# Surrogate-assisted inversion for large-scale history matching

## Comparative study between projection-based reduced-order modeling and deep neural network

Xiao, Cong; Lin, Hai-Xiang ; Leeuwenburgh, Olwijn; Heemink, Arnold

**DOI**
10.1016/j.petrol.2021.109287

**Publication date**
2022

**Document Version**
Final published version

**Published in**
Journal of Petroleum Science and Engineering

Journal of Petroleum Science and Engineering 208 (2022) 109287

Contents lists available at ScienceDirect

Journal of Petroleum Science and Engineering

journal homepage: www.elsevier.com/locate/petrol

# Surrogate-assisted inversion for large-scale history matching: Comparative study between projection-based reduced-order modeling and deep neural network

Cong Xiao [a,b,c,*], Hai-Xiang Lin [c], Olwijn Leeuwenburgh [d,e], Arnold Heemink [c]

[a] Key Laboratory of Petroleum Engineering, Ministry of Education, China University of Petroleum, Beijing, 102249, China
[b] College of Petroleum Engineering, China University of Petroleum, Beijing, 102249, China
[c] Delft Institute of Applied Mathematics, Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands
[d] Civil Engineering and Geosciences, Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands
[e] TNO Princetonlaan 6, PO Box 80015, 3508 TA Utrecht, The Netherlands

ARTICLE INFO

*Keywords:*
Reservoir simulation
Deep learning
Reduced-order modeling
Data assimilation
Stochastic optimization

ABSTRACT

History matching can play a key role in improving geological characterization and reducing the uncertainty of reservoir model predictions. Application of reservoir history matching is restricted by the huge computational cost by amongst others the many runs of the full model. Surrogate models with a reduced complexity are therefore used to reduce the computational demands. This paper presents an efficient surrogate-assisted deterministic inversion framework to primarily explore the possibility of applying deep neural network (DNN) surrogate to approximate the gradient of large-scale history matching by using auto-differentiation (AD). In combination with the deep neural network model, the AD enables us to evaluate the gradients efficiently in a parallel manner. Furthermore, the benefits of using stochastic gradient optimizers in the deep learning practice, instead of full gradient optimizers in conventional deterministic inversions, is investigated as well. Numerical experiments are conducted on a 3D benchmark reservoir model in the context of a water-flooding production scenario. The quantity of interest, e.g., dynamic saturation for an ensemble of test models, can be accurately predicted. The proposed surrogate-assisted inversion with stochastic gradient optimizer obtains a very quick convergence rate against the model and data noise for the high-dimensional history matching problem with a large number of data and parameters. In addition, we also conduct several comparisons and evaluations with our previously proposed projection-based subdomain POD-TPWL approach in terms of computational efficiency and accuracy. The subdomain POD-TPWL constructs a local surrogate model, which is repeatedly reconstructed a number of times for maintaining a satisfactory accuracy, while DNN constructs a global surrogate model based on the entire training data and generally does not require additional reconstructions. The subdomain POD-TPWL is very sensitive to how the domain is decomposed, increasing the training samples does not infinitely improve the history matching results by a fixed decomposition. Overall, these two kinds of surrogate models have demonstrated great potential in solving large-scale history matching problem. The DNN surrogate is particularly useful to generate multiple posteriors for model uncertainty quantification.

## 1. Introduction

We address the problem of computationally efficient estimation of spatially varying parameters in large-scale simulation models in petroleum engineering which is known as history matching (Aanonsen et al., 2009). Measured data are typically obtained at wells, which are normally sparsely distributed over large areas, leaving the generally heterogeneous reservoir rock in-between the wells unsampled. Alternative ways of gathering information are based on geophysical techniques, such as time-lapse (4D) seismic data, which registers the arrival time and amplitude of acoustic waves that are reflected at contrasts in acoustic impedance, which in turn is affected by the density and mechanical properties of the rock and fluids. The availability of such geophysical information is often less frequent in time and of low spatial resolution, but much denser compared to the well measure points. In contrast to sparse well data, seismic data generally provides sufficient spatial-information about the dynamic changes in the oil reservoirs (Mannseth and Fossum, 2018).

* Corresponding author at: Key Laboratory of Petroleum Engineering, Ministry of Education, China University of Petroleum, Beijing, 102249, China.
*E-mail address:* xclmjtud@yahoo.com (C. Xiao).

https://doi.org/10.1016/j.petrol.2021.109287
Received 17 September 2020; Received in revised form 6 February 2021; Accepted 26 July 2021
Available online 8 August 2021

Incorporating 4D seismic data into the history matching workflow, hereinafter referred to as seismic history matching (hereinafter referred to as SHM), has been investigated for several decades. The optimization algorithms for addressing SHM problem in the field of petroleum engineering have been investigated for several decades, among them, the gradient-based deterministic inversion is one of commonly used approaches nowadays. We generally define a nonlinear objective function as the sum of squares of the difference between the recorded data and simulated predictions through running a reservoir model. Gradient-based optimization, using the adjoint model, is used to minimize the objective function through iteratively adjusting the model parameters (Courant and Hilbert, 1962). The high dimensionality of grid-based seismic measurements and strong non-linearity, however, pose strong challenges, e.g., expensive model simulations and algorithm convergence. Although many efforts have been taken to make the implementation of the adjoint model feasible (Courtier et al., 1994), a fast evaluation of the objective function and its gradient information still remains a big computational challenge. In some cases it may not be feasible to perform the history matching within an acceptable short time frame, because it requires numerous simulations, e.g., simulations of multiple models for the gradient approximation. The surrogate modeling has recently gained more and more popularity in case many forward simulations must be performed, like in history matching problems. The main focus of this paper is on investigating surrogate-assisted optimization strategy for large-scale history matching, e.g., grid-based seismic data.

Surrogate modeling aims at providing a faster emulation with a simplified relation between the inputs and outputs of a complex model. It has the potential of accelerating the gradient-based optimization problems. Existing surrogate modeling approaches can be roughly classified into three categories: hierarchical-based, projection-based reduced-order model (ROM) and data-driven surrogate models. Reviews of surrogate modeling can be found in Asher et al. (2015). In hierarchical-based methods the surrogate is created by simplifying the representation of the physical system, such as by ignoring certain processes, or reducing the numerical resolution or the complexity, e.g., upscaling and grid coarsening (Zhang et al., 2008; Salazar et al., 2007), or the complexity, e.g., nonlinearity. Reduced-order modeling approaches have been always increasing popularity as one of the most effective means to reduce the computational effort of model-based workflows through reduction of the number of dimensions of the model. The main idea behind projection-based ROM is to construct a (linear) low-order surrogate model by projecting the dynamics of the system onto the subspace of dominant variability of the model dynamics. Most ROM strategies employ proper orthogonal decomposition (POD) (Vermeulen and Heemink, 2006; Altaf et al., 2009) of time series of model state 'snapshots' to identify an orthogonal set of basis functions of the subspace (Kaleta et al., 2011; Xiao et al., 2018). Such model-order reduction strategies have been applied with success in speeding up model simulations in computational fluid mechanics (Xiao et al., 2014, 2016) and subsurface flow simulations (Markovinović and Jansen, 2006; Cardoso et al., 2009).

The aforementioned ROM accurately represents the high-fidelity forward model in the reduced subspaces spanned by the projection basis which contains the hidden physics. By contrast, data-driven approaches purely depend on data (simulated or real data) to approximate the relation between input and output as accurately as possible, such as polynomial chaos expansion (Dai et al., 2016) and fully-connected artificial neural networks (Ahmadi, 2015). These data-driven surrogate models have demonstrated their feasible applicability to some degree, however their use is restricted only to relatively small-dimensional problems and therefore suffers from the *curse of dimensionality* and fail to work for large-scale problems. The deep neural network (DNN) has recently attracted attention because of successful applications in several fields, including computer vision (Heo et al., 2018) and image processing (Young et al., 2018). Recent advances in DNN and their promising application results have prompted research on deep-learning-based surrogate modeling for high-dimensional nonlinear systems (Mo et al., 2018, 2019). The popularity of these methods is further enhanced by the availability of open access machine learning frameworks, such as TensorFlow (Abadi et al., 2016) and PyTorch (Paszke et al., 2019).

The projection-based ROM surrogate has been applied previously, so the focus of the paper is to explore the possibility of DNN to approximate the gradients for significantly increasing the efficiency of the SHM problems, and to compare and evaluate with our previously proposed projection-based POD-TPWL approach (Xiao et al., 2019a,b). The application of projection-based ROM to gradient-based optimization problem has been extensively investigated. Cardoso et al. were the first to integrate POD with trajectory-piecewise-linearization (TPWL), i.e., referred to as POD-TPWL, for addressing oil production optimization (Cardoso and Durlofsky, 2010). Subsequently, He et al. (2014) also applied POD-TPWL to reservoir history matching problems. The standard POD-TPWL enforces us to access to the source code, which definitely hinders its applicability to more generic situations where the source code is not available, especially for the commercial simulators. To fill this gap, Vermeulen and Heemink (2006), Altaf et al. (2009) and Kaleta et al. (2011) proposed a code non-intrusive reduced-order linear model to approximate the high-dimensional nonlinear system. The adjoint of this reduced-order linear model can be easily constructed for minimizing the objective function efficiently. This method uses any simulator as black box without intruding the simulation code, however, the derivatives for the model linearization are approximated through a standard finite difference method, which involves perturbing the parameters one by one and is therefore computationally less attractive for large scale problems with many uncertain parameters. Xiao et al. have recently proposed a subdomain projection-based ROM, hereinafter referred to as subdomain POD-TPWL, through using domain decomposition and radial-basis function interpolation method (Xiao et al., 2019a,b). The domain decomposition strategy individually handles model dynamics for each spatial subdomains with much smaller dimension, while the radial-basis function interpolation method replaces the original model with a set of non-intrusive interpolation functions. Subdomain POD-TPWL has demonstrated to be very promising in speeding up gradient-based reservoir history matching.

The DNN models are generally trained by iteratively adjusting the trainable variables that parameterize the neural network model, during which the general auto-differentiation (or back-propagation in training neural networks) is used to calculate the gradient of the loss function with respect to neural network parameters, e.g., weights and biases. There is a clear similarity in the way how DNN and SHM iteratively update parameters, e.g., geological parameters in SHM. In this sense, the training of DNN is not that far from the gradient-based SHM. Due to the availability of auto-differentiation (AD) and high-performance computing units, the evaluation of gradients for DNN is much more efficient than that of conventional SHM procedure. An interesting question remains how to take full use of the computational advantage of DNN to address the SHM problem in a short time frame.

The preliminary results presented in our previous work have shown great potential of applying DNN to efficiently assist gradient-based reservoir history matching (Xiao et al., 2020). The contribution of this paper to the large-scale reservoir history matching is investigating two kinds of surrogate models and verifying their applicability to speedup gradient-based SHM problems. Specifically, the first objective of this work is to explore the possibility of using the DNN surrogate to approximate the gradient for reservoir history matching, which has not been fully investigated yet. Addressing gradient-based SHM with the use of a DNN surrogate has the advantage of facilitating the use of deep learning packages. It might benefit from powerful concepts such as AD in the deep learning packages for obtaining gradients in a very efficient manner. Since both DNN surrogate and our previously proposed subdomain POD-TPWL are intentionally used for the gradient-based history matching, a comparative study between them is identified as the second research objective in this paper.

The remainder of this paper is as follows: The gradient-based reservoir history matching framework is defined in Section 2. Section 3 describes the deep-learning surrogate model, in which a residual U-Net and an explicit concatenation of time feature are integrated to capture both spatial and temporal features. The gradient approximation for SGD is described as well. Section 4 assesses the surrogate-assisted gradient-based inversion framework on a 3D benchmark reservoir model. A comparative study between the DNN surrogate model and our previously proposed projection-based subdomain POD-TPWL is also presented. Finally, Section 5 summarizes our contribution and discusses future work.

## 2. Definition of gradient-based reservoir history matching

History matching real time-lapse seismic data requires the capability to compute seismic data from a given reservoir model. The entire workflow generally requires running two consecutive forward models: the fluid flow simulation model and the rock–physics model, e.g., Gassmann model (Gassmann, 1951). The former predicts the reservoir state variables (pressure and saturation), and then the latter simulate the seismic response from the reservoir state variables. In this study, the main data used on the history matching procedure is a saturation map observed in different times, which mimic the data obtained from interpretation of the real 4D seismic survey.

To simplify the notation without loss of generality, we give an explicit formula for a single fluid flow simulation step as follows,

$$\mathbf{x}^n = \mathbf{f}^n(\mathbf{x}^{n-1}, \mathbf{m}), \quad n = 1, \dots, N_t \tag{1}$$

where, $\mathbf{m} \in R^{N_\mathbf{m}}$ denotes the vector of spatial parameters for reservoir model. $N_\mathbf{m}$ is the total number of gridblocks. $n$ denotes the simulation step. $\mathbf{x}^n \in R^{2N_m}$ represents the state vectors (e.g., including pressure and oil/water saturation in all gridblocks). The seismic data is directly related to fluid saturation and pressure changes and provides information on the dynamic behavior of the reservoirs. For simplification, we directly measure fluid saturation in each grid-block without running the rock–physics model for generating the real elastic properties. That is to say, $\mathbf{x}^n$ directly represents the simulated data.

We are concerned with parameter estimation in heterogeneous oil reservoir models. Uncertain parameters can be estimated by minimizing an objective function, which can be formulated within the Bayesian framework (Evensen, 2009). The solution will represent the posterior probability density function (PDF) of model parameters $\mathbf{m}$ conditioned to measurements $\mathbf{d}_{obs}$ with an assumption that the prior PDF of parameters $\mathbf{m}$ is Gaussian with a mean $\mathbf{m}_b$ and covariance matrix $\mathbf{R}_m$.

Gradient descent algorithms, or more specifically a full-gradient descent (FGD), are generally used to minimize the objective function in an iterative manner. The general gradient-based history matching can be defined as a least squares optimization formula. The objective function, denoted as $J$ here, is defined using full dataset with high-fidelity model (HFM) as follows.

$$J(\mathbf{x}^1, \dots, \mathbf{x}^{N_t}, \mathbf{m}) = \frac{1}{2}(\mathbf{m} - \mathbf{m}_b)^T \mathbf{R}_m^{-1}(\mathbf{m} - \mathbf{m}_b) + \frac{1}{2}\sum_{n=1}^{N_t}(\mathbf{d}_{obs}^n - \mathbf{x}^n)^T [\mathbf{R}_{obs}^n]^{-1}(\mathbf{d}_{obs}^n - \mathbf{x}^n) \tag{2}$$

where, the measurement errors for the data gathered at the timestep $n$ are generally assumed to satisfy a Gaussian distribution $N(\mathbf{0}, \mathbf{R}_{obs}^n)$, where $\mathbf{R}_{obs}^n$ represents the measurement error covariance matrix.

After defining the objective functions, the key step of a gradient-based minimization algorithm is to determine the gradient of the objective function with respect to the parameters. In this paper, both projection-based ROM and DNN surrogate are used to approximate the gradients in an efficient manner.

## 3. Surrogate modeling using deep neural network

This section introduces the procedures of using a data-driven DNN to approximate the relation between geological parameters and saturation. The neural network architecture, model training procedure and gradient approximation for the stochastic gradient descent are described. More details about the neural network architecture, training data preparation and procedure of model training can be found in the provided supplementary material.

### 3.1. Neural network architecture

Eq. (1) describes the dependence of the state variables and parameters at two consecutive time steps, e.g., $\mathbf{x}^n$, $\mathbf{x}^{n-1}$ and $\mathbf{m}$. The main task of our proposed neural network model is to construct a simple mapping directly from permeability field $\mathbf{m}$ to saturation field $\mathbf{x}^n$ at the specific time step $n$. The relationship between simulated saturation and spatial parameter fields can be described by a nonlinear operator. To simplify the notation without loss of generality, one such generic operator $\mathbf{h}^n$ can be simply described as follows,

$$\mathbf{x}^n = \mathbf{h}^n(\mathbf{m}), \quad n = 1, \dots, N_t \tag{3}$$

and a deep-learning based surrogate model representing the time-dependent process can be presented as follows:

$$\hat{\mathbf{x}}^n = \hat{\mathbf{h}}^n(\mathbf{m}, t^n), \quad n = 1, \dots, N_t \tag{4}$$

where, $\hat{\mathbf{x}}_n \in R^{N_x \times N_y \times N_z}$ is the neural network prediction (an 3D image) for the input $\mathbf{m} \in R^{N_x \times N_y \times N_z}$ at the time $\mathbf{t}^n$. $N_x$, $N_y$ and $N_z$ represent the image size. $\theta$ denotes all the network trainable parameters. In this model, the current output $\hat{\mathbf{x}}_n$ depends only on the time-independent input permeability field $\mathbf{m}$ and time $\mathbf{t}^n$.

Recent applications of deep neural network to subsurface flow simulations have been extensively reported. For example, Jin et al. proposed a data-driven DNN surrogate model with autoregressive structure for approximating the time-varying process in reservoir simulation problem (Jin et al., 2019). Tang et al. (2019) developed a deep convolutional recurrent neural network architectures, specifically a combination of auto-encoder and a convolutional long short term memory recurrent network (convLSTM) (Xingjian et al., 2015). Although autoregressive structure excels at temporal regression tasks, they will definitely encounter a time-dependent error accumulation problem. The recurrent neural network might become computationally demanding for the long time-serial models (Zhou and Tartakovsky, 2020), although the time-dependent error accumulation problem hardly occurs.

To address the aforementioned drawbacks, we use a new deep convolutional neural network, namely time-conditioning residual U-Net (cR-U-Net), based on an integration of auto-encoder structure and the well-known residual U-Net architecture, which was originally proposed for bio-medical image segmentation (Ronneberger et al., 2015). The residual U-Net architecture has demonstrated to be very effective for the cross-domain regression problem (Zhong et al., 2019), such as the mapping from the spatial parameter fields to simulated saturation field studied in the paper. To learn the temporal features of dynamic models, a time-conditioning feature, as an additional channel, is concatenated to the low-dimensional representation features after the encoder part (Mo et al., 2018). The time feature is represented as a map of the same dimension as the low-dimensional representation features. Each element of the time feature map is equal to a specific time value. This is different from the above two approaches that use either autoregressive or recurrent structures to capture the time-serial dynamics and hence is capable of effectively mitigating the error accumulation and computation issue for the long time-serial models. In addition, embedding the time feature into the network architecture enables to predict the model states at the times which are not used in training the network model. This strategy increases the generality and interpolation ability of the cR-U-Net surrogate model.

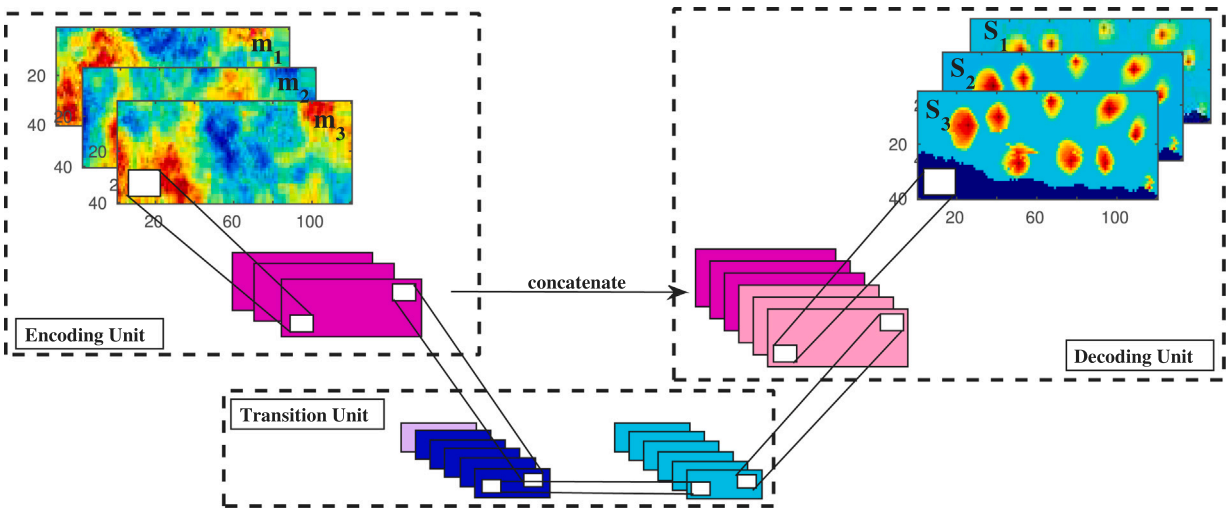



**Fig. 1.** Schematic illustration of the cR-U-Net architecture for a 2D input image following the literature (Tang et al., 2019). The cR-U-Net is composed of encoding unit, transition unit and decoding unit. The multi-scale features extracted in the encoding unit are concatenated with the upsampling features in the decoding unit to produce the final output. The transition unit concatenates and forwards the extracted multi-scale features and time feature, and then feeds them to the decoding unit. The gray shape in the transition unit represents the time feature map.

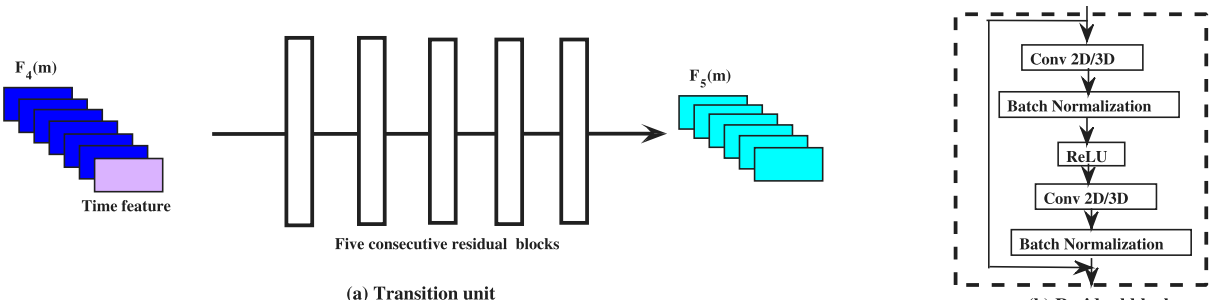



**Fig. 2.** Schematic illustration of transition unit. A stack of five residual blocks is used to propagate the output of the encoding unit. In addition, the time feature as an additional channel is fed to this unit as well to capture the flow dynamic.

The configuration of cR-U-Net is guided by the literature (Tang et al., 2019) where a similar R-U-Net architecture is designed. This cR-U-Net architecture contains two symmetric units, an encoding path and a decoding path, to capture the spatial features of the input and the output images. In addition, we also design a transition unit, including a stack of residual blocks, for connecting the encoding and the decoding unit. Fig. 1 illustrates a schematic diagram of the proposed cR-U-Net which synoptically displays the arrangement of encoding, transition and decoding unit. In this network, each convolutional block consists of three serial operations ($Conv3D - BatchNorm - ReLU$), including a 3D convolutional layer ($Conv3D$), a batch normalization layer ($BatchNorm$) and a rectified linear activation unit ($ReLU$). The symmetric encoding and decoding units are to downsample and upsample the feature maps using the 3D convolutional ($Conv3D$) and transposed 3D convolutional ($TConv3D$) operation, respectively. As illustrated in Fig. 3(a), this transition unit is composed of five residual-blocks, which intentionally construct connections between non-adjacent layers aimed at propagating the features produced from the previous layers to more deeper layers (Huang et al., 2016). The illustration of a residual block can be found in Fig. 2(b). The residual blocks, specifically residual convolutional ($resConv$) blocks, also can help cope with the gradient vanishing/explosion problem especially for very deep networks (He et al., 2016). More details about the architecture of the encoding, decoding and transition units can be found in the Supplementary material.

After configuring the neural network architecture, we employ the popular stochastic-gradient optimizer, i.e., $Adam$, to optimize the neural network parameters $\boldsymbol{\theta}$ (Kingma and Ba, 2014). Details about the iterative scheme of $Adam$ optimizer will be given in the next section. Once the neural network is trained, the predictions corresponding to new inputs are straightforward. Given an arbitrary input, repeated implementation of Eq. (4) are used to evolve this time-varying states for all $N_t$ timesteps. Specifically, the saturation $\hat{\mathbf{x}}^n$ at the $n$th timestep is sequentially predicted by providing the permeability input $\mathbf{m}$ and the time $t^n$. The computational cost for the new predictions almost can be neglected as compared to additional high-fidelity model simulations.

### 3.2. Gradient approximation using auto-differentiation (AD)

The use of DNN surrogate model in SHM enables the entire SHM procedure completely implemented using the deep learning packages. It can benefit from the deep learning practice of using minibatches of data with the stochastic-gradient descent (SGD) optimizers instead of the more conventional approach of applying an optimizer such as the FGD, e.g., as defined in Eq. (2). The robustness and faster convergence of SGD optimizers have been extensively demonstrated in the deep-learning community (Goodfellow et al., 2016). In this work, we explore the potential of applying SGD to improve large-scale history matching performance. The attractive benefits of using SGD stem from its efficiency for problems with a large number of data and parameters and its effectiveness for problems with noisy gradients.

Compared to FGD during which we have to run through all data points at $N_t$ measured time steps for a single update of the parameters in a particular iteration, SGD enables us to use the data points at only one measured time step (e.g., $n \in [1, N_t]$) or a subset $\mathbf{bz}$ of $[1, N_t]$ (e.g., a collection of several random selections of indices from $[1, N_t]$) to

update the parameters. If the size of dataset is huge, e.g., the spatially dense seismic data frequently collected at some time steps, it may take too long for FGD to minimize the objective function because we need use the full dataset for updating the parameters in each iteration. In contrast, SGD uses only one or a subset of data and it starts update the parameters rapidly.

This paper introduces the stochastic gradient as an approximation of the full gradient that can be used to speed up calculations and algorithm convergence. After training the cR-U-Net surrogate model, the approximated stochastic objective function, denoted as $J_N$, can be defined using a subset **bz** with trained neural network model correspondingly

$$J_N = \frac{1}{2}(\mathbf{m}-\mathbf{m}_b)^T \mathbf{R}_m{}^{-1}(\mathbf{m}-\mathbf{m}_b) + \frac{1}{2}\sum_{n\in\mathbf{bz}}[\mathbf{d}^n_{obs}-\hat{\mathbf{h}}^n(\mathbf{m},t^n)]^T[\mathbf{R}^n_{obs}]^{-1}[\mathbf{d}^n_{obs}-\hat{\mathbf{h}}^n(\mathbf{m},t^n)] \quad (5)$$

and its corresponding stochastic gradient with respect to the parameters **m** using the sensitivity or Jacobian matrix

$$\mathbf{g}_{N,1} = \mathbf{R}_m{}^{-1}(\mathbf{m}-\mathbf{m}_b) - \sum_{n\in\mathbf{bz}}[\frac{\partial\hat{\mathbf{h}}^n}{\partial\mathbf{m}}]^T[\mathbf{R}^n_{obs}]^{-1}[\mathbf{d}^n_{obs}-\hat{\mathbf{h}}^n(\mathbf{m},t^n)] \quad (6)$$

The computation of objective function gradient can be realized based on two methods. (1) The first method uses the model sensitivities $\frac{\partial\hat{\mathbf{h}}^n}{\partial\mathbf{m}}$, to compute the gradient $\mathbf{g}_{N,1}$, e.g., Eq. (6). Each element of this matrix is the derivative of the surrogate model outputs, $\hat{\mathbf{h}}^n(\mathbf{m})$, with respect to the parameters **m**. e.g., grid-based geological permeability in this paper. (2) The second method directly calculates the gradient of the objective function, $J_N$ with respect to **m**, denoted to be $\mathbf{g}_N$. Deep-learning packages provide sufficient built-in modules to efficiently compute the function with a least-square formula, e.g., Eq. (5), and its gradient using AD. That is, the second method does not explicitly involve the computation of sensitivities. In general, $\mathbf{g}_N$ is often cheaper to calculate than that of model sensitivities $\frac{\partial\hat{\mathbf{h}}^n}{\partial\mathbf{m}}$ in terms of the computational complexity.

Once the SHM algorithm is implemented in the deep learning packages, some sophisticated evaluations, e.g., objective function evaluation (Eq. (5)) and gradient approximation (Eq. (6)) involving high-dimensional matrix–matrix/vector multiplications, can be quickly computed. For example, they can be executed on multi-CPUs and GPUs, and run in parallel across a distributed memory computer cluster. All these features form the main motivation for investigating the use of DNN surrogate model for the SHM problem.

### 3.3. Stochastic gradient descent optimization

To minimize surrogate objective function $J_N$, a standard stochastic gradient descent (SGD) algorithm updates the parameters at $k$th optimization step

$$\mathbf{m}^{k+1} \leftarrow \mathbf{m}^k - \varepsilon^k \mathbf{g}^k_N \quad (7)$$

The aforementioned *Adam* optimization algorithm has recently seen broader applications in the community of deep learning. *Adam* leverages past gradient information to retard the descent along large gradients. This information is stored in the momentum vector **u** and squared element-wise gradient vector **v** as

$$\mathbf{u}^k = b_u\mathbf{u}^{k-1} + (1-b_u)\mathbf{g}^k_N; \quad \hat{\mathbf{u}}^k = \frac{\mathbf{u}^k}{1-\mathbf{u}^k}$$
$$\mathbf{v}^k = b_v\mathbf{v}^{k-1} + (1-b_v)[\mathbf{g}^k_N]^2; \quad \hat{\mathbf{v}}^k = \frac{\mathbf{v}^k}{1-\mathbf{v}^k} \quad (8)$$

where, $\hat{\mathbf{u}}^k$ and $\hat{\mathbf{v}}^k$ are the unbiased momentum and squared gradient vectors, respectively. The gradient descent step proceeds as follows

$$\mathbf{m}^{k+1} = \mathbf{m}^k - \zeta^k \frac{\hat{\mathbf{u}}^k}{\sqrt{\hat{\mathbf{v}}^k}+\epsilon} \quad (9)$$

The above update is performed element-wise and $\epsilon$ is a small number (e.g. $10^{-8}$) to avoid any division by zero in the implementation. Here we use commonly recommended default parameters values of $\varepsilon^k = 0.001$, $b_u = 0.9$, $b_v = 0.999$ and $\epsilon = 10^{-8}$ in this paper.

For the implementation of the SHM procedure with the deep neural network surrogate, we have used the open source deep learning package PyTorch to build a hybrid CPU/GPU computing framework, where the training data are generated using the open-source simulator OPM-*Flow* (Rasmussen et al., 2019) running on CPUs while both surrogate model training and seismic history matching are implemented on GPUs. All these features prompt the application of our proposed methodology to practical cases.

## 4. Experiments and discussion

### 4.1. Description of model settings

In the numerical experiment, a 3D benchmark model used in the SAIGUP project (Matthews et al., 2008) is used to test our proposed surrogate-assisted history matching approach. The model consists of 20 layers containing a total of 78720 active grid cells. The reservoir model describes a water-flooding system with nine producers and nine injectors, which are labeled from $P_1$ to $P_9$, and $I_1$ to $I_9$, see Fig. 3. Details about reservoir geometry, rock properties, fluid properties, and well controls are shown in Table 1.

In this case-study, the logarithmic permeability field is heterogeneous and assumed to be log-Gaussian random fields. We generate Gaussian-distributed realizations of logarithmic permeability using the Stanford Geostatistical Modeling Software (SGeMS) (Anon., 1995). One of the realizations is chosen to be the reference model as illustrated in Fig. 3. Separate random Gaussian distributions are used to model the top nine layers (zone 1) and the bottom 10 layers (zone 2), which are separated from each other by layer 10. Fig. 3(b) also shows the logarithmic permeability fields of the 1st, 10th and 11th horizontal layer. In this 3D model, the vertical permeability of the 10th layer is much smaller than that of other layers. This layer has ultra-low permeability, and therefore acts as a barrier to vertical flow.

Another strategy to decrease the number of model simulations is the reduction of parameter dimensionality, such as principle component analysis (PCA) (Chen et al., 2014), discrete cosine transformation (DCT) (Jafarpour and McLaughlin, 2008) and discrete wavelet transformation (DWT) (Chen and Oliver, 2012). In addition, several approaches, e.g., Pluri-PCA (Chen et al., 2015), optimization-based PCA (O-PCA) procedure (Vo and Durlofsky, 2015, 2016) and convolutional neural network based PCA (CNN-PCA) (Liu et al., 2018), were proposed to adapt the standard PCA to address non-Gaussian models. For the Gaussian model defined in this study, the conventional PCA reduction of the parameter space using a 95% energy cutoff criterion results in $N_{pca} = 303$ (e.g., 155 and 148 patterns for zone 1 and zone 2, respectively.) global PCA patterns. These are used to fully represent the original logarithmic permeability fields, and then generate the training and testing samples. Both samples are generated from the same distribution.

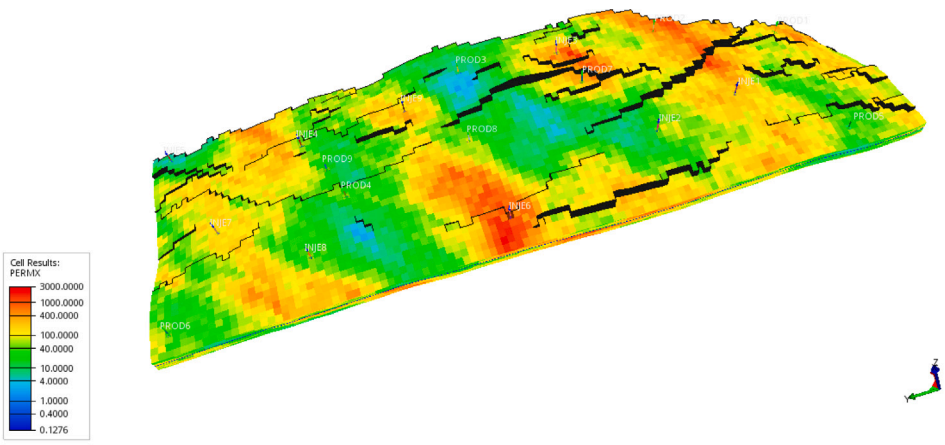
In order to quantify the history matching results, we define an error measure based on parameter misfits $e_{\mathbf{m}}$ as follows,

$$e_{\mathbf{m}} = \sqrt{\frac{\sum_{i=1}^{N_m}(\mathbf{m}^i_{true}-\mathbf{m}^i_{upt})^2}{N_m}} \quad (10)$$

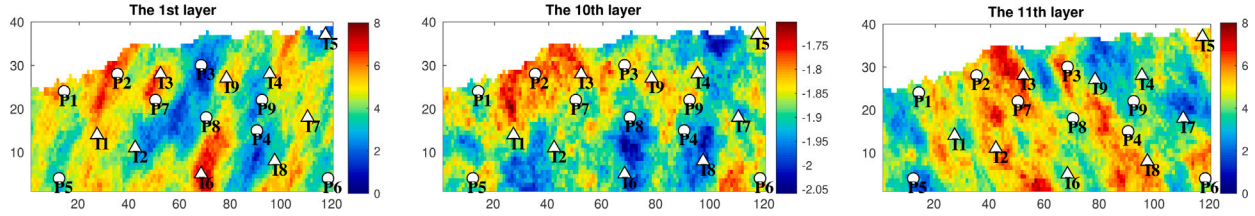
where, $\mathbf{m}^i_{true}$ and $\mathbf{m}^i_{upt}$ denote the logarithmic permeability value of the gridblock $i$ from the reference model and updated model, respectively.

Oliver discussed the expected range of the optimal objective function (Oliver et al., 2008). Since we employ an approximate surrogate model, we apply a less strict criterion (Liu et al., 2019). The objective function $J$ corresponding to the satisfactory history matching result should satisfy the inequality as follows

$$J \leqslant 3\,N_{obs} \quad (11)$$




(a) The 3D view of geological realization




(b) The 2D view of the 1st, 10th and 11th horizontal layer

**Fig. 3.** Illustrations of the spatial logarithmic permeability for the 1st, 10th and 11th horizontal layer. The upper nine layers and lower ten layers are isolated by a vertically ultra-low or almost impermeable layer, i.e, the 10th layer. The triangles and circles denote the injectors and producers, respectively.

**Table 1**
Reservoir settings using OPM for 3D benchmark SAIGUP model and the hyper parameter settings for training cR-U-Net model.

| Reservoir model settings | |
|---|---|
| Dimension | $40 \times 120 \times 20$ |
| Number of wells | 9 producers, 9 injectors |
| Constant porosity | 0.2 |
| Fluid density | 1014 kg/m$^3$, 859 kg/m$^3$ |
| Fluid viscosity | 0.4 mP·s, 2 mP·s |
| Bottom-hole pressure for producers | 15 MPa |
| Bottom-hole pressure for injectors | 30 MPa |
| Historical production time | 5400 days |
| **Hyper parameter settings** | |
| Number of realizations for training ($N_s$) | 100, 300, 500, 800, 1000 |
| Number of training samples ($N_s \times N_t$) | 1000, 3000, 5000, 8000, 10000 |
| Number of realizations for testing ($N_{test}$) | 200 |
| Learning rate | 0.001 |
| Optimizer | *Adam* |
| Batch size | 10 |
| Number of epochs | 100 |

where, $N_{obs}$ is the total number of measurements. We use the history matched permeability field, run the HFM and use the output saturation to calculate the objective function $J$. This criterion or tolerance can be used to quantify the accuracy of history matching results.

To evaluate the quality of the parameter estimation results, we will compare the value of the final objective function with both the criterion (Eq. (11)) and the reference objective function values calculated from the true model. Reconstructed parameter maps will provide a visual indication of quality of the solution. For all approaches we will list the computational cost expressed in terms of the number of HFM simulations.

### 4.2. Configuration of cR-U-Net architecture

The details about the cR-U-Net architecture are described in Table 2. The input is a three-dimensional image of logarithmic permeability $\mathbf{m}$. In our proposed workflow, since we use a PCA parameterization to generate the geological permeability fields, which are vectors, we need to reshape the geological permeability fields into three-dimensional images with the size of $N_x = 40$, $N_y = 120$ and $N_z = 20$ before feeding them into the cR-U-Net.

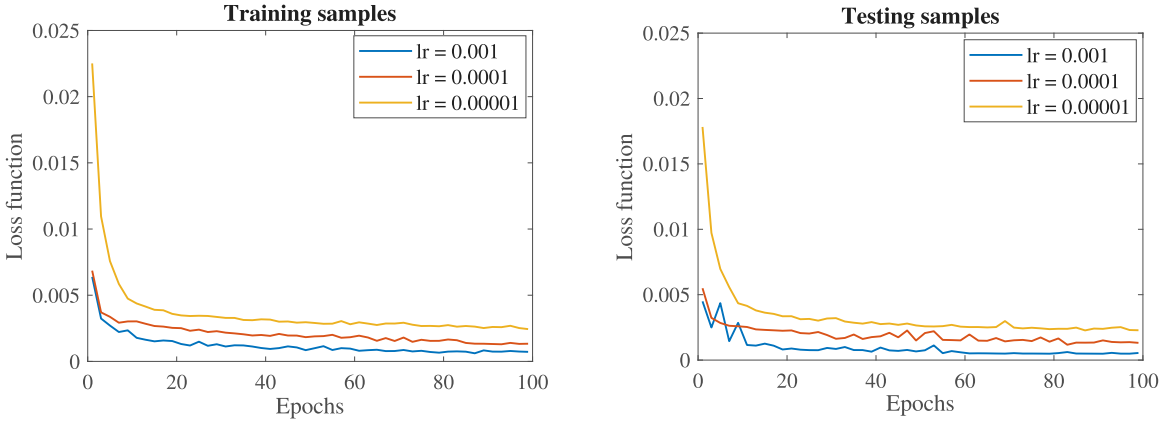
As illustrated in Fig. 4, the encoding part extracts a set of feature maps $\mathbf{F}_k(\mathbf{m})$ $(k = 1, 2, 3, 4)$ from four consecutive convolutional blocks. The final output of the encoding part consists of 128 compressed feature maps with size $2 \times 7 \times 1$, which are then concatenated with an additional time feature map of the same size $2 \times 7 \times 1$. The time should be normalized to the range [0, 1] to improve the training performance. There are in total $N_t = 10$ timesteps for this case-study, and the normalized times are therefore $[0.1, 0.2, 0.3, \ldots, 0.9, 1.0]$. For one specific timestep, all elements of the $2 \times 7 \times 1$ time feature map are filled by the normalized time, e.g., 0.1 corresponding to the 1st timestep. These 129 feature maps are fed into the transition unit for producing 128 feature maps with constant size $2 \times 7 \times 1$. Finally, these 128 feature maps are provided to the decoding unit for the output, e.g., saturation, with size of $40 \times 120 \times 20$.

Five different number of realizations for training, i.e., $N_s = 100$, 300, 500, 800, and 1000 samples, and totally different $N_{test} = 200$ testing sample are used to assess the performance of cR-U-Net. The model is run for 5400 days, and training data are collected at $N_t = 10$ intervals of 540 days each. After reorganizing the dataset, there are in total 1000, 3000, 5000, 8000 and 10000 training samples correspondingly.

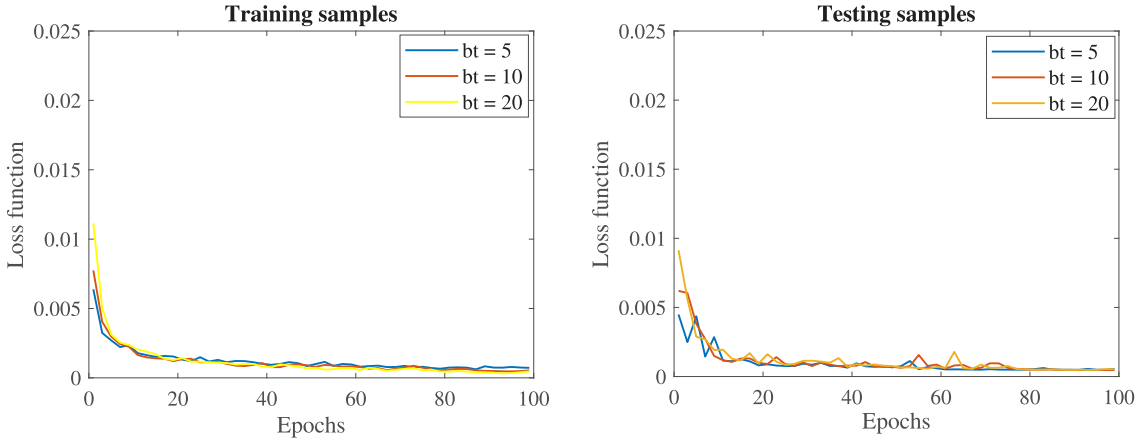
Some results related to the selection of hyperparameters, e.g., learning rate and batch size, are shown in Fig. 5. When minimizing the loss function, the learning rate determines the updating step of parameters. In order to choose a reasonable learning rate, we investigate a wide range of learning rates from 0.001 to 0.00001. Fig. 5(a) shows the evolution of losses over the training and testing samples. In this case

**Table 2**
Illustration of cR-U-Net architecture. The size of a 3D image is $N_x = 40$, $N_y = 120$ and $N_z = 20$ in this case-study.

| Unit | Layer | Output size |
|---|---|---|
| Encoder unit | Input (geological permeability images) | (40, 120, 20, 1) |
| | *Conv3D-BatchNorm-ReLU*, 16 kernels of size (3, 3, 3, 2) | (20, 60, 10, 16) |
| | *Conv3D-BatchNorm-ReLU*, 32 kernels of size (3, 3, 3, 16) | (20, 60, 10, 32) |
| | *Conv3D-BatchNorm-ReLU*, 64 kernels of size (3, 3, 3, 32) | (10, 30, 5, 64) |
| | *Conv3D-BatchNorm-ReLU*, 128 kernels of size (3, 3, 3, 64) | (10, 30, 5, 128) |
| Transition unit | Input (outputs of encoder unit + an additional time feature) | (10, 30, 5, 129) |
| | *resConv*, 128 kernels | (10, 30, 5, 128) |
| | *resConv*, 128 kernels | (10, 30, 5, 128) |
| | *resConv*, 128 kernels | (10, 30, 5, 128) |
| | *resConv*, 128 kernels | (10, 30, 5, 128) |
| | *resConv*, 128 kernels | (10, 30, 5, 128) |
| Decoder unit | Input (outputs of transition unit) | (10, 30, 5, 128) |
| | *TConv3D-BatchNorm-ReLU*, 128 kernels of size (3, 3, 3, 128), | (10, 30, 5, 128) |
| | *TConv3D-BatchNorm-ReLU*, 64 kernels of size (3, 3, 3, 128), | (20, 60, 10, 64) |
| | *TConv3D-BatchNorm-ReLU*, 32 kernels of size (3, 3, 3, 64), | (20, 60, 10, 32) |
| | *TConv3D-BatchNorm-ReLU*, 16 kernels of size (3, 3, 3, 32), | (40, 120, 20, 16) |
| | *Conv3D*, 1 kernels of size (3, 3, 3, 16), stride 1 | (40, 120, 20, 1) |




(a) learning rate (lr) selection




(b) batch size (bs) selection

**Fig. 4.** Schematic illustration of the cR-U-Net architecture using the 3D SAIGUP geological models as inputs.

study, using a smaller learning rate will result in a slower convergence. Here we choose 0.001 as the best learning rate for this problem. The batch size determines the number of training examples used in computing a gradient in each iteration step. Fig. 5(b) displays the evolution of losses with respect to the batch sizes of 5, 10, and 20, respectively. It can be clearly seen that the batch size hardly affects the training performance. As a result, we have chosen 10 as the batch size to avoid possible memory issue. The other hyperparameters settings for training the cR-U-Net model are listed in Table 2.

### 4.3. Assessment of surrogate model quality

It is useful to analyze the relative error of predicted grid-based saturation between HFM and the cR-U-Net surrogate model. The field-average relative error in saturation over all $N_t$ time intervals, denoted

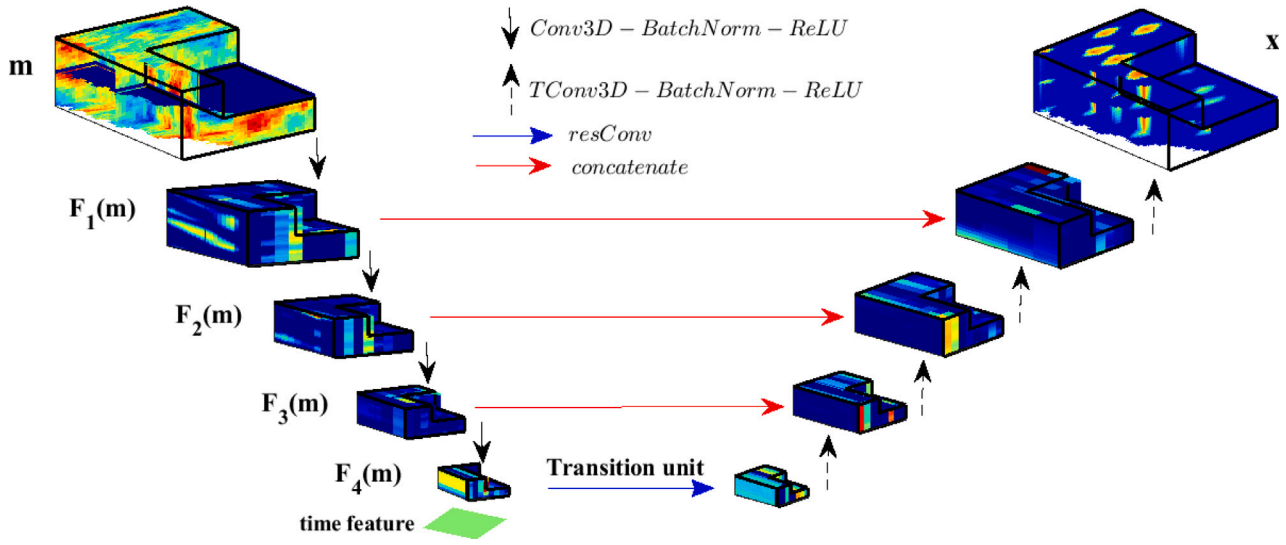



**Fig. 5.** Evolution of loss functions with respect to the number of epochs. (a) learning rate (lr) selection; (b) batch size (bs) selection. These models are trained with 100 training samples.

as $\gamma_s^n$, for all $N_{test}$ samples, is given by

$$\gamma_s = \frac{1}{N_{test} N_{\mathbf{m}} N_t} \sum_{i=1}^{N_{test}} \sum_{n=1}^{N_t} \sum_{j=1}^{N_{\mathbf{m}}} \frac{\|\hat{\mathbf{x}}^{i,n,j} - \mathbf{x}^{i,n,j}\|}{\mathbf{x}^{i,n,j}} \tag{12}$$

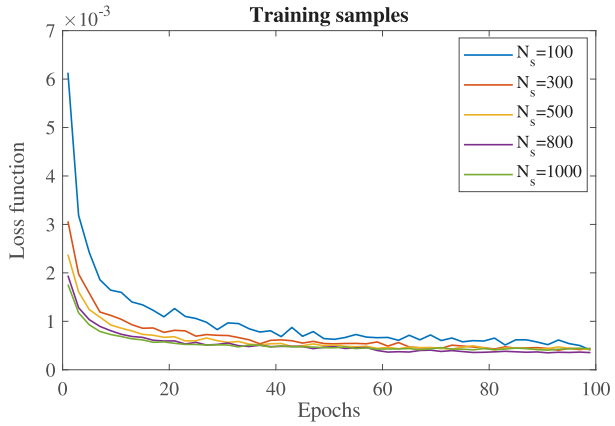
where, $\hat{\mathbf{x}}^{i,n,j}$ and $\mathbf{x}^{i,n,j}$ separately denote the saturation predicted from HFM and surrogate models for the testing sample $i$, in gridblock $j$ at the timestep $n$.

In order to analyze the overfitting issue during the training of the DNN model, we show the evolution of loss functions with respect to the epochs, see Fig. 6. It can be observed that the overfitting problem hardly occurs even when using only 100 training samples. The accuracy of the DNN surrogate model improves with the number of training samples.

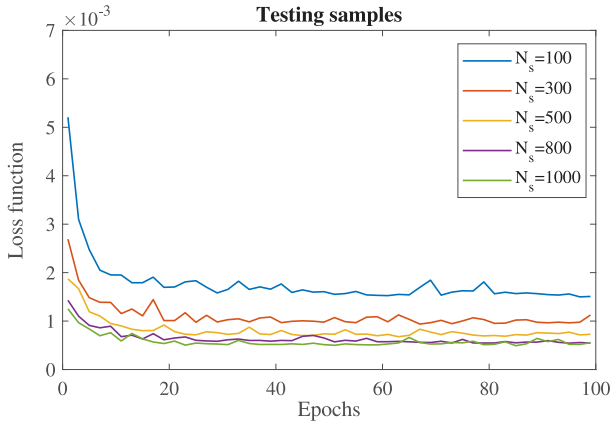
The performance of the cR-U-Net model can be investigated by comparing the predefined metrics, e.g., $\gamma_s$, relative to the number of training samples $N_s$ for the quantities of interest, i.e., saturation values. Fig. 7 shows the $\gamma_s$ value obtained on the training ensemble of $N_{test}$ = 200 realizations for the predicted saturation. The $\gamma_s$ metrics will gradually increase as the time propagation, which reflects the degraded quality of network models. It has been demonstrated that training cR-U-Net model using $N_s$ = 1000 samples significantly improve the network quality through achieving relatively low $\gamma_s$ values. Through observing the evaluated relative errors in saturation maps over the 200 random test samples, we find the overall field-average relative errors $\gamma_s$ are 2.83% and 5.37% for 1000 and 100 training samples, respectively. The small $\gamma_s$ values significantly indicate a high degree of accuracy in the saturation maps predicted from the cR-U-Net model.

In addition, the plots of the time-varying saturation predictions from the two approaches are used to visually assess the accuracy of the trained network. Fig. 8 shows the predicted saturation distribution at two time-instances, i.e., day 2700 and day 5400, corresponding to the cR-U-Net models trained using $N_s$ = 100 and 1000 samples, respectively. It clearly can be seen that the cR-U-Net surrogate model is capable of predicting the water saturation profile with a high accuracy. In addition, the corresponding relative saturation error $\gamma_s$ at these two time instances are 3.92% and 2.87%, respectively. Thus, the results in Fig. 8 can be considered to be representative in terms of surrogate model accuracy.

In terms of computational effort, the runtime for a single HFM simulation for this case is about 250 s on a machine with i5-4690 Intel CPUs (4 cores, 3.5 GHz) and 24 GB memory. The cR-U-Net is trained using an NVIDIA Tesla P100 GPU card. The simulation of a trained neural network requires about 0.1 s. However, the training stage of this network is computationally intensive, which includes the generation of training samples and additional overhead. It would be not useful




(a) Loss function values for different sizes of training set




(b) Loss function values on test set

**Fig. 6.** Evolution of loss functions with respect to the number of epochs. (a) for different sizes of training set; (b) Loss function values on test set.

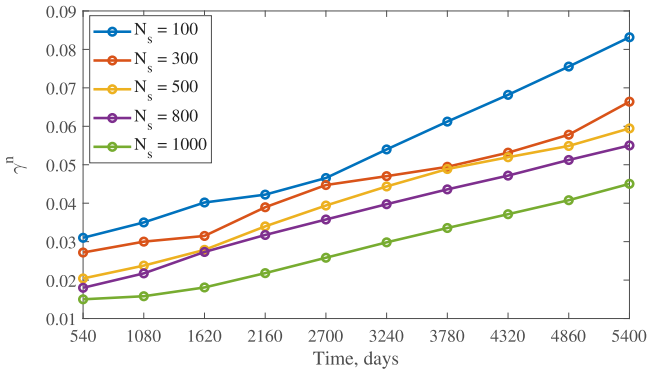



**Fig. 7.** Comparison of the field-average relative error in saturation at all $N_t$ = 10 timesteps over the full $N_{test}$ = 200 test samples. The cR-U-Net surrogate models are trained using $N_s$ = 100, 300, 500, 800, and 1000 samples, respectively.

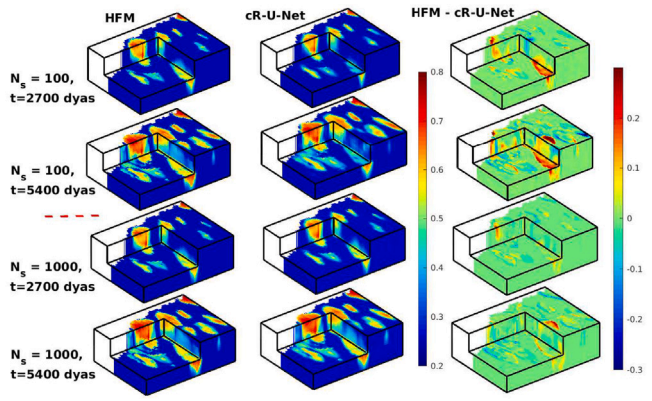



**Fig. 8.** Illustration of time-varying phase saturation predicted with cR-U-Net and HFM, and the absolute errors between the two models at day 2700 and day 5400. The cR-U-Net models are trained using $N_s$ = 1000 and 100 training samples, respectively.

to construct the cR-U-Net surrogate model unless it is to be used in the situation where numerous simulations are required. Because many simulations are required in large-scale history matching applications, the cR-U-Net surrogate models is applicable in this context. Although the training time can vary by case, it is just a small fraction of the time required with conventional history matching where numerous HFM simulations must be performed. The use of surrogate model in conjunction with the gradient-based history matching is presented in the following section.

### 4.4. History matching results using the cR-U-Net surrogate

The data used (saturation measured on all active grids at different times) mimics observations obtained from interpretation of a 4D seismic survey. The saturation values are simulated from the 'true' model simulation for every 540 days, mimicking the collection of a large amount of data, i.e., a total of 787200 measurements, from ten time instances. Normal distributed independent measurement noise with a standard deviation equal to 5% of the 'true' data value was added to all measurements. We will demonstrate the feasible applicability of our proposed DNN-assisted history matching approach to assimilate a large number of measurements.

#### 4.4.1. Study of the base-case

After training the cR-U-Net surrogate model, the gradients of the objective function with respect to the logarithmic permeability are computed analytically using AD, and then the *Adam* optimizer is implemented to update the geological parameters efficiently. Batch size

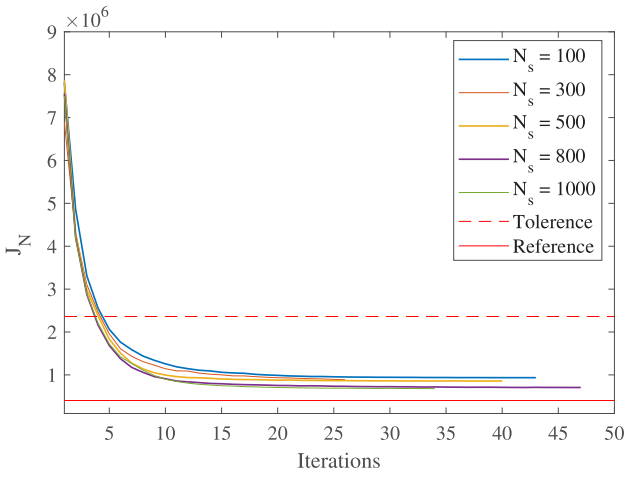



**Fig. 9.** The evolution of surrogate objective functions $J_N$ as a function of iterations. The cR-U-Net surrogates are trained using $N_s$ = 100, 300, 500, 800 and 1000 training samples, respectively. The tolerance (defined by Eq. (11)) and the reference objective function values (reflecting the impact of the data noise) are shown in red dash and solid lines, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

of one (i.e. we randomly select data from one measured timestep) is used in this base-case study. In addition to the default parameters for the *Adam* optimizer, the learning rate $\epsilon$ is set to be 0.05. We do not fix the maximum number of iterations since the cost of running the cR-U-Net models is almost negligible. The minimization algorithm is considered to have sufficiently converged when either the criterion for minimum change of objective function or the logarithmic permeability at two consecutive iterations is satisfied. These two stopping criteria are set to be 0.0001 and 0.001, respectively.

Table 3 and Fig. 9 display the history matching results, including the final objective functions, parameter misfits $e_{\mathbf{m}}$ and the required number of HFM simulations. The cR-U-Net models are trained using $N_s$ = 100, 300, 500, 800 and 1000 samples, respectively. The surrogate objective functions $J_N$ are iteratively minimized as shown in Fig. 9. Since the cR-U-Net cannot exactly represent the original HFM using a finite training set, we also calculate the high-fidelity objective functions $J$ using the original HFM. It can be seen that the values of $J$ are slightly larger than that of $J_N$ due to the fact that the cR-U-Net surrogate model introduces additional approximation errors. In order to verify the history matching results, we also display the tolerance (defined by Eq. (11)) and the reference objective function values, which are indicated by the red dash and solid lines in Fig. 9, respectively. It can be observed that the final objective functions are very close to the reference value. After assimilating a large amount of measurements, the parameter misfits $e_{\mathbf{m}}$ have been significantly decreased, e.g., from 1.1141 to 0.4123 corresponding for the case with $N_s$ = 1000 training samples. The accuracy of history matching results gradually improves with increasing number of training samples as indicated by the reductions of $e_{\mathbf{m}}$ values.

Fig. 10 shows the 2D horizontal cross sections of the updated logarithmic permeability fields for the 1st and 11th vertical layer, which can be used to assess the accuracy of the history matching results. The reference model is almost reconstructed by assimilating a large number of measurements. Surprisingly, the cR-U-Net model trained with only $N_s$ = 100 samples is already capable of achieving a posterior realization which is visually close to the reference model. These results demonstrate the effectiveness of the proposed cR-U-Net surrogate model for addressing the history matching problems, since it can provide a useful direction to update the uncertain parameters correctly.

To further illustrate the accuracy of the updated reservoir models, the predictions of saturation at both two specific time instances

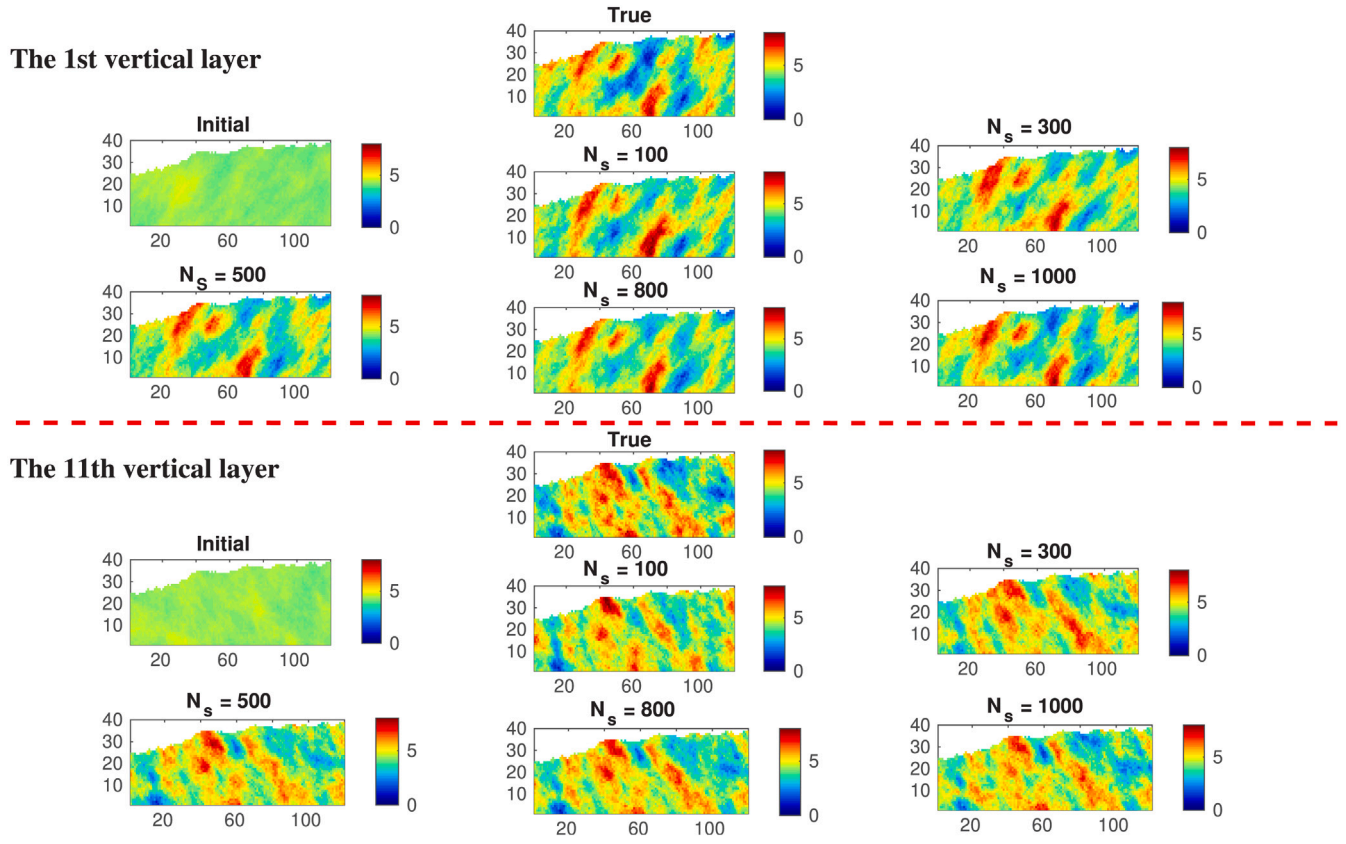



**Fig. 10.** 2D horizontal cross sections of the updated logarithmic permeability under different sample sizes $N_s$ = 100, 300, 500, 800 and 1000 from the cR-U-Net. The 1st and 11th vertical layer are shown.

**Table 3**
Summary of the history matching results using cR-U-Net surrogate model trained with different numbers of samples. These are the results for SGD with batch size one. The tolerance (Eq. (11)) and the reference objective function values for the true model are also listed.

| $N_s$ | $e_{\mathrm{m}}$ | $J_N \times 10^6$ | $J \times 10^6$ | Number of HFM simulations |
|---|---|---|---|---|
| 100 | 0.7130 | 0.9262 | 1.0224 | 100 |
| 300 | 0.6075 | 0.8425 | 0.8754 | 300 |
| 500 | 0.5087 | 0.8376 | 0.8577 | 500 |
| 800 | 0.4368 | 0.7083 | 0.7235 | 800 |
| 1000 | 0.4123 | 0.6720 | 0.7011 | 1000 |
| Reference | | | 0.4027 | |
| Tolerance | | | 2.362 | |

(e.g., day 2700 and day 5400) before and after history matching are depicted in Fig. 11. Compared to the initial models, the saturation predictions of the updated models are visually close to the predictions of the true model. The correlation coefficients $R^2$ have been increased from the initial 70% to the final 98%.

The scalability and generalization of the whole workflow has been demonstrated by estimating a totally different reference model. Fig. 12 shows the 2D horizontal cross sections of the updated logarithmic permeability fields, which further confirms the effectiveness of the proposed workflow for addressing large-scale history matching problem.

*4.4.2. Effect of the initial models*

The gradient-based optimization methods inevitably get stuck in local minima for any non-convex optimization problems, such as the history matching in this study. Fortunately, without running additional HFM simulations at the history matching stage, the proposed cR-U-Net surrogate model enables us to efficiently generate multiple posterior solutions through starting from different initial models. Fig. 13 depicts

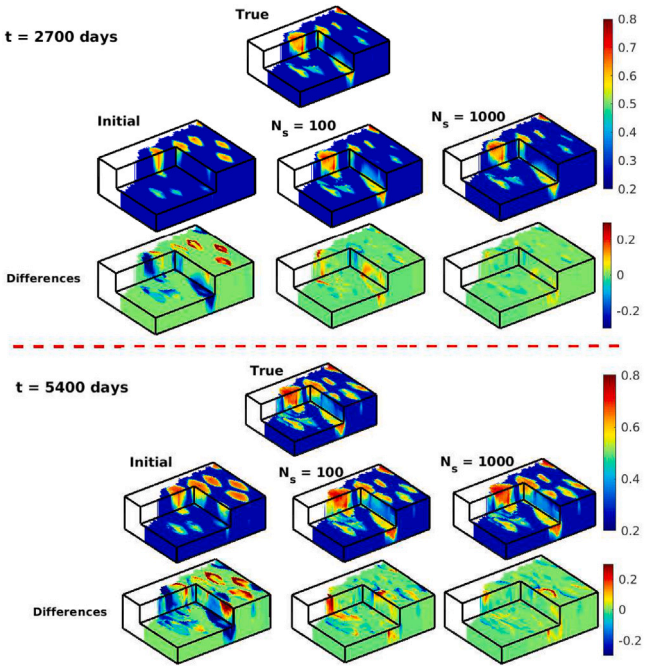



**Fig. 11.** Predictions of water saturation before and after history matching and their absolute errors at 2700 days and 5400 days of production. The cR-U-Net models are trained using $N_s$ = 100 and 1000 training samples, respectively.

the parameter misfits $e_{\mathrm{m}}$ after the history matching corresponding to three different initial models. The differences among these three

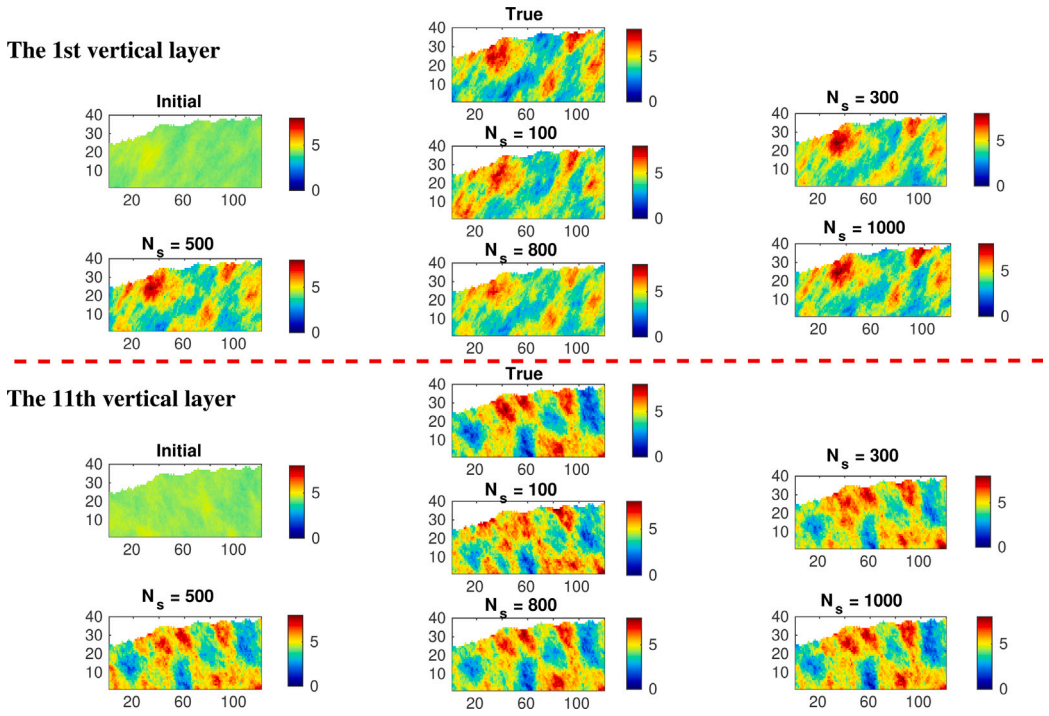



**Fig. 12.** 2D horizontal cross sections of the updated logarithmic permeability for a totally different reference model. The 1st and 11th vertical layers are shown here.

parameter misfits $e_{\mathbf{m}}$ indicate that several local minimas have been found for different initial models. It can be further visually revealed in Fig. 14 that the updated logarithmic permeability fields corresponding to these three initial models are significantly different.

During the history matching process, overfitting occurs when a model perfectly matches the data but makes poor predictions. In this section, both parameter misfits and predictions of well response are used to diagnose this issue. Fig. 15 displays a boxplot of the parameter misfits $e_{\mathbf{m}}$ of logarithmic permeability corresponding to 100 different initial models. Comparing with the initial models, the parameter misfits are gradually reduced as the number of training samples increases. We also compare in Fig. 16 the predictions of the well water injection rate and well water-cut at all nine injectors and nine producers before and after history matching. Clearly, it can be observed that the spread of these predictions is significantly reduced toward the predictions of the reference model. All these results reveal that overfitting almost does not occur during the history matching process.

In this study, the total number of HFM simulations is taken as an indicator of the computational cost, since the GPU time for running cR-U-Net is negligible compared to that for running HFM model. In order to assess the model uncertainty, we can generate multiple posterior models through repeatedly implementing the gradient-based optimization. We only need to run HFM simulations in the training stage and SHM process does not involve additional HFM simulations. Our proposed surrogate-assisted history matching method is highly efficient since the training of the cR-U-Net requires only a small number of HFM simulations. For example, the cR-U-Net model trained with only $N_s = 100$ samples is already capable of achieving good history matching results.

#### 4.4.3. FGD versus SGD optimizers

Since our proposed cR-U-Net model is not an exact representation of the high-fidelity model, incorporating the cR-U-Net into the SHM procedure inevitably introduces an additional source of uncertainty due to the approximation errors. In this section, the *Adam* optimizer is used to demonstrate the effectiveness of SGD optimization in reducing the approximation errors. The entire dataset measured at 10 time

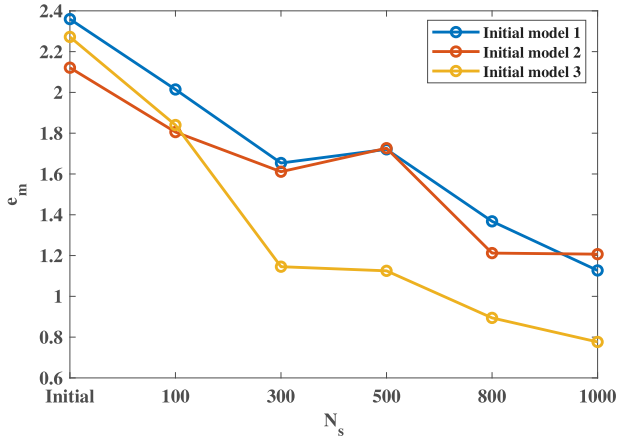



**Fig. 13.** Illustration of the parameter misfits $e_{\mathbf{m}}$ with respective to the number of training samples $N_s$ corresponding to three random initial models.

instances is divided using five different batch size, e.g., 1, 2, 3, 4 and 5, respectively. Taking the batch size 2 as an example, two random time index are selected from the 10 time steps, and then the corresponding data at these two time instances are used to update the parameters in one iteration. That is to say, the entire data will be fully used to update the parameters in 5 iterations.

Fig. 17 shows the evolution of the surrogate objective function $J_N$ with respect to the iterations and batch size for the *Adam* optimizer. It clearly can be seen that using a smaller batch size can achieve a faster convergence rate. For example, about 35 and 118 iterations are required for a batch size 1 and 5, respectively. By contrast, the FGD optimizer requires much more iterations, e.g., after 220 iterations it still has a larger objective function value.

Fig. 18 displays an ensemble of parameter misfits $e_{\mathbf{m}}$ corresponding to different batch sizes and the FGD optimize. It is noticeable that the SGD optimization achieves larger reductions of the parameter misfits than that of the FGD optimization. SGD is more effective to address

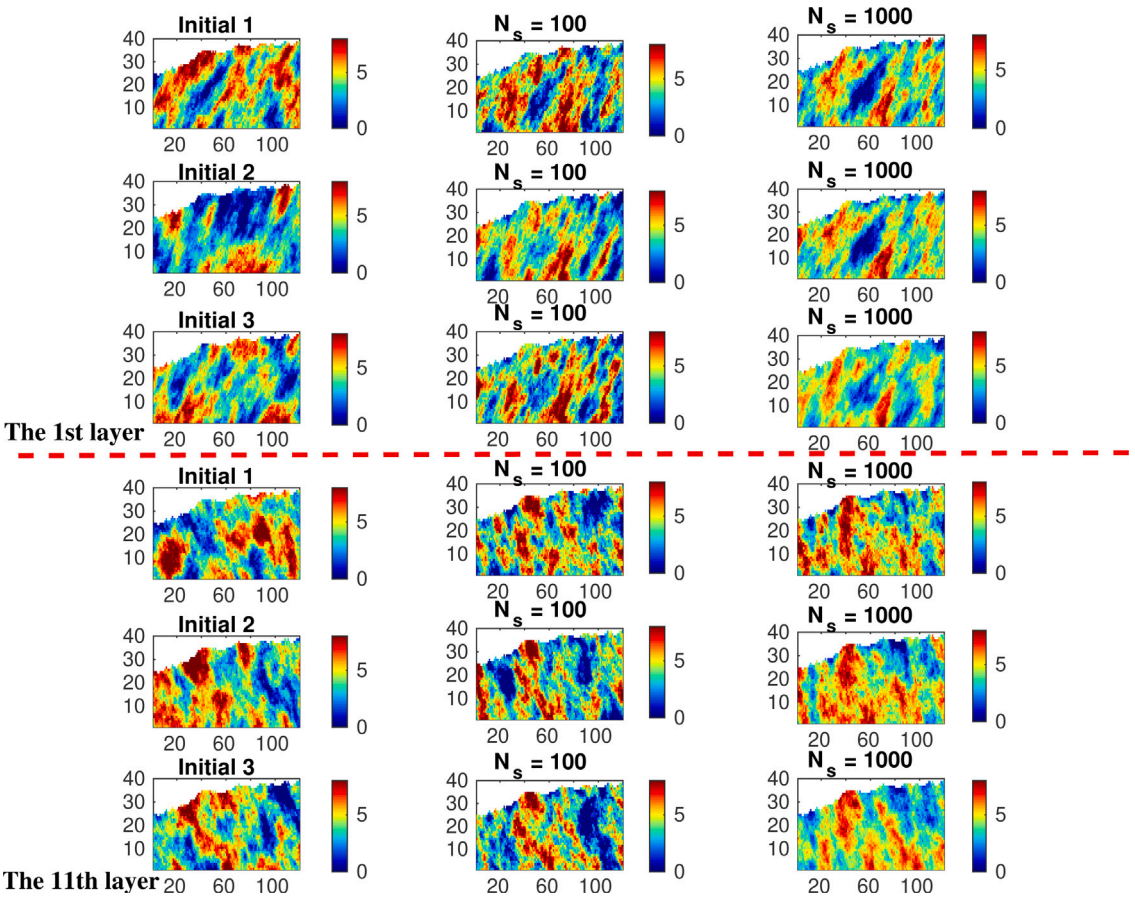



**Fig. 14.** 2D horizontal cross sections of the updated logarithmic permeability corresponding to three random initial models. The cR-U-Net models are trained using $N_s$ = 100 and 1000 samples, respectively.

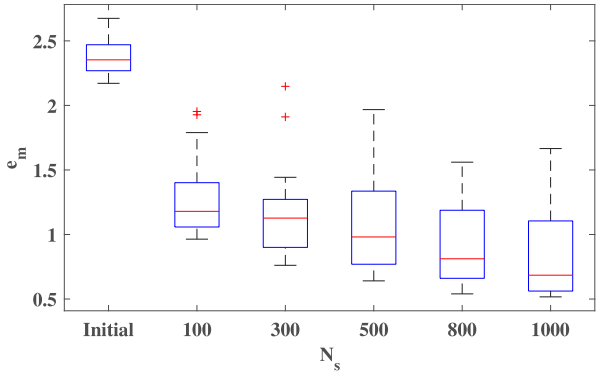



**Fig. 15.** Boxplot of the parameter misfits $e_{\mathbf{m}}$ of logarithmic permeability for a different reference model corresponding to 100 different initial models.

surrogate-assisted history matching with approximation errors. As a hyper parameter for the *Adam* optimizer, we should chose the batch size with much care. The evolution of the parameter misfits $e_{\mathbf{m}}$ corresponding to three initial models are highlighted as well in Fig. 23. It can be concluded that the optimal batch size depends on the initial model and the quality of the cR-U-Net model. For example, for the cR-U-Net trained with $N_s$ = 1000 samples, batch size 3, 2 and 5 generate the best history matching results for these three initial models, respectively.

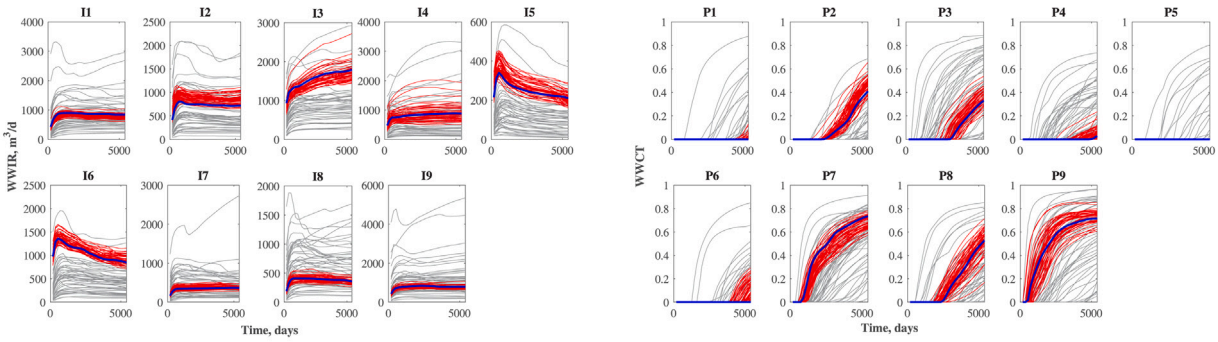
Fig. 19 depicts the updated logarithmic permeability fields corresponding to one random initial model using SGD and FGD optimization. The SGD optimization obtains similar parameter fields corresponding to different batch sizes, which are very consistent with the reference model, while the FGD optimizer generates a different parameter field. SGD has proven highly successful in the process of training DNN. There is a clear similarity in the way how DNN and SHM iteratively update parameters, e.g., geological parameters in the history matching. The stochastic optimization methods might easily move away from saddle points, while the FGD methods tend to get stuck in them. That is to say, our adopted stochastic optimization method might be more possible to escape from local minima. By contrast, the approximated gradients by FGD are inevitably fraught with approximation errors of cR-U-Net surrogate model.

### 4.5. cR-U-Net versus Subdomain POD-TPWL surrogate

Since both the DNN surrogate and our previously proposed projection-based reduced-order modeling, e.g., subdomain POD-TPWL, are intended for the gradient-based history matching, a comparative study between them is conducted in this section. More details about the derivation of subdomain POD-TPWL and model-reduced adjoint approach for gradient approximation can be found in the provided supplementary material. The saturation data are measured corresponding to the saturation values from the 'true' model simulation after 2700 days and 5400 days of production, which results in total 157440 measurements. The noisy measurements for these two time instances are shown in Fig. 20.

#### 4.5.1. Construction of the subdomain POD-TPWL

Our proposed subdomain POD-TPWL is previously tested on synthetic or simplified version of 2D models (Xiao et al., 2019a,b), and an extension of subdomain POD-TPWL to 3D cases is straightforward. It can be observed from Fig. 20 that the water-flooded area in the




(a) Well water injection rate (b) Well watercut

**Fig. 16.** Forecast of the well water injection rate and well water-cut of all nine injectors and nine producers. Dark-gray lines: initial models, blue lines: the reference model, red lines: updated models using the cR-U-Net trained with $N_s = 1000$ samples. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

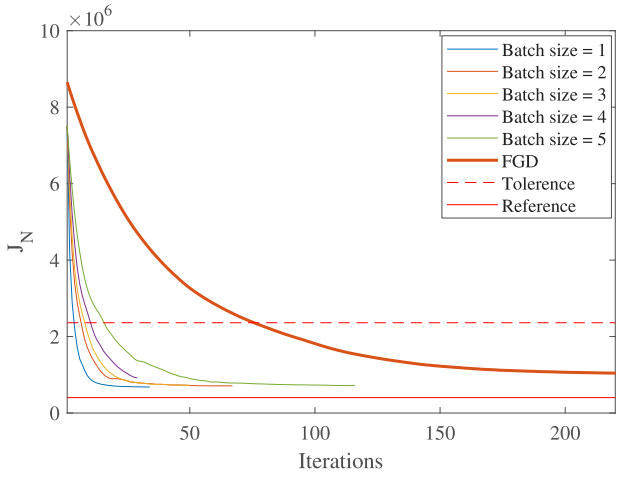



**Fig. 17.** Evolution of surrogate objective function $J_N$ as a function of the number iterations and batch size. The cR-U-Net surrogate models are trained with 1000 samples and the *Adam* optimizer is applied.

upper layers is smaller than that of lower layers, that is, the saturation and geological parameters have shorter-distance correlations in the upper layers than that of the lower layers. It should be reasonable that the zone 1 is decomposed into small subdomains, while the zone 2 is decomposed into relatively large subdomains. We use a quasi-3D decomposition consisting of two independent 2D decompositions for zone 1 and zone 2, respectively. In this case-study, we choose a fixed domain decomposition scheme, e.g., $(3 \times 4, 2 \times 3)$ decomposition. That is, $3 \times 4$ and $2 \times 3$ 2D decompositions are for zone 1 and zone 2, respectively.

We follow the procedure described in Xiao et al. (2019a,b) to determine the number of HFM simulations for the collection of pressure and saturation snapshots. 42 HFM simulations are run for collecting a total number of 7560 (42 simulation models by 180 time steps) snapshots of both pressure and saturation. Instead of taking global basis functions to define the subspace, the snapshots are first partitioned into each subdomain and then local basis functions are obtained from these partitioned snapshots. That is, these 7560 global saturation and pressure snapshots are decomposed into each subdomains. As a result, the implementation of POD for a large number of snapshots will not pose severe computational problems in this 3D model application.

In terms of computational effort, the subdomain POD-TPWL models require less than 1.5 s. However, the construction of the subdomain POD-TPWL requires 147 HFM simulation runs, among them, 42 HFM simulations are used to collect the snapshots for implementing POD, and 105 HFM simulations are used to construct the RBF function (Eq. (9) in the provided supplementary material.).

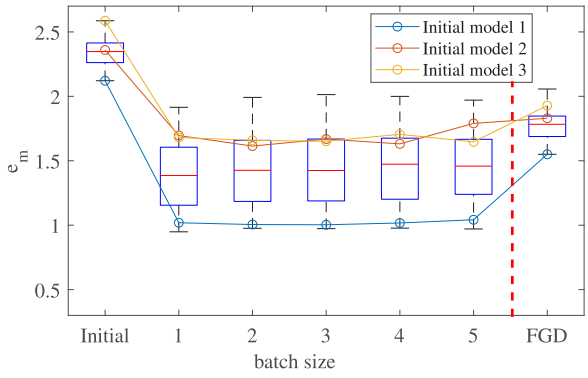
#### 4.5.2. Comparison of surrogate model quality

Fig. 21 displays the dependence of the field-average relative error $\gamma_s$ with respect to domain decomposition strategy, testing interval and number of training samples. The testing interval represents the maximum discrepancy between the testing model and the reduced-order linear model. It can be seen from Fig. 21(a) that increasing the testing interval deteriorates the accuracy of the subdomain POD-TPWL surrogate model, e.g., from 1.52% to 5.04%, for the $(3 \times 4, 2 \times 3)$ domain decomposition scheme. The comparisons of the time-varying saturation between subdomain POD-TPWL, HFM and their difference are also used to visually assess the accuracy of the subdomain POD-TPWL surrogate model. Fig. 22(a)–(b) show the predictions of spatial saturation distribution at the day 2700 and day 5400 corresponding to the testing interval $[-0.1, 0.1]$ and $[-0.3, 0.3]$, respectively. It clearly can be seen that the cR-U-Net surrogate model achieves more accurate saturation profile than that of subdomain POD-TPWL for the testing model sampled from a relatively large testing interval $[-0.3, 0.3]$ in this case-study.

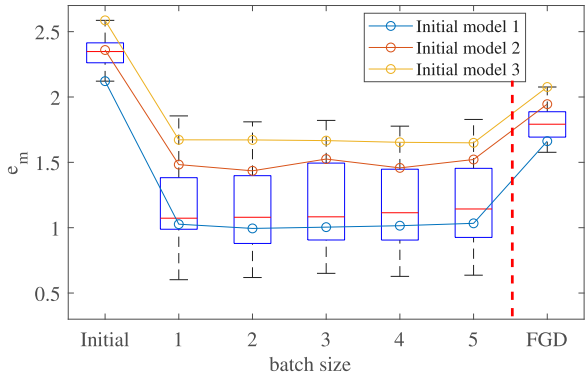
The cR-U-Net is almost not sensitive to the testing interval, which however has a significant influence on the subdomain POD-TPWL surrogate model. For the testing model sampled from a small testing interval $[-0.1, 0.1]$, these two surrogate models almost obtain comparable accuracy, however, the accuracy of the subdomain POD-TPWL decreases gradually as the testing interval increases. More deep insights about these two types of surrogate models are worth explaining. Subdomain POD-TPWL constructs reduced-order linear model around a specific trajectory using the first-order Taylor expansion. If the testing model is far away from this trajectory, the accuracy inevitably will deteriorate. This leads to the necessity in frequently reconstructing subdomain POD-TPWL once the new testing models have large discrepancies from the current reduced-order linear model. By contrast, the cR-U-Net actually constructs a global surrogate model based on the entire training data and hence retraining cR-U-Net surrogate model is not strictly required. Definitely, filling the entire parameter space as much as possible using a large number of training samples, e.g., 1000 in this study, substantially improves the accuracy.

#### 4.5.3. Comparison of history matching results

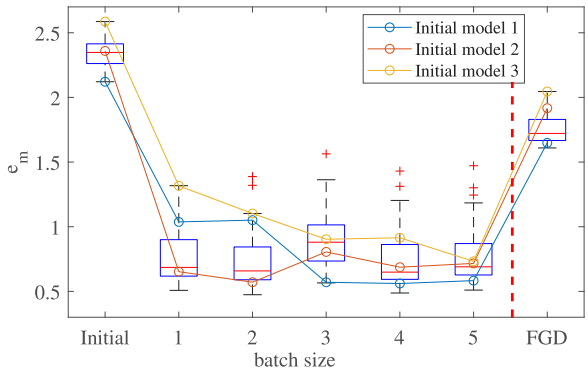
Table 4 display the final objective functions, the number of HFM simulations and the parameter misfits $e_{\mathbf{m}}$ for the cR-U-Net and subdomain POD-TPWL, respectively. In this base-case study, the subdomain POD-TPWL with $(3\times4, 2\times3)$ 3D domain decomposition has been almost convergence after 12 out-loop iterations. Although the subdomain POD-TPWL obtains comparable and even smaller final objective functions than that of cR-U-Net surrogate models, the parameter misfits $e_{\mathbf{m}}$




(a) $N_s$=100




(b) $N_s$=500




(c) $N_s$=1000

**Fig. 18.** Boxplot of the parameter misfits $e_{\mathbf{m}}$ of logarithmic permeability before and after history matching corresponding to 100 different initial models. The batch size of the *Adam* optimizer is set to 1, 2, 3, 4 and 5, respectively.

are relatively larger. For example, the final objective functions are $0.1689 \times 10^5$ and $0.3890 \times 10^5$ for the subdomain POD-TPWL and cR-U-Net trained with 300 samples, respectively, the parameter misfits $e_{\mathbf{m}}$, however, are 0.6534 and 0.6370 correspondingly. This result indicates that subdomain POD-TPWL is noticeably susceptible to the data noise and hence cannot prevent the model from overfitting to the noisy measurements. By contrast, the proposed cR-U-Net with SGD optimizer yields small parameter misfits $e_{\mathbf{m}}$, which also gradually decrease as the number of training samples.

In terms of the computational cost, the $(3\times4, 2 \times 3)$ domain decomposition requires 158 HFM simulations, among them, 42 HFM simulations are used to collect the snapshots to construct the bases for implementing POD, 105 HFM simulations are run to construct the initial subdomain reduced-order linear model at the 1st out-loop, and additional 11 HFM simulations are required in the following 11 outer-loops. In order to achieve comparable parameter misfits $e_{\mathbf{m}}$, the cR-U-Net surrogate model should be trained with at least $N_s = 300$ samples. We also present an alternative to compare these two surrogate models for almost the same computational cost. We increase the sampling points for constructing the subdomain POD-TPWL surrogate models, which generally improves the accuracy of the approximated gradients. As illustrated in Fig. 23, the parameter misfits $e_{\mathbf{m}}$ gradually decreases but much slower than those for the cR-U-Net. Corresponding to the specific domain decomposition scheme, such as $(3 \times 4, 2 \times 3)$ in this case-study, the accuracy does not infinitely improve as the number of training samples. These results further indicate the importance of designing appropriate domain decompositions.

Fig. 24 shows the 2D horizontal cross sections of updated logarithmic permeability fields for a comparison of the history matching results using the cR-U-Net and subdomain POD-TPWL surrogate models. It clearly can be seen that the true logarithmic permeability fields almost can be reconstructed. In summary, subdomain POD-TPWL is slightly more efficient than that of cR-U-Net surrogate model for generating one posterior realization for this example application. In addition, the implementation of subdomain POD-TPWL is also relatively flexible without a strict requirement for the hardware, e.g., GPU cards.

In order to generate $N_e$ posterior realizations for quantifying the model uncertainty, the subdomain POD-TPWL has to be independently implemented for each ensemble member, which requires us to repeatedly update the reduced-order models with several additional HFM simulations at the outer-iterations. 147 HFM simulations are required to construct the initial reduced-order model at the 1st out-loop. If the number of outer-loops for the $i$th ensemble member is assumed to be $N^i_{outer-loop}$, the required total number of HFM simulations for obtaining $N_e$ posterior realizations should be $(147 + \sum_{i=1}^{N_e} \times N^i_{outer-loop})$. By contrast, the cR-U-Net surrogate model obtains $N_e$ posterior realizations through starting from different initial models. As we have mentioned before, this process does not involves additional HFM simulations. Fig. 25 depicts the parameter misfits $e_{\mathbf{m}}$ of $N_e = 100$ posterior models for the cR-U-Net and subdomain POD-TPWL. It is noticeable that the cR-U-Net generates smaller parameter misfits $e_{\mathbf{m}}$ than that of the subdomain POD-TPWL. In terms of computational cost, the subdomain POD-TPWL requires about, for example an average $N_{outer-loop} = 15$ outer-loops, $1647 = 147 + 100 \times 15$ HFM simulations for generating these 100 posterior models, while the cR-U-Net only requires 100 HFM simulations for achieving even better results. The computational saving will increase linearly with the ensemble size. Overall, the cR-U-Net will be much more efficient than that of the subdomain POD-TPWL for generating an ensemble of posterior realizations.

Both methods have their own advantages and disadvantages. On the one hand, the physical interpretation of the subdomain POD-TPWL makes it easier to diagnose and comprehend. Our previous results in Xiao et al. (2019b) have indicated that the performance of subdomain POD-TPWL is very sensitive to the domain decomposition schemes. How to choose the optimal domain decomposition scheme is one of the most key steps for a successful implementation of the subdomain POD-TPWL. On the other hand, although the construction of the cR-U-Net can be realized without being explicitly programmed with the hidden and highly complex governing physics, its performance generally becomes difficult to explain. Also the choice of a DNN architecture is relatively subjective (although flexible), and its outputs lack a clear understanding. The cR-U-Net takes advantage of the high-performance computing units, such as the GPU cards, since the exiting deep learning packages can efficiently use them. The ability of a software program to fully utilize GPUs is a big advantage.

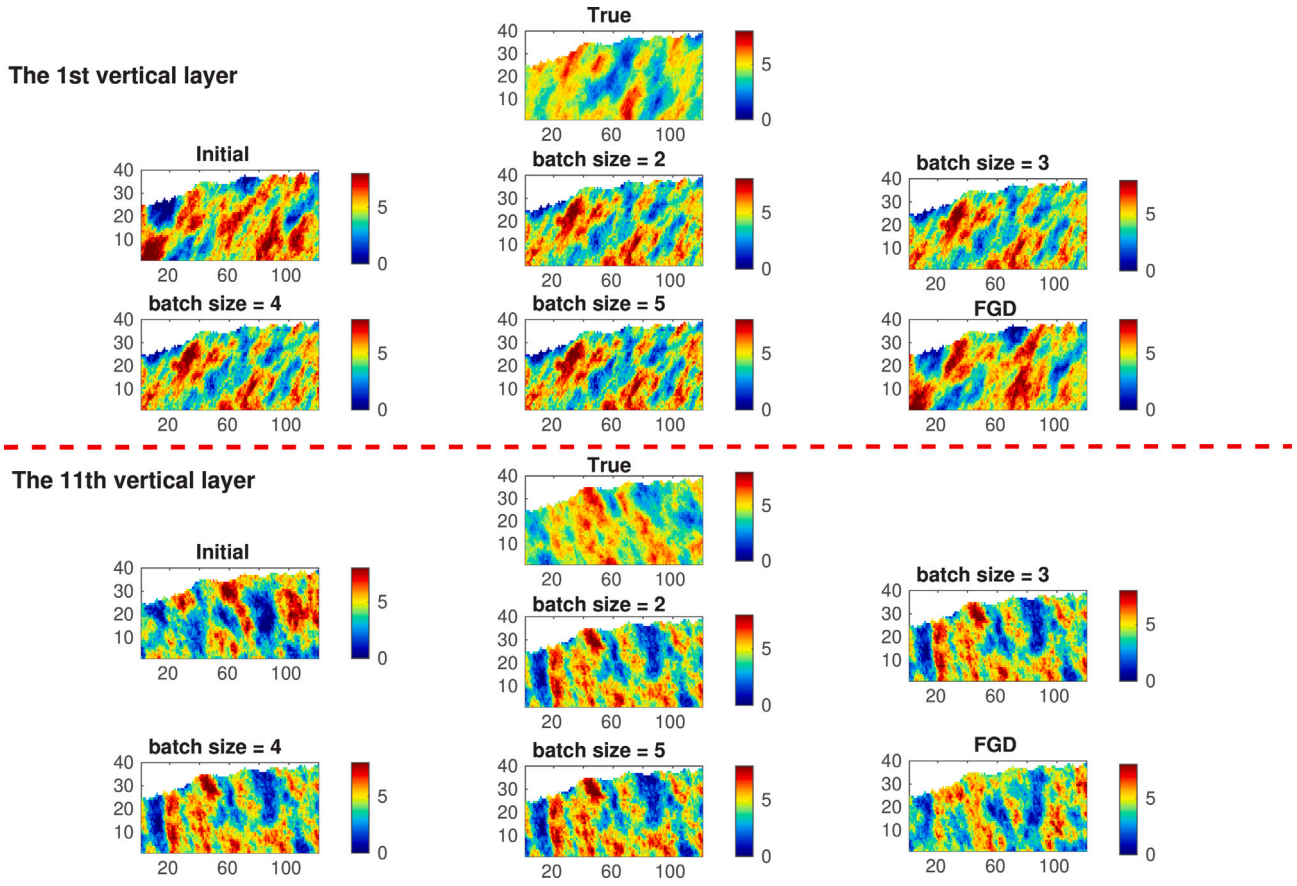



**Fig. 19.** 2D horizontal cross sections of the updated logarithmic permeability for the 1st and 11th vertical layer under different batch size for SGD and FGD optimization methods. The cR-U-Net surrogate models are trained using $N_s$ 1000 training samples.

**Table 4**
The final objective function values, parameter misfits $e_{\mathbf{m}}$ and required number of HFM simulations for the subdomain POD-TPWL and cR-U-Net surrogate models.

| | Subdomain POD-TPWL | cR-U-Net | | | | |
|---|---|---|---|---|---|---|
| | | $N_s = 100$ | $N_s = 300$ | $N_s = 500$ | $N_s = 800$ | $N_s = 1000$ |
| $J \times 10^6$ | 0.1689 | 0.4366 | 0.3890 | 0.2699 | 0.2083 | 0.2021 |
| Reference $\times 10^6$ | | | 0.0785 | | | |
| Tolerance $\times 10^6$ | | | 0.4724 | | | |
| $e_m$ | 0.6534 | 0.7190 | 0.6370 | 0.5371 | 0.4704 | 0.4374 |
| Number of HFM simulations | 158 | 100 | 300 | 500 | 800 | 1000 |

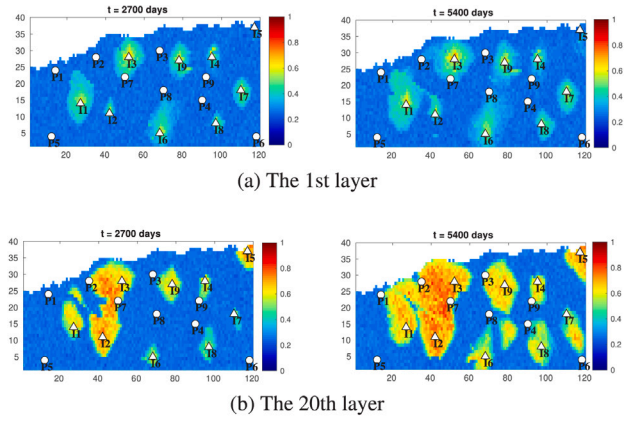
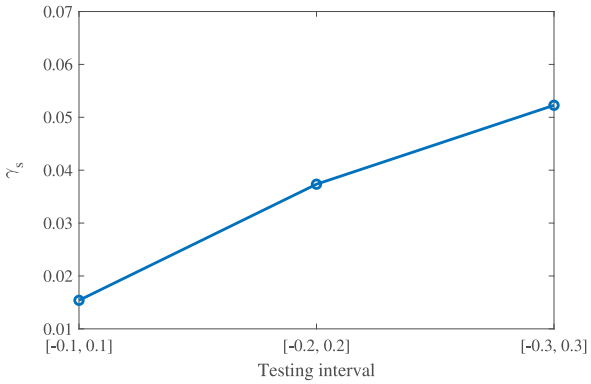


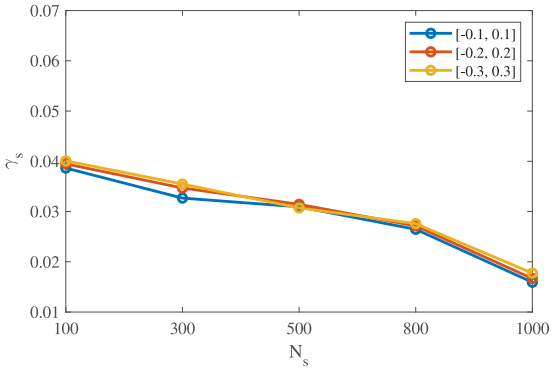

**Fig. 20.** Noise distribution of water saturation at the days 2700 and days 5400 used in this comparative study. Normal distributed independent measurement noise with a standard deviation equal to 5% of the 'true' data value, was added to all observations.

### 4.6. Additional remarks

One of the most prominent merit of the proposed surrogate-assisted history matching method with deep neural network is that it can perform massively parallel computations on GPUs. Although the availability of GPU's must be a precondition for efficiently training the cR-U-Net surrogate model, this should be not a big problem as GPUs are relatively cheap and easy to install on existing computer systems. Nevertheless, limitations of the proposed method do exist. First, the choice of a neural network architecture is flexible, but it is still relatively subjective. Second, although our proposed surrogate-assisted history matching does not involve additional HFM simulations. the intrinsic disadvantage of the DNN method appears to be that it may not be clear beforehand how many training samples are needed to obtain the desired accuracy. In order to obtain highly accurate DNN model, we have to blindly increase the number of training samples as much as possible in most cases. This inevitably leads to redundant model simulations sometimes when satisfactory results can be obtained using a small dataset. Some promising results have been obtained in this




(a) Subdomain POD-TPWL




(b) cR-U-Net

**Fig. 21.** Field-average relative error $\gamma_s$ as a function of domain decomposition, testing interval and number of training samples. (a) Subdomain POD-TPWL; (b) cR-U-Net surrogate model.

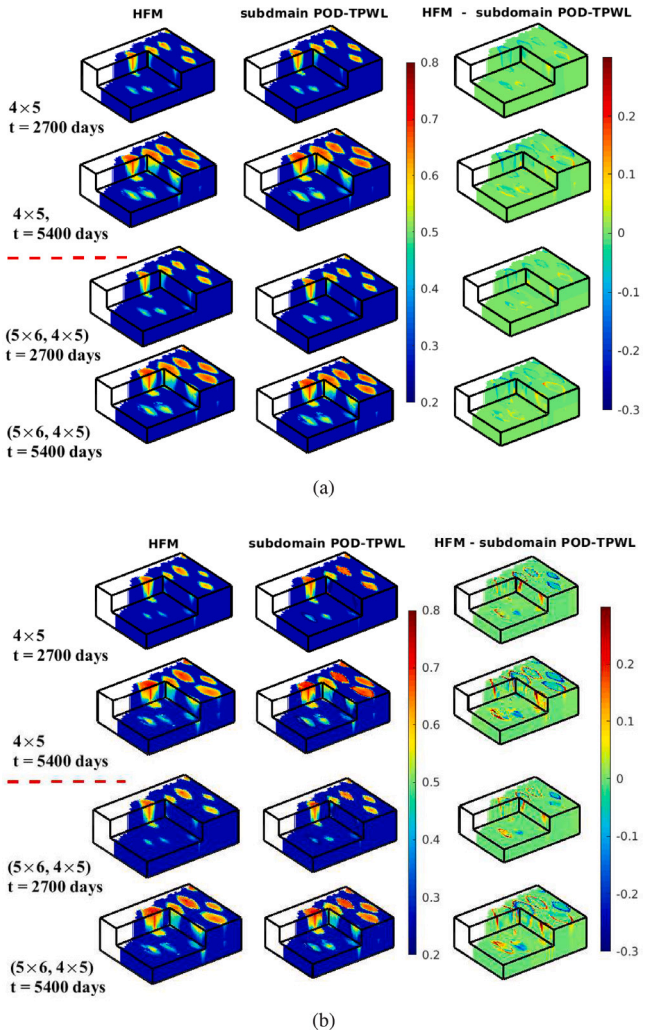



**Fig. 22.** Illustration of time-varying phase saturation for a random parameter field predicted from subdomain POD-TPWL, HFM and their residual errors at 2700 days and 5400 days of production . The cR-U-Net model is trained using $N_s$ = 1000 and 100 training samples. (a) Testing interval [−0.1, 0.1]; (b) Testing interval [−0.3, 0.3].

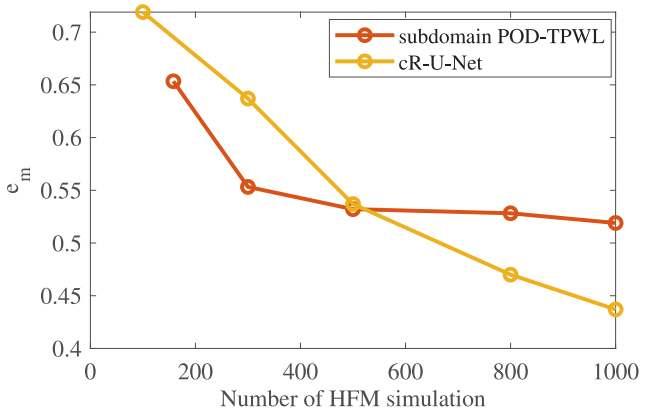



**Fig. 23.** Parameter misfits $e_m$ with respective to the number of HFM simulations for the cR-U-Net and subdomain POD-TPWL, respectively. We gradually increase the HFM simulations for constructing the subdomain POD-TPWL surrogate models.

paper, however, further research is needed to gain insights about how the DNN surrogate works for the gradient-based inversion, when it will work (weak- or strong non-linear model) and the trade-offs between accuracy and computational complexity. The degree of model non-linearity dominates the convergence performance of the gradient-based optimizations. Nevertheless, whether the non-linearity of the high-fidelity model is correctly represented by the DNN surrogate model has not been fully understood yet.

Our proposed DNN surrogate method has demonstrated to be very efficient in generating multiple posterior models. In the community of data assimilation, the ensemble-based assimilation methods are particularly useful to quantify the models uncertainty for the realistic cases and many successful applications have been reported in the literature (Emerick and Reynolds, 2012). A direct application of the proposed DNN surrogates to field-scale models with a large degree of freedom definitely poses huge computational challenges and memory requirements for training them. This drawback seemingly makes the DNN approach less attractive than that of the ensemble-based assimilation methods. However, assimilating a large number of measurements easily causes severe rank deficiency and eventually ensemble collapse problem. In general, increasing ensemble size or covariance localization have always been a standard approach to resolve this issue (Emerick and Reynolds, 2011; Yeo et al., 2014). It is, however, very challenging to design localization regions and localization methods usually require problem adaption before application. By contrast, the filter divergence issue hardly occurs in our proposed DNN-assisted history matching methodology.

In this study, the main data used in the history matching procedure are the time-dependent grid-based saturation maps predicted from the fluid flow simulation model. History matching real seismic data generally involves a coupled system (i.e., fluid flow simulation model and the rock–physics model). On the one hand, there has been recent progress

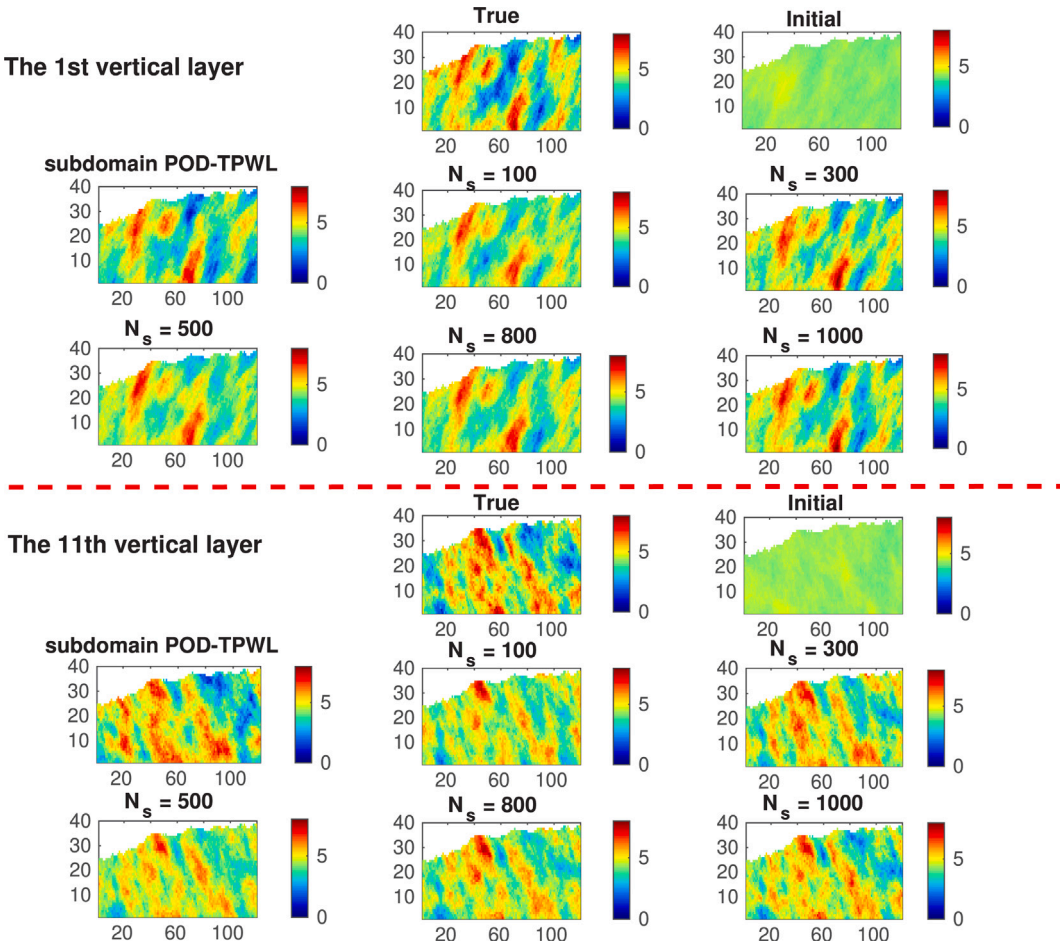



**Fig. 24.** 2D horizontal cross sections of the updated logarithmic permeability fields for assessing the accuracy of the history matching results using the cR-U-Net and subdomain POD-TPWL surrogate models. The 1st and 11th vertical layer are shown here.

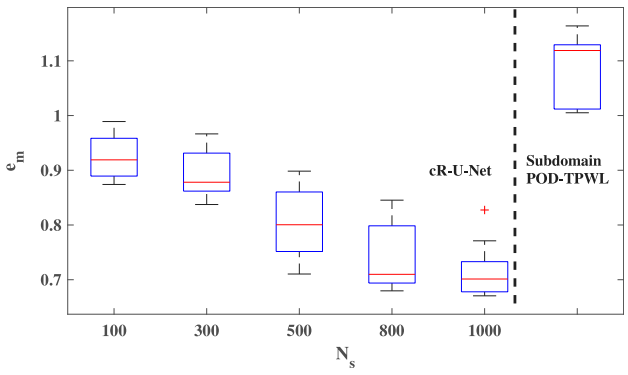



**Fig. 25.** Boxplot of the parameter misfits $e_m$ of logarithmic permeability after the history matching using cR-U-Net and subdomain POD-TPWL surrogate models. The 3D domain decomposition with adaptive strategy is implemented.

in reduced-order modeling for coupled flow and geomechanics. For example, Jin et al. proposed a reduced-order-modeling framework to simulate coupled flow-geomechanics problems (Jin et al., 2020). The conventional POD-TPWL is constructed to efficiently simulate the global pressure, saturation and stress fields. These results imply the potential of applying our approach to address coupled flow and geomechanics problems, since our proposed subdomain approach is very similar to the POD-TPWL method. On the other hand, the potential applications of deep-learning surrogate models to address time-lapse seismic data recently also have been reported. For example, Zhong et al. proposed a data-driven deep learning method to facilitate the solution of both the forward and inverse problems in seismic inversion (Zhong et al., 2020). Specifically, a cycle generative adversarial neural network (CycleGAN) model is trained to learn the bidirectional functional mappings between the reservoir dynamic property changes and seismic attribute changes, such that both forward and inverse solutions can be obtained efficiently from the trained model. Similarly, Kaur et al. design a deep learning framework for carbon dioxide ($CO_2$) saturation monitoring to determine the geological controls on storage of $CO_2$ (Kaur et al., 2020). The network is trained with a few time-lapse seismic images and their corresponding changes in saturation values. The proposed algorithm provides a framework for direct estimation of $CO_2$ saturation values and plume migration in heterogeneous formations using the time-lapse seismic data. Thus, a hybrid workflow of the above DNN models, e.g., CycleGAN and our proposed cR-U-Net can be readily used for reservoir characterization and reservoir model updates involving the use of time-lapse seismic data. Alternatively, we also can train a unified DNN framework by directly learning a mapping from geological parameters to real seismic data, which deserves to be investigated in further research.

## 5. Conclusion

We have introduced an efficient deep-learning inversion framework where the original high-fidelity model is fully replaced by a Conditional Residual U-Net (cR-U-Net) surrogate model to speedup gradient-based large-scale history matching. We use the proposed surrogate model

to estimate the unknown permeability fields of a modified version of benchmark reservoir model by assimilating a large number of synthetic measurements. This work primarily explores the potential of employing the DNN surrogate to approximate the objective function's gradient using auto-differentiation. The inherent computational saving stems from the use of GPU units, which is absolute essential but also a major benefit derived from the deep learning packages. These preliminary numerical experiments carried out in representative scenarios demonstrate that the proposed methodology is highly efficient since the required number of high-fidelity model runs for training the cR-U-Net surrogate model is relatively small. In addition, the proposed deep-learning inversion algorithm with stochastic gradient optimizer demonstrates an effective convergence performance against the model and data noise for the considered large-scale history matching problem with a large number of measurements and parameters.

The comparative study between the cR-U-Net surrogate and projection-based subdomain POD-TPWL reveals that although cR-U-Net is much more efficient in generating an ensemble of posterior realizations for uncertainty quantification, the implementation of subdomain POD-TPWL is relatively flexible and does not depend on hardware requirements, e.g., GPU cards which may not be always available at hand, and the physical interpretation of the projection-based subdomain POD-TPWL also makes it easier to diagnose and comprehend. The performance of the subdomain POD-TPWL is very sensitive to the domain decomposition schemes, however, the choice of domain decomposition is very subjective. For small sample sizes the physics based subdomain POD-TPWL is more efficient and that for larger sample sizes the error of subdomain POD-TPWL at some point does not improve any more for a fixed domain decomposition strategy. The DNN is less efficient for small sample sizes, but gets better and better with larger samples sizes. Overall, both methods have their own advantages and disadvantages. Based on the results in this study, the DNN surrogate model is recommended for the purpose of model uncertainty assessment.

There are some other aspects of the proposed methodology that could possibly be improved. The complementary advantage of the projection-based ROM and DNN opens up another avenue of research and therefore deserves to be explored further. For example, the accuracy of ROM can be improved by quantifying the discrepancy between the high-fidelity model and the ROM outputs using a DNN-based error correction procedure based on our preliminary results in Huang (2019). Or we can project the high-dimensional model into the reduced-subspace as commonly done in ROM and then construct DNN surrogate in the subspace, which results in a large reduction of computational cost and memory requirement simultaneously (Wang et al., 2020).

The training of the proposed DNN surrogate model is completely independent of the history matching process. The history matching actually has the potential of instructing the construction of a DNN surrogate model along with an improved accuracy. The surrogate model could be trained with a small number of initial training samples, and then progressively enriched with more training data (i.e. re-sampled simulation runs) close to the target distribution at the history matching process. Ideally one would have an adaptive scheme to update the model after convergence with additional training runs and test if the objective function can be further decreased or not. Under these circumstances, the DNN surrogate can perhaps be retrained relatively quickly if we only add one or several new data points and just continue from the previously trained model. In order to prevent the model from overfitting to these new training points, fine-tuning or transfer learning strategy could be useful (De et al., 2020).

## CRediT authorship contribution statement

**Cong Xiao:** Conceptualization, Methodology, Software, Writing – original draft. **Hai-Xiang Lin:** Investigation, Supervision. Writing – original draft, Writing – reviewing and editing. **Olwijn Leeuwenburgh:** Geological model generation, Model design, Writing – original draft, Writing – reviewing and editing. **Arnold Heemink:** Investigation, Supervision. Writing – original draft, Writing – reviewing and editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

The first author would like to thank the China Scholarship Council (CSC) and Science Foundation of China University of Petroleum, Beijing, China (No. ZX20210069) for providing research funding. Additional Computing resources were provided by Mathematics Physics Group, Department of Applied Mathematics at Delft University of Technology. The use of open source codes OPM-$flow$ (https://opmproject.org/) is gratefully acknowledged.

## Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.petrol.2021.109287.

## References

Aanonsen, S.I., Nævdal, G., Oliver, D.S., 2009. The ensemble Kalman filter in reservoir engineering–A review. Spe J. 14 (03), 393–412.

Abadi, Martín, Agarwal, Ashish, Barham, Paul, Brevdo, Eugene, Chen, Zhifeng, Citro, Craig, Corrado, Gregory S., Davis, Andy, Dean, Jeffrey, Devin, Matthieu, Ghemawat, Sanjay, Goodfellow, Ian J., Harp, Andrew, Irving, Geoffrey, Isard, Michael, Jia, Yangqing, Józefowicz, Rafal, Kaiser, Lukasz, Kudlur, Manjunath, Levenberg, Josh, Mané, Dan, Monga, Rajat, Moore, Sherry, Murray, Derek Gordon, Olah, Chris, Schuster, Mike, Shlens, Jonathon, Steiner, Benoit, Sutskever, Ilya, Talwar, Kunal, Tucker, Paul A., Vanhoucke, Vincent, Vasudevan, Vijay, Viégas, Fernanda B., Vinyals, Oriol, Warden, Pete, Wattenberg, Martin, Wicke, Martin, Yu, Yuan, Zheng, Xiaoqiang, 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. CoRR abs/1603.04467, arXiv:1603.04467.

Ahmadi, Mohammad Ali, 2015. Developing a robust surrogate model of chemical flooding based on the artificial neural network for enhanced oil recovery implications. Math. Probl. Eng. 2015.

Altaf, Muhammad Umer, Heemink, Arnold W., Verlaan, Martin, 2009. Inverse shallow-water flow modeling using model reduction. Int. J. Multiscale Comput. Eng. 7 (6).

Anon., 1995. GSLIB: Geostatistical software library and user's guide. Technometrics.

Asher, Michael J., Croke, Barry F.W., Jakeman, Anthony J., Peeters, Luk J.M., 2015. A review of surrogate models and their application to groundwater modeling. Water Resour. Res. 51 (8), 5957–5973.

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

Cardoso, M.A., Durlofsky, L.J., Sarma, P., 2009. Development and application of reduced-order modeling procedures for subsurface flow simulation. Internat. J. Numer. Methods Engrg. 77 (9), 1322–1350.

Chen, C., Gao, G., Honorio, J., Gelderblom, P., Jimenez, 2014. Integration of principal-component-analysis and streamline information for the history matching of channelized reservoirs. In: Spe Technical Conference and Exhibition.

Chen, Chaohui, Gao, Guohua, Ramirez, Benjamin A., Vink, Jeroen C., Girardi, Alejandro M., et al., 2015. Assisted history matching of channelized models using pluri-principal component analysis. In: SPE Reservoir Simulation Symposium. Society of Petroleum Engineers.

Chen, Yan, Oliver, Dean S., 2012. Multiscale parameterization with adaptive regularization for improved assimilation of nonlocal observation. Water Resour. Res. 48 (48), 4503.

Courant, R., Hilbert, D., 1962. Methods of Mathematical Physics. Wiley Interscience.

Courtier, Philippe, Thépaut, J.-N, Hollingsworth, Anthony, 1994. A strategy for operational implementation of 4D-var, using an incremental approach. Q. J. R. Meteorol. Soc. 120 (519), 1367–1387.

Dai, Cheng, Xue, Liang, Zhang, Dongxiao, Guadagnini, Alberto, 2016. Data-worth analysis through probabilistic collocation-based ensemble Kalman filter. J. Hydrol. 540, 488–503.

De, Subhayan, Britton, Jolene, Reynolds, Matthew, Skinner, Ryan, Jansen, Kenneth, Doostan, Alireza, 2020. On transfer learning of neural networks using bi-fidelity data for uncertainty propagation. arXiv preprint arXiv:2002.04495.

Emerick, A., Reynolds, A., 2011. Combining sensitivities and prior information for covariance localization in the ensemble Kalman filter for petroleum reservoir applications. Comput. Geosci. 15 (2), 251–269.

Emerick, A.A., Reynolds, A.C., 2012. History matching time-lapse seismic data using the ensemble Kalman filter with multiple data assimilations. Comput. Geosci. 16 (3), 639–659.

Evensen, G., 2009. Data Assimilation: The Ensemble Kalman Filter. Springer Science & Business Media.

Gassmann, Fritz, 1951. Elastic waves through a packing of spheres. Geophysics 16 (4), 673–685.

Goodfellow, Ian, Bengio, Yoshua, Courville, Aaron, 2016. Deep Learning. MIT press.

He, Jincong, Durlofsky, Louis J., et al., 2014. Reduced-order modeling for compositional simulation by use of trajectory piecewise linearization. SPE J. 19 (05), 858–872.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Identity mappings in deep residual networks. In: European Conference on Computer Vision.

Heo, Y.J., Kim, S.J., Kim, D., Lee, K., Chung, W.K., 2018. Super-high-purity seed sorter using low-latency image-recognition based on deep learning. IEEE Robot. Autom. Lett. 3 (4), 1.

Huang, Jie, 2019. Machine learning based error modeling for surrogate model in oil reservoir problem.

Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q., 2016. Densely connected convolutional networks.

Jafarpour, Behnam, McLaughlin, Dennis B., 2008. History matching with an ensemble Kalman filter and discrete cosine parameterization. Comput. Geosci. 12 (2), 227–244. http://dx.doi.org/10.1007/s10596-008-9080-3.

Jin, Zhaoyang Larry, Garipov, Timur, Volkov, Oleg, Durlofsky, Louis J., et al., 2020. Reduced-order modeling of coupled flow and quasistatic geomechanics. SPE J. 25 (01), 326–346.

Jin, Z.L., Liu, Y., Durlofsky, L.J., 2019. Deep-learning-based reduced-order modeling for subsurface flow simulation. arXiv preprint arXiv:1906.03729.

Kaleta, Małgorzata P., Hanea, Remus G., Heemink, Arnold W., Jansen, Jan-Dirk, 2011. Model-reduced gradient-based history matching. Comput. Geosci. 15 (1), 135–153.

Kaur, Harpreet, Sun, Alexander, Zhong, Zhi, Fomel, Sergey, 2020. Time-lapse seismic data inversion for estimating reservoir parameters using deep learning. In: SEG Technical Program Expanded Abstracts 2020. Society of Exploration Geophysicists, pp. 1720–1724.

Kingma, Diederik P., Ba, Jimmy, 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Liu, Yimin, Durlofsky, Louis J., et al., 2019. Multilevel strategies and geological parameterizations for history matching complex reservoir models. SPE J..

Liu, Yimin, Sun, Wenyue, Durlofsky, Louis J., 2018. A deep-learning-based geological parameterization for history matching complex models. Math. Geosci. 1–42.

Mannseth, T., Fossum, K., 2018. Assimilating spatially dense data for subsurface applications—balancing information and degrees of freedom. Comput. Geosci. 22 (5), 1323–1349.

Markovinović, R., Jansen, J.D., 2006. Accelerating iterative solution methods using reduced-order models as solution predictors. Internat. J. Numer. Methods Engrg. 68 (5), 525–541.

Matthews, J.D., Carter, J.N., Stephen, K.D., Zimmerman, R.W., Skorstad, A., Manzocchi, T., Howell, J.A., 2008. Assessing the effect of geological uncertainty on recovery estimates in shallow-marine reservoirs: The application of reservoir engineering to the SAIGUP project. Pet. Geosci. 14 (1), 35–44. http://dx.doi.org/10.1144/1354-079307-791.

Mo, S., Zabaras, N., Shi, X., Wu, J., 2018. Deep autoregressive neural networks for high-dimensional inverse problems in groundwater contaminant source identification.

Mo, S., Zhu, Y., Zabaras, N., Shi, X., Wu, J., 2019. Deep convolutional encoder-decoder networks for uncertainty quantification of dynamic multiphase flow in heterogeneous media. Water Resour. Res. 55 (1).

Oliver, D.S., Reynolds, A.C., Liu, N., 2008. Inverse Theory for Petroleum Reservoir Characterization and History Matching. Cambridge University Press.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., 2019. Pytorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems. pp. 8024–8035.

Rasmussen, A.F., Sandve, T.H., Bao, K., Lauser, A., Hove, J., Skaflestad, B., Klöfkorn, R., Blatt, M., Rustad, A.B., Sævareid, O., Lie, K.A., 2019. The open porous media flow reservoir simulator. arXiv preprint arXiv:1910.06059.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, pp. 234–241.

Salazar, Melvin Oswaldo, Villa Piamo, Jose Reinaldo, et al., 2007. Permeability upscaling techniques for reservoir simulation. In: Latin American & Caribbean Petroleum Engineering Conference. Society of Petroleum Engineers.

Tang, M., Liu, Y., Durlofsky, L.J., 2019. A deep-learning-based surrogate model for data assimilation in dynamic subsurface flow problems. arXiv preprint arXiv:1908.05823.

Vermeulen, P.T.M., Heemink, A.W., 2006. Model-reduced variational data assimilation. Mon. Weather Rev. 134 (10), 2888–2899.

Vo, Hai X., Durlofsky, Louis J., 2015. Data assimilation and uncertainty assessment for complex geological models using a new PCA-based parameterization. Comput. Geosci. 19 (4), 747–767.

Vo, Hai X., Durlofsky, Louis J., 2016. Regularized kernel PCA for the efficient parameterization of complex geological models. J. Comput. Phys. 322, 859–881.

Wang, Min, Cheung, Siu Wun, Leung, Wing Tat, Chung, Eric T., Efendiev, Yalchin, Wheeler, Mary, 2020. Reduced-order deep learning for flow dynamics. the interplay between deep learning and model reduction. J. Comput. Phys. 401, 108939.

Xiao, D., Fang, F., Buchan, A.G., Pain, C.C., Navon, I.M., Du, J., Hu, G., 2014. Non-linear model reduction for the Navier–Stokes equations using residual DEIM method. J. Comput. Phys. 263, 1–18.

Xiao, C., Heemink, A., Lin, H., Leeuwenburgh, O., 2020. Deep-learning inversion to efficiently handle big-data assimilation: Application to seismic history matching. In: ECMOR XVII, volume 2020. European Association of Geoscientists & Engineers, pp. 1–16.

Xiao, Cong, Leeuwenburgh, Olwijn, Lin, Hai Xiang, Heemink, Arnold, 2018. Non-intrusive subdomain POD-TPWL for reservoir history matching. Comput. Geosci. 1–29.

Xiao, Cong, Leeuwenburgh, Olwijn, Lin, Hai Xiang, Heemink, Arnold, 2019a. Non-intrusive subdomain POD-TPWL for reservoir history matching. Comput. Geosci. 23 (3), 537–565.

Xiao, Cong, Leeuwenburgh, Olwijn, Lin, Hai Xiang, Heemink, Arnold, 2019b. Subdomain POD-TPWL with local parameterization for large-scale reservoir history matching problems. arXiv preprint arXiv:1901.08059.

Xiao, D., Yang, P., Fang, F., Xiang, J., Pain, C.C., Navon, I.M., 2016. Non-intrusive reduced order modelling of fluid–structure interactions. Comput. Methods Appl. Mech. Engrg. 303, 35–54.

Xingjian, S.H.I., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K., Woo, W.C., 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In: Advances in Neural Information Processing Systems. pp. 802–810.

Yeo, M.J., Jung, S.P., Choe, J., 2014. Covariance matrix localization using drainage area in an ensemble Kalman filter. Energy Sources A 36 (19), 2154–2165.

Young, T., Hazarika, D., Poria, S., Cambria, E., 2018. Recent trends in deep learning based natural language processing [review article]. IEEE Comput. Intell. Mag. 13 (3), 55–75.

Zhang, Pinggang, Pickup, Gillian E., Christie, Michael A., et al., 2008. A new practical method for upscaling in highly heterogeneous reservoir models. SPE J. 13 (01), 68–76.

Zhong, Z., Sun, A.Y., Jeong, H., 2019. Predicting $CO_2$ plume migration in heterogeneous formations using conditional deep convolutional generative adversarial network. Water Resour. Res..

Zhong, Zhi, Sun, Alexander Y., Wu, Xinming, 2020. Inversion of time-lapse seismic reservoir monitoring data using CycleGAN: A deep learning-based approach for estimating dynamic reservoir property changes. J. Geophys. Res. Solid Earth 125 (3), e2019JB018408.

Zhou, Zitong, Tartakovsky, Daniel M., 2020. Markov chain Monte Carlo with neural network surrogates: Application to contaminant source identification. arXiv preprint arXiv:2003.02322.