Accelerating the solution of linear systems appearing in two-phase reservoir simulation by the use of POD-based deflation methods

Diaz Cortes, Gabriela Berenice; Vuik, Cornelis; Jansen, Jan Dirk

DOI
10.1007/s10596-021-10041-6

Publication date
2021

Document Version
Final published version

Published in
Computational Geosciences

Citation (APA)
Diaz Cortes, G. B., Vuik, C., & Jansen, J. D. (2021). Accelerating the solution of linear systems appearing in two-phase reservoir simulation by the use of POD-based deflation methods. *Computational Geosciences*, *25*(5), 1621-1645. https://doi.org/10.1007/s10596-021-10041-6

Computational Geosciences
https://doi.org/10.1007/s10596-021-10041-6

ORIGINAL PAPER

Accelerating the solution of linear systems appearing in two-phase reservoir simulation by the use of POD-based deflation methods

Gabriela Berenice Díaz Cortés[1] · Cornelis Vuik[2] · Jan-Dirk Jansen[3]

Received: 14 July 2020 / Accepted: 26 January 2021

Abstract
We explore and develop a Proper Orthogonal Decomposition (POD)-based deflation method for the solution of ill-conditioned linear systems, appearing in simulations of two-phase flow through highly heterogeneous porous media. We accelerate the convergence of a Preconditioned Conjugate Gradient (PCG) method achieving speed-ups of factors up to five. The up-front extra computational cost of the proposed method depends on the number of deflation vectors. The POD-based deflation method is tested for a particular problem and linear solver; nevertheless, it can be applied to various transient problems, and combined with multiple solvers, e.g., Krylov subspace and multigrid methods.

Keywords Deflation · Krylov Methods · Porous media · Two-phase reservoir simulation

1 Introduction

Solutions of systems of linear equations are required when simulating flow through subsurface porous media [27]. These linear systems emerge during the iterative solution of time- and space-discretized nonlinear partial differential equations that govern the porous media flow problems. In the case of multi-phase flow, the equations involve an elliptic or parabolic pressure equation coupled to one or more near-hyperbolic saturation equations.

There are several ways to treat this system of equations, among others, the Implicit Pressure Explicit Saturation (IMPES), Fully Implicit (FI) and Adaptive Implicit Methods (AIM) [13, 25]. In this work we use the IMPES scheme implemented in MRST [30], that decouples the pressure from the saturation via the fractional flow formulation, see Appendix B and [47].

For the IMPES scheme, the saturation is calculated explicitly, while pressure is kept implicit, and solving the pressure equation is the most time-consuming part, especially for large and ill-conditioned systems, which often occur because of the complex geometry and strongly heterogeneous rock properties that constitute the porous medium. Furthermore, for time-varying problems, it is required to compute a large number of simulations, which makes the solution of these problems expensive.

Even larger computational demands occur when subsurface flow models are used in workflows for optimization or uncertainty quantification which typically require the simulation of a vast number of models. Various techniques have, therefore, been developed to improve the linear solver speed.

Among others, Reduced Order Models (ROM) methods are used to capture relevant information of a high-dimensional system and to project it into a lower-dimension space [3, 10, 28, 41, 50], which is easier to solve. With these methods, essential system information can be obtained by computing a small set of basis functions from a collection of system solutions (also known as 'snapshots'). Proper Orthogonal Decomposition (POD) is a ROM method that has recently been used to accelerate the solution of the linear pressure equation resulting from reservoir simulation [24, 35, 40, 42, 48], among other applications.

For the computation of the POD basis, two main approaches are used. In the first one, a training simulation

✉ Gabriela Berenice Díaz Cortés
gbdiaz@imp.mx

✉ Cornelis Vuik
c.vuik@tudelft.nl

[1] Mexican Petroleum Institute (IMP), Eje Central Lazaro Cardenas Norte 152, Col. San Bartolo Atepehuacan, 07730, Mexico City, Mexico

[2] Delft University of Technology, Faculty of Electrical Engineering, Mathematics and Computer Science, Van Mourik Broekmanweg 6, 2628 XE Delft, Netherlands

[3] Delft University of Technology, Faculty of Civil Engineering and Geosciences, Stevinweg 1, 2628 CN Delft, Netherlands

Published online: 07 June 2021

is run, and the solutions are stored as snapshots, which are collected to obtain a POD basis. This methodology is especially suitable for solving problems with small changes in the input variables, e.g. the same well configurations but different flow rates or bottom hole pressures (*bhp*) [35, 40, 48]. The basis can also be computed on-the-fly, using, e.g., the solution of the latest time steps [16, 40, 42]. With this approach, the basis has to be adapted during the simulation. Once the basis is obtained, various POD-based strategies can be used to solve the system. In the future, we will refer to the first approach as *training phase* approach, and the second as *moving window* approach.

For the solution of a large-scale system, Markovinovic et al. [42] proposed using POD techniques to compute a good initial guess that accelerates the iterative method for simulation of two-phase flow through heterogeneous porous media. For the same type of problems, solving the problem in the small-scale domain, and projecting it back to the large-scale system was implemented by Astrid et al. [40].

For many applications, Krylov subspace iterative methods are used [19, 46] to accelerate the solution of linear systems. The speed of convergence of these methods depends on the condition number and the right-hand side (*rhs*) of the system. If the condition number is large, generally, preconditioning techniques are needed to transform the original system into a better conditioned one. If the system is Symmetric Positive (Semi) Definite (SP(S)D), a commonly used Krylov-subspace method is Conjugate Gradient (CG) [8, 9, 20, 26, 28]. If the system is not SP(S)D, methods like GMRES are used. These methods can be accelerated further with preconditioning [32, 45], or recycling strategies like augmentation and deflation [8].

For some Krylov methods, the computation of an orthogonal basis for the entire Krylov subspace is required to obtain an approximate solution. As the number of iterations increases, the work and the storage requirements grow dramatically. To reduce the extra work, some methods are restarted after some number of iterations. For the GMRES method, system information is obtained from previous solutions or from the Ritz vectors and combined with restarting, truncation and deflation techniques to accelerate the method [11, 34, 43, 44].

If the spectrum of the system contains few small eigenvalues, or if the initial error vector has small components for the lowest eigenvalues modes, the rate of convergence of the initial sublinear phase is more rapid. Such an initial error vector can be obtained using preconditioners based on multilevel meshes working from coarser meshes to finer ones [38]. In recent years, deflation techniques have been developed to accelerate the convergence of Krylov subspace methods [8, 9, 22, 26, 49]. Especially useful when a sequence of linear systems with constant or slightly varying matrices has to be solved [31]. For this technique to be effective, a deflation subspace needs to be found. This subspace is such that the smallest eigenvalues of the system are no longer hampering the convergence of the iterative method.

The reuse of system information obtained from POD has been increasingly approached recently. Carlberg et al. [28] proposed a POD-augmented CG algorithm for Krylov-subspace recycling applied to solid-mechanics problems. Another approach was developed by Pasetto et al. [10], who suggested constructing a preconditioner for the CG method, based on a POD basis for the solution of groundwater flow models. The use of the POD basis within a deflation procedure to accelerate the CG method was introduced by Diaz Cortes et al. [16] for single-phase flow simulation problems.

In this work, we extend the work of Diaz Cortes et al. [16] to two-phases, and we introduce the capture of information via POD methods with a *training phase* approach besides the *moving window* approach proposed in [16]. The acquired information is used for the construction of the above-mentioned deflation subspace. We explore the applicability of this methodology for the simulation of two-phase flow in large-scale, highly-heterogeneous porous media.

In Section 2, we give a brief overview of the methods used to solve linear systems. Section 3 is devoted to numerical experiments, where we give some examples and present some results. Finally, we formulate the conclusions. The governing equations used for the simulation of two-phase flow problems, together with the discretization schemes used in this work can be consulted in Appendices A and C.

2 Solution methods for linear systems

Iterative techniques are preferred over direct methods to approximate the solution of ill-conditioned, large and sparse linear systems. In this section, we give a brief overview of the Krylov iterative methods, used to accelerate the solution of this kind of systems. We also introduce the most popular acceleration techniques for Krylov subspace methods: preconditioning, augmentation, and deflation. Finally, we introduce the POD method that together with preconditioning and deflation are used for the acceleration of the CG method throughout this work.

Krylov subspace Given a linear system

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{1}$$

with $\mathbf{A} \in \mathbb{C}^{N\times N}$, the initial residual $\mathbf{r}^0 := \mathbf{b} - \mathbf{A}\mathbf{x}^0$, where $\mathbf{x}^0 \in \mathbb{C}^N$ is an initial guess of $\mathbf{x}$, and k is a positive integer. We define the k-dimensional Krylov subspace as:

$$\mathcal{K}_k\left(\mathbf{A}, \mathbf{r}^0\right) = span\left\{\mathbf{r}^0, \mathbf{A}\mathbf{r}^0, \ldots, \mathbf{A}^{k-1}\mathbf{r}^0\right\}. \tag{2}$$

For Krylov iterative methods, the solution can be obtained with the iteration procees presented below, for the $k-th$ iteration we have:

$$\mathbf{x}_k \in \mathbf{x}_0 + \mathcal{K}_k\left(\mathbf{M}^{-1}\mathbf{A}; \mathbf{r}_0\right).$$

Conjugate gradient (CG) method The CG method is a Krylov subspace method used for Symmetric Positive (Semi) Definite matrices (SP(S)D). For this method, the convergence bound is given by:

$$||\mathbf{x}-\mathbf{x}^k||_{\mathbf{A}} \leq 2||\mathbf{x}-\mathbf{x}^0||_{\mathbf{A}}\left(\frac{\sqrt{\kappa_2(\mathbf{A})}-1}{\sqrt{\kappa_2(\mathbf{A})}+1}\right)^k, \tag{3}$$

where the condition number $\kappa_2(\mathbf{A})$ is defined as $\kappa_2(\mathbf{A}) = \frac{\sqrt{\lambda_{max}(\mathbf{A}^T\mathbf{A})}}{\sqrt{\lambda_{min}(\mathbf{A}^T\mathbf{A})}}$. As $\mathbf{A}$ is SPD, $\kappa_2(\mathbf{A}) = \frac{\lambda_{max}(\mathbf{A})}{\lambda_{min}(\mathbf{A})}$.

Typically, the convergence of the Krylov subspace methods takes place in three phases with different rates of convergence. In the first phase, the CG method lacks information about the spectrum, and acts mainly on the information of the initial iteration error, leading to a short but rapid initial convergence rate, also known as the sublinear phase. Later, the influence of the spectral condition number becomes larger and the method enters the linear phase. Finally, when a sufficient number of extreme eigenvalue components have been damped out, the method enters the so called superlinear convergence phase [4, 6, 38].

Acceleration of the Krylov subspace methods can be achieved for the three stages. By improving the initial guess so that the error is minimized, by changing the condition number of the system, or by removing the influence of the smallest eigenvalues of the system matrix. There are several methods to achieve acceleration. Next, we present some of them.

Preconditioning To accelerate the convergence of Krylov subspace methods, the linear system is multiplied by a matrix $\mathbf{M}^{-1}$, such that the iteration matrix has a better spectrum and $\mathbf{M}^{-1}$ is cheap to compute. The resulting preconditioned system is:

$$\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}. \tag{4}$$

The convergence bound of the preconditioned iterative method is given by:

$$||\mathbf{x}-\mathbf{x}^k||_{\mathbf{A}} \leq 2||\mathbf{x}-\mathbf{x}^0||_{\mathbf{A}}\left(\frac{\sqrt{\kappa\left(\mathbf{M}^{-1}\mathbf{A}\right)}-1}{\sqrt{\kappa\left(\mathbf{M}^{-1}\mathbf{A}\right)}+1}\right)^k, \tag{5}$$

where the condition number of the preconditioned system is smaller than the one of the original system, i.e.,

$$\kappa\left(\mathbf{M}^{-1}\mathbf{A}\right) < \kappa(\mathbf{A}).$$

For the CG method, a commonly used preconditioner is the Incomplete Cholesky (IC) factorization, given by $\mathbf{A} = \mathbf{C}^T\mathbf{C} + \mathbf{R}$, with $\mathbf{R}$ the error matrix of the factorization. In this work, as preconditioner, we use the IC factorization of order 0, meaning that the non-zero entries of the $\mathbf{C}$ matrix are the same as the non-zero entries of the system matrix $\mathbf{A}$. We refer to the Conjugate Gradient method preconditioned with the IC factorization as ICCG.

2.1 Augmentation and deflation techniques for Krylov methods

Even after preconditioning, the spectrum of the preconditioned system matrix can contain few small eigenvalues that tend to slow down the convergence rate of the Krylov subspace methods. Therefore, it is required to use techniques that treat these eigenvalues. Among others, deflation and augmentation methods are used to remove the influence of these eigenvalues [12, 37, 39].

As the iteration of a Krylov method proceeds, the Ritz values converge, i.e., extreme eigenvalues of the system matrix are approximated by the corresponding Ritz values. When this happens, the method behaves as if the corresponding eigenvectors are no longer present [4, 21]. Thus, it is necessary to perform some iterations before the Ritz values are sufficiently approximated, and the influence of the corresponding extreme eigenvalues is eliminated.

Augmented Krylov subspace methods Aim to add information about the problem into the search space by including into the Krylov subspace an augmentation space $\mathcal{U}$ that, otherwise, could be obtained only after some Krylov iterations [39]. With these methods, the usual search space $\mathcal{K}_k\left(\mathbf{A}, \mathbf{r}^0\right)$ is replaced by the augmented Krylov subspace, given by:

$$\mathcal{K}_k\left(\mathbf{A}, \mathbf{r}^0\right) + \mathcal{U}. \tag{6}$$

Hence, the solution is found with the following iterative process:

$$\mathbf{x}_k \in \mathbf{x}_0 + \mathcal{K}_k\left(\mathbf{M}^{-1}\mathbf{A}; \mathbf{r}_0\right) + \mathcal{U}.$$

Deflated Krylov methods On the other hand, instead of the inclusion of the augmentation space $\mathcal{U}$ on the search space, the iterate

$$\mathbf{x}^k \in \mathbf{x}^0 + \mathcal{K}_k\left(\mathbf{A}, \mathbf{r}_0\right), \tag{7}$$

is constructed, so that the residual $\mathbf{r}^k$ satisfies the orthogonality condition given by:

$$\mathbf{r}^k = \mathbf{P}\left(\mathbf{b} - \mathbf{A}\mathbf{x}^k\right) \perp \mathcal{K}_k\left(\mathbf{A}, \mathbf{r}_0\right), \tag{8}$$

with $\mathbf{P}$ being the deflation operator that usually contains information of the extreme eigenvalues. This approach is known as deflation approach. With deflation, the effect of

the small eigenvalues can be annihilated by setting them to zero, such that the condition number of the system is reduced before the Krylov iterative process. The method is consequently accelerated, and the superlinear convergence phase is reached earlier [1, 2, 8, 22, 29]. A formal definition is presented next.

Definition 1 Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be an SPD matrix, and $\mathbf{Z} \in \mathbb{R}^{n\times p}$ be a full rank matrix. The invertible Galerkin matrix, $\mathbf{E} \in \mathbb{R}^{p\times p}$, the correction matrix, $\mathbf{Q} \in \mathbb{R}^{n\times n}$ and the deflation matrix $\mathbf{P} \in \mathbb{R}^{n\times n}$ are defined as [26, 29, 49]:

$$\mathbf{P} = \mathbf{I} - \mathbf{AQ}, \qquad \mathbf{Q} = \mathbf{ZE}^{-1}\mathbf{Z}^T, \qquad \mathbf{E} = \mathbf{Z}^T\mathbf{AZ}. \tag{9}$$

where $\mathbf{Z} \in \mathbb{R}^{n\times p}$ is called the $deflation - subspace$ matrix, and its columns are the $deflation$ vectors or $projection$ vectors.

A good selection of the deflation vectors is usually problem-dependent, and available information of the system is, in general, used to construct them. Most of the techniques used to select deflation vectors are based on eigenvectors or approximated eigenvectors, recycling vectors [16, 33], subdomain deflation vectors [9] or multigrid and multilevel based deflation vectors [7, 18, 26]. A description of some of these choices of deflation vectors is presented next.

Eigenvectors of the system matrix as deflation vectors If the matrix $\mathbf{Z}$ contains eigenvectors corresponding to the most unfavorable eigenvalues of the system matrix $\mathbf{A}$, the deflation method set them to zero and convergence of the iterative method is achieved faster. If the deflation-subspace matrix, $\mathbf{Z}$ is chosen as p eigenvectors, $\mathbf{v}_j$, of $\mathbf{A}$, i.e., $\mathbf{Z} = \left[\mathbf{v}_1, \ldots, \mathbf{v}_p\right]$, then $\mathbf{PA}$ has the same eigenvectors as $\mathbf{A}$ and the spectrum is given by [14, 49]:

$$\sigma(\mathbf{PA}) = \left\{0, \ldots, 0, \lambda_{p+1}, \ldots, \lambda_n\right\}.$$

However, the eigenvalues are usually unknown and it is costly to obtain them. Thus, approximate eigenvalues are used instead.

Recycled vectors as deflation vectors Given a linear system $\mathbf{Ax} = \mathbf{b}$ with solution $\mathbf{x} = \sum_{i=1}^{p} c_i\mathbf{x}_i$, where the $\mathbf{x}_i'$s are linearly independent (l.i.) solutions such that $\mathbf{Ax}_i = \mathbf{b}_i$, and any other solution can be constructed as a linear combination of this solutions. These solutions can be used to construct the deflation subspace matrix

$$\mathbf{Z} = \left[\mathbf{x}_1, \ldots, \mathbf{x}_p\right].$$

Using this choice of deflation vectors, the solution of the linear system is obtained within one iteration of DCG (see the proof in [14, 16]).

To understand the spectral behavior of the deflation method when using recycling vectors, we present Lemma 1, where, for a particular case, choosing l.i. solutions as deflation vectors leads to the same behaviour of the deflation method as the use of system eigenvectors.

Lemma 1 *Let* $\mathbf{Ax} = \mathbf{b}$, *be a linear system with* $\mathbf{A} \in \mathbb{R}^{n\times n}$ *symmetric with spectrum* $\sigma(\mathbf{A}) = \{\lambda_1, \ldots, \lambda_n\}$, *and eigenvectors* $\Sigma(\mathbf{A}) = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$, *such that* $\mathbf{v}_i^T\mathbf{v}_j = \delta_{ij}$. *We assume that the solution to this system can be written as* $\mathbf{x} = \sum_{i=1}^{p} a_i\mathbf{x}_i$, *where the* $\mathbf{x}_i'$*s are linearly independent (l.i.) solutions to* $\mathbf{Ax}_i = \mathbf{b}_i$. *Rewriting the* $\mathbf{x}_i'$*s as:*

$$\mathbf{x}_i = \sum_{k=1}^{n} c_k\mathbf{v}_k, \qquad \mathbf{x}_j = \sum_{m=1}^{n} \tilde{c}_m\mathbf{v}_m,$$

and taking $c_\alpha \neq \tilde{c}_\beta$, *and* $|c_\alpha| >> |c_k|$, *with* $k \in [1, \ldots, \alpha-1, \alpha+1, \ldots, n]$, *and* $|\tilde{c}_\beta| >> |\tilde{c}_m|$, $m \in [1, \ldots, \beta-1, \beta+1, \ldots, n]$, *the spectrum of the deflated system* $\mathbf{PA}$ *is given by:*

$$\sigma(\mathbf{PA}) = \left\{\lambda_1, \ldots, \lambda_{\alpha-1}, 0, \lambda_{\alpha+1}, \ldots, \lambda_{\beta-1}, 0, \lambda_{\beta+1}, \ldots, , \lambda_n\right\}.$$

Proof The l.i solutions $\mathbf{x}_i$ and $\mathbf{x}_j$ are solution to $\mathbf{Ax}_i = \mathbf{b}_i$, thus, can be expressed as

$$\mathbf{x}_i = c_\alpha\mathbf{v}_\alpha + \sum_{k=1}^{n-1} c_k\mathbf{v}_k, \qquad \mathbf{x}_j = \tilde{c}_\beta\mathbf{v}_\beta + \sum_{m=1}^{n-1} \tilde{c}_m\mathbf{v}_m. \tag{10}$$

If $|c_\alpha| >> |c_k|$, with $k \in [1, \ldots, \alpha-1, \alpha+1, \ldots, n]$, and

$|\tilde{c}_\beta| >> |\tilde{c}_m|$, with $m \in [1, \ldots, \beta-1, \beta+1, \ldots, n]$,

we have

$$\mathbf{x}_i \simeq c_\alpha\mathbf{v}_\alpha, \qquad \mathbf{x}_j \simeq \tilde{c}_\beta\mathbf{v}_\beta. \tag{11}$$

If the deflation subspace matrix is chosen as $\mathbf{Z} = \left[\mathbf{x}_i, \mathbf{x}_j\right]$, i.e., $\mathbf{Z} = \left[c_\alpha\mathbf{v}_\alpha, \tilde{c}_\beta\mathbf{v}_\beta\right]$, then, the spectrum of the deflated system becomes:

$$\sigma(\mathbf{PA}) = \{\lambda_1, \ldots, \lambda_{\alpha-1}, 0, \lambda_{\alpha+1}, \ldots, \lambda_{\beta-1}, 0, \lambda_{\beta+1}, \ldots, , \lambda_n\},$$

i.e., the eigenvalues λ_α and λ_β corresponding to the eigenvectors $\mathbf{v}_\alpha$ and $\mathbf{v}_\beta$, respectively, are removed from the system. □

Hence, if the unfavorable eigenvalues are captured in the l.i. solutions $\mathbf{x}_i$ and $\mathbf{x}_j$, the behavior of the deflated method is the same as if these eigenvectors are used as deflation vectors.

2.2 Preconditioned deflated conjugate gradient method (DPCG)

Deflation techniques are usually combined with a preconditioner M^{-1} to achieve a larger acceleration of iterative methods. When using these techniques to accelerate the CG method, this results in the DPCG method. The pseudocode of the DPCG method is given in Algorithm 1. In this work, we implement the Deflated Preconditioned Conjugate

Gradient method with the Incomplete Cholesky factorization as preconditioner; we refer to this method as DICCG.

Algorithm 1 Preconditioned deflated conjugate gradient method (DPCG), solving $\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}$.

Give an initial guess $\mathbf{x}_0$.
Compute $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$, and $\hat{\mathbf{r}}_0 = \mathbf{P}\mathbf{r}_0$,
solve $\mathbf{M}\mathbf{y}_0 = \hat{\mathbf{r}}_0$, set $\mathbf{p}_0 = \mathbf{y}_0$.
for $k = 0, \ldots,$ until convergence
  $\hat{\mathbf{w}}_k = \mathbf{P}\mathbf{A}\mathbf{p}_k$
  $\alpha_k = \frac{(\hat{\mathbf{r}}_k, \mathbf{y}_k)}{(\hat{\mathbf{w}}_k, \mathbf{p}_k)}$
  $\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_k + \alpha_k \mathbf{p}_k$
  $\hat{\mathbf{r}}_{k+1} = \hat{\mathbf{r}}_k - \alpha_k \mathbf{w}_k$
  Solve $\mathbf{M}\mathbf{y}_{k+1} = \hat{\mathbf{r}}_{k+1}$
  $\beta_k = \frac{(\hat{\mathbf{r}}_{k+1}, \mathbf{y}_{k+1})}{(\hat{\mathbf{r}}_k, \mathbf{y}_k)}$
  $\mathbf{p}_{k+1} = \mathbf{y}_{k+1} + \beta_k \mathbf{p}_k$
end for
$\mathbf{x} = \mathbf{Q}\mathbf{b} + \mathbf{P}^T\hat{\mathbf{x}}$

For this method, the error is bounded by:

$$||\mathbf{x} - \mathbf{x}_{k+1}||_{\mathbf{A}} \leq 2||\mathbf{x} - \mathbf{x}_0||_{\mathbf{A}} \left(\frac{\sqrt{\kappa_{eff}\left(\mathbf{M}^{-1}\mathbf{PA}\right)} - 1}{\sqrt{\kappa_{eff}\left(\mathbf{M}^{-1}\mathbf{PA}\right)} + 1} \right)^{k+1},$$

were $\kappa_{eff} = \frac{\lambda_{max}(\mathbf{M}^{-1}\mathbf{PA})}{\lambda_{min}(\mathbf{M}^{-1}\mathbf{PA})}$ is the effective condition number. As mentioned before, deflation sets the smallest eigenvalues to zero; then, these eigenvalues are not longer taken into account for the iteration process. Instead, $\lambda_{min}\left(\mathbf{M}^{-1}\mathbf{PA}\right)$, the smallest non-zero eigenvalue of $\mathbf{M}^{-1}\mathbf{PA}$, is used to study the convergence.

This solution method is studied throughout this work, using an Incomplete Cholesky factorization as preconditioner. As deflation vectors we use Recycled vectors in two ways: solutions of previous time steps as deflation vectors, and a Proper Orthogonal Decomposition (POD) basis as deflation vectors. This basis is obtained from previous time step solutions. To illustrate how this POD basis is constructed, next, we introduce the POD method.

2.3 Proper orthogonal decomposition (POD)

The POD method is a Model Order Reduction (MOR) method, where a high-order model is projected onto a space spanned by a small set of orthonormal basis vectors $\Psi = \left[\psi_1\, \psi_2 \,..\, \psi_p\right]$, $\Psi \in \mathbb{R}^{n\times p}$. The basis vectors $\psi_i \in \mathbb{R}^n$ are computed from a set of s 'snapshots' $\{\mathbf{x}_i\}_{i=1}^s$, obtained by simulation or experiments [42]. The vectors $\left\{\psi_j\right\}_{j=1}^p$ are p eigenvectors corresponding to the p largest eigenvalues $\left\{\lambda_j\right\}_{j=1}^p$ of the data correlation matrix $\mathbf{R} \in \mathbb{R}^{n\times n}$,

$$\mathbf{R} := \frac{1}{s}\mathbf{X}\mathbf{X}^T \equiv \frac{1}{s}\sum_{i=1}^{s} \mathbf{x}_i \mathbf{x}_i^T, \qquad \mathbf{X} := [\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_s]. \quad (12)$$

In some cases, the covariance matrix $\overline{\mathbf{R}}$ is used instead of $\mathbf{R}$, this matrix is defined as

$$\overline{\mathbf{R}} := \frac{1}{s-1}\sum_{i=1}^{s} (\mathbf{x}_i - \overline{\mathbf{x}})\,(\mathbf{x}_i - \overline{\mathbf{x}})^T, \qquad \overline{\mathbf{X}} := [\overline{\mathbf{x}}_1, \overline{\mathbf{x}}_2, \ldots \overline{\mathbf{x}}_s], \quad (13)$$

where $\overline{\mathbf{x}} = \frac{1}{s}\sum_{i=1}^{s}\mathbf{x}_i$ is the mean of the snapshots. In this work, we also normalize the snapshots, i.e., we use the following relation:

$$\overline{\mathbf{R}} := \frac{1}{s-1}\sum_{i=1}^{s} \frac{\mathbf{x}_i - \overline{\mathbf{x}}}{||\mathbf{x}_i - \overline{\mathbf{x}}||_2} \frac{(\mathbf{x}_i - \overline{\mathbf{x}})^T}{||\,(\mathbf{x}_i - \overline{\mathbf{x}})^T\,||_2}. \quad (14)$$

If the system is large, the matrix $\overline{\mathbf{R}}$ is also large, and to compute the eigenvalues can be costly. However, it is not necessary to compute the eigenvalues of $\overline{\mathbf{R}} = \overline{\mathbf{X}}\overline{\mathbf{X}}^T \in \mathbb{R}^{n\times n}$, but instead, it is possible to compute the eigenvalues of the much smaller matrix $\mathbf{C} = \overline{\mathbf{X}}^T\overline{\mathbf{X}} \in \mathbb{R}^{s\times s}$, $s << n$. To do so, we perform the Singular Value Decomposition (SVD) of $\mathbf{C} = \mathbf{V}\mathbf{D}\mathbf{V}^T$. Here $\mathbf{V} \in \mathbb{R}^{s\times s}$ are the eigenvectors of $\mathbf{C}$, and $diag(\mathbf{D})$ are the corresponding eigenvalues, which are the same as the eigenvalues of $\overline{\mathbf{R}}$. The eigenvectors of $\overline{\mathbf{R}}$ can be obtained from [17]:

$$\mathbf{U} = \overline{\mathbf{X}}\mathbf{V}\left(\Lambda^T\right)^{\frac{1}{2}} \in \mathbb{R}^{n\times n}.$$

Once the basis is computed, the high dimensional variable $\mathbf{x} \in \mathbb{R}^n$ is approximated by a linear combination of p orthonormal basis vectors [40]:

$$\mathbf{x} \approx \sum_{i=1}^{p} c_i \psi_i. \quad (15)$$

The p eigenvectors are chosen such that they contain almost the whole variability of the snapshots. Usually, they are the maximal number of eigenvalues satisfying [42]:

$$\frac{\sum_{j=1}^{p}\lambda_j}{\sum_{j=1}^{s}\lambda_j} \leq \alpha, \qquad 0 < \alpha \leq 1, \quad (16)$$

with α close to 1. The eigenvalues λ_j are ordered from large to small with λ_1 being the largest eigenvalue of $\overline{\mathbf{R}}$.

Once the basis Ψ is obtained, the linear system from Eq. 1 is projected onto the subspace spanned by the basis [40] as follows:

$$\Psi^T\mathbf{A}\Psi\mathbf{x} = \Psi^T\mathbf{b},$$

leading to the reduced model:

$$\mathbf{A}_r\mathbf{x} = \mathbf{b}_r \qquad \mathbf{A}_r \in \mathbb{R}^{p\times p}, \mathbf{b}_r \in \mathbb{R}^p.$$

The reduced model is dense; however, it is much smaller than the original system and can be solved efficiently with direct methods.

2.4 POD-based deflation method

In this work, we further explore and develop the POD-based deflation method introduced in [16]. Instead of solving the reduced model, as in [28, 35], we propose the reuse of a POD basis, Ψ, as the subspace-deflation matrix, $\mathbf{Z}$, in a deflation procedure. With this method, the residual is projected into the deflated space constructed with few elements of the basis Ψ, as presented in Eq. 8. The approximate solution is obtained from the Krylov subspace directly ($\mathcal{K}_k$) for the full system.

The problems we study contain a few small eigenvalues of the system matrix $\mathbf{A}_t$ that cause slow convergence. Deflation methods can remove these eigenvalues if the corresponding eigenvectors are used. As shown in Eq. 3, for the case of the CG method, the convergence depends on the ratio of the largest (λ_{max}) to the smallest (λ_{min}) eigenvalues. Thus, when the smallest system eigenvalue (λ_{min}) is 'removed' from the system, the convergence depends on the "effective" condition number, given as the ratio between the largest (λ_{max}) and the second smallest (λ_{min-1}) system eigenvalues [4, 5].

A POD basis, ideally, can get the main system information from a set of snapshots, $\mathbf{X}$. In particular, the basis should be an approximation of the system eigenvectors. As an example, we take as snapshot an eigenvector of the system matrix $\mathbf{A}\mathbf{v}_k = \lambda_k \mathbf{v}_k$, $\mathbf{X} = [\mathbf{v}_k]$. Then, the correlation matrix is given by

$$\mathbf{R} = \mathbf{X}\mathbf{X}^T = \mathbf{v}_k \mathbf{v}_k^T = \mathbf{v}_k \mathbf{I} \mathbf{v}_k^T,$$

which implies that $\Psi = [\mathbf{v}_k]$, and using this basis in a deflation procedure, the corresponding eigenvalue λ_i is removed from the system. Thus, if the eigenvalues of the system matrix $\mathbf{A}$ hampering the performance of an iterative method are captured on the POD basis, they can be removed using this basis as deflation vectors. Taking a snapshot $\mathbf{x}_i$ as in Lemma 1, if it contains information about one eigenvalue, i.e., $\mathbf{x}_i \sim \mathbf{v}_i$, a basis obtained from it will be the same as before, i.e., $\Psi = [\mathbf{v}_k]$. When used as a deflation vector, this eigenvalue will be removed from the system.

Methodology In this work, we study the performance of the Deflated Preconditioned Conjugate Gradient method preconditioned with Incomplete Cholesky (DICCG) and a POD basis as deflation vectors. For this method, the complete solution strategy consists of three stages:

i) **Snapshots collection**. A set $\mathbf{X}$ of snapshots, vectors $\mathbf{b}_i$ with different configurations or at various time steps, is obtained for the computation of the POD basis. To capture the snapshots, we use two approaches. The first one is a moving window approach, and the second one is a training simulation approach, both of them used for non-invariant systems,

$$\mathbf{A}^t \mathbf{x}^t = \mathbf{b}^t.$$

These approaches are further explained below.

Moving window approach: the window consists of a set of s snapshots updated for each time step. The snapshots are captured 'on the fly,' i.e., they are the most recently computed solutions, obtained during the previous time steps. Given a time step t, we collect the previous s snapshots (from $t-s-1$ to $t-1$). These snapshots are given by:

$$\mathbf{X} = \left[\mathbf{x}^{t-s-1}, \ldots, \mathbf{x}^{t-1}\right].$$

With this approach, the first t snapshots cannot be computed for the first time steps with the DICCG method, as we lack a deflation subspace matrix $\mathbf{Z}$. Instead, the first s time steps are computed with the ICCG method. The rest of the snapshots are obtained with DICCG.

Training phase approach: a full pre-simulation is run. For a pre-simulation consisting of n time steps. The set of snapshots is given by:

$$\mathbf{X} = \left[\mathbf{x}^1, \ldots, \mathbf{x}^n\right].$$

During the training phase, the bottom hole pressures (*bhp*) of the production wells P is varied randomly, and it takes values between P_1 and P_2, which implies that the right-hand side $\mathbf{b}^t$ is constantly changing during the simulation.

The solutions of the pre-simulation are obtained with the ICCG method.

ii) **POD basis computation**.The previously obtained snapshots are used to construct a POD basis (Ψ). The algorithm of the construction of the basis is presented below, Algorithm 2.

iii) **Solution of the linear system**. The POD basis is used as subspace-deflation matrix ($\mathbf{Z}$) in a deflation procedure for the solution of the linear system. The algorithms of the complete procedure are presented in Algorithm 3 for a moving window approach, and Algorithm 4 for a training phase approach.

To analyze the method's performance, we compare the total number of iterations and operations necessary to run the DICCG simulation with the

total number of iterations and operations necessary to solve the same problem using ICCG.

Algorithm 2 Computing the POD basis from a set of snapshots.

Given a set of s snapshots,
$\mathbf{X}^{1:s} = \left\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^s\right\}$*,
compute the average value of the set,
$\overline{\mathbf{x}} = \frac{1}{s}\sum_{j=1}^{s}\mathbf{x}^j$
substract the average from the original set,
$\overline{\mathbf{x}}^j = \mathbf{x}^j - \overline{\mathbf{x}}$
construct the covariance matrix,
$\overline{\mathbf{R}} := \frac{1}{s}\overline{\mathbf{X}}\overline{\mathbf{X}}^T, \qquad \overline{\mathbf{X}}^{1:s} = \left\{\overline{\mathbf{x}}^1, \overline{\mathbf{x}}^2, \ldots, \overline{\mathbf{x}}^s\right\}$
obtain the SVD of the transposed covariance matrix,
$\mathbf{V}\Lambda\mathbf{V}^T = \overline{\mathbf{R}}^T$
compute the eigenvectors of the covariance matrix,
$\mathbf{U} = \mathbf{X}\mathbf{V}\left(\Lambda^T\right)^{\frac{1}{2}}$
construct the basis (Ψ) with the eigenvectors ($\mathbf{u}^i \in \mathbf{U}$) corresponding to the p largest eigenvalues,
$\Psi^{1:p} = \{\psi^1, \ldots, \psi^p\} = \left\{\mathbf{u}^1, \ldots, \mathbf{u}^p\right\}$.
*where, $\mathbf{X}^{a:b} := \{\mathbf{x}^a, \mathbf{x}^{a+1}, \ldots, \mathbf{x}^b\}$.

Algorithm 3 Deflation, moving window variant, solving $\mathbf{A}^t\mathbf{x}^t = \mathbf{b}^t$.

Compute the solution of the initial window consisting on the snapshots obtained during the first s time steps with the ICCG method.
for $t = 1, \quad \ldots, \quad s$
Solve $\mathbf{A}^t\mathbf{x}^t = \mathbf{b}^t$ with ICCG
end for
$\mathbf{X}^{1:s} = \left\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^s\right\}$ % Window containing s snapshots
Compute the POD basis from the recently computed snapshots with Algorithm 2.
$\mathbf{Z} = \Psi^{1:p}\left(\mathbf{X}^{1:s}\right) = \left[\mathbf{v}^1, \ldots, \mathbf{v}^p\right]$

Compute the solution of the remaining time steps with the DICCG method.
for $t = s+1, \quad \ldots, \quad$ steps
Solve $\mathbf{A}^t\mathbf{x}^t = \mathbf{b}^t$ with DICCG
For the t-th time step, the window consists on the previous s snapshots, i.e., the solutions to the system from the $(t-s-1)$-th time step until the $(t-1)$-th time step.
$\mathbf{X}^{(t-s-1):(t-1)} = \left\{\mathbf{x}^{(t-s)}, \mathbf{x}^2, \ldots, \mathbf{x}^{(t-1)}\right\}$ % Window containing s snapshots
Update the POD basis from the recently computed window, see Algorithm 2.
$\mathbf{Z} = \Psi^{1:p}(\mathbf{X}^{(t-s-1):(t-1)}) = \left[\mathbf{v}^1, \ldots, \mathbf{v}^p\right]$
end for

Algorithm 4 Deflation, training phase variant, $\mathbf{A}^t\mathbf{x}^t = \mathbf{b}^t$.

Run a training phase simulation with ICCG changing randomly the producer's pressure, i.e., varying the value of the b_i's of the cells containing wells.
for $t = 1, \quad \ldots, \quad steps$
Solve $\mathbf{A}^t\mathbf{x}^t = \mathbf{b}^t$ with ICCG
end for
$\mathbf{X}^{1:steps} = \left\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^{steps}\right\}$
Compute the POD basis from the snapshots, see Algorithm 2,
$\mathbf{Z} = \Psi^{1:p}\left(\mathbf{X}^{1:steps}\right) = \left\{\mathbf{v}^1, \ldots, \mathbf{v}^p\right\}$

Run various simulations with fixed pressure* in the wells using DICCG, with $\mathbf{Z}$ as deflation matrix.
for $t = 1, \quad \ldots, \quad$ steps
Solve $\mathbf{A}^t\mathbf{x}^t = \mathbf{b}^t$ with DICCG
end for
*A pressure value is set for all the wells, and the simulation is run without changing the pressure values.

As tolerance or stopping criterion we use the relative residual, defined as the 2-norm of the residual of the k^{th} iteration divided by the 2-norm of the right-hand side of the preconditioned system,

$$\mathbf{rr}_k = \frac{||\mathbf{M}^{-1}(\mathbf{b} - \mathbf{A}\mathbf{x}_k)||_2}{||\mathbf{M}^{-1}\mathbf{b}||_2} = \frac{||\mathbf{M}^{-1}\mathbf{r}_k||_2}{||\mathbf{M}^{-1}\mathbf{b}||_2} \leq \epsilon, \tag{17}$$

throughout this work, we refer to this term as $\mathbf{rr}_k$.

The tolerance of the solvers is presented for each case. For some experiments, the true relative error is computed as:

$$e_k = \frac{||\mathbf{x} - \mathbf{x}_k||_2}{||\mathbf{x}||_2}, \tag{18}$$

where $\mathbf{x}$ is the 'true' solution, that in this case is the solution computed with backslash from Matlab, and $\mathbf{x}_k$ is the approximation computed with ICCG or DICCG.

Complexity Taking **p** as the number of deflation vectors, **m** as the sparcity of the matrix, i.e., the number of nonzero diagonals, and **s** as the number of snapshots, the approximate number of operations to solve a system with n unknowns is given in Table 1.

Besides the initial work the DICCG if a POD basis is used, the cost of the computation of the basis Ψ is $\left(4p^2 + s - 1\right)n + \left(8s^2 - 2s + 1\right)s$. More details about the

Table 1 Complexity of the methods

Method	Initialization	Iteration
ICCG	$(m^2 + 4m + 2)n$	$(4m + 9)n$,
DICCG	$((m + 2p + 4)m + (4p + 2)p + 2)n + p^3/3$	$(4m + 4p + 9)n$

computation of operations can be found in [14, 15, 17, 23] and Appendix D.

3 Numerical experiments: results and discussion

In this work, we focus on the solution of systems of linear equations resulting from water flooding for immiscible fluids, oil, and water, flowing through highly heterogeneous 2D and 3D reservoirs. To decouple pressure from saturation, we use the fractional flow formulation (see Appendix A). We solve the system with sequential schemes using the Matlab Reservoir Simulation Toolbox (MRST) [30].

The solution of the resulting transport equation is obtained with MRST using implicit schemes, and the linear pressure system is solved with the proposed POD-based DICCG methodology and compared to the ICCG method. We study various stopping criteria for the solution methods: $\epsilon = 5 \cdot 10^{-5}$, $\epsilon = 5 \cdot 10^{-7}$, and $\epsilon = 5 \cdot 10^{-9}$.

The models studied in this work are: an 'academic' layered reservoir with a contrast in permeability coefficients up to 10^6, and the SPE 10 benchmark with a contrast in permeability coefficients of $\mathcal{O}(10^7)$ [36]. We include capillary pressure and gravity terms in some of the studied cases.

3.1 Heterogeneous permeability layers

In this section, we study water injection into an 'academic' system consisting of equally-sized layers with a constant porosity field of 0.2 and different permeability values (see Fig. 1). A set of layers with permeability $\sigma_1 = 1\ mD$ is followed by layers with permeability $\sigma_2 = 10^6$ or $10^1\ mD$.

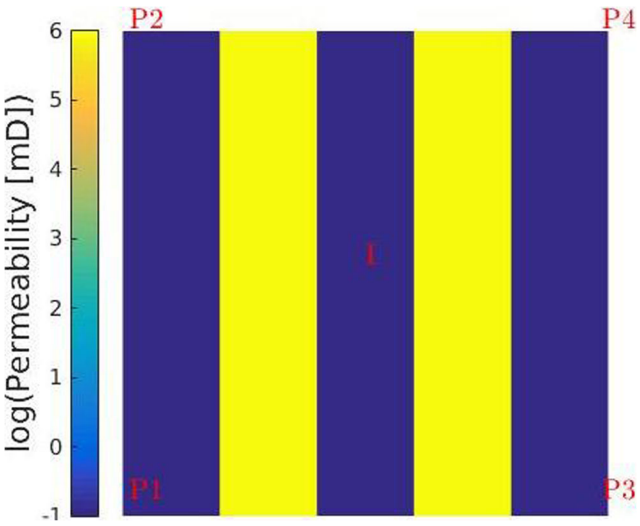

Fig. 1 Rock permeability

Table 2 Fluids properties

	Water	Oil	Units
μ	1	10	cp
ρ	1000	700	kg/m^3
k_r	$(S_w)^2$	$(1 - S_w)^2$	
C_p	$10 * (1 - S)$		bars

The contrast between permeability coefficients is given by $c_\sigma = \sigma_1/\sigma_2$.

The domain consists of a Cartesian grid of 35 x 35 cells for a 2D case, and 25 x 25 x 25 cells for a 3D case of length 8 m. The fluid properties are presented in Table 2.

The first set of experiments does not consider gravity and capillary pressure terms. Later, we include capillary pressure using the relationship $p_c = C(1 - S)$, and the 3D problem includes gravity terms. The relative permeability model used is the Corey model, with exponents $n_w = n_{nw} = 2$ (see Table 2). The studied test cases are:

TC1. $c_\sigma = 10^{-1}$, no capillary pressure terms included.
TC2. $c_\sigma = 10^{-6}$, no capillary pressure terms included.
TC3. $c_\sigma = 10^{-1}$, capillary pressure terms included.
TC4. $c_\sigma = 10^{-6}$, capillary pressure terms included.

We study water flooding with injection through the boundary and wells. When using wells, the setup consists of four producers, P_i, on the corners and one injector, I, in the center. The wells are controlled by prescribing the bottom hole pressure *bhp* values.

3.1.1 Injection through the left boundary

The injection is performed through the left boundary at a rate of 0.4 m^3/day for the 2D case and 4 m^3/day for the 3D case. The pressure is set as zero at the right boundary and the initial pressure of the reservoir is 100 bars (See Table 3). The simulation is run for 480 days with time steps of 1 day (see Table 3). We study the deflation procedure in a moving window approach (MW), with two different selections of deflation vectors, p. For the first one, we use the ten previous time step solutions, $p = 10$, represented by $DICCG_{10}$. For the second

Table 3 Boundary conditions and temporal parameters

Temporal parameters			Boundary conditions		
T_{steps}	480		$P_{0,x\neq(0,L_x)}$	100	$bars$
dT	1	day	$P_{x=L_x}$	0	$bars$
T_{total}	480	days	$Q_{x=0}$	0.4 (2D), 4 (3D)	m^3/day

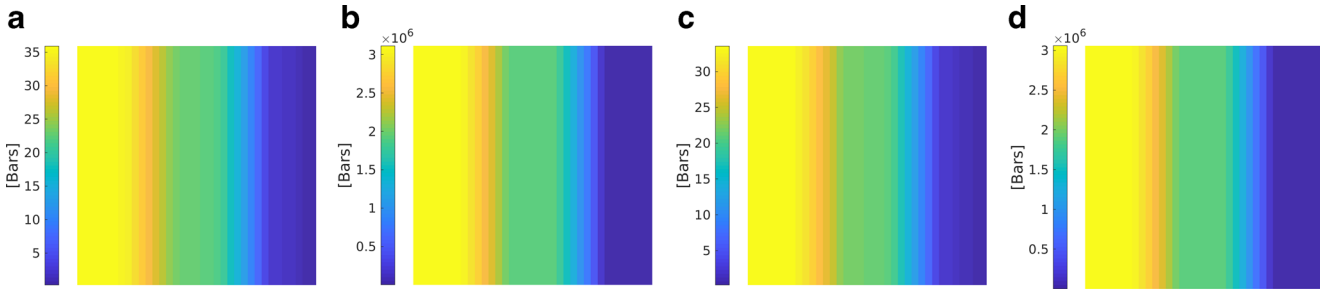

Fig. 2 Pressure field for the last time step, cases **a** TC1, **b** TC2, **c** TC3, **d** TC4

one, we choose the eigenvectors corresponding to the five largest eigenvalues of the correlation matrix of a POD methodology, $p = 5$. The latest approach is the proposal of this work, called POD-based method, and it is represented by $DICCG_{POD_5}$.

The pressure field and oil saturation are presented in Figs. 2 and 3. We observe that the pressure is higher on the boundary where water is injected, and it decreases towards the right boundary. We also observe that the layered pattern influences the pressure and saturation fields.

In Fig. 4, we present the eigenvalues of a) the system matrix, b) preconditioned system matrix, deflated and preconditioned system with c) ten snapshots, and d) five POD basis vectors as deflation vectors. We observe that for the cases with a larger contrast in permeability coefficients (TC2 and TC4) the spectrum of the system is more spread, which implies that the system is more ill-conditioned. We note that after preconditioning, the spectrum is clustered, but there are few eigenvalues smaller than the rest.

For the ill-conditioned systems (TC2 and TC4) these few eigenvalues are smaller than in the other cases (TC1 and TC3), slowing down the convergence of the method. After deflation, these small eigenvalues are set to a value close to zero (10^{-16} in machine precision). We can note that using (c) ten snapshots or (d) five POD-based deflation vectors, the smallest eigenvalues of the preconditioned system are set to zero, leading to a similar spectrum (without the removed eigenvalues). This implies a similar performance with both selection of deflation vectors.

The number of iterations necessary to achieve convergence is summarized in Table 4, where the first column shows the test case. In the second column, we present the number of deflation vectors used, p. The third one shows the number of iterations necessary to achieve convergence with the ICCG method only. Note that this number is the same for each test case, i.e., only the number of DICCG iterations and work change if a different number of deflation vector is used.

The number of iterations necessary for convergence of the deflation method is presented in the sixth column. For this example, we use the Moving Window (MW) approach. Therefore, the first p time steps are computed with the ICCG method (fourth column), and the rest of the time steps are computed with the DICCG method (fifth column). Then, we compute the percentage of DICCG iterations with respect to the total number of ICCG iterations. In the last column, we present the percentage of ICCG work required to perform the simulation. The total work of each method is computed as the number of iterations times the number of operations per iteration, plus the initial work and the extra work required to compute the POD basis.

In Table 4, we observe a significant reduction in the number of iterations and the work required when using the DICCG method. For the 2D problem, test cases TC1, TC2, the number of iterations is reduced to 14% and 12% the number of ICCG iterations when using ten snapshots as deflation vectors, and to 19% and 16% when using five POD-based vectors as deflation vectors. For the same test cases, the work required to perform the simulation is reduced to around 33% and 28% for ten deflation vectors and to 61.7% and 47.6% using five, respectively. Thus, the TC2 case presenting a contrast between permeability layers

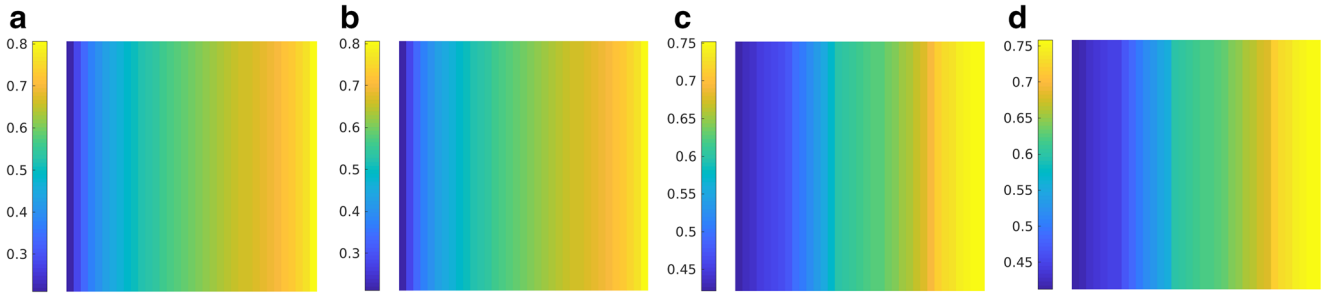

Fig. 3 Oil saturation during the last time step, cases **a** TC1, **b** TC2, **c** TC3, **d** TC4

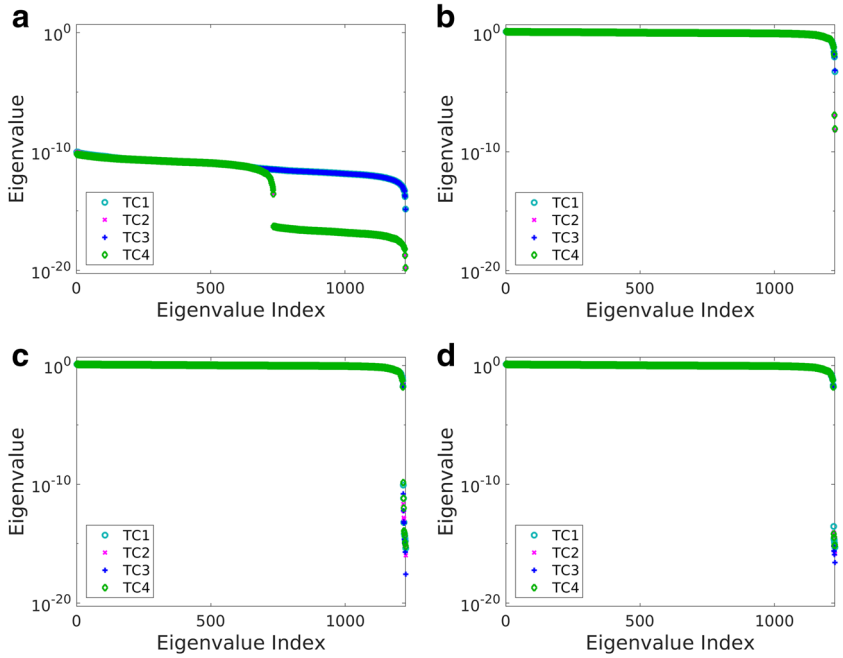

Fig. 4 Eigenvalues of **a** the system matrix, **b** preconditioned system matrix, deflated and preconditioned system with **c** ten snapshots, and **d** five POD basis vectors as deflation vectors

of 10^{-6} and no capillary pressure terms showed a slightly better performance.

If capillary terms are included (TC3 and TC4), the percentage of ICCG iterations and the total work increase. For these cases, the reduction in the number of iterations is similar for DICCG$_{10}$ and DICCG$_{POD_5}$ methods, which implies that almost the same information that is contained in the ten snapshots is captured in the five POD basis vectors.

Table 4 Number of iterations for various methods, $\epsilon = 5 \cdot 10^{-7}$

	p	Total	DICCG		Total	% of ICCG	% of ICCG
		ICCG	ICCG	DICCG	DICCG	Iterations	Work
2D Case							
TC1	10	23245	406	2830	3236	14	33.2
	5_{POD}			3929	4335	19	61.7
TC2	10	33834	194	3789	3983	12	28
	5_{POD}			5256	5450	16	47.6
TC3	10	23123	427	5792	6219	27	64
	5_{POD}			6296	6723	29	78.9
TC4	10	35507	220	6959	7179	20	48.1
	5_{POD}			7291	7511	21	55.2
3D Case							
TC1	10	24231	498	3887	4385	18	37.7
	5_{POD}			4929	5427	22	56.5
TC2	10	20260	309	3342	3651	18	37.6
	5_{POD}			3117	3426	17	52.4
TC3	10	24557	513	8103	8616	35	73.1
	5_{POD}			8472	8985	37	78.1
TC4	10	19130	267	4180	4447	23	48.5
	5_{POD}			5084	5351	28	70.9

The cases TC1 and TC3 do not contain capillary pressure terms, while the TC2 and TC4 cases contain these terms. The sub-index $_{POD}$ in the p column means that the deflation vectors are POD basis vectors, if no sub-index is included, it means that the vectors are snapshots

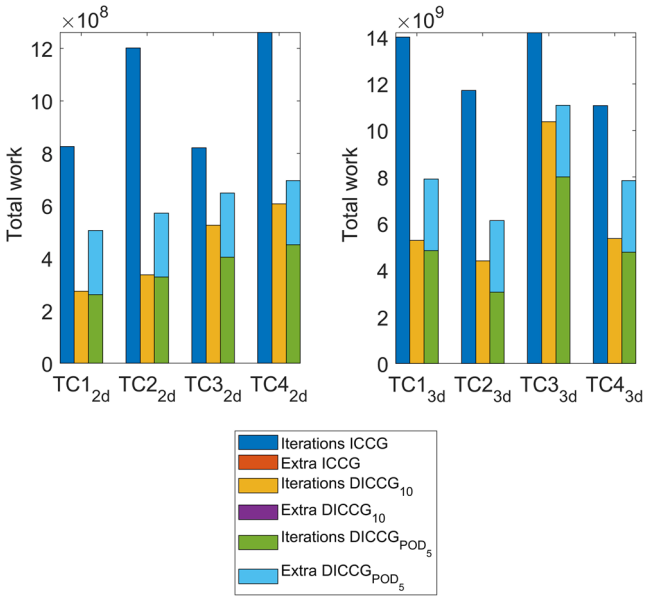

Fig. 5 Total work for various methods, $\epsilon = 5 \cdot 10^{-7}$, Left: 2D cases, Right: 3D cases. The initial work is negligible in all cases

However, computing the POD basis requires more work. Using ten snapshots, the number of iterations is reduced to 27 and 20% the number of ICCG iterations, and to 64 and 48% of the ICCG work. If five POD vectors are used, the reduction is 29 and 21% the number of ICCG iterations, and to 79 and 55% of the ICCG work.

We observe similar trends for the 2D and 3D cases, the last one, including gravity terms. However, the 3D cases are more expensive in terms of the number of iterations and the total work than the 2D cases.

In Fig. 5, we show the total number of operations required to perform the work during the iteration process and the extra work. In the case of the DICCG$_{POD_5}$ method with five POD-based deflation vectors, the extra work is due to the computation of the deflation vectors. The ICCG and the DICCG$_{10}$ methods do not have extra work involved as the ICCG method does not compute any deflation vectors, and the DICCG$_{10}$ method reuses the previously computed solutions. The initial work is negligible for all cases. Thus, it is not included in the plots.

Although the work required to perform the iterative process for the DICCG$_{POD_5}$ method is less than the work required by the DICCG$_{10}$ method (see Fig. 5), the computation of the POD basis increases the number of operations considerably for the DICCG$_{POD_5}$ method, especially for the 3D case.

Figure 6 shows the relative residual (17), and Fig. 7 presents the true relative error (18) for the 240-th time step for the studied methods, similar performance is observed for the rest of the time steps. We note that the superlinear convergence phase is reached almost immediately for the deflation methods, contrary to the ICCG method that requires some iterations before reaching this phase.

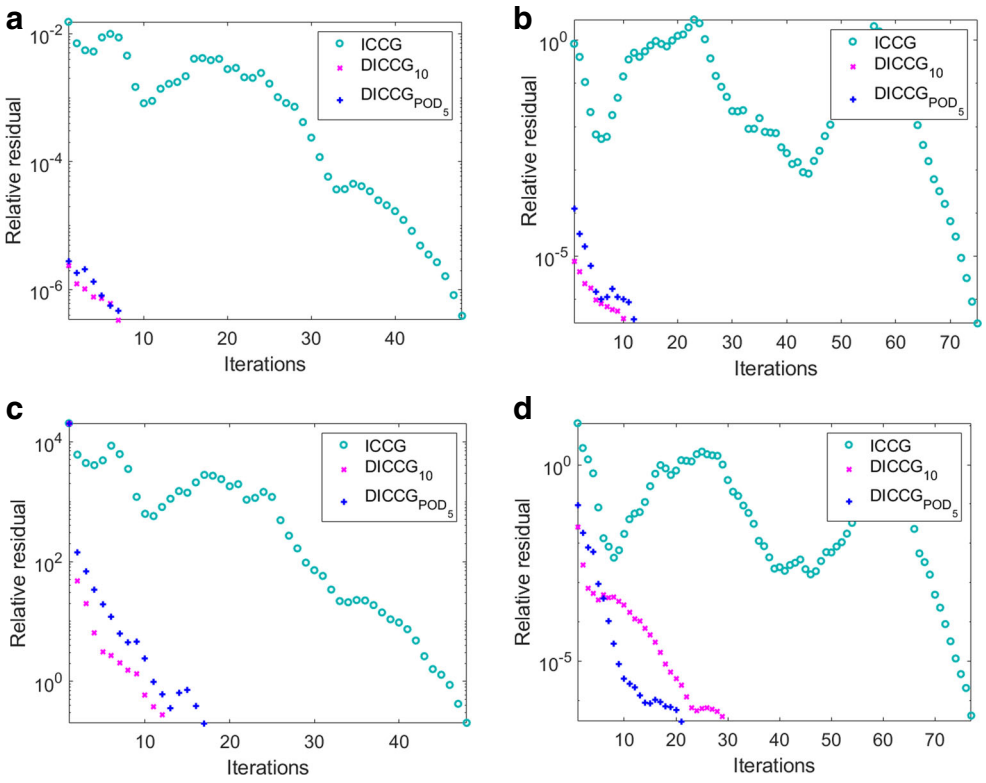

Fig. 6 Relative residual, time step 240, **a** TC1, **b** TC2, **c** TC3, **d** TC4

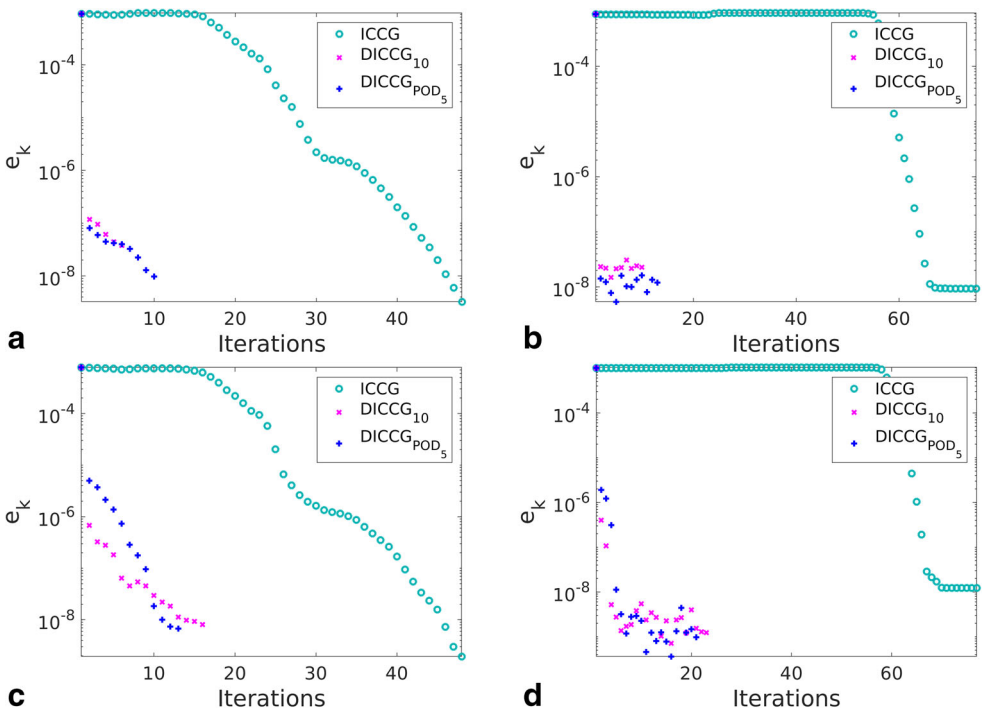

Fig. 7 True relative error, time step 240, **a** TC1, **b** TC2, **c** TC3, **d** TC4

Table 5 Number of iterations for various methods, $\epsilon = 5 \cdot 10^{-5}$

	p	Total	DICCG		Total	% of ICCG	% of ICCG
		ICCG	ICCG	DICCG	DICCG	Iterations	Work
2D Case							
TC1	10	16671	118	1310	1428	9	20.5
	5_{POD}			1629	1747	10	59
TC2	10	30803	116	2140	2256	7	17.5
	5_{POD}			2874	2990	10	38.8
TC3	10	16935	178	2153	2331	14	32.9
	5_{POD}			2600	2778	16	68.4
TC4	10	32359	127	4031	4158	13	30.6
	5_{POD}			4046	4173	13	43.1
3D Case							
TC1	10	15751	221	1351	1572	10	20.9
	5_{POD}			1472	1693	11	50.4
TC2	10	5141	63	747	810	16	33.1
	5_{POD}			613	676	13	123.9
TC3	10	16311	239	1659	1898	12	24.3
	5_{POD}			1907	2146	13	52.4
TC4	10	3998	68	734	802	20	42.2
	5_{POD}			790	858	21	166.3

The cases TC1 and TC3 do not contain capillary pressure terms, while the TC2 and TC4 cases contain these terms. The sub-index $_{POD}$ in the p column means that the deflation vectors are POD basis vectors, if no sub-index is included, it means that the vectors are snapshots

We also note that the true relative error is smaller than the relative residual in most of the cases. Furthermore, after a few DICCG iterations, an accuracy of $\mathcal{O}(10^{-8})$ is reached. If the contrast between permeability coefficients is larger (TC2 and TC4), i.e., more ill-conditioned problems, the true error is $\mathcal{O}(10^{-8})$, after one, and three iterations, respectively.

As mentioned before, the first iterations have an accuracy of $\mathcal{O}(10^{-4})$, thus, we study a case with a less strict tolerance, $\epsilon = 5 \cdot 10^{-5}$. Table 5 presents the number of iterations for the ICCG and DICCG methods together with the percentage of the ICCG iterations and work required when using the DICCG method for a tolerance of $\epsilon = 5 \cdot 10^{-5}$.

We observe that the reduction in the number of iterations when using the deflated methods ranges from 7-21% of the ICCG iterations, which is larger than when a more strict stopping criterion ($\epsilon = 5 \cdot 10^{-7}$) is required. Similarly, the reduction of the ICCG work is smaller, ranging from 17.5-68% for the 2D case.

The work required to perform the iteration process and the extra work for this problem is presented in Fig. 8 for all methods, the initial work is negligible in all cases, thus, it is not presented.

We observe that the largest reduction in work is achieved for the DICCG$_{10}$ method in all the cases. A reduction of the ICCG work is achieved when using the DICCG$_{POD_5}$ for all the cases in the 2D problem, but only for TC1 and TC3 for the 3D problem. The ICCG work required for the TC2 and TC4 is small and therefore, the cost of computing the POD basis increases the total work of the DICCG$_{POD_5}$ method.

We observe that for the DICCG$_{10}$ method, the maximum number of iterations is reached (150). This implies that the method does not converge. The set of ten deflation vectors used are the ten previous solutions, and linear independence is not guaranteed for this set. This implies that computing the Galerkin matrix $\mathbf{E}$ is not possible, and convergence of the deflation method is not guaranteed.

We also note that for the TC2, 3D case, the reduction in the number of iterations is larger if we use five POD deflation vectors. To understand this behavior, we plot the number of iterations for all time steps in Fig. 9.

On the other hand, the POD basis is linearly independent by construction. Therefore, using vectors of this basis as deflation vectors (DICCG$_{POD_5}$) does not present the same problems as the recycled snapshots (DICCG$_{10}$). As a consequence, the linear independence of the POD results in a more robust deflation method.

Note that, for the methods used in this work we need an initial guess. When solving the first time step, this initial guess is not necessarily close to the solution, hence, it needs a larger number of iterations to converge. For the following time steps, the initial guess is the solution of the previous time step, and as the system does not change dramatically from one step to the next one, the solution is reached faster than the initial time step.

Discussion We observe a reduction in the work and the number of iterations required to simulate waterflooding in a layered reservoir. If the contrast between permeability layers increases, the reduction is more significant. This latter observation shows the better performance of the POD-based deflation method for ill-conditioned systems.

We also observe that using ten snapshots or five POD basis vectors as deflation vectors gives a similar reduction in the number of iterations. However, the reduction in the work is larger if ten snapshots are used as deflation vectors due to the cost of computing the POD basis.

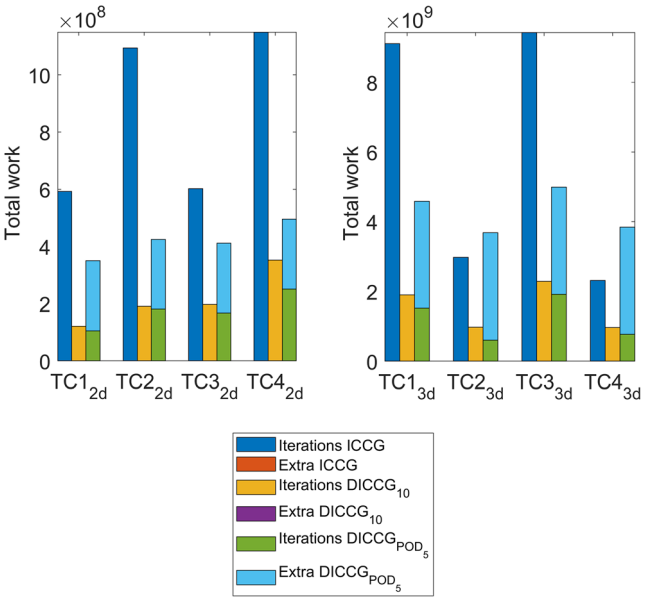

Fig. 8 Total work for various methods, $\epsilon = 5 \cdot 10^{-5}$, Left: 2D cases, Right: 3D cases. The initial work is negligible in all cases

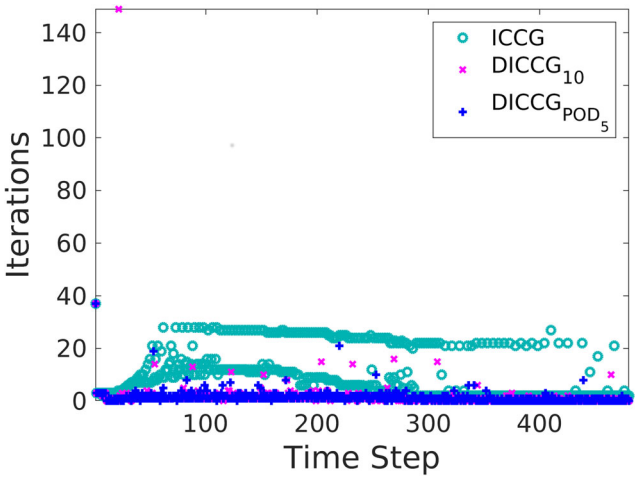

Fig. 9 Number of iterations TC2, $\epsilon = 5 \cdot 10^{-5}$

Nevertheless, using the POD basis (DICCG$_{PODs}$) avoids non-convergence problems that the DICCG method might present when using only snapshots (DICCG$_{10}$).

Including capillary pressure terms slightly deteriorates the performance of the methods. This behavior might be due to the fact that including capillarity pressure terms reflects on changes on the right-hand side, which is not effectively treated by the deflation method. However, more studies are required to confirm this hypothesis.

We note that after a few iterations, the residual is of order 10^{-4}. Thus, using a less strict stopping criterion (10^{-5} instead of 10^{-7}) results in less work of the DICCG method compared to the ICCG method. However, this is only the case for the DICCG$_{10}$ method. As a less strict criterion is used, the ICCG method also requires fewer iterations, and computing the basis for the DICCG$_{PODs}$ method leads to a more costly method.

3.2 SPE 10 benchmark

The SPE 10 model consists of 60 x 220 x 85 cells. In this work, we study a 2D case containing the top layer and a 3D case with the first 36 layers, including gravity terms. For the problem with one layer, the system matrix has a condition number of $\mathcal{O}(10^7)$, while for the 36 layers, the condition number is of $\mathcal{O}(10^8)$. As the convergence of the methods depends on the condition number, to achieve similar results, we impose a more strict stopping criterion for the 3D case. We study cases with and without capillary pressure terms:

TC1. No capillary pressure terms included.
TC2. Capillary pressure terms included.

We consider injection through the boundary and injection through wells. The wells' setup consist of one injector and four producers (see Fig. 10).

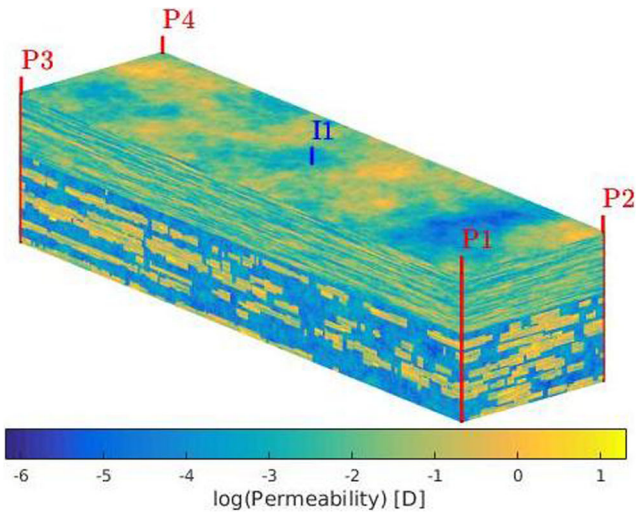

Fig. 10 Permeability field SPE 10

We compare the performance of the POD-based deflation method against the standard ICCG method for the solution of the pressure equation.

Application of the DICCG method requires the construction of a deflation matrix based on a POD basis obtained from a series of snapshots selected in two ways: a *moving window*, and a *training phase* approach (see Section 2).

For the training phase approach, the POD basis and deflation matrix are obtained off-line in a training run solved with the ICCG method. Then, a series of simulations are performed using the DICCG method with diverse values of bottom hole pressure, *bhp*, in the producers. The fluid properties and the capillary function used for this problems are the same as in Section 3.1.1 (see Table 2).

3.2.1 Injection through the left boundary, moving window approach

Water is injected through the left boundary at a constant rate of 600 m^3/day to a reservoir initially filled with oil. The initial reservoir's pressure is 500 [bars], and the pressure at the right boundary is set to zero [bars]. We simulate 480 time steps with a step size of 15 days for the 2D case. For the 3D case, we use 36 layers, and we simulate 240 time steps with a step size of 1 day and an injection rate of 1000 m^3/day (see Table 6).

For the DICCG method we use the moving window approach (MW), selecting ten snapshots (from previously computed time steps) and five POD basis vectors as deflation vectors for the 2D case. The pressure field, the water saturation and the eigenvalues of the correlation matrix are presented in Fig. 11 for the 240-th time step.

Regarding the eigenvalues, we observe that when we include capillary pressure terms (TC2) they are larger, which indicates that they still contain important system information, and therefore, more eigenvectors are required to obtain a better performance of the deflation method.

The number of iterations required to achieve an accuracy of $\epsilon = 5 \cdot 10^{-7}$ for the 2D case, and $\epsilon = 5 \cdot 10^{-9}$ for the 3D is presented in Table 7 for the ICCG and DICCG methods. We observe a reduction to 11% of the number of ICCG iterations and to 24.5% the total ICCG work when using ten snapshots as deflation vectors DICCG$_{10}$. If five

Table 6 Boundary conditions and temporal parameters

Temporal parameters				Boundary conditions			
	2D	3D			2D	3D	
T_{steps}	480	240		$P_{0,x\neq(0,L_x)}$	500	500	$bars$
dT	15	1	day	$P_{x=L_x}$	0	0	$bars$
				$Q_{x=0}$	600	1000	m^3/day

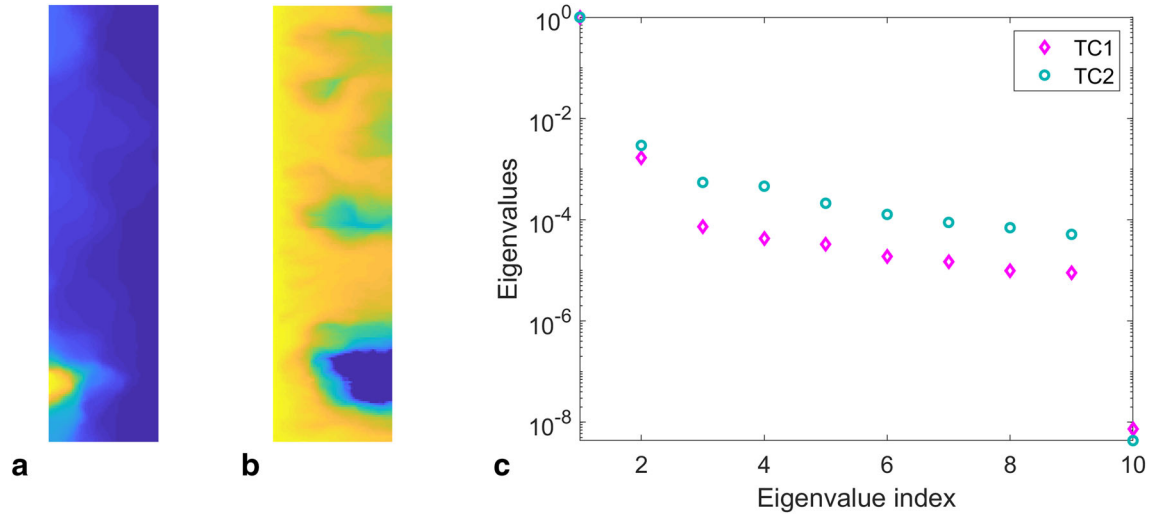

Fig. 11 **a** Pressure field, **b** Water saturation, and **c** Normalized eigenvalues of the correlation matrix, 240-th time step

POD-based deflation vectors are used DICCG$_{POD_5}$, the number of ICCG iterations is reduced to 12%, and the total ICCG work to 30.4%.

If we include capillary terms, we observe an increment in the number of iterations, and, consequently, in the total work. The percentage of ICCG iterations increases to ∼19% and the ICCG work to ∼43% for both cases. As we mentioned before, more eigenvectors are required for the TC2 to gather the same information as in TC1, this is reflected in a better DICCG performance for the TC1.

For the 3D problems, we note that we reduce the number of iterations and the total work significantly. For the TC1, we achieve a reduction of ∼27% of the ICCG work when using 10 snapshots as deflation vectors (DICCG$_{10}$), and ∼35% when using 5 POD vectors (DICCG$_{POD_5}$). For the TC2, the reductions in work are ∼72% for DICCG$_{10}$, and ∼66% for DICCG$_{POD_5}$.

In Fig. 12, we show the work required to perform each part of the simulation for the 2D case. The initial work for all methods is negligible compared with the rest of the work, thus, it is not included. The only method that requires a computation of extra work is the DICCG$_{POD_5}$ method with five POD-based deflation vectors. For this method, the extra work is the computation of the basis (see Appendix D).

We note that the work performed during the iterative process is smaller when using five deflation vectors (DICCG$_{POD_5}$). However, taking into account all the operations involved, using five POD vectors or ten solutions as deflation vectors

Table 7 Number of iterations for various methods, 2D case $\epsilon = 5 \cdot 10^{-7}$, 3D case $\epsilon = 5 \cdot 10^{-9}$

	p	Total	DICCG		Total	% of ICCG	% of ICCG
		ICCG	ICCG	DICCG	DICCG	Iterations	Work
2D Case							
TC1	10	69617	2177	5249	7426	11	25.4
	5_{POD}			6341	8518	12	30.4
TC2	10	70386	2217	10345	12562	18	42.5
	5_{POD}			11741	13958	20	43.2
3D Case							
TC1	10	63185	3456	7592	11048	17	26.9
	5_{POD}			9081	12537	20	34.8
TC2	10	55840	3064	16350	19414	35	72.4
	5_{POD}			17286	20350	36	65.7

The cases TC1, do not contain capillary pressure terms, while the TC2 cases contain these terms. The sub-index $_{POD}$ in the p column means that the deflation vectors are POD basis vectors, if no sub-index is included, it means that the deflation vectors are snapshots

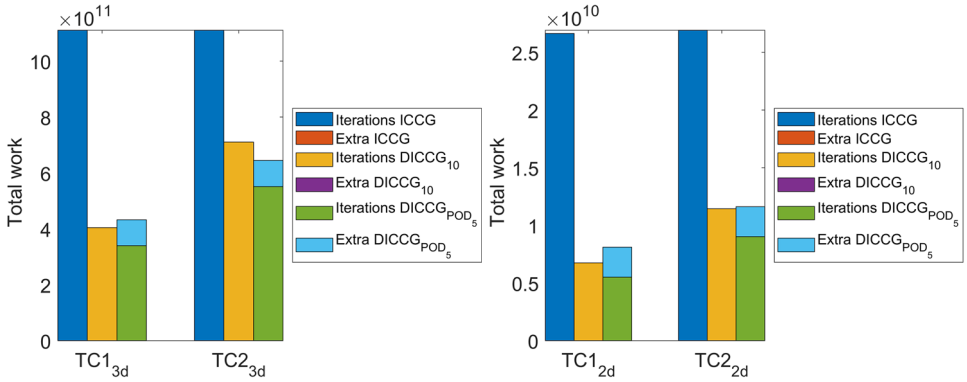

Fig. 12 Total work for various methods, $\epsilon = 5 \cdot 10^{-7}$, SPE layers, Left: 2D cases, Right: 3D cases. The initial work is negligible in all cases

requires a similar number of operations for the TC2, and slightly more for the DICCG$_{POD_5}$ method for the TC1.

The relative residual of the studied methods is shown in Fig. 13 for the 240-th time step for the 2D case, similar performance is observed for the rest of the time steps. We note that the first iteration gives a significant reduction for all methods. However, for the ICCG method, the reduction rate is small, while for the DICCG method, in both cases, this rate is larger and the desired accuracy is achieved faster. This behaviour implies that the superlinear convergence phase is achieved for the DICCG method after the first iteration.

The true error is presented in Fig. 14. Here, we observe that after the first iteration, a good approximation is achieved for the DICCG method, $\mathcal{O}(10^{-5})$, but the ICCG method takes longer to achieve the same accuracy. Thus, the relative residual does not show the behaviour of the true error for the ICCG method.

These results show that the most important system information is contained in the deflation vectors and, therefore, convergence is achieved in a small number of iterations. If the desired accuracy is small, e.g., 10^{-5}, only one iteration is required. This results are similar to the results obtained for the layered problems.

In the next section we compare the approach used here (MW) with the training phase (TP) approach for a case where water is injected through a well located in the center of the reservoir.

3.2.2 Injection through wells, MW and TP approaches

In this section, we perform a series of experiments injecting water through a well located in the center of the reservoir with a prescribed rate of injection. The fluids are produced in the corner wells, where we prescribe the pressure (see Fig. 10), i.e., we give the bottom hole pressures (*bhp*) of the wells.

For the moving window approach (MW) examples, the pressure is kept constant for all wells during all the time steps. For the training phase (TP) approach, first, we perform a complete simulation (training run) with variable pressures in the wells. The solutions obtained from the training run are used to obtain the POD-based deflation vectors. These vectors are the deflation vectors required to implement the DICCG method. We solve a series of problems with constant wells pressures.

We solve the first layer (2D case) and the first 36 layers (3D case) of the SPE 10 benchmark. For the MW approach, we select the ten previous solutions and five largest POD basis vectors as deflation vectors. For the TP approach, we select ten and five of the largest POD basis vectors as deflation vectors.

Moving window approach (MW) For this case, the pressure inside the wells is set at 275 [bars] in the producers and 1100 [bars] in the injector. The test cases of this approach are referred to as TC1$_{MW}$ when it has no capillary pressure

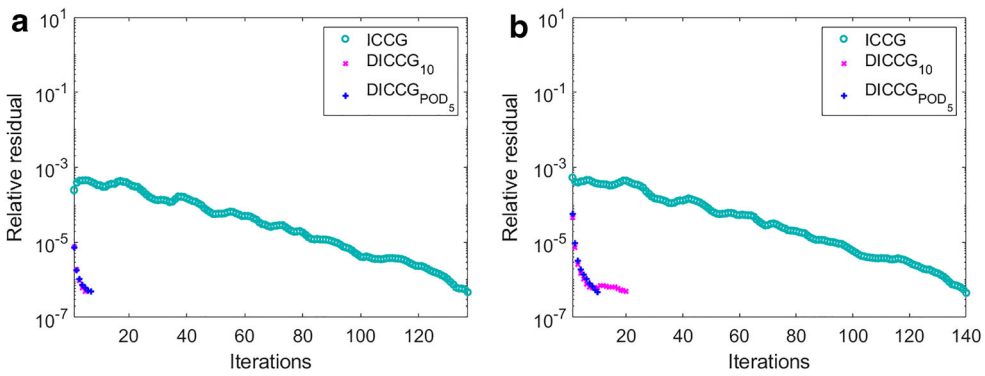

Fig. 13 Relative residual for various methods, **a** TC1, **b** TC2, injection through the left boundary

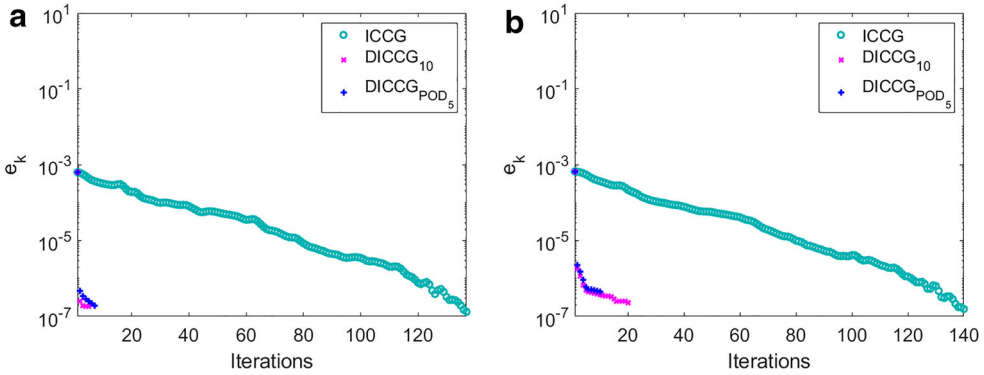

Fig. 14 True relative error for various methods, **a** TC1, **b** TC2, injection through the left boundary

terms included, and $TC2_{MW}$ when we include capillary pressure terms. The simulation is run for 480 time steps with a step size of 15 days.

Training phase approach (TP) For this set of experiments, we run a full simulation changing the pressure randomly in the wells. The solutions of this simulation are used to compute the POD basis, used later as deflation vectors. The *bhp* in the production wells is varied every two time steps during the training phase within a range of 137.5 and 275 [bars] (see Fig. 15). The pressure in the injection well is maintained constant at 1100 [bars].

Once the deflation matrix is obtained from the POD basis, we solve three cases. For the first two, we select a *bhp* in the producers inside the range used during the training run, one intermediate 200 [bars], (cases $TC3_{TP}$ and $TC4_{TP}$); and one extreme value 275 [bars] (cases $TC5_{TP}$ and $TC6_{TP}$). For the third case, we solve the problem setting the *bhp* in the wells outside the training phase range, 400 [bars] (cases $TC7_{TP}$ and $TC8_{TP}$). The simulation is run during 480 steps of 15 days. A summary of the cases studied in this section is presented below.

$TC1_{MW}$. MW approach, 2D, producers bhp: 275 [bars], no capillary pressure terms included.
$TC2_{MW}$. MW approach, 2D, producers bhp: 275 [bars], capillary pressure terms included.
$TC3_{TP}$. TP approach, 2D, producers bhp: 200 [bars], no capillary pressure terms included.
$TC4_{TP}$. TP approach, 2D, producers bhp: 200 [bars], capillary pressure terms included.
$TC5_{TP}$. TP approach, 2D, producers bhp: 275 [bars], no capillary pressure terms included.
$TC6_{TP}$. TP approach, 2D, producers bhp: 275 [bars], capillary pressure terms included.
$TC7_{TP}$. TP approach, 2D, producers bhp: 400 [bars], no capillary pressure terms included.

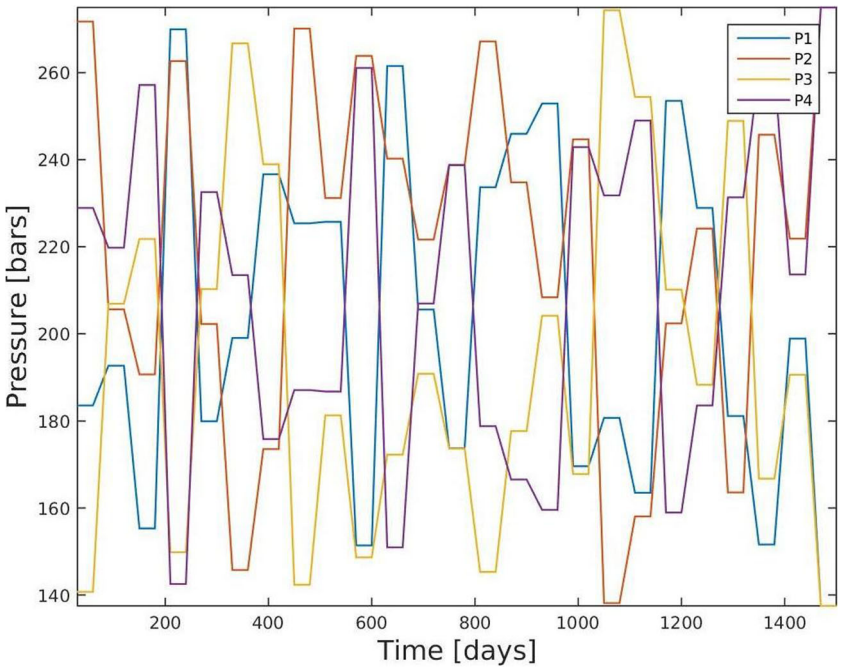

Fig. 15 Well pressures for the first time steps of the training phase

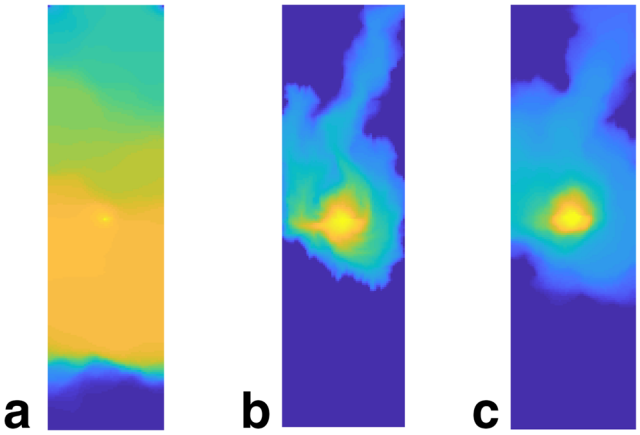

Fig. 16 **a** Pressure field, **b** Water saturation ($TC1_{MW}$), and **c** Water saturation ($TC2_{MW}$), last time step

$TC8_{TP}$. TP approach, 2D, producers bhp: 400 [bars], capillary pressure terms included.

The pressure field and the water saturation are presented in Fig. 16. The eigenvalues of the correlation matrix for the 240-th time step of the MW approach, and for the training phase run are presented in Fig. 17.

For the MW approach, we observe that the eigenvalues are similar for the two studied cases ($TC1_{MW}$ and $TC2_{MW}$). This indicates that their corresponding eigenvectors contain comparable information, and we expect a similar behaviour. For the TP approach, we note that the eigenvalues of the $TC4_{TP}$ (including capillary pressure terms) are smaller, hence, they contain less information. Therefore, we require more eigenvectors to get similar performance as in the $TC3_{TP}$ (no capillary terms included).

The number of iterations required to obtain an approximate solution with an accuracy of $\epsilon = 5 \cdot 10^{-7}$ for the ICCG and DICCG methods is presented in Table 8 for the MW and TP approaches for the 2D case. Note that (see Section 2.4), for the MW approach we use ten deflation vectors and five POD deflation vectors computed during the simulation. However, a single POD basis is obtained off-line, and this basis is used for all the test cases of the TP approach ($TC3_{TP}$-$TC8_{TP}$).

In Fig. 18 we present in a) the total work required for each part of the methods. Note that, there is extra work only for the MW approach with five POD vectors, for the TP run the deflation vectors are computed offline in a training run. In b), the percentage of total ICCG work is presented for the DICCG method with five and ten deflation vectors.

Table 8 Number of iterations for various methods, $\epsilon = 5 \cdot 10^{-7}$ for the 2D case

	p	Total	DICCG		Total	% of ICCG	% of ICCG
		ICCG	ICCG	DICCG	DICCG	Iterations	Work
$TC1_{MW}$	10	80764	2241	5691	7932	10	23.4
	5_{POD}			6667	8908	11	27.1
$TC2_{MW}$	10	78950	2256	5334	7590	10	22.9
	5_{POD}			6018	8274	10	26.3
$TC3_{TP}$	10_{POD}	80764			12044	15	35.5
	5_{POD}				20281	25	42.5
$TC4_{TP}$	10_{POD}	78950			11746	15	35.4
	5_{POD}				17659	22	37.8
$TC5_{TP}$	10_{POD}	83972			12935	15	37
	5_{POD}				21439	26	38.3
$TC6_{TP}$	10_{POD}	83279			13078	16	37
	5_{POD}				19045	23	38.3
$TC7_{TP}$	10_{POD}	72158			10279	14	33.9
	5_{POD}				17858	25	41.8
$TC8_{TP}$	10_{POD}	70819			9824	14	33
	5_{POD}				15353	22	36.7

The cases TC1, TC3, TC5 and TC7 do not contain capillary pressure terms, while the rest of the cases contain these terms. The sub-index $_{POD}$ in the p column means that the deflation vectors are POD basis vectors, if no sub-index is included, it means that the deflation vectors are snapshots

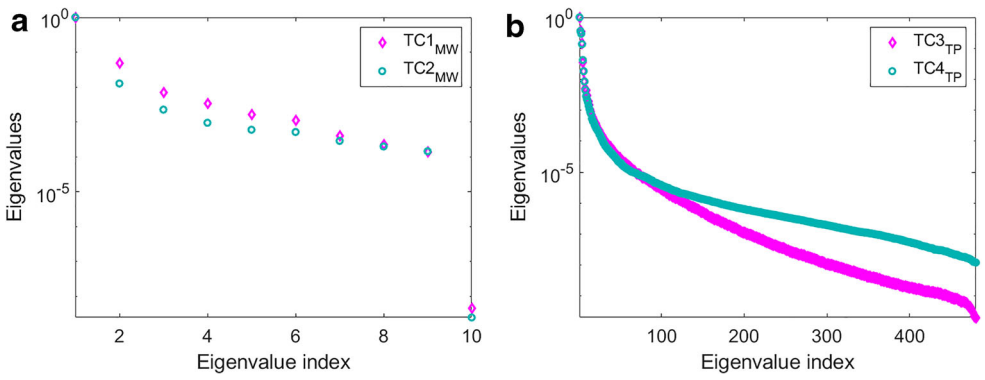

Fig. 17 Normalized eigenvalues of the correlation matrix, 240-th time step **a** MW: TC1 and TC2, **b** TP: TC3 and TC4

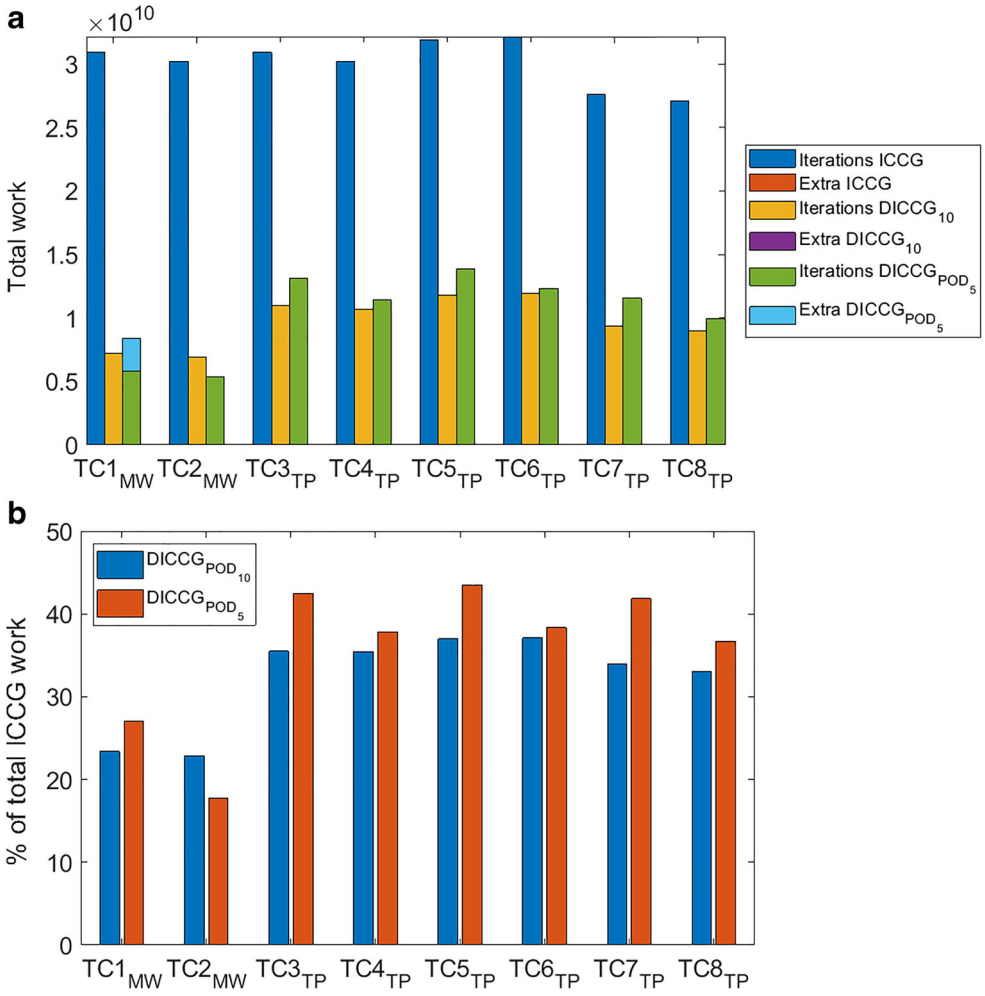

Fig. 18 **a** Total work required for each part of the methods, **b** Percentage of total ICCG work

We note that the MW approach gives a larger reduction in the number of ICCG iterations, around 10% for the 2D cases ($TC1_{MW}$ and $TC2_{MW}$). As a consequence, the reduction of the total work is also larger for this case. We also note that, using five POD-based deflation vectors for the MW case requires more work than using ten, due to the extra work required to compute the basis (see Fig. 18).

For the TP approach, we note that the results are similar for the three cases. This result implies that the basis computed from the training phase where the *bhp* of the production wells is varied in a range can be used to solve problems with different bhp of the production wells inside the range $TC3_{TP}$ and $TC4_{TP}$ (275 [bars]), in the limits $TC5_{TP}$ and $TC6_{TP}$ (200 [bars]), and outside the range $TC7_{TP}$ and $TC8_{TP}$ (400 [bars]). Furthermore, we note that the reduction of the work is slightly larger when using ten deflation vectors.

The relative residuals and true relative errors of the studied methods are presented in Figs. 19 and 20. As in the previous cases, we observe a good approximation $\mathcal{O}(10^{-4})$ after the first DICCG iteration for the relative residual, similar to the true relative error in all the studied cases. Regarding the ICCG method, even if the relative residual appears to be $\mathcal{O}(10^{-6})$ the error is 10^{-4} for the MW approach ($TC1_{MW}$ and $TC2_{MW}$) and 10^{-6} for the TP approach ($TC3_{TP}$ and $TC4_{TP}$). Thus, they do not reach the desired accuracy (10^{-7}), however, the error of the TP approach it is closer to the desired accuracy than the MW approach.

We also note a faster convergence rate for DICCG methods compared to the ICCG method during the first iterations, i.e., the superlinear convergence phase is achieved during the first iterations for the DICCG method.

The 3D cases are presented below:

$TC9_{MW}$. MW approach, 3D, producers bhp: 275 [bars], no capillary pressure terms included.
$TC10_{MW}$. MW approach, 3D, producers bhp: 275 [bars], capillary pressure terms included.
$TC11_{TP}$. TP approach, 3D, producers bhp: 275 [bars], no capillary pressure terms included.
$TC12_{TP}$. TP approach, 3D, producers bhp: 275 [bars], capillary pressure terms included.

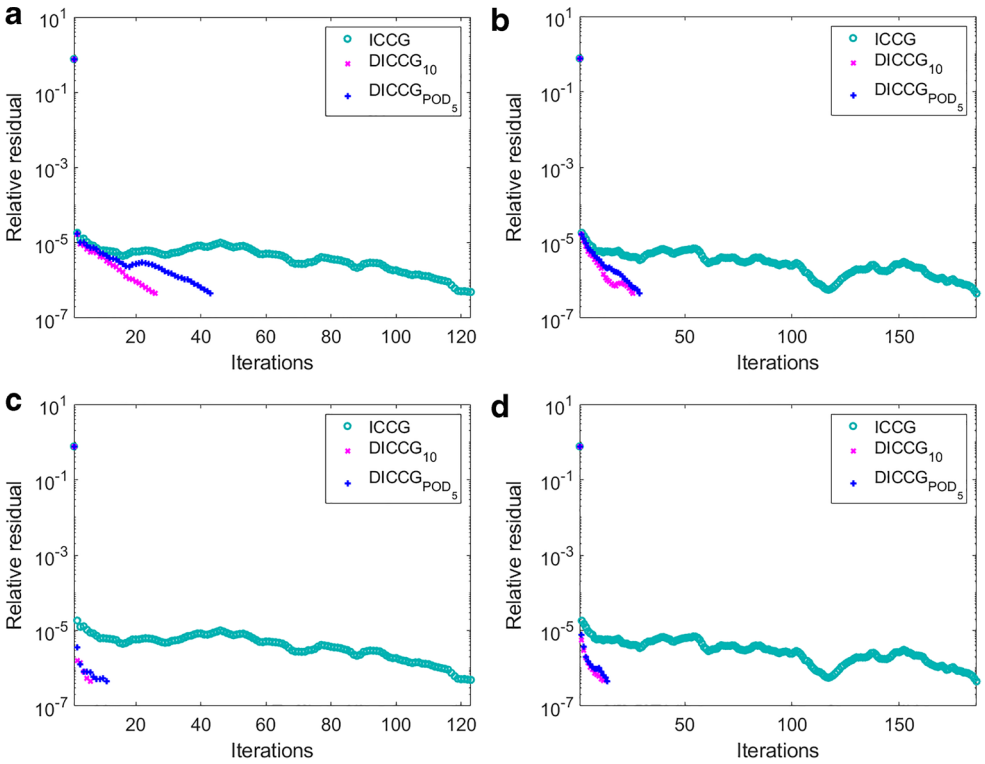

Fig. 19 Relative residual for various methods, **a** $TC1_{MW}$, **b** $TC2_{MW}$, **c** $TC3_{TP}$ **d** $TC4_{TP}$, injection through wells

For these cases (see Table 9), the reduction on the number of ICCG iterations and ICCG work is smaller than for the 2D cases. Implementing the MW approach ($TC9_{MW}$ and $TC10_{MW}$), the reduction on the ICCG work is ∼37% when using ten snapshots as deflation vectors ($DICCG_{10}$), and ∼44% using 5 POD-based deflation vectors ($DICCG_{POD_5}$). When implementing the TP approach ($TC11_{TP}$ and $TC12_{TP}$), the reduction on the ICCG work is ∼45% when

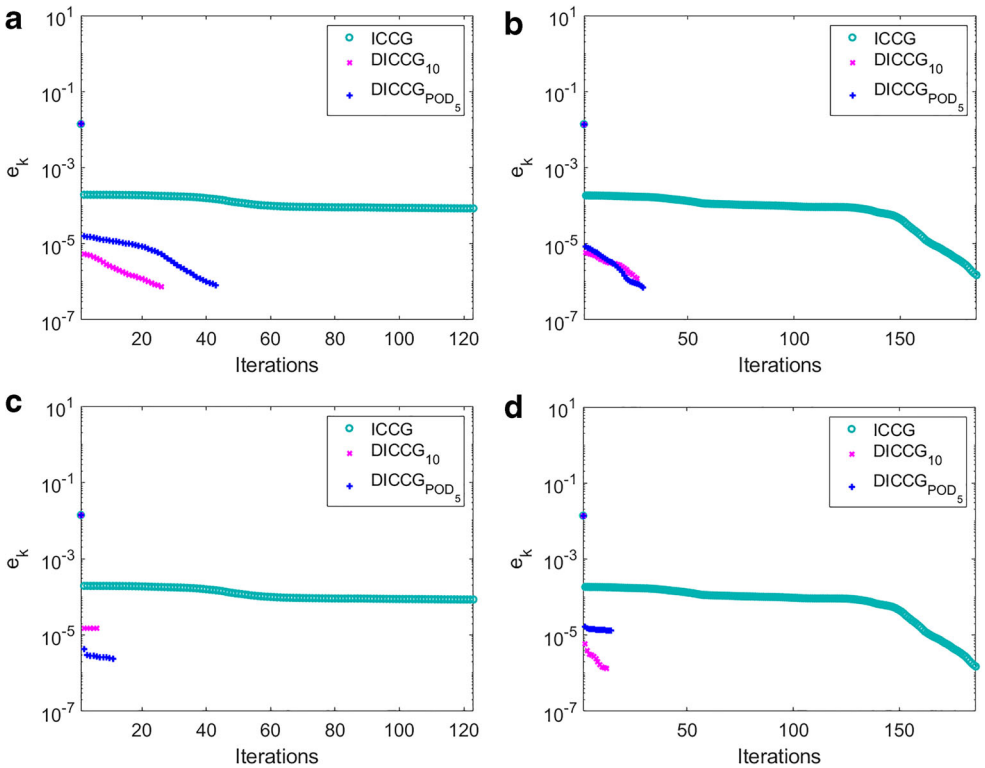

Fig. 20 True relative error for various methods, **a** $TC1_{MW}$, **b** $TC2_{MW}$, **c** $TC3_{TP}$ **d** $TC4_{TP}$, injection through wells

Table 9 Number of iterations for various methods, $\epsilon = 5 \cdot 10^{-9}$ for the 3D case

	p	Total	DICCG		Total	% of ICCG	% of ICCG
		ICCG	ICCG	DICCG	DICCG	Iterations	Work
3D case							
$TC9_{MW}$	10	87789	5280	14798	20078	23	35.2
	5_{POD}			17084	22364	25	42.3
$TC10_{MW}$	10	76878	5323	14285	19608	26	39.3
	5_{POD}			15945	21268	28	46.1
$TC11_{TP}$	10_{POD}	87789			25410	29	44.6
	5_{POD}				40959	47	71.9
$TC12_{TP}$	10_{POD}	76878			22740	30	45.6
	5_{POD}				35617	46	71.4

The cases TC9 and TC11 do not contain capillary pressure terms, while the TC10 and TC12 cases contain these terms. The sub-index $_{POD}$ in the p column means that the deflation vectors are POD basis vectors, if no sub-index is included, it means that the deflation vectors are snapshots

using ten POD-based deflation vectors ($\text{DICCG}_{POD_{10}}$), and ∼72% using 5 POD-based deflation vectors (DICCG_{POD_5}), making this approach more expensive, specially when using 5 POD based deflation vectors.

Discussion As in the previous section, we observe an important reduction on the the number of ICCG iterations when using a deflated version of this method, during the simulation of waterflooding for the SPE 10 benchmark, for the 2D case. Furthermore, the first iteration leads to a reduction of $\sim\mathcal{O}(10^{-4})$. A similar performance is observed for cases with and without capillary pressure terms. However, less work is required for the cases without capillary pressure terms. The MW approach results in a better performance of the DICCG method. However, it does not reach the desired accuracy for some test cases, but it gives a better approximation than the ICCG method. Regarding the TP approach, a single basis computed offline is used to solve diverse problems, resulting in similar performance for all of them.

For the 3D case, the reduction on the % of ICCG iterations is smaller, and as consequence the reduction on the total work is also smaller.

For the TP cases, the best performance is observed using ten POD basis vectors as deflation vectors.

4 Conclusions

In this work, we study the POD-deflation method for two-phase reservoir simulation problems. We test this methodology for several cases presenting a contrast in permeability coefficients up to $\mathcal{O}(10^8)$ and including capillary pressure and gravity terms. We study two different approaches: *Moving Window (MW)* and *Training Phase (TP)*, that differ in they way the snapshots are collected.

We compare the DICCG (Deflated Conjugate Gradient preconditioned with Incomplete Cholesky) method with the ICCG (Conjugate Gradient preconditioned with the Incomplete Cholesky factorization) method, and we found that:

- The performance of the DICCG method depends on the information contained in the POD basis. It is mainly related to the eigenvalues of the correlation matrix. If only a few eigenvalues are noticeably larger than the rest, they contain most of the system information leading to better DICCG performance.
- Using the POD basis as deflation vectors requires some extra work, however, this ensures linear independence of the vectors, making the deflation method more robust.
- The total work is reduced up to 23% the ICCG work when using the DICCG method, for the 2D case, and up to 35% for the 3D case.
- The MW approach showed a better performance than the TP approach for the studied cases.
- A slightly better performance is observed if no capillary pressure terms are included.
- For the TP approach, if a single basis, computed offline is reused for alike problems, a similar performance of the DICCG method is observed. Hence, once a basis is computed, it can be used to solve several problems.
- In some cases, the required accuracy is not achieved for both methods. However, the DICCG method is slightly more accurate. Thus, the DICCG method is more robust than the ICCG method for the studied problems.

We introduce this methodology for reservoir simulation examples, but it can be adapted to any time-varying problem. Moreover, we test it using the Preconditioned Conjugate Gradient method, but it is independent of the method, and it can also be implemented in combination with other iterative methods or preconditioners like multigrid.

Appendix A: Two-phase flow through porous media

For the simulation of two-phase (oil-water) flow through a porous medium, we can consider the phases as separated, i.e., they are immiscible and there is no mass transfer between them. We usually consider one of the fluids as the wetting phase (w), which is more attracted to the mineral particles than the other phase, known as the non-wetting phase (nw). In the case of a water-oil system, water is considered the wetting phase.

The saturation of a phase (S_α) is the fraction of void space filled with that phase in the medium. When modeling two-phase flow, the permeability of each phase, α, will be affected by the presence of the other phase. Therefore, an effective permeability, K_α, has to be used instead of the absolute permeability K, and relative permeability values are taken into account.

$$k_{r\alpha}\left(S_\alpha\right) = K_\alpha^e / K, \tag{19}$$

As in the single-phase case, the governing equations for two-phase flow in a porous medium are the mass conservation principle and Darcy's law. The mass balance equation for a phase α is given by:

$$\frac{\partial(\phi\rho_\alpha S_\alpha)}{\partial t} + \nabla\cdot(\rho_\alpha \mathbf{v}_\alpha) = \rho_\alpha q_\alpha, \tag{20}$$

and the Darcy's law reads:

$$\mathbf{v}_\alpha = -\frac{k_{r\alpha}}{\mu_\alpha} K\left(\nabla p_\alpha - \rho_\alpha g \Delta d\right), \tag{21}$$

where ρ_α, μ_α, q_α and p_α are the density, viscosity, sources and pressure of each phase, g is the gravity constant, and d is the depth of the reservoir.

Combining Darcy's law (21), the mass balance (20) and using the phase mobilities, the system reads:

$$\frac{\partial(\phi\rho_\alpha S_\alpha)}{\partial t} - \nabla\cdot(\rho_\alpha\lambda_\alpha(\nabla p_\alpha - \rho_\alpha g\Delta d)) = \rho_\alpha q_\alpha, \tag{22}$$

which is a parabolic equation for pressures and saturations, and where

$$\lambda_\alpha\left(S_\alpha\right) = \frac{K k_{r\alpha}\left(S_\alpha\right)}{\mu_\alpha}. \tag{23}$$

The previously-mentioned equations can be separated into a pressure equation and a saturation or transport equation via the fractional flow formulation [27], approach used in this work. For an immiscible, incompressible flow, the pressure equation becomes elliptic and the transport equation becomes hyperbolic. With this formulation, the pressure and transport equations are solved in separate steps in a sequential procedure, for more details see [14]. This approach is used throughout this work, therefore, we present a brief description of the method in Appendix B.

Appendix B: Fractional flow formulation

In the case of incompressible flow, the porosity ϕ and the densities ρ_α do not depend on the pressure. Considering a two-phase system with a wetting (w) and a non wetting phase (nw). Therefore, (20) for each phase reduces to:

$$\phi\frac{\partial S_w}{\partial t} + \nabla\cdot(\mathbf{v}_w) = q_w, \qquad \phi\frac{\partial S_{nw}}{\partial t} + \nabla\cdot(\mathbf{v}_{nw}) = q_{nw}. \tag{24}$$

To solve them, the fractional flow formulation takes the total Darcy's velocity as the sum of the velocity in both phases:

$$\begin{aligned}\mathbf{v} &= \mathbf{v}_w + \mathbf{v}_{nw} = -(\lambda_{nw}+\lambda_w)\nabla p_{nw} + \lambda_w\nabla p_c \\ &\quad +(\lambda_{nw}\rho_{nw} + \lambda_w\rho_w) g\Delta d.\end{aligned} \tag{25}$$

Summing up the continuity equations for the two phases we obtain:

$$\phi\tfrac{\partial(S_w+S_{nw})}{\partial t} + \nabla\cdot(\mathbf{v}_w + \mathbf{v}_{nw}) = \nabla\cdot\mathbf{v} = q, \tag{26}$$

where $q = q_{nw} + q_w$ is the total source term. Defining the total mobility as $\lambda = \lambda_{nw} + \lambda_w$ (see Eq. 23), and using Darcy's law, (26) becomes:

$$-\nabla\cdot(\lambda\nabla p_{nw}) = q - \nabla\left[\lambda_w\nabla p_c + (\lambda_{nw}\rho_{nw} + \lambda_w\rho_w) g\Delta d\right], \tag{27}$$

which is an equation for the pressure of the non wetting phase. This equation depends on the saturation via the capillary pressure p_c and the total mobility λ.

Multiplying each phase velocity by the relative mobility of the other phase and subtracting the result we get:

$$\lambda_{nw}\mathbf{v}_w - \lambda_w\mathbf{v}_{nw} = \lambda_w\lambda_{nw}[\nabla p_c + (\rho_w - \rho_{nw})g\Delta d]. \tag{28}$$

Therefore, for the wetting phase velocity, $\mathbf{v}_w$, we have:

$$\mathbf{v}_w = \frac{\lambda_w}{\lambda}\mathbf{v} + \frac{\lambda_w\lambda_{nw}}{\lambda}[\nabla p_c + (\rho_w - \rho_{nw})g\Delta d]. \tag{29}$$

We introduce the fractional flow function,

$$f_w(S_w) = \frac{\lambda_w(S_w)}{\lambda_w(S_w) + \lambda_{nw}(S_{nw})}, \tag{30}$$

which, together with the previously computed velocity $\mathbf{v}_w$, transforms the transport Eq. (20), idem for the rest of the referenced equations in Appendix A to:

$$\phi\frac{\partial S_w}{\partial t}+\nabla\cdot[f_w(\mathbf{v}_w+\lambda_{nw}\Delta\rho g\Delta d)]+\nabla\cdot(f_w\lambda_{nw}\nabla p_c)=q_w, \tag{31}$$

where $\Delta\rho=\rho_w-\rho_{nw}$.

With this approach, the system is expressed in terms of the non wetting phase pressure, (27), and the saturation of the wetting phase, (31). In the pressure equation, the coupling to saturation is present via the phase mobilities, and the derivative of the capillary function. For the saturation, we have an indirect coupling with the pressure through the total Darcy velocity, (25). With this scheme, the equations are solved for the pressure using the previously computed saturation, and the saturation is updated by substituting the computed pressure.

Appendix C: Discretization methods

In this work, we use the sequential scheme to simulate two-phase flow. With this approach, an unknown is fixed, e.g., the saturation of the wetting phase (S_w), and the resulting elliptic equation is solved for the pressure of the non-wetting phase (p_{nw}). Once p_{nw} is computed, we update the total velocity, $\mathbf{v}$, and we solve the parabolic transport equation for S_w (more details in [14]).

The resulting system depends on space and time. The space derivatives are discretized using finite differences scheme; for the temporal discretization, we use the backward Euler method, details can be found in [14]. In the examples presented in Section 3, the discretization is performed with the Matlab Reservoir Simulation Toolbox, MRST [30].

Table 10 Comparison of flops required with the ICCG and the DICCG methods

Solving $\mathbf{Ax}=\mathbf{b}$; $\mathbf{A}\in\mathbb{R}^{n\times n}$, $\mathbf{b},\mathbf{x}\in\mathbb{R}^n$.	
Initial work	
ICCG	$\left(m^2+4m+2\right)n$
DICCG	$\left(m^2+4m+2\right)n+\left(4p^2+2mp+2p\right)n+\frac{p^3}{3}-p^2-p$
$\frac{\text{DICCG}}{\text{ICCG}}$	$1+\frac{4p+2m+2}{m^2+4m+2}+\frac{(\frac{p^2}{3}-p-1)p}{(m^2+4m+2)n}$
POD basis	$[(8s-1)n+10s^2-2s+1]s$, s is the number of snapshots
Work per iteration	
ICCG	$(4m+9)n$
DICCG	$(4p+4m+9)n-p$
$\frac{\text{DICCG}}{\text{ICCG}}$	$\sim 1+\frac{4p}{4m+9}$

Appendix D: Complexity

This appendix is devoted to the computation of the number of operations necessary to implement the methods studied throughout this work. For the implementation of the POD-based deflation method, first, we need to compute the snapshots with the ICCG method. Once the snapshots are computed, we obtain the eigenvalues (Λ) and eigenvectors ($\mathbf{V}$) of the covariance matrix $\overline{\mathbf{R}}\in\mathbb{R}^{n\times n}$ by computing the SVD of $\overline{\mathbf{R}}^T=\overline{\mathbf{X}}\overline{\mathbf{X}}^T\in\mathbb{R}^{p\times p}$. The eigenvalues of both matrices are the same and the eigenvectors of $\overline{\mathbf{R}}$ can be computed from:

$$\mathbf{U}=\overline{\mathbf{X}}\mathbf{V}\left(\Lambda^T\right)^{\frac{1}{2}},$$

where $\mathbf{V}$ are the eigenvectors of $\overline{\mathbf{R}}^T$ (see [17, 40]). The eigenvectors corresponding to the largest eigenvalues are selected as the POD basis.

The operational cost of the ICCG method, the DICCG method, and the SVD process is presented in Table 10 (see [17]). For the computation of the number of flops of the DICCG method, we assume that the matrix $\mathbf{Z}$ is already given, i.e. it does not change during the iteration process. The flops are computed for sparse matrices with m the number of non-zero diagonals, and p the number of deflation vectors.

Funding This study was funded by the Mexican Insitute of Petroleum (IMP), the 'Consejo Nacional de Ciencia y Tecnología (CONACYT)', and the 'Secretaría de Energía (SENER)', through the programs: 'Programa de Captación de Talento, Reclutamiento, Evaluación y Selección de Recursos Humanos (PC-TRES)', and 'Formación de Recursos Humanos Especializados para el Sector Hidrocarburos (CONACYT-SENER Hidrocarburos, 2015-2016, CUARTO PERIODO)'.

Declarations

Conflict of Interests The authors declare that they have no conflict of interest.

References

1. Gaul, A.: Recycling Krylov subspace methods for sequences of linear systems. Ph.D. thesis, Technical University of Berlin (2014)
2. Gaul, A., Gutknecht, M., Liesen, J., Nabben, R.: A framework for deflated and augmented Krylov subspace methods. SIAM J. Matrix Anal. Appl. **34**(2), 495–518 (2013)
3. Quarteroni, A., Rozza, G.: Reduced Order Methods for Modeling and Computational Reduction, vol. 9. Springer, New York (2014)
4. Van der Sluis, A., van der Vorst, H.A.: The rate of convergence of conjugate gradients. Numer. Math. **48**(5), 543–560 (1986)
5. Van der Sluis, A., Van der Vorst, H.A.: The convergence behavior of Ritz values in the presence of close eigenvalues. Linear Algebra Appl. **88**, 651–694 (1987)
6. Beckermann, B., Kuijlaars, A.: Superlinear convergence of conjugate gradients. SIAM J. Numer. Anal. **39**(1), 300–329 (2001)
7. Smith, B., Bjorstad, P., Gropp, W.: Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations. Cambridge University Press, New York (1996)
8. Vuik, C., Segal, A., Meijerink, J.A.: An efficient preconditioned CG method for the solution of a class of layered problems with extreme contrasts in the coefficients. J. Comput. Phys. **152**, 385 (1999)
9. Vuik, C., Segal, A., Yaakoubi, L., Dufour, E.: A comparison of various deflation vectors applied to elliptic problems with discontinuous coefficients. Appl. Numer. Math. **41**(1), 219–233 (2002)
10. Pasetto, D., Ferronato, M., Putti, M.: A reduced order model-based preconditioner for the efficient solution of transient diffusion equations. Int. J. Numer. Methods Eng. **109**(8), 1159–1179 (2017)
11. De, S.turler.: Eric: Truncation strategies for optimal Krylov subspace methods. SIAM J. Numer. Anal. **36**(3), 864–889 (1999)
12. Ebadi, G., Alipour, N., Vuik, C.: Deflated and augmented global Krylov subspace methods for the matrix equations. Appl. Numer. Math. **99**, 137–150 (2016)
13. Marcondes, F., Maliska, C.R., Zambaldi, M.: A comparative study of implicit and explicit methods using unstructured voronoi meshes in petroleum reservoir simulation. J. Braz. Soc. Mech. Sci. Eng. **31**(4), 353–361 (2009)
14. Diaz Cortes, G.B.: POD-based deflation method for reservoir simulation, Ph.D. thesis, Delft University of Technology (2019)
15. Diaz Cortes, G.B., Vuik, C., Jansen, J.D.: On POD-based deflation vectors for DPCG applied to porous media problems Report, 17-1, Delft University of Technology, Delft Institute of Applied Mathematics Delft (2017)
16. Diaz Cortes, G.B., Vuik, C., Jansen, J.D.: On POD-based deflation vectors for DPCG applied to porous media problems. J. Comput. Appl. Math. **330**(Supplement C), 193–213 (2018)
17. Diaz Cortes, G.B., Vuik, C., Jansen, J.D.: POD-based deflation techniques for the solution of two-phase flow problems in large and highly heterogeneous porous media Report 18-1, Delft University of Technology, Delft Institute of Applied Mathematics, Delft (2018)
18. Klie, H., Wheeler, M.F., Stueben, K., Clees, T., et al: Deflation AMG solvers for highly ill-conditioned reservoir simulation problems. In: SPE Reservoir Simulation Symposium Society of Petroleum Engineers (2007)
19. Van der Vorst, H.A.: Iterative Krylov Methods for Large Linear Systems, vol. 13. Cambridge University Press, Cambridge (2003)
20. Van der Vorst, H.A., Dekker, C.: Conjugate gradient type methods and preconditioning. J. Comput. Appl. Math. **24**, 73–87 (1988)
21. Liesen, J., Tichý, P.: Convergence analysis of Krylov subspace methods. GAMM-Mitt. **27**(2), 153–173 (2004)
22. Tang, J., Vuik, C.: On deflation and singular symmetric positive semi-definite matrices. J. Comput. Appl. Math. **206**(2), 603–614 (2007)
23. Tjan, J.: Comparison of two-level preconditioners using deflation techniques applied to flow problems. Master's thesis, TU Delft. https://repository.tudelft.nl/islandora/object/uuid%3A5d702116-1c72-4509-b24c-b83f2039e2f8 (2018)
24. van Doren, J., Markovinović, R., Jansen, J.D.: Reduced-order optimal control of water flooding using proper orthogonal decomposition. Comput. Geosci. **10**(1), 137–158 (2006)
25. Jansen, J.D.: A Systems Description of Flow Through Porous Media. Springer, New York (2013)
26. Tang, J.M., Nabben, R., Vuik, C., Erlangga, Y.: Comparison of two-level preconditioners derived from deflation, domain decomposition and multigrid methods. J. Sci. Comput. **39**(3), 340–370 (2009)
27. Aziz, K., Settari, A.: Petroleum reservoir simulation. Chapman & Hall, London (1979)
28. Carlberg, K., Forstall, V., Tuminaro, R.: Krylov-subspace recycling via the POD-augmented conjugate-gradient method. SIAM J. Matrix Anal. Appl. **37**(3), 1304–1336 (2016)
29. Kahl, K., Rittich, H.: The deflated conjugate gradient method: Convergence, perturbation and accuracy. Linear Algebra Appl. **515**, 111–129 (2017)
30. Lie, K.-A.: An introduction to reservoir simulation using MATLAB/GNU Octave: User guide for the MAT-LAB Reservoir Simulation Toolbox (MRST). Cambridge University Press (2019)
31. Bergamaschi, L.: A Survey of low-rank updates of preconditioners for sequences of symmetric linear systems. Algorithms **13**(4), 100 (2020)
32. Benzi, M.: Preconditioning techniques for large linear systems: a survey. J. Comput. Phys. **182**(2), 418–477 (2002)
33. Clemens, M., Wilke, M., Schuhmann, R., Weiland, T.: Subspace projection extrapolation scheme for transient field simulations. IEEE Trans. Magn. **40**(2), 934–937 (2004)
34. Parks, M., De Sturler, E., Mackey, G., Johnson, D., Maiti, S.: Recycling Krylov subspaces for sequences of linear systems. SIAM J. Sci. Comput. **28**(5), 1651–1674 (2006)
35. Cardoso, M.A., Durlofsky, L.J., Sarma, P.: Development and application of reduced-order modeling procedures for subsurface flow simulation. Int. J. Numer. Methods Eng. **77**(9), 1322–1350 (2009)
36. Christie, M.A., Blunt, M.J.: Tenth SPE comparative solution project: a comparison of upscaling techniques. SPE Reserv. Eng. Eval. **4**(4), 308–317 (2001)
37. Gutknecht, M.H.: Deflated and augmented Krylov subspace methods: Basic facts and a breakdown-free deflated. MINRES, MATHEON Preprint 759 Technical University (2011)
38. Axelsson, O., Kaporin, I.: On the sublinear and superlinear rate of convergence of conjugate gradient methods. Numer. Algoritm. **25**(1-4), 1–22 (2000)
39. Coulaud, O., Giraud, L., Ramet, P., Vasseur, X.: Deflation and augmentation techniques in Krylov subspace methods for the solution of linear systems.arXiv:1303.5692 (2013)
40. Astrid, P., Papaioannou, G., Vink, J.C., Jansen, J.D.: Pressure Preconditioning Using Proper Orthogonal Decomposition. In: 2011 SPE Reservoir Simulation Symposium, The Woodlands, pp. 21–23, Texas USA (2011)
41. Vermeulen, P.T.M., Heemink, A.W., Te Stroet, C.B.M.: Reduced models for linear groundwater flow models using empirical orthogonal functions. Adv. Water Resour. **27**(1), 57–69 (2004)
42. Markovinović, R., Jansen, J.D.: Accelerating iterative solution methods using reduced-order models as solution predictors. Int. J. Numer. Methods Eng. **68**(5), 525–541 (2006)
43. Morgan, R.B.: A restarted GMRES method augmented with eigenvectors. SIAM J. Matrix Anal. Appl. **16**(4), 1154–1171 (1995)
44. Morgan, R.B.: GMRES with deflated restarting. SIAM J. Sci. Comput. **24**(1), 20–37 (2002)

45. Wang, S., Sturler, E., Glaucio, P.: Large-scale topology optimization using preconditioned Krylov subspace methods with recycling. Int. J. Numer. Methods Eng. **69**(12), 2441–2468 (2007)
46. Saad, Y. Iterative Methods for Sparse Linear Systems., 2nd edn. Society for Industrial and Applied Mathematics Philadelphia, PA, USA SIAM (2003)
47. Bjørnarå, T., Aker, E.: Comparing equations for two-phase fluid flow in porous media. In: COMSOL conference Hannover (2008)
48. Heijn, T., Markovinovic, R., Jansen, J.D., et al: Generation of low-order reservoir models using system-theoretical concepts. SPE J. **9**(02), 202–218 (2004)
49. Tang, J.: Two-Level Preconditioned Conjugate Gradient Methods with Applications to Bubbly Flow Problems. Ph.D. thesis, Delft University of Technology (2008)
50. Schilders, W., Van der Vorst, H., Rommes, J.: Model order reduction: theory, research aspects and applications, vol. 13. Springer, New York (2008)