AI-Assisted Field Development in Carbon Sequestration with Neural Operators

John Godlewski, SLB*; Philipp Witte, Microsoft

Abstract

Machine learning (ML) techniques are emerging as transformative tools for field development planning by offering rapid and accurate predictions of the subsurface. Hybrid conventional-ML workflows enable engineers to swiftly and systematically screen for optimum solutions in the nearly infinite design space.

To evaluate their effectiveness, this paper presents an end-to-end case study where Fourier neural operators (FNOs) are employed within a CO_2 well placement workflow. The workflow combines data generation with a commercial simulator, network training, and optimization. We assess the performance of FNOs by quantifying their accuracy versus training data quantity. The simulator and ML models are then coupled with a genetic optimizer for combinatorial well placement. We analyze how FNO accuracy impacts the optimization quality and the effectiveness of the FNO-surrogate and simulator-only workflows.

The FNO-enabled workflow was found to provide superior well placement results, faster total workflow time and lower compute costs under all conditions tested.

Introduction

Studies of complex subsurface reservoirs, such those investigated in hydrocarbon, geothermal, hydrogeology, and carbon sequestration, have come to rely on reservoir simulation to predict subsurface flow under wide-ranging conditions (e.g., [1-3]). Due to the large spatial and temporal extents and the nonlinearity of the governing equations, physics simulations require large computing resources and long times to solve [4]. As such, engineers can explore only a limited number of possibilities and a limited representation of uncertainty in the time available. As a result, suboptimal development plans are easily constructed.

Where large numbers of calls to a reservoir model are needed, one common approach is to build a cheap-to-evaluate surrogate model of acceptable accuracy. Such surrogates can be reduced-order models (i.e., derived from simplified physics [5,6]), data-driven models, or both.

Over the past few years, many data-driven models based on machine learning (ML) and deep learning (DL) techniques have been proposed in the context of reservoir engineering. We find that such approaches can generally be grouped into two categories. In the first class [e.g., 9-13] the neural network attempts to directly learn the relationship between a set of control variables (e.g., well location, injection rates) and the target value to be optimized (e.g. well production

*work performed while at SLB

rate, cost, or stored CO_2). The second class of approaches [e.g., 14-19] focuses on approximating the high-dimensional output of the physics themselves, for example the 3D pressure field over time.

While training a high-dimensional surrogate model is more challenging and computationally expensive, it can directly leverage extensive data from physics simulators. We postulate that high-dimensional surrogates learn a morecomplete view of the dynamic system and are better able to generalize to unseen variable combinations. Use of these models also provides spatial predictions of interest, such as CO_2 plume location, trapping mechanisms, and underdeveloped regions.

While many recent publications [12-17] focus on training fast surrogate simulators for carbon capture and sequestration (CCS), they typically only compare the trained DL-based simulators to the numerical simulator. We find there is a lack of analyses of how such a DL-based simulator performs on commercial tasks such as well placement and control. In addition, most comparisons of DL-based simulators with numerical simulators typically include only the performance at inference time while disregarding the greater workflow.

In this paper, we investigate how well an AI-based reservoir simulator performs in the context of developing a geological carbon storage site in which we are interested in optimally placing a set of CO_2 injection wells. In contrast to prior work, we consider the end-to-end workflow, including the data generation with a commercial simulator, training of the neural network, and subsequent well optimization. Our DL-based simulator is based on Fourier neural operators (FNOs) [26], an architecture that has shown better performance over convolutional neural networks (CNNs) for partial differential equations and particularly for CO_2 storage [e.g., 17]. We analyze the cost-performance benefit of the FNO-based workflow versus a conventional, simulator-only workflow.

In addition, we investigate how much data is required to train the FNO-based simulator with respect to how well it performs on the optimization task (rather than just how well it approximates the conventional reservoir simulator). We apply these techniques to a hypothetical design task based on the Sleipner field in the North Sea [18].

Well Placement Problem Statement

We aim to place four vertical injectors in the well-studied Sleipner field [19]. This combinatorial well placement problem was chosen for its high difficultly and large possible solution space. For example, even with the modest grid in this study there are over 6.8 trillion possible combinations. Heterogeneous permeability and variable well locations imply there will be complex interference patterns and plume shapes.

Locations are optimized considering the desire to inject a high quantity of CO_2 at high storage security and low capital cost [20] as combined into a single objective function.

Specifically, the quantity of CO_2 is valued as if it was receiving the US 45Q tax credits [20]. To model the storage security, we consider that the CO_2 could be in trapped in one of four ways. In order of increasing security, these are structural, capillary, solubility, and mineral trapping [22]. Multipliers are used in the objective function for CO_2 trapped in the more-desirable states. We focus on the first three storage mechanisms due to the absence of reliable geochemical data.

A large penalty function is also implemented for any CO_2 that leaks outside of the license block and into the aquifer region (see Figure 1).

Well costs are estimated by the total drilling distance needed for all locations from a platform drilling center, which can be co-optimized or fixed (co-optimized in this study). Drilling wells farther apart results in less reservoir interference but incurs a higher capital cost.

The objective function is calculated at a single time step, although a time-weighted function could also be used. The objective function is found in Equation 1 and Table 1.

$$CP * \left(CO2_{struc} + (\lambda_{cap} * CO2_{cap}) + (\lambda_{dis} * CO2_{dis}) - (\lambda_{penalty} * CO2_{leak})\right) - (D_{\{drill\}} * \ CD)$$

Equation 1: Objective function for optimization

where
$CO2_{struc}$ = Structurally trapped CO_2, metric ton
$CO2_{cap}$ = Capillary trapped CO_2, metric ton
$CO2_{dis}$ = Dissolution trapped CO_2, metric ton
$CO2_{leak}$ = CO_2 leaked from license block to adjacent aquifer, metric ton
$D_{\{drill\}}$ = Drilling distance, m
Other parameters are defined in Table 1.

Table 1: Objective function parameters

Function Parameter	Value
CP, carbon price, (USD/metric ton)	85
λ_{cap}, capillary multiplier	1.5
λ_{dis}, dissolved multiplier	2
$\lambda_{penalty,}$ leak penalty multiplier	50
CD, drilling cost (USD/m)	7600

Reservoir Model

The dataset is a modified version of the Sleipner reservoir model described by Witte et al. [23], which itself is based on the public dataset provided by Equinor [24]. This is a finite-volume physics model that can be solved with a commercial reservoir simulator. The main license block is a 64x56x66 grid. The total grid with aquifer is 84x76x66 with over 400,000 active cells.

To add heterogeneity, the porosity in the injection zone was modeled with a Gaussian random function simulation, with a spherical variogram 1000 m N/S, 500 m E/W, and 40 m vertical anisotropy. The permeability in this sand zone was defined via the porosity-permeability transform function in Equation 2. Shale breaks are assigned zero permeability.

$$k = \phi(\%)^{\{2.27\}}$$

Equation 2: Permeability - porosity transform

Next, a large, infinite-acting numerical aquifer was added with geometrically increasing cell size in all horizontal directions, as demonstrated in Figure 1 and similar to the approach in [17]. The aquifer cells provide both realistic transient pressure support and easy quantification of any CO_2 leakage outside of the main license block.

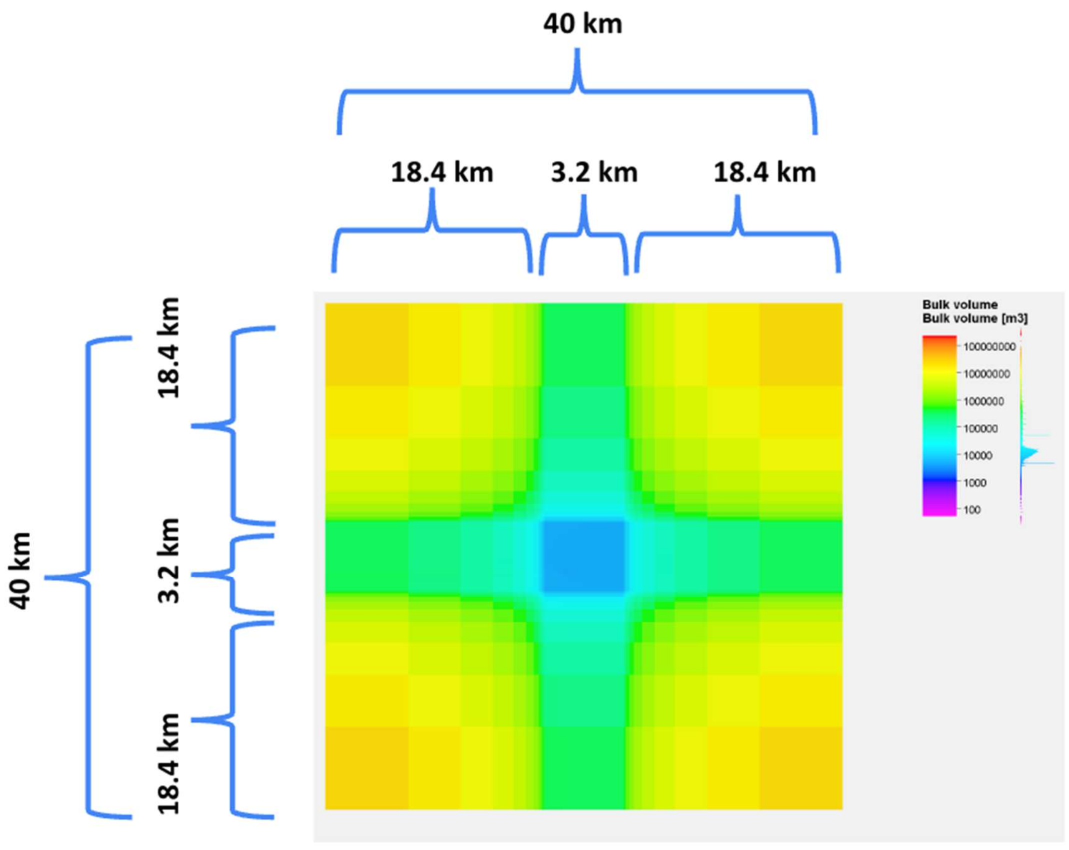

Figure 1: Cell bulk volume for Sleipner main license block (blue) and surrounding numerical aquifer

Finally, the permeability of all zones was reduced by two orders of magnitude as compared to [24], with an injection zone average permeability of 32 mD (see Figure 2). The low permeability and large aquifer were implemented to test if the FNOs could predict the interference of the wells and aquifer under complex transient conditions.

Although there are several zones with low-permeability barriers between them, they are not totally sealing. Equinor interpreted several "feeders"—or leakage points—between the zones as implemented as vertical permeability multipliers [25]. These feeders are left intact and increase the complexity of the well placement task. The yellow dots in Figure 3 show the location of the feeders.

CO_2 injection happens in the lowest zone with a simplified field strategy. All four injectors inject simultaneously at a maximum bottom hole pressure of 230 bar, assumed to be near the fracture pressure. Rate controls, including variable rate controls and time-staggered drilling schedules were tested but are outside the scope of the optimization. Injection occurs for 9 years, and then the well is shut in for an additional year.

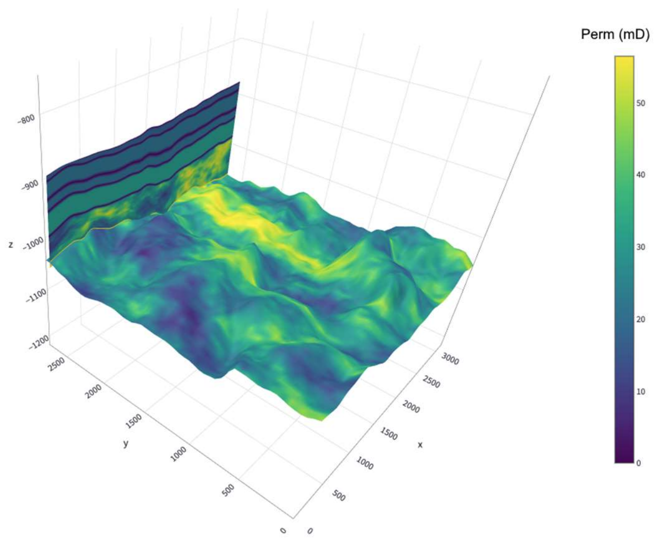

Figure 2: Modified Sleipner horizontal permeability field, main license block, 0-55 mD

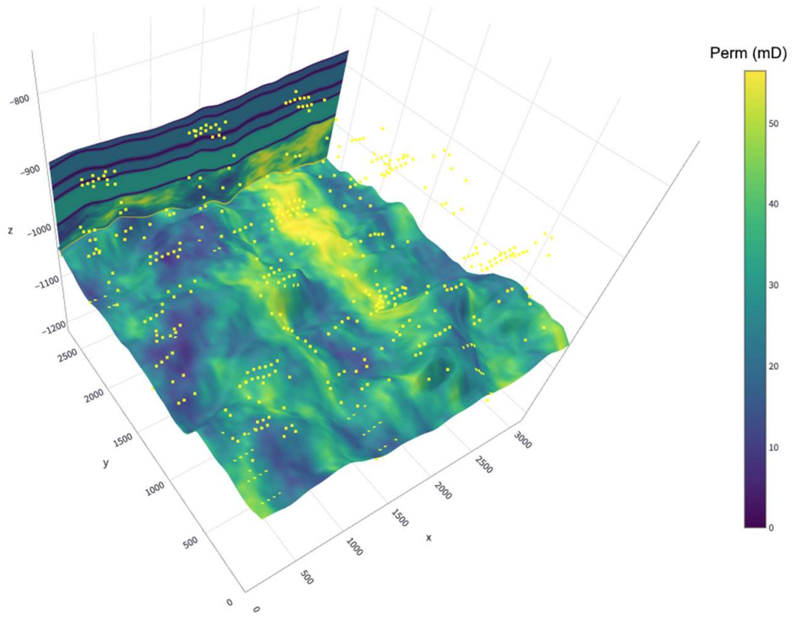

Figure 3: Horizontal permeability field (solid) and feeder locations (dots)

Hybrid Workflow

The hybrid workflow is demonstrated in Figure 4. Data from reservoir simulations are used as "ground truth" to train an AI surrogate model. After data generation and training, the optimization uses the ML model. Solution(s) are then passed back to the physics reservoir simulator for verification and any further detailed engineering.

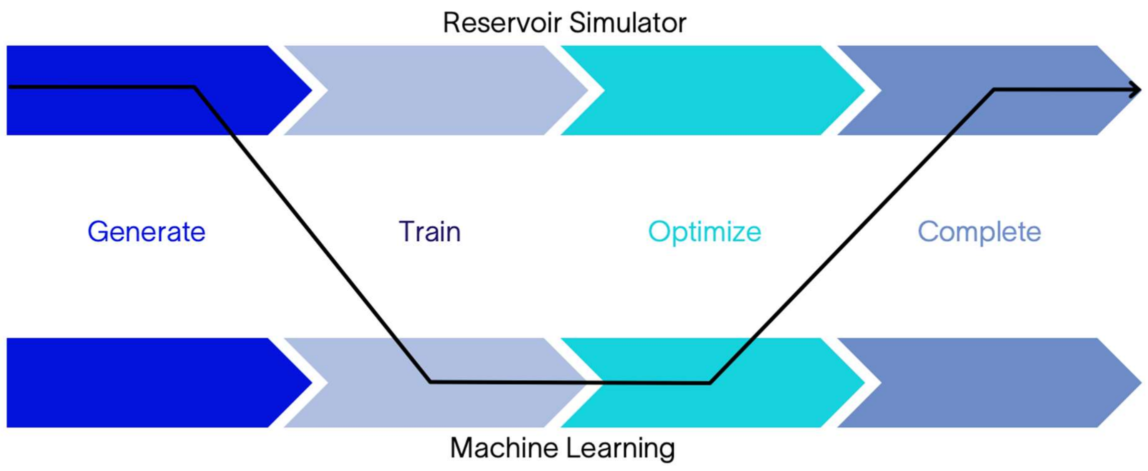

Figure 4: AI surrogate model workflow

Data Generation

Full training and test datasets were created by running reservoir physics simulations with random well locations. The output of the simulations (3D pressure, CO_2 saturation, and dissolved CO_2 versus time) were used to train the FNO. The training data generation process largely follows the workflow described in [23]. No two wells were allowed in the same cell or neighboring cells, but otherwise the training locations were fully random. Lower data requirements may be possible with targeted sampling of the design space.

A total of 4000 ECLIPSE simulations were launched so that the solution could be tested with high and low data availability. For each of the 4000 examples, the input consists of the well locations and the output consists of the simulated pressure, saturation, and dissolved CO_2 concentration at the final simulation time step. The well locations for each example are represented through a binary map (value of one at the location of the wells and zero everywhere else). This way, both the input and output data are three-dimensional tensors of the size of the simulation mesh.

Aside from the well location, the input of each training example contains the permeability in lateral and vertical directions, the porosity, and the depth (vertical location of each grid point) as additional channels. Even when these quantities do not vary between the samples, including them improves convergence during model training.

FNO Network Design Decisions and Speed

Our neural network for learning the CO_2 flow simulator is based on the Fourier neural operator (FNO) architecture [26]. Similar to spectral methods for numerical simulations, FNOs use fast Fourier transforms (FFTs) to transform the hidden states of the network to the spectral domain and learn a weighting of the frequency modes. We chose this architecture as it has shown good performance on approximating solutions of a wide range of partial differential equations, ranging from elliptic to both parabolic and hyperbolic problems [27-28].

Rather than using a 1x1 convolutional kernel in the Fourier bypass layer as in [26], we use a standard convolutional layer with a 3x3 kernel. We find that this modification leads to smaller errors in the simulation outputs, while only minorly increasing the computational cost and number of trainable parameters. As the grid definition is the same for all cases, "super-resolution" is not needed.

As in the original FNO paper [26] and in previous work on FNOs for CO_2 flow simulations [17, 29], we train the FNO in a supervised manner using the data described in the previous section. In addition to the supervised data-to-data learning, the physics conservation equations can also be encoded directly into the loss function [30-31]. While this improves accuracy, it requires finer time discretization, slowing both training and inference. It also is challenging to train these networks as the elements in the loss function must be manually balanced against each other. For these reasons, we omitted conservation equations in our subsequent experiments.

Our goal is to make the FNO simulator as fast as possible since many calls to it may be required to solve the optimization problem. To this end, compared to [29], we reduced the size of the neural network, the number of Fourier modes considered, and predicted only the final state of the reservoir for use in the objective function (excluding intermediate steps).

We independently train one 3D FNO for each of the target variables, namely pressure, saturation, and dissolved CO_2. We find that this improves the quality of the outputs compared to training a single network for predicting all three variables at once. Training time is reduced via transfer learning from one network to the next. Each neural network contains just under eight million trainable parameters and is lightweight to train and deploy. All architecture and training hyperparameters are listed in Table 2.

Training used the 3D results from between 25 and 3200 well combinations run on the ECLIPSE commercial physics simulator. FNO training was completed on between 1 and 4 V100 GPUs for 40 epochs with a batch size of 4. The model took between 5 minutes and 1 hour to train, depending on the quantity of training data used.

The physics simulation takes 8240 seconds on a quad-core CPU laptop (Intel Xeon E3-1545). The FNO surrogate model takes 0.017 seconds for inference on a single V100 GPU, representing a 500,000x speed up at inference.

Table 2: FNO architecture and hyperparameters

Number of FNO layers	4
Number of encoder/decoder layers	2
Number of Fourier modes	10
Padding size	6
Batch size	4
Number of epochs	40
Learning rate	1e-3
Adam betas	(0.9, 0.999)
LR scheduler	ReduceLROnPlateau
LR scheduler: patience	5
LR scheduler: cooldown	0
LR scheduler: reduce factor	0.25
LR scheduler: min LR rate	1e-9

FNO Accuracy vs. Data

The accuracy of the ML model depends strongly on how much data is used to train it. Figure 5 shows this dependence, which appears to exhibit linearity on a semi-log plot of the material balance error versus the number of cases used for training.

In addition to the conventional L2 mismatch, we quantify the mismatch in terms of material balance error. Assuming that the physics simulator is the "ground truth", Figure 5 shows the difference in total CO_2 injected between the ML and simulator. This is done by summing up all Equation 3 for all cells in the license block from both methods.

$$\sum_{all\ cells} V_p * \left(\left(S_{CO2} * \rho_{CO2(sc)} \right) + 〖S_{water} * Rs * Bw * \rho_{CO2(std)}) \right)$$

Equation 3: Summation of CO2 in the 3D grid

Where
V_p= pore volume, porosity*bulk volume, sm^3
S_{CO2} = saturation (pore fraction) of CO_2
$\rho_{CO2(sc)}$ = density of supercritical CO_2, a function of pressure, kg/m^3
S_{water} = saturation (pore fraction) of water, $1 - S_{CO2}$
Rs = dissolved gas ratio, sm^3/sm^3
Bw = water formation volume factor, m^3/sm^3
$\rho_{CO2(std)}$ = density of gaseous CO_2 at standard conditions, kg/m^3

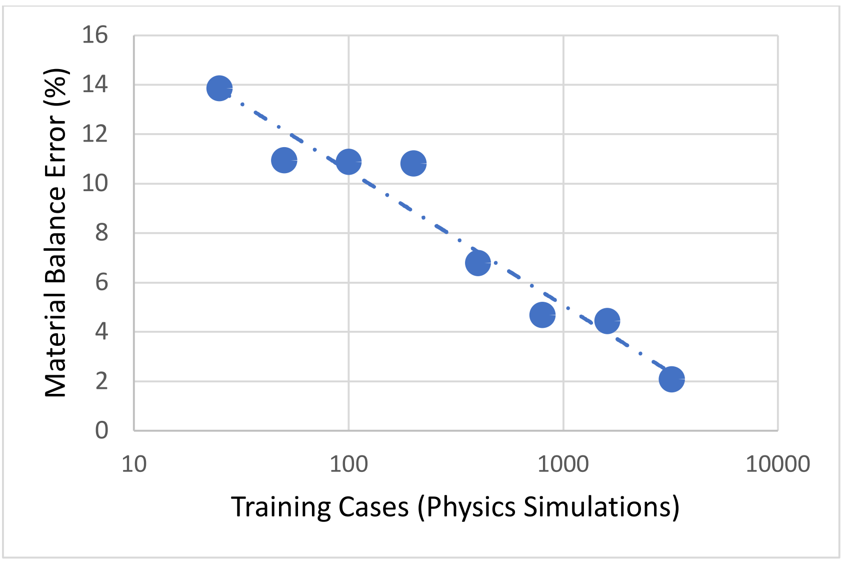

Figure 5: Training data quantity vs. material balance error for test cases

Table 3: Accuracy of the FNO models and number of training cases used

	Validation Loss			Material Balance Error %			
Number of Training Cases	Pressure	Saturation	Dissolved Gas	Free	Trapped	Dissolved	Total in Boundary
25	0.094	0.975	0.963	100.0	129.1	49.0	13.9
50	0.087	0.426	0.436	15.3	11.4	8.4	10.9
100	0.095	0.462	0.404	20.4	16.8	7.7	10.9
200	0.09	0.373	0.377	14.0	16.7	9.5	10.8
400	0.038	0.324	0.342	9.5	6.7	6.0	6.8
800	0.027	0.26	0.275	5.5	5.1	4.8	4.7
1600	0.017	0.203	0.205	4.6	6.0	2.7	4.4
3200	0.013	0.167	0.170	2.6	2.3	1.8	2.1

Visually, a comparison of the predicted values of saturation, dissolved saturation and pressure can be seen in

Figure *6* for a line-drive well pattern (which was absent from the training cases). It shows that the complexity of the CO_2 plume is largely predicted even under the effects of multi-well interference, buoyancy, and a heterogeneous, anisotropic 3D permeability field.

Figure 6 also shows that the errors are mainly concentrated in a few cells around the plume edge. In this area, the PDE's solution is a hyperbolic saturation shock front. Poor estimation of this phenomenon is a well-known limitation of many current ML techniques [27], although the mismatch can be mitigated if it is critical to the application at hand [28].

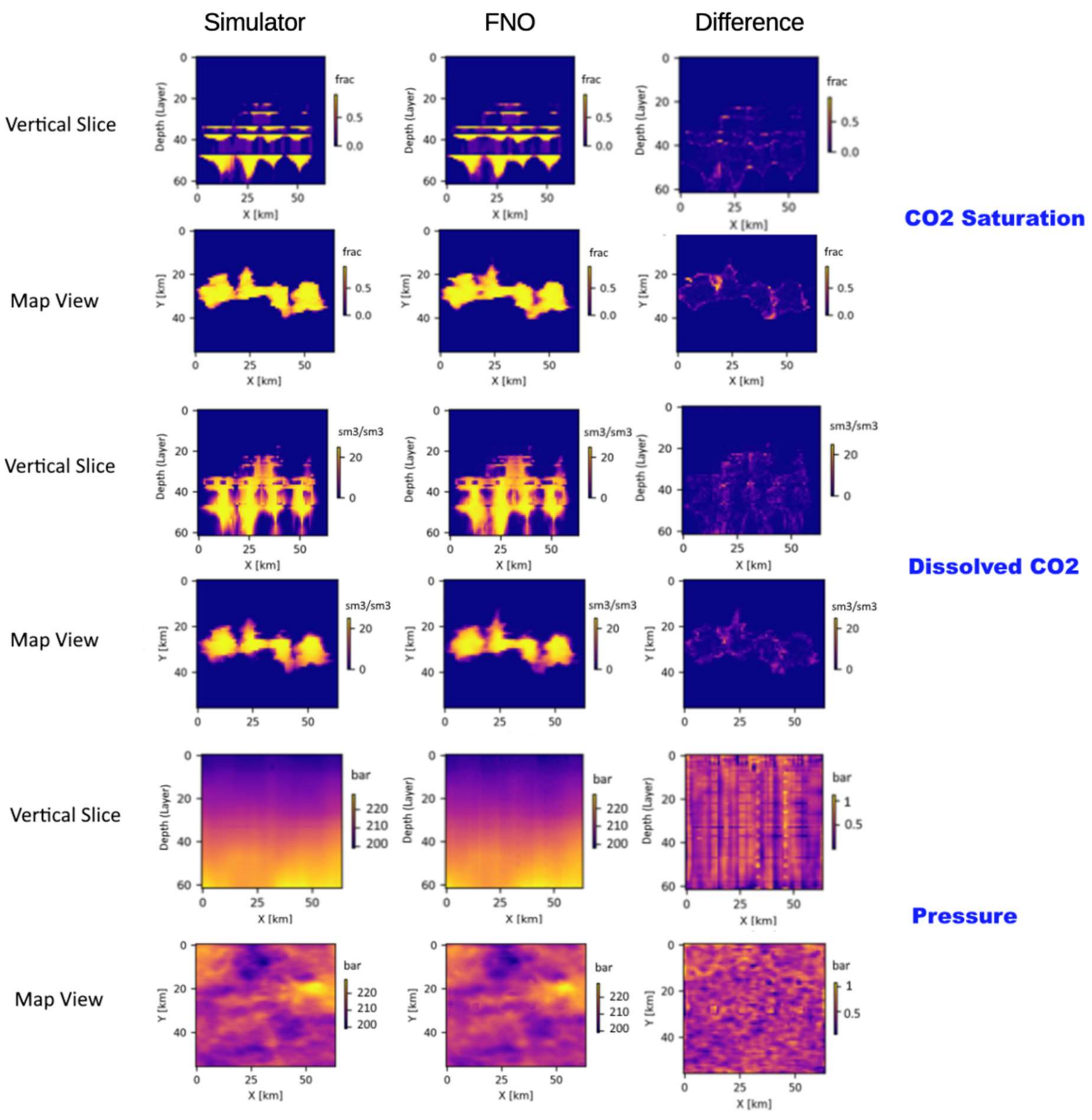

Figure 6: Visual comparison of FNO and commercial simulator for an unseen well pattern.

Optimization Process

The differential evolution optimizer from the SciPy library [32] is used to automatically place the wells, with the *i* and *j* locations of each well being treated as independent variables within the bounds of the grid.

Genetic optimization was chosen as it is a known global minimization method in similar problems [e.g., 33-35], and because it is a derivative-free method that can be used with “black box” simulators and the Fourier neural operators alike.

After the problem has converged, the well locations are run in the commercial simulator to check their final objective function results.

Optimization Quality

The eight ML models and have differing accuracies, as detailed in Figure 5 and Table 3. To test what model accuracy is required to solve the optimization problem, we run the optimizer with all eight ML models. The best well location set from each FNO model is verified with the commercial physics simulator, and the results are shown in Figure 7.

Normally we expect the closer the match is between $Obj_{simulator}$ and Obj_{FNO}, the better the optimized solution. While the more accurate FNO models do result in better well placement, seven of the eight models give results within 5% of each other while one clearly fails.

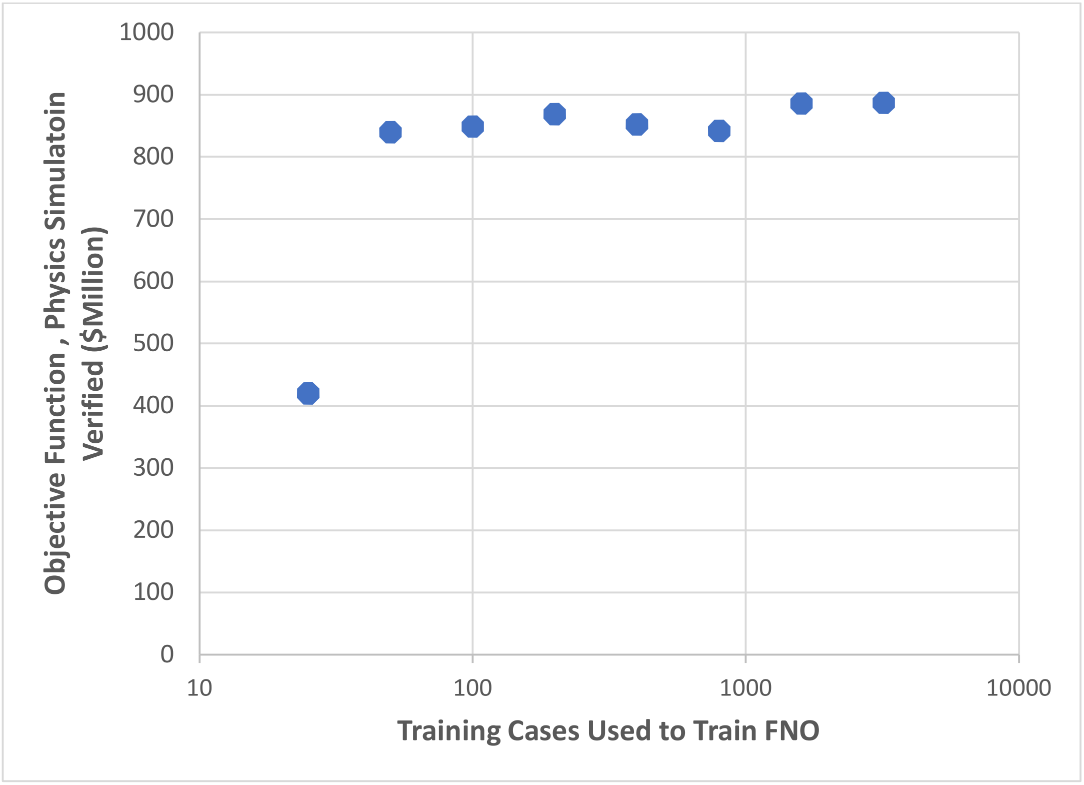

Figure 7: Quantity of training data used versus best solution found

This is because the optimizer result depends on the shape of the objective function with respect to the optimization variables. Even if the error between $Obj_{simulator}$ and Obj_{FNO} is large, if the shape of the response surface is similar, the optimizer will find a similar result.

Figure 8 shows the well locations found by the genetic optimizer for the eight ML models. We also note that the optimizer repeatedly finding the same locations with different initializations gives confidence that a nearly global optimum has been found.

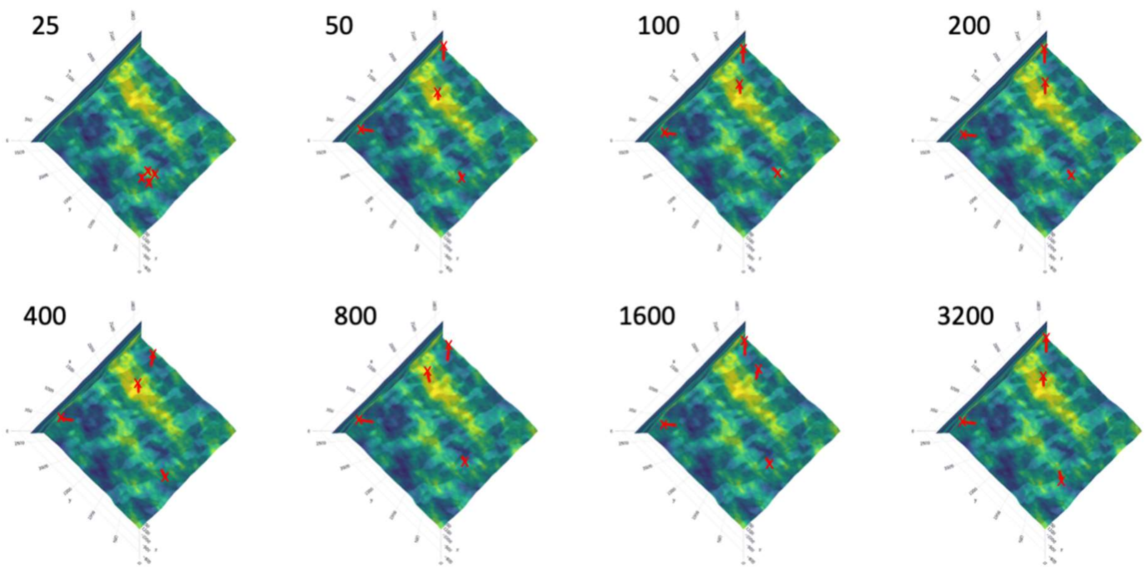

Figure 8: Well locations found by the genetic optimizer and ML models

Another way to consider the optimization quality is if the engineering team only has a certain "budget" to run physics simulations before a decision must be made. Figure 9 shows the best case found for each epoch of the simulation workflow (with 120 simulations per epoch), overlaid on the FNO results.

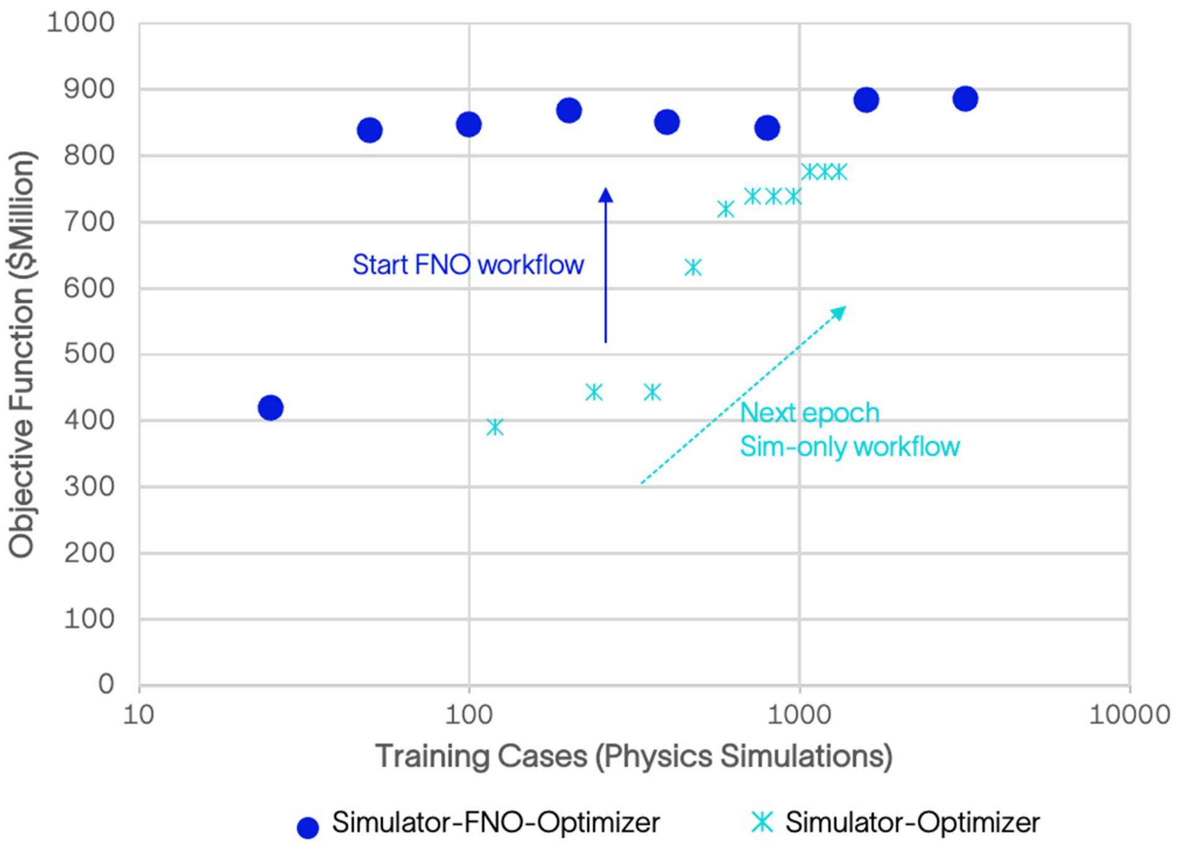

Figure 9: Physics simulations run versus best solution found, both workflows

From Figure 9, we can see that for any "simulation budget" tested, a better optimum would be achieved by taking the existing cases and training an FNO rather than proceeding to the next epoch.

Workflow Comparison

Discussions about the speed of ML vs. physics simulation are misleading as they typically only reference the performance at inference. In fact, the computational load has been shifted from inference to training and data generation. We investigate the efficiency of the FNO-surrogate approach in a holistic manner, where all such elements of a reservoir study are included.

The focus is on the total elapsed time to complete the study, as this is most often the limiting practical factor. Typically, drilling needs to occur by a certain date as a condition of the subsurface license, and the decision quality by that date is of the upmost importance. A secondary, less important, consideration is the total compute cost for the workflow.

A key characteristic of the FNO versus simulator-only workflow is the level of parallelization that can be achieved to reduce the elapsed time. When the FNO workflow is being followed, all training data can potentially be generated simultaneously with flexible cloud computing resources.

When the search space (the possible locations of all wells) is sampled randomly, the runs do not depend on each other and so are “embarrassingly parallel”. Data generation is complete once the slowest simulation has completed, and simulations with convergence problems can often be safely discarded. Figure 10 shows a Gantt chart of the workflow steps with the average parameters of the seven ML models using 50–3200 cases, as reported in Table 4 and Table 5.

In the simulator-only workflow, the genetic optimizer uses the commercial simulator to determine the objective function for every set of variables tried. The optimizer sets off 120 function calls for the population for each optimization epoch, and then adjusts the parameters for the next epoch. The total time for each epoch is thus the slowest simulation time (median time*multiplier) plus the time for needed for the optimizer itself.

The time and compute requirement are outlined for both workflows in Equation 4, Equation 5, Table 4, and Table 5.

$$T_{simopt} = \left(T_{sim} * M_{sim} + T_{opt}\right) * N_{opt\_epochs}$$
$$C_{simopt} = N_{func} * T_{sim} * C_{cpu}$$

Equation 4: Workflow time and cost for optimization with reservoir simulator backend

where
T_{simopt} =Total simulator-optimizer elapsed time
C_{simopt}= Total simulator-optimizer compute cost

$$T_{FNOopt} = T_{sim} * M_{sim} + T_{train} + N_{func} * T_{FNO} + N_{opt\_epochs} * T_{Opt}$$
$$C_{FNOopt} = N_{training} * T_{sim} * C_{cpu} + T_{train} * C_{4GPU} + N_{func} * C_{GPU}$$

Equation 5: Workflow time and cost for optimization with FNO backend

where
T_{FNOopt} = Total FNO-optimizer elapsed time
C_{FNOopt} = Total FNO-optimizer compute cost

Table 4: Time parameters and mean values

T_{sim}, median physics simulation time (min)	137.3
M_{sim}, longest case multiplier	2
N_{opt_epochs}, number of optimization epochs	97
N_{func}, total optimizer function calls	11640
T_{train}, Train time (min)	63.4

T_{Opt}, optimizer calculation time (min per epoch)	0.035
T_{FNO}, FNO inference time (min)	0.000225

Table 5: Cost parameters and mean values

Cloud Costs, January 2023	(USD/min)
C_{4GPU}, 4 V100 GPU train cost	0.2131
C_{GPU}, 1 V100 GPU inference cost	0.0576
C_{cpu}, 4 CPU simulator cost	0.0032

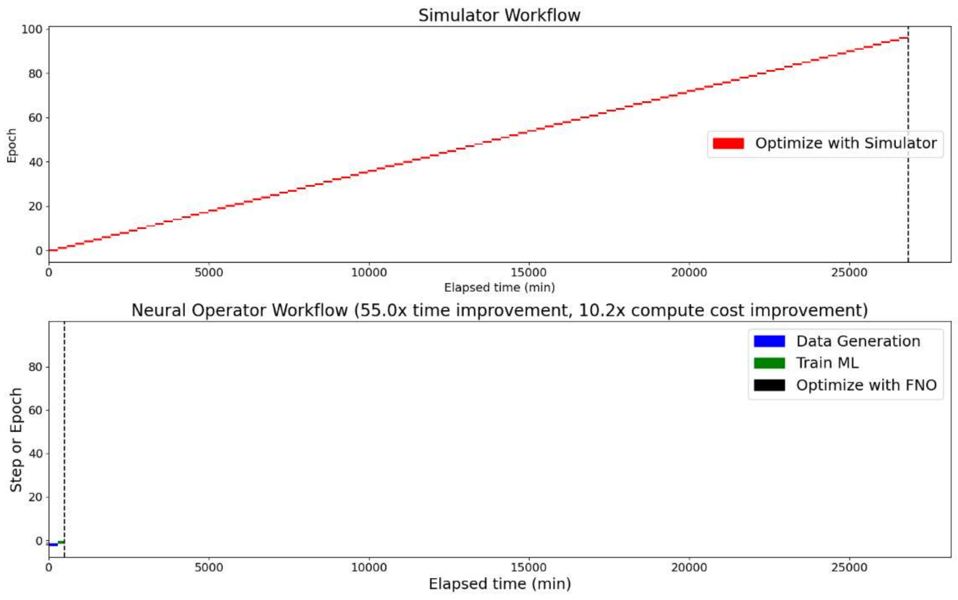

Figure 10: Gantt charts for simulator-only and FNO workflows

Using the equations and tables stated, Figure 10 shows that the ML workflow finishes on average 55x faster and at one-tenth the cloud computing cost of the simulator-only workflow.

Conclusions

Hybrid simulator-ML workflows found more profitable field development strategies under practical time and compute constraints, even when taking data generation and training into account. Near-optimal well locations were found faster and with a lower compute cost than a simulator-only workflow for multiple well locations. Depending on the use case, explicitly encoding the conservation equations into the loss function may further reduce data requirements. For easy optimization problems, the simulator-only workflow may be appropriate. However, complex optimizations and large search spaces requires more simulations/inference calls to find a solution. This dramatically increases the advantage of ML proxy models and enables a new end-to-end workflow for field development decisions.

References

[1] Blunt, Martin, Michael J. King, and Harvey Scher. "Simulation and theory of two-phase flow in porous media." *Physical Review A* 46.12 (1992): 7680.

[2] Jasak, Hrvoje. "OpenFOAM: Open source CFD in research and industry." *International Journal of Naval Architecture and Ocean Engineering* 1.2 (2009): 89-94.

[3] Lichtner, Peter C., et al. *PFLOTRAN user manual: A massively parallel reactive flow and transport model for describing surface and subsurface processes*. No. LA-UR-15-20403. Los Alamos National Lab.(LANL), Los Alamos, NM (United States); Sandia National Lab.(SNL-NM), Albuquerque, NM (United States); Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States); Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States); OFM Research, Redmond, WA (United States), 2015.

[4] Gross, Herve, and Antoine Mazuyer. "GEOSX: A multiphysics, multilevel simulator designed for exascale computing." Paper presented at the SPE Reservoir Simulation Conference, On Demand, October (2021). doi: https://doi.org/10.2118/203932-MS.

[5] Alghamdi, Ahmed, Hiba, Moaz, Aly, Moustafa, and Abeeb Awotunde. "A Critical Review of Capacitance-Resistance Models." Paper presented at the SPE Russian Petroleum Technology Conference, Virtual, October (2021). doi: https://doi.org/10.2118/206555-MS

[6] INSIM - Ying Li, Mustafa Onur, INSIM-BHP: A physics-based data-driven reservoir model for history matching and forecasting with bottomhole pressure and production rate data under waterflooding, Journal of Computational Physics, Volume 473, 2023,111714, ISSN 0021-9991,https://doi.org/10.1016/j.jcp.2022.111714.

[7] Navrátil, Jiří, et al. "Accelerating physics-based simulations using end-to-end neural network proxies: An application in oil reservoir modeling." *Frontiers in Big Data* 2 (2019): 33.

[8] Tang, Haoyu, and Louis J. Durlofsky. "Use of low-fidelity models with machine-learning error correction for well placement optimization." *Computational Geosciences* 26.5 (2022): 1189-1206.

[9] He, Jincong, et al. "Deep reinforcement learning for generalizable field development optimization." *SPE Journal* 27.01 (2022): 226-245.

[10] Wang, Nanzhe, et al. "Efficient well placement optimization based on theory-guided convolutional neural network." *Journal of Petroleum Science and Engineering* 208 (2022): 109545.

[11] Schulze-Riegert, Ralf, et al. "Ensemble-based well location optimization under subsurface uncertainty guided by deep-learning approach to 3D geological feature classification." Paper presented at the Abu Dhabi International Petroleum Exhibition & Conference, November (2020). https://doi.org/10.2118/202660-MS.

[12] Mo, Shaoxing, et al. "Deep convolutional encoder-decoder networks for uncertainty quantification of dynamic multiphase flow in heterogeneous media." *Water Resources Research* 55.1 (2019): 703-728.

[13] Shokouhi, Parisa, et al. "Physics-informed deep learning for prediction of CO2 storage site response." *Journal of Contaminant Hydrology* 241 (2021): 103835.

[14] Tang, Meng, Xin Ju, and Louis J. Durlofsky. "Deep-learning-based coupled flow-geomechanics surrogate model for CO2 sequestration." *International Journal of Greenhouse Gas Control* 118 (2022): 103692.

[15] Tang, Hewei, et al. "A deep learning-accelerated data assimilation and forecasting workflow for commercial-scale geologic carbon storage." *International Journal of Greenhouse Gas Control* 112 (2021): 103488.

[16] Yan, Bicheng, et al. "A robust deep learning workflow to predict multiphase flow behavior during geological CO2 sequestration injection and Post-Injection periods." *Journal of Hydrology* 607 (2022): 127542.

[17] Wen, Gege, et al. "Real-time high-resolution CO 2 geological storage prediction using nested Fourier neural operators." *Energy & Environmental Science* 16.4 (2023): 1732-1741.

[18] Furre, Anne-Kari, et al. "20 years of monitoring CO2-injection at Sleipner." *Energy Procedia* 114 (2017): 3916-3926.

[19] Andrew, J. Cavanagh, R. Stuart Haszeldine, and Bamshad Nazarian. "The Sleipner CO2 storage site: using a basin model to understand reservoir simulations of plume dynamics." *First Break* 33.6 (2015).

[20] Beck, Lee, et al. “The US Section 45Q Tax Credit for Carbon Oxide Sequestration: An Update”. Global CCS Institute (2020).

[21] Cameron, David A., and Louis J. Durlofsky. "Optimization of well placement, CO2 injection rates, and brine cycling for geological carbon sequestration." *International Journal of Greenhouse Gas Control* 10 (2012): 100-112.

[22] Ringrose, Philip. "How to store CO2 underground: Insights from early-mover CCS projects." SpringerBriefs in Earth Sciences (2020): 978-3.

[23] Witte, Philipp A., et al. "Fast CO2 saturation simulations on large-scale geomodels with artificial intelligence-based Wavelet Neural Operators." *International Journal of Greenhouse Gas Control* 126 (2023): 103880.

[24] “Sleipner CO2 Benchmark Model”, https://co2datashare.org/dataset/sleipner-2019-benchmark-model (2019).

[25] Andrew, J. Cavanagh, R. Stuart Haszeldine, and Bamshad Nazarian. "The Sleipner CO2 storage site: using a basin model to understand reservoir simulations of plume dynamics." *First Break* 33.6 (2015).

[26] Li, Zongyi, et al. "Fourier neural operator for parametric partial differential equations." *arXiv preprint arXiv:2010.08895* (2020).

[27] de Hoop, Maarten V., et al. "The cost-accuracy trade-off in operator learning with neural networks." *arXiv preprint arXiv:2203.13181* (2022).

[28] Wen, Gege, et al. "U-FNO—An enhanced Fourier neural operator-based deep-learning model for multiphase flow." *Advances in Water Resources* 163 (2022): 104180.

[29] Witte, Philipp A., et al. "Industry-scale CO2 Flow Simulations with Model-Parallel Fourier Neural Operators." *NeurIPS 2022 Workshop Tackling Climate Change with Machine Learning*. 2022.

[30] Raissi, Maziar, Paris Perdikaris, and George 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.

[31] Li, Zongyi, et al. "Physics-informed neural operator for learning partial differential equations." *arXiv preprint arXiv:2111.03794* (2021).

[32] https://scipy.org/

[33] Onwunalu, Jérôme E., and Louis J. Durlofsky. "Application of a particle swarm optimization algorithm for determining optimum well location and type." *Computational Geosciences* 14 (2010): 183-198.

[34] Wang, Honggang, et al. "Optimal well placement under uncertainty using a retrospective optimization framework." *SPE Journal* 17.01 (2012): 112-121.

[35] Bukhamsin, Ahmed Y., Mohammad Moravvej Farshi, and Khalid Aziz. "Optimization of multilateral well design and location in a real field using a continuous genetic algorithm." Paper presented at the SPE/DGS Saudi Arabia Section Technical Symposium and Exhibition, Al-Khobar, Saudi Arabia (2010) doi: https://doi.org/10.2118/136944-MS.