Journal of Computational Physics 413 (2020) 109456

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A deep-learning-based surrogate model for data assimilation in dynamic subsurface flow problems

Meng Tang [a,b,*], Yimin Liu [a,b], Louis J. Durlofsky [a,b]

[a] 367 Panama Street, Stanford, CA, 94305, United States of America
[b] Department of Energy Resources Engineering, Stanford University, United States of America

ARTICLE INFO

Article history:
Received 16 August 2019
Received in revised form 5 February 2020
Accepted 3 April 2020
Available online 15 April 2020

Keywords:
Surrogate model
Deep-learning
Reservoir simulation
History matching
Inverse modeling

ABSTRACT

A deep-learning-based surrogate model is developed and applied for predicting dynamic subsurface flow in channelized geological models. The surrogate model is based on deep convolutional and recurrent neural network architectures, specifically a residual U-Net and a convolutional long short term memory recurrent network. Training samples entail global pressure and saturation maps, at a series of time steps, generated by simulating oil-water flow in many (1500 in our case) realizations of a 2D channelized system. After training, the 'recurrent R-U-Net' surrogate model is shown to be capable of predicting accurate dynamic pressure and saturation maps and well rates (e.g., time-varying oil and water rates at production wells) for new geological realizations. Assessments demonstrating high surrogate-model accuracy are presented for an individual geological realization and for an ensemble of 500 test geomodels. The surrogate model is then used for the challenging problem of data assimilation (history matching) in a channelized system. For this study, posterior reservoir models are generated using the randomized maximum likelihood method, with the permeability field represented using the recently developed CNN-PCA parameterization. The flow responses required during the data assimilation procedure are provided by the recurrent R-U-Net. The overall approach is shown to lead to substantial reduction in prediction uncertainty. High-fidelity numerical simulation results for the posterior geomodels (generated by the surrogate-based data assimilation procedure) are shown to be in essential agreement with the recurrent R-U-Net predictions. Comparisons to simulation-based data assimilation results further highlight the accuracy and applicability of the recurrent R-U-Net, and suggest that it may enable the use of more formal posterior sampling methods in realistic problems.

1. Introduction

Reliable subsurface flow forecasts are essential for the effective management of oil, gas and groundwater resources. The intrinsic uncertainty in subsurface characterizations can, however, lead to substantial uncertainty in subsurface flow predictions. Inverse modeling, also referred to in this context as data assimilation or history matching, entails the calibration of geological models based on observed data of various types. The resulting posterior (history matched) models generally

* Corresponding author at: 367 Panama Street, Stanford, CA, 94305, United States of America.
E-mail addresses: mengtang@stanford.edu (M. Tang), yiminliu@stanford.edu (Y. Liu), lou@stanford.edu (L.J. Durlofsky).

https://doi.org/10.1016/j.jcp.2020.109456
0021-9991/

provide predictions with narrower uncertainty ranges, and are thus more useful for reservoir/aquifer management. Inverse modeling algorithms can, however, be very computationally intensive, particularly when the forward model entails complex physical processes and contains a large number of grid blocks. For such cases it would be very beneficial to have access to an accurate surrogate model that can be used in place of the original model for the majority of the required function evaluations.

In this paper, we introduce a new deep-learning-based surrogate model that can accurately capture the evolution of high-dimensional (global) pressure and saturation fields, along with well phase flow rate data, given the subsurface geological characterization. The surrogate model employs convolutional neural networks (CNNs) to capture the nonlinear relationship between the geological parameter map (permeability in our case) and subsurface flow state maps. A type of recurrent neural network (RNN) is applied to capture the temporal evolution of the system. Given a sufficient number of training samples (1500 in our case), this surrogate model can provide flow predictions in close agreement with the underlying flow simulator, but with a significant reduction in computational cost. Thus this approach enables the application of accurate but computationally demanding inverse modeling procedures.

There has been extensive research on constructing surrogate models for subsurface flow prediction. These can be generally classified, based on the mathematical formulation, into physics-based and data-driven procedures (though these categories are not mutually exclusive). The physics-based methods typically neglect or simplify physical or numerical aspects of the problem, through, for example, reduced-physics modeling, coarse-grid modeling, or proper orthogonal decomposition (POD) based reduced-order modeling (ROM). A variety of POD-based ROMs, in which the state variables and the system of equations are projected into low-dimensional space and then solved, have been applied for a range of subsurface flow problems [1–5]. These ROMs can be effective, although they are generally only accurate when new (test) runs are sufficiently 'close' to training runs. In addition, the application of POD-based ROMs for inverse modeling has been somewhat limited, though a few studies have shown promise in this area [6,7].

Data-driven approaches, on the other hand, rely purely on simulation data to train a statistical model to approximate the input-output relationship of interest. Along these lines, Hamdi et al. [8] applied a Gaussian process for the surrogate modeling of a 20-parameter unconventional-gas reservoir system in the context of history matching. A polynomial chaos expansion surrogate model was constructed by Bazargan et al. [9] for inverse modeling with a 40-parameter representation of a 2D fluvial channelized reservoir undergoing waterflooding. Artificial neural networks (ANNs) were applied by Costa et al. [10] to build a 16-parameter surrogate model to assist in the history matching of an oil-water system. Although these data-driven surrogates each have their own advantages and drawbacks, they share the limitation of applicability only for relatively low-dimensional problems.

Recent advances in deep neural networks, and their successful application for high-dimensional data regression in image recognition [11–13] and natural language processing [14,15], have stimulated research on deep-learning-based surrogate modeling for high-dimensional nonlinear systems. In contrast to shallow ANNs, carefully designed deep neural networks can capture complex high-dimensional nonlinearities, with relatively limited training data, while avoiding overfitting [16,17]. Within a subsurface flow setting, Zhu and Zabaras [18] first introduced a fully convolutional encoder-decoder network to approximate flow quantities. They considered single-phase steady-state flow in models characterized by Gaussian permeability fields and demonstrated that their deep convolutional neural network, trained with a limited amount of data, was able to predict high-dimensional pressure maps. Subsequent applications, again with Gaussian permeability fields, have included the prediction of CO_2 saturation plumes [19] and groundwater contaminant concentration [20] in the context of uncertainty quantification and inverse modeling, respectively. More recently, the extended deep residual dense convolutional neural network was successfully applied by Mo et al. [21] for groundwater solute transport problems with 2D and 3D non-Gaussian permeability fields parameterized using an adversarial autoencoder. These studies demonstrate the ability of deep convolutional neural networks to capture high-dimensional relationships in subsurface flow systems.

In a recent study, Jin et al. [22] presented a deep-learning-based embed-to-control (E2C) framework, where a constrained autoregressive strategy was applied on the low-dimensional representation of the state maps. This work involved a single (fixed) geological characterization, and the surrogate model was trained to predict the oil-water reservoir response for a wide range of well settings (e.g., time-varying injection/production rates or wellbore pressures). This capability is required for production optimization, where many different combinations of well settings must be considered. The E2C method involves an encoder, which projects system variables to a low-dimensional latent space, a linear transition model that approximates the low-dimensional system dynamics for the specified well settings, and a decoder that projects solutions back to physical space. The methodology and the target application addressed in [22] are quite different from those considered in this work. Here, our intent is to predict flow responses for new geomodels all subjected to the same well settings, while in [22] the goal was to predict flow for a single realization under different well settings.

In this work, we develop and apply a new surrogate model for nonlinear, two-phase, oil-water dynamic systems in formations characterized by channelized (non-Gaussian) permeability fields. The target application is data assimilation, also referred to as history matching. We propose a deep-learning-based surrogate modeling framework that adopts a recurrent neural network rather than an autoregressive strategy, as was used in, e.g., [20] and [22]. In this study, the injection and production wells are specified to operate under wellbore pressure control, in which case well rates, and thus the time evolution of the saturation field, can vary significantly from realization to realization. This results in additional challenges for the surrogate since, at a given time, the amount of fluid in the model differs between cases.

Our surrogate model is trained using numerical solutions for flow through a number of different geological realizations (drawn from a single geological scenario). The surrogate model then provides very fast predictions of pressure and saturation and well flow rate data, which can be used in the inverse modeling procedure. The deep-learning-based surrogate model developed here uses a convolutional U-Net [23] architecture to approximate the state responses from the (input) permeability field. Above this U-Net, a recurrent architecture, specifically long short term memory (LSTM) [24,25], is incorporated to capture the time-dependent evolution of the global pressure and saturation state maps.

This paper proceeds as follows. In Section 2, we provide the underlying flow equations and then describe the surrogate model, in which a residual U-Net and a recurrent architecture LSTM are combined to capture both spatial and temporal information. In Section 3, the surrogate model is applied for oil-water flow involving multiple realizations of a channelized system, with flow driven by 25 wells under pressure control. A detailed assessment of model accuracy, in terms of global states and well-rate quantities, is presented. Then, in Section 4, we apply the surrogate model to history match a channelized geomodel. A randomized maximum likelihood framework is used to generate multiple posterior realizations, and posterior (surrogate-based) flow predictions are verified through comparison to numerical flow simulations. In Section 5, we summarize this work and provide suggestions for future investigations. In the Appendices, we provide model architecture details and flow results for an additional case involving four wells in a channelized system.

2. Governing equations and recurrent R-U-Net formulation

In this section, we present the governing flow equations and then describe our deep-learning-based surrogate model for dynamic two-phase subsurface flow problems. The key aspects of the surrogate model, including model architecture, the training process, and data preprocessing, are discussed.

2.1. Governing equations for two-phase flow

In this work, we consider 2D immiscible oil-water flow problems. Combining mass conservation and Darcy's law, which relates Darcy velocity to pressure gradient and other quantities, we arrive at:

$$\nabla \cdot (\rho_j \lambda_j \mathbf{k} \nabla p_j) + q_j^{\mathrm{w}} = \frac{\partial}{\partial t}(\phi \rho_j S_j), \quad j = o,\ w. \tag{1}$$

Here j denotes phase/component, with $j = o$ for oil and w for water, ρ_j is phase density, $\lambda_j = \frac{k_{rj}}{\mu_j}$ is the phase mobility (here $k_{rj}(S_j)$ is the relative permeability, a prescribed function of phase saturation S_j that is usually derived from laboratory measurements, and μ_j is phase viscosity), $\mathbf{k}$ is the absolute permeability tensor, p_j is the phase pressure, q_j^{w} is the source/sink term (superscript w indicates well), t is time, and ϕ is the rock porosity. The governing equations are completed by noting that the phase saturations sum to unity, and that the phase pressures are related through the prescribed capillary pressure P_c; i.e., $p_o - p_w = P_c(S_w)$. In this work we neglect capillary pressure effects, which is common in large-scale reservoir simulation [26], so $p_o = p_w = p$. Note also that Eq. (1) is written for horizontal $(x - y)$ systems, so gravitational effects do not appear.

Eq. (1) is discretized using a fully implicit finite volume method, as is standard in oil reservoir simulation. The primary variables are S_w and p. Fluid is introduced and removed from the system via wells, and the source term, for a well in grid block i, is modeled using the Peaceman representation [27]:

$$\left(q_j^{\mathrm{w}}\right)_i = WI_i (\lambda_j \rho_j)_i (p_i - p^{\mathrm{w}}). \tag{2}$$

Here p_i and p^{w} denote the well block and wellbore pressure, respectively, and WI_i denotes the well index, given by

$$WI_i = \frac{2\pi k_i \Delta z}{\ln(r^0/r^{\mathrm{w}})}, \tag{3}$$

where k_i is the (isotropic) permeability in grid block i, Δz is the grid block thickness, r^{w} is the wellbore radius, and $r^0 = 0.14\sqrt{(\Delta x)^2 + (\Delta y)^2}$, where Δx and Δy are grid block dimensions in the x and y directions. Note that Eq. (3) applies for a fully penetrating vertical well, centered in the grid block, and isotropic permeability k_i. Analogous expressions have been developed for a wide range of more general cases.

We see from Eqs. (2) and (3) that, when wellbore pressure p^{w} is specified (as it is in our case), well injection and phase production rates depend directly on the well-block states p and S_w. Both of these quantities are determined from the solution of Eq. (1), so even slight inaccuracy in p and S_w can have a large impact on well rates. In addition, because the phase mobility λ_j (recall $\lambda_j = k_{rj}(S_w)/\mu_j$) is a highly nonlinear function of S_w in many two-phase scenarios, the impact of well-block saturation on rates can be particularly strong.

2.2. Data-driven surrogate modeling in reservoir simulation

In applications such as inverse modeling and uncertainty quantification, we need to solve the discretized versions of Eq. (1) hundreds or thousands of times, for different reservoir models, but under identical initial and boundary conditions. A single simulation run can be expressed as

$$\mathbf{x} = f(\mathbf{m}), \tag{4}$$

where f indicates the reservoir simulation process, $\mathbf{m} \in \mathbb{R}^{n_b}$ denotes the geological model (taken to be the permeability value in every grid block in the model), and $\mathbf{x} \in \mathbb{R}^{2n_b n_t}$ denotes the state maps (p and S_w in every grid block) at all n_t time steps in the simulation. Here $n_b = n_x n_y n_z$ is the total number of grid blocks in the model, with n_x, n_y and n_z the number of blocks in the x, y and z directions.

Data-driven surrogate modeling entails an inexpensive and nonintrusive replacement of the numerical simulator. It applies statistical or machine learning tools to approximate the relationship between state responses $\mathbf{x}$ and rock properties $\mathbf{m}$ by learning from the training dataset $\{(\mathbf{m}_1, \mathbf{x}_1), \ldots, (\mathbf{m}_{n_s}, \mathbf{x}_{n_s})\}$, where n_s is the number of training samples. Traditional machine learning algorithms such as support vector machines and random forest rely on hand-designed kernels to extract useful features. They are not applicable for mapping sets of high-dimensional input ($\mathbf{m}$) to high-dimensional output ($\mathbf{x}$).

A key capability of recent deep-learning-based methods is to simultaneously detect useful features from data and to approximate input-to-output mappings. With such an approach, we approximate the reservoir simulation process as

$$\mathbf{x} \approx \hat{\mathbf{x}} = \hat{f}(\mathbf{m}; \boldsymbol{\theta}), \tag{5}$$

where $\hat{f}$ indicates the surrogate model, $\hat{\mathbf{x}} \in \mathbb{R}^{2n_b n_t}$ denotes the approximate state responses, which are expected to be close to the simulated $\mathbf{x}$, and $\boldsymbol{\theta}$ are the deep neural network parameters determined during the training procedure. Consistent with Eq. (5), our goal in this work is to develop a surrogate model to provide the time-dependent states $\hat{\mathbf{x}}$ given a permeability map $\mathbf{m}$.

2.3. R-U-Net architecture

The multiscale spatial correlations that characterize the permeability maps determine the spatial variations of the resulting state maps. Convolutional neural networks (CNNs) are specifically designed to capture this type of spatial information. In a CNN, the lower (earlier) layers generally capture more local features, while high (later) layers capture more global information [28]. A general CNN architecture can be formulated recursively through the following expression:

$$\mathbf{F}_l = \sigma(\mathbf{W}_l * \mathbf{F}_{l-1} + \mathbf{b}_l), \tag{6}$$

where $\mathbf{F}_l$ and $\mathbf{F}_{l-1}$ denote feature maps at layers l and $l-1$, σ represents a nonlinear activation function, $\mathbf{W}_l$ designates a kernel matrix, $*$ denotes the convolution operation, and $\mathbf{b}_l$ is the bias [29]. Feature maps at different layers are also functions of the input map $\mathbf{m}$, and we define $\mathbf{F}_0 = \mathbf{m}$.

Among different CNN architectures, the U-Net architecture [23], with contracting (encoding) and symmetric expanding (decoding) paths, can efficiently capture hierarchical spatial features and approximate complex nonlinearities between input and output maps. The U-Net is built upon CNNs that do not include any fully connected layers and can take input maps of arbitrary size without the need for architecture modification [30]. U-Nets have been successfully applied for various image segmentation and regression problems [23,13]. As we will see, the U-Net architecture performs well in capturing flow responses in our experiments. We believe this is because U-Nets facilitate the flow of multiscale information between the encoding and decoding nets.

To further enhance local information flow, we introduce a residual U-Net (R-U-Net), in which residual CNN modules are added into the U-Net architecture [31]. A schematic diagram of the U-shaped R-U-Net architecture is shown in Fig. 1, where the extracted features in the encoding net are copied and concatenated onto the upsampled features in the decoding net. This enables the multiscale features extracted in the encoding net to be propagated to the corresponding decoding net. A more detailed schematic of the R-U-Net architecture is presented in Appendix A.1.

We illustrate the encoding and decoding net architectures in Figs. 2 and 3. The encoding net shown in Fig. 2 takes the permeability map as input. The extracted feature maps $\mathbf{F}_k \in \mathbb{R}^{N_{x,k} \times N_{y,k} \times N_{z,k}}$ $(k = 1, \ldots, 5)$ from different encoding blocks will later be copied and fed to the decoding net. Here, $N_{x,k}$ and $N_{y,k}$ denote the dimensions of feature map $\mathbf{F}_k$ along the x and y directions, and $N_{z,k}$ indicates the number of filters in convolutional block k. From $\mathbf{F}_1$ to $\mathbf{F}_5$, the extracted features grow from simple and local to complex and global. Residual blocks are applied to produce feature map $\mathbf{F}_5$, which is the most complex and compressed feature map. This map is fed to the decoding net.

The decoding net illustrated in Fig. 3 upsamples the global feature map $\mathbf{F}_5$ to different smaller-scale feature maps, and combines it with the corresponding smaller-scale feature maps $\mathbf{F}_k$ $(k = 1, \ldots, 4)$ extracted in the symmetric encoding path. Through this procedure the decoding net provides the target state map. The transposed convolutional block [29] is applied here for upsampling. Similar to the convolutional blocks, this block has tunable weights that must be learned from the training process to achieve optimal upsampling results.

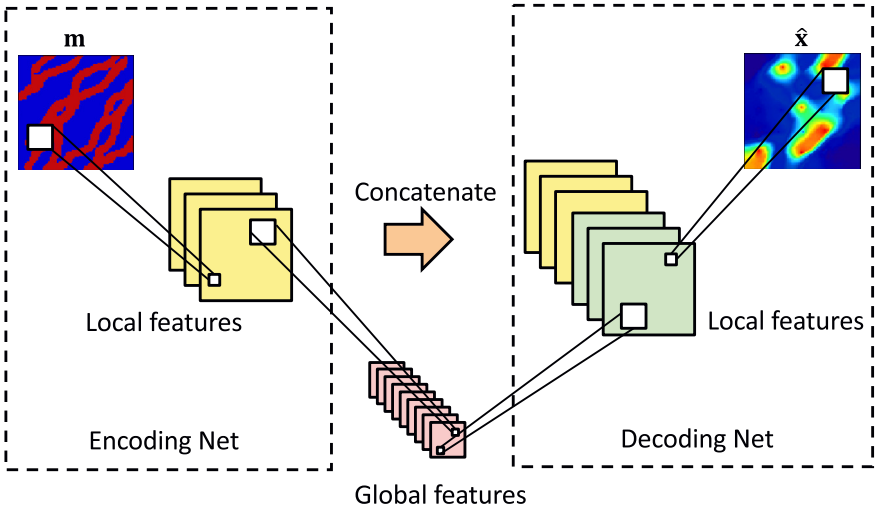


Fig. 1. Schematic illustration of R-U-Net architecture (a more detailed architecture illustration is provided in Appendix A.1). R-U-Net consists of encoding and decoding paths, where the local features extracted in the encoding path are concatenated with the upsampled features in the decoding path to produce the state map prediction.

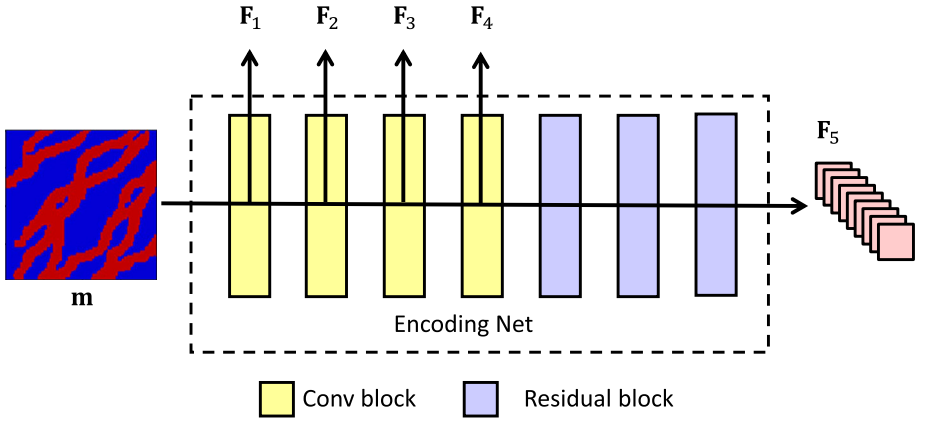


Fig. 2. Encoding net consisting of convolutional and residual blocks. The encoding net accepts the permeability map as input. The extracted multiscale features $\mathbf{F}_k \in \mathbb{R}^{N_{x,k} \times N_{y,k} \times N_{z,k}}$ $(k = 1, \ldots, 5)$ are input to the decoding net.

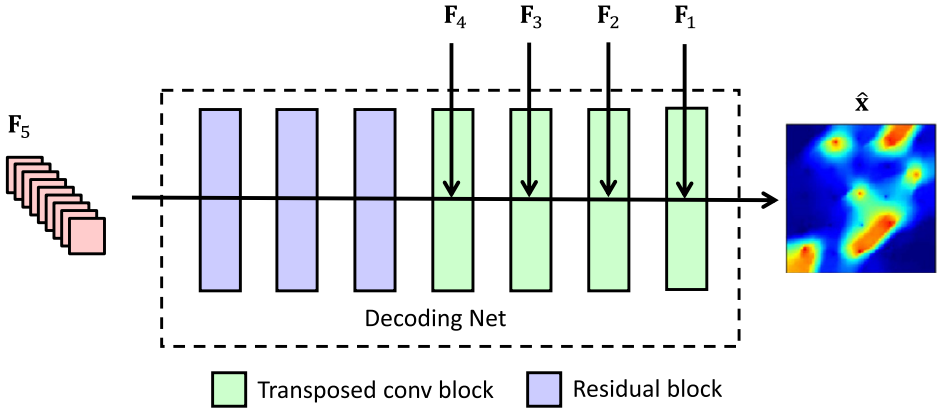


Fig. 3. Decoding net consisting of transposed convolutional (upsampling) and residual blocks. The decoding net utilizes the multiscale features $\mathbf{F}_k \in \mathbb{R}^{N_{x,k} \times N_{y,k} \times N_{z,k}}$ $(k = 1, \ldots, 5)$ extracted by the encoding net to predict the state map.

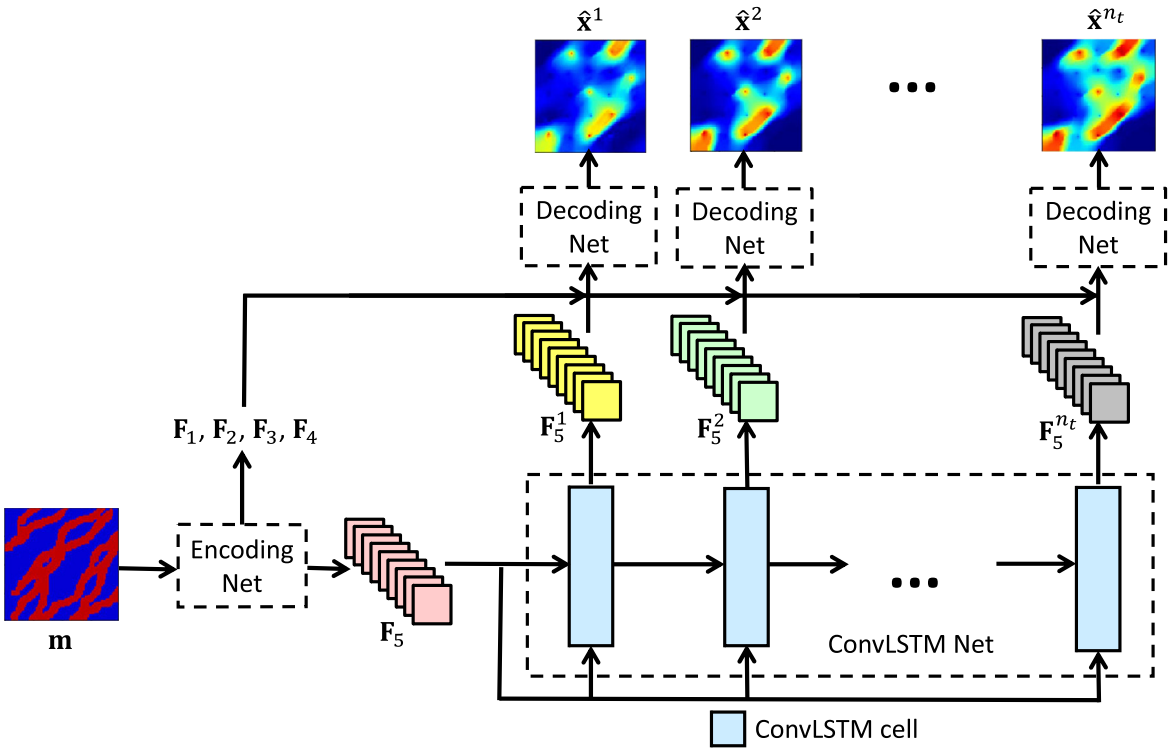


Fig. 4. Recurrent R-U-Net architecture incorporating convLSTM into the R-U-Net. The convLSTM net takes the global feature map $\mathbf{F}_5$ from the encoding net and generates a sequence of feature maps $\mathbf{F}_5^t$ $(t = 1, \ldots, n_t)$ that will be decoded into a sequence of state maps $\mathbf{x}^t$ $(t = 1, \ldots, n_t)$ separately, using the same decoding net.

The R-U-Net described thus far maps from an input permeability field to an output pressure field. However, in our inverse modeling scenarios, we are interested in reservoir dynamics, which require the surrogate model to capture the relationship between the input property map $\mathbf{m}$ and the dynamic state maps $\mathbf{x} = [\mathbf{x}^1, \ldots, \mathbf{x}^{n_t}]$ over n_t time steps. The R-U-Net as described thus far is not optimal for complex dynamic systems because the time-dependent information is not encoded and captured by the feed-forward architecture. This motivates us to investigate a recurrent R-U-Net architecture, which we now describe.

2.4. Recurrent R-U-Net architecture

To capture temporal dynamics, we apply a recurrent R-U-Net architecture. The ability of recurrent neural networks (RNNs) to capture temporal dynamics stems from the fact that the composite RNN input at the current time step contains historical information [32]. In practice, long short term memory (LSTM) [24], which is a variant of the standard RNN architecture, is often applied to treat long-term temporal dependency. This is because the set of gates used in LSTM improves information flow and solves the vanishing gradient problem that is common in standard RNNs [33].

It is desirable to incorporate an LSTM architecture into the R-U-Net architecture to capture temporal dynamics. To achieve a more compact architecture, which may significantly benefit training efficiency, the LSTM architecture is incorporated only on feature map $\mathbf{F}_5$, as illustrated in Fig. 4. This is because this feature map carries the most compressed representation of the input property map $\mathbf{m}$, and the temporal evolution of state maps can be expressed by evolving $\mathbf{F}_5$. In addition, in order to maintain the encoded spatial information while avoiding a large number of extra parameters, as would be introduced by a conventional fully connected LSTM, the convolutional LSTM (convLSTM) [25] is adopted.

The convLSTM net is composed of a chain of repeating convLSTM cells, which share the same set of (trainable) weights. In the convLSTM cell capturing information at time t, the input $\boldsymbol{\chi}^t$, output state (also referred to as the hidden state) $\mathbf{H}^t$, cell state $\mathbf{C}^t$, and the different gates are all 3D tensors. The cell state $\mathbf{C}^t$ serves as the memory of the convLSTM net and is updated via

$$\mathbf{C}^t = \mathbf{f}^t \circ \mathbf{C}^{t-1} + \mathbf{i}^t \circ \tilde{\mathbf{C}}^t, \tag{7}$$

where $\circ$ denotes the Hadamard product, $\mathbf{C}^{t-1}$ is the cell state at the previous time step, $\tilde{\mathbf{C}}^t$ is the new candidate cell state, $\mathbf{f}^t$ is the 'forget gate' that controls what information to eliminate from the previous cell state $\mathbf{C}^{t-1}$, and $\mathbf{i}^t$ is the input gate

that determines what information to update from the proposed cell state $\tilde{\mathbf{C}}^t$. The output state $\mathbf{H}^t$ is updated based on $\mathbf{C}^t$ filtered by the output gate $\mathbf{o}^t$ and is given by

$$\mathbf{H}^t = \mathbf{o}^t \circ \tanh(\mathbf{C}^t). \tag{8}$$

Here $\mathbf{o}^t$ determines which information in the cell state $\mathbf{C}^t$ is transferred to the output state $\mathbf{H}^t$. More details regarding convLSTM gates and states can be found in [25]. See Appendix A.2 for specifics on the recurrent R-U-Net architecture used here.

The convLSTM net and its variants have been used in a range of application areas, including precipitation forecasting [25], video gesture recognition [34] and MRI cardiac segmentation [35], where they have been shown to be effective in capturing both spatial and temporal information. The integration of the convLSTM net into the R-U-Net provides the recurrent R-U-Net developed in this study. As illustrated in Fig. 4, the recurrent R-U-Net takes the property map $\mathbf{m}$ as input, and the corresponding multiscale feature maps $\mathbf{F}_k$ $(k = 1, \ldots, 5)$ are extracted by the encoding net. Then the convLSTM net takes the most compressed feature map $\mathbf{F}_5$ and generates a sequence of feature maps $\mathbf{F}_5^t$ $(t = 1, \ldots, n_t)$. These are then decoded separately, by the same decoding net, into a sequence of state maps $\hat{\mathbf{x}}^t$ $(t = 1, \ldots, n_t)$. After proper training, our recurrent R-U-Net can produce a sequence of state maps $[\hat{\mathbf{x}}^1, \ldots, \hat{\mathbf{x}}^{n_t}]$ that describe reservoir dynamics for an input property map $\mathbf{m}$ (and a fixed set of well controls $\mathbf{u}$).

2.5. Training procedure

During training, in order to allow the recurrent R-U-Net to learn the temporal dynamics of the system given a permeability map $\mathbf{m}$, we minimize the difference between the sequence of target state maps $\mathbf{x}_i^t$, generated from a set of high-fidelity simulations, and the sequence of state maps $\hat{\mathbf{x}}_i^t$, found by the recurrent R-U-Net (i.e., through application of $\hat{f}(\mathbf{m}_i; \boldsymbol{\theta})$). This training set includes a sequence of states generated for each geomodel $\mathbf{m}_i$, $i = 1, \ldots, n_s$, where n_s is the total number of geomodels in the training set.

The training objective is to minimize the L^p norm of the difference between the $\mathbf{x}_i^t$ and the $\hat{\mathbf{x}}_i^t$. Extra weight is placed on the states in blocks containing wells in order to improve the accuracy of the well flow rates (computed through application of Eq. (2)). This is important here because well rates are the key data we seek to match during history matching. We note that a similar weighted loss was proposed in [20]. The specific minimization applied in this work is as follows:

$$\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{n_s}\frac{1}{n_t}\sum_{i=1}^{n_s}\sum_{t=1}^{n_t} ||\hat{\mathbf{x}}_i^t - \mathbf{x}_i^t||_p^p + \lambda \frac{1}{n_s}\frac{1}{n_t}\frac{1}{n_{\mathrm{w}}}\sum_{i=1}^{n_s}\sum_{t=1}^{n_t}\sum_{\mathrm{w}=1}^{n_{\mathrm{w}}} ||\hat{x}_i^{t,\mathrm{w}} - x_i^{t,\mathrm{w}}||_p^p. \tag{9}$$

Here λ is the additional weighting for the well states, which is applied at n_{w} well locations. Our numerical experiments showed that the use of the L^2 norm results in better predictions of the saturation maps, while the use of the L^1 norm leads to slightly more accurate pressure maps. We use two separate recurrent R-U-Nets, which are trained with an L^2 norm loss for saturation and an L^1 norm loss for pressure. As discussed below, a single net can also be used.

In the training process, the loss function is minimized by tuning the network parameters $\boldsymbol{\theta}$. The gradient of the loss function with respect to $\boldsymbol{\theta}$ is automatically computed by backpropagation [36] through the recurrent R-U-Net. In this work, we use the adaptive moment estimation (ADAM) [37] optimization algorithm, which is an extension of stochastic gradient descent (SGD). This has been found to be an effective procedure for the training of many deep neural network architectures.

The training of the recurrent R-U-Net can be accomplished efficiently, though the specific training time depends on many factors. These include the training set size, training batch size, optimizer setup and learning rate, as well as the graphics processing unit (GPU) performance. The training process converges in 80 minutes using a Nvidia Tesla V100 GPU, with a batch size of 8, for the cases considered in this study. Note that we are only able to access a portion of the GPU memory, since ours is a shared computational resource. We expect that the training time could be reduced through the use of more GPU memory. For a given set of inputs (model size, batch size, etc.), the recurrent R-U-Net should require slightly longer to train than existing surrogates, such as [20,19], due to the recurrent setup.

Although the training time can vary by case, it is essentially negligible compared to the time that would be required if we were to perform high-fidelity simulations in the history matching procedure. Specifically, our example involves a 2D geomodel defined on an 80×80 grid, and we predict pressure and saturation states at 10 time steps. After training, given a new geomodel, the recurrent R-U-Net can provide predictions for the states, at 10 time steps, in an elapsed time of about 0.01 seconds using a GPU.

There are about 2.6 million trainable parameters in our recurrent R-U-Net. As is the case with other well-designed deep neural networks, this over-parameterized network does not appear to suffer from over-fitting. A rigorous explanation for this has yet to be provided. Recent studies explain this general observation in terms of intrinsic dimension [38] and the lottery-ticket hypothesis [39], and they suggest that the number of parameters does not represent the true complexity of deep neural networks. In addition, as suggested in [18], the high-dimensional rock property and reservoir state data embed essential physical and dynamical information, and this acts to regularize the problem and thus reduce over-fitting.

2.6. Data processing

Data preprocessing is important for the effective training of deep neural networks. Constraining training data values to be near zero with proper data normalization, for example, can enhance the deep neural network training in many cases [40]. In this study, the input property map $\mathbf{m}$ is a binary geological facies map, and the output state maps $\mathbf{x}$ are water saturation and pressure in each grid block. The binary facies map is naturally represented by 0 (denoting shale/mud) and 1 (denoting sand/channel) block values, and saturation map values are physically between 0 and 1. Thus these two maps do not require any pre-processing.

Pressure map values, however, typically display large ranges, and the mean is far from 0. Given that constant-in-time wellbore pressures are specified in this study, it might seem appropriate to use a min-max normalization for all of the pressure data based on the minimum and maximum wellbore pressures applied. Such a min-max normalization for the pressure data can, however, lead to prediction error in the dynamic cases considered in this study. An alternate approach, in which we detrend the pressure data by performing separate data normalization at each time step t in the training set, was found to provide better performance. With this approach we normalize based on the time-varying grid-block pressures, not the (time-independent) wellbore pressures. Specifically, we compute the mean pressure map $\bar{\mathbf{p}}^t$ at each time step as

$$\bar{\mathbf{p}}^t = \frac{1}{n_s}\sum_{i}^{n_s} \mathbf{p}_i^t \quad (t = 1, \ldots, n_t). \tag{10}$$

At each time step t, we subtract this mean pressure map from each training pressure map to give the difference map $\hat{\mathbf{p}}_i^t$, i.e.,

$$\hat{\mathbf{p}}_i^t = \mathbf{p}_i^t - \bar{\mathbf{p}}^t, \quad i = 1, \ldots, n_s, \;\; t = 1, \ldots, n_t. \tag{11}$$

Finally, we perform min-max normalization over the difference map $\hat{\mathbf{p}}_i^t$ at each time step t via application of

$$\tilde{\mathbf{p}}_i^t = \frac{\hat{\mathbf{p}}_i^t - \min([\hat{\mathbf{p}}_1^t, \ldots, \hat{\mathbf{p}}_{n_s}^t])}{\max([\hat{\mathbf{p}}_1^t, \ldots, \hat{\mathbf{p}}_{n_s}^t]) - \min([\hat{\mathbf{p}}_1^t, \ldots, \hat{\mathbf{p}}_{n_s}^t])}, \quad i = 1, \ldots, n_s, \;\; t = 1, \ldots, n_t, \tag{12}$$

where the 'max' and 'min' operations here find the maximum and minimum scalar values for the entire model over all training pressure map samples at time step t.

The maximum and minimum values, along with all of the $\bar{\mathbf{p}}^t$ maps, are saved. These are then used for the inverse transform of the predicted pressure maps; i.e., to transform from the predicted $\tilde{\mathbf{p}}^t$ map to the physical pressure map $\mathbf{p}^t$. Numerical experimentation demonstrated that this treatment acts to enhance the accuracy of the recurrent R-U-Net pressure predictions relative to the use of a time-independent normalization.

3. Surrogate model evaluation

In this section, we describe a 2D binary facies channelized model and define the oil-water flow problem used for evaluation. We then compare the errors for the proposed recurrent R-U-Net with those for an existing autoregressive DenseED model [20]. Pressure and saturation field evolution and well responses using our model are then considered in detail. These quantities are assessed for individual realizations and for an ensemble of models. In Appendix B, results for an additional case, involving a four-well channelized system, are presented.

3.1. Flow problem setup

Binary channelized models, such as those considered here, are more demanding to treat than Gaussian models in many respects. They are, for example, much more difficult to parameterize than Gaussian models, and they are less suited for surrogate modeling since the range of flow responses can be very large. Thus the use of channelized geomodels provides a challenging test for our R-U-Net surrogate model.

One realization of the channelized system considered here, in terms of a binary facies (rock type) map, is shown in Fig. 5. The geomodel is defined on an 80×80 grid, with each grid block of size 50 m $\times$ 50 m $\times$ 10 m (in the x, y and z directions respectively). The model contains 25 wells (seven water injection wells and 18 production wells), and the geomodels are conditioned to facies type at all wells. For a particular grid block (i), $m_i = 1$ indicates channel (sand), and $m_i = 0$ indicates shale (mud). The permeability k_i is related to the facies type via the expression $k_i = a\exp(bm_i)$, with $a = 30$ md and $b = \ln(\frac{2000}{30})$. This results in $k_i = 2000$ md for blocks with sand, and $k_i = 30$ md for blocks with shale (mud). These values are within the range of actual sand and mud permeabilities in channelized systems.

All wells are specified to operate with wellbore pressure (sometimes referred to as bottom-hole pressure) specified. Injection wellbore pressures are set to 330 bar, and production wellbore pressures to 320 bar. The initial oil and water saturations are 0.9 and 0.1. The oil viscosity changes with reservoir pressure and is 0.29 cp at the initial reservoir pressure of 325 bar. Water viscosity is constant at 0.31 cp. The nonlinear oil-water relative permeability curves are shown in Fig. 6. Porosity is set to a constant value of 0.2.

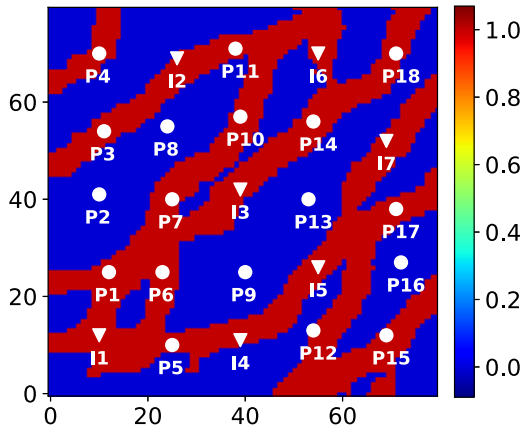


Fig. 5. Channelized 80 × 80 facies map, conditioned to facies type at 25 wells. White circles denote production wells, and white triangles denote injection wells.

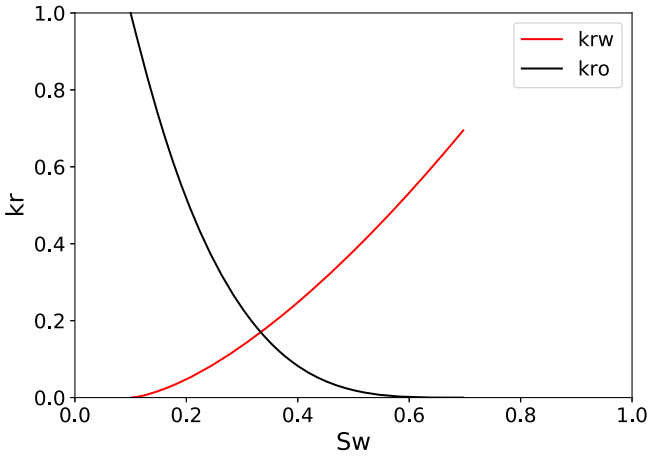


Fig. 6. Oil-water relative permeability curves.

3.2. Geomodel generation and training simulations

The recurrent R-U-Net surrogate model requires a number of training flow simulations to learn the correct mapping from the input permeability field to the dynamic output states (pressure and saturation). Geomodel realizations for this training step can be generated by a geological modeling package such as SGeMS [41]. SGeMS realizations are constructed to honor the features (i.e., multipoint spatial statistics) that exist in a prescribed geological 'training image.' Data measured at wells, referred to as hard data, are also honored in the resulting realizations.

In this work, rather than apply SGeMS directly, we use a parameterized representation for geological realizations. This parameterization is denoted CNN-PCA (PCA here indicates principal component analysis) [42,43]. This representation entails the use of a CNN to post-process PCA-based realizations (which capture two-point spatial statistics but not multipoint statistics) into geomodels with the requisite channel structure and continuity. CNN-PCA models were shown to provide flow results in close agreement with those from SGeMS realizations [42]. The key advantage of the CNN-PCA representation is that it enables us to represent the geomodel, which in this case contains 6400 grid blocks, in terms of, e.g., $O(100)$ parameters. This is very beneficial in history matching applications, since many fewer parameters need to be determined.

CNN-PCA generates high-dimensional models by first sampling a lower-dimensional variable $\boldsymbol{\xi}$ from the standard Gaussian distribution. In this study, the dimension of $\boldsymbol{\xi}$, n_ξ, is set to 100, which is generally consistent with the values used in [42,43]. Fig. 7 displays six random channelized facies models generated by CNN-PCA with $n_\xi = 100$. A final hard-thresholding step is applied to provide strictly binary fields. The white points in the figures depict the well locations, where the m_i are conditioned to honor facies data. Specifically, there are five wells drilled in mud, and 20 wells drilled in sand. Given the facies model, we construct permeability in each grid block through application of $k_i = a\exp(bm_i)$, with a and b as given previously.

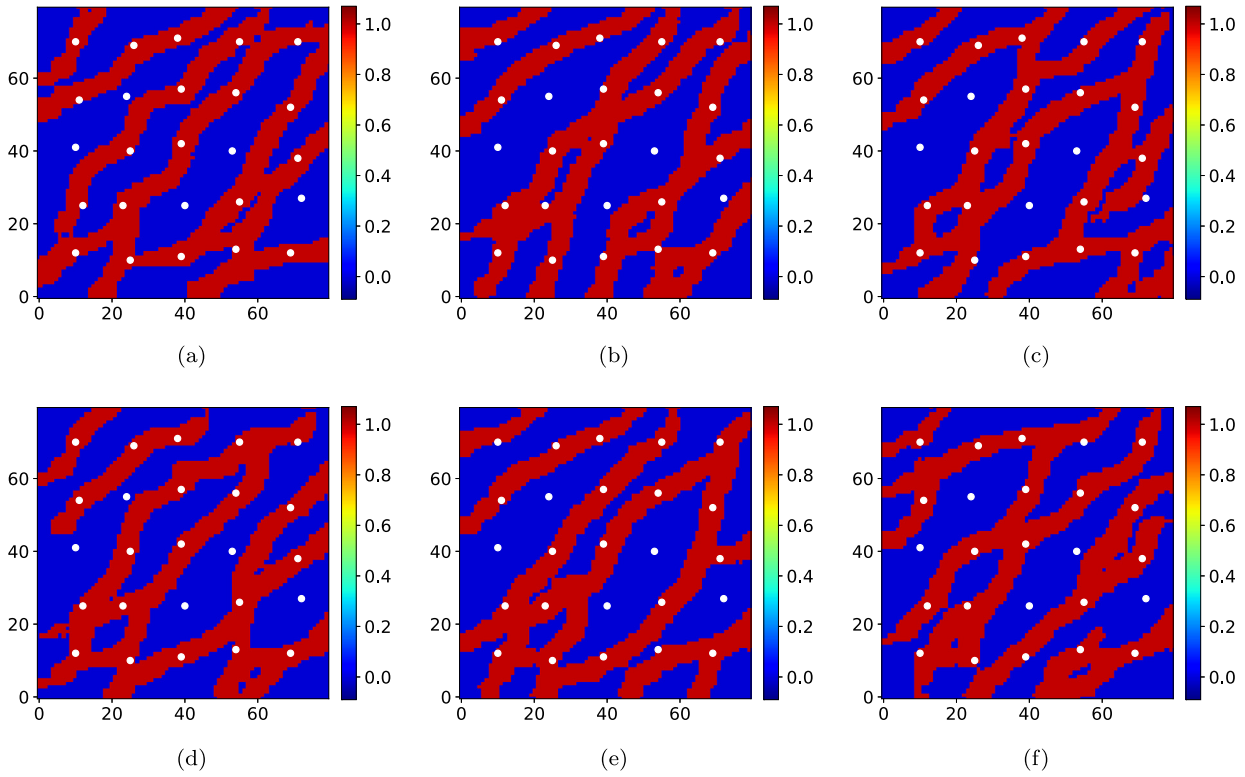


Fig. 7. Six random channelized realizations generated by CNN-PCA. White circles indicate well locations. All models honor facies data at the 25 wells. Model in (a) used in the assessments in Sections 3.4 and 3.5.

Once the geomodels are constructed, we simulate flow using a standalone 2D C++ numerical simulator. This code was verified through comparison to Stanford's Automatic Differentiation-based General Purpose Research Simulator, AD-GPRS [44], for a range of problems. Timings for the cases considered in this work are about the same between the standalone code and AD-GPRS (AD-GPRS is intrinsically faster but it involves more overhead than the standalone code, so performance on small problems may be comparable). We simulate each model over a time frame of 1000 days. We collect training data (pressure and saturation maps) at 10 time steps (at 50, 100, 200, 300, 400, 500, 600, 700, 850 and 1000 days). More variation in the states occurs at earlier times in the simulations, so the data collection is skewed to capture this.

3.3. Training procedure

As noted earlier, we found that the use of the L^1 norm loss results in more accurate recurrent R-U-Net pressure predictions, while the L^2 norm loss provides better saturation predictions. Therefore, we train two separate recurrent R-U-Nets with exactly the same architectures but with different training sets (one with pressure and one with saturation). As in existing CNN models [18,20], a single recurrent R-U-Net with two output channels could be trained to predict saturation and pressure maps simultaneously. In this case weightings for the different losses must be determined to provide results with accuracy equivalent to that achieved with separate models. The performance of a dual-output recurrent R-U-Net trained in such a way will be reported later.

There are several important hyperparameters that require specification before training, including learning rate, batch size, number of epochs, and the mixing weight λ defined in the loss function (given in Eq. (9)). The two recurrent R-U-Nets share the same hyperparameter setup, with initial learning rate $l_r = 0.003$ for the ADAM optimization algorithm, batch size $N_b = 8$, and mixing weight $\lambda = 1000$. The training of both nets converges within 200 epochs, though we observed that the training for saturation usually converges faster than for pressure. The optimal hyperparameter values can usually be found by conducting grid search or random search [45] over the specified value ranges. In our numerical experiments, we found the training to not be very sensitive to hyperparameter values after appropriate data preprocessing. Therefore, the same set of hyperparameters can be used as the initial setup for a new training set.

In this study, we use a training sample size of 1500, which means we have 1500 random channelized permeability fields (generated by CNN-PCA) and corresponding sequences of state maps (generated by the numerical simulator). Although more training data will usually lead to higher prediction accuracy, it also corresponds to higher preprocessing cost, so there is a tradeoff between these two objectives. As noted earlier, not counting the 1500 training simulations, it takes about 80 minutes to train each of the recurrent R-U-Nets applied in this study. Since we train networks (separately) for pressure

and saturation, this corresponds to a total of 160 minutes of training. These trainings can however be performed in parallel when resources are available, in which case elapsed time is only 80 minutes.

The way in which the training time scales with problem size, number of training runs, number of time steps considered, etc., is important for practical applications. This is a complicated issue because some aspects of the training are expected to scale linearly with problem size, while others scale sub-linearly (since the number of training parameters will stay the same). These scalings should be investigated and quantified in future work. We do expect, however, that with larger models training time will continue to be small compared to the time required for data assimilation using full numerical simulation.

3.4. Saturation and pressure map predictions

It is important to quantify the relative error in the surrogate model predictions for saturation and pressure. The relative error in saturation at time step t, denoted δ_S^t, for the full set of $n_e = 500$ test samples, is given by

$$\delta_S^t = \frac{1}{n_e n_b} \sum_{i=1}^{n_e} \sum_{j=1}^{n_b} \frac{\left\| (\hat{S}_w)_{i,j}^t - (S_w)_{i,j}^t \right\|}{(S_w)_{i,j}^t}, \tag{13}$$

where $(\hat{S}_w)_{i,j}^t$ and $(S_w)_{i,j}^t$ denote the saturation value provided by the surrogate model and the simulator for test sample i, in grid block j, at time step t. Here we have $n_b = 80 \times 80 = 6400$ grid blocks. The initial water saturation is 0.1, which means $(S_w)_{i,j}^t \geq 0.1$ (so the denominator is well behaved). The overall relative saturation error over $n_t = 10$ time steps, denoted δ_S, is then given by

$$\delta_S = \frac{1}{n_t} \sum_{t=1}^{n_t} \delta_S^t. \tag{14}$$

The relative pressure error at time step t, δ_p^t, is given by

$$\delta_p^t = \frac{1}{n_e n_b} \sum_{i=1}^{n_e} \sum_{j=1}^{n_b} \frac{\left\| \hat{p}_{i,j}^t - p_{i,j}^t \right\|}{p_{i,\max}^t - p_{i,\min}^t}, \tag{15}$$

where $\hat{p}_{i,j}^t$ and $p_{i,j}^t$ are defined analogously to $(\hat{S}_w)_{i,j}^t$ and $(S_w)_{i,j}^t$. The difference between the maximum grid-block pressure $p_{i,\max}^t$ and the minimum grid-block pressure $p_{i,\min}^t$, for sample i at time step t, is used to normalize the pressure error. The relative pressure error over $n_t = 10$ time steps, δ_p, is then given by

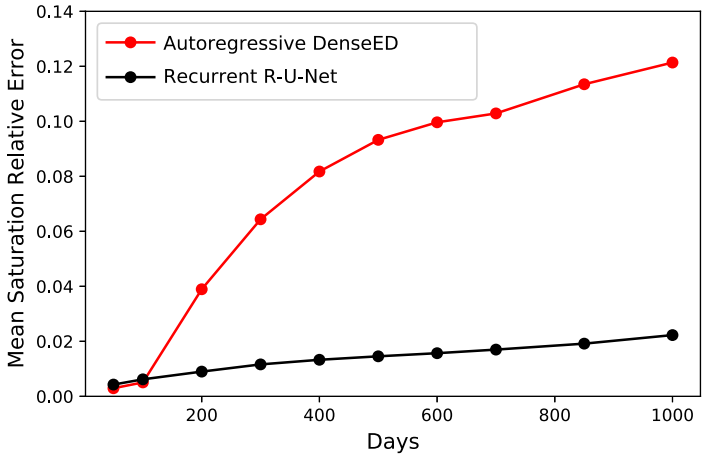
$$\delta_p = \frac{1}{n_t} \sum_{t=1}^{n_t} \delta_p^t. \tag{16}$$

Using these error measures, we now compare the performance of the proposed surrogate model with that of an existing approach – the autoregressive DenseED framework presented in [20]. The problem setup for both models is as described in Section 3.1. The pressure normalization presented in Section 2.6 is used for both models. To simplify the comparison, we eliminate the additional weighting for well block quantities; i.e., we set $\lambda = 0$ in Eq. (9). In addition, for consistency, models are trained separately for saturation and pressure with both approaches. A batch size of 8 and a learning rate decay strategy are applied with both methods until convergence is achieved.

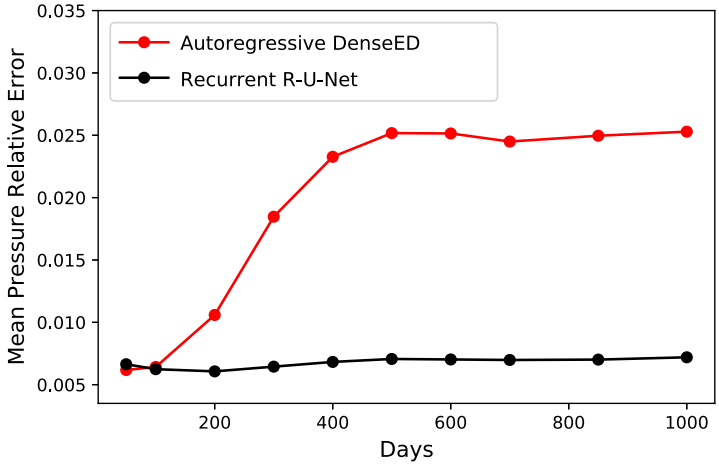
The geomodels in the test set are new (random) CNN-PCA realizations. Numerical simulation is applied to produce the reference state maps. The mean saturation and pressure relative errors for each time step are computed using Eqs. (13) and (15) over 500 test cases. Error results for both quantities are shown in Fig. 8. It is evident that errors are generally lower, and that they grow less quickly in time, using our recurrent R-U-Net than with the autoregressive DenseED procedure. Analogous behavior is also observed in Appendix B, where a four-well system is considered.

We now evaluate the performance of the recurrent R-U-Net trained with the additional weighting loss on well states, as defined in Eq. (9). The same set of 500 test cases is considered. The recurrent R-U-Net state maps are generated at 10 time steps, though here we present results at just three (representative) time steps, 50, 400 and 850 days, for a single realization. The realization considered here is shown in Fig. 7a. This particular geomodel provides saturation and pressure results that correspond to relative errors δ_S and δ_p that are slightly larger than the average relative error over the 500 test cases.

The saturation and pressure predictions provided by the recurrent R-U-Net, for the geomodel in Fig. 7a, are displayed in Figs. 9 and 10. In both plots the top row shows the recurrent R-U-Net surrogate model predictions, the middle row displays the high-fidelity simulation results, and the bottom row shows difference maps between the numerical simulation and surrogate results. It is evident in Fig. 9 that the progression of the saturation field with time is strongly impacted by the channelized permeability field, with transport occurring along the high-permeability channels. This type of saturation



(a) Saturation error



(b) Pressure error

Fig. 8. Relative saturation error δ_S^t (Eq. (13)) and pressure error δ_p^t (Eq. (15)) at different time steps using the recurrent R-U-Net procedure and an existing framework (autoregressive DenseED [20]).

distribution is quite different from that typically observed with multi-Gaussian permeability fields. We see that the recurrent R-U-Net provides generally accurate predictions for the saturation field at the three times. Small errors are, however, noticeable near saturation fronts. The pressure results in Fig. 10 similarly demonstrate the high degree of accuracy of the recurrent R-U-Net for this case. The relative error of the surrogate model for pressure is small, though unlike saturation error, pressure error is not confined to fluid fronts.

Evaluated over the 500 random test samples, we find time-averaged relative errors $\delta_S = 2.8\%$ (using Eq. (14)) and $\delta_p = 1.1\%$ (Eq. (16)). These low error values indicate a high degree of accuracy in the surrogate dynamic saturation and pressure fields. We note that the corresponding relative saturation and pressure errors for the geomodel in Fig. 7a (over all 10 time steps) are 3.1% and 1.2% respectively. Thus the results in Figs. 9 and 10 can be considered to be representative in terms of surrogate model accuracy.

As noted earlier, we could also use a single recurrent R-U-Net rather than one for pressure and one for saturation. With such a setup, using equal weights for the L^1 pressure loss and the L^2 saturation loss, and setting $\lambda = 0$ in Eq. (9), the relative errors over the 500 test samples are $\delta_S = 3.1\%$ and $\delta_p = 0.6\%$. Thus this approach is clearly viable. The pressure error given above (1.1%) is actually larger than 0.6% because the use of nonzero λ in Eq. (9) provides decreased well-block error at the expense of slightly increased global error. These single net errors could likely be improved slightly with additional tuning.

3.5. Well flow rate predictions

Well production and injection rates are often key quantities of interest, and data of this type are commonly assimilated in history matching studies. We now assess the accuracy of our recurrent R-U-Net for well-rate data. This entails the

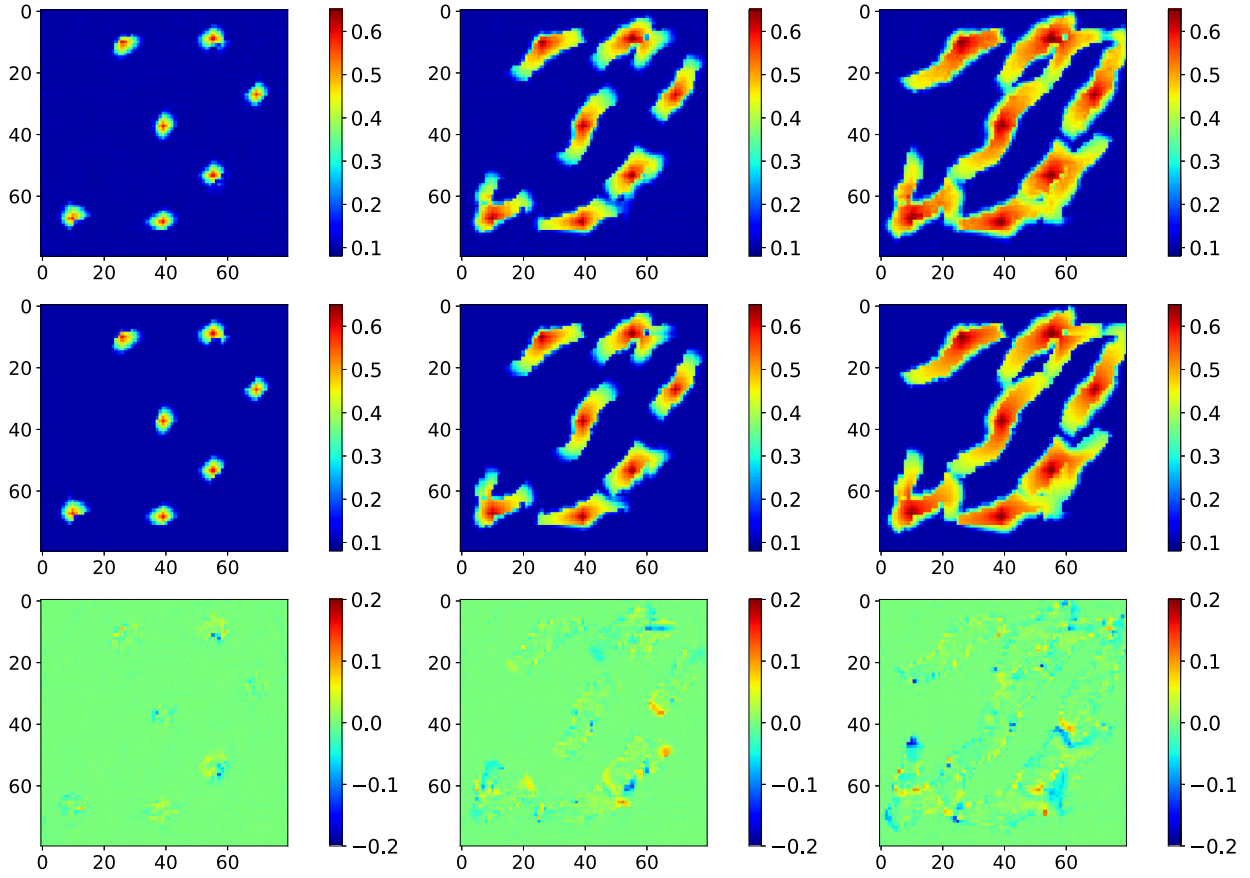


Fig. 9. Saturation maps from recurrent R-U-Net surrogate model (top row) and high-fidelity simulator (middle row), along with difference maps (bottom row), at three different times.

application of Eq. (2), which in turn requires the R-U-Net estimates for p and S_w in well blocks, along with the well index given in Eq. (3).

Fig. 11 displays comparisons of oil and water production rates for the geomodel considered in Section 3.4 (shown in Fig. 7a). Well locations are as indicated on Fig. 5. Well rate results in Fig. 11 are presented for the high-fidelity simulation run (black curves, designated 'sim' in the figure legends) and for the recurrent R-U-Net predictions (red curves, designated 'surr'), for three production wells. These rates are actually computed at only 10 particular times, but the results are presented as continuous curves. These results demonstrate that the surrogate model provides a high degree of accuracy in well flow rates, which is essential if this model is to be used for history matching. There are, however, small but noticeable discrepancies in water rate at late time in wells P3 and P7.

In order to concisely evaluate recurrent R-U-Net performance for the full set of 500 test cases, we now present results, in terms of flow statistics, for the full ensemble. Specifically, for a given well (or for the entire field), we order the 500 oil and water production rate results at each time step, and then select the result corresponding to the 10th, 50th, and 90th percentile for each quantity. These are referred to as the P_{10}, P_{50} and P_{90} responses. This type of assessment was also used in the evaluation of CNN-PCA for geomodel generation – see [42] for details.

Fig. 12 displays the P_{10}, P_{50} and P_{90} flow responses, for oil and water production rate, obtained from the surrogate model (red curves) and from numerical simulation (black curves). Oil rate results for production wells P1, P2 and P17 are shown, while water rates for wells P1, P15 and P17 are displayed. We consider P2 for oil and P15 for water in order to display a range of behaviors. The P_{50} results are shown as solid curves, while the P_{10} and P_{90} responses are shown as the (lower and upper) dashed curves. Note that the model corresponding to the P_{50} result (or the P_{10} or P_{90} result) can differ from time step to time step. Agreement is clearly very close between the two sets of results.

Note that, for well P2, the oil rates evident in Fig. 12c are very low compared to those for the other wells. This is the case because this well is located in a low-permeability (mud) region, as is evident from the well locations shown in Fig. 5. For this well even the P_{90} result corresponds to zero water production, so we do not show well P2 water rate results. We

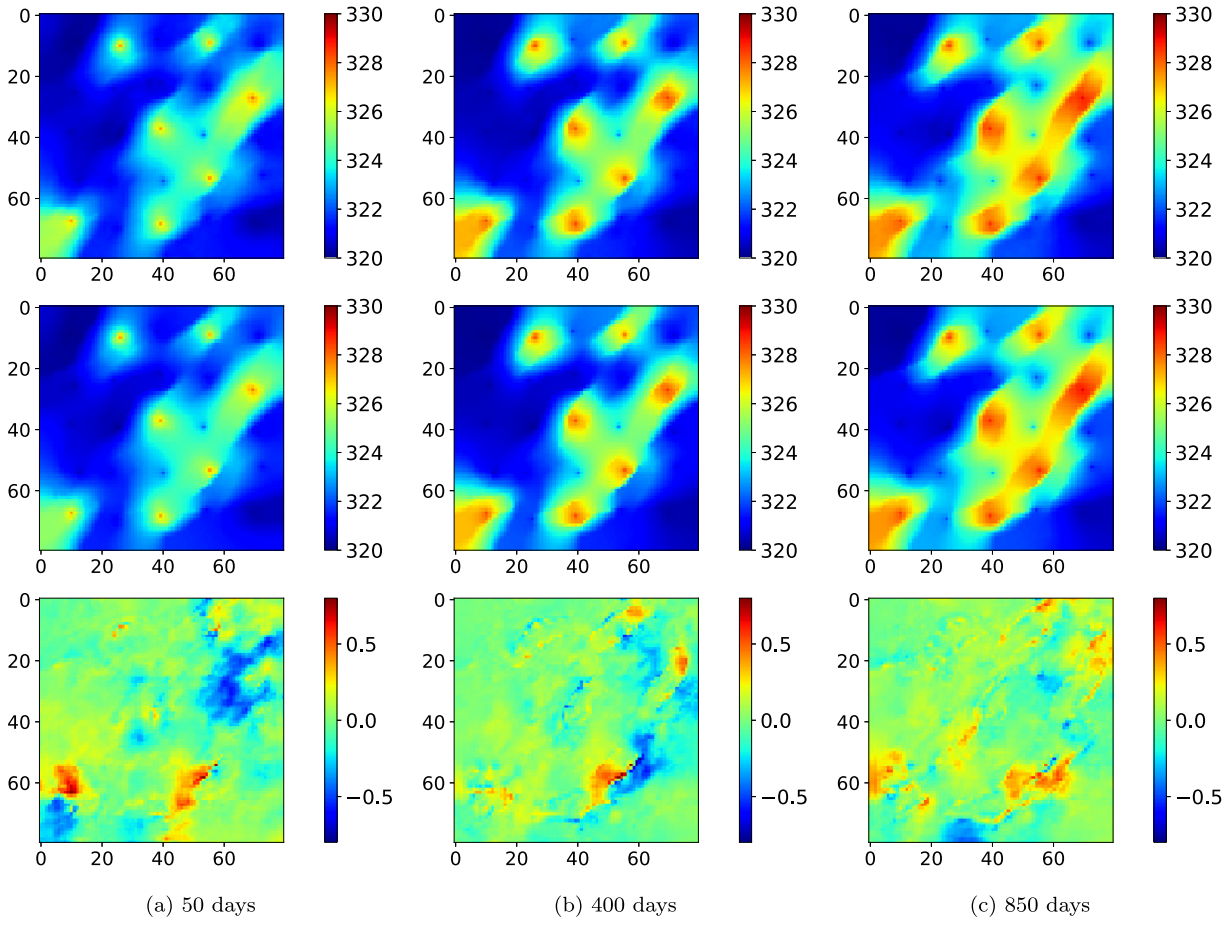


Fig. 10. Pressure maps from recurrent R-U-Net surrogate model (top row) and high-fidelity simulator (middle row), along with difference maps (bottom row), at three different times.

instead show the water production statistics for well P15. For this well it is apparent from Fig. 12d that more than half of the realizations do not produce any water over the entire 1000-day simulation time frame. It is encouraging to see that the recurrent R-U-Net results are consistent with those from the simulator even in these more extreme situations.

Although the well rate results presented thus far have been for production wells, it is important to demonstrate that the recurrent R-U-Net results for water injection are also accurate. Comparisons of water injection rate statistics between the surrogate model and the high-fidelity simulator, for well I1 and for the entire field, are shown in Fig. 13. There we see very close agreement between the two sets of results, which demonstrates that R-U-Net predictions for water injection rate are indeed accurate.

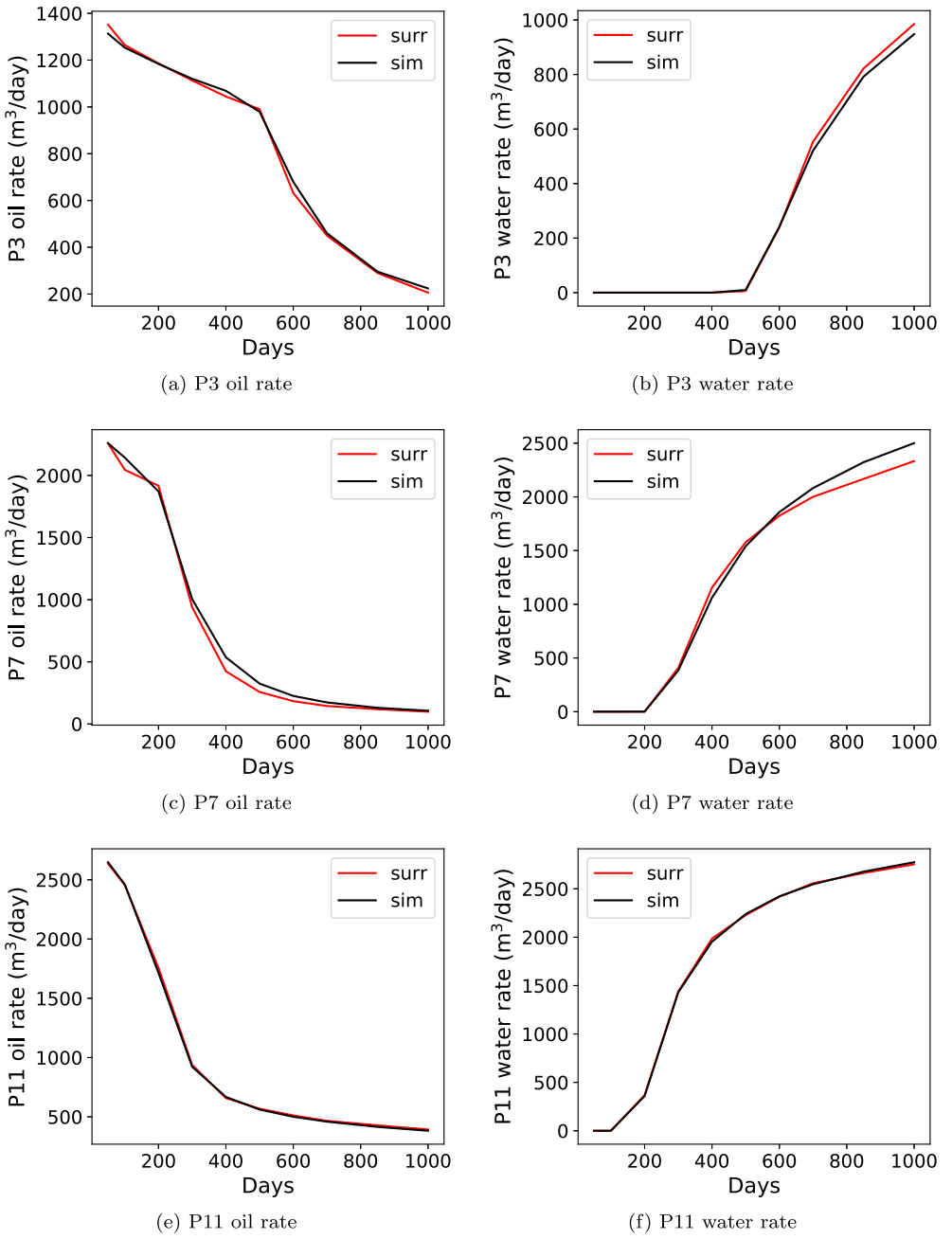
Finally, we quantify the average relative errors in oil and water production rates, and water injection rate, over all production and injection wells, respectively. For oil and water production rates, the field-average relative error at time step t ($\delta^t_{r,j}$), and the overall relative error ($\delta_{r,j}$), are given by

$$\delta^t_{r,j} = \frac{1}{n_e n_p} \sum_{i=1}^{n_e} \sum_{k=1}^{n_p} \frac{\left\| (\hat{r}_j)^t_{i,k} - (r_j)^t_{i,k} \right\|}{(r_j)^t_{i,k} + \epsilon}, \tag{17}$$

and

$$\delta_{r,j} = \frac{1}{n_t} \sum_{t=1}^{n_t} \delta^t_{r,j}, \tag{18}$$

where $(\hat{r}_j)^t_{i,k}$ and $(r_j)^t_{i,k}$ denote the phase ($j = o$ for oil and w for water) production rate from the surrogate model and the simulator for well k at time step t in test sample i. To avoid division by zero, a constant $\epsilon = 1$ is introduced in the denominator. Here n_p is the number of production wells. For injection rate relative error, we replace n_p by n_i, the number



(a) P3 oil rate (b) P3 water rate

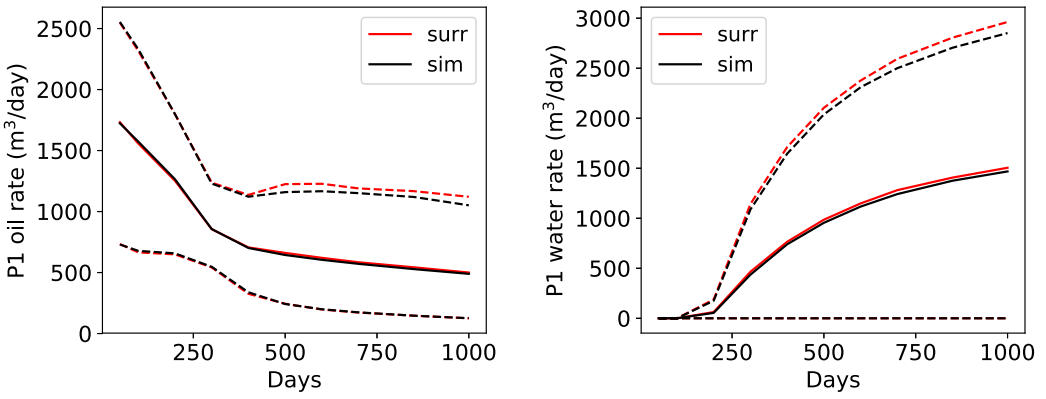
(c) P7 oil rate (d) P7 water rate

(e) P11 oil rate (f) P11 water rate

Fig. 11. Comparison of oil (left) and water (right) production rates, for three different wells, for the geomodel considered in Section 3.4. Red and black curves represent results from the recurrent R-U-Net surrogate model and the high-fidelity simulator, respectively. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

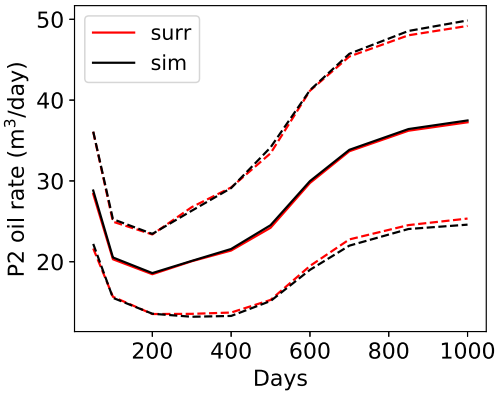
of injection wells, and compute $\delta_{r,inj}$. Note that there is no error cancellation from time step to time step in Eq. (18). The relative errors for oil and water production rates using Eq. (18) are $\delta_{r,o} = 6.4\%$ and $\delta_{r,w} = 5.8\%$, while the relative error for water injection rate is $\delta_{r,inj} = 3.5\%$. For the particular geomodel in Fig. 7a, these relative errors are comparable but slightly smaller (5.7%, 5.4%, 3.2% for oil production, water production and water injection rate, respectively).

Finally, we provide an error comparison, for a key well rate quantity, between our recurrent R-U-Net and the autoregressive DenseED model [20]. Fig. 14 displays the field-average relative error in water production rate (computed using Eq. (17)) for the two methods. For this comparison, both surrogate models are trained with $\lambda = 0$ (thus the errors for the recurrent R-U-Net in Fig. 14 are slightly higher than in the results reported above). It is evident that the recurrent R-U-Net provides more accurate well responses. This will be important in history matching applications, which we consider in the next section.

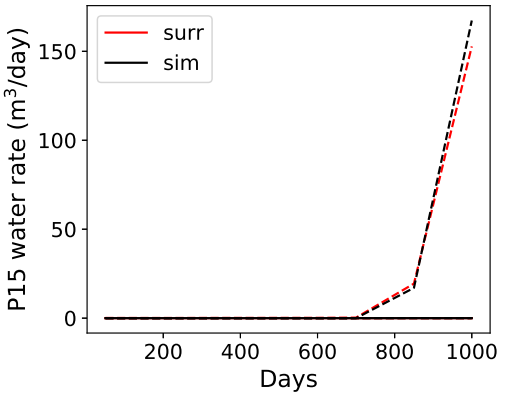


(a) P1 oil rate

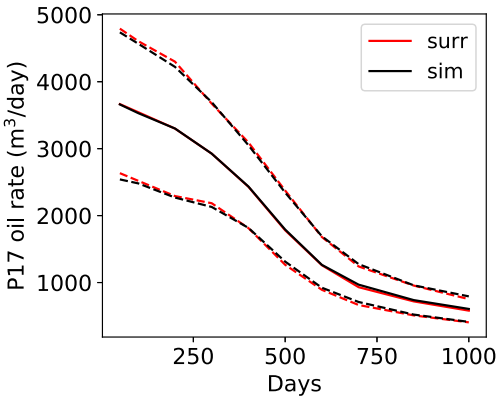
(b) P1 water rate



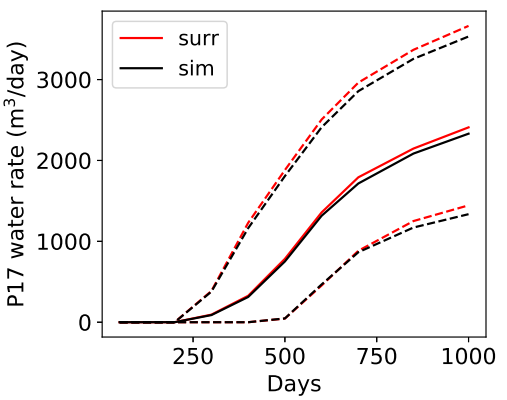
(c) P2 oil rate



(d) P15 water rate



(e) P17 oil rate

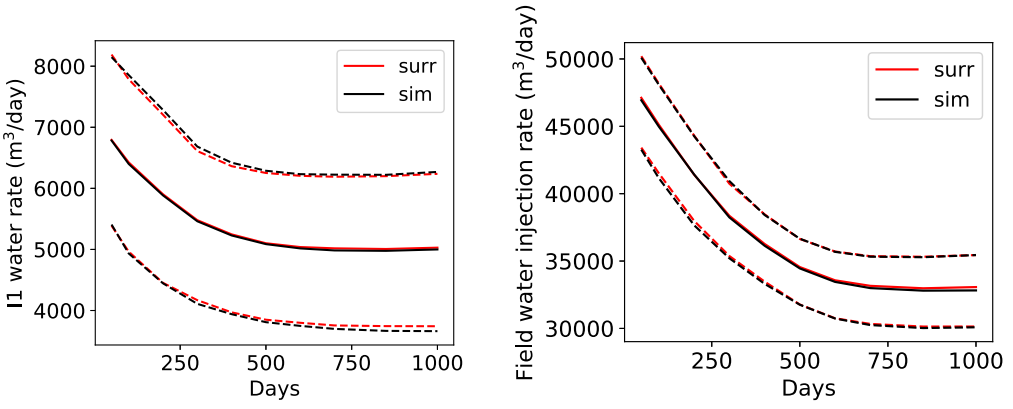


(f) P17 water rate

Fig. 12. Comparison of oil (left) and water (right) production rate statistics, for different wells, over the full ensemble of 500 test cases. Red and black curves represent results from the recurrent R-U-Net surrogate model and the high-fidelity simulator, respectively. Solid curves correspond to P_{50} results, lower and upper dashed curves to P_{10} and P_{90} results. P_{10}, P_{50} and P_{90} water production rates for well P2 are all zero (thus results are shown for well P15 instead).

4. History matching using recurrent R-U-Net surrogate model

The applicability of our surrogate model for estimating dynamic state evolution and well rates was demonstrated in Section 3. We now apply the method for a challenging history matching problem. Comparisons to a simulation-based procedure are also presented.



(a) I1 water injection rate (b) Field water injection rate

Fig. 13. Comparison of water injection rate statistics for well I1 (left) and for the entire field (right) over the full ensemble of 500 test cases. Red and black curves represent results from the recurrent R-U-Net surrogate model and the high-fidelity simulator, respectively. Solid curves correspond to P_{50} results, lower and upper dashed curves to P_{10} and P_{90} results.

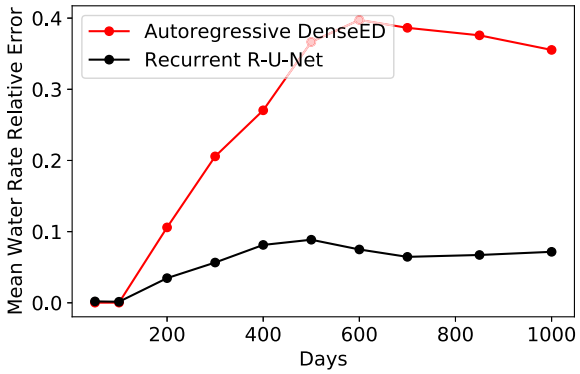


Fig. 14. Relative field-average water rate error $\delta^t_{r,w}$ (Eq. (17)) using the recurrent R-U-Net and autoregressive DenseED [20].

4.1. History matching procedure

In this study, the randomized maximum likelihood (RML) procedure is applied to generate posterior geomodels. The RML method, originally developed in [46] and [47], has been widely used both with high-dimensional geomodels (e.g., [48]) and parameterized models (e.g., [49,42]). Here we apply it with a CNN-PCA parameterization, so our treatment is analogous to that described in [42]. However, in place of high-fidelity simulations, recurrent R-U-Net predictions are used. This means we (1) represent a geomodel $\mathbf{m}$ as $\mathbf{m} \approx \mathbf{m}_{cnn}(\boldsymbol{\xi})$, where $\boldsymbol{\xi} \in \mathbb{R}^{n_\xi}$ is the low-dimensional (parameterization) variable, and (2) model the flow response as $f \approx \hat{f}(\mathbf{m}_{cnn}(\boldsymbol{\xi}))$, where the use of $\hat{f}$ (rather than f) means the recurrent R-U-Net is used in place of the numerical simulator.

RML is an optimization-based procedure in which a minimization problem is solved repeatedly to generate multiple (N_r) posterior samples. In our setting, each run provides a posterior sample designated $\boldsymbol{\xi}_{i,rml}$. The minimization problem is expressed as:

$$\begin{aligned}\boldsymbol{\xi}_{i,rml} = \underset{\boldsymbol{\xi}_i}{\text{argmin}}[&(\hat{f}(\mathbf{m}_{cnn}(\boldsymbol{\xi}_i)) - \mathbf{d}^*_{i,obs})^\intercal C_D^{-1} (\hat{f}(\mathbf{m}_{cnn}(\boldsymbol{\xi}_i)) - \mathbf{d}^*_{i,obs}) \\ &+ (\boldsymbol{\xi}_i - \boldsymbol{\xi}^*_i)^\intercal (\boldsymbol{\xi}_i - \boldsymbol{\xi}^*_i)], \quad i = 1, \ldots, N_r.\end{aligned} \tag{19}$$

Here $\mathbf{d}_{obs}$ denotes the observed data and C_D is the covariance of the data measurement error. The superscript $*$ indicates that the quantity is sampled from a normal distribution. Specifically, the (perturbed) observation data $\mathbf{d}^*_{i,obs}$ is sampled from $\mathcal{N}(\mathbf{d}_{obs}, C_D)$, and $\boldsymbol{\xi}^*_i$ is sampled from $\mathcal{N}(0, I)$. The observed and simulated data in our case include oil and water production rates and water injection rates. Though not considered here, time-lapse seismic data are available in some cases, and these data can be used to infer an approximate global saturation field. In this situation, the global saturation field provided by the recurrent R-U-Net would also enter into the formulation. We note finally that the $(\boldsymbol{\xi}_i - \boldsymbol{\xi}^*_i)^\intercal (\boldsymbol{\xi}_i - \boldsymbol{\xi}^*_i)$ term on the right-hand side of Eq. (19) acts as a regularization. The use of the CNN-PCA parameterization assures that posterior realizations are consistent (in a geostatistical sense) with prior realizations.

The minimization in Eq. (19) can be performed using various algorithms. Here we apply mesh adaptive direct search (MADS), described in [50]. MADS is a local-search (derivative-free) procedure that evaluates trial points determined by an underlying stencil. This stencil is centered around the current-best solution. Each iteration (in the MADS variant used here) entails $2l$ function evaluations, where l denotes the number of optimization variables. In our case, $l = n_\xi = 100$. MADS applies a set of strategies when no improvement is achieved with the current stencil. Convergence to a local minimum is guaranteed if particular problem criteria are satisfied. We note finally that a range of optimizers could be used for the minimization in Eq. (19).

4.2. Problem setup

The system considered here is consistent with that used in Section 3. We again have a 2D channelized model defined on an 80×80 grid. The model contains 25 wells (18 producers and seven injectors, 20 wells are in sand and five are in mud). All realizations are conditioned to facies-type at well locations. The 'true' model, which is a (new) random SGeMS realization generated from the training image used in the construction of the CNN-PCA representation, is shown in Fig. 15a.

Oil-water flow is again considered, with fluid and rock-fluid properties as described in Section 3. The total simulation time frame is 1000 days. The first 400 days are prescribed to be the history matching period (during which data are collected), and the following 600 days are the forecast period. Oil and water production rates for the 18 producers, and water injection rates for the seven injectors, at five time steps, comprise the observed data (there are thus $18 \times 5 \times 2 + 7 \times 5 = 215$ measurements to be matched). Random Gaussian noise, consistent with C_D, is added to the simulated flow response for the 'true' model to provide the observed data vector. The mean and standard deviation of the random Gaussian noise are set to zero and 5% of the corresponding true data, respectively.

In this work we generate $N_r = 100$ posterior models. In each RML run, 200 MADS iterations are performed, and at each iteration 200 function evaluations are required. Consequently, the overall history matching procedure entails $100 \times 200 \times 200 = 4 \times 10^6$ flow model evaluations. The average runtime for the numerical simulator for a single flow problem is around 10 seconds on a single CPU, while that of the deep-learning-based surrogate model is about 0.01 seconds (for batch predictions on a single GPU). We thus expect the surrogate model to reduce the total computation from about 4×10^7 seconds to 40,000 seconds. In Section 4.4 below, we apply simulation-based history matching for this case using the same RML and MADS procedures. There we will see that simulation-based history matching actually requires considerably more than 4×10^7 seconds, for reasons discussed in Section 4.4. This indicates that greater than $1000\times$ speedup is achieved using the surrogate model for this problem.

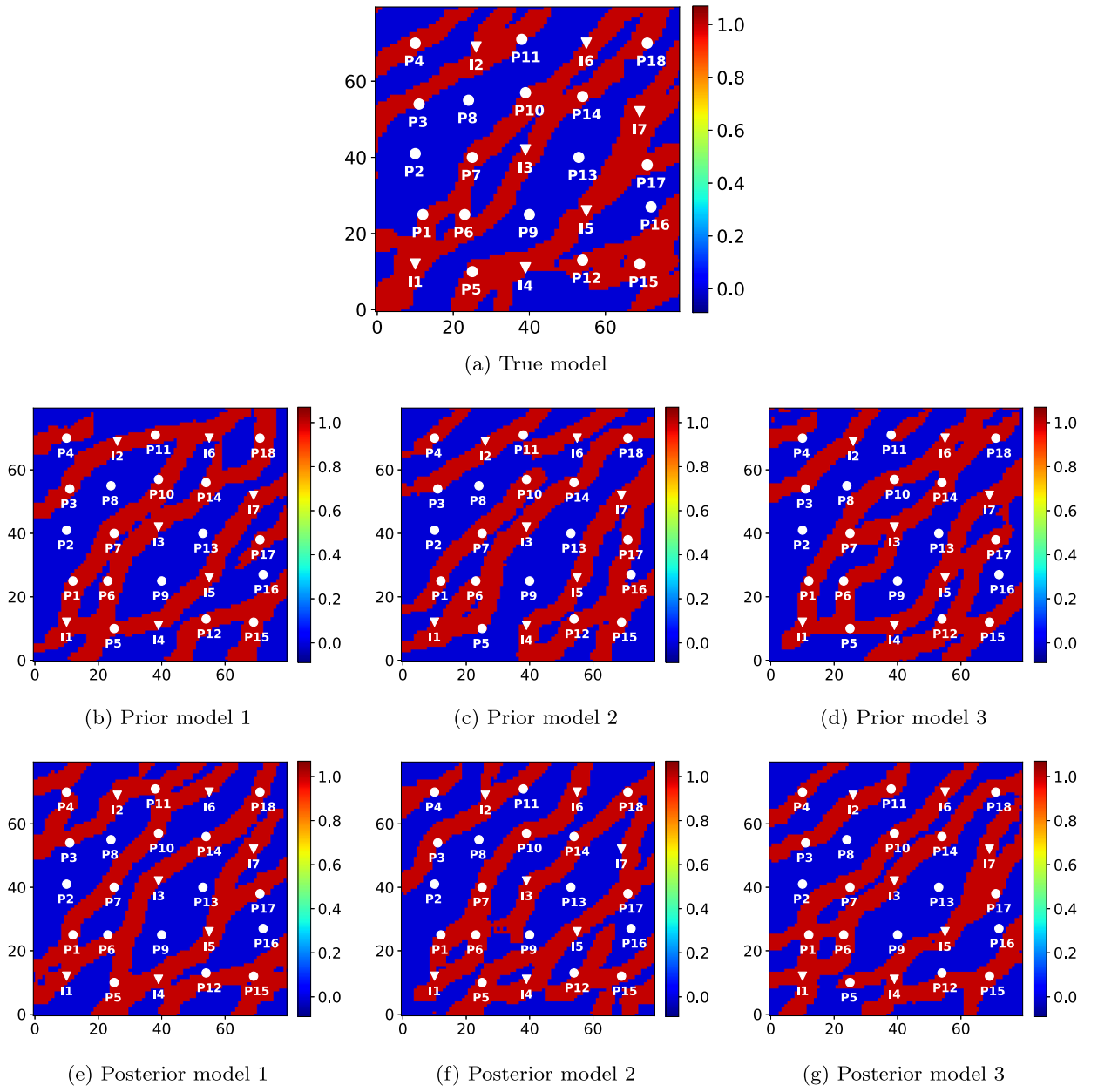
It is important to emphasize that these timings are for data assimilation using RML and MADS, with $N_r = 100$, 200 MADS iterations, and $l = n_\xi = 100$. The use of ensemble-based history matching approaches will reduce these timings dramatically. Savings could also be achieved by reducing the values of N_r and l, or by using a gradient-based optimization procedure. However, one advantage of the surrogate model in this setting is that it may enable the use of history matching algorithms that would otherwise be extremely demanding computationally.

4.3. History matching results

We now present history matching results using the recurrent R-U-Net as a surrogate for the flow simulator. Three randomly selected prior models are shown in Fig. 15b, c and d. These models are used as initial guesses in the subspace RML procedure (Eq. (19)), and the corresponding posterior models appear in Fig. 15e, f and g. Both the prior and posterior models clearly resemble the true model (Fig. 15a) in terms of their general channelized appearance. This is because CNN-PCA is able to maintain the channelized structure apparent in the original training image.

There are, however, important differences between the prior and posterior models in terms of how particular sets of wells are connected (or not connected) through sand. For example, in the true model, injector I1 and producer P5 (both are in the lower left portion of the model) are not connected through sand, while in all three prior models they are connected. In the three posterior models, wells I1 and P5 are (correctly) not connected via sand. In addition, wells I2 and P11 are connected through sand in the true model, while in prior model 3 (Fig. 15d) they are not connected. In the corresponding posterior model (Fig. 15g) they are properly connected. Sand connections are also established between wells I6 and P10, and between wells I4 and P5, in posterior model 2 (compare Fig. 15c and f).

Fig. 16 displays results for oil and water production rates for producers P1, P14 and P17. In these figures, the gray region displays the P_{10}–P_{90} interval for the prior models, the red curve denotes the true model flow response, the red circles indicate the observed data (generated by perturbing from the true model response, as described in Section 4.2), and the blue dashed curves depict the P_{10}, P_{50} and P_{90} posterior results generated using RML with the recurrent R-U-Net surrogate model. The vertical black dashed line at 400 days divides the simulation time frame into the history matching period (to the left of the line) and forecast or prediction period. It is important to note that, despite the relatively large number of wells (and thus conditioning data) in this problem, there is still very substantial well-by-well variation in production quantities, as indicated by the gray P_{10}–P_{90} prior intervals in Fig. 16. For example, for well P1, we see in Fig. 16b that more than 10% of the realizations show no water breakthrough over the full 1000-day period, while more than 10% of the models provide water rates of about 2800 m^3/day or higher at 1000 days. High levels of variability are also observed in many other well-by-well quantities.



(a) True model

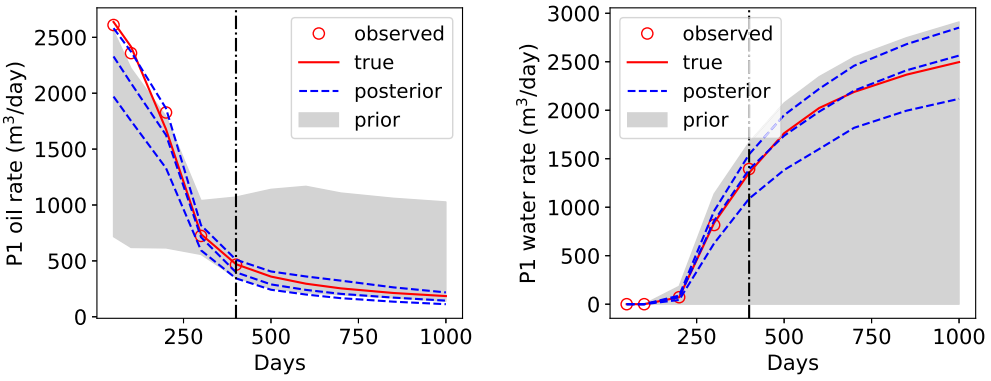
(b) Prior model 1 (c) Prior model 2 (d) Prior model 3

(e) Posterior model 1 (f) Posterior model 2 (g) Posterior model 3

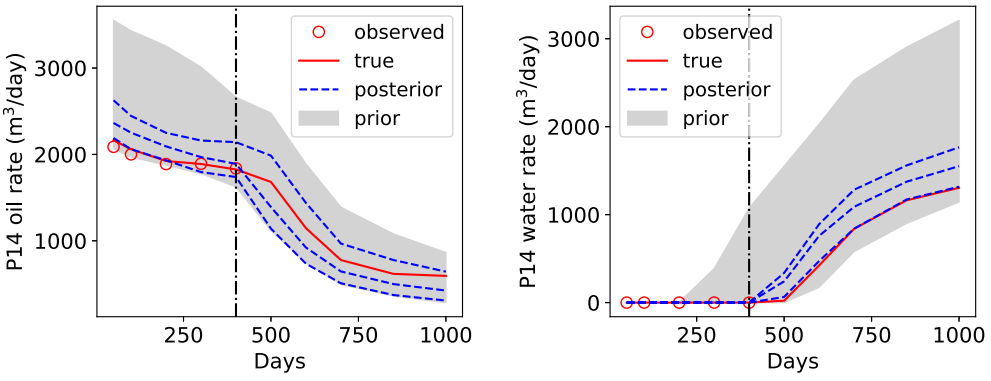
Fig. 15. Binary facies models for the 2D channelized system. All models conditioned to facies type at the 25 wells. True SGeMS model, along with three prior and (corresponding) posterior CNN-PCA models are shown. White circles denote production wells and white triangles denote injection wells.

It is clearly evident that the history matching procedure leads to a significant reduction in uncertainty; i.e., the posterior P_{10}–P_{90} ranges are overall much smaller than the prior (gray) P_{10}–P_{90} ranges. The posterior P_{10}–P_{90} interval generally captures the observed (and true) data, even when these data fall toward the edge of the prior P_{10}–P_{90} interval. This is evident in all three oil rate plots in Fig. 16. It is also interesting to note that, even when water breakthrough has not yet occurred in a particular well during the history match period (e.g., well P14 in Fig. 16d), there is still significant uncertainty reduction in the water rate prediction.

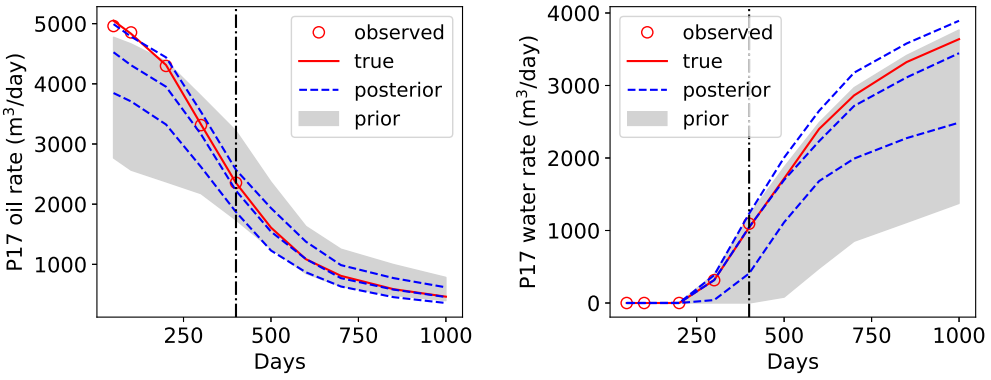
The inversion procedure provides 100 posterior geomodels, and all of the posterior flow responses for these models (in Fig. 16) were generated using the recurrent R-U-Net surrogate model. It is therefore reasonable to verify the predicted flow responses for these models by simulating them using the numerical simulator. The results of such an assessment are shown in Fig. 17. The black and red solid curves denote the P_{50} flow response using the numerical simulator and the surrogate



(a) P1 oil rate (b) P1 water rate



(c) P14 oil rate (d) P14 water rate

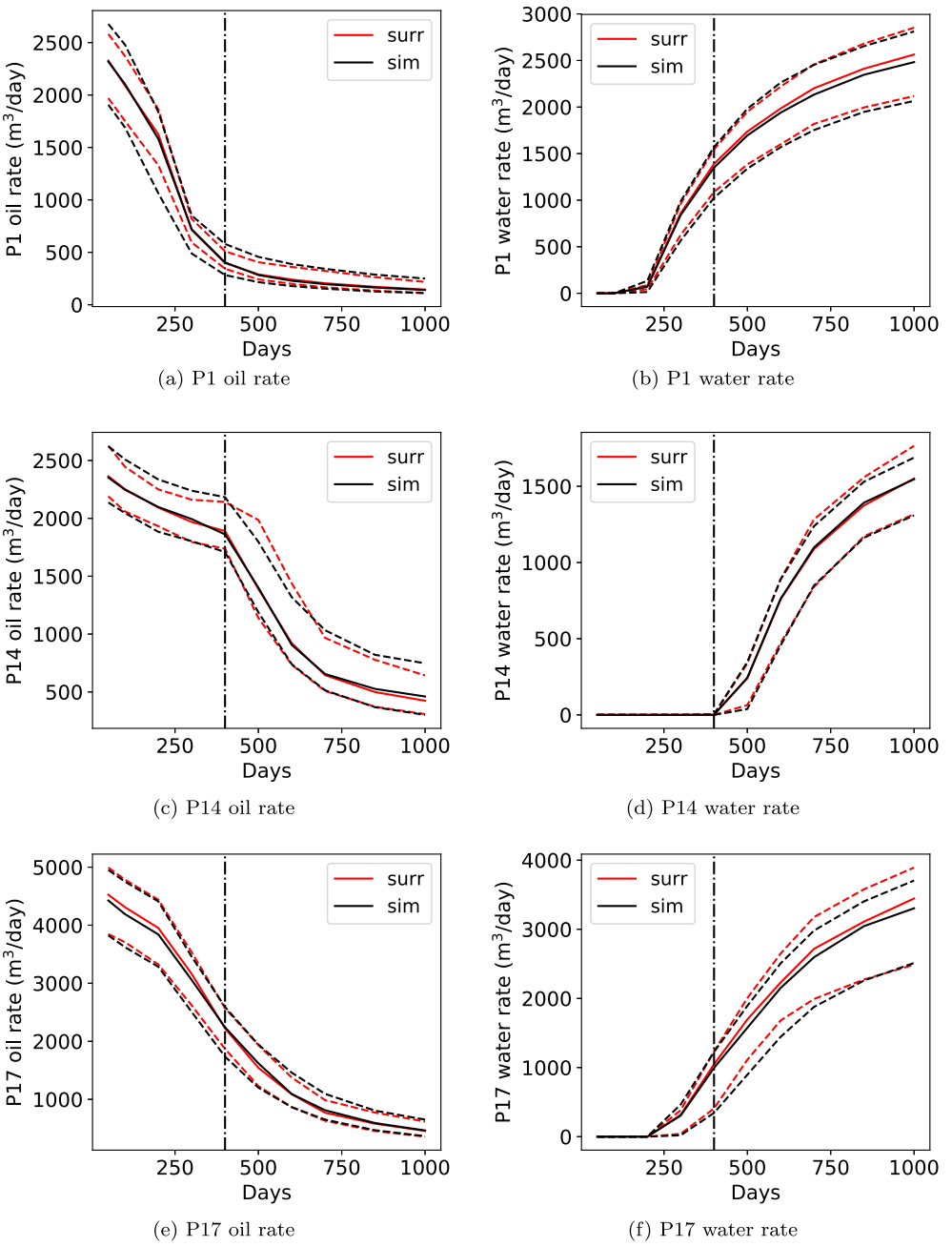


(e) P17 oil rate (f) P17 water rate

Fig. 16. Oil (left) and water (right) production for producers P1, P14 and P17. Gray regions represent the prior P_{10}–P_{90} range, red points and red lines denote observed and true data, and blue dashed curves denote the posterior P_{10} (lower), P_{50} (middle) and P_{90} (upper) predictions. Vertical dashed line divides simulation time frame into history match and prediction periods.

model, respectively. The black and red dashed curves depict the P_{10} and P_{90} flow responses from the simulator and the surrogate model. The red (surrogate model) curves in Fig. 17 are the same as the blue curves in Fig. 16, but the scales in some plots differ because the prior P_{10}–P_{90} ranges are not shown in Fig. 17.

From Fig. 17, we see that there is generally very close correspondence between the numerical simulation and recurrent R-U-Net P_{10}, P_{50}, P_{90} posterior results. Agreement in the P_{50} curves is consistently close, though small discrepancies are observed in some of the P_{10} and P_{90} curves (e.g., P_{90} curve in Fig. 17c and P_{10} and P_{90} curves in Fig. 17f). These errors are, however, very small compared to the amount of uncertainty reduction achieved by the inversion procedure.



(a) P1 oil rate (b) P1 water rate

(c) P14 oil rate (d) P14 water rate

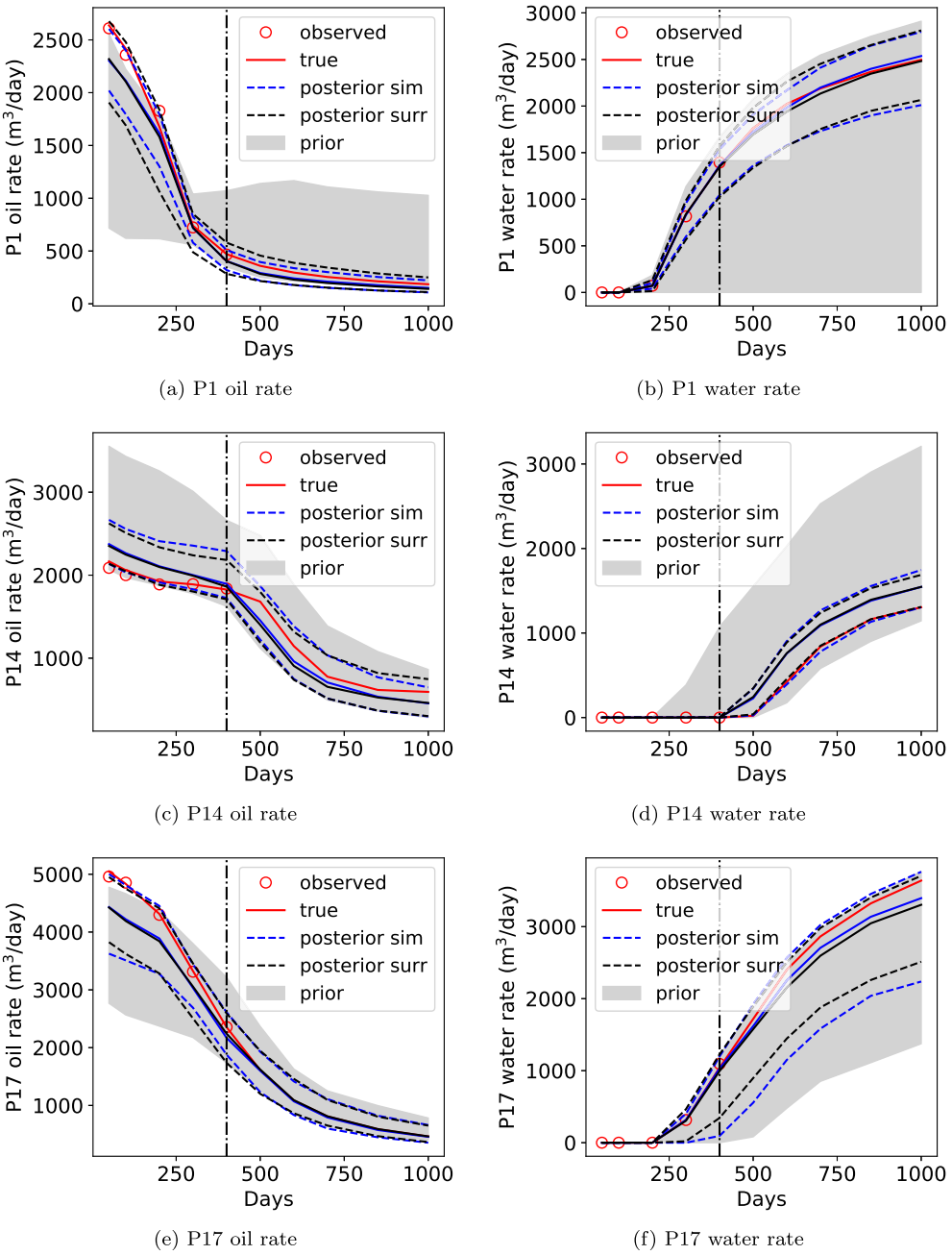
(e) P17 oil rate (f) P17 water rate

Fig. 17. Comparison of oil (left) and water (right) posterior production forecasts generated by the numerical simulator (black curves) and recurrent R-U-Net surrogate model (red curves). Solid curves correspond to P_{50} response, and dashed curves to P_{10} (lower curves) and P_{90} (upper curves) responses.

4.4. Comparison with simulation-based history matching

As a final verification of the applicability of the recurrent R-U-Net surrogate model for history matching, we compare surrogate results to those achieved using numerical simulation for all function evaluations. The same set of prior models is used for consistency, and the same RML and MADS procedures are applied to generate posterior models. This simulation-based history matching is a computationally intensive task, as discussed below.

Fig. 18 presents the comparison between history matching results using the surrogate model and the simulator. The blue lines (solid and dashed) correspond to P_{10}, P_{50}, P_{90} posterior simulation-based history matching results, while the black lines denote the corresponding surrogate results. We see that the recurrent R-U-Net surrogate model indeed provides history matching results in generally close agreement with the simulation-based results. The P_{50} results are consistently



(a) P1 oil rate (b) P1 water rate

(c) P14 oil rate (d) P14 water rate

(e) P17 oil rate (f) P17 water rate

Fig. 18. Comparison of oil (left) and water (right) production for producers P1, P14 and P17 using surrogate-based and simulation-based history matching. Gray regions represent the prior P_{10}–P_{90} range, red points and red lines denote observed and true data, blue and black curves denote the posterior P_{10} (lower), P_{50} (middle) and P_{90} (upper) predictions obtained using the numerical simulator (blue) and the surrogate (black). Vertical dashed line divides simulation time frame into history match and prediction periods.

accurate, though some small discrepancies are apparent in the P_{10} (Fig. 18a and f) and P_{90} (Fig. 18c) results. These may correspond to statistical fluctuations (since we generate only 100 posterior models), or they may be indicative of some amount of surrogate model inaccuracy.

The simulation-based history matching was achieved here by running the numerical simulator, in parallel, on 200 CPU cores. This procedure required about 180 hours to generate 100 posterior realizations. This corresponds to approximately 1.3×10^8 seconds of computation, which is more than a factor of three times our earlier estimate of 4×10^7 seconds for this simulation-based history matching. The additional time corresponds to communication, reading, writing and deleting simulation files, and idling. The latter is observed when one or more runs during an optimization iteration require more

than the average 10-second runtime. This occurs when time-step cuts are performed to assure simulator convergence. Thus we see that the speedup provided by the surrogate model for this case is very substantial.

5. Concluding remarks

In this work, a deep-learning-based surrogate model was developed to capture temporal dynamics in a high-dimensional nonlinear system, specifically an oil-water subsurface flow model. The surrogate model entails a residual U-Net architecture linked to a convolutional long short term memory recurrent network. We refer to the overall network as a recurrent R-U-Net. The recurrent R-U-Net is trained on numerically simulated dynamic saturation and pressure maps, generated for different geological realizations drawn from a particular geological scenario. In our case 2D channelized models were considered, and 1500 training simulations were performed. The recurrent R-U-Net was trained to predict dynamic saturation and pressure in every grid block at 10 particular time steps. Additional weighting at well blocks was introduced in the loss function in order to accurately capture well flow responses, since these represent the primary data used in history matching.

The recurrent R-U-Net was evaluated for oil-water reservoir simulation problems involving flow through new (test) channelized geological realizations. The ability to predict flow responses for new geomodels is essential if the recurrent R-U-Net is to be used for history matching. Comparisons with an existing surrogate model showed that our new approach leads to less error accumulation in time. Detailed comparison of the dynamic pressure and saturation maps generated by the surrogate model and the reference numerical simulator, for a particular geomodel, demonstrated that the recurrent R-U-Net can accurately predict these high-dimensional state variables. Well flow responses – specifically time-varying oil and water production rates and water injection rates – were also shown to be in close agreement. The surrogate model was then evaluated for a test ensemble of 500 new geomodels. In this assessment the P_{10}, P_{50} and P_{90} flow responses from the recurrent R-U-Net were compared to those from the numerical simulator. The high level of accuracy of the surrogate model in predicting these flow statistics demonstrates its applicability for uncertainty quantification.

The recurrent R-U-Net was next applied for history matching. This is a challenging application area, particularly when channelized (rather than multi-Gaussian) geomodels are considered. Posterior model generation was accomplished using a randomized maximum likelihood (RML) procedure, and geomodels were represented concisely (in terms of 100 parameters) using the recently developed CNN-PCA parameterization. RML is an optimization-based method, and the minimization was accomplished using mesh adaptive direct search. Significant uncertainty reduction was achieved, and the posterior (surrogate-model) predictions for oil and water production rates were shown to be reasonably accurate through comparison to numerical simulation results (for the posterior models). Comparisons between surrogate-based and full simulation-based history matching further demonstrated the applicability of the recurrent R-U-Net surrogate model. The speedup obtained using the surrogate model relative to high-fidelity numerical simulation was dramatic in this example, suggesting that the use of the recurrent R-U-Net may enable the application of more rigorous inverse modeling procedures for realistic problems. This will be considered in future work.

There are many other promising directions for future research in this general area. The surrogate model developed in this work is for 2D problems, and we plan to investigate extensions of the recurrent R-U-Net to 3D systems. The surrogate model can also be extended to treat larger and more complicated systems. Our specific interest is in multiphysics problems involving coupled flow and geomechanics, which entail additional governing equations and output state variables such as surface deformations. Surrogate models can have a very large impact for such problems since high-fidelity numerical simulations are often extremely expensive. Finally, the deep-learning-based surrogate model can potentially be extended to handle varying well control and well location variables. This could be accomplished by encoding these variables as additional input maps during training. If successful, this capability could enable the surrogate model to be used for field development optimization in addition to history matching.

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.

Acknowledgements

We are grateful to the Stanford Smart Fields Consortium and to Stanford–Chevron CoRE for partial funding of this work. We also thank the Stanford Center for Computational Earth & Environmental Science (CEES) for providing the computational resources used in this study.

Appendix A. Model architecture details

A.1. Detailed R-U-Net schematic

The detailed R-U-Net architecture is illustrated in Fig. 19. The boxes correspond to multichannel feature maps and the colored arrows represent different operations. The gray arrow shows the copy and concatenation of the extracted features in

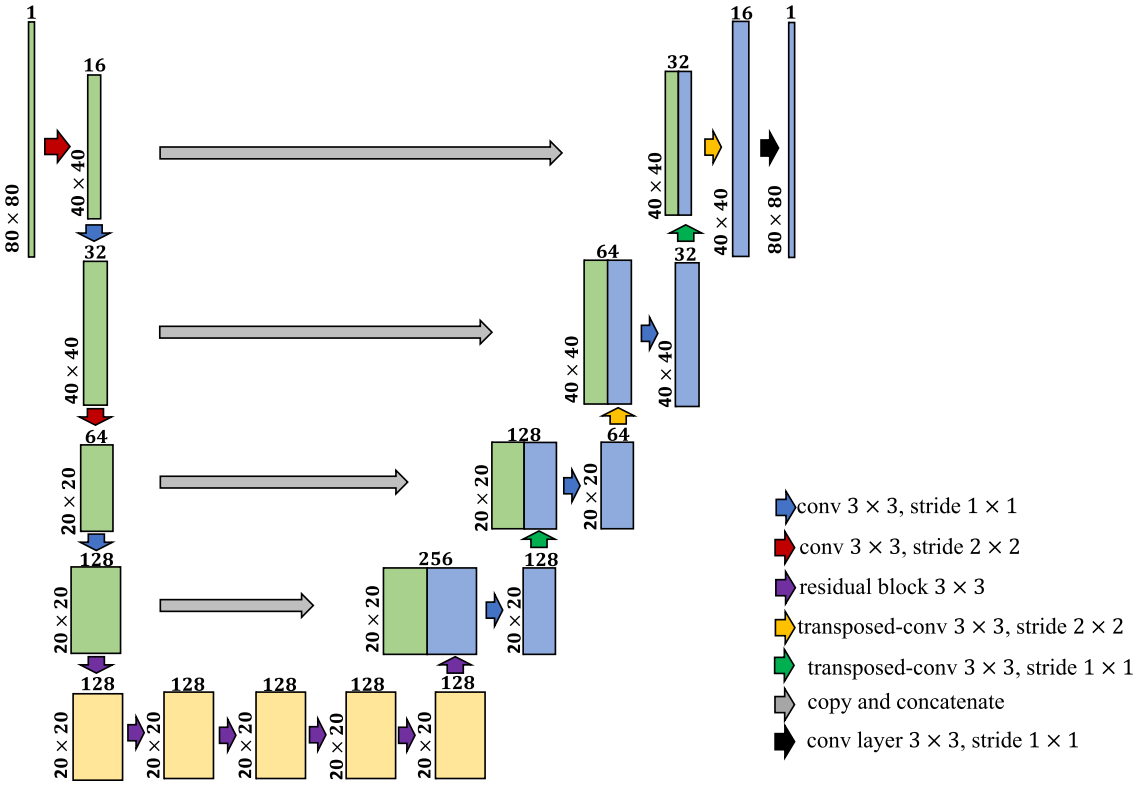
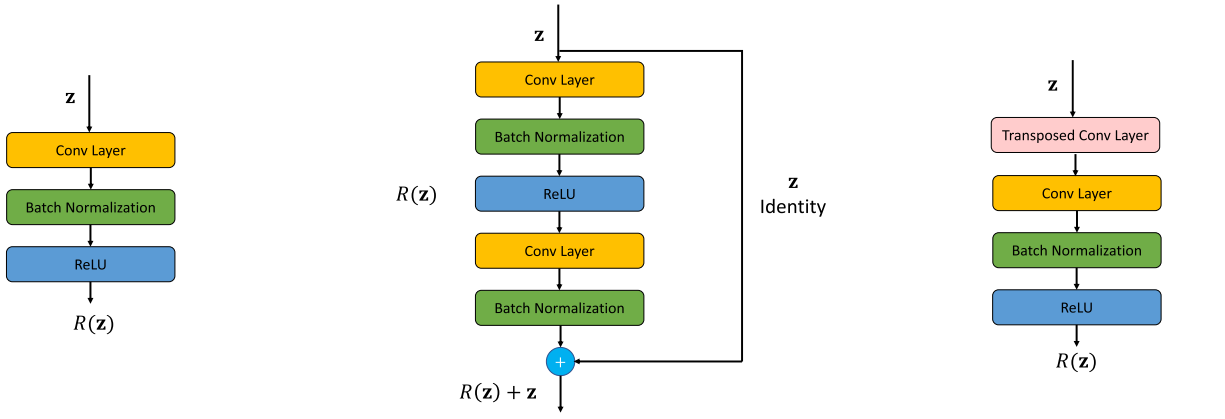


Fig. 19. Detailed schematic for the R-U-Net. Each box denotes a multichannel feature map. The number of channels N_z is indicated above each box, and the values at the left of each box indicate corresponding feature map size $N_x \times N_y$. The arrows denote the different operations.



(a) Conv block (b) Residual block (c) Transposed conv block

Fig. 20. Illustration of conv, residual and transposed conv blocks.

the encoding net to the upsampled features in the decoding net. Specifically, 'conv' in Fig. 19 represents the convolutional blocks and 'transposed-conv' denotes the transposed convolutional blocks. More details on these blocks as well as on the residual block are provided in Fig. 20.

As shown in Fig. 20, conv represents a convolutional layer followed by batch normalization and ReLU nonlinear activation, while transposed conv denotes a transposed (upsampling) convolutional layer followed by a convolutional layer, batch normalization and ReLU. A stack of two convolutional layers with 128 filters of size 3×3 constitutes a residual block, in which the first convolutional layer has skip connections with the output of the second convolutional layer.

Table 1
Recurrent R-U-Net architecture details.

Net	Layer	Output size
Encoder	Input	$(n_x, n_y, 1)$
	conv, 16 filters of size 3×3, stride 2	$(n_x/2, n_y/2, 16)$
	conv, 32 filters of size 3×3, stride 1	$(n_x/2, n_y/2, 32)$
	conv, 64 filters of size 3×3, stride 2	$(n_x/4, n_y/4, 64)$
	conv, 128 filters of size 3×3, stride 1	$(n_x/4, n_y/4, 128)$
	residual block, 128 filters of size 3×3, stride 1	$(n_x/4, n_y/4, 128)$
	residual block, 128 filters of size 3×3, stride 1	$(n_x/4, n_y/4, 128)$
	residual block, 128 filters of size 3×3, stride 1	$(n_x/4, n_y/4, 128)$
ConvLSTM	convLSTM2D block, 128 filters of size 3×3, stride 1	$(n_x/4, n_y/4, 128, n_t)$
Decoder	residual block, 128 filters of size 3×3, stride 1	$(n_x/4, n_y/4, 128, n_t)$
	residual block, 128 filters of size 3×3, stride 1	$(n_x/4, n_y/4, 128, n_t)$
	residual block, 128 filters of size 3×3, stride 1	$(n_x/4, n_y/4, 128, n_t)$
	transposed conv, 128 filters of size 3×3, stride 1	$(n_x/4, n_y/4, 128, n_t)$
	transposed conv, 64 filters of size 3×3, stride 2	$(n_x/2, n_y/2, 64, n_t)$
	transposed conv, 32 filters of size 3×3, stride 1	$(n_x/2, n_y/2, 32, n_t)$
	transposed conv, 16 filters of size 3×3, stride 2	$(n_x, n_y, 16, n_t)$
	conv layer, 1 filter of size 3×3, stride 1	$(n_x, n_y, 1, n_t)$

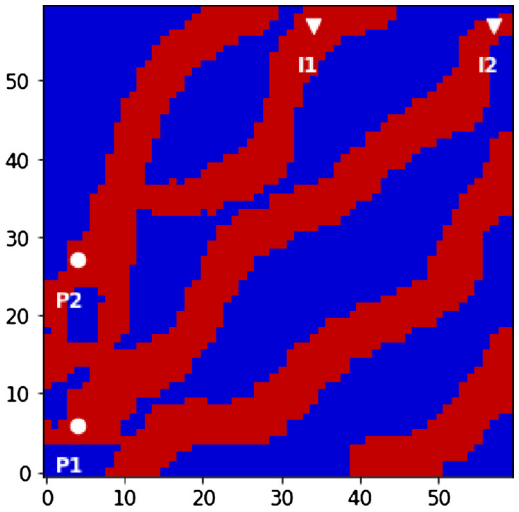


Fig. 21. Channelized 60×60 facies map, conditioned to facies type at four wells. White circles denote production wells and white triangles indicate injection wells. Model is from [42].

A.2. Recurrent R-U-Net architecture

The detailed architecture of the recurrent R-U-Net is shown in Table 1. In the table, conv, transposed conv and residual block are as described in Appendix A.1. The convLSTM2D block, which also employs 128 filters of size 3×3, performs all of the LSTM gate operations. Note that the convLSTM net generates $(n_x/4, n_y/4, 128)$ activation maps for all n_t time steps. The decoder layers process these n_t activation maps separately to produce the state maps.

Appendix B. Surrogate model results for a four-well example

In this appendix, we investigate the performance of the recurrent R-U-Net on a four-well binary channelized system. One of the realizations, along with the well locations, is shown in Fig. 21. The binary facies model and injection and production well locations are taken from Liu et al. [42]. This setup involves hard data only at the four well locations. This corresponds to much less conditioning than in the 25-well example considered earlier.

The problem specification is essentially the same as that described in Section 3.1, though here the simulation time frame is 1500 days. We apply the same recurrent R-U-Net architecture, hyperparameter settings and training procedures as were used previously. A total of 500 new test cases are simulated using the surrogate and the numerical flow simulator.

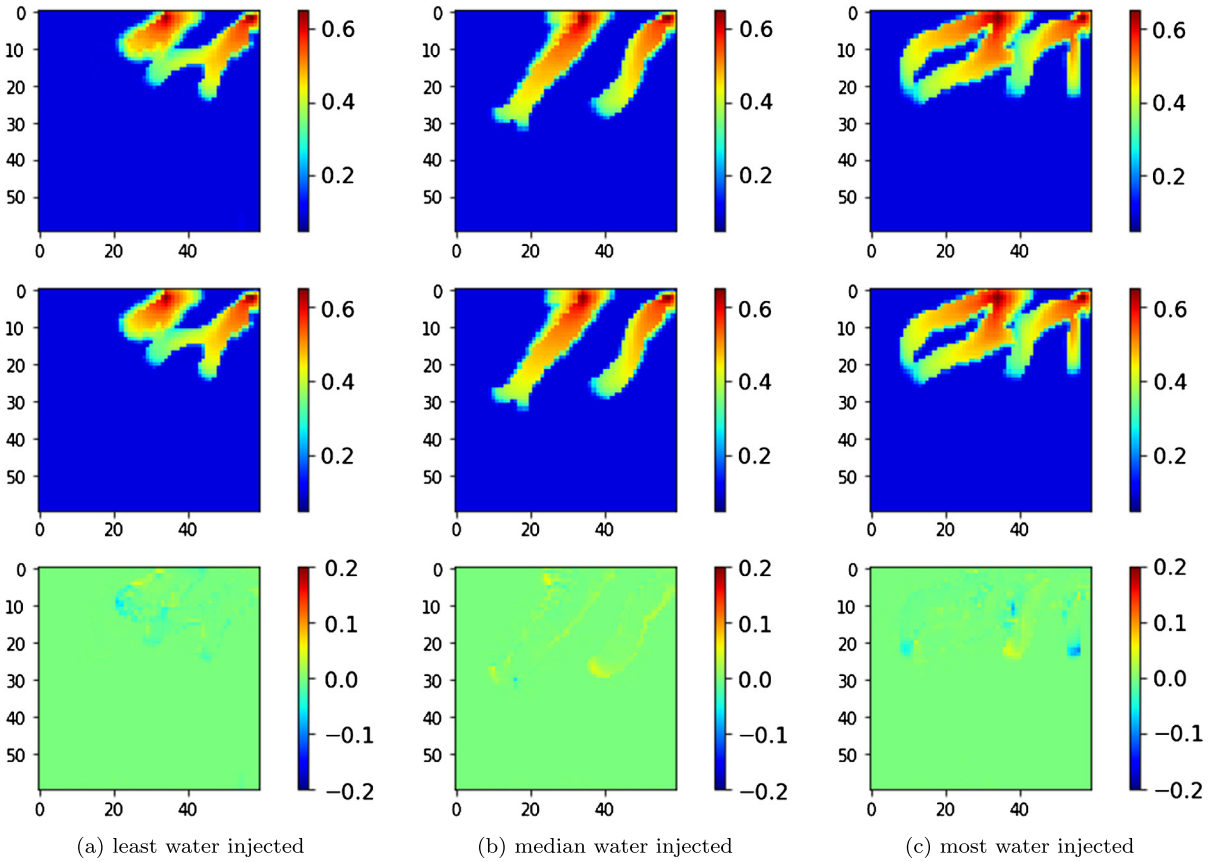


Fig. 22. Saturation maps from recurrent R-U-Net surrogate model (top row) and numerical simulator (middle row), along with difference maps (bottom row), for the test-set geomodels with (a) least, (b) median and (c) most cumulative water injected over the 1500-day simulation time frame.

For this example we present saturation and pressure maps for the cases corresponding to the least, median and most cumulative water injection over the full simulation period. Results are shown, in Figs. 22 and 23, at 1500 days. The top row displays the recurrent R-U-Net predictions, the middle row shows the numerical simulation results, and the bottom row indicates the difference. For both pressure and saturation, we see that the maps appear quite different for the three cases (in a very discrete sense for saturation), and that this variation is captured accurately by the surrogate model.

Finally, we compare the accuracy of the recurrent R-U-Net and autoregressive DenseED surrogate models for pressure and saturation predictions in this four-well example. The pressure normalization described in Section 2.6 is applied for both surrogate models, and we set $\lambda = 0$ in Eq. (9). Errors are again computed using Eqs. (13) and (15). Results are shown in Fig. 24. We again see that our recurrent R-U-Net provides generally lower error (though autoregressive DenseED is more accurate for pressure at early time) and that this error grows only slowly in time. These results are consistent with those presented earlier for the 25-well example.

References

[1] J.F. Van Doren, R. Markovinović, J.-D. Jansen, Reduced-order optimal control of water flooding using proper orthogonal decomposition, Comput. Geosci. 10 (1) (2006) 137–158.
[2] M.A. Cardoso, L.J. Durlofsky, P. Sarma, Development and application of reduced-order modeling procedures for subsurface flow simulation, Int. J. Numer. Methods Eng. 77 (9) (2009) 1322–1350.
[3] J. He, L.J. Durlofsky, Reduced-order modeling for compositional simulation by use of trajectory piecewise linearization, SPE J. 19 (05) (2014) 858–872.
[4] Y. Yang, M. Ghasemi, E. Gildin, Y. Efendiev, V. Calo, Fast multiscale reservoir simulations with POD-DEIM model reduction, SPE J. 21 (06) (2016) 2–141.
[5] Z.L. Jin, L.J. Durlofsky, Reduced-order modeling of CO_2 storage operations, Int. J. Greenh. Gas Control 68 (2018) 49–67.
[6] J. He, P. Sarma, L.J. Durlofsky, Reduced-order flow modeling and geological parameterization for ensemble-based data assimilation, Comput. Geosci. 55 (2013) 54–69.
[7] C. Xiao, O. Leeuwenburgh, H.X. Lin, A. Heemink, Non-intrusive subdomain POD-TPWL for reservoir history matching, Comput. Geosci. 23 (03) (2019) 537–565.
[8] H. Hamdi, I. Couckuyt, M.C. Sousa, T. Dhaene, Gaussian processes for history-matching: application to an unconventional gas reservoir, Comput. Geosci. 21 (2) (2017) 267–287.
[9] H. Bazargan, M. Christie, A.H. Elsheikh, M. Ahmadi, Surrogate accelerated sampling of reservoir models with complex structures using sparse polynomial chaos expansion, Adv. Water Resour. 86 (2015) 385–399.
[10] L.A.N. Costa, C. Maschio, D.J. Schiozer, Application of artificial neural networks in a history matching process, J. Pet. Sci. Eng. 123 (2014) 30–45.

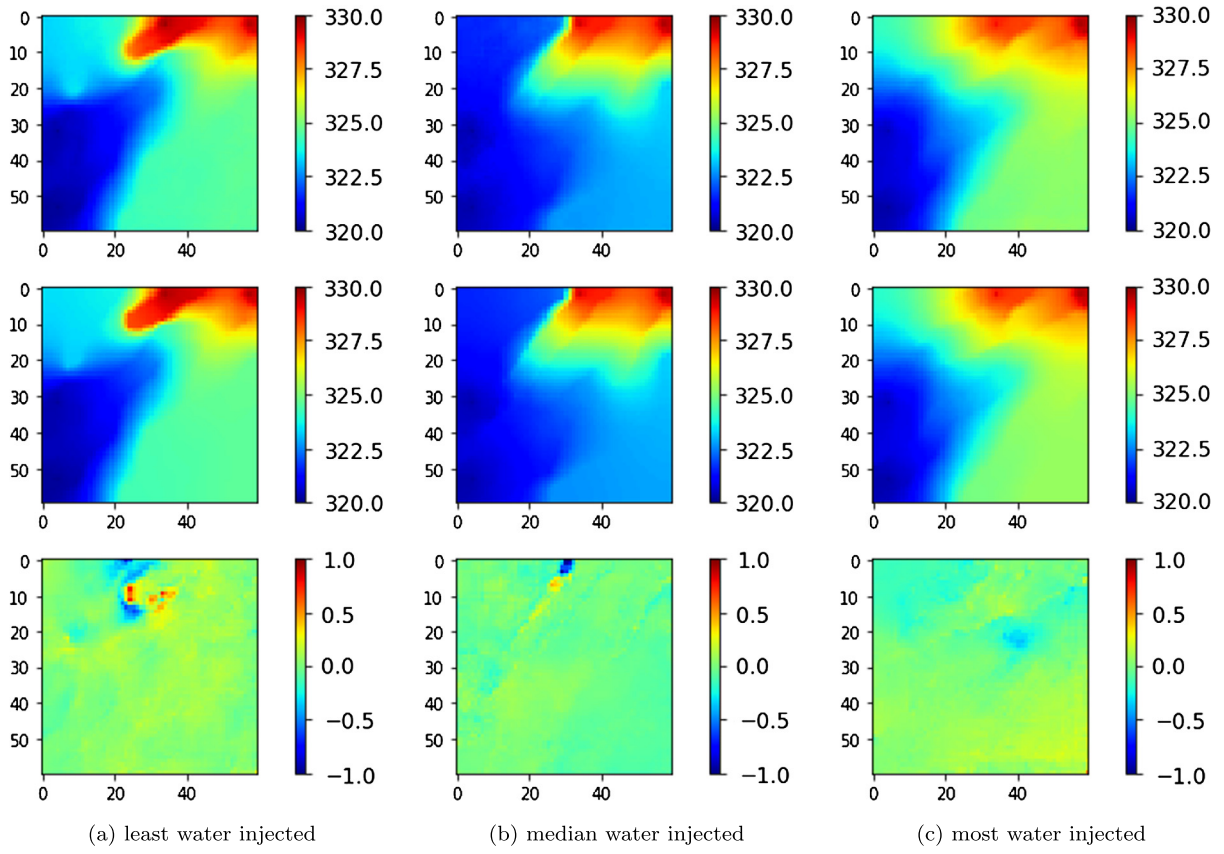
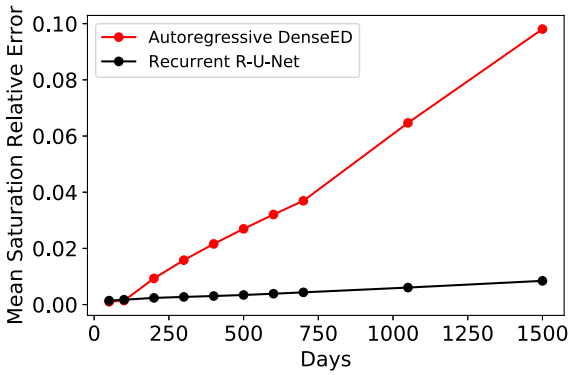
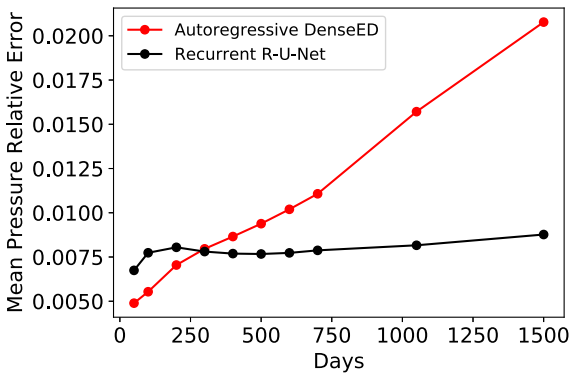


Fig. 23. Pressure maps from recurrent R-U-Net surrogate model (top row) and numerical simulator (middle row), along with difference maps (bottom row), for the test-set geomodels with (a) least, (b) median and (c) most cumulative water injected over the 1500-day simulation time frame.

[11] T. Baltrusaitis, P. Robinson, L.-P. Morency, Constrained local neural fields for robust facial landmark detection in the wild, in: Proceedings of the IEEE International Conference on Computer Vision Workshops, 2013, pp. 354–361.
[12] F. Liu, C. Shen, G. Lin, Deep convolutional neural fields for depth estimation from a single image, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5162–5170.
[13] P. Isola, J.-Y. Zhu, T. Zhou, A.A. Efros, Image-to-image translation with conditional adversarial networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1125–1134.
[14] Y. Wu, M. Schuster, Z. Chen, Q.V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Å. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, J. Dean, Google's neural machine translation system: bridging the gap between human and machine translation, arXiv:1609.08144.
[15] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova Bert, Pre-training of deep bidirectional transformers for language understanding, arXiv:1810.04805.
[16] G.K. Dziugaite, D.M. Roy, Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data, arXiv:1703.11008.
[17] S. Arora, R. Ge, B. Neyshabur, Y. Zhang, Stronger generalization bounds for deep nets via a compression approach, arXiv:1802.05296.
[18] Y. Zhu, N. Zabaras, Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification, J. Comput. Phys. 366 (2018) 415–447.
[19] S. Mo, Y. Zhu, N. Zabaras, X. Shi, J. Wu, Deep convolutional encoder-decoder networks for uncertainty quantification of dynamic multiphase flow in heterogeneous media, Water Resour. Res. 55 (1) (2019) 703–728.
[20] S. Mo, N. Zabaras, X. Shi, J. Wu, Deep autoregressive neural networks for high-dimensional inverse problems in groundwater contaminant source identification, Water Resour. Res. 55 (5) (2019) 3856–3881.
[21] S. Mo, N. Zabaras, X. Shi, J. Wu, Integration of adversarial autoencoders with residual dense convolutional networks for inversion of solute transport in non-Gaussian conductivity fields, preprint, arXiv:1906.11828.
[22] Z.L. Jin, Y. Liu, L.J. Durlofsky, Deep-learning-based reduced-order modeling for subsurface flow simulation, preprint, arXiv:1906.03729.
[23] O. Ronneberger, P. Fischer, T. Brox, U-net: convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 234–241.
[24] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[25] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, W.-C. Woo, Convolutional LSTM network: a machine learning approach for precipitation nowcasting, in: Advances in Neural Information Processing Systems, 2015, pp. 802–810.
[26] M.G. Gerritsen, L.J. Durlofsky, Modeling fluid flow in oil reservoirs, Annu. Rev. Fluid Mech. 37 (2005) 211–238.
[27] D.W. Peaceman, Interpretation of well-block pressures in numerical reservoir simulation with nonsquare grid blocks and anisotropic permeability, SPE J. 23 (03) (1983) 531–543.
[28] M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: European Conference on Computer Vision, Springer, 2014, pp. 818–833.
[29] V. Dumoulin, F. Visin, A guide to convolution arithmetic for deep learning, arXiv:1603.07285.



(a) Saturation error



(b) Pressure error

Fig. 24. Relative saturation error δ_S^t (Eq. (13)) and pressure error δ_p^t (Eq. (15)) at different time steps using the recurrent R-U-Net and autoregressive DenseED [20] procedures for the four-well channelized system.

[30] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.
[31] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[32] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, S. Khudanpur, Recurrent neural network based language model, in: Eleventh Annual Conference of the International Speech Communication Association, 2010.
[33] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE Trans. Neural Netw. 5 (2) (1994) 157–166.
[34] G. Zhu, L. Zhang, P. Shen, J. Song, Multimodal gesture recognition using 3-D convolution and convolutional LSTM, IEEE Access 5 (2017) 4517–4524.
[35] R.P. Poudel, P. Lamata, G. Montana, Recurrent fully convolutional neural networks for multi-slice MRI cardiac segmentation, in: Reconstruction, Segmentation, and Analysis of Medical Images, Springer, 2016, pp. 83–94.
[36] R. Hecht-Nielsen, Theory of the backpropagation neural network, in: Neural Networks for Perception, Elsevier, 1992, pp. 65–93.
[37] D.P. Kingma, J. Ba Adam, A method for stochastic optimization, arXiv:1412.6980.
[38] C. Li, H. Farkhoor, R. Liu, J. Yosinski, Measuring the intrinsic dimension of objective landscapes, arXiv:1804.08838.
[39] J. Frankle, M. Carbin, The lottery ticket hypothesis: finding sparse, trainable neural networks, arXiv:1803.03635.
[40] J. Sola, J. Sevilla, Importance of input data normalization for the application of neural networks to complex industrial problems, IEEE Trans. Nucl. Sci. 44 (3) (1997) 1464–1468.
[41] N. Remy, A. Boucher, J. Wu, Applied Geostatistics with SGeMS: A User's Guide, Cambridge University Press, 2009.
[42] Y. Liu, W. Sun, L.J. Durlofsky, A deep-learning-based geological parameterization for history matching complex models, Math. Geosci. 51 (6) (2019) 725–766.
[43] Y. Liu, L.J. Durlofsky, Multilevel strategies and geological parameterizations for history matching complex reservoir models, SPE J. 25 (01) (2020) 81–104.
[44] Y. Zhou, Parallel general-purpose reservoir simulation with coupled reservoir models and multisegment wells, Ph.D. thesis, Stanford University, 2012.
[45] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, J. Mach. Learn. Res. 13 (2012) 281–305.
[46] P.K. Kitanidis, Quasi-linear geostatistical theory for inversing, Water Resour. Res. 31 (10) (1995) 2411–2419.
[47] D.S. Oliver, Multiple realizations of the permeability field from well test data, SPE J. 1 (02) (1996) 145–154.
[48] G. Gao, M. Zafari, A.C. Reynolds, Quantifying uncertainty for the PUNQ-S3 problem in a Bayesian setting with RML and EnKF, in: SPE Reservoir Simulation Symposium, 2005.
[49] H.X. Vo, L.J. Durlofsky, Data assimilation and uncertainty assessment for complex geological models using a new PCA-based parameterization, Comput. Geosci. 19 (4) (2015) 747–767.
[50] C. Audet, J.E. Dennis Jr, Mesh adaptive direct search algorithms for constrained optimization, SIAM J. Optim. 17 (1) (2006) 188–217.