

U-FNO - an enhanced Fourier neural operator based-deep learning model for multiphase flow

Gege Wen^a, Zongyi Li^b, Kamyar Azizzadenesheli^c, Anima Anandkumar^b,
Sally M. Benson^a

^a*Energy Resources Engineering, Stanford University, 367 Panama
St, Stanford, 94305, CA, USA*

^b*Computing and Mathematical Sciences, California Institute of Technology, 1200 E.
California Blvd., MC 305-16, Pasadena, 91125, CA, USA*

^c*Department of Computer Science, Purdue University, 305 N University St, West
Lafayette, 47907, IN, USA*

Abstract

Numerical simulation of multiphase flow in porous media is essential for many geoscience applications. Data-driven machine learning methods provide faster alternatives to traditional simulators by training neural network models with numerical simulation data mappings. Here we present U-FNO, a novel neural network architecture for solving multiphase flow problems with superior speed, accuracy, and data efficiency. U-FNO is designed based on the newly proposed Fourier neural operator (FNO) that learns an infinite dimensional integral kernel in the Fourier space, which has shown excellent performance for single-phase flows. Here we extend the FNO-based architecture to a CO₂-water multiphase problem, and proposes the U-FNO architecture to enhance the prediction accuracy in multiphase flow systems. Through a systematic comparison among a CNN benchmark and three types of FNO variations, we show that the U-FNO architecture has the advantages of both the traditional CNN and original FNO, providing significantly more accurate and efficient performance than previous architectures. The trained U-FNO predicts gas saturation and pressure buildup with a 6×10^4 times speed-up compared to traditional numerical simulators while maintaining similar accuracy. The trained models can act as a general-purpose simulator alternative for 2D-radial CO₂ injection problems with wide ranges of permeability and porosity heterogeneity, anisotropy, reservoir conditions, injection configurations, flow rates, and multiphase flow properties.

Keywords: Multiphase flow, Fourier neural operator, Convolutional neural network, Carbon capture and storage, Deep learning

1. Introduction

Multiphase flow in porous media is important for many geoscience applications, including contaminant transport [1], carbon capture and storage (CCS) [2], hydrogen storage [3], oil and gas extraction [4], and nuclear waste storage [5]. Due to the multi-physics, non-linear, and multi-scale nature of these processes, numerical simulation is the primary approach used to solve mass and energy conservation equations for these applications [6]. These numerical simulations are often very time consuming and computationally intensive since they require fine spatial and temporal discretization to accurately capture the flow processes [7, 8]. Meanwhile, the inherent uncertainty in property distributions of heterogeneous porous medium necessitates probabilistic assessments and inverse modeling to aid engineering decisions [9, 10]. Both of these procedures require large numbers of forward numerical simulation runs and are often prohibitively expensive [11].

A number of machine learning-based methods have been proposed over the past few years to provide faster alternatives to numerical simulation. Most existing machine learning-based methods can be categorized into the following two categories: (1) data-driven finite-dimensional operators that learn Euclidean space mappings from numerical simulation data [12, 13, 14, 15, 16, 17], and (2) physics-informed/ physics-constrained/neural finite difference learning methods that parameterize the solution functions with a neural network [18, 19, 20]. The first type, finite-dimensional operators, is often implemented with convolutional neural networks (CNN). These CNN-based models have been successful in providing fast and accurate predictions for high-dimensional and complex multiphase flow problems [16, 21, 17, 22, 23]. However, CNN-based methods are prone to overfitting, therefore requiring large numerical simulation data sets that can be unmanageable as the problem dimension grows. Also, the results produced by these models are tied to the specific spatial and temporal meshes used in the numerical simulation data set. The second approach using neural finite difference methods requires separate trainings for any new instance of the parameters or coefficients [18] (*e.g.*, new permeability map or injection rate). Therefore, these methods require as much computational effort as traditional numerical solvers, if not

more. Furthermore, these methods often struggle with multiphase flow problems with shock fronts, which are common for many classes of geoscience problems [24, 25].

Recently, a novel approach, the *neural operator*, has been proposed that directly learns the infinite-dimensional-mapping from any functional parametric dependence to the solution [26, 27, 28, 29]. Unlike neural finite difference methods, neural operators are data-driven therefore require training only once. Meanwhile, neural operators are mesh-independent, so they can be trained and evaluated on different grids. Due to the cost of evaluating global neural integral operators, previously proposed neural operators have not yet been able to achieve the desirable degree of computational efficiency [30]. However, one type of neural operator, the Fourier neural operator (FNO), alleviates this issue through the implementation of a Fast Fourier Transform [30]. The FNO has shown excellent performance on single-phase flow problems with great generalization ability, and is significantly more data efficient than CNN-based methods [30].

Here we extend the FNO-based architecture to multiphase flow problems. We find that while FNO’s testing accuracy is generally higher than CNN-based models, the training accuracy is sometimes lower due to the regularization effect of the FNO architecture. To improve upon this, we present an enhanced Fourier neural operator, named U-FNO, that combines the advantages of FNO-based and CNN-based models to provide results that are both highly accurate and data efficient. Through the implementation of the newly proposed U-Fourier layer, we show that the U-FNO model architecture produces superior performance over both the original FNO [30] and a state-of-the-art CNN benchmark [17]. We apply the U-FNO architecture to the highly complex CO₂-and-water multiphase flow problem in the context of CO₂ geological storage to predict dynamic pressure buildup and gas saturation. The trained U-FNO models provide an alternative to numerical simulation for 2D-radial CO₂ injection problems with wide ranges of permeability and porosity heterogeneity, anisotropy, reservoir conditions, injection configurations, flow rates, and multiphase flow properties.

2. Methods

The goal of a neural operator is to learn an infinite-dimensional-space mapping from a finite collection of input-output observations. To formulate the problem, we define the domain $D \subset \mathbb{R}^d$ be a bounded and open set;

\mathcal{A} be the input function space; \mathcal{Z} be the output function space. \mathcal{A} and \mathcal{Z} are separable Banach spaces of functions defined on D that takes values in \mathbb{R}^{d_a} and \mathbb{R}^{d_z} respectively. $\mathcal{G}^\dagger : \mathcal{A} \rightarrow \mathcal{Z}$ is a non-linear map that satisfy the governing PDEs. Suppose we have a_j that are drawn from probability measure μ in \mathcal{A} , then $z_j = \mathcal{G}^\dagger(a_j)$. We aim to build an operator \mathcal{G}_θ that learns an approximation of \mathcal{G}^\dagger by minimizing the following problem using a cost function C .

$$\min_{\theta} \mathbb{E}_{a \sim \mu} [C(\mathcal{G}_\theta(a), \mathcal{G}^\dagger(a))] \quad (1)$$

Since $a_j \in \mathcal{A}$ and $z_j \in \mathcal{Z}$ are both functions, we use n -point discretization $D_j = \{x_1, \dots, x_n\} \subset D$ to numerically represent $a(x)_j|_{D_j} \in \mathbb{R}^{n \times d_a}$ and $z(x)_j|_{D_j} \in \mathbb{R}^{n \times d_z}$. We demonstrate in this section that the proposed U-FNO architecture learns the infinite-dimensional-space mapping \mathcal{G}_θ from a finite collections of $a(x)_j$ and $z(x)_j$ pairs. A table of notation is included in Appendix A.

2.1. U-FNO architecture

The U-FNO architecture contains the following three steps:

1. Lift input observation $a(x)$ to a higher dimensional space $v_{l_0}(x) = P(a(x))$ through a fully connected neural network transformation P .
2. Apply iterative Fourier layers followed by iterative U-Fourier layers: $v_{l_0} \mapsto \dots \mapsto v_{l_L} \mapsto v_{m_0} \mapsto \dots \mapsto v_{m_M}$ where v_{l_j} for $j = 0, 1, \dots, L$ and v_{m_k} for $k = 0, 1, \dots, M - 1$ are sequences of functions taking values in \mathbb{R}^c for channel dimension c .
3. Project v_{m_M} back to the original space $z(x) = Q(v_{m_M}(x))$ using a fully connected neural network transformation Q .

Figure 1A provides a schematic of the U-FNO architecture. Within each newly proposed U-Fourier layer, we have

$$v_{m_{k+1}}(x) := \sigma \left((\mathcal{K}v_{m_k})(x) + (\mathcal{U}v_{m_k})(x) + W(v_{m_k}(x)) \right), \forall x \in D \quad (2)$$

where \mathcal{K} is a kernel integral transformation parameterized by a neural network, \mathcal{U} is a U-Net CNN operator, and W is a linear operator, which are all learnable. σ is a non-linear activation function. Refer to Li et al. [30] for the formulation of the original Fourier layer.

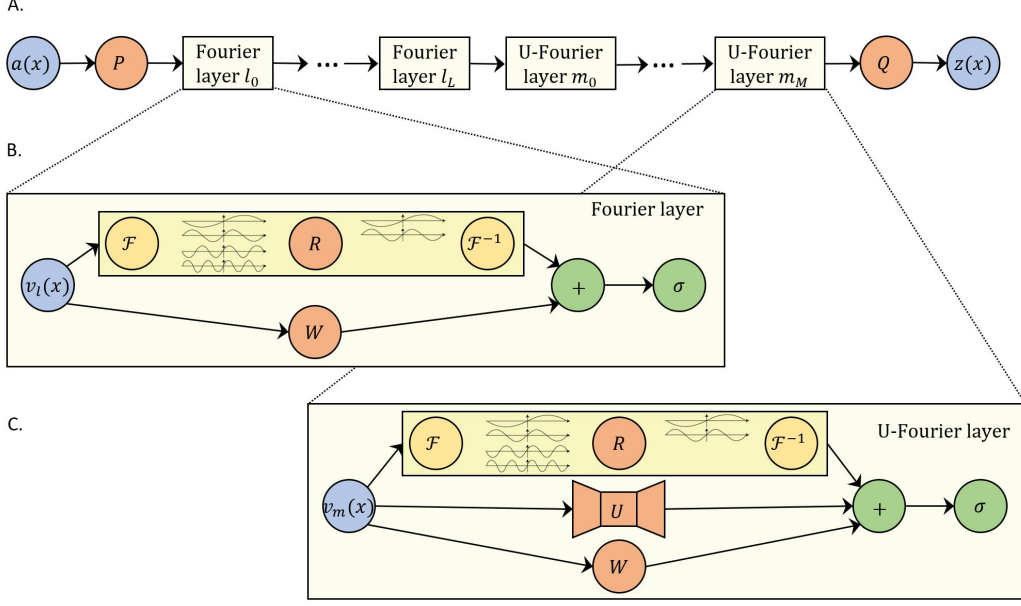


Figure 1: A. U-FNO model architecture. $a(x)$ is the input, P and Q are fully connected neural networks, and $z(x)$ is the output. B. Inside the Fourier layer, \mathcal{F} denotes the Fourier transform, R is the parameterization in Fourier space, \mathcal{F}^{-1} is the inverse Fourier transform, W is a linear bias term, and σ is the activation function. C. Inside the U-FNO layer, U denotes a two step U-Net, the other notations have identical meaning as in the Fourier layer.

2.2. Integral kernel operator in the Fourier space

The integral kernel operator in Equation 2 is defined by

$$(\mathcal{K}(v_l))(x) = \int_D \kappa(x, y) v_l(y) dy, \forall x \in D. \quad (3)$$

To efficiently parameterize kernel κ , the FNO method considers the representation v_l (and also v_m) in the Fourier space and utilizes Fast Fourier Transform (FFT) [30]. By letting $\kappa(x, y) = \kappa(x - y)$ in Equation 3 and applying the convolution theorem, we can obtain

$$(\mathcal{K}(v_l))(x) = \mathcal{F}^{-1}(\mathcal{F}(\kappa) \cdot \mathcal{F}(v_l))(x), \forall x \in D \quad (4)$$

where \mathcal{F} denotes a Fourier transform of a function $f : D \rightarrow \mathbb{R}^c$ and \mathcal{F}^{-1} is its inverse. Now, we can parameterize κ directly by its Fourier coefficients:

$$(\mathcal{K}(v_l))(x) = \mathcal{F}^{-1}(R \cdot \mathcal{F}(v_l))(x), \forall x \in D. \quad (5)$$

where R is the Fourier transform of a periodic function κ . Since we assume that κ is periodic, we can apply a Fourier series expansion and work in the discrete modes of Fourier transform.

We first truncate the Fourier series at a maximum number of modes k_{max} , and then parameterize R directly as a complex valued $(k_{max} \times c \times c)$ -tensor with the truncated Fourier coefficients. As a result, multiplication by the learnable weight tensor R is

$$(R \cdot \mathcal{F}(v_l))_{k,i} = \sum_{j=1}^c R_{k,i,j} (\mathcal{F}(v_l))_{k,j}, \quad \forall k = 1, \dots, k_{max}, \quad i = 1, \dots, c. \quad (6)$$

By replacing the \mathcal{F} by the FFT and implementing R using a direct linear parameterization, we have obtained the Fourier operator as illustrated in Figure 1B and C with nearly linear complexity.

2.3. Characteristics of the U-Fourier layer

In contrast to the original Fourier layer in FNO [30], the U-FNO architecture proposed here appends a U-Net path in each U-Fourier layer. The U-Net processes local convolution to enrich the representation power of the U-FNO in higher frequencies information. The number of Fourier and U-Fourier layers, L and M , are hyperparameters that can be optimized for the specific problem. For the multi-phase flow problem considered here, we found that the architecture with half Fourier layers and half U-Fourier layers achieves the best performance, compared to architectures with all Fourier layers or all U-Fourier layers.

Note that although the Fourier neural operator is an infinite-dimensional-operator, when we append the U-Net block, we sacrifice the flexibility of training and testing the model at different discretizations. We made this choice because the CO₂-water multiphase flow problem is very sensitive to numerical dispersion and numerical dissolution, which are both tied to a specific grid resolution. When training and testing at different grid dimensions, the numerical noise is often transformed in a nonphysical way. As a result, for this problem, we prioritize achieving higher training and testing accuracy, which the U-FNO provides.

3. Problem setting

3.1. Governing equation

We consider a multi-phase flow problem with CO_2 and water in the context of geological storage of CO_2 . The CO_2 and water are immiscible but have mutual solubility. The general forms of mass accumulations for component $\eta = \text{CO}_2$ or *water* are written as [31]:

$$\frac{\partial(\varphi \sum_p S_p \rho_p X_p^{CO_2})}{\partial t} = -\nabla \cdot \left[\mathbf{F}^{CO_2}|_{adv} + \mathbf{F}^{CO_2}|_{dif} \right] + q^{CO_2} \quad (7)$$

$$\frac{\partial(\varphi \sum_p S_p \rho_p X_p^{water})}{\partial t} = -\nabla \cdot \left[\mathbf{F}^{water}|_{adv} + \mathbf{F}^{water}|_{dif} \right]. \quad (8)$$

Here p denotes the phase of w (wetting) or n (non-wetting). In the siliciclastic rocks present at most geological storage sites, water is the wetting phase [32]. However, due to the mutual solubility of water and CO_2 , there is a small amount of CO_2 in the water phase and a small amount of water in the CO_2 phase. Here φ is the porosity, S_p is the saturation of phase p , and X_p^η is the mass fraction of component η in phase p .

For both components, the advective mass flux $\mathbf{F}^\eta|_{adv}$ is obtained by summing over phases p ,

$$\mathbf{F}^\eta|_{adv} = \sum_p X_p^\eta \mathbf{F}_p = \sum_p X_p^\eta \left(-k \frac{k_{r,p} \rho_p}{\mu_p} (\nabla P_p - \rho_p \mathbf{g}) \right) \quad (9)$$

where each individual phase flux \mathbf{F}_p is governed by the multiphase flow extension of Darcy's law. k denotes the absolute permeability, $k_{r,p}$ is the relative permeability of phase p that non-linearly depends on S_p , μ_p is the viscosity of phase p that depends on P_p , and \mathbf{g} is the gravitational acceleration.

Due to the effect of capillarity, the fluid pressure P_p of each phase is

$$P_n = P_w + P_c \quad (10)$$

$$P_w = P_w \quad (11)$$

where the capillary pressure P_c is a non-linear function of S_p . Additionally, porosity φ , density ρ_p , and the solubility of CO_2 in Equation 7 and Equation 8 are also non-linear functions that depend on P_p .

To simplify the problem setting, our simulation does not explicitly include molecular diffusion and hydrodynamic dispersion. However some unavoidable numerical dispersion resulting from approximating spatial gradients using the two-point upstream algorithm [33] is intrinsic to the numerical simulations used for the neural network training.

3.2. Numerical simulation setting

We use the numerical simulator ECLIPSE (e300) to develop the multi-phase flow data set for CO₂ geological storage. ECLIPSE is a full physics simulator that uses the finite difference method with upstream weighting for spatial discretization and the adaptive IMplicit method for temporal discretization [33]. We inject super-critical CO₂ at a constant rate into a radially symmetrical system $x(r, z)$ through a vertical injection well with a radius of 0.1 m. The well can be perforated over the entire thickness of the reservoir or limited to a selected depth interval. We simulate CO₂ injection for 30 years at a constant rate ranging from 0.2 to 2 Mt/year. The thickness of the reservoir ranges from 12 to 200 m with no-flow boundaries on the top and bottom. We use a vertical cell dimension of 2.08 m to capture the vertical heterogeneity of the reservoir. The radius of the reservoir is 100,000 m. The outer boundary is closed, but is sufficiently distant from the injection well that it behaves like an infinite acting reservoir. Two hundred gradually coarsened grid cells are used in the radial direction. Grid sensitivity studies show that this grid is sufficiently refined to capture the CO₂ plume migration and pressure buildup, while remaining computationally tractable [8]. Simulated values of the gas saturation (SG) and pressure buildup (dP) fields at 24 gradually coarsening time snapshots are used for training the neural nets. Refer to Appendix B for detailed spatial and temporal discretizations.

3.3. Variable sampling scheme

We sample two types of variables for each numerical simulation case: field variables and scalar variables. As shown in Figure 2, field variables include the horizontal permeability map (k_x), vertical permeability map (k_y), porosity map (ϕ), and injection perforation map ($perf$). The reservoir thickness b is randomly sampled in each simulation case and controls the reservoir dimension in each of the following field variables:

- k_x : The Stanford Geostatistical Modeling Software (SGeMS) [34] is used to generate the heterogeneous k_x maps. SGeMS produces permeability map according to required input parameters such as correlation

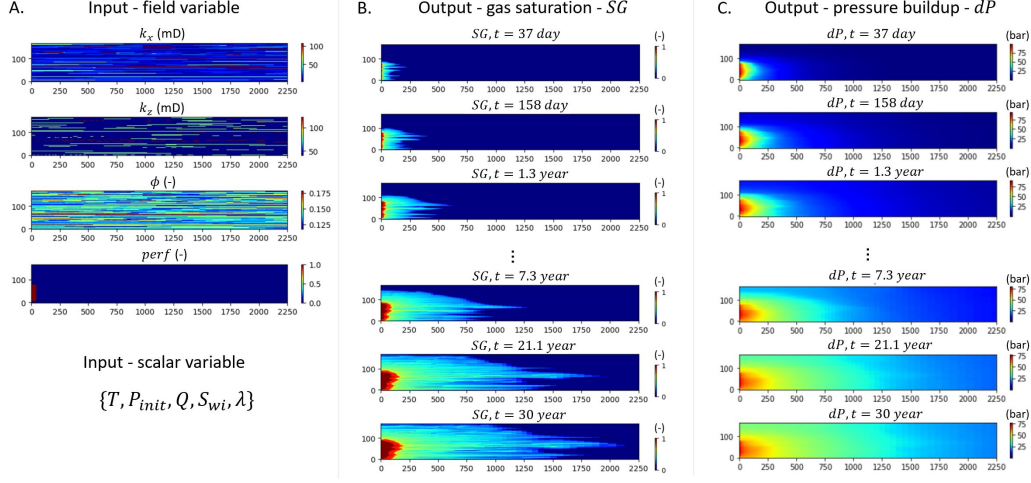


Figure 2: Example of mapping between A. input to B. output gas saturation and C. pressure buildup. A. Field and scalar channels for each case. Note that the scalar variables are broadcasted into a channel at the same dimension as the field channels. B. Gas saturation evolution for 6 out of 24 time snapshots. C. Pressure buildup evolution for 6 out of 24 time snapshots.

lengths in the vertical and radial directions, medium appearances (Appendix C), as well as permeability mean and standard deviation. A wide variety of permeability maps representing different depositional environments are included in the data set and the permeability value ranges widely from 10 Darcy to 0.001 mD. Appendix C summarizes statistical parameters that characterize the permeability maps.

- k_y : The vertical permeability map is calculated by multiplying the k_x map by the anisotropy map. To generate the anisotropy map, values of k_x are binned into n_{aniso} materials, each of which is assigned a randomly sampled anisotropy ratio. Note that the anisotropy ratio is uncorrelated with the magnitude of the radial permeability. This procedure roughly mimics a facies-based approach for assigning anisotropy values.
- ϕ : Previous studies show that porosity and permeability are loosely correlated with each other [35]. Therefore, to calculate porosity we first use the fitting relationship presented in Pape et al [35] and then perturb these values with a random Gaussian noise ϵ with mean value of zero and standard deviation of 0.005.

Table 1: Summary of input variable’s type, sampling range, distribution, and unit. All input sampling are independent with the exception of porosity and vertical permeability map. *: refer to Appendix C for a detailed statistical parameter summary for generating heterogeneous k_x map.

variable type	sampling parameter	notation	distribution	unit
field	horizontal permeability field	k_x	heterogeneous*	-
	# of anisotropic materials	n_{aniso}	$X \sim \mathcal{U}\{1, 6\}$	-
	material anisotropy ratio	k_x/k_y	$X \sim \mathcal{U}[1, 150]$	-
	porosity (perturbation)	ϕ	$\epsilon \sim \mathcal{N}(0, 0.005)$	-
	reservoir thickness	b	$X \sim \mathcal{U}[12, 200]$	m
	perforation thickness	b_{perf}	$X \sim \mathcal{U}[12, b]$	m
	perforation location	-	randomly placed	-
scalar	injection rate	Q	$X \sim \mathcal{U}[0.2, 2]$	MT/y
	initial pressure	P_{init}	$X \sim \mathcal{U}[100, 300]$	bar
	iso-thermal reservoir temperature	T	$X \sim \mathcal{U}[35, 170]$	°C
	irreducible water saturation	S_{wi}	$X \sim \mathcal{U}[0.1, 0.3]$	-
	van Genuchten scaling factor	λ	$X \sim \mathcal{U}[0.3, 0.7]$	-

- *perf*: The injection interval thickness b_{perf} is randomly sampled within the range from 12 m to the specific reservoir thickness b of that case. We placed the perforation interval on the injection well, by randomly sampling the depth of the perforation top from 0 m to $(b - b_{perf})$ m.

Visualizations of the above field variable are shown in Appendix C. Table 1 summarizes the parameter sampling ranges and distributions. The sampling parameters are independent of each other with the exception of porosity and permeability.

Scalar variables include the initial reservoir pressure at the top of the reservoir (P_{init}), reservoir temperature (T), injection rate (Q), capillary pressure scaling factor (λ) [36], and irreducible water saturation (S_{wi}). The parameter sampling range and distributions are summarized in Table 1. While the scalar variables P_{init} and T are determined independently, cases that yield unrealistic combinations of these variables are excluded. These field and scalar input variables create a very high-dimensional input space, which is very challenging for traditional CNN-based models. Nevertheless, the U-FNO model architecture handles the high-dimensional input space with excellent data efficiency.

3.4. Data configuration

Each of the field variables in Figure 2A is represented by a channel in the data input. Since we use a gradually coarsening radial grid for the numerical simulations, a logarithm conversion in the radial direction is applied in training to project the field variables onto a uniform grid that can be represented by a (96, 200) matrix. Notice that reservoir thickness is also a variable and 96 cells represents a 200m thick reservoir. When the reservoir is thinner than 200 m, we use zero-padding to denote cells that are outside of the actual reservoir. For the scalar variables, the values are simply broadcast into a matrix with dimension of (96, 200).

In addition to the input variables, we also supply the spatial grid information to the training by using one channel to denote radial cell dimensions and another channel to denote vertical cell dimensions. The temporal grid information is supplied into the network as an additional dimension. The input to each data sample is constructed by concatenating the field variables, scalar variables, spatial grids, and temporal grid together.

For the gas saturation and pressure buildup outputs as shown in Figure 2B and C, we use the same logarithm conversion to project the outputs onto a uniform grid. We then concatenate the outputs for different time snapshots to obtain a spatial-temporal 3D volume. The pressure buildup is normalized into zero-mean and unit-variance distribution. For gas saturation, we do not normalize the data because the saturation values always range from 0 to 1. The dimensions of the input and outputs are shown for in each model architecture (Appendices D to G).

The data set contains 5,000 input-to-output mappings. We use a 9/1 split to segregate the data set into 4,500 samples for training and 500 samples for testing.

3.5. Loss function design and training

We use a relative lp -loss to train the deep learning models. The lp -loss is applied to both the original output (y) and the first derivative of the output in the r -direction (dy/dr), and is written as:

$$L(y, \hat{y}) = \frac{\|y - \hat{y}\|_p}{\|y\|_p} + \beta \frac{\|dy/dr - d\hat{y}/dr\|_p}{\|dy/dr\|_p}, \quad (12)$$

where \hat{y} is the predicted output, dy/dr is the first derivative of the predicted output, p is the order of norm, and β is a hyper-parameter. This relative

loss has a regularization effect and is particularly effective when the data have large variances on the norms. Our experiments show that, compared to an MSE -loss, a relative loss significantly improves the performance for both gas saturation and pressure buildup. The second term in Equation 12 greatly improves quality of predictions for gas saturation at the leading edge of the plume. Similarly this term improves prediction of the sharp pressure buildup around the injection well. We use the l_2 -loss for gas saturation and pressure buildup since it provides faster convergence than the l_1 -loss.

As described in Section 3, our data set contains reservoirs with various thicknesses and the cells outside of the reservoir are padded with zeros for both input and output. To accommodate for the variable reservoir thicknesses, during training, we construct an active cell mask for each data sample and only calculate the loss within the mask. Our experiments show that this loss calculation scheme achieves better performance than calculating the whole field because of the better gradient distribution efficiency.

During training, the initial learning rate is 0.001 and the learning rate gradually decreases with a constant step and reduction rate. These hyperparameters are optimized for the gas saturation and pressure buildup model separately. The training stops when the loss no longer decreases.

4. Results

This section compares 4 types of model architectures: original FNO proposed in Li et al. [30], the newly proposed U-FNO in this paper, a conv-FNO that uses a `conv3d` in the place of the U-Net, and the state-of-the-art benchmark CNN used in Wen et al. [17]. All models are trained on the proposed loss function (Equation 12) and directly output the 3D ($96 \times 200 \times 24$) gas saturation and pressure field in space and time. Detailed parameters for each model are summarized in Appendices D to G.

4.1. Gas saturation

Figure 3A and Table 2 demonstrates that the best performance for both the training and testing data set is achieved with the U-FNO model. Specifically, the testing set performance represents the true predictability of the model for unseen data. The low relative loss clearly indicates the superior performance of the proposed U-FNO.

Interestingly, we notice that although the original FNO has a higher training relative loss than the CNN benchmark, the testing relative loss by the

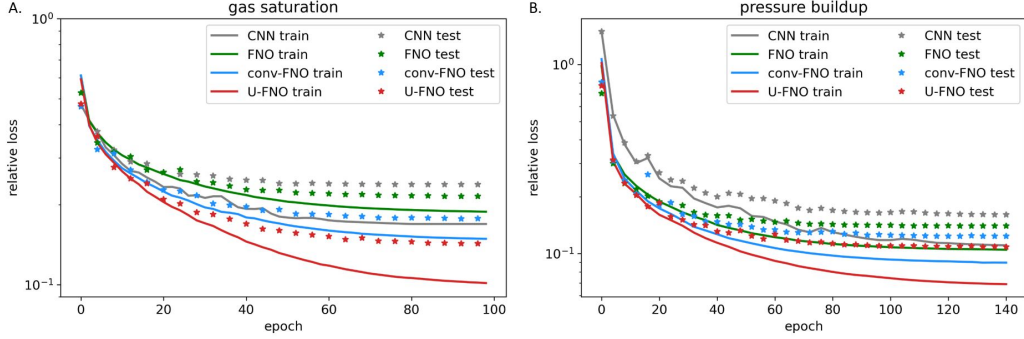


Figure 3: Training and testing relative loss evolution vs. epoch for U-FNO, FNO, conv-FNO and CNN benchmark for A. gas saturation and B. pressure buildup.

original FNO is lower than that of the CNN benchmark. This indicates that FNO has excellent generalization ability and achieves better performance than the CNN even though FNO has a higher training relative loss. Nevertheless, the original FNO has the highest relative loss in the training set due to the inherent regularization effect by using a finite set of truncated Fourier basis. The Conv-FNO and U-FNO architecture is therefore designed to enhance the expressiveness by processing the higher frequency information that are not picked up by the Fourier basis. We can observe from Figure 3A that the training loss is significantly improved even by simply adding a plain `conv3d` in the Conv-FNO case. When the FNO layer is combined with a U-Net in the U-FNO case, the model takes the advantages of both architectures and consistently produces the lowest relative loss throughout the entire training (Figure 3A).

Table 2: Average training and testing data set R^2 scores of 500 random samples for 4 models. R^2 scores measures the similarity between the prediction and the numerical simulation output; an R^2 of 1 indicates an exact match.

Model	Gas saturation (SG)				Pressure buildup (dP)			
	Train		Test		Train		Test	
	mean	std	mean	std	mean	std	mean	std
CNN	0.994	0.007	0.986	0.018	0.990	0.029	0.988	0.023
FNO	0.990	0.013	0.986	0.017	0.991	0.014	0.990	0.017
Conv-FNO	0.993	0.011	0.990	0.013	0.992	0.018	0.991	0.018
U-FNO	0.997	0.005	0.994	0.009	0.994	0.014	0.993	0.015

In addition to considering the average performance over the entire training and testing sets, we compare model predictions for four different cases with varying degrees of complexity in Figure 4. For each case, Figure 4 shows

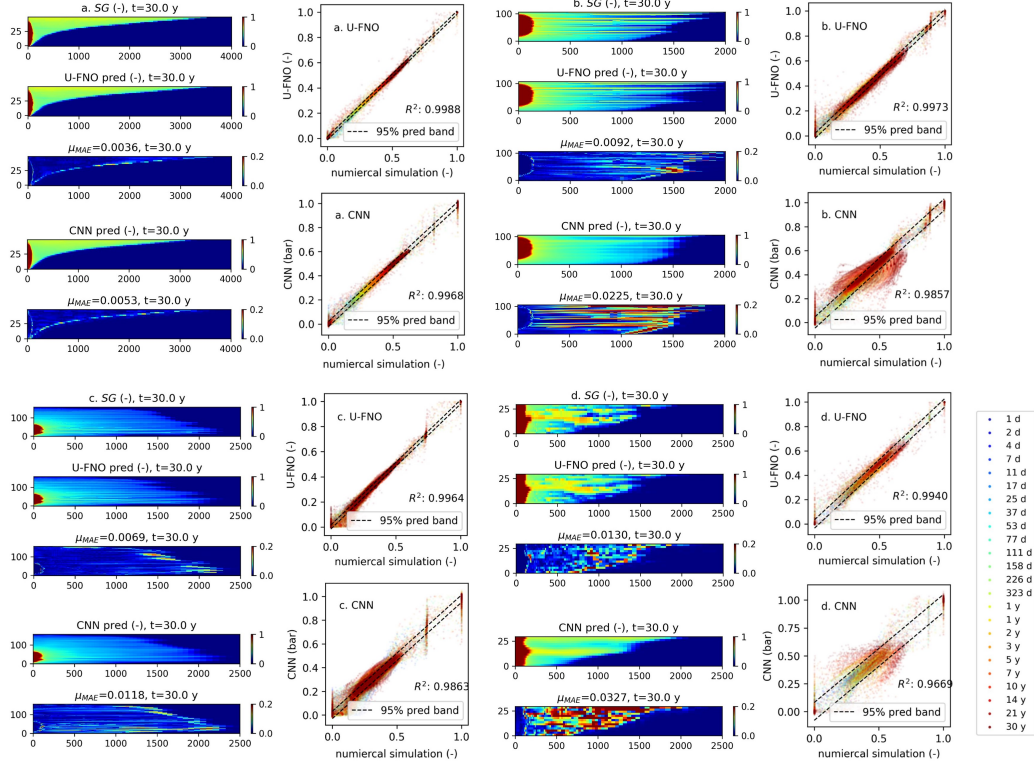


Figure 4: Visualizations and scatter plots for example a to d. In each example, visualizations show the true gas saturation (SG), U-FNO predicted, U-FNO absolute error, CNN predicted, and CNN absolute error. The mean absolute error μ_{MAE} is labeled on the U-FNO and CNN absolute error plots. Scatter plots shows numerical simulation vs. predicted by U-FNO and CNN model on each grid. The legend for all of the scatter plots is shown in the bottom right.

a comparison between the predicted and true values of the CO_2 saturation for each grid cell in the model over the entire 30 year injection period. The U-FNO has superior performance compared to the CNN for all of these examples as quantified by the higher R^2 value and narrower 95% prediction bands. Case b. and d. are especially obvious examples in which the U-FNO successfully predicts the complicated horizontal saturation variations where the CNN ignores the heterogeneity and simply predicts more uniform saturation fields.

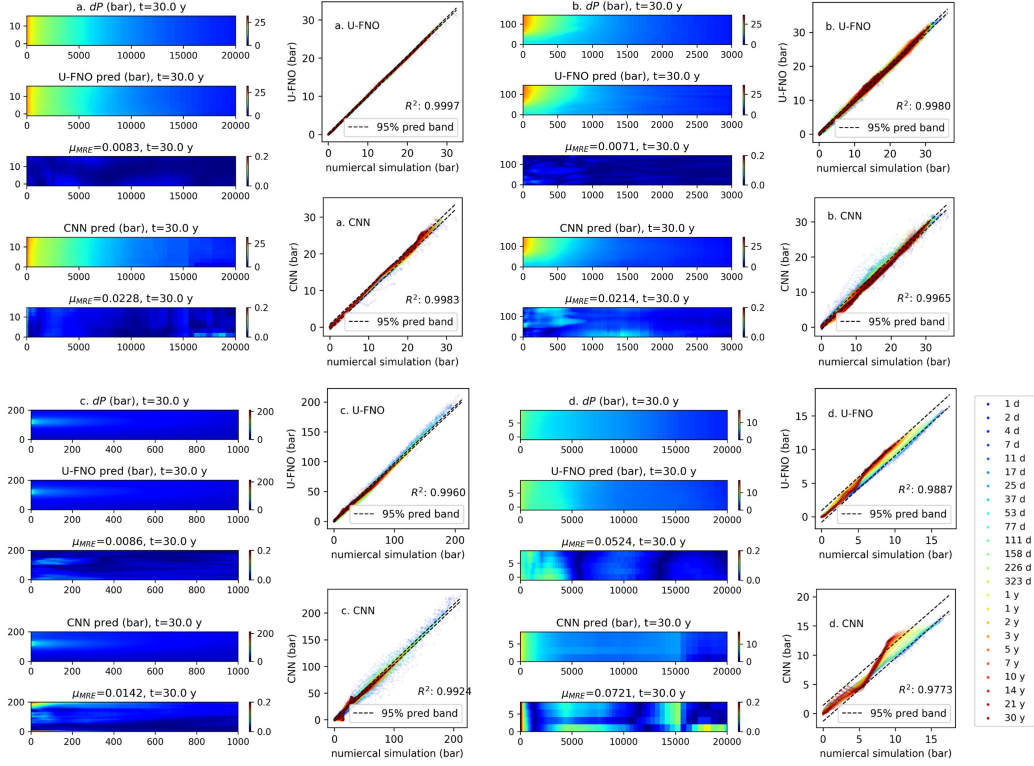


Figure 5: Visualizations and scatter plots for examples a to d. In each example, visualizations show the true pressure buildup (dP), U-FNO predicted, U-FNO relative error, CNN predicted, and CNN relative errors. The relative errors are defined as in [22]; the mean relative error μ_{MPE} is labeled on the U-FNO and CNN relative error plots. Scatter plots shows numerical simulation vs. predicted by U-FNO and CNN model on each grid. The legend for all of the scatter plots is shown in the bottom right.

4.2. Pressure buildup

For pressure buildup, the U-FNO also achieves the lowest relative error for both training and testing data sets. As shown in Figure 3B, the training and testing relative errors for the U-FNO are consistently low throughout the training process. Generally, pressure buildup models are less prone to overfitting than gas saturation models since pressure buildup fields are less heterogeneous. By comparing the R^2 scores for the training and testing sets in Table 2, we can observe that all FNO-based models produce nearly negligible overfitting.

The superior performance of the U-FNO for pressure buildup predictions is also demonstrated for the four examples shown in Figure 5. In each case

the U-FNO has higher R^2 values and narrower 95% prediction bands. Unlike the gas saturation outputs, pressure buildup distributions are challenging to predict since they have a larger radius of influence and larger differences between cases. For example, the maximum pressure buildup in the 4 examples shown in Figure 5 varies from ~ 20 bar to ~ 220 bar. Notice that the the CNN model especially struggles with cases that have large radius of influence (e.g. case d) while the U-FNO model maintains excellent accuracy at locations that are far away from the injection well.

5. Discussion

5.1. Training size analysis

The results in Section 4 demonstrate the excellent generalization ability of all FNO-based architectures. To further investigate the relationship between the training size and overfitting for the newly proposed U-FNO model, we run a set of comparisons for the U-FNO model trained with 500, 2,500 and 4,500 samples (see Figure 6). Each model is trained for the same number of epochs. From this comparison, we can observe that the gas saturation models are more prone to overfitting as indicated by the wider gaps in R^2 mean and standard deviations. For the pressure buildup, the training size of 4,500 produces a very similar R^2 score in the training and testing set. Therefore, a 4,500 training data-size is already sufficient for pressure buildup prediction and more training data will not significantly improve the performance.

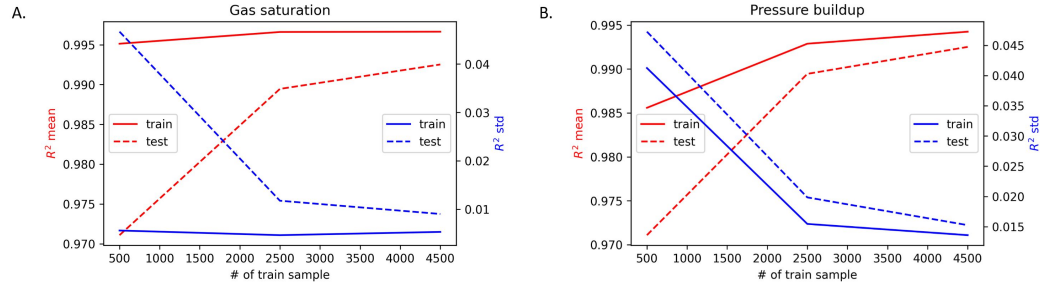


Figure 6: The training and testing set R^2 mean and standard deviation vs. the number of training sample for A. Gas saturation and B. pressure buildup.

5.2. Computational efficiency analysis

We summarize the computational efficiency of the CNN, FNO, Conv-FNO, and U-FNO in Table 3. The training and testing times are both evaluated on a Nvidia A100-SXM GPU. For the comparison, we run ECLIPSE simulations on an Intel® Xeon® Processor E5-2670 CPU. Each simulation case can utilize a fully dedicated CPU and the average run time for 1,000 random cases is 10 minutes. We run the comparison with the CPU that’s readily available to us. Other types of CPU will not change the simulation time for a significant order.

Table 3: Summary of the number of parameters, training time, and testing times required for all four models. The testing times are calculated by taking the average of 500 random cases. The speed-up is compared with average numerical simulation run time of 10 mins.

	# Parameter (-)	Training (s/epoch)	Testing		
			Gas saturation (s)	Pressure buildup (s)	Speed-up vs. numerical simulation (times)
CNN	33,316,481	562	0.050	0.050	1×10^4
FNO	31,117,541	711	0.005	0.005	1×10^5
Conv-FNO	31,222,625	1,135	0.006	0.006	1×10^5
U-FNO	33,097,829	1,872	0.010	0.010	6×10^4

All of the neural network models are at least 10^4 times faster than conventional numerical simulation during prediction. Notice that FNO-based models are significantly faster than the CNN model at testing time while slower at training time. In this problem, we prioritize the prediction accuracy and testing time over the training time, which U-FNO provides. For problems that are more sensitive to training time, one could also use the Conv-FNO which provides both high accuracy and relatively fast training.

5.3. Fourier kernel visualization

As described in Section 2, the Fourier path within each U-Fourier layer contains trainable kernel R that is parameterized in the Fourier space. Here we provide visualizations for a random selection of the Fourier kernels in the trained gas saturation and pressure buildup models. Notice that unlike traditional CNN kernels that are generally small (*e.g.*, $(3, 3, 3)$ or $(7, 7, 7)$), Fourier kernels are full field kernels that can be interpreted by any grid discretization. The kernels in this paper are 3D kernels with dimensions (r, z, t) and the examples shown in Figure 7 are the (r, z) directional slices evaluated using the data discretization. Both gas saturation and pressure buildup models contain a wide variety of kernels from low to high frequency.

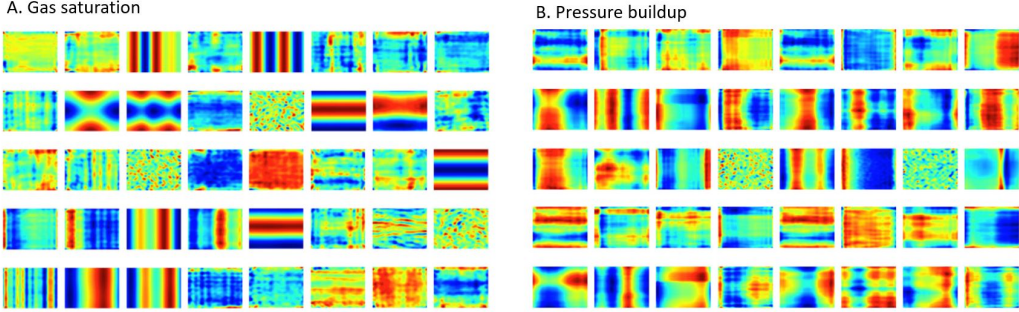


Figure 7: Visualizations of random selections of (r, z) directional kernels for trained A. gas saturation and B. pressure buildup models.

We hypothesize that the asymmetry in the r direction might be related to the gradually coarsening r -directional grid resolution, while the asymmetry in the z direction might be related to the effects of buoyancy since CO_2 is less dense than water and tends to migrate to the top of the reservoir.

6. Conclusion

This paper presents U-FNO, an enhanced Fourier neural operator for solving multiphase flow problems. We demonstrate that U-FNO predicts highly accurate flow outputs for a complex CO_2 -water multiphase flow problem in the context of CO_2 geological storage.

Through comparisons with the original FNO architecture [30] and a state-of-the-art CNN benchmark [17], we show that the newly proposed U-FNO architecture provides the best performance for both gas saturation and pressure buildup predictions. The U-FNO architecture enhances the training accuracy of a original FNO. At the same time, U-FNO maintains the excellent generalizability of the original FNO architecture. For the CO_2 -water multiphase flow application described here, our goal is to optimize for the accuracy of gas saturation and pressure fields, for which the U-FNO provides the highest performance.

The trained U-FNO model generates gas saturation and pressure buildup prediction that are 10^4 times faster than a traditional numerical solver. The significant improvement in the computational efficiency can support many engineering tasks that requires repetitive forward numerical simulations. For example, the trained U-FNO model can serve as an alternative to full physics

numerical simulators in probabilistic assessment, inversion, and site selection, tasks that were prohibitively expensive with desirable grid resolution using numerical simulation.

Code and data availability

The python code for U-FNO model architecture and the data set used in training will be released upon the publication of this manuscript. A web app hosting the pre-trained models for gas saturation and pressure buildup will be released upon the publication of this manuscript.

Acknowledgments

G. Wen and S. M. Benson gratefully acknowledges the supported by ExxonMobil through the Strategic Energy Alliance at Stanford University and the Stanford Center for Carbon Storage. Z. Li gratefully acknowledges the financial support from the Kortschak Scholars Program. A. Anandkumar is supported in part by Bren endowed chair, LwLL grants, Beyond Limits, Raytheon, Microsoft, Google, Adobe faculty fellowships, and DE Logi grant. The authors would like to acknowledge Peter Messmer for his comments and pointing out the typos.

Appendix A. Table of notations

Table A.4: Table of notations.

	Notation	Meaning
Operator learning	$D \in \mathbb{R}^d$	The spatial domain for the problem
	$a \in \mathcal{A} = D; \mathbb{R}^{d_a}$	Input coefficient functions
	$z \in \mathcal{Z} = D; \mathbb{R}^{d_z}$	Target solution functions
	$\mathcal{G}^\dagger : \mathcal{A} \rightarrow \mathcal{Z}$	The operator mapping from coefficients to solutions
	n	The size of the discretization
	\mathbf{x}	Points in the spatial domain
	$D_j = \{x_1, \dots, x_n\} \subset D$	The discretization of (a_j, u_j)
	\mathcal{G}_θ	An approximation of \mathcal{G}^\dagger
	μ	A probability measure where a_j is sampled from
	C	Cost function
U-FNO	$a(x)$	The discretized data input
	$z(x)$	The discretized data output
	$v_{l_j}(x), j = 0, \dots, L$	High dimensional representation of $a(x)$ in Fourier layers
	$v_{m_k}(x), k = 0, \dots, M - 1$	High dimensional representation of $a(x)$ in U-Fourier layers
	$Q(\cdot)$	The lifting neural network
U-Fourier layer	$P(\cdot)$	The projection neural network
	\mathcal{K}	The Kernel integral operator applied on v_l and v_m
	R	The linear transformation applied on the lower Fourier modes
	W	The linear transformation (bias term) applied on the spatial domain
	U	The U-Net operator applied on v_l and v_m
	σ	The activation function
	$\mathcal{F}, \mathcal{F}^{-1}$	Fourier transformation and its inverse
	κ	The kernel function learned from data
	k_{max}	The maximum number of modes
Governing equation	c	The number of channels
	$\eta = CO_2, water$	Components of CO ₂ and water
	$p = w, n$	Phases of wetting and non-wetting
	φ	The pore volume
	t	Time
	S_p	The saturation of phase p
	ρ_p	The density of phase p
	X_p	The mass fraction of phase p
	\mathbf{F}	Flux
	q	The source term
	P_p	The pressure of phase p
	k	The absolute permeability
	$k_{r,p}$	The relative permeability of phase p
	μ_p	The viscosity of phase p
	\mathbf{g}	Gravitational acceleration
Sampling variable	refer to Table 1	

Appendix B. Grid discretization

Table B.5: Vertical, radial, and temporal grid discretization for ECLIPSE numerical simulation runs. The radial grid width gradually coarsens as $d_{r_{min}} \times a_r^{j-1}$, for $j \in [1, \dots, i_r]$. The temporal step size gradually coarsens as $d_{t_{min}} \times a_t^{j-1}$, for $j \in [1, \dots, i_t]$.

Dimension	Parameter	Notation	Value	Unit
Vertical (z)	box boundary	z_{max}	12 to 200	m
	grid count	i_z	6 to 96	-
	grid thickness	d_z	2.08	m
Radial (r)	box boundary	r_{max}	1,000,000	m
	grid count	i_r	200	-
	minimum grid width	$d_{r_{min}}$	3.6	m
	amplification factor	a_r	1.035012	-
	well radius	r_{well}	0.1	m
Temporal (t)	total length	t_{max}	30	years
	step count	i_t	24	-
	minimum step	$d_{t_{min}}$	1	day
	amplification factor	a_t	1.421245	-

Appendix C. Heterogeneous permeability map statistical parameters and visualizations

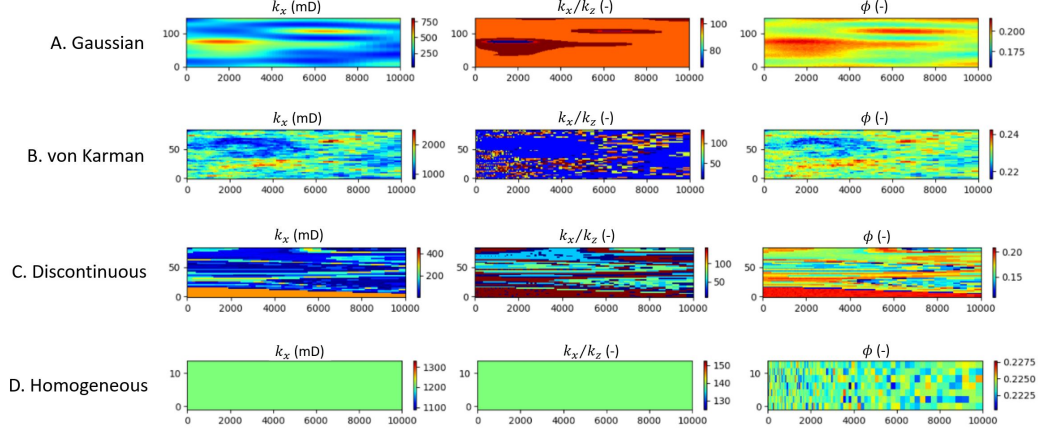


Figure C.8: Horizontal permeability map, anisotropy map, and porosity map for A. Gaussian, B. von Karman, C. Discontinuous, and D. Homogeneous medium appearances.

Table C.6: Statistical parameters of horizontal permeability (k_x) maps generated by Stanford Geostatistical Modeling Software (SGeMS) [34]. We defined the medium appearance, spatial correlation, mean, standard deviation, and contrast ratio (k_{high}/k_{low}) in each map to create a large variety of permeability maps.

Medium	Parameter	Mean	Std	Max	Min	Unit
A. Gaussian	Field average	30.8	58.3	1053	0.3	mD
	Vertical correlation	7.3	3.6	12.5	2.1	m
	Horizontal correlation	2190	1432	6250	208	m
	Contrast ratio	4.01×10^4	2.19×10^5	3.00×10^6	1.01	-
B. von Karman [37]	Field average	39.9	54.4	867.9	1.8	mD
	Vertical correlation	7.2	3.5	12.5	2.1	m
	Horizontal correlation	2.15×10^4	1.40×10^4	6.23×10^4	208	m
	Contrast ratio	2.66×10^4	1.54×10^5	2.12×10^6	1.00	-
C. Discontinuous	Field average	80.8	260.2	5281	2.0	mD
	Vertical correlation	7.2	3.6	12.5	2.1	m
	Horizontal correlation	2176	1429	6250	208	m
	Contrast ratio	2.17×10^4	1.51×10^5	2.68×10^6	1.01	-
D. Homogeneous	Field permeability	327.7	478.1	1216	4.0	mD

Appendix D. CNN benchmark model architecture

Table D.7: CNN architecture. **Conv3D** denotes a 3D convolutional layer; **BN** denotes a batch normalization layer; **ReLU** denotes a rectified linear layer; **Add** denotes an addition with the identity; **UnSampling** denotes an unSampling layer that expands the matrix dimension using nearest neighbor method, and **Padding** denotes a padding layer using the reflection padding technique. In this model, the number of total parameters is 33,316,481 with 33,305,857 trainable parameters and 10,624 non-trainable parameters. To ensure a fair comparison with the FNO-based models, we performed hyper-parameter optimization on the CNN benchmark model and trained it with the same loss function (Equation 12) as the FNO-based models.

Part	Layer	Output Shape
Input	-	(96,200,24,1)
Encode 1	Conv3D/BN/ReLU	(48,100,12,32)
Encode 2	Conv3D/BN/ReLU	(48,100,12,64)
Encode 3	Conv3D/BN/ReLU	(24,50,6,128)
Encode 4	Conv3D/BN/ReLU	(24,50,6,128)
Encode 5	Conv3D/BN/ReLU	(12,25,3,256)
Encode 6	Conv3D/BN/ReLU	(12,25,3,256)
ResConv 1	Conv3D/BN/Conv3D/BN/ReLU/Add	(12,25,3,256)
ResConv 2	Conv3D/BN/Conv3D/BN/ReLU/Add	(12,25,3,256)
ResConv 3	Conv3D/BN/Conv3D/BN/ReLU/Add	(12,25,3,256)
ResConv 4	Conv3D/BN/Conv3D/BN/ReLU/Add	(12,25,3,256)
ResConv 5	Conv3D/BN/Conv3D/BN/ReLU/Add	(12,25,3,256)
Decode 6	UnSampling/Padding/Conv3D/BN/Relu	(12,25,3,256)
Decode 5	UnSampling/Padding/Conv3D/BN/Relu	(24,50,6,256)
Decode 4	UnSampling/Padding/Conv3D/BN/Relu	(24,50,6,128)
Decode 3	UnSampling/Padding/Conv3D/BN/Relu	(48,100,12,128)
Decode 2	UnSampling/Padding/Conv3D/BN/Relu	(48,100,12,64)
Decode 1	UnSampling/Padding/Conv3D/BN/Relu	(96,200,24,32)
Output	Conv3D	(96,200,24,1)

Appendix E. FNO model architecture

Table E.8: FNO model architecture. The **Padding** denotes a padding operator that accommodates the non-periodic boundaries; **Linear** denotes the linear transformation to lift the input to the high dimensional space, and the projection back to original space; **Fourier3d** denotes the 3D Fourier operator; **Conv1d** denotes the bias term; **Add** operation adds the outputs together; **ReLu** denotes a rectified linear layer. In this model, the number of total parameters is 31,117,541.

Part	Layer	Output Shape
Input	-	(96,200,24,12)
Padding	Padding	(104, 208, 32, 12)
Lifting	Linear	(104, 208, 32, 36)
Fourier 1	Fourier3d/Conv1d/Add/ReLu	(104, 208, 32, 36)
Fourier 2	Fourier3d/Conv1d/Add/ReLu	(104, 208, 32, 36)
Fourier 3	Fourier3d/Conv1d/Add/ReLu	(104, 208, 32, 36)
Fourier 4	Fourier3d/Conv1d/Add/ReLu	(104, 208, 32, 36)
Fourier 5	Fourier3d/Conv1d/Add/ReLu	(104, 208, 32, 36)
Fourier 6	Fourier3d/Conv1d/Add/ReLu	(104, 208, 32, 36)
Projection 1	Linear	(104, 208, 32, 128)
Projection 2	Linear	(104, 208, 32, 1)
De-padding	-	(96, 200, 24, 1)

Appendix F. Conv-FNO model architecture

Table F.9: Conv-FNO model architecture. The **Padding** denotes a padding operator that accommodates the non-periodic boundaries; **Linear** denotes the linear transformation to lift the input to the high dimensional space, and the projection back to original space; **Fourier3d** denotes the 3D Fourier operator; **Conv1d** denotes the bias term; **Conv3d** denotes a 3D convolutional operator; **Add** operation adds the outputs together; **ReLu** denotes a rectified linear layer. In this model, the number of total parameters is 31,222,625.

Part	Layer	Output Shape
Input	-	(96,200,24,12)
Padding	Padding	(104, 208, 32, 12)
Lifting	Linear	(104, 208, 32, 36)
Fourier 1	Fourier3d/Conv1d/Add/ReLu	(104, 208, 32, 36)
Fourier 2	Fourier3d/Conv1d/Add/ReLu	(104, 208, 32, 36)
Fourier 3	Fourier3d/Conv1d/Add/ReLu	(104, 208, 32, 36)
Conv-Fourier 1	Fourier3d/Conv1d/Conv3d/Add/ReLu	(104, 208, 32, 36)
Conv-Fourier 2	Fourier3d/Conv1d/Conv3d/Add/ReLu	(104, 208, 32, 36)
Conv-Fourier 3	Fourier3d/Conv1d/Conv3d/Add/ReLu	(104, 208, 32, 36)
Projection 1	Linear	(104, 208, 32, 128)
Projection 2	Linear	(104, 208, 32, 1)
De-padding	-	(96, 200, 24, 1)

Appendix G. U-FNO model architecture

Table G.10: U-FNO model architecture. The **Padding** denotes a padding operator that accommodates the non-periodic boundaries; **Linear** denotes the linear transformation to lift the input to the high dimensional space, and the projection back to original space; **Fourier3d** denotes the 3D Fourier operator; **Conv1d** denotes the bias term; **UNet3d** denotes a two step 3D U-Net; **Add** operation adds the outputs together; **ReLu** denotes a rectified linear layer. In this model, the number of total parameters is 33,097,829.

Part	Layer	Output Shape
Input	-	(96,200,24,12)
Padding	Padding	(104, 208, 32, 12)
Lifting	Linear	(104, 208, 32, 36)
Fourier 1	Fourier3d/Conv1d/Add/ReLu	(104, 208, 32, 36)
Fourier 2	Fourier3d/Conv1d/Add/ReLu	(104, 208, 32, 36)
Fourier 3	Fourier3d/Conv1d/Add/ReLu	(104, 208, 32, 36)
U-Fourier 1	Fourier3d/Conv1d/UNet3d/Add/ReLu	(104, 208, 32, 36)
U-Fourier 2	Fourier3d/Conv1d/UNet3d/Add/ReLu	(104, 208, 32, 36)
U-Fourier 3	Fourier3d/Conv1d/UNet3d/Add/ReLu	(104, 208, 32, 36)
Projection 1	Linear	(104, 208, 32, 128)
Projection 2	Linear	(104, 208, 32, 1)
De-padding	-	(96, 200, 24, 1)

References

- [1] J. Bear, A. H.-D. Cheng, Modeling groundwater flow and contaminant transport, Vol. 23, Springer, 2010.
- [2] R. K. Pachauri, M. R. Allen, V. R. Barros, J. Broome, W. Cramer, R. Christ, J. A. Church, L. Clarke, Q. Dahe, P. Dasgupta, et al., Climate change 2014: synthesis report. Contribution of Working Groups I, II and III to the fifth assessment report of the Intergovernmental Panel on Climate Change, Ipcc, 2014.
- [3] L. Hashemi, M. Blunt, H. Hajibeygi, Pore-scale modelling and sensitivity analyses of hydrogen-brine multiphase flow in geological porous media, Scientific reports 11 (1) (2021) 1–13.
- [4] K. Aziz, Petroleum reservoir simulation, Applied Science Publishers 476 (1979).

- [5] B. Amaziane, M. El Ossmani, M. Jurak, Numerical simulation of gas migration through engineered and geological barriers for a deep repository for radioactive waste, *Computing and Visualization in Science* 15 (1) (2012) 3–20.
- [6] F. M. Orr, et al., *Theory of gas injection processes*, Vol. 5, Tie-Line Publications Copenhagen, 2007.
- [7] C. Doughty, Investigation of co₂ plume behavior for a large-scale pilot test of geologic carbon storage in a saline formation, *Transport in porous media* 82 (1) (2010) 49–76.
- [8] G. Wen, S. M. Benson, CO₂ plume migration and dissolution in layered reservoirs, *International Journal of Greenhouse Gas Control* 87 (May) (2019) 66–79. doi:10.1016/j.ijggc.2019.05.012.
URL <https://linkinghub.elsevier.com/retrieve/pii/S1750583619300246>
- [9] P. K. Kitanidis, Persistent questions of heterogeneity, uncertainty, and scale in subsurface flow and transport, *Water Resources Research* 51 (8) (2015) 5888–5904. doi:10.1002/2015WR017639.
- [10] C. W. Strandli, E. Mehnert, S. M. Benson, CO₂ plume tracking and history matching using multilevel pressure monitoring at the Illinois basin - Decatur project, *Energy Procedia* 63 (2014) 4473–4484. doi:10.1016/j.egypro.2014.11.483.
URL <http://dx.doi.org/10.1016/j.egypro.2014.11.483>
- [11] NAS, *Negative Emissions Technologies and Reliable Sequestration*, National Academies Press, 2018. doi:10.17226/25259.
- [12] Y. Zhu, N. Zabaras, Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification, *Journal of Computational Physics* 366 (2018) 415–447.
- [13] S. Mo, Y. Zhu, N. Zabaras, X. Shi, J. Wu, Deep convolutional encoder-decoder networks for uncertainty quantification of dynamic multiphase flow in heterogeneous media, *Water Resources Research* 55 (1) (2019) 703–728.

- [14] Z. Zhong, A. Y. Sun, H. Jeong, Predicting co2 plume migration in heterogeneous formations using conditional deep convolutional generative adversarial network, *Water Resources Research* 55 (7) (2019) 5830–5851.
- [15] M. Tang, Y. Liu, L. J. Durlofsky, A deep-learning-based surrogate model for data assimilation in dynamic subsurface flow problems, *Journal of Computational Physics* 413 (2020) 109456.
- [16] G. Wen, M. Tang, S. M. Benson, Towards a predictor for co2 plume migration using deep neural networks, *International Journal of Greenhouse Gas Control* 105 (2021) 103223. doi:<https://doi.org/10.1016/j.ijggc.2020.103223>.
URL <http://www.sciencedirect.com/science/article/pii/S1750583620306484>
- [17] G. Wen, C. Hay, S. M. Benson, Ccsnet: a deep learning modeling suite for co2 storage, *Advances in Water Resources* (2021) 104009doi:<https://doi.org/10.1016/j.advwatres.2021.104009>.
URL <https://www.sciencedirect.com/science/article/pii/S0309170821001640>
- [18] M. Raissi, P. Perdikaris, G. E. 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 (2019) 686–707.
- [19] Y. Zhu, N. Zabaras, P. S. Koutsourelakis, P. Perdikaris, Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data, *Journal of Computational Physics* 394 (2019) 56–81. arXiv:1901.06314, doi:10.1016/j.jcp.2019.05.024.
URL <https://doi.org/10.1016/j.jcp.2019.05.024>
- [20] E. Haghighat, R. Juanes, Sciann: A keras/tensorflow wrapper for scientific computations and physics-informed deep learning using artificial neural networks, *Computer Methods in Applied Mechanics and Engineering* 373 (2021) 113552.
- [21] Z. Jiang, P. Tahmasebi, Z. Mao, Deep residual u-net convolution neural networks with autoregressive strategy for fluid flow predictions in large-scale geosystems, *Advances in Water Resources* (2021) 103878.

- [22] M. Tang, Y. Liu, L. J. Durlofsky, Deep-learning-based surrogate flow modeling and geological parameterization for data assimilation in 3d subsurface flow, *Computer Methods in Applied Mechanics and Engineering* 376 (2021) 113636.
- [23] H. Wu, R. Qiao, Physics-constrained deep learning for data assimilation of subsurface transport, *Energy and AI* 3 (2020) 100044.
- [24] O. Fuks, H. Tchelepi, Physics based deep learning for nonlinear two-phase flow in porous media, in: *ECMOR XVII*, Vol. 2020, European Association of Geoscientists & Engineers, 2020, pp. 1–10.
- [25] M. M. Almajid, M. O. Abu-Elseid, Prediction of porous media fluid flow using physics informed neural networks, *Journal of Petroleum Science and Engineering* (2021) 109205.
- [26] L. Lu, P. Jin, G. E. Karniadakis, DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators, *arXiv preprint arXiv:1910.03193* (2019).
- [27] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkumar, Multipole graph neural operator for parametric partial differential equations, *arXiv preprint arXiv:2006.09535* (2020).
- [28] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkumar, Neural operator: Graph kernel network for partial differential equations, *arXiv preprint arXiv:2003.03485* (2020).
- [29] K. Bhattacharya, B. Hosseini, N. B. Kovachki, A. M. Stuart, Model reduction and neural networks for parametric pdes, *arXiv preprint arXiv:2005.03180* (2020).
- [30] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkumar, Fourier neural operator for parametric partial differential equations, *arXiv preprint arXiv:2010.08895* (2020).
- [31] K. Pruess, C. M. Oldenburg, G. Moridis, *Tough2 user’s guide version 2*, Tech. rep., Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States) (1999).

- [32] R. Pini, S. C. Krevor, S. M. Benson, Capillary pressure and heterogeneity for the co₂/water system in sandstone rocks at reservoir conditions, *Advances in Water Resources* 38 (2012) 48–59.
- [33] Schlumberger, Eclipse reservoir simulation software reference manual (2014).
- [34] N. Remy, A. Boucher, J. Wu, *Applied Geostatistics with SGeMS: A User’s Guide*, Cambridge University Press, 2009. doi:10.1017/CB09781139150019.
- [35] H. Pape, C. Clauser, J. Iffland, Variation of permeability with porosity in sandstone diagenesis interpreted with a fractal pore space model, in: *Fractals and dynamic systems in geoscience*, Springer, 2000, pp. 603–619.
- [36] B. Li, H. A. Tchelepi, S. M. Benson, Influence of capillary-pressure models on co₂ solubility trapping, *Advances in water resources* 62 (2013) 488–498.
- [37] S. Carpentier, K. Roy-Chowdhury, Conservation of lateral stochastic structure of a medium in its simulated seismic response, *Journal of Geophysical Research: Solid Earth* 114 (B10) (2009).