Abstract

In this paper, we present a global-local model reduction for fast multiscale reservoir simulations in highly heterogeneous porous media with applications to optimization and history matching. Our proposed approach identifies a low dimensional structure of the solution space. We introduce an auxiliary variable (the velocity field) in our model reduction that allows achieving a high degree of model reduction. The latter is due to the fact that the velocity field is conservative for any low-order reduced model in our framework. Because a typical global model reduction based on POD is a Galerkin finite element method, and thus it can not guarantee local mass conservation. This can be observed in numerical simulations that use finite volume based approaches. Discrete Empirical Interpolation Method (DEIM) is used to approximate the nonlinear functions of fine-grid functions in Newton iterations. This approach allows achieving the computational cost that is independent of the fine grid dimension. POD snapshots are inexpensively computed using local model reduction techniques based on Generalized Multiscale Finite Element Method (GMsFEM) which provides (1) a hierarchical approximation of snapshot vectors (2) adaptive computations by using coarse grids (3) inexpensive global POD operations in a small dimensional spaces on a coarse grid. By balancing the errors of the global and local reduced-order models, our new methodology can provide an error bound in simulations. Our numerical results, utilizing a two-phase immiscible flow, show a substantial speed-up and we compare our results to the standard POD-DEIM in finite volume setup.

Introduction

Reservoir management enables us to obtain the most favorable production scenario given the current information of the reservoir. However, proper reservoir management becomes challenging to be performed due to the intrinsic uncertainties and complexities associated with the reservoir properties (Voneiff et al., 2014). To this end, accurate results for reservoirs are obtained if fine grid discretization is induced into the model. This leads to large-scale system of nonlinear equations that needs to be solved every time step. The importance of obtaining a simpler model that can represent the physics of the full system is paramount to speed up the workflows that require several (from dozen to thousands) calls of the forward model. This is usually the case in history matching (Oliver et al., 2008; Afra et al., 2014), production optimization problems (Doren et al., 2006) and uncertainty quantifications (Jafarpour and Tarrahi, 2011).
Also, the computational time of such large-scale models become the bottleneck of fast turnarounds in the decision-making process and assimilation of real-time data into reservoir models (Gildin and Lopez, 2011; Ghasemi et al., 2012).

Over the past decade, numerous techniques have been applied in the context of porous media flow simulation to reduce the computational effort associated with the solution of the underlying coupled nonlinear partial differential equations. These techniques ranging from heuristic approach (Queipo et al., 2002; Lerlerptakdee et al., 2014), to more elegant mathematical techniques (Antoulas et al., 2001; Heijn et al., 2004), explore the idea of determining a ‘reduced complexity’ model that can approximate the full nonlinear system of equations with certain accuracy. In many cases, reduced-order modeling techniques have shown to be a viable way of mitigating computational complexity in simulation of the large-scale model, while they maintain high level of accuracy when compared with high fidelity models. In this paper, we address global-local model order reduction techniques in reservoir simulation.

In the global model reduction, one seeks to preserve the input-output behavior of the system. In recent years, many reduced-order modeling techniques have been adapted to speed up reservoir simulation and production optimization. Reduced-order modeling by projection has been used in the systems/controls-like framework, such as the balanced truncation (Heijn et al., 2004), proper orthogonal decompositions (POD) (Doren et al., 2004), the trajectory-piecewise linear (TPWL) techniques Cardoso and Durlofsky (2010), empirical interpolation methods Efendiev et al. (2013); Ghommem et al. (2013), bilinear Krylov subspace methods (Ghasemi et al., 2014) and quadratic bilinear model order reduction (Gildin and Ghasemi, 2014). In this work we focus on the POD based model order reduction technique, which performs a Galerkin projection on the space identified by important modes. Because Galerkin projection minimizes $L_2$ error for the residual over the entire domain, this approach does not yield a mass conservative scheme, in general. The mass conservation does not hold even if the solution can be exactly represented as a linear combination of the POD basis functions. To achieve a mass conservative scheme, an auxiliary variable, velocity, is introduced for which we identify basis functions.

The system for the velocity and the pressure is jointly solved by performing a projection within a mixed form of the equations. Mixed finite element methods are used in model reduction in previous findings Suwartadi (2012). In this paper, we present a POD-DEIM and to solve the coupled system at a cost that is independent of the fine grid and develop global-local model reduction. We also understand the limitations of mixed formulations when dealing with more complicated physics such as three-phase flow.

When POD is applied to a nonlinear problem, the evaluation of the nonlinear terms requires to project back to fine scale solution with similar computational cost as the original system. There are different techniques to alleviate this problem. One approach is to linearize these nonlinear functions around several known states and use these piecewise linear solutions, see (Cardoso and Durlofsky, 2010). Here, we use DEIM, where one construct another subspace for reducing the nonlinear function evaluations (Chaturantabut and Sorensen, 2010). We apply the POD-DEIM for model reduction in porous media flow, and we show that better speed-ups can be achieved without compromising the solution accuracy.

In this paper, we also apply generalized multi-scale method that has been developed to account for different scales on the discretization of the underlying partial differential equations (Efendiev and Hou, 2009). In Efendiev et al. (2012), the authors implemented a global-local multiscale model reduction framework that balances out the number of multiscale basis per node for a local reduction and the number of basis applied to the global model reduction implemented via the balanced truncation method.

**An Incompressible Two-Phase Flow Model**

In this section, we summarize the underlying partial differential equations related to porous media flow simulation. In particular, we briefly discuss two-phase oil-water systems. We consider two-phase flow in a reservoir domain (denoted by $\Omega$) under the assumption that the displacement is dominated by viscous
effects; i.e., we neglect the effects of gravity, compressibility, and capillary pressure. The two phases are water and oil, and they are assumed to be immiscible. We write Darcy’s law for each phase as follows:

\[ u_l = -\frac{k_{rl}(s)}{\mu_l} K \cdot \nabla p, \]  

(1)

where \( u_l \) is the phase velocity, \( K \) is the permeability tensor, \( k_{rl} \) is the relative permeability to phase \( l \) (\( l = o, w \)), \( s \) is the water saturation (we use \( s \) instead of \( s_w \) for simplicity) and \( p \) is pressure. Throughout the paper, we will assume that a single set of relative permeability curves is used. Combining Darcy’s law with a statement of conservation of mass allows us to express the governing equations in terms of the so-called pressure and saturation equations:

\[-\nabla \cdot (\lambda(s) K \nabla p) = q_w + q_o \quad \text{in} \quad \Omega, \]

(2)

\[ \frac{\partial s}{\partial t} + \nabla \cdot (f_w(s) u) = \frac{q_w}{\rho_w} \quad \text{in} \quad \Omega. \]

(3)

\[ u \cdot n = 0 \quad \text{on} \quad \partial \Omega : \text{no flow at boundary} \]

(4)

\[ s(t = 0) = s_0 \quad \text{in} \quad \Omega : \text{initial known saturation.} \]

(5)

where \( \phi \) is the porosity, \( \lambda \) is the total mobility defined as,

\[ \lambda(s) = \lambda_w(s) + \lambda_o(s) = \frac{k_{rw}(s)}{\mu_w} + \frac{k_{ro}(s)}{\mu_o}, \]

(6)

\[ f(s) \] is the flux function,

\[ f_w(s) = \frac{\lambda_w(s)}{\lambda(s)} = \frac{k_{rw}(s)}{k_{ro}(s) + \frac{\mu_o}{\mu_w} k_{rw}(s)}, \]

(7)

and \( u = u_w + u_o \) is the total velocity. Moreover, \( q_w \) and \( q_o \) are volumetric source terms for water and oil. The above descriptions are referred to as the fine model of the two-phase flow problem. The Eqs. (2) and (3) are nearly elliptic, see (Aarnes et al., 2007). Thus, we follow the sequential formulation; at each time step one solves for pressure first and then use the result to solve for saturation.

**Pressure Equation: Mixed Formulation**

We solve the pressure equation with mixed method (Fortin and Brezzi, 1991). By introducing the flux variable \( u = -\lambda(s) K \nabla p \), we get the following first-order system

\[(\lambda(s) K)^{-1} u + \nabla p = 0, \quad \nabla \cdot u = q. \]

(8)

The mixed formulation of Eq. (8) is to find \((p, v) \in Q \in V \) such that,

\[((\lambda K)^{-1} u, v) + (\nabla p, v) = 0 \quad \forall v \in V \]

\[(\nabla \cdot v, l) = (q, l) \quad \forall l \in Q \]

(9)

where \((\cdot, \cdot)\) is the \(L^2\) inner product and \(Q, V\) are finite dimensional spaces for pressure and velocity solutions, which typically consist of low order piecewise polynomials. For further discussion one can refer to (Aarnes et al., 2007).

After spatial discretization of the problem in (9), the system of equations takes the form,

\[
\begin{pmatrix}
B(\lambda(s)) & -C^T \\
C & 0
\end{pmatrix}
\begin{pmatrix}
u \\
p
\end{pmatrix}
= 
\begin{pmatrix}
0 \\
g
\end{pmatrix}
\]

(10)

where
\[ b_{ij} = \int (\lambda(s)k)^{-1} \psi_i \psi_j, \quad c_{ij} = \int \nabla \cdot (\psi_i \phi_j) \]

and \( \phi_i, \psi_i \in V, \phi_i \in Q, \) and \( f \) results from the sink/source terms. Here, we keep the same notations for discrete and continuous variables for simplicity.

### Saturation Equation

Generally, an implicit time (backward Euler) discretization will be followed to solve for saturation profile, while a mass conservative finite volume is used for the spatial derivative discretization. Consider a cell \( \Omega_i \) with edges \( \gamma_{ij} \) and associated normal vectors \( n_{ij} \) pointing out of \( \Omega_i \), the saturation Eq. (3) will be discretized as,

\[ s^{n+1}_i = s^n_i + \frac{\Delta t}{|\Omega_i|} \left( q^+ - \sum_j F_{ij}(s^{n+1})u_{ij} + f_w(s^{n+1})q^- \right). \]  

where \( s^n_i \) is the cell-average of the water saturation at time \( t = t_n \), \( q^+ = \max(q_i, 0) \) and \( q^- = \min(q_i, 0) \), \( u_{ij} \) is the total flux (for oil and water) over the edge \( \gamma_{ij} \) between the two adjacent cells and \( F_{ij} \) is a numerical approximation of the flux over edge \( \gamma_{ij} \) defined as,

\[ F_{ij} \approx \int_{\gamma_{ij}} (f_{ij}(s)u_{ij}) \cdot n_{ij} \, dv. \]  

There are different schemes to evaluate the integrand in (13). It is common to use first order approximation, known as upstream weighting, defined as,

\[ f_{ij}(s) = \begin{cases} f_w(s_i) & \text{if } u_{ij} \cdot n_{ij} \geq 0; \\ f_w(s_j) & \text{if } u_{ij} \cdot n_{ij} < 0. \end{cases} \]  

Our solution approach is sequential. First, the pressure equation is solved on a fine grid. Next, the saturation equation is solved implicitly. Note that the pressure and velocity equations in (10) is linear system, whereas the saturation equation in (12) is nonlinear system and can be solved for \( s^{n+1} \) by iterative methods such as Newton-Raphson efficiently. The residual as a function of saturation is defined as follows,

\[ R(s) \triangleq s - s^n - A(u) f_w(s) - \frac{\Delta t}{|\Omega|} q^+, \]

where \( s^n \) is the vector of cell-saturations at time instance \( t_n \) and \( f_w(s) \) is the vector of fractional flow values including \( f_w(s_i) \), \( A \) is a matrix implementing \( \frac{\partial}{\partial s} f_w(s)q^- - \nabla \cdot (f_w(s)u) \) on a cell by cell basis. The derivative of Residual in (15) is called Jacobian defined as,

\[ G(s) \triangleq \frac{\partial R}{\partial s} = 1 - A f'_w(s). \]

where \( f'_w \) is the explicit derivative of fractional function with respect to saturation. At each iteration of Newton method, the current state is updated as,

\[ s^{k+1} = s^k - \alpha G(s^k)^{-1} R(s^k), \]

and the iterations are stopped at each time step when the norm of the update or Jacobian are smaller than a tolerance. In the next section, our goal is to present a reduced-order model for both equations and reduce the computation runtime.

### Global Model Reduction via POD-DEIM

Global model reduction for a mixed formulation will be performed using POD and DEIM. These approaches are widely discussed in the literature and we briefly mention some basic concepts here.
POD is typically performed on the state solutions. Assume the solution states are saved at each time step and put next to each other as,

\[ S_x = [x_1, x_2, \ldots, x_{ns}] \in \mathbb{R}^{N \times ns} \tag{18} \]

where \( ns \) is the total number of snapshots, \( x_i \) can be pressure, saturation, etc at time instance \( i \) that are vectorized to be stacked in a matrix, \( S \) is usually referred to as snapshots matrix, \( N \) is the total number of grid-blocks. The goal is to find a set of basis to re-parametrize the state of the system. In POD, one finds an orthonormal \( \{ \phi_i \}_{i=1}^r \) basis such that

\[ \text{min}_{\phi_i} \sum_{j}^{ns} \left\| x_j - \sum_{i}^{r} (x_j^T \phi_i) \phi_i \right\| \tag{19} \]

It can shown (see (Volkwein and Hinze, 2005)) that the solution of this minimization problem is found by applying \( \text{svd} \) to the snapshots matrix,

\[ S_x = V \Lambda W^T \tag{20} \]

where matrices \( V \) and \( W \) are left and right projection matrices and \( \Lambda \) is a matrix containing the singular values. The projection matrix \( \phi \in \mathbb{R}^{N \times r} \) can be obtained by selecting the first \( r \) columns of the matrix \( V \).

We denote the snapshots of the pressure, the velocity, and saturation by \( S_p, S_u, S_s \), respectively. After applying a singular value decomposition on these matrices, we select the projection matrix by indicating the fraction of total energy to be captured. The fractional energy is defined as,

\[ E = \frac{\sum_{i=1}^{r} \sigma_i}{\sum_{i=1}^{ns} \sigma_i} \tag{21} \]

where \( \sigma_i \) is the \( i \)th diagonal element of matrix \( \Lambda \). The number of basis selected such that the captured energy is \( 0.9 \leq E \leq 1 \) (see discussions in (Volkwein and Hinze, 2005) for selecting modes). After obtaining the projection matrix, the pressure, velocity and saturation are projected into reduced subspace as,

\[ p = \Phi_p p_r, \quad u = \Phi_u u_r, \quad s = \Phi_s s_r \tag{22} \]

POD-Galerkin reduces computational cost moderately for systems with general nonlinear terms. To reduce the complexity of the nonlinear term, we apply DEIM, which constructs a separate subspace for nonlinear terms, selects interpolation points via a greedy strategy, and then combines the interpolation and projection to approximate the nonlinear terms in the subspace.

We briefly review the DEIM as presented in Chaturantabut and Sorensen (2010). Let \( f(\tau) \in \mathbb{R}^N \) denotes a nonlinear function where \( \tau \) refers to time \( t \) or any control parameter \( \mu \). We approximate the function \( f \) by projecting it into a subspace spanned by the basis functions \( U = (u_1, \ldots, u_m) \in \mathbb{R}^{N \times M} \) as

\[ f(\tau) \approx U c(\tau) \tag{23} \]

To compute the coefficient vector \( c \), we define a selection matrix

\[ P = [e_{p1}, \ldots, e_{pm}] \in \mathbb{R}^{N \times m} \]

where \( u_{[\tau]} = [0, \ldots, 0, 1, 0, \ldots, 0]^T \in \mathbb{R}^N \) is the \( \delta_{ij} \)th column of the identity matrix \( I_n \in \mathbb{R}^{N \times N} \) for \( i = 1, \ldots, m \). Multiplying Equation (23) by \( P^T \) and assuming that the matrix \( P^T U \) is nonsingular, we obtain

\[ f(\tau) \approx U c(\tau) = U (P^T U)^{-1} P^T f(\tau). \tag{24} \]
DEIM approximate the nonlinear function $f$ in (24) with a single subspace of dimension $m$ spanned by columns of projection matrix $U$ and will reduce the computational runtime. To summarize, the approximation of the nonlinear function $f(\tau)$, as given by (24), requires the followings:

- Computation of the projection basis $U$
- Identification of the indices $\{\varphi_1, \cdots, \varphi_m\}$

To determine the projection basis $U$, we assemble the function evaluations in a matrix $S_f \in \mathbb{R}^{N \times n_s}$ and employ the singular value decomposition (SVD) to compute the proper orthogonal decomposition (POD) modes. These modes are used as the projection basis in the approximation given by (23). As for the interpolation indices $\{\varphi_1, \cdots, \varphi_m\}$ they are selected using greedy algorithm as given in (Chaturantabut and Sorensen, 2010).

**Global Model Reduction for Pressure Equation**

After the training run and saving $S_p$, $S_u$ and $S_s$; using SVD yields POD basis matrices $\Phi_u \in \mathbb{R}^{N_u \times r_u}$ and $\Phi_p \in \mathbb{R}^{N_p \times r_p}$ and $\Phi_s \in \mathbb{R}^{N_s \times r_s}$ with $r_u << N_u$ and $r_p << N$ and $r_s << N$. Through Galerkin projection onto the subspace spanned by POD basis and substituting the new defined variables in (22) into (10), one gets the reduced pressure equation as,

$$
\begin{bmatrix}
\Phi^T \mathbf{B} \Phi_u & - \Phi^T \Phi u \Phi p \\
\Phi^T \Phi u & 0
\end{bmatrix}
\begin{bmatrix}
u \\
p_r
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
\Phi p \Phi s
\end{bmatrix}.
$$

Note that the matrix $B$ is of the form (11), which is a nonlinear function of saturation. Our goal is to solve the problem in a reduced space through entire simulation time, and avoid projecting back to fine scale solution. Thus, one need to use an approximation for the mass matrix. Here, we apply DEIM method to approximate the nonlinear function as explained in the previous section. The nonlinear function $f_\lambda(s) = (\lambda(s)K)^{-1}$ is projected onto a smaller subspace, i.e.

$$
\begin{equation}
\begin{aligned}
f_\lambda(s) & \approx \sum_{l=1}^{m} c_l U_l = U_\lambda (P_\lambda U_\lambda)^{-1} P_\lambda f_\lambda(\Phi_\lambda s_r)
\end{aligned}
\end{equation}
$$

where $m$ is the number of DEIM basis for approximating this nonlinear function, and $s_r$ is the saturation the reduced space (of size $r_s << N_s$). Substituting (25) in the definition of matrix $B$ in (11), we get

$$
\int_\Omega f_\lambda(s) \psi_s dx = \sum_{l=1}^{m} c_l \int_\Omega U_l \psi_s dx.
$$

Most of the matrices in Eqs (25) and (26), e.g. $U_\lambda (P_\lambda U_\lambda)^{-1}$, $P_\lambda \Phi_\lambda$ and $M_l = \int_\Omega U_l \psi_s dx$, $l = 1, \ldots, m$ can be precomputed in the offline stage. In the online stage, we only need to find the coefficients and do the summation $M = \sum_{l=1}^{m} c_l M_l$

**Global Model Reduction for Saturation Equation**

We will employ POD-DEIM to the whole system Eq (3). By applying POD to the saturation equation, we get

$$
\begin{equation}
\begin{aligned}
R_s(s_r^{n+1}) & \triangleq s_r^{n+1} - s_r^n - (\Phi_s)^T \left( A f_w(\Phi_s, s_r^{n+1}) + \frac{\Delta t}{\Omega} q^+ \right),
\end{aligned}
\end{equation}
$$

Next, apply DEIM to the flux function $f_w(s)$ to project it to a lower dimensional subspace and reduced the complexity, i.e.

$$
\begin{equation}
\begin{aligned}
f_w(s) & \approx U_w (P_w U_w)^{-1} P_w f_w(\Phi_\lambda s_r).
\end{aligned}
\end{equation}
$$

Substituting Eq (28) into (27) results in,
\[
\Phi^T f_w(s) = \Phi^T A U_w \left( P_f^T U_w \right)^{-1} f_w \left( P_f^T \Phi_S r \right) \tag{29}
\]

Here \( \Phi^T A U_w \left( P_f^T U_w \right)^{-1} \in \mathbb{R}^{r_f \times m_f} \), and \( P_f^T \Phi_S \in \mathbb{R}^{m_f \times r_f} \) can be precomputed; \( r_f \) is the number of POD basis for saturation, \( m_f \) is the number of DEIM points for flux. One can also approximate the Jacobian as follows,

\[
G_r(s) \triangleq \frac{\partial R_r}{\partial s} - I_{r_f} - \Phi^T A U_w \left( P_f^T U_w \right)^{-1} f''_w \left( P_f^T \Phi_S r \right) \tag{30}
\]

Now Newton-Raphson iteration is used to solve for the reduced saturation at each time instance as following,

\[
s^{k+1}_r = s^k_r - \alpha G_r(s^k) R_r(s^k), \tag{31}
\]

Thus, at each time step the saturation will be solved in the reduced space without the need to project back to the fine scale as shown in the POD-DEIM flowchart in Fig. (1).

Note that the matrix \( A \) is derived from upwind indices and depends on the velocity. Thus, one needs to address the updating of upwinding matrix during simulation. One approach is to project back to fine scale solution at each time step after the pressure equation is solved, and update the upwinding matrix based on new velocity solution. Other alternative method is to update the upwinding matrix based on reduced velocity vector, i.e. finding regions in the reduced subspace where the velocity direction has changed using the reduced velocity and only updates those regions. These issues will be investigated in our future research.

**Mass conservation in POD with finite volume discretization**

Because POD is a Galerkin projection, it will not necessarily honor mass conservation property. If we assume that the snapshots \( \eta_i \) are obtained with saturation fields \( S_i \), then the mass conservation holds if one of the snapshots, \( \eta_i \), coincides with the current solution; however, if the projection error is zero, i.e., the solution at the current time can be exactly represented by pressure snapshots, \( P = \Sigma_i d_i \eta_i \), for some \( d_i \), it does not guarantee that the current solution will be mass conservative.

One way to achieve a mass conservation is using an auxiliary variable, 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. This allows us to achieve higher degree of reduction. In the next example, we demonstrate how mass conservation can be violated.

**Example**

In this example, the reservoir model is a two-phase flow (oil-water) model under the water flooding recovery process with the structure of a 5-spot. The reservoir model is discretized using Cartesian grid of size 10ft × 10ft × 10ft. Overall the reservoir model has 45 × 45 × 1 = 2025 active cells. The permeability of the reservoir is 10 (md) homogeneous and the porosity is 0.2. The relative permeability curves is quadratic.

All the producers have constant bottom hole pressure at 2500 (psia). For training input, the injector bottom hole pressure is 3750 (psia). The reservoir model was simulated for 1000 days and the snapshots of the nonlinear fractional function was saved every 10 days. After applying \( \text{svd} \) on the snapshot matrices,
the pressure and saturation basis are obtained. We selected 13 pressure basis to preserve 0.9999 energy and 13 basis to preserve 0.99 saturation energy. These basis used to construct the projection matrices to project fine scale states to the reduced subspaces. When the reduced model was run with the same exact input and boundary conditions, the error was small and the results were close to fine scale solution. However, the reduced models are used in frameworks with different inputs other than training one. Thus, we perturbed the bottom hole pressure of the injector as $\pm 5\%$, as shown in Fig. 2. Although, this is a small perturbation and it is only in one of the input variables, the results of reduced model is far from high fidelity solution as it is shown in Fig. 3. All the produces have the same water cut as shown in Fig. 3(a), due

![Figure 2—Training and test (perturbed) bottom hole pressure of the injector](image)

![Figure 3—POD model reduction on finite volume formulation](image)

![Figure 4—Evaluation of mass conservation at final time](image)
to the symmetry in the problem. Note that it the water cut is very different towards end of simulation, because the mass conservation is violated in most of the gridblocks, as shown in Fig. 4.

We also applied POD on a mixed finite element formulation of this example. After running the same training 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 result in small error for exact input, but also replicates a very similar results to high fidelity model for test input as shown in Fig. 5. The saturation error is smaller than 2% for most of simulation time.

This example showed 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.

Local Model Reduction

The main computational cost in global POD-DEIM approach is due to computations of snapshots as well as manipulations with these snapshot vectors, such as setting up eigenvalue problems and so on. To reduce this cost in a systematic fashion, one can use local model reduction techniques. One of the local model order reduction consists of flow-based upscaling (Durlofsky, 1998) that computes upscaled permeability fields. In this paper, we will use a systematic coarse-grid models where multiscale basis functions are constructed locally. We will follow Generalized Multiscale Finite Element Concept as introduced in (Efendiev and Hou, 2009). In (Efendiev et al., 2012) where the main idea is to compute a few basis functions in each coarse block to approximate the solution.

These basis functions are computed once and allows fast computations of snapshot vectors and their low-dimensional approximations. By adding more basis functions, one can converge to the fine-scale solution.

Multiscale methods can be considered as model reduction techniques that 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 are used to solve these forward problems on a coarse grid multiple times and thus can provide a substantial speed-up.

The flow problems considered here have heterogenous permeability fields, which usually require a large number of degrees of freedom to capture all the multiple scales and features. Here we apply the
recently developed mixed GMsFEM (Chung et al., 2014), 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 pressure 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. 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 idea of the offline space is that it gives a good solution approximation with fewer basis functions.

As a result of offline computations, one can compute several multiscale basis functions per each coarse edge. Let \( \mathcal{T}^H \) be a usual conforming partition of the domain \( \Sigma \) into finite elements called coarse-grid blocks, and \( H \) is the coarse mesh size. Then each coarse grid block is divided into a connected union of fine-grid blocks, which are conforming across coarse-grid faces. This fine grid partition is denoted by \( \mathcal{T}^H \). Denote \( E^H = \bigcup_{i=1}^{N_e} \{ E_i \} \), where \( N_e \) is the number of coarse faces, and \( \mathcal{O}_i = \bigcup \{ K_j \in \mathcal{T}^H : E_i \in \partial K_j \} \) as the coarse neighborhood corresponding to the coarse face \( E_i \). Like in (Chung et al., 2014), we construct multiscale basis functions for each face \( i, 1 \leq i \leq N_e \), denoted by \( \psi_{k, i}^{\text{off}} \), \( 1 \leq k \leq l_i \), where \( l_i \) is the number of basis chosen for face \( i \). Note that the support of these basis functions is \( \partial E_i \).

The offline space is then constructed as:

\[
V_{\text{off}} = \text{span}\{ \psi_{k, i}^{\text{off}} : 1 \leq k \leq l_i, 1 \leq i \leq N_e \}.
\]

Using single-index notation:

\[
V_{\text{off}} = \text{span}\{ \psi_k^{\text{off}} : 1 \leq k \leq l_{\text{off}} \},
\]

where \( l_{\text{off}} \) is the number of velocity basis from the offline phase. Furthermore, we define

\[
R_{\text{off}} = [\psi_1^{\text{off}}, \ldots, \psi_{l_{\text{off}}}^{\text{off}}],
\]

which maps from the offline space to the fine space, where \( \psi_k^{\text{off}} \) is a vector containing the coefficients in the expansion of \( \psi_{k, i}^{\text{off}} \) in the fine-grid basis functions.

Let \( V_H = V_{\text{off}} \) and \( Q_{H} \) be the space of piecewise constant functions with respect to the coarse grid. The online stage is to find \( (p_H, u_H) \in Q_H \times V_H \) such that

\[
\int_{\Omega} (\lambda(\sigma) K)^{-1} v_H \cdot u_H \ dx - \int_{\Omega} p_H \nabla \cdot v_H \ dx = 0 \tag{32}
\]

\[
\int_{\Omega} \nabla \cdot u_H z_H \ dx = \int_{\Omega} q z_H \ dx, \tag{33}
\]

for all \( V_H \in V_H \) and \( Z_H \in Q_H \). Its matrix form is

\[
\begin{bmatrix}
R_0^T BR_{\text{off}} & -R_0^T C^T G_H \\
G_H C R_{\text{off}} & 0
\end{bmatrix}
\begin{bmatrix}
v_r \\
p_r
\end{bmatrix} =
\begin{bmatrix}
0 \\
G_H \mathcal{R}
\end{bmatrix}, \tag{34}
\]

where \( G_H \) is the restriction operator from \( Q_H \) into \( Q_{h} \), and \( v_r, p_r \) are vectors of coefficients in the expansions of the solutions \( v_{H}, p_{H} \) in the spaces \( V_{H} \) and \( Q_H \).

**Numerical Examples**

In this section, we will give some numerical examples to demonstrate the performance of the methods for solving the flow problem. The first 2D examples are dimensionless, the computational domain considered is \( \Omega = (0,1)^2 \); the coarse grid \( \mathcal{T}^H \) and the fine grid \( \mathcal{T}^h \) are \( N \times N \) and \( n \times n \) uniform meshes with \( N = 22, n = 220 \) respectively; the permeability field is shown in Fig. 6; an injector is placed on the top-left and a producer is placed on the bottom-right; the rate is 2 for training and 4 for test simulation.
respectively; end of simulation time is 1000. In all the examples, relative error for saturation calculated at \( t_i \) is computed as

\[
s_{err}(t_i) = \frac{\|s_{red}(t_i) - s_{ref}(t_i)\|_{L^2(\Omega)}}{\|s_{ref}(t_i)\|_{L^2(\Omega)}},
\]  

(35)

where \( s_{red}(t_i) \) is the solution from model reduction and \( s_{ref}(t_i) \) is the reference solution which will be specified in the examples.

**POD-DEIM Model Reduction**

During the training simulation, the Eq. (3) are solved on the fine grid. In Table 2(a), the relative \( L^2 \) errors of saturation at end of simulation time for different number of basis for each variable and total number of Newton iterations are given. The fine solution is used as reference solution. The total number of Newton iterations for the fine solution is 1299. From Table 2(a), we see that with 2 basis for velocity, 2 for pressure and 8 for saturation, 4 DEIM points for \( f_x \) and at least 9 DEIM points for the flux function, we can get 2% relative saturation error. The number of Newton iterations is much smaller than the reference solution.

**Global-Local Model Reduction**

During the training simulation, the mixed GMsFEM is employed to solve the pressure equation. On every inner coarse edge, 5 basis are selected. The dimension for velocity space is approximately 5% of the fine scale velocity space and 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 2(b), 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. From Table 2(b), we see that with 2 basis for velocity, 2 for pressure and 8 for saturation, 4 DEIM points for $f$ 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 $O(10^2)$ compared to the fine solution.

**Case Study**

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, and it is assumed that all of them are perforated only at the bottom layer. The reservoir is SPE10 comparative model (Christie and Blunt, 2001) (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 $20ft \times 10ft \times 2ft$. 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 Figs. 7a and 7b, respectively. We assumed a constant porosity of 0.2 for entire model.

For the training schedule, the producers are controlled by bottom hole pressure and the injector by injection rate as shown in Fig. 8. Note that this amount of injection was selected to make sure at least one pore volume will be injected throughout simulation time (1000 days). The initial water saturation and pressure are assumed to be 0.0 and 2500 psia respectively.
In order to apply the POD-DEIM methods, we simulated the reservoir for 1000 days and saved the snapshots of pressures, velocity, water saturations and the nonlinear fractional function every 10 days. Thus, we have 100 snapshots for each variable. 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. The singular values of the snapshot matrices are shown in Figs. 9a and 9b. 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 of it to capture the most of energy and have small error.

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. 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 (35) with fine scale solution as the reference solution. The average error is less than 5% for most of the simulation time as shown in Fig. 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 simulation is shown in Fig. 10a and compared with fine scale model. Figs. 11b and 11a show the spatial relative error at the final time in the pressure and saturation, respectively. The error in pressure is $O(10^{-3})$, and in saturation $O(10^{-1})$. Note that the error is usually larger in the cells around injector due high dynamical fluid flow.

We run the reduced model with a new test schedule as shown in Fig. 12, to make sure that the POD-DEIM model reduction is robust to input variation. This schedule is obtained by $\pm 20\%$ random

In order to apply the POD-DEIM methods, we simulated the reservoir for 1000 days and saved the snapshots of pressures, velocity, water saturations and the nonlinear fractional function every 10 days. Thus, we have 100 snapshots for each variable. 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. The singular values of the snapshot matrices are shown in Figs. 9a and 9b. 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 of it to capture the most of energy and have small error.

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. 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 (35) with fine scale solution as the reference solution. The average error is less than 5% for most of the simulation time as shown in Fig. 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 simulation is shown in Fig. 10a and compared with fine scale model. Figs. 11b and 11a show the spatial relative error at the final time in the pressure and saturation, respectively. The error in pressure is $O(10^{-3})$, and in saturation $O(10^{-1})$. Note that the error is usually larger in the cells around injector due high dynamical fluid flow.

We run the reduced model with a new test schedule as shown in Fig. 12, 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 basis of the reduced model are not updated and we used the same basis 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 Fig. 13a and compared with fine scale model. Figs. 14b and 14a show the spatial relative error at the final time in the pressure and saturation, respectively. The error in pressure is still $O(10^{-3})$, and in saturation $O(10^{-1})$. Note that even though the error is around 0.2 in
some of the cells around injector due high dynamical fluid flow, the average saturation error is smaller than 5% for most of the simulation time as shown in Fig. 13b.

Conclusions
In this paper, we present a global-local approach for simulations of two-phase flow and transport. Our approach uses POD to setup the whole system on a reduced dimensional space. In addition, DEIM is used to approximate nonlinear responses so that the resulting system is independent of the fine grid. 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 formulation (cf. (Aarnes et al., 2006, 2008)). In our proposed technique, the resulting coupled flow and transport system is solved on a reduced dimensional space using nonlinear approximations of mobility and fractional flow with DEIM. 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.

Nomenclature

\( K \): permeability, \([ft^2]\)
\( \mu \): viscosity, \([cp]\)  
\( \phi \): porosity, \([-]\)  
\( q \): volumetric flow rate, \([bbl/day]\)  
\( t \): time \([day]\)  
\( p \): pressure \([psi]\)  
\( s \): water saturation  
\( u \): phase velocity \([ft/day]\)  

Subscripts  
\( o \): oil  
\( w \): water  
\( ro \): relative to oil  
\( rw \): relative to water  

Subscripts  
\( n \): \( n \)th time step  
\( k \): \( k \)th nonlinear iteration

References  
Afra, S., Gildin, E., and Tarrahi, M., 2014. Heterogeneous reservoir characterization using efficient parameterization through higher order svd (hosvd). In American Control Conference. IEEE, Portland, Oregon, USA.  


