
A FRAMEWORK FOR DATA-DRIVEN SOLUTION AND PARAMETER ESTIMATION OF PDES USING CONDITIONAL GENERATIVE ADVERSARIAL NETWORKS

A PREPRINT

Teeratorn Kadeethum

Sibley School of Mechanical and Aerospace Engineering
Cornell University, New York, USA
tk658@cornell.edu

Daniel O'Malley

Los Alamos National Laboratory
New Mexico, USA
omalled@lanl.gov

Jan Niklas Fuhg

Sibley School of Mechanical and Aerospace Engineering
Cornell University, New York, USA
jnf853@cornell.edu

Youngsoo Choi

Lawrence Livermore National Laboratory
California, USA
choi15@llnl.gov

Jonghyun Lee

Civil & Environmental Engineering/Water Resources Research Center
University of Hawai'i at Manoa
Hawaii, USA
jonghyun.harry.lee@hawaii.edu

Hari S. Viswanathan

Los Alamos National Laboratory
New Mexico, USA
viswana@lanl.gov

Nikolaos Bouklas

Sibley School of Mechanical and Aerospace Engineering and
Center for Applied Mathematics
Cornell University, New York, USA
nb589@cornell.edu

May 28, 2021

ABSTRACT

This work is the first to employ and adapt the image-to-image translation concept based on conditional generative adversarial networks (cGAN) towards learning a forward and an inverse solution operator of partial differential equations (PDEs). Even though the proposed framework could be applied as a surrogate model for the solution of any PDEs, here we focus on steady-state solutions of coupled hydro-mechanical processes in heterogeneous porous media. Strongly heterogeneous material properties, which translate to the heterogeneity of coefficients of the PDEs and discontinuous features in the solutions, require specialized techniques for the forward and inverse solution of these problems. Additionally, parametrization of the spatially heterogeneous coefficients is excessively difficult by using standard reduced order modeling techniques. In this work, we overcome these challenges by employing the image-to-image translation concept to learn the forward and inverse solution operators and utilize a U-Net generator and a patch-based discriminator. We use heterogeneous permeability fields as an input parameter and pressure and displacement fields as outputs for the forward model, enabling calculation at least 2000 times faster than a linear finite element solver. For inverse modeling, the framework allows for estimating output in terms of the permeability field, given input of pressure and/or displacement fields, where input data is incomplete and contaminated by noise. The framework provides a speed-up of 120000 times compared to a Gaussian prior-based inverse modeling approach while also delivering more accurate inverse results. Our results

show that the proposed data-driven reduced order model has competitive predictive performance capabilities in accuracy and computational efficiency as well as training time requirements compared to state-of-the-art data-driven methods for both forward and inverse problems.

Keywords conditional generative adversarial network · reduced order modeling · heterogeneity · inverse modeling · discontinuous Galerkin · poroelasticity

1 Introduction

Many engineering applications involving porous media ranging from groundwater and contaminant hydrology to tissue engineering and drug delivery rely on physical simulations [1–6]. These simulations are usually governed by partial differential equations (PDEs). For instance, coupled hydro-mechanical (HM) processes in porous media, in which fluid flow and solid deformation tightly interact, could be described by Biot’s formulation of linear poroelasticity [7]. These PDEs could be solved analytically for simple geometries and boundary conditions [8, 9] or approximated numerically using different techniques such as finite volume or finite element methods [10–13], which we will further term as full order model (FOM). However, the latter requires substantial computational resources, especially when discontinuities arise in the solution [14, 15]. Hence, the FOM is not directly suitable to handle large-scale inverse problems, optimization, or even concurrent multiscale calculations in which an extensive set of simulations are required to be explored [15, 16].

Development of a reduced order model (ROM) [17–21] which aims to produce a low-dimensional representation of FOM could be an alternative towards handling field-scale inverse problems, optimization, or control. ROMs are generally composed of an offline and an online stage [22–25]. The offline stage begins with the initialization of uncertain input parameters, referred to as a training set. Then, the FOM is solved successively using each member of the training set as inputs, and we will refer to the corresponding solutions as snapshots from now on. Data compression techniques are then used to compress the FOM snapshots to produce basis functions that span a reduced space of very low dimensionality (compared to the original dimension) but still guarantee accurate reproduction of the snapshots [26–28]. ROMs can generate forward solutions during the online stage for any new value of parameters by seeking an approximated solution in the reduced space. The ROM methodology relies on a parametrized problem (i.e., repeated evaluations of a problem depending on parameters, which could correspond to physical properties, geometric characteristics, or boundary conditions [15, 16, 29]). However, it is difficult to parameterize spatial fields of PDE coefficients (e.g., corresponding to heterogeneous material properties) by a small number of parameters. For instance, the problem of interest in HM processes commonly involves complex subsurface structures [30–32] where the corresponding spatially distributed parameters can span several orders of magnitude and include discontinuous features. As a result, the traditional parametrized ROM might not be suitable for this type of problem because a “global” proper orthogonal decomposition (POD) approach might require a high dimensional reduced basis (see S4 for more information). Moreover, to handle inverse problems, FOM or ROM are usually used in conjunction with optimization algorithms or adjoint state methods that are not straightforward to implement and still suffer from the curse of dimensionality [33–35].

Data-driven and physics-informed modeling enabled by machine learning techniques has recently gained significant attention in scientific computing as an alternative to the aforementioned reduced order modeling techniques. Directly incorporating physical laws, Physics-informed neural networks (PINN) were proposed to solve PDEs in forward and inverse settings without the need for labeled data [36–38]. Their limitation lies in the fact that they can not efficiently handle problems with heterogeneous coefficients. Extended-PINN (xPINN) attempts to address this issue using domain decomposition [39]. However, it still cannot handle a high degree of spatial heterogeneity of the PDE coefficients as an input (see Figure 1) since it is not practical to subdivide these types of fields. Even though PINN and its variants can be used for parameter estimation problems, they cannot efficiently provide multiple inquiries for the forward problem. Data-driven learning of solution operators of PDEs has recently been proposed in DeepONet [40] and Neural Operator [40], which have been shown to approximate the solution of PDEs in a forward solution setting. These methods, however, are limited to an approximation of continuous functions, which might not be applicable to problems that deal with discontinuous input and output functional spaces, as shown in Figure 1. We note that even though the Neural Operator model [40] could generalize between spatially heterogeneous input fields, discontinuities must be smoothed prior to training. Additionally, we have found the training time required for the latter approach is exceedingly high for standard engineering applications (see S5.3).

This study focuses on a data-driven reduced order modeling approach geared towards HM processes in porous media with heterogeneous material properties, where resolving discontinuities is critical for the fidelity of the simulation. The data-driven approach is attractive because it does not require any modifications of FOM source codes, and subsequently, provides more flexibility when coupling to existing FOM platforms [41–43]. Along with the benefits of the data-driven approach, concerns arise regarding a lack of strict enforcement of physical laws. Navigating between the data-driven

and physics-informed paradigms is a challenging task, often governed by the trade-off between speed-up and accuracy as well as the availability of data.

Central to the developments in this work is an idea of generative adversarial networks (GAN). Since the introduction of GAN [44] in 2014, this architecture has gained popularity because of its generalization ability and has been widely used in image/video analysis [44, 45]. GAN is based on two networks, (1) a generator and (2) a discriminator, which compete with each other. The generator’s goal is to produce realistic output that resembles the real data (belonging to the training set). The discriminator’s job is to differentiate between the output of the generator and the real data. The training phase is complete when the discriminator cannot distinguish between real data belonging to the generator’s training set and data produced by the generator. The concept of deep convolutional neural networks (CNN) has been employed for both the generator and the discriminator of GAN to capture highly heterogeneous spatial features [46–48] and their evolution in time [49, 50]. Additionally, an augmentation of discrete labeled data has been used to control the generator’s output, termed conditional GAN or cGAN [51–54]. Still, all the aforementioned frameworks cannot be used directly to address the data-driven solution of PDEs since they are limited to only categorical conditioning, which is not sufficient for the parametrization of spatially heterogeneous input fields.

To approximate primary variables with given heterogeneous coefficients of the PDEs and to estimate the spatially heterogeneous coefficients given the primary variables, we employ the idea of the image-to-image translation framework [55]. Different from a classical cGAN in which is conditioned by labeled data (e.g., classes of object the image is representing), the image-to-image translation framework relies on conditional fields, which could be edges, masks, points, or segmentation maps [56–61]. This idea has been extended to high-resolution images [56, 57], unsupervised translation [59–61], controllable facial attribute editing [58], and controllable segmentation mapping [62]. We rely on the image-to-image translation concept because we want to construct mappings of gradients, discontinuities, and multiscale features between input and output fields. We achieve these goals through (1) nonlinear data compression (latent representation) and (2) multiscale feature mapping (skip connections), as illustrated in Figure 1. The work presented here is novel in two main ways

1. We are the first to propose and adapt the image-to-image translation concept to learn forward and inverse solution operators of PDEs. This data-driven ROM framework can efficiently parametrize highly complex input fields and generalize mappings between those input fields and their corresponding output fields. We show that the framework can handle discontinuities in both input and output fields, which is different than the current state-of-the-art studies of data-driven learning of solution operators [40, 63].
2. We illustrate that using a U-Net generator [64] instead of the deep CNN (see S5.4 for more detail and performance comparison), and a patch-based discriminator [65] along with spectral normalization [66] or Wasserstein loss with gradient penalty [67, 68] can improve the framework’s accuracy and training stability without adding costly training time. In the forward setting, this framework provides a speed-up of at least 2000-fold while maintaining a high level of accuracy. For the inverse modeling in which input data is sparsely available and noisy, the framework delivers a speed-up of 120000 times compared to a Gaussian prior-based inverse modeling approach, while its accuracy is significantly improved.

2 Results

We present two possible applications of our proposed model; (1) forward modeling: with given permeability fields, the model approximates the field of the primary variables (i.e., pressure and displacement fields), and (2) inverse modeling: with given a subset of primary variable fields, the model estimates the permeability field. We utilize a Discontinuous Galerkin finite element model of linear poroelasticity developed in [69, 70] to generate training and test sets. For input parameters to the FOM, we utilize three types of highly heterogeneous permeability fields: (1) a permeability field given as a Gaussian distribution (S5.1.1), (2) a permeability field defined by a bimodal transformation (S5.1.2), and (3) a permeability field from a Zinn & Harvey transformation (S5.1.3) [71]. We here only present results of the W model (corresponding to one out of three variants of our ROM), which is developed based on Wasserstein loss with gradient penalty (see S3.3) as it provides the best accuracy and the most training stability. We compare the performance of three models, (1) base model (S3.1), (2) SN model (S3.2), and (3) W model (S3.3) in detail for both training and test dynamics in S5.1.1, S5.1.2, and S5.1.3.

2.1 Forward modeling

Focusing on the steady-state solution of poroelasticity equations (as described in S2) while considering highly heterogeneous material properties, we train and test the ROM based on three types of highly heterogeneous permeability fields simultaneously. The input to the ROM is three types of permeability field, and the output is pressure and displacement

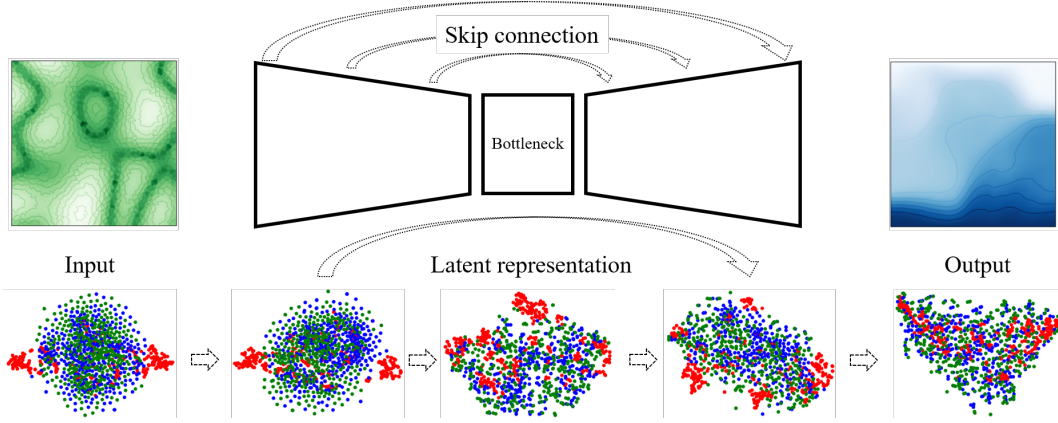


Figure 1: Illustration of strongly heterogeneous input and output fields. Our framework aims to handle continuous and/or discontinuous input and output fields through nonlinear data compression and skip connections. Red, blue, and green dots represent t-Distributed Stochastic Neighbor Embedding (t-SNE) under three different statistical distributions. We illustrate that red, blue, and green dots distribute differently for each stage of nonlinear compression. In the latent representation, all colors mix with each other. Multiscale features are captured progressively through skip connections, so that decoded input samples become uncorrelated and evenly distributed in the latent space. Therefore, without skip connection (multiscale data transfer), the output field might lose some essential features (see S5.4).

fields. Details on how we generate each permeability field, set model parameters, training, and testing of the ROM can be found in S5.1.1 for the Gaussian distribution, S5.1.2 for the bimodal transformation, and S5.1.3 for the Zinn & Harvey transformation. We illustrate three test cases (out of 3000 test examples, which are excluded from the training set) in Figure 2. The size of the training set will be discussed in the following paragraphs. From this figure, we note that our model can provide reasonable approximations of the FEM results. The difference between solutions produced by the FOM and ROM (further referred to as DIFF) is calculated by

$$\text{DIFF}(\mathbf{X}) = \left| \mathbf{X}_h - \widehat{\mathbf{X}}_h \right|. \quad (1)$$

\mathbf{X}_h is a finite-dimensional approximation of the set of primary variables, corresponding to displacement and pressure fields in this study, as calculated by the FOM. $\widehat{\mathbf{X}}_h$ is an approximation of \mathbf{X}_h produced by the ROM. An extensive discussion can be found in the **problem description and methods** section.

The maximum DIFF values lie in a range from 0.01% to 0.1% (depending on the type of permeability of the test cases). When the ROM is trained and tested separately for each type of permeability field (see Figs. S5.2, S5.5, and S5.8), the maximum DIFF values remain in the same order of magnitude. This behavior indicates that the proposed ROM can generalize efficiently among three types of microstructures. As expected, the permeability field from the Zinn & Harvey transformation case has the highest DIFF value since the permeability field exhibits extensive discontinuities over the domain and the highest contrast. The permeability field as bimodal transformation generally has greater DIFF values than the Gaussian distribution case.

We then investigate the effect of the number of training samples (N) on the model dynamics (i.e., accuracy and its trend). We have five cases with a different number of training examples; (1) $N = 1500$ training examples, (2) 3000 training examples, (3) 7500 training examples, (4) 15000 training examples, and (5) 30000 training examples. As previously mentioned, we train and test our model using three types of permeability fields; hence, we divide the portion of each field equally. For instance, assuming we have 30000 training examples, we use 10000 samples from each permeability field. In line with this, for testing, we use $N = 3000$ testing examples (1000 examples from each field) for all five cases.

The root mean square error (RMSE) of the test cases, defined as

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^N (x_i - \hat{x}_i)^2}{N}}, \quad x_i \in \mathbf{X}_h \text{ and } \hat{x}_i \in \widehat{\mathbf{X}}_h, \quad (2)$$

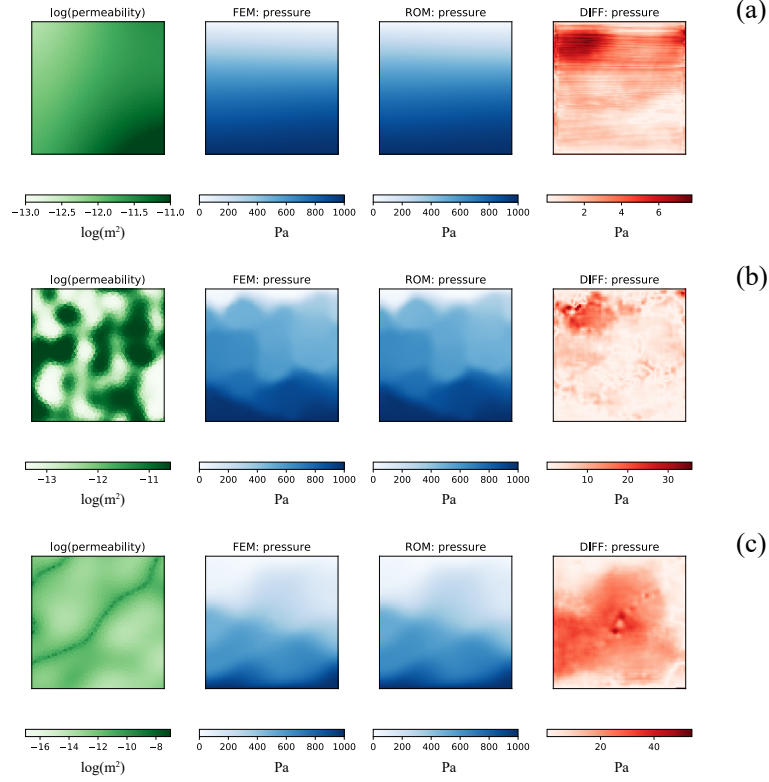


Figure 2: The results of the test case of the W model trained with 30000 examples (10000 for each Gaussian distribution, bimodal transformation, and Zinn & Harvey transformation). The model then is tested using 3000 examples (1000 for each Gaussian distribution, bimodal transformation, and Zinn & Harvey transformation). These three cases shown here are randomly picked from 3000 test examples. Note that the ranges (minimum and maximum values) of each permeability field are different. However, our model can still predict the pressure field with acceptable accuracy.

is evaluated and presented in Figure 3. We note that x_i and \hat{x}_i are the ground truth (FOM result) and approximated values (ROM result), respectively. We observe that as the number of training examples increases, the RMSE values and their trends become more accurate, see Figs. 3a-e. As we decrease the number of training samples (see 3a-c), we can observe more overfitting [72]. These behaviors are discussed in detail in S5.2.2 and S5.2.3. Our numerical experiment shows that if we keep the number of training samples greater than 7500 (2500 examples for each field), our ROM framework performs well.

So far, we have seen that the ROM accuracy is acceptable when the W model is employed. Additionally, this model exhibits a reliable training behavior. We now investigate the W model's efficacy by comparing the gained speed-up and the lost accuracy using the ROM. We apply the multifidelity Monte Carlo (MFMC) idea and criteria used in [73]. The accuracy loss is quantified through a correlation between FEM and ROM using

$$\rho_{\mathbf{X}_h, \widehat{\mathbf{X}}_h} = \frac{\text{COV}(\mathbf{X}_h, \widehat{\mathbf{X}}_h)}{\text{SD}(\mathbf{X}_h) \text{SD}(\widehat{\mathbf{X}}_h)}, \quad (3)$$

where $\rho_{\mathbf{X}_h, \widehat{\mathbf{X}}_h}$ is the Pearson correlation coefficient of \mathbf{X}_h with $\widehat{\mathbf{X}}_h$, $\text{COV}(\cdot)$ is a co-variance, and $\text{SD}(\cdot)$ is standard deviations. To quantify a trade-off between speed-up and accuracy, we use the following criterion [73, 74]

$$\frac{w_{\mathbf{X}_h}}{w_{\widehat{\mathbf{X}}_h}} > \frac{1 - \rho_{\mathbf{X}_h, \widehat{\mathbf{X}}_h}^2}{\rho_{\mathbf{X}_h, \widehat{\mathbf{X}}_h}^2}. \quad (4)$$

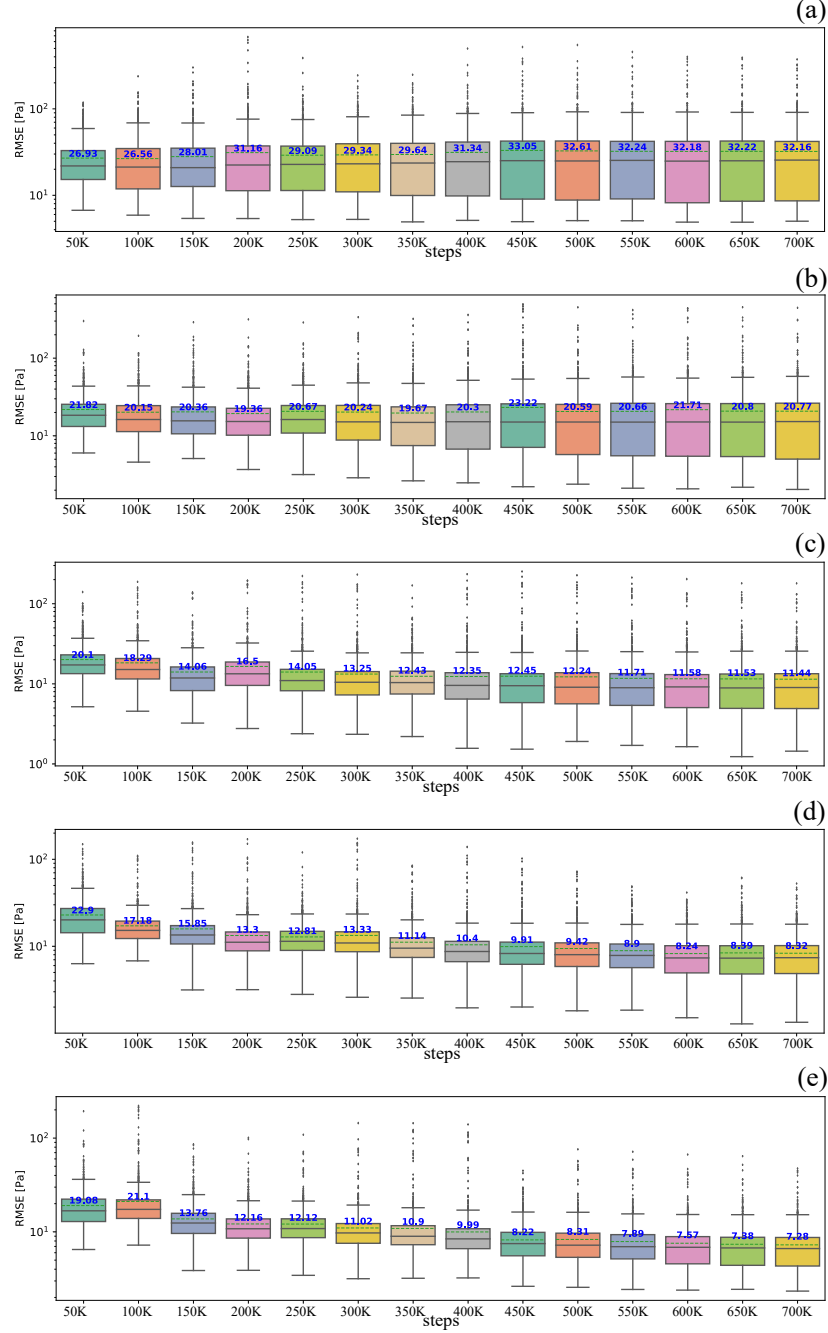


Figure 3: Root Mean Square Error (RMSE) over the training steps of the W model using (a) 1500 training samples, (b) 3000 training samples, (c) 7500 training samples, (d) 15000 training samples, and (e) 30000 training samples. We use an equal number of training examples for each permeability field. For instance, in (e), we use 10000 for each Gaussian distribution, bimodal transformation, and Zinn & Harvey transformation. Each step refers to each time we perform back-propagation, including updating both generator and discriminator's parameters.

Here, $w_{\mathbf{X}_h}$ and $w_{\widehat{\mathbf{X}}_h}$ denote the computational cost in wall times for computing \mathbf{X}_h and $\widehat{\mathbf{X}}_h$, respectively. $w_{\mathbf{X}_h}$ and $w_{\widehat{\mathbf{X}}_h}$ are approximately 4 and 0.002 seconds resulting in a speed-up of 2000. As we solved a time-dependent set of equations to obtain the steady-state response, we utilized ten steps of the time-integration scheme, resulting in 40 seconds for the FOM steady-state solution indicating a speed-up of 20000. Here, we compare to the time of just one solution step of the FOM corresponding to a linear problem for a fair comparison. Any time-dependent or nonlinear

problem we lead to a greater speed-up. We present the results of (4) for (1) 1500 training samples, (2) 3000 training samples, (3) 7500 training samples, (4) 15000 training samples, and (5) 30000 training samples in Figure 4. From this figure, we observe that all five cases satisfy (4) (i.e., the blue line is always above the red line). Moreover, as the number of training examples increase the differences between FEM and ROM results decrease (see $\frac{1-\rho^2_{\mathbf{x}_h, \widehat{\mathbf{x}}_h}}{\rho^2_{\mathbf{x}_h, \widehat{\mathbf{x}}_h}}$ value).

Note that (4) does not account for the time used to train the ROM framework. On average, it took us 12 hours to prepare the base and SN models and 14 hours to train the W model with a Quadro RTX 6000 and 24GB RAM. We train all of our models for all cases for 700000 steps. Again, a step refers to a back-propagation of the loss, including updating both generator and discriminator’s parameters throughout this section. The benefit of the proposed framework is that it is a one-time cost for the offline training phase. Then the framework could be inquired many times with much cheaper cost while providing sufficient accuracy (refer to Figure 4).

We propose the following thought experiment; a small oil & gas company needs to make decisions daily about operating ten wells. Their goal is to maximize the expected net present value (NPV). One way to compute the expected NPV is to run 1000 FEM simulations for each of the ten wells every day. This results in 10000 daily simulations. This operation might potentially be too expensive for this small company. An alternative is to use MFMC with the FEM and ROM frameworks. In this case, the company has a one-time cost of, say, 10000 model runs to train the ROM framework. On a daily basis, the company could perform 10 FEM runs, and 10000 ROM runs for each of their ten wells and get better accuracy than 1000 FEM runs per well. In this circumstance, a considerable amount of time and computing resources is saved compared to the 10000 daily FEM simulations that a Monte Carlo (MC) simulation would require. The savings become even more pronounced if the company considers multiple ways to operate the wells each day (e.g., increase the production rate, decrease the production rate, keep it the same). We note that the framework proposed here could also handle different boundary conditions by adjusting the input field to the generator and discriminator.

2.2 Inverse modeling

In contrast to the forward modeling setup where the coefficients were the input and the solution field in terms of the primary variables was the output, here we use a subset of pressure and displacement fields at the steady-state solutions of the FOM as an input, i.e., measurements, to the ROM framework. The model’s output is the reconstructed permeability field discussed in Section S5.1.3. The model’s input could have one, two, or three channels (i.e., pressure, displacement in the x-direction, and displacement in the y-direction). Throughout this section, we only show two cases; (1) input has one channel - pressure field, and (2) input has three channels - fields of pressure, displacement in the x-direction, and displacement in the y-direction. We note that one could also utilize the ROM framework as a forward approximator (as we did in the **forward modeling** section) combined with an optimization algorithm to solve the inverse problem.

We test our framework in an inverse setting against the Principal Component Geostatistical Approach (PCGA) [75, 76], which is a scalable Hierarchical Bayesian model with Gaussian prior. PCGA utilizes the low dimensional space of the prior covariance, i.e., principal components, to approximate the maximum a posteriori estimate as to the inverse solution with its linearized uncertainty and is considered as one of the state-of-the-art inverse modeling methods in geoscience [77]. It only requires a few hundreds of the forward model runs in total without any intrusive implementation and accelerates dense matrix operations through the linearly scalable fast linear algebra methods [78]. We use PCGA in conjunction with our FOM solver [69, 70] in this study. We reduce the available data points to 3% of the input in a finite-dimensional setting, and we also illustrate the performance of the framework using noisy data. Note that we represent no measurements data using a flag of -1 (e.g., -1000 in a real value for pressure field).

2.2.1 Training and testing data is not corrupted by noise

The details of cGAN architecture, parameters, and loss functions could be found in S3. We note that we only use the W model (S3.3) in the current comparison. Moreover, the input field of both training and testing data is not corrupted by noise. We note that this practice is rarely done in inverse modeling literature since data acquisition is always subject to errors. However, this problem is used to test our framework robustness before we progress to the noisy data in the following section. The PCGA parameters are set as follows; the number of principal components is 100, the prior distribution of log permeability follows a Gaussian distribution with a prior standard deviation of $2 \log(\text{m}^2)$ and exponential covariance with a scale length of $[0.1, 0.1]$. The variance of the measurement error is set to 10^2 Pa^2 .

The comparison between the W model and PCGA is shown in Figure 5 for three test cases. We note that we only use pressure as an input field for both models. Since we consider the coupled hydro-mechanical problem as in S2, the pressure data is related to the displacement through the coupled PDEs so that the pressure data alone may characterize spatially distributed permeability field better than those obtained from the pressure data in the single-phase flow configuration [79, 80]. Moreover, we use the result from the checkpoint at 50000 steps for the W model. From Figure 5,

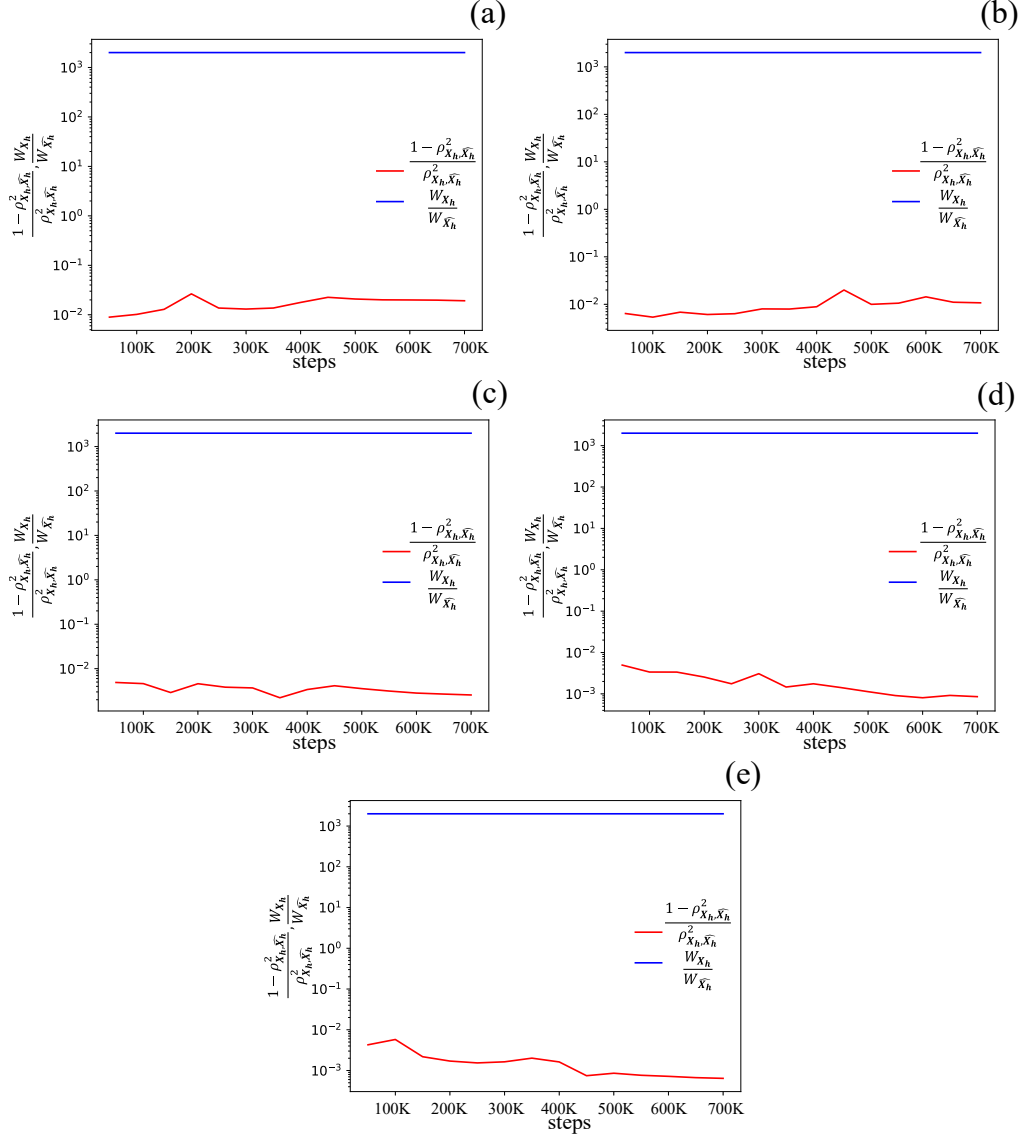


Figure 4: Results of (4) for W model using (a) 1500 training examples, (b) 3000 training examples, (c) 7500 training examples, (d) 15000 training examples, and (e) 30000 training examples. We use an equal number of training examples for each permeability field. For instance, in (e), we use 10000 for each Gaussian distribution, bimodal transformation, and Zinn & Harvey transformation. Each step refers to each time we perform back-propagation, including updating both generator and discriminator's parameters.

we observe that the W model could capture most of the structures and details with an acceptable error. On the other hand, the PCGA's inversion with sparse pressure measurements does not provide accurate results as expected since the estimated permeability fields are smoothed due to Gaussian prior assumption and lack of the displacement information that the pressure data alone cannot capture. The RMSE of the W model is from 0.17 to 0.33 $\log(\text{m}^2)$ while the RMSE of the PCGA is from 0.58 to 0.63 $\log(\text{m}^2)$.

We run the PCGA model using 36 cores (AMD Ryzen Threadripper 3970X) using the python package pyPCGA <https://github.com/jonghyunharrylee/pyPCGA>, and the inversion takes approximately four to five minutes to converge with six to eight Gauss-Newton iterations. We train the W model on a single Quadro RTX 6000, and it takes around 1 hour for the model at the 50000 steps checkpoint. For the online phase or prediction, the W model takes about 0.002 seconds. Hence, the W model could provide a speed-up of 120000. We note that even though the W model must be trained, it could be used repeatedly to estimate the permeability field much faster than the pyPCGA approach.

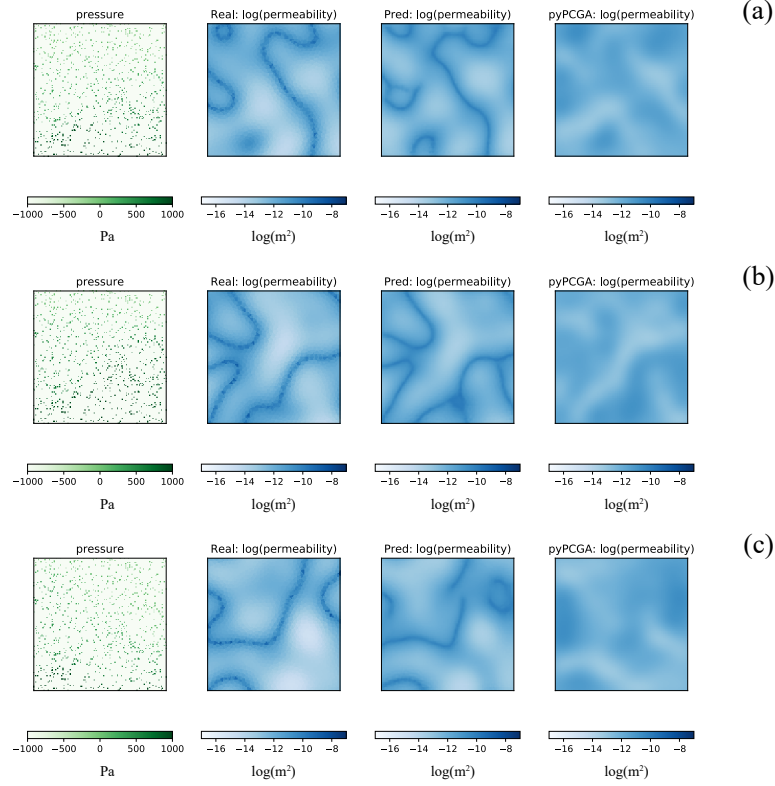


Figure 5: Three test cases' results (a), (b), and (c) of the W model and pyPCGA using 3% of input fields. For the W model, we use only pressure as its input, and the number of training examples is 10000.

We also illustrate that with more input data (pressure and displacement measurements as input), the W model could provide a better approximation, see Figure 6. The average RMSE values of the checkpoint at 50000 steps are $0.71 \log(\text{m}^2)$ and $0.52 \log(\text{m}^2)$ for the W model using only pressure as input and both pressure and displacement as input, respectively. The RMSE behavior clearly shows the overfitting behavior, as also observed in S5.2.2 and S5.2.3. This observation suggests that as the number of available data decreases, we should train the model using fewer steps.

2.2.2 Training and testing data is corrupted by noise

Next, we perform a study of the effect of added noise, which is created from the true data (\mathbf{X}_h) as follows [36, 81, 82]

$$\mathbf{X}_{h\text{noise}} = \mathbf{X}_h + \epsilon \text{SD}(\mathbf{X}_h) \mathcal{G}(0, 1), \quad (5)$$

where $\mathbf{X}_{h\text{noise}}$ is the input data with noise. The constant ϵ , which is set to 0.05, determines the noise level as it is multiplied with the standard deviation of the input field. $\mathcal{G}(0, 1)$ is a random value which is sampled from the standard normal distribution with mean and standard deviation of zero and one, respectively.

The comparison between the W model and PCGA with noisy data is illustrated in Figure 7. Like the previous section, the W model could approximate the permeability field using only the pressure field as an input with RMSE from 0.39 to $0.63 \log(\text{m}^2)$. The PCGA has approximately from 0.62 to $0.82 \log(\text{m}^2)$ of RMSE value. These results illustrate that the W model is tolerant against noise in the input data. Again, during the online phase, the W model could estimate the permeability field was significantly more efficient with respect to computational time. We note that the variance of the measurement error is set to 25^2 Pa^2 .

Figure 8 presents the evolution of the RMSE behavior of the W model using (a) pressure as input and (b) both pressure and displacement as input. The average RMSE values of the checkpoint at 50000 steps are $0.72 \log(\text{m}^2)$ and $0.63 \log(\text{m}^2)$ for the W model using only pressure as input and pressure and displacement as input, respectively. While the RMSE is not much affected by the noise for the model with only pressure as input, the model with pressure and displacement as input suffers from the noise more severely (i.e., more input data, more noise effects). Similar to S5.2.2 and S5.2.3, and the previous section, the RMSE values illustrate the overfitting behavior, and it implies that as the number of available data decreases, we should train the model with an early stopping strategy.

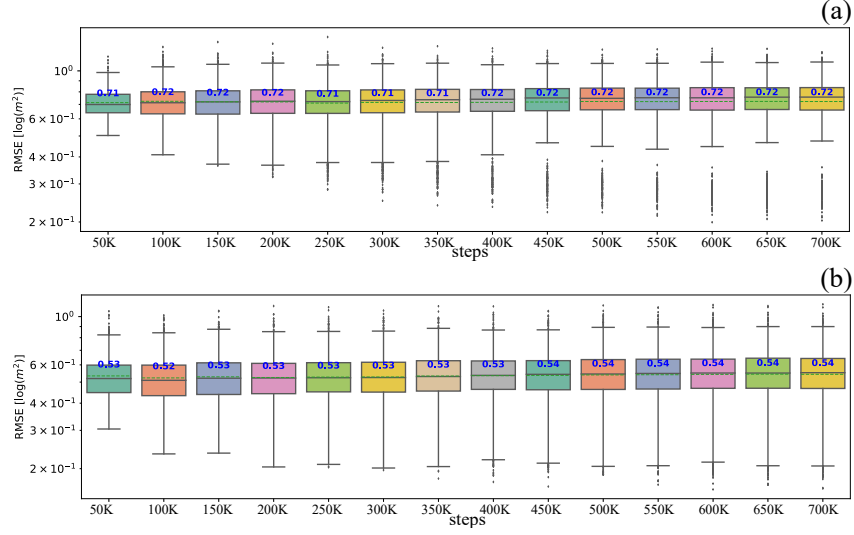


Figure 6: Root Mean Square Error (RMSE) of W model using 3% of input fields (a) pressure as input and (b) both pressure and displacement as input. The number of training examples is 10000. Each step refers to each time we perform back-propagation including updating both generator and discriminator's parameters.

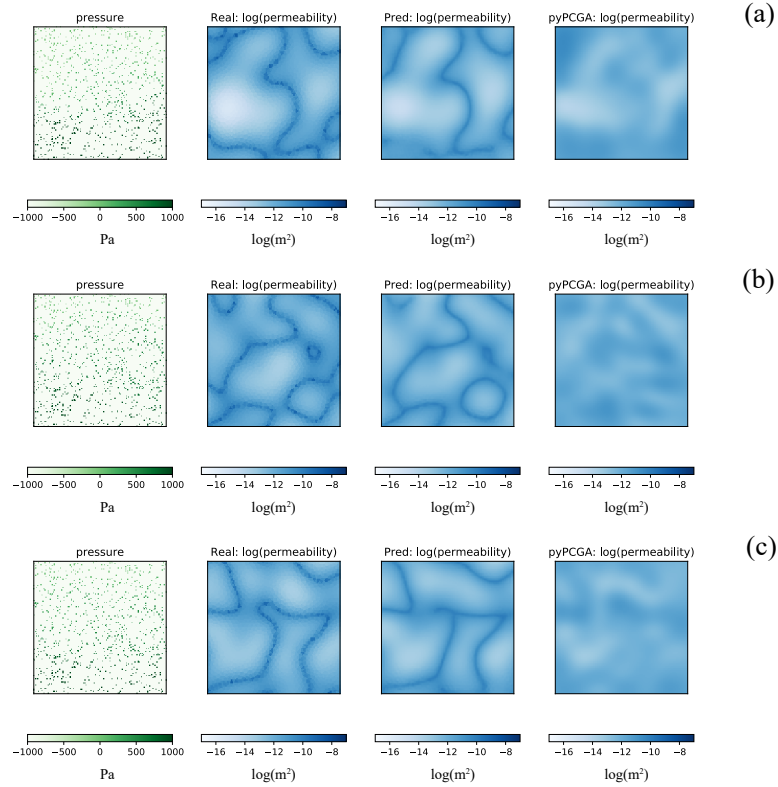


Figure 7: Three test cases' results (a), (b), and (c) with a noisy data of the W model and pyPCGA using 3% of input fields. We use only pressure as its input, and for the W model, the number of training examples is 10000.

3 Discussion

This manuscript presents a data-driven framework for solving a system of partial differential equations in a forward and inverse setting, focusing on coupled hydro-mechanical processes in heterogeneous porous media (S2) where

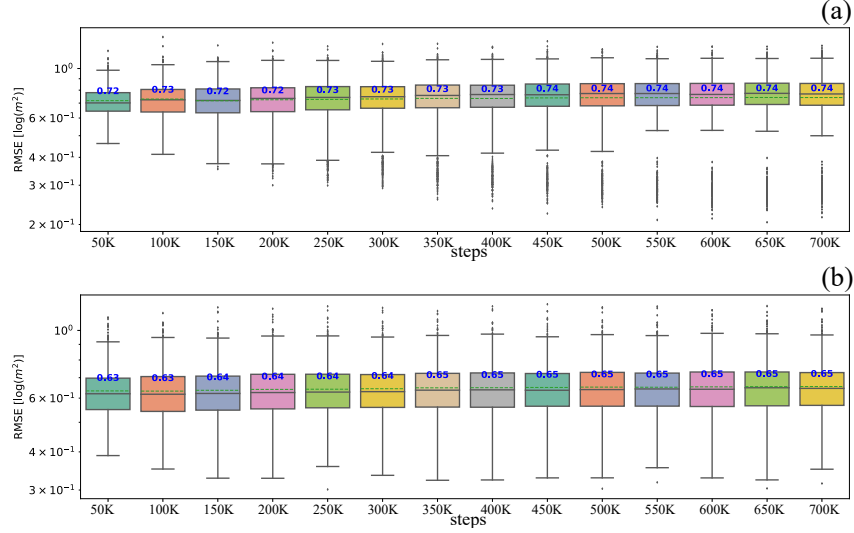


Figure 8: Root Mean Square Error (RMSE) of W model using 3% of input fields (a) pressure as input and (b) pressure and displacement as input. The number of training examples is 10000. Each step refers to each time we perform back-propagation including updating both generator and discriminator’s parameters.

discontinuities commonly arise. Our framework is developed using cGAN image-to-image translation concept, which is composed of the U-Net generator and patch-based discriminator (S3), as a means to learn the forward and inverse solution operator of a system of PDEs. Our approach uses the cGAN image-to-image translation concept to enable efficient parametrization of spatially heterogeneous input fields while mapping multiscale features to the output (see S5.4 for more detail). The model has three variations: (1) base model with classic adversary loss (S3.1), (2) SN model with spectral normalization (S3.2), and (3) W model with Wasserstein loss and gradient penalty (S3.3).

Based on S5.1.1, S5.1.2, and S5.1.3, we have illustrated that the proposed framework can efficiently learn the forward solution operator and generate solutions outside of the training set, which have acceptable accuracy with respect to the full-order model (a Discontinuous Galerkin Finite Element framework in our case). It does so, even in exploring highly heterogeneous fields of material parameters, and more specifically, permeability fields in fluid-filled porous media. This observation suggests that our framework can be utilized to solve other linear and nonlinear PDEs, exploring the effect of heterogeneous material properties such as Young’s modulus, Poisson’s ratio, or porosity in the material and structural response. Although we have two primary variables in our HM problem, corresponding to the pressure and displacement fields, we only present the results of the pressure field for conciseness. The displacement field results also have the same range of relative RMSE. The RMSE values are two to three orders of magnitude less than the maximum value of the primary variables. Moreover, the ROM with W loss, referred to as the W model in this manuscript, showcases the best accuracy and the most stable training behavior. However, the W model is computationally more expensive to train than the base and SN models. The W model’s training time for 700000 back-propagation steps is two hours greater than that of the base and SN models.

We then focus on the ability of the proposed framework to generalize in the **Results: Forward modeling** section. There, we demonstrate that the framework can approximate primary variables given highly heterogeneous permeability fields, which have significantly different characteristics and ranges. Again, we utilize three types of highly heterogeneous permeability fields: (1) a permeability field given as a Gaussian distribution, (2) a permeability field defined by a bimodal transformation, and (3) a permeability field from a Zinn & Harvey transformation. The model with only 7500 training examples (2500 examples for each field) could provide the RMSE of two orders of magnitude less than the maximum value of the primary variable. This behavior shows that our model does not require extensive training sets, which is very beneficial in practice. The developed framework could give a speedup of approximately 2000 and still be able to capture most of the high-fidelity model results (finite element method). In S5.3, we illustrate that the W model could handle a binary function as an input. Moreover, the W model has RMSE approximately four-order of magnitude less than the neural operator approach [40]. This characteristic would be beneficial to various applications such as optimization, uncertainty quantification, and multiscale modeling.

S5.2.1 shows that the framework could also be used for inverse modeling (i.e., with given fields of primary variables, we inquire the model to approximate the material property distributions - the permeability field in this case). As we observe that the W model outperforms the base and SN models in terms of training stability and model’s accuracy on

the test set, we only use the W model in the inverse modeling part. The RMSE values are two orders of magnitude less than the maximum value of the permeability field. In addition, using both pressure and displacement fields as input, the model does not provide any significant incremental accuracy compared to the model with only pressure field as input since the pressure data contains information on the displacement field through the coupled HM PDEs. We note that this statement is valid only when the full data of the input field is provided.

In many realistic cases, however, the input data is sparse, especially in subsurface operations such as geothermal, groundwater, or hydrocarbon harvesting, since measurement points are strictly limited to the location of the wells or non-intrusive geophysics and InSAR data. Therefore, we use S5.2.2 and S5.2.3 to demonstrate that our framework could also provide reasonable accuracy (i.e., still approximately two orders of magnitude less than the maximum value of the permeability field) in cases where available input data is incomplete (as low as six % of the completed data). As expected, the more insufficient the data set is, the less accurate the model becomes. In contrast to S5.2.1, where input data is complete, we could not observe a significant difference between using both pressure and displacement fields and using only pressure field as input; the W model with both pressure and displacement fields as input has higher accuracy than the W model with only a pressure field in cases where input data is incomplete. This observation reflects the fact that as the available input data is decreased (spatially), the additional information from the other fields, displacement field, in this case, could provide more information to the model and resulting in better accuracy.

We compare our framework in an inverse setting with Principal Component Geostatistical Approach (PCGA) method in **Results: Inverse modeling** section. We further reduce the available data from six % to three % of the completed data. Moreover, we test our model with cases where we have experimental measurements are corrupted by noise. Our results illustrate that the framework still could provide decent approximations of the permeability field as the RMSE values are two orders of magnitude less than the maximum value of the permeability field. Moreover, our model's performance is better than the PCGA approach since the estimated permeability fields of the PCGA approach are smoothed due to Gaussian prior assumption and lack of sensitivity in the pressure data alone. We also observe that the model with both pressure and displacement fields as input is more affected by noise in data than the model with pressure as an input alone. For a computational time comparison, our framework provides approximately a speed-up of 120000.

4 Problem description and methods

Our main target is illustrated in Figure 9. We consider a system of time-independent PDEs

$$\begin{aligned} \mathbf{F}(\mathbf{X}; \boldsymbol{\mu}) &= \mathbf{0} \quad \text{in } \Omega, \\ \mathbf{X} &= \mathbf{f}_D \quad \text{on } \partial\Omega_D, \\ -\nabla \mathbf{X} \cdot \mathbf{n} &= \mathbf{f}_N \quad \text{on } \partial\Omega_N. \end{aligned} \tag{6}$$

corresponding to quasi-static or steady-state problems, where $\Omega \subset \mathbb{R}^{n_d}$ ($n_d \in \{1, 2, 3\}$) denotes the computational domain, and $\partial\Omega_D$ and $\partial\Omega_N$ denote the Dirichlet and Neumann boundaries respectively. \mathbf{X} is a set of scalar ($\mathbf{X} \in \mathbb{R}$) or tensor valued (e.g. $\mathbf{X} \in \mathbb{R}^{n_d}$ or $\mathbb{R}^{n_d} \times \mathbb{R}^{n_d}$) generalized primary variables, and $\boldsymbol{\mu}$ are scalar ($\boldsymbol{\mu} \in \mathbb{R}$) or tensor valued (e.g. $\boldsymbol{\mu} \in \mathbb{R}^{n_d}$ or $\mathbb{R}^{n_d} \times \mathbb{R}^{n_d}$) generalized parameters for which a solution can be obtained in a finite-dimensional setting through a FOM. We are interested in developing a data-driven ROM to efficiently (1) approximate primary variables \mathbf{X} for a solution of the system of PDEs given heterogeneous fields for the generalized parameters $\boldsymbol{\mu}$, and (2) estimate the generalized parameters $\boldsymbol{\mu}$ when given information from the solution of the primary variable field \mathbf{X} that is full, partial or corrupted by noise. We note that $\boldsymbol{\mu}$ and \mathbf{X} must be able to correspond to both continuous or discontinuous fields over the domain of interest (see Figure 9) for both the forward and the inverse problem.

A summary of the proposed framework built upon the offline-online paradigm is presented in Figure 10. The first step of the offline stage represents the initialization of a training set of parameters ($\boldsymbol{\mu}$) of cardinality M (colored in blue in Figure 10). These coefficients $\boldsymbol{\mu}$ could represent physical properties, geometric characteristics, or boundary conditions. However, in this work, we focus on using $\boldsymbol{\mu}$ to represent collections of spatially heterogeneous scalar coefficients, defining the physical characteristics of complex microstructures.

In the second step (colored in green in Figure 10), we query the FOM, which can provide a solution in a finite-dimensional setting, for each parameter $\boldsymbol{\mu}$ in the training set. The FOM is used to approximate primary variables \mathbf{X}_h , which could correspond to material displacement and fluid pressure fields given the field of parameters $\boldsymbol{\mu}$ as input. We note that \mathbf{X}_h is a finite-dimensional approximation of \mathbf{X} corresponding to the FOM results, which our model takes as the ground truth. Even though the proposed framework could be applied to a wide range of PDEs, here, we will focus on the steady-state solution of the linear poroelasticity equations coupling solid deformation and fluid diffusion in porous media with highly heterogeneous permeability. For the FOM, we use the Discontinuous Galerkin finite element solver, discussed in S2, as required to handle the high degree of heterogeneity of the system.

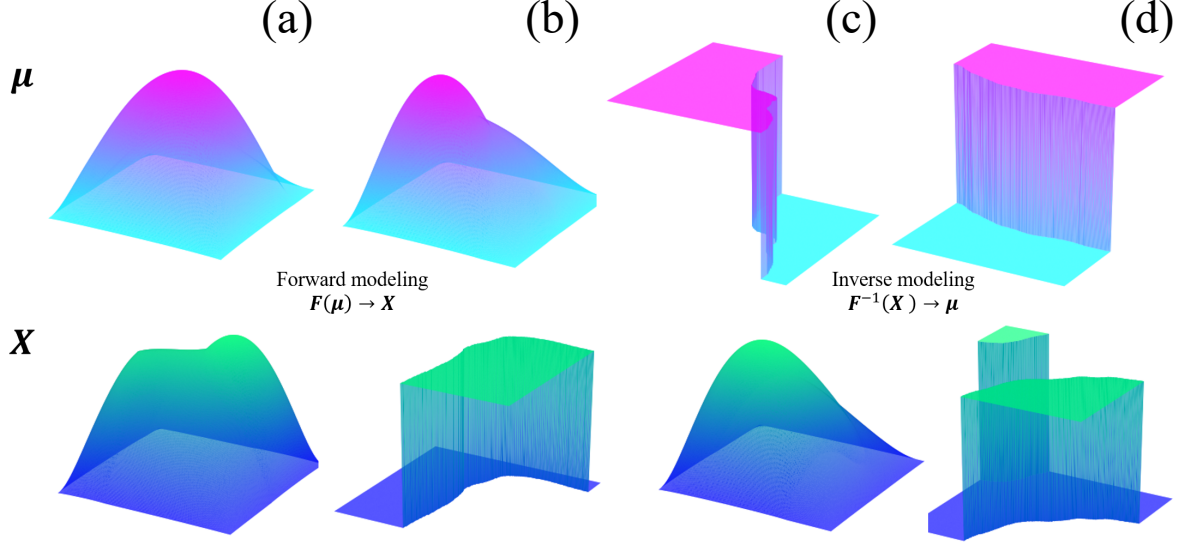


Figure 9: Main goals of data-driven model reduction framework of PDEs: (a) μ and X are continuous, (b) μ is continuous, but X is discontinuous, (c) μ is discontinuous, but X is continuous, and (d) μ and X are discontinuous.

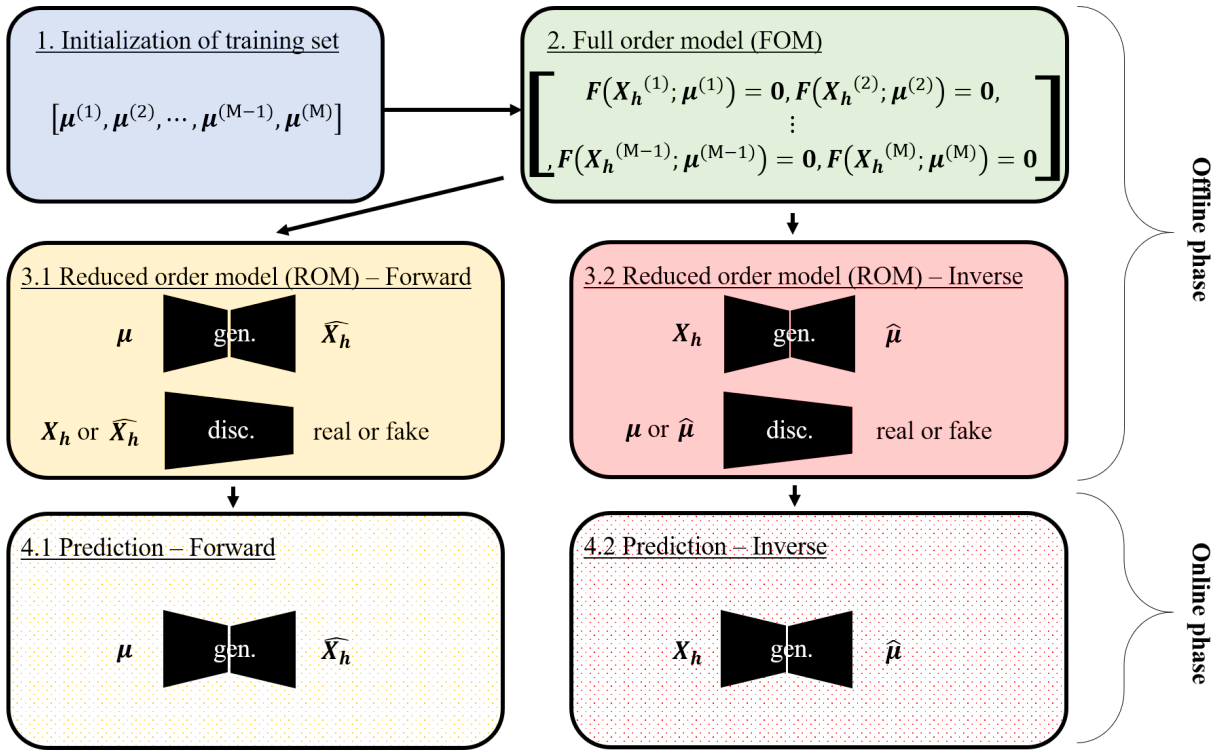


Figure 10: Summary of data-driven model reduction framework of PDEs. Gen. represents generator, and disc is discriminator. We note that X_h is an approximation of X obtaining from FOM, and \widehat{X}_h is an approximation of X_h obtaining from ROM in forward modeling setting. $\widehat{\mu}$ is an estimation of μ using ROM in inverse modeling setting.

In the third step, we build the ROM based on the cGAN image-to-image translation framework, which is discussed in detail in S3. We compare the performance of three different variants of the proposed ROM (modifying the architecture and loss function) with respect to both accuracy and computational costs. These variants of the model are extensively discussed in S3.1, S3.2, and S3.3. The training of the ROM must be adapted depending on whether it is to be used

in forward or inverse modeling. For the forward setting (colored in yellow in Figure 10), the input to the generator is μ , and the output is \widehat{X}_h . The input to the discriminator is X_h or \widehat{X}_h , and the output quantifies the proximity of the generated data to the real data. For the inverse modeling (colored in red in Figure 10), the input to the generator is X_h , and the output is $\widehat{\mu}$. The input to the discriminator is X_h or $\widehat{\mu}$, and the output again quantifies the proximity of the generated data to the real data.

Finally, during the online phase for the forward modeling (colored in dotted yellow in Figure 10), we used the trained generator to predict \widehat{X}_h with given μ . For the inverse modeling (colored in dotted red in Figure 10), the trained generator is used to estimate $\widehat{\mu}$ with given X_h . In practice, the input field for the inverse model might be incomplete because the measurement points are limited and are located sparsely. Moreover, the input might also be contaminated with noise due to uncertainty in experimental measurement. Hence, the ROM in the inverse setting is tested with incomplete and noisy data.

5 Acknowledgments

DO acknowledges support from Los Alamos National Laboratory’s Laboratory Directed Research and Development Early Career Award (20200575ECR). HV is grateful to the funding support from the U.S. Department of Energy (DOE) Basic Energy Sciences (LANLE3W1). NB acknowledges startup funds from Cornell University. YC acknowledges LDRD funds (21-FS-042) from Lawrence Livermore National Laboratory. Lawrence Livermore National Laboratory is operated by Lawrence Livermore National Security, LLC, for the U.S. Department of Energy, National Nuclear Security Administration under Contract DE-AC52-07NA27344 (LLNL-JRNL-823007).

6 CRediT authorship contribution statement

T. Kadeethum: Conceptualization, Formal analysis, Software, Validation, Writing - original draft, Writing - review & editing. **D. O’Malley:** Conceptualization, Formal analysis, Supervision, Validation, Writing - review & editing. **J.N. Fuhs:** Software, Validation, Writing - review & editing. **Y. Choi:** Conceptualization, Formal analysis, Supervision, Validation, Writing - review & editing. **J. Lee:** Software, Formal analysis, Supervision, Writing - review & editing. **H.S. Viswanathan:** Conceptualization, Supervision, Writing - review & editing. **N. Bouklas:** Conceptualization, Formal analysis, Funding acquisition, Supervision, Writing - review & editing.

7 Competing interests

The authors declare no competing interests

S1 Supplementary information

S2 Coupled hydro-mechanical processes in porous media

Even though the reduced order model presented in this manuscript (Section S3) could be applied to any types of differential equations, we here focus on coupled hydro-mechanical (HM) processes in porous media, in which fluid flow and solid deformation tightly interact. These coupled multiphysics processes are involved in various problems ranging from groundwater and contaminant hydrology to biomedical engineering [2, 3, 5, 69, 83–88]. We follow the Biot's formulation for linear poroelasticity [7, 89]. Note that although linear poroelasticity theory may oversimplify deformations in soft porous materials such as soils [90–93], this description is reasonably good for stiff materials such as rocks, which are the focus of this work. The theory of linear poroelasticity theory describes the HM problem through two coupled governing equations, namely linear momentum and mass balance equations.

Let $\Omega \subset \mathbb{R}^d$ ($d \in \{1, 2, 3\}$) denote the physical domain and $\partial\Omega$ denote its boundary. The time domain is denoted by $\mathbb{T} = (0, T]$ with $T > 0$. Primary variables used in this paper are $p : \Omega \times \mathbb{T} \rightarrow \mathbb{R}$, which is a scalar-valued fluid pressure (Pa) and $\mathbf{u} : \Omega \times \mathbb{T} \rightarrow \mathbb{R}^d$, which is a vector-valued displacement (m). The infinitesimal strain tensor $\boldsymbol{\varepsilon}$ is defined as

$$\boldsymbol{\varepsilon}(\mathbf{u}) := \frac{1}{2} (\nabla \mathbf{u} + (\nabla \mathbf{u})^\top), \quad (\text{S2.1})$$

Further, $\boldsymbol{\sigma}$ is the total Cauchy stress tensor, which may be related to the effective Cauchy stress tensor $\boldsymbol{\sigma}'$ and the pore pressure p as

$$\boldsymbol{\sigma}(\mathbf{u}, p) = \boldsymbol{\sigma}'(\mathbf{u}) - \alpha p \mathbf{I}. \quad (\text{S2.2})$$

Here, \mathbf{I} is the second-order identity tensor, and α is the Biot coefficient defined as [94]:

$$\alpha = 1 - \frac{K}{K_s}, \quad (\text{S2.3})$$

with K and K_s being the bulk moduli of the bulk porous material and the solid matrix, respectively. According to linear elasticity, the effective stress tensor relates to the infinitesimal strain tensor, and therefore to the displacement, through the following constitutive relationship, which can be written as

$$\boldsymbol{\sigma}'(\mathbf{u}) = \lambda_l \text{tr}(\boldsymbol{\varepsilon}(\mathbf{u})) \mathbf{I} + 2\mu_l \boldsymbol{\varepsilon}(\mathbf{u}). \quad (\text{S2.4})$$

where λ_l and μ_l are the Lamé constants, which are related to the bulk modulus K and the Poisson ratio ν of the porous solid as

$$\lambda_l = \frac{3K\nu}{1+\nu}, \quad \text{and} \quad \mu_l = \frac{3K(1-2\nu)}{2(1+\nu)}. \quad (\text{S2.5})$$

Under quasi-static conditions, the linear momentum balance equation can be written as

$$\nabla \cdot \boldsymbol{\sigma}(\mathbf{u}, p) + \mathbf{f} = \mathbf{0}, \quad (\text{S2.6})$$

where \mathbf{f} is the body force term defined as $\rho\phi\mathbf{g} + \rho_s(1-\phi)\mathbf{g}$, where ρ is the fluid density, ρ_s is the solid density, ϕ is the porosity, and \mathbf{g} is the gravitational acceleration vector. The gravitational force will be neglected in this study, but the body force term will be kept in the succeeding formulations for a more general case.

For this solid deformation problem, the domain boundary $\partial\Omega$ is assumed to be suitably decomposed into displacement and traction boundaries, $\partial\Omega_u$ and $\partial\Omega_t$, respectively. Then the linear momentum balance equation is supplemented by the boundary and initial conditions as:

$$\begin{aligned} \nabla \cdot \boldsymbol{\sigma}'(\mathbf{u}) - \alpha \nabla \cdot (p \mathbf{I}) + \mathbf{f} &= \mathbf{0} \quad \text{in } \Omega \times \mathbb{T}, \\ \mathbf{u} &= \mathbf{u}_D \quad \text{on } \partial\Omega_u \times \mathbb{T}, \\ \boldsymbol{\sigma}(\mathbf{u}) \cdot \mathbf{n} &= \mathbf{t}_D \quad \text{on } \partial\Omega_t \times \mathbb{T}, \\ \mathbf{u} &= \mathbf{u}_0 \quad \text{in } \Omega \text{ at } t = 0, \end{aligned} \quad (\text{S2.7})$$

where \mathbf{u}_D and \mathbf{t}_D are prescribed displacement and traction values at the boundaries, respectively, and \mathbf{n} is the unit normal vector to the boundary.

Next, the mass balance equation is given as [95, 96]:

$$\frac{1}{M} \frac{\partial p}{\partial t} + \alpha \frac{\partial \boldsymbol{\varepsilon}_v}{\partial t} + \nabla \cdot \mathbf{q} = g \quad \text{in } \Omega \times \mathbb{T}, \quad (\text{S2.8})$$

where

$$\frac{1}{M} = \left(\phi c_f + \frac{\alpha - \phi}{K_s} \right) \quad (\text{S2.9})$$

is the Biot modulus. Here, c_f is the fluid compressibility, $\varepsilon_v := \text{tr}(\boldsymbol{\varepsilon}) = \nabla \cdot \mathbf{u}$ is the volumetric strain, and g is a sink/source term. The superficial velocity vector \mathbf{q} is given by Darcy's law as

$$\mathbf{q} = -\boldsymbol{\kappa}(\nabla p - \rho \mathbf{g}). \quad (\text{S2.10})$$

Here $\boldsymbol{\kappa} = \frac{\mathbf{k}}{\mu_f}$ is the porous media conductivity, μ_f is the fluid viscosity. Again, the gravitational force, $\rho \mathbf{g}$, will be neglected in this work, without loss of generality. In addition, \mathbf{k} is the matrix permeability tensor defined as

$$\mathbf{k} := \begin{cases} \begin{bmatrix} k_{xx} & k_{xy} & k_{xz} \\ k_{yx} & k_{yy} & k_{yz} \\ k_{zx} & k_{zy} & k_{zz} \end{bmatrix} & \text{if } d = 3, \\ \begin{bmatrix} k_{xx} & k_{xy} \\ k_{yx} & k_{yy} \end{bmatrix} & \text{if } d = 2, \\ k_{xx} & \text{if } d = 1, \end{cases} \quad (\text{S2.11})$$

To simplify our problem, we assume all off-diagonal terms of (S2.11) to be zero and all diagonal terms have similar value. For the fluid flow problem, the domain boundary $\partial\Omega$ is also suitably decomposed into the pressure and flux boundaries, $\partial\Omega_p$ and $\partial\Omega_q$, respectively. In what follows, we apply the fixed stress split scheme [96, 97], assuming $(\sigma_v - \sigma_{v,0}) + \alpha(p - p_0) = K\varepsilon_v$. Then we write the fluid flow problem with boundary and initial conditions as

$$\begin{aligned} \left(\frac{1}{M} + \frac{\alpha^2}{K} \right) \frac{\partial p}{\partial t} + \frac{\alpha}{K} \frac{\partial \sigma_v}{\partial t} - \boldsymbol{\kappa} \nabla p &= g \quad \text{in } \Omega \times \mathbb{T}, \\ p &= p_D \quad \text{on } \partial\Omega_p \times \mathbb{T}, \\ -\boldsymbol{\kappa} \nabla p \cdot \mathbf{n} &= q_D \quad \text{on } \partial\Omega_q \times \mathbb{T}, \\ p &= p_0 \quad \text{in } \Omega \text{ at } t = 0, \end{aligned} \quad (\text{S2.12})$$

where $\sigma_v := \frac{1}{3} \text{tr}(\boldsymbol{\sigma})$ is the volumetric stress, and p_D and q_D are the given boundary pressure and flux, respectively.

We utilize the discontinuous Galerkin finite element model of linear poroelasticity developed in [69, 70, 98] for this study. To simplify the process, we only investigate the pressure and displacement fields at the steady-state solutions and use these solutions to train our ROM model. The mesh and boundary conditions adapted from [99] are presented in Fig. S2.1. We take $\Omega = (0, 1)^2$ corresponding to a square domain of 1m^2 area, and decompose its boundary $\partial\Omega$ with the following labels

$$\begin{aligned} \text{Left} &= \{0\} \times [0, 1], \\ \text{Top} &= [0, 1] \times \{1\}, \\ \text{Right} &= \{1\} \times [0, 1], \\ \text{Bottom} &= [0, 1] \times \{0\}. \end{aligned} \quad (\text{S2.13})$$

Throughout this study, the boundary conditions are described as follows

$$\begin{aligned} \mathbf{u}_D \cdot \mathbf{n} &= 0 \quad \text{m on Left} \times \mathbb{T}, \\ t_D &= [0, -1] \quad \text{kPa on Top} \times \mathbb{T}, \\ \mathbf{u}_D \cdot \mathbf{n} &= 0 \quad \text{m on Right} \times \mathbb{T}, \\ \mathbf{u}_D \cdot \mathbf{n} &= 0 \quad \text{m on Bottom} \times \mathbb{T}, \end{aligned} \quad (\text{S2.14})$$

for (S2.7), and

$$\begin{aligned} q_D &= 0 \quad \text{m/s} \quad \text{on} \quad \text{Left} \times \mathbb{T}, \\ p_D &= 0 \quad \text{Pa} \quad \text{on} \quad \text{Top} \times \mathbb{T}, \\ q_D &= 0 \quad \text{m/s} \quad \text{on} \quad \text{Right} \times \mathbb{T}, \\ p_D &= 1000 \quad \text{Pa} \quad \text{on} \quad \text{Bottom} \times \mathbb{T}, \end{aligned} \tag{S2.15}$$

for (S2.12). The degrees of freedom associated with this mesh are 9722 for the continuous approximation of the displacement field \mathbf{u} , and 7110 for the discontinuous approximation of the pressure p .

The following set of input parameters are used throughout this study. We fix $\alpha \approx 1$, as the porous matrix is characterized by $K = 1000$ kPa while the bulk solid is modeled by $K_s \rightarrow \infty$ kPa, $c_f = 1.0 \times 10^{-10}$ Pa $^{-1}$, $\phi = 0.2$, fluid viscosity - $\mu_f = 10^{-3}$ Pa.s, and Poisson ratio $\nu = 0.25$. The permeability in x-direction (k_{xx}) is uniquely populated for each simulation by [100]. Then the permeability field (\mathbf{k}) is set as

$$\mathbf{k} := \begin{bmatrix} k_{xx} & 0.0 \\ 0.0 & k_{xx} \end{bmatrix} \tag{S2.16}$$

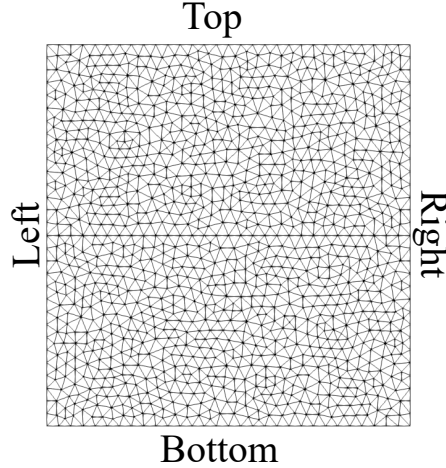


Figure S2.1: Domain, its boundaries, and mesh used for all numerical examples.

As porous media are generally heterogeneous, and the sharp contrast of phases of the porous microstructure leads to discontinuous material properties that can span several orders of magnitude, it is very challenging and time-consuming to capture multiphysics processes occurred in these media using finite volume or finite element methods [30, 32, 70, 83, 101–105]. Hence, we aim to reduce this problem’s computational cost while maintaining the acceptable accuracy through a generative model presented in Section S3.

Remark 1. We want to emphasize that we use the discontinuous Galerkin finite element model of linear poroelasticity developed in [69, 70, 98, 99] to generate training and test examples. Each FEM simulation uses the same set of input parameters except the permeability field (\mathbf{k}), which is uniquely generated for each simulation by [100]. Besides, to simplify the process, we only employ the steady-state solutions in this study. Time-dependent problems will be incorporated in our subsequent studies.

S3 Conditional generative adversarial network

We propose a data-driven model order reduction for physics-based problems using conditional generative adversarial network (cGAN) [51–53]. The framework is adapted from the idea of paired image-to-image translation developed in [55, 106]. The word ‘paired’ reflects the fact that there is a one-to-one mapping between input fields and output fields. In the past several years, this technique and its variations have been applied to many problems both supervised [56–58] and unsupervised learning [59–61].

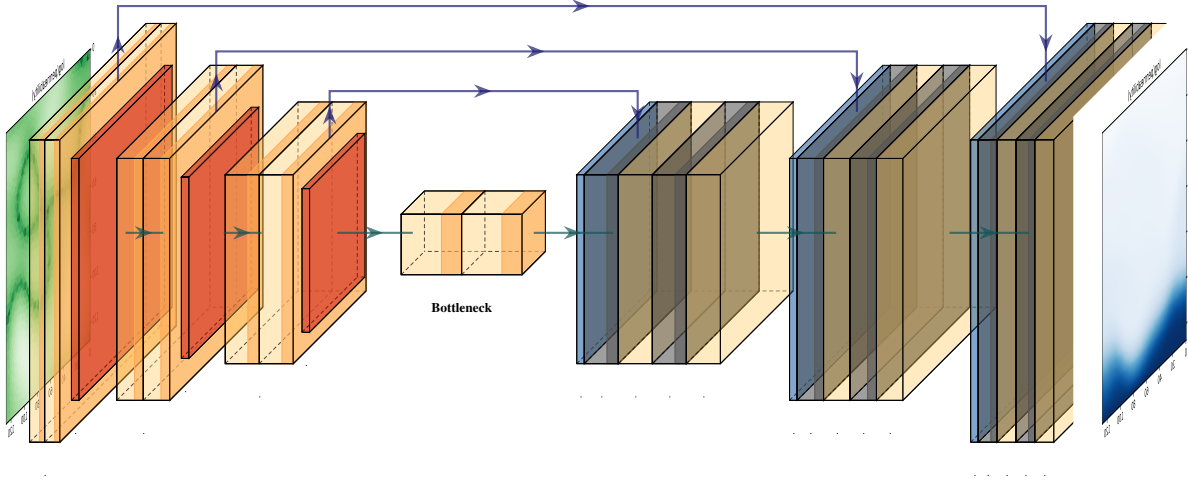


Figure S3.1: Generator: U-net visualized using PlotNeuralNet: <https://github.com/HarisIqbal88/PlotNeuralNet>

In contrast to the classical cGAN in which noise and class vectors are used as an input to the generator [52], the image-to-image translation framework employs a conditional matrix (or a conditional field) as its input [55]. This conditional field could be edges, masks, points, or segmentation maps in typical image analysis studies. These fields then are mapped to the output, which is an image. [56–61, 106]. This study, however, uses physics fields that could represent material properties, boundary and initial conditions, and measurable quantities such as pressure, temperature, or displacements as the generator’s input and output fields.

The generator used in this framework is illustrated in Fig. S3.1, and it resembled the well-established architecture of U-net, which typically is used for image segmentation [64]. The first component of the generator is a contracting block that performs two convolutions followed by a max pool operation (see Fig. S3.1 - light and dark orange blocks). The second component is an expanding block (see Fig. S3.1 - blue and pale yellow) in which it performs an upsampling, a convolution, a concatenation of its two inputs. The generator used in this study consists of six contracting and six expanding blocks.

We provide a detailed architecture of the generator used in this study in Table S3.1. Note that each contracting block uses LeakyReLU with a negative slope of 0.2 as its activation function, each expanding block uses ReLU as its activation function, the 1st convolutional layer is used to map input channel (C_{in}) to hidden layer size (H), and it does not subject to any activation function, and the 2nd convolutional layer is used to map hidden layer size (H) to output channel (C_{out}), and it subjects to the Sigmoid activation function. For the generator, $C_{in} = C_{out} = C$, and $H = 32$. The domain size is governed by $DOF_x \times DOF_y$, which is 128×128 throughout this manuscript. We note that B is batch size.

Both input or conditional (I) and output or real (O) fields of the generator are normalized to be in a range of $[0, 1]$ as follows:

$$I_i = \frac{I_i - \min(I)}{\max(I) - \min(I)} \quad \text{and} \quad O_i = \frac{O_i - \min(O)}{\max(O) - \min(O)}. \quad (S3.1)$$

where i represents each member’s index in the sets of I and O . We also note that I and O have M members (as well as the approximated output or fake fields (\hat{O})). As the last layer of the generator 2nd convolutional layer (see Table S3.1), is subjected to the Sigmoid activation function, the \hat{O} is always in a range of 0 to 1.

In the classical cGAN, input to the discriminator is a concatenation of conditional vector (e.g., $[0, 1, \dots, 8, 9]$ for MNIST dataset) and output field produced from the generator or a real field (training/testing set) [52]. On the contrary, the image-to-image translation framework uses a combination of conditional and output fields produced from the generator or a real field (training/testing set) as its input [55]. The schematic of the patch discriminator is shown in Fig. S3.2. The discriminator utilizes the contracting block (see Fig. S3.2 - light and dark orange blocks), which is also used in the generator. The purple block is used to illustrate that our discriminator output is not a single value, but a matrix [65].

Table S3.1: Generator: U-net's detail used in this study (input and output sizes are represented by $[\mathbb{B}, \mathbb{C}, \text{DOF}_x, \text{DOF}_y]$. We use hidden layers $\mathbb{H} = 32$)

Block	Input size	Output size	Batch normalization	Dropout
1 st convolutional layer	$[\mathbb{B}, \mathbb{C}, 128, 128]$	$[\mathbb{B}, 32, 128, 128]$		
1 st contracting block	$[\mathbb{B}, 32, 128, 128]$	$[\mathbb{B}, 64, 64, 64]$	✓	✓
2 nd contracting block	$[\mathbb{B}, 64, 64, 64]$	$[\mathbb{B}, 128, 32, 32]$	✓	✓
3 rd contracting block	$[\mathbb{B}, 128, 32, 32]$	$[\mathbb{B}, 256, 16, 16]$	✓	✓
4 th contracting block	$[\mathbb{B}, 256, 16, 16]$	$[\mathbb{B}, 512, 8, 8]$	✓	
5 th contracting block	$[\mathbb{B}, 512, 8, 8]$	$[\mathbb{B}, 1024, 4, 4]$	✓	
6 th contracting block	$[\mathbb{B}, 1024, 4, 4]$	$[\mathbb{B}, 2048, 2, 2]$	✓	
1 st expanding block	$[\mathbb{B}, 2048, 2, 2]$	$[\mathbb{B}, 1024, 4, 4]$	✓	
2 nd expanding block	$[\mathbb{B}, 1024, 4, 4]$	$[\mathbb{B}, 512, 8, 8]$	✓	
3 rd expanding block	$[\mathbb{B}, 512, 8, 8]$	$[\mathbb{B}, 256, 16, 16]$	✓	
4 th expanding block	$[\mathbb{B}, 256, 16, 16]$	$[\mathbb{B}, 128, 32, 32]$	✓	
5 th expanding block	$[\mathbb{B}, 128, 32, 32]$	$[\mathbb{B}, 64, 64, 64]$	✓	
6 th expanding block	$[\mathbb{B}, 64, 64, 64]$	$[\mathbb{B}, 32, 128, 128]$	✓	
2 nd convolutional layer	$[\mathbb{B}, 32, 128, 128]$	$[\mathbb{B}, \mathbb{C}, 128, 128]$		

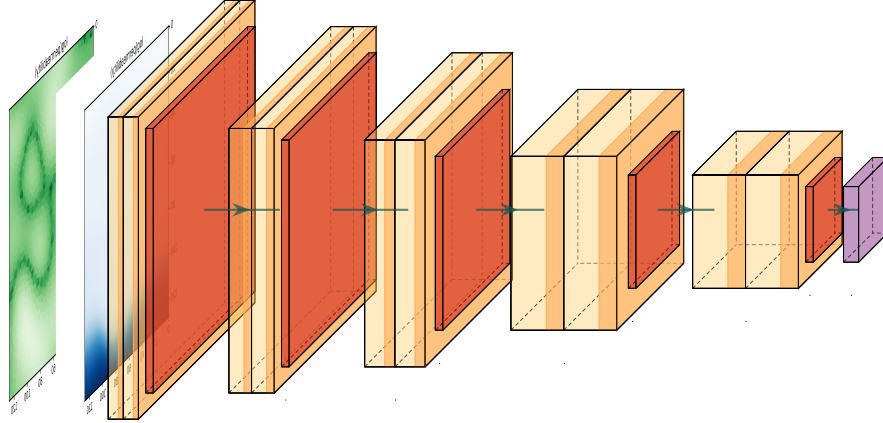


Figure S3.2: Discriminator: PatchGAN visualized using PlotNeuralNet: <https://github.com/HarisIqbal88/PlotNeuralNet>

The detail of patch discriminator architecture is shown in Table S3.2. Each contracting block (see Fig. S3.2 - light and dark orange blocks) uses LeakyReLU with a negative slope of 0.2 as its activation function, the 1st convolutional layer is used to map $\mathbb{C} + \text{conditional field}$ to $\mathbb{H} = 8$, and it does not subject to any activation function, and the 2nd convolutional layer is used to map $\mathbb{H} = 8$ to \mathbb{C} . Dissimilar to the generator where the input size is equal to output size $\text{DOF}_x \times \text{DOF}_y = 128 \times 128$, the discriminator input size is $\text{DOF}_x \times \text{DOF}_y = 128 \times 128$, while the output size is a patch matrix of size $\text{PATCH}_x \times \text{PATCH}_y = 8 \times 8$. We note that we have tried different generator and discriminator architectures such as fully connected layers, deep convolutional networks, or U-Net without residual blocks. We select these generator and discriminator architectures, Figs. S3.1 and S3.2 because they provide the best performance based on our trial-and-error process.

We note that the input of the discriminator is combined fields of input (I) and output (O) fields of the generator; hence, they are normalized by (S3.1) and range from 0 to 1. In this study, we compare three different models' performance in both accuracy and computational costs. The detail of each model is provided in the following sections.

Table S3.2: Discriminator: Patch discriminator’s detail used in this study (input size is represented by $[\mathbb{B}, \mathbb{C}, \text{DOF}_x, \text{DOF}_y]$, and output size is represented by $[\mathbb{B}, \mathbb{C}, \text{PATCH}_x, \text{PATCH}_y]$. We use hidden layers $\mathbb{H} = 8$). Note that the spectral normalization is applied only for the SN model (see Section S3.2).

Block	Input size	Output size	Batch normalization	Dropout	Spectral normalization
1 st convolutional layer	$[\mathbb{B}, \mathbb{C}+1, 128, 128]$	$[\mathbb{B}, 8, 128, 128]$			✓
1 st contracting block	$[\mathbb{B}, 8, 128, 128]$	$[\mathbb{B}, 16, 64, 64]$			✓
2 nd contracting block	$[\mathbb{B}, 16, 64, 64]$	$[\mathbb{B}, 32, 32, 32]$	✓		✓
3 rd contracting block	$[\mathbb{B}, 32, 32, 32]$	$[\mathbb{B}, 64, 16, 16]$	✓		✓
4 th contracting block	$[\mathbb{B}, 64, 16, 16]$	$[\mathbb{B}, 128, 8, 8]$	✓		✓
2 nd convolutional layer	$[\mathbb{B}, 128, 8, 8]$	$[\mathbb{B}, \mathbb{C}, 8, 8]$			✓

S3.1 Base model

As previously discussed, the cGAN model is generally composed of one generator and one discriminator (see Figs. S3.1 and S3.2). The goal of the generator is to generate an approximated output field ($\hat{\mathbf{O}}$) that looks realistic and similar to a real output field (\mathbf{O}) with a given conditional field (\mathbf{I}). The discriminator, on the other hand, tries to differentiate between $\hat{\mathbf{O}}$ and \mathbf{O} . Consequently, to train the framework (both generator and discriminator), we equivalently try to solve the following constraint

$$\min_G \max_D [\ell_a + \lambda_r \ell_r]. \quad (\text{S3.2})$$

Here, ℓ_a is an adversarial loss defined as [44, 45]

$$\ell_a = \frac{1}{\mathbb{B}} \sum_{i=1}^{\mathbb{B}} \left(O_i \cdot \log \hat{O}_i + (1 - O_i) \cdot \log (1 - \hat{O}_i) \right), \quad (\text{S3.3})$$

where, again, \mathbb{B} is batch size and $\mathbb{B} \leq M$, which is the total number of training examples. The λ_r is a penalty constant selected by users, and we set $\lambda_r = 500$ throughout this paper. The ℓ_r is read

$$\ell_r = \frac{1}{\mathbb{B}} \sum_{i=1}^{\mathbb{B}} \left| \hat{O}_i - O_i \right|. \quad (\text{S3.4})$$

We choose L1 norm (i.e., (S3.4)) over L2 norm because it has shown to improve GAN performance [107]. We would like to emphasize that according to (S3.2) and (S3.3), the discriminator’s goal is to maximize $\frac{1}{\mathbb{B}} \sum_{i=1}^{\mathbb{B}} \left(O_i \cdot \log \hat{O}_i + (1 - O_i) \cdot \log (1 - \hat{O}_i) \right)$, which equivalently means that it tries to differentiate between $\hat{\mathbf{O}}$ and \mathbf{O} . The generator of the base model, on the other hand, is not affected by $O_i \cdot \log \hat{O}_i$ term in (S3.3). Hence, the goal of the generator is to minimize $\frac{1}{\mathbb{B}} \sum_{i=1}^{\mathbb{B}} (1 - O_i) \cdot \log (1 - \hat{O}_i)$ and $\frac{1}{\mathbb{B}} \sum_{i=1}^{\mathbb{B}} \left| \hat{O}_i - O_i \right|$. By satisfying these two constraints, the generator attempts to produce realistic $\hat{\mathbf{O}}$.

We use the adaptive moment estimation (ADAM) algorithm [108] to train the framework (both generator and discriminator). The learning rate (η) is calculated as [109]

$$\eta_c = \eta_{\min} + \frac{1}{2} (\eta_{\max} - \eta_{\min}) \left(1 + \cos \left(\frac{\text{step}_c}{\text{step}_f} \pi \right) \right) \quad (\text{S3.5})$$

where η_c is a learning rate at step step_c , η_{\min} is the minimum learning rate, which is set as 1×10^{-16} , η_{\max} is the maximum or initial learning rate, which is selected as 1×10^{-4} , step_c is the current step, and step_f is the final step. We note that each step refers to each time we perform back-propagation, including updating both generator and discriminator’s parameters. In certain circumstances, one could do multiple rounds of back-propagating on the generator while updating the discriminator only once and vice versa to achieve more stable training. However, in this study, we update the generator and discriminator’s parameters one time per step.

S3.2 Spectral normalization generative adversarial networks (SN model)

Studies show that employing the loss terms and architecture in the base model in Section S3.1 may result in unstable training (e.g., mode collapse or vanishing gradient problem), which could lead to the early ceasing of the GAN learning process [66–68]. Therefore, we adapt the concept of spectral normalization proposed in [66]. This spectral normalization enforces the constraint in the discriminator that the weights matrix exhibits Lipschitz continuity (Euclidean norm of discriminator’s gradient is at most one), which could provide a certain level of functional smoothness in all directions during the training process. The spectral normalization is applied to weight matrices (\mathbf{W}) resulting in (see [66, 110])

$$\mathbf{W}_{\text{SN}} = \frac{\mathbf{W}}{\text{SN}(\mathbf{W})}. \quad (\text{S3.6})$$

We apply spectral normalization only to the discriminator as shown in Table S3.2. The loss terms for both generator and discriminator are similar to the base model (see (S3.2)).

S3.3 Wasserstein generative adversarial networks (W model)

To prevent the GAN model from mode collapse or vanishing gradient problem, we could also use the Wasserstein loss or W loss [67, 68]. The W loss employs the Earth mover’s distance, enforcing the distribution of the generator’s output to be similar to that of the actual distribution [67, 111]. Dissimilar to the previous models, the W model uses the following constraint

$$\min_G \max_D [\ell_a + \lambda_r \ell_r + \lambda_p \wp_p]. \quad (\text{S3.7})$$

where ℓ_a in this model is the Earth mover’s distance defined as

$$\ell_a = \mathbb{E}(D(\mathbf{I} + \mathbf{O})) - \mathbb{E}(D(\mathbf{I} + \hat{\mathbf{O}})). \quad (\text{S3.8})$$

Here, $D(\mathbf{I} + \mathbf{O})$ is an output matrix from the patch discriminator (see Fig. S3.2) using a real output (\mathbf{O}); $D(\mathbf{I} + \hat{\mathbf{O}})$, on the other hand, is an output matrix from the patch discriminator using an approximated output $\hat{\mathbf{O}}$ produced by the generator ($G(\mathbf{I}) = \hat{\mathbf{O}}$); $\mathbb{E}(\cdot)$ denotes an expectation; λ_p denotes a gradient penalty constant that we set to 10 throughout this study; \wp_p gradient penalty regularization; \wp_p is used to enforce Lipschitz continuity of weight matrices (\mathbf{W}), i.e., Euclidean norm of discriminator’s gradient is at most one.

We note that there are two common ways of ensuring Lipschitz continuity of \mathbf{W} ; weight clipping and gradient penalty [67, 68]. With weight clipping, \mathbf{W} of the discriminator is forced to take values between a fixed interval, which means that weights over that interval, either too high or too low, will be set to the maximum or the minimum amount allowed. One of the disadvantages is forcing the discriminator’s \mathbf{W} to a limited range of values could limit the discriminator’s ability to learn and ultimately resulting in an early termination of the learning process [68].

The gradient penalty, another method, is a much softer way to enforce the discriminator to be Lipschitz continuous. The gradient penalty adds a regularization term ($\lambda_p \wp_p$) to the loss function (S3.7). This term penalizes the discriminator when its gradient norm is higher than one by

$$\wp_p = \mathbb{E}(\|\nabla D(\mathbf{I} + \bar{\mathbf{O}})\|_2 - 1)^2 \quad (\text{S3.9})$$

where $\bar{\mathbf{O}}$ is a mixing between $\hat{\mathbf{O}}$ and \mathbf{O} , which is defined by

$$\bar{\mathbf{O}} = \epsilon \mathbf{O} + (1 - \epsilon) \hat{\mathbf{O}}. \quad (\text{S3.10})$$

We randomly select $\epsilon_i \in \epsilon$ for each $\bar{\mathbf{O}}_i$ from a uniform distribution on the interval of $[0, 1)$.

Remark 2. We note that in some literature, the ‘discriminator’ of Wasserstein generative adversarial networks or W model in this paper is referred to as ‘critic’ since its output is not bounded by 0 or 1 [67, 68]. Throughout this paper, however, we refer the ‘critic’ to ‘discriminator’ for the sake of simplicity.

Remark 3. Finite element results obtained from [69, 70, 98, 99] are based on unstructured grids. Hence, we pre-processing our data by interpolating the finite element results as well as permeability field populated by [100] to structured grids using cubic spline interpolation and use this processed data in our ROM framework.

S4 Supplementary proper orthogonal decomposition (POD) analysis

We perform proper orthogonal decomposition (POD) on the pressure field p_h obtained from FOM discussed in S2 using RBniCS project [112]. The decay of eigenvalues results of p_h using (1) permeability field as Gaussian distribution S5.1.1, (2) permeability field as bimodal transformation S5.1.2, and (3) permeability field as Zinn & Harvey transformation S5.1.3 are presented in Figure S4.1. From this figure, we could observe that the decay is very slow, which means POD, low-dimensional linear subspace, poorly approximates the p_h field. We note that the total available data is 10000. The result is based on a "global" POD approach and using "local" POD method [18] may improve a performance of POD.

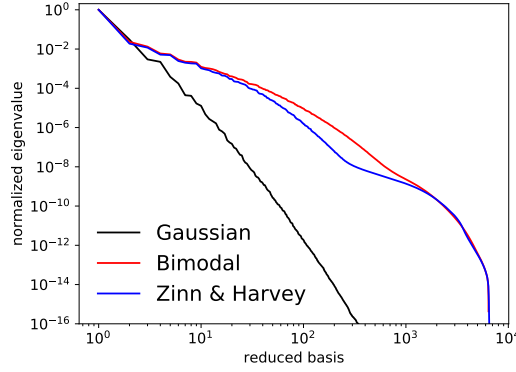


Figure S4.1: Normalized eigenvalue as a function of basis for fluid pressure field (p_h).

S5 Supplementary numerical examples

S5.1 Supplementary forward modeling

We utilize pressure and displacement fields at the steady-state solutions of discontinuous Galerkin finite element model of linear poroelasticity developed in [69, 70, 98, 99, 113] to train our ROM model. Throughout this section, we fix our framework parameters as follows; the model architecture is shown in Figs.S3.1 and S3.2 and discussed in Section S3. The input and output fields have a resolution of 128×128 . The input has one channel, but the output could have one, two, or three channels (i.e., pressure, the displacement in the x-direction, or the displacement in the y-direction). Note that we present here, the forward modeling part, only the results of pressure field p_h , for the sake of compactness. A very similar result is held for the displacement field ($[u_{x_h}, u_{y_h}] \in \mathbf{u}_h$) as well. The batch size is fixed at four as it has been shown that GAN or cGAN model generally requires a small batch size to improve model accuracy [114]. We use the adaptive moment estimation (ADAM) algorithm as an optimizer [108]. Its learning rate is 0.0001, and β is (0.5, 0.999). The L1 penalty coefficient (λ_r in (S3.2) or (S3.7)) is 500. For the W model, we use gradient penalty coefficient (λ_p in (S3.7)) of 10 as recommended by [68].

S5.1.1 Example 1: Permeability field as Gaussian distribution

This section explores the ROM model's behavior using permeability fields populated using an unconditional Gaussian distribution [100, 115]. The mean of log-permeability (base 10) field is $-12 \log(\text{m}^2)$, the variance is $1 \log(\text{m}^4)$, and the covariance kernel is Gaussian (i.e., Squared exponential) with the length-scale is 1.0 m. For the sake of brevity, all 'log' in this manuscript refers to 'log10.' We create 11000 pairs of permeability and primary variable fields. We use 10000 pairs to train the ROM model and test the model with 1000 cases. To reiterate, we have three types of model (1) base model, (2) SN model, and (3) W model as discussed in Section S3. The training losses of both generator and discriminator are presented in Fig. S5.1.

In Fig. S5.1, each step refers to each time we perform back-propagation including updating both generator and discriminator's parameters. Comparing Figs. S5.1a-c, we could observe that the generator loss of the W model behaves more stable than those of the base and SN models. Besides, there is no significant difference between the base and SN models. We also observe that the W model has better stability in training than the base and SN models for the discriminator losses. Its loss value reaches zero slower than those of the base and SN models (i.e., the learning process ceases later [67, 68])

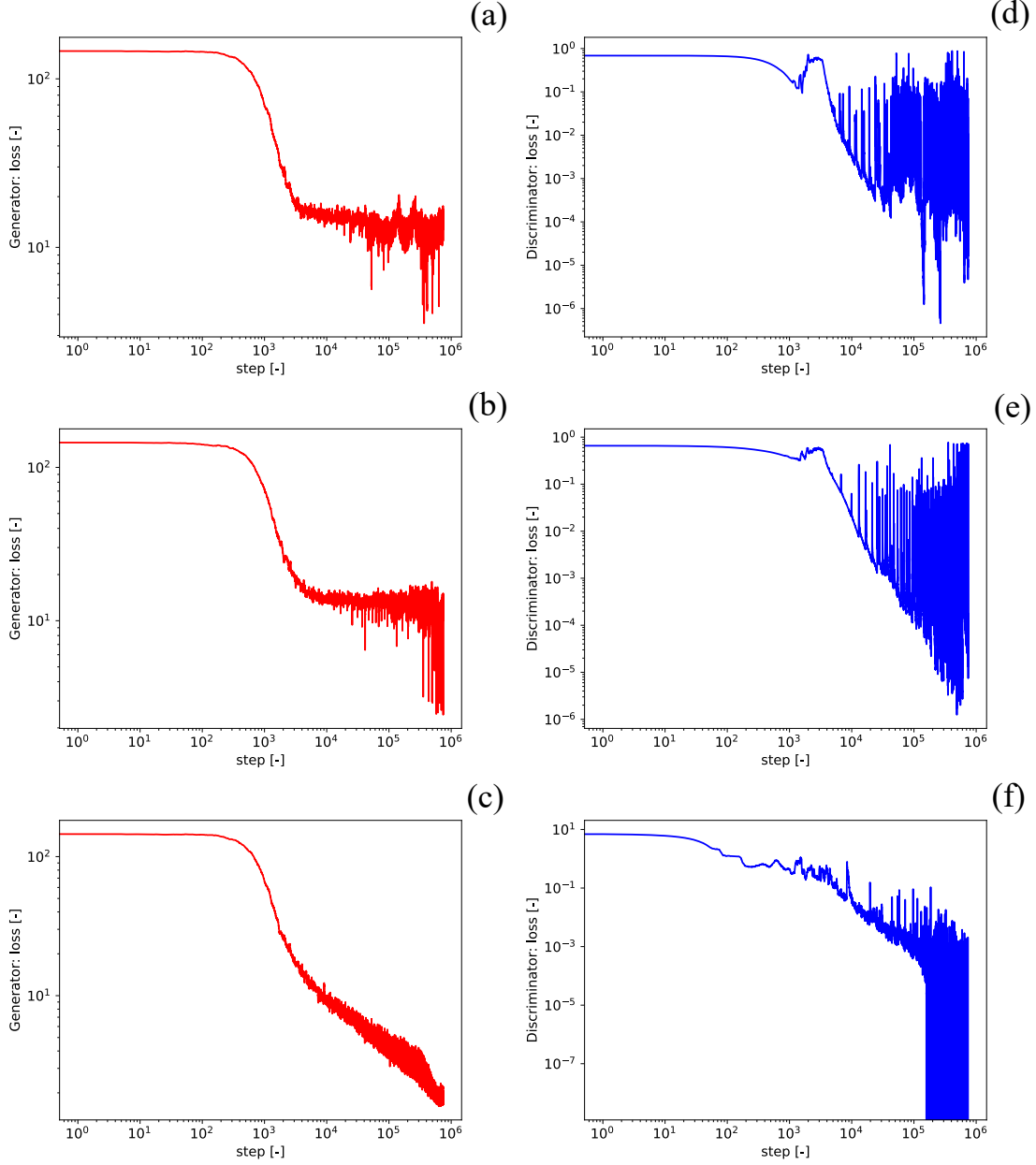


Figure S5.1: Example 1: 100th moving average of Generator loss of (a) base model, (b) SN model, and (c) W model and Discriminator loss of (d) base model, (e) SN model, and (f) W model

We then test our model with 1000 testing examples. Three of those cases are presented in Fig. S5.2. From this figure, we observe that our models could provide a decent approximation of the pressure field. The range of pressure values is from 0.0 to 1000.0 Pa; the DIFF values (1) are approximately in the range of 0.0 to 10.0 Pa, which means the errors are approximately 0.1%. Again, for the sake of compactness, we only show here the results of the pressure field, and the results of the displacement field are very similar.

The RMSE of the base, SN, and W models are shown in Fig. S5.3. We note that these RMSE values are calculated from 1000 test cases (i.e., $N = 1000$). From Fig. S5.3, one can observe that the RMSE values of the W model are the most stable, the ones from the SN model are a little bit less stable, while the results of the base model are not stable.

The best of an average value of RMSE is 10.35 Pa, which is about 1% of the maximum pressure. The maximum and minimum of RMSE values are approximately 30% and 0.4% of the maximum pressure, respectively. For the SN model,

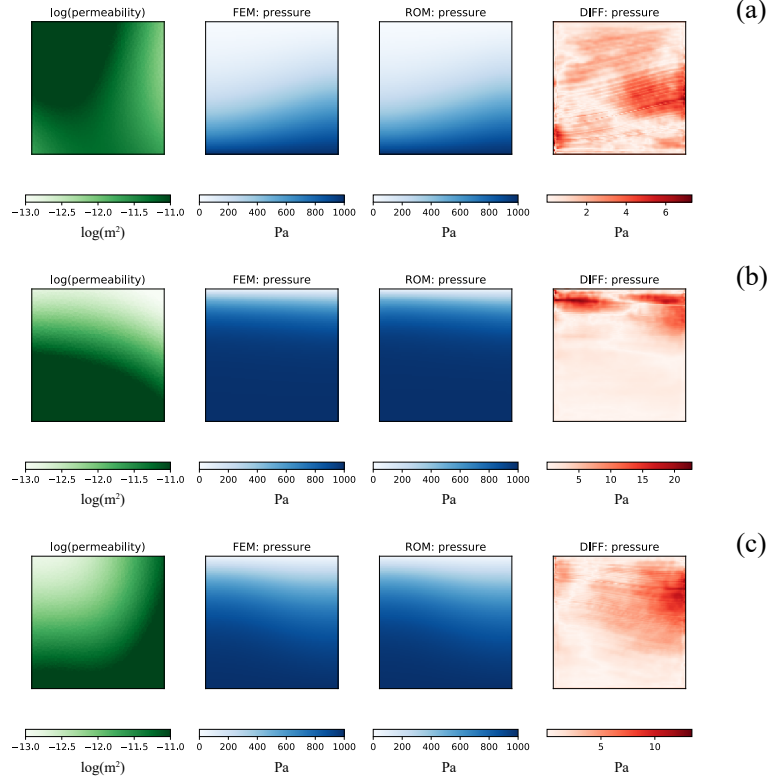


Figure S5.2: Example 1: test cases' results of the W model. Note that we train our three models (base, SN, and W models) using 10000 training examples and test them using 1000 test examples. These three cases shown here are randomly picked from 1000 test examples.

the best RMSE has an average of 5.19 Pa or 0.5% of the maximum pressure. The RMSE maximum and minimum of this model are about 20% and 0.3% of the maximum pressure, respectively. The W model's best RMSE has an average of 4.36 Pa, which is about 0.4% of the maximum pressure. The maximum and minimum RMSE values of the W model are approximately 15% and 0.1% of the maximum pressure, respectively.

S5.1.2 Example 2: Bimodal permeability fields

We then progress to a more challenging case of permeability fields that have a bimodal distribution. Again, we first populate 11000 realizations of a permeability field using a multivariate Gaussian distribution with a mean of \log_{10} of permeability field is $-12 \log(\text{m}^2)$, the variance is $1 \log(\text{m}^4)$, and the length-scale is 1.0 m. Subsequently, we transform the permeability fields using bimodal transformation [100]. Similar to the previous example, we use 10000 and 1000 examples for training and testing, respectively. The generator and discriminator losses' behaviors are illustrated in Fig. S5.4.

These behaviors of the generator and discriminator losses are slightly different from ones in Fig. S5.1. From Figs. S5.4a-c, we observe that the W model is the most stable one, and in this case, the SN model's generator loss is more stable than the one of the base model. From Figs. S5.4d-f, the W model's discriminator loss is decreased steadily, while there are no significant differences between the SN model and base model's discriminator losses. Again, the W model's discriminator loss approaches zero much later than those of the base and SN models, which means the learning process continues for a longer time [67, 68].

We present three of our test cases in Fig. S5.5. Comparing between Figs. S5.2 and S5.5, we observe that the results of the bimodal transformation example have higher errors (DIFF) as we expect. These higher errors stem from the fact that the permeability fields, in this case, contain sharper contrasts, which are harder to capture even in cases where finite element or finite volume methods are used [69, 116, 117].

The RMSE results of the three models, base model, SN model, and W model, at different steps are presented in Fig. S5.6. Similar to the previous example (Example 1), the W model provides the best accuracy and the most stable results

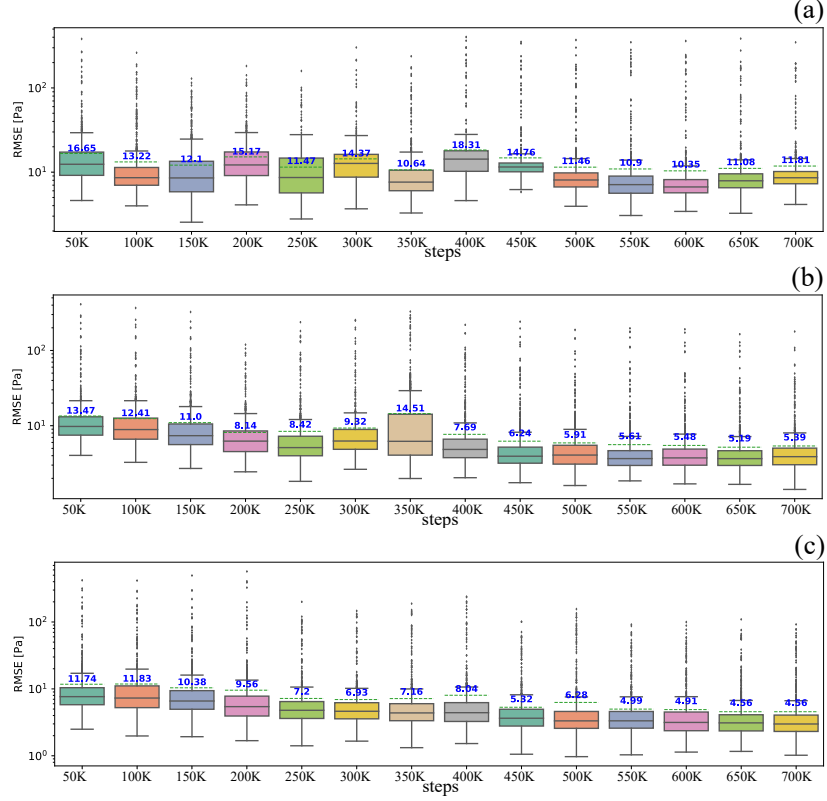


Figure S5.3: Example 1: Root Mean Square Error (RMSE) of (a) base model, (b) SN model, and (c) W model. Each step refers to each time we perform back-propagation including updating both generator and discriminator's parameters.

(i.e., the RMSE of the test cases decreases as the training progresses). Furthermore, the SN model yields better RMSE results than those of the base model. As expected, the RMSE results of the permeability field as bimodal transformation example are generally higher than those of the permeability field as Gaussian distribution example.

From Fig. S5.6, the base model has 0.06%, 1.45%, and 8.00% minimum, average, and maximum RMSE values, respectively. The SN model has 0.04%, 0.84%, and 6.00% minimum, average, and maximum RMSE values. The W model, which has the best performance, has 0.35%, 0.77%, and 6.00% minimum, average, and maximum RMSE values, respectively.

S5.1.3 Example 3: Permeability field as Zinn & Harvey transformation

For the third example, we move on to cases where there are high permeability contrasts and channels. Similar to two previous examples (Examples 1 and 2), we first populate 11000 realizations of a permeability field using an unconditional Gaussian distribution with a mean of \log_{10} of permeability field is $-12 \log(\text{m}^2)$, the variance is $1 \log(\text{m}^4)$, and the length-scale is 1.0 m. Then, we transform the generated permeability fields using Zinn & Harvey transformation [71, 100] that promotes the connectivity in the high values of the original Gaussian field. To elaborate, this transformation transforms a Gaussian random field realization to a random field with enhanced connectivity (high or low values of the original realization are connected) without changing the first and second moments (mean and covariance of the random variable). Again, we populate 11000 permeability fields and use 10000 and 1000 of those fields for training and testing, respectively. The generator and discriminator losses' behaviors are illustrated in Fig. S5.7.

Similar to two previous examples (Sections S5.1.1 and S5.1.2), the W model has the most stable training behavior, see Figs S5.7a-c. The SN model exhibits a better training behavior than the base model (i.e., its generator loss shows a longer decreasing trend.). For the discriminator's loss (see Figs S5.7d-f), the W model illustrates the longest training process similar to previous examples.

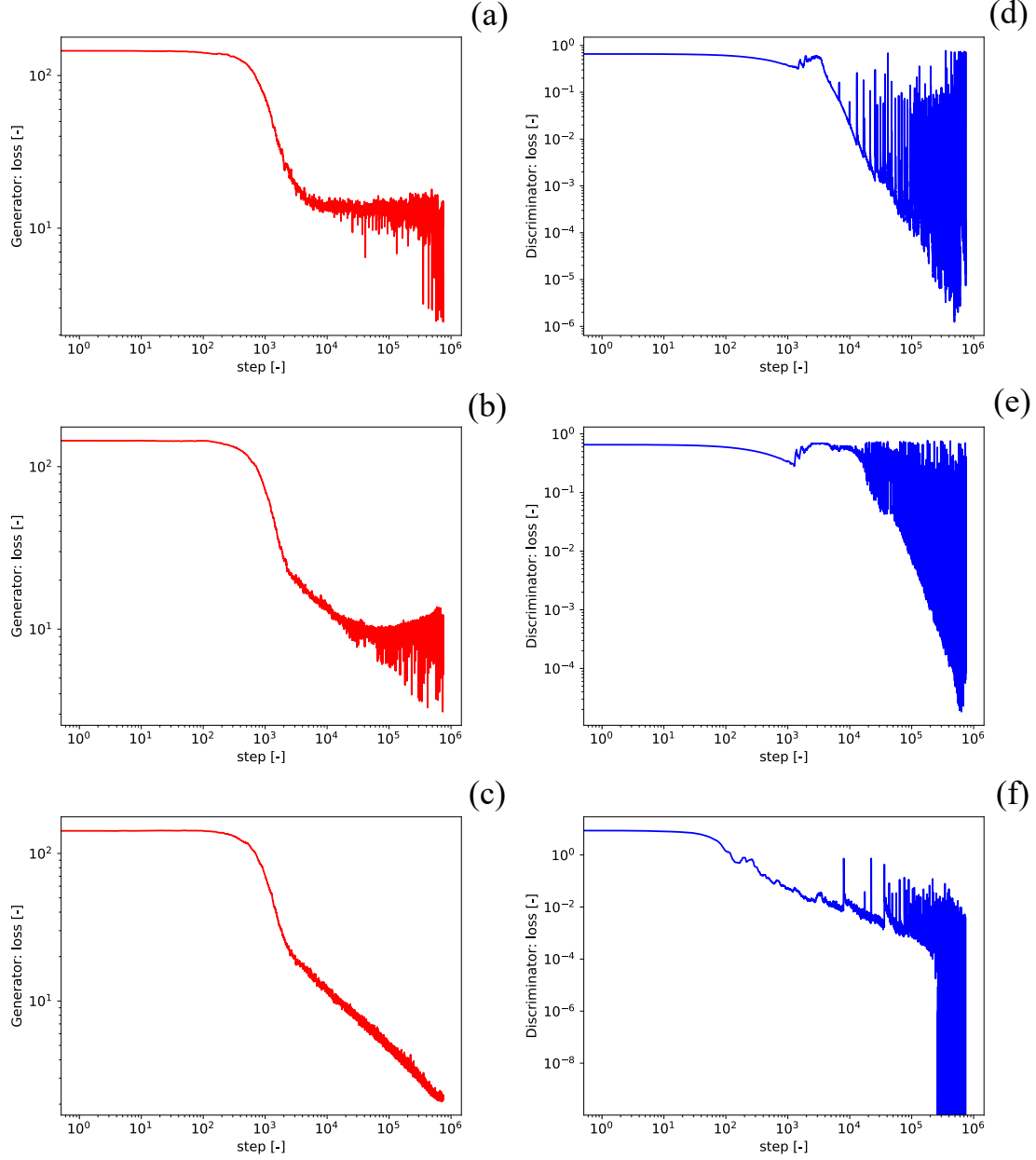


Figure S5.4: Example 2: 100th moving average of Generator loss of (a) base model, (b) SN model, and (c) W model and Discriminator loss of (d) base model, (e) SN model, and (f) W model

We present three examples of the test cases in Fig. S5.8. We observe that the ROM model could provide a decent approximation of the FEM result. As one would expect, this example's error (DIFF) is higher than the two previous examples. However, the maximum DIFF value is still two-order of magnitude less than the maximum pressure value.

The RMSE results of this example are presented in Fig. S5.9. Among three examples (Examples 1, 2, and 3), this example generally has the highest RMSE values for all three models (base, SN, and W models). Similar to two previous examples, the W model exhibits the most stable behavior (i.e., the RMSE average decreases as the training progress as well as the RMSE average is the lowest among the three models). The SN model typically has a better performance than the base model (see Fig. S5.9a-b).

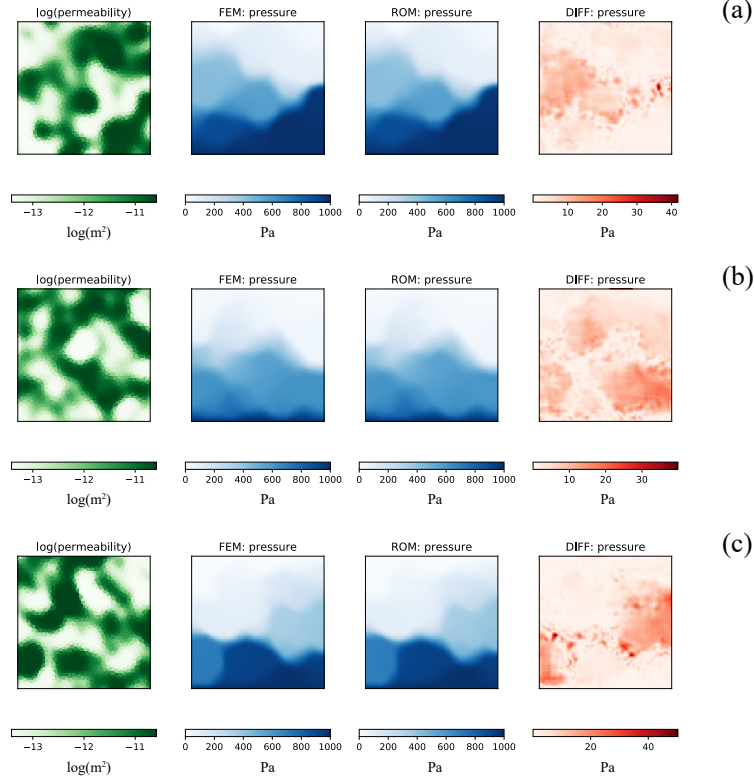


Figure S5.5: Example 2: test cases' results of the W model. Note that we train our three models (base, SN, and W models) using 10000 training examples and test them using 1000 test examples. These three cases shown here are randomly picked from 1000 test examples.

S5.2 Supplementary inverse modeling

As presented in the forward modeling sections, the W model exhibits the most stable training behavior as well as the testing accuracy; hence, throughout this section, we focus only on the results of the W model. Moreover, we use only Zinn & Harvey transformation permeability fields.

S5.2.1 Example 4: input and output fields have the same resolution

Throughout this example, the ROM's input and output fields have the same resolution (i.e., 128×128). The generator and discriminator's training losses are presented in Fig. S5.10. Both of losses are converging to zero, which show the training stability of the W model.

The test case example is shown in Fig. S5.11. From this figure, we observe that with a given pressure field (the results with given pressure, displacement in the x-direction, and displacement in y-direction fields yield approximately similar results), the ROM framework could approximate the permeability field with decent accuracy (see DIFF value).

The RMSE results with different W model steps are presented in Fig. S5.12. Fig. S5.12a represents the ROM with a pressure field as input, and Fig. S5.12b represents the ROM with pressure and displacement fields as input. We could observe that the case where we use pressure and displacement fields as input has slightly better results. The maximum RMSE values are 0.3 and 0.28 $\log(\text{m}^2)$ for the first and the second cases, respectively (the absolute value of the magnitude of $\log(\text{permeability})$ is in the range of [8, 12]). The minimum RMSE values of the two cases are similar. This reflects the fact that as the ROM framework receives more information, its accuracy improves. However, there is not much difference between these two cases.

S5.2.2 Example 5: using 75% of input fields - uniformly vs. randomly removed

From Example 4, we observe that the ROM framework could approximate the permeability field with given pressure or pressure and displacement fields with decent accuracy (i.e., the maximum RMSE values are one- to two-ordered

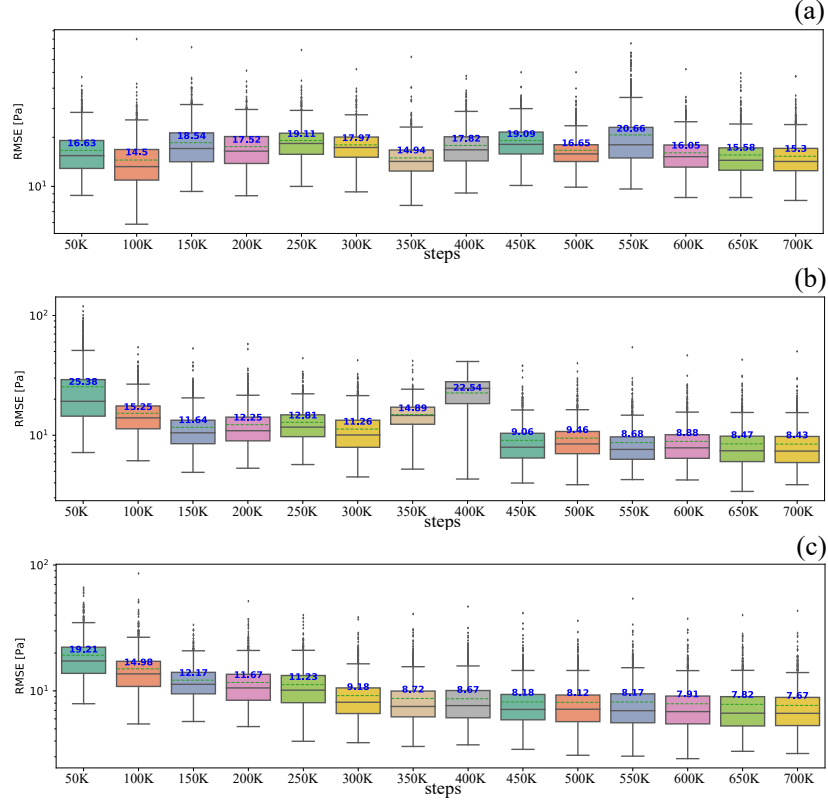


Figure S5.6: Example 2: Root Mean Square Error (RMSE) of (a) base model, (b) SN model, and (c) W model. Each step refers to each time we perform back-propagation including updating both generator and discriminator's parameters.

magnitude less than the log of the field values). This example aims to illustrate cases where the input fields have incomplete data (i.e., the input size is still 128×128 ; however, 25% of these values have no measurements.). Note that we represent no measurements data using a flag of -1 (-1000 in a real value); as mentioned in Section S3, we normalize our data for pressure, displacement, and permeability to $[0, 1]$.

The examples of test cases are presented in Figs. S5.13a-b for 25% of the input data is uniformly removed, and 25% of the input data is randomly removed, respectively. We see that the second case's DIFF values, Fig. S5.13b, are generally higher than those of the first case, Fig. S5.13a. Still, the maximum RMSE values of these two cases are two-ordered magnitude less than the log of the permeability field values.

Example 5.1: using 75% of input fields - uniformly removed

The RMSE results of the uniformly removed case are presented in Fig. S5.14. Compared to the previous example, Example 4, the RMSE values of this example are generally higher. Besides, there are differences between using only pressure as an input (case 1) and using pressure and displacement as input (case 2), see Figs. S5.14a-b. To elaborate, the maximum of RMSE average values are 0.44 and 0.31 $\log(\text{m}^2)$, see blue texts in the Fig. S5.14, for case 1 and case 2, respectively. the minimum of RMSE average values are 0.43 and 0.29 $\log(\text{m}^2)$ for case 1 and case 2, respectively. Both models seem to be improved as the number of steps are increased.

Example 5.2: using 75% of input fields - randomly removed

The RMSE behaviors of the randomly removed data points of using only pressure as an input (case 1) and using pressure and displacement as input (case 2) are illustrated in Fig. S5.15. The maximum of RMSE average values are 0.53 and 0.46 $\log(\text{m}^2)$ for case 1 and case 2, respectively, see blue texts in the Fig. S5.15. The minimum of RMSE average values are 0.50 and 0.42 $\log(\text{m}^2)$ for case 1 and case 2, respectively. Again, we observe that by using more input (i.e., case 2), the model's accuracy is improved.

Even though the number of available data is similar to Example 5.1 (i.e., 75% of the input fields), the RMSE values of this example are significantly higher than Example 5.1, see Figs. S5.14 and S5.15. This behavior reflects the fact that

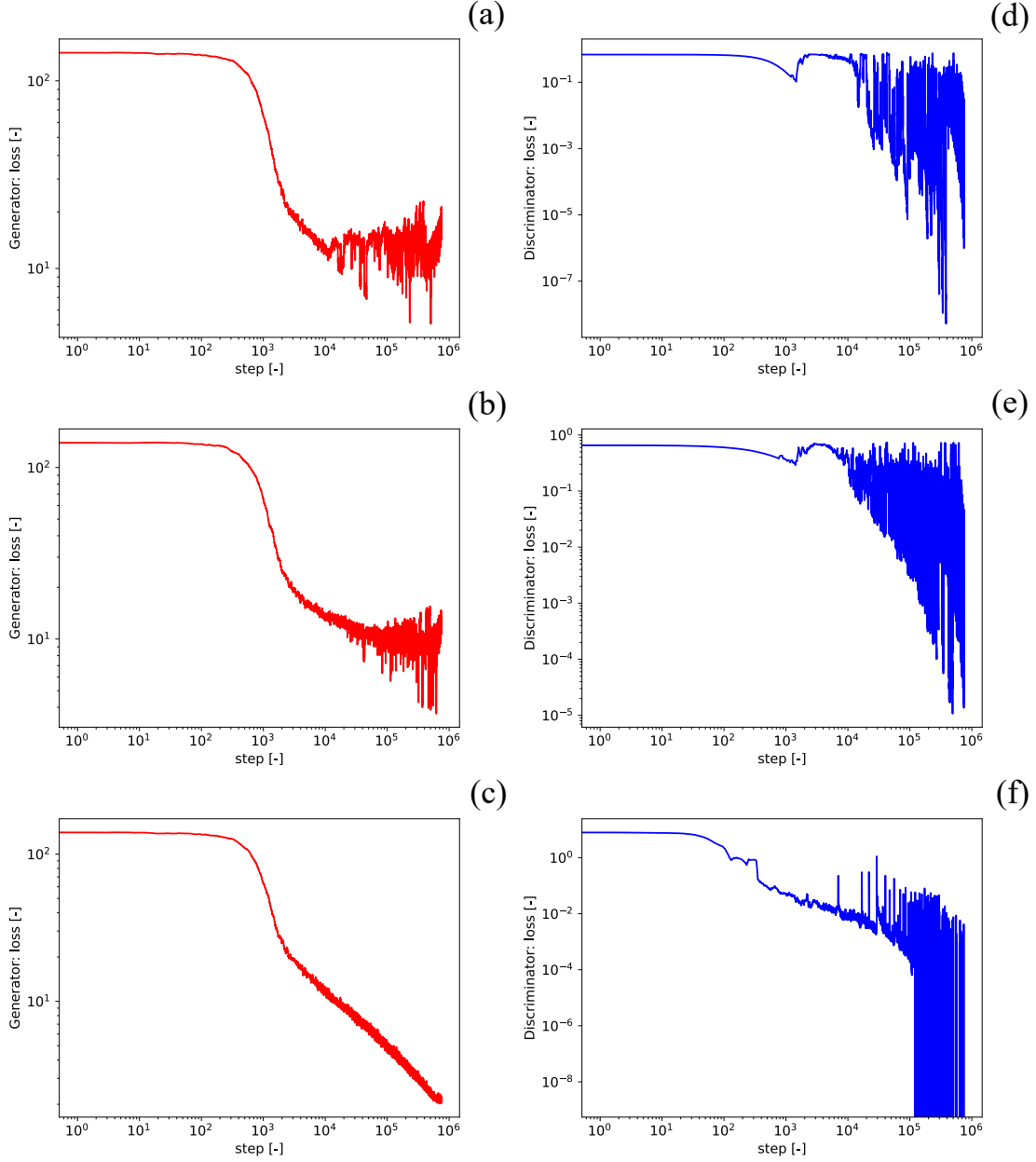


Figure S5.7: Example 3: 100th moving average of Generator loss of (a) base model, (b) SN model, and (c) W model and Discriminator loss of (d) base model, (e) SN model, and (f) W model

the available data's location could impact the model's accuracy substantially. Furthermore, we observe that the model exhibits overfitting behavior (i.e., the test cases' accuracy reduces as the number of steps increases.).

S5.2.3 Example 6: resolution of input fields' effect on model dynamics - randomly removed

In this example, we study the effects of available data on the model training dynamics as well as test cases' accuracy. We only randomly remove the data points from the input fields, (1) only pressure field or (2) pressure, displacement in the x-direction, and displacement in the y-direction fields. We present five examples of test cases in Fig. S5.16 for (a) Example 6.1 - using 44% of input fields, (b) Example 6.2 - using 24% of input fields, (c) Example 6.3 - using 12% of input fields, and (d) Example 6.4 - 6% of input fields. As expected, as the number of available data reduces, the model's accuracy deteriorates. However, the DIFF values are still one- to two-ordered magnitude less than the log of the permeability field values.

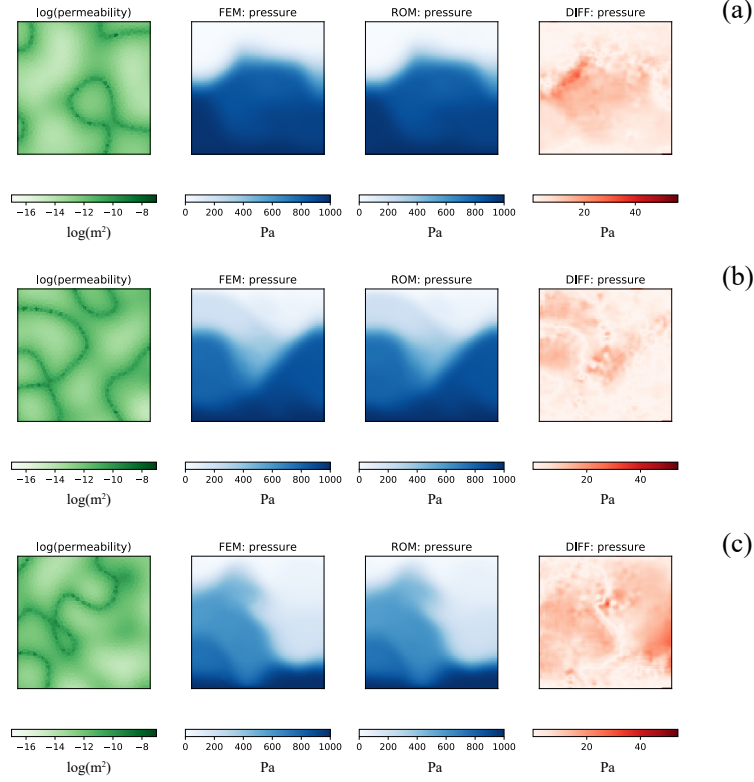


Figure S5.8: Example 3: test cases' results of the W model. Note that we train our three models (base, SN, and W models) using 10000 training examples and test them using 1000 test examples. These three cases shown here are randomly picked from 1000 test examples.

Example 6.1: using 44% of input fields

Example 6.1 uses the only 44% of original input fields, and its RSME results are shown in Fig. S5.17a using only pressure field as an input - case 1 and Fig. S5.17b using pressure and displacement fields as input - case 2. Again, case 2 (more information for the input) has better accuracy than case 1. The maximum of RMSE average values are 0.59 and 0.50 $\log(\text{m}^2)$ for case 1 and case 2, respectively, see blue texts in the Fig. S5.17. The minimum of RMSE average values are 0.56 and 0.45 $\log(\text{m}^2)$ for case 1 and case 2, respectively. As expected, the accuracy deteriorates as we reduce the number of available data (i.e., from 75% of original input fields to 44% of original input fields). Similar to Example 5.2, the model exhibits the overfitting behavior.

Example 6.2: using 24% of input fields

We now reduce the available data to 24% of original input fields. The RMSE values are presented in Fig. S5.18a using only pressure field as an input - case 1 and Fig. S5.18b using pressure and displacement fields as input - case 2. The maximum of RMSE average values are 0.62 and 0.52 $\log(\text{m}^2)$ for case 1 and case 2, respectively. The minimum of RMSE average values are 0.60 and 0.48 $\log(\text{m}^2)$ for case 1 and case 2, respectively. Similar to Example 6.1, the model with more input information (case 2) has better accuracy. The ROM framework shows the overfitting behavior. The model's accuracy is decreased as we reduce the available information from 44% of original input fields to 24% of original input fields.

Example 6.3: using 12% of input fields

We then decrease the available information from 24% of original input fields to 12% of original input fields. The RMSE behaviors are illustrated in Fig. S5.19a using only pressure field as an input - case 1 and Fig. S5.19b using pressure and displacement fields as input - case 2. The maximum of RMSE average values are 0.68 and 0.55 $\log(\text{m}^2)$ for case 1 and case 2, respectively. The minimum of RMSE average values are 0.67 and 0.51 $\log(\text{m}^2)$ for case 1 and case 2, respectively. Again, case 2 has better test cases' accuracy than those of case 1. The model still shows the overfitting behavior (i.e., the test cases' accuracy reduces as the number of steps increases.).

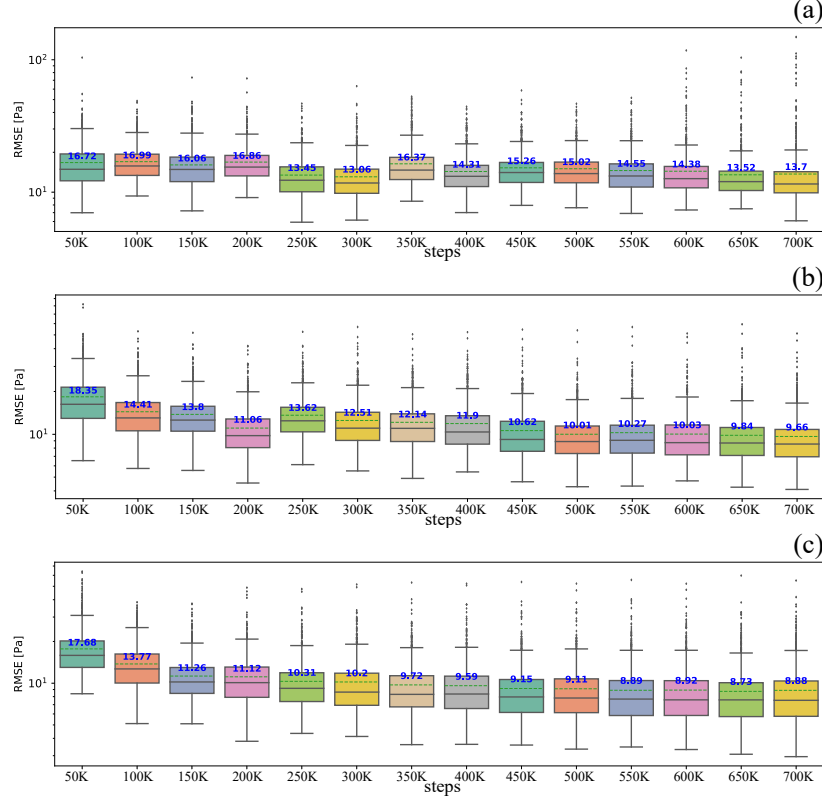


Figure S5.9: Example 3: Root Mean Square Error (RMSE) of (a) base model, (b) SN model, and (c) W model. Each step refers to each time we perform back-propagation including updating both generator and discriminator's parameters.

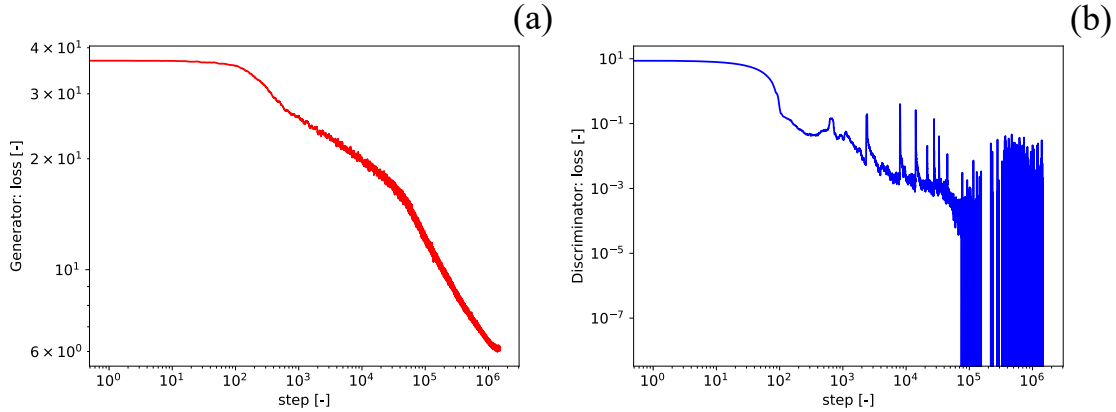


Figure S5.10: Example 4: 100th moving average of Generator and Discriminator losses of W model

Example 6.4: using 6% of input fields

Finally, we use only 6% of original input fields. The RMSE results are shown in Fig. S5.20a using only pressure field as an input - case 1 and Fig. S5.20b using pressure and displacement fields as input - case 2. The maximum of RMSE average values are 0.72 and 0.57 $\log(\text{m}^2)$ for case 1 and case 2, respectively. The minimum of RMSE average values are 0.70 and 0.54 $\log(\text{m}^2)$ for case 1 and case 2, respectively. Similar to Examples 5.2, 6.1, 6.2, and 6.3, case 2 has better test cases' accuracy than those of case 1. The model still shows the overfitting behavior.

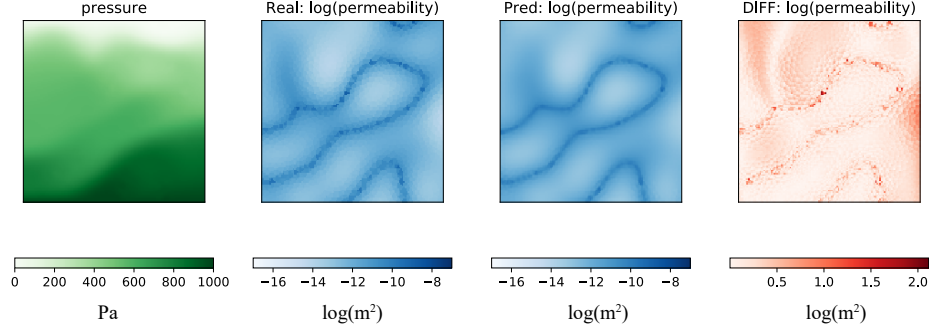


Figure S5.11: Example 4: test cases' result of the W model. Note that we train our model using 10000 training examples and test them using 1000 test examples. This case shown here is randomly picked from 1000 test examples.

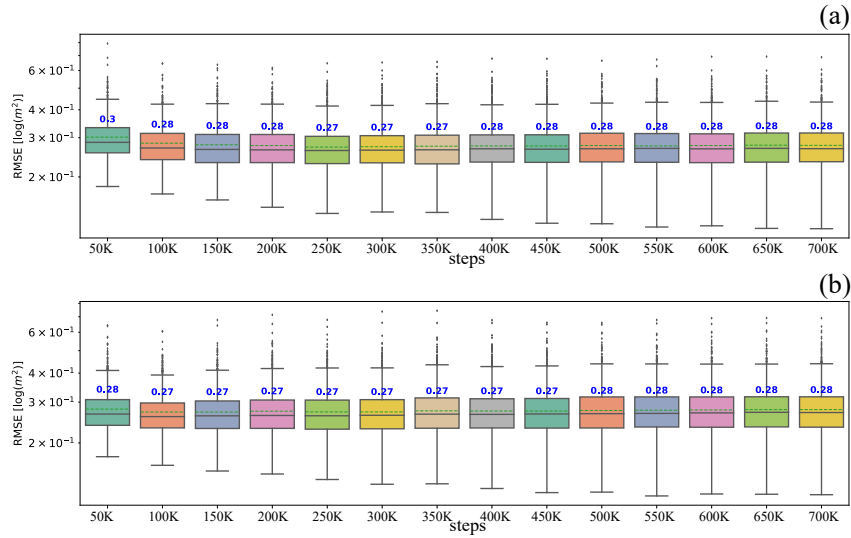


Figure S5.12: Example 4: Root Mean Square Error (RMSE) of W model using (a) pressure as input and (b) pressure and displacement as input. The number of training examples is 10000. Each step refers to each time we perform back-propagation including updating both generator and discriminator's parameters.

S5.3 Comparison with Neural Operator approach

We compare our W model with the Neural Operator approach [40]. The methods are tested on the second order elliptic PDE of the form

$$\begin{aligned} -\nabla \cdot a(\mathbf{X}) \nabla u(\mathbf{X}) &= 0 \quad \text{in } \Omega, \\ u(\mathbf{X}) &= 0, \quad \text{on } \partial\Omega_D, \end{aligned} \quad (\text{S5.1})$$

where $\Omega = [0, 1] \times [0, 1]$. We utilize the training and test data sets as provided by the authors [40]. The investigated training data consists of 500 training samples. Furthermore, we employ the original Neural Operator code provided by the authors to offer a consistent comparison.

We train the W model on a single Quadro RTX 6000, and it takes approximately 15 minutes (resolution of 32×32) for the model at the 20000 steps checkpoint. The training of the Neural Operator model takes around 3 hours for 500 epochs with a resolution of 35×35 . Consistent with the parameter settings proposed by the authors [40], a ReLU activation function and a learning rate of 10^{-4} are used. The kernel network consists of 5 layers with a width of 256 each. Both the test and training radii are set to 0.1. We found that an increase in training resolution (from 35×35 to 85×85) significantly increases the training time. This expensive training behavior may limit the utility of the Neural

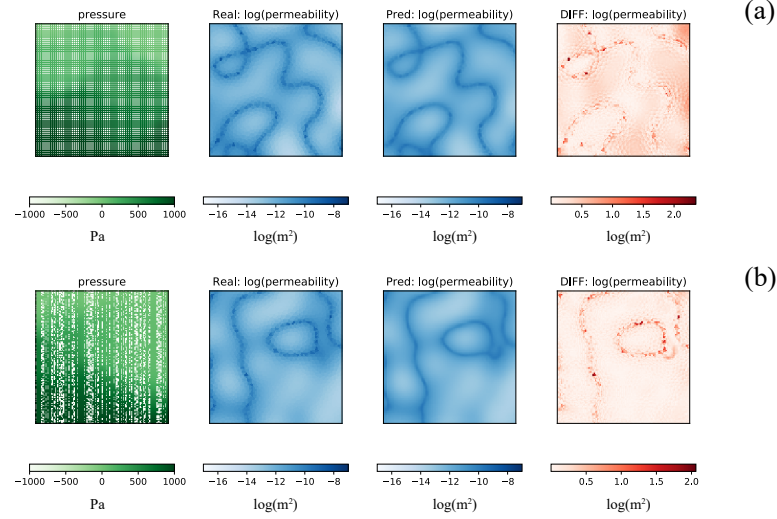


Figure S5.13: Example 5: test cases' results of the W model of (a) Example 5.1 - using 75% of input fields - uniformly removed and (b) Example 5.2 - using 75% of input fields - randomly removed. Note that we train our model using 10000 training examples and test them using 1000 test examples. Each case shown here is randomly picked from 1000 test examples.

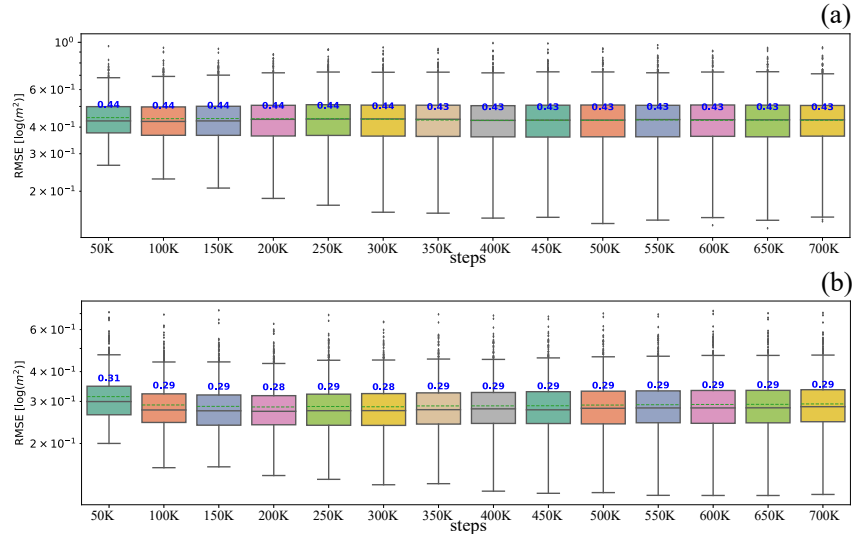


Figure S5.14: Example 5.1: Root Mean Square Error (RMSE) of W model using (a) pressure as input and (b) pressure and displacement as input. The number of training examples is 10000. Each step refers to each time we perform back-propagation including updating both generator and discriminator's parameters.

Operator approach. We project that using our available computational infrastructure and employing a training resolution of 241×241 would take more than one week to train. We have not experienced the same issues with the W model.

The generalization capabilities of the W model and the Neural Operator are compared in Figure S5.21 for three random test cases, which were not part of the training set. Overall, the RMSE of the W model is 5.69×10^{-10} , 7.65×10^{-10} , and 1.50×10^{-9} for Figures S5.21a, S5.21b, and S5.21c, respectively. The RMSE of the Neural Operator is 7.68×10^{-5} , 1.78×10^{-6} , and 1.89×10^{-6} for Figures S5.21a, S5.21b, and S5.21c, respectively. It can be seen that the proposed W model outperforms the Neural Operator approach for the investigated dataset.

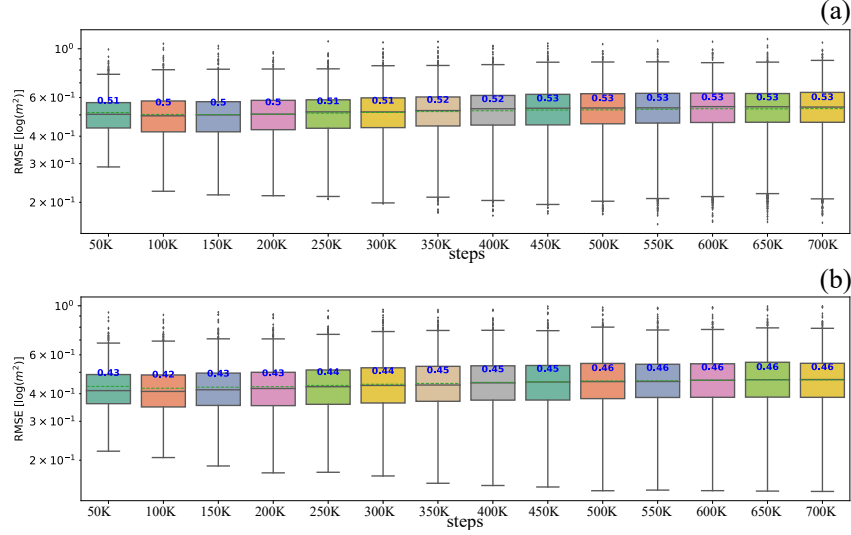


Figure S5.15: Example 5.2: Root Mean Square Error (RMSE) of W model using (a) pressure as input and (b) pressure and displacement as input. The number of training examples is 10000. Each step refers to each time we perform back-propagation including updating both generator and discriminator's parameters.

S5.4 Comparison with generator that has no skip connections

We compare our W model with two different generators. The first one uses skip connections (U-Net) as described in S3. The second one removes all skip connections and technically is a deep convolutional autoencoder. We illustrate the RMSE results in Figure S5.22 for both using permeability fields from a Zinn & Harvey transformation as input (Figure S5.22a) and permeability fields given as a Gaussian distribution, permeability fields defined by a bimodal transformation, and permeability fields from a Zinn & Harvey transformation as input (Figure S5.22b). These RMSE results are calculated from a test set. We note that for the first case, we use 10000 training examples and 1000 test examples. The second case uses 7500 training examples (2500 for each field) and 1000 test examples (334 for the Gaussian distribution, 333 for the bimodal transformation, and 333 for the Zinn & Harvey transformation).

Comparing Figures S5.9c and S5.22a, we observe that the model with a U-Net generator outperforms the model with a deep convolutional autoencoder generator in terms of accuracy. Moreover, the RMSE results of the deep convolutional autoencoder generator become worse as we use permeability fields given as a Gaussian distribution, permeability fields defined by a bimodal transformation, and permeability fields from a Zinn & Harvey transformation as input (see a comparison between Figures 3c and S5.22b). This behavior reflects that the deep convolutional autoencoder generator could not generalize as well as the U-Net generator when the permeability fields become more complex.

References

- [1] R. Middleton, W. Carey, R. Currier, J. Hyman, Q. Kang, S. Karra, J. Jiménez-Martínez, Mark L. Porter, and H. Viswanathan. Shale gas and non-aqueous fracturing fluids: Opportunities and challenges for supercritical co2. *Applied Energy*, 147:500–509, 2015.
- [2] J. Choo and W. Sun. Cracking and damage from crystallization in pores: Coupled chemo-hydro-mechanics and phase-field modeling. *Computer Methods in Applied Mechanics and Engineering*, 335:347–349, 2018.
- [3] Y. Yu, N. Bouklas, C. Landis, and R. Huang. Poroelastic effects on the time-and rate-dependent fracture of polymer gels. *Journal of Applied Mechanics*, 87(3), 2020.
- [4] V. Vinje, J. Brucker, M. Rognes, K. Mardal, and V. Haughton. Fluid dynamics in syringomyelia cavities: Effects of heart rate, CSF velocity, CSF velocity waveform and craniovertebral decompression. *The neuroradiology journal*, page 1971400918795482, 2018.
- [5] T. Kadeethum, S. Salimzadeh, and H. Nick. An investigation of hydromechanical effect on well productivity in fractured porous media using full factorial experimental design. *Journal of Petroleum Science and Engineering*, 181:106233, 2019.

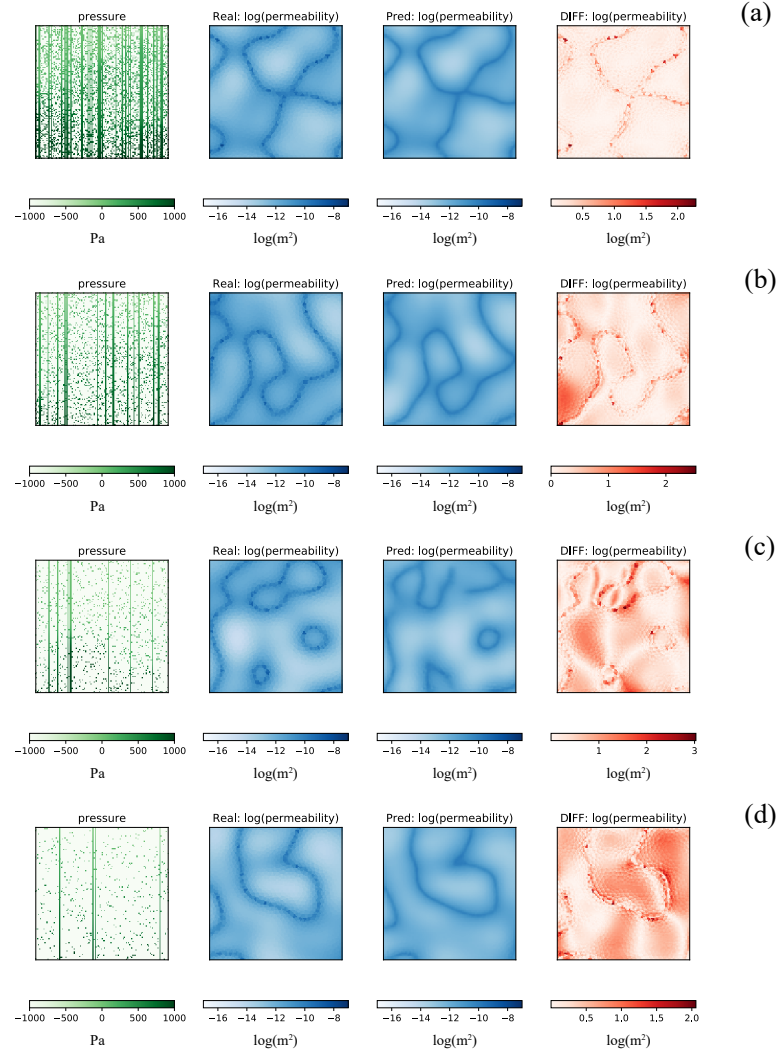


Figure S5.16: Example 6: test cases' results of the W model of (a) Example 6.1 - using 44% of input fields, (b) Example 6.2 - using 24% of input fields, (c) Example 6.3 - using 12% of input fields, and (d) Example 6.4 - 6% of input fields. Note that we train our model using 10000 training examples and test them using 1000 test examples. Each case shown here is randomly picked from 1000 test examples.

- [6] N. Bouklas and R. Huang. Swelling kinetics of polymer gels: comparison of linear and nonlinear theories. *Soft Matter*, 8(31):8194–8203, 2012.
- [7] M. Biot. General theory of three-dimensional consolidation. *Journal of applied physics*, 12(2):155–164, 1941.
- [8] H. Liu, J. Li, and Q. Zhang. Lie symmetry analysis and exact explicit solutions for general burgers' equation. *Journal of Computational and Applied Mathematics*, 228(1):1–9, 2009.
- [9] T. Kadeethum, S. Salimzadeh, and H. Nick. Well productivity evaluation in deformable single-fracture media. *Geothermics*, 87, 2020.
- [10] M. Wheeler, G. Xue, and I. Yotov. Coupling multipoint flux mixed finite element methods with continuous Galerkin methods for poroelasticity. *Computational Geosciences*, 18(1):57–75, 2014.
- [11] Q. Deng, V. Ginting, B. McCaskill, and P. Torsu. A locally conservative stabilized continuous Galerkin finite element method for two-phase flow in poroelastic subsurfaces. *Journal of Computational Physics*, 347:78–98, 2017.
- [12] J. Liu, S. Tavener, and Z. Wang. Lowest-order weak Galerkin finite element method for Darcy flow on convex polygonal meshes. *SIAM Journal on Scientific Computing*, 40(5):B1229–B1252, 2018.

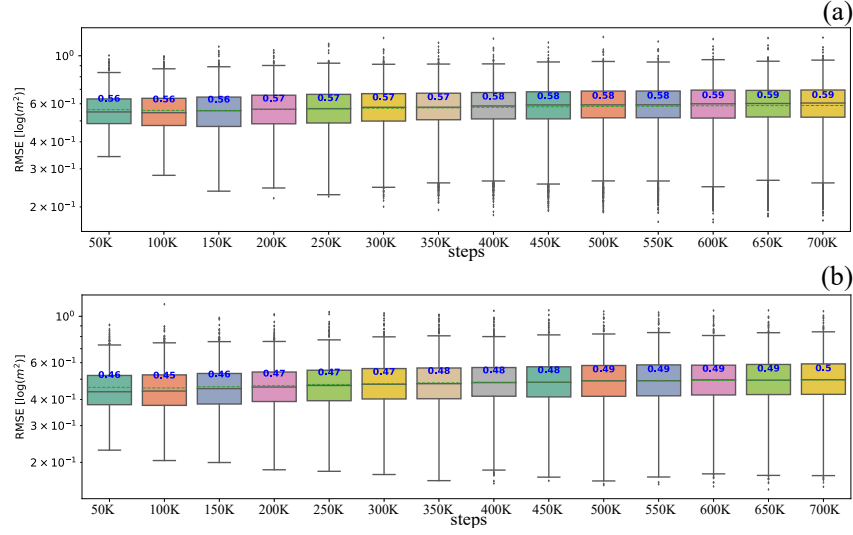


Figure S5.17: Example 6.1: Root Mean Square Error (RMSE) of W model using (a) pressure as input and (b) pressure and displacement as input. The number of training examples is 10000. Each step refers to each time we perform back-propagation including updating both generator and discriminator's parameters.

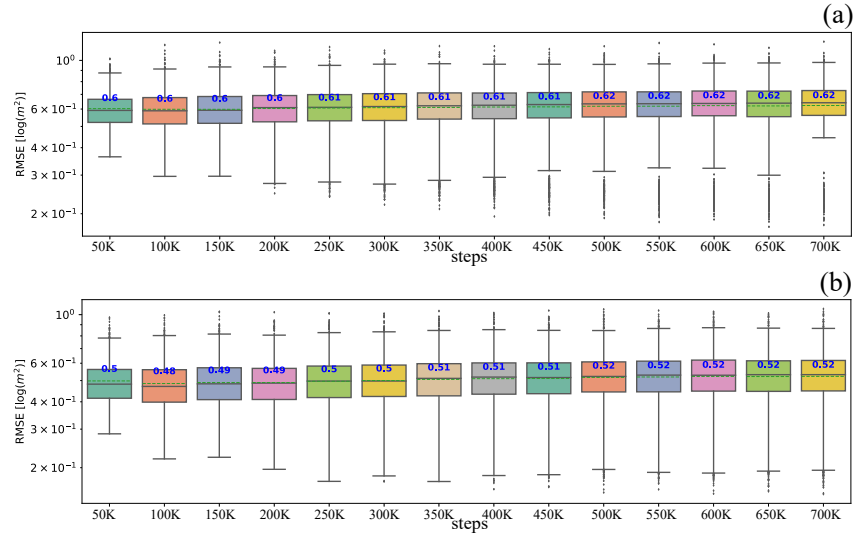


Figure S5.18: Example 6.2: Root Mean Square Error (RMSE) of W model using (a) pressure as input and (b) pressure and displacement as input. The number of training examples is 10000. Each step refers to each time we perform back-propagation including updating both generator and discriminator's parameters.

- [13] B. Devarajan and R. Kapania. Thermal buckling of curvilinearly stiffened laminated composite plates with cutouts using isogeometric analysis. *Composite Structures*, 238:111881, 2020.
- [14] P. Hansen. *Discrete inverse problems: insight and algorithms*, volume 7. Siam, 2010.
- [15] J. Hesthaven, G. Rozza, B. Stamm, et al. *Certified reduced basis methods for parametrized partial differential equations*. Springer, 2016.
- [16] F. Ballarin, A. D'amario, S. Perotto, and G. Rozza. A POD-selective inverse distance weighting method for fast parametrized shape morphing. *International Journal for Numerical Methods in Engineering*, 117(8):860–884, 2019.
- [17] W. Schilders, H. Van der Vorst, and J. Rommes. *Model order reduction: theory, research aspects and applications*, volume 13. Springer, 2008.

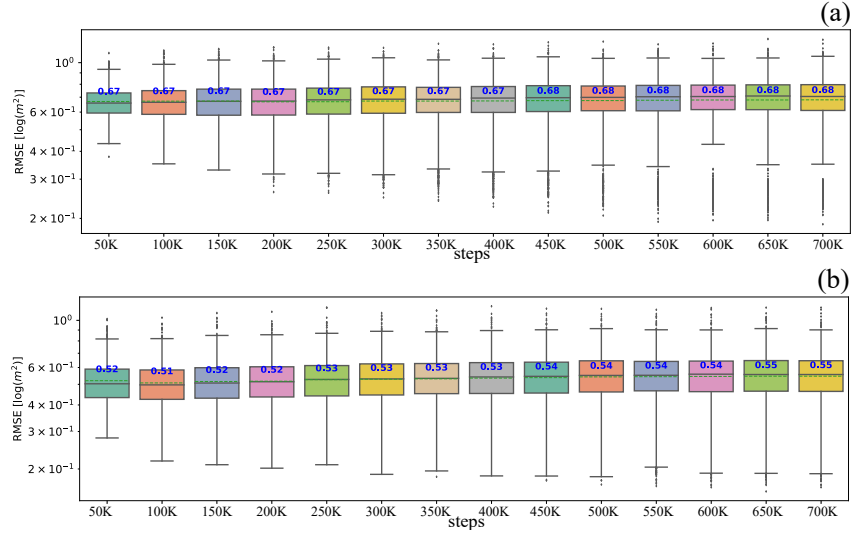


Figure S5.19: Example 6.3: Root Mean Square Error (RMSE) of W model using (a) pressure as input and (b) pressure and displacement as input. The number of training examples is 10000. Each step refers to each time we perform back-propagation including updating both generator and discriminator's parameters.

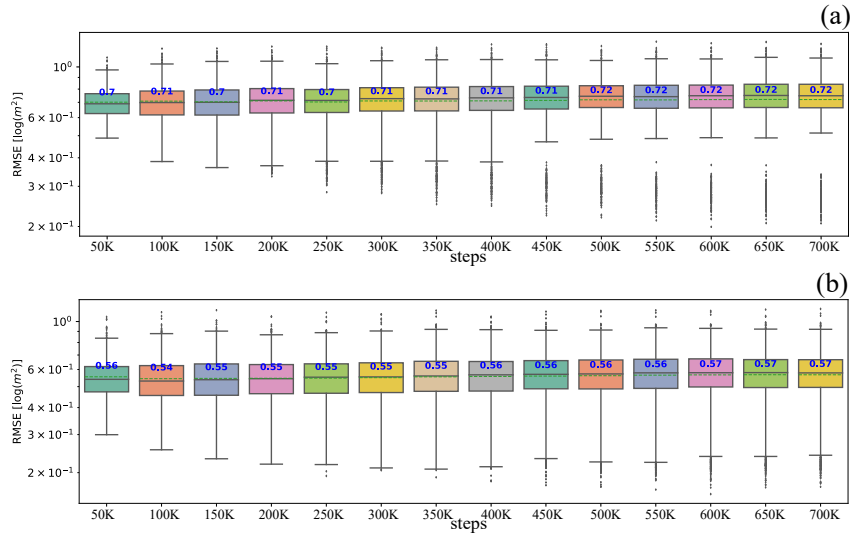


Figure S5.20: Example 6.4: Root Mean Square Error (RMSE) of W model using (a) pressure as input and (b) pressure and displacement as input. The number of training examples is 10000. Each step refers to each time we perform back-propagation including updating both generator and discriminator's parameters.

- [18] Y. Choi, G. Boncoraglio, S. Anderson, D. Amsallem, and F. Farhat. Gradient-based constrained optimization using a database of linear reduced-order models. *Journal of Computational Physics*, 423, 2020.
- [19] S. McBane and Y. Choi. Component-wise reduced order model lattice-type structure design. *Computer Methods in Applied Mechanics and Engineering*, 381, 2021.
- [20] Y. Choi, G. Oxberry, D. White, and T. Kirchdoerfer. Accelerating design optimization using reduced order models. *arXiv preprint arXiv:1909.11320*, 2019.
- [21] D. Amsallem, M. Zahr, Y. Choi, and C. Farhat. Design optimization using hyper-reduced-order models. *Structural and Multidisciplinary Optimization*, 51(4):919–940, 2015.
- [22] Y. Kim, K.M. Wang, and Y. Choi. Efficient space–time reduced order model for linear dynamical systems in python using less than 120 lines of code. *arXiv preprint arXiv:2011.10648*, 2020.

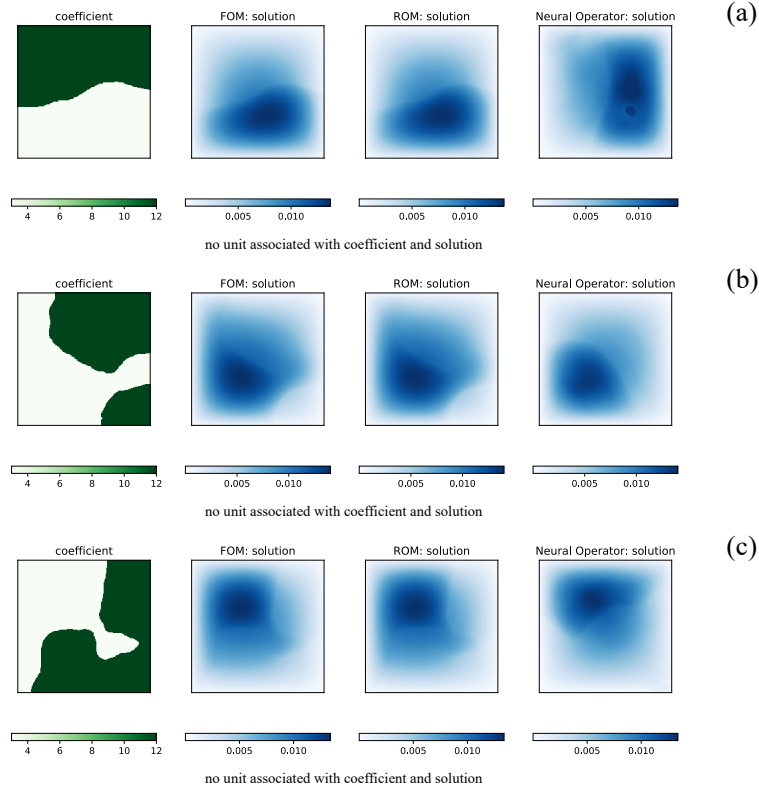


Figure S5.21: Comparison of test cases' results between the W model and Neural Operator approach. Note that we train the models using 500 training examples and test them using 524 test examples. These three cases shown here are randomly picked from 524 test examples.

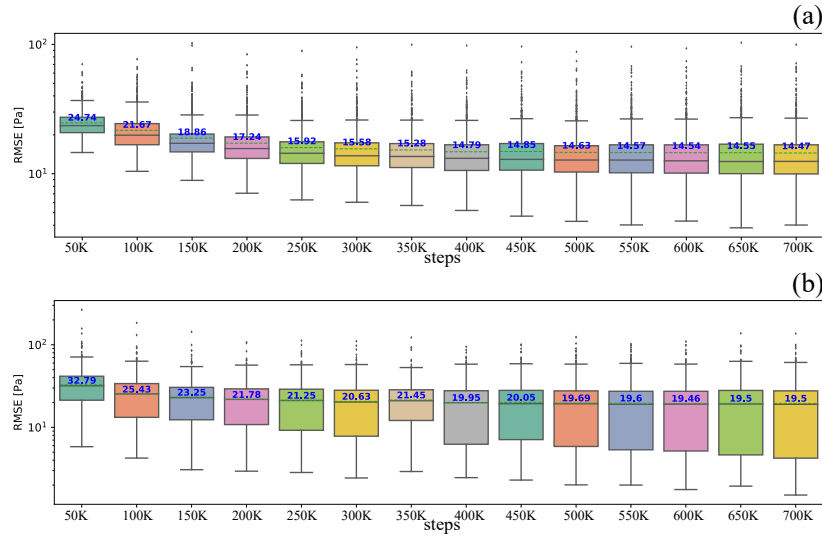


Figure S5.22: Comparison with generator that has no skip connections: Root Mean Square Error (RMSE) of W model using (a) permeability fields from a Zinn & Harvey transformation as input and (b) permeability fields given as a Gaussian distribution, permeability fields defined by a bimodal transformation, and permeability fields from a Zinn & Harvey transformation as input. Each step refers to each time we perform back-propagation including updating both generator and discriminator's parameters.

- [23] Y. Kim, Y. Choi, D. Widemann, and T. Zohdi. A fast and accurate physics-informed neural network reduced order model with shallow masked autoencoder. *arXiv preprint arXiv:2009.11990*, 2020.
- [24] C. Hoang, Y. Choi, and K. Carlberg. Domain-decomposition least-squares petrov-galerkin (dd-lspg) nonlinear model reduction. *arXiv preprint arXiv:2007.11835*, 2020.
- [25] K. Carlberg, Y. Choi, and S. Sargsyan. Conservative model reduction for finite-volume models. *Journal of Computational Physics*, 371:280–314, 2018.
- [26] V. DeCaria, T. Iliescu, W. Layton, M. McLaughlin, and M. Schneier. An artificial compression reduced order model. *SIAM Journal on Numerical Analysis*, 58(1):565–589, 2020.
- [27] J. Cleary and I. Witten. Data compression using adaptive coding and partial string matching. *IEEE transactions on Communications*, 32(4):396–402, 1984.
- [28] Y. Choi, P. Brown, W. Arrighi, R. Anderson, and K. Huynh. Space-time reduced order model for large-scale linear dynamical systems with application to boltzmann transport problems. *Journal of Computational Physics*, 424, 2021.
- [29] L. Venturi, F. Ballarin, and G. Rozza. A weighted POD method for elliptic PDEs with random inputs. *Journal of Scientific Computing*, 81(1):136–153, 2019.
- [30] S. Matthai and H. Nick. Upscaling two-phase flow in naturally fractured reservoirs. *AAPG bulletin*, 93(11):1621–1632, 2009.
- [31] B. Flemisch, I. Berre, W. Boon, A. Fumagalli, N. Schwenck, A. Scotti, I. Stefansson, and A. Tatomir. Benchmarks for single-phase flow in fractured porous media. *Advances in Water Resources*, 111:239–258, 2018.
- [32] P. Jia, L. Cheng, S. Huang, Z. Xu, Y. Xue, R. Cao, and G. Ding. A comprehensive model combining Laplace-transform finite-difference and boundary-element method for the flow behavior of a two-zone system with discrete fracture network. *Journal of Hydrology*, 551:453–469, 2017.
- [33] D. O’Malley, J. Golden, and V. Vesselinov. Learning to regularize with a variational autoencoder for hydrologic inverse analysis. *arXiv preprint arXiv:1906.02401*, 2019.
- [34] Y. Lin, D. O’Malley, and V. Vesselinov. A computationally efficient parallel levenberg-marquardt algorithm for highly parameterized inverse model analyses. *Water Resources Research*, 52(9):6948–6977, 2016.
- [35] U. Villa, N. Petra, and O. Ghattas. hippylib: An extensible software framework for large-scale inverse problems. *Journal of Open Source Software*, 3(30):940, 2018.
- [36] M. Raissi, P. Perdikaris, and G. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.
- [37] Teeratrorn Kadeethum, Thomas M Jørgensen, and Hamidreza M Nick. Physics-informed neural networks for solving nonlinear diffusivity and biot’s equations. *PloS one*, 15(5):e0232683, 2020.
- [38] Jan N Fuhg and Nikolaos Bouklas. The mixed deep energy method for resolving concentration features in finite strain hyperelasticity. *arXiv preprint arXiv:2104.09623*, 2021.
- [39] A. Jagtap and G. Karniadakis. Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5):2002–2041, 2020.
- [40] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020.
- [41] M. Mignolet, A. Przekop, S. Rizzi, and M. Spottswood. A review of indirect/non-intrusive reduced order modeling of nonlinear geometric structures. *Journal of Sound and Vibration*, 332(10):2437–2460, 2013.
- [42] J. Hesthaven and S. Ubbiali. Non-intrusive reduced order modeling of nonlinear problems using neural networks. *Journal of Computational Physics*, 363:55–78, 2018.
- [43] D. Xiao, F. Fang, A. Buchan, C. Pain, I. Navon, and A. Muggeridge. Non-intrusive reduced order modelling of the Navier–Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 293:522–541, 2015.
- [44] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.
- [45] M. Liu and O. Tuzel. Coupled generative adversarial networks. *arXiv preprint arXiv:1606.07536*, 2016.
- [46] A. Odena, V. Dumoulin, and C. Olah. Deconvolution and checkerboard artifacts. *Distill*, 1(10):e3, 2016.

- [47] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.
- [48] Y. Yu, Z. Gong, P. Zhong, and J. Shan. Unsupervised representation learning with deep convolutional neural network for remote sensing images. In *International Conference on Image and Graphics*, pages 97–108. Springer, 2017.
- [49] J. Yoon, D. Jarrett, and M. van der Schaar. Time-series generative adversarial networks. *NeurIPS*, 2019.
- [50] M. Naruse, T. Matsubara, N. Chauvet, K. Kanno, T. Yang, and A. Uchida. Generative adversarial network based on chaotic time series. *Scientific reports*, 9(1):1–9, 2019.
- [51] Y. Shen, J. Gu, X. Tang, and B. Zhou. Interpreting the latent space of gans for semantic face editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9243–9252, 2020.
- [52] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.
- [53] X. Chen, Y. Duan, R. Houthoofd, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *arXiv preprint arXiv:1606.03657*, 2016.
- [54] X. Lee, Joshua R. Waite, C. Yang, S. Pokuri, A. Joshi, A. Balu, C. Hegde, B. Ganapathysubramanian, and S. Sarkar. Fast inverse design of microstructures via generative invariance networks. *Nature Computational Science*, 1(3):229–238, 2021.
- [55] P. Isola, J. Zhu, T. Zhou, and A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.
- [56] T. Wang, M. Liu, J. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8798–8807, 2018.
- [57] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.
- [58] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1316–1324, 2018.
- [59] J. Zhu, T. Park, P. Isola, and A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.
- [60] M. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. *arXiv preprint arXiv:1703.00848*, 2017.
- [61] X. Huang, M. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 172–189, 2018.
- [62] T. Park, M. Liu, T. Wang, and J. Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2337–2346, 2019.
- [63] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. Karniadakis. Learning nonlinear operators via deepnet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.
- [64] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.
- [65] U. Demir and G. Unal. Patch-based image inpainting with generative adversarial networks. *arXiv preprint arXiv:1803.07422*, 2018.
- [66] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.
- [67] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.
- [68] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017.
- [69] T. Kadeethum, H. Nick, S. Lee, and F. Ballarin. Enriched Galerkin discretization for modeling poroelasticity and permeability alteration in heterogeneous porous media. *Journal of Computational Physics*, page 110030, 2021.

- [70] T. Kadeethum, S. Lee, and H. Nick. Finite element solvers for biot’s poroelasticity equations in porous media. *Mathematical Geosciences*, 52:977–1015, 2020.
- [71] B. Zinn and C. Harvey. When good statistical models of aquifer heterogeneity go bad: A comparison of flow, dispersion, and mass transfer in connected and multivariate gaussian hydraulic conductivity fields. *Water Resources Research*, 39(3), 2003.
- [72] I. Tetko, D. Livingstone, and A. Luik. Neural network studies. 1. comparison of overfitting and overtraining. *Journal of chemical information and computer sciences*, 35(5):826–833, 1995.
- [73] D. O’Malley, S. Karra, J. Hyman, H. Viswanathan, and G. Srinivasan. Efficient monte carlo with graph-based subsurface flow and transport models. *Water Resources Research*, 54(5):3758–3766, 2018.
- [74] B. Peherstorfer, K. Willcox, and M. Gunzburger. Optimal model management for multifidelity monte carlo estimation. *SIAM Journal on Scientific Computing*, 38(5):A3163–A3194, 2016.
- [75] J. Lee and P. Kitanidis. Large-scale hydraulic tomography and joint inversion of head and tracer data using the principal component geostatistical approach (pcga). *Water Resources Research*, 50(7):5410–5427, 2014.
- [76] J. Lee, H. Yoon, P. Kitanidis, C. Werth, and A. Valocchi. Scalable subsurface inverse modeling of huge data sets with an application to tracer concentration breakthrough data from magnetic resonance imaging. *Water Resources Research*, 52(7):5213–5231, 2016.
- [77] Hojat Ghorbanidehno, Amalia Kokkinaki, Jonghyun Lee, and Eric Darve. Recent developments in fast and scalable inverse modeling and data assimilation methods in hydrology. *Journal of Hydrology*, page 125266, 2020.
- [78] Ruoxi Wang, Chao Chen, Jonghyun Lee, and Eric Darve. Pbbfmm3d: A parallel black-box algorithm for kernel matrix-vector multiplication. *Journal of Parallel and Distributed Computing*, 154:64–73, 2021.
- [79] Peter K Kang, Jonghyun Lee, Xiaojing Fu, Seunghak Lee, Peter K Kitanidis, and Ruben Juanes. Improved characterization of heterogeneous permeability in saline aquifers from transient pressure data during freshwater injection. *Water Resources Research*, 53(5):4444–4458, 2017.
- [80] Jonghyun Lee, Amalia Kokkinaki, and Peter K Kitanidis. Fast large-scale joint inversion for deep aquifer characterization using pressure and heat tracer measurements. *Transport in Porous Media*, 123(3):533–543, 2018.
- [81] T. Kadeethum, T. Jørgensen, and H. Nick. Physics-informed neural networks for solving nonlinear diffusivity and Biot’s equations. *PLoS ONE*, 15(5):e0232683, 2020.
- [82] T. Kadeethum, T. Jørgensen, and H. Nick. Physics-informed Neural Networks for Solving Inverse Problems of Nonlinear Biot’s Equations: Batch Training. In *54th US Rock Mechanics/Geomechanics Symposium*, Golden, CO, USA, 2020. American Rock Mechanics Association.
- [83] K. Bisdom, G. Bertotti, and H. Nick. A geometrically based method for predicting stress-induced fracture aperture and flow in discrete fracture networks. *AAPG Bulletin*, 100(7):1075–1097, 2016.
- [84] R. Juanes, B. Jha, B. Hager, J. Shaw, A. Plesch, L. Astiz, J. Dieterich, and C. Frohlich. Were the May 2012 Emilia-Romagna earthquakes induced? A coupled flow-geomechanics modeling assessment. *Geophysical Research Letters*, 43(13):6891–6897, 2016.
- [85] S. Lee, M. Wheeler, and T. Wick. Pressure and fluid-driven fracture propagation in porous media using an adaptive finite element phase field model. *Computer Methods in Applied Mechanics and Engineering*, 305:111–132, 2016.
- [86] H. Nick, A. Raoof, F. Centler, M. Thullner, and P. Regnier. Reactive dispersive contaminant transport in coastal aquifers: numerical simulation of a reactive henry problem. *Journal of contaminant hydrology*, 145:90–104, 2013.
- [87] N. Bouklas, C. Landis, and R. Huang. A nonlinear, transient finite element method for coupled solvent diffusion and large deformation of hydrogels. *Journal of the Mechanics and Physics of Solids*, 79:21–43, 2015.
- [88] S. Salimzadeh, E. Hagerup, T. Kadeethum, and H. Nick. The effect of stress distribution on the shape and direction of hydraulic fractures in layered media. *Engineering Fracture Mechanics*, 215:151–163, 2019.
- [89] M. Biot and D. Willis. The elastic coefficients of the theory of consolidation. *J. appl. Mech.*, 15:594–601, 1957.
- [90] J. Choo, J. White, and R. Borja. Hydromechanical modeling of unsaturated flow in double porosity media. *International Journal of Geomechanics*, 16(6):D4016002, 2016.
- [91] R. Borja and J. Choo. Cam-Clay plasticity, Part VIII: A constitutive framework for porous materials with evolving internal structure. *Computer Methods in Applied Mechanics and Engineering*, 309:653–679, 2016.
- [92] C. Macminn, E. Dufresne, and J. Wettlaufer. Large deformations of a soft porous material. *Physical Review Applied*, 5(4):1–30, 2016.

- [93] Y. Zhao and J. Choo. Stabilized material point methods for coupled large deformation and fluid flow in porous materials. *Computer Methods in Applied Mechanics and Engineering*, 362:112742, 2020.
- [94] J. Jaeger, Neville G. Cook, and R. Zimmerman. *Fundamentals of rock mechanics*. John Wiley & Sons, 2009.
- [95] O. Coussy. *Poromechanics*. John Wiley & Sons, 2004.
- [96] J. Kim, H. Tchelepi, and R. Juanes. Stability and convergence of sequential methods for coupled flow and geomechanics: Fixed-stress and fixed-strain splits. *Computer Methods in Applied Mechanics and Engineering*, 200(13-16):1591–1606, 2011.
- [97] A. Mikelić and M. Wheeler. Convergence of iterative coupling for coupled flow and geomechanics. *Computational Geosciences*, 17(3):455–461, 2013.
- [98] T. Kadeethum, H. Nick, S. Lee, C. Richardson, S. Salimzadeh, and F. Ballarin. A Novel Enriched Galerkin Method for Modelling Coupled Flow and Mechanical Deformation in Heterogeneous Porous Media. In *53rd US Rock Mechanics/Geomechanics Symposium*, New York, NY, USA, 2019. American Rock Mechanics Association.
- [99] T. Kadeethum, F. Ballarin, and N. Bouklas. Non-intrusive reduced order modeling of poroelasticity of heterogeneous media based on a discontinuous galerkin approximation. *arXiv preprint arXiv:2101.11810*, 2021.
- [100] S. Müller and L. Schüller. GeoStat-Framework/GSTools. Zenodo, 2020.
- [101] R. Baker, H. Yarranton, and J. Jensen. *Practical reservoir engineering and characterization*. Gulf Professional Publishing, 2015.
- [102] T. Kadeethum, S. Salimzadeh, and H. Nick. Investigation on the productivity behaviour in deformable heterogeneous fractured reservoirs. In *2018 International Symposium on Energy Geotechnics*, 2018.
- [103] B. Muljadi, M. Blunt, A. Raeini, and B. Bijeljic. The impact of porous media heterogeneity on non-darcy flow behaviour from pore-scale simulation. *Advances in water resources*, 95:329–340, 2016.
- [104] C. Nicolaidides, B. Jha, L. Cueto-Felgueroso, and R. Juanes. Impact of viscous fingering and permeability heterogeneity on fluid mixing in porous media. *Water Resources Research*, 51(4):2634–2647, 2015.
- [105] T. Kadeethum, H. Nick, S. Lee, and F. Ballarin. Flow in porous media with low dimensional fractures by employing Enriched Galerkin method. *Advances in Water Resources*, 2020.
- [106] L. Ma, X. Jia, Q. Sun, B. Schiele, T. Tuytelaars, and L. Van Gool. Pose guided person image generation. *arXiv preprint arXiv:1705.09368*, 2017.
- [107] A. Pandey and D. Wang. On adversarial training and loss functions for speech enhancement. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5414–5418. IEEE, 2018.
- [108] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [109] I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.
- [110] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.
- [111] C. Frogner, C. Zhang, H. Mobahi, M. Araya-Polo, and T. Poggio. Learning with a wasserstein loss. *arXiv preprint arXiv:1506.05439*, 2015.
- [112] RBniCS - reduced order modelling in FEniCS. <https://www.rbnicsproject.org/>, 2015.
- [113] T. Kadeethum, S. Lee, F. Ballarin, J. Choo, and H. Nick. A locally conservative mixed finite element framework for coupled hydro-mechanical-chemical processes in heterogeneous porous media. *Computers & Geosciences*, page 104774, 2021.
- [114] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.
- [115] J. Jensen, L. Lake, P. Corbett, and D. Goggin. *Statistics for petroleum engineers and geoscientists*, volume 2. Gulf Professional Publishing, 2000.
- [116] H. Nick and S. Matthai. A hybrid finite-element finite-volume method with embedded discontinuities for solute transport in heterogeneous media. *Vadose Zone Journal*, 10(1):299–312, 2011.

- [117] P. Salinas, D. Pavlidis, Z. Xie, H. Osman, C. Pain, and M. Jackson. A discontinuous control volume finite element method for multi-phase flow in heterogeneous porous media. *Journal of Computational Physics*, 352:602–614, 2018.