Computational Geosciences (2006) 10:137–158
DOI: 10.1007/s10596-005-9014-2

Reduced-order optimal control of water flooding using proper orthogonal decomposition

Jorn F. M. van Doren[a,b], Renato Markovinović[a] and Jan-Dirk Jansen[a,c]

[a]*Department of Geotechnology, Delft University of Technology, PO Box 5028, 2600 GA Delft, The Netherlands.*
E-mail: j.f.m.vandoren@dcsc.tudelft.nl
[b]*8C-3-10 Delft Center for Systems and Control, Delft University of Technology, Mekelweg 2 (OCP Building) 2628 CD, Delft, The Netherlands.*
[c]*Exploratory Research, Shell International Exploration and Production, PO Box 60, 2280 AB Rijswijk, The Netherlands.*

Received 19 March 2005; accepted 8 August 2005

Model-based optimal control of water flooding generally involves multiple reservoir simulations, which makes it into a time-consuming process. Furthermore, if the optimization is combined with inversion, i.e., with updating of the reservoir model using production data, some form of regularization is required to cope with the ill-posedness of the inversion problem. A potential way to address these issues is through the use of proper orthogonal decomposition (POD), also known as principal component analysis, Karhunen–Loève decomposition or the method of empirical orthogonal functions. POD is a model reduction technique to generate low-order models using 'snapshots' from a forward simulation with the original high-order model. In this work, we addressed the scope to speed up optimization of water-flooding a heterogeneous reservoir with multiple injectors and producers. We used an adjoint-based optimal control methodology that requires multiple passes of forward simulation of the reservoir model and backward simulation of an adjoint system of equations. We developed a nested approach in which POD was first used to reduce the state space dimensions of both the forward model and the adjoint system. After obtaining an optimized injection and production strategy using the reduced-order system, we verified the results using the original, high-order model. If necessary, we repeated the optimization cycle using new reduced-order systems based on snapshots from the verification run. We tested the methodology on a reservoir model with 4050 states (2025 pressures, 2025 saturations) and an adjoint model of 4050 states (Lagrange multipliers). We obtained reduced-order models with 20–100 states only, which produced almost identical optimized flooding strategies as compared to those obtained using the high-order models. The maximum achieved reduction in computing time was 35%.

Keywords: reservoir engineering, water flooding, optimal control, proper orthogonal decomposition, Karhunen–Loève, reduced-order model, reduction

1. Introduction

With increasing computer capacities, reservoir models have become more complex and consist of an increasing number of variables (typically in the order of 10^4–10^6). Maximizing hydrocarbon production and minimizing water production of a reservoir can be done in a smart field with optimal control theory (OCT) [1,3,15,18,20]. Calculating the optimal valve settings using OCT requires several passes of forward simulation of the reservoir model and backward simulation of an adjoint system of equations. The time needed to calculate optimized controls increases with the number of grid blocks and the complexity of the reservoir model. Reduced-order modelling and reduced-order control may provide an alternative to this [5,9,10,19]. Moreover, parameters and variables of the model can be updated with history matching. In this process, output from the real reservoir is compared to output from the model. Using an optimization strategy, the model parameters are updated, and the discrepancy between the measured and the simulated output is minimized. A problem with history matching, however, is the ill-posedness, even if the correct model is assumed. There are usually many parameter combinations that produce near-identical outputs. Normally, one tries to overcome this problem by constraining the solution space for the model parameters through the addition of regularization terms to the objective function. Reduced-order models may provide an alternative, and in case of closed-loop reservoir management, a reduced-order model could be used for both flooding optimization and history matching. Another paper in this special issue describes the use of reduced-order modelling for history-matching the permeability field [16]. In that paper, the reduction method has been applied to the parameters (permeabilities) of the model, whereas we will consider a reduction of the states (pressures and saturations) of the model. Moreover, we will not address the history-matching problem but will concentrate on flooding optimization using proper orthogonal decomposition (POD). POD, also known as principal component analysis, Karhunen–Loève decomposition or the method of empirical orthogonal functions, is a frequently used tool for model reduction, but only recently it has also been used for control applications [6,8,13,14,17]. We will describe a methodology using nested loops, where the inner iterative loop makes use of a truncated basis of POD functions to calculate optimized injection and production rates. After convergence in this loop, we simulate in the outer loop the original, high-order model with the optimized rates and subsequently adapt the basis and the truncation of the POD functions. They are used in the next inner loop to calculate new optimized injection and production rates. We will describe an example in which we applied the methodology to a two-dimensional, two-phase reservoir model and compared full-order optimal control with reduced-order optimal control.

2. High-order reservoir model

To generate a reduced-order model with POD, we first need to run a full-order simulation and produce snapshots. For the full-order model, we use a two-dimen-

sional, two-phase, black oil, reservoir simulator, developed in-house and written in MATLAB [7]. We used a five-point finite difference discretization in space and a semi-implicit discretization in time. Spatial discretization of the original partial differential equations yields the following state-space ordinary differential equation in continuous time t:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) = \mathbf{A}_c(\mathbf{x})\mathbf{x}(t) + \mathbf{B}_c(\mathbf{x})\mathbf{u}(t), \tag{1}$$

$$\mathbf{A}_c = [\mathbf{V}(\mathbf{x}(t))\mathbf{W}(\mathbf{x}(t))]^{-1}\mathbf{T}(\mathbf{x}(t)), \mathbf{B}_c = \mathbf{W}(\mathbf{x}(t))^{-1}, \tag{2,3}$$

where $\mathbf{x}$ is the state vector containing oil pressures p_o and water saturations S_w for each grid block, $\mathbf{V}$ is a diagonal mass matrix with entries that are a function of grid block volume and fluid densities, $\mathbf{W}$ is a block diagonal matrix with entries that are primarily a function of compressibility and porosity, $\mathbf{T}$ is a block pentadiagonal matrix containing the transmissibilities for oil and water and $\mathbf{u}$ is the input vector containing water rates q_w at the injectors and liquid rates $q_l = q_o + q_w$ at the producers [2,12]. We chose not to use a well model, but to use rate-constrained injectors and producers. $\mathbf{A}_c$ and $\mathbf{B}_c$ are actually not explicit functions of state but functions of state-dependent parameters. The closure equations for each grid block are $S_o + S_w = 1$ and $p_o - p_w = p_{cow}$, where S_o is the oil saturation, p_w is the water pressure and $p_{cow}(S_w)$ is the capillary pressure. The initial conditions are specified as $\mathbf{x}(0) = \mathbf{x}_0$. Semi-implicit Euler discretization by treating the state and input vectors implicitly, but the matrix coefficients explicitly, can be written as

$$\frac{\mathbf{x}(k+1) - \mathbf{x}(k)}{\Delta t} = \mathbf{A}_c(\mathbf{x}(k))\mathbf{x}(k+1) + \mathbf{B}_c(\mathbf{x}(k))\mathbf{u}(k), \tag{4}$$

resulting in

$$\mathbf{x}(k+1) = \mathbf{f}_d(\mathbf{x}(k), \mathbf{u}(k+1)) = \mathbf{A}_d(\mathbf{x}(k))\mathbf{x}(k) + \mathbf{B}_d(\mathbf{x}(k))\mathbf{u}(k), \tag{5}$$

where k is the discrete time and where the time step-dependent matrices $\mathbf{A}_d$ and $\mathbf{B}_d$ have now been defined as

$$\mathbf{A}_d(\mathbf{x}(k)) = [\mathbf{I} - \Delta t\mathbf{A}_c(\mathbf{x}(k))]^{-1}, \ \mathbf{B}_d(\mathbf{x}(k)) = \Delta t\mathbf{A}_d(\mathbf{x}(k))\mathbf{B}_c(\mathbf{x}(k)). \tag{6,7}$$

In our numerical implementation, the matrix inverses in equation (6) are not computed, since it is more efficient to solve the equivalent system of linear equations

$$[\mathbf{I} - \Delta t\mathbf{A}_c(\mathbf{x}(k))]\mathbf{x}(k+1) = \mathbf{x}(k) + \Delta t\mathbf{B}_c(\mathbf{x}(k))\mathbf{u}(k) \tag{8}$$

for the unknown state $\mathbf{x}(k+1)$.

3. Proper orthogonal decomposition

Proper orthogonal decomposition has been developed in fluid mechanics and is a mathematical technique to describe 'coherent structures,' which represent low-order dynamics in turbulent flow [6,17]. An approximation of the system dynamics is obtained by projecting the original n-dimensional state space onto an l-dimensional subspace as follows. First, during simulation of an n-dimensional discrete-time model, we record a total of κ snapshots for the oil pressure state $\mathbf{x}_p$ and the water saturation state $\mathbf{x}_s$. In our case, the dimension n will be equal to twice the number of grid blocks. If we have a reservoir model with one horizontal layer of $\overline{m} \times \overline{n}$ grid blocks, the vectors $\mathbf{x}_p$ and $\mathbf{x}_s$ will have $\overline{m}\overline{n}$ elements each, and therefore, the full-state vector $\mathbf{x}$ will have length $n = 2\overline{m}\overline{n}$. We keep the pressure and the saturation states segregated because they correspond to different physical processes and will consequently generate different dominant structures. Moreover, it allows us to choose a different degree of reduction for the pressures and the saturations. For clarity of notation, we will omit the indication of pressure or saturation for the variables in this section, but we note that all steps in the order reduction process should be performed twice, once for the pressures and once for the saturations. After subtracting the mean $\overline{\mathbf{x}} = (1/\kappa)\sum_{i=1}^{\kappa} \mathbf{x}(i)$ from the snapshots, we construct a data matrix:

$$\mathbf{X} := [\mathbf{x}'(1),\ \mathbf{x}'(2), \ldots,\ \mathbf{x}'(\kappa)] = [\mathbf{x}(1) - \overline{\mathbf{x}},\ \mathbf{x}(2) - \overline{\mathbf{x}}, \ldots,\ \mathbf{x}(\kappa) - \overline{\mathbf{x}}]. \tag{9}$$

Subtraction of the mean implies that matrix $\mathbf{R}_n = \mathbf{X}\mathbf{X}^T/(\kappa - 1)$ is the $n \times n$ covariance matrix of state variables as captured by the snapshots. It has at most rank $\kappa - 1$ because it has been constructed from κ snapshots that are dependent through the equation for the mean. A potential benefit of the subtraction of the mean is an increased level of detail in the reduced-order description in case of near-parallel snapshot vectors $\mathbf{x}(i)$. The goal of POD is, given the data matrix $\mathbf{X}$, to find a transformation

$$\mathbf{x}' = \mathbf{\Phi}_l \mathbf{z} + \mathbf{r}, \tag{10}$$

where $\mathbf{\Phi}_l$ is an $n \times l$ transformation matrix, $\mathbf{z}$ is a reduced state vector of length l and $\mathbf{r}$ are residuals, such that the squared sum of the snapshot residuals, $\sum_{i=1}^{\kappa} \|\mathbf{r}(i)\|^2$, is minimized. It can be shown, see, e.g., [6], that this minimum is given by

$$\sum_{i=1}^{\kappa} \|\mathbf{r}(i)\|^2 = \sum\nolimits_{j=l+1}^{\kappa} \lambda_j, \tag{11}$$

where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_\kappa \geq \lambda_{\kappa+1} = \ldots = \lambda_n = 0$ are the ordered solutions of the eigenvalue problem

$$\mathbf{R}_n \boldsymbol{\varphi}_i = \lambda_i \boldsymbol{\varphi}_i, \tag{12}$$

and $\boldsymbol{\varphi}_i$ ($i = 1, \ldots, n$) are the corresponding eigenvectors. Because the rank of $\mathbf{R}_n$ can be at most $\kappa - 1$, we do not need to solve equation (12) but may solve the much smaller eigenvalue problem

$$\boldsymbol{\psi}^T \mathbf{R}_\kappa = \boldsymbol{\psi}^T \lambda, \tag{13}$$

where $\mathbf{R}_\kappa = \mathbf{X}^T\mathbf{X}$ is a $\kappa \times \kappa$ matrix. Noting that $\mathbf{R}_n$ and $\mathbf{R}_\kappa$ are symmetric, we can write

$$\boldsymbol{\Lambda}_n = \boldsymbol{\Phi}^T \mathbf{R}_n \boldsymbol{\Phi} \quad \text{and} \quad \boldsymbol{\Lambda}_\kappa = \boldsymbol{\Psi}^T \mathbf{R}_\kappa \boldsymbol{\Psi}, \tag{14, 15}$$

where $\boldsymbol{\Lambda}_n$ and $\boldsymbol{\Lambda}_\kappa$ are $n \times n$ and $\kappa \times \kappa$ diagonal matrices with ordered eigenvalues λ_i on the diagonal, respectively, whereas $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$ are $n \times n$ and $\kappa \times \kappa$ orthogonal matrices containing the eigenvectors $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$ as columns and rows, respectively. The required (right) eigenvectors $\boldsymbol{\varphi}$ can now be obtained from the (left) eigenvectors $\boldsymbol{\psi}$ with the aid of the relationship

$$\boldsymbol{\Phi} = \mathbf{X}\boldsymbol{\Psi}\boldsymbol{\Lambda}_\kappa^{-1/2}. \tag{16}$$

According to equation (11), the squared sum of the snapshot residuals is determined by the $\kappa - l$ highest eigenvalues. The eigenvectors corresponding to the remaining l eigenvalues, i.e., the first l columns of matrix $\boldsymbol{\Phi}$, form the optimal transformation matrix $\boldsymbol{\Phi}_l$ [6]. We may, alternatively, compute the eigenvectors $\boldsymbol{\varphi}$ with the aid of the singular value decomposition of the data matrix [4]:

$$\mathbf{X} = \boldsymbol{\Phi}\boldsymbol{\Sigma}\boldsymbol{\Psi}^T, \tag{17}$$

where the $n \times \kappa$ matrix $\boldsymbol{\Sigma}$ is given by

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_\kappa \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}, \tag{18}$$

Here $\sigma_1 \geq \ldots \geq \sigma_l >> \sigma_{l+1} \geq \ldots \geq \sigma_\kappa \geq 0$ are the singular values of $\mathbf{X}$ and are the square roots of the eigenvalues λ_i, $i = 1, 2, \ldots, \kappa$. It is simply verified that equations (14) and (15) can be obtained from equation (17) by working out the matrix products $\mathbf{X}\mathbf{X}^T$ and $\mathbf{X}^T\mathbf{X}$. Note that we will have at least one singular value equal to 0 because the rank of $\mathbf{R}_\kappa$ is at most $\kappa - 1$. The number of singular values l, i.e., the number of

POD basis functions that we want to keep, can now be determined as follows. The total amount of relative 'energy' present in the snapshots can be expressed as $E_{tot} = \sum_{i=1}^{\kappa-1} \sigma_i^2$. The reduced number of basis functions is the largest number $l \in \{1, \ldots, \kappa - 1\}$ that satisfies

$$E = \frac{\sum_{i=1}^{l} \sigma_i^2}{E_{tot}} \leq \alpha, \tag{19}$$

where α denotes the fraction of relative energy we want to be captured. If the singular values, ordered by magnitude, display a clear drop, the system apparently has a natural set of dominant singular values. Otherwise, the choice of α becomes somewhat arbitrary. Frequently used cut-off levels are $0.9 < \alpha < 1.0$. Note that we may choose different cut-off criteria for the pressures and the saturations. The transformation matrix $\mathbf{\Phi}_l$ is now taken as the first l columns of the matrix $\mathbf{\Phi}$, and we obtain the transformation:

$$\mathbf{x} \simeq \mathbf{\Phi}_l \mathbf{z} + \overline{\mathbf{x}}. \tag{20}$$

4. Reduced-order reservoir model

After replacing the '$\simeq$' sign by the '=' sign, and dropping the subscript l to simplify the notation, we can substitute relation (20) into equation (5) to obtain

$$\mathbf{z}(k+1) = \mathbf{\Phi}^T[\mathbf{f}(\mathbf{\Phi}\mathbf{z}(k) + \overline{\mathbf{x}}, \mathbf{u}(k)) - \overline{\mathbf{x}}]. \tag{21}$$

This transformation can be interpreted as a discrete-time differential equation in reduced-order state space, obtained by projecting the normalized state vector $\mathbf{x}' = \mathbf{x} - \overline{\mathbf{x}}$ of the original problem on the reduced-order space. As described in [5], it may appear at first sight as if the reduced-order model does not lead to a reduction in simulation time. If we compute $\mathbf{z}(k+1)$ explicitly, we may even obtain a slight increase in simulation time, because every time step, we have to perform two additional transformations (from $\mathbf{z}$ to $\mathbf{x}$ and back) because we need the original state vector $\mathbf{x}$ to compute the functions $\mathbf{f}$. However, this increase will, in general, be offset by an increase in the minimum time step required for stability. More importantly, if we consider implicit or semi-implicit computation of $\mathbf{z}(k+1)$, a considerable efficiency gain may be achieved. Applying the transformation $\mathbf{\Phi}$ to equation (8), we obtain

$$[\mathbf{I} - \Delta t \mathbf{A}_c(\mathbf{\Phi}\mathbf{z}(k) + \overline{\mathbf{x}})]\mathbf{\Phi}\mathbf{z}(k+1) = \mathbf{\Phi}\mathbf{z}(k) + \Delta t \mathbf{B}_c(\mathbf{\Phi}\mathbf{z}(k) + \overline{\mathbf{x}})\mathbf{u}(k). \tag{22}$$

In our implementation, we successfully used

$$\underbrace{\mathbf{\Phi}^T[\mathbf{I} - \Delta t \mathbf{A}_c(\mathbf{\Phi}\mathbf{z}(k) + \overline{\mathbf{x}})]\mathbf{\Phi}}_{lxl}\mathbf{z}(k+1) = \mathbf{z}(k) + \mathbf{\Phi}^T \Delta t \mathbf{B}_c(\mathbf{\Phi}\mathbf{z}(k) + \overline{\mathbf{x}})\mathbf{u}(k), \tag{23}$$

which is obtained from equation (22) by premultiplying with $\mathbf{\Phi}^T$. The number of state variables is thus reduced from $n = 2\overline{m}\overline{n}$ to $l = l_p + l_s$. Note that we now use the variable l to indicate the *total* number of reduced state variables. The matrix dimensions for the total system are consequently reduced from $n \times n$ to $l \times l$. The simulation time of the reduced-order model using semi-implicit discretization is decreased because we have to solve l equations instead of n equations in the full-order model, where $l << n$. For fully implicit simulation, where more than one system of equations has to be solved during every time step, the decrease in simulation time is expected to be even higher. Unfortunately, the original pentadiagonal matrix structure (heptadiagonal for three-dimensional systems) is changed to a full matrix, because we multiply the pentadiagonal matrix with a full matrix $\mathbf{\Phi}$ from the left side and from the right side. This counteracts the computational advantage obtained by reducing the size of the state vector. When we simulated reduced-order reservoir models with the same controls as the original full-order models, we obtained almost identical states, as long as a sufficient fraction of the relative energy of the full-order model was preserved. However, if we strongly altered the controls, and therefore the structures of the states, the states of the full-order model were less well represented by the reduced-order model. This is in line with the findings of other authors [13]. Because it is not possible to specify a priori the validity of a reduced-order model, we will use a nested approach in the development of the optimization methodology below, such that the reduced-order results are frequently validated by the full-order model.

5. Reduced-order optimal control

Adjoint-based OCT is an effective technique to optimize the settings of control variables $\mathbf{u}(k)$ (e.g., valve positions or flow rates) over the life of the reservoir in order to maximize an objective function $J = \sum_{k=1}^{K} J_k(\mathbf{x}(k), \mathbf{u}(k))$, [1,3,15,18,20]. The objective function typically represents ultimate recovery or a 'simple net-present value (NPV),' i.e., the sum of the incremental discounted oil production income and water injection and production costs over the life of the reservoir [3]. OCT is a gradient-based optimization technique, where the gradients are obtained with the aid of an adjoint equation in terms of Lagrange multipliers $\boldsymbol{\lambda}$. The multipliers represent the objective function's sensitivities to changes in the state variables and originate from adding the dynamic system as a constraint to the objective function. In our application, the controls are formed by the injection and production rates in the smart well segments at every time step. Following the derivation in [3], the adjoint equation can be written as in discrete time as

$$\boldsymbol{\lambda}(k)^T \left(\frac{\partial \mathbf{g}(k-1)}{\partial \mathbf{x}(k)} \right) = \left[-\boldsymbol{\lambda}(k+1)^T \frac{\partial \boldsymbol{g}(k)}{\partial \mathbf{x}(k)} - \frac{\partial J_k(k)}{\partial \mathbf{x}(k)} \right], \tag{24}$$

where

$$\mathbf{g}(k) = \mathbf{x}(k+1) - \mathbf{f}_d(\mathbf{x}(k), \mathbf{u}(k+1)) = \mathbf{0} \tag{25}$$

is a compact representation of the system equation (5). For our implementation, instead of the full-order model, we added the reduced-order model as a constraint to the objective function J with the aid of a set of low-order Lagrange multipliers $\boldsymbol{\mu}$:

$$\overline{J}_{red} = \sum_{k=0}^{K-1} \left[J_k(\boldsymbol{\Phi}\mathbf{z}(k), \mathbf{u}(k)) + \boldsymbol{\mu}(k+1)^T \boldsymbol{\Phi}^T \mathbf{g}(\boldsymbol{\Phi}\mathbf{z}(k+1), \boldsymbol{\Phi}\mathbf{z}(k), \mathbf{u}(k)) \right], \tag{26}$$

where we note that we need to add $\overline{\mathbf{x}}$ to each product $\boldsymbol{\Phi}\mathbf{z}$, in line with equation (20). Taking the first variation of equation (26), and reworking the results, we obtain a reduced-order equation in terms of reduced-order Lagrange multipliers:

$$\boldsymbol{\mu}(k)^T \underbrace{\boldsymbol{\Phi}^T \frac{\partial \mathbf{g}(k-1)}{\partial \mathbf{x}(k)} \boldsymbol{\Phi}}_{lxl} = \left[-\boldsymbol{\mu}(k+1)^T \underbrace{\boldsymbol{\Phi}^T \frac{\partial \mathbf{g}(k)}{\partial \mathbf{x}(k)} \boldsymbol{\Phi}}_{lxl} - \underbrace{\frac{\partial J_k(k)}{\partial \mathbf{x}(k)} \boldsymbol{\Phi}}_{1xl} \right], \tag{27}$$

Starting from the final condition $\boldsymbol{\mu}(K) = \mathbf{0}$ it can be integrated backward in time. Because the derivatives in equation (27) consist of state-dependent parameters, we first calculate the full-order derivatives. They are then transformed and reduced by projecting them on the axes of the low-order model. After calculating $\boldsymbol{\mu}$ every time step, we can calculate:

$$\frac{\partial \mathcal{L}(k)}{\partial \mathbf{u}(k)} = \frac{\partial J_k(k)}{\partial \mathbf{u}(k)} + \boldsymbol{\mu}(k+1)^T \left\{ \boldsymbol{\Phi}^T \frac{\partial \mathbf{g}(k)}{\partial \mathbf{u}(k)} \right\}. \tag{28}$$

We compute improved controls using a steepest ascend method according to $\mathbf{u}_{new} = \mathbf{u}_{old} + \varepsilon \partial \mathcal{L}(k) / \partial \mathbf{u}(k)$ where ε is a weight factor [3]. The computational advantage of using reduced-order models in OCT is that the system of equations involves only l unknowns, whereas the original system involved $n = 2\overline{m}\overline{n}$ unknowns. This decreases the simulation time considerably, especially for large systems where $l << n$. Unfortunately, the original block pentadiagonal matrix structure of $\partial \mathbf{g}(k-1)/\partial \mathbf{x}(k)$ and the block-diagonal matrix structure of $\partial \mathbf{g}(k)/\partial \mathbf{x}(k)$ are changed to full matrices, because we multiply them with full matrices $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi}^T$. This counteracts the computational advantage obtained by using reduced-order optimal control.

6. Methodology

The implementation of the full-order OCT algorithm for water flooding was described in [3]. In reduced-order optimal control based on POD (see figure 1), we first

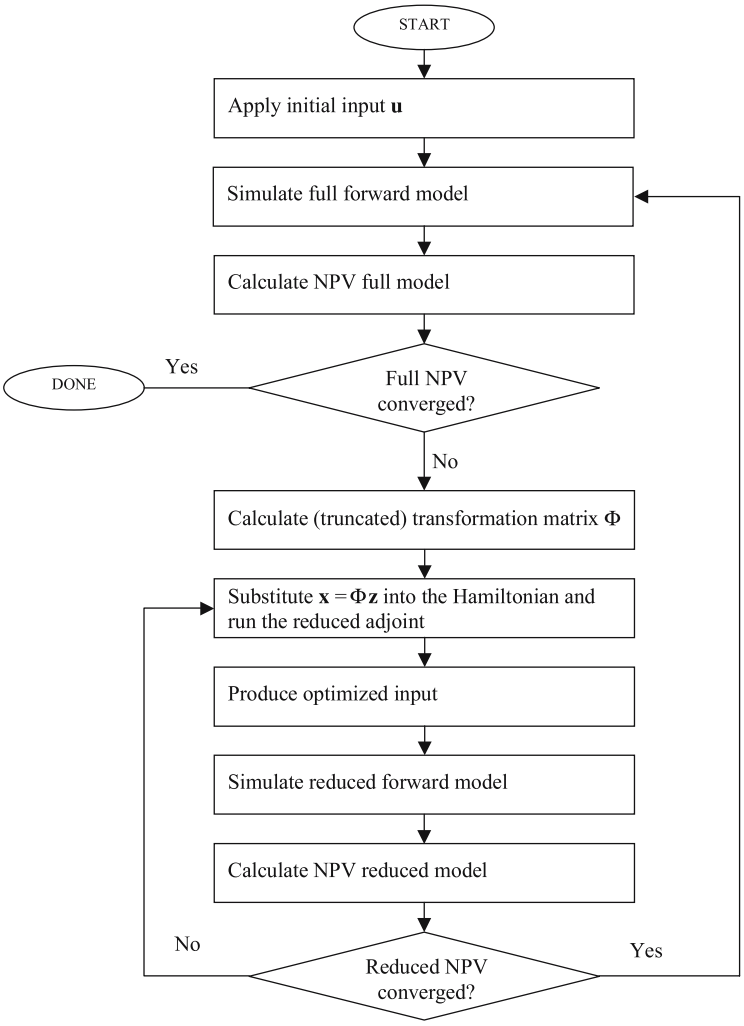

Figure 1. Flow chart for reduced-order OCT for water flooding.

simulate the dynamical behavior of the system over time interval 0 to K with an initial choice of $\mathbf{u}$ and compute the NPV. Following [1], the initial choice of $\mathbf{u}$ reflects a flooding strategy with conventional horizontal wells with constant pressure along the well bores. Every time step, we record and store a total of κ snapshots of pressures and saturations and calculate POD transformation matrices $\mathbf{\Phi}$. Now, instead of using the full-order derivatives of the system, we use the reduced-order derivatives for the backward calculation and calculate $\boldsymbol{\mu}$ with equation (27). Based on the derivatives computed with equation (28), we compute new controls and use them for the next reduced-order forward simulation. For this simulation, we use the same transformation

matrices $\mathbf{\Phi}$. This means that the computational 'overhead' of calculating $\mathbf{\Phi}$ is shared by multiple runs of the reduced-order model. To determine convergence of the inner loop, we use a convergence criterion c. The inner loop has converged when the NPV of a reduced-order forward simulation is less than c times the NPV of the previous reduced-order simulation. Convergence of the inner loop may occur because a local maximum of the NPV has been reached or because the controls have changed too much to be accurately captured in the reduced system representation. Entering the outer loop again, we use the improved controls in a full-order forward simulation and verify if the controls have indeed maximized the NPV. If necessary, the transformation matrices $\mathbf{\Phi}$ are replaced with new ones that reflect the altered dynamics, and the inner loop is repeated. The outer loop has converged when the NPV of the full-order forward simulation is less than the NPV of the previous full-order simulation.

We implemented the methodology in a MATLAB algorithm. The advantage of the methodology is that we use reduced-order forward simulations and reduced-order optimal control, which have a shorter simulation time. A disadvantage is that an improved control of the reduced-order model is not necessarily an improved control for the full-order model. In the numerical example below, we will see that in our example, this is, however, not a problem. Assessment of the robustness of this approach requires further research on more realistic reservoir models.

7. Numerical example

To test the methodology, we used a two-dimensional model with 2025 (45×45) grid blocks, which is the same model as used in [3,11,15,16]. The dimensions of the reservoir were $450 \times 450 \times 10$ m, and the permeability field is shown in figure 2. Initially, the reservoir is completely saturated with oil. We assigned liquid compressibilities of 1×10^{-10} Pa^{-1} to both water and oil. At the left side of the reservoir, we introduced one horizontal water injector divided in 45 segments by interval control valves. At the right side, we introduced one horizontal producer, also divided in 45 segments. The controls are therefore formed by the 90 injection and production rates in the smart well segments at every time step. The objective function represents a simple NPV, defined as the sum of the incremental discounted oil production income and water production costs over the life of the reservoir. The wells were operated without well model under rate constraint: the liquid rate in an injection segment was equal to the water rate, and in a production segment, the rate was equal to the sum of water rate and oil rate. The total production and injection rates were equal to each other during the entire simulation time. During optimization, the flow rates were redistributed maintaining a constant total sum of the rates. In the NPV calculation, we used an oil price $r_o = \$80/\mathrm{m}^3$ and a produced water cost $r_w = \$20/\mathrm{m}^3$. We compared the NPV obtained with the reduced-order and full-order optimal control algorithms with the NPV of a reference case. In the reference case, the injection and production rates were constant over time and a function of water and oil

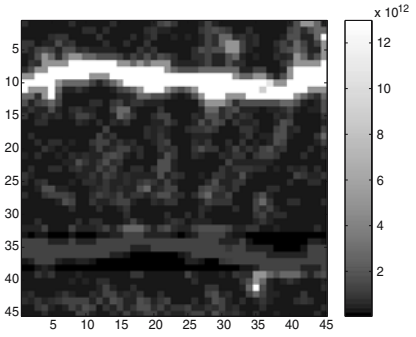

Figure 2. Permeability field (m^2).

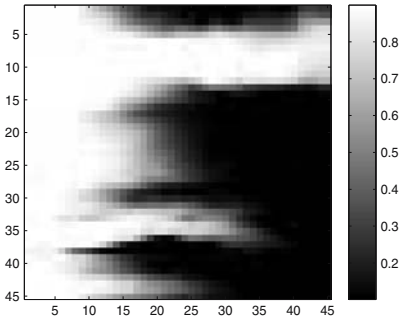

Figure 3. Final water saturation after 1 PV production for the reference case. *Dark*: oil; *light*: water.

mobility, reflecting a conventional water flood where the wells are operated at constant bottom hole pressure. We simulated the reservoir model for 949 days with variable time step size, and in this period, we injected and produced one pore volume of liquid. We obtained a saturation distribution as depicted in figure 3. The total NPV for the reference case is \$10.1 million.

7.1. Full-order optimal control example

Starting from the reference case, we ran the full-order control algorithm. With a 2.4-GHz Pentium 4 processor and 1-GB RAM memory, it took 8701 s (148 min) to run the full-order algorithm. We reached convergence after 19 full-order forward simulations and 18 full-order backward simulations. The average simulation time for the full-order forward simulation was 144 s and, for the full-order backward simulation, was 266 s. The resulting optimized rates are given in the left and the middle pictures of figure 4, which correspond to a final oil–water saturation distribution as depicted in the right picture of figure 4. The corresponding NPV versus the number of iterations has been plotted in figure 5. It can be seen that the

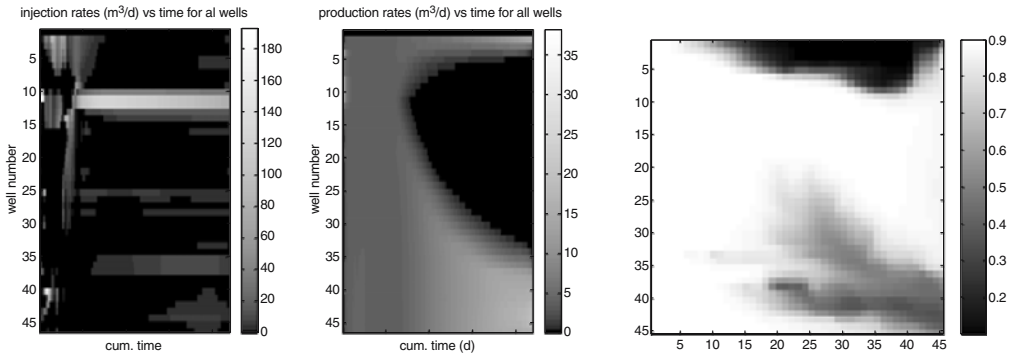

Figure 4. Optimized injection rates (*left*) and production rates (*middle*) in m^3/d vs. the simulation time, calculated with the full-order control algorithm. At the right the resulting water saturation distribution. *Dark*: oil; *light*: water.

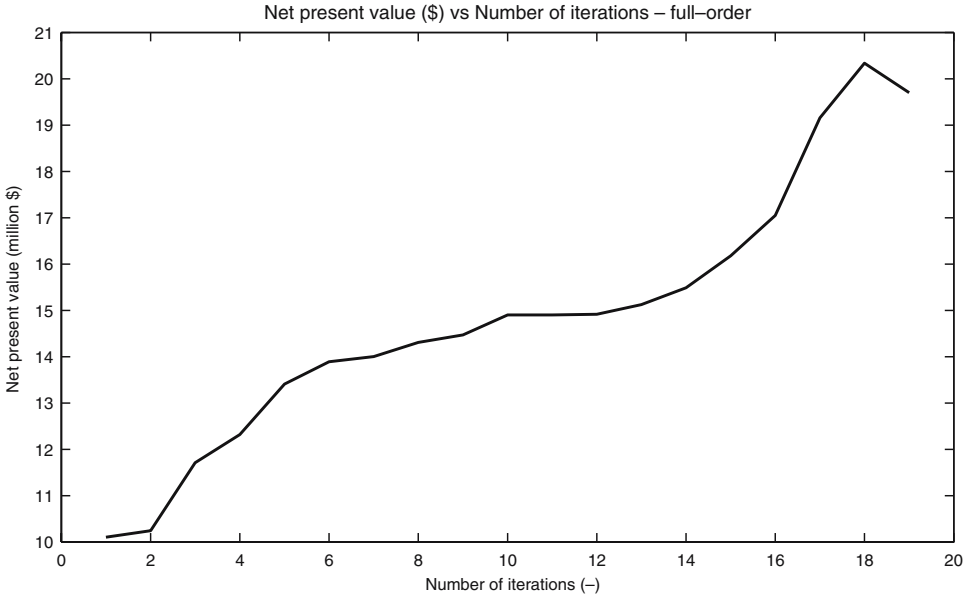

Figure 5. NPV vs. number of iterations in full-order control algorithm.

NPV first stabilizes around \$15 million and later increases further to its maximum value of \$20.3 million. The NPV of the last iteration is slightly less than the previous one. When we continued with the algorithm, we did not observe a further increase in NPV. Because our optimization procedure is a local one, the maximum NPV should be regarded as a lower bound of the possible improvements.

7.2. Reduced-order optimal control example

For reduced-order optimal control, we used the algorithm as described in section 6. With an energy level of 0.999 as cut-off criterion for POD, we obtained a final NPV of $19.3 million, which is an increase of 87% with respect to the reference case. The maximum NPV obtained with reduced-order control approached the NPV obtained with full-order optimal control to within 95%. We reached convergence in 5661 s, which is a reduction with 35% of the time used for the full-order optimal control. In this example, we needed 10 full-order forward simulations and 13 reduced-order forward simulations. In the first iterations, matrices $\mathbf{\Phi}$ were shared two to four times before the inner loop converged and matrices $\mathbf{\Phi}$ were updated. The reduction in simulation time for the reduced-order forward simulation was 34% and, for the reduced-order backward model, was 38%. In order to maintain an energy level of 0.999, we need for the reference case in total 30 POD basis functions. The number of POD basis functions gradually increased when we used improved controls, and for the optimal case, we used 49 POD basis functions. This speaks in favor of our nested approach where we adapt the transformation matrix after a full-order forward simulation. Table 1 shows the results in more detail. In figure 6, the resulting rates for this case are given in the left and the middle, which correspond to a final oil–water saturation distribution as shown in the right. The resulting rates and the final saturation distribution obtained with reduced-order optimal control differed from the resulting rates and final saturation distribution obtained with full-order optimal control. Apparently, we ended up in two different optima.

When we specified a lower energy level of 0.99, we required less POD basis functions, and the simulation time for the reduced-order simulations therefore decreased. We needed, however, more simulations in order to converge. Conversely, when we increased the energy level to 0.9999, we required a larger number of basis functions and the simulation time increased. The average simulation time for the full-order forward simulation was, in the latter case, almost equal to the average simulation time for the reduced-order forward model. The reduced-order backward simulations were still faster. Figure 7 displays the NPV versus the number of iterations in the full-order control algorithm and the reduced-order algorithm using energy levels of 0.99, 0.999 and 0.9999. The NPV obtained with reduced-order optimal control increased steeper in the beginning than the NPV obtained with full-order optimal control. After two full-order forward simulations using an energy level of 0.999, we already reached an increase in NPV of 72%. After four full-order simulations, the NPV slowly converged to its maximum. Changing the convergence criterion c of the inner loop from 1.01 to 1.10, we observed no significant influence on the maximum NPV (see table 1). Also, the total simulation times were almost identical. Another parameter that we varied was the number of time steps between updating the parameter-dependent matrices in the inner loop ($\mathbf{A}_d$ and $\mathbf{B}_d$ of the forward equation and the derivative matrices of the adjoint equation). Figure 8 displays the NPV versus the number of

Table 1
Results of full-order and reduced-order control algorithms for different energy levels, convergence criteria and number of time steps between updating the parameter-dependent matrices in the inner loop.

	Simulation time		Forward full		Forward reduced		Adjoint full		Adjoint reduced		No. of POD basis functions reference			No. of POD basis functions optimal			Final NPV $
	s	h	No. of calls	s/call	No. of calls	s/call	No. of calls	s/call	No. of calls	s/call	S_w	p	Total	S_w	p	Total	
Energy level = 0.99	5809	1.61	10	124	15	87	0	0	15	151	17	2	19	14	7	21	19,479,000
Energy level = 0.999	5661	1.57	9	151	13	100	0	0	13	164	28	2	30	28	13	41	19,315,000
Energy level = 0.9999	6746	1.87	9	129	14	131	0	0	14	200	42	2	44	58	25	83	19,371,000
Inner convergence = 1.01	5477	1.52	9	126	13	100	0	0	13	166	28	2	30	28	13	41	19,315,000
Inner convergence = 1.05	5661	1.57	9	151	13	100	0	0	13	164	28	2	30	28	13	41	19,315,000
Inner convergence = 1.10	5592	1.55	9	128	13	103	0	0	13	170	28	2	30	28	13	41	19,315,000
Constant time step = 1	5661	1.57	9	151	13	100	0	0	13	164	28	2	30	18	13	31	19,315,000
Constant time step = 2	5855	1.63	13	126	17	74	0	0	17	104	28	2	30	29	11	40	18,781,000
Constant time step = 3	4923	1.37	11	128	16	69	0	0	16	82	28	2	30	40	14	54	18,858,000
Constant time step = 4	5972	1.66	9	126	19	98	0	0	19	92	28	2	30	75	28	103	18,110,000
Full-order optimal control	8702	2.42	19	144	0	0	18	266	0	0	–	–	–	–	–	–	20,335,000
Full forward reduced backward	3774	1.05	11	130	0	0	0	0	10	165	28	2	30	28	11	39	19,139,000

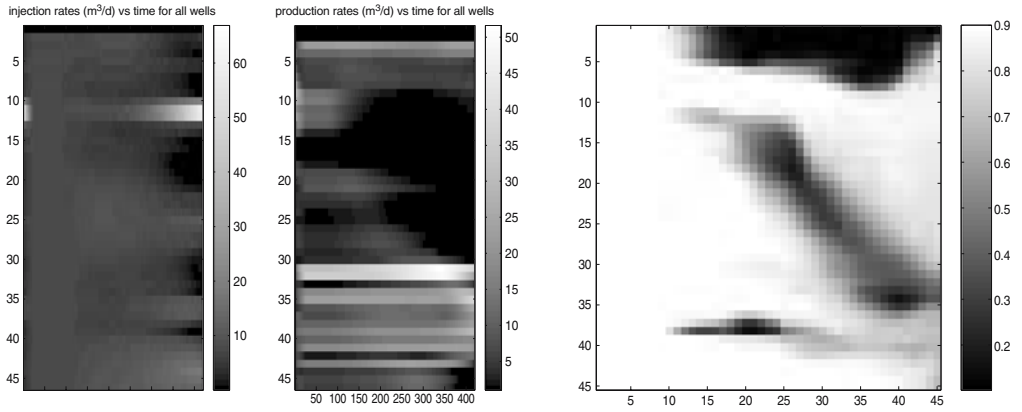

Figure 6. Optimized injection rates (*left*) and production rates (*middle*) in m^3/d vs. simulation time calculated with reduced-order control algorithm and energy level 0.999. At the right the resulting water saturation distribution. *Dark*: oil; *light*: water.

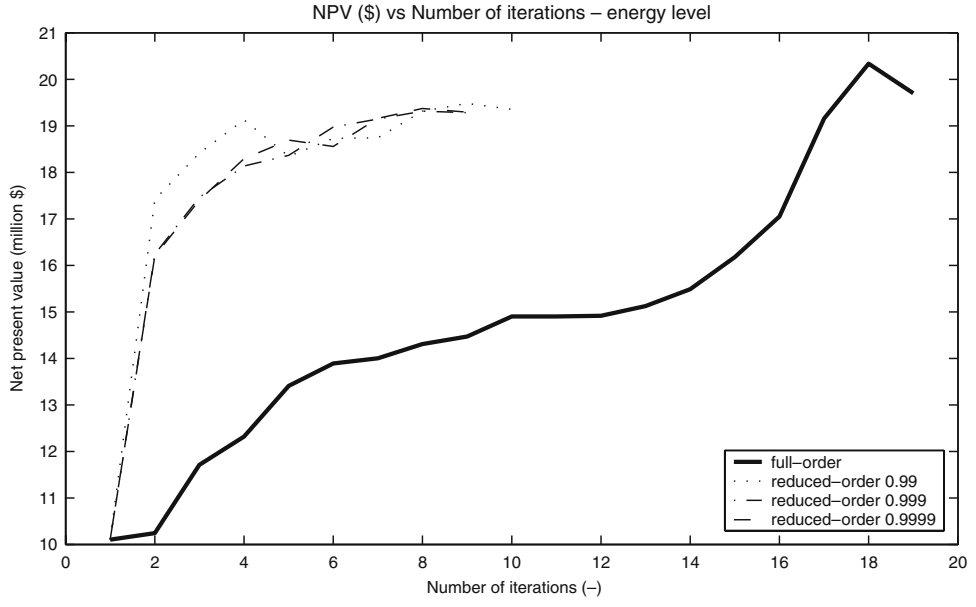

Figure 7. NPV vs. number of iterations in the reduced-order control algorithm, for different energy levels.

iterations using 1, 2, 3 and 4 time steps between the updates. Again, we see a major increase of the NPV after two full-order forward simulations and a subsequent slow converge to the maximum. By not updating the matrices every time step, we introduce an error in calculating the improved controls. This is evident from the seventh and

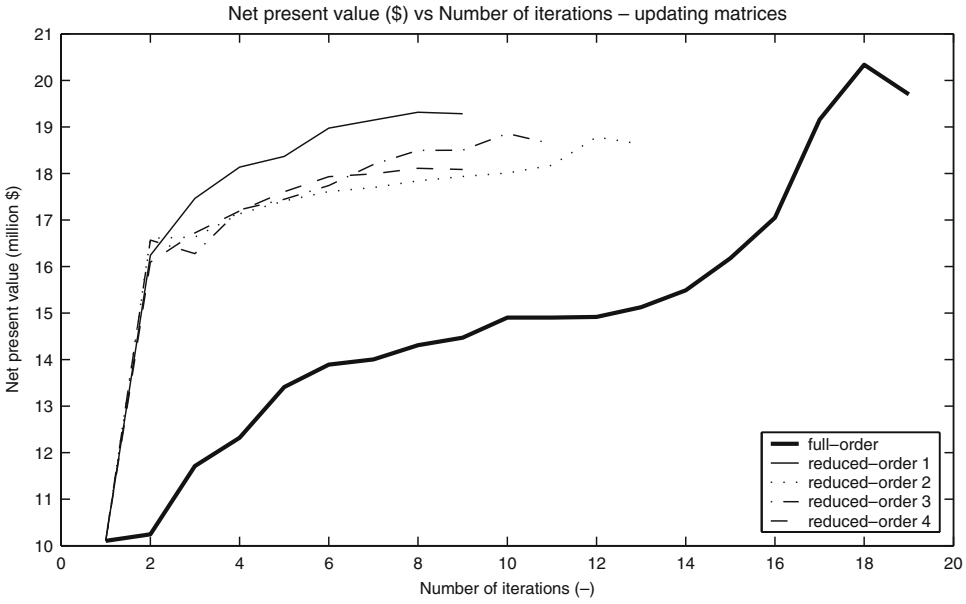

Figure 8. NPV versus number of iterations in the reduced-order control algorithm, for different numbers of time steps between the updating the parameter-dependent matrices.

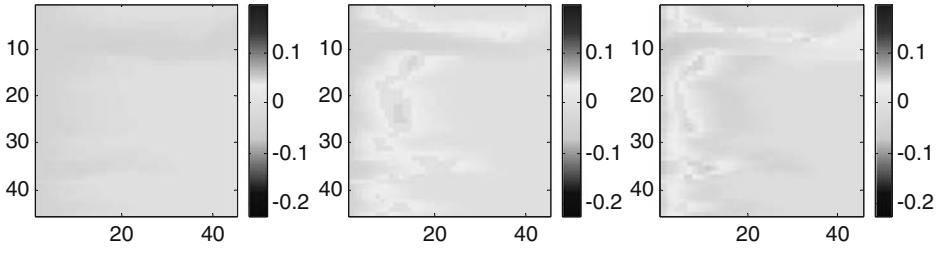

Figure 9. Example of the first three saturation POD basis functions projected on a 45 × 45 matrix.

eighth columns of table 1, which represent the number of retained basis functions, for a given energy level, during the first and the last iteration in the inner loop, respectively. It can be observed that not updating the parameter-dependent matrices results in the need to use more basis functions in the later iterations. This effect is probably caused by spurious dynamics resulting from the more abrupt changes in the system parameters in case of less frequent updating.

7.3. *Energy level, number of snapshots and number of grid blocks*

Figure 9 depicts a physical interpretation of the saturation POD basis vectors for our example of a 45 × 45 reservoir model. Because the vectors have a length equal to

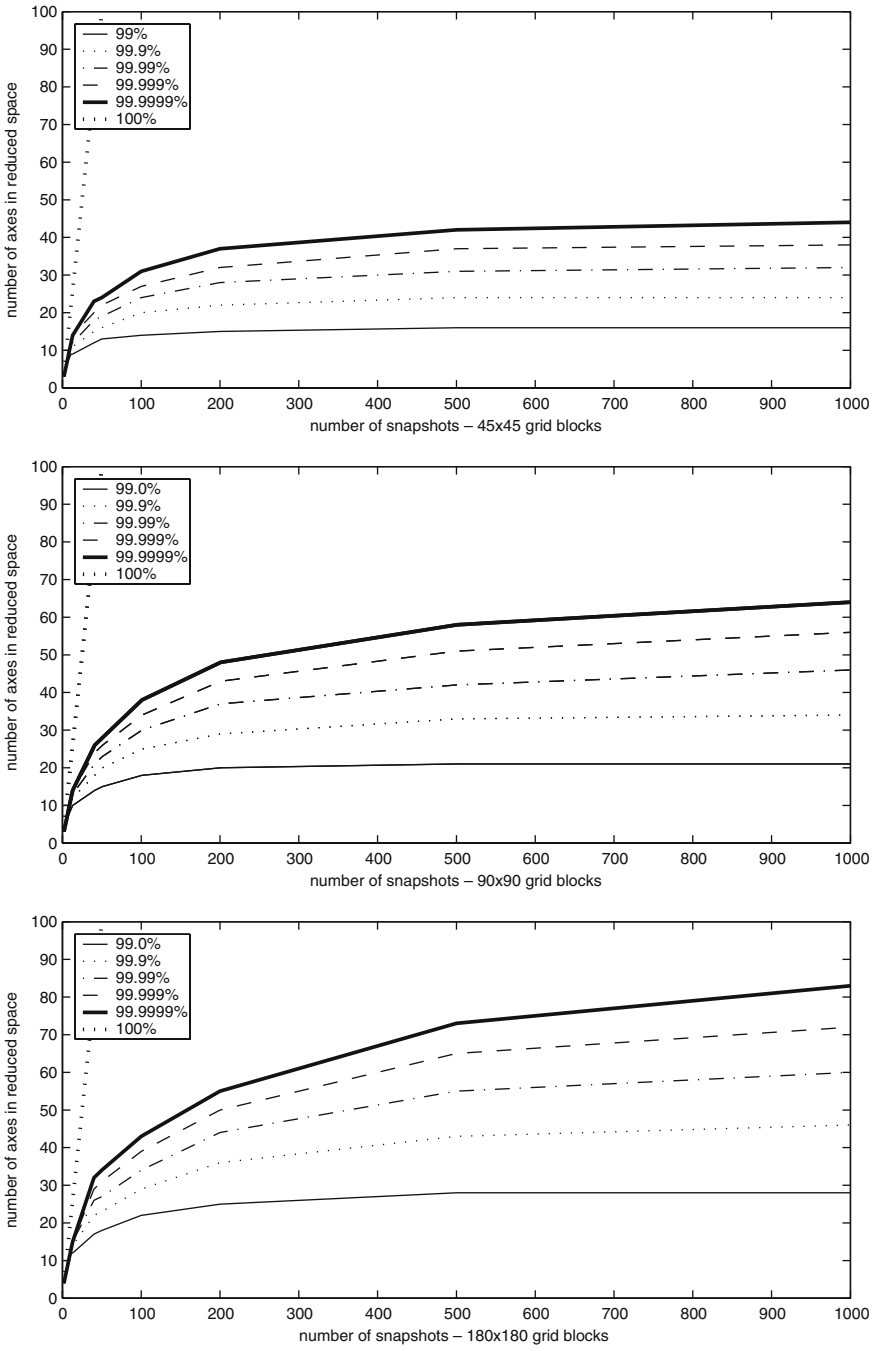

Figure 10. The three graphs represent the number of POD basis functions required to preserve a certain energy level, as a function of the number of snapshots, for a 45 $\times$ 45 reservoir model (*top*), a 90 $\times$ 90 reservoir model (*middle*) and a 180 $\times$ 180 reservoir model (*bottom*).

the total number of grid blocks, we can reshape and project them on a 45×45 matrix. In the figure, this is done for the first three vectors, and we can recognize the spatial characteristics of the permeability field (depicted in figure 2) and of the oil–water front, which are both represented in the snapshots.

In figure 10, we present three graphs that display the number of POD basis functions l versus the number of snapshots used in the calculation of $\mathbf{\Phi}$. For the first graph, we simulated the reference case and generated a set of 1000 snapshots at identical time intervals. For the second graph, we used the same case, except that each grid block was divided in four grid blocks with identical permeabilities resulting in a 90×90 reservoir model. Also, for this mode, we generated a set of 1000 snapshots at identical time intervals. For the 180×180 reservoir model, we used the same procedure, which resulted in a reservoir model with 32,400 grid blocks. We varied the number of snapshots by choosing from the set of 1000 snapshots a subset consisting of, respectively, 2, 3, 5, 9, 11, 13, 41, 51, 101, 201, 501 and 1000 snapshots at identical time intervals. The figures illustrate how an increasing number of retained basis functions corresponds to an increasing energy level. We needed an increasing number of POD basis functions when we increased the number of snapshots for a given energy level, which indicates that added snapshots are not a linear combination of the earlier snapshots, i.e., that they contain new information. More interestingly, we can conclude that the number of POD basis functions l increases with the number of grid blocks in a non-linear fashion: when the number of grid blocks is multiplied by 16, the number of POD functions at an energy level of 99.99% for 1000 snapshots is only multiplied by 1.87. This illustrates that the reduced-order representation mainly represents the dominant structures present in the snapshots. Because we did not change the permeability field, but merely used a larger number of grid blocks to describe it, the dominant structures in the dynamics of the state variables also did not change very much. The small increase in basis functions captures some increased detail at the boundaries of the dominant structures. This implies that for a reservoir model with dominant large-scale geological features, the computational efficiency of reduced-order simulation will increase with an increasing model size. For a model that lacks clear dominant structures, e.g. one that has heterogeneities with a small correlation length, this increase in computational efficiency is less pronounced or even absent.

8. Conclusion

In the example discussed, we found that reduced-order optimal control of water flooding using POD improved the NPV with respect to an uncontrolled reference case. Within a shorter simulation time, the NPV obtained by the full-order optimal control algorithm was approached closely by the NPV obtained by the reduced-order algorithm. The increase in computational efficiency was achieved by reducing the number of states in the forward and backward simulations considerably and,

consequently, the number of equations that needed to be solved every time step. Considering a reservoir model with 4050 states (2025 pressures, 2025 saturations) and an adjoint model of 4050 states (Lagrange multipliers), we obtained reduced-order models with 20–100 states only. The NPV obtained by reduced-order optimal control was approached to within 95% of the NPV obtained by full-order optimal control. The resulting reduction in computing time was 35%. In general, the number of POD basis functions preserving a certain fixed level of relative energy increases during optimization, which speaks in favor of our nested reduced-order optimal control algorithm where we adapt the transformation matrix after simulating the full-order reservoir model with improved controls.

Nomenclature

$\mathbf{A}$	system matrix
$\mathbf{B}$	input matrix
c	convergence criterion in inner loop
E	relative energy present in snapshots
$\mathbf{f}$	nonlinear system function vector
$\mathbf{g}$	nonlinear system function vector
$\mathbf{I}$	unit matrix
J	objective function ($)
k	discrete time step counter
K	total number of time steps
l	reduced system order
$\mathcal{L}$	Lagrangian ($)
$\overline{m}$	number of grid blocks in $\mathbf{x}$ direction
n	system order (number of state variables)
$\overline{n}$	number of grid blocks in y direction
N	number of wells
p	pressure (Pa)
$\mathbf{P}$	$n \times n$ matrix of right eigenvectors
q	flow rate (m^3/s)
r	price per unit volume ($/$m^3$)
$\mathbf{r}$	residual vector

$\mathbf{R}$	covariance matrix
t	time (s)
S	saturation
$\mathbf{T}$	transmissibility matrix
$\mathbf{u}$	input vector
$\mathbf{V}$	mass matrix
$\mathbf{W}$	compressibility/porosity matrix
$\mathbf{x}$	state vector
$\mathbf{X}$	data matrix, containing state vectors
z	transformed state variable
$\mathbf{z}$	transformed state vector
α	fraction of relative energy
ε	weighting factor
κ	number of snapshots in POD
λ	eigenvalue
$\boldsymbol{\lambda}$	vector of Lagrange multipliers
$\boldsymbol{\mu}$	vector of reduced-order Lagrange multipliers
σ	singular value
$\boldsymbol{\Sigma}$	diagonal matrix of singular values
$\boldsymbol{\varphi}$	right eigenvector
$\boldsymbol{\Phi}$	matrix of right eigenvectors
$\boldsymbol{\Phi}_l$	truncated matrix of right eigenvectors
$\boldsymbol{\psi}$	left eigenvector
$\boldsymbol{\Psi}$	matrix of left eigenvectors

Subscripts

c	continuous
cow	oil–water capillary pressure
d	discrete
i	counter

l	liquid
o	oil
p	pressure
s	saturation
w	water

References

[1] H. Asheim, Maximization of water sweep efficiency by controlling production and injection rates, paper SPE 18365, in: *Proc. SPE European Petroleum Conference*, London, UK (1988).

[2] K. Aziz and A. Settari, *Petroleum Reservoir Simulation* (Applied Science Publishers, London, UK, 1979).

[3] D.R. Brouwer and J.D. Jansen, Dynamic optimisation of water flooding with smart wells using optimal control theory, SPE J. (2004) 391–402, December.

[4] G.H. Golub and C. Van Loan, *Matrix Computations*, 3rd Edition (John Hopkins Univ. Press, Baltimore, MD, 1983).

[5] T. Heijn, R. Markovinović and J.D. Jansen, Generation of low-order reservoir models using system-theoretical concepts, paper SPE 79674, in: *Proc. Reservoir Simulation Symposium*, Houston, TX, USA, 3–5 February (2003).

[6] P. Holmes, J.L. Lumley and G. Berkooz, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry* (Cambridge University Press, Cambridge, 1996).

[7] http://www.mathworks.com.

[8] H.V. Ly and H.T. Tran, Modeling and control of physical processes using proper orthogonal decomposition, Math. Comput. Model. 33 (2001) 223–236.

[9] R. Markovinović, E.L. Geurtsen, T. Heijn and J.D. Jansen, Generation of low-order reservoir models using POD, empirical gramians and subspace identification, in: *Proc. 8th European Conf. on the Mathematics of Oil Recovery (ECMOR VIII)*, Freiberg, Germany, E31 (2002) pp. 1–10.

[10] M. Meyer and H.G. Matthies, Efficient model reduction in non-linear dynamics using the Karhunen–Loève expansion and dual-weighted-residual methods, Comput. Mech. (2003) 179–191.

[11] G. Naevdal, D.R. Brouwer and J.D. Jansen, Water flooding using closed-loop control, Comput. Geosci. (2006), DOI: 10.1007/s10596-005-9010-6.

[12] D.W. Peaceman, *Fundamentals of Numerical Reservoir Simulation* (Elsevier Scientific Publishing Company, Amsterdam, The Netherlands, 1977).

[13] R.D. Prabhu, S.S. Collis and Y. Chang, The influence of control on proper orthogonal decomposition of wall-bounded turbulent flows, Phys. Fluids 13(2) (2001) 520–537.

[14] S.S. Ravindran, Reduced-order adaptive controllers for fluid flows using POD, J. Sci. Comput. 14(4) (2000) 457–478.

[15] P. Sarma, K. Aziz and L.J. Durlofsky, Implementation of adjoint solution for optimal control of smart wells, paper SPE 92864, in: *Proc. SPE Reservoir Simulation Symposium*, Houston, TX, USA (2005).

[16] P. Sarma, L.J. Durlofsky, K. Aziz and W.H. Chen, Efficient real-time reservoir management using adjoint-based optimal control and model updating, Comput. Geosci. (2006), DOI: 10.1007/s10596-005-9009-z.

[17] L. Sirovich, Turbulence and the dynamics of coherent structures. Part 1: Coherent structures, Q. Appl. Math. 45(3) (1987) 561–571.
[18] B. Sudaryanto and Y.C. Yortsos, Optimization of fluid front dynamics in porous media using rate control. I. Equal mobility fluids, Phys. Fluids 12(7) (2000) 1656–1670.
[19] P.T.M. Vermeulen, A.W. Heemink and C.B.M. Te Stroet, Reduced models for linear groundwater flow models using empirical orthogonal functions, Adv. Water Resour. 27 (2004) 54–69.
[20] G.A. Virnovski, Water flooding strategy design using optimal control theory, in: *Proc. 6th European Symposium on IOR*, Stavanger, Norway (1991) pp. 437–446.