \kappa g(\lambda_w\rho_w + \lambda_o\rho_o)\nabla z\Big) \;=\; q_t. \tag{2.10}$$

The coupled flow-transport system (with $\mathbf{u}, p, s$ as primary variables) is:

$$\nabla\cdot\Big(-\kappa\lambda_t(s)\nabla p - \kappa g(\lambda_w(s)\rho_w + \lambda_o(s)\rho_o)\nabla z\Big) \;=\; q_t \quad \text{in} \quad \Omega, \tag{2.11}$$

$$\mathbf{u} + \kappa\lambda_t\nabla p + \kappa g(\lambda_w\rho_w + \lambda_o\rho_o)\nabla z \;=\; 0 \quad \text{in} \quad \Omega, \tag{2.12}$$

$$\phi\frac{\partial s}{\partial t} + \nabla\cdot\Big(f_w(s)(\mathbf{u} - \kappa g\lambda_o(s)(\rho_w-\rho_o)\nabla z)\Big) \;=\; q_w \quad \text{in} \quad \Omega, \tag{2.13}$$

which can be solved with proper initial and boundary conditions.

The system (2.11)-(2.13) is a coupled nonlinear system with an elliptic equation and a parabolic equation. We solve the system following a sequential method. Saturation from previous step (or initial condition) is used to compute the saturation-dependent terms in (2.11), e.g. λ_t, and then solve (2.11), (2.9) for the total velocity. Then, total velocity stays constant while saturation is solved from (2.13) and advance in time. Next, with the

new saturation states, we update the saturation dependent terms in (2.11) and solve for the velocity states. The process keeps going in this way until the end of the simulation time.

2.2 Global model reduction method using POD-DEIM

In this section, we review the concept of POD-DEIM based global model reduction method, which can reduce the dimension of large-scale ODE systems regardless of their origin. A large source of such systems is the semidiscretization of time dependent PDEs. After spatial discretization for a nonlinear PDE, we get a system of nonlinear ODEs of the form

$$\frac{d}{dt}\mathbf{y}(t) = \mathbf{B}\mathbf{y}(t) + \mathbf{f}(\mathbf{y}(t)), \tag{2.14}$$

where $\mathbf{y}(t) \in \mathbb{R}^n, \mathbf{B} \in \mathbb{R}^{n\times n}$ is a constant matrix, and $\mathbf{f}(\mathbf{y}(t))$ is a nonlinear function.

We start our exposition by reviewing the global model reduction framework. Suppose k ($k \ll n$) number of POD basis $\{\boldsymbol{\phi}_1, \cdots, \boldsymbol{\phi}_k\}$ are given. We describe POD basis construction in Section 2.2.1. The solution of (2.14) is approximated as a linear combination of the POD bases $\{\boldsymbol{\phi}_1, \cdots, \boldsymbol{\phi}_k\}$, i.e., $\mathbf{y}(t) \approx \sum_{i=1}^{k} \tilde{y}_i \boldsymbol{\phi}_i$, which can be written as a matrix form $\mathbf{y}(t) = \mathbf{V}_k\tilde{\mathbf{y}}(t)(\tilde{\mathbf{y}} \in \mathbb{R}^k)$. Here $\mathbf{V}_k \in \mathbb{R}^{n\times k}$ is an orthonormal matrix consisting of the POD bases $\{\boldsymbol{\phi}_1, \cdots, \boldsymbol{\phi}_k\}$. Substituting $\mathbf{y}(t) = \mathbf{V}_k\tilde{\mathbf{y}}(t)$ in (2.14), and by Galerkin projection of the system (2.14) onto $\mathbf{V}_k$, we get the reduced system of (2.14) (the super index T means the transpose of a matrix):

$$\frac{d}{dt}\tilde{\mathbf{y}}(t) = \mathbf{V}_k^T\mathbf{B}\mathbf{V}_k\tilde{\mathbf{y}}(t) + \mathbf{V}_k^T\mathbf{f}(\mathbf{V}_k\tilde{\mathbf{y}}(t)). \tag{2.15}$$

The quality of the above POD Galerkin approximation depends on the choice of the global POD bases. The POD technique constructs a set of global basis functions from a singular value decomposition (SVD) of snapshots, which are discrete samples of trajec-

tories associated with a particular set of boundary conditions and inputs. Therefore, the empirically generated global bases depend on the sampling method. It is expected that the samples (snapshots) will be representative of the solution manifold of the problem of interest. Among the various techniques for obtaining global bases, POD constructs a reduced basis that is optimal in the sense that a certain approximation error concerning the snapshots is minimized. The details of POD are presented in Section 2.2.1.

To evaluate the nonlinear term $\mathbf{V}_k^T\mathbf{f}(\mathbf{V}_k\tilde{\mathbf{y}}(t))$ in (2.15), we first need to project the solution from the reduced space to the fine solution space, then evaluate the nonlinear function $\mathbf{f}$ at all the n fine components, finally project the n fine components back onto $\mathbf{V}_k$. The process results in a cost as expensive as solving the original system. To handle this in a computational efficient way, we use DEIM [25] to reduce the nonlinear term, which is reviewed in Section 2.2.2. The DEIM interpolant of $\mathbf{f}$ is defined by $(\mathbf{U}, \mathbf{P})$. The DEIM approximation of $\mathbf{f}$ is derived, based on $(\mathbf{U}, \mathbf{P})$, as

$$\mathbf{f}(\mathbf{y}(t)) \approx \mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T\mathbf{f}(\mathbf{y}(t)). \tag{2.16}$$

Put (2.16) back into (2.15), we obtain

$$\frac{d}{dt}\tilde{\mathbf{y}}(t) = \mathbf{V}_k^T\mathbf{B}\mathbf{V}_k\tilde{\mathbf{y}}(t) + \mathbf{V}_k^T\mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T\mathbf{f}(\mathbf{V}_k\tilde{\mathbf{y}}(t)). \tag{2.17}$$

If the function f evaluates componentwise at its input vector, we can get

$$\frac{d}{dt}\tilde{\mathbf{y}}(t) = \mathbf{V}_k^T\mathbf{B}\mathbf{V}_k\tilde{\mathbf{y}}(t) + \underbrace{\mathbf{V}_k^T\mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}}_{\text{precomputed:}k\times m}\underbrace{\mathbf{f}(\mathbf{P}^T\mathbf{V}_k\tilde{\mathbf{y}}(t))}_{m\times 1}. \tag{2.18}$$

Note that the term $\mathbf{V}_k^T\mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}$ in (2.18) does not depend on t, and therefore it can be precomputed before solving the system of ODEs. Note also that $\mathbf{P}^T\mathbf{V}_k\tilde{\mathbf{y}}(t) \in \mathbb{R}^n$

in (2.18) can be obtained by extracting the rows $\mathbf{p}_1, \cdots, \mathbf{p}_m$ of $\mathbf{V}_k$ and then multiplying against $\tilde{\mathbf{y}}$, which requires $2mk$ operations. Therefore, if $\alpha(m)$ denotes the cost of evaluating m components of f, the complexity for computing this approximation of the nonlinear term becomes roughly $O(\alpha(m) + 4km)$, which is independent of dimension n of the full-order system (2.14).

The POD-DEIM model reduction method consists of two stages. In the offline stage, snapshots for the solution states and nonlinear functions are generated using experimental data or by running the full-order model several times using different given inputs, for example, boundary conditions. The boundary conditions that generate representative solution space may not be known a priori. Then, we perform POD on the snapshots to get the reduced subspaces for the states, and interpolants for the nonlinear functions. The offline stage possibly has a high numerical cost, but it is performed only once. In the online stage, the approximate solution of the problem is obtained by writing the solution as a linear combination of the reduced basis, which was fixed at the offline phase. The reduced model can be used in the online stage many times to yields rapidly the outputs of interest. Next, we describe in more details the aforementioned process.

2.2.1 Proper Orthogonal Decomposition (POD)

The POD bases in Euclidean space are specified formally in this section. We refer to [81] for more details on the POD bases in general Hilbert space.

Given a set of snapshots $\{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{n_s}\} \in \mathbb{R}^n$, let $\mathbb{Y} = \text{span}\{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{n_s}\}$ and $r = \dim(\mathbb{Y})$. A POD bases of dimension $k < r$ consists of a set of orthonormal vectors $\{\boldsymbol{\phi}_i\}_{i=1}^k \in \mathbb{R}^n$ whose linear span best approximates the space $\mathbb{Y}$. The basis set $\{\boldsymbol{\phi}_i\}_{i=1}^k \in \mathbb{R}^n$ is derived by solving the minimization problem

$$\min_{\{\boldsymbol{\phi}_i\}_{i=1}^k} \sum_{j}^{n_s} \left\| \mathbf{y}_j - \sum_{i=1}^{k} (\mathbf{y}_j^T \boldsymbol{\phi}_i) \boldsymbol{\phi}_i \right\|_2^2 \tag{2.19}$$

with $\boldsymbol{\phi}_i^T \boldsymbol{\phi}_j = \delta_{ij}$, where δ_{ij} is the Kronecker delta.

It is easy to get that the solution to (2.19) is given by the set of the left singular vectors of the *snapshot matrix* $\mathbb{S}_y = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{n_s}] \in \mathbb{R}^{n \times n_s}$. Applying SVD to $\mathbb{S}_y$, we get

$$\mathbb{S}_y = \mathbf{V}\ \mathbf{\Lambda}\ \mathbf{W}^T, \tag{2.20}$$

where $\mathbf{V} = [\mathbf{v}_1, \cdots, \mathbf{v}_r] \in \mathbb{R}^{n \times r}$, $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_r] \in \mathbb{R}^{n_s \times r}$ are the left and right projection matrices, and $\mathbf{\Lambda} = \text{diag}(\sigma_1, \cdots, \sigma_r) \in \mathbb{R}^{r \times r}$ with $\sigma_1 \geq \sigma_2 \geq \cdots \sigma_r > 0$. The rank of $\mathbb{S}_y$ is $r \leq \min(n, n_s)$. Then the POD bases are $\{\mathbf{v}_i\}_{i=1}^k$. The minimum 2-norm error from approximating the snapshots using the POD bases is then given by

$$\sum_j^{n_s} \left\| \mathbf{y}_j - \sum_{i=1}^{k} (\mathbf{y}_j^T \mathbf{v}_i) \mathbf{v_i} \right\|_2^2 = \sum_{i=k+1}^{r} \sigma_i^2. \tag{2.21}$$

The appropriate number of POD bases (say N_p) to be retained for a prescribed accuracy can be determined, for example, by means of the fractional energy

$$E = \frac{\sqrt{\sum_{i=1}^{N_p} \sigma_i^2}}{\sqrt{\sum_{i=1}^{n_s} \sigma_i^2}}. \tag{2.22}$$

(see discussions in [112] for selecting modes). A small number of POD basis is retained if the singular values decay fast. This decay depends on the intrinsic dynamics of the system and the selection of the snapshots. In the next section, we review DEIM, which is applied to reduce the nonlinear terms.

2.2.2 Discrete Empirical Interpolation Method (DEIM)

In the presence of nonlinearity, the standard POD-Galerkin projection method reduces dimension in a way that far fewer degrees of freedom are calculated, but the complexity of evaluating the nonlinear terms remains that of the fine problem. DEIM combines both in-

terpolation and projection to improve the dimension reduction efficiency of POD-Galerkin projection method. Specifically, DEIM works in the following way.

In the offline stage, we collect the snapshots of the nonlinear function $\mathbf{f}(\mathbf{y}(t))$ as:

$$\{\mathbf{f}(\mathbf{y}(t_1)), \cdots, \mathbf{f}(\mathbf{y}(t_M))\} \in \mathbb{R}^n.$$

To obtain the DEIM basis $\mathbf{U} \in \mathbb{R}^{n \times m}$, we perform POD to the snapshot matrix (with the above snapshots as column vectors) and select the first m eigen-vectors corresponding to the dominant eigen-values. We approximate $\mathbf{f}$ as

$$\mathbf{f} = \mathbf{U}\mathbf{c}.$$

Then interpolation is used to compute the coefficients $\mathbf{c}$. The interpolation points are obtained by the DEIM Algorithm 1 [25]. With $\mathbf{U}$ as input, DEIM selects inductively m distinct interpolation points

$$\{p_1, \cdots, p_m\} \in \{1, \cdots, n\}$$

and assemble the DEIM interpolation points matrix $\mathbf{P} = [\mathbf{e}_{p_1}, \cdots, \mathbf{e}_{p_m}] \in \mathbb{R}^{n \times m}$, where $\mathbf{e}_i \in \{0, 1\}^n$ is the i-th canonical unit vector. In Algorithm 1, the process starts from choosing the first interpolation index $j_1 \in \{1, \cdots, n\}$ with respect to the entry of the first POD basis with largest magnitude. The remaining interpolation indices, j_i for $i = 2, \cdots, m$ are selected so that each of them corresponds to the entry with the largest magnitude of the residual $\mathbf{r} = \mathbf{U}(:, i) - \mathbf{V}\mathbf{c}$ from line 7 of Algorithm 1. The term $\mathbf{r}$ is the error between the input $\mathbf{U}(:, i)$ and its approximation $\mathbf{V}\mathbf{c}$ from interpolating the bases $\{\mathbf{U}(:, 1), \cdots, \mathbf{U}(:, i-1)\}$ at the indices $\{j_1, \cdots, j_{i-1}\}$.

The DEIM interpolant of $\mathbf{f}$ is defined by $(\mathbf{U}, \mathbf{P})$ as

$$\mathbf{f}(\mathbf{y}(t)) \approx \mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T\mathbf{f}(\mathbf{y}(t)),$$

where $\mathbf{P}^T\mathbf{f}(\mathbf{y}(t))$ samples $\mathbf{f}$ at m components only.

The error bound for the DEIM approximation is given in LEMMA 2.2.1, see [25].

Lemma 2.2.1. *Let $\mathbf{f} \in \mathbb{R}^n$ be an arbitrary vector. $(\mathbf{U}, \mathbf{P})$ are as defined in Algorithm 1, The DEIM approximation for $\mathbf{f}$ of order $m \leq n$ is*

$$\hat{\mathbf{f}} = \mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T\mathbf{f}.$$

An error bound for $\hat{\mathbf{f}}$ is given by

$$\|\mathbf{f} - \hat{\mathbf{f}}\|_2 \leq C\epsilon(\mathbf{f}),$$

where $C = \|(\mathbf{P}^T\mathbf{U})^{-1}\|_2$, and $\epsilon(\mathbf{f}) = \|(\mathbf{I} - \mathbf{U}\mathbf{U}^T)\mathbf{f}\|_2$ is the error of the best 2-norm approximation for $\mathbf{f}$ from the space Range($\mathbf{U}$). The constant C is bounded by

$$C \leq \frac{(1+\sqrt{2n})^{m-1}}{|\mathbf{e}_{p_1}^T\mathbf{U}(:,1)|} = (1+\sqrt{2n})^{m-1}\|\mathbf{U}(:,1)\|_\infty^{-1}.$$

We give an example of reducing a 2-Dimensional nonlinear parametrized function (modified from the one given in [62]) by using DEIM. Consider a function $f : \Omega \times \mathbb{D} \to \mathbb{R}$ defined as:

$$f(x, y; \boldsymbol{\mu}) = \frac{1}{\sqrt{(x-\mu_1)^2 + (y-\mu_2)^2 + \mu_1^2}}, \tag{2.23}$$

where $(x, y) \in \Omega = [0.01, 0.99] \times [0.01, 0.99]$ and $\boldsymbol{\mu} = (\mu_1, \mu_2) \in [-1, 0] \times [-1, 0]$. We uniformly discretize the domain Ω into rectangles, with 20 sub-intervals in both x and

Algorithm 1 Interpolation points for DEIM

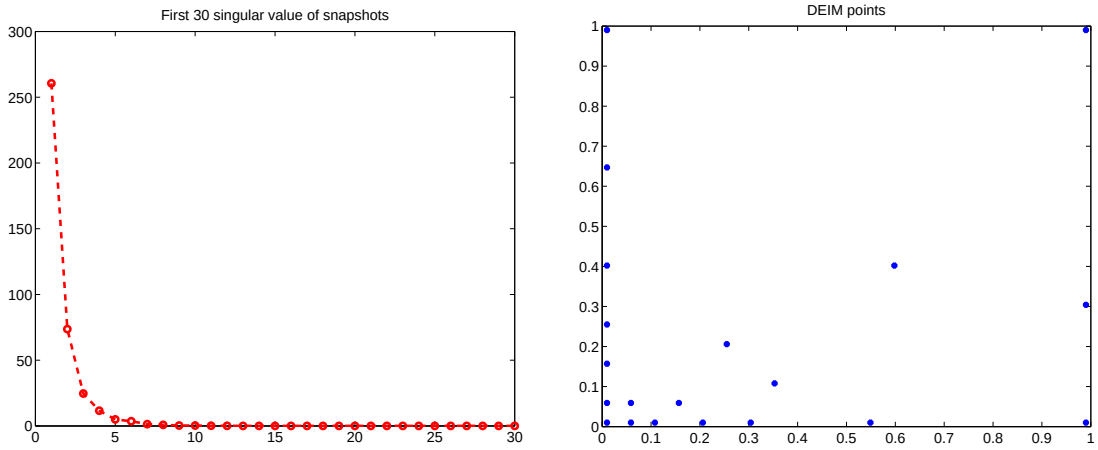
Notation: $\mathbf{e}_{s,k}$ is a standard coordinate vector of length s and with 1 at position k. Brackets $[\quad]$ denote concatenation of two or more matrices. $\mathbf{U}(:,i)$ denotes the i_{th} column of matrix $\mathbf{U}$.

Input : Global POD basis matrix $\mathbf{U}$
$(n, m) \leftarrow$ size($\mathbf{U}$)
$j_1 \leftarrow$ the position of $\mathbf{U}(:,1)$ with largest absolute entry
$\mathbf{V} = [\mathbf{U}(:,1)], \mathbf{P} = [\mathbf{e}_{n,j_1}]$
for $i = 2 : n$ **do**
  $\mathbf{c} \leftarrow \mathbf{P}^T\mathbf{V}\mathbf{c} = \mathbf{P}^T\mathbf{U}(:,i)$
  $\mathbf{r} \leftarrow \mathbf{U}(:,i) - \mathbf{V}\mathbf{c}$
  $j_i \leftarrow$ the position of $\mathbf{r}$ with largest absolute entry
  $\mathbf{V} \leftarrow [\mathbf{V}, \mathbf{U}(:,i)], \mathbf{P} \leftarrow [\mathbf{P}, \mathbf{e}_{n,j_i}]$
end for
Output : $\mathbf{P}$

y directions. Therefore, the full dimension is 441. To generate the snapshots, we select parameters $\boldsymbol{\mu}^{\text{snap}} = (\mu_1^{\text{snap}}, \mu_2^{\text{snap}})$ uniformly from the parameter domain $\mathbb{D}$. A set of 256 pairs of parameters are used for the snapshots. In this case, each snapshot corresponds to a column vector of function values for a given pair of parameter at the 441 grid points. To capture 99.9% of energy, 11 out of 256 basis must be retained. The first 30 singular values of these snapshots are shown in Figure 2.1a. Figure 2.1b presents the locations of the first 20 spatial points chosen by the DEIM algorithm using the POD basis as input. Since the function increase fast near the origin, most of the selected points distribute close to the origin. In Figure 2.2, we pick any pair parameter $\boldsymbol{\mu} = (-0.02, -0.5)$, and compare the approximation from POD, DEIM with the nonlinear function of dimension 441. We use 11 basis for these approximations, and achieve a good accuracy.

2.3 Coarse and fine grids

Local model reduction techniques such as MsFEM requires the use of coarse and fine grids. We introduce the notion of coarse and fine grids here. First we divide the compu-



(a) Singular values. (b) First 20 DEIM points selected by DEIM.

Figure 2.1: First 30 singular values and first 20 DEIM points.

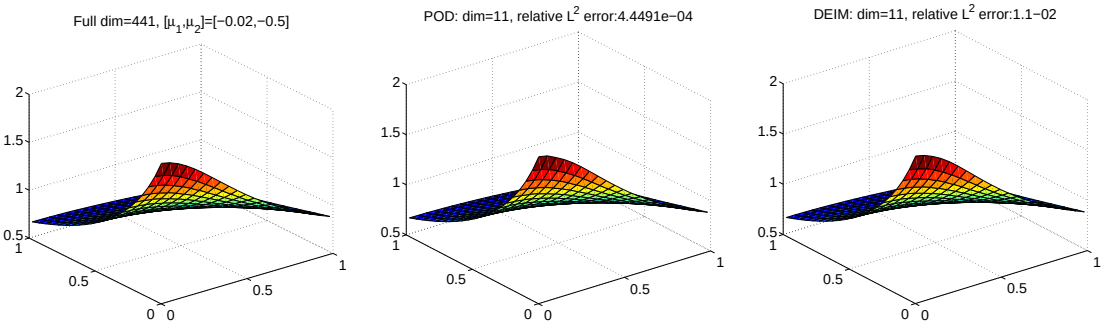


Figure 2.2: Comparison of the nonlinear function of dimension 441 with the POD and DEIM approximation of dimension 11 at $\mu = (-0.02, -0.5)$.

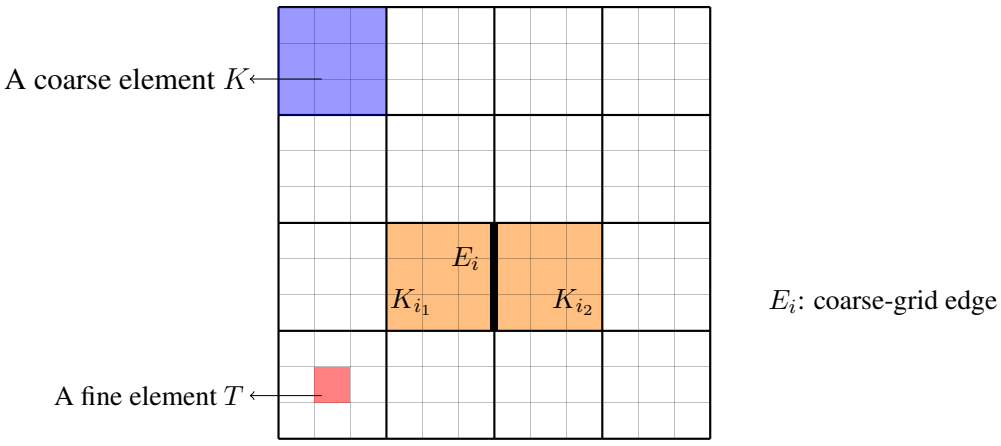


Figure 2.3: Illustration of coarse and fine grids.

tational domain Ω into non-overlapping polygonal coarse blocks K_i with diameter $O(H_i)$ so that $\overline{\Omega} = \cup_{i=1}^{N} \overline{K}_i$. We allow nonconforming decomposition for these blocks. We call E_H a coarse face (edge) of the coarse block K_i if $E_H = \partial K_i \cap \partial K_j$ or $E_H = \partial K_i \cap \partial\Omega$. Let $\mathcal{E}_H(K_i)$ be the set of all coarse edges of a coarse block K_i and $\mathcal{E}_H = \cup_{i=1}^{N} \mathcal{E}_H(K_i)$.

In each coarse block K_i, we introduce a shape regular discretization $\mathcal{T}_h(K_i)$ with rectangle elements (denoted as $T_j, j = 1, \cdots, i_n$) of size h_i. We denote the faces (edges) of T_j by e_h. Let $\mathcal{T}_h = \cup_{i=1}^{N} \mathcal{T}_h(K_i)$, $\mathcal{E}_h(K_i)$ be the set of all faces of the discretization $\mathcal{T}_h(K_i)$ and $\mathcal{E}_h^0(K_i)$ be the set of all interior edges of the dicretization $\mathcal{T}_h(K_i)$ and set $\mathcal{E}_h = \cup_{i=1}^{N} \mathcal{E}_h(K_i)$. Figure 2.3 gives an illustration of the design of the two grids, including neighborhoods and elements subordinated to the coarse discretization.. The black lines represent the coarse grid, and the gray lines represent the fine grid. For each coarse edge E_i, we define a coarse neighborhood ω_i as the union of all coarse blocks containing the edge E_i, which is the orange area in the figure. In the following, we introduce briefly the GMsFEM that is developed based on these two-level grids.

2.4 Local model reduction method via GMsFEM

Subsurface fluid flow problems usually exhibit multiscales and high contrast. The recently developed GMsFEM is one of the most popular methods to deal with such multicale problems. The main idea of GMsFEM is to construct a small dimensional solution space that can be used to approximate the multiscale solution with certain accuracy in an efficient manner. In contrast to the bases in the global reduction framework, the bases from GMsFEM are constructed locally on the coarse grid.

For the flow-transport system (2.11)-(2.13), the simulation of the flow problem accounts for most of the simulation time, mainly due to the high variability and the strong heterogeneity exhibited in the permeability field. In this respect, multiscale methods to simulate the flow problem in the forward simulations can save offline computational time. Moreover, since an accurate velocity field with conservative property from the flow equation is important for the simulation of the transport problem, we will consider numerical methods based on the mixed form of the flow problem.

For the sake of simplicity, we consider the following second order elliptic differential equation in mixed form:

$$\boldsymbol{u} + \kappa \nabla p = 0 \qquad \text{in } \Omega, \tag{2.24a}$$

$$\nabla \cdot \boldsymbol{u} = f \qquad \text{in } \Omega, \tag{2.24b}$$

with Neumann boundary condition $\mathbf{u} \cdot \mathbf{n} = 0$, where $\Omega \subset \mathbb{R}^d$ $(d = 2, 3)$ is a bounded polyhedral domain with outward unit normal vector $\boldsymbol{n}$ on the boundary, $f \in L^2(\Omega)$, κ represents the permeability field that varies over multiple spacial scales.

Here, we take the recently developed mixed GMsFEM [31] for illustration, where fine-scale features are incorporated into a set of coarse-grid basis functions for the flow

velocities. By using the multiscale basis functions, we can retain efficiency of solving the flow equation on a coarse grid, while at the same time yield a conservative velocity field on the underlying fine grid. The main idea of the mixed GMsFEM is to divide the computation into offline and online stages. During the offline stage, we construct a snapshot space and then the offline space via spectral decomposition of the snapshot space. At the online stage, for each input space element, a low-dimensional space in each coarse block is generated by solving local problems in the offline space. The snapshot space should be large enough so that the snapshot vectors preserve the essential properties of the solution and provide a good approximation space. The main feature of the offline space is that it gives a good solution approximation with fewer basis functions.

As defined in the last section, the coarse neighborhood of the coarse face E_i is $\omega_i = \bigcup\{K_j \in \mathcal{T}^{\mathcal{H}}; E_i \in \partial K_j\}$. Let N_e be the number of coarse faces. Like in [31], we construct multiscale basis functions for each face $E_i, 1 \leq i \leq N_e$, denoted by $\boldsymbol{\Psi}_k^{i,\text{off}}, 1 \leq k \leq l_i$, where l_i is the number of basis chosen for face i. The support of these basis functions is ω_i. The offline space is then constructed as:

$$\mathbf{V}_{\text{off}} = \text{span}\{\boldsymbol{\Psi}_k^{i,\text{off}} : 1 \leq k \leq l_i, 1 \leq i \leq N_e\}. \tag{2.25}$$

Using single-index notation:

$$\mathbf{V}_{\text{off}} = \text{span}\{\boldsymbol{\Psi}_k^{\text{off}} : 1 \leq k \leq l_{\text{off}}\}, \tag{2.26}$$

where l_{off} is the number of velocity basis from the offline stage. Furthermore, we define

$$\mathbf{R}_{\text{off}} = [\boldsymbol{\psi}_1^{\text{off}}, \ldots, \boldsymbol{\psi}_{l_{\text{off}}}^{\text{off}}], \tag{2.27}$$

which maps from the offline space to the fine space, where $\boldsymbol{\psi}_i^{\text{off}}$is a vector containing the

coefficients in the expansion of $\mathbf{\Psi}_i^{\text{off}}$ in the fine-grid basis functions.

Let $\mathbf{V}_H = \mathbf{V}_{\text{off}}$ and W_H be the space of piecewise constant functions with respect to the coarse grid, and define $\mathbf{V}_H^0 = \mathbf{V}_H \bigcap \{\mathbf{u} \in \mathbf{V}_H : \mathbf{u} \cdot \mathbf{n} = 0 \text{ on } \partial\Omega\}$ as a subspace of $\mathbf{V}_H$ consisting of elements with zero normal component on $\partial\Omega$.

The online stage is to find $(p_H, \mathbf{u}_H) \in Q_H \times \mathbf{V}_H^0$ such that

$$\begin{aligned} \int_\Omega \kappa^{-1}\mathbf{v}_H \cdot \mathbf{u}_H - \int_\Omega p_H \nabla \cdot \mathbf{v}_H &= 0 \quad \forall\, \mathbf{v}_H \in \mathbf{V}_H^0, && (2.28) \\ \int_\Omega \nabla \cdot \mathbf{u}_H z_H &= \int_\Omega f z_H \quad \forall\, z_H \in Q_H. && (2.29) \end{aligned}$$

Let $Q_h \times \mathbf{V}_h$ be the piecewise constant polynomials and the standard lowest-order Raviart-Thomas space for (2.24a)-(2.24b) on the fine grid $\mathcal{T}_h$. Then the fine grid solution $(p_h, \mathbf{u}_h) \in Q_h \times \mathbf{V}_h^0$ satisfies:

$$\begin{aligned} \int_\Omega \kappa^{-1}\mathbf{v}_h \cdot \mathbf{u}_h - \int_\Omega p_h \nabla \cdot \mathbf{v}_h &= 0 \quad \forall\, \mathbf{v}_h \in \mathbf{V}_h^0, && (2.30) \\ \int_\Omega \nabla \cdot \mathbf{u}_h z_h &= \int_\Omega f z_h \quad \forall\, z_h \in Q_h, && (2.31) \end{aligned}$$

with $\mathbf{V}_h^0 = \mathbf{V}_h \bigcap \{\mathbf{u} \in \mathbf{V}_h : \mathbf{u} \cdot \mathbf{n} = 0 \text{ on } \partial\Omega\}$. Its matrix representation is:

$$\begin{pmatrix} \mathbf{B} & -\mathbf{C}^T \\ \mathbf{C} & 0 \end{pmatrix} \begin{pmatrix} \vec{\mathbf{u}}_h \\ \vec{p}_h \end{pmatrix} = \begin{pmatrix} 0 \\ \mathbf{F}_h \end{pmatrix}, \tag{2.32}$$

where $\vec{\mathbf{u}}_h, \vec{p}_h$ are vectors of coefficients in the expansions of the solutions $\mathbf{u}_h \in \mathbf{V}_h, p_h \in Q_h$ respectively.

The corresponding matrix form for (2.28)-(2.29) is

$$\begin{pmatrix} \mathbf{R}_{\text{off}}^T \mathbf{B} \mathbf{R}_{\text{off}} & -\mathbf{R}_{\text{off}}^T \mathbf{C}^T \mathbf{G}_H \\ \mathbf{G}_H^T \mathbf{C} \mathbf{R}_{\text{off}} & 0 \end{pmatrix} \begin{pmatrix} \vec{\mathbf{u}}_H \\ \vec{p}_H \end{pmatrix} = \begin{pmatrix} 0 \\ \mathbf{G}_H^T \mathbf{F}_h \end{pmatrix}, \tag{2.33}$$

where $\mathbf{G}_H$ is the restriction operator from Q_H into Q_h, and $\vec{\mathbf{u}}_H, \vec{p}_H$ are vectors of coefficients in the expansions of the solutions $\mathbf{u}_H, p_H$ in the spaces $\mathbf{V}_H$ and Q_H.

3. POD-DEIM MODEL REDUCTION FOR MULTI-PHASE FLOWS *

3.1 Introduction

In this section, we discuss the construction of reduced-order models based on POD-DEIM for multiphase flows in heterogeneous porous media, see [117]. The two-phase incompressible flow problem we consider here can be formulated as a coupled (flow-transport) system of a pressure equation and a nonlinear saturation (near hyperbolic) equation. We discretize the pressure equation by mixed finite element method, which solves for pressure and the Darcy velocity (flux) simultaneously. The velocity field obtained through mixed finite element method is local mass conservative. In the offline stage, the process of running training simulations to produce snapshots for velocity can be carried out via Mixed Generalized Multiscale Finite Element Method (MGMsFEM), which compares to the fine solver, can save computational cost. Since the velocity snapshots are local mass conservative, the POD bases obtained through POD are also mass conservative. This characteristic entails that only a few number of velocity POD bases should be retained. The reduced-order models developed this way have significant model-order reduction as will be shown in the numerical examples.

The rest of the section is organized as follows. In Section 3.2, we give the space and time discretization for the two-phase flow system, and briefly discuss the upstream upwinding scheme for the phase fluxes. In Section 3.3, we describe the construction of reduced model for the two-phase flow system by using POD-DEIM. In Section 3.4, we present several numerical examples to demonstrate the efficiency of our method.

*Reprinted with permission from "Fast Multiscale Reservoir Simulations With POD-DEIM Model Reduction" by Y. Yang, M. Ghasemi, E. Gildin, Y. Efendiev and V. Calo, *SPE Journal*, doi:10.2118/173271-PA, 2016. Copyright 2016 by SPE.

3.2 Discretization

We solve the pressure equation (2.10) with a mixed finite element method [52] which solves for pressure and velocity simultaneously. The velocity field obtained in this way is local mass conservative. We will explain the details as follows.

From (2.5) and (2.9), we get the following first-order system

$$\begin{aligned} (\lambda_t(s)\kappa)^{-1}\mathbf{u} + \nabla p &= \mathbf{f}_z, \\ \nabla \cdot \mathbf{u} &= q_t, \end{aligned} \tag{3.1}$$

where $\mathbf{f}_z = (\lambda_t(s)\kappa)^{-1}\Big(- \kappa g(\lambda_w\rho_w + \lambda_o\rho_o)\nabla z\Big)$.

Let $Q_h, \mathbf{V}_H, \mathbf{V}_H^0$ be as defined in Section 2.4. Then the fine grid solution $(p_h, \mathbf{u}_h) \in Q_h \times \mathbf{V}_h^0$ satisfies:

$$\int_\Omega (\lambda_t(s)\kappa)^{-1}\mathbf{v}_h \cdot \mathbf{u}_h - \int_\Omega p_h \nabla \cdot \mathbf{v}_h = \int_\Omega \mathbf{f}_z \cdot \mathbf{v_h} \quad \forall\, \mathbf{v}_h \in \mathbf{V}_h^0, \tag{3.2}$$

$$\int_\Omega \nabla \cdot \mathbf{u}_h z_h = \int_\Omega f z_h \quad \forall\, z_h \in Q_h. \tag{3.3}$$

Written in matrix form, we get:

$$\begin{pmatrix} \mathbf{B}(s) & -\mathbf{C}^T \\ \mathbf{C} & 0 \end{pmatrix} \begin{pmatrix} \vec{\mathbf{u}}_h \\ \vec{p}_h \end{pmatrix} = \begin{pmatrix} \mathbf{F}1 \\ \mathbf{F}2 \end{pmatrix}, \tag{3.4}$$

where $\mathbf{B}(s) = \{b_{ij}\}$, $\mathbf{C} = \{c_{ik}\}$, $\mathbf{F}1 = \{f1_k\}$ and $\mathbf{F}2 = \{f2_k\}$ are defined as,

$$b_{ij} = \int_{T_i} \big(\lambda_t(s)\kappa\big)^{-1}\boldsymbol{\psi}_i \cdot \boldsymbol{\psi}_j\ , \quad c_{ik} = \int_{T_i} \phi_k \nabla \cdot \boldsymbol{\psi}_i, \tag{3.5}$$

$$f1_k = \int_{T_k} \boldsymbol{\psi}_k \cdot \mathbf{f}_z, \quad f2_k = \int_{T_k} \phi_k q_t. \tag{3.6}$$

Here $\{\boldsymbol{\psi}_i\}$ are basis in $\mathbf{V}_H^0$, and $\{\phi_k\}$ are basis in Q_h.

Next we describe the discretization for the transport equation:

$$\phi\frac{\partial s}{\partial t} + \nabla \cdot \mathbf{u}_w \quad = \quad q_w \quad \text{in} \quad \Omega, \tag{3.7}$$

where

$$\mathbf{u}_w = f_w(\mathbf{u} - \kappa g \lambda_o(\rho_w - \rho_o)\nabla z) = f_w \mathbf{u} - f_2 \mathbf{C}_g, \tag{3.8}$$

with $\mathbf{C}_g = \kappa g(\rho_w - \rho_o)\nabla z$, $f_2 = f_w\lambda_o$. We use the fully-implicit backward Euler method for the temporal discretization, and finite volume method for the spatial discretization.

Consider a control-volume $\tilde{T}_i$ with edges $\tilde{e}_{ij}$ and associated normal vectors $\mathbf{n}_{ij}$ pointing out of $\tilde{T}_i$. The discrete conservation equation of the water phase (3.7), for $\tilde{T}_i$ can be written as

$$s_i^{n+1} = s_i^n + \frac{\Delta t}{|\tilde{T}_i|\phi}\left(q_t^+ - \sum_j F_{ij}(s_i, s_j)\mathbf{u}_{ij}\cdot\mathbf{n}_{ij} + \sum_j G_{ij}(s_i, s_j)\mathbf{C}_g \cdot \mathbf{n}_{ij} + f_w(s_i^{n+1})q_t^-\right), \tag{3.9}$$

where the superscript n represents time step, s_i^n is the cell-average of the water saturation at time $t = t_n$, $q^+ = \max(q_{ti}, 0)$, $q^- = \min(q_{ti}, 0)$, $\mathbf{u}_{ij}$ is the total viscous flux over the edge $\tilde{e}_{ij}$ between the two adjacent cells, and F_{ij}, G_{ij} are numerical approximation of the viscous fraction flow function and gravitational fraction flow over the edge $\tilde{e}_{ij}$ respectively. Written in vector form, we get

$$\mathbf{S}^{n+1} = \mathbf{S}^n + \frac{\Delta t}{|\tilde{T}_i|\phi}\left(\mathbf{F}^{n+1} + \mathbf{G}^{n+1} + \mathbf{Q}_\mathbf{w}\right), \tag{3.10}$$

where $\mathbf{F}^{n+1} = \mathbf{F}(\mathbf{S}^{n+1}), \mathbf{G}^{n+1} = \mathbf{G}(\mathbf{S}^{n+1})$ represent the numerical approximation of the fluxes from viscous and gravitational forces across all grid block interfaces respectively, and $\mathbf{Q}_\mathbf{w}$ denotes the vector for wells. In contrast to the linear pressure equation, the dis-

cretized saturation equation (3.10) is a system of nonlinear algebraic equations as the flux terms are nonlinearly dependent on the saturation. We use the iterative solver Newton-Raphon's method to solve this nonlinear system. The residual is defined as:

$$\mathbf{R}(\mathbf{S}) = \mathbf{S} - \mathbf{S}^n - \frac{\Delta t}{|\tilde{T}_i|\phi}\Big(\mathbf{F}(\mathbf{S}) + \mathbf{G}(\mathbf{S}) + \mathbf{Q_w}\Big), \tag{3.11}$$

and the Jacobian matrix is expressed as:

$$\mathbf{J}(\mathbf{S}) = \mathbf{I} - \frac{\Delta t}{|\tilde{T}_i|\phi}\left(\frac{\partial \mathbf{F}}{\partial \mathbf{S}} + \frac{\partial \mathbf{G}}{\partial \mathbf{S}}\right), \tag{3.12}$$

where $\mathbf{I}$ is the identity matrix with dimension equals to the number of grid blocks, and $\frac{\partial \mathbf{F}}{\partial \mathbf{S}}$, $\frac{\partial \mathbf{G}}{\partial \mathbf{S}}$ are matrices. For $\frac{\partial \mathbf{F}}{\partial \mathbf{S}}$, its row has the form: $\{\frac{\partial \mathbf{F}}{\partial s_1}, \frac{\partial \mathbf{F}}{\partial s_2}, \cdots, \frac{\partial \mathbf{F}}{\partial s_n}\}$. As can be seen in (3.9), for each row, $\mathbf{F}, \mathbf{G}$ only depend on a few saturations, therefore most of the elements in these matrices are zero.

We note that F, G in Equation (3.9) depend on the two saturations on either side of the interface $\tilde{e}_{ij}$: s_i and s_j. Some upstream upwinding should be applied to determine the dependency. Otherwise, the numerical solution may encounter oscillations, overshoots, undershoots (e.g. saturation less than zero or greater than 1), or converge to an incorrect solution.

We employ the implicit Hybrid Upwinding scheme as proposed in [84, 83, 67] for the numerical fluxes, such that the upstream for the numerical viscous flux is determined by the direction of the total velocity across the interface, and the upstream for the numerical buoyancy flux is fixed in the way that oil flows upward while water flows downward. Besides being locally conservative, and being able to yields monotone physically consistent numerical solutions, this approach is particularly beneficial in the POD-DEIM model reduction context, since the upwinding directions are fixed, making it possible to train the

snapshots for the nonlinear functions. The Hybrid Upwinding scheme is mathematically translated as the following:

The viscous F, and buoyancy G, fractional-flow functions at the interface between control-volumes i and j are expressed respectively as:

$$F(s_i, s_j) = F(s_{ij}), \quad G(s_i, s_j) = G(s_{ij}).$$

We define upwinding with respect to the direction of the total velocity as:

$$s_{ij} = \begin{cases} s_i & \text{if } \mathbf{u}_{ij} \cdot \mathbf{n}_{ij} \geq 0; \\ s_j & \text{if } \mathbf{u}_{ij} \cdot \mathbf{n}_{ij} < 0 \end{cases} \tag{3.13}$$

and the upwinding scheme for the numerical buoyancy flux G as:

$$s_{w,ij} = s_j \quad \text{and} \quad s_{o,ij} = 1 - s_i \quad \text{if} \quad (y_i - y_j)g \geq 0,$$

$$s_{w,ij} = s_i \quad \text{and} \quad s_{o,ij} = 1 - s_j \quad \text{if} \quad (y_i - y_j)g < 0.$$

In the next section, we will discuss the construction of reduced model for the coupled flow-transport system via POD-DEIM.

3.3 POD-DEIM model reduction

In this section, we describe the model reduction procedure iva POD-DEIM. We start with reducing the pressure equation using POD-projection. Projection based approaches construct reduced order models of order $r \ll n$ that can approximate the original system of order n from a subspace spanned by reduced bases (POD bases).

From the training simulations, we save the snapshots for the states (pressure, velocity and saturation), denoted as $\mathbb{S}_p$, $\mathbb{S}_{\mathbf{u}}$ and $\mathbb{S}_s$ respectively. Take pressure for example. Each

snapshot is a vector of pressure values at each grid element at some time step generated with some boundary condition. Then we perform SVD to these snapshot matrices to obtain the POD basis matrices $\Phi_{\mathbf{u}} \in \mathbb{R}^{n_u \times r_u}$, $\Phi_p \in \mathbb{R}^{n_c \times r_p}$ and $\Phi_s \in \mathbb{R}^{n_c \times r_s}$ with $r_u \ll n_u$, $r_p \ll n_c$ and $r_s \ll n_c$, where n_u is the total number of faces in the mesh. Replacing $\mathbf{u}$ by $\Phi_{\mathbf{u}}\mathbf{u}_r$, and p by $\Phi_p p_r$ in equation (3.2), and projecting (3.2) onto the subspaces spanned by the POD basis, one gets the reduced system for the pressure equation as,

$$\begin{pmatrix} \Phi_{\mathbf{u}}^T \mathbf{B}(s) \Phi_{\mathbf{u}} & -\Phi_{\mathbf{u}}^T \mathbf{C}^T \Phi_p \\ \Phi_p^T \mathbf{C} \Phi_{\mathbf{u}} & 0 \end{pmatrix} \begin{pmatrix} \vec{\mathbf{u}}_r \\ \vec{p}_r \end{pmatrix} = \begin{pmatrix} \Phi_{\mathbf{u}}^T \mathbf{F}1 \\ \Phi_p^T \mathbf{F}2 \end{pmatrix}, \tag{3.14}$$

where $\vec{\mathbf{u}}_r, \vec{p}_r$ are coefficients for the POD bases of velocity and pressure respectively.

Next we describe the reduction for the saturation equation (3.7). Replacing $\mathbf{S}$ in (3.11) and (3.12) by $\Phi_s \mathbf{S}_r$, and premultiplying by Φ_s^T, we get the reduced forms of the residual and Jacobian:

$$\mathbf{R}_r(\mathbf{S}) = \mathbf{S}_r - \mathbf{S}_r^n - \frac{\Delta t}{|\tilde{T}_i|\phi} \Phi_s^T \Big(\mathbf{F}(\Phi_s \mathbf{S}_r) + \mathbf{G}(\Phi_s \mathbf{S}_r) + \mathbf{Q}_{\mathbf{w}} \Big), \tag{3.15}$$

$$\mathbf{J}_r(\mathbf{S}) = \mathbf{I}_r - \frac{\Delta t}{|\tilde{T}_i|\phi} \Phi_s^T \left(\frac{\partial \mathbf{F}(\Phi_s \mathbf{S}_r)}{\partial \mathbf{S}} + \frac{\partial \mathbf{G}(\Phi_s \mathbf{S}_r)}{\partial \mathbf{S}} \right), \tag{3.16}$$

where $\mathbf{I}_r$ is the identity matrix with dimension r_s.

To avoid full evaluation of the nonlinear terms, we use DEIM to approximate the nonlinear functions on a set of inpterpolation points. These interpolation points are selected by the DEIM Algorithm 1 or 2 which are grid blocks that are essential in reconstructing the nonlinear terms with certain accuracy.

3.4 Numerical examples

In this section, several numerical examples are given. Relative saturation error and water cut are used to evaluate the accuracy of approximation solutions. Fine grid solution will be considered as reference solution. The relative saturation error at time step i is defined as

$$e_s^i := \frac{\|\mathbf{s}_{\text{red}}^i - \mathbf{s}_{\text{ref}}^i\|_{L^2}}{\|\mathbf{s}_{\text{ref}}^i\|_{L^2}}. \tag{3.17}$$

Water cut represents the fraction of water produced in relation to the total production. Saturation are computed at different pore volume injected (PVI). PVI is defined as

$$\text{PVI} = \int \frac{Q}{P_{\text{volume}}},$$

where P_{volume} is the total pore volume of the system, $Q = \int_{\partial\Omega^{\text{out}}} \mathbf{v} \cdot \mathbf{n}$ is the total flow rate, and $\partial\Omega^{\text{out}}$ is the outflow boundary.

3.4.1 Mass conservation in POD with finite volume method

POD Galerkin projection method does not necessarily honor mass conservation. A simple example in this section will be shown about the violation of mass conservation by POD projection based on finite volume method. One way to achieve mass conservation is by introducing an auxiliary variable, in our case, a mass-conservative velocity field. By constructing POD basis functions for the velocity field, we can guarantee that the velocity field is conservative because it consists of a linear combination of velocity basis functions which are mass conservative. This allows us to achieve higher degree of reduction.

In our simulations, there are no sources or sinks except in the well locations, where we enforce the exact mass conservation. In the non-well blocks, we need to conserve the mass by guaranteeing the sum of fluxes is zero. Because each basis function for the flux satisfies this property, thus, any of their linear combination will be mass conservative.

In the next example, we demonstrate how mass conservation can be violated. The reservoir model is a two-phase flow (oil-water) model under the water flooding recovery process with the structure of a 5-spot (four producers on the corners and one injector in the middle of the domain). The reservoir model is discretized using Cartesian grid of size $10\text{f}t \times 10\text{f}t \times 10\text{f}t$. Overall the reservoir model has $45 \times 45 \times 1 = 2025$ active cells. The permeability field of the reservoir is 10 (md) homogeneous and the porosity is 0.2. The relative permeability curves are quadratic.

All the producers are controlled by a constant bottom hole pressure at 2500 (psia). For the training input, the injector bottom hole pressure is 3750 (psia). The reservoir model was simulated for 1000 days by using finite volume method and the snapshots of the nonlinear fractional function were saved every 10 days. Therefore, all the states (pressure and saturation) have 100 snapshots. After applying SVD on the snapshot matrices, the pressure and saturation bases are obtained. We selected 13 pressure bases to preserve 0.9999 energy and 13 bases to preserve 0.99 saturation energy. These bases are used to construct the projection matrices to project fine scale states to the reduced subspaces. For the reduced model with the same exact input and boundary conditions as the training problem, the errors are small and the results are close to the fine scale solution. However, the reduced models are expected to be used in frameworks with different inputs from the training one. We perturbed the bottom hole pressure of the injector by $\pm 5\%$, as shown in Figure 3.1. Although, this is a small perturbation and it is only in one of the input variables, the results of the reduced model is far from the high fidelity solution as it is shown in Figure 3.2. Figure 3.2(b) presents the relative saturation error with respect to time step. We see that this error is more than 10% for most of the time steps. All the producers have the same water cut (red line) from the high fidelity simulation as shown in Figure 3.2(a), due to the symmetry in the problem. Note that the water cut from the reduced model(green) is very different towards end of simulation from the one(red) of the

fine model, because the mass conservation is violated in most of the gridblocks, as shown in Figure 3.3.

We also implemented POD in the context of a mixed finite element formulation of this example. After running the same training simulation and saving the snapshots, we selected 8 velocity basis to preserve 0.99 of its energy, 2 pressure basis to preserve 0.995 energy and 13 basis to preserve 0.99 saturation energy. The reduced model on mixed formulation not only results in small error for exact input, but also replicates a very similar results to the high fidelity model for test input as shown in Figure 3.4. The saturation error is smaller than 2% for most of simulation time.

These examples demonstrate that reduced order models that only use pressure field to construct reduced model, are sensitive to changes in boundary conditions. Thus, one need to have many basis to keep the error small or to reformulate the problem (as in the mixed finite volume) and solve for the velocity at the same time. This way we make sure the results are mass conservative and one does not need a lot of basis to obtain accurate results.

The next two examples are 2D, dimensionless. The computational domain considered is $\Omega = (0,1)^2$; the coarse grid $\mathcal{T}^H$ and the fine grid $\mathcal{T}^h$ are 22×22 and 220×220 uniform meshes respectively; the permeability field is shown in Figure 3.5; an injector is placed on the top-left and a producer is placed on the bottom-right; the rate is fixed as 2 for training and 4 for test simulation respectively; end of simulation time is 1000.

3.4.2 POD-DEIM model reduction

This example presents the results of using POD-DEIM model reduction to simulate the system (2.11)-(2.13). Note that the training simulation is carried out on the fine grid. In Table 3.1, the relative saturation errors at end of simulation time of using different number of basis for states and nonlinear functions, and the total number of Newton iterations

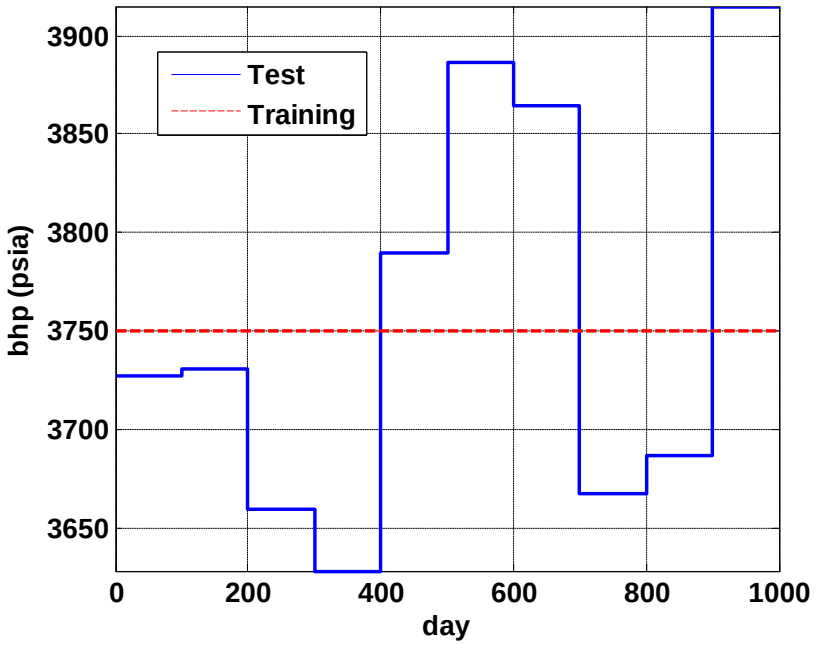
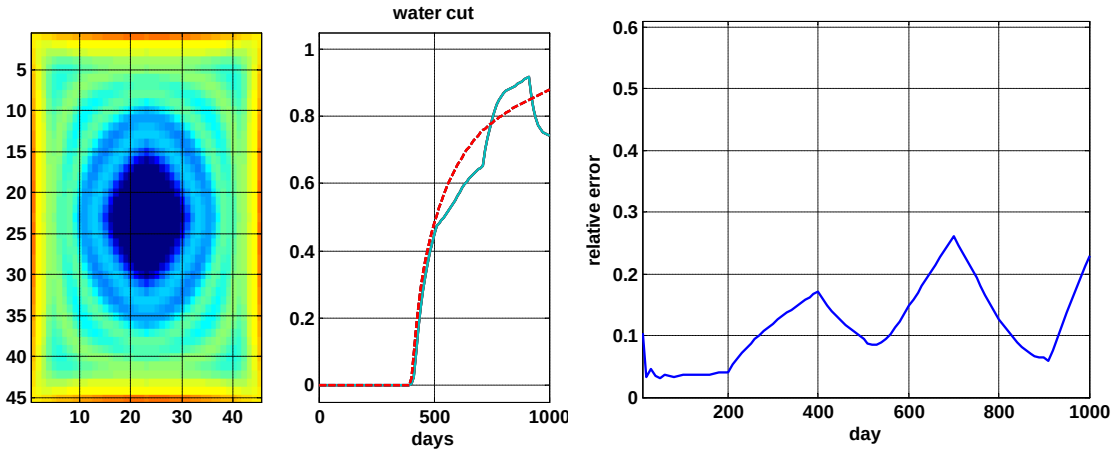


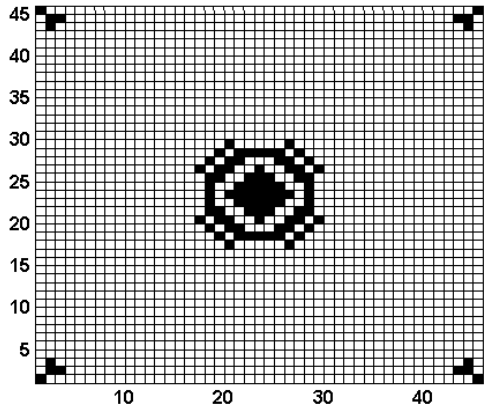
Figure 3.1: Training and test (perturbed) bottom hole pressure of the injector.



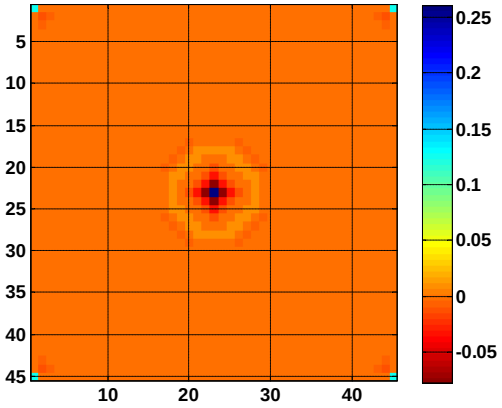
(a) Final water saturation and water cut.

(b) Water saturation error in reduced model.

Figure 3.2: POD model reduction on finite volume formulation.

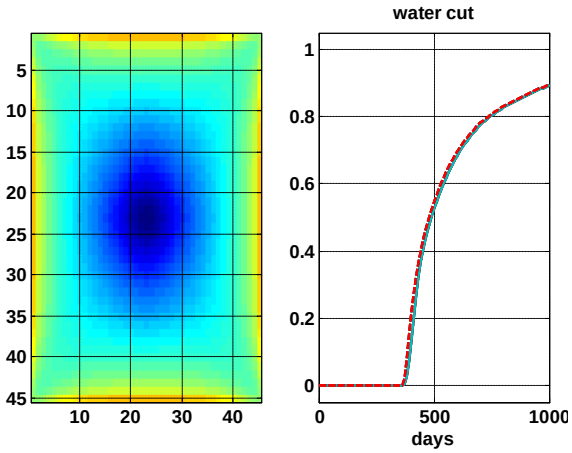


(a) Gridblocks where mass conservation does not hold.

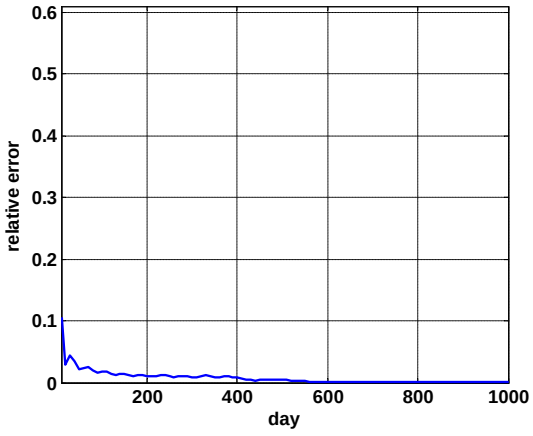


(b) Mass conservation violation scaled as PVI.

Figure 3.3: Evaluation of mass conservation at final time.



(a) Final water saturation and water cut.



(b) Water saturation error in reduced model.

Figure 3.4: POD model reduction on mixed finite element formulation.

Table 3.1: Relative saturation error, Global (POD-DEIM).

u-p-s-f_λ-f_w basis	error w.r.t fine	Newton iter
1-1-6-6-20	0.0306	279
1-1-8-6-20	0.0273	286
1-2-8-6-20	0.0272	286
2-2-8-2-20	0.0230	285
2-2-8-4-8	0.9577	313
2-2-8-4-10	0.0223	293
2-2-8-4-15	0.0227	288
2-2-8-4-20	0.0227	287
2-2-8-6-8	0.9417	313
2-2-8-6-20	0.0226	287

are given. The fine solution is used as reference solution. The total number of Newton iterations for the fine solution is 1299. In Table 3.1, if we fixed the number of POD basis for velocity, pressure and saturation as $2, 2, 8$ respectively, from the cases $2-2-8-4-8, 2-2-8-6-8$ with error greater than 90% and $2-2-8-4-10$ with error about 2%, we can see that the number of basis for the flux f_w should be greater than 8. The error here consists of the following two parts:

$$\|s_{\text{POD-DEIM}} - s_{\text{POD}}\| + \underbrace{\|s_{\text{POD}} - s_{\text{ref}}\|}_{\text{irreducible error}}.$$

With fixed number of POD bases for velocity, pressure and saturation, the irreducible error is fixed. Therefore, when the error touches a point (here about 2.2% which is small enough), adding more basis for the nonlinear functions won't help. But we can reduce the irreducible error by using more POD basis for velocity, pressure and saturation and therefore improve the whole error. But since 2.2% is sufficiently small for reservoir simulations, we consider that this is not necessary.

Table 3.2: Relative saturation error as defined in (3.17), Global-Local.

u-p-s-f_λ-f_w basis	error w.r.t fine	error w.r.t MS	Newton iter
1-1-6-6-20	0.038728	0.029552	273
1-1-8-6-20	0.036653	0.026391	278
1-2-8-6-20	0.034247	0.022856	278
2-2-8-2-20	0.032796	0.021027	278
2-2-8-4-8	0.53262	0.52982	308
2-2-8-4-10	0.11633	0.11379	315
2-2-8-4-11	0.041929	0.033591	309
2-2-8-4-15	0.032849	0.020842	285
2-2-8-6-8	0.58091	0.57806	308
2-2-8-6-20	0.033017	0.021201	279

3.4.3 Global-Local model reduction

As mentioned earlier, to save computational time for the offline stage, we can use model reduction techniques to simulate the training models. In this example, for the training simulation, the mixed GMsFEM is employed to solve the pressure equation. On every inner coarse edge, 5 multiscale velocity bases are selected. The dimension for velocity space is approximately 5% of the fine scale velocity space size. The relative L^2 error of saturation between the MGMsFEM solution and the fine solution at end of simulation time is about 2%. In Table 3.2, the relative L^2 errors of saturation at end of simulation time for different number of basis and total number of Newton iterations are given. The MGMsFEM solution is used as reference solution. The total number of Newton iterations for the reference solution is 1319. In this table, we see that with 2 basis for velocity, 2 for pressure and 8 for saturation, 4 DEIM points for f_λ and at least 11 DEIM points for the flux function, we can get less than 5% relative saturation error. The computational time of the local model reduction decreased by a factor of $\mathcal{O}(10^2)$ compared to that of the fine solution.

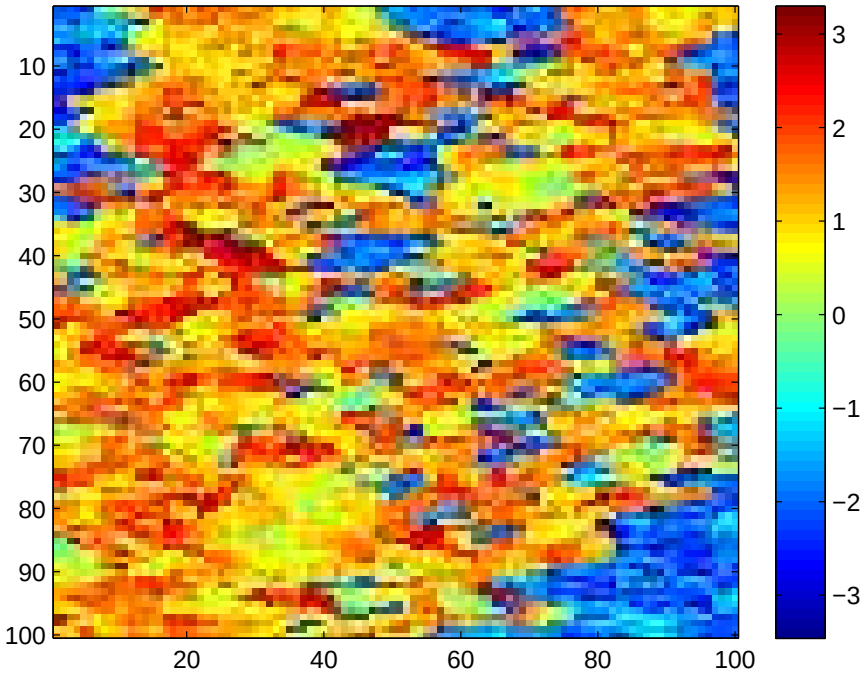
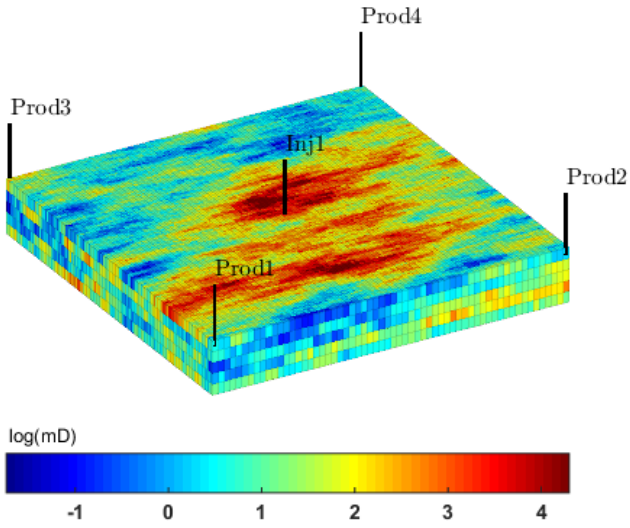


Figure 3.5: Logarithm of permeability field.

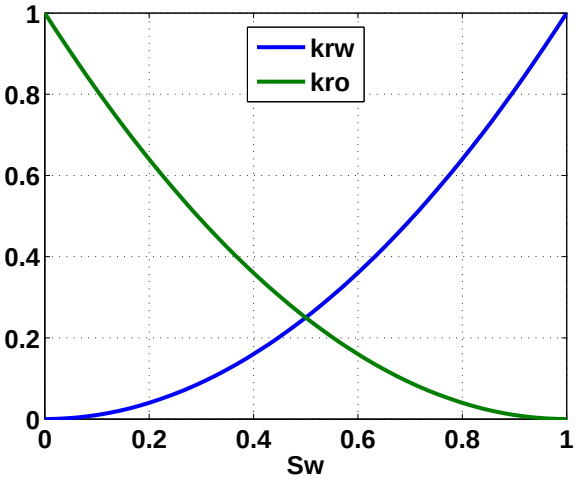
3.4.4 Case study with viscous only

In this section we apply the model reduction methodologies for a two-phase flow (oil-water) reservoir model under the water flooding recovery process with the structure of a 5-spot. Here we have an injector in the center of the reservoir and four producers in the corners, see Figure 3.6a, and it is assumed that all of them are perforated only at the bottom layer. The reservoir is SPE10 comparative model [28] (five layers of 10th-14th). This model is synthetic but can be representative of a real reservoir with large heterogeneity.

This reservoir is discretized using Cartesian grid of size $20\text{ft} \times 10\text{ft} \times 2\text{ft}$. Overall the reservoir model has $60 \times 220 \times 5 = 66000$ active cells. The fluid viscosity ratio is $\mu_w/\mu_o = 0.1$. The absolute heterogeneous permeability and the relative permeability curves are depicted in Figure 3.6a and Figure 3.6b, respectively. We assume a constant porosity of 0.2 for the entire model.

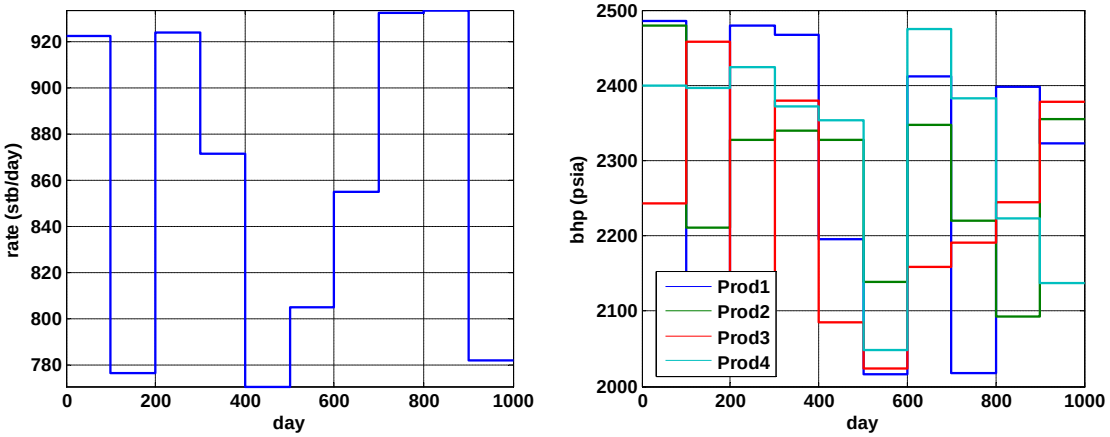


(a) Permeability field.



(b) Relative permeability.

Figure 3.6: SPE10 - 5 layers



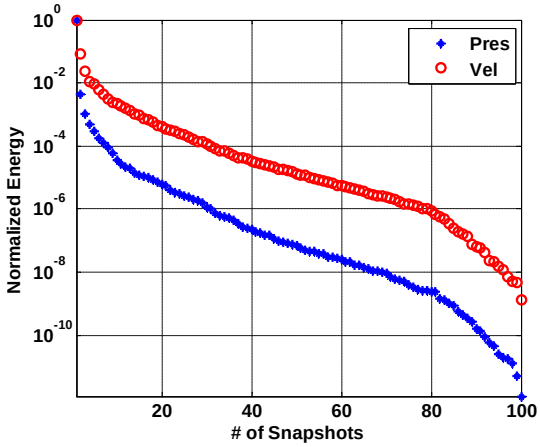
(a) Injection rates.

(b) Bottom hole pressure.

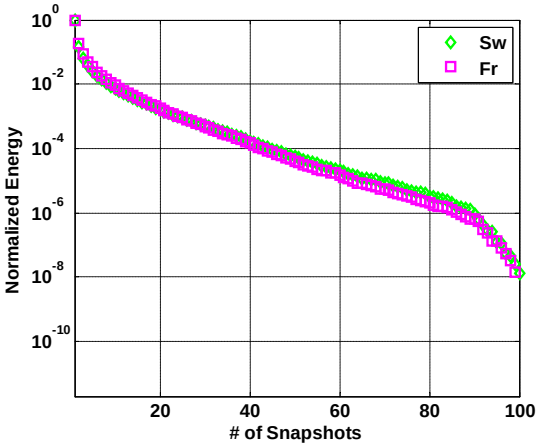
Figure 3.7: Training schedule.

For the training schedule, the producers are controlled by bottom hole pressure and the injector by injection rate, as shown in Figure 3.7. Note that this amount of injection is selected to ensure at least one pore volume is injected throughout simulation time (1000 days). The initial water saturation and pressure are assumed to be 0.0 and 2500 psia respectively.

In the offline stage, we simulate the reservoir for 1000 days and save the snapshots of pressures, velocity, water saturations and the nonlinear fractional function every 10 days. Thus, we have 100 snapshots for each variables. Each snapshot is reshaped to a column vector and is stacked in a snapshot matrix. After applying SVD to each matrix, one can find the basis as explained in previous sections (see Table 3.3 for the CPU time to obtain the basis). The singular values of the snapshot matrices are shown in Figure 3.8a and Figure 3.8b. As can be seen, there is a faster decay in the singular values for the pressure and velocity compared to saturation and fractional function. Thus, we need more basis for saturation and nonlinear functions to capture the most of the energy and to have small error.

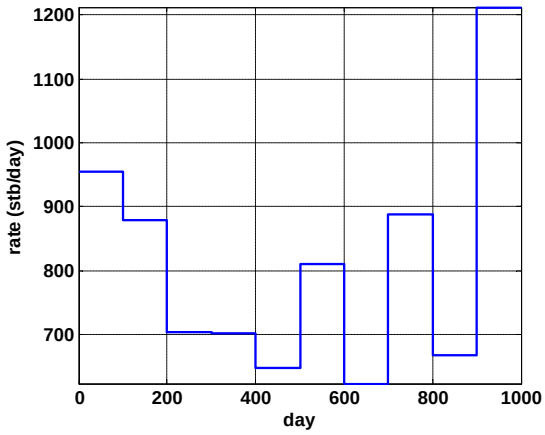


(a) Pressure (Pres) and velocity (Vel).

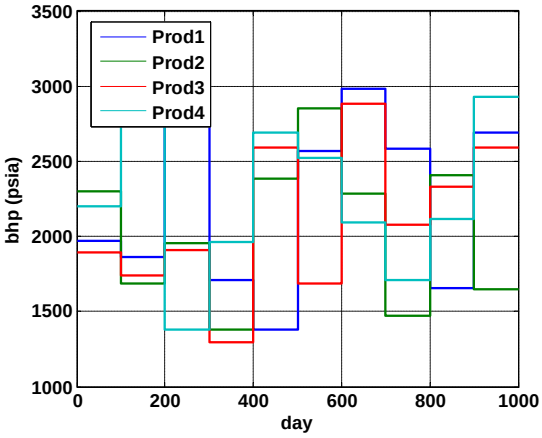


(b) Saturation (Sw) and fractional function (Fr).

Figure 3.8: Singular values of snapshot matrix.



(a) Injection rate.



(b) Bottom hole pressure.

Figure 3.9: Test schedule ($\pm 20\%$ random variation in training)

Table 3.3: Overhead time for calculating the basis.

	Overhead time (sec)
pressure basis	1.9
velocity basis	0.6
saturation basis	0.7
fractional flow basis	0.9

Table 3.4: Compare fine and reduced scale model.

	Fine Scale	POD-DEIM	Final Relative Error
number of pressure unknowns	66000	2	–
number of velocity unknowns	183400	12	–
number of saturation unknowns	66000	20	–
number of fractional flow unknowns	66000	25	–
pressure Eq. elapsed time (sec)	8309	91.8	0.01
saturation Eq. elapsed time (sec)	315	17.5	0.05
total simulation elapsed time (sec)	8627	112	–
speed-up	–	77	–

The selection criteria here was to capture at least 99% of the energy of snapshots. The number of basis is compared for reduced model to the original fine scale one in Table 3.4. For example, we reduce the dimension of velocity from 183400 to 2. It is obvious that several orders of magnitude in model order reduction is obtained in this example. The pressure equation runtime reduced more than 90 times and the saturation equation around 18 times. Overall the reduced model can be run 77 times faster than the original fine scale one.

This error is computed by Equation (3.17) with fine scale solution as the reference solution. The average error is less than 5% for most of the simulation time as shown in Figure 3.10b, indicating that the reduced model is a good approximation. The final water saturation at the bottom layer and water cut for all the producers after 1000 days of sim-

ulation are shown in Figure 3.10a and compared with fine scale model. Figure3.11b and Figure 3.11a show the spatial relative error at the final time in the pressure and saturation, respectively. The error in pressure is $\mathcal{O}(10^{-3})$, and in saturation $\mathcal{O}(10^{-1})$. Note that the error is usually larger in the cells around injector due to high dynamical fluid flow.

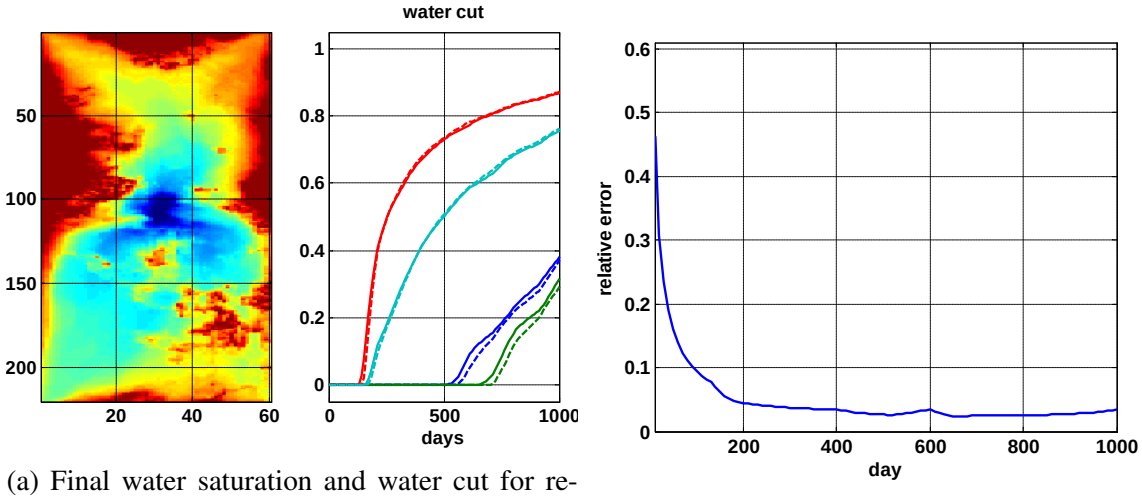
We run the reduced model with a new test schedule as shown in Figure 3.9, to make sure that the POD-DEIM model reduction is robust to input variation. This schedule is obtained by $\pm 20\%$ random perturbation of the training schedule. Note that the bases of the reduced model are not updated and we used the same bases obtained from training snapshots.

The final water saturation at the bottom layer and water cut for all the producers after 1000 days of simulation with the Test schedule is shown in Figure 3.12a and compared with fine scale model. Figure 3.13b and Figure 3.13a show the spatial relative error at the final time in the pressure and saturation, respectively. The error in pressure is still $\mathcal{O}(10^{-3})$, and in saturation $\mathcal{O}(10^{-1})$. Note that even though the error is around 0.2 in some of the cells around injector due to high dynamical fluid flow, the average saturation error is smaller than 5% for most of the simulation time as shown in Figure 3.12b.

3.4.5 Case study with gravity

The numerical example here is the five-spot 85th layer of the SPE10 comparative model with gravity in the y direction. Other settings of the wells are the same as in the case with viscous only.

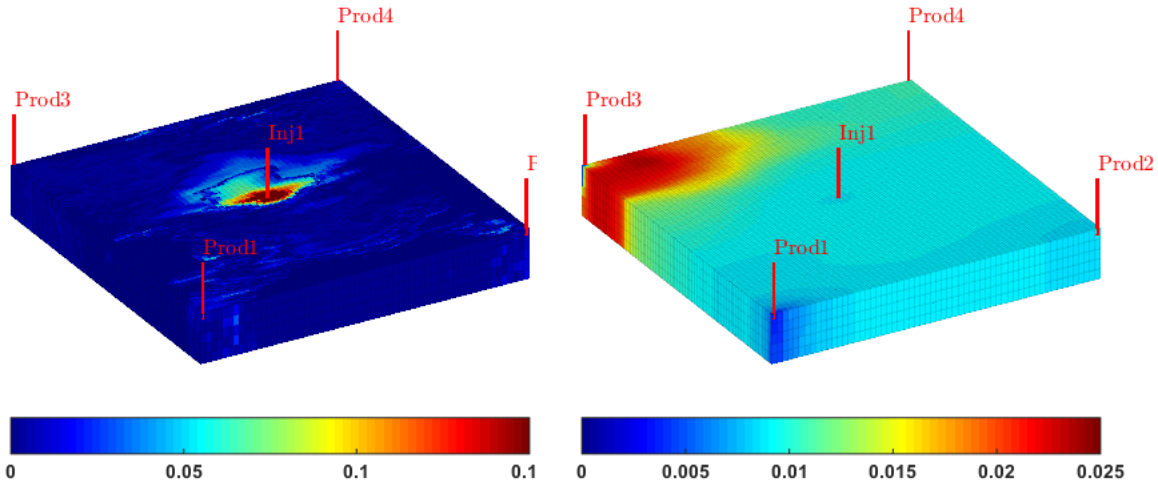
The challenging part of the method for gravity is that the same number of DEIM interpolation locations as the POD modes does not seem to be sufficient to give a good approximation of the nonlinear functions. We propose a modified DEIM algorithm, which allows more DEIM interpolation points. The details of the algorithm is given in Algorithm 2 (following [8]).



(a) Final water saturation and water cut for reduced (solid) and high fidelity (dashed) model.

(b) Relative saturation error advancing with respect to time.

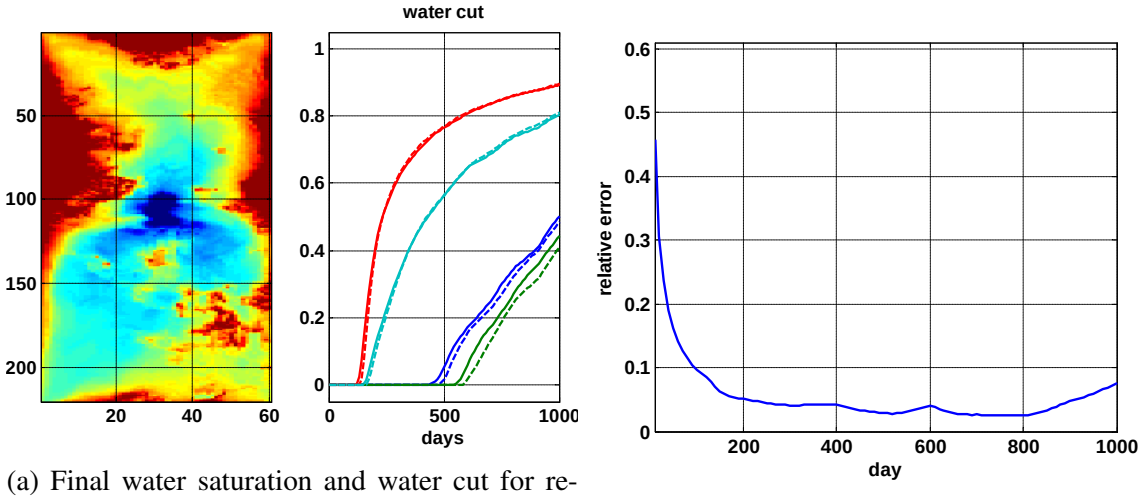
Figure 3.10: Results for training schedule.



(a) Relative saturation error.

(b) Relative pressure error.

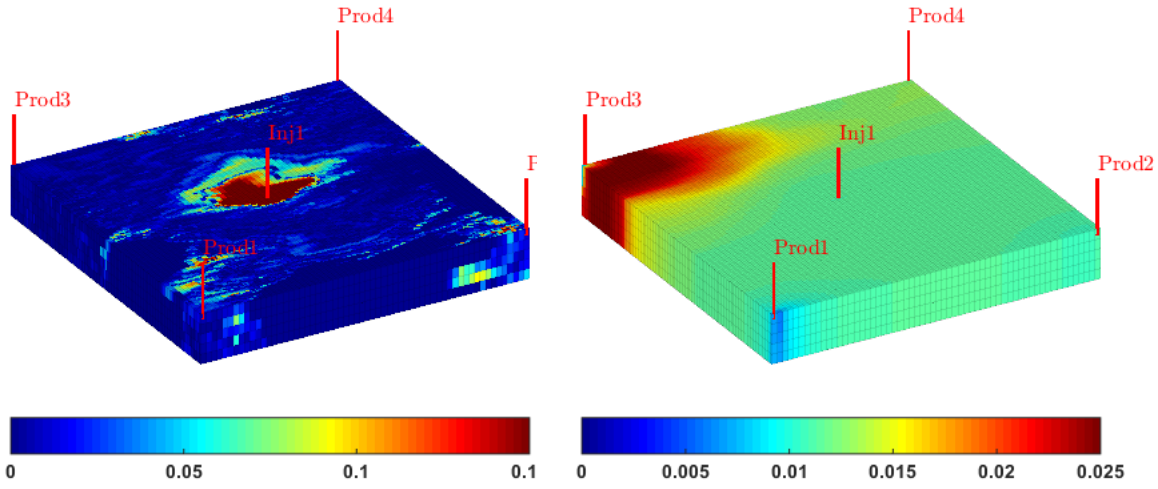
Figure 3.11: Final relative error with the training schedule.



(a) Final water saturation and water cut for reduced (solid) and high fidelity (dashed) model.

(b) Relative saturation error advancing with respect to time.

Figure 3.12: Results for test schedule.



(a) Relative saturation error.

(b) Relative pressure error.

Figure 3.13: Final relative error with the test schedule.

In Algorithm 1 (original DEIM algorithm), the process starts from selecting the first interpolation index $j_1 \in \{1, \cdots, n\}$ corresponding to the entry of the first POD basis matrix with largest magnitude. The remaining interpolation indices, j_i for $i = 2, \cdots, m$ are selected so that each of them corresponds to the entry with the largest magnitude of the residual $\mathbf{r} = \mathbf{U}(:, i) - \mathbf{Vc}$ from line 7 of Algorithm 1. The term $\mathbf{r}$ is the error between the input $\mathbf{U}(:, i)$ and its approximation $\mathbf{Vc}$ from interpolating the basis $\{\mathbf{U}(:, 1), \cdots, \mathbf{U}(:, i-1)\}$ at the indices $\{j_1, \cdots, j_{i-1}\}$. The difference between DEIM in Algorithm 1 and the modified DEIM in Algorithm 2 is that, in modified DEIM, we choose $N_{gp} > 1$ number of interpolation points instead of 1 for each empirical mode in each step (without repeating the previously selected points), and solve a least square problem to get the coefficients $\mathbf{c}$.

Algorithm 2 Interpolation points for modified DEIM

Notation: $\mathbf{e}_{s,k}$ is a standard coordinate vector of length s and with 1 at position k. Brackets $[\ \ ]$ denote concatenation of two or more matrices. $\mathbf{U}(:, i)$ denotes the i_{th} column of matrix $\mathbf{U}$.
Input : Global POD basis matirx $\mathbf{U}$, number of points N_{gp} for each column
$(n, m) \leftarrow$ size($\mathbf{U}$)
$J \leftarrow$ the first N_{gp} positions of $\mathbf{U}(:, 1)$ with largest absolute entries
$\mathbf{U} = [\mathbf{U}(:, 1)], \mathbf{P} = [e_{n,J_1}, \cdots, e_{n,j_{s_1}}]$
for $i = 2 : m$ **do**
  $\mathbf{c} \leftarrow \text{argmin}_{\mathbf{q}} \|\mathbf{P}^T\mathbf{U}(:, i) - \mathbf{P}^T\mathbf{Vq}\|$
  $\mathbf{r} \leftarrow \mathbf{U}(:, i) - \mathbf{Vc}$
  $J_i \leftarrow$ the first N_{gp} positions of $\mathbf{r}$ with largest absolute entries
  $J \leftarrow J \bigcup (J_i/J) = \{j_1, \cdots, j_{s_i}\}$
  $\mathbf{V} \leftarrow [\mathbf{V}, \mathbf{U}(:, i)], \mathbf{P} \leftarrow [\mathbf{P}, \mathbf{e}_{n,J_1}, \cdots, \mathbf{e}_{n,j_{s_i}}]$
end for
Output : $\mathbf{P}$

The number of POD basis for velocity, pressure, saturation and number of DEIM basis for the nonlinear functions are given in Table 3.5, where N_{gp} is as defined in Algorithm 2. Each row represents the number of DEIM points for one of the nonlinear functions.

Table 3.5: Number of DEIM points corresponding to different nonlinear functionals defined in 3.8 using Algorithm 2. d refers to the derivative. Number of basis for states: $N_v = 20, N_p = 5, N_s = 25$

	$N_{gp} = 2$	$N_{gp} = 4$	$N_{gp} = 8$	$N_{gp} = 10$	$N_{gp} = 15$	$N_{gp} = 20$
$N_{f_w} = 50$	90	127	187	220	284	371
$N_{df_w} = 65$	125	199	306	346	468	559
$N_{f_2} = 50$	84	126	189	205	264	327
$N_{df_2w} = 65$	125	193	312	363	472	592
$N_{df_2o} = 55$	101	138	195	222	287	355

For example, N_{f_w} refers to the number of emprical modes used in approximating f_w. $N_{f_w} = 50$ refers to the number of DEIM points in the original algorithm. The other elements of this row correspond to the number of DEIM points for the choice of N_{gp}. df_w, f_2, df_2w, df_2o refer to the functional defined in the total flux and their derivatives. In Figure 3.14, relative saturation errors with respect to the time with different number of DEIM points are given. For the comparison, the results of the projection errors for the nonlinear functions are also given. We note that for $N_{gp} \geq 8$, we observe a reasonable error. Our numerical results show that for N_{gp} values 1 and 4, the errors can reach 100 %, which we do not present here.

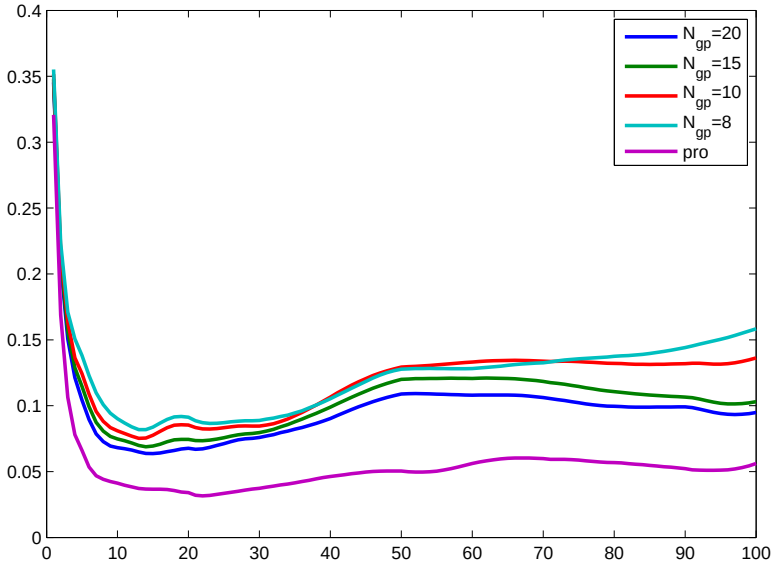


Figure 3.14: Saturation errors advancing with respect to time.

4. ONLINE ADAPTIVE GLOBAL-LOCAL MODEL REDUCTION FOR MULTI-PHASE FLOWS IN HETEROGENEOUS MEDIA *

4.1 Introduction

POD projection type model reduction typically consists of an offline stage followed by an online stage. In the offline stage, the low dimension solution space, also called as the POD subspace, is generated. In the online stage, a reduced system for the problem at hand is constructed by projecting onto the POD subspace. The online solution relies only on the pre-computed quantities from the offline stage. Therefore, a good approximation from the reduced model can be expected only if the offline information is a good representation of the problem of interest.

Adaptivity in the context of model reduction has received much attention. Offline adaptive methods [65] seek to provide a better reduced solution space while the reduced system is constructed in the offline stage. However, once the reduced system is derived, it stays fixed and is kept unchanged in the online stage. Therefore, the online solution relies only on the pre-computed information from the offline stage which is not incorporated at the online phase. Unlike the offline adaptation, online adaptive methods modify the reduced system during the computations in the integration stage. Most of the existing global online adaptive methods [94, 86, 7] rely only on precomputed quantities from the offline stage. These methods work fine when the offline data contains representative information for the solution to the problem of interest. Accurate reduction methods for inputs or controls that result in a solution outside the span of the snapshots generated in the offline stage are desired. We are motivated to consider online adaptive method by incorporating new data that becomes available online, see [48]. Therefore, our method can deal with cases

*Reprinted with permission from "Online Adaptive Local-Global Model Reduction for Flows in Heterogeneous Porous Media" by Y. Efendiev, E. Gildin, and Y. Yang, *Computation*, 4(2), 2016. © 2016 by the authors

that the solution of the problem at hand lies out of the span of the snapshots generated in the offline stage.

In the online stage, we propose a criterion to determine when adaptions are required. Once the criterion is satisfied, we adapt the POD subspaces, the DEIM space and the DEIM interpolation points for the nonlinear functions by incorporating new data that becomes available online. For the adaption of the POD subspace, we propose a global-local strategy. First, instead of global error indicators based on the residual which are expensive to compute, we use local error indicators to monitor when global POD basis adaption is needed. By using local error indicators, the computational time can be reduced as one avoids computing global error indicators. Secondly, we employ a local model order reduction method, *i.e.*, the generalized multiscale finite element method (GMsFEM) [49, 30, 47], to solve the global residual problem to get the new global POD basis function. Based on local error indicators [33, 32, 36, 24], some local multiscale basis functions are updated and used to compute the online global modes inexpensively. For the adaption of DEIM, we employ a modified low rank updates method investigated in [95], which introduces computational costs that scale linearly in the number of unknowns of the full-order system. We remark that the offline global-local approaches have been studied in the literature [5, 43, 40, 97]. In the proposed method, we develop online multiscale basis functions for both local and global updates.

The rest of this section is organized as follows. Section 4.2 devotes to developing the adaptive POD reduction method by adopting data obtained from the online stage. In Section 4.3, we introduce an online low-rank adaptive DEIM method to improve accuracy for the reduction of nonlinear functions. Section 4.4 presents numerical examples to demonstrate the efficiency of our method.

4.2 Online adaptive global-local POD-projection

We adapt the POD subspaces at some particular instants as time proceeds. Two issues are addressed here: at which instants to adapt, and how to adapt the POD subspaces. The updates are performed online, therefore the goal is to keep the computational cost as low as possible.

The main idea of adaptive global-local POD model order reduction method is summarized in Algorithm 3. The superindex G stands for global, superindex L stands for local, and superindex T means transpose of a matrix. Φ_{k-1}^{G} is the initial global POD matrix for time step $k-1$. Similarly, Φ_{k-1}^{L} is the initial local multiscale basis matrix for time step $k-1$. In Figure 4.1, a flowchart corresponding to Algorithm 3 is given. In Figure 4.2, we present an schematic description of the adaptive POD reduction method. We note that there is no global fine problem computation in the method.

Algorithm 3 Adaptive Global-Local POD Model Order Reduction Method

OFFLINE STAGE:

1: Construction of snapshots for states, local off-line space (consists of local multiscale basis) by GMsFEM

2: Construction of POD subspaces (POD projection matrices)

ONLINE STAGE : for step k adaption

3: INPUT : Global POD basis matrix Φ_{k-1}^{G}, local off-line space Φ_{k-1}^{L}

4: Solve the reduced system: $(\Phi_{k-1}^{G})^{T}\mathbf{A}\Phi_{k-1}^{G}\mathbf{X}_{r}^{k-1} = (\Phi_{k-1}^{G})^{T}\mathbf{F}_{k}$ (Global reduced-order model)

5: Compute local error indicators, and decide if adaption is needed. If yes, go to 6. Otherwise, go to next time step $k+1$

6: Solve the global residual problem $\mathbf{A}\boldsymbol{\phi}^{k,\text{online}} = \mathbf{F}_{k} - \mathbf{A}\Phi_{k-1}^{G}\mathbf{X}_{r}^{k-1}$ for $\boldsymbol{\phi}^{k,\text{online}}$ by adaptive local method with initial local off-line space Φ_{k-1}^{L}

7: Update the POD subspace by Adaptive-POD-1 or Adaptive-POD-2

8: OUTPUT : Global POD basis matrix Φ_{k}^{G}, local offline space Φ_{k}^{L}

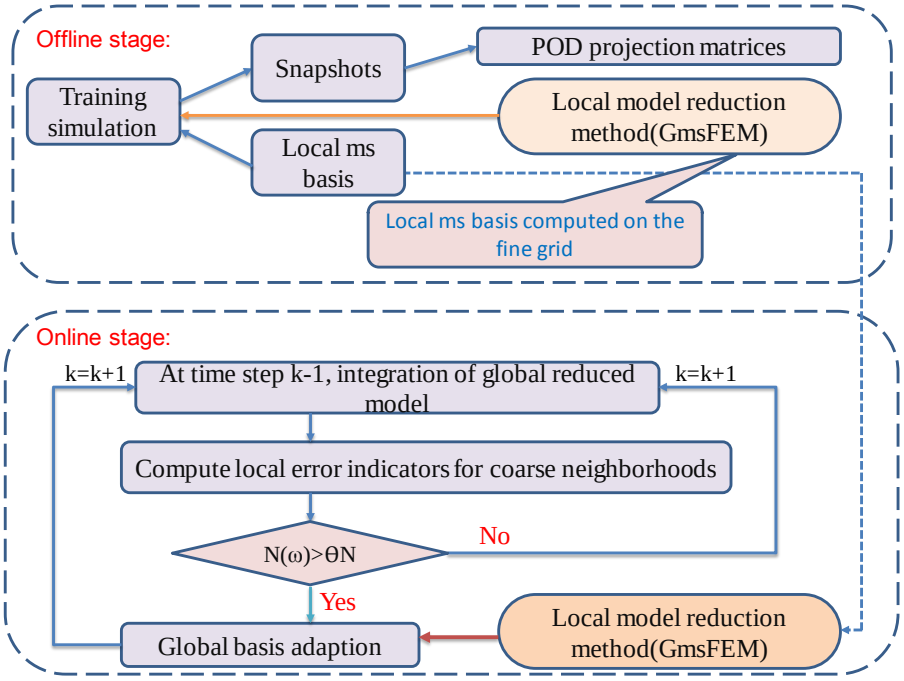


Figure 4.1: Flowchart of the online adaptive global-local POD method.

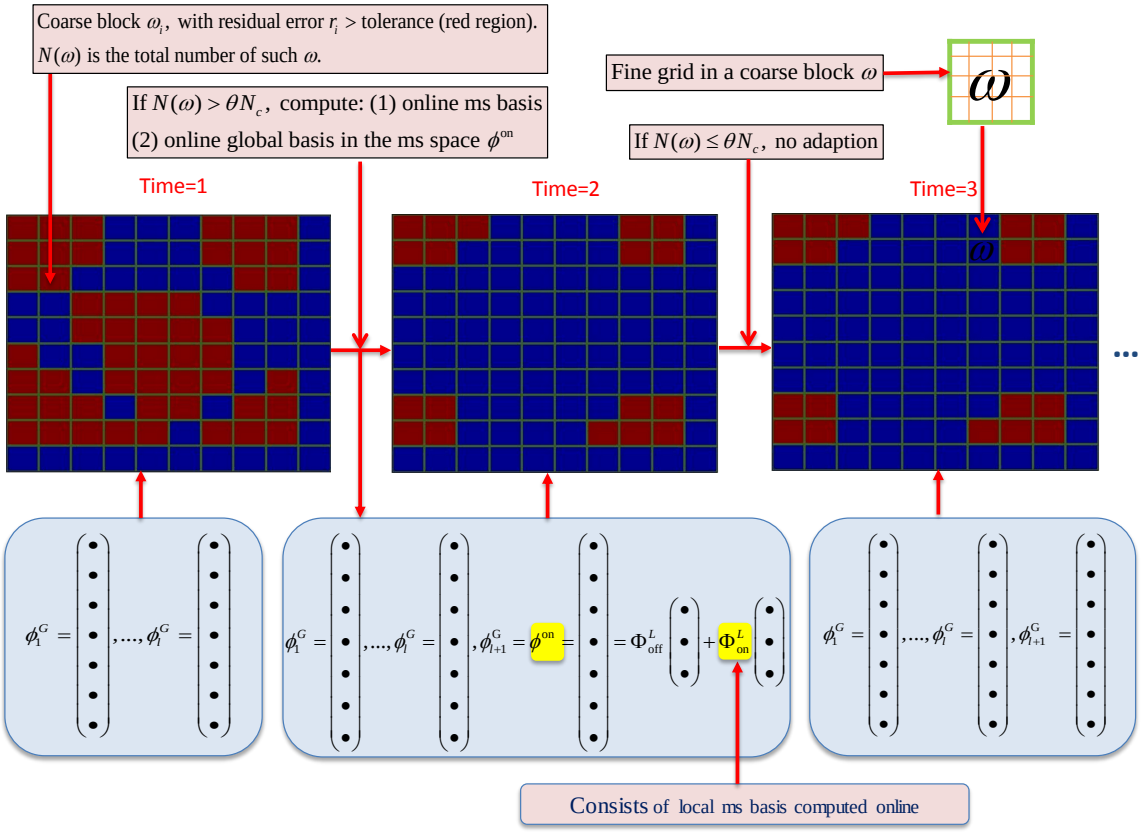


Figure 4.2: Schematic description of the online adaptive global-local POD method.

In the offline stage, a training simulation is run by the GMsFEM. Local multiscale bases are constructed, and snapshots for states at some time steps are saved. Next, POD is performed on the snapshots to construct the POD projection matrices. In the online stage, at every time step we solve a global reduced system, which is usually very small, for example, in Section 4.4.1, the largest dimension is 8. Next we use our criterion to decide if global basis adaption is necessary. The criterion is specified in the next paragraph. If adaption is needed, a new global basis is obtained by solving a residual problem with GMsFEM. Again, in Section 4.4.1, a residual problem is of size about 10^4; in Table 4.1, the size of the reduced problem of the residual problem by GMsFEM is about 600.

It is important to monitor when the approximation from the reduced system is no longer sufficiently accurate according to a particular criterion, leading to adaptation of the POD subspaces. This adaption must be performed when the system dynamics has deviated from the current POD subspace as reflected by the residual evaluation. In [17], some residual based POD modes were added to the old POD subspace for Navier-Stokes equations. In [99], a second error estimate based on a second Galerkin system to account for situations in which qualitative changes in the dynamics occur was introduced. In this case, two reduced systems must be solved at a time, thus incurring higher computational cost. In [98], a residual based indicator was used in nonlinear dissipative systems. Here, we propose to use local error indicators based on the multiscale basis, which is cheaper to be computed. The main idea is as follows:

- For coarse blocks $\omega_1, \omega_2, \cdots, \omega_N$, compute their corresponding H^{-1} norm of the residual, and denote as $r_1, r_2, \cdots, r_{N_c}$.
- Count the total number $N(\omega)$ of coarse blocks with $r_i >$ a certain error tolerance. Here a large error means the current POD modes cannot give a good representation of the solution features in that coarse block.
- If $N(\omega) > \theta N_c$, then adaption is needed, θ is a fraction of the total number of coarse blocks.

In general, one can use a threshold for the sum of the local indicators. The choice of the threshold is typically based on experimental results and analysis. The larger the threshold is, more updates are performed. Next, we describe how to update the POD subspace.

For the POD subspace adaption at step k, suppose N_{k-1} number of initial POD modes $\{\boldsymbol{\phi}_1^G, \cdots, \boldsymbol{\phi}_{N_{k-1}}^G\}$ are given. Φ^G is the POD matrix with $\{\boldsymbol{\phi}_1^G, \cdots, \boldsymbol{\phi}_{N_{k-1}}^G\}$ as column vectors. We obtain the online snapshot vector $\boldsymbol{\phi}^{k,\text{online}}$ by solving a residual problem (for

linear system), or the high fidelity system (for nonlinear system). We illustrate how to get the new online global basis for the linear case. Suppose we are solving Equation (2.14). At every time step, we need to solve a linear system of the form

$$\mathbf{A}\mathbf{X} = \mathbf{F}_k. \tag{4.1}$$

The POD reduced order system for the equation (4.1) is

$$(\Phi^G)^T \mathbf{A} \Phi^G \mathbf{X}_r = (\Phi^G)^T \mathbf{F}_k. \tag{4.2}$$

The residual is $\mathbf{res} = \mathbf{F}_k - \mathbf{A}\Phi^G \mathbf{X}_r$. If update is needed, then we solve the residual problem

$$\mathbf{A}\boldsymbol{\phi}^{k,\text{online}} = \mathbf{res} \tag{4.3}$$

to obtain the online basis $\boldsymbol{\phi}^{k,\text{online}}$.

Note that equation (4.3) is expensive to solve because it is a fine-grid problem. We propose to use the local model reduction method as described in Section 2.4 to solve it. In the following, we introduce two methods to adapt the POD subspace after $\boldsymbol{\phi}^{k,\text{online}}$ is obtained. One way is to add $\boldsymbol{\phi}^{k,\text{online}}$ directly to $\{\boldsymbol{\phi}_1^G, \cdots, \boldsymbol{\phi}_{N_{k-1}}^G\}$, therefore the new POD subspace is the span of $\{\boldsymbol{\phi}_1^G, \cdots, \boldsymbol{\phi}_{N_{k-1}}^G, \boldsymbol{\phi}^{k,\text{online}}\}$. In this way, the number of POD modes increases by one after every adaption. We denote this method as Adaptive-POD-1.

It is possible that as time proceeds, some modes in the POD subspace become negligible (redundant) in representing the near future dynamics, therefore they can be removed from the POD subspace. This removal can be achieved in various ways. In our simulations, we follow the following procedure [98]. Apply POD to the set of vectors

$$\{w_1\boldsymbol{\phi}_1^G, \cdots, w_{N_{k-1}}\boldsymbol{\phi}_{N_{k-1}}^G, \boldsymbol{\phi}^{k,\text{online}}\}$$

, where the weights w_i are defined as

$$w_i = \min\left\{ \frac{\sigma_i}{\sqrt{\sum_{j=1}^{N_{k-1}} \sigma_j^2}}, \frac{\langle |C_i| \rangle}{\sqrt{\sum_{j=1}^{N_{k-1}} \langle |C_j| \rangle^2}} \right\}$$

where σ_i are singular values associated with $\boldsymbol{\phi}_i$, C_i is the amplitude of $\boldsymbol{\phi}_i^G$, *i.e.*, the coefficient of mode $\boldsymbol{\phi}_i^G$, and $\langle |C_i| \rangle$ is the temporal mean value of $|C_i|$ obtained from the reduced system starting from the first instant since the last update until $k-1$. In this way, the weights w_i eliminate those modes whose average energy decreased considerably since the last update. Therefore only a small number of POD modes is retained through the whole simulation. We denote this method as Adaptive-POD-2. We compare the performance of these two methods in the numerical experiment for two-phase flow in Section 4.4. The next section we discuss the adaption for the reduction of nonlinear terms.

4.3 Online adaptive DEIM

In this section, we summarize the idea of online adaptive DEIM as in [95]. First, we adapt the DEIM space by rank one updates. Secondly, we adapt the DEIM interpolation points. The main steps for this DEIM adaption are given in Algorithm 4. It is shown in [95] that the run time cost of the adaption scales linearly with the number of degrees of freedom of the full-order system.

For the basis adaption for the nonlinear function $\mathbf{f}(\mathbf{y}(t))$ at step k, suppose an initial DEIM interpolant $(\mathbf{U}_{k-1}, \mathbf{P}_{k-1})$ (with m DEIM points) is given. We extend the interpolation points matrix $\mathbf{P}_{k-1}$ as

$$\mathbf{S}_k = [\mathbf{e}_{p_1}, \cdots, \mathbf{e}_{p_m}, \mathbf{e}_{p_{m+1}}, \cdots, \mathbf{e}_{p_{m+m_s}}].$$

The extra m_s points are chosen from $\{1, \cdots, n\} \backslash \{p_1, \cdots, p_m\}$ randomly. Therefore, we get a new DEIM interpolant $(\mathbf{U}_{k-1}, \mathbf{S}_k)$. We note that $\mathbf{U}_{k-1}$ is still the same. Next

Algorithm 4 Online Adaptive DEIM [95]

INPUT : $(\mathbf{U}_{k-1}, \mathbf{P}_{k-1})$, number of sampling points m_s, window size w

Select m_s points from $\{1, 2, \cdots, n\}$ that do not repeat the interpolation points at step $K-1$.

Assemble sampling points matrix $\mathbf{S}_k$ by combining $\mathbf{P}_{k-1}$ and the selected points.

Compute the coefficient matrix $\mathbf{C}_k$ for $[\hat{\mathbf{y}}(t_k), \hat{\mathbf{y}}(t_{k-1}), \cdots, \hat{\mathbf{y}}(t_{k-w+1})]$ with interpolant $(\mathbf{U}_{k-1}, \mathbf{S}_k)$

Solve the minimization problem (4.5) for $\boldsymbol{\alpha}_k, \boldsymbol{\beta}_k$

Update POD subspace $\mathbf{U}_k = \mathbf{U}_{k-1} + \boldsymbol{\alpha}_k \boldsymbol{\beta}_k^T$

Compute $\text{diag}(\mathbf{U}_k^T \mathbf{U}_{k-1})$

Find the index i of the pair of basis vectors which are nearest to orthogonal

Let $\mathbf{u}_k$ be the $i-$ th column of the adapted basis $\mathbf{U}_k$

Store all other $m-1$ columns of $\mathbf{U}_k$ in $\hat{\mathbf{U}}_k \in \mathbb{R}^{n \times (m-1)}$

Store all other $m-1$ columns of $\mathbf{P}_{k-1}$ in $\hat{\mathbf{P}}_k \in \mathbb{R}^{n \times (m-1)}$

Approximate $\mathbf{u}_k$ with the DEIM interpolant $(\hat{\mathbf{U}}_k, \hat{\mathbf{P}}_k)$ as $\hat{\mathbf{u}}_k = \hat{\mathbf{U}}_k (\hat{\mathbf{P}}_k^T \hat{\mathbf{U}}_k)^{-1} \hat{\mathbf{P}}_k^T \mathbf{u}_k$

Find the index p_k^i that corresponds to largest residual of $|\hat{\mathbf{u}}_k - \mathbf{u}_k|$

Update interpolation matrix $\mathbf{P}_k$: if $\mathbf{e}_{p_k^i}$ is not a column of $\mathbf{P}_{k-1}$, then for the $i-$ th column of $\mathbf{P}_{k-1}$, replace interpolation point $\mathbf{e}_{p_{k-1}^i}$ with $\mathbf{e}_{p_k^i}$

OUTPUT : $(\mathbf{U}_k, \mathbf{P}_k)$

we define a window of size w that contains the vectors of $\hat{\mathbf{y}}(t_k), \hat{\mathbf{y}}(t_{k-1}), \cdots, \hat{\mathbf{y}}(t_{k-w+1})$ obtained in the previous online computations; an alternative is to also include intermediate states from the Newton-Raphson iterations of the previous online computations. We then construct DEIM approximations of the nonlinear function $\mathbf{f}$ at the vectors

$$\hat{\mathbf{y}}(t_k), \hat{\mathbf{y}}(t_{k-1}), \cdots, \hat{\mathbf{y}}(t_{k-w+1})$$

with the DEIM interpolant $(\mathbf{U}_{k-1}, \mathbf{S}_k)$. For each $\hat{\mathbf{y}}(t_{k_i})$, the coefficient is derived as

$$\mathbf{c}_k(\mathbf{y}(t_{k_i})) = (\mathbf{S}_k^T \mathbf{U}_{k-1})^+ \mathbf{S}_k^T \mathbf{f}(\hat{\mathbf{y}}(t_{k_i}))$$

which are put as columns in the coefficient matrix $\mathbf{C}_k \in \mathbb{R}^{m \times w}$. Here $(\mathbf{S}_k^T \mathbf{U}_{k-1})^+ \in \mathbb{R}^{m \times (m+m_s)}$ is the Moore-Penrose pseudo-inverse.

We then derive two vectors $\boldsymbol{\alpha}_k \in \mathbb{R}^n$ and $\boldsymbol{\beta}_k \in \mathbb{R}^m$ such that the adapted DEIM basis $\mathbf{U}_k = \mathbf{U}_{k-1} + \boldsymbol{\alpha}_k \boldsymbol{\beta}_k^T$ minimizes the Frobenius norm of the residual at the sampling points given by $\mathbf{S}_k$

$$\|\mathbf{S}_k^T (\mathbf{U}_k \mathbf{C}_k - \mathbf{F}_k)\|_F^2 \tag{4.4}$$

where $\mathbf{F}_k = [\mathbf{f}(\hat{\mathbf{y}}(t_k)), \cdots, \mathbf{f}(\hat{\mathbf{y}}(t_{k-w+1}))]$.

Define the residual matrix $\mathbf{R}_k = \mathbf{U}_{k-1} \mathbf{C}_k - \mathbf{F}_k$ and transform (4.4) into

$$\|\mathbf{S}_k^T \mathbf{R}_k + \mathbf{S}_k^T \boldsymbol{\alpha}_k \boldsymbol{\beta}_k^T \mathbf{C}_k\|_F^2.$$

Thus the vectors $\boldsymbol{\alpha}_k$ and $\boldsymbol{\beta}_k$ of the update $\boldsymbol{\alpha}_k \boldsymbol{\beta}_k^T \in \mathbb{R}^{n \times m}$ are a solution of the minimization problem

$$\arg \min_{\boldsymbol{\alpha}_k \in \mathbb{R}^n, \boldsymbol{\beta}_k \in \mathbb{R}^m} \|\mathbf{S}_k^T \mathbf{R}_k + \mathbf{S}_k^T \boldsymbol{\alpha}_k \boldsymbol{\beta}_k^T \mathbf{C}_k\|_F^2. \tag{4.5}$$

The vectors $\boldsymbol{\alpha}_k, \boldsymbol{\beta}_k$ are obtained by solving a generalized eigenvalue problem as stated

in Theorem 4.3.1, we refer to [95] for details.

Theorem 4.3.1. *If* $\|\mathbf{S}_k^T\mathbf{R}_k\mathbf{C}_k^T\|_F > 0$*, and after the transformation of Lemma 4.3.2, an optimal update* $\boldsymbol{\alpha}_k\boldsymbol{\beta}_k^T$ *with respect to (4.5) is given by setting* $\boldsymbol{\alpha}_k = \mathbf{S}_k\boldsymbol{\alpha}_k^{'}$ *and* $\boldsymbol{\beta}_k = \mathbf{Q}_k\boldsymbol{\beta}_k^{'}$*, where* $\boldsymbol{\alpha}_k^{'}$ *and* $\boldsymbol{\beta}_k^{'}$ *are defined as in Lemma 4.3.3, and* $\mathbf{Q}_k \in \mathbb{R}^{m\times r}$ *is given as in Lemma 4.3.2. If* $\|\mathbf{S}_k^T\mathbf{R}_k\mathbf{C}_k^T\|_F = 0$*, an optimal update with respect to (4.5) is* $\boldsymbol{\alpha}_k = 0$ *and* $\boldsymbol{\beta}_k = 0$*.*

Lemma 4.3.2. *Let* $\mathbf{C}_k \in \mathbb{R}^{m\times w}$ *has rank* $r < m$*, i.e.,* $\mathbf{C}_k$ *does not have full row rank. There exists a matrix* $\mathbf{Z}_k \in \mathbb{R}^{r\times w}$*, with rank* r*, and a matrix* $\mathbf{Q}_k \in \mathbb{R}^{m\times r}$ *with orthonormal columns such that*

$$\|\mathbf{S}_k^T\mathbf{R}_k + \mathbf{a}\mathbf{b}^T\mathbf{C}_k\|_F^2 = \|\mathbf{S}_k^T\mathbf{R}_k + \mathbf{a}\mathbf{z}^T\mathbf{Z}_k\|_F^2,$$

for all $\mathbf{a} \in \mathbb{R}^{m+m_s}$ *and* $\mathbf{b} \in \mathbb{R}^m$*, where* $\mathbf{z}^T = \mathbf{b}^T\mathbf{Q}_k \in \mathbb{R}^r$*.*

Lemma 4.3.3. *Let* $\mathbf{C}_k \in \mathbb{R}^{m\times w}$ *has rank* m*, i.e., full row rank, and assume* $\|\mathbf{S}_k^T\mathbf{R}_k\mathbf{C}_k^T\|_F > 0$*. Let* $\boldsymbol{\beta}_k^{'} \in \mathbb{R}^m$ *be an eigenvector corresponding to the largest eigenvalue* $\lambda \in \mathbb{R}$ *of the generalized eigenvalue problem*

$$\mathbf{C}_k(\mathbf{S}_k^T\mathbf{R}_k)^T(\mathbf{S}_k^T\mathbf{R}_k)\mathbf{C}_k^T\boldsymbol{\beta}_k^{'} = \lambda\mathbf{C}_k\mathbf{C}_k^T\boldsymbol{\beta}_k^{'},$$

and set $\boldsymbol{\alpha}_k^{'} = (-1/\|\mathbf{C}_k^T\boldsymbol{\beta}_k^{'}\|_2^2)\mathbf{S}_k^T\mathbf{R}_k\mathbf{C}_k^T\boldsymbol{\beta}_k^{'}$*. The vectors* $\boldsymbol{\alpha}_k^{'}$ *and* $\boldsymbol{\beta}_k^{'}$ *are a solution of the minimization problem*

$$\arg\min_{\mathbf{a}\in\mathbb{R}^{m+m_s},\mathbf{b}\in\mathbb{R}^m} \|\mathbf{S}_k^T\mathbf{R}_k + \mathbf{a}\mathbf{b}^T\mathbf{C}_k\|_F^2. \tag{4.6}$$

We note that the sampling points are only used for the DEIM basis adaptation. Ultimately the DEIM interpolation points are used for the DEIM approximation. With the adapted DEIM basis $\mathbf{U}_k$ at step k, we also adapt the DEIM interpolation points. The s-

tandard DEIM greedy algorithm is computationally expensive to implement in the online stage, since it recomputes all m interpolation points. It is unnecessary to change all interpolation points after a rank-one update to the DEIM basis. Therefore, we adapt the DEIM interpolation points as follows. First, we compute the dot product between the old and the adapted DEIM basis

$$\text{diag}(\mathbf{U}_k^T \mathbf{U}_{k-1}). \tag{4.7}$$

If the dot product of two normalized vectors is one, then they are colinear and the adapted basis vector has not been rotated with respect to the previous vector at step $k-1$. If it is zero, they are orthogonal. We select the basis vector $\mathbf{u}_k$ of $\mathbf{U}_k$ that corresponds to the component of (4.7) with the lowest absolute value. The new interpolation point p_k^i is derived from $\mathbf{u}_k$ following the standard DEIM algorithm. It then replaces the interpolation point p_{k-1}^i. Other interpolation points are unchanged.

4.4 Numerical examples

4.4.1 Single-phase flow

In this section, we consider the following single-phase flow problem:

$$\frac{\partial p}{\partial t} + \nabla \cdot (\kappa \nabla p) = q \quad \text{in} \quad \Omega, \tag{4.8}$$

$$p = 0 \quad \text{on} \quad \partial\Omega \quad \text{(boundary condition)}, \tag{4.9}$$

$$p(t=0) = p_0 \quad \text{in} \quad \Omega \quad \text{(initial condition)}. \tag{4.10}$$

Discretizing in a finite element space, for example Q_1 which consists of piecewise linear functions on Ω, for space and using backward Euler method for time discretization,

we obtain the following algebraic system:

$$(\mathbf{M} + \Delta t \mathbf{S})\tilde{\mathbf{p}}^n = \mathbf{M}\tilde{\mathbf{p}}^{n-1} + \Delta t \mathbf{Q}^n \quad \text{in} \quad \Omega, \tag{4.11}$$

$$\tag{4.12}$$

where $\mathbf{M}_{ij} = \int_\Omega \psi_i \psi_j$ is the mass matrix, $\mathbf{S}_{ij} = \int_\Omega \kappa \nabla \psi_i \cdot \nabla \psi_j$ is the stiffness matrix, $\psi_i, \psi_j \in \mathbf{Q}_1$, Δt is the time step size.

The computational domain is $\Omega = (0,1)^2$, a uniform fine mesh with $n = 100$ is used. Our proposed approach can also be used for solving problems on unstructured grids. The size of the full-order system at every time step is $10,000$. For the coarse mesh, we use the mesh size $1/10$. The final simulation time is 10, time-stepping size is $\Delta t = 0.1$, therefore there are 100 snapshots in total. The logarithm of the permeability field is shown in Figure 3.5. In the offline stage, the right-side hand function q for the training simulation is only nonzero on the five blue regions in Figure 4.3. The values of q with respect to time is given in the left figure of Figure 4.4. In the test simulation, q is given in the right figure of Figure 4.4, two nonzero regions are added, one is nonzero through 60–80, the other is nonzero through 90–100, which are the red ones in Figure 4.3. There is a large decrease in the value $q1$ at time instant 35. In Table 4.2, the average L^2, H^1 errors with different number of POD basis are presented. We can see that even if we use all the 100 POD basis, the average L^2 error is still about 18%. In Table 4.1, we start with 5 initial global POD basis. The column G^{off} means the number of initial global POD basis for each time instants, while the column L^{off} means the number of initial local basis functions, and the column L^{add} means the number of local basis added to solve the residual problem (4.3). For instance, for the third row, the first column shows that we need to update at time instant 2; the second column means for time instant 2, the number of initial global POD basis is 6; the third column means for time instant 2, the number of initial local basis

functions is 484; the forth column means we add 126 local basis to the multiscale solution space, from which the global basis is obtained. For the whole test simulation, we add basis at time instants 1, 2 and 35, then we get about 1% average L^2 error. The relative L^2, H^1 errors with respect to time are shown in Figure 4.5, which show a good agreement of our numerical solution with the reference solution. The reference solution is computed on the fine grid. Here, since the number of POD basis is already very small, we only use the Adaptive-POD-1 method.

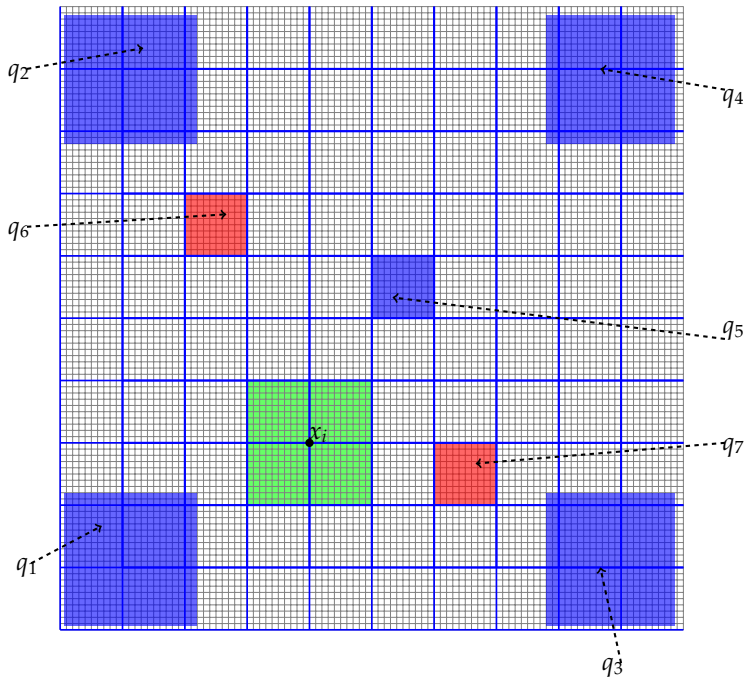


Figure 4.3: Grid information and q.

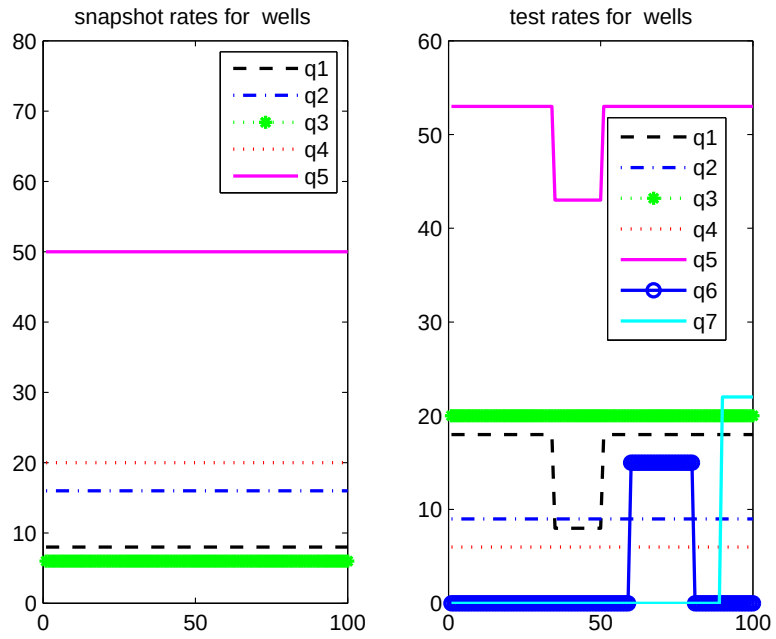


Figure 4.4: Injection rate schedules for training and test models.

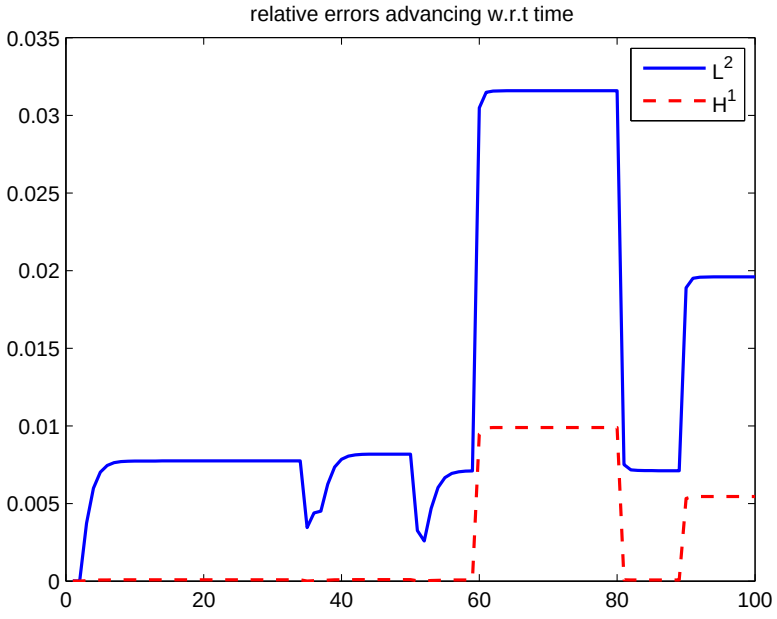


Figure 4.5: Relative errors advancing in time for the adaptive POD.

Table 4.1: Adaptive-POD-1 with 5 initial global POD basis.

Time instant to add	G^{off}	L^{off}	L^{add}
1	5	484	119
2	6	484	116
35	7	484	126
Average error	L^2	H^1	
	0.0135	0.0027	

Table 4.2: Average POD errors.

Number of POD basis	L^2	H^1
2	0.6114	0.3254
5	0.3470	0.1676
8	0.2956	0.1378
100	0.1830	0.0796

4.4.2 An incompressible two-phase flow model

In this section, we consider two-phase flow in a reservoir domain (denoted by Ω) under the assumption that the displacement is dominated by viscous effects; i.e., we neglect the effects of gravity, compressibility, and capillary pressure.

We apply the online adaptive model reduction method to an oil-water flow model with the structure of a 5-spot. A 5-spot pattern is a well configuration that involves 1 producing

well (or 1 injector well) in the center and 4 injecting wells (or 4 producers) in the corners of a Cartesian grid. It can be seen as a five wells (point source) in the whole domain. It is designed to maximize the reservoir sweep as water is being injected into the reservoir. Here, we set four injection wells in the corners and one production well in the middle of the domain. All the wells are rate controlled. The computational domain is $\Omega = (0, 1)^2$, a uniform fine mesh with $h = 1/220$ is used. The fluid viscosity ratio is $\mu_w/\mu_o = 0.2$. To get the snapshots, we simulate the flow with end of simulation time 800, time stepping size $\Delta t = 4$, with well rates as shown in the left in Figure 4.6. From this forward (training) simulation, we save the snapshots of the states and the nonlinear functions at every 2 units in time. Therefore, 100 snapshots are saved for the states and nonlinear functions. Each snapshot is reshaped to a column vector and is stacked in a snapshot matrix. After applying POD to each matrix, we get the POD subspace. For the test simulation, total simulation time is $400, \Delta t = 4$, and the well rates are given in the right of Figure 4.6.

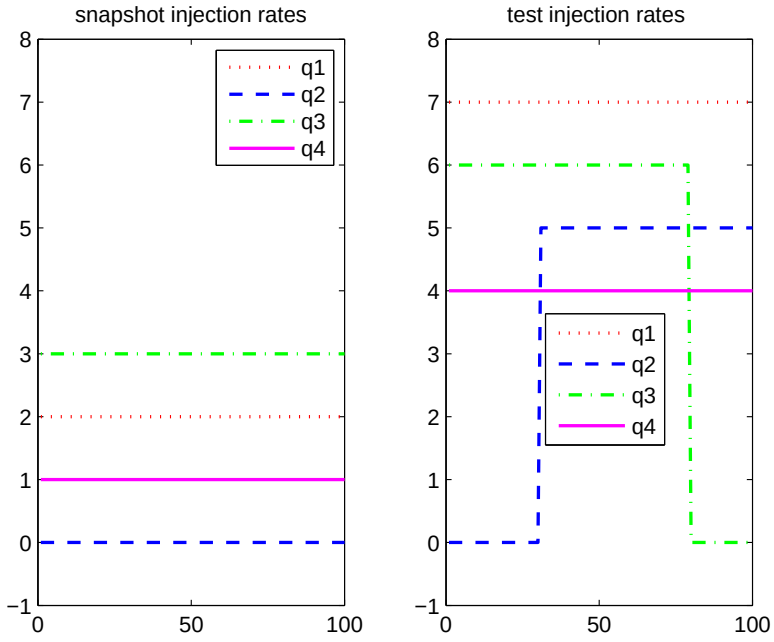


Figure 4.6: Injection rate schedules for training and test models.

From Figure 4.6, we see that the rates are constant for the training simulation, and well 2 is shut off all the time; while for the test simulation, well 2 is open at the time instant 31 and well 3 is closed from 80 to the end of simulation time. Large deviations in the rates of the training and test simulation start at three points: time instant 1, time instant 31, and time instant 80. These deviations are expected to result in requirement of online adaption. We use the L^2 norm of the residual as indicators for adaption. To better understand the impact of injection rates on the whole system, we will investigate the method for the pressure and saturation equations separately first. For the adaptive DEIM application, the number of sampling points is $m_s = 800$, and the window size is $w = 25$.

In Case 1, we apply POD reduction to the pressure equation only, and solve the saturation equation on the fine grid. Note that only POD reduction is used, no DEIM is involved since the pressure equation is linearly dependent on the pressure (we assume incompress-

ible condition). The selection criterion (2.22) for choosing the number of initial POD bases is to capture 99.99% of the energy of the snapshots. In Figure 4.7, relative saturation errors for 3 scenarios with 5 initial POD basis for flux are given. The red line is for static POD without updating, the errors are greater than 10% during the simulation time, and there are jumps at time instants {31,80}. The blue line is for adaptive POD with adaption at time instants {1,31}. With these two adaptions, the error before time instant 80 is under 10%, and increases drastically at time instant 80. The dot purple line is for adaptive POD with adaption at time instants {1,31,80}, which gives a good accuracy. Figure 4.8 presents the comparison of water-cut (water flux fractional function $f_w(s)$) corresponding to these 3 scenarios with the fine water-cut. Both figures show that adaption is necessary in order to obtain a good accuracy.

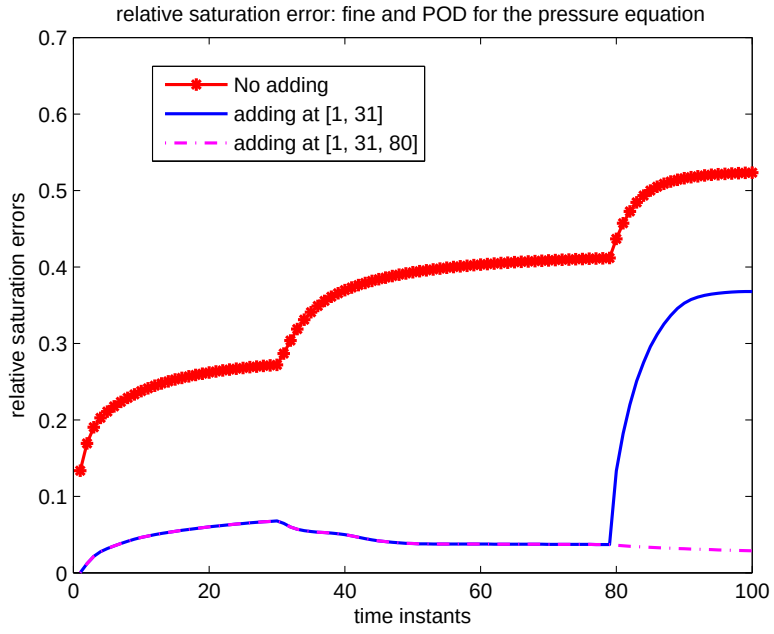


Figure 4.7: Relative saturation errors advancing in time: POD for the pressure equation.

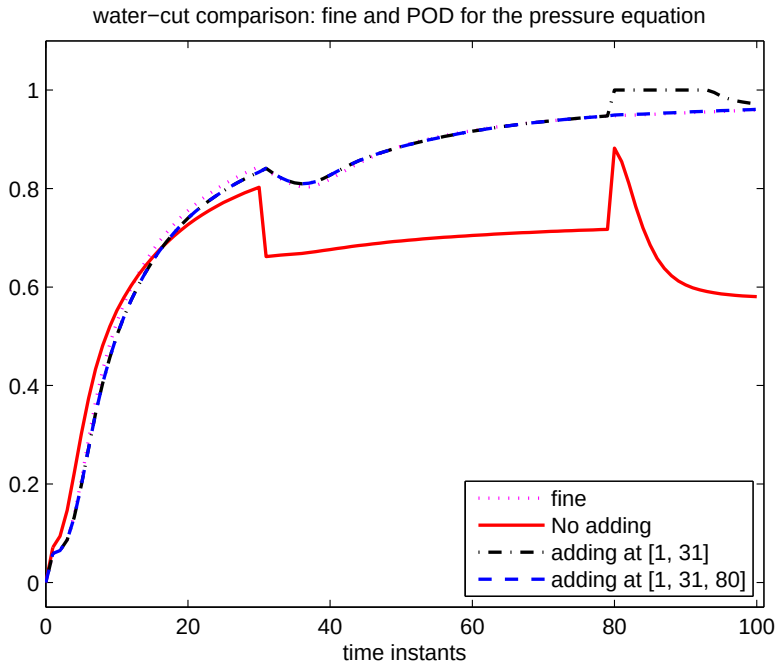


Figure 4.8: Water-cut of online adptive POD for the pressure equation.

In Case 2, we apply model reduction to the saturation equation only while solving the pressure equation on the fine grid. First, we consider POD without DEIM. As pointed out in Section 2.2, due to the nonlinearity of the saturation equation, no improvement in efficiency will be achieved if no DEIM is used. We only consider this case to compare with the case for POD-DEIM in the next example. We will see that the POD-DEIM needs more updates than POD. The selection criterion (2.22) for choosing the number of initial saturation POD basis is to capture 99% of the energy of the snapshots. 20 initial saturation POD basis is selected. Updates are needed at $\{1, 4, 31, 32, 33, 35, 36, 38, 40, 42, 43\}$. In Table 4.3, the numbers of saturation POD basis after updating for Adaptive-POD-1, Adaptive-POD-2 are presented. For example, the number of basis for Adaptive-POD-1 after updating at time instant 33 is 25, the number of basis for Adaptive-POD-2 after 4 is 16. The relative saturation errors for both Adaptive-POD-1 and Adaptive-POD-2 are

given in Figure 4.9, we can see that Adaptive-POD-2 delivers almost the same accuracy while using much less number of basis compared with Adaptive-POD-1. The comparison of water-cut between the solutions of Adaptive-POD-1, Adaptive-POD-2 and the fine solution is shown in Figure 4.10, which shows a good agreement. Second, we apply POD-DEIM to the saturation equation. We use the adaptive DEIM technique as described in Section 4.3 to approximate f_w. 15 initial DEIM basis functions are selected for f_w. The comparison of water-cut between the solution of Adaptive-POD-2-DEIM, Adaptive-POD-2-DEIM and the fine solution is shown in Figure 4.11, they are in good agreement in general. The saturation errors for both Adaptive-POD-1-DEIM and Adaptive-POD-2-DEIM are given in Figure 4.12. We can see that the error of Adaptive-POD-2-DEIM is less than Adaptive-POD-1-DEIM, while the number of basis for Adaptive-POD-2 is less than Adaptive-POD-1.

Table 4.3: Number of POD basis for saturation after each update of POD subspace for Equation (2.6).

Method \ Time instant	1	4	31	32	33	35	36	38	40	42	43
Adaptive-POD-1	21	22	23	24	25	26	27	28	29	30	31
Adaptive-POD-2	22	16	8	8	7	7	7	7	7	7	7

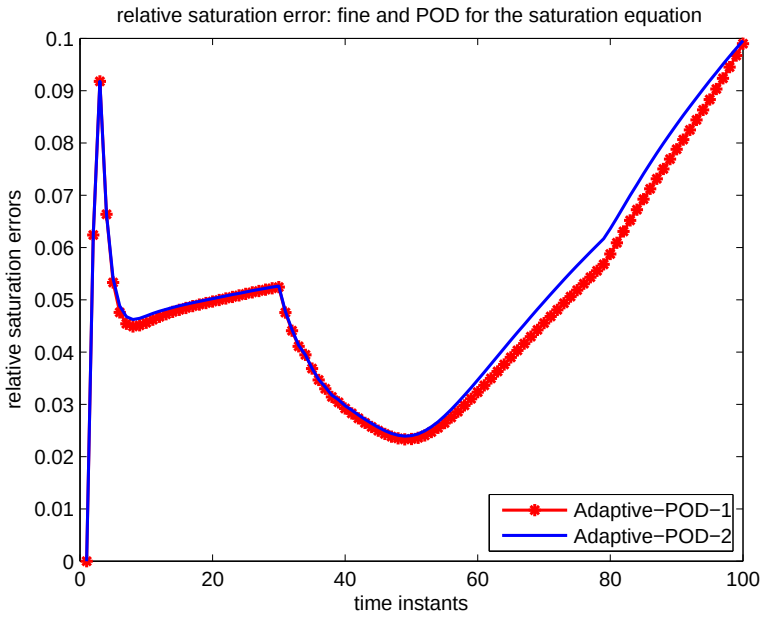


Figure 4.9: Relative saturation errors advancing in time: POD for the saturation equation.

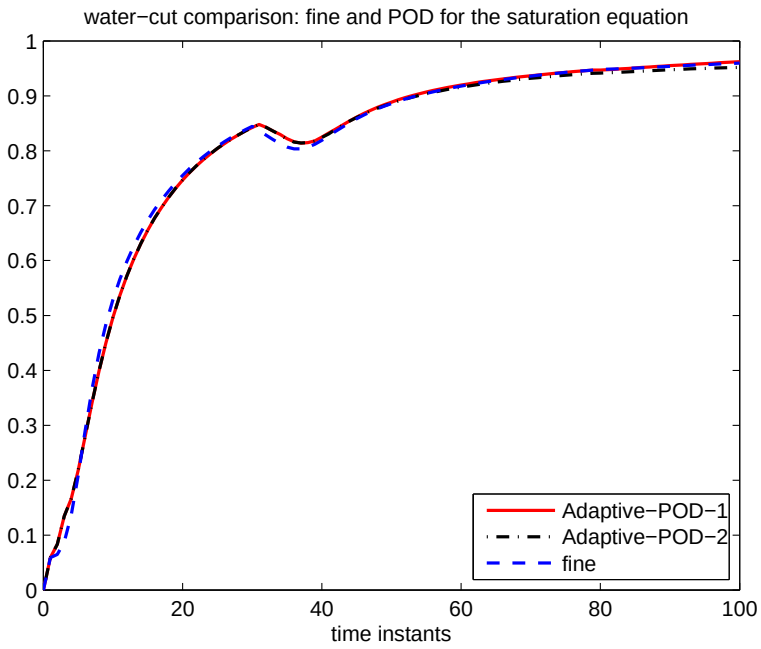


Figure 4.10: Water-cut of online adptive POD for the saturation equation.

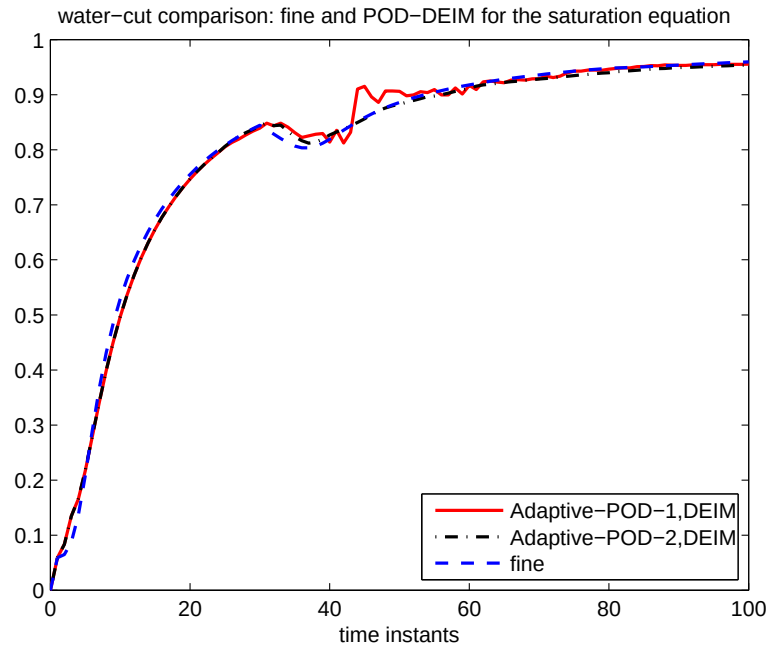


Figure 4.11: Water-cut of online adaptive POD-DEIM for the saturation equation.

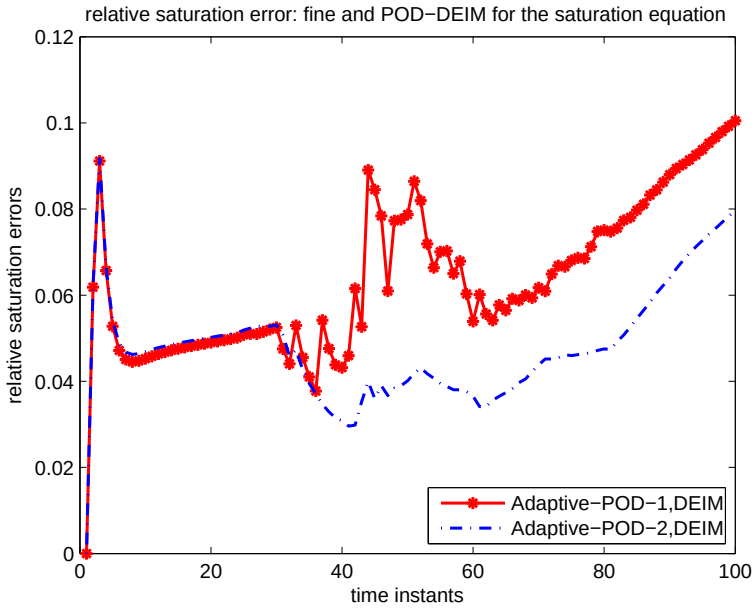


Figure 4.12: Relative saturation errors advancing in time: POD-DEIM for the saturation equation.

In Case 3, apply our model reduction method to the coupled system, *i.e.*, POD for the pressure equation and POD-DEIM for the saturation equation. Velocity needs updates at $\{1, 31, 36, 48, 80\}$, and saturation needs 13 updates at instants as listed in the Table 4.4. The numbers of saturation POD basis after updating for Adaptive-POD-1, Adaptive-POD-2 are shown in Table 4.4. The saturation error is given in Figure 4.13, from there we see that Adaptive-POD-2 has better performance than Adaptive-POD-1. The comparison of water-cut between the solution of Adaptive-POD-1, Adaptive-POD-2 and the fine solution is shown in Figure 4.14. We observe good agreement.

Table 4.4: Number of POD basis for saturation after each update of POD subspace for the coupled system.

Time instant / Method	1	4	31	32	34	35	36	38	39	47	53	54	60
Adaptive-POD-1	21	22	23	24	25	26	27	28	29	30	31	32	33
Adaptive-POD-2	21	16	8	8	7	7	7	7	7	7	7	7	7

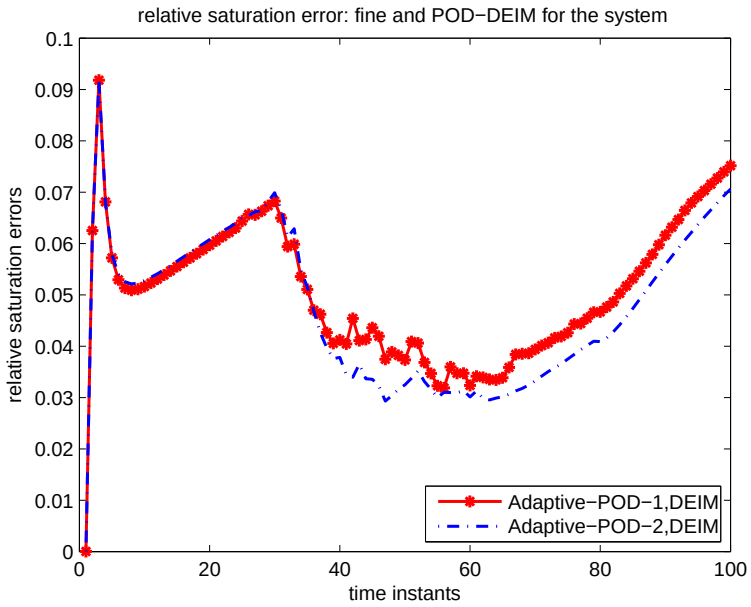


Figure 4.13: Relative saturation errors advancing in time: POD-DEIM for the coupled system.

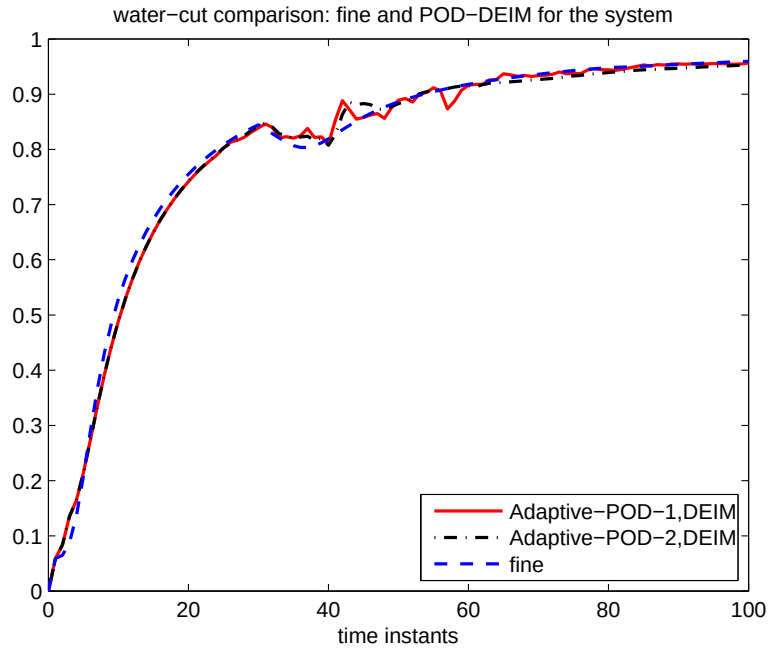


Figure 4.14: Water-cut of online adaptive POD-DEIM for the coupled system.

5. AN ENRICHED MULTISCALE MORTAR SPACE FOR HIGH CONTRAST FLOW PROBLEMS

5.1 Introduction

In previous sections, we have developed global-local approaches using known local GMsFEM techniques, such as mixed GMsFEM. In this section, I present some of my work on the development of a local GMsFEM using mortar finite element framework. This approach develops multiscale basis functions for the interfaces. The mortar GMsFEM approach can be used for multi-phase flow [35].

In the mortar framework, the connectivity of the sub-grid variations is typically enforced by using a Lagrange multiplier. For multiscale problems, the choice of the mortar space for the Lagrange multiplier requires a very careful construction, in order to obtain an efficient and robust method. To construct an accurate mortar space with a small dimension, we will apply the recently developed GMsFEM, which offers a systematic approach for model reduction. In particular, we first create a local snapshot space for every coarse edge. We obtain this space by first solving some local problems on a small region containing an edge, and then restricting the solutions to the edge. Next, we select the dominated modes within the snapshot space using an appropriate POD technique. These dominated modes form the basis for the mortar space. We will apply our mortar space in two related formulations. The first one is a mixed GMsFEM using the mortar formulation. This method gives a coarse-grid solver for the problem (2.24a)-(2.24b). The second one is a coarse space for some preconditioners applied to the fine scale discretization of (2.24a)-(2.24b).

We also study the effects of using oversampling techniques, randomized snapshots and different sizes of local problem domain on the robustness and accuracy of both the multiscale solver and preconditioners. Our work share some similarities with multiscale

hybridizable discontinuous Galerkin method [51, 50]. However, our basis construction is different. The mortar mixed finite element has the feature of global mass conservation, which is important in industrial reservoir simulations.

On the other hand, based on our proposed mortar space, we will construct effective and robust two-level preconditioners to solve the algebraic system arising from fine scale discretization by some iterative methods. Our approach uses the solutions of small local problems and a coarse problem in constructing the preconditioners for the fine-scale system. There are many different types of robust two-level preconditioners [105, 105, 60, 91, 55, 56, 46, 39, 77, 106, 78, 76, 79, 92] developed as mentioned in Section 1.1.2. We use the multiscale mortar space as coarse space for the preconditioners.

This section is arranged as follows. In Section 5.2 we introduce preliminary information, including the mortar variational form of the flow problem, its finite element approximation, and an interface problem. Section 5.3 focuses on the description of the construction of the multi-scale mortar space. In Section 5.4, we discuss the design of two-level preconditioners by using the multi-scale mortar space as coarse space. Numerical results are given in 5.5, which show that the proposed coarse space gives promising ability to deal with problems in media with complicated inclusions and long channels that may across coarse edges.

5.2 Preliminaries

In this section, we give some basic definitions. We also present the formulations of a fine-scale discretization for (2.24a)-(2.24a) (with boundary condition $\mathbf{u}\cdot\mathbf{n} = 0$ on $\partial\Omega$) and its domain decomposition formulation, as well as the formulation for a mixed multiscale method for (2.24a)-(2.24a).

5.2.1 Variational form

With the notion of coarse and fine grids defined in Section 2.3, we introduce the following spaces

$$L_2(K_i) = \left\{ p : \int_{K_i} p^2 < \infty \right\},$$

$$H(\text{div}; K_i) = \left\{ \mathbf{v} \in L_2(K_i)^d : \text{div}(\mathbf{v}) \in L_2(K_i) \right\}.$$

Denote $(\cdot,\cdot)_{K_i}$ for the $L_2(K_i)$ or $L_2(K_i)^d$ inner product, and $\langle \cdot, \cdot \rangle_{\partial K_i}$ for the duality pairing on boundaries and interfaces. For each subdomain i, define

$$\mathbf{V}_i = \{ \mathbf{v} \in H(\text{div}; K_i) : \mathbf{v} \cdot \mathbf{n}|_{\partial\Omega \cap \partial K_i} = 0 \} \quad \text{and} \quad \mathbf{V} = \oplus_{i=1}^N \mathbf{V}_i,$$

$$W_i = L_2(K_i) \quad \text{and} \quad W = \left\{ w \in L_2(\Omega) : \int_\Omega w = 0 \right\},$$

$$M_i = H^{1/2}(E_i), \quad \text{and} \quad M = \oplus_{i=1}^N M_i.$$

The variational form for the system (2.24a)-(2.24a) (with boundary condition $\mathbf{u} \cdot \mathbf{n} = 0$ on $\partial\Omega$) using domain decomposition is formulated as: find $\mathbf{u} \in \mathbf{V}$, $p \in W$ and $\lambda \in M$ such that for each $1 \le i \le N$,

$$(\kappa^{-1}\boldsymbol{u}, \boldsymbol{v})_{K_i} - (p, \nabla \cdot \boldsymbol{v})_{K_i} + \langle \lambda, \boldsymbol{v} \cdot \boldsymbol{n}_i \rangle_{E_i} = 0 \qquad \forall\, \boldsymbol{v} \in \boldsymbol{V}_i, \tag{5.1a}$$

$$(\nabla \cdot \boldsymbol{u}, w)_{K_i} = (f, w)_{K_i} \quad \forall\, w \in W_i, \tag{5.1b}$$

$$\sum_{i=1}^N \langle \boldsymbol{u} \cdot \boldsymbol{n}_i, \mu \rangle_{E_i} = 0 \qquad \forall\, \mu \in M. \tag{5.1c}$$

5.2.2 The finite element approximation

Let $\mathbf{V}_{h,i} \times W_{h,i} \subset \mathbf{V}_i \times W_i$ be any of the mixed finite element spaces satisfying the inf-sup condition for which $\nabla \cdot \mathbf{V}_{h,i} = W_{h,i}$, e.g., the Raviart-Thomas spaces. Define

$\mathbf{V}_h = \oplus_{i=1}^N \mathbf{V}_{h,i}$ and $W_h = \oplus_{i=1}^N W_{h,i}/\mathbb{R}$ for the global discrete flux and pressure. Let $M_{H,i}, M_{h,i} \subset L_2(E_i)$ be the local coarse and fine mortar finite space respectively, and $M_H = \oplus_{1\leq i\leq N} M_{H,i}, M_h = \oplus_{1\leq i\leq N} M_{h,i}$ be the entire coarse and fine mortar finite element spaces.

We formulate the finite element approximation as: find $\mathbf{u}_h \in \mathbf{V_h}$, $p_h \in W_h$ and $\lambda_H \in M_H$ such that for each $1 \leq i \leq N$,

$$(\kappa^{-1}\boldsymbol{u}_h, \boldsymbol{v}_h)_{K_i} - (p_h, \nabla\cdot \boldsymbol{v}_h)_{K_i} + \langle \lambda_H, \boldsymbol{v}_h\cdot\boldsymbol{n}_i\rangle_{E_i} = 0 \qquad \forall\, \boldsymbol{v}_h \in \boldsymbol{V}_{h,i}, \tag{5.2a}$$

$$(\nabla\cdot\boldsymbol{u}_h, w_h)_{K_i} = (f, w_h)_{K_i} \quad \forall\, w_h \in W_{h,i}, \tag{5.2b}$$

$$\sum_{i=1}^{N} \langle \boldsymbol{u}_h\cdot\boldsymbol{n}_i, \mu_H\rangle_{E_i} = 0 \qquad \forall\, \mu_H \in M_H. \tag{5.2c}$$

We note that the coarse mortar space is used in this system. Similar system holds using fine mortar space $\lambda_h \in M_h$. Local conservation is enforced by (5.2b), and (5.2c) enforces weak continuity of flux across the interfaces with respect to the mortar space M_H.

Some examples of the mortar space M_H include the span of piecewise constants or linear (or higher order polynomial) functions or trigonometric functions.

The linear system for (5.2a)-(5.2c) is:

$$\begin{pmatrix} \mathbf{B} & -\mathbf{C} & \mathbf{D} \\ -\mathbf{C}^T & 0 & 0 \\ \mathbf{D}^T & 0 & 0 \end{pmatrix} \begin{pmatrix} \vec{\mathbf{u}} \\ \vec{p} \\ \vec{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ -\mathbf{f} \\ \mathbf{0} \end{pmatrix}, \tag{5.3}$$

where $\mathbf{B}_{i,j} = (\kappa^{-1}\boldsymbol{v}_i, \boldsymbol{v}_j), \mathbf{C}_{i,j} = (w_j, \nabla\cdot\boldsymbol{v}_i)$, and $\mathbf{D}_{i,j} = \langle \mu_j, \boldsymbol{v}_i\cdot\mathbf{n}\rangle$. Here $\vec{\mathbf{u}}, \vec{p}, \vec{\lambda}$ are vectors of coefficients in the expansions of the solutions $\mathbf{u}_h, p_h, \lambda_H$ in their corresponding spaces respectively.

By Schur complement, we end up solving the following system by eliminating $\vec{\mathbf{u}}$ and

$\vec{p}$:

$$\mathbf{S}\vec{\lambda} = \mathbf{b}, \tag{5.4}$$

where

$$\mathbf{S} = \mathbf{D}^T\big(\mathbf{B}^{-1} - \mathbf{B}^{-1}\mathbf{C}(\mathbf{C}^T\mathbf{B}^{-1}\mathbf{C})^{-1}\mathbf{C}^T\mathbf{B}^{-1}\big)\mathbf{D},$$

$$\mathbf{b} = \mathbf{D}^T\mathbf{B}^{-1}\mathbf{C}(\mathbf{C}^T\mathbf{B}^{-1}\mathbf{C})^{-1}\mathbf{f}.$$

The Schur complement system (5.4) can be solved by a Krylov method, such as the Preconditional Conjugate Gradient (PCG) [38, 96], or the Generalized Minimal Residual (GMRES) [42, 104].

5.2.3 Interface problem

The mortar multiscale method and the preconditioners developed in this section are based on a domain decomposition formulation of (2.24a)-(2.24a). We will present the mortar multiscale method in this section, and derive the preconditioner in Section 5.4. The main feature of the mortar mixed finite element method is that it could be implemented by just solving a global system on the coarse mesh together with the solutions of some local problems.

Define bilinear forms $a_{H,i} : M_{H,i} \times M_{H,i} \to \mathbb{R}, i = 1, \cdots, N$ by

$$a_{H,i}(\lambda, \mu) = -\left\langle \mathbf{u}_h^*(\lambda) \cdot \mathbf{n}_i, \mu \right\rangle |_{E_i},$$

and $a_H : M_H \times M_H \to \mathbb{R}$ by

$$a_H = \sum_{i=1}^{N} a_{H,i}(\lambda, \mu),$$

where $\left(\mathbf{u}_h^*(\lambda), p_h^*(\lambda)\right) \in \mathbf{V}_h \times W_h$ solves (λ given, $f = 0$)

$$\left(\kappa^{-1}\boldsymbol{u}_h^*(\lambda), \boldsymbol{v}_h\right)_{K_i} - \left(p_h^*(\lambda), \nabla \cdot \boldsymbol{v}_h\right)_{K_i} = -\left\langle \lambda, \boldsymbol{v}_h \cdot \boldsymbol{n}_i \right\rangle_{E_i} \qquad \forall\, \boldsymbol{v}_h \in \boldsymbol{V}_{h,i}, \tag{5.5a}$$

$$\left(\nabla \cdot \boldsymbol{u}_h^*(\lambda), w_h\right)_{K_i} = 0 \qquad \forall\, w_h \in W_{h,i}, \tag{5.5b}$$

for each $1 \leq i \leq N$.

Define linear functionals $g_{H,i} : M_{H,i} \to \mathbb{R}$ by

$$g_{H,i}(\mu) = \left\langle \bar{\boldsymbol{u}}_h \cdot \mathbf{n}_i, \mu \right\rangle |_{E_i},$$

and $g_H : M_H \to \mathbb{R}$ by

$$g_H(\mu) = \sum_{i=1}^{N} g_{H,i}(\mu),$$

where $(\bar{\boldsymbol{u}}_h, \bar{p}_h) \in \mathbf{V}_h \times W_h$ solves ($\lambda = 0, f$ given) for $1 \leq i \leq N$

$$\left(\kappa^{-1}\bar{\boldsymbol{u}}_h, \boldsymbol{v}_h\right)_{K_i} - \left(\bar{p}_h, \nabla \cdot \boldsymbol{v}_h\right)_{K_i} = 0 \qquad \forall\, \boldsymbol{v}_h \in \boldsymbol{V}_{h,i}, \tag{5.6a}$$

$$\left(\nabla \cdot \bar{\boldsymbol{u}}_h, w_h\right)_{K_i} = (f, w_h)_{K_i} \quad \forall\, w_h \in W_{h,i}. \tag{5.6b}$$

Define the coarse variational interface problem about the mortar pressure as: find $\lambda_H \in M_H$ such that

$$a_H(\lambda_H, \mu) = g_H(\mu) \quad \forall\, \mu \in M_H. \tag{5.7}$$

It is proven in [11] that the interfacpe problem (5.7) produces the solution of (5.2a)-(5.2c) via

$$\boldsymbol{u}_h = \mathbf{u}_h^*(\lambda) + \bar{\boldsymbol{u}}_h, p_h = \tilde{p}_h - \frac{1}{|\Omega|}\int_\Omega \tilde{p}_h,$$

where $\tilde{p}_h = p_h^*(\lambda) + \bar{p}_h$.

The solution of the interface problem (5.7), interpreted from the point view of multi-

scale method, is to construct multiscale basis functions over the coarse blocks. First we design a basis for M_H. For each interface E_i, from the set of mortar basis λ_H associated with this interface, we can obtain the multiscale basis $\mathbf{u}_h^*(\lambda_H)$ over the coarse domains K_{i_1} and K_{i_2}. From these, we get a system of equations from (5.7) directly, and solve it in any appropriate way.

The interface bilinear form $a_H(\cdot,\cdot)$ is symmetric and positive semi-definite on M_H and this system can be solved by preconditioned conjugate gradient method [12, 38, 96]. We will construct a preconditioner in Section 5.4. On the other hand, the mortar space M_H for problems with high contrast heterogeneous media needs to be carefully designed. Thus, in the next section, we will introduce a multiscale mortar space. In particular, we will replace the space M_H by our multiscale mortar space.

5.3 Multiscale mortar space

In this section, we present the construction of our multiscale mortar space. The space can be used as an approximation space for a multiscale mortar method for the equation (5.7), and as a coarse solver for a class of preconditioners, which will be presented in Section 5.4. Our multiscale mortar space consists of a set of multiscale basis functions, which are defined only on the coarse skeleton $\mathcal{E}_H$. To construct these basis functions, we will first define a set of snapshot functions for each coarse edge. The snapshots represent various modes of the solutions, and are typically of large size. To find the snapshots, we will solve some local problems in a small subdomain covering an edge, and then restrict the solutions to the edge. We will next define some PODs and use them to extract dominant modes within the snapshot space. These dominant modes form the multiscale basis functions. Notice that these basis functions capture some information of the permeability field within neighboring coarse blocks of an edge.

5.3.1 Construction of local snapshot space

Let $E_i \in \mathcal{E}_H^0$ be an interior edge and let ω_i be the corresponding coarse neighborhood covering E_i (see Figure 2.3). We will first construct a set of local snapshots $\{\psi_j^{E_i}\}_{j=1}^{N_{\omega_i}}$ defined on ω_i, and the local snapshot space for E_i is defined as:

$$V_{\text{snap}}^{E_i}(\omega_i) = \text{span}\{\psi_j^{E_i}\}_{j=1}^{N_{\omega_i}}.$$

The snapshots can be given explicitly or computed via solutions of local boundary value or local spectral problems in ω_i. We use the following approach. For each coarse edge E_i, we define

$$W_i(\partial\omega_i) = \{w_j^i \mid w_j^i = 1 \quad \text{on} \quad e_j^i; \quad w_j^i = 0 \quad \text{on} \quad \partial\omega_i - e_j^i, 1 \leq j \leq N_{\omega_i}\},$$

where N_{ω_i} is the number of fine edges on $\partial\omega_i$, and e_j^i is the j-th fine edge on $\partial\omega_i$. In Figure 5.1, we give an illustration of a fine edge e_j^i on E_i. To construct the snapshots, we solve the following problems ($j = 1, \cdots N_{\omega_i}$):

$$\begin{aligned} \mathbf{u}_j^i + \kappa\nabla p_j^i &= 0 \quad \text{in } \omega_i, \\ \nabla \cdot \mathbf{u}_j^i &= 0 \quad \text{in } \omega_i, \\ p_j^i &= w_j^i \quad \text{on } \partial\omega_i. \end{aligned}$$

Using the solutions of the above local problems, we obtain the basis for the snapshot space: $\{\psi_j^E = p_j^i|_{E_i}, j = 1, \cdots N_{\omega_i}\}$ associated with E_i. Finally, we construct the snapshot space as $V_{\text{snap}}^{E_i}(\omega_i) = \text{span}\{\psi_j^E, j = 1, \cdots N_{\omega_i}\}$.

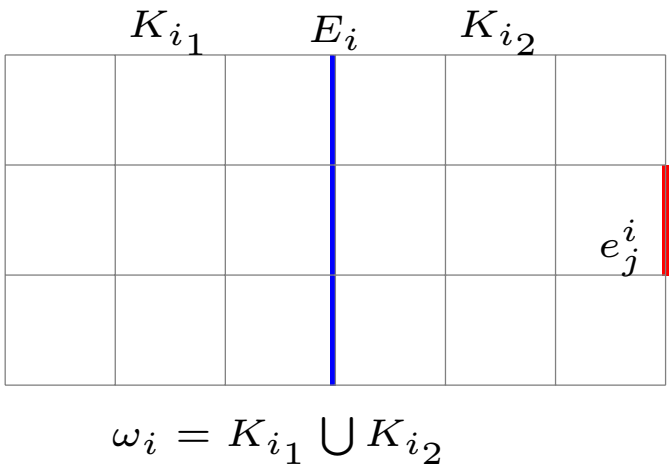


Figure 5.1: Illustration of a fine edge on a coarse edge.

5.3.2 Construction of multiscale mortar space

To construct our multiscale mortar space, we will apply a space reduction technique to the snapshot space $V_{\text{snap}}^{E_i}(\omega_i)$ to obtain a smaller dimensional space. In particular, we will perform POD to $V_{\text{snap}}^{E_i}(\omega_i)$ and then select the first l_i dominant modes Ψ_j^i. In this way, we obtain the offline space corresponding to the coarse face E_i, which is

$$V_{\text{off}}(E_i) = \text{span}\{\Psi_j^i, 1 \leq j \leq l_i\}. \tag{5.8}$$

The global offline space is

$$V_{\text{off}} = \text{span}\{\Psi_j^i, 1 \leq j \leq l_i, 1 \leq i \leq N_e\}. \tag{5.9}$$

To simplify the notations, we use the single-index notation:

$$V_{\text{off}} = \text{span}\{\Psi_i^{\text{off}} : 1 \leq i \leq N_{\text{off}}\},$$

where $N_{\text{off}} = \sum_{i=1}^{N_e} J_i$. We define

$$\mathbf{R}_{\text{off}} = [\boldsymbol{\psi}_1^{\text{off}}, \ldots, \boldsymbol{\psi}_{N_{\text{off}}}^{\text{off}}],$$

which maps from the offline space to the fine space, and $\boldsymbol{\psi}_i^{\text{off}}$ is a vector containing the coefficients in the expansion of Ψ_i^{off} in the fine-grid basis functions. We use this multiscale offline basis to enrich the constant mortar space M_H.

5.3.3 Oversampling and randomized snapshot

One can use an oversampling technique to improve the accuracy of the method. The main idea of the oversampling approach is to use larger domains ω_i^+ instead of ω_i to compute snapshot, see Figure 5.3 for examples of oversampling domain of a coarse edge. On the other hand, a typical choice of $V_{\text{snap}}^E(\omega_i)$ is local flow solutions as we explained earlier, which are constructed by solving local problems with all possible boundary conditions. This can be expensive and for this reason, one can use randomized boundary conditions [20], and construct only a few more snapshots than the number of desired local basis functions. We will test the performance of these techniques in the numerical examples in Section 5.5.

5.4 Two-level domain decomposition preconditioners

Previous sections are devoted to the design of a good coarse multiscale mortar space to achieve a desired accuracy. In this section, we will apply our multiscale mortar space in an iterative method aiming to solve the fine scale system. We propose a two-level iterative method to solve the fine scale problem with the previously introduced multiscale mortar space as the coarse space, see [14, 116] for the case of polynomial and homogenized multiscale basis. The two-level preconditioner $\mathbf{M}^{-1}$ includes two parts, a local fine scale preconditioner $\mathbf{M}_{loc}^{-1}$ to smooth out fine-scale error, and a global coarse-scale precon-

ditioner $\mathbf{M}_0^{-1}$ for exchanging global information. We begin with the construction of global coarse preconditioners.

5.4.1 Global coarse preconditioners

We denote the coarse basis functions as $\{\Psi_i\}_{i=1}^{N_{\text{off}}}$, where N_{off} is the total number of coarse basis functions, which is usually larger than the total number of coarse edges due to the use of enriched basis. Then we can define the coarse space as

$$M_H = \text{span}\{\Psi_i\}_{i=1}^{N_{\text{off}}}$$

and the coarse matrix $\mathbf{A}_0 = \mathbf{R}_0 \mathbf{S} \mathbf{R}_0^T$, where $\mathbf{R}_0^T = [\Psi_1, \Psi_2, ..., \Psi_{N_{\text{off}}}]$, and $\mathbf{S}$ is the matrix corresponding to bilinear form $a_H(\xi_i, \xi_j) : M_H \times M_H \rightarrow R$ defined in equation (5.7). Then, the coarse preconditioner is defined as

$$\mathbf{M}_0^{-1} = \mathbf{R}_0^T \mathbf{A}_0^{-1} \mathbf{R}_0.$$

The selection of coarse space M_H is quite important to the performance of the preconditioner. We will use the multiscale basis (5.9) constructed in the previous section to form the mortar space M_H.

5.4.2 Local preconditioners

The local preconditioner is $\mathbf{M}_{\text{loc}}^{-1}$ defined neighborhood wise, to compute contributions from each block. Specifically, let $\mathcal{R}_i : M_H \rightarrow M_H|_{E_i}$ be the restriction operator from $\mathcal{E}_H$ to E_i and let $\mathbf{R}_i$ be the corresponding matrix representation. For each coarse edge E_i, we consider a domain $\omega_i^+ \supset E_i$ (see Figure 5.3 for the illustration of ω_i^+) to design the local preconditioner. Similarly, we define the restriction operator from $\mathcal{E}_H$ to $E_i^+ = \mathcal{E}_H \cap \omega_i^+$ as $\mathcal{P}_i : M_H \rightarrow M_H|_{E_i^+}$ and its corresponding matrix as $\mathbf{P}_i$. We note that ω_i^+ can be the same

as ω_i. Then we define the local preconditioner as

$$\mathbf{M}_{\text{loc}}^{-1} = \sum_i \mathbf{R}_i^T (\mathbf{P}_i \mathbf{S} \mathbf{P}_i^T)^{-1} \mathbf{P}_i. \tag{5.10}$$

The implementation of $\mathbf{A}_i^{-1} = (\mathbf{P}_i \mathbf{S} \mathbf{P}_i^T)^{-1}$ is equivalent to solving a homogeneous Dirichlet boundary condition problem on ω_i^+ with local residual as source. We note that the local preconditioner defined here is about each edge, not each coarse block.

If $\omega_i^+ = \omega_i$, then we have $\mathbf{R}_i = \mathbf{P}_i$, which implies that $\mathbf{M}_{\text{loc}}^{-1}$ is symmetric and we can use PCG as outside accelerator. In this case, the local computational domain is twice the size of K_i, therefore the cost (although it is offline) of applying local preconditioners may be expensive especially in 3D case. To reduce the computational cost, we can consider the case $\omega_i^+ \neq \omega_i$, which is called the restrictive local preconditioner [19]. This will not only reduce the computation of applying local preconditioner, but also decrease the number of iterations since it includes the distant information. In this case $\mathbf{M}_{\text{loc}}^{-1}$ is no longer symmetric, and we can choose algorithm such as GMRES as the outer accelerator.

Remark 5.4.1. *Restrictive local preconditioner is quite similar to the idea of oversampling, both utilize a larger domain than standard domain to perform computation and then take restriction. Both method shows better performance than standard method.*

5.4.3 Two-level preconditioners

We combine the local preconditioner and coarse preconditioner in two ways to form the two-level preconditioners. The first approach is the *additive* preconditioner

$$\mathbf{M}_{\text{add}}^{-1} = \mathbf{M}_0^{-1} + \mathbf{M}_{\text{loc}}^{-1}. \tag{5.11}$$

The second is the *hybrid* preconditioner

$$\mathbf{M}_{\text{hyb}}^{-1} = \mathbf{M}_0^{-1} + (I - \mathbf{P}_0)\mathbf{M}_{\text{loc}}^{-1}(I - \mathbf{P}_0^T). \tag{5.12}$$

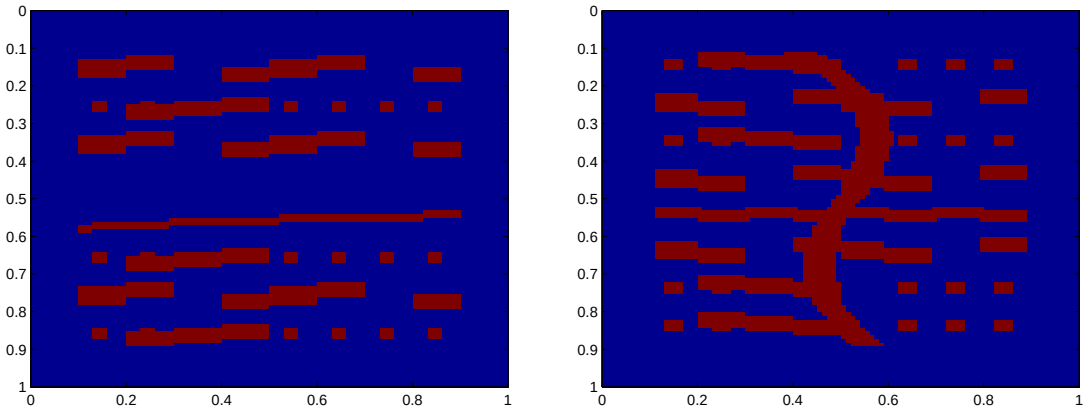
where $\mathbf{P}_0 = \mathbf{M}_0\mathbf{S}$ is the Schwarz projection operator, see [109] for details. For more details about the two-level preconditioners we adopt here, we refer to [14, 116] and reference therein. We remark again that the main ingredient in the above preconditioners is the use of our mortar multiscale space constructed in Section 5.3.

5.5 Numerical examples

In this section, we present some representative examples to show the performance of our method. We consider two models with permeability κ depicted in Figure 5.2. We note that $\kappa = 1$ in the blue region and $\kappa = \eta(\gg 1)$ in the red region. We will consider two high contrast cases: $\eta = 10^4, \eta = 10^6$ in the following examples. As it is shown, these two models contains high contrast, short and long channels, and isolated inclusions. We will first demonstrate the performance of the multiscale solver by showing the error of multiscale solution against the fine scale (reference) solution. Next we report the results of two-level preconditioners with the coarse space formed by using our multiscale mortar space. We consider different snapshot spaces computed on different domains, and also consider the two preconditioners.

We divide the domain $\Omega = (0, 1)^2$ into $N_H \times N_H$ square coarse elements. In each coarse element, we generate a uniform $N_h \times N_h$ fine scale square elements. Therefore, the domain was divided into $N_f \times N_f$ fine elements, where $N_f = N_H \times N_h$. The number of degrees of freedom for the fine solver is $5 \times N_f^2 + 2 \times N_f \times (N_f - 1)$, while the number of degree of freedom for the multiscale solver is $N_b \times 2 \times N_H \times (N_H - 1)$, where N_b is the number of multiscale basis on each coarse edge.

Constant sources and homogeneous Dirichlet boundary condition are considered. We



(a) κ_1. (b) κ_2.

Figure 5.2: Permeability fields.

use $\begin{pmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{pmatrix}$ to define local computational domain of the snapshot and the local preconditioner. See Figure 5.3 for the illustration of d_{ij}. In total, 4 computational domains to generate the multiscale space are considered:

Domain 1: No oversampling: $\begin{pmatrix} N_h & 0 \\ 0 & N_h \end{pmatrix}$

Domain 2: oversampling case a: $\begin{pmatrix} N_h & 1 \\ 1 & N_h \end{pmatrix}$

Domain 3: oversampling case b: $\begin{pmatrix} [N_h/2] & 1 \\ 1 & [N_h/2] \end{pmatrix}$

Domain 4: oversampling case c: $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$

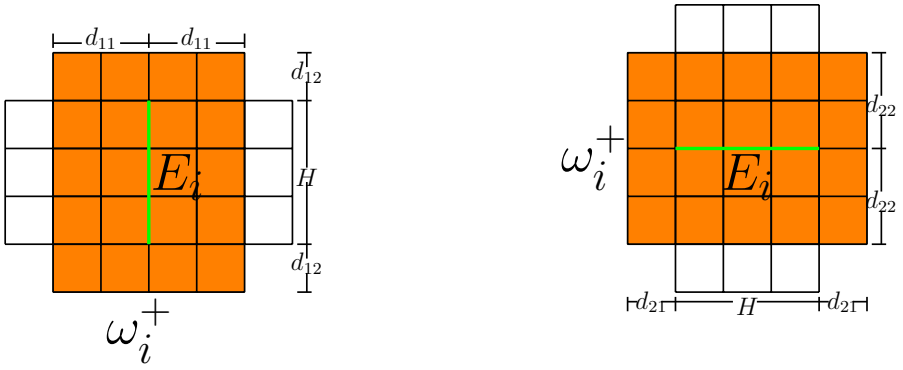


Figure 5.3: Illustration of an oversampled neighborhood.

5.5.1 Coarse grid multiscale solution

In this subsection, we study the error decay of multiscale solution by adding more multiscale basis. We consider using 4 types of multiscale basis as well as polynomial basis for comparison. We demonstrate the influence of snapshot, mesh size and contrast of the permeability on the multiscale solution. We define the following error to quantify the accuracy of coarse grid multiscale solution.

$$e_p := \frac{\|p_{ms} - p_f\|_{L^2,\Omega}}{\|p_f\|_{L^2,\Omega}}, \quad e_{\boldsymbol{u}} := \frac{\|\boldsymbol{u}_{ms} - \boldsymbol{u}_f\|_{\kappa,\Omega}}{\|\boldsymbol{u}_f\|_{\kappa,\Omega}}$$

where $\|\boldsymbol{u}\|^2_{\kappa,\Omega} = \int_\Omega \kappa^{-1}\boldsymbol{u}^2$.

The 4 types of multiscale basis based on the way of generating the snapshot are:

Case 1: full snapshot on domain 1

Case 2: full snapshot on domain 2

Case 3: full snapshot on domain 3

Case 4: $(N_h + 2)$ randomized snapshot on domain 2.

Table 5.1 and Table 5.2 present the numerical results for model 1 with mesh setting $N_H = 5, N_h = 20$ and $N_H = 10, N_h = 10$ respectively. The first column is the number of

multiscale basis for each coarse edge. The rest of the columns present both errors $e_p, e_{\boldsymbol{u}}$ for using polynomial basis, and 4 types of multiscale basis as described before. As it is shown, by adding basis, both types of errors decrease for all types of basis considered. Moreover, the error decay of all multiscale basis cases are faster than the polynomial basis case. For example, in Table 5.1, the relative weighted velocity error $e_{\boldsymbol{u}}$ decreases from 78.6% to 6.2% in case 2 when $N_H = 5$; however, the corresponding error decreases from 78.2% to 26.7% for polynomial basis. We remark that 6% error is tolerable in industry flow simulation. By using 5 basis on each coarse edge, the error e_p for the polynomial case is 10.3%, while this error for the multiscale basis cases are below ten percent. Note that, by using 5 basis on each coarse edge, the number of degree of freedom of multiscale solver is about 0.3% of that of the fine solver. By comparing case 1 and case 2(oversampling case), we can observe obvious improvement by applying oversampling although only 1 fine scale element is added to the snapshot domain. We remark that adding more fine scale elements to the domain of computing snapshot will further improve the solution. From the results of case 2 and case 4, we can see that the performance of randomized snapshot is also comparable with full snapshot, and it is better than case 3 and case 1. In Figure 5.4, for model 1, the reference solution and the corresponding multiscale solutions with different number of basis on each coarse edge are presented. The upper left is the reference solution. The upper right is the multiscale solution with 1 basis on each coarse edge. This solution has obvious discontinuity across the coarse edges. The bottom left shows the multiscale solution with 3 basis on each coarse edge, which is closer to the reference solution. However, we can see slight difference between this solution and reference solution. The bottom right is the multiscale solution with 5 basis on each coarse edge, this solution is almost identical with the reference solution. This figure shows that the additional multiscale basis functions are important to capture all the features of the solution.

By comparing the errors in Table 5.1 and Table 5.2, we notice that smaller coarse grid

size can improve the accuracy of the solution. For example, the second column in Table 5.1, e_u for polynomial basis drops from 63.0% to 10.3%, while the same error in Table 5.2 drops from 60.9% to 5.0%; the third column in Table 5.1, $e_{\boldsymbol{u}}$ for Case 1 drops from 78.6% to 8.8%, while the same error in Table 5.2 drops from 77.8% to 2.7%.

Table 5.3 and Table 5.4 show corresponding results for model 2. We observe similar results as model 1. Both types of errors decrease for all types of basis considered by adding more basis. The error decay of all multiscale basis cases are faster than the polynomial basis case. For example, if $N_H = 10$ by adding to 5 basis on each coarse edge, the error e_p for polynomial case decreases to 1.9%; while the errors for multiscale cases decrease to 0.6%, 0.04%, 1.1%, 0.08% respectively. By comparing case 1 and case 2, we can again observe obvious improvement by applying oversampling although only 1 fine scale element is added to the snapshot domain. From the results of case 2 and case 4, we can see that the performance of randomized snapshot is also comparable with full snapshot, and it is better than case 3 and case 1.

In Figure 5.5, for model 2, the reference solution and the corresponding multiscale solutions with different numbers of basis on each coarse edge are presented. The upper left is the reference solution. The upper right is the multiscale solution with 1 basis on each coarse edge. Apparently the solution is discontinuous across the coarse edges. The bottom left is the multiscale solution with 3 basis on each coarse edge, which shows better agreement with the reference solution. However, there are still some detailed features missing in the multiscale solution . The bottom right is the multiscale solution with 5 basis on each coarse edge, which has good agreement with the reference solution. This figure shows that the additional multiscale basis functions are important to capture all the features of the solution. These results for model 2 demonstrate that our multiscale solver can handle permeability field with long channel effects.

We also test the robustness of our method by varying the order of high contrast. The

results are presented in Figure 5.7 and Figure 5.6. In Figure 5.7, the errors e_p(left), $e_{\boldsymbol{u}}$ (right) for model 1 with high contrast order $\eta = 10^4, 10^6$ with coarse $N_H = 5, N_h = 10$ are plotted. The black dashed line is for the case $10^4, N_H = 5$, the blue dashed line is for the case $10^6, N_H = 5$. The two lines are identical. The green dashed line is for the case $10^4, N_H = 10$, the red dashed line is for the case $10^6, N_H = 10$. The two lines are also identical.

Figure 5.6 displays the corresponding results for model 2. Though the lines do not overlap, the difference is very small. We note that the snapshot we used for both cases comes from case 2. From these examples, we see that our method produces robust results independent of the order of high contrast. Robustness will further be confirmed by our preconditioner results in following parts.

Table 5.1: Relative error between multiscale solution and fine scale solution with different types of basis for model 1, $N_H = 5$, $\eta = 10^4$. "Nb" represent the number of basis per coarse edge.

Nb	Polynomial		Case 1		Case 2		Case 3		Case 4	
	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$
1	0.630	0.786	0.630	0.786	0.630	0.786	0.630	0.786	0.630	0.786
2	0.432	0.647	0.438	0.649	0.357	0.581	0.351	0.581	0.427	0.640
3	0.288	0.519	0.157	0.374	0.139	0.354	0.143	0.357	0.129	0.341
4	0.120	0.291	0.060	0.232	0.014	0.104	0.050	0.207	0.038	0.171
5	0.103	0.267	0.012	0.088	0.006	0.062	0.028	0.157	0.008	0.073

Table 5.2: Relative error between multiscale solution and fine scale solution with different types of basis for model 1, $N_H = 10$, $\eta = 10^4$. "Nb" represent the number of basis per coarse edge.

Nb	Polynomial		Case 1		Case 2		Case 3		Case 4	
	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$
1	0.609	0.778	0.609	0.778	0.609	0.778	0.609	0.778	0.609	0.778
2	0.133	0.322	0.140	0.329	0.130	0.316	0.133	0.320	0.106	0.287
3	0.103	0.278	0.026	0.128	0.019	0.114	0.039	0.167	0.026	0.135
4	0.069	0.232	0.012	0.092	0.010	0.083	0.029	0.142	0.004	0.050
5	0.050	0.197	0.001	0.027	3.9e-04	0.013	0.002	0.035	6.7e-04	0.020

Table 5.3: Relative error between multiscale solution and fine scale solution with different types of basis for model 2, $N_H = 5$, $\eta = 10^4$. "Nb" represent the number of basis per coarse edge.

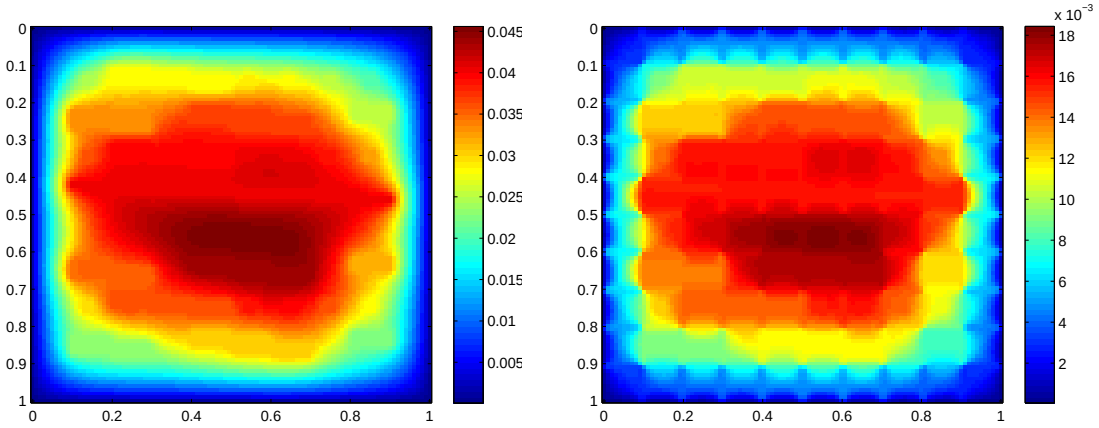
Nb	Polynomial		Case 1		Case 2		Case 3		Case 4	
	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$
1	0.668	0.806	0.668	0.806	0.668	0.806	0.668	0.806	0.668	0.806
2	0.508	0.699	0.549	0.728	0.410	0.627	0.447	0.655	0.466	0.651
3	0.332	0.560	0.265	0.494	0.151	0.370	0.151	0.370	0.142	0.358
4	0.118	0.316	0.136	0.348	0.038	0.178	0.039	0.181	0.041	0.187
5	0.077	0.244	0.061	0.211	0.012	0.089	0.009	0.075	0.026	0.153

Table 5.4: Relative error between multiscale solution and fine scale solution with different types of basis for model 2, $N_H = 10$, $\eta = 10^4$. "Nb" represent the number of basis per coarse edge.

Nb	Polynomial		Case 1		Case 2		Case 3		Case 4	
	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$	e_p	$e_{\boldsymbol{u}}$
1	0.593	0.769	0.593	0.769	0.593	0.769	0.593	0.769	0.593	0.769
2	0.055	0.205	0.118	0.304	0.090	0.266	0.090	0.267	0.074	0.245
3	0.044	0.175	0.031	0.146	0.011	0.082	0.017	0.109	0.014	0.098
4	0.029	0.142	0.015	0.101	0.002	0.035	0.007	0.065	0.004	0.049
5	0.019	0.115	0.006	0.055	4.4e-04	0.011	0.001	0.023	7.6e-04	0.017

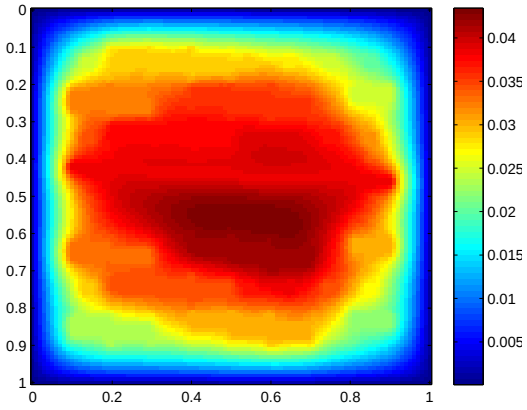
5.5.2 Preconditioner

In this subsection, we present the numerical results of using multiscale basis to form the coarse space for the two level additive Schwarz domain decomposition preconditioner. We use PCG as outer accelerator if the two-level preconditioner is symmetric, otherwise GMRES with restarted number of 2 (GMRES(2)) is applied. We adopt the techniques in [57] to implement the local preconditioner M_{loc}^{-1}. Then the dominant computation can be done offline and it can be parallelized with coloring techniques. The dimension of each local preconditioner is equal to the number of fine scale edges in the adjacent coarse elements, which is quite small and thus can be precomputed and saved. Direct solver is used to implement M_0^{-1}. We consider both additive and hybrid preconditioners. We are particularly interested in the robustness (robust is defined as that the iteration number is independent of contrast) of the method, and in each simulation we consider three types of

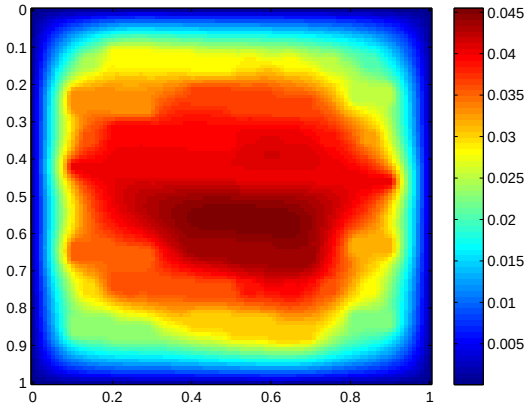


(a) Fine-scale solution.

(b) Coarse-scale solution(1 basis).

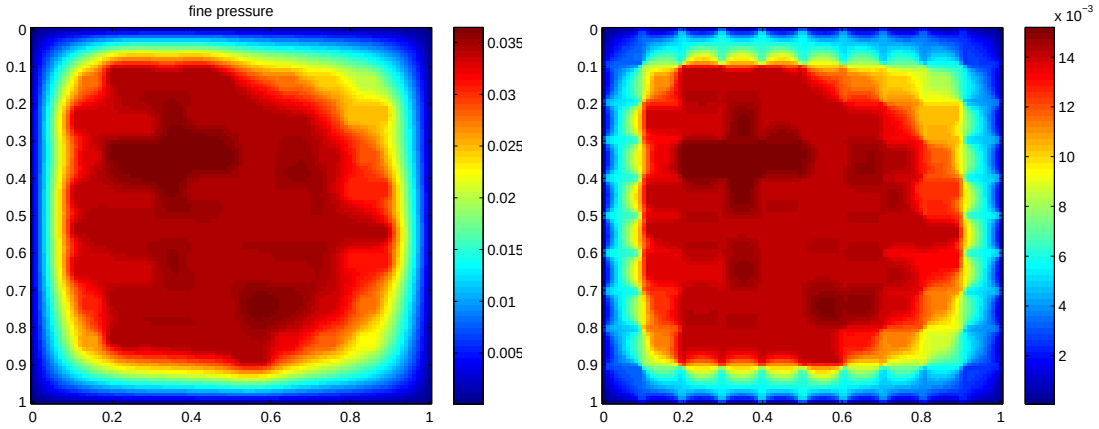


(c) Coarse-scale solution(3 basis).



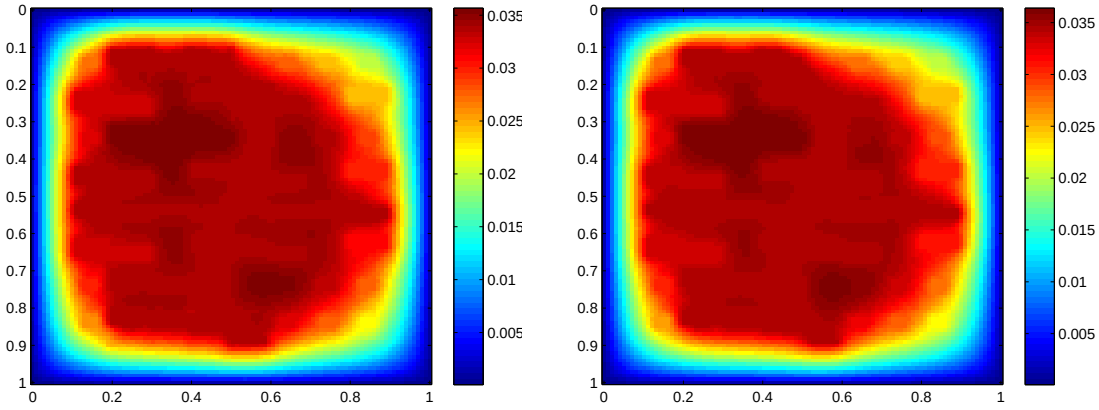
(d) Coarse-scale solution(5 basis).

Figure 5.4: Comparison of the coarse-scale solutions with the reference (fine-scale) solution, $N_H = 10$, $\eta = 10^4$, model 1.



(a) Fine-scale solution.

(b) Coarse-scale solution(1 basis).



(c) Coarse-scale solution(5 basis).

(d) Coarse-scale solution(5 basis).

Figure 5.5: Comparison of the coarse-scale solutions with the reference (fine-scale) solution, $N_H = 10$, $\eta = 10^4$, model 2.

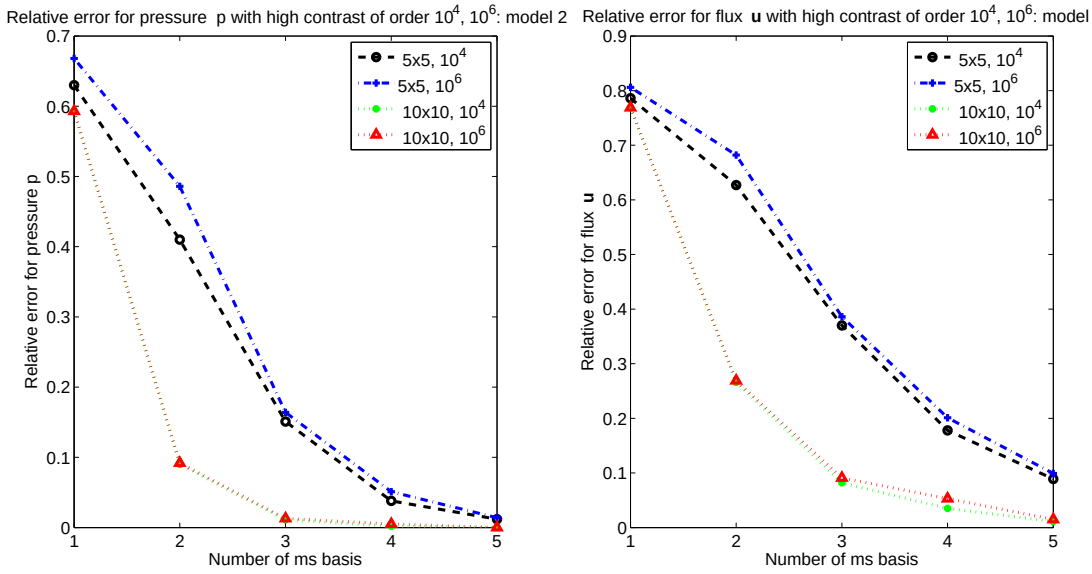


Figure 5.6: Relative error for p (left), $\mathbf{u}$ (right) with contrast order $\eta = 10^4$ and $\eta = 10^6$ for model 2, basis generation case 2.

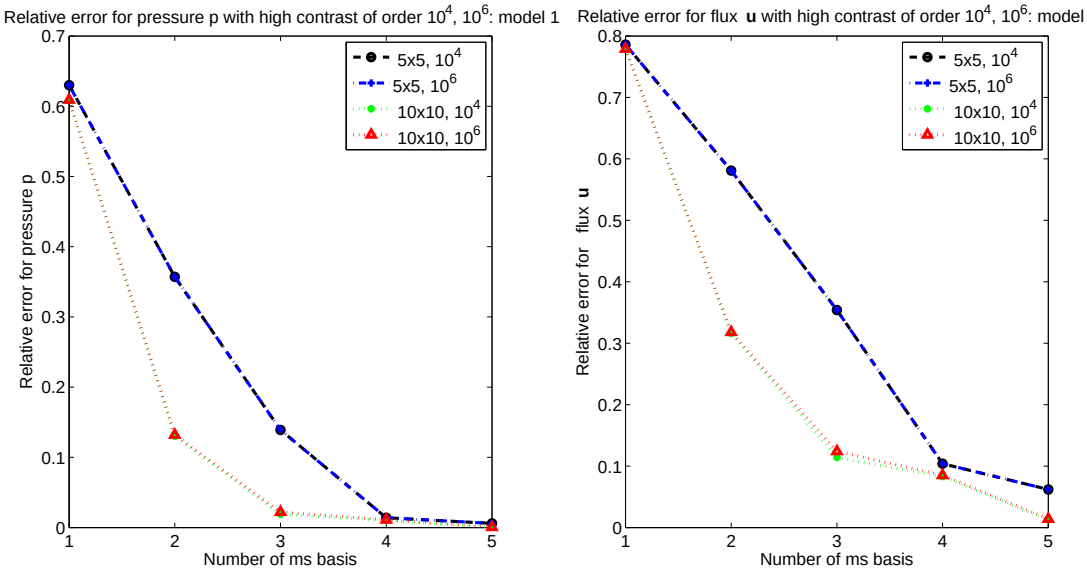


Figure 5.7: Relative error for p (left), $\mathbf{u}$ (right) with contrast order $\eta = 10^4$ and $\eta = 10^6$ for model 1, basis generation case 2.

contrast to test the robustness. The initial guess is zero, and the stopping criterion is that the residual is reduced by a factor of 10^7 in L^2 norm.

The first test involves comparing the PCG iteration number of using multiscale basis and polynomial basis to form coarse space of the coarse preconditioner, which is shown in Table 5.5. The first column gives the order of high contrast. The second column is the number of PCG iterations of using polynomial to form the coarse space. The rest of the columns give the number of PCG iterations of using multiscale basis from Case 3 and Case 4 to form the coarse space. We can see clearly that for each case, the iteration number of multiscale basis is generally much smaller than that of polynomial basis. If polynomial basis is used, the preconditioner is not robust with respect to the contrast of the media. For example, for the hybrid method in the second column, PCG iteration number is 13 for contrast 10^2, and increases to 39 for contrast 10^6. While for the multiscale basis, iteration number is almost independent of contrast. Hybrid preconditioner performs better than additive preconditioner in terms of iteration number, however, hybrid preconditioner requires to apply the coarse preconditioner twice in each iteration.

Next, we focus our study on the influence of the domain size on local preconditioner. In Table 5.6, the GMRES(2) iteration numbers of using multiscale basis from case 3 with four types of computational domain are presented. Two multiscale basis on each coarse edge is used to form the coarse space. The first column gives the order of high contrast. The rest of the columns give the number of GMRES(2) outer iterations for the four types of computational domain. As it is shown, a restrictive local preconditioner (cases of Domain 2, 3, 4) can reduce the iteration number approximately by half by comparing the results of Domain 1 and the rest of the 3 domain cases. For example, the number of iteration for the additive case from Domain 1 for contrast of order $10^2, 10^4, 10^6$ is 20, 18, 18 respectively, while from Domain 2 the number of iteration for the additive case is 10, 11, 11. Moreover, iteration number is almost independent of contrast for all types of domains. In Table 5.7,

the GMRES(2) iteration numbers of using multiscale basis from case 4 with four types of computational domain are presented. We observed similar results as in Table 5.6. Since multiscale basis from case 4 uses randomized technique for snapshot, it is acceptable that the number of iterations are slightly larger than the number of iterations of corresponding cases in in Table 5.6.

Table 5.5: PCG iteration number with different types of coarse space, 2 basis is used for coarse space, $N_H = 5$.

Contrast	Polynomial		Case 3		Case 4	
	additive	hybrid	additive	hybrid	additive	hybrid
10^2	23	13	25	16	26	16
10^4	> 40	28	27	16	29	17
10^6	> 40	39	27	16	28	16

Table 5.6: GMRES(2) iteration number with different types of preconditioners and different local preconditioner, 2 basis is used for coarse space with multiscale basis case 3, $N_H = 5$.

Contrast	Domain 1		Domain 2		Doamin 3		Domain 4	
	additive	hybrid	additive	hybrid	additive	hybrid	additive	hybrid
10^2	20	10	10	5	10	5	11	6
10^4	18	10	11	5	12	5	13	7
10^6	18	10	11	5	12	5	11	6

Table 5.7: GMRES(2) iteration number with different types of preconditioners and different local preconditioner, 2 basis is used for coarse space with multiscale basis case 4, $N_H = 5$.

Contrast	Domain 1		Domain 2		Doamin 3		Domain 4	
	additive	hybrid	additive	hybrid	additive	hybrid	additive	hybrid
10^2	17	10	15	7	12	8	16	9
10^4	20	8	18	5	18	10	18	7
10^6	18	10	14	6	12	7	16	9

6. CONCLUSIONS

In this dissertation, we mainly focus on the development and application of global, local model reduction methods for subsurface flow simulation. We now summarize each component.

We present a reduced-order modeling approach for simulations of two-phase flow and transport. The approach incorporates the use of POD and DEIM to construct a reduced model that is less computational expensive while reproduce the full-order input/output behavior with certain accuracy. Our approach uses POD Galerkin projection to setup the whole system on a reduced dimensional subspace. In addition, DEIM is used to approximate nonlinear terms so that the resulting system is solved at the online stage with a much less computational cost compared to the fine-grid solver. However, because POD based techniques use Galerkin (or Petrov-Galerkin) projections, the resulting system may not have mass conservation property. We use reduced dimensional basis for velocity field within mixed finite element framework [2, 3]. We show a combination of multiscale methods with reduced-order modeling in an approach called global-local model reduction. In this approach, the global snapshots are computed using local multiscale methods based on Generalized Multiscale Finite Element Method where a few multiscale basis functions are computed for each coarse region and re-used for all input data. One can control accuracy of local as well as global approaches by adding additional basis functions in each coarse region or adding global basis functions.

We also propose an online adaptive global-local POD-DEIM model reduction method for nonlinear systems where both POD subspaces and DEIM interpolants are adapted during the online stage by incorporating online information locally. In the global adaptive framework, we employ local techniques to realize adaption leading to computational sav-

ings in evaluating the residual. Our approach is particularly useful for situations where it is desired to solve the reduced system for inputs or controls that result in a solution outside the span of the snapshots generated in the offline stage. This happens in many applications, such as inverse problem and optimization, where the final solution path may be difficult to predict before the problem is solved. Our method also offers an alternative to constructing a robust reduced system even if a potential initial poor choice of snapshots is used, since it can absorb representative information into the POD solution space along the online simulation process.

We have also worked on developing a local multiscale approach. The method is based on a mixed formulation of the problem, the concepts of domain decomposition, and mortar techniques. The multiscale basis functions that fully resolve the problem within the subdomains are constructed from local problems by following the framework of the Generalized Multiscale Finite Element Method (GMsFEM). Using the proposed multiscale mortar space, we construct a multiscale finite element coarse grid solver for the flows. Further, we design both two-level *additive*, *hybrid* preconditioners which can be used within a Krylov accelerator such as PCG or GMRES as an exact solver. These two-level preconditioners consist of a local smoothing preconditioner based on block Jacobi(BJ), blocked by subdomain interfaces, and a coarse preconditioner based on subdomain interfaces using the enriched multiscale mortar space. Finally, we present some numerical examples to show that the proposed coarse space gives promising ability to deal with problems in media with complicated inclusions and long channels that may across coarse edges.

REFERENCES

[1] J. E. Aarnes, V. L. Hauge, and Y. Efendiev. Coarsening of three-dimensional structured and unstructured grids for subsurface flow. *Advances in Water Resources*, 30(11):2177–2193, 2007.

[2] J. E. Aarnes, S. Krogstad, and K. A. Lie. A hierarchical multiscale method for two-phase flow based upon mixed finite elements and nonuniform coarse grids. *Multiscale Modeling & Simulation*, 5(2):337–363, 2006.

[3] J. E. Aarnes, S. Krogstad, and K. A. Lie. Multiscale mixed/mimetic methods on corner-point grids. *Computational Geosciences*, 12(3):297–315, 2008.

[4] Z. Alghareeb. *Optimal reservoir management using adaptive reduced order models*. PhD thesis, Massachusetts Institute of Technology, 2015.

[5] M. Alotaibi, V. M. Calo, Y. Efendiev, J. Galvis, and M. Ghommem. Global–local nonlinear model reduction for flows in heterogeneous porous media. *Computer Methods in Applied Mechanics and Engineering*, 292:122–137, 2015.

[6] D. Amsallem and C. Farhat. An interpolation method for adapting reduced-order models and application to aeroelasticity. *AIAA Journal*, 46(7):1803–1813, 2008.

[7] D. Amsallem and C. Farhat. An online method for interpolating linear parametric reduced-order models. *SIAM Journal on Scientific Computing*, 33(5):2169–2198, 2011.

[8] D. Amsallem, M. Zahr, Y. Choi, and C. Farhat. Design optimization using hyper-reduced-order models. *Structural and Multidisciplinary Optimization*, pages 1–22, 2014.

[9] R. Araya, C. Harder, D. Paredes, and F. Valentin. Multiscale hybrid-mixed method. *SIAM Journal on Numerical Analysis*, 51(6):3505–3531, 2013.

[10] T. Arbogast. Analysis of a two-scale, locally conservative subgrid upscaling for elliptic problems. *SIAM Journal on Numerical Analysis*, 42(2):576–598, 2004.

[11] T. Arbogast, L. C. Cowsar, M. F. Wheeler, and I. Yotov. Mixed finite element methods on nonmatching multiblock grids. *SIAM Journal on Numerical Analysis*, 37(4):1295–1315, 2000.

[12] T. Arbogast, G. Pencheva, M. F. Wheeler, and I. Yotov. A multiscale mortar mixed finite element method. *SIAM Multiscale Modeling & Simulation*, 6(1):319–346, 2007.

[13] T. Arbogast and H. Xiao. A multiscale mortar mixed space based on homogenization for heterogeneous elliptic problems. *SIAM Journal on Numerical Analysis*, 51(1):377–399, 2013.

[14] T. Arbogast and H. Xiao. Two-level mortar domain decomposition preconditioners for heterogeneous elliptic problems. *Computer Methods in Applied Mechanics and Engineering*, 292:221–242, 2015.

[15] P. Astrid, S. Weiland, K. Willcox, and T. Backx. Missing point estimation in models described by proper orthogonal decomposition. *IEEE Transactions on Automatic Control*, 53(10):2237–2251, 2008.

[16] C. F. Barnes and R. L. Frost. Residual vector quantizers with jointly optimized code books. *Advances in Electronics and Electron Physics*, 84:1–59, 1992.

[17] M. Bergmann, C. H. Bruneau, and A. Iollo. Enablers for robust POD models. *Journal of Computational Physics*, 228(2):516–538, 2009.

[18] T. Bui-Thanh, M. Damodaran, and K. Willcox. Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition. *AIAA Journal*, 42(8):1505–1516, 2004.

[19] X. Cai and M. Sarkis. A restricted additive Schwarz preconditioner for general sparse linear systems. *SIAM Journal on Scientific Computing*, 21(2):792–797, 1999.

[20] V. M. Calo, Y. Efendiev, J. Galvis, and G. Li. G. Randomized oversampling for generalized multiscale finite element methods. *Multiscale Modeling & Simulation*, 14(4):482–501, 2016.

[21] Y. Cao, J. Zhu, Z. Luo, and I. M. Navon. Reduced order modeling of the upper tropical pacific ocean model using proper orthogonal decomposition. *Computers & Mathematics with Applications*, 52(8):1373–1386, 2006.

[22] M. A. Cardoso and L. J. Durlofsky. Use of reduced-order modeling procedures for production optimization. *SPE Journal*, 15(2):426–435, 2010.

[23] M. A. Cardoso, L. J. Durlofsky, and P. Sarma. Development and application of reduced-order modeling procedures for subsurface flow simulation. *International Journal for Numerical Methods in Engineering*, 77(9):1322–1350, 2009.

[24] H. Y. Chan, E. T. Chung, and Y. Efendiev. Adaptive mixed gmsfem for flows in heterogeneous media. *arXiv:1507.01659*, 2015.

[25] S. Chaturantabut and D. C. Sorensen. Discrete empirical interpolation for nonlinear model reduction. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010.

[26] P. Chen, A. Quarteroni, and G. Rozza. A weighted reduced basis method for elliptic partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 56(6):3163–3185, 2013.

[27] Z. Chen and T. Y. Hou. A mixed multiscale finite element method for elliptic problems with oscillating coefficients. *Mathematics of Computation*, 72(242):541–576, 2003.

[28] M. A. Christie and M. J. Blunt. Tenth SPE comparative solution project: a comparison of upscaling techniques. SPE-72469. *SPEREE*, 4:308–317, 2001.

[29] E. T. Chung, Y. Efendiev, and S. Fu. Generalized multiscale finite element method for elasticity equations. *GEM-International Journal on Geomathematics*, 5(2):225–254, 2014.

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

[31] E. T. Chung, Y. Efendiev, and C. S. Lee. Mixed generalized multiscale finite element methods and applications. *Multiscale Modeling & Simulation*, 13(1):338–366, 2015.

[32] E. T. Chung, Y. Efendiev, and W. Leung. Residual-driven online generalized multiscale finite element methods. *Journal Computational Physics*, 302:176, 2015.

[33] E. T. Chung, Y. Efendiev, and G Li. An adaptive gmsfem for high-contrast flow problems. *Journal of Computational Physics*, 273:54, 2014.

[34] E. T. Chung, Y. Efendiev, G. Li, and M. Vasilyeva. Generalized multiscale finite element methods for problems in perforated heterogeneous domains. *Applicable Analysis*, pages 1–26, 2015.

[35] E. T. Chung, S. Fu, and Y. Yang. An enriched multiscale mortar space for high contrast flow problems. *arXiv:1609.02610*, 2016.

[36] E. T. Chung, W. T. Leung, and S. Pollock. Goal-oriented adaptivity for gmsfem. *Journal of Computational and Applied Mathematics*, 296:625–637, 2016.

[37] E. T. Chung, W. T. Leung, and M. Vasilyeva. Mixed GMsFEM for second order elliptic problem in perforated domains. *Journal of Computational and Applied Mathematics*, 304:84–99, 2016.

[38] B. Cockburn, J. Gopalakrishnan, and R. Lazarov. Unified hybridization of discontinuous Galerkin, mixed, and continuous Galerkin methods for second order elliptic problems. *SIAM Journal on Numerical Analysis*, 47(2):1319–1365, 2009.

[39] V. Dolean, F. Nataf, R. Scheichl, and N. Spillane. Analysis of a two-level Schwarz method with coarse spaces based on local Dirichlet-to-Neumann maps. *Computational Methods in Applied Mathematics*, 12(4):391–414, 2012.

[40] L. Durlofsky, Y. Efendiev, and V. Ginting. An adaptive local–global multiscale finite volume element method for two-phase flow simulations. *Advances in Water Resources*, 30:576–588, 2007.

[41] L. J. Durlofsky. Numerical calculation of equivalent grid block permeability tensors for heterogeneous porous media. *Water Resources Research*, 27:699–708, 1991.

[42] H. C. Edwards. A parallel multilevel-preconditioned GMRES solver for multiphase flow models in the Implicit Parallel Accurate Reservoir Simulator. Technical Report 98-04, TICAM, University of Texas at Austin, 1998.

[43] Y. Efendiev, J. Galvis, and E. Gildin. Local–global multiscale model reduction for flows in high-contrast heterogeneous media. *Journal of Computational Physics*, 231(24):8100–8113, 2012.

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

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

[46] Y. Efendiev, J. Galvis, R. Lazarov, and J. Willems. Robust domain decomposition preconditioners for abstract symmetric positive definite bilinear forms. *ESAIM: Mathematical Modelling and Numerical Analysis*, 46(5):1175–1199, 2012.

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

[48] Y. Efendiev, E. Gildin, and Y. Yang. Online adaptive local-global model reduction for flows in heterogeneous porous media. *Computation*, 4(2), 2016.

[49] Y. Efendiev and T. Y. Hou. *Multiscale finite element methods: theory and applications*, volume 4. Springer, 2009.

[50] Y. Efendiev, R. Lazarov, M. Moon, and K. Shi. A spectral multiscale hybridizable discontinuous Galerkin method for second order elliptic problems. *Computer Methods in Applied Mechanics and Engineering*, 292:243–256, 2015.

[51] Y. Efendiev, R. Lazarov, and K. Shi. A multiscale HDG method for second order elliptic equations. Part I. Polynomial and homogenization-based multiscale spaces. *SIAM Journal on Numerical Analysis*, 53(1):342–369, 2015.

[52] M. Fortin and F. Brezzi. *Mixed and hybrid finite element methods*. Springer, 1991.

[53] S. Fu, Y. Efendiev, and R. Gao, K.and Gibson. Multiscale modeling of acoustic wave propagation in 2d heterogeneous media using local spectral basis functions. In *SEG Technical Program Expanded Abstracts*, volume 2013, pages 3553–3558, 2013.

[54] D. Galbally, K. Fidkowski, K. Willcox, and O. Ghattas. Non-linear model reduction for uncertainty quantification in large-scale inverse problems. *International Journal for Numerical Methods in Engineering*, 81(12):1581–1608, 2010.

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

[56] J. Galvis and Y. Efendiev. Domain decomposition preconditioners for multiscale flows in high contrast media: reduced dimension coarse spaces. *SIAM Multiscale Modeling & Simulation.*, 8(5):1621–1644, 2010.

[57] B. Ganis and I. Yotov. Implementation of a mortar mixed finite element method using a multiscale flux basis. *Computer Methods in Applied Mechanics and Engineering*, 198(49):3989–3998, 2009.

[58] L. Gao, X. Tan, and E. T. Chung. Application of the generalized multiscale finite element method in parameter-dependent PDE simulations with a variable-separation technique. *Journal of Computational and Applied Mathematics*, 300:183–191, 2016.

[59] M. Ghasemi, Y. Yang, E. Gildin, Y. Efendiev, and V. Calo. Fast multiscale reservoir simulations using POD-DEIM model reduction. In *SPE Reservoir Simulation Symposium*, Houston, Texas, Feb 2015. Society of Petroleum Engineers. SPE 173271.

[60] G. Graham, P. O. Lechner, and R. Scheichl. Domain decomposition for multiscale PDEs. *Numerische Mathematik*, 106(4):589–626, 2007.

[61] D. Gratton. Reduced-order, trajectory piecewise-linear models for nonlinear computational fluid dynamics. *Master's thesis, Massachusetts Institute of Technology*, 2004.

[62] M. A. Grepl, Y. Maday, N. C. Nguyen, and A. T. Patera. Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations. *Mathematical Modeling and Numerical Analysis*, 41(3):575–605, 2007.

[63] E. J. Grimme. *Krylov projection methods for model reduction*. PhD thesis, University of Illinois at Urbana-Champaign, 1997.

[64] S. Gugercin and A. C. Antoulasy. A survey of model reduction by balanced truncation and some new results. *International Journal of Control*, 77(8):748–766, 2004.

[65] B. Haasdonk and M. Ohlberger. Adaptive basis enrichment for the reduced basis method applied to finite volume schemes. *Finite Volumes for Complex Applications V*, pages 471–479, 2008.

[66] K. C. Hall, J. P. Thomas, and E. H. Dowell. Proper orthogonal decomposition technique for transonic unsteady aerodynamic flows. *AIAA Journal*, 2000.

[67] F. P. Hamon, B. T. Mallison, and H. A. Tchelepi. Implicit hybrid upwind scheme for coupled multiphase flow and transport with buoyancy. *Computer Methods in Applied Mechanics and Engineering*, 311:599 – 624, 2016.

[68] C. Harder, D. Paredes, and F. Valentin. A family of multiscale hybrid-mixed finite element methods for the Darcy equation with rough coefficients. *Journal of Computational Physics*, 245:107–130, 2013.

[69] J. He and L. Durlofsky. *Reduced-order modeling for compositional simulation using trajectory piecewise linearization*. Paper SPE 163 presented at the SPE Reservoir Simulation Symposium, The Wooldlands, Texas, USA, 2013.

[70] T. Heijn, R. Markovinovic, and J. D. Jansen. Generation of low-order reservoir models using system-theoretical concepts. *SPE Journal*, 9(2):202–218, 2004.

[71] T. Hughes, G. Feijoo, L. Mazzei, and J. Quincy. The variational multiscale method - a paradigm for computational mechanics. *Computer Methods in Applied Mechanics and Engineering*, 166:3–24, 1998.

[72] O. Iliev, R. Lazarov, and J. Willems. Variational multiscale finite element method for flows in highly porous media. *Multiscale Modeling & Simulation*, 9.4:1350–1372, 2011.

[73] P. Jenny, S. H. Lee, and H. Tchelepi. Multi-scale finite volume method for elliptic problems in subsurface flow simulation. *Journal of Computational Physics*, 187:47–67, 2003.

[74] I. Kalashnikova and M. F. Barone. Efficient non-linear proper orthogonal decomposition (pod)/Galerkin reduced order models with stable penalty enforcement of boundary conditions. *International Journal for Numerical Methods in Engineering*, 00:1–28, 2011.

[75] A .Kellems, S. Chaturantabut, D. Sorensen, and S. Cox. Morphologically accurate reduced order modeling of spiking neurons. *International Journal for Numerical Methods in Engineering*, 28(3):477–94, 2010.

[76] E. T. Kim, H. H.and Chung and C. Xu. A BDDC algorithm with adaptive primal constraints for staggered discontinuous Galerkin approximation of elliptic problems with highly oscillatory coefficients. *To appear in Journal of Computational and Applied Mathematics*.

[77] H. H. Kim and E. T Chung. A BDDC algorithm with enriched coarse spaces for two-dimensional elliptic problems with oscillatory and high contrast coefficients. *SIAM Multiscale Modeling & Simulation*, 13(2):571–593, 2015.

[78] H. H. Kim, E. T. Chung, and J. Wang. BDDC and FETI-DP algorithms with adaptive coarse spaces for three-dimensional elliptic problems with oscillatory and high contrast coefficients. *arXiv preprint arXiv:1606.07560*, 2016.

[79] A. Klawonn, P. Radtke, and O. Rheinbach. FETI-DP methods with an adaptive coarse space. *SIAM Journal on Numerical Analysis*, 53(1):297–320, 2015.

[80] K. Kunisch and S. Volkwein. Control of the burgers equation by a reduced-order approach using proper orthogonal decomposition. *Journal of Optimization Theory and Applications*, 102:345–371, 1999.

[81] K. Kunisch and S. Volkwein. Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics. *SIAM Journal on Numerical Analysis*, 40(2):492–515, 2002.

[82] A. Tan Yong Kwang. *Reduced basis method for 2nd order wave equation: application to one-dimensional seimic problem*. PhD thesis, Massachusetts Institute of Technology, 2005.

[83] S. H. Lee and Y. Efendiev. C 1-continuous relative permeability and hybrid upwind discretization of three phase flow in porous media. *Advances in Water Resources*, 96:209–224, 2016.

[84] S. H. Lee, Y. Efendiev, and H. A. Tchelepi. Hybrid upwind discretization of nonlinear two-phase flow with gravity. *Advances in Water Resources*, 82:27–38, 2015.

[85] J. L. Lumley. Atmospheric turbulence and radio wave propagation. *Journal of Computational Chemistry*, 23(13):1236–1243, 1967.

[86] Y. Maday and B. Stamm. Locally adaptive greedy approximations for anisotropic parameter reduced basis spaces. *SIAM Journal on Scientific Computing*, 35(6):A2417–A2411, 2013.

[87] M.Barrault, Y. Maday, N. Nguyen, and A. T. Patera. An empirical interpolation method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339(9):667–672, 2004.

[88] M. Meyer and H. G. Matthies. Efficient model reduction in non-linear dynamics using the Karhunen-Loeve expansion and dual-weighted-residual methods. *Computational Mechanics*, 31:179–191, 2003.

[89] L. Mu, J. Wang, and X. Ye. A weak Galerkin generalized multiscale finite element method. *Journal of Computational and Applied Mathematics*, 305:68–81, 2016.

[90] N. C. Nguyen, A. T. Patera, and J. Peraire. A best points interpolation method for efficient approximation of parametrized functions. *International Journal for Numerical Methods in Engineering*, 73:521–543, 2008.

[91] D. S. Oh. An overlapping Schwarz algorithm for Raviart-Thomas vector fields with discontinuous coefficients. *SIAM Journal on Numerical Analysis*, 51(1):297–321, 2013.

[92] D. S. Oh, O. B. Widlund, S. Zampini, and C. R. Dohrmann. BDDC algorithms with deluxe scaling and adaptive selection of primal constraints for Raviart-Thomas vector fields. Technical report, tech. rep., Courant Institute, New York University, 2015. TR2015-978, 2016.

[93] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 6*, 6(2):559–572, 1901.

[94] B. Peherstorfer, D. Butnaru, K. Willcox, and H.-J. Bungartz. Localized discrete empirical interpolation method. *SIAM Journal on Scientific Computing*, 36(1):A168–A192, 2014.

[95] B. Peherstorfer and K. Willcox. Online adaptive model reduction for nonlinear systems via low-rank updates. *SIAM Journal of Scientific Computing*, 37(4):2123–2150, 2015.

[96] G. Pencheva and I. Yotov. Balancing domain decomposition for mortar mixed finite element methods. *Numerical Linear Algebra with Applications*, 10(1-2):159–180, 2003.

[97] M. Presho and S. Ye. Reduced-order multiscale modeling of nonlinear p-laplacian flows in high-contrast media. *Computational Geosciences*, 19:921–932, 2015.

[98] M. L. Rapun, F. Terragni, and J. M. Vega. Adaptive POD-based low-dimensional modeling supported by residual estimates. *International Journal for Numerical Methods in Engineering*, 104:844–868, 2015.

[99] M. L. Rapun and J. M. Vega. Reduced order models based on local POD plus Galerkin projection. *Journal of Computational Physics*, 229(8):3046–3063, 2010.

[100] S. S. Ravindran. Adaptive reduced-order controllers for thermal flow system using proper orthogonal decomposition. *SIAM Journal of Scientific Computing*, 23(6):1924–1942, 2002.

[101] M. Rewienski. *A trajectory piecewise linear approach to model order reduction of nonlinear dynamical systems*. PhD thesis, Massachusetts Institute of Technology, 2003.

[102] C. W. Rowley. Model reduction for fluids using balanced proper orthogonal decomposition. *International Journal of Bifurcation and Chaos*, 15(3):997–1013, 2005.

[103] C. W. Rowley, T. Colonius, and R. M. Murray. Model reduction for compressible flows using POD and Galerkin projection. *Physica D*, 189:115–129, 2004.

[104] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7:856–869, 1986.

[105] M. Sarkis. Nonstandard coarse spaces and Schwarz methods for elliptic problems with discontinuous coefficients using non-conforming elements. *Numerische Mathematik*, 77(3):383–406, 1997.

[106] N. Spillane, V. Dolean, P. Hauret, F. Nataf, C. Pechstein, and R. Scheichl. Abstract robust coarse spaces for systems of PDEs via generalized eigenproblems in the overlaps. *Numerische Mathematik*, 126(4):741–770, 2014.

[107] E. Suwartadi. *Gradient based methods for production optimization of oil reservoirs*. PhD thesis, Massachusetts Institute of Technology, 2012.

[108] R. Åđtefanescu and I. M. Navon. POD/DEIM nonlinear model order reduction of an ADI implicit shallow water equations model. *Journal of Computational Physics*, 237:95–114, 2013.

[109] A. Toselli and O. Widlund. *Domain decomposition methods: algorithms and theory*, volume 34. Springer, 2005.

[110] J. F. M. van Doren, R. Markovinovic, and J. D. Jansen. Reduced-order optimal control of water flooding using proper orthogonal decomposition. *Computational Geosciences*, 10:137–158, 2006.

[111] P. T. M. Vermeulen, A. W. Heemink, and C. B. M. T. Stroet. Reduced models for linear groundwater flow models using empirical orthogonal functions. *Advances in Water Resources*, 27:57–69, 2004.

[112] S. Volkwein and M. Hinze. Proper orthogonal decomposition surrogate models for nonlinear dynamical systems: error estimates and suboptimal control. In *Reduc-*

tion of Large-Scale Systems, P. Benner, V. Mehrmann, D. C. Sorensen (eds.), Lecture Notes in Computational Science and Engineering, volume 45, pages 261–306. 2005.

[113] M. F. Wheeler, T. Wildey, and I. Yotov. A multiscale preconditioner for stochastic mortar mixed finite elements. *Computer Methods in Applied Mechanics and Engineering*, 200(9-12):1251–1262, 2011.

[114] M. F. Wheeler, G. Xue, and I. Yotov. A multiscale mortar multipoint flux mixed finite element method. *ESAIM Mathematical Modelling and Numerical Analysis*, 46(4):759–796, 2012.

[115] X. H. Wu, Y. Efendiev, and T. Y. Hou. Analysis of upscaling absolute permeability. *Discrete and Continuous Dynamical Systems, Series B.*, 2:158–204, 2002.

[116] H. Xiao. *Multiscale mortar mixed finite element methods for flow problems in highly heterogeneous porous media*. PhD thesis, The University of Texas at Austin, 2013.

[117] Y. Yang, M. Ghasemi, E. Gildin, Y. Efendiev, and V. Calo. Fast multiscale reservoir simulations with pod-deim model reduction. *SPE Journal*, 2016.

[118] M. J. Zandvliet. *Model-based lifecycle optimization of well locations and production settings in petroleum reservoirs*. PhD dissertation, Delft University of Technology, 2008.