# Multifidelity Genetic Transfer: An Efficient Framework for Production Optimization

**Faliang Yin,** China University of Petroleum, East China; **Xiaoming Xue,** City University of Hong Kong; **Chengze Zhang,** Exploration and Development Research Institute of PetroChina Tarim Oilfield Company; **Kai Zhang*,** China University of Petroleum, East China; **Jianfa Han,** Exploration and Development Research Institute of PetroChina Tarim Oilfield Company; and **BingXuan Liu, Jian Wang,** and **Jun Yao,** China University of Petroleum, East China

## Summary

Production optimization led by computing intelligence can greatly improve oilfield economic effectiveness. However, it is confronted with huge computational challenge because of the expensive black-box objective function and the high-dimensional design variables. Many low-fidelity methods based on simplified physical models or data-driven models have been proposed to reduce evaluation costs. These methods can approximate the global fitness landscape to a certain extent, but it is difficult to ensure accuracy and correlation in local areas. Multifidelity methods have been proposed to balance the advantages of the two, but most of the current methods rely on complex computational models. Through a simple but efficient shortcut, our work aims to establish a novel production-optimization framework using genetic transfer learning to accelerate convergence and improve the quality of optimal solution using results from different fidelities. Net present value (NPV) is a widely used standard to comprehensively evaluate the economic value of a strategy in production optimization. On the basis of NPV, we first established a multifidelity optimization model that can synthesize the reference information from high-fidelity tasks and the approximate results from low-fidelity tasks. Then, we introduce the concept of relative fidelity as an indicator for quantifying the dynamic reliability of low-fidelity methods, and further propose a two-mode multifidelity genetic transfer learning framework that balances computing resources for tasks with different fidelity levels. The multitasking mode takes the elite solution as the transfer medium and forms a closed-loop feedback system through the information exchange between low- and high-fidelity tasks in parallel. Sequential transfer mode, a one-way algorithm, transfers the elite solutions archived in the previous mode as the population to high-fidelity domain for further optimization. This framework is suitable for population-based optimization algorithms with variable search direction and step size. The core work of this paper is to realize the framework by means of differential evolution (DE), for which we propose the multifidelity transfer differential evolution (MTDE). Corresponding to multitasking and sequential transfer in the framework, MTDE includes two modes, transfer based on base vector (b-transfer) and transfer based on population (p-transfer). The b-transfer mode incorporates the unique advantages of DE into fidelity switching, whereas the p-transfer mode adaptively conducts population for further high-fidelity local search. Finally, the production-optimization performance of MTDE is validated with the egg model and two real field cases, in which the black-oil and streamline models are used to obtain high- and low-fidelity results, respectively. We also compared the convergence curves and optimization results with the single-fidelity method and the greedy multifidelity method. The results show that the proposed algorithm has a faster convergence rate and a higher-quality well-control strategy. The adaptive capacity of p-transfer is also demonstrated in three distinct cases. At the end of the paper, we discuss the generalization potential of the proposed framework.

## Introduction

Production optimization is supposed to optimize an appropriate development strategy of well sets, including location, bottomhole pressure, flow rates, and conversion scheme, to maximize the NPV or accumulative oil production (Zhang et al. 2020a, 2020b; Zhao et al. 2020c). It has become an indispensable process in oil/gasfield development in the context of continuous fluctuations in international oil prices and declining production of many large oil fields. Moreover, with the technological developments of digital and smart fields, the economic benefits achieved by production optimization are increasingly considerable.

However, because of the complexity of design variables and reservoir heterogeneity, production optimization is challenged by the high dimensionality and multimodal of the optimization objective function (Zhao et al. 2020a, 2020b). Commercial numerical simulators with full-scale reservoir models are often used to build correlations between well-control settings and oilfield-development outcomes; nevertheless, it means that the gradient information required for optimization cannot be obtained from the packaged software. Derivative-free stochastic global optimization algorithms such as genetic algorithm and particle-swarm optimization are frequently applied to overcome this obstacle. Using stochastic and evolving operators, this kind of algorithm can search for the global optimum by means of the objective-function value instead of calculating or approximating the gradient information. However, the run time of a single simulation can take hours or even tens of hours as the scale of reservoir increases and, moreover, this time-consuming step needs to be performed thousands of times during the global optimization process. Therefore, improving the speed of searching for the optimal scheme has been thought of as an essential work in production optimization.

The black-box system derived from the finite-difference numerical-simulation method with the meshed reservoir model is the most widely used evaluation scheme in production optimization. However, the high computational cost is a large impediment to fully assessing the potential of feasible regions in a limited time. Many efforts have been made to improve the efficiency of simulations, including building simplified models or obtaining approximate results from substitute physical concepts that are easier to calculate, and these are known as low-fidelity approaches. Several methods have been proposed to simplify the flow partial-differential equations of large-scale grids for numerical simulation. At present, commercial numerical software generally integrates the grid-roughing module to simplify the scale of solving partial-differential equations by generating a rough reservoir model. In addition, the streamline numerical-simulation method based on implicit pressure/explicit saturation has become an efficient and reliable choice for waterflood optimization (Bratvedt et al. 1992;

*Corresponding author; email: zhangkai@upc.edu.cn

Original SPE manuscript received for review 29 September 2020. Revised manuscript received for review 8 November 2020. Paper (SPE 205013) peer approved 1 December 2020.

Crane et al. 2000). This method converts the fluid migration between grids in the standard finite-difference simulator into a 1D flow space solution, thus saving several times of calculation time. This advantage is particularly significant in large-scale reservoir cases.

Another powerful low-fidelity approach is to build a data-driven proxy model instead of the numerical simulator. Using a trained data-driven model for computation can save much more time. At present, the proxies, including engineering models and regression models, have achieved remarkable results in the field of production optimization. On the one hand, the engineering model introduces the physical theory or engineering information as the basis of the model and then adjusts the parameters through the training data to improve the approximation performance of the model. The interwell numerical-simulation model (Zhao et al. 2016; Guo and Reynolds 2018, 2019) approximates the waterflood reservoir as a number of interwell units with transmissibility and control pore volume. It predicts production performance by solving the mass-material-balance and front-tracking equation. The capacitance/resistance model (Yousef et al. 2006; Weber 2009) takes into account the effects of compressibility (capacitance) and transmissibility (resistance) to measure the fluctuations between the indicators of injectors and the producers. The artificial neural network-discrete empirical interpolation method (Foroud and Seifi 2016; Foroud et al. 2018) learns the nonlinear mapping of control parameters with pressure and saturation at points of interest using the artificial neural network, and then constructs a flow dynamic system to replace the simulator for production optimization through the discrete empirical interpolation method. On the other hand, the regression model is also a popular alternative, and uses machine learning or statistical methods to directly establish the relationship between well-control parameters and NPV. This proxy model generally works with evolutionary algorithms and learns online with iterative updates of the population (known as infilling) (Yondo et al. 2018). Kriging, also known as the Gaussian process (Sacks et al. 1989), has been extensively used for decades because of its ability to predict mean-squared error, but often requires dimension reduction when dealing with high-dimensional problems such as production optimization (Chen et al. 2020a). Radial basis function (Broomhead and Lowe 1988; Chen et al. 2020b, 2021) and support-vector regression (Drucker et al. 1997; Guo and Reynolds 2018) have more advantages in dealing with high-dimensional problems. Previous studies show that such methods have achieved remarkable results in production optimization. In addition, artificial neural networks have the potential to replace numerical simulators because of their universal approximation ability (Foroud et al. 2014; Golzari et al. 2015; Teixeira and Secchi 2019). However, the applicability of different kinds of neural networks is still an open question.

Although the previous low-fidelity methods greatly improve the calculation efficiency of the objective function in the optimization search, the quality of the final result cannot be guaranteed through the simplified system. Therefore, a tradeoff between speed and accuracy is required, which leads us to focus on multifidelity methods. For an engineering-optimization problem, when additional information about the problem is available, the multifidelity approach is used to compensate for the shortcomings of the single-fidelity approach (Forrester and Keane 2009). The main principle is to speed up the search by combining a large amount of cheap but inaccurate evaluation information with a small amount of accurate but expensive evaluation information (de Baar and Roberts 2017). How to balance the call frequency of tasks with different fidelities and make efficient use of high-fidelity information is the key to improving the computational efficiency of the multifidelity method. In current research, the multifidelity technique of optimization has multiple meanings.

For a given observed object/solution, precise measurement/evaluation (high fidelity) often requires high experimental/computational costs, and if there is a relatively coarse approximation (low fidelity), it can be used as an additional information aid for high-fidelity optimization. In this respect, cokriging (Kennedy and O'Hagan 2000), a geostatistical interpolation method with multisource inputs, is most widely used at present (Forrester et al. 2007; Le Ravalec-Dupin 2012; Thenon et al. 2016; Zaytsev and Burnaev 2017). However, as a kind of surrogate model with complex structure, the parameter training and correlation-matrix calculation in the construction process of cokriging need to consume a considerable computational cost. In addition, the setting of hyperparameters affects the approximation performance of different problems, which implies an additional budget for sensitivity analysis.

For a definite evaluation criterion (objective function), the optimization algorithm adaptively adjusts the precision level of the search pattern or approximation model (such as the sizes of the search grid or model training set) (Wang et al. 2016). Efficient allocation of computing resources at different stages of the search is achieved by fine modeling and searching only in potential regions. This type of multifidelity technique is generally attached to surrogate-assisted optimization. However, the performance of such methods depends on the accuracy of the fitness-landscape prediction of the model and is vulnerable to the "curse of uncertainty," especially in high-dimensional problems.

Without building additional models, this work tries to balance the advantages of the preceding two ideas. Therefore, we focus on transfer learning to achieve this goal. Transfer learning mimics the human-learning behavior of using previous knowledge to solve new problems more quickly. Differing from traditional machine learning from scratch, it can learn by way of some knowledge from source tasks to solve target tasks that are related but not the same (Gupta et al. 2018). In this case, if effective information can be extracted from the source task, the cost of solving the target task can be greatly reduced. For a certain problem, the inherent similarity between the high- and low-fidelity domains provides the feasibility of performing the information transfer. At present, the multifidelity approach has been applied in a multiform way in the field of transfer learning (Lim et al. 2008; Yuan et al. 2017). This work considers a shortcut, transferring by means of the population of the optimization algorithm. Population-based algorithms generally serve as the optimizer for production-optimization problems, which can generate a large number of available points of input/output data during the search process. In recent years, genetic transfer designed for evolutionary algorithms has attracted extensive attention as a new direction of transfer learning (Koçer and Arslan 2010; Neill et al. 2017; Iqbal et al. 2019). It takes individuals in population as the medium of transfer to construct the relationships among populations of similar evolutionary tasks. Previous studies under genetic programming have shown that the application of genetic transfer can improve the search performance significantly. However, these efforts focus on a "one-way algorithm" (i.e., the sequential transfer of populations between similar tasks). The downside of this approach is that because this is a one-time migration, only the final result of the source task is considered.

Therefore, this work attempts to establish a new multifidelity genetic transfer framework combining multitasking and sequential transfer to adaptively select the transfer strategy at different search stages. In the context of multifidelity optimization, especially during the exploration phase, we proposed an information-exchange mechanism with the two fidelity tasks in parallel, which is called the multitasking mode. Then, aiming at the misdirection of the low-fidelity task in the local area, the concept of relative fidelity is introduced to quantify the credibility of a low-fidelity result. The subsequent sequential transfer mode will choose an appropriate time according to the relative fidelity to send the result of the multitasking mode to the high-fidelity task without any loss. The framework is implemented by adjusting the parameters and search strategy of the optimization algorithm without the need to build an additional proxy model. The multifidelity genetic transfer framework proposed in this paper applies to any population-based algorithm, such as evolutionary algorithms or swarm-intelligence algorithms. It can also be extended to other expensive engineering problems, such as history matching and uncertainty quantification when additional information can be obtained.

The core work of this paper is to practice the proposed framework through DE to obtain an efficient and robust multifidelity optimization algorithm, which is called MTDE. Corresponding to the multitasking and sequential transfer in the framework, MTDE has two modes, b-transfer and p-transfer, which perform different fidelity tasks with the media of basis vector and population, respectively. In addition, four centralized mutation strategies under the multifidelity context are proposed to improve the local search capability of the algorithm. For the b-transfer mode, a low-cost population-establishment method is presented for population transfer to the high-fidelity domain.

The rest of this article is organized as follows. The Multifidelity Optimization Model section discusses the mathematical modeling of production optimization under the background of multifidelity. Then we quantify the fidelity and present the multifidelity genetic transfer framework. Third, the algorithm structure of MTDE is introduced. Furthermore, we test MTDE in three cases and analyze the results. Finally, we discuss and summarize this work.

## Multifidelity Optimization Model

The aim of production optimization is to obtain the most economical production scenario in the life cycle. NPV is generally used to evaluate the benefit of a strategy, taking into account the cost of water injection and sewage treatment in the development process. The design variable $x$ is expressed as a $d$-dimensional vector, where $d$ is the number of wells multiplied by the number of control timesteps. The objective function $f(x)$ can be normally formulated as

$$f(x) = NPV(x) = \sum_{k=1}^{K} \Delta t^k \frac{r_o Q_{o,k} - r_{wp} Q_{wp,k} - r_{wi} Q_{wi,k}}{(1+b)^{t_k}}, \quad (1)$$

where $x$ is an $n$-dimensional vector of design parameter in this optimization problem that represents well controls; $K$ is the total number of timesteps, and $k$ is the current timestep index; $\Delta t^k$ is the length of the $k$th timestep (in days); and $Q_{o,k}$, $Q_{wp,k}$, and $Q_{wi,k}$ denote average oil-production rate, water-production rate, and water-injection rate at the $k$th timestep, respectively; $r_o$ is the crude-oil revenue; $r_{wp}$ and $r_{wi}$ are the cost of sewage treatment and water injection, respectively; $b$ is the annual discount rate; and $t_k$ is the total development time at the $k$th timestep (in years).

The mathematical assessment paradigm in Eq. 1 is commonly used in the field of production optimization, and the numerical simulator usually serves for the calculation of the NPV function. We expect that the evaluation for the candidate solution $x$ is as consistent with the fact as possible. However, because the numerical simulation relies on assumptions and a theoretical model, the result inevitably displays errors. To describe this phenomenon, the concept of fidelity is introduced. Fidelity mentioned in this paper is defined as the accuracy of assessment for the quality of the candidate solution.

From the perspective of cost effectiveness, the idea of multifidelity constructs a model by integrating information from multiple fidelity levels to improve the efficiency of optimization. In the context of multifidelity, optimization modeling involves search tasks in a distinct fidelity domain. For simplicity, two fidelity levels are considered in this work. Herein, the optimization task based on the high-fidelity method and that based on the low-fidelity method are labeled by $T_h$ and $T_l$, respectively, and their corresponding domains are $D_h$ and $D_l$. It needs to be noted that $T_h$ is the ultimate goal of multifidelity optimization, while $T_l$ serves as a subsidiary of $T_h$ to obtain candidate solutions. We assume $f_h$ is the objective function of the candidate solution evaluated by the high-fidelity method, whereas $f_l$ represents the low-fidelity objective function. Hence the multifidelity optimization model in this structure can be expressed as

$$\max_{x \in \Omega \subset R^d} f(x), \quad (2)$$

$$f(x) = f_h\{x, \underset{x}{\operatorname{argmax}}[f_l(x)]\}, \quad (3)$$

$$\text{subject to } \mathrm{C}(x) \leq 0, \quad (4)$$

$$x^{\mathrm{lb}} \leq x \leq x^{\mathrm{ub}}, \quad (5)$$

where $f$ is the multifidelity objective function organized by $f_h$, which includes two input modes: searching directly for feasible domain of $x$ and searching with the optimized results obtained by $f_l$. Its realization will be discussed in the next section. Eqs. 4 and 5 represent the nonlinear constraint $\mathrm{C}(x)$ and boundary of decision variables, respectively, which limits the range of feasible region $\Omega$. In this work, the nonlinear constraint is set in the numerical simulator.

## Multifidelity Genetic Transfer

To construct a multifidelity framework using an evolutionary algorithm, a direct approach is considered. For a candidate solution $x$ produced by evolutionary operators (e.g., crossover, mutation), it will be first evaluated by the low-fidelity objective function $f_l$. If this candidate solution is fitter than the currently known low-fidelity optimum, it will be recorded as the low-fidelity optimum $x_{l\text{best}}$. However, the low-fidelity results are not entirely reliable. It is necessary to call the high-fidelity objective function $f_h$ to guarantee the optimality in the high-fidelity domain. Only in this case will this solution be saved as the new high-fidelity optimum $x_{h\text{best}}$. This strategy is called greedy multifidelity, which has been tested to be an effective method to filter out low-fidelity error interference, as shown later in the paper.

However, calling a high-fidelity function is a time-consuming process, and in particular, assigning computational resources to a low-fidelity task is a more efficient option for those phases where the low-fidelity function is sufficiently credible. Therefore, the purpose of this work is to reduce expensive high-fidelity function calls and to make full use of accurate high-fidelity information feedback to guide searches in the low-fidelity domain.

Using the multifidelity optimization model introduced previously, this section introduces a multifidelity genetic transfer learning framework for global optimization algorithms. First, a fidelity evaluation index, which changes dynamically with the optimization search process, is introduced, and two genetic transfer modes are proposed according to the index. Also note that this section only introduces the general transfer framework, and the implementation of the particular optimization algorithm will be presented later in this paper.

**Fidelity Evaluation Indices.** The effect of fidelity on optimization is our primary concern. The high-fidelity objective function $f_h$ and the low-fidelity objective function $f_l$ are descriptions of the same problem at different levels of fidelity. The high-fidelity objective function with fine model has accurate approximation performance, while the low-fidelity objective function with rough model has the advantage of high-speed calculation. However, this does not mean that the low-fidelity objective function has a smoother fitness landscape. Both should have a consistent trend from the global perspective, or else the low-fidelity objective function provides little value for optimization. The error of the low-fidelity objective function is usually expressed as the false optimum and the deflected optimum. To illustrate this error, **Fig. 1** shows a simple example. $x_{1h}$ and $x_{1l}$ represent the same local optimum. This kind of local optimum deviating from the true position is defined as deflected optimum. The optimum $x_2$ of the low-fidelity function does not correspond to any high-fidelity optimum, in which case it is defined as the false optimum. Therefore, we make a reasonable assumption that the low-fidelity method has good effectiveness in the global search process in the early stage, but when the latter converges to the local area, the accuracy of the low-fidelity method is greatly reduced.

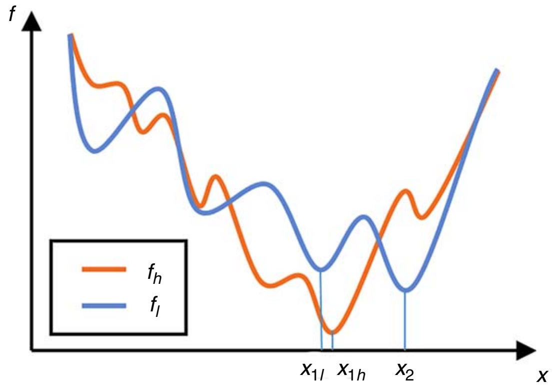


**Fig. 1—Simple case of the fitness landscape of low- and high-fidelity functions.**

Considering this characteristic, the principle of multifidelity is that, on the premise of ensuring sufficient accuracy of the evaluation, we hope to propose a balanced strategy that mainly calls the low-fidelity objective function with low calculation cost but only calls the high-fidelity objective function when necessary. This requires a decision using the fidelity requirements of the model and method in the optimization search process. For this principle, two evaluation indices of fidelity, absolute fidelity and relative fidelity, are proposed to assist decision making. The former serves for sorting fidelity levels, while the latter is used to evaluate the dynamic reliability of low-fidelity methods in the search process.

Absolute fidelity is inversely related to the difference/distance between the simulation results and the reality. It is usually a priori knowledge, depending on the quality of the model or the principle of the simulation method (e.g., the fidelity of the mesh-coarsening model is lower than that of the initial case, and the fidelity of the capacitance/resistance method is lower than that of the finite-difference method).

The difference between low- and high-fidelity models is usually reflected in the details of local scale. Nonetheless, the upward/downward trend in the global perspective is sufficiently reliable. To quantify the dynamic reliability of the low-fidelity method in the optimization process from global to local refinement, the concept of relative fidelity is introduced. Relative fidelity $R_f$ is defined as the difference/distance between the result from the used low-fidelity method and that from the assessment on fidelity at the reference level, which can be expressed by the person correlation coefficient of the $f_l$ and $f_h$ evaluation results according to sampling points over a period of time. For visual illustration, 10 generated sample points are selected from the early and late optimization stages of Case 1, which is tested later. As shown in **Figs. 2a and 2b,** the correlation coefficient showed significant differences in distinct optimization stages. Therefore, the relative fidelity is expressed as

$$R_f = \frac{\mathrm{cov}(F_h^n, F_l^n)}{\sigma_h^n \cdot \sigma_l^n}, \quad \ldots \quad (6)$$

where $F_h^n = [F_{h1}, F_{h2}, \ldots, F_{hn}]$ is the fitness label vector of latest $n$ sampling points evaluated by $f_h$; likewise, $F_l^n = [F_{l1}, F_{l2}, \ldots, F_{ln}]$ is evaluated by $f_l$, and $\sigma_h^n, \sigma_l^n$ are their standard deviations. $R_f$ will act as a key signal of reliability for the next mode switch. According to the property of the correlation coefficients, $|R_f| \in [0, 1]$ increases with the increase of the correlation between the results of $f_l$ and $f_h$. A positive value represents a positive correlation between $f_l$ and $f_h$, or vice versa. The low-fidelity method makes sense for optimization only when $R_f$ is greater than 0.

**Multifidelity Genetic Transfer Framework.** Aiming at the phenomenon of relative fidelity, the previously mentioned greedy multifidelity method is improved to the multifidelity method using genetic transfer learning. It mainly relies on low-fidelity tasks and can choose the right time to dock high-fidelity tasks. Moreover, it can also extract guidance information from the obtained results to promote convergence.

Transfer learning tries to reduce the cost of solving the target task by constituting a priori knowledge from the solving experience of repeated tasks or tasks with similar domains. The emphasis is how to identify and extract effective information from the source task. Transfer effectiveness is related to the proportion of the overlap between distinct domains, which clearly indicates the great potential of transfer learning for multifidelity optimization. If we consider a low-fidelity task $T_l$ as a source task and a high-fidelity task $T_h$ as a target task, the correlation between the corresponding domains $D_l$ and $D_h$ is significant for the same problem.

Population-based metaheuristic algorithms are widely favored in solving black-box problems such as production optimization because of their nonderivative character. This family of algorithms based on the theory of biomimetic swarm intelligence attach search paradigms to the individuals (or genetic materials) updated with iterations. The maturity of the individuals leads to convergence. The decisive role of genetic material in the search process and its independence from the algorithm provides feasibility and effectiveness for genetic transfer learning.

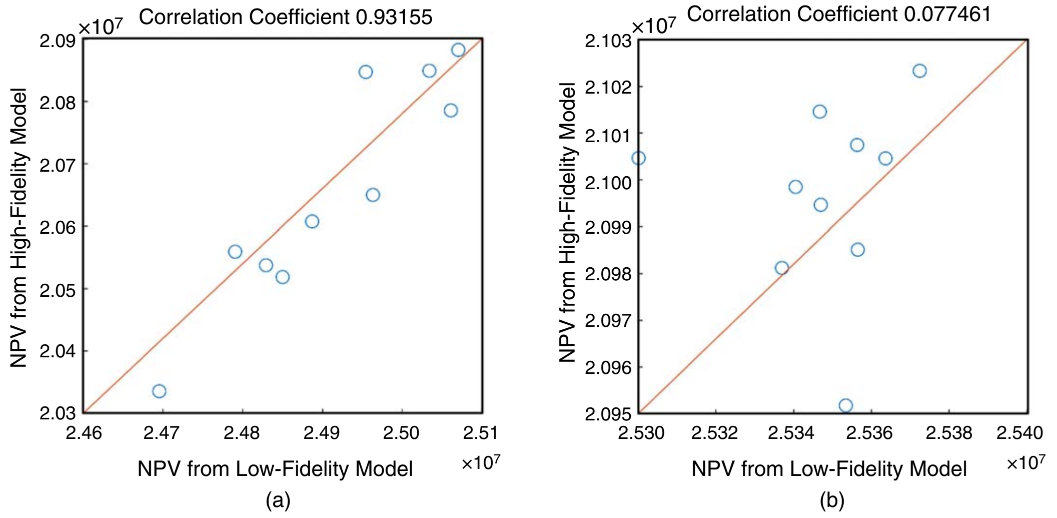
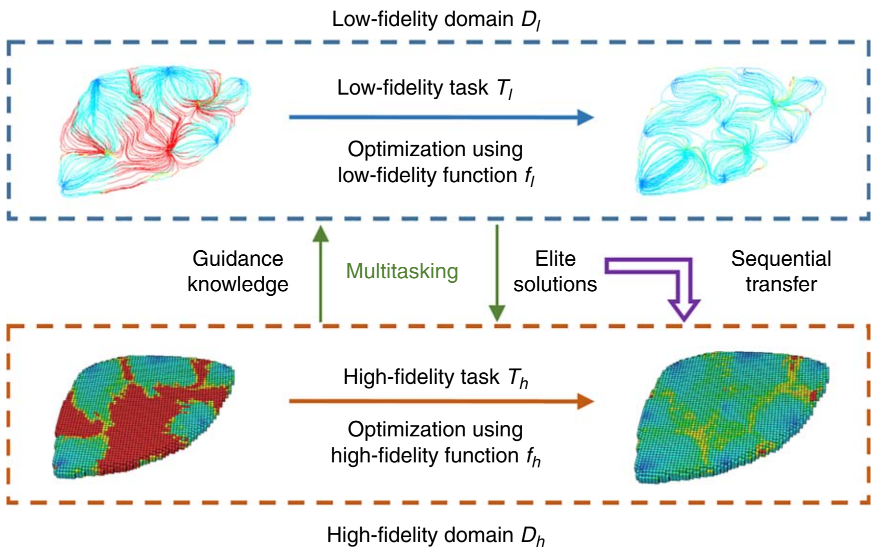


**Fig. 2—Correlation coefficients in the distinct stages of optimization: (a) early stage; (b) late stage.**

One of the most widely used ideas in genetic transfer is to locate the target task population to the dominant region by the transference of elite individuals according to the knowledge obtained from source task. This work further develops this approach in the context of multifidelity. According to the principles mentioned previously, the proposed multifidelity transfer-learning structure in this work includes two modes, including multitasking and sequential transfer (shown in **Fig. 3**), responding to the aforementioned relative fidelity $R_f$.



Low-fidelity domain $D_l$
Low-fidelity task $T_l$
Optimization using low-fidelity function $f_l$
Guidance knowledge
Multitasking
Elite solutions
Sequential transfer
High-fidelity task $T_h$
Optimization using high-fidelity function $f_h$
High-fidelity domain $D_h$

**Fig. 3—Multifidelity genetic transfer framework.**

Multitasking constructs a continuous information-exchange mechanism, including elite solution transmission from $T_l$ to $T_h$ and feedback guidance from $T_h$ to $T_l$. It is appropriate for the early stage of multifidelity optimization, when the $R_f$ of the low-fidelity method is still sufficient. In this mode, low- and high-fidelity tasks are parallel in space and serial in time. The memory spaces of tasks with different fidelity levels are independent of each other. $T_l$ with a lower computational cost is used to evaluate the solution in the current population when $T_h$ is dormant; $T_h$ is activated only when a fitter solution is found by $T_l$.

To be specific, we adopt the global search beginning with $T_l$; that is, $f_l$ is taken as the basis to evaluate the new solutions generated by the random operator rather than $f_h$. If and only if the $T_l$ finds a new minimum point, this elite solution will be transferred to the $T_h$ for verification. The result of $T_h$ will serve as a feedback signal to guide the following search direction of $T_l$ by adjusting the search step size or introducing the local optimization algorithm. In this way, the convergence of $T_l$ is accelerated to make up for the running time of $T_h$. According to our case study, the acceleration effect of transfer optimization is more prominent than the computational cost of $T_h$. Meanwhile, each elite solution with dual labels generated by $f_l$ and $f_h$ will be archived into the individual pool as the material for subsequent optimization.

Sequential transfer is the seamless continuation of multitasking. In the late convergence stage of the $T_l$, when the $R_f$ reduces to the threshold (0 in this work), we believe that the $f_l$ is no longer credible. Hence $T_l$ is terminated, and $T_h$ will start by organizing a new population, all or part of which are from the sorted individual pool. All the elite solutions in the pool with labels are from $f_h$; thus, there is no additional training cost for sequential transfer. It is worth noting that sequential transfer is started adaptively according to $R_f$ and is not a necessary step; it might not be triggered if the $f_l$ maintains a good performance of reliability.

## MTDE

**Differential Evolution (DE).** We first review its classic form. DE (Storn and Price 1997; Das and Suganthan 2011), known for its simplicity and robustness, is a population-based global optimization algorithm by means of parallel direct search. Similar to other evolutionary algorithms, DE employs mutation, crossover, and selection operators. It introduces randomness into the algorithm mainly through a mutation operator using weighted difference to search in the feasible domain. Using the unique mutation strategies of DE, we propose an efficient global optimization algorithm: MTDE. Applying the multitasking and sequential transfer described previously to DE, two transfer modes are set for MTDE, including b-transfer based on basis vector and p-transfer based on population.

The population $\{x\}_N = \{x_1^g, x_2^g, \ldots, x_N^g\}$ containing $N$ individuals (candidate solutions) is organized for MTDE, where $g$ is the current generation. For a $d$-dimensional problem, the current individual is $x_i^g = [x_{i1}^g, x_{i2}^g, \ldots, x_{id}^g], i \in [1, N]$ . The iterative optimization process of MTDE includes three classical operators, which we will discuss in the following paragraphs.

Mutation adds up the basis vector (also called the target vector) and the differential variation to obtain the mutant vector. Diversity of mutation strategies is one of the highlights of DE. According to the mode of the basis vector, the commonly used strategies include "rand" type and "best" type, and the proposed MTDE refers to both of them. For each individual $x_i^g$ in the population, create a corresponding mutant vector $v_i^g = [v_{i1}^g, v_{i2}^g, \ldots, v_{id}^g], i \in [1, N]$ according to DE/rand/1,

$$v_i^g = x_{r1}^g + Mu \cdot (x_{r2}^g - x_{r3}^g), \quad \ldots \quad (7)$$

and DE/best/1,

$$v_i^g = x_{\text{best}}^g + Mu \cdot (x_{r1}^g - x_{r2}^g), \quad \ldots \quad (8)$$

where $x_{\text{best}}^g$ is the current best individual, and $r1$, $r2$, $r3 \in [1,N]$ are random integer indices that are different from each other and also the current index $i$. At the $g$th generation, the population of mutant vectors is denoted as $\{v\}_{NP} = \{v_1^g, v_2^g, \ldots, v_N^g\}$. $Mu$ is the amplification factor controlling the proportion of differential variation, and its adaptive adjustment according to search stage will be discussed later. In this work, the adaptive mutation strategy and amplification factor constitute the core content of feedback from $T_h$ to $T_l$ in multitasking.

Crossover is another operator with randomness, and is used to generate the trial vector $u_i^g = [u_{i1}^g, u_{i2}^g, \ldots, u_{id}^g], i \in [1, N]$ by

$$u_{ij}^g = \begin{cases} v_{ij}^g & \text{if } \mathrm{rand}(0,1) \le r_c \text{ or } j = j_{\text{rand}} \\ x_{ij}^g & \text{otherwise}, \end{cases} \quad \ldots \quad (9)$$

where $r_c$ is the crossover rate and $j_{\text{rand}}$ is a random index $\in [1,d]$. The population of trial vectors at the $g$th generation is denoted as $\{u\}_N = \{u_1^g, u_2^g, \ldots, u_N^g\}$.

Selection is the last operator to update the population by deciding whether to keep $x_i^g$ or $u_i^g$ for the next generation $g$+1, which can be expressed as

$$x_i^{g+1} = \begin{cases} u_i^g & \text{if } f_l(u_i^g) < f_l(x_i^g) \\ x_i^g & \text{otherwise}. \end{cases} \quad \ldots \quad (10)$$

In the selection operator of standard DE, the fitness value is taken as the judgment condition of the update, and the objective function is called to calculate each individual in the population and the corresponding test vector. Because this process involves a large amount of computation, we prefer to use the $T_l$ to generate the fitness label of candidate solutions for updating the population, and only generate the high-fidelity fitness label for the key individuals determined by the principles discussed later.

MTDE adopts the preceding three traditional operators and adjusts the mutation strategy dynamically and parameter adaptively. According to the reliability of the used low-fidelity method, the algorithm is divided into two stages: b-transfer of multitasking and p-transfer of sequential transfer. The overall flow chart is shown in **Fig. 4** and the pseudocode of MTDE is shown in the appendix**.**

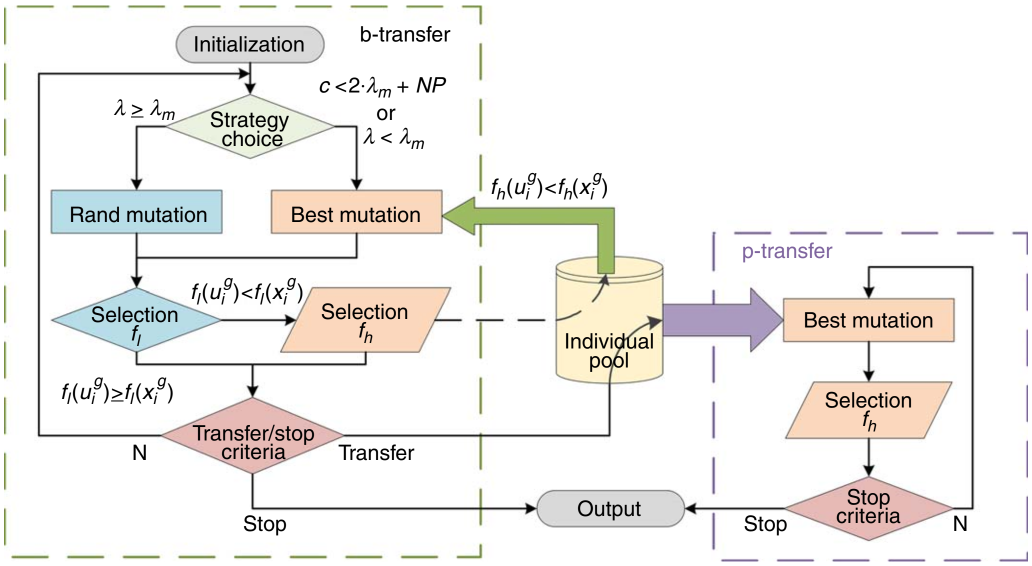


**Fig. 4—Flow chart of MTDE.**

**B-Transfer Mode.** The proposed b-transfer corresponds to the aforementioned multitasking mode, which is a transfer strategy carried by the basis vector of differential mutation. To minimize the frequency of high-fidelity function calls, MTDE applies $T_l$ to the selection operator and only uses $T_h$ to assist the construction of mutation operators. The mode of b-transfer refers to the idea of a memetic algorithm that combines the global search based on population with the heuristic local search based on individual. For the structural and functional characteristics of differential mutation operator, it can be realized through the dynamic adjustment of mutation strategy and amplification factor according to Eqs. 7 and 8. This mode uses the guideline with mutation-selection closed-loop feedback to alternate $T_l$ with $T_h$.

The switch of mutation strategies with different basis vectors is one of the guiding approaches of b-transfer. The basis vector occupies a larger proportion in the mutation vector than differential variation. Therefore, the choice of basis vector determines the search pattern of the algorithm. The basis vector of the "rand" strategy is randomly selected. Thus the generated mutation vector shows remarkable diversity, which means that the algorithm has better performance of exploration. It can avoid falling into a local optimum prematurely, but the convergence speed is slow. Consequently, it is suitable for global search in the early stage of optimization. On the contrary, the basis vector of the "best" strategy is the current best solution, so that all the mutation vectors of the population will be established with the best basis vector as the center. At this time, the algorithm has a strong ability of exploitation, and it is easier to search in the neighborhood for the current best solution. However, for multimodal problems such as production optimization, it is easy to fall into a local optimum, so it is more suitable for local search in the late stage of optimization.

Adjusting the amplification factor *Mu* is another guiding approach of b-transfer. *Mu* controls the search step size of the algorithm. A smaller *Mu* produces less disturbance, so it is easier to find a better solution near the basis vector. In this case, the algorithm has strong local search ability, but might fall into a local optimum. The increase of *Mu* means a larger step size, which makes the algorithm strong in global search but slow in convergence.

The preceding features enlighten us to adjust the mutation strategy and amplification factor adaptively for the different stages of the search process. Hence, to accelerate the convergence speed and ensure the quality of results, b-transfer follows three principles:

- A: In the exploration stage (the early stage), the strategy with strong global search capability is adopted to locate the current optimal solution to the potential region as far as possible.
- B: In the exploitation stage (the later stage of search), the current strategy should be closer to local search, with fine search near the current optimal solution.
- C: Considering the stall limit for the exploitation stage as well, the search strategy will be changed to try to escape from the local optimum.

Each of these principles reflects a search state, and correspondingly, there are three switching situations: location (Principles A/B), dispersion (Principles B/C), and relocation (Principles C/B). A feedback mechanism based on real-time monitoring of dynamic performance is presented for this. The "rand" strategy with increasing *Mu* and the "best" strategy with decreasing *Mu* are respectively regarded as global and local search strategies in the b-transfer mode.

In initialization, each individual in the initial population $\{x\}_N$ is evaluated by $f_l$ to obtain the corresponding low-fidelity label $F_l$. The $f_h$ is then called to evaluate the best individual in the initial population to obtain its high-fidelity label $F_h$. This individual with $F_h$ will be archived in an independent storage space, the individual pool $\{\text{pool}\}_t$, in which all the candidates labeled by the $f_h$ will be stored throughout b-transfer. Eq. 11 describes how the individual pool is initialized.

$$\text{pool}_1 \leftarrow [x_{m*}, f_h(x_{m*})];\, m^* = \underset{x_i \in \{x\}_N}{\text{argmax}}[f_l(x_i)]. \quad (11)$$

Exploration starts with initialization, which corresponds to Principle A and tries to locate the optimal solution of the current population to the region near the global optimum rather than a local one. This stage is implemented through iterations of the "rand" strategy with a deterministic constant, which can be assigned as a hyperparameter, or adaptively determined by the introduced guidance boundary parameter $\lambda_m$, denoted as

$$\lambda_m = \left\lceil \frac{T_{\max}}{T_{\text{ini}}} \right\rceil + d, \quad (12)$$

where $T_{\max}$ is the maximum time, $T_{\text{ini}}$ is the time taken by initialization, and $d$ is the dimension of design variables. As the threshold of iteration, $\lambda_m$ will be used multiple times in the proposed MTDE. In the adaptive strategy, the maximum iteration number of the exploration stage is set to double $\lambda_m$. The "rand" mutation strategy with a fixed amplification factor $Mu_0$ is executed throughout the exploration stage. Meanwhile, the individual pool is updated through the $T_h$. Assuming that the parameter $t^*$ is the update times of the optimal solution (including initialization), there are $t^*$ pairs of individuals with $F_h$ in the $\{\text{pool}\}_{t^*}$. Eq. 13 shows the updating of the individual pool in the b-transfer process.

$$\text{pool}_{t^*} \leftarrow [u_{m^*}, f_h(u_{m^*})];\, m^* = \underset{u_i \in \{u\}_N \cup \text{pool}_{t^*-1}}{\text{argmax}}[f_l(u_i)]. \quad (13)$$

At the end of the exploration stage, the individual in the pool with the best high-fidelity fitness value will serve as the basis vector for the "best" strategy in the exploitation stage, namely location (Principles A/B).

The exploitation stage, occupying most of the b-transfer running time, proceeds after the exploration, in which $T_l$ is guided by the result of $T_h$ to alternate between dispersion (Principles B/C) and relocation (Principles C/B). In this stage, $T_l$ is still used by the selection operator to update the population, whereas $T_h$ is used to update the basis vectors in the "best" strategy. First, according to Principle B, the exploitation stage begins with the local search according to the "best" strategy, with a basis vector of the best individual.

Considering the basis vectors assisted by multifidelity information, four optional "best" strategies are proposed in Eqs. 14 through 17. For *g*th generation,

MTDE/*h*best/1,

$$v_i^g = x_{h\text{best}}^g + Mu \cdot (x_{r1}^g - x_{r2}^g); \quad (14)$$

MTDE/*l*best/1,

$$v_i^g = x_{l\text{best}}^g + Mu \cdot (x_{r1}^g - x_{r2}^g); \quad (15)$$

MTDE/*p*best/1,

$$v_i^g = \begin{cases} x_{l\text{best}}^g + Mu \cdot (x_{r1}^g - x_{r2}^g) & r_{\text{rand}} < R \\ x_{h\text{best}}^g + Mu \cdot (x_{r1}^g - x_{r2}^g) & r_{\text{rand}} \geq R; \end{cases} \quad (16)$$

MTDE/*r*best/1,

$$v_i^g = [R_f \cdot x_{l\text{best}}^g + (R_f - 1) \cdot x_{h\text{best}}^g] + Mu \cdot (x_{r1}^g - x_{r2}^g), \quad (17)$$

where $x_{l\text{best}}$ and $x_{h\text{best}}$ are the optimal solutions obtained $T_l$ and $T_h$, respectively; $R_f$ is relative fidelity, and $r_{\text{rand}} \in [0,1]$ is a random number. **Fig. 5** visually shows the search characteristics of the four best strategies. The "*h*best" strategy shown in Fig. 5a considers only high-fidelity results when determining the basis vectors, which can filter the influence of false optimal points but will also miss deflected optimal points. Instead, the "*l*best" strategy searches near the best advantage found by all the low-fidelity tasks, as shown in Fig. 5b. Because the $T_l$ is updated more frequently, this strategy can result in a more active population. Considered together, the other two strategies make decisions according to relative fidelity. The "*p*best" strategy shown in Fig. 5c adopts a random pointer to keep the selection probability of the basis vector consistent with relative fidelity. In this case, if the low-fidelity method is more reliable, the basis vector will be more inclined to choose the low-fidelity best solution. The "*r*best" strategy shown in Fig. 5d is a similar approach, which selects a balance point in space.

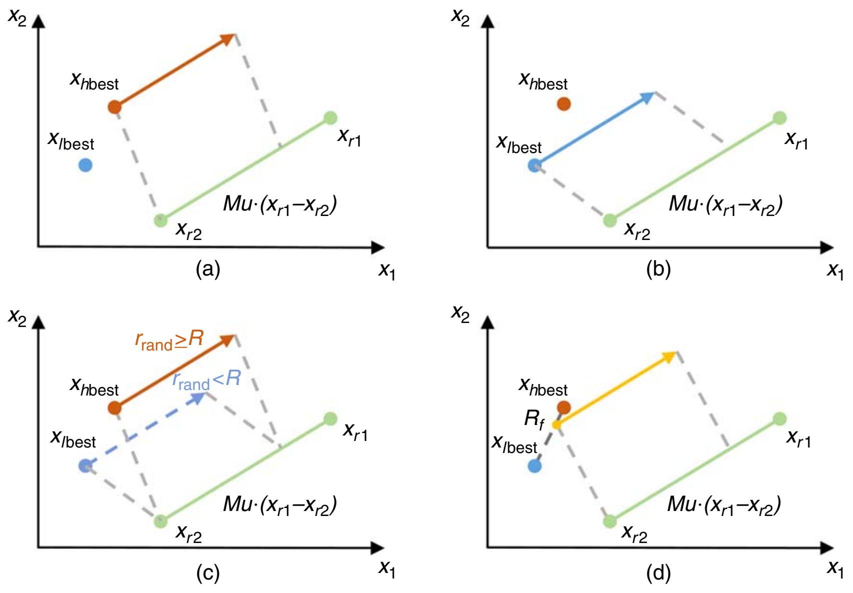


**Fig. 5—Characteristics of the four best strategies with multifidelity information. (a) MTDE/$_{h\text{best}}$/1, (b) MTDE/$_{l\text{best}}$/1, (c) MTDE/$_{p\text{best}}$/1, (d) MTDE/$_{r\text{best}}$/1.**

The local search launched around the *t*th optimal solution is denoted as best$_t$. The stall count $\lambda$, recording the stall generations of best$_t$, is introduced to monitor the healthiness of search, which is set to 0 at the beginning of best$_t$. If the new optimal solution is not obtained in the current iteration under best$_t$ (the selection operator with $T_l$ does not update the population, or the new optimal solution found by $T_l$ does not pass the verification by $T_h$), let $\lambda \leftarrow \lambda + 1$.

When $\lambda > \lambda_m$, we conclude that the algorithm falls into a local optimal. According to Principle C, b-transfer stops best$_t$ and changes the mutation strategy to "rand" for global search, until a new optimal point is found, namely dispersion (Principles B/C). In addition, $\lambda$ still counts in this case.

If and only if an optimal solution reducing the value of the high-fidelity function is found, set $\lambda \leftarrow 0$.

We assume that the current optimal solution lies in a potential region. According to Principle B, this individual becomes the basis vector to start the local search process of best$_{t+1}$, namely relocation (Principles C/B).

Besides the switch of variation strategy mentioned previously, feedback control of $T_h$ is also reflected in the dynamic regulation of the amplification factor. Here, we introduce the hyperbolic tangent function that is widely used in machine learning,

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \quad (18)$$

where tanh is an odd function with the range between –1 and 1, and the steepness of landscape decreases as the absolute value of input increases. Based on this property, it serves as the guidance of amplification factor regulation by $T_h$ task, which is dominated by the stall count $\lambda$. To combine the influence of the amplification factor on search with the characteristics of different mutation strategies, we assigned the part of the tanh on the left side of the *x*-axis to the "rand" strategy and the part on the right side of the *x*-axis to the "best" strategy.

$$\text{flag} = \begin{cases} 0 \text{ "best" strategy} \\ 1 \text{ "rand" strategy}. \end{cases} \quad (19)$$

The amplification factor is dynamically adjusted according to

$$Mu = Mu_0 + (-1)^{\text{flag}+1} \cdot \beta \cdot \tanh\left(3 \cdot \frac{\lambda - \lambda_m \cdot \text{flag}}{\lambda_m}\right), \quad (20)$$

where $Mu_0$ is the initial amplification factor and $\beta$ is the variation range of the amplification factor. The fraction in tanh is the distance between stall count and the guide boundary parameter $\lambda_m$, and the constant 3 is used to balance the effective interval of the function. For the "best" strategy, the fraction, which can be regarded as the normalized $\lambda$, increases gradually from 0, leading to a decrease in the amplification factor $Mu$ as the stall algebra increases. This will lead the algorithm to reduce the size of the search step if it fails to improve the current best solution. As for the "rand" strategy, because $\lambda$ begins with $\lambda_m$, this initial value needs to be subtracted first. On the contrary, $Mu$ will increase with stall generations, and therefore, the algorithm will increase its ability to escape from the local optimum through a larger disturbance. In this way, the amplification factor $Mu$ can be dynamically valued between ($Mu-\beta$, $Mu+\beta$) according to the current requirement observed from the feedback derived from $T_h$.

The b-transfer will alternate between dispersal (Principles B/C) and relocation (Principles C/B) in the exploitation stage according to this closed-loop feedback mechanism until the transfer criteria or stop criteria are met, and then the algorithm will start the p-transfer mode or deliver the final output result accordingly.

**P-Transfer Mode.** Normally, $f_l$ might not meet the accuracy requirement in the whole optimization process. Especially in the late stage of convergence, the errors of the low-fidelity method will interfere with the search direction of the algorithm. Sequential transfer will be triggered according to the dynamic variation of relative fidelity $R_f$. In this mode, we will transfer the multifidelity optimization process from the multitasking mode that uses both $T_l$ and $T_h$ to the search with $T_h$ only. Because the population is taken as the transfer medium, here we call this mode p-transfer. Unlike the alternative tasks mode of b-transfer, the task switch of p-transfer is a sequential and irreversible one-way algorithm.

In p-transfer, because the objective function used for the selection operator has been changed from $f_l$ to $f_h$, the results of multitasking need to be transferred to $T_h$ by establishing a new population. To reduce the cost of transfer, the individual pool $\{\text{pool}\}_t$ established in b-transfer is used as the resource of p-transfer, because the samples within are all labeled with $f_h$. Therefore, if all the individuals in the new population are from p-transfer, the cost of initialization with the high-fidelity function is eliminated, which is otherwise computationally expensive. If the current capacity of the individual pool is less than the population size of p-transfer, the population should be supplemented by random sampling.

After the previous search process, we assume that the population has converged to a worthy region, so best/1/bin strategy is used for further search with $T_h$. It should be noted that the p-transfer is triggered adaptively; as a result, this mode is not necessary for all cases. If the accuracy of low-fidelity method still meets the requirement, p-transfer will not start; instead, we will maintain the b-transfer mode with lower computational cost, seeking to increase the potential to find the global optimum through more search attempts.

Therefore, appropriate transfer criteria are required to accurately select the transfer time. At the end of each iteration of b-transfer, the relative fidelity $R_f$ will be checked according to Eq. 6 using the latest $n$ optimal solutions. We assume that if $R_f$ satisfies the requirements, the landscape of $f_l$ is still consistent with $f_h$. On the contrary, when the relative fidelity is less than the set threshold, p-transfer is considered to be triggered. It is worth noting that $T_h$ with few iterations have little value for optimization, so the remaining computational resource should also be considered before transfer. Criterion 1 is used to consider the preceding principles,

$$R_f \leq \varepsilon_0 \quad \&\& \quad T_{\text{res}} > T_f \times (N^* + N),$$

where the threshold $\varepsilon_0$ is the lowest allowable relative fidelity, which is set to 0 in this work. $T_{\text{res}}$ is the rest allowed time for optimization. $T_f$ is the cost of a single run of $f_h$. $N$ is the population size and $N^*$ is the number of new sampling points—not within $\{\text{pool}\}_t$—for the transfer population. The symbol && represents logical AND.

However, relying only on the relative fidelity as the p-transfer condition cannot handle the situation that the $T_l$ has already converged to a local optimum. Thus we set Criterion 2, taking into account the stall count $\lambda$,

$$\lambda \geq \lambda_M,$$

where the threshold $\lambda_M$ is the maximum allowable stall count. A small threshold can result in an inadequate search in b-transfer; conversely, the computational cost will be wasted in local optimal with a large threshold. We recommend setting $\lambda_M = 4\times\lambda_m$.

Considering relative fidelity $R_f$ and stall count $\lambda$, we set the transfer criteria for p-transfer as Criterion 1 or Criterion 2.

## Case Study

In this section, the proposed MTDE is applied to three production optimization cases of waterflood reservoirs. Meanwhile, using high- and low-fidelity numerical simulation, we test the cases with the same parameter setting to compare the advantages of the multifidelity method. To further demonstrate the improvement of the proposed multifidelity genetic transfer method, we also use an aforementioned greedy multifidelity control group, which does not involve transfer optimization, and the additional high-fidelity information only acts as a filter. All schemes are derived from DE. For distinguishing purposes, the test schemes with high- and low-fidelity functions are denoted as H-DE and L-DE, respectively. The DE algorithm under greedy multifidelity framework is called GMDE. The parameter settings of the four schemes of controlled trial are shown in **Table 1.** The initial population is generated by Latin hypercube sampling (LHS) (McKay et al. 2000), and each set of comparison tests controls the same initial population. Five independent runs are performed for each set of tests.

Without loss of generality, the evaluations of candidate solutions involved in optimization are performed using Eclipse$^{\text{SM}}$ Reservoir Simulation Software (Schlumberger Limited, Sugar Land, Texas, USA). The results of different fidelity are generated by means of two optional numerical-simulation models contained in the software: the black-oil and streamline models. The black-oil model can obtain the exact numerical solution by the fully implicit finite-difference method, which is considered to be totally stable and robust. Assuming the simulation results are close enough to the real production, they serve as the high-fidelity results for reference in the case study. The streamline model based on implicit pressure/explicit saturation also establishes pressure equations in the grid system, while the difference lies in the calculation of saturation. The numerical simulation based on the streamline model can establish the streamline field according to the equipotential surface of one-time pressure calculation. The fluid moves along streamlines in the direction of the pressure gradient rather than between the grids, which can reduce the simulation model to a series of 1D streamline simulations. Although the simplification results in a loss of accuracy, the simulation based on the streamline model can significantly reduce the computational cost and is therefore used to obtain low-fidelity results in the case study.

| Scheme Name | Optimization Algorithm | Fidelity Level | $Mu$ / $Mu_0$ | $r_c$ | $N$ |
|---|---|---|---|---|---|
| H-DE | DE/rand/1/bin | high- | 0.6 | 0.5 | 25 |
| L-DE | DE/rand/1/bin | low- | 0.6 | 0.5 | 25 |
| GMDE | GMDE/rand/1/bin | multi- | 0.6 | 0.5 | 25 |
| MTDE | MTDE/*p*best/1/bin | multi- | 0.6 | 0.5 | 25 |

Table 1—Settings of the four schemes for case study.

In addition, because the evaluation times of high- and low-fidelity objective functions differ greatly, the number of iterations cannot measure the cost of different schemes. Therefore, a fairer way is considered (i.e., setting the stop criteria as the maximum running time and comparing the quality of optimal solution obtained by each scheme in the limited computational budget).

**Case 1: Egg Model.** The egg model (Jansen et al. 2014) is a seven-layer channelized reservoir model with 25,200 grids (18,553 active grids). Typical properties of reservoir model are shown in **Table 2.** The permeability-field distribution is shown in **Fig. 6,** where the black spots represent the well locations. The reservoir consists of four production wells at constant pressure, and bottomhole-pressure target is set as 395 bar. Eight injection wells located around and in the center are controlled by surface flow targets, which will be design variables for production optimization. For each injection well, the allowable upper bound is 79.5 m$^3$/d and the lower bound is zero. The well-control strategy is from the start time of production. The 10-year production cycle is divided into five timesteps. Therefore, the dimension of the optimization problem is $8 \times 5 = 40$. In the NPV formula, we set the oil revenue as 50 USD/m$^3$, the water-injection cost as 3 USD/m$^3$, and the water-production cost as 4 USD/m$^3$. Discount rates are not taken into account.

| Properties | Value |
|---|---|
| Model size | $60 \times 6 \times 7$ |
| Depth | 4000 m |
| Porosity | 0.2 |
| Connate water saturation | 0.1 |
| Density of oil | 900 kg/m$^3$ |
| Initial pressure | 400 bar |
| Viscosity of oil | 2 cp |
| Oil formation volume factor | 1 m$^3$/std m$^3$ |
| Oil compressibility | $1.0\times10^{-5}$ bar$^{-1}$ |

Table 2—Properties of egg model.

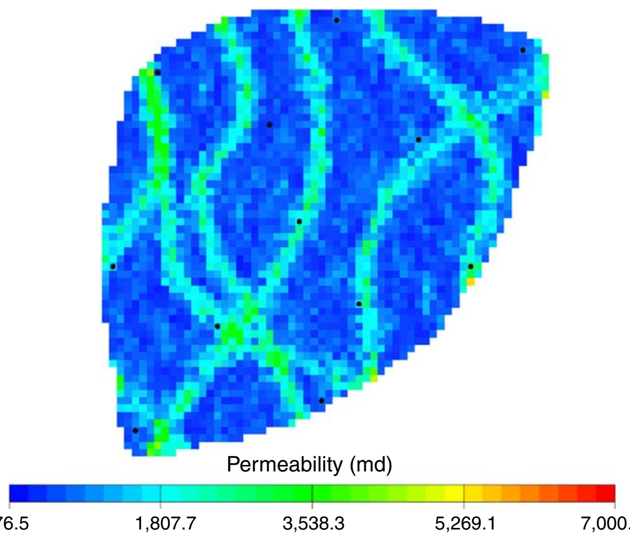


**Fig. 6—Permeability field of egg model.**

In this test case, the computational budget of optimization is 28,800 seconds. **Fig. 7** shows the convergence curve of each scheme in the egg model. It should be noted that the ordinate shows the NPV value obtained by the high-fidelity objective function. Therefore, for L-DE, which does not involve the high-fidelity objective function in optimization, its convergence curve is obtained by calculating the high-fidelity function value of all its historical optimum, and this process, as a subsequent data processing, is not included in the cost. From Fig. 7, we can see that H-DE has the worst result because it allows the least number of iterations within a certain time. L-DE and MTDE converge fast in the early stage. However, the curve of L-DE oscillates in the late stage and ends with a value worse than the historical optimum. This is because the relative fidelity of the low-fidelity method decreases rapidly in the late stage of convergence, so that the search is deceived by the false optimum. On the contrary, GMDE converges more slowly in the early stage because of the additional high-fidelity cost, but the improvement is achieved by filtering the false optimum with high-fidelity information. The proposed MTDE has the best convergence performance. We can observe the rapid rise of curve in the b-transfer stage guided by high fidelity. After a short stall in the middle stage (around 10,000 seconds), a great improvement was made again, which was attributed to the p-transfer mode. After the convergence in the low-fidelity domain, p-transfer will establish a new population based on the individual pool and continue to search in the high-fidelity domain, which is not available in other schemes. All schemes are repeated five times to ensure the results are reliable, and the distribution of these results can be seen in **Fig. 8.** All the results from MTDE are better than those from the others. The important intermediate parameters of MTDE are listed in **Table 3.**

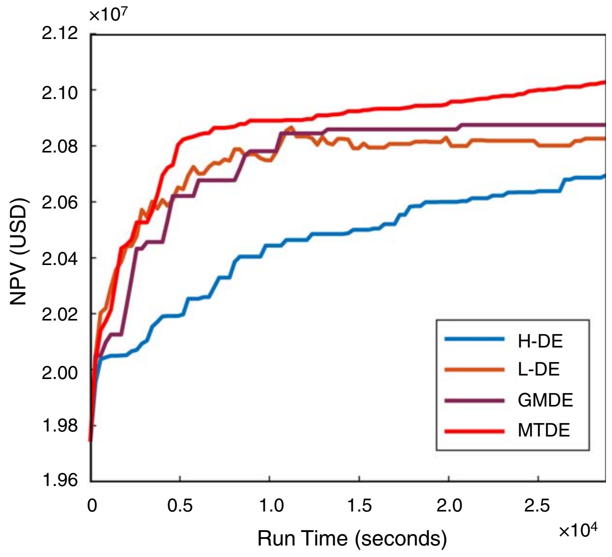


**Fig. 7—NPV vs. run time by H-DE, L-DE, GMDE, and MTDE for egg model.**

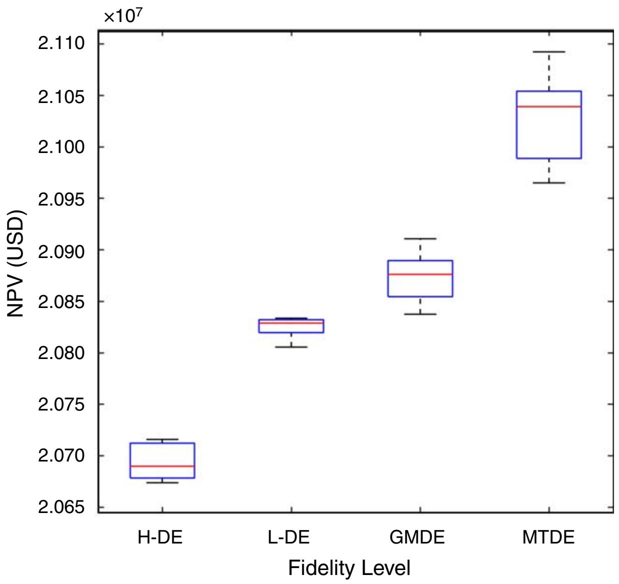


**Fig. 8—Distribution of NPV of four schemes with five independent runs.**

We also analyze the quality of the optimal solution from the perspective of oilfield production and development. The final optimal well-control strategy from each test scheme is shown in **Fig. 9,** where the initial value is the best individual of the initial population generated by LHS. **Fig. 10** shows the effect of each scheme in terms of water control and oil stabilization. It can be seen that the strategy obtained by MTDE achieves higher oil production through less injection volume. The remaining oil distribution of the first layer of the model is shown in **Fig. 11.**

| Intermediate Parameters | Mean Value |
|---|---|
| Cost of a single run of $F_l$ | 6.10 seconds |
| Cost of a single run of $F_h$ | 221.11 seconds |
| Frequency of $T_l$ | 2,992 |
| Frequency of $T_h$ in b-transfer | 33 |
| Frequency of $T_h$ in p-transfer | 149.2 |
| Fitness of initial population | $2.00\times10^7$ |
| Fitness of b-transfer result | $2.09\times10^7$ |
| Fitness of p-transfer result | $2.10\times10^7$ |

Table 3—Intermediate parameters of MTDE in egg model.

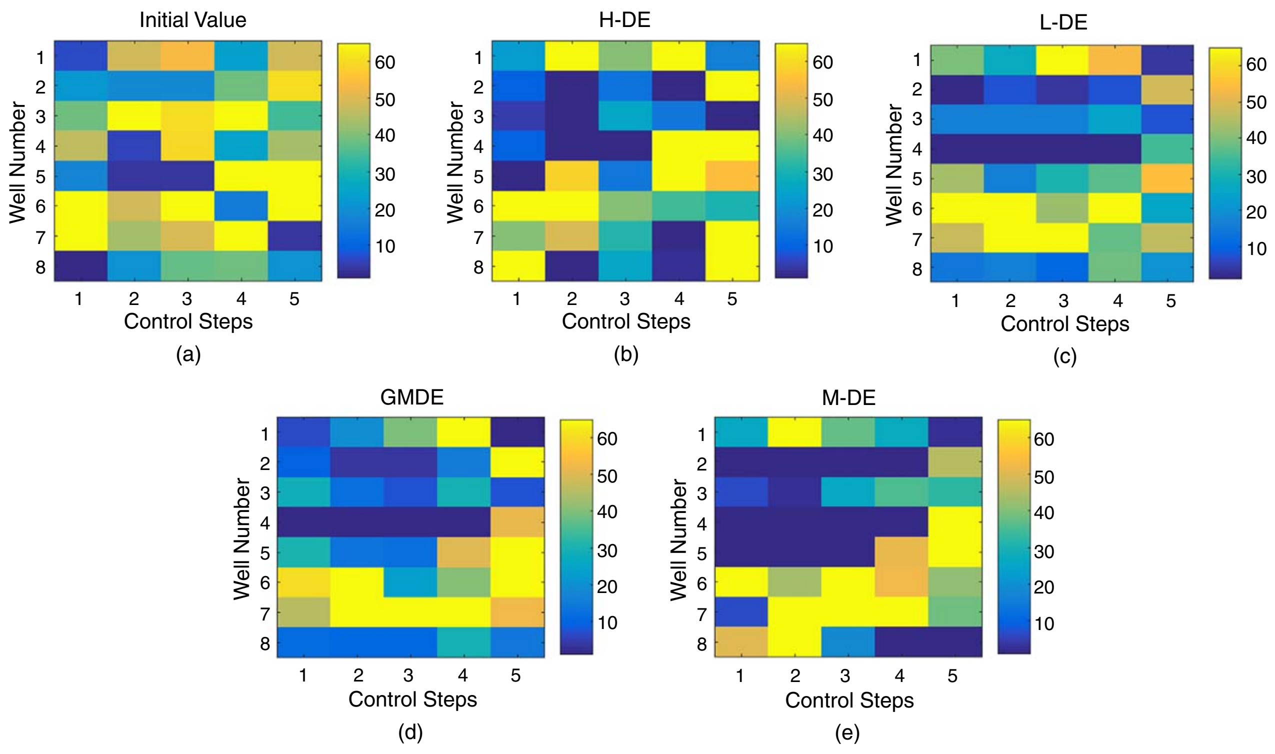


**Fig. 9—Optimal rate target well controls provided by LHS, H-DE, L-DE, GMDE, and MTDE for egg model.**

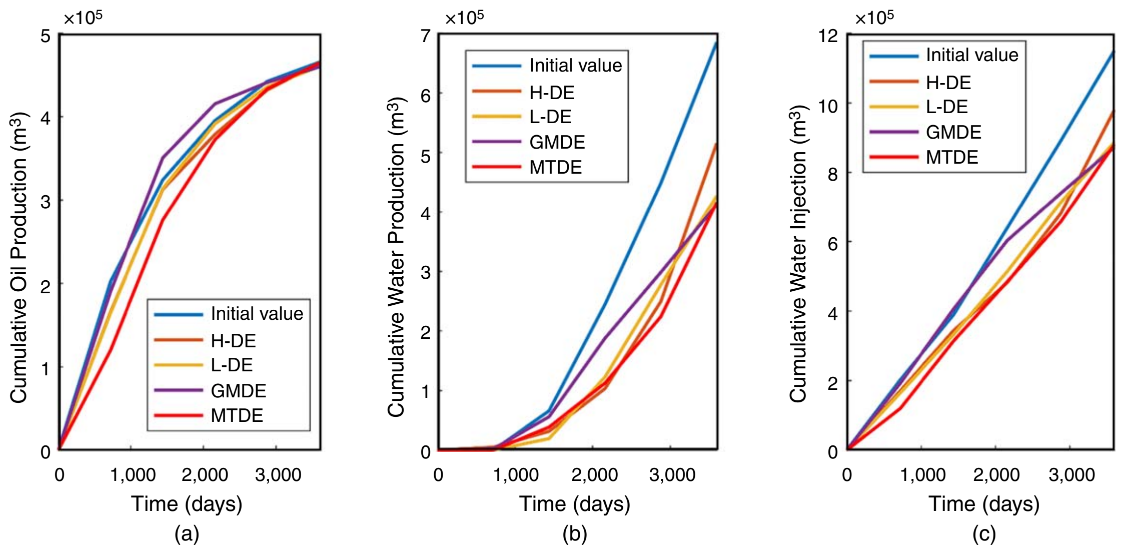


**Fig. 10—Results of optimum control provided by LHS, H-DE, L-DE, GMDE, and MTDE for egg model: (a) cumulative oil production vs. time, (b) cumulative water production vs. time, and (c) cumulative water injection vs. time.**

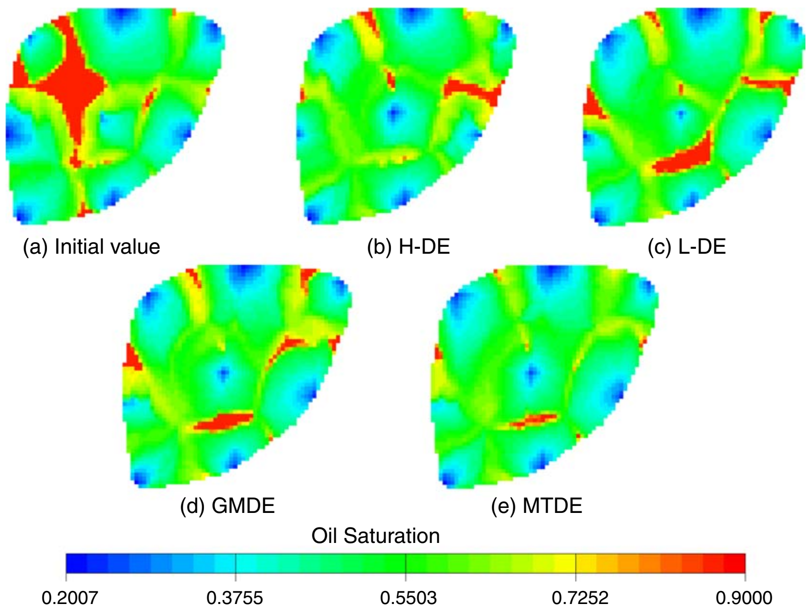


**Fig. 11—Remaining oil distribution of the first layer of the egg model by LHS, H-DE, L-DE, GMDE, and MTDE.**

**Case 2: Unconformity Reservoir Model.** The reservoir model is built under the condition of unconformable strata, with a total of 19 layers, and an inactive layer is set in every other layer. History matching has been performed using the observed data of oil-production rate, water-production rate, and water cut. The model contains 59,295 grids, of which 37,091 grids are active. The permeability distribution is shown in **Fig. 12.** The model includes eight injection wells and 15 production wells controlled by surface flow targets that are set as design variables. For the injectors, the upper bound of the water-injection rate is 150 $m^3$/d and the lower bound is zero. For the producers, the upper bound of the fluid-producing rate is 80 $m^3$/d and the lower bound is zero. Low formation pressure will lead to dissolved gas release, so we set an additional constraint, the bottomhole-pressure lower bound at 80 bar. The reservoir has been in production for 457 days and the injection/production strategy for the next 5 years is required, which is divided into five timesteps. Therefore, the dimension of the optimization problem is $(8 + 15) \times 5 = 115$. In the calculation of NPV, we set the oil revenue as 50 USD/$m^3$, the water-injection cost as 3 USD/$m^3$, and the water-production cost as 4 USD/$m^3$ according to the actual situation, and set the discount rate as 0%. Typical properties of reservoir model are shown in **Table 4.**

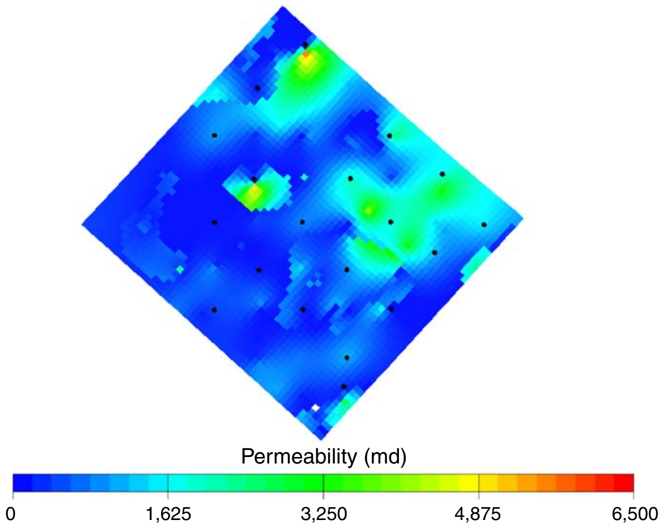


**Fig. 12—Permeability field of unconformity reservoir model.**

| Properties | Value |
|---|---|
| Model size | $55 \times 51 \times 19$ |
| Depth | 1278.5 m |
| Connate water saturation | 0.208 |
| Density of oil | 957 kg/$m^3$ |
| Initial pressure | 135 bar |
| Transmissibility | 20 cp-res $m^3$/d/bar |

Table 4—Properties of the unconformity reservoir model.

The maximum optimization time of this test case is 50,000 seconds. The convergence curves of the four schemes are shown in **Fig. 13.** H-DE performs the worst because it iterates slowly. For the schemes working with low fidelity, lower computational cost means that more attempts can be made within the feasible region. The performance of these three algorithms is similar in the early stage of search, but the L-DE method oscillates violently after convergence. Each update of L-DE comes from a new optimum found by the low-fidelity task, but in fact, as can be seen from the convergence curve, these are false optimums. GMDE is not affected by this error thanks to the auxiliary high-fidelity task. MTDE, by contrast, makes better use of additional high-fidelity information. With the help of the transfer-feedback mechanism, MTDE accurately locates the potential region in the high-fidelity domain and thus obtains the best results. The distribution of results of each scheme is shown in **Fig. 14.** The important intermediate parameters of MTDE are shown in **Table 5.**

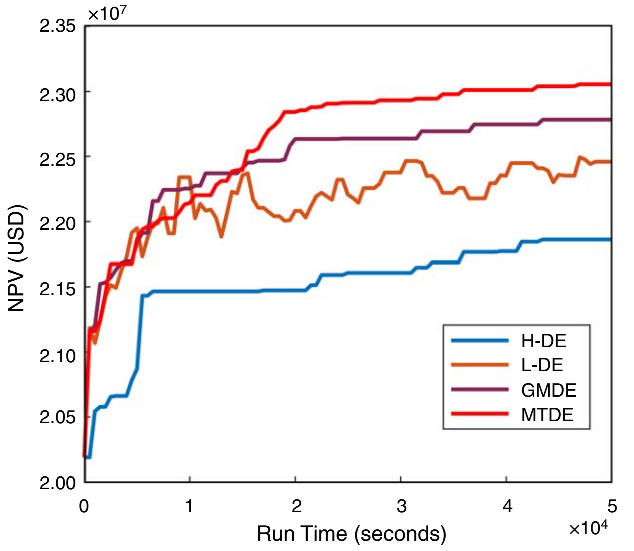


**Fig. 13—NPV vs. run time by H-DE, L-DE, GMDE, and MTDE for unconformity reservoir model.**

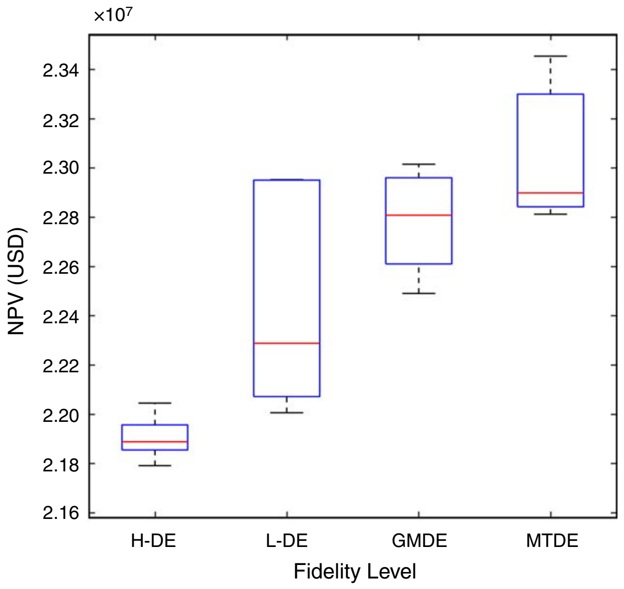


**Fig. 14—Distribution of NPV of four schemes with five independent runs.**

We also evaluate the attractiveness of each optimal solution obtained by the schemes. **Fig. 15** reveals these optimal well-control strategies. The effects of water control and oil stabilization of each scheme are shown in **Fig. 16.** The proposed MTDE significantly increased cumulative oil production, and both water injection and water production were lower than GMDE, which was second in oil production. A typical layer of the model is selected to show the oil-saturation distribution (**Fig. 17**) after the completion of each strategy. It can be observed that MTDE achieves a higher recovery percentage in many regions.

**Case 3: Edgewater-Reservoir Model.** The test case is an edgewater-reservoir model with 33 layers. History matching has been performed using the observed data of oil-production rate, water-production rate, and water cut. The model contains 374,616 grids, of which 199,822 grids are active. The permeability-field distribution is shown in **Fig. 18.** The reservoir involved seven injection wells and 14 production wells that are controlled by surface flow targets, all of which needed to be optimized. For the injectors, the upper bound of the water-injection rate is 300 $m^3$/d and the lower bound is zero. For the producers, the upper bound of the fluid producing rate is 150 $m^3$/d and the lower bound is zero. Meanwhile, the bottomhole-pressure lower bound for the production well is 190 bar. Production optimization starts from scratch to plan the injection-production strategy for the next 4 years with 1 timestep/yr. Therefore, the dimension of the optimization problem is $(7+14)\times 4 = 84$. In the calculation of NPV, we set the oil revenue as 80 USD/$m^3$, the water-injection cost as 5 USD/$m^3$, and the water-production cost as 5 USD/$m^3$, according to the actual situation, and assume that the discount rate is zero. Typical properties of reservoir model are shown in **Table 6.**

| Intermediate Parameters | Mean Value |
|---|---|
| Cost of a single run of $F_l$ | 20.33 seconds |
| Cost of a single run of $F_h$ | 593.48 seconds |
| Frequency of $T_l$ | 1,501.8 |
| Frequency of $T_h$ in b-transfer | 27.2 |
| Frequency of $T_h$ in p-transfer | 28.4 |
| Fitness of initial population | $2.12\times10^7$ |
| Fitness of b-transfer result | $2.29\times10^7$ |
| Fitness of p-transfer result | $2.31\times10^7$ |

Table 5—Intermediate parameters of MTDE in the unconformity reservoir model.

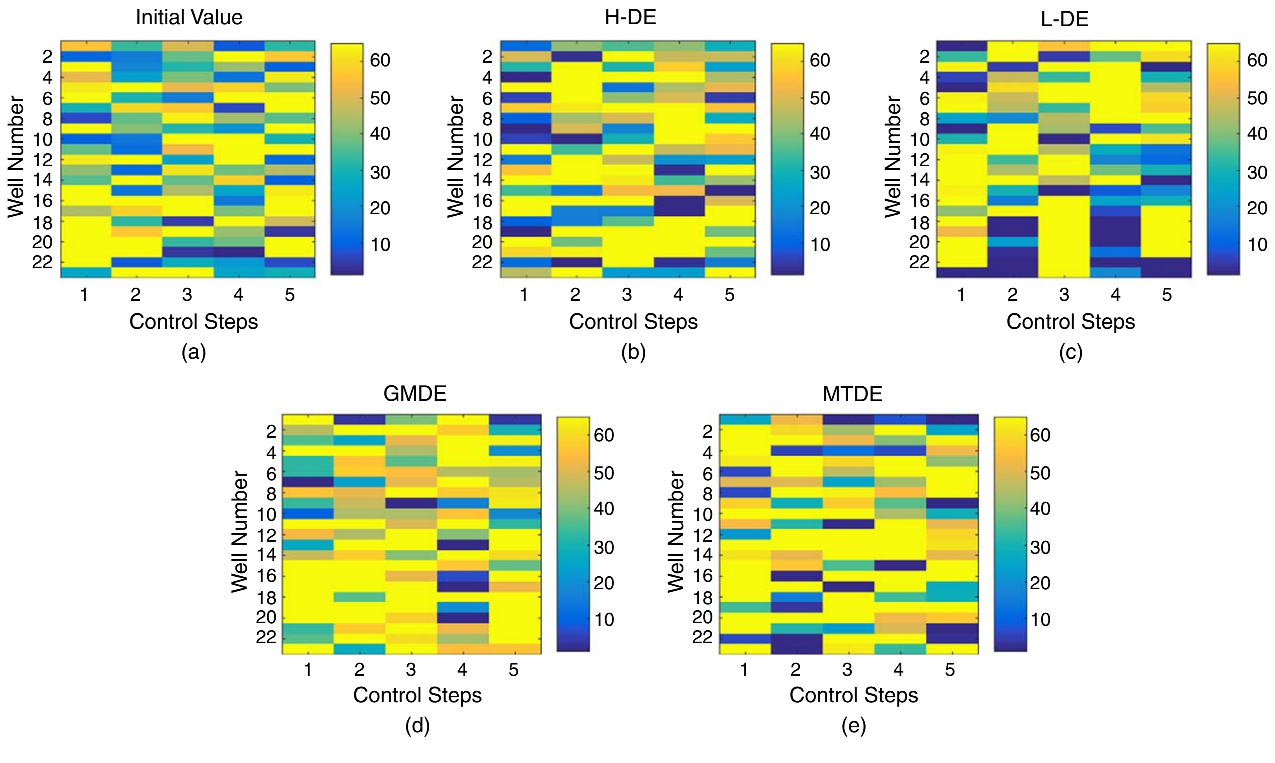


**Fig. 15—Optimal rate target well controls provided by LHS, H-DE, L-DE, GMDE, and MTDE for unconformity reservoir model.**

The time limit of this test case is 86,400 seconds. The convergence curves of the four schemes are shown in **Fig. 19.** Because the high-fidelity objective function in this case is extremely expensive, H-DE shows a distinct disadvantage. Compared with L-DE and GMDE, MTDE has the high-fidelity information feedback under the b-transfer mode, which provides faster convergence speed and stronger global search capability. It is worth mentioning that by observing the change of relative fidelity during search, the low-fidelity objective function in this case presents high credibility. According to the adaptive transfer criteria, MTDE does not perform p-transfer in this case, and thus avoids expensive high-fidelity numerical simulation. As for MGDE, there is no advantage over L-DE because of the additional computing overhead of the multifidelity task. The distributions of the results of five tests for each scheme are shown in **Fig. 20.** All the solutions generated by MTDE have significant advantages. The important intermediate parameters of MTDE are shown in **Table 7.**

Furthermore, we discuss the optimal well-control strategy generated by each scheme. **Fig. 21** shows the optimal well control obtained by each scheme. **Fig. 22** compares the water-control and oil-stabilization effects of each scheme. We can observe that the injection/production strategy from MTDE can produce the maximum amount of oil with relatively little water injection and water production. In the exploitation process, the strategy keeps a low water content. The pressure field and oil-saturation distribution of each strategy are shown in **Fig. 23** (edgewater area not shown). MTDE generates a uniform displacement, and this strategy is consistent with the production demand.

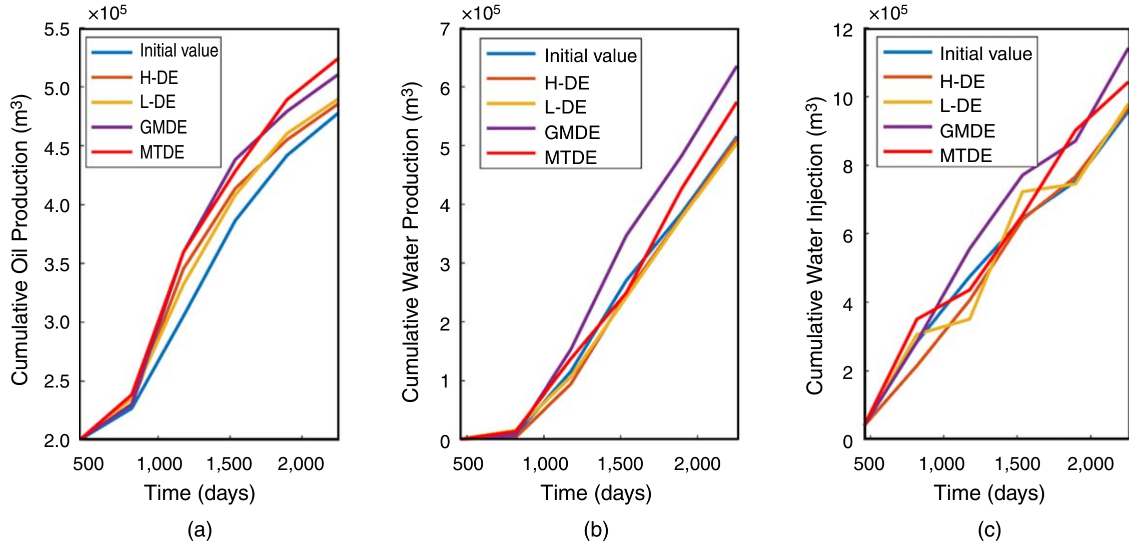


Fig. 16—Results of optimum control provided by LHS, H-DE, L-DE, GMDE, and MTDE for unconformity reservoir model: (a) cumulative oil production vs. time, (b) cumulative water production vs. time, and (c) cumulative water injection vs. time.

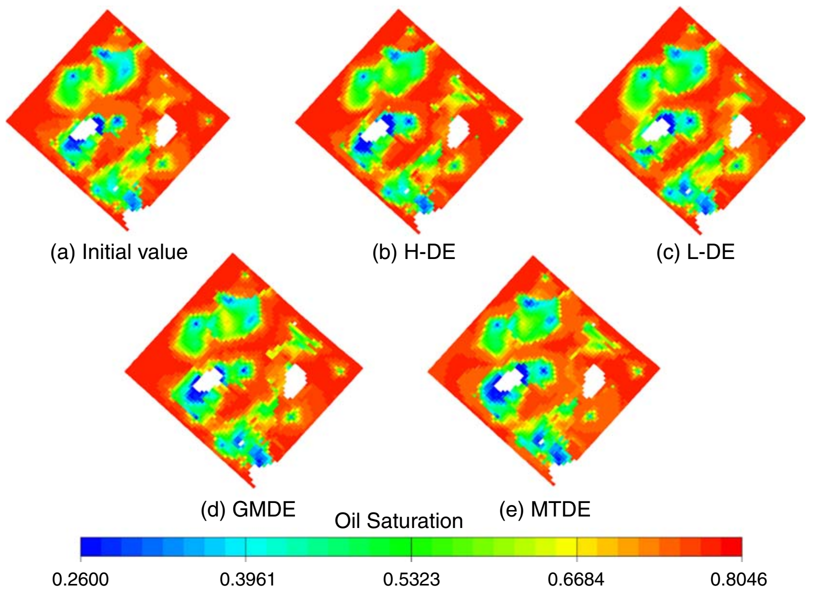


Fig. 17—Remaining oil distribution of the first layer of unconformity reservoir model by LHS, H-DE, L-DE, GMDE, and MTDE.

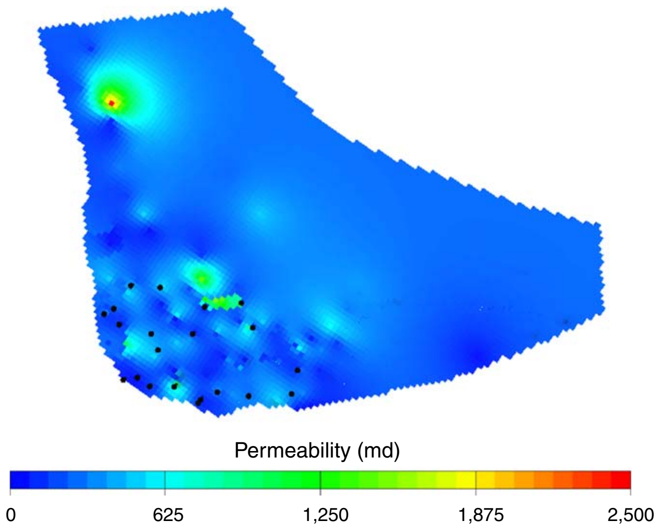


Fig. 18—Permeability field of edgewater-reservoir model.

| Properties | Value |
|---|---|
| Model size | 132 × 86×33 |
| Depth | 2165.8 m |
| Connate water saturation | 0.3 |
| Density of oil | 931 kg/m$^3$ |
| Initial pressure | 225 bar |

Table 6—Properties of edgewater-reservoir model.

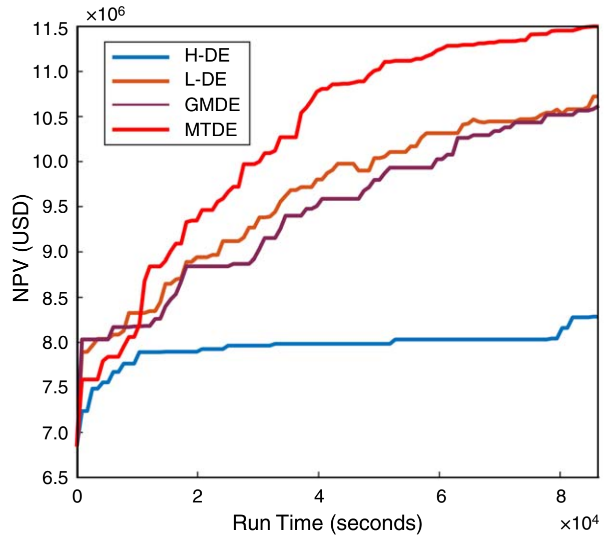


**Fig. 19—NPV vs. run time by H-DE, L-DE, GMDE, and MTDE for edgewater-reservoir model.**

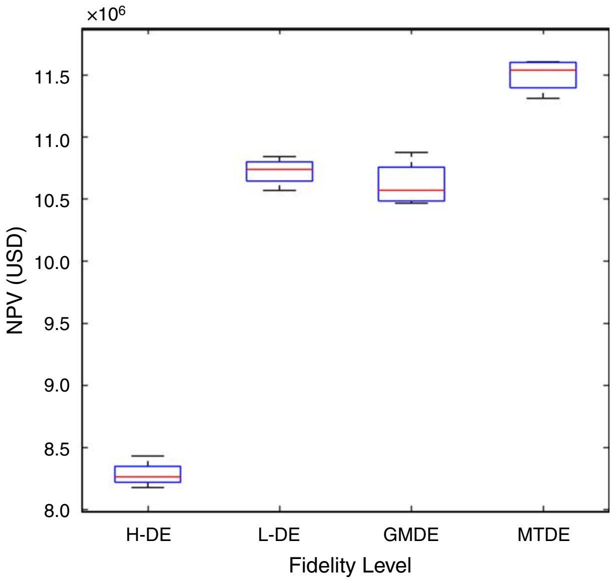
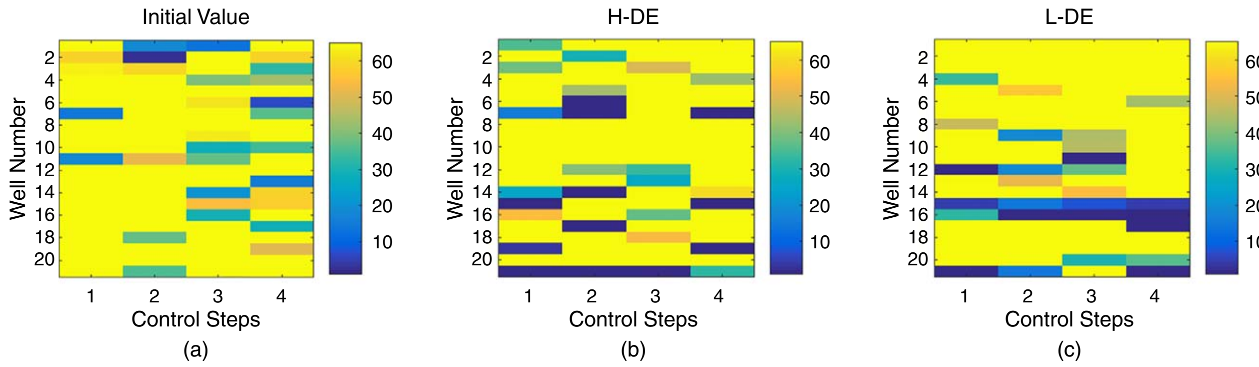


**Fig. 20—Distribution of NPV of four schemes with five independent runs.**

| Intermediate Parameters | Mean Value |
|---|---|
| Cost of a single run of $F_l$ | 84.64 seconds |
| Cost of a single run of $F_h$ | 1,234.29 seconds |
| Frequency of $T_l$ | 854.75 |
| Frequency of $T_h$ in b-transfer | 35 |
| Frequency of $T_h$ in p-transfer | 0 |
| Fitness of initial population | $7.59\times10^6$ |
| Fitness of b-transfer result | $1.15\times10^7$ |
| Fitness of p-transfer result | – |

Table 7—Intermediate parameters of MTDE in edgewater-reservoir model.



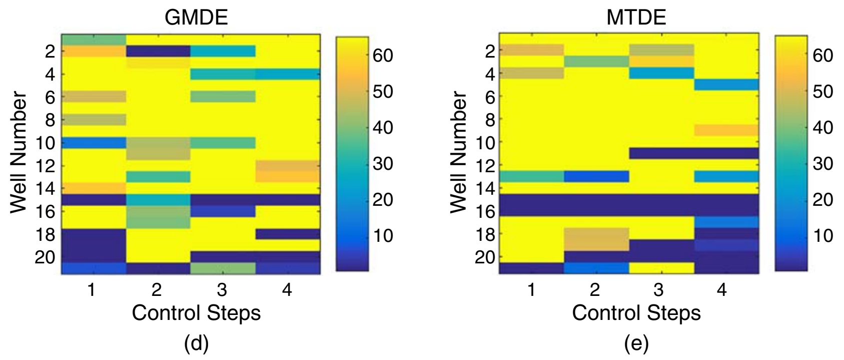


Fig. 21—Optimal rate target well controls provided by LHS, H-DE, L-DE, GMDE, and MTDE for edgewater-reservoir model.

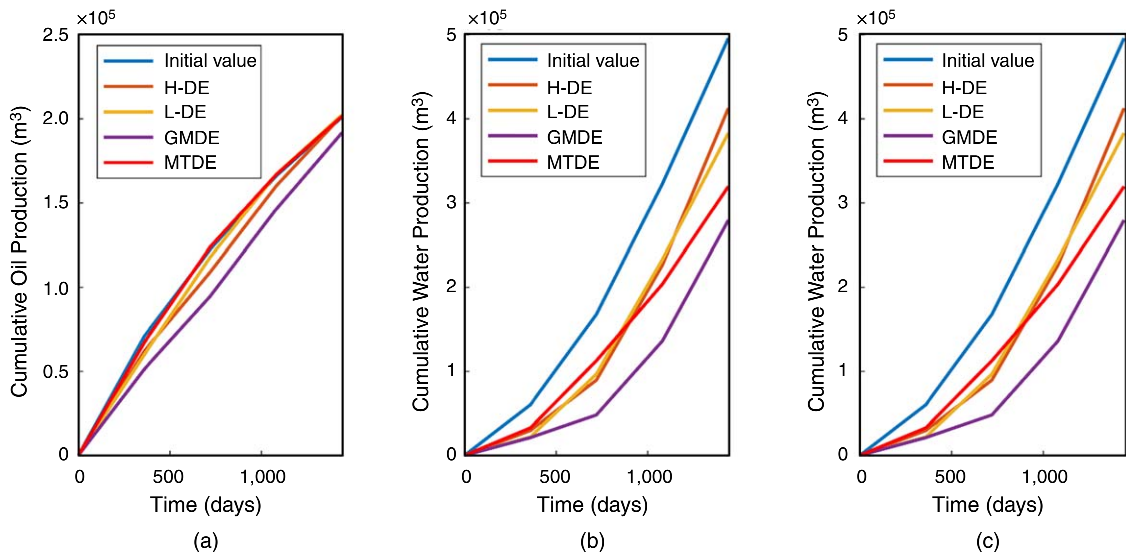


Fig. 22—Results of optimum control provided by LHS, H-DE, L-DE, GMDE, and MTDE for edgewater-reservoir model: (a) cumulative oil production vs. time, (b) cumulative water production vs. time, and (c) cumulative water injection vs. time.

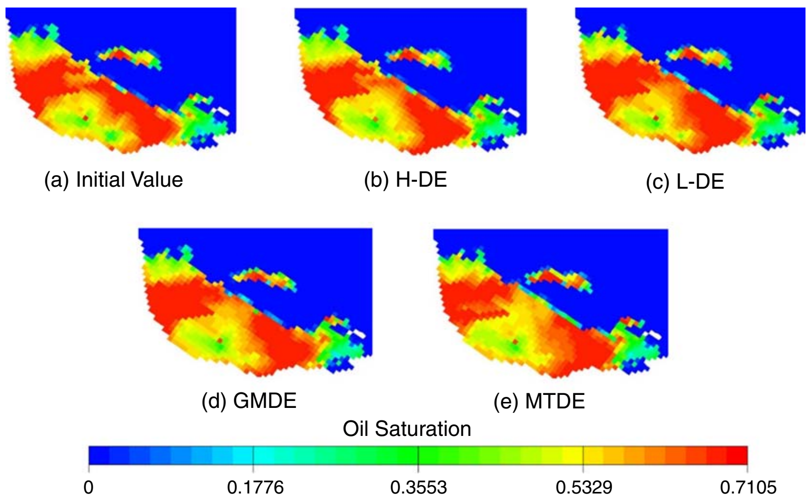


**Fig. 23—Remaining oil distribution of the first layer of edgewater-reservoir model by LHS, H-DE, L-DE, GMDE, and MTDE.**

## Discussion and Conclusions

In this work, we first introduce the optimization model in the context of multifidelity. Using this, a genetic transfer multifidelity optimization framework is constructed. Without loss of generality, differential evolution is chosen as the optimization algorithm for the framework. It is worth noting that any population-based algorithm that can adjust the step size and search direction can be used in the proposed framework (for example, global best, individual best, and learning factor in particle-swarm optimization). Therefore, the proposed framework can be further extended to the classical evolutionary algorithms and swarm-intelligence algorithms and also their improved versions. In addition, only two levels of fidelity are covered in this paper for simplicity, and both low- and high-fidelity information come from commercial numerical simulators. Our next step is to consider integrating the data-driven model mentioned in the Introduction section into a multifidelity optimization framework in place of a numerical simulator. Computing resources will further be saved in a proactive way to obtain inexpensive, low-fidelity information. Compared with the existing surrogate-assisted optimization algorithms, establishing a generalized genetic transfer framework is considered; this is suitable for any form of a data-driven model and population-based optimization algorithm. Moreover, because of the separation of low- and high-fidelity tasks in storage space, and the large difference in their evaluation time, paralleling multiple low-fidelity tasks and a high-fidelity task also has the potential to improve the computational efficiency of the framework.

The proposed MTDE algorithm consists of two modes. The b-transfer mode establishes an efficient multifidelity information-feedback mechanism. The p-transfer mode migrates population from the individual pool adaptively according to the relative fidelity information. We test MTDE in three production-optimization cases and compare it with the results from low-fidelity, high-fidelity, and greedy multifidelity methods. MTDE shows excellent performance in convergence speed and global search capability and generates the most-attractive well-control strategy.

## Nomenclature

| | |
|---|---|
| $b$ = | annual discount rate |
| $D_h$ = | high-fidelity domain |
| $D_l$ = | low-fidelity domain |
| $f_h$ = | high-fidelity objective function |
| $f_l$ = | low-fidelity objective function |
| $F_h$ = | fitness label evaluated by $f_h$ |
| $F_l$ = | fitness label evaluated by $f_l$ |
| $k$ = | current timestep index |
| $K$ = | total number of timesteps |
| $Mu$ = | amplification factor |
| $N$ = | population size |
| $\{\text{pool}\}$ = | individual pool |
| $Q_{o,k}$ = | oil-production rate, m$^3$/d |
| $Q_{wi,k}$ = | water-injection rate, m$^3$/d |
| $Q_{wp,k}$ = | water-production rate, m$^3$/d |
| $r_c$ = | crossover rate |
| $r_o$ = | crude-oil revenue, USD/m$^3$ |
| $r_{wi}$ = | cost of water injection, USD/m$^3$ |
| $r_{wp}$ = | cost of sewage treatment, USD/m$^3$ |
| $R_f$ = | relative fidelity |
| $t_k$ = | total development time at *k*th timestep, year |
| $T_l$ = | low-fidelity task |
| $T_h$ = | high-fidelity task |
| $T_{\text{ini}}$ = | initialization time, seconds |
| $T_{\text{max}}$ = | maximum time, seconds |
| $T_{\text{res}}$ = | rest allowed time, seconds |
| $\text{u}_\text{i}^\text{g}$ = | trial vector |
| $\text{v}_\text{i}^\text{g}$ = | mutant vector |

| | |
|---|---|
| $x_{h\text{best}}$ | = high-fidelity optimum |
| $x^{\text{lb}}$ | = lower boundary |
| $x_{l\text{best}}$ | = low-fidelity optimum |
| $x^{\text{ub}}$ | = upper boundary |
| $\Delta t^k$ | = length of $k$th timestep, days |
| $\varepsilon_0$ | = lowest allowable relative fidelity |
| $\lambda$ | = stall count |
| $\lambda_m$ | = guidance boundary parameter |
| $\lambda_M$ | = maximum allowable stall count |

**Superscript**

| | |
|---|---|
| $g$ | = generation |

**Subscripts**

| | |
|---|---|
| $h$ | = high-fidelity method |
| $i$ | = index of current individual |
| $l$ | = low-fidelity method |

## Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant Nos. 51722406, 52074340, and 51874335), the Shandong Provincial Natural Science Foundation (Grant No. JQ201808), the Fundamental Research Funds for the Central Universities (Grant No. 18CX02097A), the Major Scientific and Technological Projects of China National Petroleum Corporation (Grant No. ZD2019-183-008), the Science and Technology Support Plan for Youth Innovation of University in Shandong Province (Grant No. 2019KJH002), the National Science and Technology Major Project of China (Grant No. 2016ZX05025001-006), and the 111 Project (Grant No. B08028).

## References

Bratvedt, F., Bratvedt, K., Buchholz, C. F. et al. 1992. A New Front-Tracking Method for Reservoir Simulation. *SPE Res Eng* **7** (1): 107–116. SPE-19805-PA. https://doi.org/10.2118/19805-PA.

Broomhead, D. and Lowe, D. 1988. Radial Basis Functions, Multi-Variable Functional Interpolation and Adaptive Networks. Royal Signs and Radar Establishment Memorandum No. 4148, London, UK.

Chen, G., Li, Y., Zhang, K. et al. 2021. Efficient Hierarchical Surrogate-Assisted Differential Evolution for High-Dimensional Expensive Optimization. *Inf. Sci*. **542** (4 January): 228–246. https://doi.org/10.1016/j.ins.2020.06.045.

Chen, G., Zhang, K., Xue, X. et al. 2020a. Surrogate-Assisted Evolutionary Algorithm with Dimensionality Reduction Method for Water Flooding Production Optimization. *J Pet Sci Eng* **185** (February): 106633. https://doi.org/10.1016/j.petrol.2019.106633.

Chen, G., Zhang, K., Zhang, L. et al. 2020b. Global and Local Surrogate-Model-Assisted Differential Evolution for Waterflood Production Optimization. *SPE J*. **25** (1): 105–118. SPE-199357-PA. https://doi.org/10.2118/199357-PA.

Crane, M., Bratvedt, F., Bratvedt, K. et al. 2000. A Fully Compositional Streamline Simulator. Paper presented at the SPE Annual Technical Conference and Exhibition, Dallas, Texas, USA, 1–4 October. SPE-63156-MS. https://doi.org/10.2118/63156-MS.

Das, S. and Suganthan, P. N. 2011. Differential Evolution: A Survey of the State-of-the-Art. *IEEE Trans Evol Comput* **15** (1): 4–31. https://doi.org/10.1109/TEVC.2010.2059031.

de Baar, J. H. S. and Roberts, S. G. 2017. Multifidelity Sparse-Grid-Based Uncertainty Quantification for the Hokkaido Nansei-Oki Tsunami. *Pure Appl Geophys* **174** (8): 3107–3121. https://doi.org/10.1007/s00024-017-1606-y.

Drucker, H., Burges, C. J. C., Kaufman, L. et al. 1997. Support Vector Regression Machines. In *Neural Information Processing Systems*, Vol. 9, ed. M. C. Mozer, J. I. Jordan, and T. Petsche, 155–161. Cambridge, Massachusetts, USA: MIT Press.

Foroud, T. and Seifi, A. 2016. A Guided Pattern Search with a Non-Intrusive Reduced Order Modeling for Oil Production Optimization: Brugge Field Case Study. *J Pet Sci Eng* **147** (November): 570–584. https://doi.org/10.1016/j.petrol.2016.09.026.

Foroud, T., Baradaran, A., and Seifi, A. 2018. A Comparative Evaluation of Global Search Algorithms in Black Box Optimization of Oil Production: A Case Study on Brugge Field. *J Pet Sci Eng* **167** (August): 131–151. https://doi.org/10.1016/j.petrol.2018.03.028.

Foroud, T., Seifi, A., and AminShahidi, B. 2014. Assisted History Matching Using Artificial Neural Network Based Global Optimization Method–Applications To Brugge Field and a Fractured Iranian Reservoir. *J Pet Sci Eng* **123** (November): 46–61. https://doi.org/10.1016/j.petrol.2014.07.034.

Forrester, A. I. J. and Keane, A. J. 2009. Recent Advances in Surrogate-Based Optimization. *Prog. Aerosp. Sci*. **45** (1–3): 50–79. https://doi.org/10.1016/j.paerosci.2008.11.001.

Forrester, A., Sobester, A., and Keane, A. 2007. Multi-Fidelity Optimization via Surrogate Modelling. *Proc. R. Soc. A* **463** (2088): 3251–3269. https://doi.org/10.1098/rspa.2007.1900.

Golzari, A., Haghighat Sefat, M., and Jamshidi, S. 2015. Development of an Adaptive Surrogate Model for Production Optimization. *J Pet Sci Eng* **133** (September): 677–688. https://doi.org/10.1016/j.petrol.2015.07.012.

Guo, Z. and Reynolds, A. C. 2018. Robust Life-Cycle Production Optimization with a Support-Vector-Regression Proxy. *SPE J*. **23** (6): 2409–2427. SPE-191378-PA. https://doi.org/10.2118/191378-PA.

Guo, Z. and Reynolds, A. C. 2019. INSIM-FT in Three-Dimensions with Gravity. *J Comput Phys* **380** (1 March): 143–169. https://doi.org/10.1016/j.jcp.2018.12.016.

Gupta, A., Ong, Y.-S., and Feng, L. 2018. Insights on Transfer Optimization: Because Experience is the Best Teacher. *IEEE Trans Emerg Top Comput Intell* **2** (1): 51–64. https://doi.org/10.1109/tetci.2017.2769104.

Iqbal, M., Al-Sahaf, H., Xue, B. et al. 2019. Genetic Programming with Transfer Learning for Texture Image Classification. *Soft Comput*. **23** (23): 12859–12871. https://doi.org/10.1007/s00500-019-03843-5.

Jansen, J.-D., Fonseca, R.-M., Kahrobaei, S. et al. 2014. The Egg Model—A Geological Ensemble for Reservoir Simulation. *Geosci Data J* **1** (2): 192–195. https://doi.org/10.1002/gdj3.21.

Kennedy, M. and O'Hagan, A. 2000. Predicting the Output from a Complex Computer Code when Fast Approximations Are Available. *Biometrika* **87** (1): 1–13. https://doi.org/10.1093/biomet/87.1.1.

Koçer, B. and Arslan, A. 2010. Genetic Transfer Learning. *Expert Syst Appl* **37** (10): 6997–7002. https://doi.org/10.1016/j.eswa.2010.03.019.

Le Ravalec-Dupin, M. 2012. Optimizing Well Placement with Quality Maps Derived from Multi-Fidelity Meta-Models. Paper presented at the SPE Europec/EAGE Annual Conference, Copenhagen, Denmark, 4–7 June. SPE-154416-MS. https://doi.org/10.2118/154416-MS.

Lim, D., Ong, Y., Jin, Y., et al. 2008. *Evolutionary Optimization with Dynamic Fidelity Computational Models*, Vol. 5227. Berlin, Germany: Lecture Notes in Computer Science Series, Springer.

McKay, M. D., Beckman, R. J., and Conover, W. J. 2000. A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics* **42** (1): 55–61. https://doi.org/10.1080/00401706.2000.10485979.

Neill, D. O., Al-Sahaf, H., Xue, B. et al. 2017. Common Subtrees in Related Problems: A Novel Transfer Learning Approach for Genetic Programming. *Proc*., IEEE Congress on Evolutionary Computation, San Sebastian, Spain, 5–8 June, 1287–1294. https://doi.org/10.1109/CEC.2017.7969453.

Sacks, J., Welch, W. J., Mitchell, T. J. et al. 1989. Design and Analysis of Computer Experiments. *Statist. Sci*. **4** (4): 409–423. https://doi.org/10.1214/ss/1177012413.

Storn, R. and Price, K. 1997. Differential Evolution—A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *J Glob Optim* **11** (4): 341–359. https://doi.org/10.1023/A:1008202821328.

Teixeira, A. F. and Secchi, A. R. 2019. Machine Learning Models To Support Reservoir Production Optimization. *IFAC-PapersOnLine* **52** (1): 498–501. https://doi.org/10.1016/j.ifacol.2019.06.111.

Thenon, A., Gervais, V., and Le Ravalec, M. 2016. Multi-Fidelity Proxy Models for Reservoir Engineering. Paper presented at ECMOR XV–15th European Conference on the Mathematics of Oil Recovery, Amsterdam, The Netherlands, 29 August–1 September. https://doi.org/10.3997/2214-4609.201601831.

Wang, H., Jin, Y., and Jansen, J. O. 2016. Data-Driven Surrogate-Assisted Multiobjective Evolutionary Optimization of a Trauma System. *IEEE Trans Evol Comput* **20** (6): 939–952. https://doi.org/10.1109/TEVC.2016.2555315.

Weber, D. B. 2009. *The Use of Capacitance-Resistance Models To Optimize Injection Allocation and Well Location in Water Floods*. PhD dissertation, University of Texas, Austin, Texas, USA.

Yondo, R., Andrés, E., and Valero, E. 2018. A Review on Design of Experiments and Surrogate Models in Aircraft Real-Time and Many-Query Aerodynamic Analyses. *Prog. Aerosp. Sci*. **96** (January): 23–61. https://doi.org/10.1016/j.paerosci.2017.11.003.

Yousef, A. A., Gentil, P. H., Jensen, J. L. et al. 2006. A Capacitance Model To Infer Interwell Connectivity from Production and Injection Rate Fluctuations. *SPE Res Eval & Eng* **9** (6): 630–646. SPE-95322-PA. https://doi.org/10.2118/95322-PA.

Yuan, Y., Ong, Y., Gupta, A. et al. 2017. Objective Reduction in Many-Objective Optimization: Evolutionary Multiobjective Approaches and Comprehensive Analysis. *IEEE Trans Evol Comput* **22** (2): 189–210. https://doi.org/10.1109/TEVC.2017.2672668.

Zaytsev, A. and Burnaev, E. 2017. Large Scale Variable Fidelity Surrogate Modeling. *Ann Math Artif Intell* **81** (1–2): 167–186. https://doi.org/10.1007/s10472-017-9545-y.

Zhang, K., Zhao, X., Zhang, L. et al. 2020a. Current Status and Prospect for the Research and Application of Big Data and Intelligent Optimization Methods in Oilfield Development. *Journal of China University of Petroleum (Edition of Natural Science)* (4): 28–38. https://doi.org/10.3969/j.issn.1673-5005.2020.04.004.

Zhang, L., Xu, C., Zhang, K. et al. 2020b. Production Optimization for Alternated Separate-Layer Water Injection in Complex Fault Reservoirs. *J Pet Sci Eng* **193** (October): 107409. https://doi.org/10.1016/j.petrol.2020.107409.

Zhao, H., Kang, Z., Zhang, X. et al. 2016. A Physics-Based Data-Driven Numerical Model for Reservoir History Matching and Prediction with a Field Application (Associated Discussion Available as Supporting Information). *SPE J*. **21** (6): 2175–2194. SPE-173213-PA. https://doi.org/10.2118/173213-PA.

Zhao, M., Zhang, K., Chen, G. et al. 2020a. A Surrogate-Assisted Multi-Objective Evolutionary Algorithm with Dimension-Reduction for Production Optimization. *J Pet Sci Eng* **192** (September): 107192. https://doi.org/10.1016/j.petrol.2020.107192.

Zhao, M., Zhang, K., Chen, G. et al. 2020b. A Classification-Based Surrogate-Assisted Multiobjective Evolutionary Algorithm for Production Optimization Under Geological Uncertainty. *SPE J*. **25** (5): 2450–2469. SPE-201229-PA. https://doi.org/10.2118/201229-PA.

Zhao, X., Zhang, K., Chen, G. et al. 2020c. Surrogate-Assisted Differential Evolution for Production Optimization with Nonlinear State Constraints. *J Pet Sci Eng* **194** (November): 107441. https://doi.org/10.1016/j.petrol.2020.107441.

## Appendix A—Pseudocode of MTDE

```
Algorithm MTDE
Input population size $N$, initial amplification factor $Mu_0$, crossover rate $r_c$, maximum time $T_{\max}$, lower boundary $x^{lb}$, upper boundary $x^{ub}$
Output $x_{h\mathrm{best}}$, $F_{h\mathrm{best}}$

1     Sampling population $\{x\}_N$ using LHS($NP$, $x^{lb}$, $x^{ub}$)
2     / *** start b-transfer mode *** /
2-1   Initialization
      Initialize population $\{x\}_N$ using $f_l$ and individual pool $\{pool\}_1$ using Eq. 11;
      get $F_{l\mathrm{best}} = f_l(x_{l\mathrm{best}})$ and $F_{h\mathrm{best}} = f_h(x_{h\mathrm{best}})$;
      calculate $\lambda_m$ using Eq. 12; $c = N$;
      While the computational budgets $T_{\max}$ have not exhausted do
         For $i = 1$ to $N$ do
```

Table A-1—Pseudocode of MTDE.

**Algorithm** MTDE

```
2-2     / *** adaptive strategy (a part) *** /
          If $c < 2*\lambda_m + N$ do Exploration (Principle A): $flag = 1$ , $Mu = Mu_0$;
          Else if $c == 2*\lambda_m + N$ do Exploitation (Principle B): $flag = 0$ , $\lambda = 0$;
          Else do
            If $\lambda < \lambda_m$ do Exploitation (Principle B): $flag = 0$;
            Else do Exploitation (Principle C): $flag = 1$; End if
          calculate $Mu$ using Eq. 20
          End if
2-3     / *** Mutation and crossover *** /
          If $flag = 0$ do get mutation vector $v_i^g$ using Eqs. 14 through 17; End if
          If $flag = 1$ do get mutation vector $v_i^g$ using Eq. 7; End if
          Get crossover vector $u_i^g$ according to Eq. 9;
          Check the constraints
2-4     / *** adaptive strategy (b part) *** /
          Get $F_l = f_l(u_i^g)$; $c = c + 1$;
          If $F_l < F_{l\text{best}}$ do
            $F_{l\text{best}} = F_l$; get $F_h = f_h(u_i^g)$; update $\{\text{pool}\}_t$;
            If $F_h < F_{h\text{best}}$ do
              $F_{h\text{best}} = F_h$, $x_{h\text{best}} = u_i^g$; $\lambda = 0$; update $R_f$;
            Else do $\lambda = \lambda + 1$; End if
          Else do $\lambda = \lambda + 1$; End if
2-5     / *** Selection *** /
          Update $x_i^g$ according to Eq. 10;
          Check transfer criteria, start p-transfer if satisfied;
          Check stop criteria, output $x_{h\text{best}}$ and $F_{h\text{best}}$ if satisfied;
        End for
      End while
      / *** end b-transfer mode *** /
3     / *** start p-transfer mode *** /
      Construct transfer population $\{x\}_N$ by sorted $\{\text{pool}\}_t$ , and
      If $t < N$ do infill rest part of $\{x\}_N$ using LHS($N$, $x^{\text{lb}}$, $x^{\text{ub}}$) and initialize; End if
      While the computational budgets $T_{\max}$ have not exhausted do
        For $i = 1$ to $N$ do
          mutation using Eq. 8; crossover using Eq. 9; selection using Eq. 10;
          Check stop criteria, output $x_{h\text{best}}$ and $F_{h\text{best}}$ if satisfied;
        End for
      End while
      / *** end p-transfer mode *** /
```

Table A-1 (continued)—Pseudocode of MTDE.