INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2009; **77**:1322–1350
Published online 29 August 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/nme.2453

Development and application of reduced-order modeling procedures for subsurface flow simulation

M. A. Cardoso[1,*,†], L. J. Durlofsky[1] and P. Sarma[2]

[1]*Department of Energy Resources Engineering, Stanford University, Stanford, CA 94305, U.S.A.*
[2]*Chevron Energy Technology Company, P.O. Box 6019, San Ramon, CA 94583, U.S.A.*

SUMMARY

The optimization of subsurface flow processes is important for many applications, including oil field operations and the geological storage of carbon dioxide. These optimizations are very demanding computationally due to the large number of flow simulations that must be performed and the typically large dimension of the simulation models. In this work, reduced-order modeling (ROM) techniques are applied to reduce the simulation time of complex large-scale subsurface flow models. The procedures all entail proper orthogonal decomposition (POD), in which a high-fidelity training simulation is run, solution snapshots are stored, and an eigen-decomposition (SVD) is performed on the resulting data matrix. Additional recently developed ROM techniques are also implemented, including a snapshot clustering procedure and a missing point estimation technique to eliminate rows from the POD basis matrix. The implementation of the ROM procedures into a general-purpose research simulator is described. Extensive flow simulations involving water injection into a geologically complex 3D oil reservoir model containing 60 000 grid blocks are presented. The various ROM techniques are assessed in terms of their ability to reproduce high-fidelity simulation results for different well schedules and also in terms of the computational speedups they provide. The numerical solutions demonstrate that the ROM procedures can accurately reproduce the reference simulations and can provide speedups of up to an order of magnitude when compared with a high-fidelity model simulated using an optimized solver. Copyright © 2008 John Wiley & Sons, Ltd.

Received 3 January 2008; Revised 2 June 2008; Accepted 22 July 2008

KEY WORDS: reduced-order model; proper orthogonal decomposition; reservoir simulation; production optimization; missing point estimation

1. INTRODUCTION

Computational optimization procedures are now being actively developed for a variety of applications involving subsurface flow. Our particular interest here is in the optimization of oil production,

*Correspondence to: M. A. Cardoso, Department of Energy Resources Engineering, Stanford University, Stanford, CA 94305, U.S.A.
†E-mail: mcardoso@stanford.edu

though the relevant computational techniques are equally valid for optimizing aquifer remediation operations, minimizing saltwater intrusion into fresh water aquifers, and optimizing the geological sequestration of carbon dioxide. For these and related applications, the number of flow simulations that must be performed during the optimization can be very large. Depending on the optimization algorithm employed, this number can be tens or hundreds (with gradient-based methods) or even thousands (using direct search or stochastic procedures such as genetic algorithms). For practical applications, flow simulation models generally contain between 10^4 and 10^6 cells. These models typically require two or more unknowns per grid block and must be integrated in time. Thus there is a significant need to reduce the computational requirements for the flow simulation model.

Reduced-order modeling (ROM) procedures [1–4] have been studied extensively in recent years. These techniques represent a means for achieving reductions in simulation time and have been used for the optimization of complex large-scale systems in several areas. For example, ROMs have been applied for applications involving non-linear circuit design [5], tubular chemical reactors [6], large-scale upper ocean circulation models [7], aerodynamics [8], wind turbine evaluations [9], and control of thermally coupled flow models [10]. At the heart of ROM is the assumption that the solution of the forward model can be accomplished using many fewer degrees of freedom. One of the widely used ROM techniques for non-linear systems is the proper orthogonal decomposition (POD) approach, which is the method applied in this work.

POD was first proposed by Lumley [1] to identify coherent structures in dynamical systems. In the POD procedure the relevant states of a model are characterized by a set of orthonormal basis functions. These basis functions are generated by the eigen-decomposition (or singular value decomposition, SVD) of a covariance matrix constructed from a set of computed solutions referred to as 'snapshots'. POD allows the representation of the state space of the forward model by the first few (most relevant) basis functions. A ROM is obtained by projecting the original governing equations onto the POD basis functions. This enables a significant reduction in the number of unknowns that must be computed.

There have been several applications of ROM to subsurface flow problems. Vermeulen *et al.* [11] and Vermeulen and Heemink [12, 13] considered groundwater flow involving a single (water) component. For a heterogeneous model with 33 000 active grid blocks, Vermeulen *et al.* [11] reported speedups between 70 and 625, depending on the method used to compute the time-varying coefficients.

ROM for non-linear two-phase (oil–water) flow has also been addressed [14–16]. For example, van Doren *et al.* [15] developed an optimal control reduced-order method for water flooding. They successfully applied their procedure to a heterogeneous 2D model containing 2025 grid blocks, though the speedups achieved were modest. Markovinović *et al.* [16] proposed the use of ROMs to accelerate the iterative linear solver used for the full (high-fidelity) model. Thus, in their procedure, the ROM was used in conjunction with the full model. Markovinović *et al.* considered complex models (e.g. a 3D model containing 93 500 grid blocks) and achieved up to a factor of three in speedup. This method, however, requires that the full system be simulated along with the ROM and it also requires that the reduced-order basis be updated during the course of the simulation. Further, it is to be expected that the impact of the ROM (and thus the degree of speedup) would be less if a highly optimized linear solver was used for the high-fidelity problem.

The developments cited above are promising in that they demonstrate the application of POD procedures to porous media flow problems. Previous work does not, however, demonstrate the application of ROM techniques for practical non-linear multiphase cases (involving highly heterogeneous geological characterizations containing, e.g. 10^4–10^5 grid blocks) in which reduced-order models can be used in place of full high-fidelity reservoir flow simulations. In addition, there does not appear to have been any implementations of ROM procedures into general-purpose subsurface flow simulators.

The objective of our work is to further the development and application of POD methods for subsurface flow problems. Specifically, by incorporating new ROM procedures developed within other application areas, we aim to apply these approaches to more practical reservoir simulation models and to achieve better computational performance. In addition, our developments are incorporated into a general-purpose flow simulator [17, 18], so they can be used for a wide variety of flow problems.

Two recent procedures are applied in this work that have not been used previously within the context of subsurface flow modeling. These are (1) a clustering technique to optimize the snapshots for the eigen-decomposition problem [19] and (2) a missing point estimation (MPE) procedure to reduce the dimension of the POD basis vectors [20]. The clustering technique is applied to the set of snapshots to achieve a more regular distribution of states. This acts to improve the accuracy of the resulting POD basis matrix. The MPE procedure enables the determination of the POD coefficients using information from only a selected number of grid blocks. We will show that the use of these techniques leads to improvements in the efficiency of the resulting ROM representation.

This paper proceeds as follows. In Section 2 we describe the governing equations, discretization, and the POD procedure for modeling subsurface flow. This includes a description of the clustering and MPE procedures and a description of the implementation of POD into an existing general-purpose simulator. Then, in Section 3, we present results for a number of cases involving a simulation model of a geologically complex reservoir containing 60 000 grid blocks. Different specifications for the time-varying bottom hole pressures (BHPs) of the production wells are considered. These results demonstrate the robustness and computational speedup of the POD techniques for realistic cases. Additional findings and a discussion of some of the limitations of the POD procedures are presented in Section 4.

2. MODELING PROCEDURE

In this section we first define the flow model and then present the governing equations and the discretized system. We then review the POD procedure and the resulting ROM. The clustering and MPE techniques are then described, as is the implementation of the ROM into our general-purpose research simulator.

2.1. Reservoir simulation model

Oil reservoir simulation involves the solution of equations governing the flow of reservoir fluids (oil, gas and water) through porous subsurface formations. More general formulations track individual components (e.g. methane, ethane, etc.) and/or include thermal effects. See [21] for more discussion. For simplicity, here we consider oil–water flows in the absence of gravity. The simulator into which our ROM is implemented is quite general, so more complex systems could be considered with relatively slight modifications.

Oil–water systems, with gravity neglected, are described by the following mass balance equations:

$$\frac{\partial}{\partial t}(\phi \rho_j s_j) + \nabla \cdot (\rho_j \mathbf{u}_j) + m_j = 0 \tag{1}$$

Here j, with $j = \mathrm{w}$ for water and $j = \mathrm{o}$ for oil, designates phase or component (they can be used interchangeably in this case because there is no mass transfer between phases). Equation (1) is written for each component. In these equations, t is time, ϕ is porosity (void fraction of the rock), ρ_j is the phase density, s_j is the phase saturation (volume fraction) and m_j is the source/sink term. The Darcy (or superficial) velocity of phase j, designated $\mathbf{u}_j$, is given by

$$\mathbf{u}_j = -\frac{k_{\mathrm{r}j}(s_j)}{\mu_j} \mathbf{k} \nabla p_j \tag{2}$$

where $k_{\mathrm{r}j}$ is the relative permeability to phase j, μ_j is phase viscosity, $\mathbf{k}$ is the permeability tensor and p_j is phase pressure.

In addition to Equations (1) and (2), we must also prescribe a capillary pressure relationship, of the form $p_{\mathrm{c}}(s_{\mathrm{w}}) = p_{\mathrm{o}} - p_{\mathrm{w}}$, and specify that the saturations sum to one ($s_{\mathrm{o}} + s_{\mathrm{w}} = 1$) at every point. This gives a total of four equations for the four unknowns ($p_{\mathrm{o}}, p_{\mathrm{w}}, s_{\mathrm{o}}, s_{\mathrm{w}}$). However, because the saturation constraint and capillary pressure relationship can be easily incorporated into the governing equations, only two primary equations (e.g. Equation (1) written for water and oil) must be solved for two primary unknowns, taken here to be p_{o} and s_{w}. The solution of these equations is complicated by a number of factors, including the fact that $\mathbf{k}$ can vary by several orders of magnitude over short distances and that non-linearities in $k_{\mathrm{r}j}(s_j)$ lead to the formation of shocks and rarefactions in the saturation field.

For the discretized models, we consider logically Cartesian systems (meaning blocks follow a logical i, j, k ordering) containing a total of n_{c} grid blocks. The state vector $\mathbf{x}$ contains the two primary unknowns:

$$\mathbf{x} = [(p_{\mathrm{o}})_1, (s_{\mathrm{w}})_1, (p_{\mathrm{o}})_2, (s_{\mathrm{w}})_2, \ldots, (p_{\mathrm{o}})_{n_{\mathrm{c}}-1}, (s_{\mathrm{w}})_{n_{\mathrm{c}}-1}, (p_{\mathrm{o}})_{n_{\mathrm{c}}}, (s_{\mathrm{w}})_{n_{\mathrm{c}}}]^{\mathrm{T}} \tag{3}$$

The discretized fully implicit version of Equation (1) can be expressed as [22]

$$\mathbf{T}^{n+1}\mathbf{x}^{n+1} - \mathbf{D}^{n+1}(\mathbf{x}^{n+1} - \mathbf{x}^{n}) - \mathbf{Q}^{n+1} = \mathbf{R} \tag{4}$$

where $\mathbf{T}$ is a block pentadiagonal matrix for 2D grids and a block heptadiagonal matrix for 3D grids, $\mathbf{D}$ is a block diagonal matrix, $\mathbf{Q}$ represents the source/sink terms and $\mathbf{R}$ is the residual vector. The time level is designated by the superscript n or $n+1$. The $\mathbf{T}^{n+1}\mathbf{x}^{n+1}$ term represents transport effects while the $\mathbf{D}^{n+1}(\mathbf{x}^{n+1} - \mathbf{x}^{n})$ term represents accumulation. The matrices $\mathbf{T}$, $\mathbf{D}$ and $\mathbf{Q}$ depend on $\mathbf{x}$ and must be updated at each iteration of every time step.

Equation (4) is non-linear and is solved by applying Newton's method to drive the residuals to zero:

$$\mathbf{J}\boldsymbol{\delta} = -\mathbf{R} \tag{5}$$

where $\mathbf{J}$ is the Jacobian matrix given by $J_{ij} = \partial R_i / \partial x_j$ and $\delta_i = x_i^{n+1,v+1} - x_i^{n+1,v}$ with v and $v+1$ indicating iteration level.

2.2. *Proper orthogonal decomposition*

As noted in the Introduction, in order to generate a POD reduced-order basis, a time simulation of the full flow model must first be performed and the states of the system saved. These states, represented by $\mathscr{S}$ 'snapshots', comprise solutions for p_{o} and s_{w} for all n_{c} grid blocks at particular times. We denote the matrices $\mathbf{x}_p$ and $\mathbf{x}_s$ to represent the system states for p_{o} and s_{w} (from here on designated simply as p and s), respectively:

$$\mathbf{x}_p = \begin{bmatrix} x_{p_1}^1 & x_{p_1}^2 & \cdots & x_{p_1}^{\mathscr{S}} \\ x_{p_2}^1 & x_{p_2}^2 & \cdots & x_{p_2}^{\mathscr{S}} \\ \vdots & \vdots & \ddots & \vdots \\ x_{p_{n_\text{c}}}^1 & x_{p_{n_\text{c}}}^2 & \cdots & x_{p_{n_\text{c}}}^{\mathscr{S}} \end{bmatrix}_{n_\text{c}\times\mathscr{S}}, \quad \mathbf{x}_s = \begin{bmatrix} x_{s_1}^1 & x_{s_1}^2 & \cdots & x_{s_1}^{\mathscr{S}} \\ x_{s_2}^1 & x_{s_2}^2 & \cdots & x_{s_2}^{\mathscr{S}} \\ \vdots & \vdots & \ddots & \vdots \\ x_{s_{n_\text{c}}}^1 & x_{s_{n_\text{c}}}^2 & \cdots & x_{s_{n_\text{c}}}^{\mathscr{S}} \end{bmatrix}_{n_\text{c}\times\mathscr{S}}$$

Each column of these matrices contains the solution of the system for a specific simulation time. The superscript indicates the snapshot number and the subscript the grid block index. The pressure states are normalized by computing $x_{p_i} = (p_i - p_{\min})/(p_{\max} - p_{\min})$, where p_i is the simulated grid block pressure and $p_{\max}$ and $p_{\min}$ are the maximum and minimum pressures encountered during the high-fidelity flow simulation. A similar normalization is applied to the saturation states.

The basic POD procedure we employ is well established and is discussed in a variety of publications (e.g. [2, 3]). The development in this section follows the work of van Doren *et al.* [15]. As in their approach, we apply the ROM procedure separately for the pressure and saturation states. This is appropriate because the two variables are governed by distinct physics (as can be readily demonstrated by manipulating Equations (1) and (2) into so-called pressure and saturation equations; see [21, 22] for details). After the snapshots are obtained, the mean of the snapshots is computed (Equation (6)) and the data matrix $\mathbf{X}$ is determined by subtracting the mean from each snapshot (Equation (7)):

$$\bar{\mathbf{x}} = \frac{1}{\mathscr{S}} \sum_{i=1}^{\mathscr{S}} \mathbf{x}_i \tag{6}$$

$$\mathbf{X} = [\mathbf{x}_1 - \bar{\mathbf{x}}, \mathbf{x}_2 - \bar{\mathbf{x}}, \ldots, \mathbf{x}_{\mathscr{S}} - \bar{\mathbf{x}}]_{n_\text{c}\times\mathscr{S}} \tag{7}$$

Next, a covariance matrix $\mathbf{C}$ is determined by applying the method of snapshots [2]. With this method, instead of computing a matrix of dimensions $n_\text{c} \times n_\text{c}$, a much smaller matrix of dimensions $\mathscr{S} \times \mathscr{S}$ is computed:

$$\mathbf{C} = \mathbf{X}^{\text{T}}\mathbf{X} \tag{8}$$

from which the following eigen-decomposition problem is solved:

$$\mathbf{C}\boldsymbol{\Psi} = \lambda\boldsymbol{\Psi} \tag{9}$$

where $\boldsymbol{\Psi}$ represents the eigenvectors and λ the eigenvalues of $\mathbf{C}$, respectively. We do not actually perform an eigen-decomposition of Equation (9) but rather a SVD. Alternatively, we can perform SVD directly on $\mathbf{X}$. The POD basis functions are then written as a linear combination of the snapshots:

$$\varphi_j = \sum_{i=1}^{\mathscr{S}} \mathbf{\Psi}_{i,j}\mathbf{X}_i \tag{10}$$

The eigenvalues are related to the energy of the system. By arranging the eigenvalues in decreasing order, this energy can be used to selectively retain eigenvectors, meaning those eigenvectors possessing small amounts of energy can be removed from the basis matrix. The energy in the first l eigenvectors is given by

$$E_l = \frac{\sum_{i=1}^{l} \lambda_i}{E_t} \tag{11}$$

where λ_i represents the energy of each basis function and $E_t = \sum_{i=1}^{\mathscr{S}} \lambda_i$. The number of basis functions retained for the pressure unknown, designated l_p, is simply the number of eigenvalues necessary to provide a specified fraction of E_t. The number of basis functions for the saturation unknown, l_s, is determined analogously.

Following application of Equation (11), the basis matrix $\mathbf{\Phi}$ will contain only l columns, where $l = l_p + l_s$. We refer to this reduced $\mathbf{\Phi}$ matrix as $\mathbf{\Phi}_l$. Now, the reduced state vector, designated $\mathbf{z}$, can be related to the full oil pressure and water saturation states $\mathbf{x}$ as follows:

$$\mathbf{x} \simeq \mathbf{\Phi}_l \mathbf{z} + \bar{\mathbf{x}} \tag{12}$$

As indicated above, for an oil–water system two basis matrices will be created, $\mathbf{\Phi}_{l_p}$ and $\mathbf{\Phi}_{l_s}$ for the oil pressure and water saturation states, respectively. Combining both matrices, Equation (12) can be represented in matrix form as

$$\begin{bmatrix} x_{p_1} \\ x_{s_1} \\ x_{p_2} \\ x_{s_2} \\ \vdots \\ x_{p_{n_c-1}} \\ x_{s_{n_c-1}} \\ x_{p_{n_c}} \\ x_{s_{n_c}} \end{bmatrix} \simeq \begin{bmatrix} \varphi_{p_1}^1 & \cdots & \varphi_{p_1}^{l_p} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \varphi_{s_1}^1 & \cdots & \varphi_{s_1}^{l_s} \\ \varphi_{p_2}^1 & \cdots & \varphi_{p_2}^{l_p} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \varphi_{s_2}^1 & \cdots & \varphi_{s_2}^{l_s} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \varphi_{p_{n_c}}^1 & \cdots & \varphi_{p_{n_c}}^{l_p} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \varphi_{s_{n_c}}^1 & \cdots & \varphi_{s_{n_c}}^{l_s} \end{bmatrix} \begin{bmatrix} z_{p_1} \\ \vdots \\ z_{p_{l_p}} \\ z_{s_1} \\ \vdots \\ z_{s_{l_s}} \end{bmatrix} + \begin{bmatrix} \bar{x}_{p_1} \\ \bar{x}_{s_1} \\ \bar{x}_{p_2} \\ \bar{x}_{s_2} \\ \vdots \\ \bar{x}_{p_{n_c-1}} \\ \bar{x}_{s_{n_c-1}} \\ \bar{x}_{p_{n_c}} \\ \bar{x}_{s_{n_c}} \end{bmatrix} \tag{13}$$

where $\mathbf{x}$ is of dimension $2n_c$, $\mathbf{\Phi}_l$ is a $2n_c \times (l_p + l_s)$ matrix, $\mathbf{z}$ has dimension $l_p + l_s$, and $\bar{\mathbf{x}}$ is of dimension $2n_c$. The benefits of the method are achieved because $l_p + l_s \ll 2n_c$; i.e. the dimension of $\mathbf{z}$ is much less than that of $\mathbf{x}$.

An important aspect of the ROM process is the generation of snapshots (see, e.g. Burkdardt *et al.* [19]). The information embodied in the snapshots is used to compute the POD basis and therefore impacts strongly the ability of the reduced-order basis to represent the states of the system. The predictive capabilities of the model are important because our target application is production optimization. This means that we require the ROM to be predictive for a variety of different flow scenarios, and these scenarios are not known at the time the snapshots are generated.

We will not consider the issue of snapshot selection in detail in this work. Rather, we apply a simple heuristic approach for snapshot generation that appears to perform adequately for current purposes. We note, however, that detailed treatments exist for snapshot selection that we may consider in future work. For example, Kunisch and Volkwein [23] developed the optimality system proper orthogonal decomposition (OS-POD), which addresses the problem of un-modeled dynamics, as occurs when the optimum solution is considerably different from the training simulation. Bui-Thanh *et al.* [24] presented an alternate procedure that employs a greedy algorithm for the selection of snapshots in high-dimensional space.

2.3. Reduced-order oil–water reservoir model

Our intent is now to introduce the reduced state $\mathbf{z}$ into the discretized flow equations. This will allow us to solve for only $l_p + l_s$ unknowns rather than $2n_c$ unknowns. By inserting Equation (12) into the discrete model (Equation (4)) we obtain

$$\mathbf{T}^{n+1}(\mathbf{\Phi}_l \mathbf{z}^{n+1} + \bar{\mathbf{x}}) - \mathbf{D}^{n+1}\mathbf{\Phi}_l(\mathbf{z}^{n+1} - \mathbf{z}^n) - \mathbf{Q}^{n+1} = \mathbf{R} \tag{14}$$

Premultiplying Equation (14) by $\mathbf{\Phi}_l^{\mathrm{T}}$, the reduced form of the residual equation is given by

$$\mathbf{\Phi}_l^{\mathrm{T}}\mathbf{T}^{n+1}(\mathbf{\Phi}_l \mathbf{z}^{n+1} + \bar{\mathbf{x}}) - \mathbf{\Phi}_l^{\mathrm{T}}\mathbf{D}^{n+1}\mathbf{\Phi}_l(\mathbf{z}^{n+1} - \mathbf{z}^n) - \mathbf{\Phi}_l^{\mathrm{T}}\mathbf{Q}^{n+1} = \mathbf{\Phi}_l^{\mathrm{T}}\mathbf{R} = \mathbf{R}_{\mathrm{r}} \tag{15}$$

where the reduced residual $\mathbf{R}_{\mathrm{r}}$ has dimension $l = l_p + l_s$ rather than $2n_c$.

Similarly, the reduced Jacobian can be computed by multiplying the original Jacobian by the basis matrix from the left and right sides:

$$\mathbf{J}_{\mathrm{r}} = \mathbf{\Phi}_l^{\mathrm{T}}\mathbf{J}\,\mathbf{\Phi}_l \tag{16}$$

The reduced Jacobian has dimensions $l \times l$, in contrast to the $2n_c \times 2n_c$ dimensions of the original Jacobian. However, the heptadiagonal matrix structure of the original Jacobian (for 3D systems) is lost, and $\mathbf{J}_{\mathrm{r}}$ is in general a full matrix. Finally, Newton's method can be applied to the reduced $(l \times l)$ system of equations:

$$\mathbf{J}_{\mathrm{r}}\boldsymbol{\delta}_{\mathrm{r}} = -\mathbf{R}_{\mathrm{r}} \tag{17}$$

where $(\delta_{\mathrm{r}})_i = z_i^{n+1,v+1} - z_i^{n+1,v}$.

Although the reduced system of equations is of much lower dimension than the full system, the procedure described above still requires the construction of the full-order Jacobian and residual at every iteration in order to form $\mathbf{J}_{\mathrm{r}}$ and $\mathbf{R}_{\mathrm{r}}$. Thus, the speedup resulting from this procedure will be somewhat limited. An alternate approach that does not require the construction of the full $\mathbf{J}$ and $\mathbf{R}$ will be discussed in Section 2.5.

2.4. Clustering snapshots

The POD basis is optimal in the sense that it captures the most information on average from a set of snapshots [10]. Specifically, given a snapshot set, for any number of basis functions POD minimizes the average of the square distance between the centered snapshots $(\mathbf{x}^i - \bar{\mathbf{x}})$ and the projections $\mathbf{\Phi}_l \mathbf{z}^i$:

$$E_p = \frac{1}{\mathscr{S}} \sum_{i=1}^{\mathscr{S}} \|\mathbf{x}^i - \bar{\mathbf{x}} - \mathbf{\Phi}_l \mathbf{z}^i\|^2 \tag{18}$$

where E_p is the quantity minimized.

In our application, the snapshots are simply the states computed at each time step (or every nth time step, with n an integer) of the full-order model. This means that, depending on the time stepping strategy employed in the simulation of the full-order model, the snapshots may not be evenly distributed in state space. In other words, snapshots might be concentrated in relatively small fractions of state space, and this may bias the resulting $\mathbf{\Phi}_l$ by overweighting certain regions of state space. Thus, although E_p in Equation (18) is still minimized, this minimization is based on a set of snapshots that is not as representative of the overall state space as it could be.

We address this issue by clustering the snapshots such that the cluster centroids are spaced approximately evenly in state space. By applying the POD procedure to the cluster centroids, we are then able to generate a slightly more accurate basis matrix. For this purpose we apply the K-means clustering procedure available in Matlab [25]. Our use of this clustering approach was motivated by the centroidal Voronoi tesselation (CVT) ROM introduced by Burkardt *et al.* [19]. CVT reduced-order modeling also starts with a snapshot set, but instead of determining a POD basis from the snapshot set, a clustering technique is applied to determine the generators (cluster centroids) of a CVT of the snapshot set, and those generators are the reduced-order basis. Our approach also applies a clustering technique to the snapshot set, but instead of determining generators, the cluster centroids are used in the POD methodology.

Matlab's [25] K-means procedure uses a two-stage iterative algorithm to minimize the sum of distances between snapshots and centroids summed over all $\mathscr{C}$ clusters. The initial cluster centroid positions are chosen by randomly selecting $\mathscr{C}$ snapshots from the original $\mathscr{S}$ snapshots. Each iteration of the first stage consists of simultaneously reassigning snapshots to their nearest cluster centroids, followed by recalculation of cluster centroids. In the second stage, snapshots are reassigned only if this reduces the sum of distances, after which cluster centroids are recomputed. The overall algorithm converges to a local minimum. By running it several times with random starting points, the optimization space is sampled more globally. The clustering is performed separately for the pressure and saturation snapshots.

Applying this procedure, $\mathscr{S}$ snapshots are reduced to $\mathscr{C}$ centroids. Then the POD procedure is applied to the $\mathscr{C}$ centroids. In our implementation, in order to achieve a high degree of computational efficiency for this clustering, we have the option of performing most of the clustering computations in a reduced space. The resulting algorithm is very efficient so the clustering introduces negligible overhead.

2.5. *Missing point estimation*

The key attribute of the basis matrix $\mathbf{\Phi}_l$ is its ability to approximate the $2n_\mathrm{c}$ states in the spatial domain using relatively few basis functions. However, the ROM procedure still requires substantial computation because the ROM is constructed from the full model [26, 27]. Specifically, as in a standard simulation, at each iteration of each time step we must first construct the full Jacobian matrix, after which the reduced Jacobian can be formed using Equation (16). Both the initial construction of the full Jacobian and the formation of the reduced Jacobian via two matrix multiplications ($\mathbf{J}_r = \mathbf{\Phi}_l^T \mathbf{J} \mathbf{\Phi}_l$) are time consuming. We note that iterative linear solution techniques could be applied to solve Equation (17) (here we use a direct method), in which case $\mathbf{J}_r$ would not need to be constructed explicitly. We did not investigate iterative techniques here, though we note that the MPE procedure described below should also improve their performance.

The MPE procedure was introduced by Astrid *et al.* [28–30]. This method represents the basis functions in terms of column vectors not of length n_c (as in Equation (13)) but rather of length n_m, with $n_m < n_c$. This acts to reduce the size of $\mathbf{\Phi}_l$ and can therefore lead to computational speedup. Astrid *et al.* applied MPE to a model of a glass melt feeder where the temperature distribution had to be controlled very precisely. In the following, we describe the use of MPE for our application.

The basis matrix $\mathbf{\Phi}_l$ is an orthonormal basis, meaning that its columns are normal and mutually orthogonal. This means that the magnitude of each column is 1 and the condition number of $\mathbf{\Phi}_l^T \mathbf{\Phi}_l$ is also near 1 (i.e. $1+\varepsilon$, where $\varepsilon \sim \mathcal{O}(10^{-10})$). The condition number is computed as

$$\mathcal{K}(\mathbf{\Phi}_l^T \mathbf{\Phi}_l) = \frac{\lambda_{\max}(\mathbf{\Phi}_l^T \mathbf{\Phi}_l)}{\lambda_{\min}(\mathbf{\Phi}_l^T \mathbf{\Phi}_l)} \tag{19}$$

where $\lambda_{\max}(\mathbf{\Phi}_l^T \mathbf{\Phi}_l)$ and $\lambda_{\min}(\mathbf{\Phi}_l^T \mathbf{\Phi}_l)$ are the maximum and minimum eigenvalues of $\mathbf{\Phi}_l^T \mathbf{\Phi}_l$, respectively.

The idea behind the MPE is to remove a number of rows of the $\mathbf{\Phi}_l$ basis matrix while allowing the condition number to increase only to a specified value. The new basis matrix will be designated $\mathbf{\Phi}_l^* \in \mathbb{R}^{n_m \times l}$, where n_m represents the number of grid blocks retained. Astrid [20] presented two different methods for grid block selection. The first method classifies the grid blocks by considering one block at a time and then computing $e = \| \mathbf{\Gamma}_l^T \mathbf{\Gamma}_l - \mathbf{I} \|$, where $\mathbf{\Gamma}_l$ represents the row of $\mathbf{\Phi}_l$ corresponding to the target block and $\mathbf{I}$ is the identity matrix. The blocks are then ordered and those corresponding to the smallest values of e are retained, with the condition number for the resulting $\mathbf{\Phi}_l^*$ computed using Equation (19). This procedure is continued until a target condition number is reached.

As reported by Astrid [20], this method requires that a relatively large number of grid blocks be retained. To overcome this limitation Astrid implemented a greedy algorithm. This procedure applies the algorithm described above to select the first block. Then, all remaining $(n_c - 1)$ blocks are considered one at a time and the block that gives the minimum condition number for the two-block system is retained. The procedure is repeated until the target condition number is reached. We found this method to perform reasonably well for small models but it is too time consuming for application to large problems (e.g. for the 60 000 grid block example considered in Section 3).

We achieved the greatest reduction in the number of rows of $\mathbf{\Phi}_l$ using a sequential QR decomposition (SQRD) approach. This approach was suggested by Y. Efendiev (private communication), who applied it for conditioning permeability fields generated using Karhunen–Loève expansions. This was done within the context of history matching reservoir models to production data using Markov chain Monte Carlo methods [31].

We implemented SQRD in Matlab and applied it to classify the grid blocks from the most important to the least important. For the pressure state, given $\mathbf{\Phi}_{l_p}^T \in \mathbb{R}^{l_p \times n_c}$, QR decomposition finds the l_p orthogonal vectors that correspond to the l_p greatest independent vectors of $\mathbf{\Phi}_{l_p}^T$ (a set of n_c vectors in $\mathbb{R}^{l_p}$ is linearly dependent if $n_c > l_p$). This set of l_p vectors is then removed from $\mathbf{\Phi}_{l_p}^T$ and the process is repeated until all grid blocks are classified. The same procedure is applied for the saturation state.

We then apply the condition number criterion to construct a new basis matrix. The number of grid blocks to be added in each step is specified (n_{b}). Then, following the classification order, n_{b} grid blocks are selected for the pressure state (n_{b_p}) and n_{b} grid blocks are selected for the saturation state (n_{b_s}). The number of grid blocks added in each step k (n_{b}^k) is given by the union of n_{b_p} and n_{b_s}, $n_{\mathrm{b}}^k = n_{\mathrm{b}_p} \cup n_{\mathrm{b}_s}$. Grid blocks (rows of $\mathbf{\Phi}_l^*$) are added in this way until the target condition number for $\mathbf{\Phi}_l^*$ (which includes both pressure and saturation information) computed by Equation (19) is reached. For the cases studied here the condition number that produced an appropriate balance between speedup and error was generally between 5 and 10.

2.6. Implementation in general purpose research simulator

Stanford's general purpose research simulator (GPRS), originally formulated by Cao [17] and recently enhanced and extended by Jiang [18], has evolved into a modeling package containing many advanced capabilities and features. These include linkages to adjoint-based optimal control procedures [32, 33], advanced linear solvers [18, 34], and models for 'smart wells', e.g. multi-branched wells containing downhole sensors and inflow control devices [18]. All of the simulations presented in the following sections were performed using a new version of GPRS, which is able to handle the POD procedure described above. The simulator also contains a partial implementation of MPE. We now briefly describe these implementations.

The ROM described previously can be separated into off-line (pre-processing) and in-line portions. The off-line portion, executed just once, contains all of the computations needed to construct the ROM. Figure 1 shows the flow chart for the off-line portion. This entails first running a training simulation and recording snapshot sets for pressure and saturation. Then, following the procedure given in Section 2.4, the snapshots can be clustered, the basis functions generated (Section 2.2) and the grid blocks selected by the MPE procedure (Section 2.5).

The in-line portion of the ROM, integrated with the reservoir simulator, is depicted in Figure 2. This ROM can be applied for a variety of different simulation scenarios. The basis functions and selected grid blocks (if MPE is used), as determined from the off-line procedure, are inputs. Then, within the Newton loop, the standard Jacobian matrix and residual vector are generated, after which the reduced Jacobian and residual are formed using $\mathbf{\Phi}_l$ or $\mathbf{\Phi}_l^*$ if MPE is being used. Newton's method is then applied to the reduced system of equations, after which the full state vector ($\mathbf{x}$) is reconstructed using $\mathbf{\Phi}_l$ (note that $\mathbf{\Phi}_l$ rather than $\mathbf{\Phi}_l^*$ is used in this step even if MPE is applied). The model initialization and time stepping are exactly as in the standard simulator.

Our current implementation of the MPE in GPRS is restricted to code at the level of the linear solver. By this we mean that we use MPE to speed up the matrix multiplications ($\mathbf{\Phi}_l^{\mathrm{T}}\mathbf{J}\mathbf{\Phi}_l$) and in the solution of the reduced-order system, but not elsewhere in the code. This is clearly suboptimal, as the full implementation of MPE would allow us to construct residual equations and Jacobian entries for only n_m (rather than n_{c}) grid blocks. This full implementation would, however, require code modifications throughout GPRS, while the solver-level implementation leads to much more limited modifications. The ROMs generate small but full matrices. We therefore apply direct solution techniques for these linear systems.

In the following section, we will compare the performance and timing of the ROM to full GPRS simulations, so some discussion of the linear solvers applied in GPRS is appropriate. The linear system of equations arising in the full simulation model is very sparse. For this solution, GPRS employs the iterative generalized minimum residual (GMRES) solver along with various preconditioners. The constrained pressure residual (CPR) preconditioner [18, 34] is the most

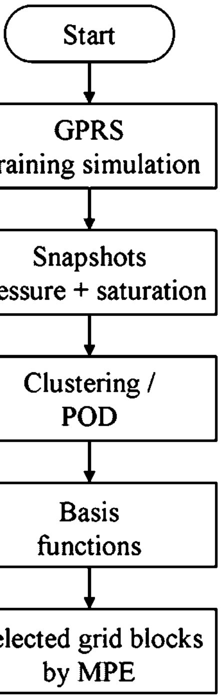

Figure 1. Flow chart for the off-line portion of the ROM.

advanced option. This preconditioner is specially designed for the reservoir simulation equations (it takes full advantage of the elliptic character of the underlying pressure equation) and is therefore highly efficient. Most of our comparisons will be against full simulation models that use CPR, so we are comparing against what we believe to be (essentially) the best current procedure. More generic preconditioners, such as ILU(0), are also available within GPRS and will be considered.

3. RESERVOIR SIMULATION USING ROM

We now illustrate the application of the POD to a realistic reservoir simulation model. We will consider each component in turn (construction of the reduced basis, clustering, MPE) and then demonstrate the ability of the reduced model to provide accurate predictions for cases that differ from the initial training simulation. Simulation times and errors due to the various ROM procedures are quantified at the end of this section.

The simulation model, shown in Figure 3, is a portion of a very large geological model developed by Castro [35]. This model represents a fluvial channel system. Realistic permeability and porosity values were assigned to channel sands and to background 'mud' regions. The model is 3D and contains a total of 60 000 grid blocks (with $n_x = 75$, $n_y = 100$, and $n_z = 8$, where n_x, n_y, and n_z designate the number of grid blocks in the corresponding coordinate direction). Five production wells and four water injection wells drive the flow.

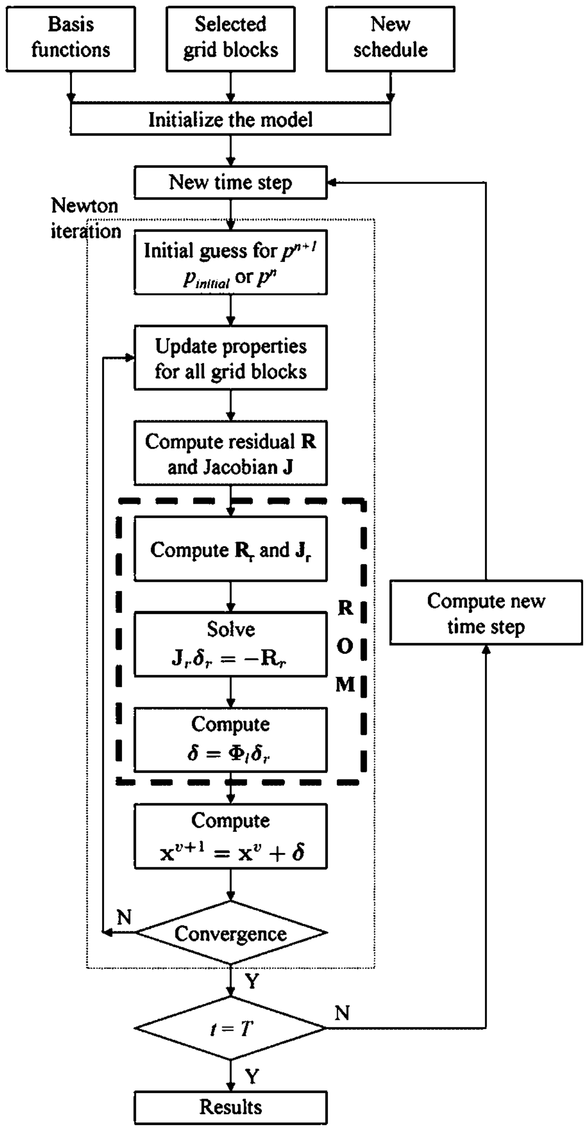
Basis functions
Selected grid blocks
New schedule
Initialize the model
New time step
Newton iteration
Initial guess for p^{n+1} $p_{initial}$ or p^n
Update properties for all grid blocks
Compute residual **R** and Jacobian **J**
Compute $\mathbf{R}_r$ and $\mathbf{J}_r$
Solve $\mathbf{J}_r\delta_r = -\mathbf{R}_r$
Compute $\delta = \Phi_l\delta_r$
R O M
Compute $\mathbf{x}^{v+1} = \mathbf{x}^v + \delta$
Convergence
N
Y
$t = T$
N
Y
Results
Compute new time step

Figure 2. Flow chart for the in-line portion of the ROM.

Figure 4 shows the permeability in the x-direction for all eight layers. A number of channelized geological features are clearly evident. The mean permeability and porosity are 2600 mD and 0.26, respectively, for the channels and 20 mD and 0.08 outside the channels. The production wells are completed in layers 1, 2, and 3 and the injector wells in layers 7 and 8.

The initial saturations of oil and water are 0.85 and 0.15, respectively, and the residual oil saturation is 0.30. The oil viscosity at standard conditions is 1.16 cp and the water viscosity is 0.325 cp. Capillary pressure is neglected and the fluid densities are set to be equal. The relative permeabilities for the oil and water phases are shown in Figure 5.

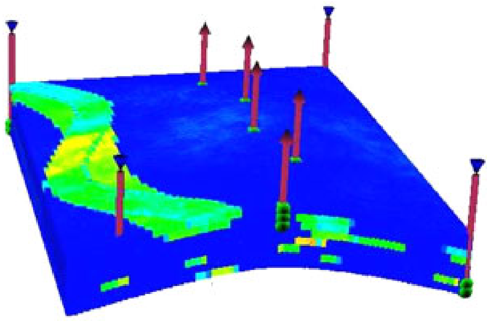

Figure 3. Synthetic 3D reservoir model with five production wells and four injection wells.

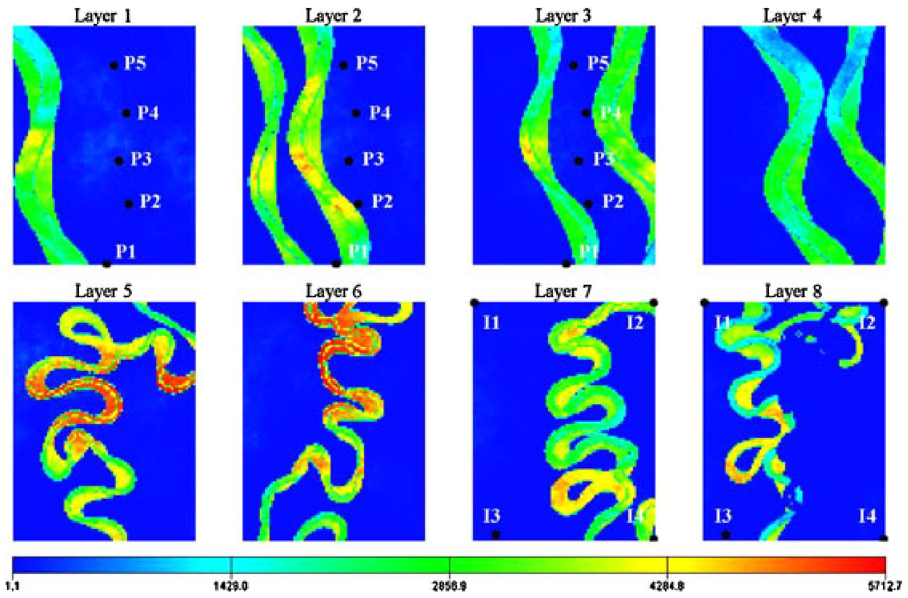

Figure 4. Permeability in the x-direction for all eight layers.

To extract the information needed to reproduce the behavior of the system, a full run (referred to as the training simulation) was performed. As indicated earlier, the conditions applied for this training simulation impact the quality of the reduced basis, so they should be selected with care. Here we apply a heuristic procedure in which we vary the bottom hole pressures (BHPs) of the production wells randomly and independently over their expected ranges (between 3500 and 4500 psia). These BHPs are changed every 100 days and the resulting schedule is shown in Figure 6. The injection well BHPs are constant in time for each well, though the actual values vary from well to well (the BHPs for injectors 1 and 2 are 6500 psia, for injector 3 BHP is 7000 psia and for injector 4 BHP is 6000 psia). The maximum time step allowed was 20 days and a total of 211 snapshots for the oil pressure and water saturation states were recorded.

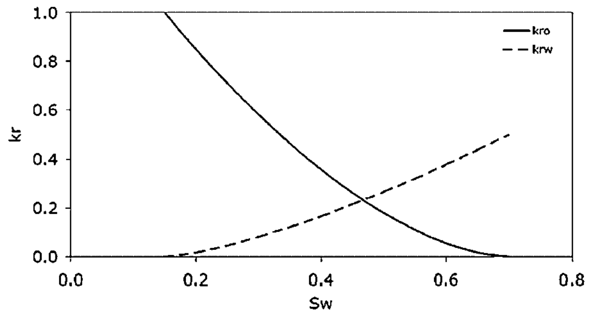

Figure 5. Relative permeability curves for the oil and water phases.

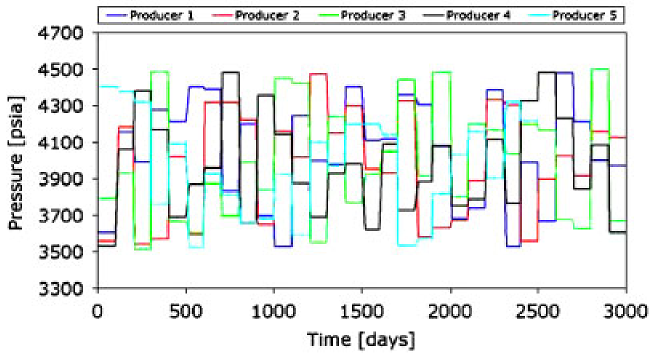

Figure 6. Schedule for bottom hole pressures for the production wells.

Applying the POD approach provides the eigenvalue spectrums for pressure and saturation shown in Figure 7. It is evident that the maximum eigenvalue for the pressure state is almost 10^3 while the minimum is around 10^{-16}, which means that the pressure eigenvalues vary over around 19 orders of magnitude. For the water saturation state this variation is also substantial, about 18 orders of magnitude.

We seek to form a basis such that the fraction of energy in the pressure state that we ignore is very low, around 10^{-6}, and the fraction ignored in the saturation state is around 10^{-4}. This requires that we retain the first 14 eigenvalues for the oil pressure state and the first 26 eigenvalues for the water saturation state. Therefore, the basis matrix $\mathbf{\Phi}_l \in \mathbb{R}^{2n_c \times l}$, where $l = l_p + l_s$, has $14+26=40$ basis vectors. This means that, while the standard reservoir simulation model needs to solve Equation (5) for $2n_c = 120000$ unknowns, the reduced-order reservoir simulation model needs to solve Equation (17) for only 40 unknowns.

We present the first six pressure and saturation basis functions for the first layer in Figures 8 and 9. The fraction of energy contained in each basis function is also indicated. These basis functions clearly reflect the locations of the wells and the pressure and saturation gradients, though it is otherwise difficult to attribute direct physical interpretation to their detailed shapes.

Implementing the clustering technique presented in Section 2.4, the number of snapshots for each state is reduced from 211 to 50. Ignoring the same amounts of energy as before, the number

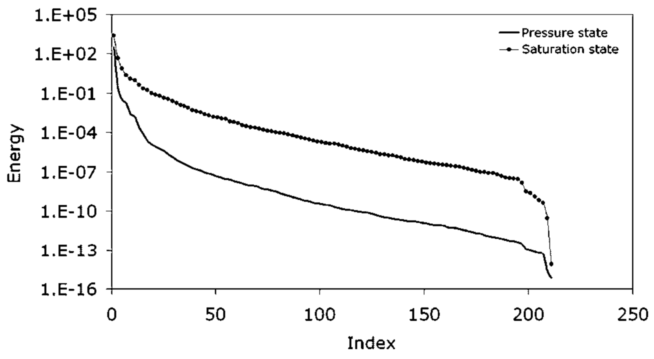

Figure 7. Eigenvalue variation for pressure and saturation matrices.

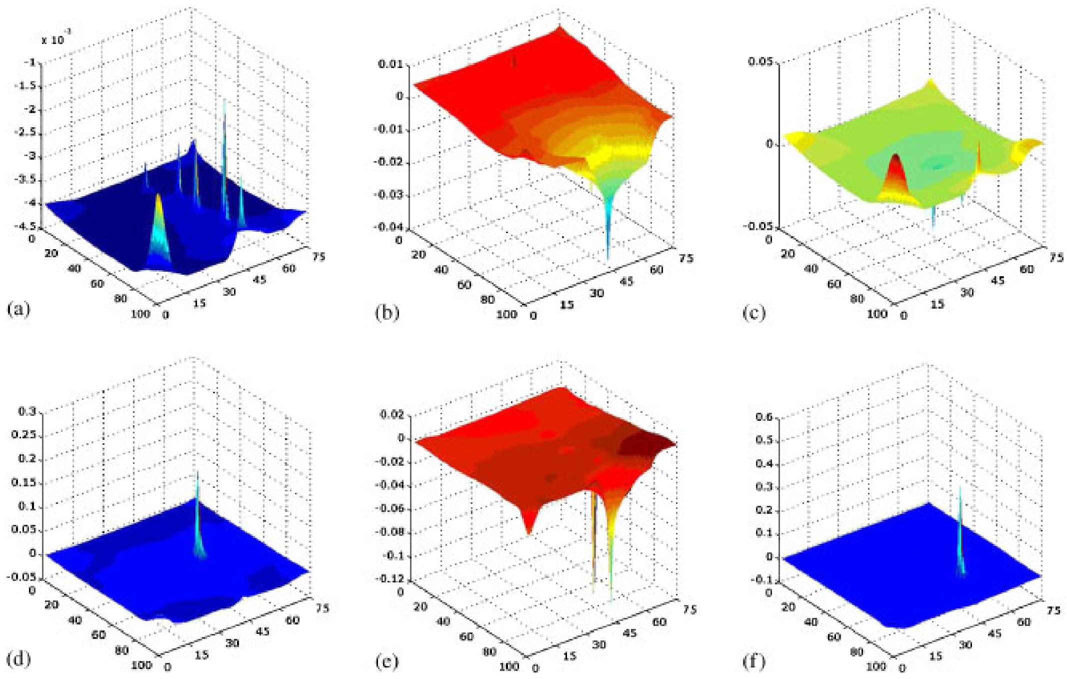
Figure 8. Six most important reduced-order basis functions for the oil pressure state (layer 1): (a) Energy: 9.9257×10^{-1}; (b) energy: 6.26×10^{-3}; (c) energy: 7.57×10^{-4}; (d) energy: 1.42×10^{-4}; (e) energy: 9.85×10^{-5}; and (f) energy: 6.40×10^{-5}.

of basis functions required for the oil pressure state decreases from 14 to 12, while for the water saturation state it decreases from 26 to 22. Thus the reduced-order reservoir simulation model now requires only 34 unknowns.

Applying the MPE technique to the clustered snapshots and using the grid block selection procedure described in Section 2.5, the grid blocks are ordered from the most important to the

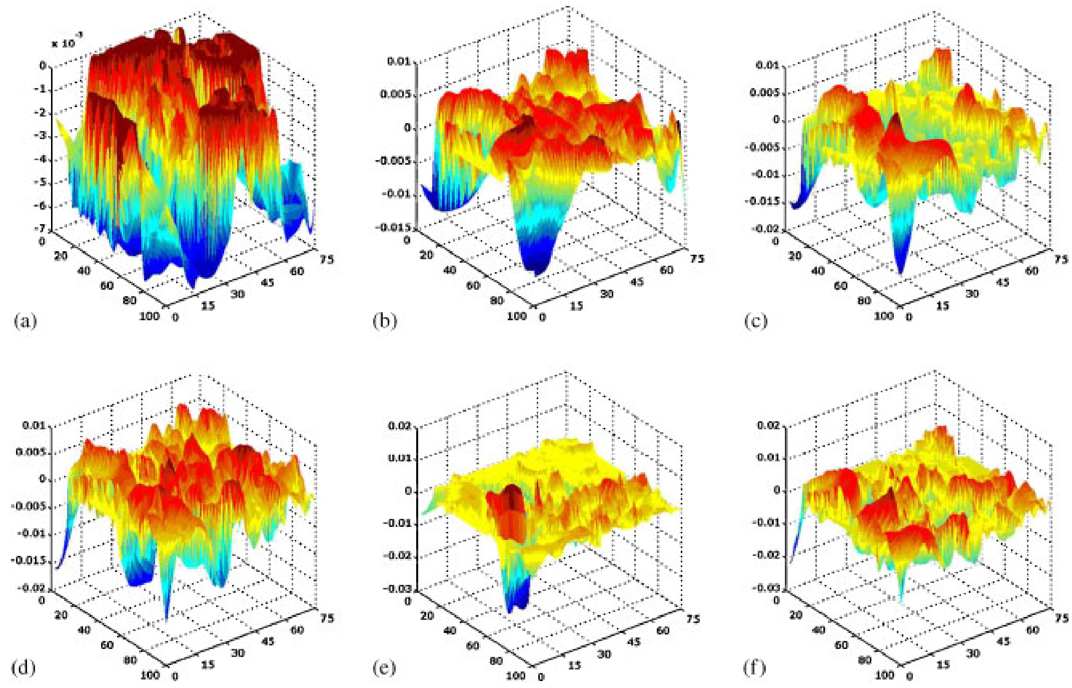

Figure 9. Six most important reduced-order basis functions for the water saturation state (layer 1): (a) Energy: 8.79×10^{-1}; (b) energy: 9.03×10^{-2}; (c) energy: 1.70×10^{-2}; (d) energy: 4.52×10^{-3}; (e) energy: 2.87×10^{-3}; and (f) energy: 1.84×10^{-3}.

least important in terms of their impact on the condition number. Allowing the condition number to increase to 5, the number of grid blocks selected is $n_m = 19441$. Figure 10 shows that the selected grid blocks (shown in black) are related to the wells and the high-permeability channels, as indicated for layer 1, where the circles indicate the location of the producer wells, the arrows the injector wells and the polygons regions of high permeability. The basis matrix, now of dimension $n_m \times 34$, can be applied to efficiently compute the reduced Jacobian appearing in Equation (16). As indicated in Section 2.6, the MPE procedure was implemented into GPRS only at the solver level. Thus, the full benefits of this treatment will not be realized in the simulation results presented below.

The ability of the ROM to reproduce the training simulation will be tested using the three basis function matrices obtained above. The first matrix was generated using only POD, $\mathbf{\Phi}_l \in \mathbb{R}^{120\,000\times40}$; the second matrix was generated using clustered snapshots and POD, $\mathbf{\Phi}_l \in \mathbb{R}^{120\,000\times34}$; and the third was formed using clustered snapshots, POD and MPE, $\mathbf{\Phi}_l^* \in \mathbb{R}^{38\,882\times34}$.

Figures 11 and 12 display the oil and water flow rates for all producer wells and the injection rates for all injector wells using the three procedures. The total quantity of injected fluid is equal to 0.67 of the total pore volume of the system. However, because of the residual (non-flowing) oil and water fractions, this corresponds to 1.22 mobile pore volumes of fluid. The reference oil and water flow rates from the training simulation (red circles for oil and blue for water) are very well reproduced using POD only (solid red circles for oil and blue for water), clustered snapshots + POD (solid yellow circles for oil and light blue for water), and clustered snapshots + POD + MPE (solid

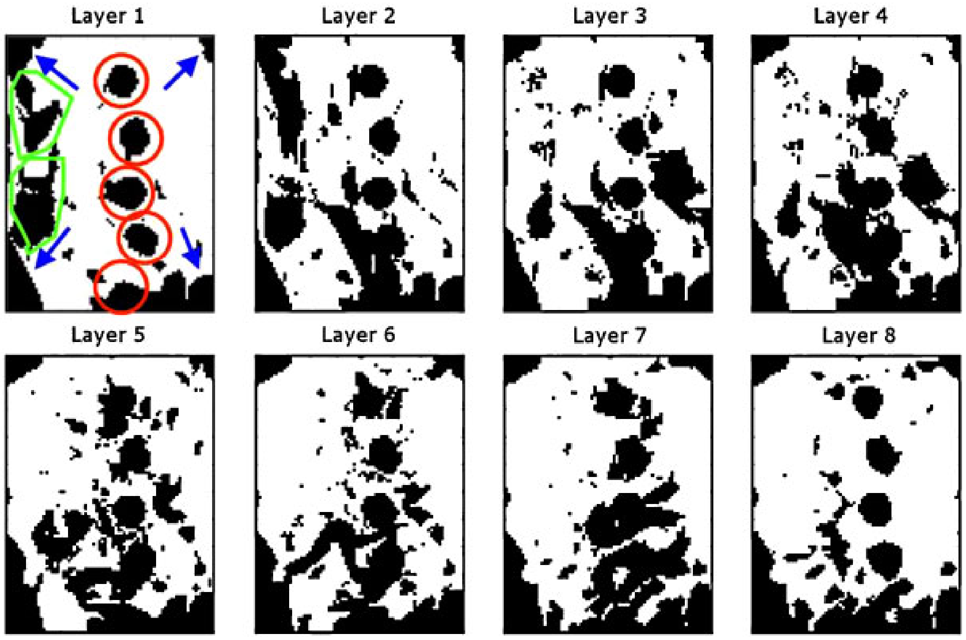

Figure 10. Grid blocks selected (in black) by MPE procedure.

black dots for oil and gray for water). The reference water injection rate from the training simulation (green circles) is also well reproduced using POD (solid green circles), clustered snapshots+POD (solid light green circles) and clustered snapshots+POD+MPE (black dots). These results are very encouraging and indicate that the snapshot set and the basis contain sufficient information to reproduce the training simulation. We can also observe that the clustering approach enables a reduction in the size of the basis matrix without losing significant accuracy in the simulation results. We note that the use of POD only (without clustered snapshots) with 34 unknowns (the number used for the clustered snapshots+POD case) provided less accurate simulation results than those obtained using clustered snapshots+POD. This demonstrates the benefit of the clustering procedure. Finally, using fewer than $\frac{1}{3}$ of the grid blocks, the MPE technique provides results that continue to agree with the training simulation.

We next consider two different flow scenarios to evaluate the predictive capability of the three ROMs.

3.1. *Prediction using ROM—schedule I*

The schedule for the BHPs of the production wells is shown in Figure 13. The overall range (between 3650 and 4350 psia) is smaller than the range used in the training simulation. In this example, the BHP is changed every 200 days, while in the training simulation it was changed every 100 days. The injection well BHPs are the same as in the training simulation. The maximum time step for this case was increased to 50 days.

Figure 14 shows the oil and water flow rates for producer wells 1 and 4 using the three basis matrices. The flow rates from the high-fidelity (reference) solution (red circles for oil and blue for

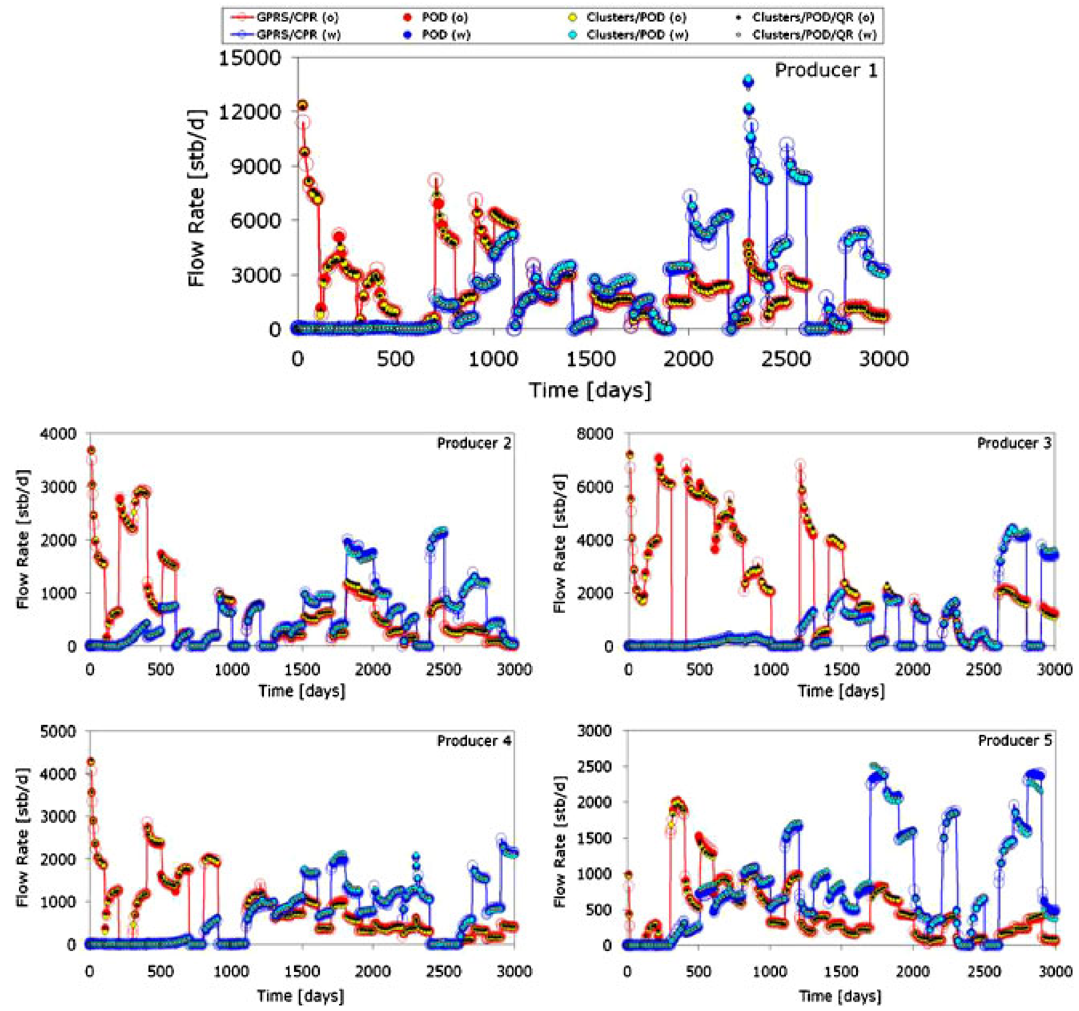

Figure 11. Oil and water flow rates for the training simulation using full-order GPRS and three ROMs.

water) are, in general, well reproduced using the three ROMs. Relatively small mismatches can be observed, however, in some of the results. The results for water injection rates for the injector wells 1 and 4, shown in Figure 15, demonstrate close agreement with the reference simulation, although some slight mismatches can be observed. Similar results (though not shown) were obtained for producer wells 2, 3, and 5 and injector wells 2 and 3.

3.2. Prediction using ROM—schedule II

The schedule for the production well BHPs is shown in Figure 16. These BHPs are changed every 200 days, and the overall range, between 3350 and 4650 psia, is larger than the range used in the training simulation. The specifications for the injection wells are the same as in the previous cases.

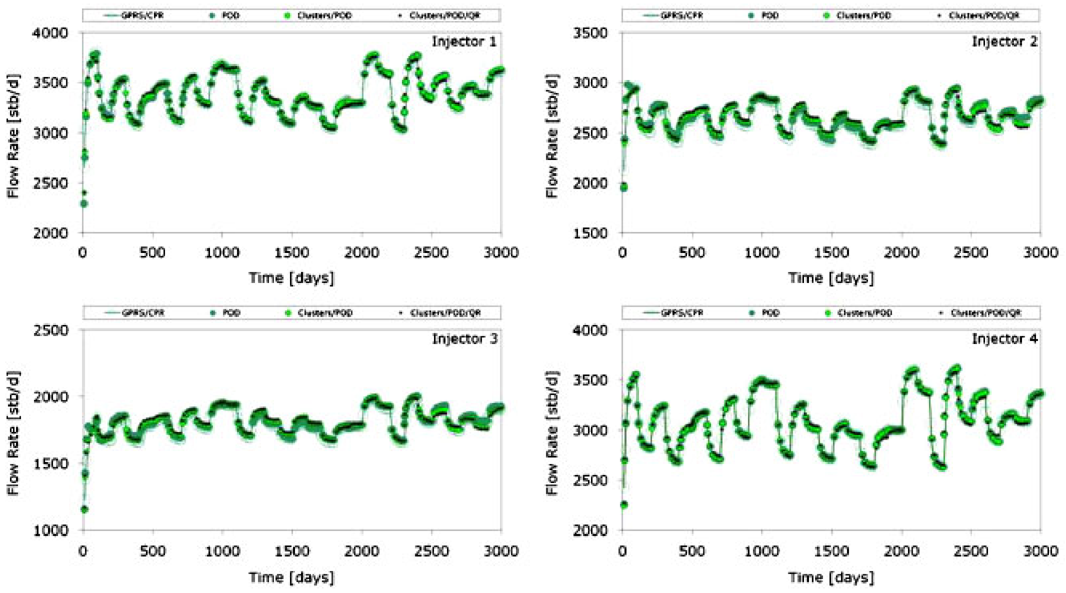

Figure 12. Injection rate for the training simulation using full-order GPRS and three ROMs.

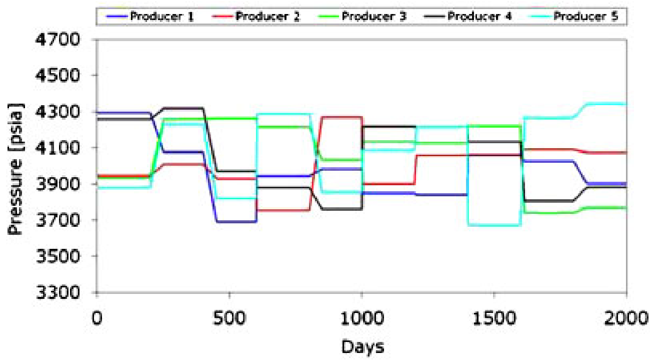

Figure 13. Bottom hole pressure for the producer wells using schedule I.

Figure 17 shows the oil and water flow rates for schedule II for producer wells 1 and 4 using the three basis matrices. The flow rates from the high-fidelity solution are again reasonably well reproduced using the three ROMs, though some errors are evident. The results for water injection rates, shown in Figure 18, also demonstrate generally close agreement with the reference simulation. Taken in total, these results are encouraging as they demonstrate the ability of the ROM to provide flow results for scenarios that differ substantially from the training simulations. This is important if the ROM is to be used within the context of optimization.

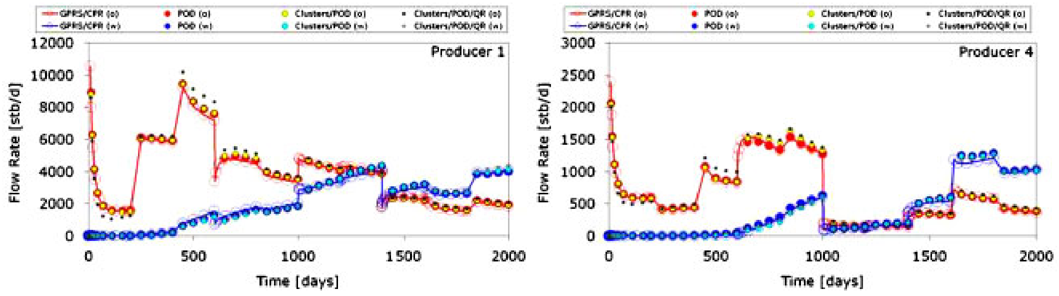

Figure 14. Oil and water flow rates for schedule I with three ROMs.

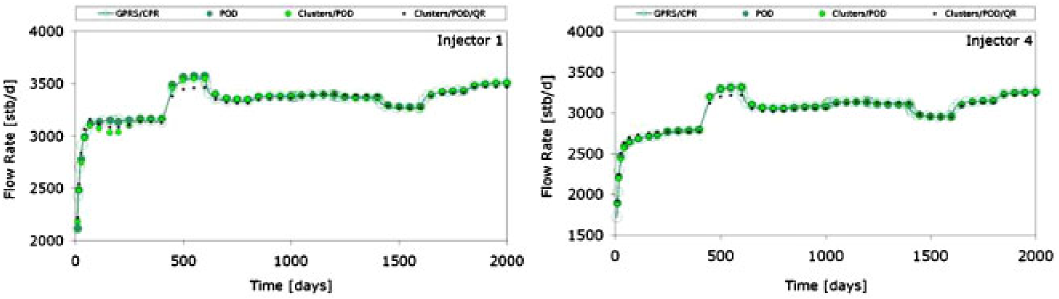

Figure 15. Water injection rate for schedule I with three ROMs.

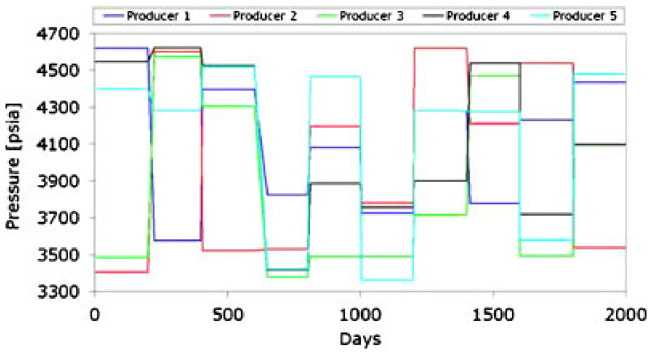

Figure 16. Bottom hole pressure for the producer wells using schedule II.

3.3. Simulation time reduction with ROMs

The simulation times for the high-fidelity models and the three ROMs for the base case and schedules I and II are summarized in Figure 19. In the figure, the first set of bars refers to the base case, the second set to schedule I and the third set to schedule II. For each set the first bar represents the simulation time for the high-fidelity simulation using the fastest current solution procedure (GMRES solver with CPR preconditioner). The second, third and fourth bars in each set

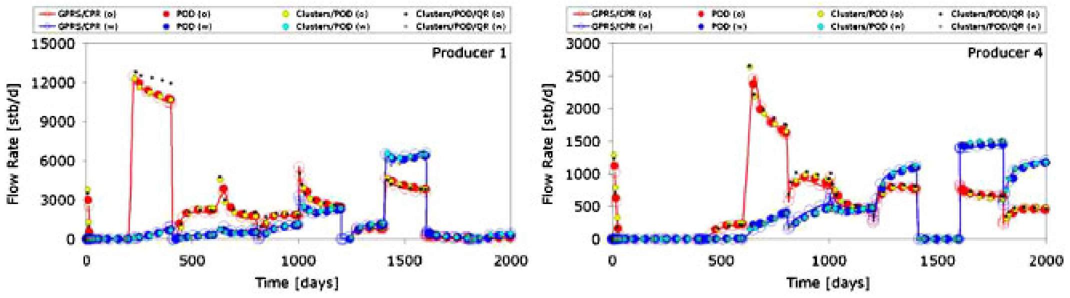

Figure 17. Oil and water flow rates for schedule II with three ROMs.

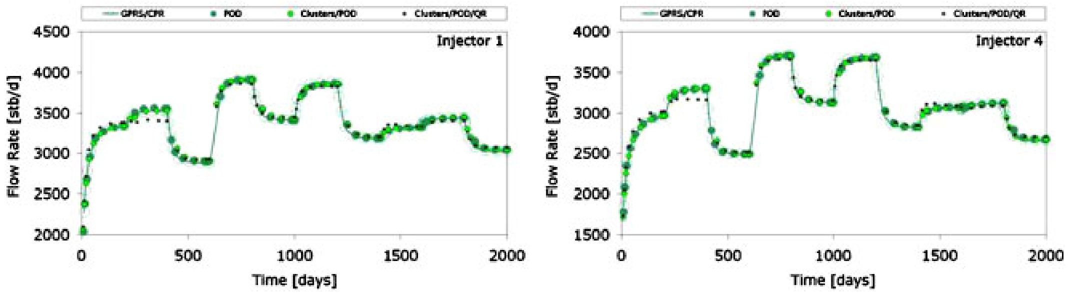

Figure 18. Water injection rate for schedule II with three ROMs.

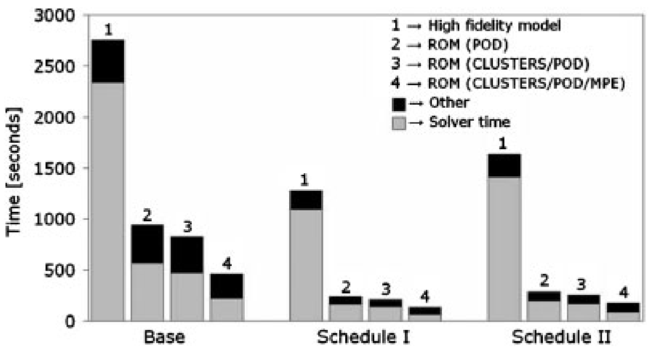

Figure 19. Comparison of simulation time.

show the simulation times for the three ROMs (POD, clusters+POD, and clusters+POD+MPE, respectively). Additionally, for each bar the gray segment represents the time spent solving the linear system of equations and the black segment the time spent in all other computations (computing block properties, forming the residual vector and Jacobian matrix, etc.).

For the base case the first set of bars shows that the ROM with POD was able to reduce the simulation time by a factor of ~2.9. Applying the ROM with clusters+POD the simulation time was reduced by a factor of ~ 3.3. Further reduction, namely a factor of ~ 5.9, was obtained when the ROM with clusters+POD+MPE was employed. The speedups achieved using the various ROMs, though significant, are not dramatic. This is because our ROM procedures target only the solver. Thus, even if the ROM reduces the solver time to zero, we are still left with the remaining simulation computations. We note, however, that a full MPE implementation (which was not introduced here) would provide additional speedup. This is because, in contrast to the other ROM procedures, a full MPE implementation would also reduce the time required for residual and Jacobian construction.

The computation times for schedules I and II simulations are indicated (respectively) by the second and third sets of bars. For schedule I, the speedups resulting from the ROMs are greater than those observed for the base case (e.g. about a factor of 9.5 for clusters+POD+MPE). This is due in part to the ability of the ROMs to take larger time steps than the high-fidelity model. Similar results were obtained using schedule II. As indicated by the third set of bars, the speedup for the clusters+POD+MPE ROM was 9.8.

3.4. *Comparison of ROMs with ILU(0) preconditioner*

As explained in Section 2.6, the full-order simulations are performed using GPRS with the GMRES solver and the CPR preconditioner. This is a highly developed preconditioner that provides essentially the best available linear solution methodology for many reservoir simulation applications. Its inherent speed, therefore, limits the degree of speedup achievable using our algorithms.

It may also be appropriate, therefore, for us to evaluate speedups relative to a more generic solver. This is because, for many applications, a highly specialized solver plus preconditioner combination will not be available. For example, our current gradient-based production optimization algorithm, which requires the solution of an adjoint equation in conjunction with the simulation model, is not yet compatible with GMRES+CPR but instead uses ILU(0) as a preconditioner [32, 33]. Thus, we now assess the speedup achieved by the ROM procedures relative to GMRES+ILU(0).

We consider the schedule I predictive scenario. The timings for GPRS using the ILU(0) and CPR preconditioners relative to the ROM (clusters+POD+MPE) are shown in Figure 20. Note that the time axis is logarithmic and the simulation time using preconditioner ILU(0) was normalized to 1. For this case, the preconditioner CPR provides a speedup of about 73 and the ROM (clusters+ POD+MPE) a quite dramatic speedup of about a factor of 700 relative to GPRS with ILU(0). This is because, for the full (high-fidelity) simulation using GPRS with ILU(0), the solver requires almost all of the computation time, so our ROM procedure has substantial impact.

3.5. *Quantification of error using ROMs*

Errors arise from the ROM simulations for two main reasons. These are (1) we discard a large number of basis vectors which, although less important than the basis vectors retained, do carry some information and (2) the ROM is in general applied to models that differ from the training simulation used to generate the reduced basis. It is therefore appropriate to quantify the ROM error for the various cases.

This error quantification could be accomplished in a variety of manners. Here we apply a simple procedure and focus on average error in oil and water (injected and produced) rates. For each production well m, at every simulated time step i, the oil production rate in the reference simulation ($Q^{m,i}_{\mathrm{o,full}}$) and in each ROM simulation ($Q^{m,i}_{\mathrm{o,ROM}}$) is computed. Results from the ROMs are interpolated to provide rates at the times computed in the reference simulation (with care being

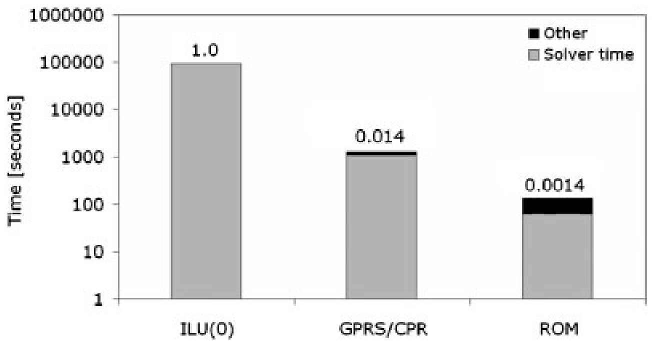

Figure 20. Comparison of simulation time for schedule I using full model with ILU(0) and CPR preconditioners and ROM.

taken to avoid interpolating across discontinuities in well BHP). The time-average error for each well, designated E^m, is then calculated as the average of the absolute differences, normalized by the time-average oil flow rate $\bar{Q}^m_{\mathrm{o,full}}$ for the well:

$$E^m = \frac{1}{n_{\mathrm{t}}\bar{Q}^m_{\mathrm{o,full}}} \sum_{i=1}^{n_{\mathrm{t}}} |Q^{m,i}_{\mathrm{o,full}} - Q^{m,i}_{\mathrm{o,ROM}}| \tag{20}$$

where n_{t} is the total number of time steps. The overall averaged error, designated E, is then given by

$$E = \frac{1}{n_{\mathrm{w}}} \sum_{m=1}^{n_{\mathrm{w}}} E^m \tag{21}$$

where n_{w} is the number of production wells. The same procedure is applied for the water production and water injection rates.

Values for E computed via Equation (21) for the training simulation and the two schedules are presented in Tables I–III for oil production, water production, and water injection rates, respectively. It is evident from the tables that the training simulation case results in the smallest errors for all of the ROMs for oil rate and water production rate; for water injection rate the training and schedule I errors are the same for the clusters+POD+MPE model. Specifically, the errors for all models for the training case are less than 5.1% in all three tables. It is not surprising that reproduction of the training runs is generally the most accurate as the basis vectors were generated using snapshots from this case. For schedules I and II, errors in oil production rate vary from about 4.5 to 8.5% for the different models; for water production rate errors vary from about 6.4 to 9.1%, and for water injection rate the errors vary from about 1.0 to 1.9%. Thus, for all simulations the errors in these injection and production rates are reasonable, less than 10% (when quantified in this way). We note that there is a tradeoff between speedup and accuracy, as further speedup could be achieved by decreasing the dimension of $\mathbf{\Phi}$, though this will generally result in increased error. This effect will be quantified in the next section.

Another error that arises in reservoir simulation is mass balance error. The finite volume formulation is constructed to conserve mass, but mass balance errors can still appear as a result of non-zero tolerances in the linear and non-linear (Newton) solution loops. Reservoir simulators

Table I. Errors in the oil rate for the various ROMs.

ROM	Training	Schedule I	Schedule II
POD	0.0164	0.0539	0.0453
Clusters+POD	0.0294	0.0492	0.0526
Clusters+POD+MPE	0.0424	0.0849	0.0761

Table II. Errors in the water production rate for the various ROMs.

ROM	Training	Schedule I	Schedule II
POD	0.0202	0.0911	0.0636
Clusters+POD	0.0406	0.0761	0.0707
Clusters+POD+MPE	0.0509	0.0734	0.0718

Table III. Errors in the water injection rate for the various ROMs.

ROM	Training	Schedule I	Schedule II
POD	0.0064	0.0139	0.0105
Clusters+POD	0.0078	0.0100	0.0134
Clusters+POD+MPE	0.0109	0.0109	0.0193

typically compute global relative mass balance errors for each component (with the mass balance error normalized relative to the total mass originally in place). For the high-fidelity models, these mass balance errors are typically about 10^{-7}. For the ROMs they are somewhat larger, up to 10^{-2}, for example, when MPE is applied. The ROMs also do not guarantee that water saturation always remains between the physical limits of s_{wc} (connate water saturation) and $1-s_{\text{or}}$ (where s_{or} is residual oil saturation). However, when saturation values outside of the physical range did occur, the violations were quite small (maximum of 0.001) and no special treatment was required.

4. FURTHER OBSERVATIONS

In this section we consider the behavior of the ROM procedure when (1) fewer basis vectors are used and (2) when the flow problem is more non-linear or includes additional physics. These results are of interest because they indicate some of the limitations of the current ROM implementation and therefore suggest directions for future investigations.

We first assess the impact of retaining less energy (and thus fewer basis functions) in the pressure and saturation bases. As the number of basis functions is reduced, we expect to achieve a higher degree of speedup but also a loss of accuracy. Three different reduced-order bases are considered, each of which neglects different fractions of the energy. All three of these ROMs were generated using only POD (i.e. no clustering or MPE procedures were applied). For these runs the ROMs are applied to reproduce the base case (this is the same base case as was used in the examples in Section 3).

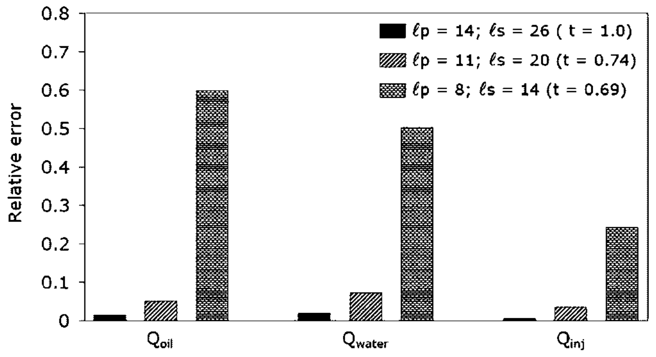

Figure 21. Tradeoff between speedup and accuracy for the training simulation.

The first ROM is the same as that used in Section 3. As indicated in Section 3, this basis ignores 10^{-6} of the energy in the pressure state and 10^{-4} of the energy in the saturation state. Thus, 14 basis functions are used for pressure and 26 for saturation. The simulation time for this case is normalized to 1. The second ROM ignores a slightly larger fraction of the total energy: 5×10^{-6} for the pressure state and 2.5×10^{-4} for the saturation state. This ROM contains 11 basis functions for pressure and 20 for saturation. The simulation using this basis requires 0.74 as much time as the first ROM. The third ROM neglects a still larger fraction of the total energy: 5×10^{-5} for pressure and 6×10^{-4} for saturation. Only eight basis functions are now used for pressure and 14 for saturation. The normalized simulation time for this ROM is 0.69.

The results for errors in oil production rate, water production rate, and water injection rate using the three ROMs are summarized in Figure 21. Errors are computed using Equation (21). It is evident that error increases as l_p and l_s decrease. This increase is relatively slight when $l=l_p+l_s$ decreases from 40 to 31, but the increase in error is large when l is further decreased to 22. The results using the first and second ROMs are quite acceptable (errors for all quantities less than 0.1), though for the third ROM the errors increase appreciably (e.g. 0.60 for oil rate). This demonstrates that, even though a large fraction of energy (>0.999) may be captured by the basis, simulation accuracy can still be very sensitive to the number of basis functions used in $\mathbf{\Phi}$. In our current implementation, the appropriate l_p and l_s must be determined through numerical experimentation, as we do not yet have an accurate means for the *a priori* determination of the size of $\mathbf{\Phi}$.

We next consider the impact of increased non-linearity on the performance of the ROM. For the models considered, the key source of non-linearity is the oil and water relative permeability curves. This non-linearity can be adjusted through the so-called Corey exponents a and b in the relative permeability functions:

$$k_{\text{ro}}=k_{\text{ro}}^{0}\left(\frac{1-s_{\text{w}}-s_{\text{or}}}{1-s_{\text{wc}}-s_{\text{or}}}\right)^{a} \tag{22}$$

$$k_{\text{rw}}=k_{\text{rw}}^{0}\left(\frac{s_{\text{w}}-s_{\text{wc}}}{1-s_{\text{wc}}-s_{\text{or}}}\right)^{b} \tag{23}$$

The original relative permeability curves, shown in Figure 5, were determined experimentally and were specified in tabular form. Expressed in the form of Equations (22) and (23), these curves correspond approximately to $a=1.7$ and $b=1.5$. We now consider curves with $a=3$ and $b=3$. These curves are plotted with the original curves in Figure 22. The other parameters used for the new curves are, with reference to Equations (22) and (23), $k_{\text{ro}}^0=1.0$, $k_{\text{rw}}^0=0.5$, $s_{\text{wc}}=0.15$, and $s_{\text{or}}=0.3$.

The training simulation with well BHPs described in Section 3 was run using these more non-linear relative permeability curves. The number of POD basis functions needed to reproduce the high-fidelity results with errors comparable to those presented in Section 3.5 was then determined through numerical experimentation. The number of basis functions required was 29 for pressure (as compared with 14 for the previous case) and 84 for saturation (as compared with 26 for the previous case). This basis neglects about the same amount of energy for the pressure state as the previous case, but for the saturation state it neglects only 10^{-5} of the total energy, as compared with 10^{-4} for the previous case. Owing to the use of larger l_p and l_s, the speedup using the ROM with the more non-linear relative permeability curves is less than that achieved with the previous case. Specifically, in this case we achieve a very modest speedup of ~ 1.3, as compared with a speedup of ~ 2.9 using the original curves (though we note that more substantial speedups could be obtained through use of the clustering and MPE procedures). This demonstrates the impact of high degrees of non-linearity on the ROM and the need for better treatments for such cases.

Another significant issue is the impact of gravitational effects on the performance of the ROM. In the presence of gravitational effects, Equation (2) becomes:

$$\mathbf{u}_j = -\frac{k_{rj}(s_j)}{\mu_j}\mathbf{k}(\nabla p_j - \rho_j g \nabla D) \tag{24}$$

where ρ_j is the density of phase j, g is gravitational acceleration, and D is depth. Gravitational effects in the saturation equation can be quantified through the density difference ($\Delta\rho$) between the oil and water phases (we note that a more complete quantification of this effect would be in terms of a dimensionless gravity number, though for current purposes the use of $\Delta\rho$ will suffice). The impact of gravitational effects on ROMs is here assessed by applying POD to high-fidelity snapshots generated with varying $\Delta\rho$. For all cases the simulation time was 1000 days and 170 snapshots were generated.

Results are displayed in Figure 23, where we present the variation in eigenvalues for the pressure and saturation states for models with $\Delta\rho=$ 0, 1, 5, and 20 lb/ft^3. It is apparent from the figure that the decay in the magnitudes of the eigenvalues is slower, for both pressure and saturation, with

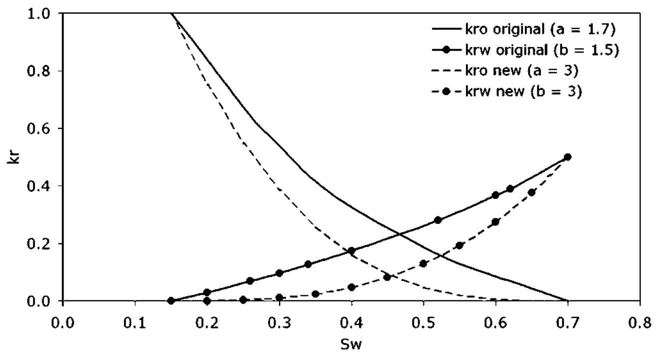

Figure 22. Original and modified relative permeability curves.

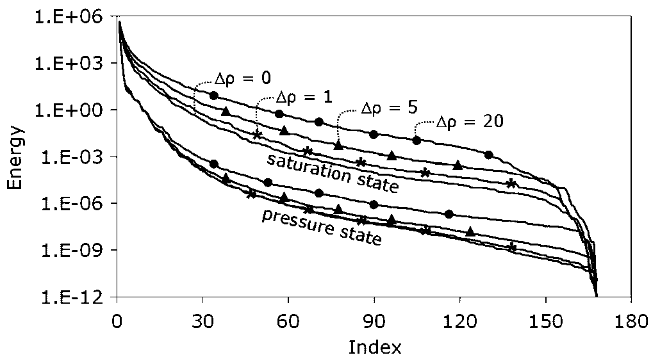

Figure 23. Eigenvalue variation for pressure and saturation for models with density differences.

increasing $\Delta\rho$. In general, the slower the decay in the eigenvalues, the larger the number of basis functions that must be retained in the ROM. Thus, we expect to observe decreasing computational efficiency using ROMs as $\Delta\rho$ is increased. Numerical experiments confirm this expectation.

There are many practical subsurface flow cases for which the relative permeability curves are not highly non-linear and gravitational effects are small relative to viscous effects. For such cases, the ROM procedures applied here would be expected to be effective. However, for cases with highly non-linear relative permeability curves and/or strong gravitational effects, further enhancements in our ROM procedures will likely be required before they can be applied in practice.

5. SUMMARY

In this work a reduced-order model (ROM) based on the proper orthogonal decomposition (POD) technique was extended, incorporated into a general-purpose simulator, and applied for a subsurface flow application. The POD procedure provides a mechanism to extract a reduced-order basis of small dimension that contains the key features of the states encountered during the simulation of a full (high-fidelity) reservoir model. This requires the recording of snapshots of the solution states (oil pressure and water saturation) at many time steps during a 'training' simulation followed by a SVD of the resulting covariance or data matrix. A snapshot clustering procedure was implemented to reduce the number of snapshots, enabling the use of fewer vectors in the reduced-order basis. Further reduction of the computational burden was achieved through application of the missing point estimation (MPE) technique. Using a condition number criterion, MPE allows the most important grid blocks to be selected, leading to a reduction in the number of rows in the basis matrix.

POD, clustering, and MPE are all off-line (pre-processing) computations and were implemented in a stand-alone Matlab application. The ROM framework was implemented in Stanford's general-purpose research simulator (GPRS). Using this implementation ROMs with different complexities can be applied, specifically, ROM with POD only, ROM using clustered snapshots and POD, and ROM using clustered snapshots, POD and MPE.

All of these ROM procedures were applied to a realistic 3D reservoir model with 60 000 grid blocks. This corresponds to 120 000 unknowns (oil pressure and water saturation in each block) at each time step. The ROMs were able to capture the dynamics of these models using only about 40 unknowns (the exact number varied depending on the specific ROM procedure). Results for production and injection rates, which are typically the key inputs to an optimization procedure, were shown to be in fairly close agreement with reference (high-fidelity) simulation results for different production schedules. For all cases the same sets of basis vectors were used, meaning that only one full-order simulation was required to generate the information needed to construct the ROMs. This indicates that these models retain a reasonable level of robustness, which is important if they are to be used in optimization applications.

Computational speedups for the ROM using clustered snapshots, POD and MPE, evaluated relative to full-order GPRS simulations using a specialized linear solver and preconditioner combination (GMRES solver and CPR preconditioner), were between 6 and 10 depending on the case. Comparison to a full-order GPRS simulation using a much simpler preconditioner (ILU(0)) showed a speedup of about 700. This highlights the fact that the ROM procedures considered here are very well suited for situations where the linear solver occupies the great majority of the computational time. Otherwise, the maximum speedup attainable is more limited.

The models considered here (oil–water systems, equal density fluids, capillary pressure neglected) are still relatively simple, in terms of multiphase flow physics, compared with many of the models that arise in practice. The presence of strong gravitational effects as well as high degrees of non-linearity in the relative permeability functions was shown to limit the effectiveness of the ROM procedures implemented in this work. The extension and enhancement of the ROM techniques to address these limitations will be the subject of future work.

ACKNOWLEDGEMENTS

We thank Yalchin Efendiev (Texas A&M University) for suggesting the use of sequential QR decomposition for the missing point estimation procedure. We are also grateful to two anonymous reviewers for their insightful comments and suggestions on the initial version of this paper.

REFERENCES

1. Lumley JL. Atmospheric turbulence and radio wave propagation. *Journal of Computational Chemistry* 1967; **23**(13):1236–1243.
2. Sirovich L. Turbulence and the dynamics of coherent structures part I–III. *Quarterly of Applied Mathematics* 1987; **45**(3):561–590.
3. Chatterjee A. An introduction to the proper orthogonal decomposition. *Current Science* 2000; **78**(7):808–817.
4. Antoulas AC, Sorensen DC. Approximation of large-scale dynamical systems: an overview. *International Journal of Applied Mathematics and Computer Science* 2001; **11**(5):1093–1121.
5. Astrid P, Verhoeven A. Application of least squares MPE technique in the reduced order modeling of electrical circuits. *17th International Symposium on Mathematical Theory of Networks and Systems*, Kyoto, Japan, July 2006.
6. Zheng D, Hoo KA, Piovoso MJ. Low-order model identification of distributed parameter systems by a combination of singular value decomposition and the Karhunen–Loève expansion. *Industrial and Engineering Chemistry Research* 2002; **41**(6):1545–1556.
7. Cao Y, Zhu J, Luo Z, Navon IM. Reduced order modeling of the upper tropical Pacific Ocean model using proper orthogonal decomposition. *Computers and Mathematics with Applications* 2006; **52**(8–9):1373–1386.
8. Bui-Thanh T, Damodaran M, Willcox K. Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition. *AIAA Journal* 2004; **42**(8):1505–1516.
9. Meyer M, Matthies HG. Efficient model reduction in non-linear dynamics using the Karhunen–Loève expansion and dual-weighted-residual methods. *Computational Mechanics* 2003; **31**:179–191.

10. Ravindran SS. Adaptive reduced-order controllers for thermal flow system using proper orthogonal decomposition. *SIAM Journal on Scientific Computing* 2002; **23**(6):1924–1942.
11. Vermeulen PTM, Heemink AW, Te Stroet CBM. Reduced models for linear groundwater flow models using empirical orthogonal functions. *Advances in Water Resources* 2004; **27**:57–69.
12. Vermeulen PTM, Heemink AW. Inverse modeling of groundwater flow using model reduction. *Water Resources Research* 2005; **41**(6):1–13.
13. Vermeulen PTM, Heemink AW. Model-reduced variational data assimilation. *Monthly Weather Review* 2006; **134**:2888–2899.
14. Heijn T, Markovinović R, Jansen JD. Generation of low-order reservoir models using system-theoretical concepts. *SPE Journal* 2004; **9**(2):202–218.
15. van Doren JFM, Markovinović R, Jansen JD. Reduced-order optimal control of water flooding using proper orthogonal decomposition. *Computational Geosciences* 2006; **10**:137–158.
16. Markovinović R, Jansen JD. Accelerating iterative solution methods using reduced-order models as solution predictors. *International Journal for Numerical Methods in Engineering* 2006; **68**(5):525–541.
17. Cao H. Development of techniques for general purpose simulators. *Ph.D. Thesis*, Stanford University, 2002.
18. Jiang Y. A flexible computational framework for efficient integrated simulation of advanced wells and unstructured reservoir models. *Ph.D. Thesis*, Stanford University, 2007.
19. Burkardt J, Gunzburger M, Lee HC. Centroidal Voronoi tessellation-based reduced-order modeling of complex systems. *SIAM Journal on Scientific Computing* 2006; **28**(2):459–484.
20. Astrid P. Reduction of process simulation models: a proper orthogonal decomposition approach. *Ph.D. Thesis*, Eindhoven University of Technology, 2004.
21. Gerritsen MG, Durlofsky LJ. Modeling fluid flow in oil reservoirs. *Annual Review of Fluid Mechanics* 2005; **37**:211–238.
22. Aziz K, Settari A. *Fundamentals of Reservoir Simulation*. Elsevier: Amsterdam, 1986.
23. Kunisch K, Volkwein S. Proper orthogonal decomposition for optimality systems. *ESAIM*: *Mathematical Modelling and Numerical Analysis* 2008; **42**:1–23.
24. Bui-Thanh T, Willcox K, Ghattas O, van Bloemen Waanders B. Goal-oriented, model-constrained optimization for reduction of large-scale systems. *Journal of Computational Physics* 2007; **224**(2):880–896.
25. MathWorks. The MathWorks, Inc. http://www.mathworks.com.
26. Nguyen NC, Patera AT, Peraire J. A 'best points' interpolation method for efficient approximation of parameterized functions. *International Journal for Numerical Methods in Engineering* 2008; **73**(4):521–543.
27. Bashir O, Willcox K, Ghattas O, van Bloemen Waanders B, Hill J. Hessian-based model reduction for large-scale systems with initial-condition inputs. *International Journal for Numerical Methods in Engineering* 2008; **73**(6):844–868.
28. Astrid P. Fast reduced order modeling technique for large scale LTV systems. *American Control Conference*, Boston, MA, U.S.A., July 2004.
29. Astrid P, Weiland S, Willcox K, Backx T. Missing point estimation in models described by proper orthogonal decomposition. *43rd IEEE Conference on Decision and Control*, Atlantis, Paradise Island, Bahamas, December 2004.
30. Astrid P, Weiland S. On the construction of POD models from partial observations. *44th IEEE Conference on Decision and Control and the European Control Conference*, Seville, Spain, December 2005.
31. Efendiev Y, Hou T, Luo W. Preconditioning Markov chain Monte Carlo simulations using coarse-scale models. *SIAM Journal on Scientific Computing* 2006; **28**(2):776–803.
32. Sarma P, Aziz K, Durlofsky LJ. Implementation of adjoint solution for optimal control of smart wells (SPE paper 92864). *2005 SPE Reservoir Simulation Symposium*, Houston, TX, U.S.A., October 2005.
33. Sarma P, Durlofsky LJ, Aziz K, Chen WH. Efficient real-time reservoir management using adjoint-based optimal control and model updating. *Computational Geosciences* 2006; **10**:3–36.
34. Cao H, Tchelepi HA, Wallis JR, Yardumian H. Parallel scalable unstructured CPR-type linear solver for reservoir simulation (SPE paper 96809). *2005 Annual Technical Conference and Exhibition*, Dallas, TX, U.S.A., October 2005.
35. Castro SA. A probabilistic approach to jointly integrate 3D/4D seismic production data and geological information for building reservoir models. *Ph.D. Thesis*, Stanford University, 2007.