Comput Geosci
DOI 10.1007/s10596-013-9383-x

ORIGINAL PAPER

A derivative-free methodology with local and global search for the constrained joint optimization of well locations and controls

Obiajulu J. Isebor · Louis J. Durlofsky · David Echeverría Ciaurri

Received: 7 February 2013 / Accepted: 28 October 2013

Abstract In oil field development, the optimal location for a new well depends on how it is to be operated. Thus, it is generally suboptimal to treat the well location and well control optimization problems separately. Rather, they should be considered simultaneously as a joint problem. In this work, we present noninvasive, derivative-free, easily parallelizable procedures to solve this joint optimization problem. Specifically, we consider Particle Swarm Optimization (PSO), a global stochastic search algorithm; Mesh Adaptive Direct Search (MADS), a local search procedure; and a hybrid PSO–MADS technique that combines the advantages of both methods. Nonlinear constraints are handled through use of filter-based treatments that seek to minimize both the objective function and constraint violation. We also introduce a formulation to determine the optimal number of wells, in addition to their locations and controls, by associating a binary variable (drill/do not drill) with each well. Example cases of varying complexity, which include bound constraints, nonlinear constraints, and the determination of the number of wells, are presented. The PSO–MADS hybrid procedure is shown to consistently outperform both stand-alone PSO and MADS when solving the joint problem. The joint approach is also observed to provide superior performance relative to a sequential procedure.

O. J. Isebor (✉) · L. J. Durlofsky
Department of Energy Resources Engineering,
Stanford University, Stanford, CA 94305, USA
e-mail: oisebor@alumni.stanford.edu

L. J. Durlofsky
e-mail: lou@stanford.edu

D. Echeverría Ciaurri
T. J. Watson Research Center, IBM,
Yorktown Heights, NY 10598, USA
e-mail: decheve@us.ibm.com

Keywords Derivative-free optimization · Field development optimization · Well placement · Production optimization · Reservoir simulation-based optimization · Nonlinear programming

Mathematics Subject Classifications (2010) 90-08 · 90C11 · 90C26 · 90C30 · 90C56 · 90C90

1 Introduction

The development of computational optimization procedures for oil field operations has been an area of active research in recent years. Optimization techniques have been developed for several types of field development and operational decisions, including the determination of the optimal type and location of new wells and the optimal operation of existing wells (as discussed below). Traditionally, the optimization of well location has been considered separately from the optimization of well operation (the latter is often referred to as well control or production optimization). However, recent work has demonstrated that, as would be expected, the optimal location of a new well depends on how the well is to be operated [6, 33, 45]. Thus, the optimization of well position and well control should be considered as a joint optimization problem, rather than as two separate (sequential) optimization problems. Our goal in this paper is to develop and test derivative-free procedures that enable the joint optimization of well location and control under general (nonlinear) constraints.

The well control problem involves determining optimal values for continuous operating variables, such as well rates or bottomhole pressures, in order to maximize an objective function (e.g., net present value, cumulative oil production, etc.). The well placement optimization problem involves

Published online: 27 November 2013

maximizing an objective function by varying well types and well locations. During the solution of the well placement optimization problem, various treatments can be used for the well controls, including fixed bottomhole pressures or "reactive" control. In the latter case, production wells are closed (shut in) when water cut exceeds a specified limit [45].

As shown by Bellout et al. [6] and Zandvliet et al. [45], the optimized well locations depend on the control strategy used during the well placement optimization. Thus, any approach that does not aim at optimizing both well locations and controls simultaneously can be expected to provide theoretically suboptimal results. Recent investigations have therefore addressed the joint, or simultaneous, optimization of well placement and well control variables for oil field problems [6, 27, 33] and problems in geological carbon storage [9]. In assessments of joint versus sequential approaches, Bellout et al. [6] and Li and Jafarpour [33] found their joint approaches to outperform sequential procedures. Humphries et al. [27], by contrast, did not find the joint approach to consistently provide superior results relative to their sequential procedure. This may be due to the fact that they included some heuristics for well control in their well placement optimizations, or to some other algorithmic treatments.

A variety of methods have been applied to solve the well control, well placement, and joint placement and control problems. Due to their more continuous nature, well control problems are amenable to solution using gradient-based optimization methods such as sequential quadratic programming [34]. Adjoint formulations allow for efficient (but simulator-invasive) computation of gradients and have been used effectively for these problems (see, e.g., Brouwer and Jansen [8] and Sarma et al. [40]). Wang et al. [42] and Forouzanfar et al. [22] applied gradient-based methods for well control optimization that heuristically approached the joint problem by eliminating wells that did not satisfy minimum injection or production criteria. Derivative-free methods have also been used for well control problems [1, 17, 25]. These approaches may require large numbers of function evaluations (reservoir simulations) when compared with gradient-based methods, but they are often easily parallelized, so elapsed time can be greatly reduced with distributed computing.

Due to the effects of reservoir heterogeneity, well placement optimization problems can display very rough optimization surfaces, with multiple local optima [36]. Thus, these problems are frequently solved using stochastic search procedures (which avoid getting trapped in some local optima) including Genetic Algorithms [24, 44], Particle Swarm Optimization [36], and stochastic perturbation methods [5]. Gradient-based approaches have also been applied for these problems [39, 45, 46], though these methods may get trapped in relatively poor local optima.

Our intent in this paper is to introduce and apply several new procedures for the joint optimization problem. The underlying optimization techniques considered are noninvasive (with respect to the simulator) derivative-free methods that naturally parallelize. Noninvasive (also known as black-box) methods are required when one does not have access to simulator source code. Echeverría Ciaurri et al. [18] illustrated the applicability of noninvasive derivative-free optimization methods to well control, well placement, and history matching problems. Our solution of the combined well placement and control problem entails the use of Mesh Adaptive Direct Search (MADS), which is a local optimization technique with established convergence theory, and Particle Swarm Optimization (PSO), a stochastic global search procedure. A new hybrid PSO–MADS procedure that combines the positive features of these two methods will be presented and applied.

Our framework also includes a treatment for general (nonlinear) constraints in the optimization. These are handled using filter-type approaches, as described in [17, 28], in which the constrained problem is essentially viewed as a biobjective optimization where the two objectives are the maximization of the net present value (for example) and the minimization of the constraint violation. The filter method for MADS was introduced by Audet and Dennis [3]. Here, we describe new filter treatments for PSO and the PSO–MADS hybrid. We also introduce a problem formulation that includes drill/do not drill binary variables that are associated with each well. This allows the procedure to add or eliminate wells and thus optimize the number of wells to be drilled (subject to a specified maximum).

An important aspect of the general problem that is not considered here is the treatment of geological uncertainty. We note, however, that our framework is readily compatible with approaches that model the impact of uncertainty by optimizing over multiple realizations, as demonstrated by Isebor [28]. Our methodology requires a large number of function evaluations (flow simulations), and although the computational effort is substantially reduced by distributed computing, further reduction could be accomplished through use of surrogate models. This could entail, for example, the use of reduced-order numerical models [11], quadratic approximations, e.g., as used in the Bound Optimization by Quadratic Approximation (BOBYQA) algorithm [23, 38], or kriging procedures within Efficient Global Optimization [30].

This paper proceeds as follows. First, the optimization problem statement is presented, together with the sequential and proposed joint solution approaches. We then describe the underlying derivative-free optimization methods used in

our framework: MADS [4, 32], PSO [15], and the PSO–MADS hybrid method, which is a variant of the PSwarm algorithm introduced by Vaz and Vicente [41]. We also describe filter techniques for handling nonlinear constraints in each of the methods. Next, optimization results for well placement and well control are presented. We consider a case with only bound constraints, a case that additionally includes nonlinear constraints, and a case with nonlinear constraints where we also optimize the number of wells. We conclude with a summary and suggestions for future work in this area.

2 Problem statement and solution approaches

In this section, we present the optimization problem for combined well placement and control. Two types of approaches for its solution are then described—the more commonly used sequential procedure and our proposed joint approach. These techniques are discussed with reference to waterflood operations (injection of water into a reservoir initially containing oil), though the procedures we develop are also applicable for other recovery processes.

2.1 Optimization problem

The goal of the optimization is to determine the optimal locations of some number of injection and production wells, along with the optimal well settings as a function of time. Later, we will introduce a treatment for additionally optimizing the number of wells. Well settings, also referred to as well controls, can be specified in terms of well flow rates or bottomhole pressures (BHPs). In this work, the well controls are BHPs. We seek to optimize the net present value or NPV of the operation (other objective functions could of course be considered) by optimizing the locations and controls in a joint, rather than sequential, manner. The optimization problem can be stated as follows:

$$\min_{\mathbf{x}\in X,\mathbf{u}\in U} \; f(\mathbf{x},\mathbf{u}), \quad \text{subject to} \;\; \mathbf{c}(\mathbf{x},\mathbf{u}) \le \mathbf{0}, \tag{1}$$

where f is the objective function we seek to minimize (e.g., $f = -\text{NPV}$ if we wish to maximize NPV) and $\mathbf{c} \in \mathbb{R}^m$ is the vector of m nonlinear constraint functions. The bounded sets $X = \{\mathbf{x} \in \mathbb{Z}^{n_1};\ \mathbf{x}_l \le \mathbf{x} \le \mathbf{x}_u\}$ and $U = \{\mathbf{u} \in \mathbb{R}^{n_2};\ \mathbf{u}_l \le \mathbf{u} \le \mathbf{u}_u\}$ define the allowable values for the discrete well placement variables $\mathbf{x}$ and continuous well control variables $\mathbf{u}$. The set X could define possible drilling locations within the reservoir boundaries and the set U could define allowable BHP ranges.

Even though the actual well locations are real-valued, the well placement variables are modeled as integers because the simulator used in this work requires well locations to be defined in terms of discrete grid blocks. The wells are assumed to be vertical with locations stated in terms of discrete areal (x, y) coordinates. Thus, we have $n_1 = 2(N_\text{I} + N_\text{P})$, where N_I and N_P are the number of injection and production wells, respectively. In more general cases involving deviated or multilateral wells, additional variables would be required to describe well locations. The well controls are represented by piecewise constant functions in time with N_T intervals. Thus, $n_2 = N_\text{T}(N_\text{I} + N_\text{P})$.

We optimize undiscounted NPV, given by

$$\begin{aligned}\text{NPV} = &-\underbrace{\sum_{j=1}^{N_\text{I}+N_\text{P}} C^j}_{\text{drilling costs}} - \sum_{j=1}^{N_\text{I}}\sum_{k=1}^{N_\text{t}} \Delta t_k \underbrace{c_\text{iw} q_\text{iw}^{j,k}(\mathbf{x},\mathbf{u})}_{\text{water injection cost}} \\ &+ \sum_{j=N_\text{I}+1}^{N_\text{I}+N_\text{P}} \sum_{k=1}^{N_\text{t}} \Delta t_k \left(\underbrace{p_\text{o} q_\text{o}^{j,k}(\mathbf{x},\mathbf{u})}_{\text{oil revenue}} - \underbrace{c_\text{pw} q_\text{pw}^{j,k}(\mathbf{x},\mathbf{u})}_{\text{water disposal cost}} \right),\end{aligned} \tag{2}$$

where C^j is the cost to drill well j, N_t is the number of time steps in the reservoir simulation, Δt_k represents the time step size at time step k, c_iw and c_pw are the costs per barrel of injected and produced water, and p_o is the sale price of produced oil. The terms $q_\text{iw}^{j,k}$, $q_\text{pw}^{j,k}$, and $q_\text{o}^{j,k}$ represent the rates of injected and produced water, and produced oil, from well j in time step k. Note that these rates are functions of the optimization variables and are obtained from the reservoir simulator. The reservoir simulator used in this work is Stanford University's General Purpose Research Simulator (GPRS) [10].

2.2 Sequential solution approach

The well placement and control problems are commonly addressed in a decoupled manner, with the well placement part solved first and the well control optimization solved second, in contrast to the joint optimization proposed in Eq. 1. Using a sequential procedure, the well placement variables are optimized with an assumed control strategy by solving

$$\mathbf{x}_S^* = \arg\min_{\mathbf{x}\in X} \; f(\mathbf{x},\mathbf{u}_0), \quad \text{subject to} \;\; \mathbf{c}(\mathbf{x},\mathbf{u}_0) \le \mathbf{0}, \tag{3}$$

where $\mathbf{u}_0 \in U$ defines the assumed control strategy. Possible control strategies include using constant BHPs (typically set at the bounds, implying wells injecting or producing at maximum rates) or the potentially more effective "reactive" control strategy, which is used in this work when we perform sequential optimizations. Under the reactive control strategy considered here, injection wells always operate at their maximum BHP bounds. Production wells operate at their minimum BHP bounds until a prescribed limit is reached. This limit can be defined in terms of a maximum

allowable fraction of water in the produced fluid, or (analogously) when the water production cost ($c_{\mathrm{pw}} q_{\mathrm{pw}}^{j,k}$) exceeds oil production revenue ($p_{\mathrm{o}} q_{\mathrm{o}}^{j,k}$) for the well. Although it can give reasonable results in some cases, reactive control represents a heuristic treatment which will, in general, be suboptimal. After the solution of Eq. 3, the well locations are fixed at $\mathbf{x}_S^*$ and the following well control optimization problem is solved:

$$\mathbf{u}_S^* = \arg\min_{\mathbf{u}\in U} \; f(\mathbf{x}_S^*, \mathbf{u}), \quad \text{subject to} \;\; \mathbf{c}(\mathbf{x}_S^*, \mathbf{u}) \leq \mathbf{0}. \tag{4}$$

As discussed in Section 1, a number of different optimization techniques—both gradient-based and derivative-free—have been applied for the problems defined by Eqs. 3 and 4. Here, we will introduce a hybrid technique that combines two noninvasive derivative-free optimization procedures for the solution of both problems. The specific approaches incorporated are Particle Swarm Optimization, which has been used for the well placement problem [36], and Mesh Adaptive Direct Search, similar to Generalized Pattern Search [2], which has been applied for well control [17].

2.3 Joint solution approach

The sequential approach defined above has the advantage of solving two smaller problems (well placement, of dimension n_1, and well control, of dimension n_2) instead of one large problem (of dimension $n_1 + n_2$). As shown by Bellout et al. [6] and Zandvliet et al. [45], the controls applied during the well placement optimization affect the optimized well locations. Thus, any approach that does not optimize the location and control variables simultaneously can be expected to be suboptimal. This motivates the joint optimization of the problem defined in Eq. 1.

As noted in Section 1, joint optimization approaches have been developed previously. Bellout et al. [6] presented a procedure based on a nested bi-level optimization. In that method, well placement is the master problem, and in order to evaluate the objective function associated with a particular configuration of wells, the well controls are optimized to a certain degree. This approach allows the use of different optimization methods for the two problems, and in their implementation, the well placement was accomplished using derivative-free direct search procedures, while the well control subproblem was addressed with an efficient adjoint-based gradient technique. Li and Jafarpour [33] used an iterative procedure in which they alternated between optimizing well placement and well control. Again, different optimization approaches were used for the two problems. The results presented in both studies demonstrated the advantages of joint optimization compared to sequential procedures.

Our approach can be seen as an alternative to these earlier treatments, but we also introduce several important extensions. We solve the joint well placement and control problem with a single optimization method, in contrast to the earlier procedures that addressed the joint problem but used different treatments for the two subproblems. We also include general (nonlinear) constraints in our framework (which do not appear to have been considered previously for the joint problem) and present a hybrid optimization procedure that includes a stochastic global search method (PSO) along with a local direct search technique (MADS). We additionally introduce a formulation, with binary decision variables, that enables the optimization of the number of wells along with the well locations and controls. We now describe the detailed procedures and the overall workflow.

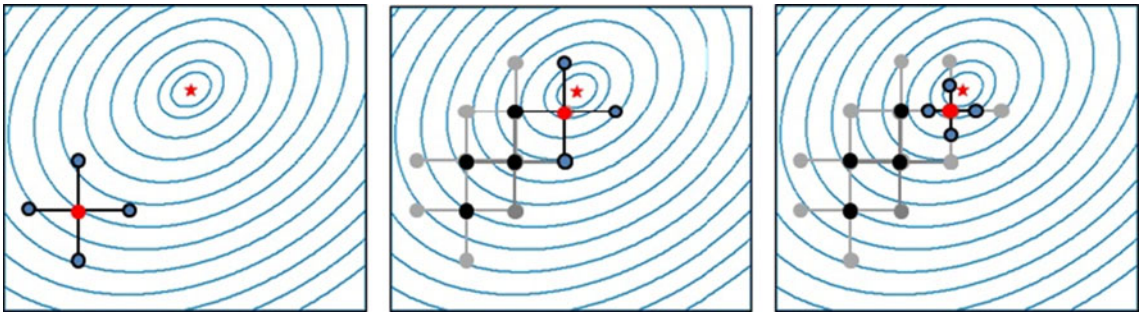
3 Optimization framework and methods

We consider derivative-free optimization techniques, an important class of optimization methods applicable to problems where gradients are not available, difficult or expensive to obtain, or ill-defined. Echeverría Ciaurri et al. [18] have illustrated the applicability of such methods to problems in oil field development and operation. After discussing the two optimization techniques used in this work, we present a hybrid approach that combines their positive features. As noted earlier, the key developments in this work include the new PSO–MADS hybrid algorithm and the consistent filter-based treatment of nonlinear constraints in both stand-alone PSO and the PSO–MADS hybrid. For further details on the optimization procedures discussed here, please see Isebor [28].

In our descriptions in this section, we consider the following general optimization problem:

$$\min_{\mathbf{x}\in\Omega} \; f(\mathbf{x}), \quad \text{subject to} \;\; \mathbf{c}(\mathbf{x}) \leq \mathbf{0},$$
$$\text{with} \;\; \Omega = \{\mathbf{x} \in \mathbb{R}^n \;:\; \mathbf{x}_l \leq \mathbf{x} \leq \mathbf{x}_u\}. \tag{5}$$

Note that the vector $\mathbf{x}$ now contains both discrete and continuous variables (we use $\mathbf{x}$ here in place of $(\mathbf{x}, \mathbf{u})$ in Eqs. 1 and 2 to simplify the presentation). As we will see later in Section 4.3, $\mathbf{x}$ can also contain binary categorical variables that determine the number of wells. The dimension of $\mathbf{x}$ varies with the number of wells and control periods and is usually on the order of tens to hundreds. The derivative-free methods presented here are not suitable for problems with many hundreds or thousands of optimization variables because the computational expense, which scales with



(a) Initial polling stencil (b) Successive polling iterations (c) Contraction of polling stencil

Fig. 1 Illustration of a polling sequence in $\mathbb{R}^2$. The *red star* designates a local optimum, the *red circles* are poll centers around which the polling stencil for the current iteration is defined (with polling stencil size Δ_k^p), the *blue circles* are poll points to be evaluated at each iteration, and the *black circles* indicate the sequence of previously evaluated poll centers

the number of variables, would be excessive on a current (typical) computer cluster.

3.1 Local derivative-free method: Mesh Adaptive Direct Search

Pattern search algorithms are a family of optimization methods based on polling, which is the local exploration of the objective function surface around the current iterate. Polling is illustrated in Fig. 1 and proceeds as follows. At any iteration k, a polling stencil is centered at the current best solution $\mathbf{x}_k$ (the poll center), as depicted in Fig. 1a. The stencil comprises a set of directions, of which at least one is a descent direction, and a poll stencil size, Δ_k^p. The objective function is evaluated at the stencil end points, and if one of these trial poll points leads to an improvement in the objective function, the center of the stencil is moved to this point for the next iteration $k+1$. See Fig. 1b for a sequence of polling iterations with improvements. If no stencil poll point yields improvement, the stencil size is reduced, as illustrated in Fig. 1c, and polling continues with the smaller stencil size.

If the stencil orientation is fixed (e.g., as shown in Fig. 1) at every iteration, the resulting method is essentially Generalized Pattern Search (GPS) [2]. However, if the stencil orientation varies from iteration to iteration, in a manner such that polling is done in an asymptotically dense set of directions, we have the MADS algorithm of Audet and Dennis [4]. A key difference between GPS and MADS is that in MADS, we have an underlying mesh with mesh size Δ_k^m on which the poll points must lie and $\Delta_k^m \leq \Delta_k^p$, whereas in GPS, we have a single stencil size with $\Delta_k^m = \Delta_k^p$. At unsuccessful iterations, by allowing Δ_k^m to decrease faster than Δ_k^p, the MADS algorithm is able to access more possible polling directions. Consistent with this, Audet and Dennis [4] present results indicating that MADS yields better solutions than GPS for constrained problems. Thus, we apply MADS in this study.

Pattern search algorithms such as MADS often include provision for an optional search step, in addition to the poll step. The search step enables great flexibility as it allows the use of any method to generate a finite number of search points in each iteration (these points could be generated anywhere in the same mesh as the polling points in an attempt to "globalize" the optimization process). The search step does not disrupt the convergence characteristics provided by polling [4]. See Fig. 2 for an illustration of possible search, $S_k = \{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3\}$, and poll points, $P_k = \{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4\}$, generated at some iteration k of an optimization with two variables.

As the algorithm progresses, the independent evolution of the mesh and poll size parameters is designed such that the set of MADS poll directions becomes dense in the space

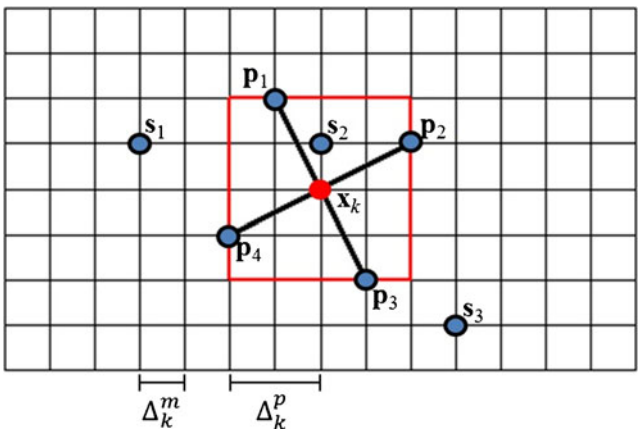


Fig. 2 Example of MADS directions in the case $n = 2$, at iteration k of the algorithm

of optimization variables, meaning that potentially every direction can be explored [4]. For discrete variables, the mesh is modified such that the coordinates corresponding to these variables are constrained to have discrete points. Note that the MADS polling process parallelizes naturally since, at every iteration, the objective function evaluations at the poll points can be accomplished in a distributed fashion. The basic MADS algorithm is summarized in Fig. 3 (see [4] for details).

Later, we will describe the use of PSO for the search step, which leads to significant global exploration of the solution space. The MADS poll step ensures theoretical local convergence and is based on poll directions that vary with iteration. In each iteration, after the search and poll steps, a final update step is performed. To accomplish this update, the algorithm must first determine if the iteration is a success or a failure. In the unconstrained case (i.e., nonlinear constraints are absent), a successful iteration occurs when the objective function is improved. For problems with nonlinear constraints, a filter method [21], as described below, is used for this assessment.

Different stopping criteria for terminating MADS can be considered. In our implementation, the optimization is terminated if the mesh or poll size parameters, Δ_k^m or Δ_k^p, which are reduced at every unsuccessful iteration, decrease beyond specified thresholds or if a specified maximum number of iterations is reached. In pattern search methods, it can be seen that the convergence of the mesh or stencil sizes to zero implies the convergence of the gradient of the cost function to zero [31]. In all of the examples presented in this paper, the mesh size criterion terminates the MADS optimization process.

Initialization: Let $\mathbf{x}_0$ be the initial guess such that $f(\mathbf{x}_0)$ is finite and let M_0 be the mesh defined over the solution domain with initial mesh size $\Delta_0^m = \Delta_0^p > 0$. For $k = 0, 1, 2, \ldots$, perform the following:

1. **Search:** Use some finite strategy to seek an improved mesh point, i.e., $\mathbf{x}_{k+1} \in M_k$ defined by Δ_k^m, such that $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$, for minimization.
2. **Poll:** If the search step was unsuccessful, evaluate f at points in the poll set defined by a stencil, with random poll directions, of size Δ_k^p centered at $\mathbf{x}_k$ in order to find an improved mesh point, $\mathbf{x}_{k+1}$.
3. **Update:** If in the search or poll steps an improved mesh point is found, update $\mathbf{x}_k$ to $\mathbf{x}_{k+1}$ and set $\Delta_{k+1}^m \geq \Delta_k^m$ and $\Delta_{k+1}^p \geq \Delta_k^p$. Otherwise, set $\mathbf{x}_{k+1} = \mathbf{x}_k$, $\Delta_{k+1}^m = \theta \Delta_k^m$ and $\Delta_{k+1}^p = \phi \Delta_k^p$, with contraction parameters $0 < \theta < \phi < 1$.

Fig. 3 MADS algorithm, with a search-poll paradigm

3.2 Constraint treatment in MADS

The bound constraints on the variables in Eq. 5 are enforced by performing the following coordinate-wise projection of trial points onto Ω:

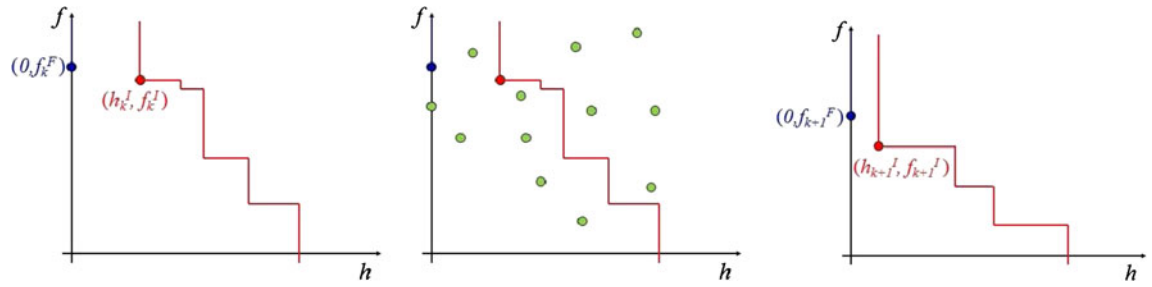
$$\text{proj}_{\Omega}(x_i) = \begin{cases} x_{l_i} & \text{if } x_i < x_{l_i}, \\ x_{u_i} & \text{if } x_i > x_{u_i}, \\ x_i & \text{otherwise.} \end{cases} \tag{6}$$

For nonlinear constraints, the filter method is used. Filter methods [17, 21] are step-acceptance mechanisms that seek to avoid the robustness issues that may exist with penalty function methods and other more traditional constraint handling approaches. The use of filters can be seen as an add-on to an optimization algorithm. Instead of combining the objective function and constraint violation into a single function, as is done when using penalty functions, the problem in Eq. 5 is viewed as a biobjective optimization in which we aim at minimizing both the objective function $f(\mathbf{x})$ and an aggregate constraint violation function, defined as follows:

$$h(\mathbf{x}) = \left[\sum_{j=1}^{m} \left(\max \left(c_j(\mathbf{x}), 0 \right) \right)^2 \right]^{1/2}. \tag{7}$$

The second objective of minimizing $h(\mathbf{x})$ is preferred over optimizing $f(\mathbf{x})$ because the solution determined by the optimization algorithm should be feasible. Using terminology from multiobjective optimization, a point $\mathbf{x}_a$ is said to dominate another point $\mathbf{x}_b$ (written as $\mathbf{x}_a \prec \mathbf{x}_b$) if and only if $f(\mathbf{x}_a) \leq f(\mathbf{x}_b)$ and $h(\mathbf{x}_a) \leq h(\mathbf{x}_b)$, with at least one of these being a strict inequality. A filter is defined as a list of pairs $\left(h(\mathbf{x}_f), f(\mathbf{x}_f)\right)$ such that no pair dominates another pair. An iterate $\mathbf{x}_k$ is considered to be acceptable, or "unfiltered," if $(h(\mathbf{x}_k), f(\mathbf{x}_k))$ is not dominated by any pair in the filter. Refer to [21] and [34] for more detailed discussions of the filter method and to [17] for its application to generally constrained production optimization problems with continuous variables. We now describe the use of the filter method with MADS (this discussion follows that given by Audet and Dennis [3] for GPS).

In adapting the filter method for MADS, a filter at iteration k is defined as the set of infeasible points that are not dominated by any other points evaluated in the optimization process up to iteration k. The evaluated feasible points are considered separately and are not strictly part of the definition of a filter. At iteration k, two types of solutions are defined, as illustrated in Fig. 4a: the best feasible solution $(0, f_k^{\text{F}})$ and the closest-to-feasible or least infeasible solution in the filter, $(h_k^{\text{I}}, f_k^{\text{I}})$.



(a) Filter at the start of iteration k (b) Evaluated search/poll points at k (c) Updated filter at iteration $k+1$

Fig. 4 Illustration of the progression of a filter from iteration k to $k+1$

During polling in the MADS algorithm, either one of these solutions can be used as the poll center, with preference given to the best feasible solution. We will refer to the best feasible solution as the primary poll center and to the least infeasible point as the secondary poll center. Even if there is a best feasible solution, it can still be useful to poll around the least infeasible point in the filter. This enables the algorithm to explore a different, and possibly more promising, part of the parameter space.

In our implementation, a MADS iteration that generates an unfiltered point is considered a successful iteration. If no feasible point has been found up to the current iteration, the polling is performed around the least infeasible point in the filter. If feasible points have been found, and if an iteration in which we poll around the best feasible point (primary poll center) is unsuccessful, in the next iteration, the polling stencil size is reduced and the secondary poll center is considered. Polling around the secondary poll center continues until an unsuccessful iteration, after which the stencil size is reduced and we return to polling around the primary poll center. We use the least infeasible point as the secondary poll center rather than another filter point (with better objective function value) because the minimization of h is preferred over the optimization of f to ensure feasible solutions.

The filter is updated at successful iterations as there are new unfiltered points that dominate some of the current filter points, as illustrated in Fig. 4. At unsuccessful iterations, the mesh and poll size parameters, Δ_k^m and Δ_k^p, are decreased and the filter remains the same since there are no new unfiltered points. It is interesting to note that, at the end of the optimization (and at no extra cost), the points of the final filter give a quantitative indication of the sensitivity of the objective function to the constraints.

Filter methods, when compared to techniques that simply discard infeasible points, have the advantage of using infeasible points to enrich the search for an optimal solution. Filter methods have been combined with gradient-based methods [21] as well as with derivative-free algorithms [3, 17]. For more details on the MADS algorithm and the filter constraint treatments used in this work, see [3, 28, 32].

3.3 Global derivative-free method: Particle Swarm Optimization

PSO algorithms are a family of global stochastic search procedures introduced by Eberhart and Kennedy [15]. They are population-based methods and entail a "swarm" (population) of "particles" (potential solutions) that move through the solution space with certain "velocities." The PSO method has been applied in many application areas, including well placement optimization [36]. The behavior of PSO algorithms is dependent on a few parameters. Fernández Martínez et al. [19, 20] analyzed the stability properties of various PSO algorithms, and their work provides guidelines for choosing parameters that result in particular behaviors (e.g., more explorative versus exploitative).

The PSO algorithm involves a swarm of S particles (S is the population or swarm size). Each particle has a location in the search space and a velocity. The new position of particle j in iteration $k+1$, denoted here as $\mathbf{x}_{k+1}^j$, is determined as follows:

$$\mathbf{x}_{k+1}^j = \mathbf{x}_k^j + \mathbf{v}_{k+1}^j \Delta t \quad \forall\, j \in \{1, \ldots, S\}, \tag{8}$$

where $\mathbf{v}_{k+1}^j$ is the velocity of particle j in iteration $k+1$ and Δt is a "time" increment. Consistent with standard PSO implementations [12], we set $\Delta t = 1$. The velocity vector associated with each particle j is given by

$$\mathbf{v}_{k+1}^j = \underbrace{\omega \mathbf{v}_k^j}_{\text{inertial term}} + \underbrace{c_1 D_{k+1}^1 (\mathbf{y}_k^j - \mathbf{x}_k^j)}_{\text{cognitive term}} + \underbrace{c_2 D_{k+1}^2 (\hat{\mathbf{y}}_k^j - \mathbf{x}_k^j)}_{\text{social term}}, \tag{9}$$

where ω, c_1, and c_2 are called the inertial, cognitive, and social parameters, respectively. The matrices D_{k+1}^1 and D_{k+1}^2 are diagonal, with elements randomly drawn from a uniform distribution with range $[0, 1]$. The inertial term tends to move the particle in the direction in which it was previously moving, with the idea of continuing in a promising search direction. The cognitive term causes particle j to be attracted to its own previous best position, $\mathbf{y}_k^j$. The social

term causes each particle j to also be attracted to the best position, $\hat{\mathbf{y}}_k^j$, found through iteration k by any particle in its "neighborhood" (the definition of "best" for cases with and without constraint violation will be provided below). In our work, we use the PSO parameters recommended by Clerc [12] ($\omega = 0.729, c_1 = c_2 = 1.494$), which were shown to perform well for a suite of test problems.

The concept of "neighborhood" is used within PSO to specify the set of particles that particle j "sees," i.e., the particles to which it has information links. In a global neighborhood topology, each particle "sees" all other particles. In this case, there is a single (global best) $\hat{\mathbf{y}}_k$, given by

$$\hat{\mathbf{y}}_k = \arg\min_{\mathbf{z}\in\{\mathbf{y}_k^1,\ldots,\mathbf{y}_k^S\}} f(\mathbf{z}). \tag{10}$$

In this work, we use a random neighborhood topology [13], where particle j is linked to a probabilistically determined subset of the swarm. The linkages are altered (randomly) after iterations where there is no improvement in the best solution. This approach was found to be robust and to provide satisfactory performance for well placement problems by Onwunalu [35]. This reference should be consulted for further discussion of PSO neighborhood topologies.

Different stopping criteria can be used in the PSO algorithm. In our implementation, the optimization process is terminated after a given number of iterations is reached or when the norm of the velocities for all particles is smaller than a specified threshold. If all of the velocities are sufficiently small, this usually indicates that the diversity between the particles has been lost, meaning the swarm has collapsed.

As is the case with MADS, the PSO algorithm is easily parallelizable since, at each iteration, the evaluation of all particles in the swarm can be performed concurrently. We note that all variables in our PSO implementation are treated as continuous. We round to the nearest integer to provide discrete variables when necessary.

It is worth observing that even though the PSO algorithm attempts to search globally and does have a stochastic component (which enables it to avoid poor local optima), we cannot realistically expect to include enough particles to "cover" a high-dimensional search space. Thus, PSO should be viewed as an algorithm capable of providing some amount of global exploration in the optimization search space. In general, global search procedures such as PSO might be expected to be more computationally expensive than local optimization methods (since global approaches need to explore a much larger region of the search space). Particular implementations, such as that used for the results reported in Section 4, may however display efficient performance, possibly at the expense of less global exploration.

3.4 Constraint treatment in PSO

To satisfy bound constraints, the coordinates of a PSO-generated point that are outside the bounds are projected using Eq. 6, and the corresponding coordinates of the velocity vector for that particle are set to zero. For treating general (nonlinear) constraints, global stochastic search methods typically employ techniques that either discard infeasible solutions (and thus only consider feasible solutions), as in [26], or they apply penalty function approaches, as in [14] and [37]. In our work, we use filters for each PSO particle, as illustrated in Fig. 5, in a manner we now describe.

For problems without nonlinear constraints, the previous best position for PSO particle j and the neighborhood best solution, $\mathbf{y}_k^j$ and $\hat{\mathbf{y}}_k^j$ in Eq. 9, are determined based only on the objective function value. If the problem has nonlinear constraints, a filter is constructed from the history of each particle, as illustrated in Fig. 5. When particle j is evaluated at a new position, its filter is updated if the new position is "unfiltered." The modification to the original PSO method is mainly in the manner in which we define the previous best position for each particle and the neighborhood best, for use in Eq. 9. If particle j has been feasible in previous iterations, $\mathbf{y}_k^j$ is the feasible point with the best objective function value. If particle j has not occupied any feasible position, then $\mathbf{y}_k^j$ is taken to be the least infeasible point in the filter for particle j, as indicated by the red circles in Fig. 5. The neighborhood best position is defined as the best feasible $\mathbf{y}_k^j$ (in terms of objective function value) or, if there are no feasible previous positions, as the least infeasible point from all

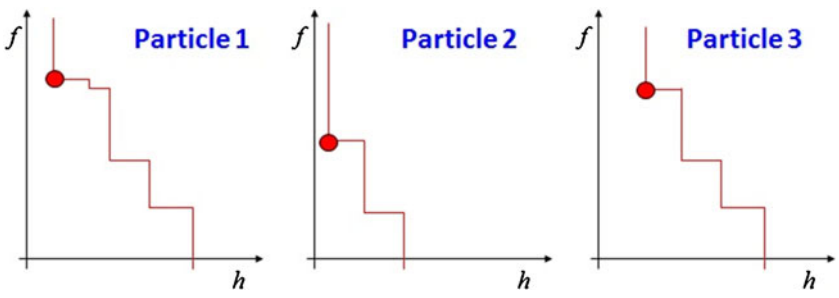


Fig. 5 Illustration of filters for three particles in a PSO swarm at iteration k, with the least infeasible point in each filter highlighted

filters in the neighborhood. In Fig. 5, the neighborhood best for the three particles would be the least infeasible point in the filter for particle 2.

3.5 Hybrid PSO–MADS procedure

Pattern search methods such as MADS are local methods that are designed to achieve convergence (from arbitrary starting points) to points that satisfy local optimality conditions. Although the use of a large initial stencil size enables some amount of global search, the MADS method is not expected to provide the same degree of global exploration as a population-based stochastic search procedure such as PSO with a reasonable swarm size. Therefore, in this work, we exploit the global search nature of PSO and the rigorous convergence to stationary points provided by MADS by creating a PSO–MADS hybrid. PSO is incorporated into the algorithm as the search step of the MADS procedure.

The hybrid used in our work is essentially an extension of the PSwarm algorithm [41], which was developed for bound constrained problems. Our hybrid implementation is different from PSwarm in that we use MADS instead of coordinate search during the polling process, and we treat nonlinear constraints using the filter-based approaches described previously. Also, we use a random neighborhood topology in the PSO stage with particle links updated after unsuccessful iterations. Figure 6 presents a detailed description of our PSO–MADS hybrid algorithm and Fig. 7 illustrates the overall workflow.

The hybrid search method begins with an initial swarm of particles, including one or more user-defined initial guesses if provided, and it then applies one iteration of PSO (using Eqs. 8 and 9). Consecutive iterations where the search step is successful are equivalent to consecutive iterations of the stand-alone PSO algorithm. In the hybrid implementation, for a PSO swarm with S particles, there are S particle filters built from the history of each particle and a main filter (denoted $\mathcal{F}^{\text{main}}$ in Fig. 6) constructed from all points evaluated in the optimization process. Note that the least infeasible point from all particle filters is the least infeasible point in the main filter.

A PSO search step is designated as successful if the global best position improves, implying generation of a new global best position that dominates the previous global best in terms of objective function f or constraint violation h. This definition of success is different than that used for MADS, where an iteration is deemed successful if any unfiltered points are found. This stricter PSO criterion is applied to avoid performing many PSO iterations during which the filter of the best particle remains unchanged (i.e., the best particle does not improve), even though the filter of a clearly suboptimal particle continues to change. This treatment acts to accelerate the convergence of the overall hybrid algorithm. Note that, as indicated in Fig. 6, in the search step, PSO uses a random neighborhood topology with local

Initialization: Choose swarm size S, stopping tolerances $\Delta_{\text{tol}} > 0$ and $v_{\text{tol}} > 0$, and initial mesh and poll sizes $\Delta_0^m = \Delta_0^p > 0$. Initialize number of PSO searches, $l = 0$, specify a maximum number of PSO search iterations, $l_{\max}$, and a maximum number of total iterations in the hybrid algorithm, $k_{\max}$. Randomly generate initial PSO particle information links (random neighborhood topology), initial swarm positions $\mathbf{x}_0^j$ and small velocities $\mathbf{v}_0^j$ with $||\mathbf{v}_0^j|| > v_{\text{tol}}\ \forall\ j \in \{1,\ldots,S\}$. Evaluate $f\left(\mathbf{x}_0^j\right)$ and $h\left(\mathbf{x}_0^j\right)$, and set $\mathbf{y}_0^j = \mathbf{x}_0^j\ \forall\ j \in \{1,\ldots,S\}$ with $\hat{\mathbf{y}}_0$ being the best feasible $\mathbf{y}_0^j$, or least infeasible $\mathbf{y}_0^j$ if there are none feasible.
Initialize particle filters $\mathcal{F}^j$ using each initial particle position, $\mathbf{x}_0^j$, and a main filter $\mathcal{F}^{\text{main}}$ using $\mathbf{x}_0^j\ \forall\ j \in \{1,\ldots,S\}$.
For $k = 0, 1, 2, \ldots, k_{\max}$, do:

1. **Search:** Skip if previous poll step was successful, if $\max_{j\in\{1,\ldots,S\}} ||\mathbf{v}_k^j|| < v_{\text{tol}}$, or if $l \geq l_{\max}$.
   $l \leftarrow l + 1$ (increment number of PSO searches). Randomly generate new PSO particle information links if previous step is unsuccessful.
   Compute $\mathbf{v}_{k+1}^j$ and $\mathbf{x}_{k+1}^j$ using equations (8) and (9). Evaluate $f\left(\mathbf{x}_{k+1}^j\right)$ and $h\left(\mathbf{x}_{k+1}^j\right)\ \forall\ j \in \{1,\ldots,S\}$. Set $\hat{\mathbf{y}}_{k+1} = \hat{\mathbf{y}}_k$.
   For $j = 1, 2, \ldots, S$, do:
   - If $\mathbf{x}_{k+1}^j$ is unfiltered w.r.t. $\mathcal{F}^j$, update $\mathcal{F}^j$, and if w.r.t. $\mathcal{F}^{\text{main}}$, update $\mathcal{F}^{\text{main}}$, then
     - set $\mathbf{y}_{k+1}^j$ to the best feasible point for particle j or least infeasible point in $\mathcal{F}^j$ if no feasible points for particle j.
     - if $\mathbf{y}_{k+1}^j \prec \hat{\mathbf{y}}_{k+1}^j$ then set $\hat{\mathbf{y}}_{k+1}^j = \mathbf{y}_{k+1}^j$ (update neighborhood best; only deem search step and iteration successful if global best, $\hat{\mathbf{y}}$, gets updated).
   - Otherwise set $\mathbf{y}_{k+1}^j = \mathbf{y}_k^j$ (deem search step unsuccessful).
2. **Poll:** Skip if successful search step performed. Choose $\hat{\mathbf{y}}_k$ as the poll center and evaluate f and h at all poll points.
   - If there exists a MADS poll direction $\mathbf{d}_k$ such that $\hat{\mathbf{y}}_k + \Delta_k^p \mathbf{d}_k$ is unfiltered w.r.t $\mathcal{F}^{\text{main}}$, then
     - if $\hat{\mathbf{y}}_k + \Delta_k^p \mathbf{d}_k \prec \hat{\mathbf{y}}_k$, then set $\hat{\mathbf{y}}_{k+1} = \hat{\mathbf{y}}_k + \Delta_k^p \mathbf{d}_k$ (update the global best particle position).
     - update filters $\mathcal{F}^{\text{main}}$ and $\mathcal{F}^{\hat{j}}$, where $\hat{j}$ is the index of the global best PSO particle.
     - set $\Delta_{k+1}^m \geq \Delta_k^m$ and $\Delta_{k+1}^p \geq \Delta_k^p$ (optionally expand mesh and poll sizes; deem poll step and iteration successful).
   - Otherwise, if all poll points $\hat{\mathbf{y}}_k + \Delta_k^p \mathbf{d}_k\ \forall\ \mathbf{d}_k$ are filtered, then
     - set $\hat{\mathbf{y}}_{k+1} = \hat{\mathbf{y}}_k$.
     - set $\mathbf{x}_{k+1} = \mathbf{x}_k$, $\Delta_{k+1}^m = \theta \Delta_k^m$ and $\Delta_{k+1}^p = \phi \Delta_k^p$ with contraction parameters $0 < \theta < \phi < 1$ (contract mesh and poll sizes; deem poll step and iteration unsuccessful).
3. If $\Delta_{k+1}^m < \Delta_{\text{tol}}$, then stop. Otherwise, $k \leftarrow k + 1$ and go to Step 1.

Fig. 6 Details of PSO–MADS hybrid algorithm

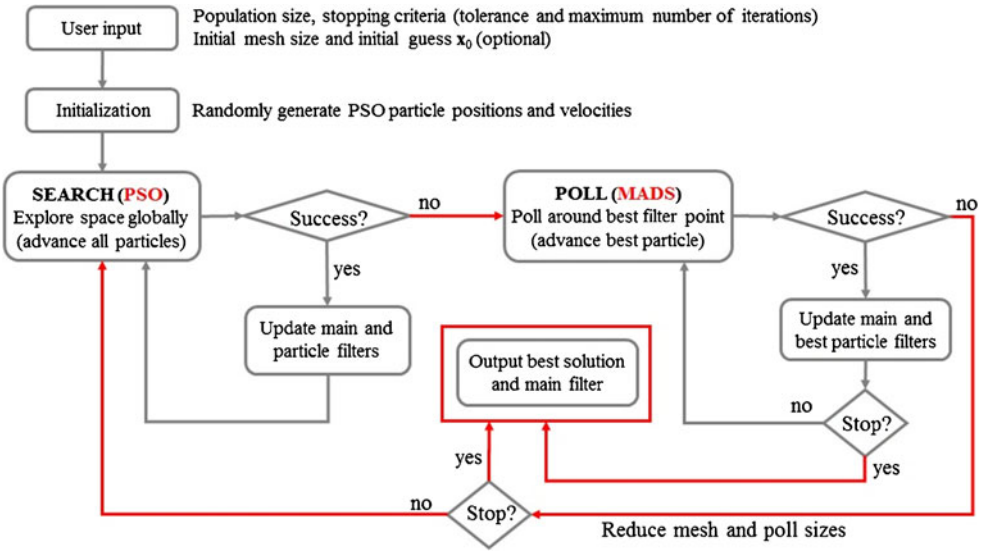


Fig. 7 Flowchart of PSO–MADS hybrid implementation (*gray arrows* indicate optimization within the PSO or MADS components and *red arrows* indicate termination of algorithm or coupling between the different components of the hybrid)

neighborhood best positions, $\hat{\mathbf{y}}^j$ (which appears in the social term of the PSO velocity equation). The definition of a successful PSO iteration is, however, based on improvement of the global best position, $\hat{\mathbf{y}}$, since subsequent MADS polling will be performed around this point.

When the search step is not successful (i.e., does not provide improvement), the MADS poll step is performed, centered on the best position from the swarm (least infeasible point or best feasible point computed for any of the particles). The polling continues as long as consecutive poll steps are successful (i.e., better feasible points or unfiltered points with respect to $\mathcal{F}^{\text{main}}$ are generated). We consider $\mathcal{F}^{\text{main}}$ here rather than the filter associated with the best particle because $\mathcal{F}^{\text{main}}$ contains more complete information on the progress of the overall search.

Consecutive iterations where the poll step is successful are equivalent to consecutive iterations of the stand-alone MADS algorithm (polling only, without a search step). At these successful MADS iterations, $\mathcal{F}^{\text{main}}$ and the particle filter corresponding to the best PSO particle, around which polling is being performed, are updated with the new unfiltered poll points. If the polling is unsuccessful, the mesh and poll sizes are reduced and the hybrid algorithm returns to the PSO search. PSO iterations are then resumed with the best particle position that (frequently) has been updated by MADS. This new global best impacts the velocities of other PSO particles through the social term in Eq. 9. The PSO search continues until an unsuccessful iteration, after which a new best particle (which could be, and indeed is in most of our runs, different from that in the previous MADS step) is used to initiate MADS. This alternation between PSO and MADS continues until the termination of the hybrid algorithm.

The PSO stage of the hybrid algorithm terminates when the particle velocities are smaller than a prescribed tolerance v_{tol} (this normally indicates that the swarm has collapsed), or when a maximum number of PSO iterations $l_{\max}$ is reached. The hybrid algorithm terminates when the mesh size decreases below a given tolerance Δ_{tol} or when a maximum number of hybrid iterations $k_{\max}$ is reached.

4 Example problems and results

The methods described in the preceding section will now be applied to solve the optimization problem defined by Eqs. 1 and 2 for a synthetic field subject to waterflooding. The geological model is represented on a two-dimensional 60×60 grid. The permeability field is shown in Fig. 8, together with an initial guess for the locations of the five wells (two injection and three production wells) used in the examples. Table 1 presents key simulation and optimization parameters.

The permeability field shown in Fig. 8 is the same as that used by Bellout et al. [6] in their example cases, though

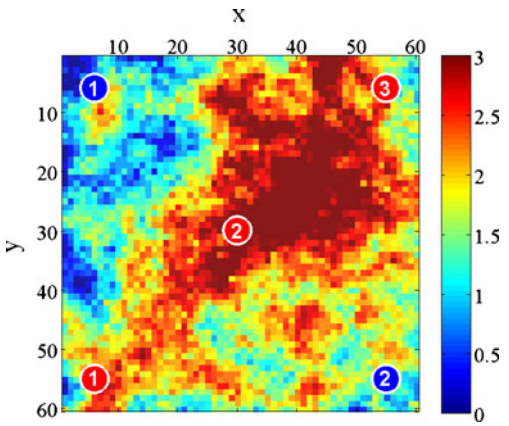


Fig. 8 Geological model ($\log_{10}$ of permeability field, with permeability expressed in millidarcys) used for all examples, showing initial-guess injection (in *blue*) and production (in *red*) well locations

Table 1 Simulation and optimization parameters

Grid cell dimensions	130 ft × 130 ft × 20 ft
Initial pressure p_i, at datum	4,012 psi at 8,620 ft
μ_o and μ_w at p_i	0.5 and 0.3 cp
ρ_o and ρ_w	53.1 and 62.4 lbm/ft^3
B_o and B_w at p_i	1.00 RB/STB
p_o, c_{pw} and c_{iw}	\$80, \$10, and \$5/STB
Drilling cost	\$20 million per well
Injection BHP range	4,100–6,000 psi
Production BHP range	1,000–3,500 psi
Maximum water injection rate	9,000 STB/day
Minimum oil production rate	4,000 STB/day
Maximum fluid production rate	9,000 STB/day
Maximum well water cut	0.7
Minimum well-to-well distance	1,300 ft

other aspects of the problem specification are different. The production time frame is 2,920 days, with the well BHPs updated every 584 days, for a total of five control intervals. The BHP is held constant over each control interval.

Results for three cases will be presented. The first case involves only bound constraints, the second case additionally incorporates the nonlinear constraints listed in Table 1 (last five rows), and the third case includes the nonlinear constraints plus binary optimization variables that allow us to also determine the optimum number of wells. The total number of optimization variables for cases 1 and 2 is 35 (two areal location variables and five control variables for each of the five wells), while in the third case, there are 40 (an additional binary variable for each well).

4.1 Case 1: bound constraints only

For this case, the nonlinear field rate and well water cut constraints are not included in the optimization. Thus, the problem involves only the bound constraints on well BHPs. The MADS, PSO, and PSO–MADS methods are applied to this problem. For each MADS iteration, a maximum of $2n$ function evaluations is performed in the polling process ($2n = 70$ for this case). We note that the evaluation of some of the poll points can be avoided if these points have already been visited in previous iterations and the objective and constraint values are stored in a cache memory. The initial MADS mesh sizes correspond to 20 % of the variable ranges.

In the stand-alone PSO iterations, a swarm size of 50 particles is used, which implies that about 50 function evaluations are performed in each PSO iteration (fewer if some positions have been previously evaluated and saved). The same parameters used for the stand-alone PSO and MADS iterations are considered for the hybrid PSO–MADS method. The function evaluations in all three methods are parallelized using a computing cluster. For these runs, we typically have about 50 processors available, so to convert from total simulations (proportional to total computational time) to equivalent simulations (proportional to actual elapsed time), the number of total simulations should be divided by 50 (we note that the overhead in the parallelization process, which leads to a true speedup that is less than 50, is not accounted for in this conversion).

Considering the stochastic nature of these algorithms and the fact that the optimization surface is expected to contain multiple optima, each of the three methods is run five times starting from five different initial guesses. The NPVs for the five initial guesses, together with their mean and standard deviation, designated <NPV> and σ, are shown in Table 2. Note that MM designates million.

The first of these runs has as the initial guess in MADS the well locations indicated in Fig. 8, with the injection well BHPs at their upper bounds (6,000 psi) and production well BHPs at their lower bounds (1,000 psi). The remaining four initial guesses for the MADS runs are randomly generated from a uniform distribution within the bounds of the problem. In this and all subsequent examples, in each of the PSO and PSO–MADS runs, the starting position for one of the particles in the initial swarm is the initial guess used in the corresponding MADS run, while the starting positions for the other particles are randomly generated.

The optimization results for the three derivative-free methods are summarized in Fig. 9 and Table 3. Figure 9 shows the NPV evolution versus the number of simulations, averaged over the five runs, for the three methods. From this figure, we see that the PSO–MADS hybrid (red curve) outperforms its component methods. Table 3 presents the final optimized NPVs for all of the runs, together with the mean and standard deviation of the optimized NPVs over the five runs. The best NPV from each method is italicized in the table.

Figure 9 and Table 3 highlight some of the characteristics of the three methods. As a result of its global search

Table 2 NPVs from the five initial guesses used in the optimizations (the best value is italicized)

Run #	Initial guess NPV [\$ MM]
1	*1,015*
2	554
3	283
4	−663
5	580
<NPV>	354
σ	626

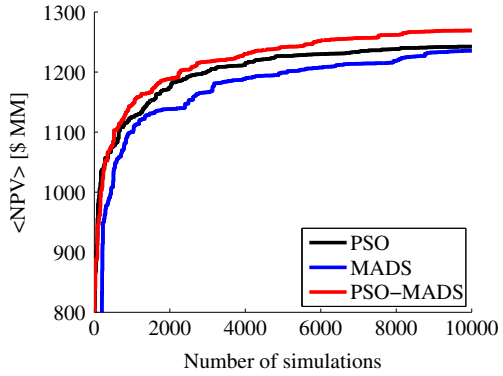


Fig. 9 Evolution of mean NPV for the five runs (case 1)

nature, PSO is able to avoid poor local optima and provide fairly robust solutions (evident from the relatively small σ in Table 3). However, in contrast to MADS, the PSO algorithm is not supported by local convergence theory and we therefore cannot guarantee that the solutions obtained satisfy any optimality conditions. MADS results, on the other hand, can depend strongly on the initial guess, and because of its local search nature, MADS may converge to poor local optima, as is the case for the second run. Since the PSO–MADS hybrid combines some of the advantages of the PSO and MADS algorithms, it displays strong convergence to better quality solutions than stand-alone MADS and PSO, along with better robustness features than PSO, which is evident from the smaller σ for PSO–MADS. We note additionally that, even though the total number of simulations required for these optimizations is around 10,000 (see Fig. 9), the number of equivalent simulations using 50 processors, which gives an indication of elapsed time, is ideally around 200.

Figure 10 shows the optimal well locations together with the injection and production well BHP controls for the best PSO–MADS solution, with NPV of $1,304 million. Comparing Figs. 8 to 10a we see that the optimized well locations are close to the initial guess locations, implying that the initial guess was quite reasonable (note that the best NPVs for all three methods correspond to the runs with this initial guess). The slight shifts in well locations, together with the modifications to the well controls shown in Fig. 10b and c, account for the observed 17 % improvement in NPV from the initial guess to the PSO–MADS optimized solution.

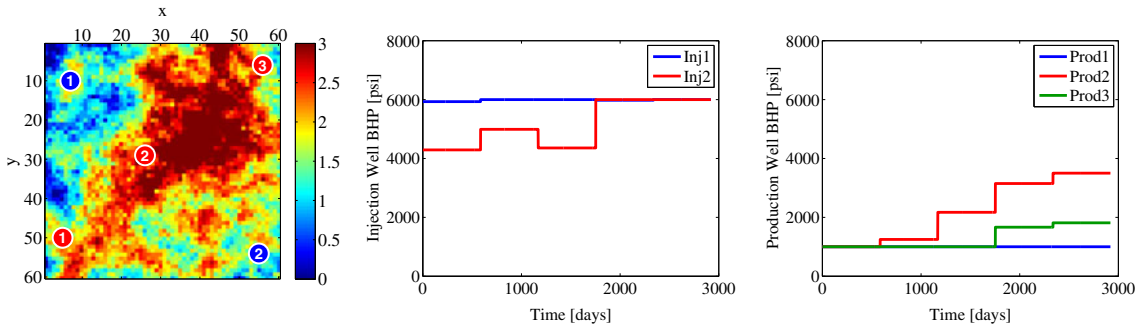
Table 3 Final NPVs from five runs for the three different methods (case 1, the best values are italicized)

Run #	PSO NPV [$ MM]	MADS NPV [$ MM]	PSO–MADS NPV [$ MM]
1	*1,283*	*1,297*	*1,304*
2	1,257	1,083	1,247
3	1,263	1,262	1,289
4	1,239	1,247	1,262
5	1,169	1,289	1,244
<NPV>	1,242	1,236	1,269
σ	44	88	26

The results presented in Fig. 9 and Table 3 are obtained by solving the combined well placement and control problem in the joint fashion proposed in this paper. In order to compare our joint approach to a sequential procedure, we also solve the problem sequentially. In this case, we first address the well placement problem (Eq. 3) using reactive controls (wells operate at their BHP limits, with production wells closed when water cut exceeds an economic limit of 0.9) and PSO–MADS with the same parameters as used for the joint solution. Then, with the optimized locations from Eq. 3, we solve the well control problem (Eq. 4), again using PSO–MADS. We run the sequential approach five times with the same initial guesses as were used for the joint optimizations.

Figure 11 displays the average performance from the sequential and joint approaches, while Table 4 presents the final optimized NPVs for all five runs, together with the mean and standard deviation of the optimized NPVs. Since the sequential method involves solving two smaller optimization problems (the well placement problem has 10 variables and the control problem has 25 variables, compared to 35 variables for the joint approach), it exhibits faster overall convergence (see Fig. 11). Despite the faster convergence, the solutions from the sequential procedure display lower NPVs on average than those from the joint approach. This is because, in contrast to the joint procedure, the sequential approach does not completely capture the coupling between the well placement and well control problems. In addition, the sequential approach appears to be less robust, as is evident from the larger σ in Table 4. Our observation that the joint approach provides better solutions than the sequential procedure is consistent with the findings reported in [6, 33], though (as noted in Section 1) Humphries et al. [27] did not observe consistent improvement with their joint formulation.

Although our joint optimization procedure differs in several respects from that applied by Bellout et al. [6], it is nonetheless instructive to compare our results with theirs. In the second example in their paper, a problem very similar to that considered above was addressed (they used the permeability field shown in Fig. 8, though they specified smaller grid block sizes and higher costs for produced and injected water, and they used more control steps). We modified our problem specification to enable solution of the same problem using our joint PSO–MADS hybrid optimization procedure. Bellout et al. [6] ran their optimization using different direct search procedures for well placement (the best results, in terms of average objective function value, were



(a) Optimized well locations (b) Optimized injection BHPs (c) Optimized production BHPs

Fig. 10 The best PSO–MADS solution, showing well locations and BHP versus time profiles (case 1)

achieved using the Hooke–Jeeves direct search method) and an adjoint-based gradient technique for well control. Their results for nine optimization runs (using different initial guesses) show a mean NPV of $363 million and a standard deviation of $27 million over the nine runs. For the same case, we achieved a mean NPV of $361 million and a standard deviation of $19 million (again over nine runs). Thus, the two procedures appear to be quite comparable in terms of the quality of results. Both of these mean NPVs are well above the mean NPV of about $300 million reported for sequential optimization in this example [6].

Bellout et al. [6] required about 4,000 simulations for each of their optimization runs, whereas we use around 30,000 simulations for our runs (note that this problem contains 10 control steps, resulting in 60 optimization variables, which are more than the 35 optimization variables in case 1). Their approach requires fewer runs because they apply an efficient gradient-based procedure for the well control optimization, with gradients computed using the adjoint technique of Sarma et al. [40]. Although it is very efficient, the adjoint approach is simulator invasive and does not readily parallelize. This means that part of their nested approach can be easily parallelized (the well placement part, solved with direct search procedures) and part of it cannot (the well control part, solved with a gradient-based technique). Our method, by contrast, is fully parallel and would effectively lead, if 60 computing cores were available on a cluster, to 500 equivalent simulations instead of 30,000.

The approaches used by Bellout et al. [6] and Li and Jafarpour [33] essentially decompose the joint problem into two smaller subproblems. This enables the application of a specialized optimization method for each of the two subproblems, e.g., an adjoint-based gradient technique can be used for the well control subproblem. These decomposition approaches can be advantageous in some cases, particularly if the adjoint-based optimization has been parallelized. They typically entail increased code complexity relative to that associated with single-level approaches, however, since a second optimization must be nested within the higher-level optimization. In addition, parallelizing the adjoint-based optimization code is far more challenging than the parallelization associated with the derivative-free optimization algorithms applied in this work. In any event, there are clearly relative advantages and disadvantages between methods that apply decomposition and the single-level procedure considered here. The choice of approach will depend on the number of available computing cores, the availability

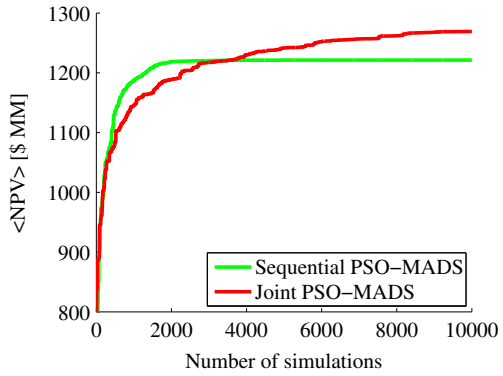


Fig. 11 Evolution of mean NPV from five PSO–MADS runs for sequential and joint procedures (case 1)

Table 4 Final NPVs from five runs for sequential and joint procedures (case 1, the best values are italicized)

Run #	PSO–MADS seq. NPV [$ MM]	PSO–MADS joint NPV [$ MM]
1	*1,290*	*1,304*
2	1,235	1,247
3	1,132	1,289
4	1,205	1,262
5	1,245	1,244
<NPV>	1,221	1,269
σ	59	26

of an adjoint procedure for the problem under consideration, and the amount of computation associated with the two subproblems.

4.2 Case 2: nonlinear constraints

Case 1 above dealt with only bound constraints. We now treat a case that also includes nonlinear constraints, which render the problem more difficult. The nonlinear constraints considered are the well distance, field rate, and well water cut constraints listed in Table 1. The rate and water cut constraints are nonlinear in nature because the relationship between individual well BHPs (the control variables) and the field rates and well water cuts involve reservoir simulation (i.e., nonlinear function evaluations). We use the same PSO, MADS, and PSO–MADS parameters as in case 1. The three algorithms, together with the filter constraint handling techniques implemented for each method (as described in Sections 3.2 and 3.4), are applied to solve the joint well placement and control problem. We again run each method five times using the same initial guesses as in case 1 (recall that one initial guess involves the well locations shown in Fig. 8).

Figure 12 displays the evolution of mean NPV for the feasible solutions for the three methods tested, and the results for all runs are summarized in Table 5. The curves in Fig. 12 do not appear until several thousand simulations have been performed because the initial guesses, with NPVs shown in Table 2, and the earlier simulations lead to infeasible solutions (i.e., solutions that violate the nonlinear constraints). From the results in Fig. 12 and Table 5, it is again apparent that the PSO–MADS procedure outperforms its component methods. Note that the optimized NPVs achieved in this case are lower, for all runs, than those for case 1 (shown in Table 3). This is consistent with the fact that this case involves a more constrained problem.

It is important to note that, even though the early solutions using all three methods are infeasible, through use of the filter method, we improve the objective function value and reduce the constraint violation simultaneously. Hence, by the time a feasible solution is found, its NPV is already relatively high. Specifically, for this case, the mean NPV for the five initial guesses (all of which are infeasible) is $354 million, while for all three algorithms, the mean NPV when feasible solutions appear (after about 4,000 simulations) is in the range of $950–$1100 million. This observation highlights the effectiveness of the filter constraint handling techniques applied in this work.

Table 5 Final NPVs from five runs for the three different methods (case 2, the best values are italicized)

Run #	PSO NPV [$ MM]	MADS NPV [$ MM]	PSO–MADS NPV [$ MM]
1	*1,225*	1,143	1,206
2	1,060	1,032	*1,228*
3	1,063	972	1,154
4	1,138	1,076	1,200
5	1,134	*1,157*	1,120
<NPV>	1,124	1,076	1,182
σ	68	77	44

The final filter from the best PSO–MADS joint optimization run is shown in Fig. 13. Before aggregation into $h(\mathbf{x})$ in Eq. 7, each of the nonlinear inequality constraints is normalized as follows:

$$c_j(\mathbf{x}) \le c_{j,\max} \quad \text{becomes} \quad \bar{c}_j(\mathbf{x}) = \frac{c_j(\mathbf{x})}{c_{j,\max}} - 1 \le 0, \qquad (11)$$

where $c_{j,\max}$ is the constraint limit for constraint c_j (e.g., maximum field water injection rate limit of 9,000 STB/day in Table 1). Consistent with Table 5, the best feasible solution has an NPV of $1,228 million. The points in the plot constitute the infeasible points that define the filter. This filter illustrates the trade-off between the objective function,

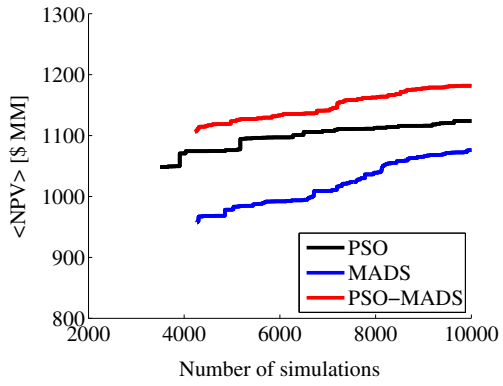


Fig. 12 Evolution of mean NPV for the five runs (case 2)

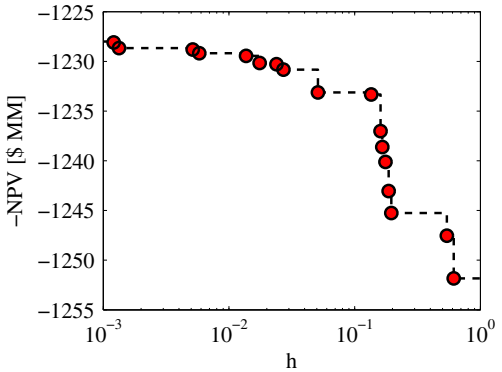


Fig. 13 Final filter, excluding points dominated by the best feasible solution, from the best PSO–MADS joint optimization run (case 2)

$f(\mathbf{x})$, and the aggregate constraint violation, $h(\mathbf{x})$. It quantifies how much the constraints need to be relaxed in order to improve the objective function by a certain amount.

In this example, for most of the filter points with small $h(\mathbf{x})$ values, only the minimum field oil production is violated. The filter in Fig. 13 shows that if we are able to accommodate a constraint violation of about 0.2 or 20 %, i.e., we allow the minimum oil production rate to be relaxed from 4,000 to 3,200 STB/day, there is a field development scenario where the NPV increases from $1,228 million to $1,245 million while satisfying the relaxed constraints. This increase in NPV occurs because the algorithm has greater flexibility in maximizing the objective. In this particular case, it may seem counterintuitive to have NPV increase when the minimum oil production rate constraint is decreased. However, in order to maintain oil production rates above 4,000 STB/day at all times, less oil is produced at earlier times. When the minimum oil constraint limit is decreased to 3,200 STB/day, higher oil rates are possible at earlier times, and this leads to an overall increase in NPV. Note that this increase is not due to a discount rate effect (since we are optimizing undiscounted NPV), but rather to the fact that, in a less-constrained problem, the optimizer has more flexibility in maximizing NPV.

As in case 1, we also performed sequential (rather than joint) optimizations for case 2, using the same optimization parameters. In these runs, a maximum water cut of 0.9 is used within the simulator. This limit triggers the "reactive control" strategy, which is active during the well placement portion of the sequential optimization. Within the optimizer, however, the maximum water cut constraint is still 0.7, consistent with the joint optimization results above. Different limits are used in the simulator and the optimizer because, with the filter constraint handling, the actual water cut constraint of 0.7 can be violated during the course of the optimization. By setting a limit within the simulator, we stipulate that the maximum water cut that can occur during the optimization is 0.9. We emphasize that all feasible solutions do satisfy the maximum water cut constraint of 0.7.

The PSO–MADS procedure was applied for both the well placement and well control stages. Figure 14 and Table 6 provide comparisons of the joint and sequential approaches. Four of the five sequential runs yield feasible solutions, with a mean NPV of $1,105 million and a standard deviation of $116 million. This is inferior in terms of average performance and robustness to the results achieved using the joint approach (<NPV> of $1,182 million, σ of $44 million). In addition, the best of the five solutions for the joint approach has an NPV that is 3.5 % higher than that from the best solution for the sequential approach. In Fig. 15, we present maps of the final oil saturation from the best solutions for both approaches. The well configurations are different between

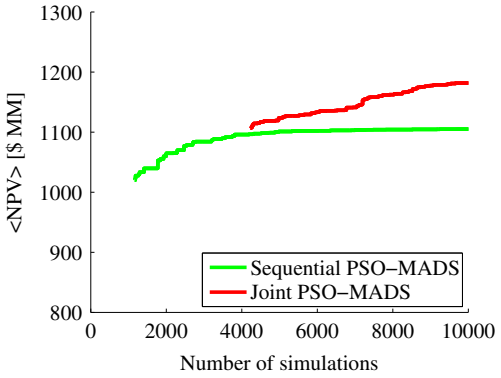


Fig. 14 Evolution of mean NPV from five PSO–MADS runs for sequential and joint procedures (case 2)

the two solutions, and it is evident that the solution for the joint approach provides a slightly better overall sweep (compare, e.g., the saturation fields in the upper right corners in both figures).

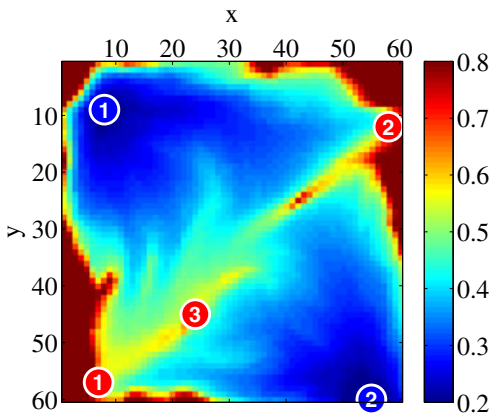
The individual well rates for the best PSO–MADS solution for the joint approach are presented in Fig. 16. These rates are the simulator outputs that result from the optimized well BHPs. Figure 16a shows the well rates for the two injection wells depicted in Fig. 15b, and 16b displays the oil production (solid lines) and water production (dashed lines) rates for the three production wells. From these plots, we see that the two injection wells operate at somewhat similar rates, while the production well rates vary significantly.

4.3 Case 3: determination of optimum number of wells

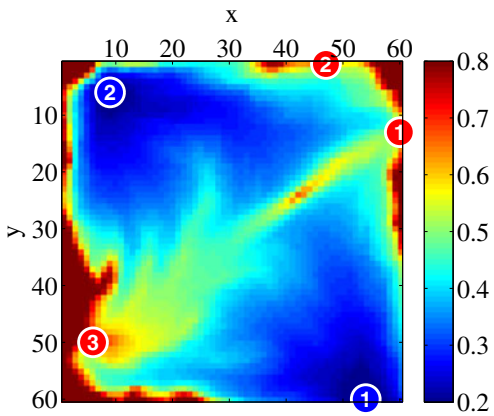
It is evident from the results in Fig. 16 that some wells operate at much higher rates than others. This leads us to question whether all five wells are required and suggests that we optimize the number of wells along with well locations and controls. To this end, we introduce a binary (drill/do not drill) optimization variable, z_j, for each well j, as in [16], in addition to the well placement and control variables. If z_j is equal to 0, then well j is not drilled, and the

Table 6 Final NPVs from five runs for sequential and joint procedures (case 2, the best values are italicized)

Run #	PSO–MADS seq. NPV [$ MM]	PSO–MADS joint NPV [$ MM]
1	*1,187*	1,206
2	–	*1,228*
3	1,107	1,154
4	941	1,200
5	1,186	1,120
<NPV>	1,105	1,182
σ	116	44

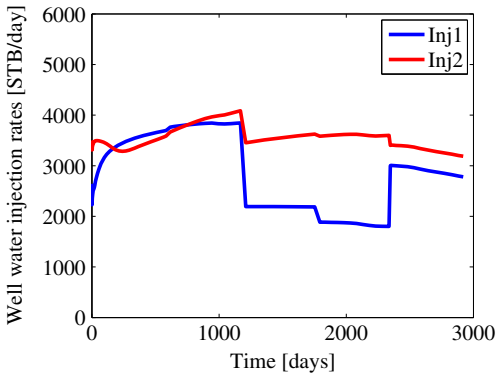


(a) Oil saturation from sequential approach

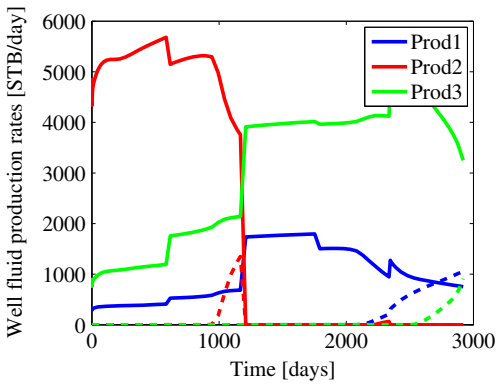


(b) Oil saturation from joint approach

Fig. 15 Final oil saturation maps (*red* indicates oil and *blue* indicates water) and well configurations from the best PSO–MADS solutions for the sequential and joint approaches (case 2)



(a) Water injection rates



(b) Oil and water production rates (solid and dashed lines, respectively)

Fig. 16 Injection and production rates for the best PSO–MADS solution for the joint approach (case 2)

well location and control variables corresponding to well j have no meaning and are ignored. In this case, we do not incur a well cost and there are no injected or produced fluids for that well. However, if $z_j = 1$, the well is drilled, the well cost is incurred, and there will be injected or produced fluids from the well.

Including the binary categorical variables, the problem in Eq. 5 is now a true mixed-integer nonlinear programming (MINLP) problem. Such problems are difficult to solve, especially when the objective and constraint functions are computationally expensive to evaluate, as is the case here. In this work, we present an initial approach for addressing this problem in the context of field development. Existing MINLP solution approaches, such as Branch and Bound and Outer Approximation [7], could also be considered. Here, we simply apply the integer treatment already implemented in our derivative-free optimization strategy (restriction to integer mesh in MADS and variable rounding in PSO). We now have $N_I + N_P$ binary variables (drill/do not drill) in addition to the $n_1 + n_2$ well location and control variables.

Case 3, which we now consider, is identical to case 2 in terms of problem specification and bound and nonlinear constraints, but it also includes binary categorical variables. We now have a total of 40 optimization variables: 10 integer well location variables, 25 continuous well control variables, and 5 binary categorical variables. The three joint optimization algorithms (PSO, MADS, and PSO–MADS) are again each run five times from different initial guesses (one user-supplied and four randomly generated, which are different from those used in cases 1 and 2 as we now have binary variables). The results from these runs are presented in Fig. 17 and Table 7. In this table, in addition to the final NPVs, we also show the optimized number of wells for each run.

We again see that the PSO–MADS hybrid outperforms its component methods in terms of mean NPV, best solution, and robustness. In addition, it is interesting to note that the best results are obtained when only four wells are drilled instead of five, leading us to believe that for this case, the optimal number of wells is indeed four (two injectors and two producers). Of the five runs for each method, the MADS algorithm converges to a solution with four wells once, the

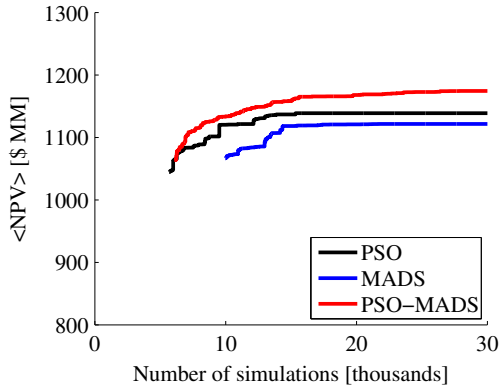


Fig. 17 Evolution of mean NPV for the five runs (case 3)

PSO algorithm twice, and the hybrid PSO–MADS method four times.

The inclusion of the binary variables leads to a more difficult optimization problem, as is apparent from the fact that about two to three times more simulation runs are required in case 3 than in case 2 (see Figs. 12 and 17). Although the NPV for the best run (run 1) in case 3 exceeds that of the best run in case 2, it is evident that some of the solutions in case 2 display better NPVs than some of the solutions in case 3. This again reflects the difficulty of this optimization and might suggest that rather than performing the optimization with binary (drill/do not drill) variables, we instead perform a sequence of optimizations with different numbers of wells specified. This sequence of optimizations can be seen as a different strategy, which evaluates all of the binary variable combinations instead of explicitly including these variables in a single optimization.

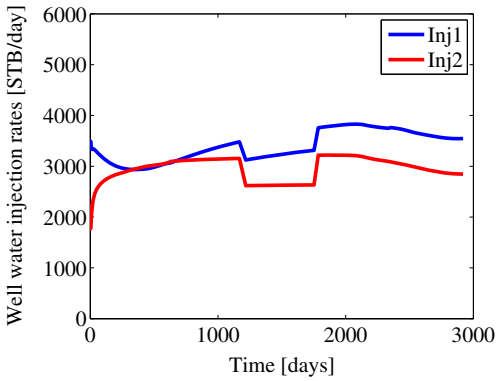
The problem with such an approach is that, even with only five wells and a maximum of three producers and two injectors, there are multiple combinations to consider (for this example, there are nine relevant combinations—two producers and one injector, one producer and two injectors, etc.). In cases with many wells, this type of exhaustive approach may become much more expensive than the explicit inclusion of binary variables. We note that more systematic optimization procedures, such as Branch and Bound, can be applied for this problem. Branch and Bound is, however, more computationally demanding than our approach here. See Isebor [28] and Isebor et al. [29] for more discussion of this issue and for optimization results comparing Branch and Bound to PSO–MADS.

The best PSO–MADS solution (from case 3, run 1) has injection and production rates as shown in Fig. 18. Comparing the rates in Fig. 18 to those in Fig. 16, we see that the four-well solution has some attractive features, i.e., the oil rates are less variable and water breakthrough is delayed. The optimized well configurations, with final oil saturation maps, for the worst, median, and best case 3 PSO–MADS solutions, together with the initial guess configurations used for these runs, are presented in Fig. 19. From these figures, we see that even with initial guesses that lead to poor sweep

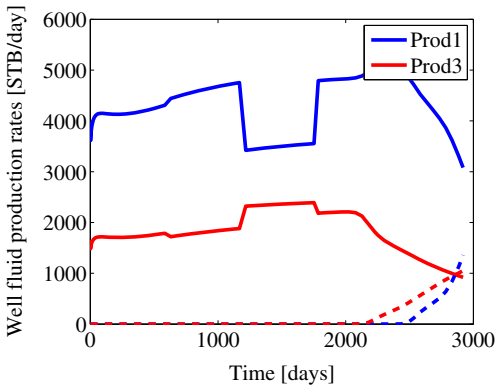
Table 7 Final NPVs from five runs for the three different methods (case 3, the best values are italicized)

Run #	PSO NPV [$ MM] (# of wells)	MADS NPV [$ MM] (# of wells)	PSO–MADS NPV [$ MM] (# of wells)
1	1,138 (5)	*1,223* (4)	*1,247* (4)
2	*1,197* (4)	1,127 (5)	1,162 (4)
3	1,151 (4)	1,073 (5)	1,158 (4)
4	1,111 (5)	1,194 (5)	1,143 (5)
5	1,096 (5)	991 (5)	1,161 (4)
<NPV>	1,139	1,122	1,174
σ	39	94	41

Number of wells for each run is shown in parentheses

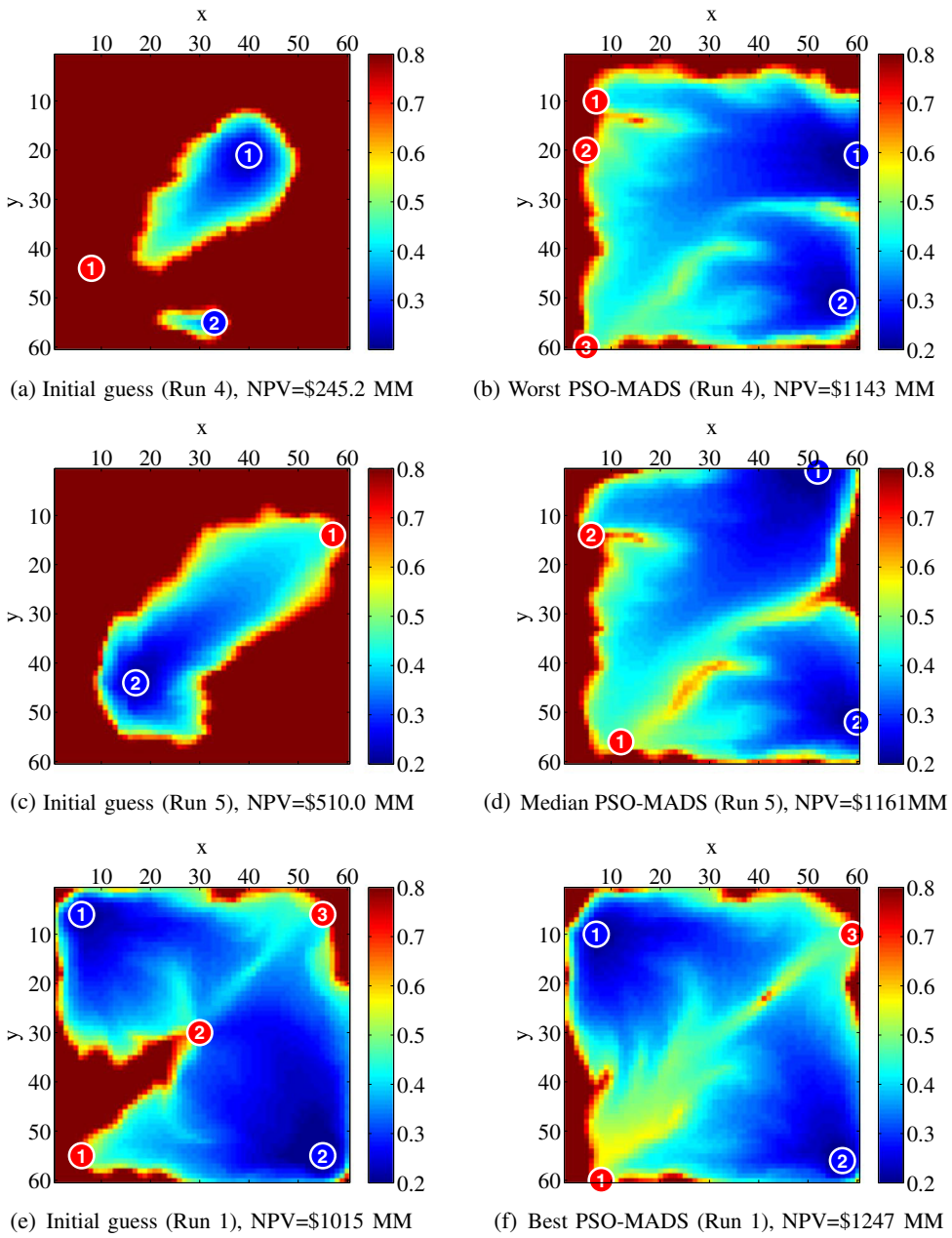


(a) Water injection rates



(b) Production rates for oil (solid lines) and water (dashed lines)

Fig. 18 Injection and production rates for the best PSO–MADS solution (case 3)



(a) Initial guess (Run 4), NPV=$245.2 MM

(b) Worst PSO-MADS (Run 4), NPV=$1143 MM

(c) Initial guess (Run 5), NPV=$510.0 MM

(d) Median PSO-MADS (Run 5), NPV=$1161MM

(e) Initial guess (Run 1), NPV=$1015 MM

(f) Best PSO-MADS (Run 1), NPV=$1247 MM

Fig. 19 Well configurations and final oil saturation maps (*red* indicates oil, *blue* indicates water) from the initial guesses and optimized solutions for the worst, median, and best PSO–MADS runs (case 3)

and NPV, the PSO–MADS procedure is able to optimize the number of wells and their associated locations and controls to provide reasonable results. Note that one of the initial guesses contains only two wells, though the corresponding optimized solution has four wells. Comparing Figs. 15b and 19f, we see that by including the binary variables in the optimization, we obtain a four-well solution that is comparable to the five-well solution in case 2, except that one of the production wells in the upper right portion of the model has been eliminated. Nonetheless, the sweep in the two cases appears quite comparable.

The NPV from the best PSO–MADS solution improves from $1,228 million for case 2 (five wells) to $1,247 million for case 3 (four wells). This increase in NPV of $19 million corresponds closely to the cost of drilling one well ($20 million). For case 2, the best PSO–MADS solution yields a cumulative oil production of 17.9 million STB and cumulative water injection of 18.7 million STB; for case 3, the corresponding values are 17.8 million STB of oil production and 18.5 million STB of water injection. Thus, the fluid injection and production volumes for the two cases are quite similar, though the solution in case 3 accomplishes this

oil recovery with one less well. This example illustrates the benefits that can be achieved by optimizing the number of wells along with the location and control variables.

5 Concluding remarks

In this paper, we presented methods for the joint optimization of well locations and controls (BHPs). Our focus was on the use of noninvasive derivative-free methods that parallelize naturally. The methods considered include a local direct-search optimization method (MADS), a stochastic global search procedure (PSO), and a hybrid technique (PSO–MADS) that combines the advantages of both methods. We applied a filter-based treatment for nonlinear constraint handling. With this approach, the problem is viewed as a biobjective optimization in which we seek to minimize both the objective function and the constraint violation. We additionally incorporated a binary variable (drill/do not drill) that enables the determination of the optimum number of wells.

We presented three example cases of increasing complexity involving multiple vertical wells, several well control periods, and different types of constraints. All of the optimizations were performed in a distributed computing environment. For all cases, the PSO–MADS hybrid method was shown to outperform the stand-alone MADS and PSO approaches, both in terms of average NPV and standard deviation of NPV over multiple runs, demonstrating that the hybrid algorithm does indeed improve upon its component methods. The stand-alone PSO procedure was observed to provide better solutions in terms of cost function value, using fewer function evaluations, than the stand-alone MADS algorithm. This may be related to the particular implementations considered. For the first two examples, we included comparisons against a sequential method (where we first optimize well locations and then well controls), and the joint approach was shown to provide superior results. The last example was a nonlinearly constrained problem that additionally included the determination of the optimum number of wells. In this case, the optimization provided an improvement in NPV by eliminating one of the production wells that was included in the previous examples (where the number of wells was fixed). Our treatment for optimizing the number of wells represents an initial approach to the MINLP problem, and the use of more sophisticated MINLP solution techniques should be investigated.

There are a number of other areas in which future research may be directed. It will be of interest to apply our joint optimization procedures to realistic three-dimensional models. This should not introduce conceptual challenges, but will result in longer computation times. It may therefore be useful to introduce some type of surrogate modeling procedure. Existing approaches include the use of kriging [30], quadratic approximations [23, 38], and reduced-order numerical models [11]. The study of alternative local and global search algorithms and implementations might improve our understanding of the performance behavior observed for different optimization procedures.

It is also important to explore the use of more rigorous MINLP approaches, such as Branch and Bound [7], to solve the MINLP problem given by Eq. 5. This will require the relaxation of the categorical variables (i.e., their treatment as continuous variables), together with the evaluation of different strategies for this relaxation. Some work in this direction is presented in Isebor [28] and Isebor et al. [29]. Finally, the incorporation of geological uncertainty in the optimization deserves attention. Techniques along the lines of the Retrospective Optimization procedure introduced by Wang et al. [43] could be applied for this purpose.

Acknowledgments We thank the industrial affiliates of the Stanford Smart Fields Consortium for partial funding of this work. We also acknowledge the Stanford Center for Computational Earth & Environmental Science (CEES) for providing the computational resources used in this research.

References

1. Almeida, L.F., Tupac, Y.J., Lazo Lazo, J.G., Pacheco, M.A., Vellasco, M.M.B.R.: Evolutionary optimization of smart-wells control under technical uncertainties. Paper SPE 107872 presented at the Latin American & Caribbean petroleum engineering conference, Buenos Aires, Argentina (2007)
2. Audet, C., Dennis Jr., J.E.: Analysis of generalized pattern searches. SIAM J. Optim. **13**(3), 889–903 (2002)
3. Audet, C., Dennis Jr., J.E.: A pattern search filter method for nonlinear programming without derivatives. SIAM J. Optim. **14**(4), 980–1010 (2004)
4. Audet, C., Dennis Jr., J.E.: Mesh adaptive direct search algorithms for constrained optimization. SIAM J. Optim. **17**(1), 188–217 (2006)
5. Bangerth, W., Klie, H., Wheeler, M.F., Stoffa, P.L., Sen, M.K.: On optimization algorithms for the reservoir oil well placement problem. Comput. Geosci. **10**(3), 303–319 (2006)
6. Bellout, M.C., Echeverría Ciaurri, D., Durlofsky, L.J., Foss, B., Kleppe, J.: Joint optimization of oil well placement and controls. Comput. Geosci. **16**(4), 1061–1079 (2012)
7. Bonami, P., Biegler, L.T., Conn, A.R., Cornuéjols, G., Grossmann, I.E., Laird, C.D., Lee, J., Lodi, A., Margot, F., Sawaya, N., Wächter, A.: An algorithmic framework for convex mixed integer nonlinear programs. Discret. Optim. **5**(2), 186–204 (2008)
8. Brouwer, D.R., Jansen, J.D.: Dynamic optimization of waterflooding with smart wells using optimal control theory. SPE J. **9**(4), 391–402 (2004)
9. Cameron, D.A., Durlofsky, L.J.: Optimization of well placement, CO_2 injection rates, and brine cycling for geological carbon sequestration. Int. J. Greenhouse Gas Control. **10**, 100–112 (2012)
10. Cao, H.: Development of techniques for general purpose simulators. Ph.D. thesis, Department of Petroleum Engineering, Stanford University (2002)

11. Cardoso, M.A., Durlofsky, L.J.: Linearized reduced-order models for subsurface flow simulation. J. Comput. Phys. **229**, 681–700 (2010)
12. Clerc, M.: The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation, pp. 1951–1957 (1999)
13. Clerc, M.: Particle Swarm Optimization. ISTE, London (2006)
14. Deb, K.: An efficient constraint handling method for genetic algorithms. Comput. Methods Appl. Mech. Eng. **186**(2–4), 311–338 (2000)
15. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micromachine and Human Science, pp. 39–43 (1995)
16. Echeverría Ciaurri, D., Conn, A.R., Mello, U.T., Onwunalu, J.E.: Integrating mathematical optimization and decision making in intelligent fields. Paper SPE 149780 presented at the SPE intelligent energy conference and exhibition, Utrecht, The Netherlands (2012)
17. Echeverría Ciaurri, D., Isebor, O.J., Durlofsky, L.J.: Application of derivative-free methodologies for generally constrained oil production optimization problems. Int. J. Math. Model. Numer. Optim. **2**(2), 134–161 (2011)
18. Echeverría Ciaurri, D., Mukerji, T., Durlofsky, L.J.: Derivative-free optimization for oil field operations. In: Yang, X.S., Koziel, S. (eds.) Computational Optimization and Applications in Engineering and Industry, Studies in Computational Intelligence, pp. 19–55. Springer, New York (2011)
19. Fernández Martínez, J.L., García Gonzalo, E.: The generalized PSO: a new door to PSO evolution. J. Artif. Evol. Appl. **2008**, 1–15 (2008)
20. Fernández Martínez, J.L., García Gonzalo, E., Fernández Alvarez, J.P.: Theoretical analysis of particle swarm trajectories through a mechanical analogy. Int. J. Comput. Intell. Res. **4**(2), 93–104 (2008)
21. Fletcher, R., Leyffer, S., Toint, P.: A brief history of filter methods. Tech. rep. Mathematics and Computer Science Division, Argonne National Laboratory (2006)
22. Forouzanfar, F., Li, G., Reynolds, A.C.: A two-stage well placement optimization method based on adjoint gradient. Paper SPE 135304 presented at the SPE annual technical conference and exhibition, Florence, Italy (2010)
23. Forouzanfar, F., Reynolds, A.C., Li, G.: Optimization of the well locations and completions for vertical and horizontal wells using a derivative-free optimization algorithm. J. Pet. Sci. Eng. **86–87**, 272–288 (2012)
24. Guyaguler, B., Horne, R.N.: Uncertainty assessment of well-placement optimization. SPE J. **7**(1), 24–32 (2004)
25. Harding, T.J., Radcliffe, N.J., King, P.R.: Optimization of production strategies using stochastic search methods. Paper SPE 35518 presented at the European 3-D reservoir modelling conference, Stavanger, Norway (1996)
26. Hu, X., Eberhart, R.: Solving constrained nonlinear optimization problems with particle swarm optimization. In: Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics (2002)
27. Humphries, T.D., Haynes, R.D., James, L.A.: Simultaneous optimization of well placement and control using a hybrid global-local strategy. In: Proceedings of the 13th European Conference on the Mathematics of Oil Recovery, Biarritz, France (2012)
28. Isebor, O.J.: Derivative-free optimization for generalized oil field development. Ph.D. thesis, Department of Energy Resources Engineering, Stanford University (2013)
29. Isebor, O.J., Echeverría Ciaurri, D., Durlofsky, L.J.: Generalized field development optimization using derivative-free procedures. Paper SPE 163631 presented at the SPE reservoir simulation symposium, The Woodlands, Texas (2013)
30. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. J. Glob. Optim. **13**(4), 455–492 (1998)
31. Kolda, T.G., Lewis, R.M., Torczon, V.: Optimization by direct search: new perspectives on some classical and modern methods. SIAM Rev. **45**(3), 385–482 (2003)
32. Le Digabel, S.: Algorithm 909: NOMAD: Nonlinear optimization with the MADS algorithm. ACM Trans. Math. Softw. **37**(4), 44:1–44:15 (2011)
33. Li, L., Jafarpour, B.: A variable-control well placement optimization for improved reservoir development. Comput. Geosci. **16**(4), 871–889 (2012)
34. Nocedal, J., Wright, S.J.: Numerical Optimization, 2nd edn. Springer, New York (2006)
35. Onwunalu, J.E.: Optimization of field development using particle swarm optimization and new well pattern descriptions. Ph.D. thesis, Department of Energy Resources Engineering, Stanford University (2010)
36. 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)
37. Parsopoulos, K.E., Vrahatis, M.N.: Particle swarm optimization method for constrained optimization problems. In: Proceedings of the Euro-International Symposium on Computational Intelligence (2002)
38. Powell, M.J.D.: The BOBYQA algorithm for bound constrained optimization without derivatives. Tech. rep., Department of Applied Mathematics and Theoretical Physics, University of Cambridge (2009)
39. Sarma, P., Chen, W.H.: Efficient well placement optimization with gradient-based algorithm and adjoint models. Paper SPE 112257 presented at the SPE intelligent energy conference and exhibition, Amsterdam, The Netherlands (2008)
40. 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)
41. Vaz, A.I., Vicente, L.N.: A particle swarm pattern search method for bound constrained global optimization. J. Glob. Optim. **39**(2), 197–219 (2007)
42. Wang, C., Li, G., Reynolds, A.C.: Optimal well placement for production optimization. Paper SPE 111154 presented at the SPE Eastern regional meeting, Lexington, Kentucky (2007)
43. Wang, H., Echeverría Ciaurri, D., Durlofsky, L.J., Cominelli, A. SPE J. **17**(1), 112–121 (2012)
44. Yeten, B., Durlofsky, L.J., Aziz, K.: Optimization of nonconventional well type, location and trajectory. SPE J. **8**(3), 200–210 (2003)
45. Zandvliet, M., Handels, M.: Adjoint-based well-placement optimization under production constraints. SPE J. **13**(4), 392–399 (2008)
46. Zhang, K., Li, G., Reynolds, A.C., Yao, J., Zhang, L.: Optimal well placement using an adjoint gradient. J. Pet. Sci. Eng. **73**(3–4), 220–226 (2010)