# Machine Learning for Robust Structural Uncertainty Quantification in Fractured Reservoirs

Dashti, Ali[1*]; Stadelmann, Thilo[2]; Kohl, Thomas[1]

1. Institute of Applied Geosciences, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
2. Centre for AI, Technikumstrasse 71, Zurich University of Applied Sciences, 8400 Winterthur, Switzerland

* Corresponding author: Ali.dashti@kit.edu

## Abstract

Including uncertainty is essential for accurate decision-making in underground applications. We propose a novel approach to consider structural uncertainty in two enhanced geothermal systems (EGSs) using machine learning (ML) models. The results of numerical simulations show that a small change in the structural model can cause a significant variation in the tracer breakthrough curves (BTCs). To develop a more robust method for including structural uncertainty, we train three different ML models: decision tree regression (DTR), random forest regression (RFR), and gradient boosting regression (GBR). DTR and RFR predict the entire BTC at once, but they are susceptible to overfitting and underfitting. In contrast, GBR predicts each time step of the BTC as a separate target variable, considering the possible correlation between consecutive time steps. This approach is implemented using a chain of regression models. The chain model achieves an acceptable increase in RMSE from train to test data, confirming its ability to capture both the general trend and small-scale heterogeneities of the BTCs. Additionally, using the ML model instead of the numerical solver reduces the computational time by six orders of magnitude. This time efficiency allows us to calculate BTCs for 2'000 different reservoir models, enabling a more comprehensive structural uncertainty quantification for EGS cases. The chain model is particularly promising, as it is robust to overfitting and underfitting and can generate BTCs for a large number of structural models efficiently.

Keywords: machine learning, uncertainty quantification, structural uncertainty, EGS

# 1 Introduction

Numerical simulations of physical systems described by differential equations are essential in engineering. Advancements in hardware have enabled computing units to solve coupled nonlinear differential equations, encompassing a wide range of phenomena, from weather forecasting (Bauer et al., 2015) to blood circulation in living bodies (Doost et al., 2016). However, these methods are computationally intensive and highly sensitive to specific cases. Besides the huge energy consumption of these computational infrastructures (Benoit et al., 2018), their availability is also limited. Furthermore, parameter tuning, sensitivity analysis (Borgonovo and Plischke, 2016), and uncertainty quantification (Abbaszadeh Shahri et al., 2022; Soize, 2017) demand up to millions of simulations.

Machine learning (ML) methods have gained significant traction across various fields (Brunton and Kutz, 2022; Stadelmann et al., 2019), including geothermal applications (Okoroafor et al., 2022). In this context, data-driven and physics-informed ML (physics-informed neural network, PINN) techniques are of great interest (Carleo et al., 2019; Raissi et al., 2019). PINNs and their diverse descendants are ceaselessly flourishing to replace numerical solvers (Karniadakis et al., 2021; Kharazmi et al., 2019; Knapp et al., 2021; Yu et al., 2022); however, their accuracy and time-efficiency for solving complex problems is still a subject of development (Degen et al., 2023).

One of the challenges in geothermal applications is characterizing fluid flow through complex underground networks. While the geometry of a fracture can define the general direction of flow, the local variation of petrophysical properties impacts the specific pathways (Meakin and Tartakovsky, 2009). The enhanced geothermal system (EGS), as an engineered underground reservoir, strongly relies on high flow rate circulation through the impermeable matrix. To enhance the reservoir's permeability, the cold fracturing fluid is injected to create new fractures or reopen the pre-existing ones (e.g. Kohl and Mégel, 2007). Hence, a complex underground fracture/flow pattern can be observed in any EGS example like the model presented by Egert et al. (2020).

Integrating local data coming from wells with field measurements like tracer tests (Cao et al., 2020) can provide insights into the EGS situation. Tracer test campaigns usually yield breakthrough curves (BTCs), which are widely used to extract properties of the porous media and fracture network. However, each measuring method is error-prone resulting in inherent uncertainty (Bond, 2015; Wellmann et al., 2010). Therefore, incorporating structural uncertainties in numerical simulations in EGS settings makes the flow forecast more realistic (Zhou et al., 2022).

This study proposes to replace computationally demanding simulations with speedy ML models to quantify structural uncertainty estimations derived from tracer data in two different EGS settings. By state-of-the-art ML methods like decision tree regression (DTR), random forest regression (RFR), and gradient boosting regression (GBR), multifold BTCs are generated on top of pure time-consuming numerical simulations. We train reliable ML models to map geometric data from the uncertain fractures of the EGS reservoir to the simulated BTC. The position of the variating structural elements is used as the input feature, and the entire BTC is chosen as the target variable. The proposed ML model correlates the entire BTC with input features, rather than using a time window to predict the future.

# 2 Methodology

## 2.1 Tracer models

Tracer flow in two different cases are applied in this study. The conceptual model introduced by Dashti et al. (2023) is used here as the first case. The model for the first case is called the ‘simple case’ because it is a highly simplified version of an EGS with a doublet configuration. The conceptual model contains two main transmissive/open faults that are connected to an injection and production well. There is also an additional sub-horizontal fault/fracture structure making a connection between the major faults at greater depth. However, data related to this structure are subject to uncertainty since this fault is located far from the drilling trajectory, and its existence as a conduit is confirmed only by additional geophysical surveys or hydraulic testing. Figure *1* provides a schematic view of the model, where two sub-vertical faults intersect with the injection and production wells and are labeled as Fault_Inj and Fault_Pro, respectively. The sub-horizontal fault, referred to as Fault_Con in the figure, is represented by thin green lines, as it connects the two major faults. Dashti et al. (2023) introduced a range of structural scenarios and perturbed the location of the sub-horizontal fault 50 times to investigate the impact of structural uncertainty on flow.

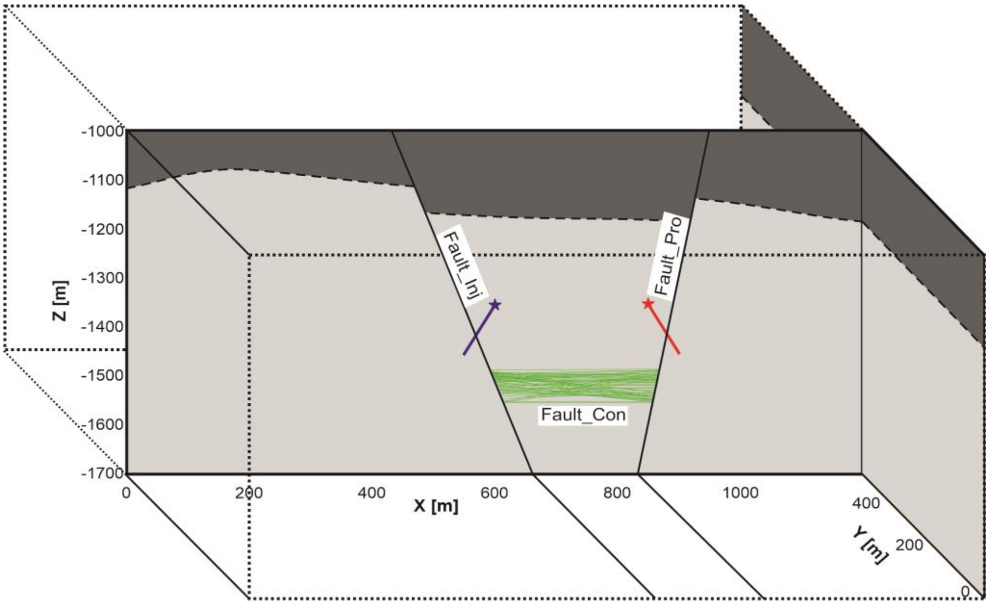


Figure 1. A schematic view of the simple case. The two certain sub-vertical faults (Fault_Inj and Fault_Pro) are shown as continuous black lines and the thinner green lines show traces of the uncertain sub-vertical fault (Fault_Con). Each green trace makes a unique structural scenario.

To comprehensively evaluate the performance of ML methods, a second, more intricate fracture network model (named as complex case) was developed (Figure 2). The ‘complex case’ incorporates seven fractures, with two designated as uncertain. The impact of varying these two fractures' depth and dip angle on tracer flow was assessed through 100 scenarios. All scenarios shared identical material properties, while the uncertain fractures' dip and depth were varied. The modelling assumptions of the complex case are similar to the simple case which is already addressed in Dashti et al. (2023).

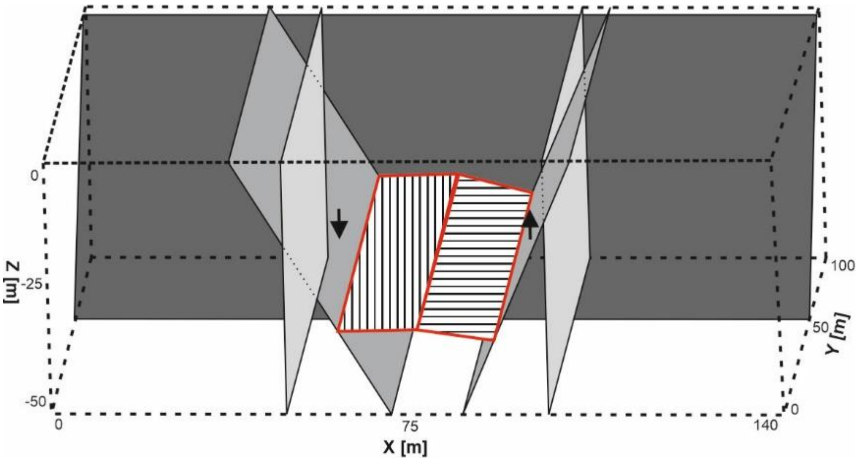


Figure 2. A complex EGS setting with seven fractures. Certain (five) fractures are shown as grey surfaces with varying shades and solid black borders while the two uncertain fractures are highlighted via the thick red border and hashed infill. Two arrows show the location of the injection and production wells.

## 2.2 Machine learning model

The ML model in this study predicts the tracer concentrations over time, i.e. the BTCs for two cases. Time series estimation for different applications is a well-documented topic (Gudmundsdottir and Horne, 2020; Weigend and Gershenfeld, 1994). For example, Alakeely and Horne (2020) introduced a recurrent neural network to predict the future by incorporating historical data. Such methods predict the system's long-term performance based on a moderate duration of the monitoring data. However, our study predicts the entire time series making the ML models applicable for cases without any historical data.

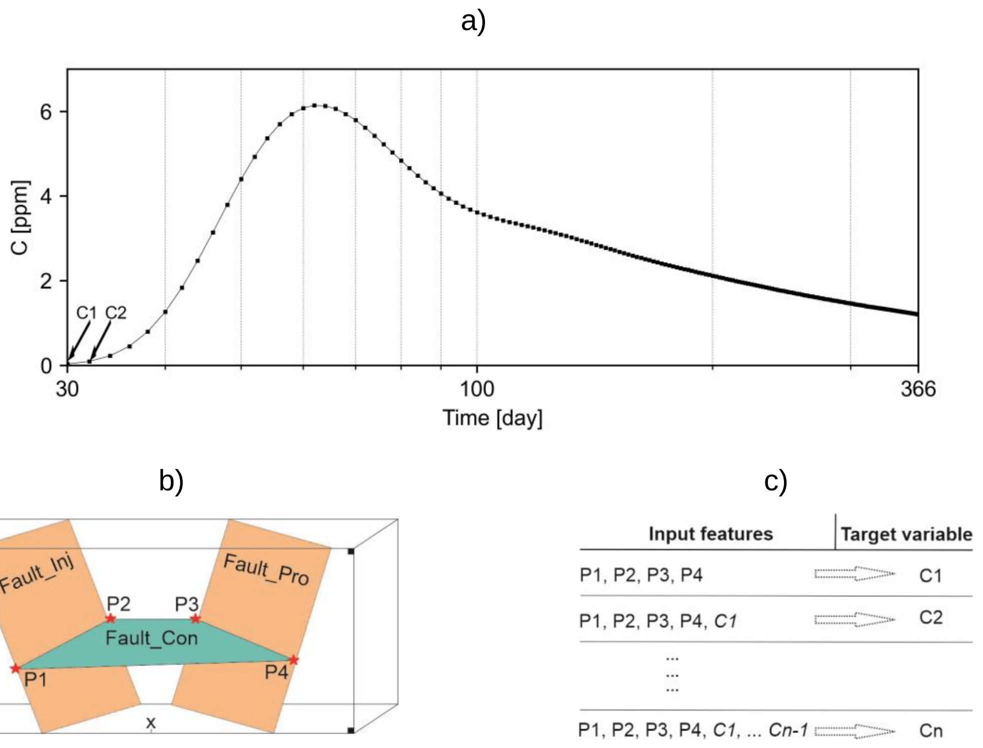
Due to the nature of the problem, two different strategies are developed.

- Strategy 1: Two ML models, DTR and RFR, are trained to independently predict the tracer concentration values. Both models predict the entire time steps of the BTC, using the input features. In this study DTR and RFR correlate structural information of the geological model with the tracer concentration. While in DTR a single tree is trained to capture the relation between the input features and target variable, RFR cultivates several trees in parallel (bagging). DTR is simple to implement and interpret, but it can be prone to overfitting. Therefore, the more complex RFR is also included in this study. The mathematical foundations of DTR and RFR are well-documented in the literature e.g. Kotsiantis (2013), Liu et al. (2012) and XU et al. (2005).
- Strategy 2: A GBR model is used to predict the concentration value at each time step by correlating it with the previous prediction. The GBR is an ensemble method that combines multiple simple and weak learners sequentially (bagging) to improve the overall performance of the model. This approach, denoted as the chain model, requires GBR to be executed for each time step of the BTC. Details of this approach are elaborated in the following.

### 2.2.1 Chain GBR model

Figure *3* provides an overview of the chain regression model for the simple case. A BTC, serving as the target variable, is presented in Figure 3-a. The input features are composed of the structural geometric information from the reservoir model with the coordinates of four corners of the uncertain sub-horizontal fracture (P1, P2, P3, and P4 in Figure 3-b). The model correlates the x/z coordinates with the BTC concentration values, i.e. the y-coordinate data remain fixed across all scenarios for the sake of simplicity. All the governing equations and modelling assumptions behind the calculation of the BTCs are fully addressed in Dashti et al. (2023). For the complex case, coordinates of the two uncertain fracture surfaces are used as the input feature while the BTC data are target variables.

The chain model predicts the BTC concentration values in a sequential manner. It starts by predicting the concentration for the first time step (C1) based on the input features (Figure *3*-b and c). For the second and following time steps (C2), the model uses the previous values, i.e. C1, along with the input features. Some errors can exist in the predicted C1 by GBR. However, to predict C2, the input feature list still contains 8 coordinate values than have a higher impact compared to the recently predicted C1. This gradual addition of the predicted values can help the chain model to adjust the weight of added features, i.e. previously predicted concentrations. Figure 3-c illustrates how concentration values from previous steps concatenate in the input features' list. To predict the first concentration value (C1) in the GBR chain model, the input feature list initially contains eight values. To predict the concentration for the last time step of the simple case (C169), the input feature list contains eight coordinates and 168 previously predicted concentration values. In the complex case, the BTC includes 140 concentration values. The input feature list of the DTR and RFR models remains fixed, because these two methods predict all the time steps of the BTC merely based on the coordinates of the fractures.



| Input features | Target variable |
|---|---|
| P1, P2, P3, P4 ⇨ | C1 |
| P1, P2, P3, P4, *C1* ⇨ | C2 |
| ...<br>...<br>... | |
| P1, P2, P3, P4, *C1, ... Cn-1* ⇨ | Cn |

Figure 3. Workflow developed for chain GBR model a) A BTC representing concentration values, C, versus logarithmic time scale. b) Four corners of the sub-horizontal uncertain fault, P1, P2, P3, and P4, are used in the ML model to predict the first concentration value (C1) for the simple case. c) To predict the second concentration value (C2), the first predicted value (C1) is also included besides the coordinates of four corners. In each time step, the previous values are added up to the list of input features.

The GBR algorithm (Friedman, 2002) is selected due to its simplicity, bagging nature, and efficiency as a predictor for the chain model. Like other supervised ML algorithms (Gupta, 2022), GBR learns a function that maps the input feature/s ($x$) to target variable/s $f(x)$ with the minimum loss:

$$F(x) = argminL\left(f(x), \hat{f}(x)\right) \text{Eq. 1}$$

where $L$ is the loss function and $\hat{f}(x)$ represents the prediction. The loss function is chosen based on the type of learning (e.g., regression, classification) and the type of the target variable (e.g., discrete, continuous). Squared error ($L_2$) loss (Bühlmann and Yu, 2003) is a simple and efficient loss function when outliers are not expected and is hence chosen here:

$$L = \sum_{i=1}^{n} \frac{1}{2}\left(f(x) - \hat{f}(x)\right)^2 \qquad \text{Eq. 2}$$

ML methods generating an ensemble of predicting models in parallel (bagging methods like RFR) or sequential (boosting methods like GBR) are more reliable than models consisting of a single strong predictive model (like DTR) (Fanelli et al., 2013; Shu and Burn, 2004). Boosting methods like GBR can have a better performance for working on small data sets compared to bagging methods that distribute the data set between different predictors. GBR starts with a very simple model ($F_0(x)$), trying to fit a straight horizontal line (average of target variable). In fact, the derivation of the loss function with respect to the predictions establishes the average value as the best guess for the first tree. In the next round, the GBR algorithm maps the input features to the residuals (remaining errors) of the previous tree, a process that can be interpreted as performing gradient descent on the negative derivative of the difference between prediction and target variable w.r.t. the prediction (Breiman, 1998). The use of residuals rather than absolute values is another reason for choosing GBR. This allows for the inclusion of residuals contributed by recently error-prone predicted concentration values into the model. In subsequent rounds, new decision trees are trained based on the accumulated residuals of the whole ensemble (Schapire, 2003):

$$\hat{F}_m(x) = F_{m-1}(x) + \alpha_m \hat{f}_m(x) \quad \text{Eq. 3}$$

where $\hat{F}_m(x)$ represents the final general function that connects input features to the target variable, $F_{m-1}(x)$ contains the information from all previous tress, $\alpha$ is the learning rate that avoids overfitting and $\hat{f}_m(x)$ represents the last tree that is correlating remaining residuals and the input features. Low learning rates decrease the impact of each tree, i.e., more trees will be needed but the model also will be more generalized. GBR minimizes the error of each tree and uses the remaining errors as the target variable of the next tree. In this way, the model is trained based on its minimized errors and aggregates several trees with decreasing errors. He et al. (2022) delved into the details of the GBR.

### 2.2.2 ML model optimization and quality control

Each ML model has two types of arguments: 1) inputs that include hyperparameters (parameters related to the model's architecture) and features selected by the user for predicting the target variable/s, and 2) output arguments that consist of internal weights and the target variable/s. The ML model is trained to minimize the error by tuning its input arguments, allowing the learning algorithm to optimize the output arguments and achieve better scores on the withheld test set (Alpaydin, 2020; Hutter et al., 2019). This iterative process, known as hyperparameter tuning (Raschka and Mirjalili, 2019) involves optimization of parameters such as the learning rate, number of trees, maximum depth of trees, etc. to decrease the error. Determining the optimal number of trees poses a challenge due to the bias-variance trade-off (Oshiro et al., 2012; Probst et al., 2019). Another hyperparameter, the maximum depth of a tree, is defined as the longest path between the root node (first node) and the leaf node (last node).

Grid search is a hyperparameter tuning method that allows input arguments to be defined as a range rather than a single value. It performs an exhaustive search over all possible combinations of values to identify the model with the lowest error i.e. highest score. For the RFR model, the number of trees and maximum depth is considered as arrays with 20 and 10 elements, respectively that result in 200 combinations. For the DTR model also maximum depth of each tree, the minimum number of samples in a leaf node and the minimum number of samples for splitting an internal node are tuned. In total, an ensemble of 540 models has been calculated using hyperparameter tuning for the DTR method. In the GBR algorithm of the chain model default values are used. Conventionally, higher score values are preferred over lower ones, and therefore we also tried to find out the combination with the maximum negated mean squared error (MSE) using the grid search.

To evaluate the model's performance, k-fold cross-validation (Zhang et al., 1999) has been employed. Rather than splitting the input data into train and test, it randomly splits them arbitrarily into k number of ”splits”. Then, the ML model will keep one split as the test and all others as the train sets. In the case of splitting data into five splits, the same number of models will be run and in each run, splits will be shuffled. This five-run procedure will be performed for all the assumed 200 combinations of hyperparameters in the grid search for the RFR method. Therefore, it finally creates 1'000 ML models – each of them being an ensemble of individual trees – and the ensemble with the highest score will be used for the final prediction. In this study, we follow the recommendations in the literature (An et al., 2007; Erdogan Erten et al., 2022) and use five splits for cross-validation for all three methods. Training (online) time for the 1,000 ML models of the RFR model on a Core i7 laptop is approximately 10 seconds. For DTR, with an ensemble of 2,700 ML models, the online time remains to be around 10 seconds. The simplicity of DTR compared to RFR results in faster computation. The chain model proves to be the most time-consuming approach, taking around 70 seconds for training without any hyperparameter optimization. Several hyperparameters were tested for the chain model, but the online time only increased without improving the model's accuracy. Therefore, default values were chosen for the chain model. For both the simple and complex cases several values have been tried for the learning rate in hyperparameter tuning but in the end, the default one (0.1) has been used. The required time for predicting a new solution with the trained models (offline time) remains in the range of milliseconds. To access the input data and trained ML models of two cases, please refer to the code and data availability section.

Figure 4-a and b show the distribution of the negative MSE scoring metric in train and test splits, focusing on two tuned hyperparameters of the RFR model in the simple case. The average of the MSE in the four train splits is presented in Figure 4-a. The distribution of the average scoring metric in the train splits is influenced by both the number and maximum depth of trees. Based on Figure *4*-a, the accuracy of the model increases as both the maximum depth of trees and the number of trees increase. However, the score distribution in the test split (Figure 4-b) is more complicated. The scores in the test split are generally lower than those in the train splits (-0.04 to -0.004 versus -0.018 to -0.003). While the score distribution for the train split promises high accuracy of the model by increasing the two hyperparameters, the heterogeneous distribution in Figure 4-b raises doubts on this conclusion. The presented example in Figure 4 concludes that determining the optimal combination even for only two hyperparameters is not a straightforward task. Going to higher dimensions can make the situation more complicated and unsolvable. Therefore, methods like grid search identify the best combination of tuned hyperparameters.

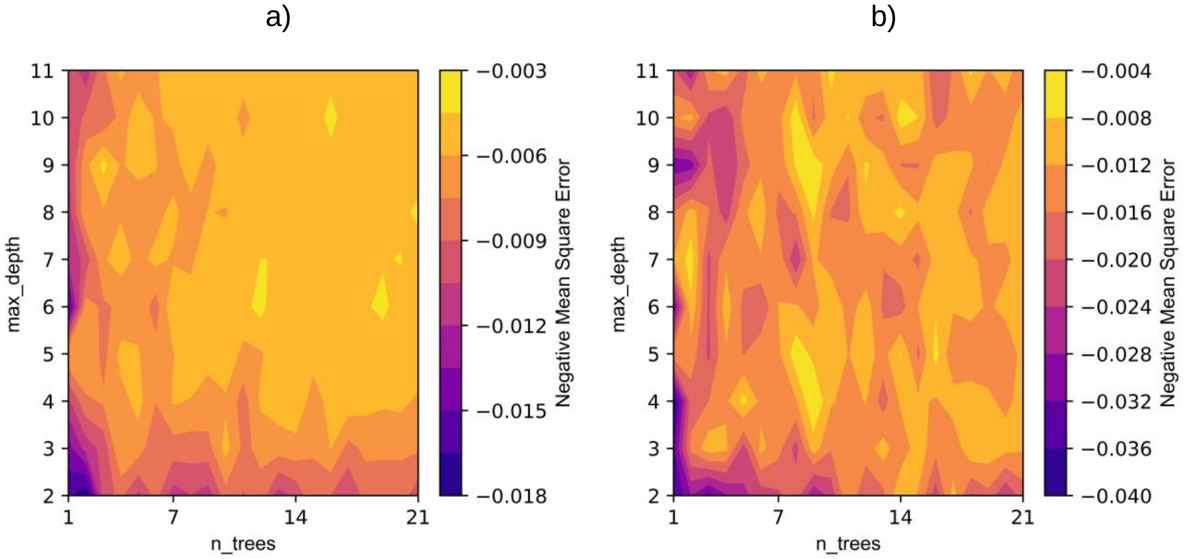


Figure 4. Change in the accuracy of the ML model with respect to different combinations of two hyperparameters of the RFR model on the train (a) and test (b) splits. The accuracy distribution in the train split (a) is smooth and higher accuracies can be achieved by increasing the number of threes and maximum depth of each tree. Subplot b depicts the more patchy and anisotropic behavior of the accuracy with respect to the hyperparameters.

# 3 Results

## 3.1 Simple case

Dashti et al. (2023) employed numerical simulations to assess the effects of uncertainty in structural models using 50 different structural scenarios in a simplified EGS setting. In these synthetic models, a 24-hour tracer injection on day eight of the simulation was assumed and monitored along a one-year time span in the production well (see Figure 5-a with e.g. peak concentration time varying between days 54 and 68). To better present the variations, a box plot (Figure 5-b) is generated by extracting the highest concentration value from each BTC and normalizing them based on their median. The variation of the tracer peak concentration time, as well as a 25% fluctuation in peak magnitude, emphasize the significance of structural uncertainty, which can introduce unexpected deviations in the results of important field tests. The appearance of a second peak between days 100 and 150 in Figure 5-a is due to the reinjection of the tracer, not multiple flow paths or stagnation zones. The first 30 days of the simulation are disregarded due to negligible concentration (almost zero) of the tracer in the production well during that period.

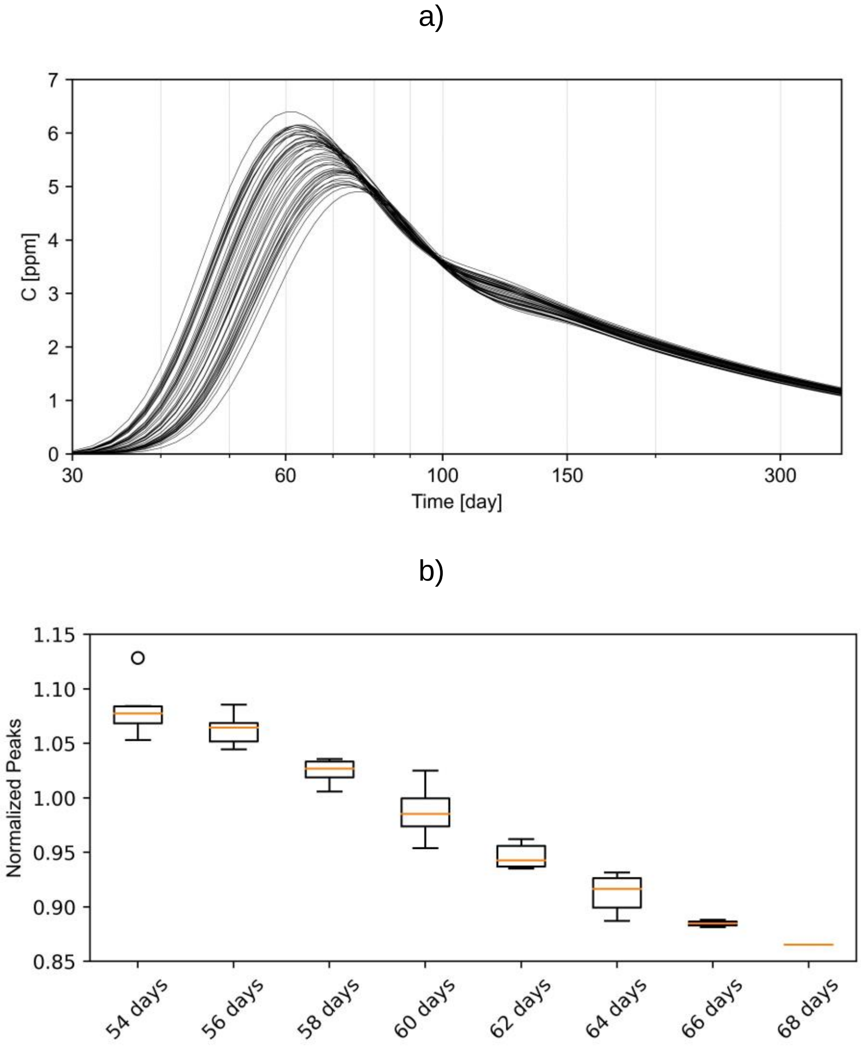


Figure 5. a) Unique BTCs simulated using the finite element solver and used as target variables for the ML models. BTCs are different from each other due to changing structural models. b) A box plot visualizing the normalized peak concentration values versus the time of the calculated peak (analysis based on Dashti et al. (2023)).

Results of the k-fold cross-validation in Figure *6* show how RMSE varies in five splits of the three ML methods. The average RMSE of the chain model is lower than the DTR and RFR. Apart from the higher absolute accuracy, the homogeneity of the model's performance is another important factor to consider. Based on Figure *6*, RMSE values in the DTR model show higher standard deviations. The higher standard deviation of RMSE for the DTR model suggests that it is overfitting the training data. Overfitting occurs when a model learns the training data too well and is unable to generalize to new data. In the case of the DTR model, this may be due to the fact that it is a single-tree model. Hence it is more likely to memorize the training data than an ensemble model like the RFR or chain model. In this study, the simplicity of the DTR model is the main factor leading to overfitting issues. The RFR model mitigates overfitting by initiating multiple parallel trees that distribute the input data. The chain model also incorporates several sequential models that consistently outperform a single model. Overall, the chain model is the most accurate and robust ML model for predicting BTCs in cases without any historical data. It has a lower average RMSE and a lower standard deviation of RMSE than the RFR and DTR models.

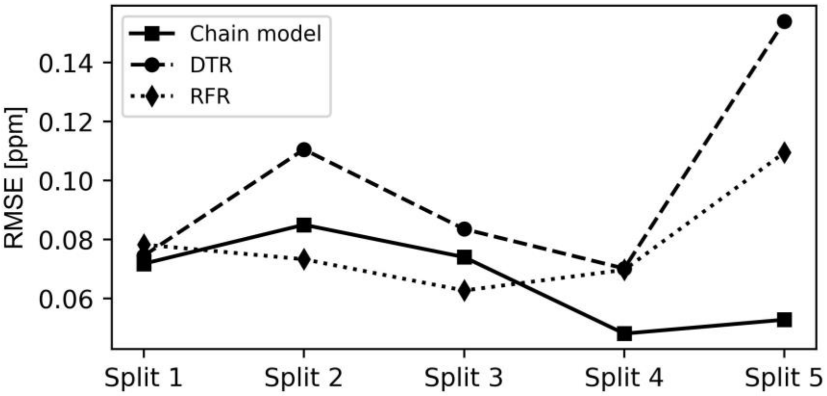


Figure 6. Accuracy distribution of the three designed ML models within their splits. RMSE values are represented as accuracy parameters.

To better assess the trained models and prevent information leakage, two additional scenarios are imported into the three ML models. The trained ML models are then utilized to predict the BTCs of these two new test scenarios . Table *1* presents the accumulated RMSEs of these two test scenarios (test set) and models' input data (train set). The ML models exhibit an increase in error when transitioning from train to test scenarios. However, even for the two new test scenarios, the RMSE remains at an acceptable level. The DTR model had the largest difference in RMSE between the train and test sets, which clearly indicates overfitting. The RFR and the chain models yield a better balance in terms of RMSE between the train and test data, suggesting their improved performance and ability to generalize.

Table 1. RMSE values of the three designed ML models within the train and test sets.

| | DTR | RFR | chain model |
|---|---|---|---|
| train set | $1.1 \times 10^{-4}$ | $1.0 \times 10^{-4}$ | $1.2 \times 10^{-4}$ |
| test set | $1.5 \times 10^{-1}$ | $4.0 \times 10^{-2}$ | $5.2 \times 10^{-2}$ |

Figure *7* shows the numerically simulated BTCs of two test scenarios and the outputs of three ML methods. For one of the test scenarios, all three ML methods achieved similar and reliable results compared to the simulation results. For the other test scenario, the DTR method was less accurate than the other methods, likely due to overfitting. The RFR and chain models had similar levels of accuracy.

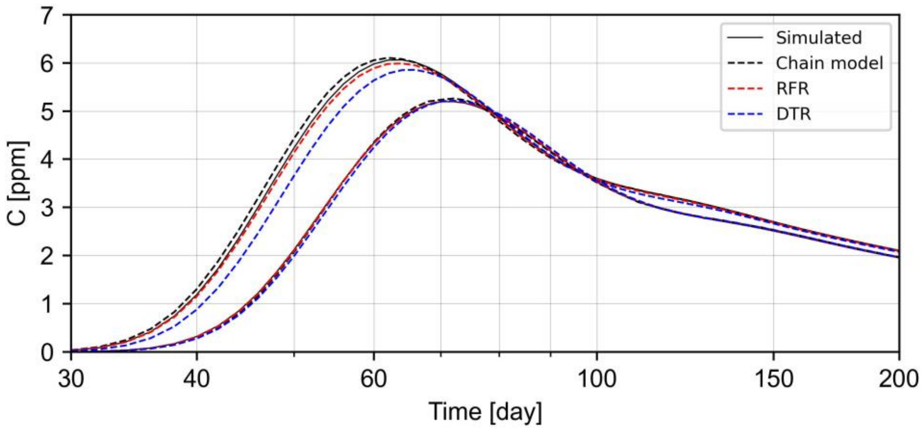


Figure 7. Two different test cases were investigated to understand the accuracy of ML models. The chain model and RFR have a high accuracy in both cases.

To further evaluate the trained models, an additional set of 2'000 different structural scenarios is generated and imported into ML models. In this step, only the connecting fault is perturbed, and the coordinates of its four corners are inputted into the three ML models. Figure 8 provides a visualization of the BTCs generated by the three ML models. These 6'000 BTCs presented in Figure 8 are calculated in the scale of milliseconds using DTR, RFR, and chain models. Two extreme cases from the training data are highlighted with blue color and dots to illustrate the boundaries of expectations. The RFR method perfectly follows the trend, generating 2'000 almost unique and parallel BTCs (Figure 8-a), which suggests that it may be underfitting the training data. The underfit models have a high bias due to oversimplifications and ignoring the underlying patterns in the train data. This problem can directly originate from the insufficient input data used to train the RFR model. The bagging procedure of RFR splits 50 input data sets into parallel bags making it difficult for each tree to be a balanced predictor. On the other hand, DTR has generated far fewer unique BTCs as shown in Figure 8-b. The covered area with BTC curves in Figure 8-a and b differs dramatically. DTR mainly repeats what it has observed in the training step. As Figure 9 shows, only a few new BTCs are generated and the majority of 2'000 BTCs overlap the 50 BTCs used in the training step.

The chain model consistently generated more reliable BTCs compared to RFR and DTR (Figure 8-c). However, in some cases, the chain model generated BTCs with irregular patterns, such as concentration values fluctuating around the peak. Despite these local discrepancies, the chain model is still the most reliable ML model for predicting BTCs.

Another notable point is that all the three data-driven ML methods are unable to be used for extrapolation. Even the frequency of generated BTCs decreases close to the extreme point for three subplots shown in Figure 8. This issue is the worst with the DTR method while the chain model has generated more BTCs in the adjacency of the extreme cases.

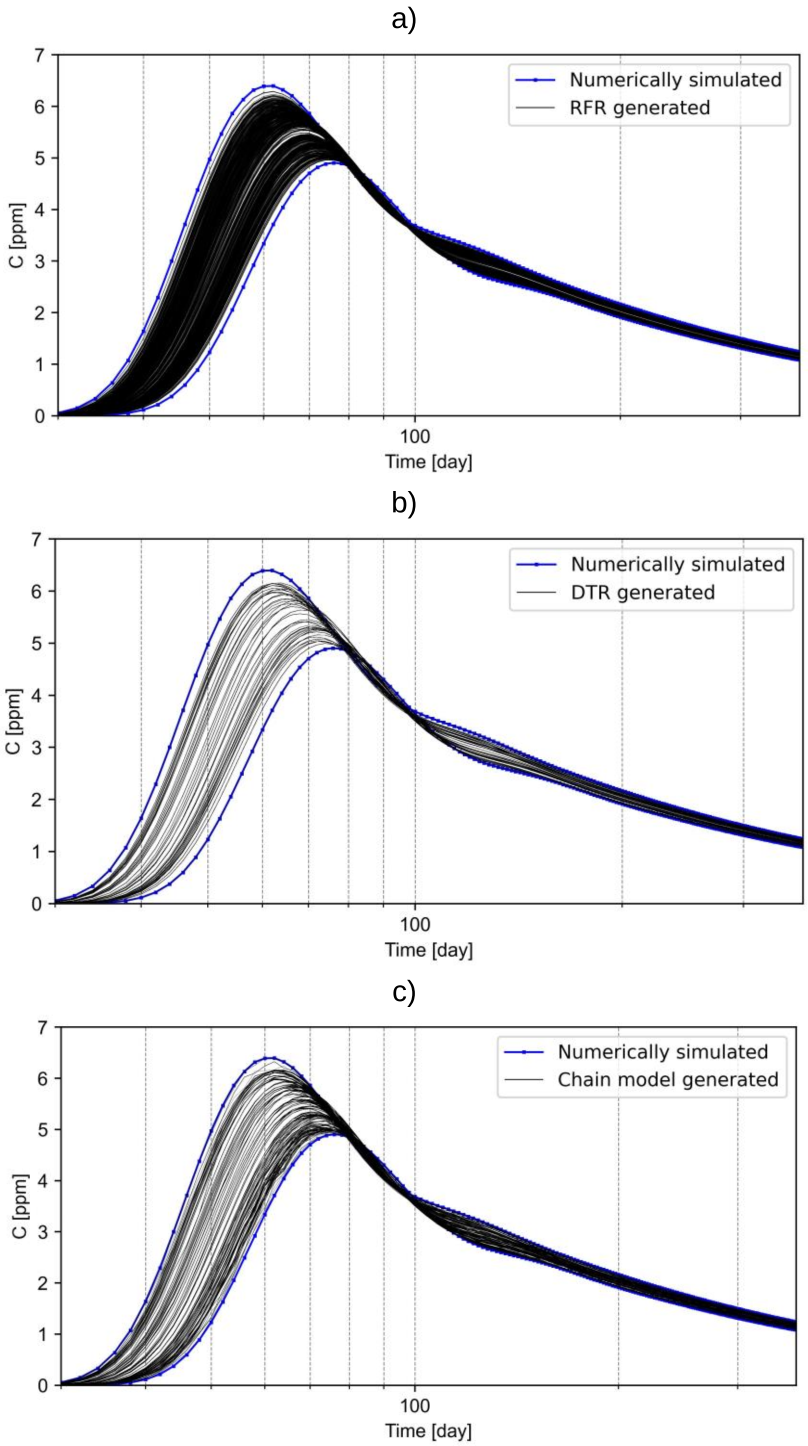


Figure 8. Two thousand generated BTCs using RFR (a), DTR (b), and chain model (c). Two extreme cases coming from the simulation are highlighted as blue curves with dots.

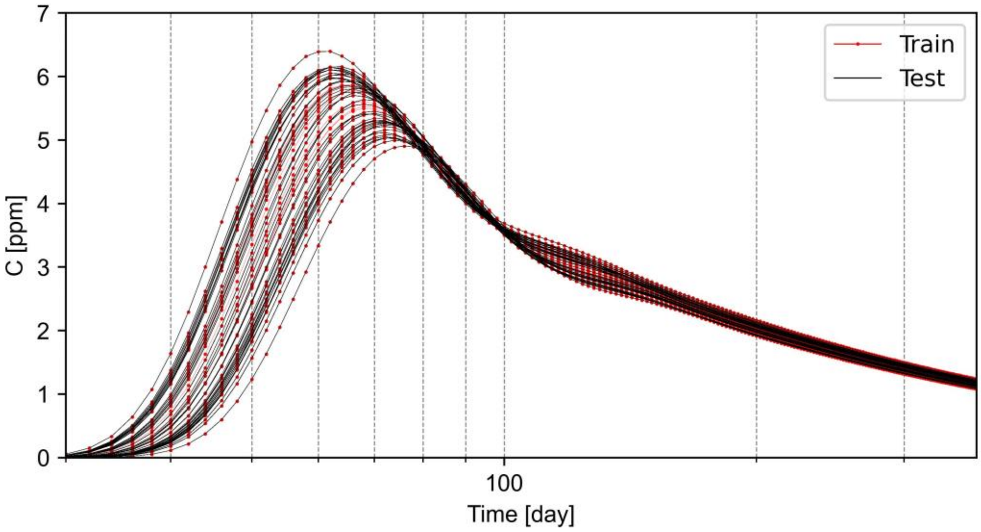


Figure 9. Most of the 2'000 BTCs generated by DTR (named Test and shown as solid black line) exactly match the input data used for training the model.

## 3.2 Complex case

For the complex case, 100 BTCs are simulated in the numerical solver and used to train and test the three ML models. The number of scenarios has increased compared to the simple case (with 50 simulations) due to the complexity of the model. In the complex case, a 24-hour tracer injection on day five of the simulation is assumed and monitored for two months in the production well. The simulation time is decreased due to the shorter/faster connection between the injection and production wells. Figure 10 shows a 2D section of the 100 unique pathways that connect injection and production wells. Two pathways are plotted with red and blue colors and are used to test the validity of the ML methods because they have not been used in the training process. Test 1 scenario visually demonstrates how the two connecting fractures can have different depths and dipping angles.

Figure 11 shows the numerically simulated BTCs for 100 scenarios of the complex case. Similar to the simple case, the peak concentration time and magnitude of the BTCs vary due to the change in the geometrical properties of the fracture network. The color coding of the train and test scenarios (1 and 2) remains consistent with Figure 10.

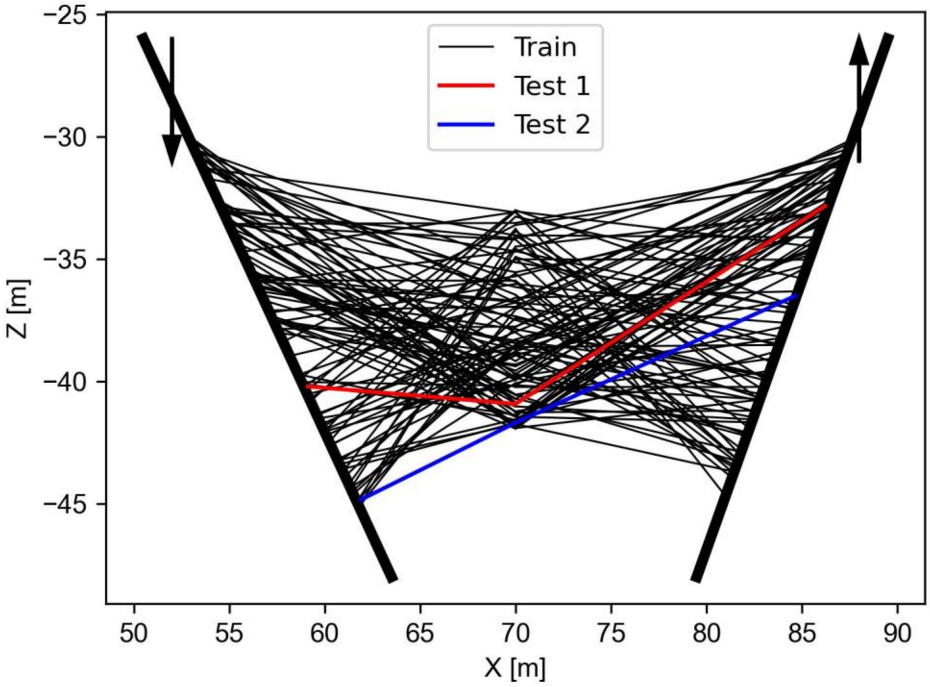


Figure 10. A 2D cross-section from the middle of the complex model. Thin black lines represent the trace of the two uncertain fractures that connect certain fractures shown via two thick black lines. The red and blue traces represent the geometry of the uncertain fractures in two tests. Arrows show the location of the injection and production wells.

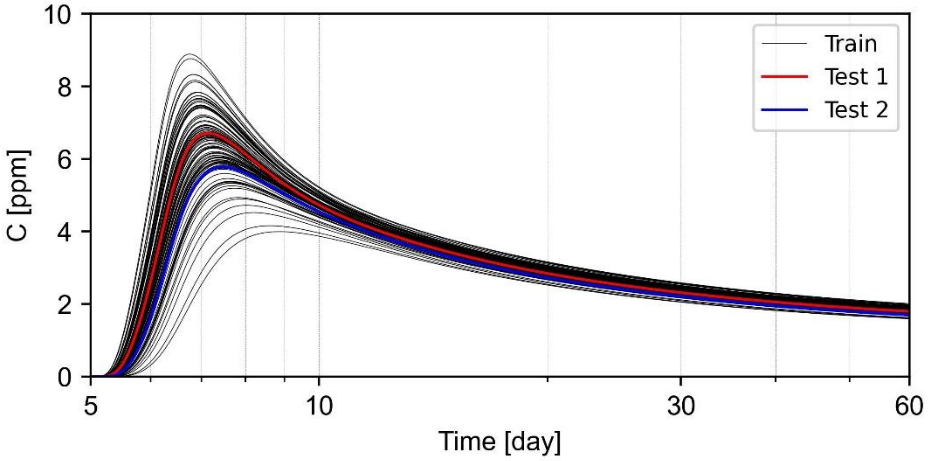


Figure 11. Thin black curves represent 98 BTCs simulated using the finite element solver. Two test scenarios are also named as Test 1 and Test 2. To see the geological model of the test cases refer to Figure 10.

Two test scenarios of the ML methods are shown in Figure 12. The RMSE values confirm the higher accuracy of the chain method. The cumulative RMSE for both scenarios is 0.05 ppm for the chain model, 0.18 ppm for DTR, and 0.16 ppm for RFR. Notably, all three machine learning models were employed with the same hyperparameters for both the simple and complex cases.

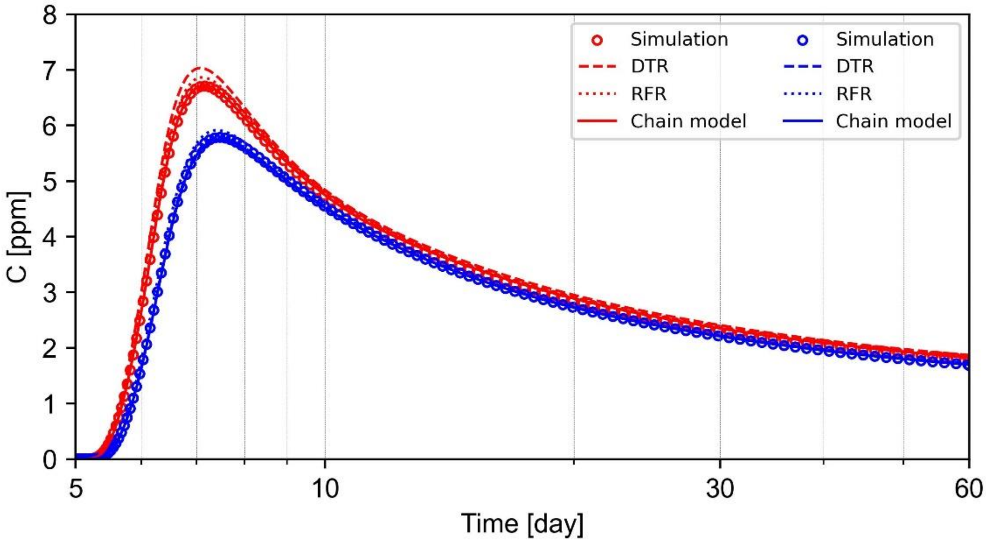


Figure 12. Simulation and ML-generated results for Test 1 are plotted as red circles and lines. Results related to Test 2 are plotted as blue circles and lines.

# 4 Discussion

Detailing the observed errors is crucial for future work aimed at improving the interpretability of the ML methods' performance. The (negligible) discrepancy likely stems from the distribution of test scenarios and size of input data (50 and 100 scenarios). This finding underscores the sensitivity of data-driven models to input data distribution. As extrapolation is a known challenge for such models, selecting a test sample near the boundary in this study exemplifies this limitation. A uniform high-density sampling strategy may prove more effective than the Gaussian distribution.

Even with large datasets, data-driven ML methods can still deviate from the underlying physics. Degen et al. (2023) proposed promising physics-based ML methods using order reduction techniques e.g., non-intrusive reduced basis, to build the solution based on basis functions that preserve the structure of the physics. In this study, we employed a sequence of concentration values as the target variable, allowing the ML models to learn the temporal relationships. Three tested ML methods have been able to capture the trend for two different cases. The current limitation is that the concentration prediction is restricted to a single point within the model. However, our strategy can be extended to develop ML models that predict the target variable at various points over time.

Meanwhile, the ML methods were significantly faster than the numerical solver, with up to six orders of magnitude reduction in computational time. To numerically solve the problem of the simple case, 12 cores on a high-performance computing cluster should run for 4 hours. The whole time for constructing (offline) and applying (online) the ML models remains in the scale of seconds. This substantial reduction makes uncertainty analysis feasible using fast and reliable ML models, without relying on time-consuming simulations that typically span multiple days. This concept can also be suited for including structural uncertainties in more complicated

EGS settings with several intersecting fractures.

## 5 Conclusion and outlook

This study presents a novel approach for using ML methods that enables quantifying the impact of structural uncertainty on BTCs in two EGS reservoirs. The approach was the first test to expand the range of structural reservoir models using ML techniques, based on an original set of a limited number of the numerical scenarios. This meets the specific requirement of uncertainty quantification, which is to provide a broad range of scenarios.

Different ML approaches are trained using the available numerical simulations to predict the BTCs based on the geometries of the perturbed elements. One ML approach used DTR and RFR algorithms to predict the entire BTC at once. Another ML approach employed a chain of GBR models to predict each time step of the BTCs while considering the correlation between consecutive time steps. The DTR model suffered from overfitting, while the RFR and chain models were more reliable, achieving an acceptable accuracy with a balanced accumulated RMSE in train and test scenarios. In the simple case, the RMSE for the DTR model jumped from 0.00011 to 0.15 between train and test scenarios, while for the RFR and chain models, it reached from 0.0001 to 0.04 and from 0.00012 to 0.052, respectively.

The trained ML models are further applied to generate BTCs for 2'000 unique structural scenarios in the model with a simple geometry. The chain model was more accurate than the RFR and DTR models. The RFR method produced 2'000 BTCs that closely follow the trend observed in the training set indicating the underfitting issue, whereas DTR can only replicate the BTCs from the training set. The chain model captures both the general trend and small-scale patterns of the data. However, the accuracy and reliability in all three methods decreases for test cases that are close to the boundaries of the input test data. A uniform sampling for selecting the input data can help the ML methods to have a wide and homogeneous distribution in the test data.

The presented approach can be adopted for a broader number of forward calculation schemes. This opens up new possibilities for more complex fractured rock settings. Rather than coordinates of one/two fractures, a more complex structural network from a real-world EGS case can be used as the input features for the ML methods.

While only structural models were varied herein to assess their impact on the BTCs, future applications could encompass modifications to specific petrophysical properties of the reservoir, further expanding the possibilities of stochasticity. Conversely, integrating more data into the model, such as BTC's or hydraulic testing data obtained from specific EGS well configurations (e.g. Schill et al., 2017), can reduce structural uncertainties. This allows for the rapid elimination of non-viable models using ML-driven routines.

Harnessing the computational efficiency of ML, this innovative approach can be transformed into a surrogate model, effectively representing the core of an inverse, backward calculation scheme for parameter identification. This transformation has the potential to replace conventional analytical solutions, which are currently the primary method for estimating parameters from tracer campaigns. The ML-based surrogate model offers several advantages, including significantly faster calculation speeds and the ability to capture the non-uniqueness inherent in mathematical solutions. In this framework, BTC data serve as the primary input, while the parameters of the complex EGS reservoir represent the target variables.

**Competing interests**

The authors declare that they have no conflict of interest.

**Acknowledgements**

Ali Dashti is receiving the financial support from The German Academic Exchange Service (Deutscher Akademischer Austauschdienst: DAAD) to do his PhD in Germany as the Research Grants-Doctoral programmes in Germany 2019/20. This organization is appreciated for giving the opportunity to researchers. The study is also part of the Helmholtz portfolio project Geoenergy. The support from the program “Renewable Energies”, under the topic “Geothermal Energy Systems”, is gratefully acknowledged. Two anonymous reviewers are appreciated for their insightful comments.

**Code and data availability**

Required data and developed ML methods for the simple and complex cases are documented and available in a Zenodo repository (https://zenodo.org/records/10810243).

# 6 References

Abbaszadeh Shahri, A., Shan, C., Larsson, S., 2022. A Novel Approach to Uncertainty Quantification in Groundwater Table Modeling by Automated Predictive Deep Learning. Nat Resour Res 31 (3), 1351–1373.

Alakeely, A., Horne, R.N., 2020. Simulating the Behavior of Reservoirs with Convolutional and Recurrent Neural Networks. SPE Reservoir Evaluation & Engineering 23 (03), 992–1005.

Alpaydin, E., 2020. Introduction to machine learning, Fourth edition ed. Adaptative Computation and Machine Learning. The MIT Press, Cambridge, London, XXIV, 682, [2] strony.

An, S., Liu, W., Venkatesh, S., 2007. Fast cross-validation algorithms for least squares support vector machine and kernel ridge regression. Pattern Recognition 40 (8), 2154–2162.

Bauer, P., Thorpe, A., Brunet, G., 2015. The quiet revolution of numerical weather prediction. Nature 525 (7567), 47–55.

Benoit, A., Lefèvre, L., Orgerie, A.-C., Raïs, I., 2018. Reducing the energy consumption of large-scale computing systems through combined shutdown policies with multiple constraints. The International Journal of High Performance Computing Applications 32 (1), 176–188.

Bond, C.E., 2015. Uncertainty in structural interpretation: Lessons to be learnt. Journal of Structural Geology 74, 185–200.

Borgonovo, E., Plischke, E., 2016. Sensitivity analysis: A review of recent advances. European Journal of Operational Research 248 (3), 869–887.

Breiman, L., 1998. Arcing classifier (with discussion and a rejoinder by the author). Ann. Statist. 26 (3).

Brunton, S.L., Kutz, J.N., 2022. Data-driven science and engineering: Machine learning, dynamical systems, and control. Cambridge University Press, Cambridge, United Kingdom, New York, NY, pages cm.

Bühlmann, P., Yu, B., 2003. Boosting With the L2 Loss. Journal of the American Statistical Association 98 (462), 324–339.

Cao, V., Schaffer, M., Taherdangkoo, R., Licha, T., 2020. Solute Reactive Tracers for Hydrogeological Applications: A Short Review and Future Prospects. Water 12 (3), 653.

Carleo, G., Cirac, I., Cranmer, K., Daudet, L., Schuld, M., Tishby, N., Vogt-Maranto, L., Zdeborová, L., 2019. Machine learning and the physical sciences. Rev. Mod. Phys. 91 (4).

Dashti, A., Gholami Korzani, M., Geuzaine, C., Egert, R., Kohl, T., 2023. Impact of structural uncertainty on tracer test design in faulted geothermal reservoirs. Geothermics 107, 102607.

Degen, D., Caviedes Voullième, D., Buiter, S., Hendriks Franssen, H.-J., Vereecken, H., González-Nicolás, A., Wellmann, F., 2023. Perspectives of Physics-Based Machine Learning for Geoscientific Applications Governed by Partial Differential Equations.

Doost, S.N., Ghista, D., Su, B., Zhong, L., Morsi, Y.S., 2016. Heart blood flow simulation: a perspective review. Biomedical engineering online 15 (1), 101.

Egert, R., Korzani, M.G., Held, S., Kohl, T., 2020. Implications on large-scale flow of the fractured EGS reservoir Soultz inferred from hydraulic data and tracer experiments. Geothermics 84, 101749.

Erdogan Erten, G., Yavuz, M., Deutsch, C.V., 2022. Combination of Machine Learning and Kriging for Spatial Estimation of Geological Attributes. Nat Resour Res 31 (1), 191–213.

Fanelli, G., Dantone, M., Gall, J., Fossati, A., van Gool, L., 2013. Random Forests for Real Time 3D Face Analysis. Int J Comput Vis 101 (3), 437–458.

Friedman, J.H., 2002. Stochastic gradient boosting. Computational Statistics & Data Analysis 38 (4), 367–378.

Gudmundsdottir, H., Horne, R., 2020. Prediction Modeling for Geothermal Reservoirs Using Deep Learning. PROCEEDINGS, 45th Workshop on Geothermal Reservoir Engineering.

Gupta, M., 2022. A Comparative Study on Supervised Machine Learning Algorithm. IJRASET 10 (1), 1023–1028.

He, J., Li, K., Wang, X., Gao, N., Mao, X., Jia, L., 2022. A Machine Learning Methodology for Predicting Geothermal Heat Flow in the Bohai Bay Basin, China. Nat Resour Res 31 (1), 237–260.

Hutter, F., Kotthoff, L., Vanschoren, J., 2019. Automated Machine Learning. Springer International Publishing, Cham.

Karniadakis, G.E., Kevrekidis, I.G., Lu, L., Perdikaris, P., Wang, S., Yang, L., 2021. Physics-informed machine learning. Nat Rev Phys 3 (6), 422–440.

Kharazmi, E., Zhang, Z., Karniadakis, G.E., 2019. Variational Physics-Informed Neural Networks For Solving Partial Differential Equations.

Knapp, E., Battaglia, M., Stadelmann, T., Jenatsch, S., Ruhstaller, B., 2021. XGBoost Trained on Synthetic Data to Extract Material Parameters of Organic Semiconductors, in: 2021 8th Swiss Conference on Data Science (SDS). 2021 8th Swiss Conference on Data Science (SDS), Lucerne, Switzerland. 6/9/2021 - 6/9/2021. IEEE, pp. 46–51.

Kohl, T., Mégel, T., 2007. Predictive modeling of reservoir response to hydraulic stimulations at the European EGS site Soultz-sous-Forêts. International Journal of Rock Mechanics and Mining Sciences 44 (8), 1118–1131.

Kotsiantis, S.B., 2013. Decision trees: a recent overview. Artif Intell Rev 39 (4), 261–283.

Liu, Y., Wang, Y., Zhang, J., 2012. New Machine Learning Algorithm: Random Forest, in: Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M.Y., Weikum, G., Liu, B., Ma, M., Chang, J. (Eds.), Information Computing and Applications, vol. 7473. Lecture Notes in Computer Science. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 246–252.

Meakin, P., Tartakovsky, A.M., 2009. Modeling and simulation of pore-scale multiphase fluid flow and reactive transport in fractured and porous media. Rev. Geophys. 47 (3).

Okoroafor, E.R., Smith, C.M., Ochie, K.I., Nwosu, C.J., Gudmundsdottir, H., Aljubran, M., 2022. Machine learning in subsurface geothermal energy: Two decades in review. Geothermics 102, 102401.

Oshiro, T.M., Perez, P.S., Baranauskas, J.A., 2012. How Many Trees in a Random Forest?, in: Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M.Y., Weikum, G., Perner, P. (Eds.), Machine Learning and Data Mining in Pattern Recognition, vol. 7376. Lecture Notes in Computer Science. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 154–168.

Probst, P., Wright, M.N., Boulesteix, A.-L., 2019. Hyperparameters and tuning strategies for random forest. WIREs Data Mining Knowl Discov 9 (3).

Raissi, M., Perdikaris, P., Karniadakis, G.E., 2019. 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.

Raschka, S., Mirjalili, V., 2019. Python machine learning: Machine learning and deep learning with python, scikit-learn, and tensorflow 2, Third edition ed. Packt Publishing, Limited, Birmingham, 741 pp.

Schapire, R.E., 2003. The Boosting Approach to Machine Learning: An Overview, in: Bickel, P., Diggle, P., Fienberg, S., Krickeberg, K., Olkin, I., Wermuth, N., Zeger, S., Denison, D.D., Hansen, M.H., Holmes, C.C., Mallick, B., Yu, B. (Eds.), Nonlinear Estimation and Classification, vol. 171. Lecture Notes in Statistics. Springer New York, New York, NY, pp. 149–171.

Schill, E., Genter, A., Cuenot, N., Kohl, T., 2017. Hydraulic performance history at the Soultz EGS reservoirs from stimulation and long-term circulation tests. Geothermics 70, 110–124.

Shu, C., Burn, D.H., 2004. Artificial neural network ensembles and their application in pooled flood frequency analysis. Water Resour. Res. 40 (9).

Soize, C., 2017. Uncertainty Quantification 47. Springer International Publishing, Cham.

Stadelmann, T., Tolkachev, V., Sick, B., Stampfli, J., Dürr, O., 2019. Beyond ImageNet: Deep Learning in Industrial Practice, in: Braschler, M., Stadelmann, T., Stockinger, K. (Eds.), Applied Data Science. Springer International Publishing, Cham, pp. 205–232.

Weigend, A.S., Gershenfeld, N.A. (Eds.), 1994. Time series prediction: Forecasting the future and understanding the past : proceedings of the NATO Advanced Research Workshop on Comparative Time Series Analysis, held in Santa Fe, New Mexico, May 14-17, 1992. A Proceedings volume, Santa Fe Institute studies in the sciences of complexity XV. Routledge, Taylor & Francis Group, New York, NY, 643 pp.

Wellmann, J.F., Horowitz, F.G., Schill, E., Regenauer-Lieb, K., 2010. Towards incorporating uncertainty of structural data in 3D geological inversion. Tectonophysics 490 (3-4), 141–151.

XU, M., WATANACHATURAPORN, P., VARSHNEY, P., ARORA, M., 2005. Decision tree regression for soft classification of remote sensing data. Remote Sensing of Environment 97 (3), 322–336.

Yu, J., Lu, L., Meng, X., Karniadakis, G.E., 2022. Gradient-enhanced physics-informed neural networks for forward and inverse PDE problems. Computer Methods in Applied Mechanics and Engineering 393, 114823.

Zhang, G., Y. Hu, M., Eddy Patuwo, B., C. Indro, D., 1999. Artificial neural networks in bankruptcy prediction: General framework and cross-validation analysis. European Journal of Operational Research 116 (1), 16–32.

Zhou, D., Tatomir, A., Niemi, A., Tsang, C.-F., Sauter, M., 2022. Study on the influence of randomly distributed fracture aperture in a fracture network on heat production from an enhanced geothermal system (EGS). Energy 250, 123781.