Comput Geosci (2017) 21:247–266
DOI 10.1007/s10596-016-9610-3

ORIGINAL PAPER

Dimensionality reduction for production optimization using polynomial approximations

Nadav Sorek[1] · Eduardo Gildin[1] · Fani Boukouvala[3] · Burcu Beykal[2,4] · Christodoulos A. Floudas[2,4]

Received: 20 May 2016 / Accepted: 21 December 2016 / Published online: 27 January 2017

Abstract The objective of this paper is to introduce a novel paradigm to reduce the computational effort in waterflooding global optimization problems while realizing smooth well control trajectories amenable for practical deployments in the field. In order to overcome the problems of slow convergence and non-smooth impractical control strategies, often associated with gradient-free optimization (GFO) methods, we introduce a generalized approach which represent the controls by smooth polynomial approximations either by a polynomial function or by a piecewise polynomial interpolation, which we denote as function control method (FCM) and interpolation control method (ICM), respectively. Using these approaches, we aim to optimize the coefficients of the selected functions or the interpolation points in order to represent the well-control trajectories along a time horizon. Our results demonstrate significant computational savings, due to a substantial reduction in the number of control parameters, as we seek the optimal polynomial coefficients or the interpolation points to describe the control trajectories as opposed to directly searching for the optimal control values (bottom hole pressure) at each time interval. We demonstrate the efficiency of the method on two and three-dimensional models, where we found the optimal variables using a parallel dynamic-neighborhood particle swarm optimization (PSO). We compared our FCM-PSO and ICM-PSO to the traditional formulation solved by both gradient-free and gradient-based methods. In all comparisons, both FCM and ICM show very good to superior performances.

Keywords Optimization dimensionality reduction · Polynomial control method · Control set cardinality reduction · Parametrization · Waterflooding optimization · Production optimization · Adjoint method · Particle swarm optimization · Smooth well-control

1 Introduction

The evolution of smart downhole flow-control valves as part of an intelligent completion of a well bore has encouraged engineers in the last decades to optimize valves control settings along the well bore and has opened the gate to optimal control (see many examples in [1–6]). It has also lead to greater optimization complexity, as the optimization entails each valve setting in different time intervals as decision (optimization) variables, which form a high-dimensional search space. As the number of wells and the number

C. A. Floudas, deceased on August 14, 2016.

✉ Eduardo Gildin
egildin@tamu.edu

Nadav Sorek
nadav.sorek@tamu.edu

Fani Boukouvala
fani.boukouvala@chbe.gatech.edu

Burcu Beykal
burcubeykal@tamu.edu

[1] Harold Vance Department of Petroleum Engineering, Texas A&M University, College Station, TX 77843-3116, USA

[2] Artie McFerrin Department of Chemical Engineering, Texas A&M University, College Station, TX 77843, USA

[3] School of Chemical & Biomolecular Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0100, USA

[4] Texas A&M Energy Institute, Texas A&M University, 302D Williams Administration Bldg, College Station, TX 77843-3372, USA

of discretized time steps become increasingly large, the optimization problem begins to suffer from the well-known mathematical-algorithmic hurdle, the curse of dimensionality. That is, as the mathematical space grows, the task of finding an optimal combination of all decision-variables becomes progressively harder.

The solution of this class of well-control optimization problems is directly tied with the optimization solver which will be used. Traditionally, this optimal control problem is solved by a gradient-based method, where the gradients are acquired using the adjoint method [43]. Gradient-based optimization (GBO) methods use derivative information of the simulation to guide the search, and under certain continuity assumptions of the control profiles, as well as the prerequisite that derivatives can be calculated and are not noisy, they are designed to have a globally convergent behavior towards local stationary points, regardless of the initialization of the algorithm. A review of globally convergent gradient-based optimization methods for dynamic process optimization using interior-point methods can be found in [8]. Local gradient-based methods have the advantages of robustness and providing smooth solutions; however, they require an initial point which might affect the final local solution obtained.

On the other hand, recently, there is a rising interest in solving the well-control production optimization problem in waterflooding scenario using gradient-free optimization (GFO) methods, usually in conjunction with additional problems such as well-placement. Gradient-free optimization algorithms are either stochastic and based purely on sampling [15, 20] or use surrogate approximating functions [9, 10], due to the fact that calculation or approximation of the true gradients is unreliable (noisy derivative information) or not possible (due to proprietary codes or high computational cost of the simulation). GFO methods can be generically used for any simulation since they do not depend on derivatives, however, they tend to lead to non-smooth profiles for well-control optimization problems. The interested reader is referred to a recent review article with a comprehensive discussion on deterministic mixed integer nonlinear programming and constrained derivative-free optimization methods, algorithms, and applications [11].

Although many of the GFO methods aim at obtaining a global solution, they might converge to poor solutions when the curse of dimensionality is present. In fact, in a more pragmatic sense, it is often necessary to accept good quality solutions rather than the sole (global) optimal solution. Therefore, one of the most appealing approaches to obtain good quality solutions and to improve GFO performances for any high-dimensional problem would be to apply an optimization dimensionality reduction (ODR) by some parametrization procedure. Here, ODR refers to a reduction in the number of decision-variables within a given global optimization problem. The aim of this work is to implement efficient ODR methods which yield good quality solutions within a feasible computational time using gradient-free method. Comparison to gradient-based method performances is also provided.

Note that we distinguish between model order reduction (MOR) and ODR. While the first aims at improving the efficiency by accelerating a single forward simulation using model reduction techniques (see examples in [18, 23, 26–28, 41]), the second accelerates the optimization by a reduced optimization representation. In other words, MOR methods reduce the cardinality of the simulation variable set, while ODR methods reduce the cardinality of the optimization variables set. Work that combines the two approaches is currently underway, though out of the scope of this work.

1.1 Production optimization using polynomial approximation

In this paper, we address ODR for the production optimization step of closed-loop reservoir management (CLRM) [32] in a waterflooding campaign. We consider a fixed location for all wells and we strive to find an optimal (or near-optimal) controls trajectory of each well. We introduce a general approach which uses smooth polynomial approximations of the original decision variable space, which leads to a significant reduction in the decision space. We obtain the polynomial trajectories by two different distinct approaches: functional control method (FCM) and interpolation control method (ICM).

In the FCM, we look for optimal function coefficients to describe well-controls along a time horizon. In particular, in this paper, we use a polynomial control strategy, named polynomial control method (PCM), in which a well-control cardinality set is reduced from the number of control time steps to the number of polynomial coefficients.

In the second approach, the ICM, we look for an optimal piecewise polynomial interpolant. The optimization variables in this case are the interpolation points (knots). To the best of our knowledge, this is the first work which treats the waterflooding production optimization problem with the interpolation concept.

FCM for rate control optimization was first introduced by Awontunde [2] where he considered both polynomial and trigonometrical control functions to represent the rate control trajectories. He observed that for relatively large problems the cosine and linear approaches outperform the constant rate and stepwise approaches (up to a specified number of iterations. That is, conclusions were drawn before convergence). He also observed that low degree polynomials perform better than higher ones (he tested linear, cubic, and fifth degree polynomials), which is counterintuitive as one might expect high degree polynomials to better capture the system dynamics. In this work, we present that higher objective function values can be achieved using higher degree polynomial functions, which also overcomes the aforementioned conflict.

1.2 Production optimization solution is often bang-singular

The work in [2] performed FCM with wells flow rates as the optimization (decision) variables. Here, on the other hand, we use the bottom hole pressures (BHP) as the optimization variables, which might require a different treatment. For example, Brouwer and Jansen [12] observed that optimal rates are smooth, but that the optimal valve settings sometimes follow a bang-bang (on-off) behavior. Following this work and the work by Sudaryanto and Yortsos [48, 49], Zandvliet et al. [53] investigated why and under what conditions reservoir flooding problems can be expected to have bang-bang optimal solutions. They showed that these problems may have bang-bang optimal solutions in the cases where upper and lower bounds are the sole constraints on the control. They concluded that the optimal settings are sometimes pure bang-bang (solutions values are only on the boundaries) and sometimes bang-bang in combination with so-called singular arcs (in which the control trajectory proceeds smoothly).

This insight is highly relevant for the GFO polynomial approach, as it is obligatory for any polynomial formulation to be able to impose controls on the boundaries at any given time interval along the simulation. Unfortunately, the formulation shown in [2] only allows the controls to be on the bounds only at initial or terminal time and thus probably misses good quality solutions (see further discussion in Section 5.1). Interestingly, while GBO tend to reach to smooth solutions, GFO methods indeed produce bang-bang (or near bang-bang) type solutions, where the controls repeatedly vary between low and high values. Some examples can be found in the control solutions obtained by different GFO methods in [14, 22, 55]. Though these type of solutions might represent the true optimum, they might be unfavorable from a production-engineering point of view, as repeated large changes might cause damage to equipment and formation. Also, from a reservoir simulation point of view, drastic changes in the well schedule controls might cause simulation convergence problems. Thus, smooth polynomial curves might serve as a possible remedy for these problems.

In certain cases, bang-bang control is desirable and researchers have developed optimization methods to be able to differential dynamic programming techniques to solve such systems [31]. In other cases, it is desirable to smooth out the bang-bang profiles in order to obtain practically implementable control strategies. There are many previous attempts within the optimal control community to reduce the numerical issues by smoothing the bang-bang type of solutions. As an example, we will note the work of Silva and Trelat [47] in which the author imposed regularization on the minimal time control problem for a single-input control affine system. They introduced a penalization parameter, ϵ, which smooths the control as its value increases and, conversely, provides a bang-bang solution as it tends to zero. They showed, however, that their method fails when the control is singular. Following [53], which showed that indeed reservoir flooding problems are linear in the control but at the same time, they might result in bang-singular solutions. This concludes that the proposed regularization method is not appropriate for the reservoir flooding problems. Other attempts to solve bang-bang control problems involve a variety of smoothing techniques, such as quadratic penalty functions and logarithmic barrier function [7], which again allow the smoothing of the control profile as a small parameter ϵ is tuned. A very recent article provides detailed convergence analysis of regularization of control problems with bang-bang solutions [44].

This paper proceeds as follows. In Section 2, we review the previous attempts to perform ODR for well-control production optimization problem. Then, in Section 3, we outline the traditional well-control production optimization formulation. In Section 4, we present a motivating example, solved by the traditional formulation, which demonstrates the importance of proposing polynomial formulations for the problem. In Sections 5.1 and 5.2, we present our polynomial FCM formulation and the piecewise polynomial ICM, respectively. Furthermore, in Section 6, we demonstrate the efficiency of our proposed method by providing computational results using two- and three-dimensional models. We conclude in Section 7 with final remarks and suggestions for future work.

2 ODR in well-control production optimization

In this section, we briefly review previous attempts, different than polynomial function discussed in [2], to reduce the size of the well-control production optimization problem. Lien et al. [36] proposed two multi-scale procedures named ordinary multi-scale and refinement indicators, where they performed multiple optimization cycles starting from a coarse problem representation. The ordinary multi-scale method simply splits each coarse variable into two new refined variables during each optimization cycle. In the second approach, the authors refine on some of the variables, rather than all of them, by using refinement indicators equal to the gradient magnitude of the objective function with respect to the variables. The two approaches parameterized the problem in both time and space. That is, a coarse decision variable could represent either several different time steps (parameterization in time) or a group of control valves (parameterization in space), which all share the same control settings. A similar approach can also be found in [46].

Oliveira and Reynolds [42] further developed the multiscale approach, by adaptive parametrization that enables a merging procedure, in addition to the splitting procedure. Thus, unlike the previous method, their approach can start with a refined parametrization as they enable both refining and coarsening of the problem representation. In their methodology, three criteria need to hold in order to merge a variable with its previous control step variable. If the three criteria do not hold, this variable is split into a predefined number of new variables (they tested two cases of splitting by factors of two and four). The three criteria are the control-variation, gradient-variation, and gradient magnitude. In a scenario when the gradient cannot be acquired (e.g., when the adjoint model is not available), the procedure considers only the first criteria. Their code accelerates the convergence of the first optimization cycles by imposing loose convergence conditions, which gradually become tighter in subsequent optimization cycles. They also treated bang-bang control solution by smoothing it with a covariance function.

We note here that there has been a growing interest in the literature of jointly optimizing well-placement and well-control. This approach can be carried by two different algorithms in a nested framework, one for well-placement and the other for well-control optimization (for example, see [6]). However, more recently, there has been extensive research on solving the entire field development problem with a single algorithm. With this approach, well-placement variables, well-control variables, and, in some cases, other categorical variables, all reside in the same vector [3, 22, 29, 30, 40, 45]. Though we did not apply well-placement optimization in this paper, we state that applying ODR to the control variable sets would decrease the size of the overall optimization vector and improve generalized field development strategies solved by a single algorithm (as done in [2]).

In the following sections, we describe the traditional well-control optimization formulation. Then, we show how results from the traditional approach motivates formulating polynomial ODR approach.

3 Traditional box-constrained well-control optimization formulation

In this section, we describe the traditional box-constrained well-control optimization problem formulation. As mentioned in the introduction, such problems have been predominantly solved insofar by two optimization approaches: stochastic global search algorithms and local search algorithms. The local search can be either gradient-based [19, 32, 36, 43, 51, 52], approximated-gradient-based [13, 21, 55], or gradient-free [6, 14]. Hybrid methods of global and local search have been suggested as well [30].

In this work, we used particle swarm optimization (PSO) as the global optimizer. We also conduct a comparison to the rate of the objective function growth achieved by GBO local search algorithms while acquiring the gradients with the adjoint method as implemented in [34]. Further discussion regarding the algorithm specifications used in this work are found in Sections 4.1 and 6.2 and Appendix.

3.1 Optimization problem

The traditional box-constrained well-control optimization problem, called here $P1$, is defined as follows:

$$P1 : \max_{\mathbf{u}} \mathrm{NPV}(\mathbf{u}) \text{ subject to } \begin{cases} g(\mathbf{u}) = 0 \\ \mathbf{u} \in \mathbb{U} \end{cases} \tag{1}$$

Where the equality constraints $g(\mathbf{u}) = 0$ are the discretized partial differential equations (PDE) in a reservoir simulation. In this work, we used the sequential solver in Matlab Reservoir Simulation Toolbox (MRST) for two-phase incompressible flow of a wetting and non-wetting fluids [35]. By neglecting gravity effects, the PDE can be described as follows:

$$\nabla \cdot \mathbf{v} = q, \mathbf{v} = -\mathbf{K}(\lambda \nabla p) \tag{2}$$

$$\phi \frac{\partial s_w}{\partial t} + \nabla \cdot (f_w(s_w)\mathbf{v}) = q_w, \tag{3}$$

where $\mathbf{v}$ denotes the Darcy velocity, q total liquid flux, p pressure, s_w water saturation, $\mathbf{K}$ permeability tensor, λ the (total) mobility, $f_w(s_w)$ the water fractional flow as a function of saturation, ϕ porosity, and t time. First, pressure and fluxes are provided by fixing the saturation values and solving Eq. 2. Subsequently, fluxes are used in Eq. 3 to solve for the saturation in the next simulation time step. For further details the reader is referred to [34, 35].

The NPV objective function, shown in Eq. 4, accounts for revenue associated with produced oil and for the cost of handling produced and injected water (which incurred as a result of pumping and separation requirements),

$$\mathrm{NPV} = \sum_{k=1}^{N_{tsim}} \Delta t_k \cdot (1+r)^{-\frac{t_k}{t_{ref}}} \times \left(\sum_{j=1}^{N_p} r_o q_o^{j,k}(\mathbf{u}) - \sum_{j=1}^{N_p} c_{wp} q_{wp}^{j,k}(\mathbf{u}) - \sum_{j=1}^{N_i} c_{wi} q_{wi}^{j,k}(\mathbf{u}) \right), \tag{4}$$

where $q_o^{j,k}$, $q_{wp}^{j,k}$, $q_{wi}^{j,k}$ are the flow rates of the oil, water produced and water injected for well j at the simulation time interval k, respectively. The revenue from a unit of oil produced, the cost of a unit of water produced and a unit of water injected are represented by r_o, c_{wp}, and c_{wi}, respectively. r is a discount rate with respect to a time reference t_{ref}, and Δt_k represents the length of each of the N_{tsim} time steps in the simulation. N_i and N_P are the total number of injection and production wells, respectively.

3.2 The solution space

In the traditional optimization framework, the controls over time for each well are represented by a piecewise constant function with N_t time intervals. That is, the well-controls are held constant during an interval and updated for the next interval. Note that N_t is the number of control (well-schedule) time steps, while N_{tsim} is the number of simulation time steps (as used in Eq. 4). If needed, a surjective function maps one or several simulation time steps to a unique control step index, and thus, $N_t \leq N_{tsim}$ [16]. For the sake of simplicity, we used explicit sequential solver, where the number of control time steps in the well-schedule is equal to the number of simulation time steps, and thus $N_t = N_{tsim}$.

Let $N_w = (N_i + N_p)$ be the total number of injection and production wells, $j = 1, 2 \ldots N_w$ denote the well number, and $k = 1, 2 \ldots N_t$ denote a time interval. Each well-control variable $u_{j,k}$ can be described either by flow rates or by BHPs with the corresponding upper and lower bounds. Let us define the vector of well-control variables as $\mathbf{u} = (u_{1,1}, u_{1,2}, \ldots, u_{1,N_t}, \ldots, u_{N_w,N_t})$. Then, all the possible $\mathbf{u}$ vectors forms the solution space $\mathbb{U}$:

$$\mathbb{U} = \{\mathbf{u} \in \mathbb{R}^{N_w \cdot N_t} : \mathbf{u_{lb}} \leq \mathbf{u} \leq \mathbf{u_{ub}}\}, \tag{5}$$

where the box-constraints vectors inequality follows to definition below,

Definition 1 (**Vector Relation**) For two vectors $\mathbf{v} = (v_0, \ldots, v_m)$, $\mathbf{w} = (w_0, \ldots, w_m) \in \mathbb{R}^m$ we write $\mathbf{v} \leq \mathbf{w}$ if for every $0 \leq i \leq m$, $v_i \leq w_i$.

We note that an efficient log transformation can eliminate the hard bounds shown in Eqs. 1 and 5, transforming the constrained optimization problem to an unconstrained problem (see [24, 52]). However, this implementation was not considered in this current work.

The measure for the size of a vector space is its dimension, which is defined as the cardinality of its basis vectors set. Since all $u_{i,j}$s are linearly independent, the dimension of $\mathbb{U}$ can be defined as follows:

$$dim(\mathbb{U}) = N_w \cdot N_t. \tag{6}$$

In Section 4, we present an example solved by the traditional methods, namely global search and local search, which in turn motivates obtaining a new formulation for the well-control optimization problem. Then, in Section 5, we propose alternative formulations for the problem.

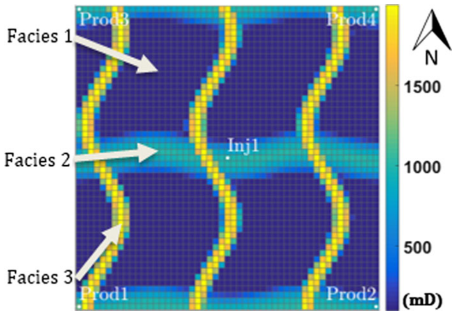
4 Motivating example

This section introduces a synthetic, two-dimensional and channelized reservoir, which serve as a motivating example for the proposed methods.

The model includes 51 by 51 grids consisting of four producing wells and one injector arranged in a five spot pattern. The grid cell dimensions are 20 ft × 20 ft × 20 ft. Figure 1a shows the permeability distribution as well as the five wells. There are three permeability sub-regions in three distinct geological facies: the original low-permeability geological environment (facies 1), east-west moderate permeability channels formed by ancient rivers (facies 2), and newer high-permeability channels whose flow eroded the environment from north to south (facies 3). Figure 1b provides a table with the minimum, maximum, and average permeability values for each sub-region as well as the homogeneous assigned porosity. The initial water saturation is 0.2 (equal to residual water saturation).

As mentioned, we used MRST flow transport two phase incompressible solvers as our forward model simulator [33, 34]. Throughout this work, we controlled all wells by

Fig. 1 **a** Permeability field and wells for the motivating example. **b** Permeability and porosity ranges for each facies



(a)

	Min Perm (mD)	Max Perm (mD)	Ave. Perm (mD)	Porosity
Facies 1	100	100	100	0.2
Facies 2	102.3	2000	1366.7	
Facies 3	100.2	1000	679.2	

(b)

adjusting the BHP, so that we determine the optimal BHP for each well at all the different time intervals. We simulated a one year production and adjusted the control every 10 days, leading to 185 optimization variables. Throughout this paper, we considered an oil price of \$100 per barrel, produced and injected water treatment costs of \$10 per barrel, and a discount rate of 10%.

4.1 Global search versus local search

We performed the optimization for this example using two approaches: global search with a parallelized dynamic-neighborhood particle swarm optimization (PSO) algorithm, as detailed in Appendix, and local search with the adjoint aggressive-line search method MRST implementation [35]. In the local search GBO, the iterations continued until the norm of the projected gradient and the relative objective function change were below a threshold (10^{-4} and 10^{-6}, respectively). In the PSO, the optimization continued until a predefined number (here, 20) consecutive iterations stalled below a relative objective function change (10^{-6}, similar to the local search GBO method).

Due to parallelization, each PSO iteration duration is approximately equivalent to one simulation run. On the other hand, gradient-based iterations have to be performed sequentially, as each simulation run (forward or adjoint) is dependent on the previous iteration. Therefore, even though gradient-based methods, in general, require far fewer simulation runs up to convergence, the choice of algorithm, in terms of execution time, depends on the computing capabilities at hand.

Figure 2 shows the best solution found by the two algorithms. We can see that the global optimizer found a solution with a slightly better NPV with more even water distribution throughout the reservoir. Though the convergence mechanism and convergence criteria differ between the two methods, our main goal in this section is to compare the shape of the control trajectory solutions found by each method, rather than the obtained objective function values.

We can see from Fig. 2 that while the gradient-based method exhibits a smooth solution, the stochastic nature of the global optimizer provides an erratic and impractical control strategy. This control strategy might not be acceptable since such dramatic repeated changes in the pressure will quickly cause a fatigue to production instruments and will damage the equipment. Therefore, in developing a new optimization method, it is better to achieve a smooth and practical solution for the control trajectories, while maintaining the ability to obtain better NPV values. Next, we discuss how the adjoint smooth solution, shown in Fig. 2, inspired us to develop a new method.

4.2 Smooth gradient-based solution as a motivator for polynomial approximation

Our notion of controlling a well with a polynomial approximation started from observing the smooth solution obtained by the adjoint gradient-based solution. Figure 3 shows the results of fitting these smooth trajectories to different polynomial functions (with degrees ranging from one to four). Of course, a better fit is observed for higher degrees, as higher order polynomials can capture multimodal trajectories (for

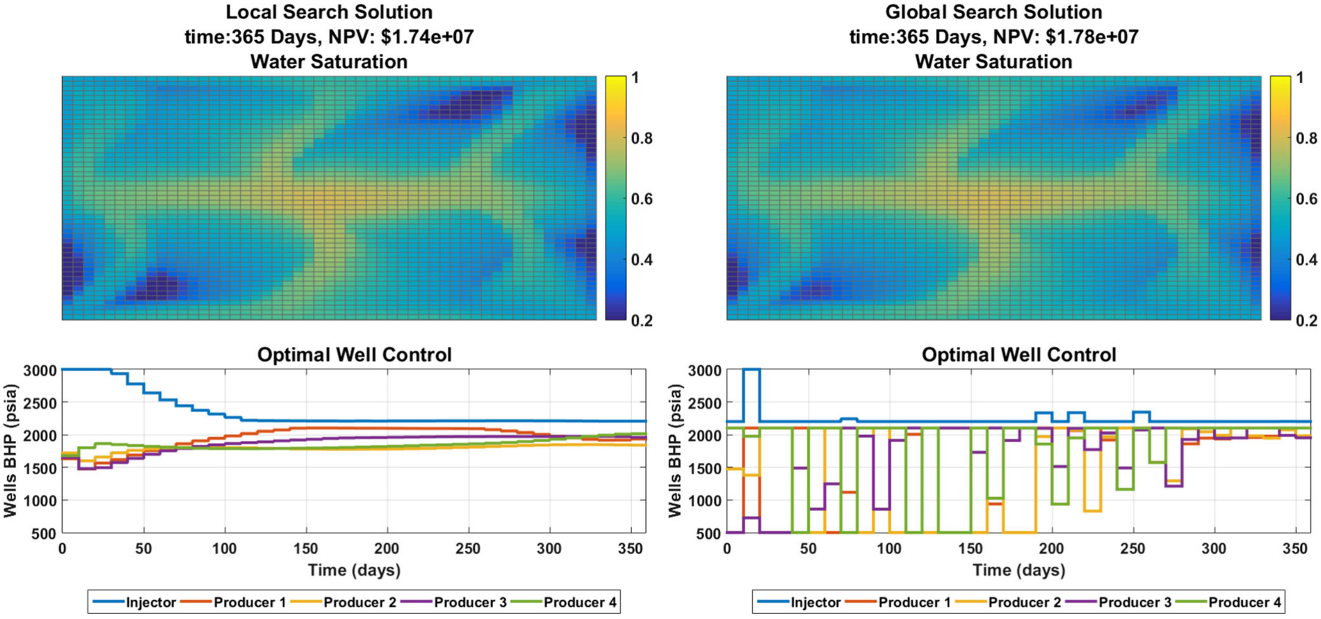
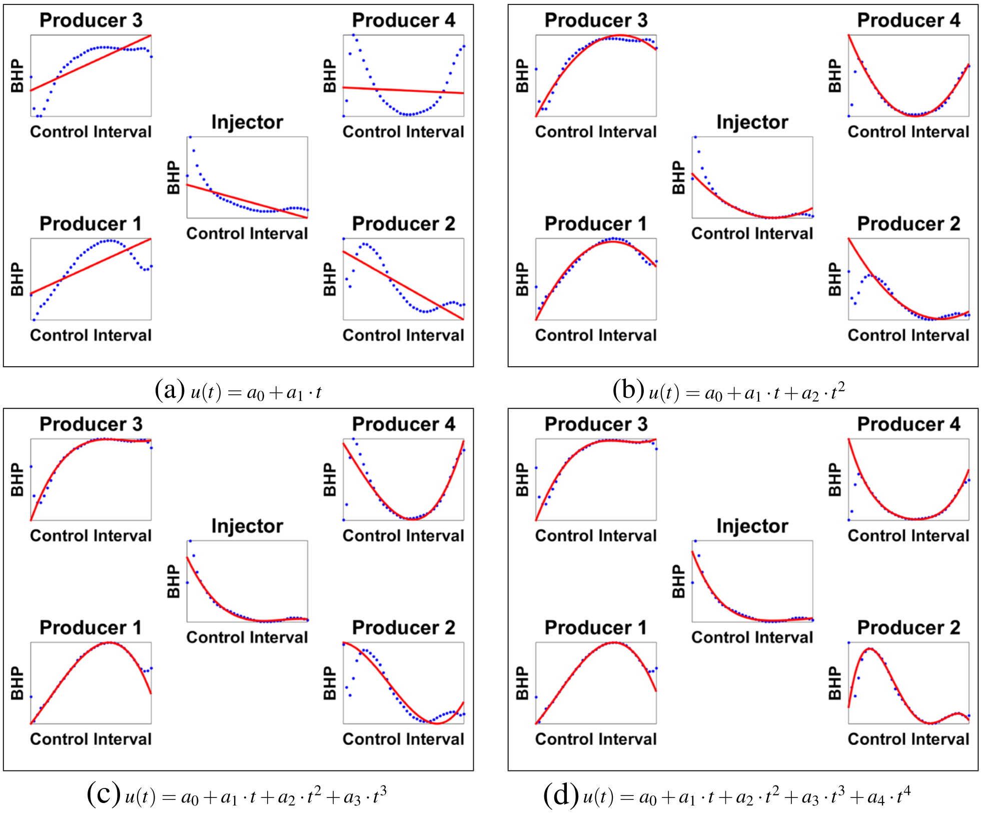


Fig. 2 Best solution found by (*left*) local (gradient-based) search and (*right*) global (stochastic) search



(a) $u(t) = a_0 + a_1 \cdot t$

(b) $u(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2$

(c) $u(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2 + a_3 \cdot t^3$

(d) $u(t) = a_0 + a_1 \cdot t + a_2 \cdot t^2 + a_3 \cdot t^3 + a_4 \cdot t^4$

Fig. 3 Polynomials functions (*red curves*) fitted to adjoint gradient-based solution (*blue dots*)

example, see Producer 2 trajectory). However, it has been repeatedly argued [32, 51, 55] that the objective function hypersurface is believed to have a high plateau where there might be an infinite number of control realizations that give similar field life-cycle NPV. Hence, it might be possible (but there is no guarantee) to find smaller degree control polynomials amongst the infinite options lies in the objective function's high plateau.

The ICM, on the other hand, is an alternative to obtain a polynomial curve approximation by performing piecewise polynomial interpolation between selected points in the original solution space ($\mathbb{U}$). This might be an attractive option, as the bounds on the variables are well defined, as opposed to the FCM where the boundary should be chosen heuristically.

In the next two sections, we present the new optimization formulation for the FCM, followed by the formulation for the ICM.

5 Polynomial approximation

5.1 Function control method

In order to achieve dimensionality reduction, while generating smooth and achievable control, let us assign a polynomial function to describe the control for each well. Equation 7 shows a general form of a polynomial well-control function:

$$\tilde{\mathbf{u}}(a_0 \cdots a_n, \tilde{t}) = a_0 + a_1 \cdot \tilde{t} + a_2 \cdot \tilde{t}^2 + \cdots + a_n \cdot \tilde{t}^n, \qquad (7)$$

$$\tilde{t} = 2 \cdot \frac{t - t_0}{t_{final} - t_0} - 1 \qquad (8)$$

$$\tilde{\mathbf{u}} = 2 \cdot \frac{\mathbf{u} - u_{lb}}{u_{ub} - u_{lb}} - 1 \qquad (9)$$

where, $\mathbf{u}$ again describes the control values (in units of either flow rate or pressure) at each time interval, a_0 through a_n represent the polynomial coefficients, t represents the time passed from production initiation, and $\tilde{\mathbf{u}}$ and $\tilde{t}$ are the normalized (and shifted) control and time, respectively.

As mentioned before, polynomial function control method for production optimization was proposed in [2]. In this contribution, the author defined the fractional time as the ratio of the length of time counted from the beginning of operating the well to the total life of the field. Thus, the independent variable scaled from zero to one. As shown in Eqs. 8 and 9, here, we took a different approach by normalizing both the operation time and the control values and shifting their scale between −1 and 1. This shifting enables to better capture multi-modality polynomial behaviors from both sides of the vertical and horizontal axes.

In addition, the work in [2] bounded the variable to $[-1/n, 1/n]$ where n is the polynomial degree, in order to restrict the polynomial to stay within the original control bounds. This restriction allowed the control to be on the boundary at only one point in time (at the start or at the end of the simulation).

However, following the already mentioned works by Sudaryanto and Yortsos [48, 49] and Zandvliet et al. [53], which show bang-bang or bang-singular solutions, the optimal control trajectory can be found on the boundary in several intervals. Therefore, we allow the polynomial to take values out of the $[-1, 1]x[-1, 1]$ normalized-time-controls box, by setting the hard bounds on the polynomial coefficients to be also $[-1, 1]$. In the case where the polynomial is indeed above (or below) the hard bounds, we set the control in the simulation schedule to be on the nearest boundary.

Similarly to the discussion in Section 3.2, let $j = 1, 2 \ldots N_w$ denote the well number, and $p = 1, 2 \ldots n + 1$ denote the polynomial coefficient index. Let us now define a new well-control variables vector as $\bar{a} = \left(a_{1,1}, a_{1,2}, \ldots, a_{1,n+1}, \ldots, a_{N_w,n+1}\right)$, where according to Eq. 7, $n = \textit{polynomial degree}$. In this formulation, the polynomial coefficients, $a_{j,p}$, are now the optimization decision variables.

Note that we distinguish between vectors that reside in the original space $\mathbb{U}$ and those which reside in the reduced space $\mathbb{P}$ (the polynomial space). Thus, for the sake of notational clarity, we write $\bar{x}$ for elements in the space $\mathbb{R}^{(n+1)N_w}$ and $\mathbf{x}$ for elements in $\mathbb{R}^{N_t N_w}$.

By inserting Eq. 7 as additional constraints into Eq. 1 and adding hard bounds constraints for the polynomial coefficients, we arrived at the new problem formulation, denoted here as $P2$, as shown in Eq. 10.

$$P2 : \max_{\bar{a}} \text{NPV}\left(\mathbf{u}(\bar{a}, t)\right) \text{ subject to } \begin{cases} g(\mathbf{u}) = 0 \\ \mathbf{u}(\bar{a}, t) \in \mathbb{U} \\ \bar{a} \in \mathbb{P} \end{cases}, \tag{10}$$

where $\mathbb{P} = \{\bar{a} \in \mathbb{R}^{(n+1)N_w} : \bar{a}_{lb} \leq \bar{a} \leq \bar{a}_{ub}\}$. With this formulation, the number of decision variables is independent of both optimization horizon time and the number of control intervals. More precisely, we reduce the solution space dimension by a factor of $N_t/(n + 1)$ as infered from Eq. 11.

$$dim(\mathbb{P}) = N_w \cdot (n + 1), \tag{11}$$

5.1.1 Smoothness constraints

Although only hard bounds constraints were considered in this work, we mention here how one can impose smoothness constraints by adding additional constraints to P2 (Eq. 10). Taking the derivative of the polynomial, as shown in Eq. 12, will enable determining the maximum allowable change in a control in every control interval. This facilitates controlling the smoothness of the obtained solution.

$$\frac{d\mathbf{u}(\bar{a}, t)}{dt} = \sum_{i=1}^{n} i \cdot a_i \cdot t^{i-1}. \tag{12}$$

Note that for a linear case, the polynomial slope is not time-dependent. Thus, by formulating FCM as a linear control, we might limit the maximum allowable control change by only setting a hard bound on the coefficient a_1 for each well. On contrary, for FCM with higher polynomial degrees, the constraints are time-dependent and should be introduced as additional inequality constraints in problem $P2$. A quadratic control requires imposing linear constraints, and cubic and higher degree PCMs require non-linear constraints. However, in this work, we imposed only hard bound constraints, as we want to test the smoothness of the obtained control trajectories without adding linear and non-linear constraints (if using Eq. 12).

5.2 Interpolation control method

The keen reader might notice that the hard bounds for the functions coefficients in FCM are determined heuristically. Thus, one can avoid the hurdle of debating which hard bounds to impose on the coefficients by using piecewise polynomial interpolation. In this approach, which we call the interpolation control method (ICM), the decision variables are control values at selected equidistant points in time. Those variables have the same bounds as the original optimization variables. Then, the control is determined based on a piecewise polynomial interpolation between these points.

Among different piecewise polynomial interpolation methods, we chose cubic spline interpolation, which requires continuity up to the second derivatives at the interpolation points (also called knots) and thus yields highly smooth curves. We used “not-a-knot” cubic spline, which also requires third derivative continuity at the first and one before the last points [5, 17].

For n selected equidistant points per each well j, we shall denote each selected point as $z_{j,p}$, where again $p = 1, 2 \ldots n$. Let $\bar{z} = (z_{j,1}, z_{j,2}, \ldots, z_{j,n+1})$, and define $P3$ as

$$P3 : \max_{\bar{z}} \text{NPV}\left(\mathbf{u}(\bar{z}, t)\right) \text{ subject to } \begin{cases} g(\mathbf{u}) = 0 \\ \mathbf{u}(\bar{z}, t) \in \mathbb{U} \\ \bar{z} \in \mathbb{Z} \end{cases}, \tag{13}$$

where $\mathbb{Z} = \{\bar{z} \in \mathbb{R}^{(n+1)N_w} : \bar{z}_{lb} \leq \bar{z} \leq \bar{z}_{ub}\}$.

At this point, it should be mentioned that using either FCM or ICM, the size of the optimization problem remains the same even if the simulation control time-step is decreased. This might enable obtaining a more detailed control scheme. However, decreasing the simulation control time-step might increase the computational time of a

Table 1 Variables hard bounds in both example 1 and example 2

Control method	Control variables	Units	Injectors lower bound	Injectors upper bound	Producers lower bound	Producers upper bound
FCM	$c_{i,j}$	[−]	−1	1	−1	1
ICM	$\mathbf{z_j}$	$psia$	2,200	3,000	500	2,100
Free control	$a_{i,j}$	$psia$	2,200	3,000	500	2,100

single forward model. Unlike several well-control parameterization approaches presented in the literature, the number of FCM and ICM parameters stay constant throughout the optimization process, which facilitates a simpler implementation. Next, we show two application examples.

6 Computational results

We tested our proposed method on two cases with increasing complexity. First, we solved the same two-dimensional problem as in the motivating example. In the second example, we solved the realistic UNISIM benchmark [25] based on the Namorado reservoir from the Campos Basin offshore Brazil [4], where we incorporated a total of 25 wells as implemented in the work shown in [41].

In order to test our method, we performed an optimization procedure for different polynomial degrees and for different number of interpolation points. We compared the FCM and ICM results with traditional optimization methods, employing both gradient-based and gradient-free approaches.

6.1 Example 1: FCM and ICM for two-dimensional reservoir model

In the first example, we used the same two-dimensional problem settings as for the earlier motivating example (see Fig. 1). We formulated the optimization problems as shown in Eqs. 10 and 13. We ran the optimization for different polynomial controls with degrees ranging from one to four (linear, quadratic, cubic, and quartic polynomials) and equivalently, for two, three, four, and five interpolation points. We compared the performances of different approaches along with a comparison with the traditional control formulation. We call this traditional formulation as

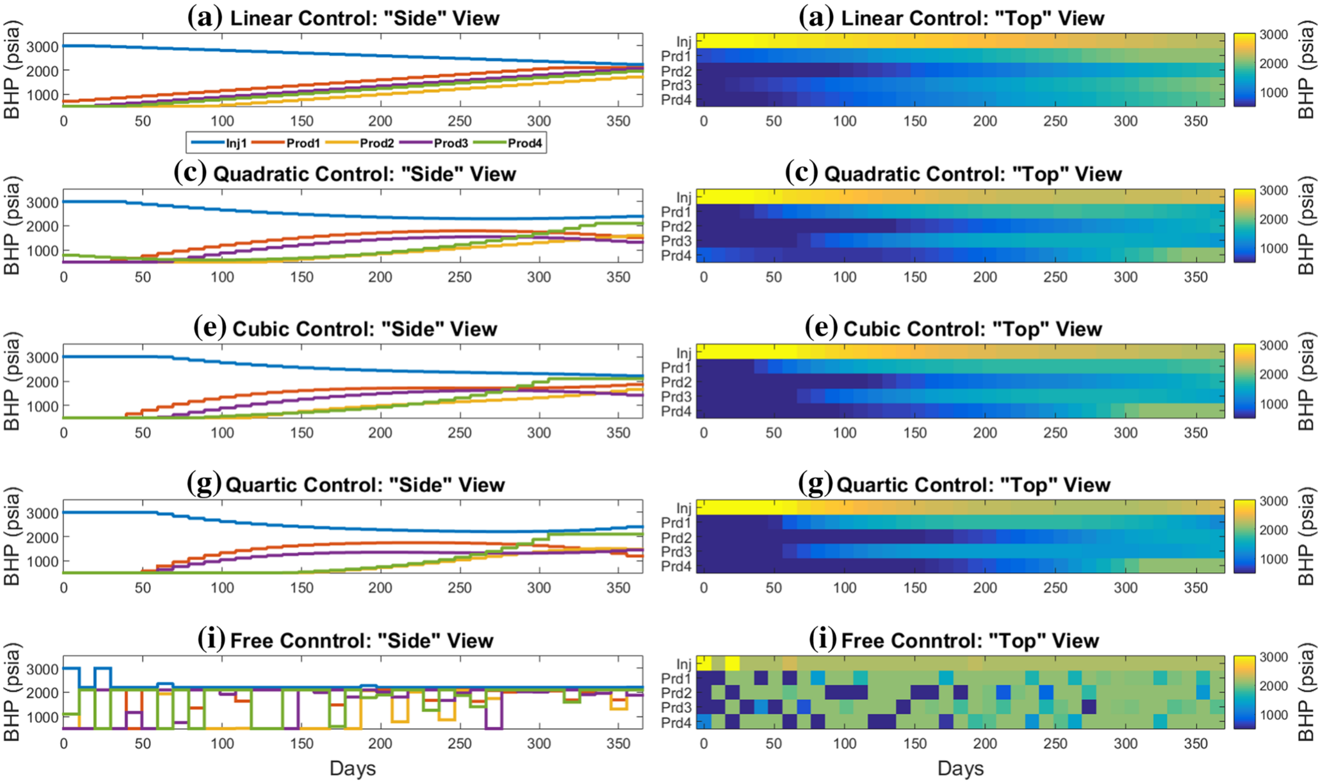


Fig. 4 Example 1: FCM and free controls–results of the RUN WITH THE best objective function found among all 10 runs

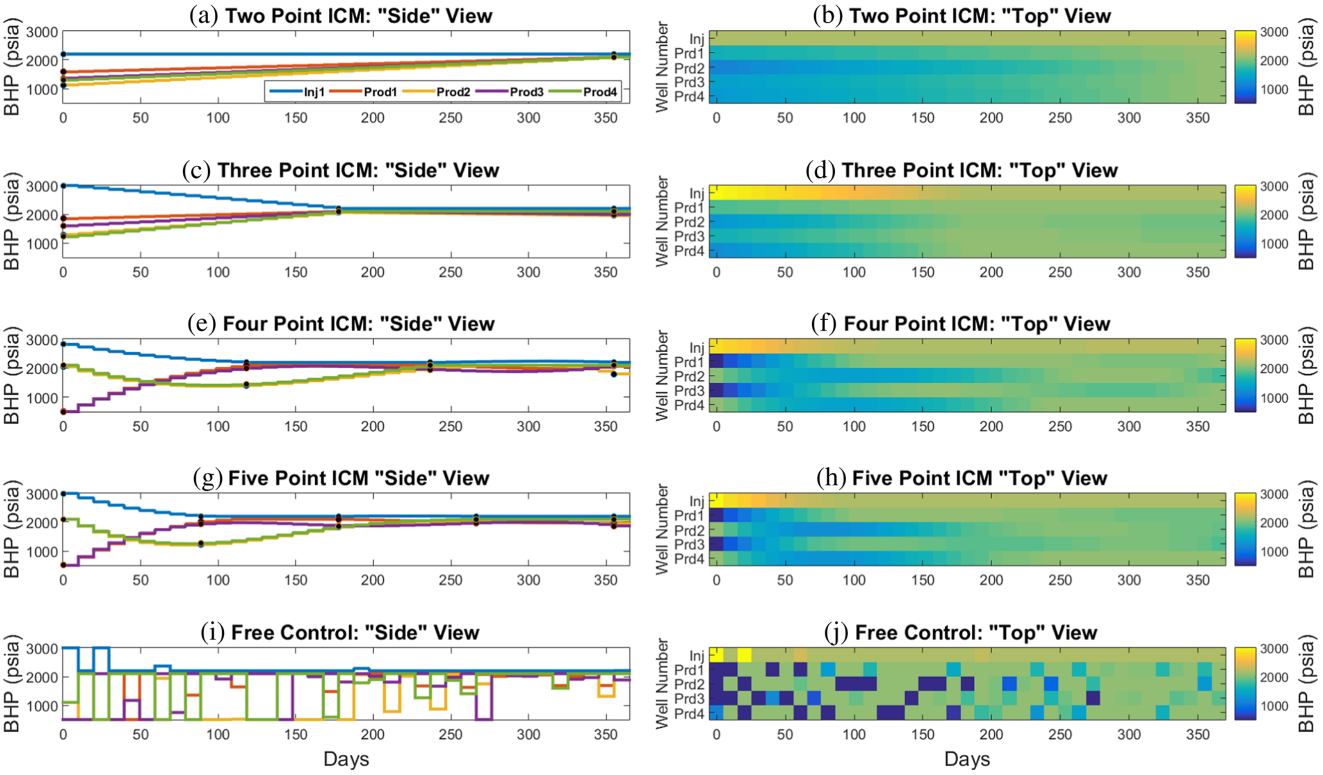
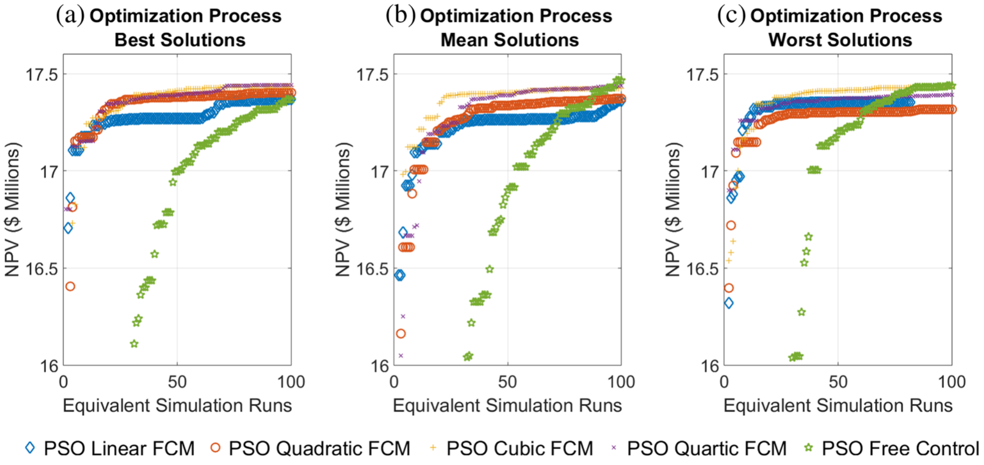


Fig. 5 ICM and free controls–results of the RUN WITH THE best objective function found among all 10 runs

"Free" control, since the control trajectory is free to have any irregular shape, without being confined to follow any polynomial trajectory. In Table 2, we present the control set cardinality for each method, as followed from Eqs. 6 and 11, and dimensionality reduction as a ratio between the free method and the ICMs and FCMs. We can see that the problem becomes smaller as the polynomial degree or the number of interpolation points decrease. We show in Table 1 the box constraints used in this work, which are applied for both example 1 and next in example 2. For all cases, we used PSO as the global search algorithm. Since PSO is a stochastic optimizer, we ran the simulation multiple times in order to obtain statistically meaningful interpretations regarding the robustness of the compared methods. Each case was run 10 times, each with a different seed random generator. Note that the obtained solution of the gradient-based method is already provided in Fig. 2. For all gradient-free methods, Figs. 4 and 5 show the controls of the best solution found among the 10 runs. We provide the results in a "side" view such that the polynomial shape can be observed. We also provide a "top" view for the reader who is more comfortable to this sort of visualization. In both plots, in addition to the FCM and ICM, we show also the free control for a comparison. Note to the bang-bang solution of the free method

Fig. 6 Example 1: FCM and free controls—first 100 iterations of the best, worst, and closest to mean runs out of the 10 runs



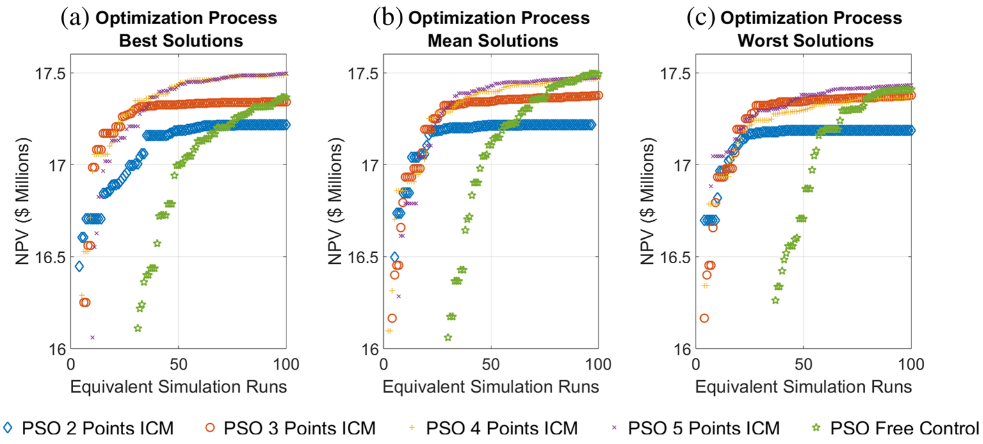


Fig. 7 Example 1: ICM and free controls—first 100 iterations of the best, worst, and closest to mean runs out of the 10 runs

(though not pure bang-bang, as some values are not exactly on the boundaries). On the other hand, in the FCM and ICM solutions, we observe smooth polynomial behavior between the bounds. Whenever the polynomials went out of bounds, the actual control was set to the bound value. We can see that even without imposing Eq. 12 as a constraint, the solutions are very smooth and amendable for deployment in the field. It seems that due to high water cut all solutions drive the controls to gradually minimize the draw-down as can be inferred from the decreasing distance between the injector and producers curves. Figures 6 and 7 show the optimization progress along the first 100 iterations for the best, worst, and closest to the mean of all 10 runs. Note to the rapid objective function (NPV) improvement in the FCM and ICM comparing to the free method. This fast improvement is due to efficient search in a small dimension space, which will have a major impact for larger optimization problems, as we will see in the next example (Table 2).

Table 3 shows the average (mean), best, worst, and standard deviations results of all 10 runs for each method. The results are given in terms of the obtained NPV, the number of simulations and the number of equivalent simulation runs (equal to one parallelized PSO iteration). We can see that in addition to dimensionality reduction, the flexibility and multi-modality of the function is also essential, as inferred from the results shown in Table 3. Note that in all categories (mean, best, and worst) among the ICMs and FCMs, a higher polynomial degree (or number of interpolation points) corresponds to a superior NPV at convergence, along with larger number of iterations and simulation runs. Thus, we observe a trade-off between the rate of improvement and the quality of the solution.

Table 2 Control set cardinality and solution space dimensionality reduction for example 1

Control method	Control set cardinality	Dimensionality reduction ratio
Linear		
Two points	10	18.50
Quadratic		
Three points	15	12.33
Cubic		
Four points	20	9.25
Quartic		
Five points	25	7.40
Free	185	1.00

6.2 Example 2: three-dimensional UNISIM benchmark reservoir model

Under the curse of dimensionality paradigm, the idea is that the larger the problem, the computational cost should be exponentially increased, and thus, the effects of a dimensionality reduction should be more evident. Thus, we test ICM and FCM on a much larger system, namely the UNISIM-I benchmark.

The original high resolution reference benchmark (with approximately 3.5 million active grid blocks) is based (with some modifications) on the structural, petrophysical, and facies model of the Namorado oil field, located in Campos Basin, Brazil [25]. Avansi and Schiozer [1] upscaled the reference model into a medium-scale reservoir in order to make the model applicable to reservoir management optimization procedures that require many simulation calls. In our work, we used this upscaled version of UNSIM-I-D, which consists of 20 layers, a 100 × 100 × 8 m grid cell resolution, and about 37,000 active grid blocks (see [50] for further model specifications).

Figure 8a shows the permeability distribution, in a log_{10} scale, of the UNISM-I-D model. In addition to the four vertical producing wells provided as part of the benchmark, we incorporated additional 10 horizontal producing wells and 11 horizontal injection wells, as it was done in [41].

Table 3 Example 1: Results of 10 optimization runs, (function tolerance: 10^{-6} for 20 consecutive stall iterations)

	Best		Mean		Worst		Standard deviation	
Control method	NPV $ millions	Total sim. (equiv. sim.) runs	NPV $ millions	Total sim. (equiv. sim.) runs	NPV $ millions	Total sim. (equiv. sim.) runs	NPV $ thousands	Total sim. (equiv. sim.) runs
Linear	17.37	3720 (124)	17.36	3894 (129.8)	17.35	2550 (85)	6.64	1086.5 (36.22)
Quadratic	17.42	12,120 (404)	17.40	11,190 (373.0)	17.34	10,128 (337.60)	3.57	798.77 (26.63)
Cubic	17.47	12,480 (416)	17.45	13,800 (460.0)	17.43	7560 (252)	14.20	4529.40 (150.98)
Quartic	17.51	42,150 (1,405)	17.51	29,244 (974)	17.50	28,020 (934)	7.07	6632.96 (221.10)
Two points	17.22	3270 (109)	17.21	3513 (171.1)	17.19	3360 (112)	9.48	1127.07 (37.57)
Three points	17.39	9720 (324)	17.39	7785 (259.5)	17.39	7440 (248)	0.62	1226.48 (40.88)
Four points	17.49	12,420 (414)	17.48	15,399 (513.3)	17.4	9720 (324)	27.11	5705.12 (190.17)
Five points	17.53	14,160 (472)	17.52	13,404 (446.8)	17.51	12,180 (406)	10.41	2429.56 (80.99)
Free	17.74	59,040 (1,968)	17.67	38,295 (1,276.5)	17.62	33,660 (1,222)	38.69	9264 (308.79)

That is, we considered in total 25 wells for the production optimization procedure (see Fig. 8b).

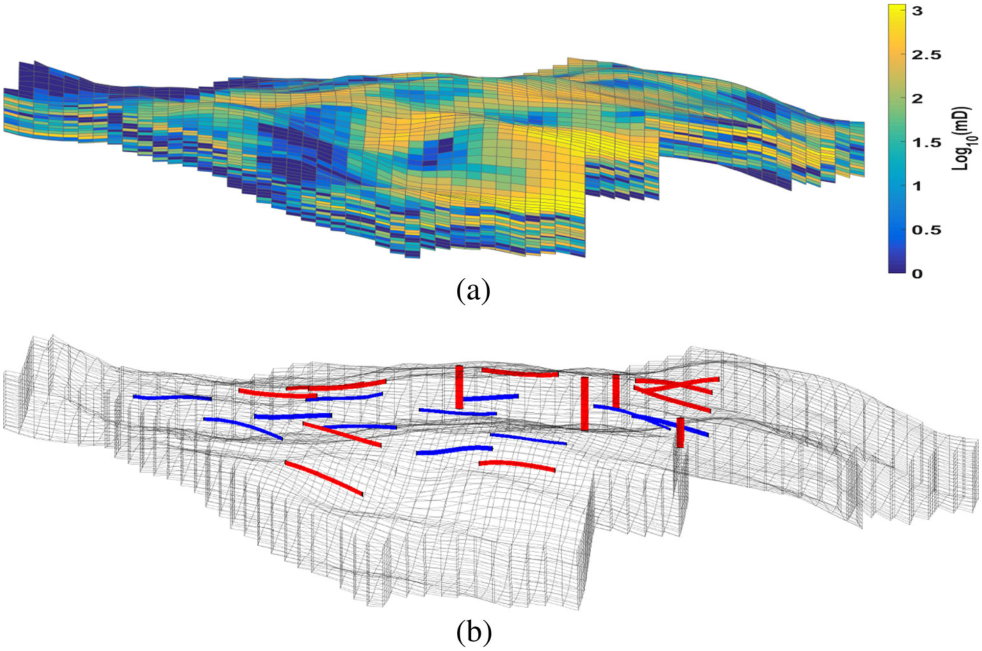
We set the simulation horizon time to be 5 years and adjusted the controls each month. Hence, the control set per each well included 60 decision (design) variables, and the total cardinality for all 25 wells amounted to 1500 variables for problem P1 as dictated by Eq. 6.

As in example 1, we solved the problem with both P1, P2, and P3 formulations (Eqs. 1, 10, and 13, respectively). Due to relatively long simulation time of this reservoir model, we limit the number of methods tested. We chose quadratic and cubic polynomials, and their dimension-equivalent methods, the three and four points interpolation. Following the results of example 1, these methods yield a combination of fast convergence and good quality solutions.

Following again from the results of the first example, which show excessive number of simulation runs without significant improvement, we loosen the convergence criteria for all methods solved by PSO algorithm from 20 stalled iterations below a relative objective function change of 10^{-6} in example 1, to 12 stalled iterations with a 10^{-4} threshold (see Appendix for more details). In Table 4, we present the obtained control set cardinality for each method, as followed from Eqs. 6 and 11, and the dimensionality reduction ratio.

Due to the large computational cost, we conducted the optimization for each one of the PSO implementation only three times, each with a different seed random generator. These mere three optimization repetitions were sufficient to demonstrate the increasing effectiveness (comparing to free

Fig. 8 UNISIM Benchmark: **a** Permeability distribution. **b** Completed wells intervals of the four vertical and 10 horizontal producing wells (in *red*) and 11 horizontal injection wells (in *blue*) used for production optimization (following the work in [41])



(a)

(b)

Table 4 Control set cardinality and solution space dimensionality reduction for example 2

Control method	Control set cardinality	Dimensionality reduction ratio
Quadratic		
Three points	75	20
Cubic		
Four points	100	15
Free	1500	1

PSO) of the ICM and FCM when solving a bigger problem. Table 5 shows the average (mean), best, worst, and standard deviations results of the three runs of each tested method. As predicted, the improvement in computation effort is considerably more profound in a larger scale problem. While the Free method used an excessive number of simulation runs and did not converge within the optimization time limitation (36 h), the FCMs and ICMs quickly converged to a considerable higher value.

Although gradient-based methods might converge to a local optimum, they have the obvious advantage of faster convergence and the ability to cope with considerable large number of decision variables, if implemented wisely. We obtained the gradients using MRST adjoint implementation, and compared different optimization methods provided by the built-in Matlab *fmincon* function [38]. Based on preliminary tests, we performed to the different algorithms implemented in *fmincon*, we found three that perform better than the others. These three constrained optimization algorithms are active-set, interior point with the BFGS inverse Hessian quasi-Newton approximation (IP-BFGS), and interior point with limited memory BFGS inverse Hessian quasi-Newton approximation (IP-LBFGS).

6.3 Example 2: optimization process comparison and discussion

Figure 9 shows three comparison of the processes along the optimization iterations. First, Fig. 9a compares the best runs of the gradient-free methods, where we can see how a relatively high dimensional problem (comparing to example 1) impacts the optimization process. Note that the objective functions of ICMs and FCMs increase and converge much faster than the Free PSO. All ICMs and FCMs converged within less than 12 h, while the free PSO did not converge and terminated after 36 h with an objective function values almost 10% lower than the ICMs and FCMs results.

Figure 9b compares the gradient-based methods. We ran all three methods once from a single starting point, with a maximum of 1000 iterations. Note that IP-LBFGS performed far better than the other gradient-based methods (converging faster to a significant higher NPV). These superior performances of the L-BFGS algorithm is consistent with previous results for large-scale optimization problems (see [39, 54]).

Note also from Fig. 9b that IP-BFGS did not converge and the optimization ended when reached to the maximum allowable number of iterations. On the other hand, the active-set method had more than 1000 simulation runs, as each iteration might consist of several simulation runs when a quadratic sub-problem is solved. Active-set converged after 736 iterations (with 1570 simulation runs), when the magnitude of directional derivative in search direction was less than 2×10^{-6} and maximum constraint violation was less than 10^{-6}. Contrarily, IP-LBFGS converged after only 644 iterations (with 716 simulation runs). Similar to the active-set method, the IP-LBFGS had more simulation runs than iterations. The reason here is that the interior point algorithm might not accept a particular step, for example when an attempted step does not decrease the merit function (see [38]). Thus, the algorithm attempts a new step and

Table 5 Example 2: Results of 10 optimization runs, (function tolerance: 10^{-4} for 12 consecutive stall iterations)

	Best		Mean		Worst		Standard deviation	
Control method	NPV $ millions	Total sim. (equiv. sim.) runs	NPV $ millions	Total sim. (equiv. sim.) runs	NPV $ millions	Total sim. (equiv. sim.) runs	NPV $ thousands	Total sim. (equiv. sim.) runs
Quadratic	17.57	5220 (174)	17.47	4190 (139.67)	17.33	4410 (147)	122.16	1155.8 (38.53)
Cubic	17.67	4410 (147)	17.60	5140 (171.33)	17.48	2940 (98)	100.94	725.8 (24.19)
Three points	17.58	3510 (117)	17.53	3660 (122.00)	17.48	3750 (125)	50.04	130.77 (4.36)
Four points	17.68	4680 (156)	17.65	4830 (161.00)	17.62	4980 (166)	42.96	212.1 (7.1)
Free	15.94[a]	18,915 (630.5)	16.09[a]	18,960 (632)	15.80[a]	18,870 (629)	145.0	45.00[b] (1.5)[b]

[a] Optimization ended due to maximum allowable execution time: 36 h. All other values in the table were obtained within less than 12 h.

[b] Following [a], all optimization had approximately same number of iterations and thus low standard deviation.

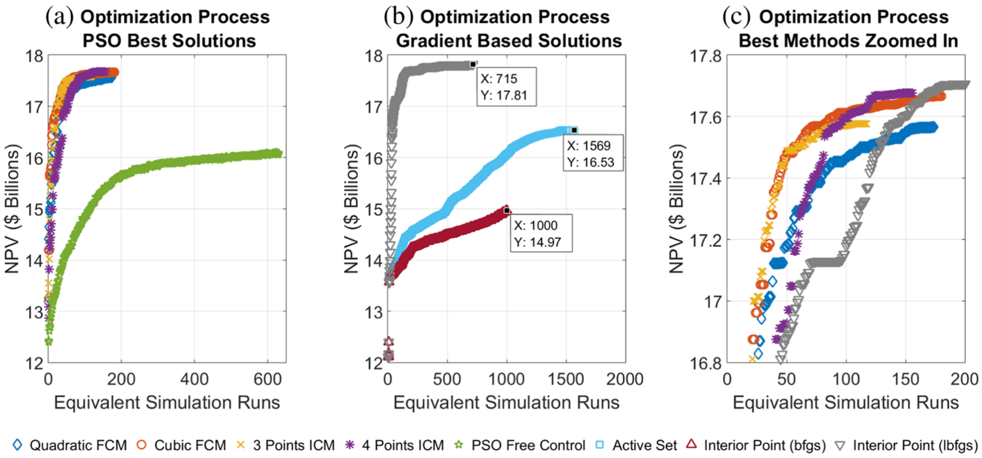


Fig. 9 Example 2: **a** Gradient-free methods best runs comparison. **b** Gradient-based methods comparison with final solutions. **c** Zoomed-in to the first 200 iterations of the ICMs, PCMs, and the IP-LBFGS methods

perform additional simulation. Comparing Fig. 9a, b, the FCMs and ICMs performed much better than the active-set and the IP-BFGS.

Figure 9c is a zoom-in to the first 200 iterations of the best methods performed: ICMs, PCMs, and the IP-LBFGS. Note that while the PSO-ICMs and PSO-FCMs grow faster than the IP-LBFGS initially, the IP-LBFGS gradually exceeds all other methods. Figure 10 shows this process as an instantaneous difference between each ICM or FCM method to the IP-LBFGS method.

Following the obtained results, some additional remarks are in order:

- If the objective function has indeed a flat plateau, we shall assume in this discussion that the optimal solution can be described by the IP-LBFGS solution. Its value as shown in Fig. 9b is $17.81 billions.
- Interestingly, each best run of the dimension-equivalent ICM and FCM converged to a similar objective function value, and thus to a similar distance from the optimal solution.
- As can be seen from Fig. 10, the difference from the optimal solution is rather small: 1.35 and 0.8% for the cases of three and four variables per well, respectively.
- Similar to the results of example 1, a higher degree polynomial or a higher number of interpolation points, leads to better objective function values as well as an increased number of iterations up to convergence (see mean results in Table 5).
- The FCMs and ICMs objective function values can improve and reach the optimal solution by adding more variables per well. This can be done iteratively by refining the controls with adding more interpolation points or polynomial coefficients.
- Tightening the PSO convergence criteria will improve the obtained results.

6.4 Example 2: control solutions comparison and discussion

Figures 11 and 12 show the best control results obtained by the FCMs, ICMs, free PSO, and the gradient-based methods. Again, we can see the erratic control behavior obtained by the free PSO (refer to Figs. 11e and 12e). Interestingly, the FCMs and ICMs exhibit smoother results than all gradient-based methods, even without imposing smoothness with Eq. 12. From Fig. 9, we can see that the two FCMs, two

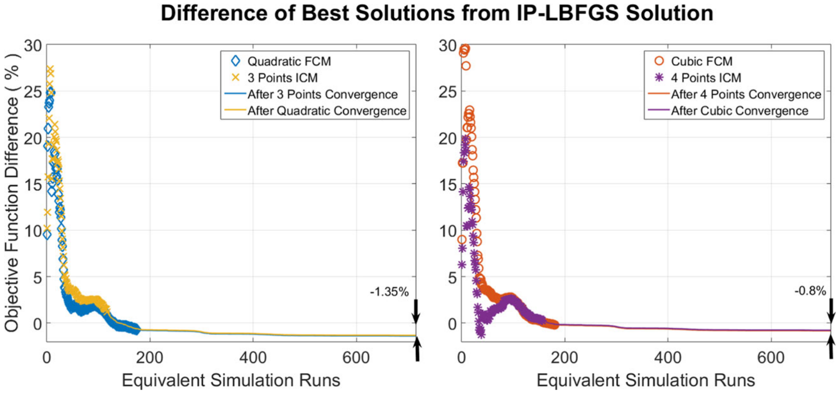


Fig. 10 Example 2: Percentage difference between the best solutions of FCMs and ICMs to IP-LBFGS. *Left*: Three variables per well. *Right*: Four variables per well. *Markers and lines* are differences before and after convergence, respectively. Final differences after IP-LBFGS convergence are shown

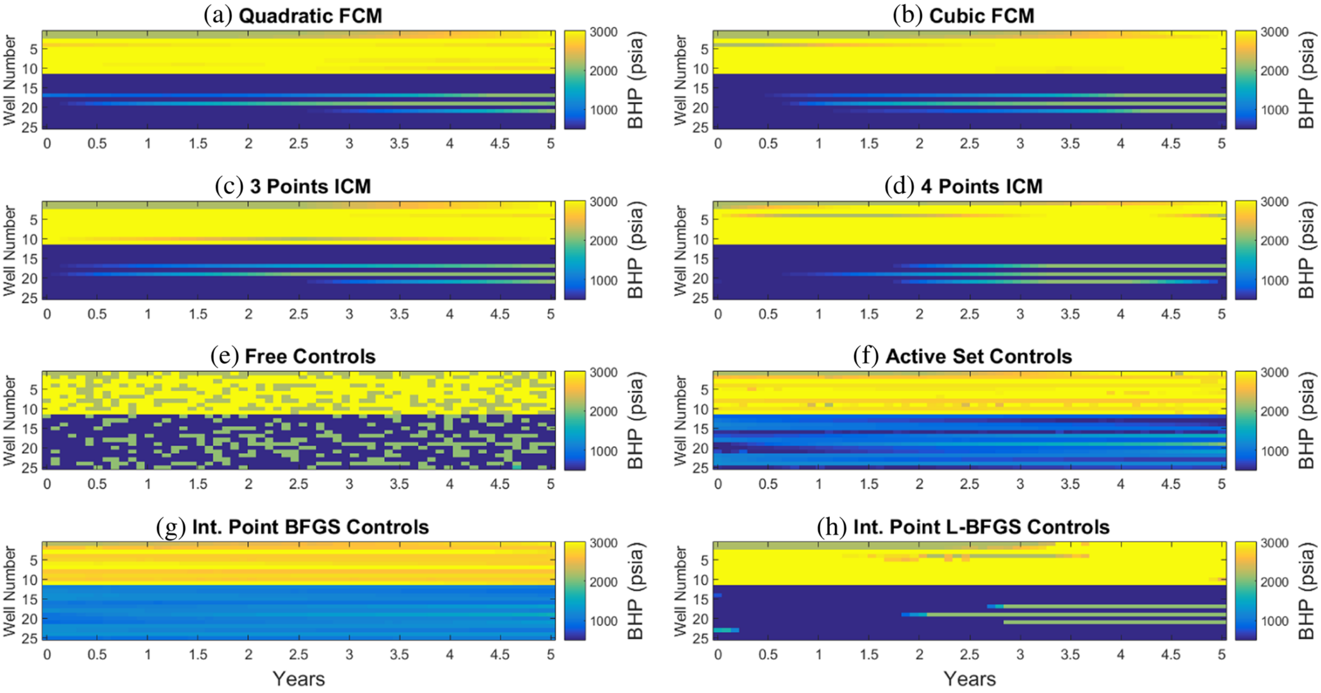


Fig. 11 *Top* view: control results obtained by the ICMs, FCMs , free PSO, and the gradient-based methods

ICMs and the IP-LBFGS obtained the best objective function values. From Fig. 11a–d, h, we observe that these five also share similar control patterns, where most wells were controlled by the maximum (injectors) or minimum (producers) allowable BHP. In all five, there are three producers wells (numbered 17, 19, and 21) in which their pressure is increase from minimum to maximum allowable BHP during the simulated 5 years. This is consistent with the problem of delaying the water breakthrough and minimizing water cut (as can be seen in Fig. 13). Note to the pure bang-bang

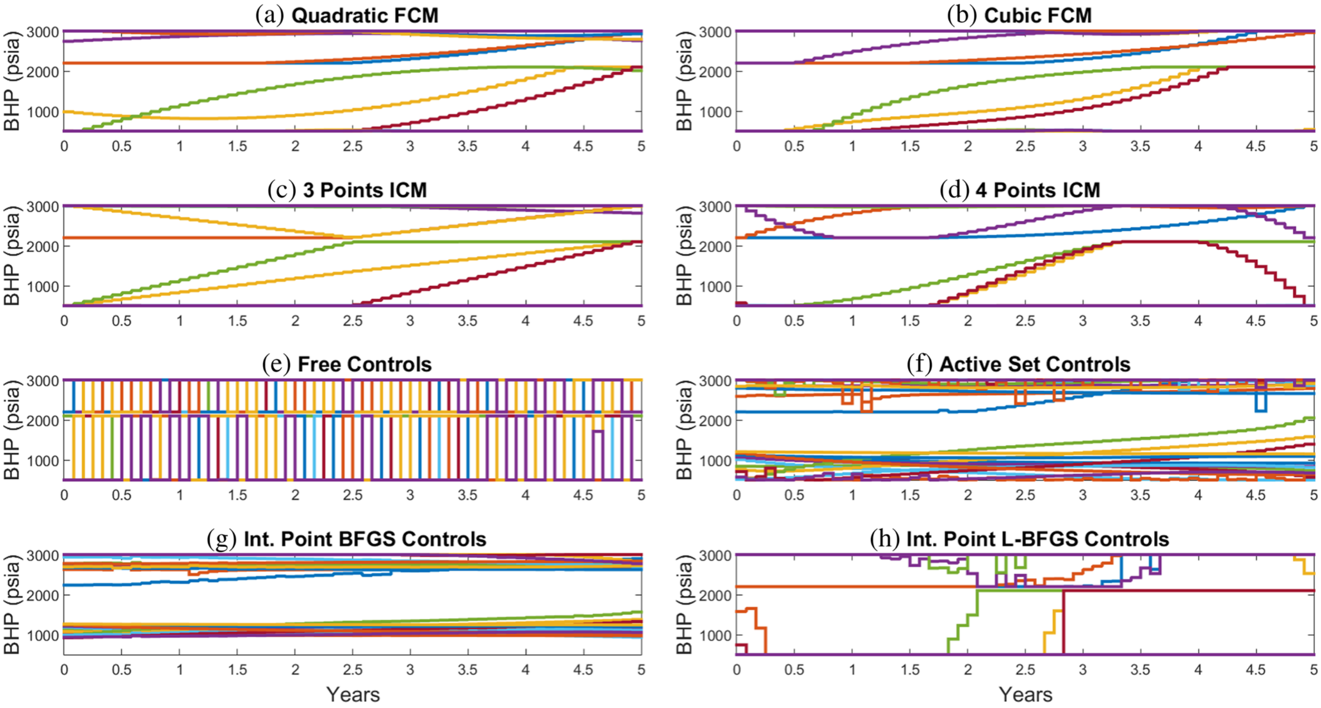


Fig. 12 *Side* view: control results obtained by the ICMs, FCMs , free PSO, and the gradient-based methods

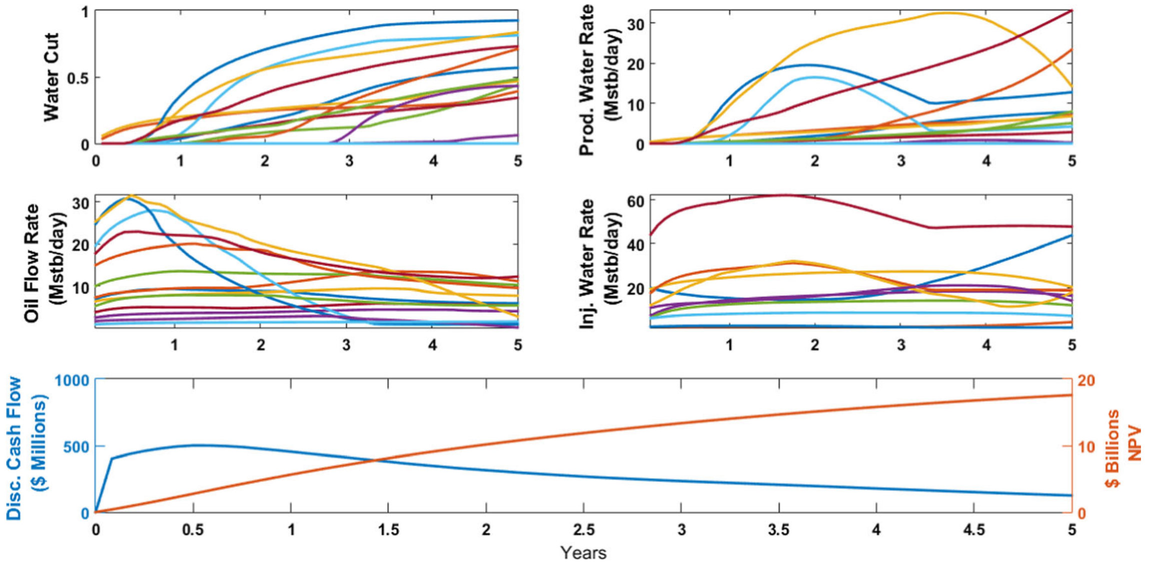


Fig. 13 Best run of four points interpolation for example 2—water cut, producers' oil and water rates, injector water rates, and economics measures

solution of well 21, and to the short singular arc of wells 17 and 19 in the solution obtained by IP-LBFGS. Also from Fig. 13, we observed that the project has not yet reached the economic limit. This proves the need for setting the terminal time as an additional optimization parameter as previously suggested in [53]. Also, note that with ICM or FCM parametrizations, adding a terminal time as another decision variable would only increase the dimensionality of the problem by one. This might, however, require some adjustments to the parallel computing work-flow, as the simulation durations might vary significantly.

6.5 Example 2: average simulation duration comparison

As previously done by others [30, 45], we use the term "equivalent simulation runs" as a performance comparison measure. We note here that this comparison criterion is not "one to one" computational time convertible. That is, one equivalent simulation run may not have the same duration amongst all methods. Since each PSO simulation (Free, ICM, or FCM) was executed in parallel on a separate node, we observed some increase in average simulation time, due to communication between nodes. On the other hand, in the gradient-based methods, which were ran on a single node sequentially, most of the equivalent simulation runs include one adjoint simulation run in addition to the forward simulation run. Due to the high control resolution, each adjoint simulation run consumed a considerable amount of time, about 80% of one forward simulation run time. Thus, the average simulation duration of the gradient-based and gradient-free methods are rather similar, as can be observed from Table 6.

Table 6 Average simulation duration for each method

Method	Average sim. duration (min.)
Act. Set	3.19
Four points PSO	3.37
Free PSO	3.42
Three points PSO	3.49
Cubic PSO	3.61
Quad PSO	3.83
IP-BFGS	3.91
IP-LBFGS	4.01

7 Conclusions and final remarks

In this paper, we provide two parametrization procedures for waterflooding production optimization problems by projecting the original infinite dimensional controls space into a polynomial subspace. The goal of these parametrization procedures is twofold: acceleration of a gradient-free optimization process and leading the controls to gain smooth solutions, which might be required from a production engineering point of view. We named our polynomial approximation approaches as Functional Control Method (FCM) and Interpolation Control Method (ICM). In the first method, the FCM, the (normalized) controls of each well are represented by a polynomial function with a normalized time as its variable. In the second method, the ICM, piecewise cubic polynomials are obtained by a "not-a-knot" smooth spline interpolation between equidistant points in time. The optimization variables are either the polynomial coefficients, or the values sampled from the original space (BHP in this paper) at the selected points for interpolation.

We stressed the importance that any polynomial parametrization should be able to produce bang-singular solutions. Following this discussion, we showed how we improved previous FCM formulation shown in the literature by allowing the polynomial to take values out of the actual bounds, where in these cases the control values are set to the bound. We also show different normalization procedure, which better capture the polynomial multi-modality behavior.

We tested our proposed methods on two examples with increasing complexity: a two-dimensional synthetic channelized reservoir and the realistic three-dimensional UNISIM benchmark. In the first example, we considered a five spot pattern where we have extensively tested four different FCMs (and ICMs): linear (two points), quadratic (three points), cubic (four points), and quartic (five points). Furthermore, in a second three-dimensional example, we imposed horizontal and vertical wells (25 in total) where we compared two FCMs (and two ICMs): quadratic (three points) and cubic (four points). We used particle swarm optimization as the gradient-free optimizer in a high performance computing environment.

In both examples, we showed that ICM outperforms FCM. We attributed this behavior to the well-defined bounds on the decision variables in the ICM when compared to the FCM. We also observed that the higher the polynomial degree or the number of interpolation points, the higher the obtained objective function value, along with increasing number of iterations up to convergence. We showed that as the problem scale increases, the contribution of FCM or ICM is greater. In the larger problem, the traditional formulation solved by PSO did not converge within maximum allowed CPU time, while our approach did converge to a considerably higher NPV within significantly fewer iterations. When comparing to gradient-based method, the FCMs and ICMs performed better than the active-set and the IP-BFGS. While the ICMs and FCMs grow initially faster than the IP-LBFGS, the IP-LBFGS gradually exceeded all other methods. The final difference from the best solution found by IP-LBFGS was in the range of 0.8–1.35%. We further discussed that testing ICM or FCM with higher number of variables per well and tightening the convergence criteria might close this gap, which can also open some scope for future work. As we stated, the future work might include also the terminal time as the only additional optimization variable.

Another avenue for future work is whether the solution space has a unique global solution or is the objective function has a high plateau with many solutions. This can be done for example with multi-start local optimization approaches. Moreover, other optimization methods to find the optimal function coefficients or interpolation points should be considered. Specifically, deterministic global optimization methods will be considered, which are expected to be more consistent in finding the globally optimal function coefficients.

Though ICM outperforms FCM in the experiments shown in this paper, this might not always be the case. Other functions than natural polynomials might be used and lead to better performances. Specifically, our group currently works with orthogonal polynomial basis functions, which seems to be a promising future research avenue.

We conclude that our method might serve as a handy tool for "black-box" and gradient-free optimization researchers who aim to optimize general field development (which entail more decision variables such as well-placement). When solving solely the optimal control problem, our methods can be useful in the absence of an adjoint model. If the adjoint model is available, our model can serve as a global search phase before switching into efficient local optimizer (such as IP-LBFGS used in this work) to refine the solution.

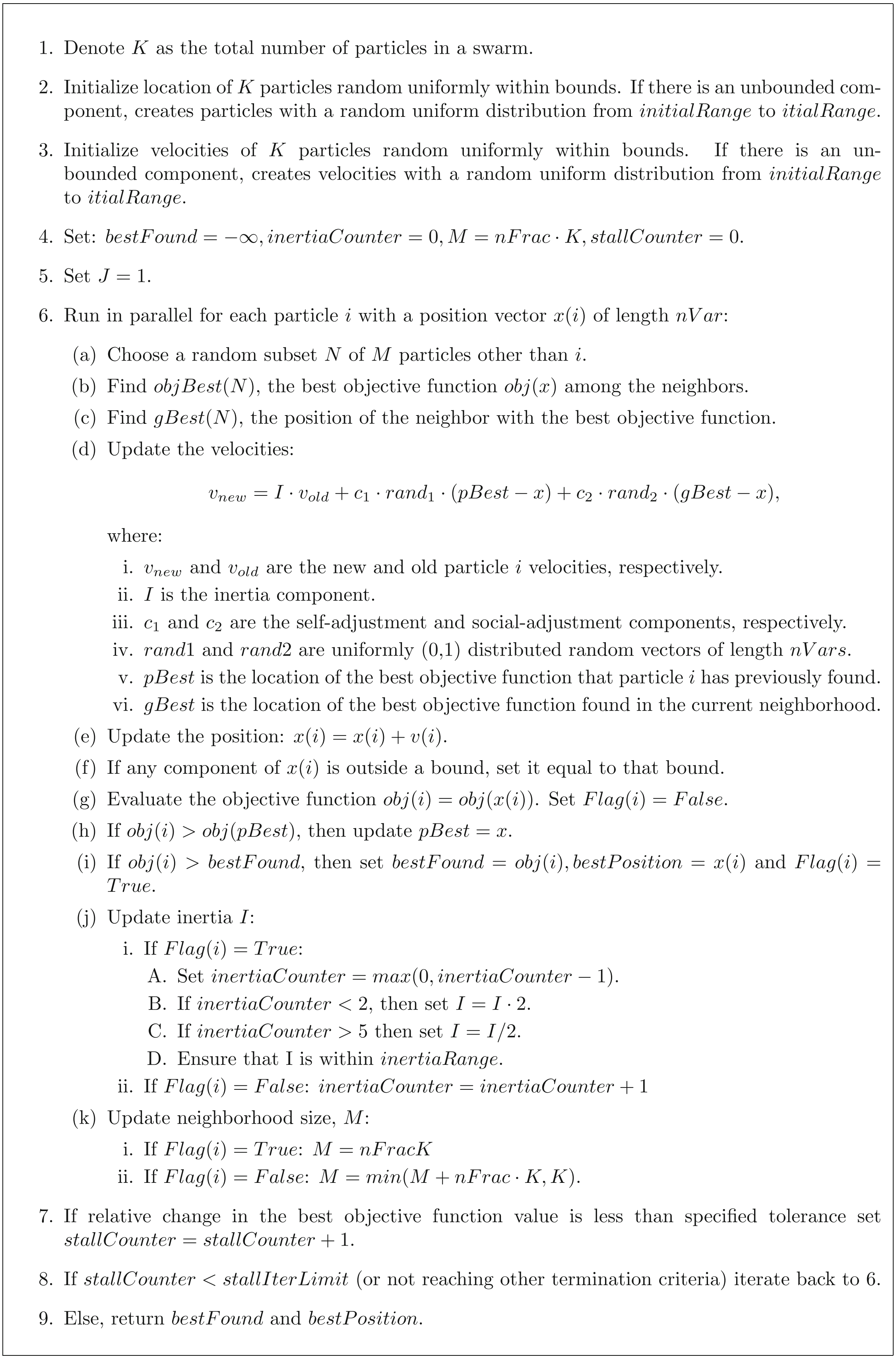
Acknowledgments E. Gildin and N. Sorek would like to thank the Foundation CMG for supporting this project through the FCMG chain at Texas A&M. The authors also acknowledge the valuable discussions with Hardikkumar Zalavadia and Alexander Tarakanov.

Appendix: Global search using PSO

Since we used particle swarm optimization (PSO) as our main optimizer to find the polynomial coefficients, here, we describe the PSO algorithm and its inputs in details. We used a PSO version with a variable neighborhood size and variable inertia magnitude. We followed, in general, the documentation showed in [37]. PSO is an optimization technique developed by Dr. Eberhart and Dr. Kennedy in 1995 [20]. This is a population based stochastic method, inspired by the social behavior of fish schooling or bird flocking.

In each iteration, each particle has a location (defined by its vector values) and a velocity. The velocity updating equation is the core of the PSO algorithm. The equation updates the new velocity, which carries each particle from one location (a feasible solution) to a neighboring location (a neighbor feasible solution). Figure 14 shows a PSO pseudo-code used in our work, for a maximization problem. Table 7 shows the specific inputs used in the two problems presented in this paper.

As can be seen in the pseudo-code, the algorithm combines meta-heuristic features with local search approach. A particle is "attracted" to the best solution found by him



```
1. Denote $K$ as the total number of particles in a swarm.
2. Initialize location of $K$ particles random uniformly within bounds. If there is an unbounded component, creates particles with a random uniform distribution from $initialRange$ to $itialRange$.
3. Initialize velocities of $K$ particles random uniformly within bounds. If there is an unbounded component, creates velocities with a random uniform distribution from $initialRange$ to $itialRange$.
4. Set: $bestFound = -\infty, inertiaCounter = 0, M = nFrac \cdot K, stallCounter = 0$.
5. Set $J = 1$.
6. Run in parallel for each particle $i$ with a position vector $x(i)$ of length $nVar$:
   (a) Choose a random subset $N$ of $M$ particles other than $i$.
   (b) Find $objBest(N)$, the best objective function $obj(x)$ among the neighbors.
   (c) Find $gBest(N)$, the position of the neighbor with the best objective function.
   (d) Update the velocities:
           $v_{new} = I \cdot v_{old} + c_1 \cdot rand_1 \cdot (pBest - x) + c_2 \cdot rand_2 \cdot (gBest - x),$
       where:
       i. $v_{new}$ and $v_{old}$ are the new and old particle $i$ velocities, respectively.
       ii. $I$ is the inertia component.
       iii. $c_1$ and $c_2$ are the self-adjustment and social-adjustment components, respectively.
       iv. $rand1$ and $rand2$ are uniformly (0,1) distributed random vectors of length $nVars$.
       v. $pBest$ is the location of the best objective function that particle $i$ has previously found.
       vi. $gBest$ is the location of the best objective function found in the current neighborhood.
   (e) Update the position: $x(i) = x(i) + v(i)$.
   (f) If any component of $x(i)$ is outside a bound, set it equal to that bound.
   (g) Evaluate the objective function $obj(i) = obj(x(i))$. Set $Flag(i) = False$.
   (h) If $obj(i) > obj(pBest)$, then update $pBest = x$.
   (i) If $obj(i) > bestFound$, then set $bestFound = obj(i), bestPosition = x(i)$ and $Flag(i) = True$.
   (j) Update inertia $I$:
       i. If $Flag(i) = True$:
          A. Set $inertiaCounter = max(0, inertiaCounter - 1)$.
          B. If $inertiaCounter < 2$, then set $I = I \cdot 2$.
          C. If $inertiaCounter > 5$ then set $I = I/2$.
          D. Ensure that I is within $inertiaRange$.
       ii. If $Flag(i) = False$: $inertiaCounter = inertiaCounter + 1$
   (k) Update neighborhood size, $M$:
       i. If $Flag(i) = True$: $M = nFracK$
       ii. If $Flag(i) = False$: $M = min(M + nFrac \cdot K, K)$.
7. If relative change in the best objective function value is less than specified tolerance set $stallCounter = stallCounter + 1$.
8. If $stallCounter < stallIterLimit$ (or not reaching other termination criteria) iterate back to 6.
9. Else, return $bestFound$ and $bestPosition$.
```

Fig. 14 Pseudo-code for PSO maximization with a dynamic neighborhood (adopted from Matlab documentation)

Table 7 Parameter values used in both examples for the particle swarm optimization algorithm

	Example 1	Example 2
Function tolerance	10^{-6}	10^{-4}
stallIterLimit	20	12
K	30	
nFrac	0.25	
inertiaRange	(0.1, 1.1)	
c1, c2	1.49	

and to the best solution found by his current dynamic-neighborhood, M. Thus, the local search here can be viewed as "attraction" to the good quality solutions.

However, at the same time, the algorithm introduces few variables that add stochastic search which might explore meta-heuristically the new areas. These variables are $rand_1$ and $rand_2$ that add random effect to each source of attraction.

Additional global exploration is achieved by adaptively changing the inertia parameter, I, and the dynamic neighborhood M, according to the optimization process. For example, once a better solution is found, the neighborhood is set to its minimum size. Thus, a particle can be attracted to a direction, which might be very different than a direction achieved, if the total swarm was considered as the neighborhood. This dynamical neighborhood approach was not presented in several previous works that utilized PSO to solve the problem under consideration.

References

1. Avansi, G.D., Schiozer, D.J.: UNISIM-I: Synthetic model for reservoir development and management applications. Int. J. Model. Simul. Pet. Ind. **9**(1), 21–30 (2015)
2. Awotunde, A.A.: On the joint optimization of well placement and control. In: SPE Saudi Arabia Sect. Tech. Symp. Exhib. Society of Petroleum Engineers (2014)
3. Awotunde, A.A.: Generalized field-development optimization with well-control zonation. Comput. Geosci. **20**(1), 213–230 (2016)
4. Bacoccoli, G., Morales, R.G., Campos, O.A.J.: The Namorado oil field: a major oil discovery in the Campos Basin, Brazil. In: Giant Oil Gas Fields Decad. 1968-1978, vol. 30, pp. 329–338 (1980)
5. Behforooz, G.: The not-a-knot piecewise interpolatory cubic polynomial. Appl. Math. Comput. **52**(1), 29–35 (1992)
6. Bellout, M.C., Ciaurri, D.E., Durlofsky, L.J., Foss, B., Kleppe, J.: Joint optimization of oil well placement and controls. Comput. Geosci. **16**, 1061–1079 (2012)
7. Bertrand, R., Epenoy, R.: New smoothing techniques for solving bang-bang optimal control problems—numerical results and statistical interpretation. Optim. Control Appl. Methods **23**(4), 171–197 (2002)
8. Biegler, L.T., Cervantes, A.M., Wachter, A.: Advances in simultaneous strategies for dynamic process optimization. Chem. Eng. Sci. **57**(4), 575–593 (2002)
9. Boukouvala, F., Floudas, C.A.: ARGONAUT: Algorithms for global optimization of constrained grey-box computational problems. Optim. Lett. 1–19. doi:10.1007/s11590-016-1028-2 (2014)
10. Boukouvala, F., Hasan, M.M.F., Floudas, C.A.: Global optimization of general constrained grey-box models: new method and its application to constrained PDEs for pressure swing adsorption. J. Global Optim. 1–40. doi:10.1007/s10898-015-0376-2 (2015)
11. Boukouvala, F., Misener, R., Floudas, C.A.: Global optimization advances in mixed-integer nonlinear programming, MINLP, and constrained derivative-free optimization, CDFO. Eur. J. Oper. Res. **252**(3), 701–727 (2015)
12. Brouwer, D., Jansen, J.D.: Dynamic optimization of waterflooding with smart wells using optimal control theory. SPE J. **9**(4), 391–402 (2004)
13. Chen, Y., Oliver, D.S., Zhang, D.: Efficient ensemble-based closed-loop production optimization. SPE Symp. Improv. Oil Recover. Society of Petroleum Engineers (2008)
14. Ciaurri, D.E., Isebor, O.J., Durlofsky, L.J.: Application of derivative-free methodologies to generally constrained oil production optimisation problems. Int. J. Math. Model. Num. Optim. **2**(2), 134–161 (2011)
15. Ciaurri, D.E., Mukerji, T., Durlofsky, L.J.: Derivative-free optimization for oil field operations. In: Computational Optimization and Applications in Engineering and Industry, pp. 19–55. Springer (2011)
16. Codas Duarte, A., Foss, B., Camponogara, E.: Output-constraint handling and parallelization for oil-reservoir control optimization by means of multiple shooting. SPE J. **20**(4), 856–871 (2015)
17. De Boor, C.: Convergence of cubic spline interpolation with the not-a-knot condition. Tech. rep., DTIC Document (1985)
18. de Holanda, R.W., Gildin, E., Jensen, J.L.: Improved waterflood analysis using the capacitance-resistance model within a control systems framework. In: SPE Lat. Am. Caribb. Pet. Eng. Conf. Society of Petroleum Engineers (2015)
19. Doublet, D.C., Aanonsen, S.I., Tai, X.C.: An efficient method for smart well production optimisation. J. Pet. Sci. Eng. **69**(1), 25–39 (2009)
20. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, vol. 1, pp. 39–43. New York (1995)
21. Fonseca, R., Leeuwenburgh, O., Van den Hof, P., Jansen, J.D.: Improving the ensemble-optimization method through covariance-matrix adaptation. SPE J. **20**(1), 155–168 (2014)
22. Forouzanfar, F., Poquioma, W.E., Reynolds, A.C.: Simultaneous and sequential estimation of optimal placement and controls of wells using a covariance matrix adaptation algorithm. SPE J. **21**(2), 501–521 (2015)
23. Fragoso, M., Horowitz, B., Jose Roberto, P.R.: Retraining criteria for TPWL/POD surrogate based waterflodding optimization. In: SPE Reserv. Simul. Symp., pp. 23–25. Society of Petroleum Engineers (2015)
24. Gao, G., Reynolds, A.C.: An improved implementation of the LBFGS algorithm for automatic history matching. SPE J. **11**(1), 5–17 (2006)
25. Gaspar, A.T., Avansi, G.D., dos Santos, A.A.d.S., Filho, J.C.v.H., Schiozer, D.J.: UNISIM-I-D: Benchmark studies for oil field 1103 development and production strategy selection. Int. J. Model. Simul. Pet. Ind. 9(1) (2015)
26. Ghasemi, M., Yang, Y., Gildin, E., Efendiev, Y., Calo, V.: Fast multiscale reservoir simulations using POD-DEIM model reduction. In: SPE Reserv. Simul. Symp., pp. 2325. Society of Petroleum Engineers (2015)

27. Ghommem, M., Gildin, E., Ghasemi, M.: Complexity reduction of multiphase flows in heterogeneous porous media. SPE J. **21**(1), 144–151 (2016)
28. He, J., Durlofsky, L.J.: Reduced-order modeling for compositional simulation by use of trajectory piecewise linearization. SPE J. **19**(5), 858–872 (2014)
29. Humphries, T.D., Haynes, R.D., James, L.A.: Simultaneous and sequential approaches to joint optimization of well placement and control. Comput. Geosci. **18**, 433–448 (2014)
30. Isebor, O.J., Ciaurri, D.E., Durlofsky, L.J.: Generalized field-development optimization with derivative-free procedures. SPE J. **19**(5), 891–908 (2014)
31. Jacobson, D.: Differential dynamic programming methods for solving bang-bang control problems. IEEE Trans. Automat. Contr. **13**(6), 661–675 (1968)
32. Jansen, J.D., Brouwer, R., Douma, S.G.: Closed loop reservoir management. In: SPE Reserv. Simul. Symp., pp. 24. Society of Petroleum Engineers (2009)
33. Krogstad, S., Lie, K.A., Møyner, O., Nilsen, H.M., Raynaud, X., ard Skaflestad: SPE 173317-MS MRST-AD—an open-source framework for rapid prototyping and evaluation of reservoir simulation problems. SPE Reserv. Simul. Symp, pp. 232523–25 (2015)
34. Lie, K.A. An introduction to reservoir simulation using MATLAB: User guide for the Matlab reservoir simulation toolbox (MRST) (2015)
35. Lie, K.A., Krogstad, S., Ligaarden, I.S., Natvig, J.R., Nilsen, H.M., Skaflestad, B.: Open-source MATLAB implementation of consistent discretisations on complex grids. Comput. Geosci. **16**, 297–322 (2012)
36. Lien, M., Brouwer, D., Mannseth, T., Jansen, J.D.: Multiscale regularization of flooding optimization for smart field management. SPE J. **13**(2), 195–204 (2008)
37. MathWorks: Particle swarm optimization algorithm (2015). http://www.mathworks.com/help/gads/particle-swarm-optimization-algorithm.html
38. MathWorks: Constrained nonlinear optimization algorithms (2016). http://www.mathworks.com/help/optim/ug/constrained-nonlinear-optimization-algorithms.html
39. Oliver, D.S., Reynolds, A.C., Liu, N.: Optimization for nonlinear problems using sensitivities. In: Inverse Theory Pet. Reserv. Charact. Hist. Matching, chap. 8 (2008)
40. Onwunalu, J.E., Durlofsky, L.J.: Application of a particle swarm optimization algorithm for determining optimum well location and type. Comput. Geosci. **14**(1), 183–198 (2010)
41. Pinto, M.A.S., Ghasemi, M., Sorek, N., Gildin, E., Schiozer, D.J.: Hybrid optimization for closed-loop reservoir management. In: SPE Reserv. Simul. Symp., vol. 3, pp. 15001510. Society of Petroleum Engineers (2015)
42. Reynolds, A.C., Oliveira, D.: An adaptive hierarchical algorithm for estimation of optimal well controls. In: SPE Reserv. Simul. Symp., pp. 1–26 (2013)
43. Sarma, P., Durlofsky, L.J., Aziz, K., Chen, W.H.: Efficient real-time reservoir management using adjoint-based optimal control and model updating. Comput. Geosci. **10**(1), 3–36 (2006)
44. Seydenschwanz, M.: Convergence results for the discrete regularization of linear-quadratic control problems with bang–bang solutions. Comput. Optim. Appl. **61**(3), 731–760 (2015)
45. Shirangi, M.G., Durlofsky, L.J.: Closed-loop field development under uncertainty by use of optimization with sample validation. SPE J. **20**(5), 908–922 (2015)
46. Shuai, Y., White, C.D., Zhang, H., Sun, T.: Using multiscale regularization to obtain realistic optimal control strategies. In: SPE Reserv. Simul. Symp., pp. 2123. Society of Petroleum Engineers (2011)
47. Silva, C., Trélat, E.: Smooth regularization of bang-bang optimal control problems. IEEE Trans. Automat. Contr. **55**(11), 2488–2499 (2010)
48. Sudaryanto, B., Yortsos, Y.C.: Optimization of fluid front dynamics in porous media using rate control. I. Equal mobility fluids. Phys. Fluids **12**, 1656 (2000)
49. Sudaryanto, B., Yortsos, Y.C.: Optimization of displacements in porous media using rate control. In: SPE Annu. Tech. Conf. Exhib. Society of Petroleum Engineers (2001)
50. UNICAMP: UNISIM-I: Benchmark case (2016)
51. van Essen, G., Van den Hof, P., Jansen, J.D.: Hierarchical long-term and short-term production optimization. SPE J. **16**(1), 191–199 (2011)
52. Wang, C., Li, G., Reynolds, A.C.: Production optimization in closed-loop reservoir management. SPE J. **14**(3), 506–523 (2009)
53. Zandvliet, M., Bosgra, O., Jansen, J.D., Van den Hof, P., Kraaijevanger, J.: Bang-bang control and singular arcs in reservoir flooding. J. Petrol. Sci. Eng. **58**(1), 186–200 (2007)
54. Zhang, F., Reynolds, A.C.: Optimization algorithms for automatic history matching of production data. In: ECMOR VIII-8th European Conference on the Mathematics of Oil Recovery (2002)
55. Zhao, H., Chen, C., Do, S., Oliveira, D., Li, G., Reynolds, A.C.: Maximization of a dynamic quadratic interpolation model for production optimization. SPE J. **18**(6), 1012–1025 (2013)