Optim Eng (2015) 16:441–481
DOI 10.1007/s11081-014-9268-4

Adjoint-based surrogate optimization of oil reservoir water flooding

Eka Suwartadi · Stein Krogstad · Bjarne Foss

Received: 26 May 2012 / Accepted: 19 October 2014 / Published online: 16 November 2014
© Springer Science+Business Media New York 2014

Abstract Maximizing economical asset of oil reservoirs is a simulation-based optimization involving large-scale simulation models. In this work we propose the use of reduced-order models for solving optimization problems in oil reservoir simulation using a Lagrangian barrier method for the treatment of nonlinear inequality constraints. The optimization with reduced-order models is done by employing a trust-region proper orthogonal decomposition (TRPOD) algorithm. In addition to the POD method, we also build a reduced-order model based on a discrete empirical interpolation method. In the algorithm, the first-order gradient of the objective function is computed by using the adjoint method, while the inverse of the second-order gradient is approximated using the BFGS method. The reduced-order models involve both the forward (state) and backward (adjoint) equations. Three optimization case examples in production optimization of oil reservoirs are used to study the method. They show that the TRPOD method works efficiently while simultaneously honoring the nonlinear constraints.

Keywords Model order reduction · Adjoint method · Nonlinear constraints · Production optimization · Oil reservoirs

Mathematics Subject Classification MSC 90C30 · MSC 49M37 · MSC 49J20

E. Suwartadi (✉) · B. Foss
Department of Engineering, Cybernetics Norwegian University Science and Technology (NTNU), 7491 Trondheim, Norway
e-mail: eka.suwartadi@ieee.org

S. Krogstad
Department of Applied Mathematics, SINTEF ICT, Blindern, Oslo, Norway

1 Introduction

As the rise of human population and the modernization of civilization, the need for fossil fuel energy has been increasing. To fulfill this demand, maximizing oil production from an oil reservoir is a challenging task. An oil reservoir is usually modeled by partial differential equations (PDEs), where the geological model is in the order of 10^6–10^9 discretized grids. The geological model is developed by geologists which is then further used by reservoir engineers. In this paper we focus on the production optimization of oil reservoirs with emphasis on water flooding which is typically handled by the reservoir engineers. The injected water aims to sweep remaining oil efficiently. With the current state-of-the-art computing power, reservoir simulation models are usually reduced to the order between 10^4 and 10^6 grid blocks. This process is known as *upscaling* since it creates a coarse model from the geological model. Upscaling is done based on geophysical interpretation by the reservoir engineers. This involves heuristics and can be a time-consuming runtime simulation process both with regards to computations and human labour (see e.g., Aarnes et al. (2007)).

Model order reduction techniques can be used to facilitate the upscaling process. The use of model order reduction techniques has been around in the reservoir simulation research since early 2000, see e.g., Markovinovic et al. (2002) and Markovinovic et al. (2002). The work of Heijn et al. (2004) compared methods for reduced-order modeling which treat oil reservoirs both as linear and nonlinear models. The methods originate from systems and control theory. Balanced truncation, subspace identification, and proper orthogonal decomposition (POD) methods were compared. Based on the examples considered in Heijn et al. (2004), the conclusion was that the POD method gave the best approximation of the oil reservoir dynamics. In follow up work Doren et al. (2006), Markovinovic and Jansen (2006), the POD method was used for gradient-based production optimization. The adjoint equations were derived using reduced-order models of the state equations. The POD method generates reduced-order models with global basis functions. Another approach was presented in Krogstad et al. (2011) where a multiscale method was applied to compute local basis functions. In that work an optimization problem using a real geometry of an oil reservoir was solved in 15 minutes, compared to a normal length of hours or even days. In a more recent work a combination of multiscale and POD methods was presented in Krogstad (2011), yielding a local POD basis function. The selection of local basis functions in a multiscale method is done by considering physical aspects such as fault locations and flux boundaries, which is more intuitive to the reservoir engineers since the reservoir model is divided into some coarsened segments, where each of the segment has its own local basis function.

The use of the trajectory-piecewise-linearization (TPWL) method, which models the oil reservoir as a linear time varying (LTV) system along selected operating points, was proposed in Cardoso and Durlofsky (2010). The same authors also proposed the use of missing point estimation (MPE)-POD (Astrid et al. (2008)) in Cardoso et al. (2009) and further used the TPWL-based model order reduction for production optimization in Cardoso and Durlofsky (2010). Two optimization methods were presented in Cardoso and Durlofsky (2010). These were the gradient-based and generalized pattern search methods. None of the production optimization papers mentioned above discuss the state or nonlinear output constraint problem. In more recent work, the use of approximate dynamic programming combined with POD method was proposed in Wen et al. (2011). This work used the penalty method to handle the state constraint problem.

In other areas, the POD method has been used for constraints handling in low-fidelity model optimization. Among these, the trust-region POD (TRPOD), originally proposed in Fahl (2000) for unconstrained optimization problems, was further developed for constrained optimization. The idea is that the POD method gives a good approximation of the high-fidelity model by updating POD basis functions in limited (or "*trusted*") operating points. During the course of optimization the decision variables are always changed, therefore the POD basis functions need to be updated using the new update of decision variables. Without this updating, the POD basis functions represent the previous/old decision variables, which are no longer valid and give a poor approximation of the high-fidelity model. The constrained TRPOD, which means optimization using reduced-order models in the presence of (equality/inequality) constraints, was initiated in Alexandrov et al. (2001). The authors developed penalty, augmented Lagrangian, and SQP-like methods. A similar approach was used in Robinson (2007), where POD, space mapping methods, and their combination were proposed for constructing the reduced-order models. Furthermore, the use of the filter method, for nonlinear constraints handling, in low-fidelity models optimization along with TRPOD was presented in Agarwal (2010), Agarwal and Biegler (2011).

In this work, we follow the TRPOD method and to handle the state constraints we use the Lagrangian barrier method, which is a continuation of our work in Suwartadi et al. (2010). To best of our knowledge, the TRPOD method has not been applied to the reservoir simulation problem. Hence, the contribution of this work is to apply the TRPOD method to the production optimization of oil reservoirs. Furthermore, we consider nonlinear inequality constraints. Our method is a gradient-based optimization method which uses the POD method for computing basis functions for state and adjoint equations. Since we have implemented the adjoint method in the high-fidelity model and to avoid the difficulty of re-implementating the adjoint-based gradient in the reduced-order model, we take snapshots of the adjoint equations as well. Thus, the reduced-order models in this work consist of reduced-order state and adjoint equations. Our approach is different from the work of Doren et al. (2006), Markovinovic and Jansen (2006) where the reduced order model for the adjoint equations were derived based on the forward reduced order model.

It should be noted that there are many variants of the POD methods in addition ones mentioned above. A combined POD and discrete empirical interpolation method (DEIM), where DEIM is a variant of EIM Barrault et al. (2004), was recently proposed in Chaturantabut and Sorensen (2010). This work pointed out that the POD method is only good for approximating linear or bi-linear terms of equations. As shown in an example in Chaturantabut and Sorensen (2011), for nonlinear systems, the POD method in conjunction with DEIM gives considerable

CPU time speedup compared to the POD method alone. Since the oil reservoir models contain highly nonlinear terms, in this work we also compare the POD and POD-DEIM methods. The application of DEIM to optimization problems involving oil reservoir models is another contribution of this work.

The outline of this paper is the following. In Sect. 2 we describe the oil reservoir model which consists of pressure and saturation equations representing the state variables. We refer to these state equations as forward equations. In this section we also derive the adjoint equations and the reduced-order models. The production optimization problem is explained in Sect. 3, which basically is an economic optimization problem. In Sect. 4, we present the algorithms for the TRPOD method and the Lagrangian barrier method for nonlinear constraint handling. The algorithms use the TRPOD method in the inner iteration and the Lagrangian barrier in the outer iteration. This means the TRPOD method is used within the Lagrangian barrier iteration. Case examples that use 2D and 3D oil reservoirs are presented and the results are discussed in Sect. 5. Finally, based on the case example results we conclude this paper in Sect. 6.

In this paper we use standard linear algebra notations for describing mathematical equations. The superscript T is used to denote vector or matrix transpose. Matrices and vectors are written with bold letters while scalars are typed as ordinary letters.

2 Oil reservoir model

Water flooding is the most common secondary recovery technique for oil reservoirs. During early stages of oil reservoir production, the pressure in the reservoir is high enough to support production alone. However, water is often injected to provide additional pressure support in the reservoir and thereby increase recovery.

We assume the reservoir is above the *bubble point* so that the oil component is in liquid form only. Furthermore, we assume the process is isothermal, the liquids are incompressible, immiscible (water and oil cannot be mixed), no capillary pressure between oil and water, no gravity effect, and no-flow at the boundary of the reservoirs.

2.1 Forward model

The oil reservoir is governed by the continuity equation which expresses conservation of mass. We refer the model exposition here to Aarnes et al. (2007). The state equations consist of *pressure* and *saturation* equations. Let Ω be a porous media domain with boundary $\partial\Omega$. The pressure equation is given by

$$\mathbf{v} = -\mathbf{K}\lambda_t(s)\nabla p, \quad \nabla \cdot \mathbf{v} = q \quad \text{in} \quad \Omega, \tag{1}$$

where $\mathbf{v}$ is the Darcy velocity, $\mathbf{K}$ is the permeability tensor, p is the pressure, s is the water saturation, and q is the volumetric source/sink term. Finally λ_t is the total mobility, which in this setting is the sum of the water and oil mobility functions,

$$\lambda_t(s) = \lambda_w(s) + \lambda_o(s) = \frac{k_{rw}(s)}{\mu_w} + \frac{k_{ro}(s)}{\mu_o}. \tag{2}$$

Here, k_{rw}, k_{ro} and μ_w, μ_o are the water and oil relative permeabilities and viscosities, respectively. Assuming no-flow boundaries means that the normal component of the Darcy velocity across boundaries is zero.

The saturation equation is given by

$$\phi \frac{\partial s}{\partial t} + \nabla \cdot (f_w(s)\mathbf{v}) = q_w \quad \text{in} \quad \Omega, \tag{3}$$

where ϕ is the porosity and q_w is the volumetric water source. Finally, f_w is the water fractional flow function $f_w(s) = \lambda_w(s)/\lambda_t(s)$, which is also known as water cut. The nonlinear behavior of the above equations is mainly dictated by the shape of the relative permeability functions, which in this paper are taken to be quadratic. The relative permeability data are obtained from laboratory experiments using small portions of rocks which do not generally represent the rock properties of the whole reservoir. Hence, uncertainties are unavoidable.

Equations (1) and (3), which are elliptic and parabolic PDEs respectively, are solved numerically. Hence, we need to discretize the equations. We discretize the domain Ω into a set of polyhedral grid blocks $\{E_i\}$, where a grid block E contains faces e_k, $k = 1, \ldots, n_E$. Let $\mathbf{v}_E = \left(v_{e_1}, v_{e_2}, \ldots, v_{e_{n_E}}\right)$ be the outward pointing flux vectors corresponding to the faces of E, p_E the pressure at the grid block center, and $\boldsymbol{\pi}_E$ the pressures at the grid faces. Then, the discretized pressure equation for a single grid-block is

$$\begin{aligned} \mathbf{v}_E =& \lambda(s_E)\mathbf{T}_E(p_E - \boldsymbol{\pi}_E) \\ &\sum_{i=1}^{n_E} v_i|e_i| = q_E, \end{aligned} \tag{4}$$

where $\mathbf{T}_E$ is the transmissibility matrix, and q_E is the source/sink term in block E. Here, we discretize according to the two-point flux-approximation (TPFA) (see e.g., Aziz and Settari (1979)), which will result in diagonal transmissibility matrices.

The boundary conditions are only located at wells since we assume no-flow boundary. As in (1) the sink/source terms represent injector/producer wells. The wells are modeled by the Peaceman equation Peaceman (1983) as follows

$$q_E^w = -\lambda(s_E) WI_E^w \left(p_E - p_E^w\right). \tag{5}$$

Here q_E^w is the flow rate from well w into grid block E and p_E^w is the wellbore pressure (assumed to be constant since we neglect gravity and wellbore flow effects). Finally, WI_E^w is the Peaceman well-index for the grid block E.

The discretized pressure Eq. (4) and the well Eq. (5) can be combined such that they construct the following linear equation

$$\begin{pmatrix} \mathbf{B}^n(\mathbf{s}^{n-1}) & \mathbf{0} & \mathbf{C} & \mathbf{D} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_w^n(\mathbf{s}^{n-1}) & \mathbf{C}_w & \mathbf{0} & \mathbf{D}_{w,N} \\ \mathbf{C}^T & \mathbf{C}_w^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{D}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_{w,N}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{v}^n \\ -\mathbf{q}_w^n \\ -\mathbf{p}^n \\ \boldsymbol{\pi}^n \\ \mathbf{p}_{w,N}^n \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ -\mathbf{D}_{w,D}\mathbf{p}_{w,D}^n(\mathbf{u}^n) \\ \mathbf{0} \\ \mathbf{0} \\ -\mathbf{q}_{tot,N}^n(\mathbf{u}^n) \end{pmatrix}. \quad (6)$$

Here, the first and the second rows in the block-matrix above represent to Darcy's law as in (1) and (5) for all grid blocks. The third row corresponds to mass conservation for all grid blocks and the last two rows refer to continuity of fluxes for all grid block faces. The solution vector of the block-matrix equation above is

$$\begin{bmatrix} \mathbf{v}^n & -\mathbf{q}_w^n & -\mathbf{p}^n & \boldsymbol{\pi}^n & \mathbf{p}_{w,N}^n \end{bmatrix}^T$$

include the fluxes, the well rates, the grid-block pressures, the face and well pressures, and the wellbore pressure, respectively. The matrices $\mathbf{B}$, $\mathbf{B}_w$, $\mathbf{C}$, and $\mathbf{C}_w$ are block diagonal with each block corresponding to a grid block. Similarly, each column of $\mathbf{D}$ and $\mathbf{D}_{w,N}$ correspond to a unique face. Superscript n represents the time step and $\mathbf{u}^n$ is the control input at time step n, which could be either bottom-hole pressure (BHP): $\mathbf{u}^n = \mathbf{p}_{w,N}^n$ or well rate: $\mathbf{u}^n = \mathbf{v}^n$. The block-matrix Eq. (6) is solved for time step n using the default linear solver in MATLAB which is a direct sparse method (see Davis (2006)). We note that when TPFA is used, the pressure Eq. (6) can be reduced to a system of cell pressure-unknowns only, while the current implementation uses a mixed formulation where fluxes and cell pressures are solved for simultaneously.

We discretize the saturation Eq. (3) using a standard upstream weighted implicit finite volume method to form

$$\mathbf{s}^n = \mathbf{s}^{n-1} + \triangle t^n \, \mathbf{D}_{PV}^{-1}\big(\mathbf{A}(\mathbf{v}^n) f_w(\mathbf{s}^n) + \mathbf{q}(\mathbf{v}^n)_+\big). \quad (7)$$

Here, $\triangle t^n$ is the time step and $\mathbf{D}_{PV}$ is the diagonal matrix containing the grid block pore volumes. The matrix $\mathbf{A}(\mathbf{v}^n)$ is the sparse flux matrix based on the upstream weighted discretization scheme, and $\mathbf{q}(\mathbf{v}^\mathbf{n})_+$ is the vector of positive sources (in this setting, water injection rates). We note that the matrix $\mathbf{A}$ and vector $\mathbf{q}$ are linear functions of $\mathbf{v}^n$. The discretized saturation Eq. (7) is solved implicitly for the current time step $n+1$ using a Newton-Raphson method.

As seen the Eqs. (6) and (7) are coupled. The solution strategy to solve these equations is first solving the discretized pressure Eq. (6) using initial water saturation values, and then solve the discretized saturation Eq. (7). This procedure is repeated forward in time until the final time is reached. This kind of solution strategy is known as a sequential-splitting method Aarnes et al. (2007). The model used in this work is implemented in Lie et al. (2011). For convenience, we write the discrete state Eqs. (6) and (7) in an implicit form $\mathbf{F}(\widetilde{\mathbf{x}}, \widetilde{\mathbf{u}}) = 0$ as

$$
\begin{aligned}
\mathbf{F}(\widetilde{\mathbf{x}}, \widetilde{\mathbf{u}}) &= \begin{pmatrix} \mathbf{F}^1(\mathbf{p}^1, \mathbf{s}^0, \mathbf{s}^1, \mathbf{u}^1) \\ \vdots \\ \mathbf{F}^N(\mathbf{p}^N, \mathbf{s}^{N-1}, \mathbf{s}^N, \mathbf{u}^N) \end{pmatrix} = 0 \\
\mathbf{x}^{nT} &= (\mathbf{p}^{nT}, \mathbf{s}^{nT}), \quad n = 1, ..., N, \\
\widetilde{\mathbf{x}}^T &= (\mathbf{x}^{1T}, ..., \mathbf{x}^{NT}), \\
\widetilde{\mathbf{u}}^T &= (\mathbf{u}^{1T}, ..., \mathbf{u}^{NT}).
\end{aligned} \tag{8}
$$

The state vectors and control input vectors are stacked for all time instances from $n = 1, \ldots, N$. The dimension of $\widetilde{\mathbf{x}}$ and $\widetilde{\mathbf{u}}$ depends on the number of grid blocks and time steps.

2.2 Adjoint equations

Let $\mathcal{J}(\widetilde{\mathbf{x}}, \widetilde{\mathbf{u}}) = \sum_{n=1}^{N} \mathcal{J}^n(\mathbf{x}^n, \mathbf{u}^n)$ be an objective function, and denote by $\nabla_{\widetilde{\mathbf{u}}}\mathcal{J}$ the gradient with respect to a control input $\widetilde{\mathbf{u}}$. The detailed description of the objective function $\mathcal{J}(\widetilde{\mathbf{x}}, \widetilde{\mathbf{u}})$ will be explained in Sect. 3. We then construct an augmented objective function or Lagrangian functional

$$
\begin{aligned}
\mathcal{L}(\widetilde{\mathbf{x}}, \widetilde{\mathbf{u}}, \boldsymbol{\lambda}) &= \mathcal{J}(\widetilde{\mathbf{x}}, \widetilde{\mathbf{u}}) + \boldsymbol{\lambda}^T \mathbf{F}(\widetilde{\mathbf{x}}, \widetilde{\mathbf{u}}) \\
&= \sum_{n=1}^{N} \left(\mathcal{J}^n(\mathbf{x}^n, \mathbf{u}^n) + \boldsymbol{\lambda}^{nT} \mathbf{F}\left(\mathbf{x}^n, \mathbf{x}^{n-1}, \mathbf{u}^n\right) \right),
\end{aligned} \tag{9}
$$

for $n = 1, \ldots, N$, where

$$
\begin{aligned}
\boldsymbol{\lambda}^{nT} F = \quad & \boldsymbol{\lambda}_v^{nT}(\mathbf{B}^n \mathbf{v}^n - \mathbf{C}\mathbf{p}^n + \mathbf{D}\boldsymbol{\pi}^n) \\
& + \boldsymbol{\lambda}_{q_w}^{nT}\left(-\mathbf{B}_w^n \mathbf{q}_w - \mathbf{C}_w \mathbf{p}^n + \mathbf{D}_{w,N} \mathbf{p}_{w,N}^n(\mathbf{u}^n)\right) \\
& + \boldsymbol{\lambda}_p^{nT}\left(\mathbf{C}^T \mathbf{v}^n - \mathbf{C}_w^T \mathbf{q}_w^n\right) \\
& + \boldsymbol{\lambda}_\pi^{nT} \mathbf{D}^T \mathbf{v}^n \\
& + \boldsymbol{\lambda}_{p_{w,N}}^{nT}\left(-\mathbf{D}_{w,N}^T \mathbf{q}_w^n + \mathbf{q}_{tot,N}^n(\mathbf{u}^n)\right) \\
& + \boldsymbol{\lambda}_s^{nT}\left(\mathbf{s}^n - \mathbf{s}^{n-1} - \triangle t^n \mathbf{i}^n\right).
\end{aligned}
$$

Here $\mathbf{i}^n = \mathbf{D}_{PV}^{-1}\left(\mathbf{A}(\mathbf{v}^n) f_w(\mathbf{s}^n) + \mathbf{q}(\mathbf{v}^n)_+\right)$. By choosing $\boldsymbol{\lambda}$ that makes $\nabla_{\tilde{\mathbf{x}}}\mathcal{L} = \mathbf{0}$, we arrive at the adjoint equations

$$
\left(\frac{\partial F(\mathbf{x}^n, \mathbf{x}^{n-1}, \mathbf{u}^n)}{\partial \mathbf{x}^n}\right)^T \boldsymbol{\lambda}^n + \left(\frac{\partial F(\mathbf{x}^{n+1}, \mathbf{x}^n, \mathbf{u}^{n+1})}{\partial \mathbf{x}^n}\right)^T \boldsymbol{\lambda}^{n+1} = -\left(\frac{\partial \mathcal{J}^n(\mathbf{x}^n, \mathbf{u}^n)}{\partial \mathbf{x}^n}\right)^T, \tag{10}
$$

for $n = N, \ldots, 1$. The details of (10) are

$$\begin{pmatrix} \mathbf{B}^n(\mathbf{s}^{n-1}) & \mathbf{0} & \mathbf{C} & \mathbf{D} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_w^n(\mathbf{s}^{n-1}) & \mathbf{C}_w & \mathbf{0} & \mathbf{D}_{w,N} \\ \mathbf{C}^T & \mathbf{C}_w^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{D}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_{w,N}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\lambda}_v^n \\ \boldsymbol{\lambda}_{q_w}^n \\ \boldsymbol{\lambda}_p^n \\ \boldsymbol{\lambda}_\pi^n \\ \boldsymbol{\lambda}_{p_{w,N}}^n \end{pmatrix} = \begin{pmatrix} \left(\frac{\partial \mathbf{i}^n}{\partial \mathbf{v}^n}\right)^T \boldsymbol{\lambda}_s^n - \left(\frac{\partial \mathcal{J}}{\partial \mathbf{v}^n}\right)^T \\ \left(\frac{\partial \mathcal{J}^n}{\partial \mathbf{q}_w^n}\right)^T \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix}, \tag{11}$$

for the corresponding pressure equation and the following for the saturation

$$\begin{aligned} \left(\mathbf{I} - \Delta t \left(\frac{\partial \mathbf{i}^n}{\partial \mathbf{s}^n}\right)^T\right) \boldsymbol{\lambda}_s^n = \; & \boldsymbol{\lambda}_s^{n+1} - \left(\frac{\partial \mathcal{J}^n}{\partial \mathbf{s}^n}\right)^T \\ & - \left(\frac{\partial}{\partial \mathbf{s}^n}\left(\mathbf{B}^{n+1}\mathbf{v}^{n+1}\right)\right)^T \boldsymbol{\lambda}_v^{n+1} \\ & + \left(\frac{\partial}{\partial \mathbf{s}^n}\left(\mathbf{B}_w^{n+1}\mathbf{q}_w^{n+1}\right)\right)^T \boldsymbol{\lambda}_{q_w}^{n+1}. \end{aligned} \tag{12}$$

Using the fact that at the final time $\boldsymbol{\lambda}_\alpha^N = \mathbf{0}$ for $\alpha = \{v, q_w, p, \pi, p_{w,N}, s\}$, we are able to compute the Lagrangian multiplier for each time step backward in time. It should be noted that (11) and (12) are linear equations and they are solved using the direct sparse method as well. Finally using the obtained Lagrangian multipliers values, the gradient with respect to $\widetilde{\mathbf{u}}$ is

$$\nabla_{\widetilde{\mathbf{u}}} \mathcal{L}^n = \frac{\partial \mathcal{J}^n(\mathbf{x}^n, \mathbf{u}^n)}{\partial \mathbf{u}^n} + \boldsymbol{\lambda}^{nT} \frac{\partial F(\mathbf{x}^n, \mathbf{x}^{n-1}, \mathbf{u}^n)}{\partial \mathbf{u}^n}. \tag{13}$$

2.3 Reduced-order models

2.3.1 POD method

In order to build a reduced-order model based on the POD method, we need to take snapshots of the high-fidelity model described in (6) and (7). Let $\mathbf{x}^T = [\mathbf{p} \quad \mathbf{s}] = [\mathbf{x}_p^T \quad \mathbf{x}_s^T] \in \mathbb{R}^{n_x}$ be the snapshot of the solution of the forward equations with n_x as the dimension of the solution, which is the number of grid block. Given a set of snapshots $\{\mathbf{x}_1, \ldots, \mathbf{x}_\Xi\} \in \mathbb{R}^{n_x \times \Xi}$, the snapshot matrices are

$$\mathbf{x}_p = \begin{bmatrix} \mathbf{x}_{p_1}^1 & \mathbf{x}_{p_1}^2 & \cdots & \mathbf{x}_{p_1}^\Xi \\ \mathbf{x}_{p_2}^1 & \mathbf{x}_{p_2}^2 & \cdots & \mathbf{x}_{p_2}^\Xi \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_{p_{n_x}}^1 & \mathbf{x}_{p_{n_x}}^2 & \cdots & \mathbf{x}_{p_{n_x}}^\Xi \end{bmatrix}_{n_x \times \Xi}, \quad \mathbf{x}_s = \begin{bmatrix} \mathbf{x}_{s_1}^1 & \mathbf{x}_{s_1}^2 & \cdots & \mathbf{x}_{s_1}^\Xi \\ \mathbf{x}_{s_2}^1 & \mathbf{x}_{s_2}^2 & \cdots & \mathbf{x}_{s_2}^\Xi \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_{s_{n_x}}^1 & \mathbf{x}_{s_{n_x}}^2 & \cdots & \mathbf{x}_{s_{n_x}}^\Xi \end{bmatrix}_{n_x \times \Xi} \tag{14}$$

It should be noted there is no additional computational time to build the snapshot matrices since they are merely solutions of the forward state equations. Let $\mathcal{V} =$

$\operatorname{span}\{\mathbf{x}_1, \ldots, \mathbf{x}_\Xi\}$, the POD basis function is a solution of an optimization problem for finding orthonormal vectors $\{\psi_i\}_{i=1}^{\ell}$, where $\ell \leq \operatorname{rank}(\mathcal{V})$. The optimization formulation is

$$\min_{\{\psi_i\}_{i=1}^{\ell}} \mathcal{J}(\psi_1, \ldots, \psi_\ell) := \sum_{j=1}^{\Xi} \left\| \mathbf{x}_j - \sum_{i=1}^{\ell} \left(\mathbf{x}_j^T \psi_i \right) \psi_i \right\|_2^2 \\ \text{subject to}\, \psi_i^T \psi_j = \delta_{ij} = \begin{cases} 1 & \text{if}\, i = j \\ 0 & \text{otherwise} \end{cases}. \tag{15}$$

We define a Lagrangian functional

$$\mathcal{L}(\psi_1, \ldots, \psi_\ell, \lambda_{11}, \ldots, \lambda_{\ell\ell}) = \mathcal{J}(\psi_1, \ldots, \psi_\ell) + \sum_{i,j=1}^{\ell} \lambda_{ij} \left(\psi_i^T \psi_j - \delta_{ij} \right). \tag{16}$$

The necessary optimality conditions, $\frac{\partial \mathcal{L}}{\partial \psi_i} = 0$ and $\frac{\partial \mathcal{L}}{\partial \lambda_{ij}} = 0$, give us an eigenvalue problem

$$\sum_{j=1}^{\Xi} \mathbf{x}_j \left(\mathbf{x}_j^T \psi_i \right) = \lambda_{ii} \psi_i, \; \text{for}\, i = 1, \ldots, \ell \tag{17}$$

or by setting $\lambda_i = \lambda_{ii}$ and $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_\Xi] \in \mathbb{R}^{\mathrm{n_x} \times \Xi}$, then the problem reads

$$\mathbf{X}\mathbf{X}^T \psi_i = \lambda_i \psi_i, \; \text{for}\, i = 1, \ldots, \ell. \tag{18}$$

To compute the solution of (18), we decompose the vector $\mathbf{X}$ using singular value decomposition (SVD), that is,

$$\mathbf{X} = \mathbf{U} \boldsymbol{\Sigma} \mathbf{V}^T, \tag{19}$$

where $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_{n_x}] \in \mathbb{R}^{\mathrm{n_x} \times \mathrm{n_x}}$ and $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_\Xi] \in \mathbb{R}^{\Xi \times \Xi}$ are orthogonal matrices, and $\boldsymbol{\Sigma} \in \mathbb{R}^{\mathrm{n_x} \times \Xi}$ is the diagonal matrix with diagonal arranged in a decreased order, that is, $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_\Xi \geq 0$. In other words,

$$\mathbf{U}^T \mathbf{X} \mathbf{V} = \boldsymbol{\Sigma}. \tag{20}$$

Moreover, it follows that for $1 \leq i \leq \Xi$ that

$$\mathbf{X}\mathbf{v}_i = \sigma_i \mathbf{u}_i, \; \mathbf{X}^T \mathbf{u}_i = \sigma_i \mathbf{v}_i, \; \mathbf{X}\mathbf{X}^T \mathbf{u}_i = \sigma_i^2 \mathbf{u}_i. \tag{21}$$

The solution of problem (18) is a POD basis $\psi_i = \mathbf{u}_i$ and $\lambda_i = \sigma_i^2 > 0$ for $i = 1, \ldots, \ell \leq d = \dim \mathcal{V}$. The minimized objective function (15) is then

$$\mathcal{J}(\psi_1, \ldots, \psi_\ell) := \sum_{j=1}^{\Xi} \left\| \mathbf{x}_j - \sum_{i=1}^{\ell} \left(\mathbf{x}_j^T \psi_i \right) \psi_i \right\|_2^2 = \sum_{i=\ell+1}^{d} \lambda_i. \tag{22}$$

To determine the dimension of ℓ, the singular value is cut according to the following '*energy*' truncation

$$E = \frac{\sum_{i=1}^{\ell} \sigma_i}{\sum_{i=1}^{\Xi} \sigma_i} < \alpha, \tag{23}$$

where typically $0.9 \le \alpha < 1$. This choice of truncation is a rather heuristic consideration Volkwein (2003). We follow what is commonly used in the literature. In other work, one may use quadratic summation of the singular value, see e.g., Doren et al. (2006), Markovinovic and Jansen (2006).

The POD method is applied to the state and adjoint equations. Let ℓ_p, ℓ_s, and n_p, n_s be the dimension of the pressure and saturation equations in reduced-order and high-fidelity models respectively, where $\ell_p \ll n_p$ and $\ell_s \ll n_s$. Then transformation from the reduced-order to the high-fidelity model is

$$\begin{aligned} \mathbf{x}_p &= \mathbf{V}_p \hat{\mathbf{x}}_p + \overline{\mathbf{x}}_p, \\ \mathbf{x}_s &= \mathbf{V}_s \hat{\mathbf{x}}_s + \overline{\mathbf{x}}_s. \end{aligned} \tag{24}$$

The high-fidelity model is represented by $\mathbf{x}_p \in \mathbb{R}^{n_p}$, $\mathbf{x}_s \in \mathbb{R}^{n_s}$ and their respective averages during the snapshots $\overline{\mathbf{x}}_p \in \mathbb{R}^{n_p}$ and $\overline{\mathbf{x}}_s \in \mathbb{R}^{n_s}$, i.e.,

$$\overline{\mathbf{x}} = \frac{1}{\Xi} \sum_{i=1}^{\Xi} \mathbf{x}_i. \tag{25}$$

In the reduced-order space, $\hat{\mathbf{x}}_p \in \mathbb{R}^{\ell_p}$ and $\hat{\mathbf{x}}_s \in \mathbb{R}^{\ell_s}$, the forward equations now become

$$\begin{aligned} &\mathbf{V}_p^T \begin{pmatrix} \mathbf{B}^n(\mathbf{s}^{n-1}) & \mathbf{0} & \mathbf{C} & \mathbf{D} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_w^n(\mathbf{s}^{n-1}) & \mathbf{C}_w & \mathbf{0} & \mathbf{D}_{w,N} \\ \mathbf{C}^T & \mathbf{C}_w^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{D}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_{w,N}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \mathbf{V}_p \hat{\mathbf{x}}_p^n \\ &= \mathbf{V}_p^T \left(\begin{pmatrix} \mathbf{0} \\ -\mathbf{D}_{w,D}\mathbf{p}_{w,D}^n(\mathbf{u}^n) \\ \mathbf{0} \\ \mathbf{0} \\ -\mathbf{q}_{tot,N}^n(\mathbf{u}^n) \end{pmatrix} - \begin{pmatrix} \mathbf{B}^n(\mathbf{s}^{n-1}) & \mathbf{0} & \mathbf{C} & \mathbf{D} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_w^n(\mathbf{s}^{n-1}) & \mathbf{C}_w & \mathbf{0} & \mathbf{D}_{w,N} \\ \mathbf{C}^T & \mathbf{C}_w^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{D}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_{w,N}^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix} \overline{\mathbf{x}}_p \right), \end{aligned} \tag{26}$$

$$\hat{\mathbf{s}}^n = \hat{\mathbf{s}}^{n-1} + \triangle t\, \mathbf{V}_s^T \mathbf{D}_{PV}^{-1} \big(\mathbf{A}(\mathbf{v}^n) f_w(\mathbf{V}_s \hat{\mathbf{s}}^n + \bar{\mathbf{s}}) + \mathbf{q}(\mathbf{v}^n)_+ \big). \tag{27}$$

Similarly, we also take snapshots of the adjoint Eqs. (11) and (12) and obtain reduced-order adjoint equations. The reduced-order corresponding adjoint pressure and saturation respectively are

$$\mathbf{V}_{ap}^{T}\begin{pmatrix} \mathbf{B}^{n}(\mathbf{s}^{n-1}) & \mathbf{0} & \mathbf{C} & \mathbf{D} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_{w}^{n}(\mathbf{s}^{n-1}) & \mathbf{C}_{w} & \mathbf{0} & \mathbf{D}_{w,N} \\ \mathbf{C}^{T} & \mathbf{C}_{w}^{T} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{D}^{T} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_{w,N}^{T} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}\mathbf{V}_{ap}\begin{pmatrix} \hat{\boldsymbol{\lambda}}_{v}^{n} \\ \hat{\boldsymbol{\lambda}}_{q_w}^{n} \\ \hat{\boldsymbol{\lambda}}_{p}^{n} \\ \hat{\boldsymbol{\lambda}}_{\pi}^{n} \\ \hat{\boldsymbol{\lambda}}_{p_{w,N}}^{n} \end{pmatrix}$$
$$= \mathbf{V}_{ap}^{T}\left\{\begin{pmatrix} \left(\frac{\partial \mathbf{i}^{n}}{\partial \mathbf{v}^{n}}\right)^{T}\boldsymbol{\lambda}_{s}^{n} - \left(\frac{\partial \mathcal{J}}{\partial \mathbf{v}^{n}}\right)^{T} \\ \left(\frac{\partial \mathcal{J}^{n}}{\partial \mathbf{q}_{w}^{n}}\right)^{T} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} - \begin{pmatrix} \mathbf{B}^{n}(\mathbf{s}^{n-1}) & \mathbf{0} & \mathbf{C} & \mathbf{D} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_{w}^{n}(\mathbf{s}^{n-1}) & \mathbf{C}_{w} & \mathbf{0} & \mathbf{D}_{w,N} \\ \mathbf{C}^{T} & \mathbf{C}_{w}^{T} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{D}^{T} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_{w,N}^{T} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}\begin{pmatrix} \boldsymbol{\lambda}_{v} \\ \boldsymbol{\lambda}_{q_w} \\ \boldsymbol{\lambda}_{p} \\ \boldsymbol{\lambda}_{\pi} \\ \boldsymbol{\lambda}_{p_{w,N}} \end{pmatrix}\right\}, \tag{28}$$

$$\begin{aligned} \mathbf{V}_{as}^{T}\left(\mathbf{I} - \Delta t\left(\frac{\partial \mathbf{i}^{n}}{\partial \mathbf{s}^{n}}\right)^{T}\right)\mathbf{V}_{as}\hat{\boldsymbol{\lambda}}_{s}^{n} = {} & \mathbf{V}_{as}^{T}\left\{\boldsymbol{\lambda}_{s}^{n+1} - \left(\frac{\partial \mathcal{J}^{n}}{\partial \mathbf{s}^{n}}\right)^{T} - \left(\frac{\partial}{\partial \mathbf{s}^{n}}\left(\mathbf{B}^{n+1}\mathbf{v}^{n+1}\right)\right)^{T}\boldsymbol{\lambda}_{v}^{n+1}\right\} \\ & + \mathbf{V}_{as}^{T}\left\{\left(\frac{\partial}{\partial \mathbf{s}^{n}}\left(\mathbf{B}_{w}^{n+1}\mathbf{q}_{w}^{n+1}\right)\right)^{T}\boldsymbol{\lambda}_{q_w}^{n+1}\right\} \\ & - \mathbf{V}_{as}^{T}\left\{\left(\mathbf{I} - \Delta t\left(\frac{\partial \mathbf{i}^{n}}{\partial \mathbf{s}^{n}}\right)^{T}\right)\bar{\boldsymbol{\lambda}}_{s}\right\}, \end{aligned} \tag{29}$$

where $\mathbf{V}_{ap}$ and $\mathbf{V}_{as}$ are the basis functions for the corresponding adjoint pressure and saturation equations, and $\bar{\boldsymbol{\lambda}}$ is the average snapshot of Lagrangian multipliers of adjoint equation solutions. As seen in all reduced-order Eqs. (11), (12), (28) and (29), the reconstruction of high-fidelity equations are needed in order to solve the reduced-order equations. Hence, after solving the reduced-order equations we reconstruct the high-fidelity solution through the transformation (24). This will inevitably give an overhead in the computational time. Nonetheless, we still gain computational reduction in CPU time compared to the high-fidelity model run.

2.3.2 POD-DEIM

Let us consider the water saturation Eq. (7) in the following form

$$\mathbf{s}^{n+1} = \mathbf{s}^n + \triangle t^n\, \mathbf{D}_{PV}^{-1}\left(\mathbf{A}(\mathbf{v}^n) f_w\left(\mathbf{s}^{n+1}\right) + \mathbf{q}(\mathbf{v}^n)_+\right). \tag{30}$$

This equation is solved for time step $n+1$ implicitly using Newton-Raphson method, that is,

$$0 \equiv G\left(\mathbf{s}^{n+1}\right) = \mathbf{s}^{n+1} - \mathbf{s}^n - \triangle t^n\, \mathbf{D}_{PV}^{-1}\left(\mathbf{A}(\mathbf{v}^n) f_w\left(\mathbf{s}^{n+1}\right) + \mathbf{q}(\mathbf{v}^n)_+\right). \tag{31}$$

Given an initial guess $\tilde{\mathbf{s}}$, then by Taylor expansion, the equation above is approximated by

$$0 = G\left(\mathbf{s}^{n+1}\right) \approx G(\widetilde{\mathbf{s}}) + G'(\widetilde{\mathbf{s}})\left(\mathbf{s}^{n+1} - \widetilde{\mathbf{s}}\right), \tag{32}$$

where the solution $\mathbf{s}^{n+1}$ is obtained through $\mathbf{s}^{n+1} = \tilde{\mathbf{s}} + d\tilde{\mathbf{s}}$. The changes $d\tilde{\mathbf{s}} = \mathbf{s}^{n+1} - \tilde{\mathbf{s}}$ satisfies the linear equation $-G'(\tilde{\mathbf{s}})d\tilde{\mathbf{s}} = G(\tilde{\mathbf{s}})$. The Jacobian $G'(\tilde{\mathbf{s}})$ is

$$G'(\tilde{\mathbf{s}}) = \mathbf{I} - \triangle t^n\, \mathbf{D}_{PV}^{-1}\mathbf{A}(\mathbf{v}^n) f_w'(\mathbf{s}^n)$$

Note that the terms $f_w'(\mathbf{s}^n)$ and $f_w(\mathbf{s}^{n+1})$ are evaluated componentwise, which means that they are evaluated at each gridblock.

The reduced-order water saturation equation can be written as follows

$$\hat{\mathbf{s}}^{n+1} = \hat{\mathbf{s}}^n + \Delta t^n\, \mathbf{D}_{PV}^{-1}\left(\underbrace{\mathbf{V}_{\mathbf{s}}^{\mathbf{T}}}_{\ell_s \times n_s}\underbrace{\mathbf{A}(\mathbf{v}^{\mathbf{n}})}_{n_s \times n_s}\underbrace{f_w\left(\mathbf{V}_s\hat{\mathbf{s}}^{n+1} + \bar{\mathbf{s}}\right)}_{n_s \times 1} + \mathbf{V}_s^T q(\mathbf{v}^n)_+\right). \tag{33}$$

As seen above, we still need to evaluate the nonlinear term, which is in this case is the water cut $f_w(\mathbf{s}^{n+1})$, in high-fidelity dimension n_s. Similarly, in solving (32) in the reduced-order space, we evaluate the Jacobian in high-fidelity. This is obviously not desirable. To mitigate this, we construct another reduced-order model for the water cut term. This is when the POD-DEIM Chaturantabut and Sorensen (2010) comes into play. The method projects the nonlinear term onto a lower dimension, such that

$$\mathbf{f}(\tau) \simeq \boldsymbol{\Phi} c(\tau) + \bar{\mathbf{f}} \tag{34}$$

where $\boldsymbol{\Phi} = [\Phi_1, \ldots, \Phi_m] \in \mathbb{R}^{\mathrm{n_x \times m}}$, $c(\tau)$ is the corresponding vector coefficient, and $\bar{\mathbf{f}}$ is the average value of the nonlinear term in the snapshot.

The vector $c(\tau)$ is determined by selecting the appropriate m rows from the overdetermined $\mathbf{f}(\tau) \simeq \boldsymbol{\Phi} c(\tau) + \bar{\mathbf{f}}$. The selection is done by a matrix $\mathbf{P} = \left[\mathbf{e}_{\wp_1}, \ldots, \mathbf{e}_{\wp_m}\right] \in \mathbb{R}^{n_x \times m}$, where $\mathbf{e}_j$ is the j-th column of the identity matrix. If $\mathbf{P}^T\boldsymbol{\Phi}$ is invertible, the coefficient vector $c(\tau)$ can be determined from

$$\mathbf{P}^T\mathbf{f}(\tau) = \mathbf{P}^T\boldsymbol{\Phi} c(\tau) + \mathbf{P}^T\bar{\mathbf{f}}, \tag{35}$$

with some rearrangement, we end up with

$$c(\tau) = \left(\mathbf{P}^T \boldsymbol{\Phi}\right)^{-1} \mathbf{P}^T \left(\mathbf{f}(\tau) - \bar{\mathbf{f}}\right). \tag{36}$$

Finally, the high-fidelity nonlinear approximation of (34) is

$$\mathbf{f}(\tau) \simeq \boldsymbol{\Phi} c(\tau) + \bar{\mathbf{f}} = \left\{ \underbrace{\boldsymbol{\Phi}\left(\mathbf{P}^T \boldsymbol{\Phi}\right)^{-1}}_{\ell \times m} \underbrace{\mathbf{P}^{\mathbf{T}}\left(\mathbf{f}(\tau) - \bar{\mathbf{f}}\right)}_{m \times 1} \right\} + \bar{\mathbf{f}}. \tag{37}$$

From this equation, we now need to construct the $\boldsymbol{\Phi}$ and $\mathbf{P}$ matrices. $\boldsymbol{\Phi}$ is selected as the POD basis function of the water cut f_w, while the $\mathbf{P}$ matrix (interpolation index) is determined by Algorithm 1. The `max` in the algorithm refers to MATLAB function `max`. Therefore, $[\,\rho \quad \wp_\ell\,] = \max\{|r|\}$ means $\rho = \left|r_{\wp_\ell}\right| = \max_{\mathrm{i}=1,\ldots,\mathrm{n}}\{|r_i|\}$.

Algorithm 1 POD-DEIM

```
INPUT: $\{\Phi_l\}_{l=1}^{m} \subset \mathbb{R}^{n_x}$
OUTPUT: $\wp = [\wp_1, \ldots, \wp_m]^T \in \mathbb{R}^m$
1. $[\,\rho \ \wp_1\,] = \max\{|\mathbf{\Phi}_1|\}$
2. $\boldsymbol{\Phi} = [\Phi_1]$, $\mathbf{P} = [\mathbf{e}_{\wp_1}]$, $\wp = [\wp_1]$
3. for $l = 2$ to $m$ do
       Solve $\left(\mathbf{P}^T \boldsymbol{\Phi}\right)\mathbf{c} = \mathbf{P}^T \Phi_l$ for $\mathbf{c}$
       $\mathbf{r} = \mathbf{u}_l - \boldsymbol{\Phi}\mathbf{c}$
       $[\,\rho \ \wp_l\,] = \max\{|\mathbf{r}|\}$
       $\boldsymbol{\Phi} \leftarrow [\,\mathbf{U} \ \mathbf{\Phi}_l\,]$, $\mathbf{P} \leftarrow [\,\mathbf{P} \ \mathbf{e}_{\wp_l}\,]$, $\bar{\wp} \leftarrow \begin{bmatrix} \wp \\ \wp_l \end{bmatrix}$
   end for
```

We employ the POD-DEIM just for the forward saturation equation, since this is the only equation that contains the nonlinear water cut term. So now the reduced-order equation of water saturation is

$$\hat{\mathbf{s}}^{n+1} = \hat{\mathbf{s}}^n + \triangle t^n \mathbf{D}_{PV}^{-1}\left(\boldsymbol{\Upsilon} + \mathbf{V}_s^T \mathbf{q}(\mathbf{v}^n)_+\right), \tag{38}$$

where $\boldsymbol{\Upsilon} = \underbrace{\mathbf{V}_s^T \mathbf{A}(\mathbf{v}^n) \boldsymbol{\Phi}^T \left(\mathbf{P}^T \boldsymbol{\Phi}\right)^{-1}}_{\ell_s \times m} \underbrace{\mathbf{P}^{\mathbf{T}}\left(\mathbf{f_w}\left(\mathbf{V_s}\hat{\mathbf{s}}^{\mathbf{n+1}} + \bar{\mathbf{s}}\right) - \bar{\mathbf{f}}_{\mathbf{w}}\right)}_{m \times 1}$. Here, by using the interpolation matrix $\mathbf{P}$ we evaluate the nonlinear water cut term f_w in the reduced-space of dimension m rather than in the high-fidelity space of dimension n_s.

One may notice there is a nonlinear dependence in the pressure Eq. (6) as well, that is, in the term $\mathbf{B}^n(\mathbf{s}^{n-1})$, involving water saturation from the previous time step. Because (6) is a linear equation and the POD method has proved to be good for linear terms, we do not apply POD-DEIM for the pressure equations. Nevertheless, this opens an opportunity for future investigation. To this end, we introduce the implicit form of reduced-order equation equivalent to (8) as $\mathbf{F}(\tilde{\mathbf{x}}_r, \tilde{\mathbf{u}}) = 0$ in the following

$$\begin{aligned}\mathbf{F}(\tilde{\mathbf{x}}_r,\tilde{\mathbf{u}}) &= \begin{pmatrix} \mathbf{F}^1(\hat{\mathbf{p}}^1,\hat{\mathbf{s}}^0,\hat{\mathbf{s}}^1,\mathbf{u}^1) \\ \vdots \\ \mathbf{F}^N(\hat{\mathbf{p}}^N,\hat{\mathbf{s}}^{N-1},\hat{\mathbf{s}}^N,\mathbf{u}^N) \end{pmatrix} \\ \hat{\mathbf{x}}^{nT} &= \left(\hat{\mathbf{p}}^{nT},\hat{\mathbf{s}}^{nT}\right), \quad n=1,\ldots,N, \\ \tilde{\mathbf{x}}_r^T &= \left(\hat{\mathbf{x}}^{1T}, \quad \ldots, \quad \hat{\mathbf{x}}^{NT}\right), \\ \tilde{\mathbf{u}}^T &= \left(\mathbf{u}^{1T}, \quad \ldots, \quad \mathbf{u}^{NT}\right).\end{aligned} \tag{39}$$

$\tilde{\mathbf{x}}_r$ is the stacked reduced-order state vector from the solution of the forward equations.

3 Production optimization problem

By injecting water into reservoirs, cumulative oil production may increase. This is cast as the following optimization problem using a reduced-order model

$$\begin{aligned} &(\hat{\mathcal{P}}:)\max_{\tilde{u}\in\mathbb{R}^{n_{\tilde{u}}}} \mathcal{J}(\tilde{\mathrm{x}}_{\mathrm{r}},\tilde{u}) \\ &\text{subject to} :\mathbf{F}(\tilde{\mathbf{x}}_r,\tilde{\mathbf{u}}) = 0 \\ &g(\mathbf{u}^n) \geq 0, \ \forall n=1,\ldots,N \\ &h(\hat{\mathbf{x}}^n,\mathbf{u}^n) \geq 0, \ \forall n=1,\ldots,N \\ &\mathbf{x}^0 \text{ is given.}\end{aligned}$$

$\mathcal{J}(\tilde{\mathbf{x}}_r,\tilde{\mathbf{u}})$, $g(\mathbf{u}^n)$, $h(\hat{\mathbf{x}}^n,\mathbf{u}^n)$ are assumed $\mathcal{C}^1$. The control input and the state constraints are represented by $g:\mathbb{R}^{n_{\tilde{u}}} \to \mathbb{R}^{n_g}$ and $h:\mathbb{R}^{n_x\times n_{\tilde{u}}} \to \mathbb{R}^{n_h}$, respectively. The objective function is given by $\mathcal{J}:\mathbb{R}^{n_{\tilde{x}_r}\times n_{\tilde{u}}} \to \mathbb{R}$ and the state equations are posed as implicit constraints. The state variables and the control inputs are dependent, therefore we are able to perform the optimization in the control input space of $\tilde{\mathbf{u}}$ instead of in the space of $(\tilde{\mathbf{x}}_r,\tilde{\mathbf{u}})$. To this end, we denote the objective as $\mathcal{J}(\tilde{\mathbf{u}})$ omitting $\mathcal{J}(\tilde{\mathbf{x}}_r(\tilde{\mathbf{u}}),\tilde{\mathbf{u}})$. In this work, we use the recovery factor (RF) as the objective function

$$\mathcal{J}(\tilde{\mathbf{u}}) = \frac{\sum_{i=1}^{N_{gb}} \mathbf{D}_{PV_i}(1-s_i^N(\tilde{\mathbf{u}}))}{V_{gb}} \times 100, \tag{40}$$

where $\mathbf{D}_{PV_i}$ is the diagonal element of the pore volume matrix, $s_i^N(\tilde{\mathbf{u}})$ is the water saturation at grid block i at the final time step N, $\tilde{\mathbf{u}}$ the control input in this case is well rate, N_{gb} is the number of grid blocks in the reservoir, and V_{gb} is the total pore volume of the reservoir. In other words, the recovery factor represents the percentage of oil that can be produced from the reservoir. Water flooding can normally give a recovery factor somewhere between 20 and 40 %, meaning that 20–40 % of the oil is extracted from the reservoir.

The optimization problem in high-fidelity is described as

$$
\begin{aligned}
(\mathcal{P}:)\ & \max_{\tilde{\mathbf{u}}\in\mathbb{R}^{n_{\tilde{\mathbf{u}}}}} \mathcal{J}\left(\tilde{\mathbf{x}}_r, \tilde{\mathbf{u}}\right) \\
\text{subject to}:\ & \mathbf{F}(\tilde{\mathbf{x}}, \tilde{\mathbf{u}}) = 0 \\
& g(\mathbf{u}^n) \geq 0, \quad \forall n = 1,\ldots,N \\
& h(\mathbf{x}^n, \mathbf{u}^n) \geq 0, \quad \forall n = 1,\ldots,N \\
& \mathbf{x}^0 \text{ is given.}
\end{aligned}
$$

Let $\mathcal{U}_{opt}$ and $\hat{\mathcal{U}}_{opt}$ be the solutions of the optimization problem in high-fidelity $\mathcal{P}$ and reduced-space $\hat{\mathcal{P}}$, respectively. In Hinze and Volkwein (2005), the error estimate between $\mathcal{U}_{opt}$ and $\hat{\mathcal{U}}_{opt}$ is

$$
\begin{aligned}
&\left\|\mathcal{U}_{opt} - \hat{\mathcal{U}}_{opt}\right\| \\
&\leq c_p \left\{ \left\|\mathbf{x}^0 - \boldsymbol{\Phi}\hat{\mathbf{x}}^0\right\| + \left\|\tilde{\mathbf{x}} - \boldsymbol{\Phi}\tilde{\tilde{x}}\right\| + \left\|\boldsymbol{\lambda}_{\tilde{\mathbf{x}}}(\tilde{\mathbf{x}}(\tilde{\mathbf{u}})) - \boldsymbol{\Phi}(\boldsymbol{\lambda}_{\tilde{x}}(\tilde{\mathbf{x}}(\tilde{\mathbf{u}})))\right\| + \sqrt{\sum_{i=\ell+1}^{d} \boldsymbol{\lambda}_i} \right\}.
\end{aligned} \tag{41}
$$

Here c_p is a positive constant, Φ are the basis functions (eigenvectors) obtained from the POD method, $\boldsymbol{\lambda}_{\tilde{\mathbf{x}}}$ is the Lagrangian multiplier in the adjoint Eqs. (11) and (12), and the last term contains $\boldsymbol{\lambda}$ from the residual of POD truncation as in (22).

The second term of the error estimate (41) $\|\tilde{\mathbf{x}} - \boldsymbol{\Phi}\tilde{\mathbf{x}}_r\|$ can be reduced by taking snapshots of the state equations. Similarly, the term $\|\boldsymbol{\lambda}_{\tilde{\mathbf{x}}}(\tilde{\mathbf{x}}(\tilde{\mathbf{u}})) - \boldsymbol{\Phi}(\boldsymbol{\lambda}_{\tilde{x}}(\tilde{\mathbf{x}}(\tilde{\mathbf{u}})))\|$ may give smaller error by taking snapshots of the adjoint equations. We will proceed with this approach and will explain it in the next section.

4 Solution method

In this section we explain how to use a reduced-order model to solve the optimization problem $\hat{\mathcal{P}}$.

4.1 Trust-region POD

The optimization is performed using a reduced-order model, which is known as surrogate optimization. The principle of surrogate optimization is depicted in Fig. 1. During the course of optimization many simulation runs are needed, and therefore by using a reduced-order model the goal is to reduce simulation runtime. Furthermore, as mentioned in the introduction there are alternative ways to construct reduced-order models. Here we will use POD and DEIM described in the previous section.

To maintain the quality of reduced-order models, we apply a trust-region framework. The trust-region framework is used as the globalization strategy in gradient-based optimization Conn et al. (2000). In a trust-region globalization strategy a quadratic approximation is used to approximate the objective function while in the surrogate optimization trust-region framework, which is called the trust-region POD (TRPOD) method, one builds a POD-based reduced-order model. The method will in

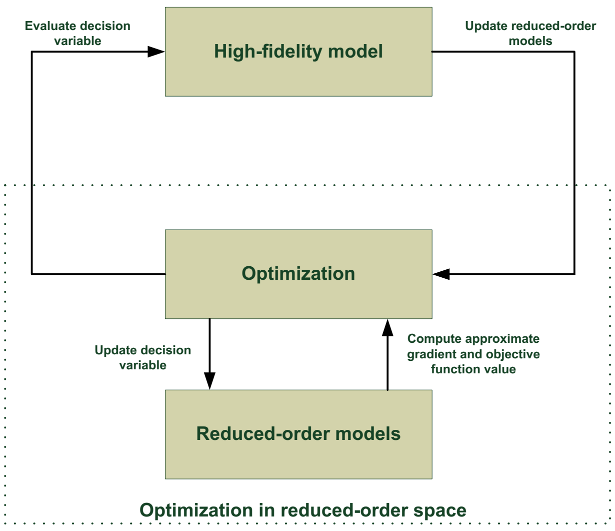

Evaluate decision variable
High-fidelity model
Update reduced-order models
Optimization
Update decision variable
Compute approximate gradient and objective function value
Reduced-order models
Optimization in reduced-order space

Fig. 1 Optimization in reduced space (surrogate optimization). Optimization is performed using reduced-order models (ROMs) and the ROMS are updated according to the trust-region rule. This figure is modified after Alexandrov et al. (2001)

principle enlarge its region when good approximations are obtained and reduce the region when the quality of model approximation is poor, or keep the region if the approximation quality is the same as at the previous iteration. The quality of approximation is measured by checking the value of the objective function in the high-fidelity model. Finally, the method will terminate due to some stopping criteria. The details of this method is explained in Algorithm 2 with some remarks below.

During the k-th iteration the TRPOD method solves the following subproblem

$$\begin{aligned} \max_{\boldsymbol{\delta}\in\mathbb{R}^{n_{\tilde{\mathbf{u}}}}} & \mathcal{J}^{\mathcal{R}}_{b,k}(\tilde{\mathbf{u}}+\boldsymbol{\delta}) \\ \text{s.t. } & \mathbf{F}_k(\tilde{\mathbf{x}}_r,\tilde{\mathbf{u}}+\boldsymbol{\delta})=0 \\ & g_k(\mathbf{u}^n+\boldsymbol{\delta})\geq 0, \quad \forall n=1,\ldots,N \\ & h_k(\hat{\mathbf{x}}^n,\mathbf{u}^n+\boldsymbol{\delta})\geq 0, \quad \forall n=1,\ldots,N \\ & \|\boldsymbol{\delta}\|_\infty \leq \triangle_k \end{aligned} \tag{42}$$

The optimized variables in the subproblem are the steps $\boldsymbol{\delta}$, where the length, expressed in infinite norm, is bounded by a trust-region radius $\triangle_k$, and the implicit form of reservoir dynamics

$$\mathbf{F}_k(\tilde{\mathbf{x}}_r,\tilde{\mathbf{u}}+\boldsymbol{\delta}) = \begin{pmatrix} \mathbf{F}^1\left(\hat{\mathbf{p}}^1,\hat{\mathbf{s}}^0,\hat{\mathbf{s}}^1,\mathbf{u}^1_k+\boldsymbol{\delta}^1_k\right) \\ \vdots \\ \mathbf{F}^N\left(\hat{\mathbf{p}}^N,\hat{\mathbf{s}}^{N-1},\hat{\mathbf{s}}^N,\mathbf{u}^N_k+\boldsymbol{\delta}^N_k\right) \end{pmatrix}. \tag{43}$$

$\mathcal{J}^{\mathcal{R}}_{b,k}$ is the modified objective function using the Lagrangian barrier method (44), explained in next subsection, evaluated using the (forward) reduced-order model.

Algorithm 2 Trust-region POD method

```
Step 0: Initialization Choose $0 < \eta_1 < \eta_2 < 1 \leq \eta_3$, $0 < \gamma_1 \leq \gamma_2 < 1 \leq \gamma_3$, an initial trust-region radius $\triangle_0$, minimum radius $\triangle_{min}$, maximum radius $\triangle_{max}$, and set $k = 0$.
Step 1: Definition of POD-based models Compute snapshots of forward and adjoint equations, $\boldsymbol{X}_k$ based on the control $\tilde{\mathbf{u}}_k$, compute $\mathcal{J}_b(\tilde{\mathbf{u}}_k)$. Compute POD basis functions for forward and adjoint equations and build reduced-order models for both equations.
Step 2: Step calculation Compute step $\boldsymbol{\delta}_k$ by solving the subproblem (42)
Step 3: Definition of trust-region ratio Compute new snapshots $\boldsymbol{X}_{k+}$ based on $\tilde{\mathbf{u}}_k + \boldsymbol{\delta}_k$ and $\mathcal{J}_b(\tilde{\mathbf{u}}_k + \delta_k)$.
 Define the ratio
   $$\rho_k = \frac{ared_k(\delta_k)}{pred_k(\delta_k)} = \frac{\mathcal{J}_b(\tilde{\mathbf{u}}_k + \boldsymbol{\delta}_k) - \mathcal{J}_b(\tilde{\mathbf{u}}_k)}{\mathcal{J}_{b,k}^{\mathcal{R}}(\tilde{\mathbf{u}}_k + \boldsymbol{\delta}_k) - \mathcal{J}_{b,k}^{\mathcal{R}}(\tilde{\mathbf{u}}_k)}.$$
Step 4: Trust-region update:
    •If $\rho_k \geq \eta_3$ :
        Set $\tilde{\mathbf{u}}_{k+1} = \tilde{\mathbf{u}}_k + \boldsymbol{\delta}_k$,
        $\mathcal{J}_b(\tilde{\mathbf{u}}_{k+1}) = \mathcal{J}_b(\tilde{\mathbf{u}}_k + \boldsymbol{\delta}_k)$, $\boldsymbol{X}_{k+1} = \boldsymbol{X}_{k+}$.
        Update trust-region radius $\triangle_{k+1} = \min(\triangle_{max}, \gamma_2 \triangle_k)$.
        Set $k = k + 1$ and go to Step 1.
    •If $\eta_2 \leq \rho_k < \eta_3$ :
        Set $\tilde{\mathbf{u}}_{k+1} = \tilde{\mathbf{u}}_k + \boldsymbol{\delta}_k$,
        $\mathcal{J}_b(\tilde{\mathbf{u}}_{k+1}) = \mathcal{J}_b(\tilde{\mathbf{u}}_k + \boldsymbol{\delta}_k)$, $\boldsymbol{X}_{k+1} = \boldsymbol{X}_{k+}$.
        Update trust-region radius $\triangle_{k+1} = \triangle_k$
        Set $k = k + 1$ and go to Step 1.
    •If $\eta_1 \leq \rho_k < \eta_2$ :
        Set $\tilde{\mathbf{u}}_{k+1} = \tilde{\mathbf{u}}_k + \boldsymbol{\delta}_k$,
        $\mathcal{J}_b(\tilde{\mathbf{u}}_{k+1}) = \mathcal{J}_b(\tilde{\mathbf{u}}_k + \boldsymbol{\delta}_k)$, $\boldsymbol{X}_{k+1} = \boldsymbol{X}_{k+}$.
        Update trust-region radius $\triangle_{k+1} = \gamma_2 \triangle_k$.
        Set $k = k + 1$ and go to Step 1.
    •If $\rho_k < \eta_1$ or $\rho_k = \infty$ :
        Set $\tilde{\mathbf{u}}_{k+1} = \tilde{\mathbf{u}}_k$,
        Update trust-region radius $\triangle_{k+1} = \gamma_1 \triangle_k$.
        Set $k = k + 1$ and go to Step 2.
```

Remark 1 The subproblem (42) is not the standard quadratic model approximation as in the trust-region globalization strategy. Instead, it is an approximation of the high-fidelity model. In Fahl (2000), an algorithm based on the Cauchy condition is used to solve the subproblem. In this work, we use the KNITRO optimization package Byrd et al. (2006) for finding optimal steps $\boldsymbol{\delta}_k$.

Remark 2 The stopping criteria applied in high-fidelity optimization are usually the absolute changes in the objective function or constraints violation as described in Algorithm 3. In the TRPOD method, the stopping criterion based on the trust-region radius. If the trust-region radius is less than the minimum trust-region radius $\triangle_{min}$, then the optimization is terminated. Since the objective function of the surrogate model can be lower (maximization case) than in the previous iteration, which may yield negative value of ρ_k, we take absolute value of ρ_k. Moreover the value of ρ_k can be infinite due to constraint violation, we hence reduce the trust-region radius.

Remark 3 The bound constraint in the high-fidelity optimization $g(\mathbf{u}^n)$ is adjusted in the reduced-space optimization due to the infinite norm constraints on the steps $\boldsymbol{\delta}_k$. The optimization in surrogate model may be stopped if the bound constraint in high-fidelity optimization is violated.

Remark 4 The trust-region parameters in the TRPOD method are chosen as follows

$$\eta_1 = 0.02,\ \eta_2 = 0.5,\ \eta_3 = 1,\ \gamma_1 = 0.25,\ \gamma_2 = 0.5,\ \gamma_3 = 1.5.$$

The small value of $\eta_1 = 0.02$ means we accept small improvement in the objective function value.

Remark 5 The discussion on convergence and convergence rate of the algorithm can be found in Fahl (2000), Agarwal (2010).

It should be noted that to speed up the optimization convergence we employ the BFGS method using the first-order gradient from the adjoint method. Alternatively, one may use the SR1 algorithm to approximate the Hessian matrix, which is quite common in the trust-region scheme.

4.2 Lagrangian barrier methods

Since we also handle state constraints, in this work we employ the Lagrangian barrier method (a simplified version of Conn et al. 1997), which requires a Lagrangian barrier function

$$\mathcal{J}_b(\tilde{\mathbf{u}}, \boldsymbol{\lambda}, \mu) = \mathcal{J}(\tilde{\mathbf{u}}) + \mu \sum_{i=1}^{n_h} \lambda_i \log(h_i(\hat{\mathbf{x}}^n, \mathbf{u}^n)). \tag{44}$$

Here μ is the barrier parameter and λ_i is the componentwise Lagrange multiplier estimates, which are updated during the course of optimization. The Lagrangian barrier method is described in Algorithm 3. The TRPOD method is used in step 1 of the Lagrangian barrier method. This method will terminate either due to (most likely) objective function criterion or constraint violation. We refer to Suwartadi et al. (2012) for further details of algorithm discussion and its uses for production optimization.

Algorithm 3 Lagrangian Barrier method

```
Step 0: **Initialization**
  - Set feasible initial solution of control $\tilde{\mathbf{u}}_0$, step $\boldsymbol{\delta}$, and positive initial multiplier $\boldsymbol{\lambda}_0$
  - Set convergence tolerances:
    - gradient tolerance $\omega_* \ll 1$
    - constraint violation $\eta_* \ll 1$
    - objective function changes $\epsilon_*$
  - Set positive barrier parameter $\mu_0$ and $\tau < 1$
  - Set initial convergence tolerance: $\omega_0$.
Perform iteration: $k = 0, 1, 2, \ldots$
  Step 1: Inner iteration
    - Find $\boldsymbol{\delta}$ such that $\left\|\nabla \mathcal{J}_{b,k}^{\mathcal{R}}\right\| \le \omega_k$
  Step 2: Test for convergence
    - If $\left\|[h_i(\tilde{\mathbf{x}}_k)]_{i=1}^m\right\| \le \eta_*$
     or $|\mathcal{J}_{k+1}(\tilde{\mathbf{x}}, \tilde{\mathbf{u}}) - \mathcal{J}_k(\tilde{\mathbf{x}}, \tilde{\mathbf{u}})| \le \epsilon_*$ then stop.
    - Check $\left\|\bar{\boldsymbol{\lambda}}_{h,k} - \boldsymbol{\lambda}_{h,k}\right\| \le \frac{\tau_k}{\mu_k}$
    - If this holds continue to Step 3.
     Otherwise go to Step 4.
  Step 3: Update Lagrangian Multipliers
    - $\mu_{k+1} = \mu_k$
    - $\omega_{k+1} = \tau_k \omega_k$
    - $\boldsymbol{\lambda}_{h,k+1} = \bar{\boldsymbol{\lambda}}_h(\tilde{\mathbf{x}}_k, \boldsymbol{\lambda}_{h,k}, \mu_{k+1})$
    - Continue to Step 1.
  Step 4: Update Barrier parameter
    - $\mu_{k+1} = \tau_k \mu_k$
    - $\tau_{k+1} = \mu_k^{0.5}$
    - Continue to Step 1.
```

5 Case examples

In this section, we present four case examples. The first case will compare POD and DEIM in building reduced-order models in terms of CPU time and its accuracy. The second example will demonstrate how the TRPOD method works in an optimization case without the presence of nonlinear output constraints. The last two case examples will show how the TRPOD and Lagrangian methods handle the nonlinear output constraints in surrogate optimization. Simulations for these case examples were done on a 64-bit Linux box with Intel(R) Xeon(R) CPU @ 3.00GHz. All the SVD computation in the case examples are done by using the SVD function in MATLAB. We use the *economical* option (*SVD("*", "con")*) in order to choose the eigenvectors corresponding to the largest singular values.

5.1 Case 1

The reservoir model in this case is taken from layer 10 of SPE 10th comparative study Christie and Blunt (2001). The grid consists of 60×220 gridblocks, where the dimension of a grid block is $10\,\text{ft} \times 20\,\text{ft} \times 2\,\text{ft}$. The connate oil saturation and residual water saturation are zero. The porosity, for simplicity, is set homogenously to 0.3, while the permeability is heterogenous as depicted in Fig. 2. The oil to water mobility ratio is set to 0.2 and initial water saturation is zero. The well configuration is a 5-spot pattern with an injector in the middle and four producers at the corners. The simulation is run for 1,200 days and the control inputs are the well rates. We divide the control inputs into 40 intervals, which means we can change the well rates every 30 days. The number of control variables are $40 \times 5 = 200$. Initial injection rate is set to $0.5\ \frac{PVI}{1{,}200\,days}$. Moreover, the snapshots of the forward and adjoint equations are taken from the 40 control intervals.

In this case example, we compare reduced-order models obtained from the POD and DEIM methods. The reduced-order models are constructed based on the snapshots of the forward and adjoint equations. For the POD method, the snapshots for the forward equations comprise the solution of pressures and water saturation for 40 control steps. While for the DEIM method, we need additional water cut snapshots representing the nonlinear terms, which is also from 40 control steps. In the adjoint equations, since there is no nonlinear term, we apply the POD method. Thus, the snapshot for the adjoint equations will be the solution of the adjoint equations. We will explain the reduced-order models for both types of equations in the following subsection.

5.1.1 Forward equations

The runtime of the high-fidelity model is described in Table 1. To build reduced-order models for the forward equations, we choose an energy level truncation. We vary the value of energy truncation in order to know a good value or dimension of the reduced-order models. Furthermore, we define the error of the reduced-order model by the following equation

$$\mathcal{E}:=\frac{\|\mathbf{s}^N-\hat{\mathbf{s}}^N\|_2}{\|\mathbf{s}^N\|_2}, \tag{45}$$

where $\mathbf{s}^N$ is water saturation at final time step N, and $\hat{\mathbf{s}}^N$ is the reduced-order water saturation at the final time step.

Figures 3, 4, and 5 depict the singular values of the state variables: pressure, saturation, and the nonlinear term, water cut.

We then run simulations with a variation of energy truncations and the results are displayed in Table 2. In general the POD method gives significant speedup for the pressure equation compared to that of the high-fidelity model runtime described in Table 1. However, only a slight CPU time reduction is obtained for the saturation equation. On the other hand, DEIM gives more speedup for the saturation equation. The approximation errors and the CPU time speedups decrease when the number of basis functions increase.

To show the quality of reduced-order model POD and DEIM, we display water saturation at the end time using energy truncation 90 % for the pressure, 90 % for the water saturation, and 90 % for the water cut in Fig. 6.

5.1.2 Adjoint equations

Here, we continue to vary the energy truncation of the adjoint equations. The runtime for the high-fidelity model of the adjoint equations is described in Table 3. Furthermore, we plot the singular values of the corresponding pressure and saturation equations in Figs. 7 and 8.

Similarly, we have done some simulations using variation of energy truncations and the results are described in Table 4. We define the error of the adjoint-gradient in the reduced-order model by the following equation

$$\mathcal{E}_{\text{grad}} := \frac{\|\mathbf{grad}-\hat{\mathbf{grad}}\|_2}{\|\mathbf{grad}\|_2}, \tag{46}$$

which compare the gradient in high-fidelity (**grad**) and in reduced-order ($\hat{\mathbf{grad}}$).

Both the POD and DEIM methods, shown in Table 4, give speedup in runtime compare to the adjoint equations in high-fidelity described in Table 3. We run both POD and DEIM for the adjoint equations since they need forward reduced-order models. Furthermore, the CPU time for corresponding adjoint saturation is comparable to that of high-fidelity runtime. This is because of the sparsity property in the linear adjoint saturation equation. In the high-fidelity equation the adjoint saturation is solved using a sparse linear solver. However, in a reduced-order model we loose the sparsity structure of the adjoint saturation equation. One may get better speedup for the adjoint saturation equation if the reservoir model has a larger number of grid blocks.

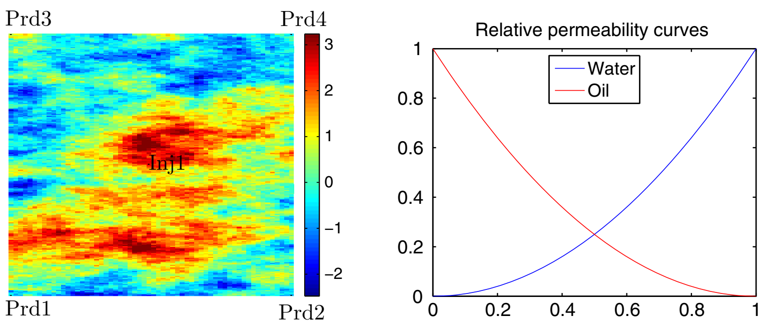


Fig. 2 The logarithm of permeability field in millidarcy (mD), well location and relative permeability curves. The well locations follow the 5-spot pattern in which four producers (Prd1, Prd2, Prd3, Prd4) are placed in the corners and one injector (Inj1) in the middle

Table 1 CPU time measured in second for forward equations using the high-fidelity model

Pressure Eq.	Saturation Eq.	Total time
15.6	5.3	20.9

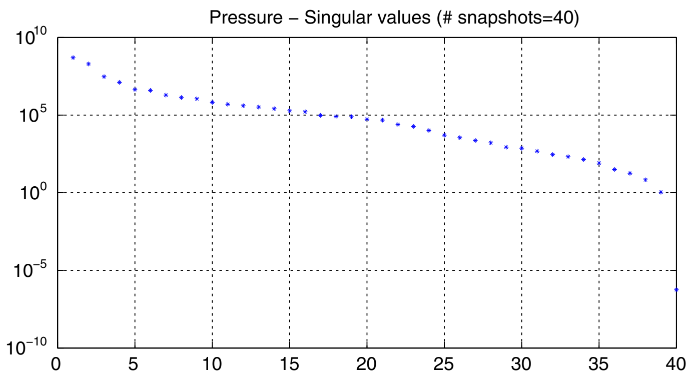


Fig. 3 Singular values of pressure snapshots

We also present the quality of the gradient approximation in the reduced-order models in Fig. 9, where the truncation is 90 % for both the corresponding pressure and saturation.

5.1.3 *Effect of perturbations*

In order to know the robustness of basis functions, we first build a reduced order model using DEIM with 90 % energy truncation for pressure, saturation, and water cut. We then change well rates at producer wells around 5, 10, and 20 % in the sense

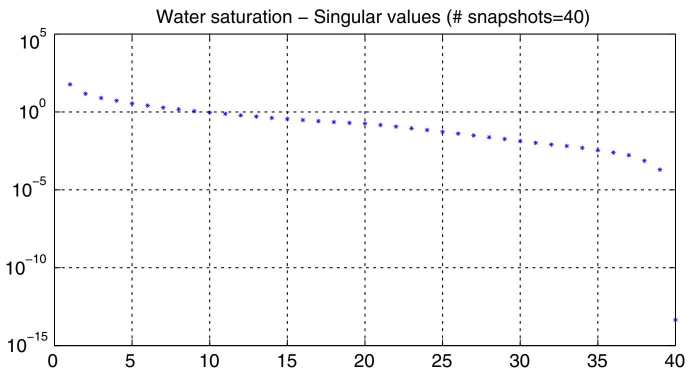


Fig. 4 Singular values of water saturation snapshots

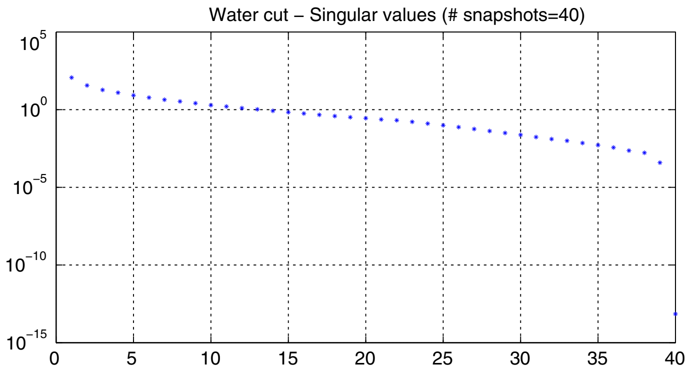


Fig. 5 Singular values of water cut snapshots

that we perturb the initial well rates when the basis functions are constructed. As seen in Figs. 10 and 11 below, the reduced-order model is good enough to approximate the high-fidelity model. However, it is not good enough for the 20 % perturbation as depicted in Fig. 12. The relative error saturation is the water saturation difference between high-fidelity model and reduced-order model divided by saturation in high-fidelity model.

5.2 Case 2

In this case we set up a surrogate optimization without any output constraints. The goal is to show the performance of TRPOD method compared to the optimization using a high-fidelity model. Since there are no nonlinear output constraint, the constraints appear only on the control, that is, bound constraints and an equality constraint due to the incompressible flow (the total injector rate must be equal the total producer rate). The objective function in this case is net present value (NPV) with oil price $80\frac{\$}{m^3}$, water separation cost $19\frac{\$}{m^3}$, and water injection cost $1\frac{\$}{m^3}$. It

Table 2 Comparison POD and DEIM in the variation of energy truncation

Energy truncation			# Basis functions			CPU time (s)			Error Sat.
Pres. (%)	W. Sat. (%)	W. Cut (%)	Pres.	W. Sat.	W. Cut	Pres. Eq.	Sat. Eq.	Total time	
POD									
90	90	–	4	9	–	4.2	3.3	7.5	2.1×10^{-2}
	95	–		13	–		3.6	7.8	1.3×10^{-2}
	99	–		24	–		4.6	8.8	4.0×10^{-3}
95	90	–	5	9	–	4.5	4.4	8.9	2.1×10^{-2}
	95	–		13	–		4.8	9.3	1.3×10^{-2}
	99	–		24	–		5.4	9.9	3.0×10^{-3}
99	90	–	11	9	–	5.1	4.1	9.2	2.1×10^{-2}
	95	–		13	–		4.1	9.2	1.3×10^{-2}
	99	–		24	–		4.8	9.9	3.0×10^{-3}
99.9	99.9	–	24	35	–	6.9	6.2	13.1	1.0×10^{-4}
99.99	99.99	–	35	40	–	9.6	6.4	16.0	1.0×10^{-4}
DEIM									
90	90	90	4	9	9	4.4	2.5	6.9	2.0×10^{-2}
	95	95		13	13		2.7	7.1	1.0×10^{-2}
	99	99		24	22		3.8	8.1	1.0×10^{-2}
95	90	90	5	9	9	4.6	2.3	6.9	2.0×10^{-2}
	95	95		13	13		2.7	7.3	1.0×10^{-2}
	99	99		24	22		4.1	8.7	9.0×10^{-3}
99	90	90	11	9	9	5.2	2.6	7.8	2.0×10^{-2}
	95	95		13	13		2.8	8.0	1.0×10^{-2}
	99	99		24	22		3.8	9.0	1.0×10^{-2}
99.9	99.9	99.9	24	35	34	6.8	5.0	11.8	1.0×10^{-3}
99.99	99.99	99.99	35	40	38	9.8	5.3	15.1	1.0×10^{-4}

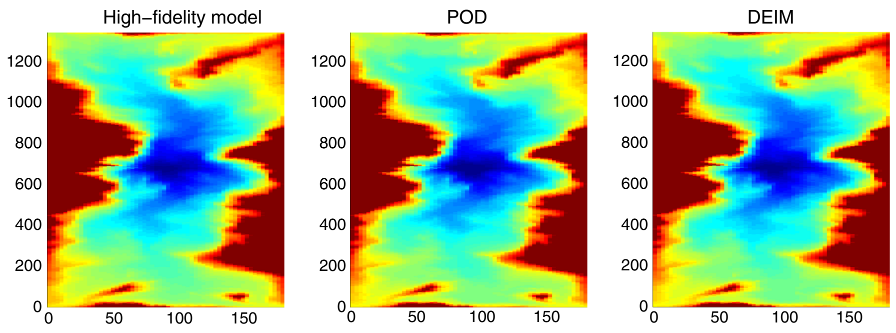


Fig. 6 Comparison of water saturation at final time for the high-fidelity model and reduced order models; POD and DEIM

Table 3 CPU time measured in second for adjoint equations using the high-fidelity model

Pressure Eq.	Saturation Eq.	Total Time
15.9	2.1	18.0

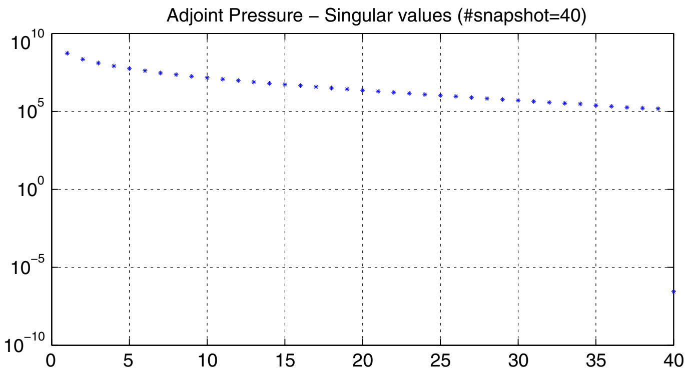


Fig. 7 Singular values of corresponding adjoint pressure equation snapshots

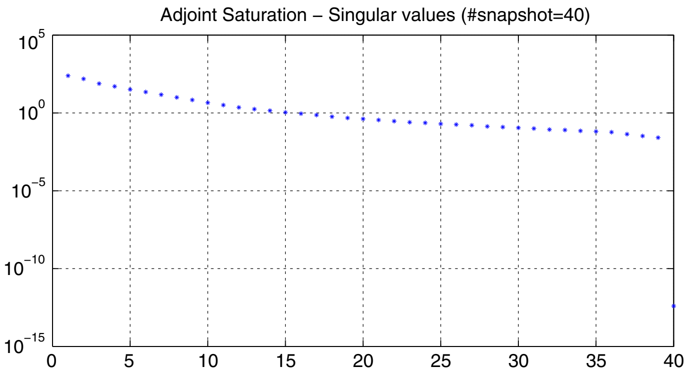


Fig. 8 Singular values of corresponding adjoint water saturation equation snapshots

should be noted there is no augmented objective function in this case. Furthermore, we continue using the reservoir setting described in case 1 with initial injection rate is 0.4 PVI for 1,200 simulation days (40 control intervals). The control inputs are well rates at producer and injector wells. The reduced-order model is built using DEIM due to its faster CPU time than POD.

To fully capture the dynamics of reservoir, we extend the simulation a bit further until 1,800 days ensuring a water cut value of 0.80 reaches all producer wells (based on the price setting). This is performed when building the initial basis functions but not during the basis functions update within the TRPOD strategy. We use an energy level of 99 % for the forward pressure, 95 % for saturation equation, and 95 % for the water cut. For the adjoint equations we use 95 % energy level for the

Table 4 Adjoint POD and DEIM in variation of energy truncation

Fwd. En. Trunc.			Adj. En. Trunc.		# Adj. Basis func.		CPU time (s)			Error
Pres. (%)	S_w (%)	F_w (%)	Adj. Pres. (%)	Adj. S_w (%)	Adj. Pres.	Adj. S_w (%)	Adj. Pres.	Adj. S_w	Total time	
POD										
90	90	–	90	90	7	6	6.5	2.1	8.6	8.0×10^{-3}
	95	–	95	95	10	8	8.2	2.2	10.4	5.0×10^{-3}
	99	–	99	99	4	24	15.9	2.4	18.3	4.0×10^{-3}
95	90	–	90	90	7	6	7.3	2.1	9.4	8.0×10^{-3}
	95	–	95	95	10	8	8.0	2.1	10.1	4.0×10^{-3}
	99	–	99	99	19	13	16.1	2.4	18.5	2.0×10^{-3}
99	90	–	90	90	7	6	7.1	2.1	9.2	7.0×10^{-3}
	95	–	95	95	10	8	8.7	2.1	10.8	3.0×10^{-3}
	99	–	99	99	19	13	16.1	2.4	18.5	1.0×10^{-3}
99.9	99.9	–	99.9	99.9	34	26	26.8	3.7	30.5	1.0×10^{-3}
DEIM										
90	90	90	90	90	7	6	6.6	2.2	8.8	9.0×10^{-3}
	95	95	95	95	10	8	8.9	2.3	11.1	3.0×10^{-3}
	99	99	99	99	19	13	16.2	2.4	18.6	4.0×10^{-3}
95	90	90	90	90	7	6	6.5	2.1	8.6	8.0×10^{-3}
	95	95	95	95	10	8	8.8	2.2	13.0	4.0×10^{-3}
	99	99	99	99	19	13	16.1	2.4	18.5	2.0×10^{-3}
99	90	90	90	90	7	6	6.8	2.1	8.9	8.0×10^{-3}
	95	95	95	95	10	8	8.8	2.2	13.0	4.0×10^{-3}
	99	99	99	99	19	13	16.2	2.4	18.6	1.0×10^{-3}
99.9	99.9	99.9	99.9	99.9	34	26	28.3	3.8	32.1	1.0×10^{-3}

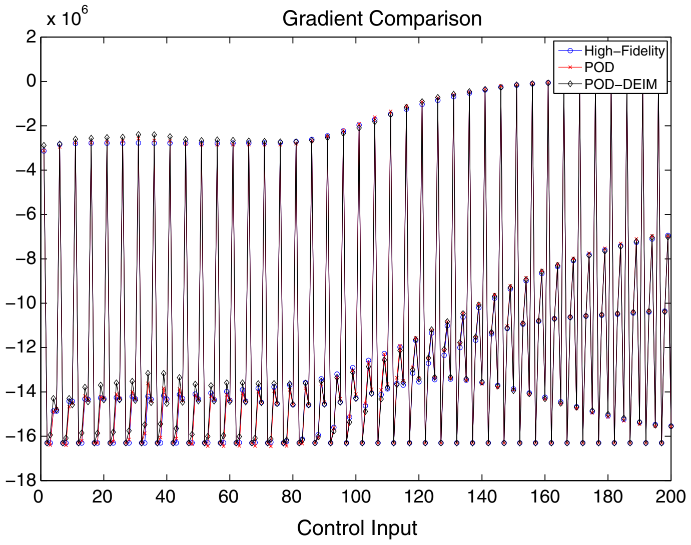


Fig. 9 Comparison of adjoint-gradient in high-fidelity and reduced-order models (POD and DEIM)

corresponding pressure and saturation equations. Using these energy truncations, an initial forward reduced-order model consists of 12, 16, and 15 basis functions for the pressure, saturation, and water cut, respectively. The interpolation points are shown in Fig. 13. The adjoint reduced-order model has 13 and 8 basis functions for the corresponding pressure and saturation equations, respectively.

We then run surrogate optimization. To evaluate the optimization, we also run the optimization using the high-fidelity model. The stopping criteria are the absolute gradient tolerance 10^{-8} and absolute step length 10^{-8}. These stopping criteria apply both for reduced and high-fidelity model optimizations. The surrogate optimization is run with an initial trust-region radius, $\triangle_0$, set to $0.03 \times \frac{V_{gb}}{1{,}200\,\text{days}}$, the maximum trust-region radius, $\triangle_{max}$, is $0.03 \times \frac{V_{gb}}{1{,}200\,\text{days}}$. This maximum trust-region radius represents the bound in which the reduced order model is robust enough to the control input perturbation. Moreover, the minimum trust-region radius $\triangle_{min}$, is $10^{-3} \times \frac{V_{gb}}{1{,}200\,\text{days}}$. The minimum trust-region ratio ρ_{min} is set 0.001. The bound constraints on the control input are set between 0 and $0.6 \times \frac{V_{gb}}{1{,}200\,\text{days}}$.

After running the optimization, the evolution of the objective function is depicted in Fig. 14. Note that the number of iterations in the high-fidelity optimization represents the number of inner iteration while the number of iterations in the surrogate optimization denotes the number of outer iterations, involved in the TRPOD method. Table 5 describes the runtime and obtained objective function values. The details of the surrogate optimization are described in Table 6, where $\hat{\mathcal{J}}$ is the objective function in reduced-order model and $\mathcal{J}$ is the objective function evaluated in high-fidelity model. The surrogate optimization terminates due to the minimum trust-region radius.

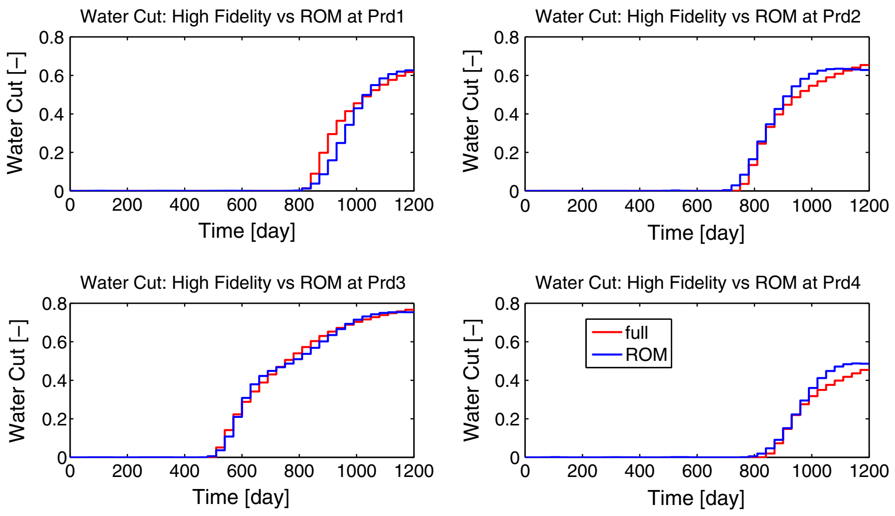


Fig. 10 5 % variation of producers well rates with relative error in saturation approximation of 0.021

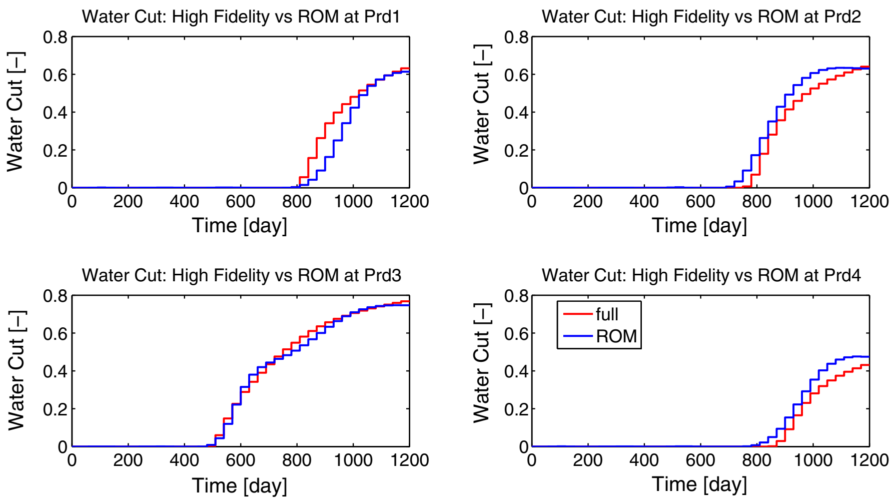


Fig. 11 10 % variation of producers well rates with relative error in saturation approximation of 0.029

In Table 6 above, the trust-region ratio ρ determines when the basis functions must be updated as well as when the radius ratio will be enlarged or shrunk. The trust-region parameter settings are described in Sect. 4. The basis functions and the trust-region will not be updated and enlarged if the trust-region ratio ρ has value less than 0.02. The trust-region radius will be enlarged only if the trust-region ratio is larger than 1. Otherwise, it will be kept or reduced. Hence, starting from the fifth iteration, seen in Table 6, the trust-region radius is shrunk. In the seventh iteration, the trust-region radius is kept as the same previous iteration and is decreased

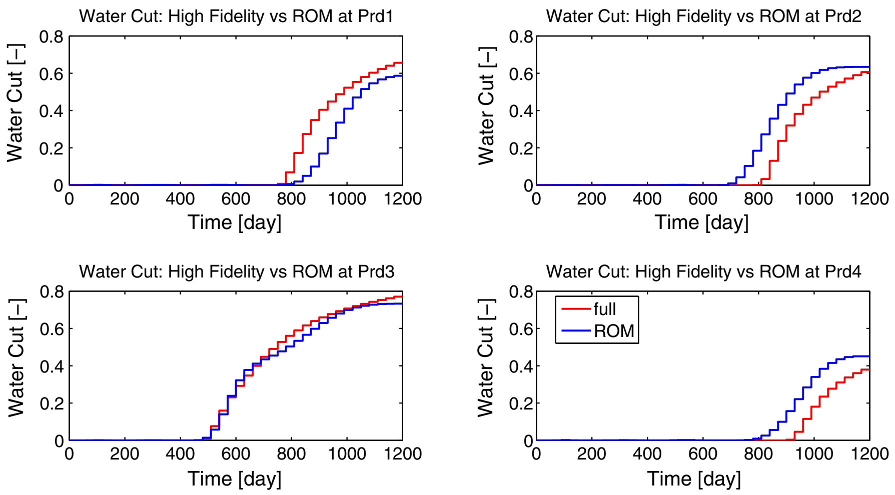


Fig. 12 20 % variation of producers well rates with relative error in saturation approximation of 0.052

afterwards. Consequently, the optimization terminates due to the minimum trust-region radius criterion. Fig. 15 shows the comparison of optimization solutions in high-fidelity and surrogate optimization. Here, we denote the injector rate with positive sign and producer rate with negative sign.

Next, we run another optimization with initial injection rate of 0.5 $\frac{PVI}{1{,}200\,days}$ using the same parameter values (initial trust-region radius). This initial rate is closer to the optimization solution in a high-fidelity model. The objective function evolutions are described in Fig. 16. The details of the optimization in the reduced-space model can be seen in Table 8 and CPU time speedup is described in Table 7. The optimization in high-fidelity model terminates due to the step length tolerance, while in surrogate optimization stops because it hits the upper bound constraint, that is, $0.6 \times \frac{V_{gb}}{1{,}200\,\text{days}}$.

It turns out that the surrogate optimization reaches a different local maximum than optimization with high-fidelity model. However, the runtime is still quite cheap. The optimization in high-fidelity seems to converge in the same local maxima but with the cost of higher CPU time. The speedup factor in this case can be up to 20 times.

5.3 Case 3

This case is a continuation of the previous case, which uses the same reservoir, and well setting, however, with the inclusion of output constraints and a more accurate reduced-order model with an increased energy truncation for saturation and water cut to 99 %. The objective function is now recovery factor (RF), described in (40). In this case we constrain the water fractional flow (water cut), which is function of

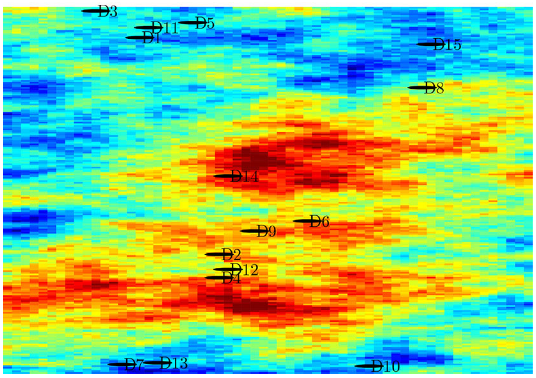


Fig. 13 Interpolation points (represented with D) for the nonlinear water cut term are located at grid blocks: 12076, 4285, 13031, 3445, 12622, 5495, 314, 10308, 5129, 282, 12437, 3746, 378, 7106, and 11869

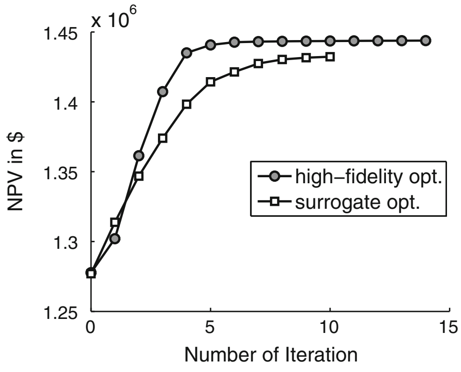


Fig. 14 Evolution of the objection functions using the initial injection 0.4 $\frac{PVI}{1{,}200\,days}$

Table 5 Comparison of optimization in high-fidelity and reduced-space using initial injection of 0.4 $\frac{PVI}{1{,}200\,days}$

Comparison	Full-model	POD-DEIM
NPV (in $)	1.44×10^6	1.43×10^6
CPU time (s)	5,715	1,270

water saturation, at the producer wells. We limit the water cut for the producer wells at the final time to $f_{w,max}$, which is set to 0.80. To this end, the augmented objective function is

$$\mathcal{J}_b(\tilde{\mathbf{u}}, \boldsymbol{\lambda}, \mu) = \mathcal{J}(\tilde{\mathbf{u}}) + \mu \sum_{i=1}^{4} \lambda_i \log\left(f_{w,max} - f^N_{w,prod_i}\right). \tag{47}$$

The parameter settings in this case are: $\boldsymbol{\lambda}_0 = [1 \quad 1 \quad 1 \quad 1]^T$, $\tau = 0.1$, $\mu_0 = 10^4$, $\omega_0 = 10^{-6}$, absolute maximum water cut tolerance $\eta_* = 10^{-6}$, and absolute objective function changes $\epsilon_* = 10^{-4}$ percent of recovery factor. We choose an active set algorithm in KNITRO to the handle control input constraints $g(\tilde{\mathbf{u}})$. The initial trust-region radius, $\triangle_0$, and the maximum trust-region radius, $\triangle_{max}$, are set

Table 6 Iteration in surrogate optimization using initial injection of 0.4 $\frac{PVI}{1{,}200\,days}$

k	ρ	$\hat{\mathcal{J}}$	$\mathcal{J}$
0	–	1.27×10^6	1.27×10^6
1	1.16×10^1	1.28×10^6	1.32×10^6
2	1.07×10^0	1.31×10^6	1.35×10^6
3	9.28×10^{-1}	1.34×10^6	1.37×10^6
4	8.09×10^{-1}	1.37×10^6	1.39×10^6
5	2.37×10^{-1}	1.44×10^6	1.41×10^6
6	2.34×10^{-1}	1.41×10^6	1.42×10^6
7	5.67×10^{-1}	1.42×10^6	1.43×10^6
8	3.44×10^{-1}	1.43×10^6	1.43×10^6
9	3.88×10^{-1}	1.43×10^6	1.43×10^6
10	3.28×10^{-1}	1.43×10^6	1.43×10^6

equally to $0.01 \times \frac{V_{gb}}{1200\,\text{day}}$, the minimum trust-region radius $\triangle_{min}$, is $10^{-4} \times \frac{V_{gb}}{1200\,\text{day}}$, and minimum trust-region ratio ρ_{min} is 0.001. The bound constraints on the control input are set between 0 and $0.8 \times \frac{V_{gb}}{1{,}200\,\text{days}}$. We run two optimizations with different initial injector settings.

5.3.1 *Initial injection rate 0.5* $\frac{PVI}{1{,}200\,days}$

We start the optimization with an initial injector rate 0.5 $\frac{PVI}{1{,}200\,days}$. The optimization with the high-fidelity model stops due to the objective function change criterion. Furthermore, Table 9 describes the results and constraint violations are shown in Fig. 18. The comparison of the objective function evolution is displayed in Fig. 19. The infinite objective function value in the reduced-order space indicates that the output constraint is violated.

The surrogate optimization terminates because of the minimum trust-region radius. POD-DEIM in this case gives a speedup more than 1.3 times. The output constraints are active only for *Prd*1 in the reduced-order model while for the other production wells, the constraints are very close to being active.

In Fig. 20 the optimization solutions using high-fidelity and reduced-order model are shown. The producer rates are shown in negative sign while the injector rate is in positive sign. Again, it is clear the surrogate optimization resulted in a local maximum. The figure also shows water saturation at final time step. As seen, the water saturation is slightly different around *Prd*4 area.

5.3.2 *Initial injection rate 0.4* $\frac{PVI}{1{,}200\,days}$

We continue the optimization using reduced-order model with different initial solution. Here, we set initial injection rate to 0.4 $\frac{PVI}{1{,}200\,days}$. The results are shown in

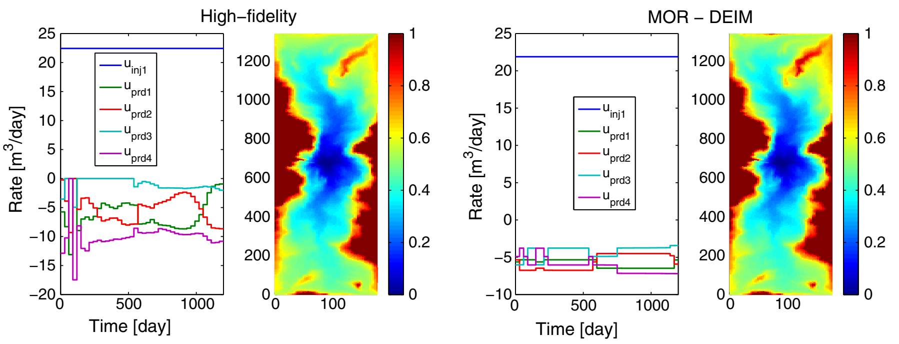


Fig. 15 Optimization solutions in high-fidelity and reduced-space models and water saturation at final time

Fig. 16 Evolution of the objection functions using the initial injection of 0.5 $\frac{PVI}{1{,}200\, days}$

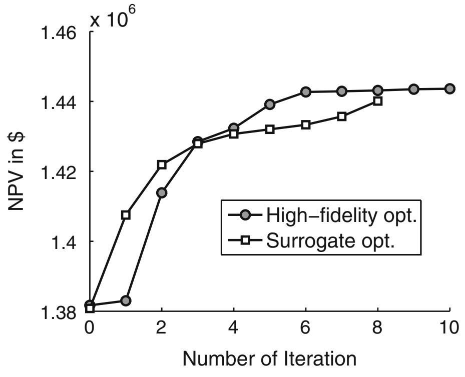


Table 7 Comparison of optimization in high-fidelity and reduced-space using initial injection of 0.5 $\frac{PVI}{1{,}200\, days}$

Comparison	Full-model	POD-DEIM
NPV ($)	1.44×10^6	1.44×10^6
CPU time (s)	26,681	1,269

Table 10, constraints satisfaction in Fig. 21. Similarly, the optimization terminates due to the minimum trust-region radius and $Prd1$ is active, while the other production wells are almost active. The evolution of the objective function is shown in Fig. 22, and control input solution is in Fig. 23.

5.4 Case 4

This case originates from the Norne comparative study Rwechungura et al. (2010) with a simplified model. The reservoir is depicted in Fig. 24 and there are 6 wells.

Table 8 Iteration in surrogate optimization using initial injection of 0.5 $\frac{PVI}{1{,}200\,days}$

k	ρ	$\hat{\mathcal{J}}$	$\mathcal{J}$
0	–	1.38×10^6	1.38×10^6
1	1.06×10^0	1.36×10^6	1.41×10^6
2	2.16×10^{-1}	1.42×10^6	1.42×10^6
3	4.75×10^{-1}	1.41×10^6	1.43×10^6
4	1.31×10^{-1}	1.43×10^6	1.43×10^6
5	7.09×10^{-1}	1.43×10^6	1.43×10^6
6	1.45×10^0	1.43×10^6	1.43×10^6
7	1.03×10^0	1.43×10^6	1.44×10^6
8	3.69×10^{-1}	1.42×10^6	1.44×10^6

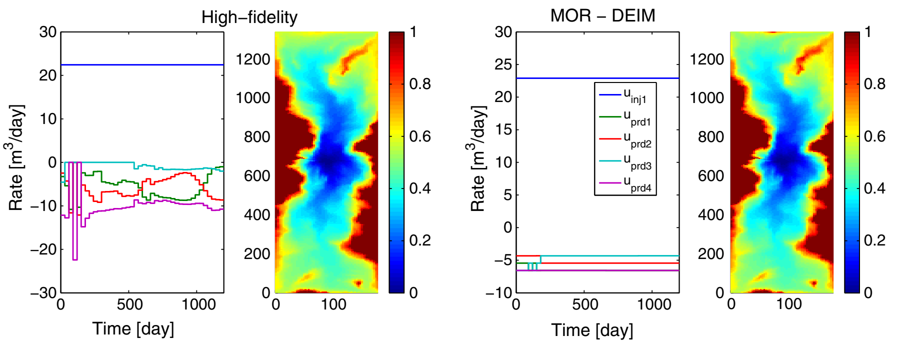


Fig. 17 Optimization solutions in high-fidelity and reduced-space models and water saturation at final time

Initial water saturation and pressures at each grid block are set to 0.2 and 40 bar, respectively. The mobility ratio between water and oil is 1 to 5. The end points of connate water saturation and oil saturation are both set to 0.2. The relative permeability curves are displayed in Fig. 24. The simulation is run for 500 days and divided into 50 control intervals. Thus, in total the controls consist of 300 variables. The controls in this case are well rates. Similar to the previous cases, we deal with equality constraints due to incompressible flow, that is, total injection rate must equal total production rate. In addition, we set bound constraint on the injectors, which is set lower than 2 $\frac{PVI}{500\,days}$. The initial water injection rates are 0.25 and 0.30 of total pore volume using constant rates. The number of snapshots for building the reduced-order models is the same as the number control of intervals, which is 50 snapshots.

In this case we constrain the total water production at the final control interval to 5×10^{-3} of the pore volume of the reservoir and define this constraint as $Q_{w,max}$. Hence, in this case the augmented objective function is

Table 9 Optimization results. The water injected is measured in pore volume injected (PVI)

Comparison	High-fidelity model	POD-DEIM
Recovery factor (%)	48.41	47.96
CPU time (s)	12,054	8,809
Total water Injected (in PVI)	0.77	0.75

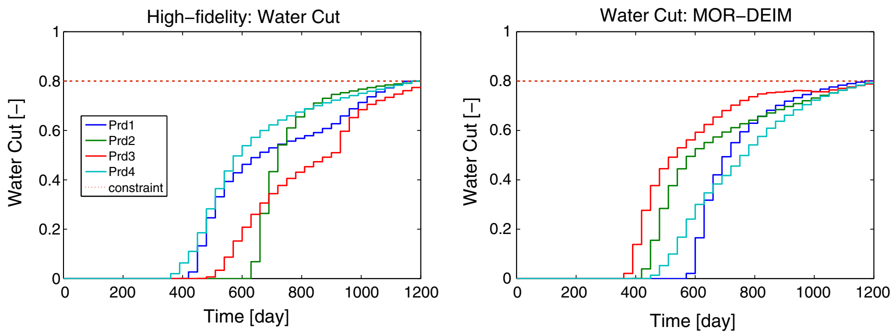


Fig. 18 The state constraints satisfaction, i.e., the water-cut at final time step

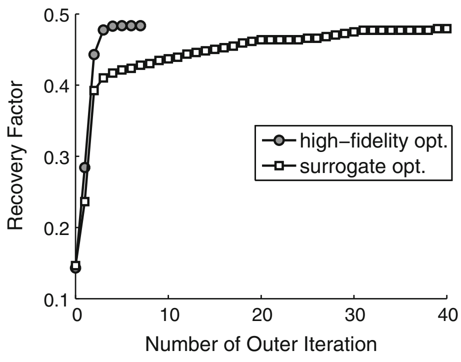


Fig. 19 Comparison of the objective function evolution in high-fidelity and reduced-space. The objective value from reduced-space in the figure is the objective function evaluation using the high-fidelity model given the solution of surrogate optimization

$$\mathcal{J}_b(\tilde{\mathbf{u}}, \lambda, \mu) = \mathcal{J}(\tilde{\mathbf{u}}) + \mu\lambda \log\left(Q_{w,max} - \sum_{i=1}^{n_h=4} Q_{w,i}^N \right). \tag{48}$$

Here the output constraint is just a scalar, that is, the total water production of each producer well at final control interval.

The reduced-order models are constructed using 90 % energy truncation for the pressure and saturation equations, and 90 % for the non-linear water cut term. This results in 4, 7, 7 basis functions for the pressure, saturation and water cut, respectively. For the adjoint equations, we use a 95 % energy level both for the

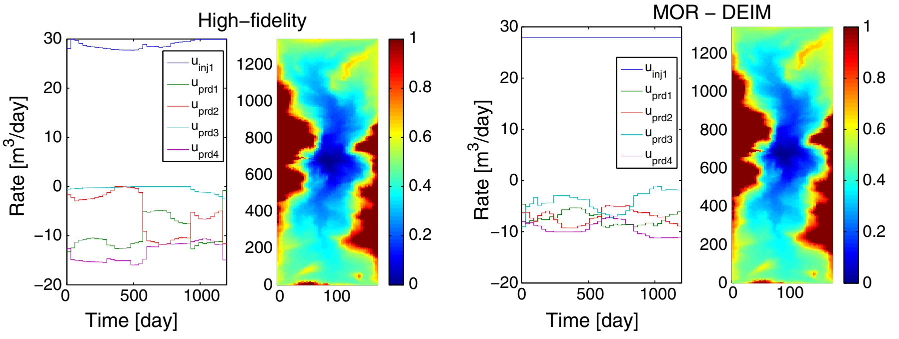


Fig. 20 Optimization solutions in high-fidelity and reduced-space models and water saturation at final time

corresponding pressure and saturation equations. To this end, the reduced-order adjoint equations have dimension 16 and 4, respectively. The CPU time comparison of the full-model and reduced-order models are described in Table 11. The POD-DEIM results are consistently faster than the standard POD. Significant speedup is obtained for the forward equations, but the corresponding saturation equation is again similar to the first case. Due to sparsity property in the high-fidelity adjoint equation and the dense matrix in the reduced-order models, the speedup of the corresponding saturation adjoint equations is not that significant. To this end, we use the reduced-order model of POD-DEIM for surrogate optimization.

The parameter settings in this case are: $\lambda = 1$, $\tau = 0.1$, $\mu = 10^7$, $\omega_0 = 10^{-3}$, and absolute total water production tolerance $\eta_* = 10^{-4}$. We choose an active set algorithm in KNITRO to handle control input constraints $g(\tilde{\mathbf{u}})$. The maximum trust-region radius, $\triangle_{max}$, is $0.1 \times \frac{V_{gb}}{500\,\text{day}}$, and minimum trust-region radius $\triangle_{min}$, is $0.001 \times \frac{V_{gb}}{500\,\text{day}}$ and initial trust-region radius, which are $\triangle_0 = 0.1 \times \frac{V_{gb}}{500\,days}$.

5.4.1 *Initial injection rate* $0.25\,\frac{PVI}{500\,days}$

We firstly run the optimization with initial injection $0.25\,\frac{PVI}{500\,days}$. The results are summarized in Table 12, the constraint satisfaction in Fig. 25, and comparison of optimized control inputs in Fig. 26. The optimization terminated due to the minimum trust-region radius.

5.4.2 *Initial injection rate* $0.3\,\frac{PVI}{500\,days}$

We then run optimization with initial injection rate $0.3\,\frac{PVI}{500\,days}$. The results are described in Table 13, Figs. 27 and 28. The optimization in this case terminates due to the minimum trust-region radius.

Table 10 Optimization results. The water injected is measured in pore volume injected (PVI)

Comparison	High-fidelity model	POD-DEIM
Recovery factor (%)	48.35	47.88
CPU time (s)	13,615	11,181
Total water Injected (in PVI)	0.77	0.74

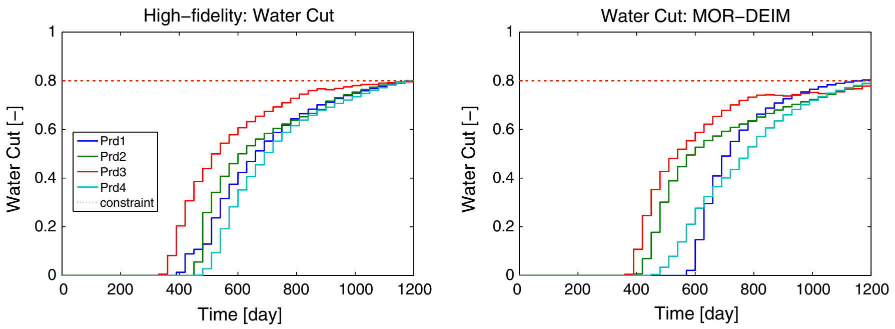
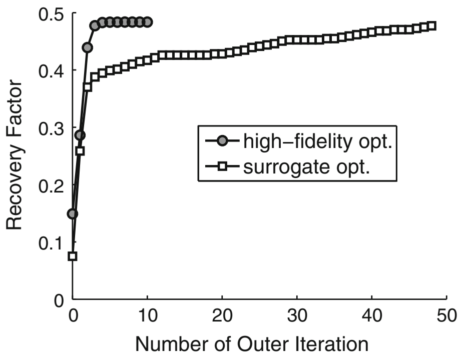


Fig. 21 The state constraints satisfaction, i.e., the water-cut at final time step

Fig. 22 Comparison of the objective function evolution in high-fidelity and reduced-space. The objective value from reduced-space in the figure is the objective function evaluation using the high-fidelity model given the solution of surrogate optimization



5.5 Discussion

We have presented four case examples. The first case example shows the quality of a reduced-order model given a variation of energy truncation. It turns out that the POD method is more CPU intensive than the POD-DEIM simulation run. Reducing the energy truncation will reduce CPU time at the expense of accuracy. The case example uses a 2D reservoir consisting of 13200 grid blocks. The POD method results in significant speedup for the pressure equation. However, for the saturation equation the CPU time in the reduced-order model is marginally faster than the high-fidelity model. In the same case example it is shown that DEIM can slightly

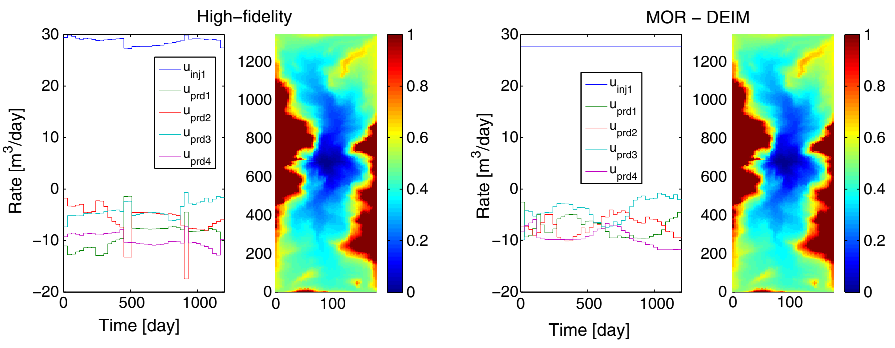
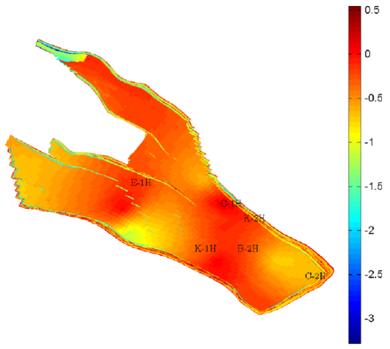


Fig. 23 Optimization solutions in high-fidelity and reduced-space models and water saturation at final time



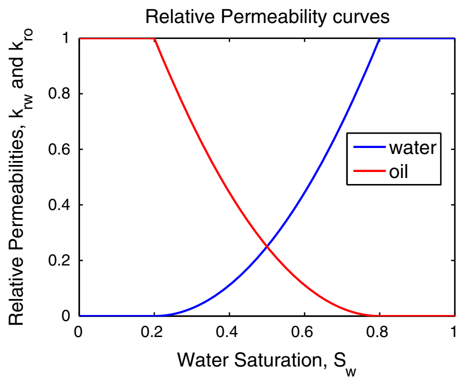


Fig. 24 Norne field, a 3D reservoir, with six wells: four producers (E-1H, K-1H, B-2H, K-2H) and 2 injectors (C-1H and C-2H). Permeability field is plotted in millidarcy (mD). The *right hand* figure shows relative permeabilities

improve the CPU time of the saturation equation. Furthermore, in the saturation adjoint equation we loose the sparsity property. Consequently, the sparse linear solver, which is used to solve the adjoint equation, consumes more CPU time.

In the second case, we demonstrate the performance of the TRPOD method. Based on the first case, we continue the optimization only with the DEIM reduced-order model because the method gives more speedup than POD. Using two different initial controls (injector rate), the TRPOD method is trapped into local maxima with comparable objective function (NPV) values. This implies that the choice of initial control and initial trust-region radius are important considerations. The speedups in this case are significant, between 5 and 20 times for the two different initial controls.

In the third case, we introduce output constraints in the surrogate optimization. The output constraints are water cut at the producer wells. This represents a multidimensional output constraint problem. The TRPOD combined with Lagrangian barrier does not achieve the same solution as the high-fidelity optimization. We

Table 11 Comparison of CPU time of forward and adjoint equations

CPU Time (s)	High-fidelity model	POD	POD-DEIM
Forward equations			
Pressure Eq.	154	32	29
Saturation Eq.	51	19	11
Total time forward Eqs.	205	51	40
Adjoint equations			
Pressure Eq.	215	92	86
Saturation Eq.	18	15	15
Total time adjoint Eqs.	233	107	101

have tried to use two different initial controls. The water cut value at one of the producer wells is active, while at the other wells are almost active. The speedups in this case are slightly faster than the high-fidelity optimization, which are around 27 and 18 %.

In the fourth case, we apply TRPOD and the Lagrangian barrier method to constrain total water production. This case is a one-dimensional output constraint problem. The results from this case show that our proposed method is able to make the optimization in reduced-space converges to a better local maxima than that of high-fidelity optimization. Moreover, the constraint is active and the speed up factor is up to four times.

The third and the fourth case examples show the performance of TRPOD for nonlinear constraint handling. As this work focuses on the nonlinear constraint handling, the speedup factor can be obtained up to 400 % for the 3D oil reservoir while only 27 % for the 2D oil reservoir.

Apart from the results above, both the TRPOD and Lagrangian barrier methods rely on some parameter settings. In the Lagrangian barrier method we need to supply suitable values of initial μ and its stopping criteria, and the TRPOD method needs more parameters, which are the minimum trust-region radius $\triangle_{min}$, its initial value $\triangle_0$, and its maximum value $\triangle_{max}$. These parameters values have important impact on the optimization results. To find good values for them, it would be interesting to use a derivative-free optimization method, see e.g. Audet and Orban (2006), rather than to use a heuristic approach.

Another point worthwhile to note is the fact that gradient-based optimization is sensitive to initial guess values. We have therefore run some optimizations with different initial controls. The results consistently show that the surrogate optimization has substantially lower CPU time while honoring the nonlinear output constraints.

6 Conclusion

The use of the TRPOD method in two case examples has been presented in this paper. Two kinds of model order reduction techniques, the POD and POD-DEIM

Table 12 Optimization results with initial injection rate 0.25 $\frac{PVI}{500\,days}$. The water injected is measured in pore volume injected (PVI)

Comparison	High-fidelity model	POD-DEIM
Recovery factor (%)	37.52	37.61
CPU time (in hours)	9.20	2.32
Total water Injected (in PVI)	0.31	0.32

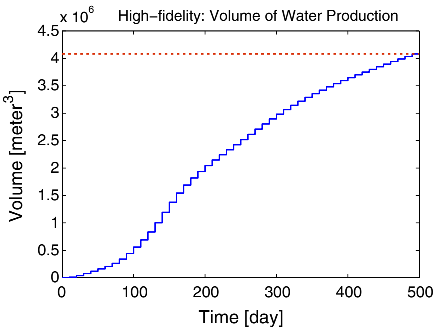


Fig. 25 The output constraints satisfaction, i.e., the total volume of water production at the final time step

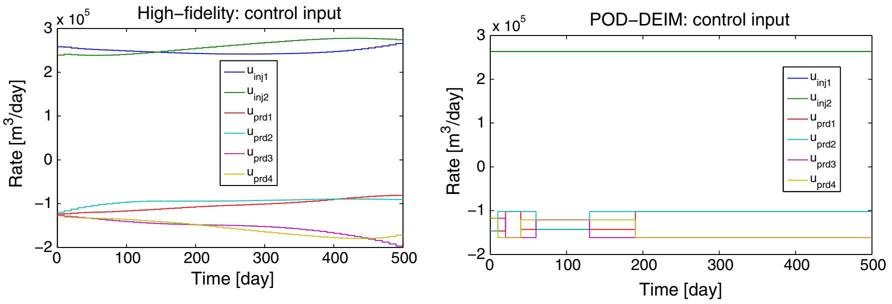


Fig. 26 Optimization solutions in high-fidelity and reduced-space models

Table 13 Optimization results with initial injection rate 0.3 $\frac{PVI}{500\,days}$. The water injected is measured in pore volume injected (PVI)

Comparison	High-fidelity model	POD-DEIM
Recovery factor (%)	37.29	37.28
CPU time (in hours)	7.30	3.57
Total Water Injected (in PVI)	0.32	0.32

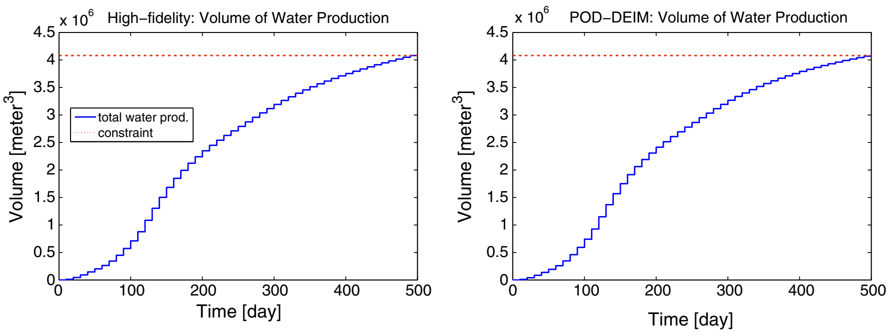


Fig. 27 The output constraints satisfaction, i.e., the total volume of water production at the final time step

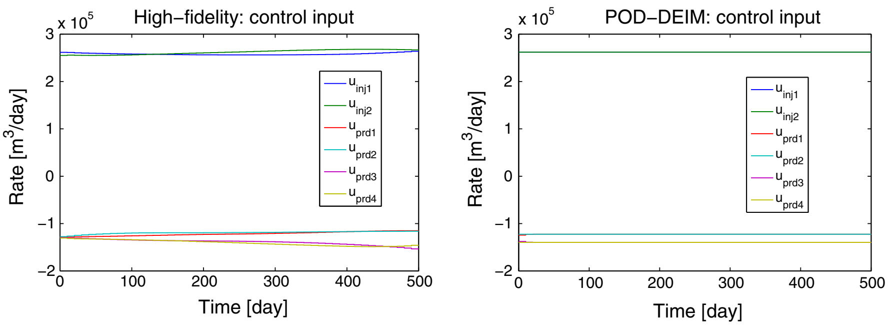


Fig. 28 Optimization solutions in high-fidelity and reduced-space models

methods, have been presented. Because of the nonlinear nature of oil reservoirs, particularly the water saturation equation, the POD method may result in a slight speedup in terms of CPU runtime. To get more CPU time speedup, we use POD-DEIM that is consistently faster for the forward saturation equation. In the 2D reservoir example, the sparse linear solver seems to be efficient to solve the linear equation of the adjoint systems. Hence, the corresponding adjoint-saturation equation in the reduced-order model cannot be much faster. The surrogate optimization using the POD-DEIM reduced-order model has shown to give considerable speedup and may also give a comparable objective function value. In addition, result from surrogate optimization can be used as an initial guess to an optimization algorithm using a high-fidelity model.

The Lagrangian barrier method is sensitive to the choice of algorithm parameters, such as the barrier multiplier. In addition, the TRPOD method also requires suitable choices of parameter values. The choice of these parameters will affect the optimization solution.

The state equations in this work are solved using a sequential method, that is, implicit-pressure and implicit-saturation solvers. The POD-DEIM is then applied to the implicit water saturation equation. In commercial reservoir simulators, a fully-implicit method, that is, implicit solutions for both pressure and saturation are commonly used. We foresee the use of POD-DEIM in fully-implicit reservoir simulators may give even better speedups than observed in this work.

In this work the choice of POD basis functions is paramount. Due to the limited operating point, the POD basis functions are updated with a trust-region strategy during the course of optimization. However, there exist many variants of POD methods that extend the operating range of the POD model. For example the extrapolation strategy proposed in Burkardt et al. (2006), may give better approximation. Furthermore, the TRPOD method depends heavily on some important parameters, namely, the initial and maximum trust-region radius. Method like adaptive TRPOD Sachs (2009) can be applied to partially overcome dependency of these parameters.

Acknowledgments The authors gratefully acknowledge support from the Integrated Operation Centers at NTNU (www.iocenter.no). The authors would also like to thank the anonymous referees for their valuable input.

References

Aarnes JE, Gimse T, Lie KA (2007) An introduction to the numerics of flow in porous media using Matlab. Springer Verlag, New York

Agarwal A (2010) Advanced strategies for optimal design and operation of pressure swing adsorption processes. Ph.D. thesis, Carnegie Mellon University

Agarwal A, Biegler LT (2011) A trust-region framework for constrained optimization using reduced order modeling. Optim Eng 14:3–35. doi:10.1007/s11081-011-9164-0

Alexandrov NM, Lewis RM, Gumbert CR, Green LL, Newman PA (2001) Approximation and model management in aerodynamic optimization with variable-fidelity models. J Aircr 38(6):1093–1101

Astrid P, Weiland S, Willcox K, Backx T (2008) Missing point estimation in models described by proper orthogonal decomposition. IEEE Trans Autom Control 53(10):2237–2251

Audet C, Orban D (2006) Finding optimal algorithmic parameters using derivative-free optimization. SIAM J Optim 17:642–664

Aziz K, Settari A (1979) Petroleum reservoir simulation. Applied Science Publisher, London

Barrault M, Maday Y, Nguyen NC, Patera AT (2004) An 'empirical interpolation' method: application to efficient reduced basis discretization of partial differential equations. Comptes Rendus Mathematique 339(9):667–672

Burkardt J, Gunzburger M, Lee HC (2006) Centroidal voronoi tessellation-based reduced-order modeling of complex systems. SIAM J Sci Comput 28(2):459–484

Byrd RH, Nocedal J, Waltz RA (2006) Knitro: an integrated package for nonlinear optimization. Large-Scale Nonlinear Optim 83:35–59. doi:10.1007/0-387-30065-1-4

Cardoso MA, Durlofsky LJ (2010) Linearized reduced-order models for subsurface flow simulation. J Comput Phys 229(3):681–700

Cardoso MA, Durlofsky LJ (2010) Use of reduced-order modeling procedures for production optimization. SPE J 15(2):426–435

Cardoso MA, Durlofsky LJ, Sarma P (2009) Development and application of reduced-order modeling procedures for subsurface flow simulation. Int J Numer Methods Eng 77(9):1322–1350

Chaturantabut S, Sorensen DC (2010) Nonlinear model order reduction via discrete empirical interpolation. SIAM J Sci Comp 32(5):2737–2764

Chaturantabut S, Sorensen DC (2011) Application of POD and DEIM on dimension reduction of non-linear miscible viscous fingering in porous media. Math Comput Model Dyn Syst 17(4):337–353

Christie MA, Blunt MJ (2001) Tenth SPE comparative solution project: a comparative of upscaling technique. SPE Reserv Eval Eng 4:308–317

Conn AR, Gould NIM, Toint PL (1997) A globally convergent Lagrangian Barrier algorithm for optimization with general inequality constraints and simple bounds. Math Comput 66(217):216–288

Conn AR, Gould NIM, Toint PL (2000) Trust-region methods. SIAM, Philadelphia

Davis TA (2006) Direct methods for sparse linear systems. SIAM, Philadelphia

Fahl M (2000) Trust-region methods for flow control based on reduced order modelling. Ph.D. thesis, Trier University

Heijn T, Markovinovic R, Jansen JD (2004) Generation of low-order reservoir models using system-theoretical concepts. SPE J 9(2):202–218

Hinze M, Volkwein S (2005) Proper orthogonal decomposition surrogate models for nonlinear dynamical systems: error estimates and suboptimal control. In: Benner P, Mehrmann V, Sorensen DC (eds) Reduction of large-scale systems, lecture notes in computational science and engineering, vol 45. Springer, Berlin

Krogstad S (2011) A sparse basis POD for model reduction of multiphase compressible flow. SPE-141973. In: Proceeding of the SPE 2011 reservoir simulation symposium

Krogstad S, Hauge VL, Gulbransen AF (2011) Adjoint multiscale mixed finite elements. SPE J 16(1):162–171. doi:10.2118/119112-PA

Lie KA, Krogstad S, Ligaarden IS, Natvig LR, Nilsen HM, Skafltestad B (2011) Open-source MATLAB implementation of consistent discretisations on complex grids. Comput Geosci Online First (2011). DOI 10.1007/s10596-011-9244-4. URL http://sintef.org/Projectweb/MRST/

Markovinovic R, Geurtsen EL, Heijn T, Jansen JD (2002) Generation of low-order reservoir models using POD, empirical gramians and subspace identification. In: Proc. 8th European conference on the mathematics of oil recovery (ECMOR). Freiberg, Germany, pp. E31-1-E31-10

Markovinovic R, Geurtsen EL, Jansen JD (2002) Subspace identification of low-order reservoir models. In: Proceeding of IV international conference on computational methods in water resources, Delft, The Netherlands. pp. 281–288

Markovinovic R, Jansen JD (2006) Accelerating iterative solution methods using reduced-order models as solution predictors. Int J Numer Methods Eng 68(5):525–541

Peaceman D (1983) Interpretation of well-block pressures in numerical reservoir simulation with nonsquare grid blocks and anisotropic permeability. SPE J 23(3):531–543

Robinson TD (2007) Surrogate-based optimization using multifidelity models with variable parameterization. Ph.D. thesis, Massachusetts Institute of Technology

Rwechungura R, Suwartadi E, Dadashpour M, Kleppe J, Foss B (2010) The norne field case -a unique comparative case study. SPE 127538. In: Proceeding of SPE intelligent energy conference and exhibition, Utrecht, The Netherlands

Sachs EW (2009) Adaptive trust region POD algorithms. URL http://www.mathematik.uni-trier.de/∼schu/poster_DFG.pdf

Suwartadi E, Krogstad S, Foss B (2010) A Lagrangian-Barrier Function for Adjoint State Constraints Optimization of Oil Reservoir Water Flooding. In: Proceeding of IEEE conference on decision and control 2010, Atlanta, Georgia, USA

Suwartadi E, Krogstad S, Foss B (2012) Nonlinear output constraints handling for production optimization of oil reservoirs. Comput Geosci 16(2):499–517

van Doren JFM, Markovinovic R, Jansen JD (2006) Reduced-order optimal control of water flooding using proper orthogonal decomposition. Comput Geosci 10(1):137–158

Volkwein S (2003) Model reduction using proper orthogonal decomposition. In: Lecture Note. Institute of Mathematics and Scientific Computing. University of Graz. URL http://www.uni-graz.at/imawww/volkwein/POD.pdf

Wen Z, Durlofsky LJ, van Roy B (2011) Use of approximate dynamic programming for production optimization. SPE 141677. In: Proceeding of the SPE 2011 reservoir simulation symposium