UNIVERSITY OF LEEDS

Deep Learning for Groundwater Prediction

Maria Luisa Taccari

Submitted in accordance with the requirements for the degree of PhD in Civil Engineering

The University of Leeds

Faculty of Engineering

School of Civil Engineering

May 2024

Intellectual Property and Publications

I, Maria Luisa Taccari, confirm that the work submitted in this thesis, which is by publications, is my own.

In all these publications, I have served as the leading and corresponding author and I have appropriately acknowledged the contributions of co-authors where applicable. My contributions included conceptualizing the research ideas, developing the methodology, creating and implementing the software, validating the results, conducting formal analysis, curating the data, writing the original draft, reviewing and editing the manuscript, and creating visualizations to represent the data and findings. Chapter 1 serves as the introduction to the thesis. Each subsequent chapter corresponds to a publication, as follow:

1. Chapter 2: **M.L. Taccari**, J. Nuttall, X. Chen, H. Wang, B. Minnema, and P. K. Jimack, "Attention U-Net as a surrogate model for groundwater prediction", *Advances in Water Resources, vol. 163, p. 104169, 2022.*

2. Chapter 3: **M.L. Taccari**, O. Ovadia, H. Wang, A. Kahana, X. Chen and P. K. Jimack, "Understanding the Efficacy of U-Net & Vision Transformer for Groundwater Numerical Modelling", *Synergy of Scientific and Machine Learning Modeling - ICML 2023 Workshop.*

3. Chapter 4: **M.L. Taccari**, H. Wang, S. Goswami, M. De Florio, J. Nuttall, X. Chen, and P. K. Jimack, "Developing a cost-effective emulator for groundwater flow modeling using deep neural operators", *Journal of Hydrology, vol. 630, p. 130551, 2024.*

4. Chapter 5: "Spatial-Temporal Graph Neural Networks for Groundwater Data", Submitted

and Under Review.

All of these publications, except for the last one which is currently under review, have undergone rigorous peer review. Chapter 6 concludes the thesis, summarizing the key findings and contributions.

Acknowledgements

The past three and a half years of my PhD journey have been incredibly enriching and fulfilling. Even with more knowledge and a more critical eye, I cannot deny that I have the same curiosity and excitement for my research as when I started.

Foremost, I extend my sincere thanks to my supervisors, whose trust and flexibility have been crucial for my research. They believed in my decisions, helping me to move forward with confidence and creativity. I'm especially thankful to Prof. Peter K. Jimack for his support and timely advice. His ability to push me, tempered with an understanding of when to step back, has given me guidance that I've truly valued. I am also grateful to Dr. He Wang for his constant presence and insightful opinions, which I have always held in high esteem. His input has been fundamental to my academic growth. A special thank you to Dr. Xiaohui Chen for initiating this project and fully supporting my enthusiasm and ideas. Similarly, Dr. Jonathan Nuttall deserves my thanks for his support that often bridged the roles of supervisor and friend, providing a foundation of both professional guidance and personal encouragement.

My visits to Deltares and my stay at Brown University were among the highlights of my PhD experience, thanks to the support and the incredible learning opportunities they presented. The period at Brown University, in particular, was filled with intense work, invaluable friendships, and learning. I owe a debt of gratitude to Prof. George Karniadakis for his hospitality and to his research group for their amazing support. Similarly, the SciML group at the University of Leeds has been a place of intellectual growth, embodying a community that we proudly built together. The support and friendship in this group have been truly inspiring.

Finally, but certainly not least, I must acknowledge the many people from both my professional and personal life who have supported me in various ways, from navigating administrative procedures to offering suggestions, providing support, and sharing the happiest and most significant memories of these years with me. As I close this chapter of my life, I am reminded that the end of my PhD is not the end of learning or of the connections I have made. These relationships and the knowledge I have gained are foundations upon which I will continue to build.

Abstract

This thesis explores the integration of advanced machine learning techniques, particularly deep learning, in enhancing groundwater prediction models. The primary focus is on developing new surrogate models that leverage deep neural networks for simulating groundwater flow, bridging the gap between traditional hydrological methods and contemporary data science approaches.

The research journey begins with the application of synthetic data and computer vision techniques and progressively advances towards handling sparse data and real-world scenarios. The thesis comprises four key papers, each contributing to the development of machine learning models for groundwater prediction. These models include convolutional encoder-decoder networks (Attention U-Net and U-Net integrated with Vision Transformer) for accurate steady-state response prediction, the DeepONet framework for generalized groundwater flow modeling under data-sparse scenarios, and finally spatial-temporal graph neural networks for long-term forecasting of groundwater levels. The research demonstrates the ability of these models to handle complex hydrological systems, predict accurately under varying conditions, and efficiently process both high-dimensional inputs and sparse data.

Overall, this thesis contributes to the field of hydrology by establishing advanced machine learning models as viable alternatives for predictive groundwater level modeling, particularly noted for their accuracy, computational efficiency, and adaptability to diverse scenarios. The findings pave the way for future research, focusing on applying these models to larger and more complex datasets for practical use in groundwater management and decision-making.

Contents

List of Figures

LIST OF TABLES

List of Tables

Chapter 1

Introduction

1.1 Motivation

Groundwater is an essential component of the global water supply and is crucial for various purposes including residential, industrial, and agricultural needs [32]. It plays a pivotal role in sustaining ecosystems, supporting biodiversity, and ensuring water security, especially in regions prone to water scarcity and drought conditions. The significance of groundwater resources is underscored by representing one-third of the world's freshwater [7], emphasizing the necessity of maintaining these resources for drinking water, irrigation, and industrial needs. However, the sustainability of groundwater is increasingly threatened by factors such as overexploitation, population growth, and climate change, which lead to environmental consequences such as aquifer depletion, land subsidence, and water quality degradation [35, 14]. Given these challenges, accurately forecasting groundwater levels (GWL) becomes paramount to the effective management and preservation of groundwater resources in order to mitigate the impacts of global population surges, urbanization, and climate-induced water shortages [28], ensuring their availability for future generations. The discipline of groundwater hydrology focuses on the effective management of groundwater systems, predicting aquifer responses to changes in their state for informed decision-making. Specifically, it seeks to forecast the distribution of water levels in aquifers at a given time [1]. The prediction of GWL is a complex task due to the dynamic and nonlinear interactions among various hydrological and meteorological variables, including rainfall, evaporation, and temperature [13]. These interactions, coupled with boundary constraints, affect GWL fluctuations, making the task of modeling these variations a challenging, yet essential, endeavor for sustainable water resource management.

In addressing the challenges of forecasting groundwater levels, a variety of numerical models have been extensively employed. These include finite difference methods (FDM) [21], finite volume methods [11], finite element methods (FEM) [29], and element-free methods [22]. Insights derived from these models into the past, current, and future states of GWL are crucial for policymakers and practitioners in the water sector to devise sustainable resource management strategies. Traditional solvers of partial differential equations, like FEM and FDM, calculate the hydraulic head as the potential energy that drives groundwater flow by iteratively solving an implicit system of equations across discretized time and flow domains. This process generates a system of equations that represent water balances at each model node or cell, necessitating the computation of unknown heads at these nodes to achieve a balance across the system [20]. These numerical models in complex, large domains face challenges in accuracy and computational demand [23]. Their resolution, execution time, and memory use scale steeply with the resolution. Moreover, once the equation is solved on a discretization, this cannot be altered, making it hard to integrate observations of various resolutions into a grid system. The precision and dependability of these numerical models also significantly depend on the availability of extensive hydrogeological data and detailed information about aquifer properties [27]. Simulating groundwater flow mandates the reconstruction of subsurface heterogeneities and aquifer physical properties, typically based on scarce direct observations. Inverse modeling, aimed at estimating unknown system parameters, requires multiple forward model runs, thus becoming highly computationally intensive as the number of unknown parameters increases. The high dependency on large volumes of data, the complexity of model calibration, and the substantial computational resources required pose significant limitations to the efficacy of traditional numerical models in groundwater modeling. Moreover, the challenges of integrating diverse data sources, defining efficient grid sizes for solving differential equations, and demarcating domain boundaries further complicate groundwater modeling efforts [33].

In recent years, new GWL forecasting techniques based on machine learning algorithms have gained popularity, as will be described in the following section. The next section highlights how these advanced computational models open new possibilities for accurate and efficient groundwater level forecasting. The purpose of this following section is to provide a succinct overview of the most relevant literature across the breadth of this thesis, noting that each subsequent chapter contains more focused, and detailed, literature reviews of the specific machine learning methodologies that are relevant to the specific context of the research in that chapter. Those seeking an extensive understanding of deep learning applications in groundwater flow analysis should consult the studies by [13, 27]. Additionally, for a foundational understanding of deep learning principles, the work by [8] serves as an essential resource.

1.2 Deep Learning for Groundwater Prediction

Artificial Intelligence (AI) is devoted to creating machines capable of performing tasks that typically require human intelligence [8]. Within this domain, *Machine Learning (ML)* centers on developing algorithms that enable computers to learn from and make decisions based on data, without being explicitly programmed. In recent years, the advent of ML and AI algorithms has revolutionized groundwater level (GWL) forecasting, offering methods that circumvent the need for detailed physical characteristics of GWL systems. Taking a step further, *Deep Learning*, a specialized branch of ML, utilizes neural networks with multiple layers that can process and interpret vast and complex datasets. The recent advancements in computational capabilities, algorithm development, and data availability have catalyzed significant progress in deep learning. Deep Learning models are renowned for their effectiveness in tasks such as classification, regression, and clustering, and have demonstrated superior performance across various fields, surpassing human abilities in complex tasks like strategic game playing and image classification [26, 3, 9]. These algorithms, including artificial neural networks (ANNs), leverage mathematical concepts to detect, classify, or predict GWL fluctuations based on data provided [13]. ANNs attempt to simulate the neural arrangement of the human brain, involving interconnected neurons linked by adjustable weights. The Multi-Layer Perceptron (MLP), a prevalent form of ANN , features a design with an input layer, an output layer for delivering predictions, and several hidden layers that facilitate the learning of complex patterns by enabling non-linear transformations at each stage. This sequential processing from input to output, coupled with the ability to adjust weights based on prediction errors through backpropagation, underscores the ANN's utility in modeling GWL dynamics.

This thesis explores how machine learning, particularly deep learning techniques, can enhance the predictive accuracy and efficiency of groundwater models. Deep neural networks, recognized as universal function approximators [10], are increasingly employed as surrogate models for solving problems in physics and engineering [12]. The evolution of AI in hydrological studies has seen the adoption of deep learning models such as recurrent neural networks (RNNs), including Long Short-Term Memory (LSTM) and Gated recurrent units (GRUs), to address the limitations of classical machine learning models in learning long-term dependencies. These developments have significantly improved the prediction efficiency on various time scales [34, 31]. LSTMs, designed for long-term time series data prediction, are equipped with memory cells capable of retaining critical information about historical events, making them well-suited for extracting non-linear spatio-temporal groundwater patterns. Zhang et al.'s study employing LSTM models to simulate water table fluctuations using a range of inputs over 14 years showed extremely high accuracy, illustrating the potential of these models to outperform traditional ANN approaches. Other studies utilize image-to-image regression techniques, employing deep convolutional neural networks to map input images to output states. Mo et al. [18] proposes a deep convolutional encoder-decoder as a surrogate model for dynamic multiphase flow problems. They demonstrated the model's capability in handling high-dimensional inputs and accurately predicting pressure and saturation fields at arbitrary time instances.

Tao H. et. al recommends that future research in GWL prediction models should place a greater emphasis on the time scale of predictions [27]. While a majority of existing studies have focused on developing predictive models on monthly and daily scales, there's a significant need to extend these efforts to long-term GWL forecasting. Yearly predictions are crucial for long-term water resource management and planning, offering decision-makers valuable insights for developing strategic policies to ensure water sustainability. Additionally, addressing the challenge of missing groundwater data is essential for improving the accuracy and reliability of GWL predictions. Groundwater observations often suffer from data sparsity and missing values due to various factors such as instrument failure or inadequate monitoring management systems, which can substantially degrade data quality and increase uncertainty in spatio-temporal groundwater analysis [5]. This study addresses these limitations by first developing general models that are not trained on data from a specific location but rely on the groundwater partial differential equation. Secondly, by applying the models to a specific location, the research tackles the challenges of sparse and missing data.

To facilitate a fair comparison of models, several key considerations should be taken into account, especially given the vast and complex landscape of possible models. Firstly, datasets should be divided into training, validation, and testing sets to prevent overfitting and ensure that models are evaluated on unseen data. Cross-validation techniques, such as k-fold cross-validation, can also help to reduce the impact of dataset bias [15]. Additionally, hyperparameter tuning should be conducted using systematic approaches, such as grid search or Bayesian optimization, to ensure optimal hyperparameters are selected for each model [2]. As highlighted by Bouthillier et al. [4], arbitrary choices in the machine learning process can skew results, making it crucial to account for variance in benchmarks to detect meaningful improvements. Their study found that performance fluctuations are influenced by various factors, with data splitting, parameter initialization, and hyperparameter choices having the greatest impact. The authors recommend randomizing sources of variation, such as data sampling, augmentation, parameter initialization, and hyperparameter choices, and accounting for variance in the results comparison. Ensuring a fair comparison between models requires optimizing each model equally, conducting multiple evaluations, and applying statistical tests to determine significant performance differences. Furthermore, the choice of evaluation metrics should reflect the specific goals of the GWL prediction task. This approach enables a coherent representation of study outcomes and facilitates the comparison of different models, including traditional GWL forecasting models. As an example from weather forecasting, the evaluation protocol of WeatherBench 2 [24], which provides a standardized protocol for AI models, closely follows the forecast verification methods used by the World Meteorological Organization and operational weather centers. This ensures that the evaluation of deep learning models is accessible to the expert community. Although there is no universally recognized set of benchmark datasets and metrics in groundwater modeling, regression models are typically used for GWL modeling [13]. Common evaluation metrics include Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Relative Error (RE), and Coefficient of Determination (R2). Special performance measures like Peak Elevation Criteria (PEC) and Low Elevation Criteria (LEC) are also used to evaluate models against critical parameters. However, RMSE and R2 are the predominant metrics in GWL modeling studies. Lastly, model evaluation in practice should also consider other metrics such as training time to reach a performance level and memory usage [25].

1.3 Research Question and Objectives

The central research question guiding this thesis is to find the most suitable neural network-based approaches for predicting groundwater flow based on the knowledge of the underlying Partial Differential Equations (PDEs). This research develops surrogate models for GWF prediction. In essence, the goal of the thesis is to create models that accurately map inputs to outputs, effectively solving the fundamental equations governing groundwater flow. The project aims to develop models suitable for groundwater management that not only expedite and refine predictions for groundwater management but also exhibit versatility across diverse real-world conditions, including different boundary settings and soil types, and efficiently incorporate observed data. Additionally, the models ideally possess properties that are independent of mesh definitions and are robust to model uncertainties.

Moreover, this thesis is structured to demonstrate the application of different machine learning techniques in groundwater modeling. The research starts from synthetic data and computer vision techniques and advances towards handling sparse data in real-world scenarios. It encompasses a series of four key papers, each integral to the overarching aim of advancing groundwater prediction models through the application of machine learning and neural networks.

Outlined within this thesis are specific objectives aimed at:

- **Objective 1:** Develop a simple model capable of learning the steady-state solution of the governing PDE in a homogeneous domain under constant boundary conditions. This serves as the foundation for creating a basic deep learning model for groundwater systems.
  Treated in Chapter 2.

- **Objective 2:** Progress to more complex and general models that can learn the steady-state solution in heterogeneous domains, accommodating a variety of boundary conditions. This step addresses the diversity and complexity inherent in real-world hydrological systems.
  Treated in Chapter 2.

- **Objective 3:** Conduct a comparative analysis between the developed models and other existing methods. This comparison aims to identify the most effective techniques and approaches for further development of the model.

Treated in Chapters 3 and 4.

- **Objective 4:** Focus on inferring solutions from limited and varied point measurements, reflecting the challenges of working with incomplete or indirect data as typically found in real-world scenarios.

  Treated in Chapters 3, 4, and 5.

- **Objective 5:** Extend the models to transient solutions of the PDE, incorporating the crucial dimension of time, which reflects the dynamic nature of groundwater flow.

  Treated in Chapters 3, 4, and 5.

- **Objective 6:** Apply the developed models in a real-world case study, benchmarking their predictions against actual measured ground truth data to assess their accuracy and applicability in practical situations.

  Treated in Chapter 5.

Each objective progressively builds upon the previous, collectively working towards the creation of sophisticated, accurate, and practical tools for groundwater modeling and prediction.

1.4 Progression of Thesis

This section summarizes each of the four papers included in the thesis, detailing how the research evolved from initial explorations using synthetic data and computer vision techniques to more advanced applications involving sparse data, neural operators, and real-world data with graph neural networks.

Paper 1 (Chapter 2): Attention U-Net as a Surrogate Model for Groundwater Prediction

This paper proposes a novel application of the Attention U-Net [19], a convolutional encoder-decoder neural network, as a surrogate model for groundwater prediction. It focuses on generating solutions for hydraulic head in heterogeneous groundwater systems using input parameters and boundary conditions. The attention mechanism allows the network to focus on relevant domain areas, resulting in detailed and accurate hydraulic head field predictions. The model shows significant efficiency and accuracy improvements over traditional numerical solvers, demonstrating its potential as a surrogate model in groundwater prediction.

Paper 2 (Chapter 3): Developing a Cost-Effective Emulator for Groundwater Flow Modeling Using Deep Neural Operators

This study introduces an emulator using a deep neural operator (DeepONet) [17] framework for groundwater flow modeling. The neural operator is capable of mapping between infinite-dimensional function spaces and is trained and evaluated with sparse data. The model accurately predicts the impact of abstraction in confined aquifers and adeptly handles various problem setups, including forward time-dependent problems, inverse analyses, and nonlinear systems. The paper also presents a novel extension to the DeepONet architecture, showing excellent performance in generating predictions for different hydraulic conductivity fields and pumping well locations.

Paper 3 (Chapter 4): Understanding the Efficacy of U-Net & Vision Transformer for Groundwater Numerical Modelling

This paper compares U-Net, U-Net integrated with Vision Transformers (ViT) [6], and Fourier Neural Operator (FNO) [16] in modeling time-dependent forward problems in groundwater systems. The paper highlights the potential of U-Net-based models in real-world groundwater modeling applications, particularly in data-sparse scenarios. The integration of U-Net with Transformers is shown to enhance predictive capabilities, making it a suitable choice for groundwater modeling tasks.

Paper 4 (Chapter 5): Spatial-Temporal Graph Neural Networks for Groundwater Data

In this study, spatial-temporal graph neural networks (ST-GNNs) [30] are applied to predict groundwater levels in the Overbetuwe area, Netherlands. Utilizing a comprehensive dataset of 395 groundwater level time series, along with auxiliary meteorological and hydrological data, the ST-GNN model effectively integrates spatial and temporal dynamics. The approach shows significant improvements over traditional models, particularly in handling missing data and providing accurate long-term forecasts, highlighting ST-GNNs potential in groundwater level prediction.

References

[1] J. Bear and A. Verruijt. *Modeling Groundwater Flow and Pollution.* Hingham, MA: Kluwer Boston Inc., 1987.

[2] J. Bergstra and Y. Bengio. "Random search for hyper-parameter optimization". In: *Journal of Machine Learning Research* 13.Feb (2012), pp. 281–305.

[3] M. Bojarski et al. "End to end learning for self-driving cars". In: (2016).

[4] Xavier Bouthillier et al. *Accounting for Variance in Machine Learning Benchmarks.* 2021. arXiv: `2103.03098 [cs.LG]`.

[5] Monidipa Das and Soumya K Ghosh. "A deep-learning-based forecasting ensemble to predict missing data for remote sensing analysis". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 10.12 (2017), pp. 5228–5236.

[6] A. Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.* `https://arxiv.org/abs/2010.11929`. arXiv preprint arXiv:2010.11929. 2021.

[7] Frédéric Frappart and Venkatesh M. Merwade. "Editorial: Groundwater systems worldwide". In: *Frontiers in Earth Science* 10 (2022). ISSN: 2296-6463. DOI: `10.3389/feart.2022.1097789`. URL: `https://www.frontiersin.org/articles/10.3389/feart.2022.1097789`.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, 2016.

[9] K. He et al. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *Proceedings of the IEEE international conference on computer vision.* 2015, pp. 1026–1034.

[10] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators". In: *Neural networks* 2.5 (1989), pp. 359–366.

[11] Pierre Jamin et al. "Direct measurement of groundwater flux in aquifers within the discontinuous permafrost zone: an application of the finite volume point dilution method near Umiujaq (Nunavik, Canada)". In: *Hydrogeology Journal* 28.3 (2020), pp. 869–885.

[12] George Em Karniadakis et al. "Physics-informed machine learning". In: *Nature Reviews Physics* 3.6 (2021), pp. 422–440.

[13] Junaid Khan et al. “A Comprehensive Review of Conventional, Machine Leaning, and Deep Learning Models for Groundwater Level (GWL) Forecasting”. In: *Applied Sciences* 13.4 (2023), p. 2743.

[14] Leonard F Konikow and Eloise Kendy. “Groundwater depletion: A global problem”. In: *Hydrogeology Journal* 13 (2005), pp. 317–320.

[15] M. Kuhn and K. Johnson. *Applied Predictive Modeling*. Springer, 2013.

[16] Zongyi Li et al. “Fourier neural operator for parametric partial differential equations”. In: *arXiv preprint arXiv:2010.08895* (2020).

[17] L. Lu et al. “Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators”. In: *Nature Machine Intelligence* 3.3 (Mar. 2021. doi: https://doi.org/10.1038/s42256-021-00302-5).

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

[19] O. Oktay et al. *Attention u-net: Learning where to look for the pancreas*. 2018.

[20] T. Olsthoorn. *Finite Difference Groundwater Modeling in Python*. 2016.

[21] Padam Jee Omar et al. “Groundwater modelling using an analytic element method and finite difference method: An insight into Lower Ganga river basin”. In: *Journal of Earth System Science* 128.7 (2019). Cited by: 22; All Open Access, Bronze Open Access. DOI: 10.1007/s12040-019-1225-3. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85068756160&doi=10.1007%2fs12040-019-1225-3&partnerID=40&md5=c113b69f1b822a7803224c3e6494c0c1.

[22] Tinesh Pathania et al. “Simulation of groundwater flow in an unconfined sloping aquifer using the element-free Galerkin method”. In: *Water Resources Management* 33 (2019), pp. 2827–2845.

[23] D. Pulido-Velazquez et al. “A general methodology to simulate groundwater flow of unconfined aquifers with a reduced computational cost”. In: *Journal of Hydrology* 338.1 (2007), pp. 42–56.

[24] Stephan Rasp et al. *WeatherBench 2: A benchmark for the next generation of data-driven global weather models*. 2024. arXiv: 2308.15560 [physics.ao-ph].

[25] Vijay Janapa Reddi et al. "Mlperf inference benchmark". In: *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*. IEEE. 2020, pp. 446–459.

[26] D. Silver et al. "Mastering the game of Go with deep neural networks and tree search". In: *Nature* 529.7587 (2016), pp. 484–489.

[27] Hai Tao et al. "Groundwater level prediction using machine learning models: A comprehensive review". In: *Neurocomputing* 489 (2022), pp. 271–308.

[28] Isabel Tubau et al. "Quantification of groundwater recharge in urban environments". In: *Science of The Total Environment* 592 (2017), pp. 391–402. ISSN: 0048-9697. DOI: `https://doi.org/10.1016/j.scitotenv.2017.03.118`. URL: `https://www.sciencedirect.com/science/article/pii/S0048969717306307`.

[29] C Ukpaka, S Nwozi-Anele Adaobi, and C Ukpaka. "Development and evaluation of trans-amadi groundwater parameters: The integration of finite element techniques". In: *Chem. Int* 3 (2017), p. 306.

[30] Zonghan Wu et al. "Connecting the dots: Multivariate time series forecasting with graph neural networks". In: *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 2020, pp. 753–763.

[31] A. Wunsch, T. Liesch, and S. Broda. "Groundwater level forecasting with artificial neural networks: a comparison of long short-term memory (LSTM), convolutional neural networks (CNNs), and non-linear autoregressive networks with exogenous input (NARX)". In: *Hydrology and Earth System Sciences* 25.3 (2021), pp. 1671–1687. DOI: `10.5194/hess-25-1671-2021`. URL: `https://hess.copernicus.org/articles/25/1671/2021/`.

[32] IS Zektser and Hugo A Loaiciga. "Groundwater fluxes in the global hydrologic cycle: past, present and future". In: *Journal of hydrology* 144.1-4 (1993), pp. 405–427.

[33] Nejat Zeydalinejad. "Artificial neural networks vis-à-vis MODFLOW in the simulation of groundwater: A review". In: *Modeling Earth Systems and Environment* 8.3 (2022), pp. 2911–2932.

[34] Jianfeng Zhang et al. "Developing a Long Short-Term Memory (LSTM) based model for predicting water table depth in agricultural areas". In: *Journal of Hydrology* 561 (2018), pp. 918–929. ISSN: 0022-1694. DOI: `https://doi.org/10.1016/j.jhydrol.2018.04.065`. URL: `https://www.sciencedirect.com/science/article/pii/S0022169418303184`.

[35] Xiang-Lian Zhou, Kai-Yong Huang, and Jian-Hua Wang. “Numerical simulation of groundwater flow and land deformation due to groundwater pumping in cross-anisotropic layered aquifer system”. In: *Journal of Hydro-environment Research* 14 (2017), pp. 19–33. ISSN: 1570-6443. DOI: https://doi.org/10.1016/j.jher.2016.08.001. URL: https://www.sciencedirect.com/science/article/pii/S1570644316302647.

Chapter 2

Attention U-Net as a Surrogate Model for Groundwater Prediction

Abstract

Numerical simulations of groundwater flow are used to analyze and predict the response of an aquifer system to its change in state by approximating the solution of the fundamental groundwater physical equations. The most used and classical methodologies, such as Finite Difference (FD) and Finite Element (FE) Methods, use iterative solvers which are associated with high computational cost. This study proposes a physics-based convolutional encoder-decoder neural network as a surrogate model to quickly calculate the response of the groundwater system. Holding strong promise in cross-domain mappings, encoder-decoder networks are applicable for learning complex input-output mappings of physical systems. This manuscript presents an Attention U-Net model that attempts to capture the fundamental input-output relations of the groundwater system and generates solutions of hydraulic head in the whole domain given a set of physical parameters and boundary conditions. The model accurately predicts the steady state response of a highly heterogeneous groundwater system given the locations and piezometric head of up to 3 wells as input. The network learns to pay attention only in the relevant parts of the domain and the generated hydraulic head field corresponds to the target samples in great detail. Even relative to coarse finite difference approximations the proposed model is shown to be significantly faster than a comparative state-of-the-art numerical solver, thus providing a base for further development of the presented networks as surrogate models for groundwater

prediction.

2.1 Introduction

Groundwater resources are of major importance for residential, industrial and agricultural use. However, the quality and availability of groundwater supplies are significantly affected by their overexploitation around the world, population growth and climate extremes [2]. Consequently, a demanding need exists for quick and accurate evaluation of multiple management alternatives over long time horizons. The last 30 years have seen the development of several physics-based numerical models for simulating groundwater systems, with Finite Difference (FD) and Finite Element (FE) discretizations of the partial differential equations (PDEs) as the most used and classical methodologies [26, 27]. These techniques calculate the hydraulic head by iteratively solving an implicit system of equations at each time step in the discretized time and flow domains. Running the groundwater model in a complex system within a large domain and with reasonable accuracy incurs numerical challenges and an excessive computational demand [33, 3]. As the computational cost increases super-linearly with the number of unknowns in the discretization, long runtimes are a major challenge when high resolution is required or when many executions are necessary, such as in uncertainty analysis, sensitivity analysis, and inverse modelling. Mens et al. [14], discuss the case of the National Water Model (NWM) that is used for national policy-making on drought risk management in the Netherlands and whose heavy computational burden poses limits to quickly responding to policy questions. The authors advocate the need for a fast simple model that describes all relevant processes and is quick enough to explore many scenario and strategy combinations for long time series. Furthermore, groundwater flow simulations require the reconstruction of subsurface heterogeneities and the physical properties of the aquifer as inputs to the model, for which only limited direct observations are available. Inverse modelling is used to estimate the unknown parameters of the system, taking into account their stochasticity.

Traditional approaches to the inversion problem correspond to iterative techniques and necessitate a large number of forward model runs. As the number of unknown parameters increases, forward operations become extremely computationally demanding. Surrogate models are cheaper-to-run models which approximate the response of a complex and computationally intensive model. Surrogate models have been used in a number of groundwater studies, such as for optimization design [32, 6] and uncertainty quantification problems[7, 39], to name a few. Reduced-fidelity models simplify the level of complexity of the physical processes of the full-order model, e.g. by projecting the governing equations into a transformed space of smaller dimension. Projection-based techniques can accurately retain the underlying structure of the full-order model; however these methods can suffer stability and robustness issues [22, 15], they are highly code-intrusive and they cannot efficiently treat strong nonlinearity [4]. Data-driven models learn the response of the system from the simulation data in a supervised manner. Gaussian processes have been successfully applied to uncertainty quantification tasks for which the training data are limited but they rely on specific *a priori* assumptions on the relationship between the input and the outputs and have high computational costs when dealing with large datasets. [18, 1]

Deep neural networks are universal function approximators and are becoming increasingly common surrogate models for solving problems within the fields of physics and engineering. These techniques have been applied for solving PDEs in high-dimensional settings and nonlinear systems, with potential applications in parameter estimation and uncertainty quantification. The reader is referred to Karniadakis et al. [17] for a review on the strengths, limitations, current applications and outlook of this class of deep learning algorithms.

Recently interest has grown for learning complex nonlinear, multiscale, and high dimensional mappings of subsurface processes. In the work of Geneva and Zabaras [12], convolutional neural networks (CNNs) for physics-constrained learning show exceptional performance, with solutions obtained an order of magnitude faster than with state-of-the-art numerical solvers. They train deep auto-regressive convolutional neural network models to learn the dynamics of three transient PDEs (1D Kuramoto-Sivashinsky equation, 1D Burgers' equation and the 2D coupled Burgers' system) without any off-line training data. Several studies adopted an adversarial network framework for surrogate methods for a single-phase flow forward model and a multiphase flow forward model [34, 40]. Dagasan et al., [8], argue that the use of a conditional generative adversarial network (cGAN) as a surrogate forward model for groundwater systems can reduce the computational time by up to 80% compared to the numerical solver MODFLOW.

Deep neural networks were chosen in this study largely due to their scalability and their ability to learn based on a few *a priori* assumptions. The first refers to the capacity to learn from massive amounts of data. Compared to a Gaussian process, whose runtime scales poorly with the size of the datasets, deep neural networks can assimilate large amounts of multi-fidelity observational data, even in partly understood, uncertain and high-dimensional contexts. Compared to reduced order model techniques, which aim to bring the physical relationships of full order models at a much lower dimension, deep neural networks do not assume any prior assumption that constrains the relationship between input and output samples. This flexibility can lead deep neural networks to learn complex relationships, thus increasing their modelling power but at the cost of a lower interpretability.

The encoder-decoder architecture consists of a contracting and an expansive path. It shows robust and accurate performance in various tasks including machine translation problems [38], semantic segmentation [25] or depth regression [9]. Initially developed for biomedical image segmentation [28], U-Net is an encoder-decoder network which uses fully convolutional networks and requires highly limited training samples. U-net based architectures have been applied across a wide spectrum of application areas, such as image super resolution, style transfer, text-to-image translation and image-to-image translation [16]. Mo et al. [24], developed a deep convolutional encoder-decoder network method as a surrogate model for transient multiphase flow models. Given the large approximation errors in the concentration fields near the source release location, the authors assign an additional weight to the loss at the eight pixels around the source release location in order to improve the surrogate predictive capability. Attention models address this limitation by allowing the model to learn to focus selectively on the relevant parts of the input. Attention has recently become an essential component of neural architectures within diverse application domains [5, 11, 19]. Attention U-net makes use of attention gates in order to focus on specific parts of the image that are of importance while paying little attention to unnecessary areas [25, 31].

The purpose of this paper is to propose an Attention U-Net network as a surrogate model for the forward operator in groundwater modelling. The encoder-decoder model learns the mapping between model inputs and output for deterministic, steady-state solutions of the two-dimensional groundwater flow equation given a highly heterogeneous subsurface domain. The surrogate model accurately captures the nonlinear relationship between the hydraulic conductivity and the subsurface groundwater map. The model dynamically pays attention to only the parts of the input where flow can take place in a manner that helps the network in learning the mapping effectively.

The rest of the paper is organized as follows. Section 2 presents the adopted image-to-image deep learning approach and the architecture of the Attention U-Net employed. Section 3 provides an overview of the problem formulation and model set-up along with training of the surrogate model. The proposed method is evaluated with and without attention gates in section 4. Finally, the conclusions are formulated in the last section.

2.2 Methodology

2.2.1 Surrogate Modelling as Image-to-Image Regression

A surrogate model $\hat{f}(x, \theta) \approx \hat{y}$ approximates the 'ground-truth' function $y = f(x)$ where $f : X \to Y$ is the mapping between the input domain X and the output domain Y, $x \in X$ is the input, $y \in Y$ is the output, and θ are the model parameters. In the case of forward solving of PDEs with machine learning, the ground truth mapping represents some combination of the solution of the PDEs governing the physical system, and the surrogate model $\hat{y} = \hat{f}(x, \theta)$ is trained using a dataset D of N simulation data: $D = \{(x^i, y^i)\}_{i=1}^{N}$.

By adopting an image-to-image regression approach, the surrogate modelling can be treated as an image regression problem. By solving the PDE over a spatial domain, such as 2D regular grids, the simulation data can be thought of as images, with inputs $x^i \in \mathbb{R}^{d_x \times H \times W}$ and outputs $y^i \in \mathbb{R}^{d_y \times H \times W}$ where d_x and d_y are the number of input and output channels respectively, each with a resolution of $H \times W$ (height $\times$ width). The surrogate modelling problem becomes an image-to-image regression problem with the regression function $\hat{f} : \mathbb{R}^{d_x \times H \times W} \to \mathbb{R}^{d_y \times H \times W}$[41].

2.2.2 Encoder-decoder model

Encoder-decoder is a learning method with an analysis path (encoder) and a synthesis path (decoder). The encoder network transforms high-dimensional unlabeled input data x into low-dimensional embeddings z (latent space) and the decoder maps z to the intended output y = decoder $\circ$ encoder(x). The input is passed through a series of layers that progressively down sample until a bottleneck layer, at which point the decoder restores the spatial dimensions to produce the output images. Intuitively, the model corresponds to a coarse-refine process: the encoder reduces the spatial dimension of the input image to high-level coarse features, and the decoder recovers the spatial dimension by refining the coarse features. The assumption is that the input and output images share the underlying structure, or they are different renderings of the same underlying structure, that is their structures are roughly aligned [16].

As the goal of this study is to generate a targeted output image corresponding to given inputs, the Encoder-Decoder model learns the mapping x $\rightarrow$ y from a conditioning input image x to the output image y. The network converts images from the source to target domains, where the first corresponds to the initial, boundary conditions and model parameters and the latter to the resolution of the governing equation given those constrains.

2.2.3 Deep Convolutional Neural Networks

CNNs [23, 29] are popular deep learning networks specialized in image processing [21, 35]. While the first layers detect basic features, deeper convolutional layers learn higher representation. A convolution layer is a linear transformation that highlights the presence of a given feature in the map while preserving spatial information in the input image [13]. Given a 2-D input image and a square kernel ω with size m, the convolutional layer outputs the value at location (i, j) by summing up the contributions from the previous layer cells y^{l-1} weighted by the filter components; then, the nonlinearity σ is applied.

$$y^{l}_{ij} = \sigma\left(\sum_{a=0}^{m}\sum_{b=0}^{m} w_{ab}\, y^{l-1}_{(i+a)(j+b)}\right) \tag{2.1}$$

The stride of the convolutional layer is a parameter that determines the number of pixel shifts between two successive moves of the filter, while the padding indicates the amount of pixels with value zero added at each side of the boundaries of the input. The rectified linear unit function (ReLU) is a piecewise linear function that outputs the input if it is positive and zero if negative. The Leaky ReLU with slope coefficient modifies the function to allow a small, negative, output when the input is negative:

$$\sigma(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{otherwise} \end{cases} \tag{2.2}$$

Batch normalization and dropouts are used to stabilize training and mitigate overfitting [30]. A dropout layer selects a random set of units from the preceding layer and ignores their output, while batch normalization standardizes the layer's inputs by calculating the mean and standard deviation across the batch.

2.3 Application

2.3.1 Groundwater model and datasets

Consider steady-state groundwater flow in saturated media satisfying the fundamental governing equation [36]:

$$\nabla \cdot (K \nabla h) + q = 0 \tag{2.3}$$

The piezometric head h [L] is the field variable of interest, $\boldsymbol{K}$ is the input hydraulic conductivity [L/T] and q represents the source (or sink) terms [L^3 T^1].

The problem of this study consists of steady-state flow in a single-layer model representing a heterogeneous confined aquifer. Initially, in this work, only Dirichlet boundary conditions are considered and the groundwater head values are fixed in the cells in which the allocated head is known. The model takes in an input image with three channels: head values, boundary markers and spatially varying hydraulic conductivity (Figure 2.1). Dirichlet boundary conditions are imposed on the four sides of the square domain. Head is constant at up to three random locations across the domain, representing wells. The source term q is set to zero. The second channel of the input image is a binary mask where the boundary markers identify the cells with a fixed value, i.e. well locations and boundary cells as defined by the first source image. The last input channel defines the heterogeneous media. The conductivity field $\boldsymbol{K}$ of the highly-heterogeneous aquifer is a Gaussian random field [37] in which the values of hydraulic conductivity are taken from a finite set of values.

This application example demonstrates the capability of an Attention U-Net to successfully learn and simulate a common hydrologic situation using an image-to-image translation approach. The model is trained to predict the output fields consisting of the spatial components of the groundwater head in the domain. These predictions are compared against simulation results obtained by the fully-implicit finite difference model MODFLOW [26], here called “target images”, bearing in mind that finite difference results provide an approximation of the partial differential equation and are not error free.

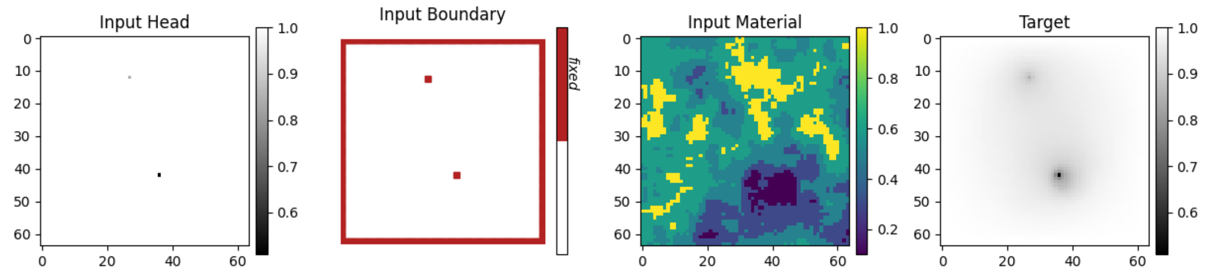


Figure 2.1: Input and output channels for an example taken from the training dataset. (Left to Right) The values of the piezometric head at the boundaries (input- channel #1); the location of the boundaries (input- channel #2); the hydraulic conductivity field (input- channel #3); the hydraulic head in the whole domain (output).

2.3.2 Network architecture

The Encoder-Decoder model is implemented as Attention U-Net [25, 28] and the employed network architecture is shown in Figure 2.2 (for the case of a 64×64 input image, as used in our computational tests). This is an encoder-decoder model with skip connections. In the down-sampling half the inputs are encoded with a series of CNNs with kernels of size 4, stride of 2 and padding set to 1. In each block, CNN is followed by a Batch Normalization layer, Dropout with rate 0.5 and a Leaky ReLU with slope 0.3. As the number of filters increases to 512 and the size of the input images reduces to 4x4, the encoder captures high-level abstract information. In the up-sampling half the representations are expanded spatially and the number of channels is reduced by a series of CNNs and up-sampling layers. The last up-sampling layer is followed by a transposed convolution layer with a sigmoid activation function to ensure predicted values between 0 and 1. Skip connections link the layers in the encoder with corresponding layers with the same-sized feature map in the decoder [28]. The only difference between Attention U-Net and the original U-Net architecture is that in the Attention U-Net network skip connections are additionally passed through attention gates, which use additive soft attention [25]. The attention coefficients are larger if the vector from the next lowest layer of the network in the up-sampling path and the corresponding vector from the encoder going through the skip connection are aligned. The weights are multiplied element-wise to the original vector which passes along in the skip connection. In this way, the attention gate (AG) mechanism allows the U-Net to suppress irrelevant regions and focus more on target structures of varying size and shape. For reproducibility full details of the network architecture are described in Appendix 3.A.

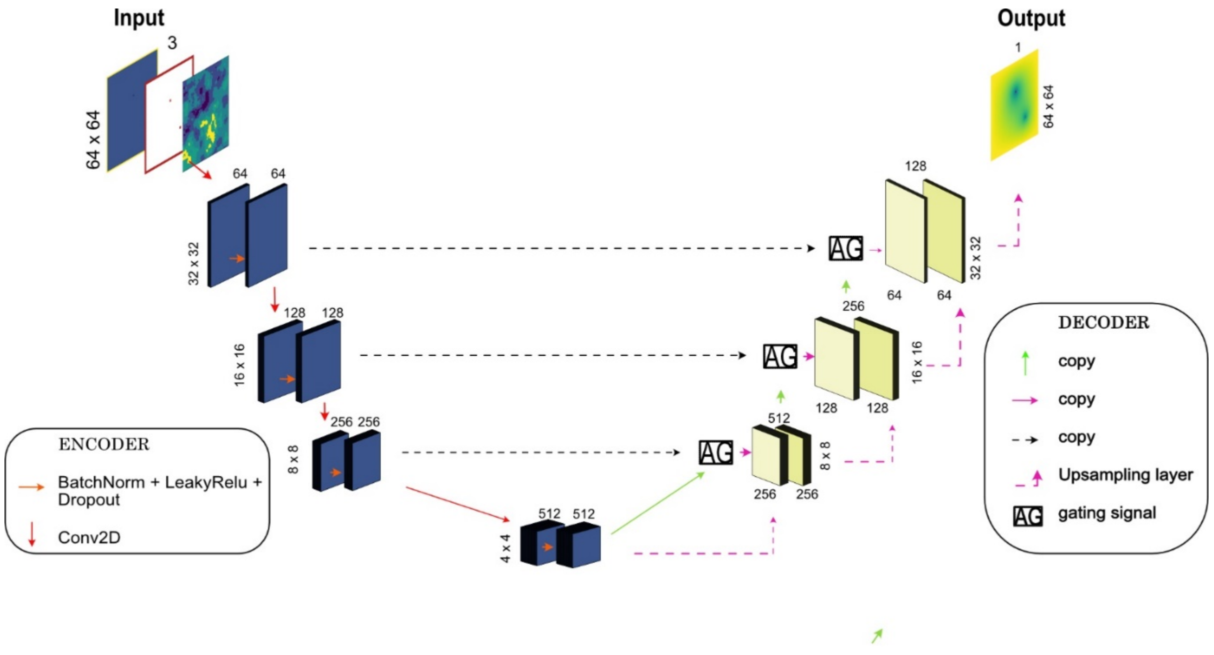


Figure 2.2: Attention U-net architecture as the surrogate model. The model has three input channels and one output channel as shown illustrated in Figure 2.1. The first part is the down-sampling half (left, in blue): the inputs are encoded with a series of CNNs with kernels 4x4 and stride of 2 and down-sampling layers. The second half (right, in yellow) is the up-sampling part: the representations are expanded spatially and the number of channels is reduced.

2.3.3 Loss function

The aim of the regression task is to minimize the mean square error (MSE) between the generated samples and training data. The network computes the average loss across a mini-batch of size $\mathrm{N_b}$:

$$\mathcal{L}_{MSE} = \frac{1}{N_b}\sum_{j=1}^{N_b}\left(\frac{1}{n}\sum_{i=1}^{n}(y_{j,i}-\hat{y}_{j,i})^2\right) \tag{2.4}$$

where y is the training image, $\widehat{y}$ is the image generated by the network and n denote the total number of pixels of each image. The networks tries to be near the ground truth output in an L2 sense.

2.3.4 Network Training

The model is trained in supervised fashion. For the examples described in the following section a dataset consisting of 32000 training data is used, and the development of the loss function is compared with 8000 validation data. The size of the training dataset is big enough to ensure the model's ability to generalize and the generation time is less than 3 hours.

The losses are minimized using the Adam optimizer [20] with a starting learning rate is 8×10^4. The network is trained for 130 epochs. Training converged after approximately 2 hours, by training the models on an Intel(R) Xeon(R) GPU Tesla K80.

The quality of the trained network is evaluated by reporting the coefficient of determination (R^2) and the root mean squared error (RMSE) between each pixel value from the target image and each pixel value from the generated image:

$$R^2 = 1 - \frac{\sum_{i=1}^{N} \|y_i - \hat{y}_i\|_2^2}{\sum_{i=1}^{N} \|y_i - \overline{y}\|_2^2}, \quad RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \|y_i - \hat{y}_i\|_2^2} \tag{2.5}$$

where y is the target image, $\overline{y}$ is the mean of the target images of the dataset $\overline{y} = \frac{\sum_{i=1}^{N} y_i}{N}$, $\widehat{y}$ is the network prediction, and N is the total number of samples. The two selected metrics between them yield complementary and representative information for the evaluation of the trained model.

2.4 Results and Discussion

2.4.1 Model predictions

This section presents both qualitative and quantitative techniques to test the performance of the model. The test case considers a square domain $\Omega = [0, 64] \times [0, 64]$ consisting of 64 rows and 64 columns, with the width of each cell equal to one. The boundary is assumed to be constant head boundary with head of 1, while the imposed head values in the wells lie in the range [0.5,1). The number of wells, their locations and their values are randomly selected and vary for each data sample. The conductivity field, K, of the highly-heterogeneous aquifer is generated as a continuous Gaussian random field, which is then discretized into a finite set of values. The heterogeneous hydraulic conductivity field has values belonging to 5 different classes (0.1, 0.325, 0.55, 0.775, 1.) which can be thought of as 5 soil types distributed within the model. The model is trained with loss function in Equation 2.4 and the target response is the finite difference simulation.

To illustrate the superior performance of the proposed Attention U-Net architecture against the original U-Net network architecture, the U-Net network without attention gates is also trained using the same training sets and parameters. At the end of the training, the Attention U-Net network achieves a RMSE of 1.98×10^{-3} and a R^2 score of 0.996, while those obtained by U-Net are 3.78×10^{-3} and 0.986, respectively (Figure 2.3).

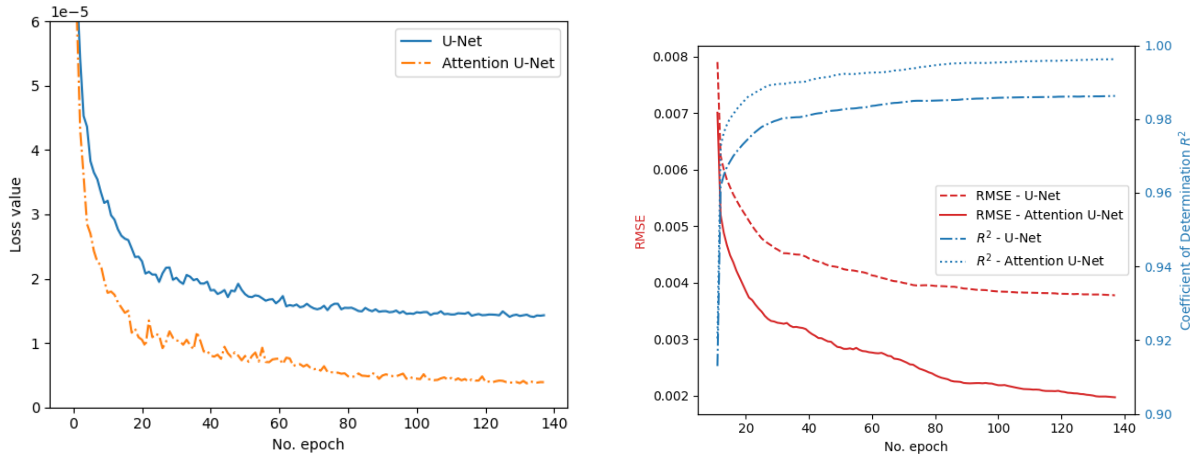


Figure 2.3: (Left to Right) Loss curves for U-Net (solid line) and Attention U-Net (dashed line); RMSE and R^2 scores of the model evaluated on the training dataset for U-Net and Attention U-Net.

Figure 2.4 provides a comparison of generated images of groundwater head with the target images for 5 random examples taken from the test dataset with a set size of 4000 samples. The Attention U-Net model has learnt to map the flow patterns: it generates accurate predictions for varied input samples that are unseen during training. The predictions match the target images very well: the model predicts the correct value of groundwater head and the pattern of its distribution. The model is able to identify and focus on salient image regions: the attention coefficients are highest at the boundary of the domain and near the well locations, while they are low in the areas with small head distribution gradients.

When trained without attention gates, U-Net can predict the values of the groundwater head in the domain, but the generated outputs have some minor deviations especially at a distance from the source area and the head gradients are smaller. The use of the attention mechanism significantly improves the accuracy of the results.

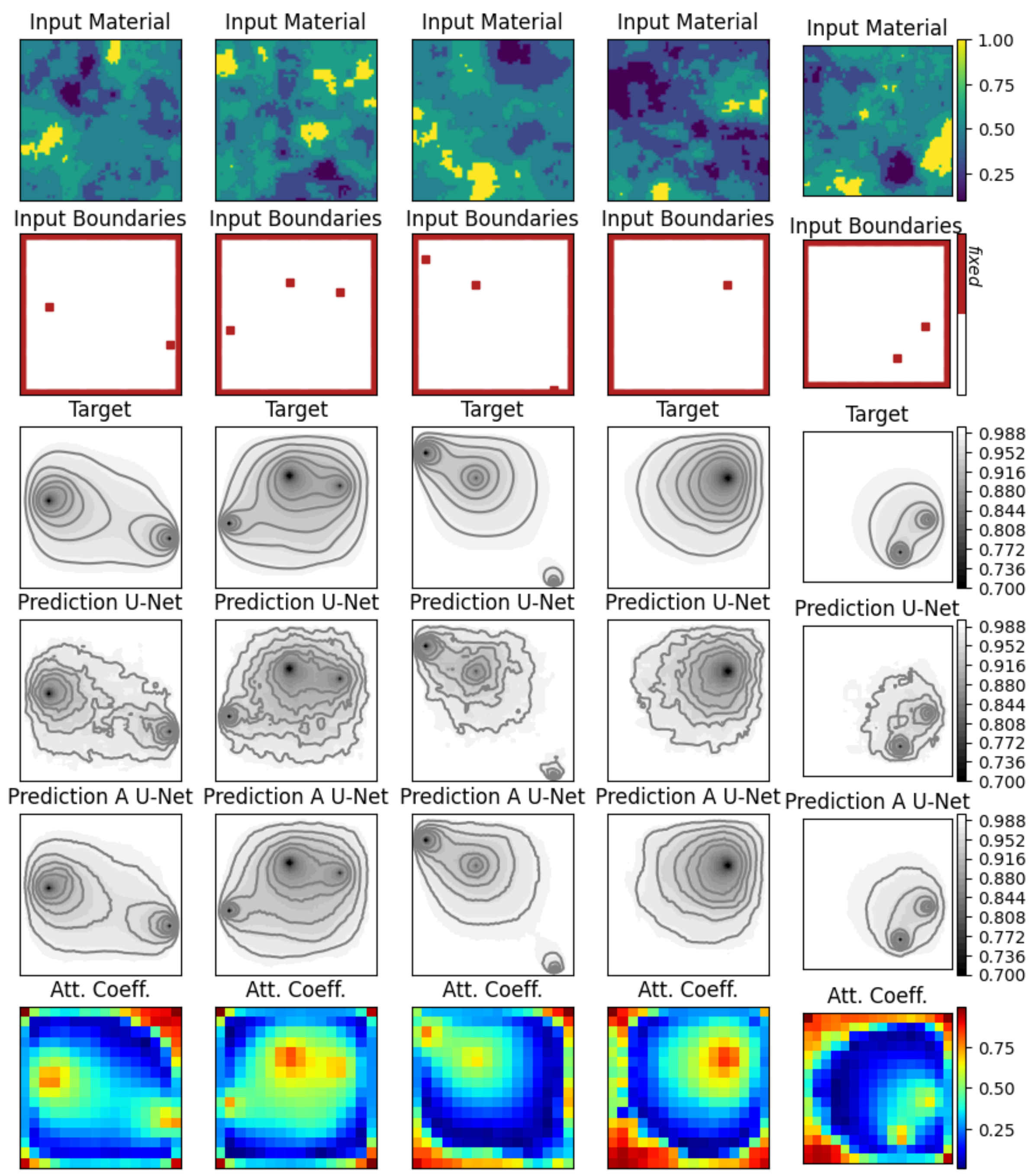


Figure 2.4: Comparison between the target sample (MODFLOW) and the learned solution (prediction) for the U-Net and the Attention U-Net models for five randomly selected samples from test dataset. (From top to bottom) The input spatially varying hydraulic conductivity; the location of the input boundaries; the target prediction; the result for the U-Net; the result for the Attention U-Net (A U-Net) ; the attention coefficients learnt by the Attention U-Net. The images in each row share the same colour map with the values given in the rightmost column. The contour lines in the target and prediction images represent the values: 0.9, 0.92, 0.94, 0.96, 0.98.

Figure 2.5 visualizes the attention coefficients obtained from two test images with respect to training epochs. During the first 20 epochs, the loss function rapidly decreases (Figure 2.3, top) and the attention gates learn to identify the location of the wells, the boundaries and a rough outline of the area with large head distribution gradients. By training the network for longer epochs, the attention coefficients are gradually updated and refined to focus on areas with large head distribution gradients.

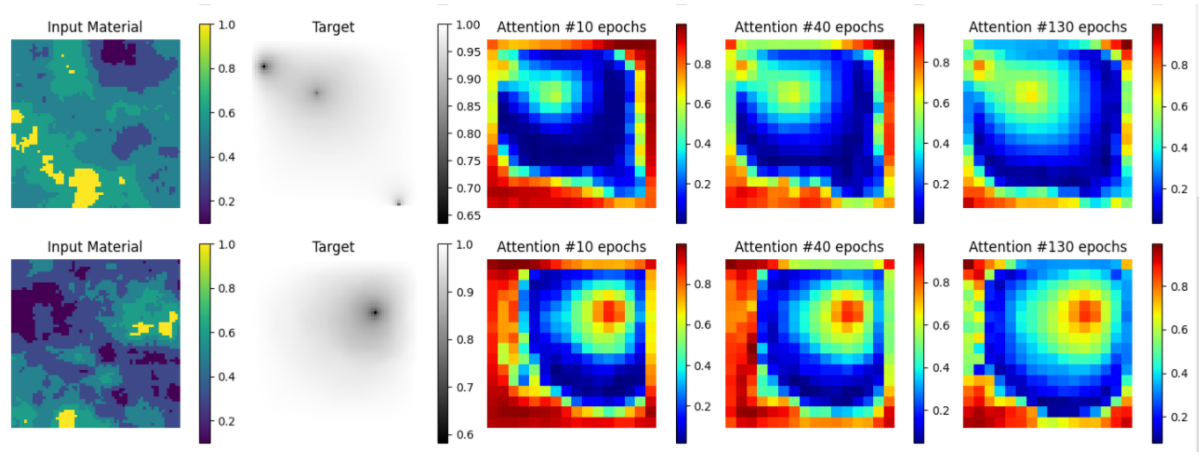


Figure 2.5: Example of attention coefficients learnt by the Attention U-Net network across different training epochs (10, 40, 130) for two random samples of the test dataset.

Appendix 3.B shows the 5 predictions with highest and lowest mean square error between the generated samples and training data out of 500 random samples from the test dataset. The errors are localized near the wells and the difference between the generated and target images is almost negligible even for the samples with the highest error.

2.4.2 Model evaluation

To test the performance of the model, its computational power is compared with the MODFLOW engine. Table 2.1 presents the processing time required for running the forward operators averaged on 10 examples. In order to have a fair comparison between the two, the tests are performed on the same hardware. The CPU used is Intel(R) Xeon(R) CPU @ 2.20GHz and the GPU is Tesla K80. The results demonstrate that Attention U-Net requires less computational power than MODFLOW. This experiment reveals a 75% computational reduction for the data-driven model, showing its capability to be used in forward simulations with less computational demand than the state-of-the-art numerical solver. When applying the method to computationally more expensive forward models, such as in large-scale non-linear system, the computational cost of the neural network will remain low and significant computational savings can be expected.

Table 2.1: Wall-clock Time Comparison

Method	**Hardware**	**Backend**	**Wall-clock time (s)**
Finite Difference	Intel(R) Xeon(R) CPU @ 2.20GHz, GPU Tesla K80	MODFLOW FloPy	0.184
Attention U-Net	Intel(R) Xeon(R) CPU @ 2.20GHz, GPU Tesla K80	Tensorflow	0.046

Dropout at inference time can be considered equivalent to Bayesian approximation in deep Gaussian processes and the neural network uncertainty can be quantified following the approach proposed by Gal and Ghahramani [10]. At test time, the same input is passed 1000 times to the network with random dropout; the mean and the standard deviation of the generated images give an estimation of the prediction interval. Figure 2.6 presents the results for 3 random samples: the uncertainty is null at the boundaries and highest in the vicinity of the wells, which is also the region with highest errors. Compared to the finite difference solver, whose response is deterministic, this method allows one to estimate the uncertainty of the model.

The generalization capabilities of the network are presented in AppendiX 2.C. The model is able to extrapolate to out-of-distribution inputs, especially for different values of hydraulic conductivity and less so for increasing number of wells.

It is worth pointing out that the effect of using attention gates on the uncertainty and generalization capabilities of the model has not been addressed in the current study. Future work should investigate this relation and consequently explore how generalization on out-of-distribution input samples can be improved.

2.5 Conclusion and Future Work

This paper presents a convolutional encoder-decoder network to quickly calculate the steady-state response of a groundwater system. The data-driven surrogate model is trained and tested in different scenarios in which the groundwater head values in the whole domain need to be inferred from the hydraulic head at the locations of the wells. The square domain is a Gaussian random field with a spatially varying hydraulic conductivity. When trained by minimizing

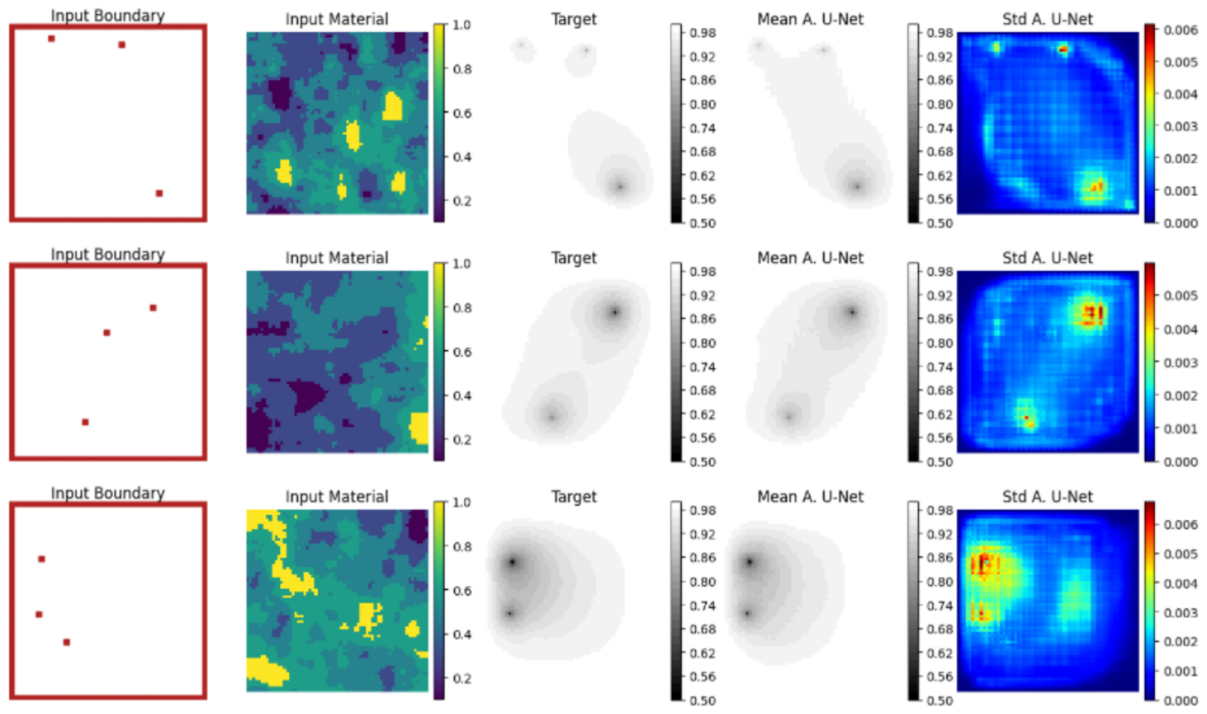


Figure 2.6: Model uncertainty: estimate of output mean and standard deviation of the surrogate model for three randomly selected samples from test dataset. From left to right: the location of the input boundaries; the input spatially varying hydraulic conductivity; simulated output obtained with MODFLOW; estimate mean of 1,000 predicted output with the Attention U-net surrogate; estimate of output standard deviation obtained with the data-driven surrogate model.

the departure from the target images, the proposed U-Net model easily learns the nonlinear relation between inputs (hydraulic conductivity fields and boundary conditions) and output (the hydraulic head field).

A significant contribution of the proposed framework is to incorporate attention gates, which allow the network to identify and focus on the salient regions of the image. The visualization of the attention coefficients demonstrate that the model has learnt to pay attention to areas with large head distribution gradients. The attention mechanism improves the network's approximation accuracy and reduces the model uncertainty. The application of the data-driven surrogate method in solving forward simulations gives very accurate results but requiring much less computational time than the state-of-the-art numerical solver.

One attractive property of this methodology is that the learning is carried out offline. Training converged after less than 3 hours on an Intel(R) Xeon(R) GPU Tesla K80, which can be considered as a low training time compared to typical deep learning models. Once the model is trained, its weights and parameters do not need to be further tuned. The choice of the hyperparameters and the specificities of the U-Net architecture have been chosen based upon manual variation (as opposed to systematic optimization) to give accurate results with low computational time with little apparent sensitivity. Future work could include a more robust hyperparameter tuning study, with a quantitative sensitivity analysis.

In the current study, only Dirichlet boundary conditions were applied to the borders of the domain and the locations of the wells. An additional natural extension of our work is to investigate how well the model generalizes to different and mixed types of boundary conditions. Discretization is another important factor to consider. The present work has been limited to data samples with the same resolution. Many questions remain open related to the discretization of the sample data: e.g. the generalization of the trained model to different discretizations and the amount of training data required if the model needs to be retrained for different resolutions.

The authors plan to further develop the presented model for more complex, larger and uncertain systems. This could include time dependent problems, three dimensional simulations and coupled transport through porous media – all of which are likely to require larger training data sets and potentially deeper networks. Another potential extension is the incorporation of prior information directly into the learning process by imposing a physics constraint in the loss function. Physics-informed learning could increase the speed of inference while requiring less data for the training process. Finally, in this study the network has been trained using synthetic data but the potential use of the proposed model holds promises for the solution of practical applications due to its data-driven nature.

2.A Appendix A: Network architecture

This appendix discusses details in the models used. Both U-Net and Attention U-Net have 4 downsampling layers and 4 upsampling layers (Table 2.2). Each layer consists of a series of CNNs with kernels of size 4, stride of 2 and padding set to 1, followed by a Batch Normalization layer and Dropout with rate 0.50. The nonlinear activation is Leaky ReLU with slope 0.3 for the downsampling layers and ReLU for the upsampling ones (Table 2.3 and Table 2.4). Skip connections concatenate the layers in the encoder with corresponding layers in the decoder. The network of Attention U-Net additionally has attention gates which are implemented as according to the work of Oktay et al. [25]. The total number of parameters of the network is 1.36×10^7, of which 7.35×10^6 are for the attention gates.

Table 2.2: Network architecture: internal layers, input and output feature maps, and number of parameters.

Layers	**Input Shape**	**Output Shape**	**Parameters**
Input Layer	(64, 64, 3)	(64, 64, 3)	0
Downsampling*	(64, 64, 3)	(32, 32, 64)	3072
Downsampling	(32, 32, 64)	(16, 16, 128)	131584
Downsampling	(16, 16, 128)	(8, 8, 256)	525312
Downsampling	(8, 8, 256)	(4, 4, 512)	2099200
Attention Gate	[(4, 4, 1024), (8, 8, 256)]	(8, 8, 256)	3412481
Upsampling	(4, 4, 1024)	(8, 8, 256)	2164992
Concatenate - Skip Connection	[(8, 8, 256), (8, 8, 256)]	(8, 8, 512)	0
Attention Gate	[(8, 8, 512), (16, 16, 128)]	(16, 16, 128)	3150337
Upsampling**	(8, 8, 512)	(16, 16, 128)	1066112
Concatenate - Skip Connection	[(16, 16, 128), (16, 16, 128)]	(16, 16, 256)	0
Attention Gate	[(16, 16, 256), (32, 32, 64)]	(32, 32, 64)	788737
Upsampling**	(16, 16, 256)	(32, 32, 64)	266816
Concatenate - Skip Connection	[(32, 32, 64), (32, 32, 64)]	(32, 32, 128)	0
Conv2DTranspose	(32, 32, 128)	(64, 64, 1)	2049
*without Batch Normalization and without Dropout			
**without Dropout			
Total parameters: 13,610,692 Trainable parameters: 13,604,548 Non-trainable parameters: 6,144			

Table 2.3: Second downsampling layer with input (32, 32, 64) with 128 filters of size 4x4, stride equal to 2 and zero padding.

Layers	**Input Shape**	**Output Shape**
Conv2D	(32, 32, 64)	(16, 16, 128)
Batch Normalization	(16, 16, 128)	(16, 16, 128)
Dropout	(16, 16, 128)	(16, 16, 128)
LeakyReLU	(16, 16, 128)	(16, 16, 128)

Table 2.4: Last upsampling layer with input (16, 16, 256) with 64 filters of size 4x4, stride equal to 2 and padding set equal to 1.

Layers	**Input Shape**	**Output Shape**
UpSampling2D	(16, 16, 256)	(64, 64, 256)
Conv2D	(64, 64, 256)	(32, 32, 64)
Batch Normalization	(32, 32, 64)	(32, 32, 64)
Dropout	(32, 32, 64)	(32, 32, 64)
ReLU	(32, 32, 64)	(32, 32, 64)

2.B Appendix B: Worst and best Model

This appendix shows the 5 predictions with highest and lowest mean square error out of 500 random samples from the test dataset. The samples with highest errors present multiple wells with wide plumes which cover most of the domains (Figure 2.7); on the contrary, the best predictions are those in which the salient region is limited (Figure 2.8). In all cases, highest errors are localized near the wells. It is worth noticing that even when the error is higher, the MSE is in the order of 10^{-5} and the difference between the generated and target images is almost negligible.

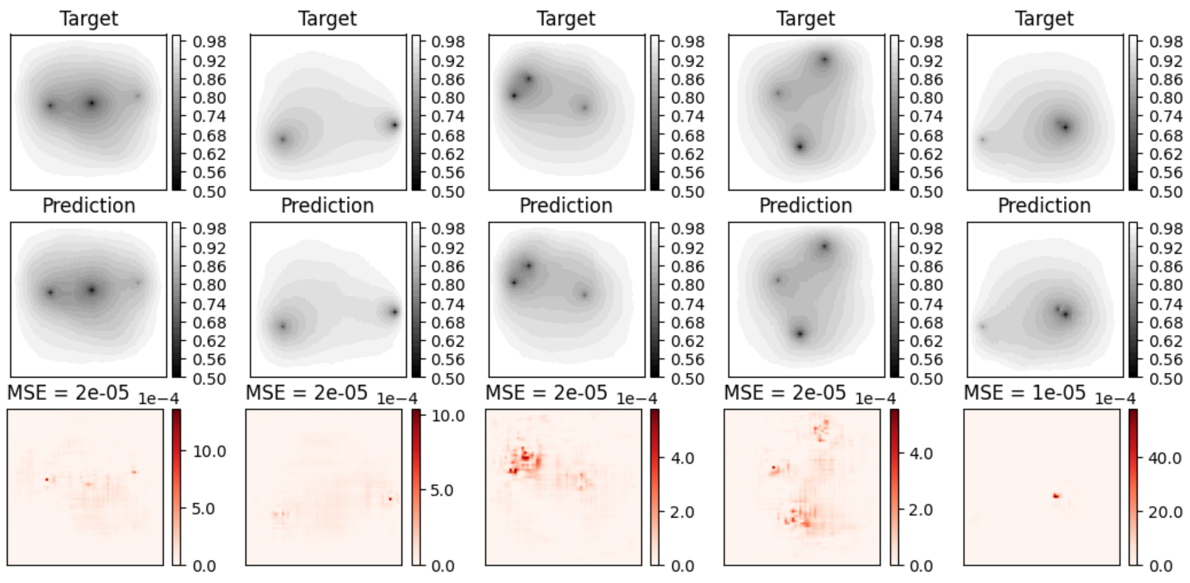


Figure 2.7: Worst 5 predictions out of 500 test samples (highest MSE values)

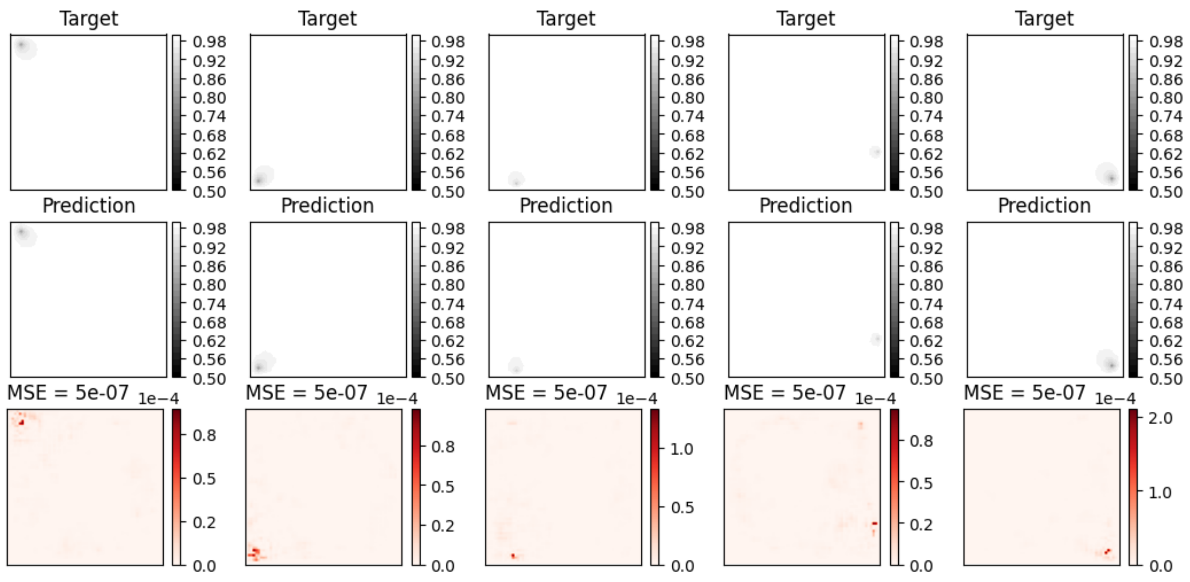


Figure 2.8: Best 5 predictions out of 500 test samples (lowest MSE values)

2.C Appendix C: Generalization

So far the model has been both trained and tested for different scenarios in which the groundwater head values in the whole domain is inferred from the piezometric head at the locations of up to three wells and the spatially varying hydraulic conductivity has values belonging to 5 different classes between 0 and 1. Here we consider testing the model on cases that have different numbers of well locations and different values for the hydraulic conductivity between 0 and 1. Figure 2.9) shows the MSE error for the model tested on four new input distributions. The figure shows that the model is able to generalize well given different values of the hydraulic conductivity, but less so for increasing number of wells.

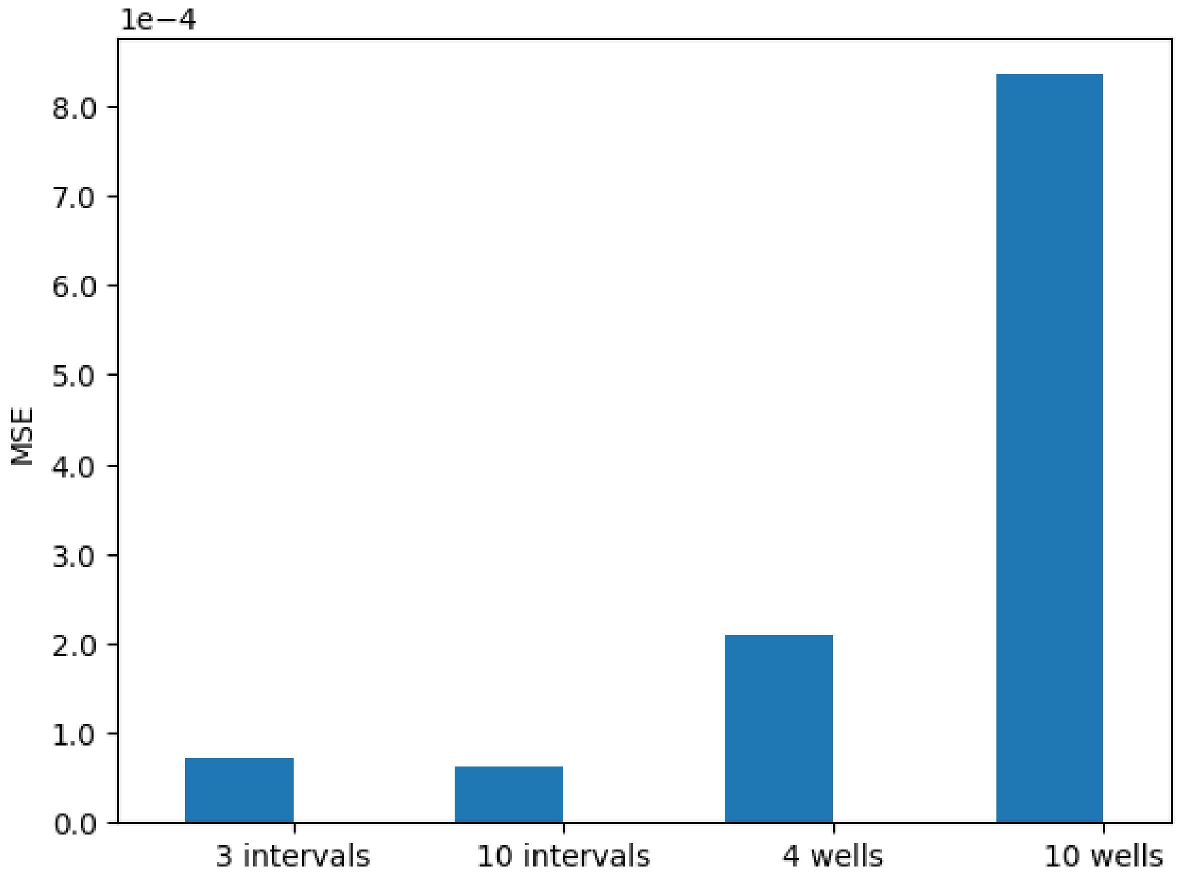


Figure 2.9: Generalization to new input distributions. MSE error when the model is evaluated with hydraulic conductivity having 3 values; hydraulic conductivity values belonging to 10 intervals; 4 wells in random locations the domain; 10 wells in random locations the domain. Each test set contains 1000 samples.

References

[1] S. Atkinson and N. Zabaras. “Structured Bayesian Gaussian process latent variable model: Applications to data-driven dimensionality reduction and high-dimensional inversion”. In: *Journal of Computational Physics* 383 (2019), pp. 166–195.

[2] J. Bear and A. Verruijt. *Modeling Groundwater Flow and Pollution.* Hingham, MA: Kluwer Boston Inc., 1987.

[3] M. Bojarski et al. “End to end learning for self-driving cars”. In: (2016).

[4] S. Chaturantabut and D.C. Sorensen. “Nonlinear model reduction via discrete empirical interpolation”. In: *SIAM Journal on Scientific Computing* 32 (2010), pp. 2737–2764.

[5] S Chaudhari et al. “An attentive survey of attention models”. In: *ACM Transactions on Intelligent Systems and Technology* 12 (2021), pp. 1–32.

[6] V. Christelis, R.G. Regis, and A. Mantoglou. “Surrogate-based pumping optimization of coastal aquifers under limited computational budgets”. In: *Journal of Hydrology* 20 (2018), pp. 164–176.

[7] D. Crevillén-García et al. “Uncertainty quantification for flow and transport in highly heterogeneous porous media based on simultaneous stochastic model dimensionality reduction”. In: *Transport in Porous Media* 126 (2019), pp. 79–95.

[8] Y. Dagasan, P. Juda, and P. Renard. “Using Generative Adversarial Networks as a Fast Forward Operator for Hydrogeological Inverse Problems”. In: *Groundwater* 58 (2020). DOI: 10.1111/gwat.13005.

[9] D. Eigen, C. Puhrsch, and R. Fergus. *Depth map prediction from a single image using a multi-scale deep network.* 2014.

[10] Y Gal and Z Ghahramani. “Dropout as a bayesian approximation: Representing model uncertainty in deep learning”. In: *International conference on machine learning.* PMLR. 2016, pp. 1050–1059.

[11] A Galassi, M Lippi, and P Torroni. “Attention in natural language processing”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[12] N. Geneva and N. Zabaras. “Modeling the dynamics of PDE systems with physics-constrained deep auto-regressive networks”. In: *Journal of Computational Physics* 403 (2020), p. 109056.

[13] I Goodfellow, Y Bengio, and A Courville. *Deep learning.* MIT press, 2016.

[14] K. He et al. “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”. In: *Proceedings of the IEEE international conference on computer vision.* 2015, pp. 1026–1034.

[15] C. Huang, K. Duraisamy, and C. Merkle. “Challenges in reduced order modeling of reacting flows”. In: *2018 Joint Propulsion Conference.* 2018, p. 4675.

[16] P Isola et al. “Image-to-image translation with conditional adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2017, pp. 1125–1134.

[17] G.E. Karniadakis et al. “Physics-informed machine learning”. In: *Nature Reviews Physics* 3 (2021), pp. 422–440. DOI: 10.1038/s42254-021-00314-5.

[18] M.C. Kennedy and A. O’Hagan. “Predicting the output from a complex computer code when fast approximations are available”. In: *Biometrika* 87 (2000), pp. 1–13.

[19] S Khan et al. “Transformers in vision: A survey”. In: (2021).

[20] DP Kingma and J Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[21] A Krizhevsky, I Sutskever, and GE Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems.* Vol. 25. 2012, pp. 1097–1105.

[22] T. Lassila et al. “Model order reduction in fluid dynamics: challenges and perspectives”. In: *Model Reduction of Parametrized Systems* (2014), pp. 235–273.

[23] Y LeCun et al. “Handwritten digit recognition with a back-propagation network”. In: *Advances in neural information processing systems.* Vol. 2. 1989.

[24] S Mo et al. “Deep convolutional encoder-decoder networks for uncertainty quantification of dynamic multiphase flow in heterogeneous media”. In: *Water Resources Research* 55 (2019), pp. 703–728.

[25] O. Oktay et al. *Attention u-net: Learning where to look for the pancreas.* 2018.

[26] T. Olsthoorn. *Finite Difference Groundwater Modeling in Python.* 2016.

[27] D. Pulido-Velazquez et al. “A general methodology to simulate groundwater flow of unconfined aquifers with a reduced computational cost”. In: *Journal of Hydrology* 338.1 (2007), pp. 42–56.

[28] O. Ronneberger, P. Fischer, and T. Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[29] DE Rumelhart, GE Hinton, and RJ Williams. “Learning internal representations by error propagation”. In: *California Univ San Diego La Jolla Inst for Cognitive Science* (1985).

[30] T Salimans and DP Kingma. “Weight normalization: A simple reparameterization to accelerate training of deep neural networks”. In: *Advances in neural information processing systems*. Vol. 29. 2016, pp. 901–909.

[31] J Schlemper et al. “Attention gated networks: Learning to leverage salient regions in medical images”. In: *Medical Image Analysis* 53 (2019), pp. 197–207.

[32] A.J. Siade et al. “Reduced-dimensional Gaussian process machine learning for groundwater allocation planning using swarm theory”. In: 56.e2019WR026061 (2020).

[33] D. Silver et al. “Mastering the game of Go with deep neural networks and tree search”. In: *Nature* 529.7587 (2016), pp. 484–489.

[34] A.Y. Sun. “Discovering state-parameter mappings in subsurface models using generative adversarial networks”. In: *Geophysical Research Letters* 45 (2018), pp. 11137–11146.

[35] C Szegedy et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.

[36] DK Todd. “Seepage and groundwater flow, numerical analysis by analog and digital methods, K. R. Rushton and S. C. Redshaw, Wiley, New York, 1979. No. of pages: 339”. In: *Earth Surface Processes and Landforms* 5 (1980), p. 399. DOI: 10.1002/esp.3760050409.

[37] E Vanmarcke. *Random fields: analysis and synthesis*. The MIT Press, Cambridge, Mass, 1983.

[38] Y. Wu et al. *Google’s neural machine translation system: Bridging the gap between human and machine translation*. 2016.

[39] X. Yu et al. “Deep learning emulators for groundwater contaminant transport modelling”. In: *Journal of Hydrology* 590 (2020), p. 125351.

[40] Z. Zhong, A.Y. Sun, and H. Jeong. “Predicting co2 plume migration in heterogeneous formations using conditional deep convolutional generative adversarial network”. In: *Water Resources Research* 55 (2019), pp. 5830–5851.

[41] Y Zhu and N Zabaras. “Bayesian Deep Convolutional Encoder-Decoder Networks for Surrogate Modeling and Uncertainty Quantification”. In: *arXiv preprint arXiv:1801.06879* (2018).

Chapter 3

Understanding the Efficacy of U-Net & Vision Transformer for Groundwater Numerical Modelling

Abstract

This paper presents a comprehensive comparison of various machine learning models, namely U-Net, U-Net integrated with Vision Transformers (ViT), and Fourier Neural Operator (FNO), for time-dependent forward modelling in groundwater systems. Through testing on synthetic datasets, it is demonstrated that U-Net and U-Net + ViT models outperform FNO in accuracy and efficiency, especially in sparse data scenarios. These findings underscore the potential of U-Net-based models for groundwater modelling in real-world applications where data scarcity is prevalent.

3.1 Introduction

Groundwater numerical models, such as MODFLOW [8], are crucial for water resource management, although they are computationally demanding. To alleviate this, surrogate modelling through data-driven methods offers efficient approximations of these complex numerical techniques.

Neural Operators [12, 15], particularly the Fourier Neural Operator (FNO) [11], have been at the forefront of recent advances, having shown potential to approximate arbitrary continuous functions. However, the computational demand of FNO is particularly high during training phase while these neural operators require architectural enhancements to deliver promising results in subsurface problems [17, 9]. This is evident in the work of Wen et al. [17], where the integration of FNO with U-Net architecture showed improved accuracy, speed, and data efficiency in multiphase flow problems. However, Gupta and Brandstetter's work [6], showing that U-Net outperforms FNOs across various fluid mechanics problems, raises a question about the necessity of neural operators when the vanilla U-Net architecture already exhibits remarkable performance.

Recently, transformers [16] have seen considerable success in various fields, including physical systems [2, 10], for which the datasets are typically smaller compared to other domains. Only one study explores the use of transformers in groundwater modeling [13], demonstrating that the models were outperformed by both GRU and LSTM models to predict groundwater levels across various stations in France with meteorological and hydrological data.

Finally, the integration of U-Net with Transformers, as exemplified in studies like TransUNet [3] and ViTO [14], has demonstrated their utility across a broad range of applications, particularly in the field of medical image segmentation and operator learning for inverse PDE problems. Yet, the applicability of these combinations in addressing time-dependent forward problems, real-world data scenarios, and in situations with sparse data, remain areas yet to be fully explored.

Several studies, such as the one by Brakenhoff et al. [1, 13], primarily focus on individual time series when analysing the impact of various hydrological stressors, including pumping rates, precipitation excess, and river stage variations, on groundwater levels of individual monitoring wells. While this approach provides valuable insights, it does not account for spatial correlations, thereby limiting its use to existing time series or monitoring wells. Similarly, previous comparisons have been predominantly limited to specific models like LSTM, CNNs and NARX in the context of groundwater level forecasting [18], leaving room for broader explorations.

In this paper, we present a comprehensive comparison among models—specifically U-Net, U-Net integrated with Vision Transformers (U-Net+ViT), and Fourier Neural Operator (FNO)—for their efficacy in modeling time-dependent forward and inverse problems in groundwater systems. We test our model extensively on synthetic datasets, simulating conditions from the Overbetuwe region in the Netherlands, including sparse data scenarios. We show that both U-Net and U-Net+ViT are particularly well-suited to these important sparse data scenarios, with the addition of the Transformer providing enhanced predictive capability in many cases.

3.2 Methodology

3.2.1 Example of study site and data

This subsection provides context and rationale for our study via an example case study based upon the polder region of Overbetuwe in the Netherlands (Figure 3.1). This region showcases the characteristic Dutch system of water management where the area is divided into several polders in a mix of agriculture, nature, and urban environments. Alongside its sparse data and heterogeneous soil, these unique characteristics underscore the inherent complexities of water management in similar settings, making this dataset a suitable choice for our research. The subsoil is primarily composed of clay and sandy clay, with soil properties being determined via borehole and cone penetration tests. The study area features numerous observation wells for monitoring groundwater heads while well fields (indicated as groundwater usage facilities in the figure) are utilized for the extraction of drinking water. The work of Brakenhoff et al. [1] considers a dataset consisting of 250 head time series, with daily recordings starting from the year 1990 and drawdown attributed to the extraction from up to four well fields.

For the purposes of this study, we employ synthetic data to validate the proposed methodology, with the intention to subsequently apply the validated method to the real-world data of the Overbetuwe region. Figure 3.2 represents a sample of the high-fidelity labeled dataset, which is constructed using the U.S. Geological Survey (USGS) finite-difference flow model, MODFLOW. The model is composed of a single-layer representation of a confined aquifer with a 128×128 grid. The aquifer's heterogeneity is reflected through varying horizontal hydraulic conductivity within the bounds $k \in [0.1, 0.5]$ m/d. The hydraulic conductivity fields in our study are created using random fields which are then thresholded to delineate different classes. A maximum of ten pumping wells are extracting water with variable rates in the range $Q \in [0, 30]$ m^3/d over a simulation period of $T = 10$ days. The pumping wells are located in random locations which

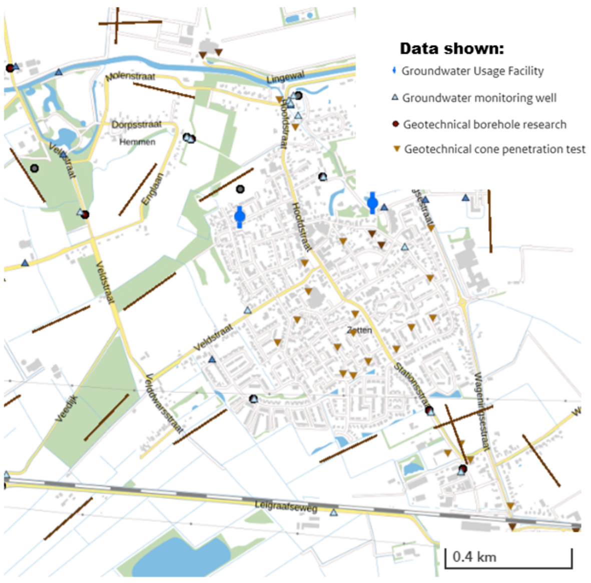


Figure 3.1: A representation of the target study area located in the polder region of Overbetuwe, Netherlands. [4]

vary for each sample. The boundary conditions are delineated as Dirichlet, with the head equal to zero, mimicking a polder encircled by ditches where a stable water level is maintained through a comprehensive network of pumping stations.

The datasets consist of $N_{train} = 5000$ training instances and $N_{test} = 1000$ testing instances. To mirror the inherent sparsity of real-world data, a data selection strategy is adopted for the test dataset. The locations of the boreholes for estimating the hydraulic conductivity are chosen following a radial distribution pattern, and a helical pattern is used for the wells monitoring hydraulic head (Figure 3.2).

3.2.2 Architectures

The architectures of the three models under comparison in this study encompass the U-Net structure, a U-Net with attention mechanism in the bottleneck, and the Fourier neural operator (FNO).

The U-Net architecture is designed with an encoder-decoder structure where the decoder receives the upsampled feature map, which is then concatenated with the corresponding feature map from the encoder through a skip connection. Detailed diagrams of the U-Net encoder and decoder can be found in Figures 3.6 and 3.7 in Appendix 3.A. The encoder consists of three bottleneck blocks, where each block utilizes three layers of Conv2d, Instance Normalization,

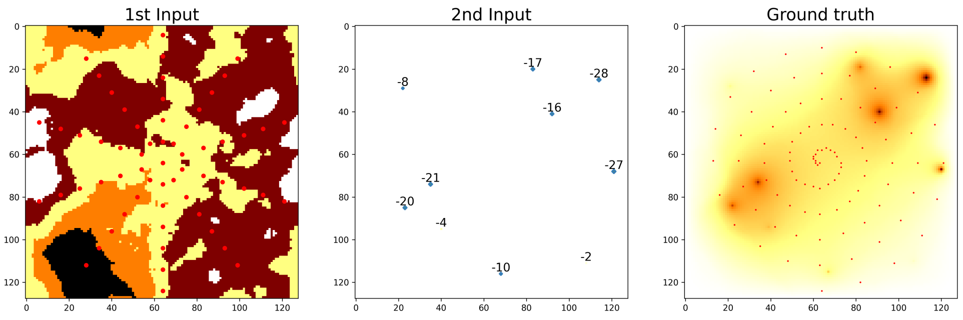


Figure 3.2: An test sample of the high-fidelity dataset constructed using MODFLOW. The first column showcases the heterogeneous hydraulic conductivity, the second column presents the randomly positioned pumping wells, and the third column depicts the resulting hydraulic head. The red dots (column 1 and 3) represent a selection of data points—following radial and helical patterns, respectively—emulating the sparse observations in real-world scenarios.

and GELU activation to extract spatial features. These blocks increase the number of channels by a factor of 2 and perform downsampling with a stride of 2. The decoder is composed of a series of upsampling blocks, where each block consists of a bilinear upsampling operation (Upsample), followed by a double convolution operation. Each convolution within the decoder is followed by Instance Normalization and GELU activation function. The bottleneck consists on a single convolutional layer. In the time-dependent scenario, the time series data of the historical pumping rates is processed through two layers of feed-forward neural network (FNN) prior to being concatenated to the input for the latent space representation (Figure 3.6).

The second model, here called UNet+ViT, employs the Vision Transformer (ViT) [5], in the latent space representation of the U-Net. This implementation is based on the methods used in TransUNet [3] and ViTO [14]. The input is tokenized into a sequence of flattened 2D patches, each of size 1×1. Positional information is retained by employing trainable convolutional projection to learn and add specific position embeddings to the patch embeddings. The structure of the Transformer includes L blocks, with each block comprising Multi-Head Attention (MSA) and FNN. This configuration involves the use of 2 blocks, each with 2 Multihead Self-Attentions, and a FNN composed of 128 neurons. For a more detailed visualization of the Vision Transformer, attention block, and multihead attention, please refer to Appendix 3.A, Figure 3.8.

The Fourier neural operator (FNO) [11] model leverages the fast Fourier Transform to parameterize the integral kernel directly in the Fourier space. The implementation of FNO for the

$2D$ Darcy Flow problem as presented in [11] is followed in this study. The total amount of parameters of FNO corresponds to 2.38 million, that is 15 times more than UNet+ViT (151k) and 17 times more than UNet (137k).

3.3 Results

3.3.1 Forward problem with sparse observations

This section presents the prediction of the hydraulic head at sparse monitoring wells after a constant 10-day pumping period under two different training conditions. We employ distinct sampling strategies for both input and output data in our methodology. Our training data is sampled from a regular quadratic grid, while for testing we have explored other arrangements, such as radial and helical, to understand their potential impact on the prediction performance. In the first scenario, training is conducted using sparse data, with a spacing of 20 grid points for the input hydraulic conductivity field and a spacing of 8 for the output hydraulic head. Testing is then carried out on sparse data points, following the radial and helical patterns delineated in subsection 3.2.1. The resulting root mean square error (RMSE) is found to be 5.2×10^{-2}, 3.5×10^{-2} and 8.1×10^{-2} for the vanilla U-Net, the UNet+ViT models and FNO respectively. These results underline the superior performance of the UNet+ViT model in handling sparse data, exhibiting a lower RMSE compared to both the vanilla U-Net and the FNO models.

In contrast, when training is performed using the entire field and testing on the same sparse dataset, the error marginally escalates to 3.9×10^{-1} for FNO, 3.8×10^{-1} for UNet and 3.6×10^{-1} for UNet+ViT model. This outcome is anticipated considering the training set exhibits sparsity in the first scenario, but not in the latter. Additionally, Figure 3.3 displays the prediction over the entire domain, resulting in a lower RMSE of 1.0×10^{-2} for FNO, 1.7×10^{-2} and 1.9×10^{-2} for the vanilla U-Net and UNet+ViT models, respectively. The FNO model, while superior when dealing with full data, exhibits the highest predictive error under sparse data observations. These results highlight the practical advantages of the U-Net and especially UNet+ViT model in real-world scenarios for which data sparsity is common.

It should be noted that traditional simpler neural networks and other machine learning techniques may not provide adequate solutions for this specific problem. This assertion is backed by a comparison of the results from a fully connected neural network, a linear regression model and a random forest, detailed in Appendix 3.B. Despite the substantial number of trainable parameters, reaching 51.17 million, inherent to the fully connected neural network and the application of linear regression and random forest, these methods significantly underperform compared to the U-Net, the UNet+ViT models, and FNO.

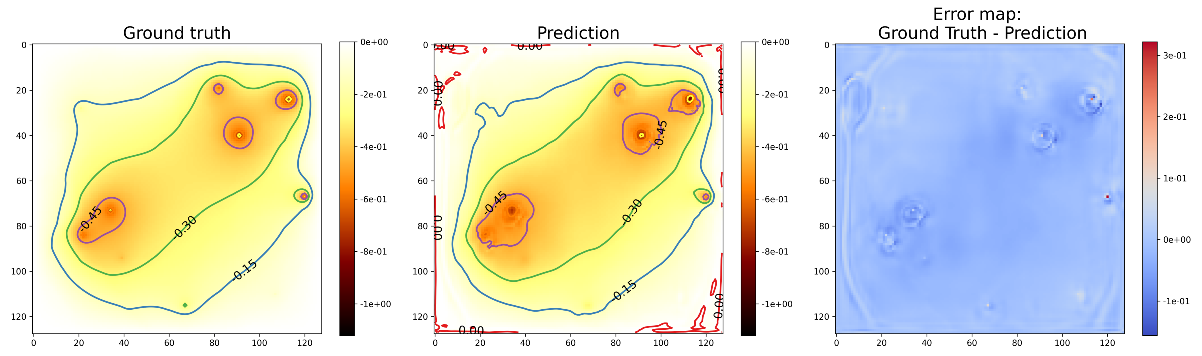


Figure 3.3: Predictions over the entire domain are shown, with the UNet+ViT model exhibiting a RMSE of 1.9×10^{-2}. The figure demonstrates the model's ability to accurately capture the spatial distribution of the hydraulic head across the entire domain.

3.3.2 Identification of pumping wells

In this section, we focus on an inverse problem: specifically the identification of pumping wells. This task requires determining the locations and rates of pumping wells based on the observed hydraulic heads. Throughout these experiments, we employ a single hydraulic conductivity field, which, while spatially varying, remains identical across all samples within the dataset.

In evaluating the performance of our models, we use both RMSE and accuracy. The RMSE calculates the average difference between the true and the predicted value for each pump location in the test dataset, giving a quantitative measure of the prediction error. Complementing this, the accuracy was determined by counting the proportion of correct pump predictions, where a prediction is considered correct if the predicted and actual pump locations align. This gives a sense of how often the model correctly identifies the location of pumps.

The U-Net model performs optimally, achieving an RMSE of 5.6×10^{-2}. Interestingly, the integration of the Vision Transformer with the U-Net model does not confer any additional precision in this scenario, yielding a near RMSE of 6.1×10^{-2}. The FNO model exhibits a higher RMSE of 1.1×10^{-1}, indicating a somewhat lower accuracy in identifying the pumping well locations.

To visually illustrate these results, Figure 3.4 presents a test sample using the U-Net + ViT model. It demonstrates an accuracy of 93% in locating the pumps, calculated across the entire test dataset. The figure visualizes the model's ability to accurately identify the positions and the pumping rate of the wells. In comparison, the FNO model achieved a notably lower detection accuracy of 79% in the same task.

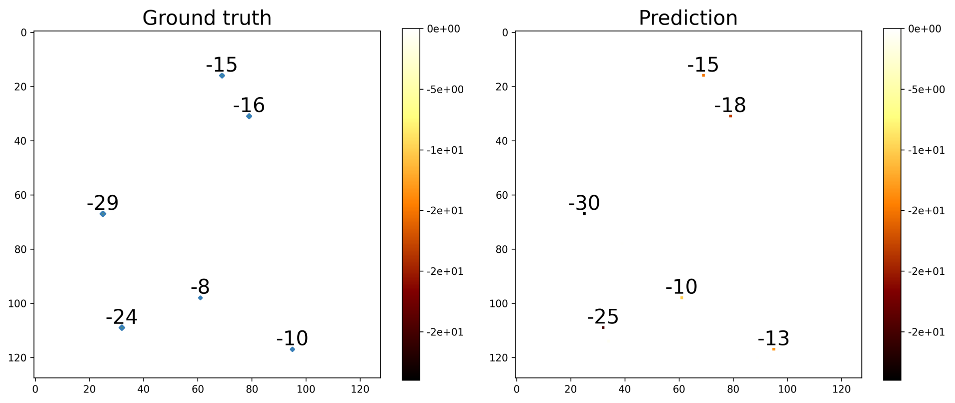


Figure 3.4: Test sample results for the identification of pumping wells using the U-Net + ViT model. The model accurately identifies both the positions of the pumping wells with an accuracy of 93 % across the entire test dataset.

3.3.3 Example results for time series data

This section unveils the results achieved from the analysis of time series data, starting with a simplified scenario, for which the inputs are the varying hydraulic conductivity field and the pumping rate of a single pump which varies over a 10-day simulation period. Results are evaluated in terms of root mean square error (RMSE) with a focus on the comparison of different configurations of the U-Net architecture with transformers. Figure 3.5 presents a comparison of results over 5 time frames for the U-Net with the Vision Transformer under autoregressive testing conditions.

The RMSE for each method was calculated to quantify the models' performance. The U-Net architecture alone yielded an RMSE of 1.79×10^{-2}. When supplemented with a Vision Transformer, consisting of 2 attention blocks and 2 heads, the performance improves, registering an RMSE of 1.67×10^{-2}. However, increasing the complexity of the Vision Transformer to 8 blocks and 8 heads did not further improve the performance, instead, it led to a slight degradation in the RMSE (1.77×10^{-2}). Adding an Axial Transformer [7] to the U-Net architecture also did not enhance the performance, yielding an RMSE of 1.83×10^{-2}. These results suggest that

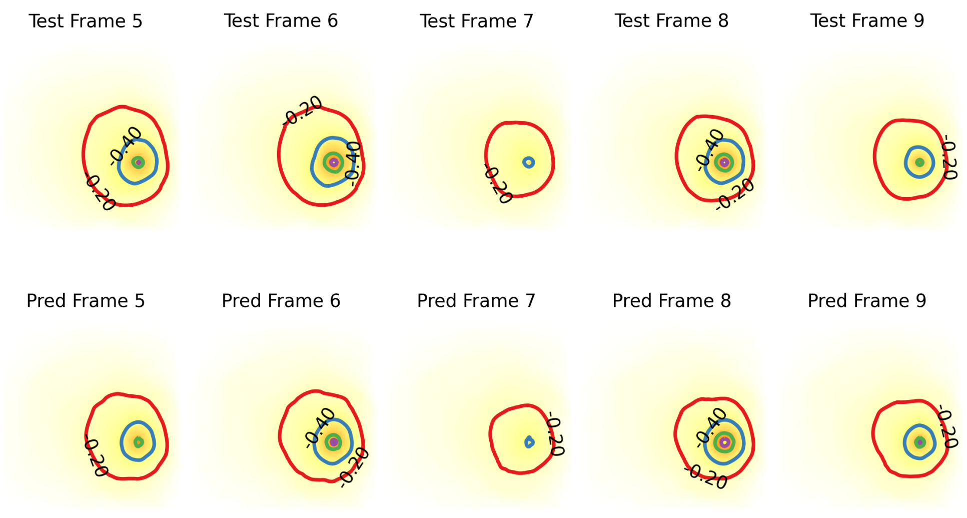


Figure 3.5: Depiction of autoregressive testing results from the UNet+ViT model over the final five frames of a ten-frame test sequence. This sequence simulates a single variable-rate pump operating over a ten-day period within a region characterized by a diverse hydraulic conductivity field. The top row presents the ground truth, while the lower row displays the predicted outcomes. Contour lines in the images represent groundwater levels, which are color-coded for enhanced visual clarity. Each successive frame is employed as input to the U-Net-Vision Transformer system, aiding in the prediction of the subsequent frame.

while adding a Vision Transformer to the U-Net architecture leads to performance improvement, increasing the complexity of the latent space does not necessarily do so.

3.4 Conclusion

This paper explores and evaluates the capabilities of different machine learning models, with a particular focus on U-Net, U-Net integrated with Vision Transformers (ViT), and Fourier Neural Operator (FNO), in the context of predicting hydraulic head in groundwater studies.

Our analysis and testing, conducted on synthetic datasets designed to simulate the conditions from the Overbetuwe region in the Netherlands and including scenarios with sparse data, firmly establish that both U-Net and U-Net + ViT models are particularly adept at dealing with such tasks. Importantly, these models are also preferred due to their fewer requisite parameters.

Specifically, in the case of sparse observation scenarios, the vanilla U-Net and the U-Net + ViT models outperformed the FNO model. In particular the performance of the UNet+ViT model was superior when handling sparse data, highlighting the potential of the model in real-world applications, where data scarcity is a common issue. The U-Net model demonstrated optimal performance in identifying pumping wells. Interestingly, the integration of the Vision Transformer with the U-Net model did not confer any additional accuracy in this scenario. As for the analysis of time series data, supplementing the U-Net architecture with a Vision Transformer improved the model performance, recording an RMSE of 1.67×10^{-2} compare to 1.79×10^{-2} of the vanilla U-Net. However, increasing the complexity of the Vision Transformer did not further enhance the model performance, indicating that a more complex architecture does not necessarily yield better results.

Future research will involve applying this validated methodology to real-world data, beginning with the Overbetuwe region in the Netherlands. This will offer an opportunity to further validate and refine the model, accounting for the sparsity and uncertainties inherent in real-world data.

3.A Appendix A

This appendix provides detailed diagrams of the model structures.

3.B Appendix B

This appendix sets out to examine whether simpler machine learning models, specifically a fully connected neural network, a linear regression model, and a Random Forest model, can achieve the same level of accuracy as more advanced models like the U-Net, the UNet+ViT models, and FNO in predicting groundwater levels.

The particular Random Forest model tested here used 30 estimators. The fully connected neural network, employed for this comparison, comprises three hidden layers, each containing 1000 nodes and using ReLU activation functions. The model holds an impressive count of 51.17 million trainable parameters.

Unfortunately, none of the models was able to predict accurately the groundwater levels neither capturing the location of the wells. Specifically, the fully connected neural network and the linear regression model yielded high RMSEs of 1.17×10^{-1} and 1.24×10^{-1}, respectively. The Random Forest model fared slightly better, achieving a lower RMSE of 1.02×10^{-1}, but it still fell short of the U-Net, the UNet+ViT models, and FNO.

Figure 3.10 visually contrasts the predictions of these simpler models gainst the ground truth.

Their significant underperformance becomes evident when compared to more sophisticated models. For a comparison of these results with accurate outcomes produced by the UNet+ViT model, the reader is directed to Figure 3.3.

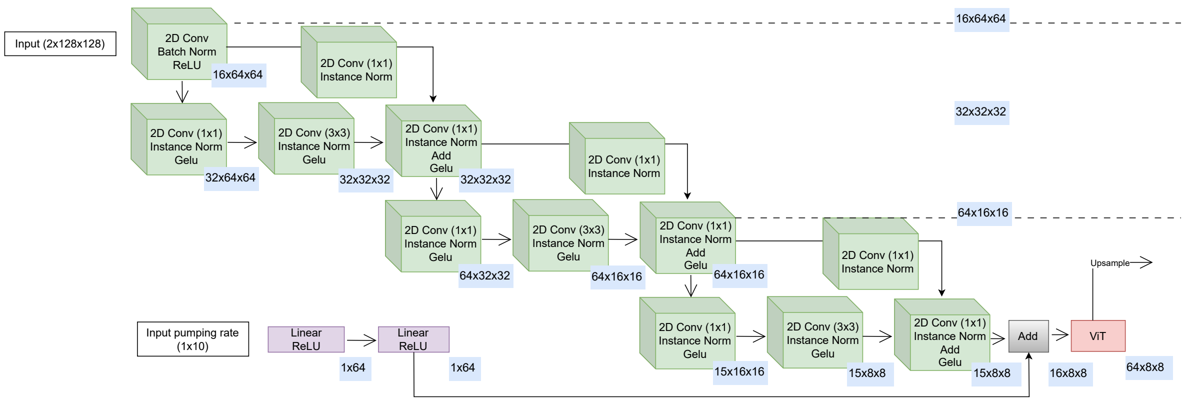


Figure 3.6: U-Net Encoder: The encoder consists of 3 bottleneck blocks, each comprising 3 convolutional blocks (Conv2d, Instance Normalization, and GELU). These blocks increase the channel dimension by a factor of 2 and perform downsampling with a stride of 2, extracting spatial features. In time-dependent scenarios, the pumping rate time series undergoes processing through a two-layer feed-forward neural network. The resulting processed data is then concatenated with the input, creating the latent space representation.

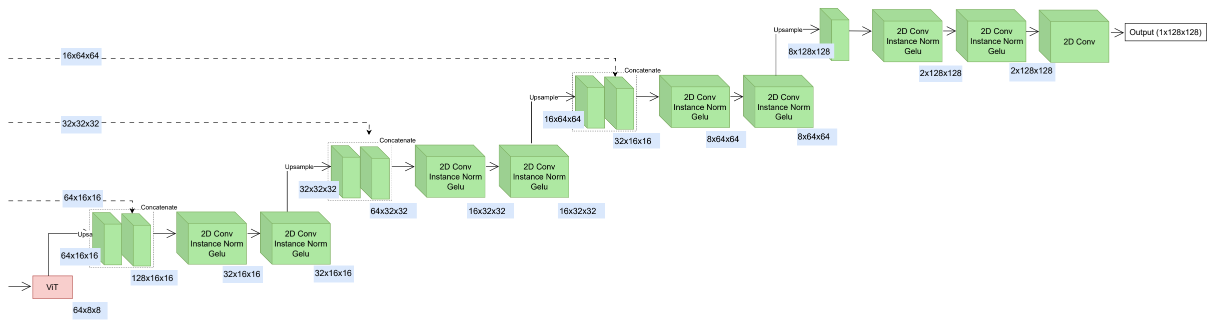


Figure 3.7: U-Net Decoder: The decoder is composed of a series of upsampling blocks, where each block consists of a bilinear upsampling operation followed by a double convolution operation.

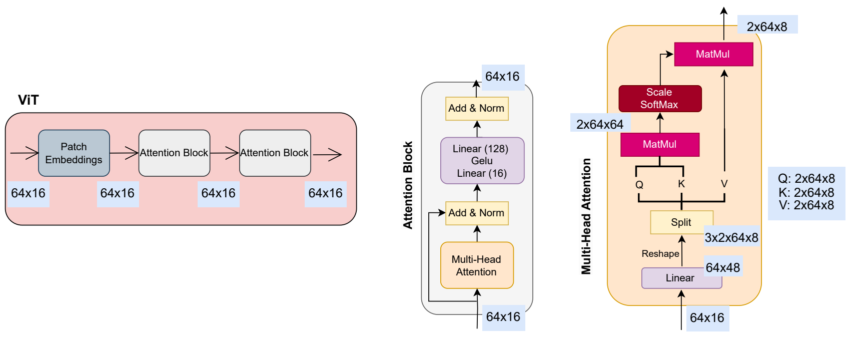


Figure 3.8: The Vision Transformer (ViT) and its components: The input is tokenized into a sequence of flattened 2D patches, each of size 1×1. The structure of the Transformer encoder includes 2 attention blocks, each containing Multi-Head Attention block and fully connected layers.

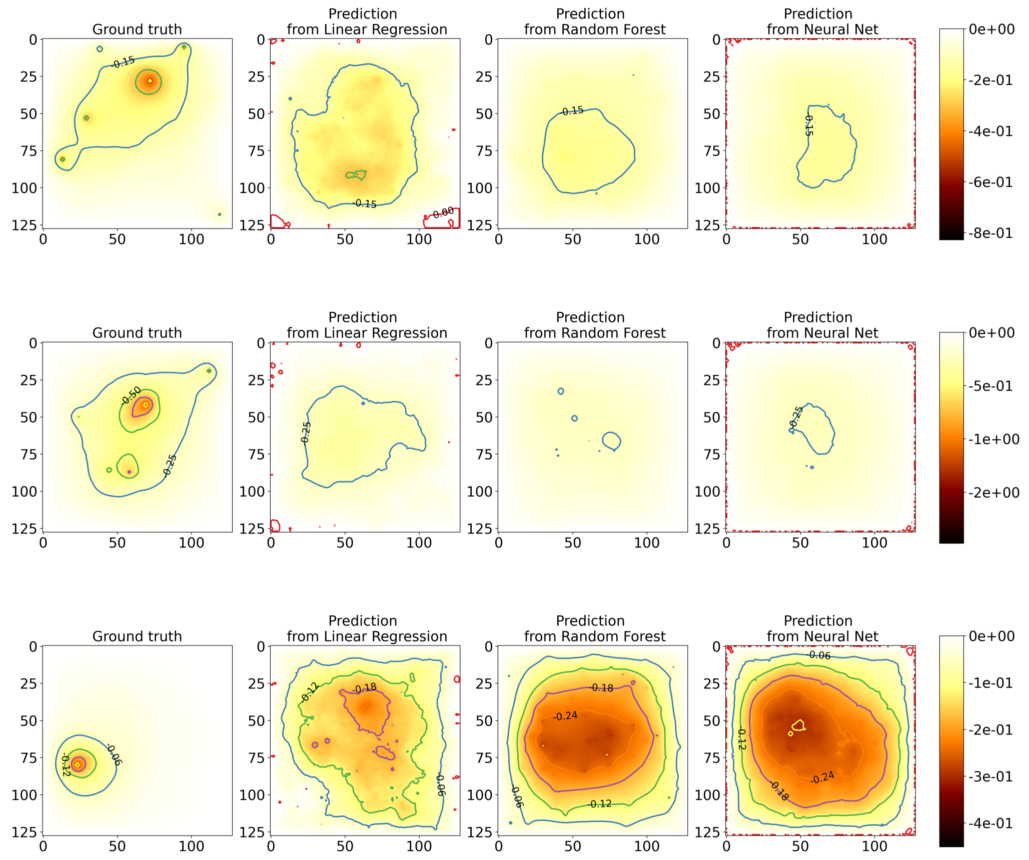


Figure 3.9: These figures illustrate the testing results using a fully connected neural network, a linear regression model, and a Random Forest model. The leftmost column row of each figure presents the ground truth, while the remaining columns display the predicted outcomes generated by each respective model. Contour lines represent groundwater levels, color-coded for clarity.

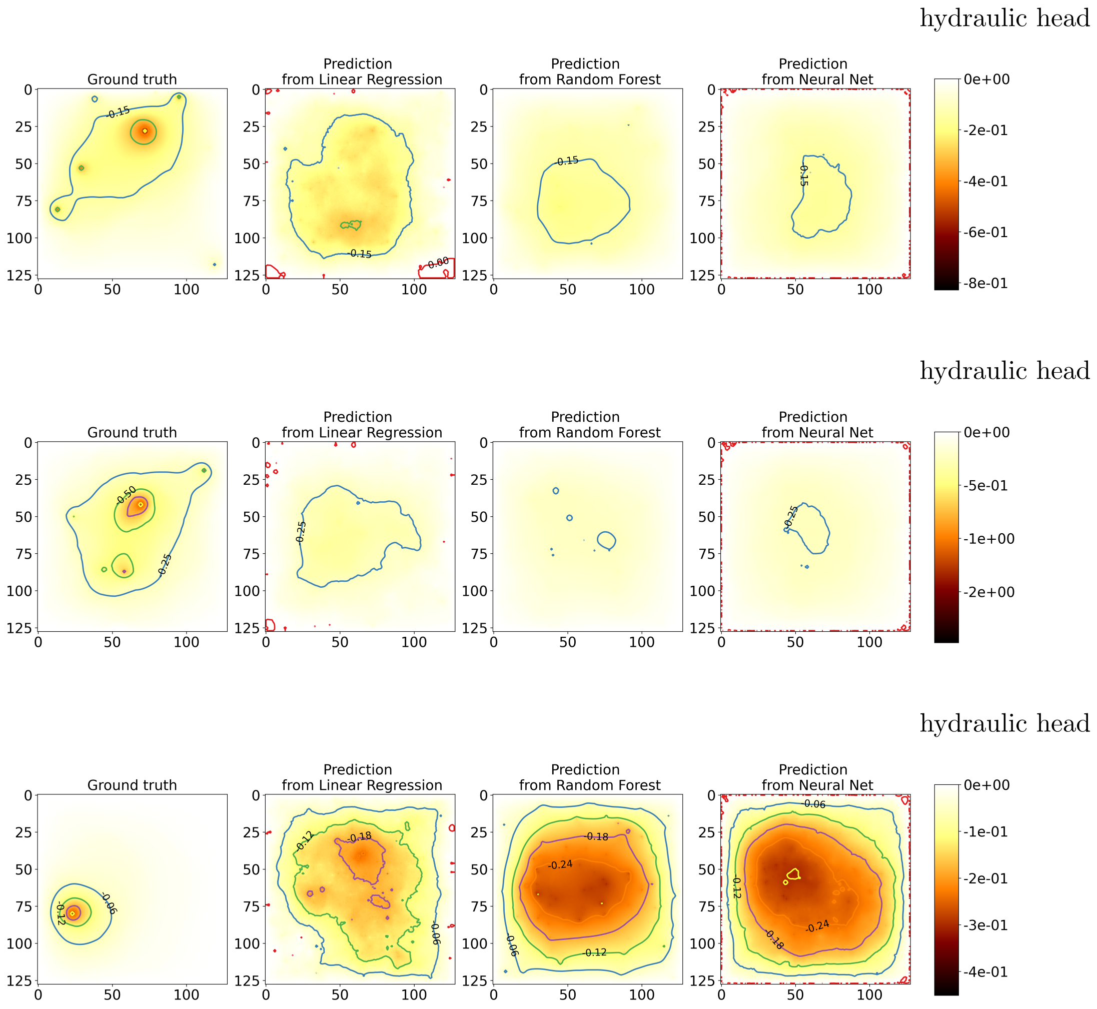


Figure 3.10: These figures illustrate the testing results using a fully connected neural network, a linear regression model, and a Random Forest model. The leftmost column of each figure presents the ground truth, while the remaining columns display the predicted outcomes generated by each respective model. Contour lines represent groundwater levels, color-coded for clarity.

References

[1] D. A. Brakenhoff et al. “Application of Time Series Analysis to Estimate Drawdown From Multiple Well Fields”. In: *Frontiers in Earth Science* 10 (2022).

[2] S. Cao. *Choose a Transformer: Fourier or Galerkin.* `https://arxiv.org/abs/2105.14995`. arXiv preprint arXiv:2105.14995. 2021.

[3] J. Chen et al. *TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation.* `https://arxiv.org/abs/2102.04306`. arXiv preprint arXiv:2102.04306. 2021.

[4] dinoloket. *DINO loket.* `https://www.dinoloket.nl/en/subsurface-data`. 2023.

[5] A. Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.* `https://arxiv.org/abs/2010.11929`. arXiv preprint arXiv:2010.11929. 2021.

[6] J. K. Gupta and J. Brandstetter. *Towards Multi-spatiotemporal-scale Generalized PDE Modeling.* `https://arxiv.org/abs/2209.15616`. arXiv preprint arXiv:2209.15616. 2022.

[7] J. Ho et al. *Axial Attention in Multidimensional Transformers.* `https://arxiv.org/abs/1912.12180`. arXiv preprint arXiv:1912.12180. 2019.

[8] J. D. Hughes et al. “The MODFLOW Application Programming Interface for simulation control and software interoperability”. In: *Environmental Modelling & Software* 148 (2022).

[9] Z. Jiang et al. *Fourier-MIONet: Fourier-enhanced multiple-input neural operators for multiphase modeling of geological carbon sequestration.* `https://arxiv.org/abs/2303.04778`. arXiv preprint arXiv:2303.04778. 2023.

[10] Z. Li, K. Meidani, and A. B. Farimani. *Transformer for Partial Differential Equations' Operator Learning.* `https://arxiv.org/abs/2205.13671`. arXiv preprint arXiv:2205.13671. 2023.

[11] Zongyi Li et al. “Fourier neural operator for parametric partial differential equations”. In: *arXiv preprint arXiv:2010.08895* (2020).

[12] L. Lu et al. *Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators.* 2021.

[13] N. Mellouli, M. L. Rabah, and I. R. Farah. "Transformers-based time series forecasting for piezometric level prediction". In: *2022 IEEE International Conference on Evolving and Adaptive Intelligent Systems (EAIS)*.

[14] O. Ovadia et al. *ViTO: Vision Transformer-Operator*. `https://arxiv.org/abs/2303.08891`. arXiv preprint arXiv:2303.08891. 2023.

[15] J. H. Seidman et al. *NOMAD: Nonlinear Manifold Decoders for Operator Learning*. `https://arxiv.org/abs/2206.03551`. arXiv preprint arXiv:2206.03551. 2022.

[16] Ashish Vaswani et al. *Attention Is All You Need*. 2017. arXiv: `1706.03762 [cs.CL]`.

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

[18] A. Wunsch, T. Liesch, and S. Broda. "Groundwater level forecasting with artificial neural networks: a comparison of long short-term memory (LSTM), convolutional neural networks (CNNs), and non-linear autoregressive networks with exogenous input (NARX)". In: *Hydrology and Earth System Sciences* 25.3 (2021), pp. 1671–1687.

Chapter 4

Developing a cost-effective emulator for groundwater flow modeling using deep neural operators

Abstract

Current groundwater models face significant challenges in their implementation due to heavy computational burdens. To overcome this, our work proposes a cost-effective emulator that efficiently and accurately forecasts the impact of abstraction in an aquifer. Our approach uses a deep neural operator (DeepONet) framework to learn operators that map between infinite-dimensional function spaces via deep neural networks. The goal is to infer the distribution of hydraulic heads in a confined aquifer in the presence of a pumping well. We successfully tested the DeepONet framework on multiple problems, including forward time-dependent problems, an inverse analysis, and a nonlinear system. Additionally, we propose a novel extension of the DeepONet-based architecture to generate accurate predictions for varied hydraulic conductivity fields and pumping well locations that are unseen during training. Our emulator's predictions match the target data with excellent performance, demonstrating that the proposed model can act as an efficient and fast tool to support a range of tasks that require repetitive forward numerical simulations or inverse simulations of groundwater flow problems. Overall, our work provides a promising avenue for developing cost-effective and accurate groundwater models.

4.1 Introduction

The computational efficiency of existing numerical models for groundwater becomes an issue when dealing with large-scale or highly nonlinear systems, particularly where decision-making relies on real-time simulation inference. The nonlinear nature of many groundwater problems necessitates the use of an iterative process to solve equations, while the computational cost can escalate rapidly when re-calibrating the initial model to incorporate new observational data [5]. Consequently, the heavy computational burden of groundwater models can limit their implementation in decision-making processes, as is the case for the National Water Model (NWM) used to manage drought risk in the Netherlands, for example [26].

In recent years, the use of deep learning techniques for functional approximation has gained significant attention due to their potential in developing efficient low-fidelity models that can approximate expensive numerical methods in a wide range of applications [18, 31, 20, 1, 15]. One area where such approximations have been successful is in using deep convolutional neural networks (CNNs) as surrogates for dynamic multiphase flow problems [33, 28, 27, 38, 34, 32]. The deep CNN-based surrogate models treat the problem as an image-to-image regression, where the input and output functions are represented as images and the deep CNNs learn the mapping between them. The resulting models are capable of accurately predicting pressure and saturation fields with highly heterogeneous aquifer conductivity fields at arbitrary time instances. However, employing deep CNNs as surrogate models has several challenges. It is limited to problems where the input and output functions are defined on a lattice grid, and the training data encompasses all grid values within the computational domain. The solution cannot be evaluated at any arbitrary query point lying within the trained domain. The accuracy of the model and its architecture both depend on the mesh resolution, meaning that the model must be retrained for different mesh resolutions to maintain its accuracy [39]. Furthermore, independent simulations need to be performed for every different domain geometry, input parameter set, or initial/boundary conditions (I/BCs).

For the generalization of the solution, we need to look into higher levels of abstraction to learn the mapping from an input function space to an output function space (and not a vector space as in functional regression). To that end, the universal approximation theorem for operators [3] is suggestive of the potential application of deep neural networks in learning nonlinear operators from data. The neural operators, introduced in 2019 in the form of deep operator networks (DeepONet)[24], learn the mapping between two infinite dimensional Banach spaces, providing a unique simulation framework for real-time prediction of multi-dimensional complex dynamics. Once trained, the DeepONet is discretization invariant, which means the same network parameters are shared across different parameterizations of the underlying functional data, and hence can be used to obtain the solution at any arbitrary spatial and temporal location (interpolation). Furthermore, a recent theoretical work [21] has shown that DeepONet can break the curse of dimensionality in the input space.

In this paper, we demonstrate through multiple problem setups that DeepONet provides fast and accurate inferences for both explicit as well as implicit operators and hence can be employed as an efficient surrogate model in approximating various quantities of interest in the domain of subsurface flows. The reader can refer to section 4.3 for details of the method and an overview of related works. Specifically, we employ the DeepONet framework to design an efficient emulator model to estimate the impact of abstraction in the distribution of hydraulic heads in a heterogeneous confined aquifer. We demonstrate, for the first time, that DeepONet can be applied effectively to groundwater problems and we illustrate some of the potential benefits of this learning approach in the domain of subsurface flows. The proposed method can efficiently learn solutions to both the forward and inverse problems, the latter being notoriously difficult and time-consuming with traditional methods. We successfully employ DeepONet for fast inference of a nonlinear system, which would require the use of iterative methods using standard numerical solvers. Finally, we propose a modification to the vanilla DeepONet in order to successfully predict the distribution of spatially varying groundwater heads given a well that is randomly positioned in the heterogeneous aquifer.

The paper is organized as follows. Section 4.2 discusses the problem statement for groundwater flows and the experiments analyzed in this study: *i*) mapping from the hydraulic conductivity field to the hydraulic head, *ii*) the mapping from a pumping well location and the hydraulic conductivity field to the hydraulic head, *iii*) time-dependent forward problems mapping from the pumping rates to the hydraulic head, *iv*) a nonlinear problem with a head-dependent boundary condition, and finally *v*) learning an inverse mapping from the hydraulic head to the hydraulic conductivity field. Section 4.3 summarizes the DeepONet framework and provides an overview of the model setup, along with training of the surrogate model. In section 5.3, we apply the neural operator to the experimental setups discussed above while section 4.5 summarizes the findings and outlines the future directions.

4.2 Problem statement

This work focuses on learning the non-linear operator that is represented by the solution of the governing equation of groundwater flow employing a neural operator. In this section, we introduce the partial differential equation (PDE) governing the groundwater flow and the computational limitations of a typical numerical solver. We then discuss the experiments which have been designed to illustrate the range of applicability of the proposed method.

4.2.1 Governing Partial Differential Equation

The governing PDE that defines the movement of groundwater on a two-dimensional space combines Darcy's Law and the principle of conservation of mass [13], and is written as:

$$S_s \frac{\partial h}{\partial t} - \nabla \cdot (K \nabla h) = q_s, \tag{4.1}$$

which is constrained by certain boundary conditions. In Equation 4.1, h is the hydraulic head [L], K is spatially varying hydraulic conductivity field [L/T], q_s is the volumetric flux of groundwater sources and sinks per unit volume [1/T], S_s is the specific storage [1/L], and t [T] is time. A table describing all the notations can be found in 5.E.

The U.S. Geological Survey's MODFLOW [13] has been a broadly used in groundwater flow simulation for over 30 years, widely adopted by researchers, consultants, and governments to analyze aquifer behaviors and manage water resources [14, 5]. The model enables assessment of impacts from changes in water withdrawals, recharge rates, and new source introductions. However, its computational demands can be significant for detailed models over extended periods, especially when solving optimization problems with the aim of maximizing groundwater withdrawals given some constraints.

Furthermore, solving an inverse problem for inferring aquifer material properties requires multiple simulations either to discover the missing physics or to calibrate the free parameters of the formulated inverse problem. Such computational burden motivates the development of deep neural network-based emulators to provide predictions with acceptable accuracy while substantially reducing the computational costs at run time. In this work, we have considered a deep neural operator-based surrogate model that is a viable alternative to numerically approximating the governing equation for multiple input functions and thus efficiently forecast the impact of abstraction in an aquifer.

4.2.2 Operator learning task

An operator, denoted by $\mathcal{G}$, is a mathematical function that takes one or more functions as input and produces another function as output. Given an input function $u(x) \in \mathbb{R}^{d_x}$ and an output function $v(x) \in \mathbb{R}^{d_y}$, the operator $\mathcal{G}$ is defined as $\mathcal{G} : u(x) \in \mathbb{R}^{d_x} \mapsto v(x) \in \mathbb{R}^{d_y}$, where $\mathbb{R}^{d_x}$ and $\mathbb{R}^{d_y}$ represents the dimensionality of the inputs and the outputs, respectively, and x denotes the spatial and temporal coordinates which define the output space. A PDE may be regarded as an operator: the input space consists of the functions required to specify the problem definition, such as initial and boundary conditions (ICs/BCs), forcing functions, and coefficients (which may vary spatially and temporally). The output space is the Sobolev space in which the solution of the PDE lies. Our goal is to approximate the PDE introduced in 4.2.1 with a neural operator $\mathcal{G}_\theta$ where θ collectively represents the parameters of the neural operator, the weights, $\mathbf{W}$ and the biases, $\mathbf{b}$. The mathematical formulation of DeepONet is introduced in subsection 4.3.1. We demonstrate the effectiveness of approximating subsurface flows with a neural operator using five computational experiments which are introduced in the next section.

4.2.3 Computational Experiments

The focus of this study is to develop a fast emulator for groundwater flow. We aim to demonstrate that the proposed framework can be used to efficiently estimate the pumping-induced change of the groundwater level, relative to the level before pumping, in a highly heterogeneous confined aquifer (Experiments E_1 and E_2). Furthermore, we employ the framework for solving time-dependent forward problems (Experiments E_3) and nonlinear systems (Experiment E_4), where the solution is conventionally obtained through an iterative process, and hence is heavily time-consuming. Finally, we apply the model to inverse problems (Experiment E_5), which require a large number of forward numerical simulations if a traditional numerical solver is employed. This section introduces the computational experiments (E_1 - E_5) designed in the context of subsurface flow presented in subsection 4.2.1. A visual description of different experiments considered in this work is presented in Table 4.1. Details of the data generation to consider heterogeneity within the aquifer are presented in 4.B.

The description of the experiments, E# are as follows.

- **E1: Forward problem for fixed well location, $\mathcal{G}_\theta : K(x) \mapsto h(x)$:** The goal of this experiment is to learn the solution operator, $\mathcal{G}_\theta$ that maps the spatially varying conductivity field, $K(x)$ to the hydraulic head, $h(x)$ at some subsequent timestep. In other words, the learning goal is to infer the distribution of hydraulic head in a heterogeneous confined aquifer with one fully penetrating well that starts pumping at a constant rate q_s at $t = 0$. The solution is the prediction of the distribution of hydraulic head $h(x)$ at the time instance $t = T$ given a spatially varying hydraulic conductivity field $K(x)$ and under the assumption of no prescribed flows or heads along the boundary of the domain.

- **E2: Multiple input functions, $\mathcal{G}_\theta : [K(x), x_P] \mapsto h(x)$:** The goal of this experiment is to learn a solution operator to approximate the hydraulic head at a time T, given spatially varying hydraulic conductivity $K(x)$ and the location of one or multiple pumping wells x_P as input functions.

- **E3: Time-dependent Forward problem, $\mathcal{G}_\theta : q(t) \mapsto h(x,t)$:** The aim of this experiment is to learn the operator $\mathcal{G}_\theta$ that maps the temporal variation in the pumping rate, $q(t)$, to the hydraulic head $h(x,t)$ at different points in the domain and at different time instances. The hydraulic conductivity is assumed to be constant and the same for all samples. The well locations are also fixed. Three cases are considered in this experiment: (1) Prediction at a single point when having one pumping well (elaborated in 4.C); (2) Prediction over the whole 2D domain when having one pumping well; (3) Prediction over the whole 2D domain when having three pumping wells, each with its own variable pumping rate. In each case, the goal is to predict the hydraulic head distribution over time, given the varying pumping rate(s) as input.

- **E4: Nonlinear Forward problem, $\mathcal{G}_\theta : K(x) \mapsto h(x)$:** The scenario of this experiment is that a pumping well is located in the center of the domain and a head-dependent well is fixed at a different location within the domain. While a pumping well has specified flow boundaries, *i.e.*, the flow is not a function of the head, the specified flow of the head-dependent well is calculated as a function of the hydraulic head, *i.e.*, q_s is $q_s(h)$ in

Equation 4.1. The goal of learning the operator $\mathcal{G}$ is to approximate the mapping between the hydraulic conductivity $K(x)$ of the heterogeneous aquifer and the distribution of hydraulic head $h(x)$ directly. In a traditional solver, nonlinearities are resolved using an iteration loop by repeatedly formulating and solving the governing equation using heads from the previous iteration until the residual of the governing equation is within a specified tolerance. The proposed neural operator-based solution eliminates the need for iterative solvers.

- **E5: Inverse problem,** $\mathcal{G}_\theta : h(x) \mapsto \mathcal{K}(x)$: The aim of this experiment is to learn an inverse operator that approximates the spatially varying hydraulic conductivity field $K(x)$ given the hydraulic head on a domain at different time instances. Understanding that the problem does not have a non-trivial solution, we acknowledge that having more observations of the solution field increases the chances of finding a unique solution that the model converges to. To constrain the solution space in the inverse modeling process, we incorporate sparse observations of hydraulic conductivity $K_0(x)$ as additional inputs. Moreover, in recognition of the fact that hydraulic conductivity is often observed at sparse points in reality, we have conducted additional tests on the model. Specifically, while we trained the model using the whole input data, during the testing phase we only provided sparse input hydraulic conductivity data. This experimental setup represents a more realistic scenario where sparse observations of hydraulic conductivity are available and thus highlights the utility of the model in such situations.

4.3 Solution operator approximation methods

In this section, we introduce the architecture of the deep operator network, DeepONet, and discuss some of the recent works where neural operators have been employed to solve PDEs. Consider two separable Banach spaces, $u = u(\Omega; \mathbb{R}^{d_x})$ and $v = v(\Omega; \mathbb{R}^{d_y})$, where Ω is a bounded open set in $\mathbb{R}^D$, and $\mathbb{R}^{d_x}$ and $\mathbb{R}^{d_y}$ are the dimensionality of the inputs and the outputs, respectively. The nonlinear map $\mathcal{G}$, arising from the solution of a time-dependent PDE (Equation 4.1) at some time T, maps from u to v where, for example in E1, u is $K(x)$ and v is $h(x, t = T)$. The objective is to approximate the nonlinear operator $\mathcal{G}$ through a parametric mapping as $\mathcal{G} : u \times \Theta \to v$ or $\mathcal{G}_\theta : u \to v$, where $\mathcal{G}_\theta$ represents the parametric mapping and Θ is a finite-dimensional parameter space. The optimal parameters θ^* are found by training a neural

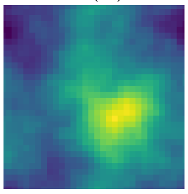
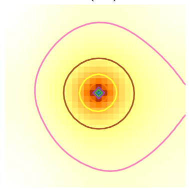
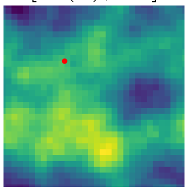
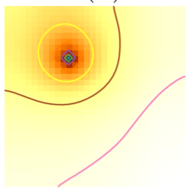
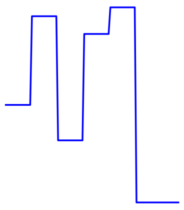
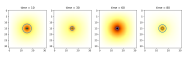
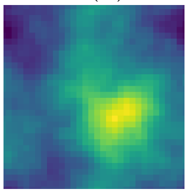
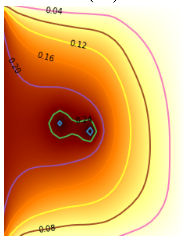
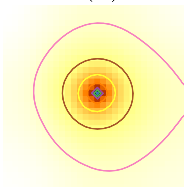
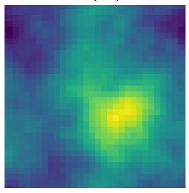
Experiment	**Input Function** $u(x)$	**Output Function** $v(x)$
E1: Forward problem	$K(x)$ 	$h(x)$ 
E2: Multiple input functions	$[K(x), x_P]$ 	$h(x)$ 
E3: Time-dependent problem	$q(t)$ 	$h(x,t)$ 
E4: Nonlinear system	$K(x)$ 	$h(x)$ 
E5: Inverse problem	$h(x)$ 	$\mathcal{K}(x)$ 

Table 4.1: A schematic representation of the experiments under consideration in this work. The input/output functions and representative plots to demonstrate the task that the operator learns are shown.

operator using backpropagation on a dataset of $\{\mathbf{u}_j, \mathbf{v}_j\}_{j=1}^{N}$ generated on a discretized domain.

4.3.1 Deep Operator Network

The universal approximation theorem for operators proposed by Chen and Chen [3] states that shallow neural networks, of sufficient width, are capable of approximating any nonlinear continuous functional or operator to arbitrary accuracy. This theorem is based on a particular neural network model which is composed of two concurrent sub-networks and the outputs of the networks are combined by an inner product. Motivated by the universal approximation theorem, the deep operator Network (DeepONet) [24] was proposed to learn the mapping between Banach spaces with infinite dimensions. The DeepONet architecture consists of two deep neural networks (DNNs): the branch net encodes the input function, u, at fixed sensor points, $\{x_1, x_2, \ldots, x_m\}$, while the trunk net encodes the information related to the spatiotemporal coordinates, $\zeta = \{x_i, y_i, t_i\}$, at which the solution operator is evaluated to compute the loss function. The learning process takes place in a general setting, meaning that the sensor locations ($x_{i\,i=1}^{\,m}$) at which the input functions, u are evaluated don't have to be evenly spaced, but they must be consistent across all input function evaluations. The branch net takes $[u(x_1), u(x_2), \ldots, u(x_m)]^T$ as input and outputs $[b_1, b_2, \ldots, b_q]^T \in \mathbb{R}^q$, while the trunk network takes ζ as input and produces $[t_1, t_2, \ldots, t_q]^T \in \mathbb{R}^q$ as output. These two subnetwork outputs are combined through a dot product to produce the desired result. A bias ($b_0 \in \mathbb{R}$) is added in the final stage to increase expressiveness, resulting in $\mathcal{G}(u)(\zeta) \approx \sum_{i=k}^{q} b_k t_k + b_0$. The optimized values of the trainable parameters θ can be obtained by minimizing a mean square error loss function. Figure 4.1 illustrates the architecture of the vanilla DeepONet proposed in [24].

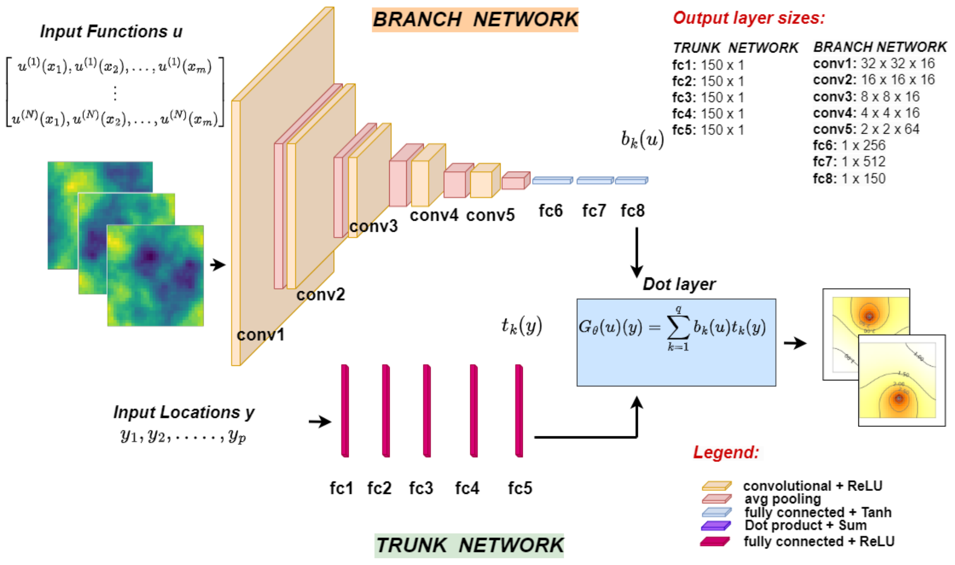


Figure 4.1: Schematic representation of the network architecture of vanilla DeepONet employed in this work. In this work, we have considered a CNN as a branch net and a fully connected feed forward neural network as trunk net. In the figure, "fc" refers to fully connected blocks and "conv" refers to convolutional blocks. The outputs of the branch and the trunk networks are combined through an inner product to approximate the solution operator.

4.3.2 Related Works

DeepONet has shown remarkable success in diverse fields of applications like approximating irregular ocean waves [2], learning stiff chemical kinetics [10], bubble dynamics [23], microstructure evolution [29] *etc.*, where the network is trained using large datasets for solving a forward mode problem. Additionally, some recent work has been focused on learning the mapping of multiple input functions to the solution field [16, 11]. Prior work of DeepONet in the area of subsurface flow problems has been to learn the mapping from the conductivity field to the hydraulic head in simple and complex geometries through data-driven [25] and physics-informed approaches [8, 7, 37]. In both these works, the PDE governing the subsurface flow (Darcy's equation) has been employed as an application to demonstrate the framework proposed in the corresponding work. In another study, an operator-level transfer learning framework [9] was proposed, where Darcy's equation was employed as an example to demonstrate the approach. The idea behind operator transfer learning is to train a source model with sufficient labeled data from a source domain under a standard regression loss and transfer the learned variables to a second target model, which is trained with very limited labeled data from a target (different but related) domain under a hybrid loss function that is the sum of the regression loss and a conditional embedding operator discrepancy loss. Furthermore, another operator-level transfer learning framework was proposed in [17], where Darcy's equation was solved on an L-shaped domain (source) and transferred to an L-shaped domain with a hole. The implementation of a hybrid solver (HINTS) approach could directly handle the change in target geometry and does not require retraining of the operator. None of the works discussed above deal with the additional specific storage term in Equation 4.1 and is not dedicated to employing the operator for multiple scenarios in subsurface flows. Furthermore, for the first time, the operator network is designed to solve an inverse problem in **E5** to learn the hydraulic conductivity field, $K(\boldsymbol{x})$, from the hydraulic head, $h(\boldsymbol{x})$ and sparse observations of $K(\boldsymbol{x})$.

In an independent work of Lanthaler et al. [21], the authors provide a theoretical analysis of the approximation and generalization errors of DeepONet. They accomplish this by decomposing the error into encoding, approximation, and reconstruction errors and theorizing the lower and upper bounds of the total error. Their analysis indicates that the accuracy of DeepONet can deteriorate in the presence of discontinuities or sharp features that are not fixed in space, while DeepONet can accurately learn operators that produce discontinuities or sharp features fixed in space. This is in line with our observation and we propose a modified architecture to deal with such scenarios. According to Hadorn [12], it struggles to learn sharp features for each location without increasing the basis functions from the trunk net. Unfortunately, increasing basis functions becomes infeasible for high dimensional problems. An effective modification to overcome this bottleneck of DeepONet to deal with translational invariance is to eliminate the invariance. Both Shift-DeepONet [12] and FlexDeepONet [35] add pre-transformation sub-networks to shift, rotate and scale the data. The input functions to the branch network are passed through these additional networks, which learn the re-scaling and re-centering of these functions. A transformation layer combines the learned shift, rotation, and scale parameters with the spatial coordinates of the evaluation points: the outputs of this layer are the inputs of the trunk network, such that the basis functions of the trunk net depend on the input functions. Similarly, another extension of DeepONet introduces two encoders, as inputs of the branch and the trunk network [36]. The embedded features are inserted into the hidden layer of both sub-networks using point-wise multiplication. This novel architecture appears to be more resilient than the conventional DeepONet architecture to vanishing gradient pathologies.

In the current work, we consider the vanilla version of the DeepONet as firstly introduced in [24] which is characterized by two separate networks for the branch and the trunk and which has the benefit of a simpler architecture. In the later part of the work, we propose a modified version of DeepONet in order to overcome the limitations of the vanilla DeepONet in dealing with a source term that is not always defined at the same location, and leads to sharp gradients in the solution field. The next section presents the details of the employed network architectures.

4.3.3 Network architecture and training

In the time-independent cases, the branch net is considered as a convolution neural network (CNN) that takes as input the functions, u evaluated on a lattice grid of size 32×32, which is consistent for all the experiments carried out in this work. For the experiments **E1** and **E4**, we have a CNN with one input channel, however, for **E2** and **E5** we have two input channels, where the second channel denotes the location of the well and sparse observations of the target hydraulic conductivity, respectively. When considering sparse observations of the target hydraulic conductivity, the input for this channel remains consistent in size and positions without observations are populated with null values to ensure a uniform structure.

The inputs to the trunk net are the coordinate of 128 evaluation points, which are randomly sampled in the domain and are distinct for each training sample. The details of the network architecture are provided in Table 4.2. A schematic representation of the network is shown in Figure 4.1.

Table 4.2: Architecture details of vanilla DeepONet employed for all the time-independent experiments.

	Layer	**Kernel Size**	**Width**	**Activation**	**Output**
			Branch Network		
1	Conv2D	5×5	16	ReLU	$32 \times 32 \times 16$
2	Avg-Pool	2×2			$16 \times 16 \times 16$
3	Conv2D	5×5	8	ReLU	$16 \times 16 \times 16$
4	Avg-Pool	2×2			$8 \times 8 \times 16$
5	Conv2D	5×5	4	ReLU	$8 \times 8 \times 16$
6	Avg-Pool	2×2			$4 \times 4 \times 16$
7	Conv2D	5×5	4	ReLU	$4 \times 4 \times 16$
8	Avg-Pool	2×2			$2 \times 2 \times 16$
9	Conv2D	5×5	4	ReLU	$2 \times 2 \times 64$
10	Avg-Pool	2×2			Reshaped to 1×64
11	Fully connected		1024	Tanh	1×256
12	Fully connected		1024	Tanh	1×512
13	Fully connected		1024		1×150
			Trunk Network		
14	Fully connected		150	ReLU	150×1
15	Fully connected		150	ReLU	150×1
16	Fully connected		150	ReLU	150×1
17	Fully connected		150	ReLU	150×1
18	Fully connected		150	ReLU	150×1

In the vanilla DeepONet architecture, the solution operator is approximated as the sum of the products of the outputs of the branch and the trunk net. However, for experiment $E2$, we noticed that informing the trunk network about the location of the pumping well (input function) is key for good learning. For this reason, we propose a novel DeepONet architecture. As illustrated in Figure 4.2, each output of the pooling layers of the branch network is combined with the output of each layer of the trunk net. The tensor coming from the branch net is flattened and followed by a dense NN layer with *Sigmoid* activation function. Given the fact that the resulting vector (whose weights can be interpreted as coefficients) has the same dimension as the corresponding hidden layer of the trunk net, the two vectors can be merged via an inner product. The result propagates through the following layers of both the trunk net and, after being reshaped and concatenated to the output of the pooling layer, the branch net. We demonstrate that this architecture can accurately predict the high gradients of the hydraulic head in the experiment **E2**, for which the vanilla DeepONet gives a smoother prediction.

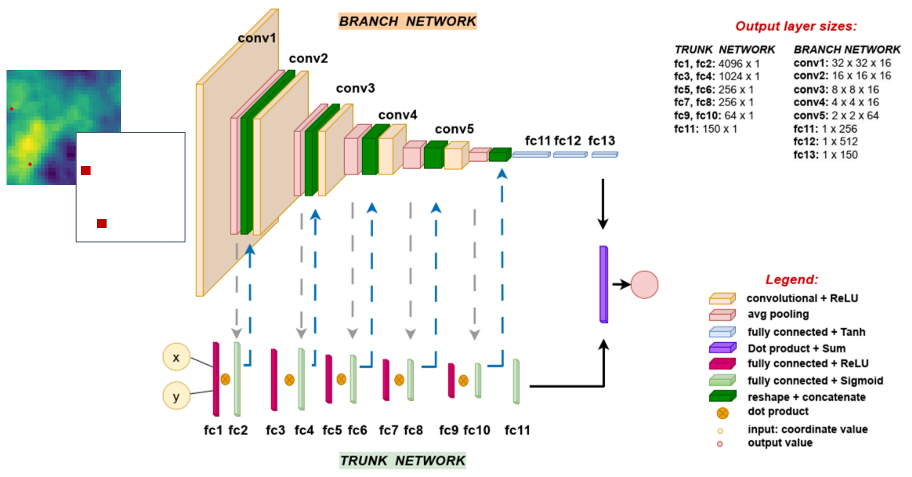


Figure 4.2: Schematic of the novel DeepONet architecture proposed for experiment **E2**. The architecture is specifically designed to take into account the varying locations of the pump. The basis functions approximated using the trunk net can be modified according to the position of the well. In the figure, "fc" refers to fully connected blocks and "conv" refers to convolutional blocks.

In the time-dependent cases (E3), the branch and the trunk are fully connected feed forward neural networks of 7 layers each. All evaluation points are used as input to the trunk network, while the input to the branch network is a vector with the pumping rates in time, which are concatenated in the case of multiple pumping wells. The detailed architecture can be visualized in Figure 4.3.

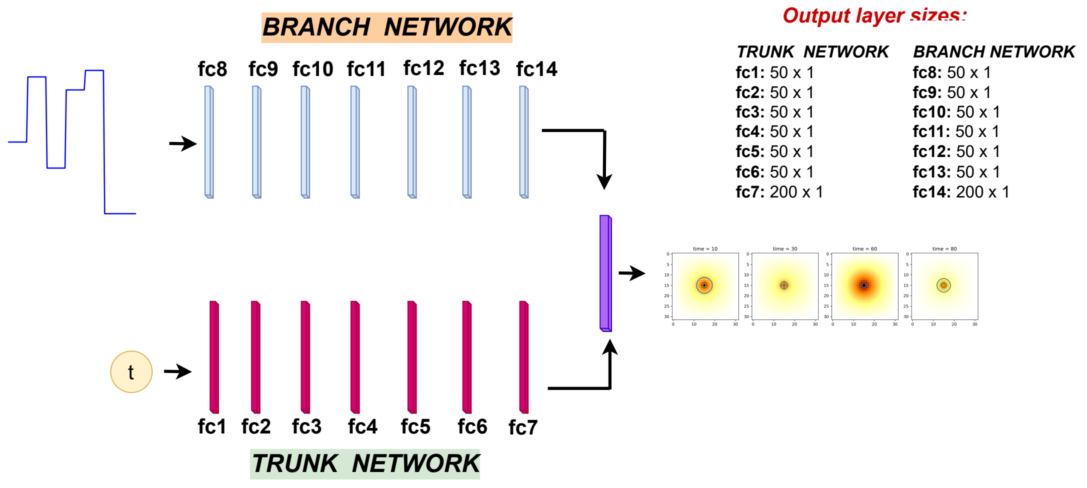


Figure 4.3: Detailed architecture of the network used for the time-dependent case E3. In these scenarios, both the branch and trunk are fully connected feed forward neural networks with 7 layers each. In the figure, "fc" refers to each fully connected block. Time coordinates serve as input to the trunk network, while the branch network receives a time-based vector of pumping rates, which are concatenated when dealing with multiple pumping wells. As in the traditional DeepONet setup, the outputs from the branch and trunk networks are merged through an inner product to approximate the solution operator.

The implementation is carried out using the JAX framework on a single NVIDIA GeForce RTX 3080. For all test cases, the datasets consist of $N_{train} = 1000$ training data and $N_{test} = 200$ test data. The network is trained using the Adam optimizer [19] with an initial learning rate of 5×10^{-4} which exponentially decays every 1000 iteration with a rate of 0.9 and a batch size of 100 for maximum 10^5 iterations. We monitor the loss after every 100 iterations and trigger an early stopping if the value of the loss for the test data does not decrease after 2×10^4 iterations.

4.4 Results

In this section, we demonstrate through the previously discussed experiments that DeepONet can be employed for approximating a range of groundwater flow simulation problems, accurately and efficiently. We also provide a comparative study of the two architectures: the vanilla DeepONet and the novel DeepONet architecture proposed in this paper. All models are trained on a few sparse points defined on the domain but are evaluated over the whole domain for the test data. Please refer to 4.B for specific details regarding the datasets utilized for both training and testing purposes. The mean square error (MSE) is used as an error metric that corresponds to the loss function used during training, calculated as the square difference between the target and the predicted fields for all the models and all the experiments. In 4.D, we provide a comparative study of the accuracy of DeepONet with other popular deep neural network architectures.

4.4.1 $E1$: Forward problem for fixed well location

In this experiment, the location of the well is considered the same for the training and the testing dataset and the operator learns the mapping from the smooth hydraulic conductivity field K to the hydraulic head: $\mathcal{G}_\theta^{F1} : K(x) \mapsto h(x)$. Training the network takes 283 seconds for a total of 24,600 iterations, and the error metrics are computed as $MSE_{train} = 1.8 \times 10^{-5}$ on N_{train} samples and $MSE_{test} = 2.5 \times 10^{-4}$ on N_{test} samples. Figure 4.4 (top row) shows a typical comparison between the predicted and target values of the hydraulic head given a heterogeneous hydraulic conductivity field. Both the inputs and the outputs are normalized along each individual channel, that is the variables are re-scaled into the range $[0, 1]$. As can be seen in the prediction plot, DeepONet can predict the pressure buildup and the sharp increase of the hydraulic head around the well very accurately. We conducted multiple analyses by altering the number of neurons, layers, kernel size, batch size, and training data, among other factors. We observed only slight changes in the error rate, and significant deterioration occurred only when the data was not normalized or the learning rate was too high. Additionally, increasing the number of query points or sampling more frequently around the well did not improve the accuracy of predictions. However, it is important to note that these conclusions were drawn under the condition that the points were already sufficient to capture the sharp increase of the hydraulic head around the well. We emphasize that a low number of points would indeed decrease accuracy, as enough points are needed to learn the high gradient at the well.

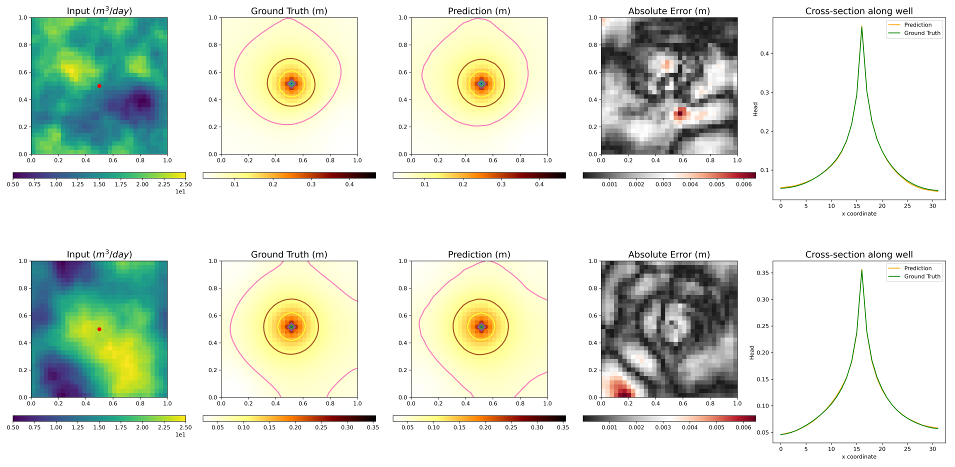


Figure 4.4: Prediction of the hydraulic head, $h(\boldsymbol{x})$ obtained from a trained vanilla DeepONet (Experiment $E1$) for two test cases with an unseen heterogeneous hydraulic conductivity field. The location of the well is fixed at $(x, y) = (16, 16)$, which is the same for the training and the testing dataset.

In a second experiment, the hydraulic conductivity is channeled. The channeled hydraulic conductivity array is created using a customized function that generates a two-dimensional sinusoidal channel pattern. The values of hydraulic conductivity are set to binary, with $k_1 = 5m/day$ representing one type of facies and $k_2 = 25m/day$ representing the other type of facies. The operator learns the mapping: $\mathcal{G}_\theta^{F1} : K(x) \mapsto h(x)$, with K being the hydraulic conductivity field and h the hydraulic head. The DeepONet model demonstrates an impressive level of accuracy in predicting the hydraulic head under such channeled hydraulic conductivity conditions. The average MSE training and testing errors are 4.1×10^{-5} and 8.6×10^{-5} respectively. In Figure 4.4, one can see a typical comparison between the predicted and target values of the hydraulic head given a heterogeneous hydraulic conductivity field. The variables are re-scaled into the range $[0, 1]$. Despite the strong heterogeneity in hydraulic conductivity, the DeepONet model can predict the pressure buildup and the sharp increase of the hydraulic head around the well very accurately. If the trained model is applied to a different type of soil, where the second facies is a gravel with $k_2 = 500m/day$, the test error significantly increases to 4.1×10^{-3}. The increase of the test error is not surprising given that the model has been trained exclusively with samples corresponding to $k_2 = 25m/day$.

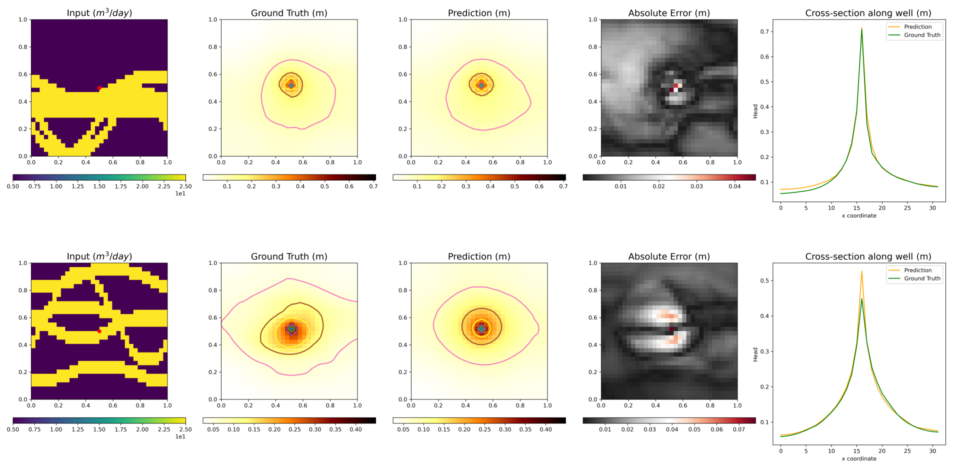


Figure 4.5: Prediction of the hydraulic head, $h(\boldsymbol{x})$ obtained from a trained vanilla DeepONet for two test cases with a channeled hydraulic conductivity field, with $k_1 = 5m/day$ representing one type of facies and $k_2 = 25m/day$ representing the other type of facies. The location of the well is fixed at $(x, y) = (16, 16)$, which is the same for the training and the testing dataset. The first column represents the channeled hydraulic conductivity field, while the second and third columns denote the ground truth and the prediction of DeepONet for the hydraulic head, respectively.

4.4.2 $E2$: Forward problem for varying well locations

In the next experiment, we further expand the network capabilities to learn the solution for unseen locations of the pumping well. According to the original formulation of the DeepONet, the branch network encodes the input functions, which are the hydraulic conductivity field and the position of the source term (the well). We observed that the vanilla architecture could not capture the sharp gradients located in the region near the pumping well when the well was shifted to several locations. The reader can refer to the last column of Figure 4.6 for visualization of vanilla DeepONet predictions for three representative test samples. In this visualization, the location of the well (forcing term) is encoded as a binary map and concatenated to the hydraulic conductivity field as an additional channel of the existing CNN of the branch net. We conducted multiple computational experiments to explore different approaches to encoding the forcing term as an input to the branch net. We tried using two separate branch networks for each input function and giving the location of the source term in various forms, such as by using Cartesian coordinates or the distance between each point of the domain and the well's location or a Gaussian function centered at the well's location. However, these experiments were not successful, as they led to either highly inaccurate predictions with incorrect determination of the extent and location of the pressure front or smoother predictions near the source terms.

4.E explores the reason for the lower prediction accuracy of the vanilla DeepONet for these test cases through the lens of the singular value decomposition (SVD). Finally, we found that informing the trunk net of the location of the source term is necessary for good learning in this class of problem. When the input to the trunk network is the concatenation of the coordinates in which the network evaluates the solution and the coordinates of the pumping well, the predictions match the reference solutions very well for different distributions of $K(\boldsymbol{x})$ and for varying locations of the pumping well (the error metric, evaluated on the training and testing datasets respectively, yields values of 2.4×10^{-5} and 2.7×10^{-4}, respectively). In general, the role of incorporating well locations into the trunk network lies in enabling effective localization of the basis functions around the wells. However, in practical scenarios, where the number of wells could vary significantly, it is impractical to predetermine the number of well coordinates for the trunk network. Moreover, this approach becomes inefficient for more complex scenarios, such as multiple wells within the domain, different pumping rates of the wells, or features not localized at a single point (like rivers and drains). In such cases, we decided to encode the well location as a binary map, which is then concatenated to the hydraulic conductivity field as input to the branch network. As our experiments showed that informing the trunk network with the location of the forcing term is key for good learning when the location of the forcing term varies among the training data, we link the branch and the trunk net with the newly proposed architecture of DeepONet (Figure 4.2) described in 4.3.3. As shown in the first two rows of Figure 4.6, the architecture that interlinks the hidden layers of the two sub-networks substantially outperforms the vanilla DeepONet for scenarios with a single pumping well. The training process takes approximately 324 seconds, and the computed error metrics equate to 6.6×10^{-5} and 2.6×10^{-4} on the training and testing datasets, respectively. Extending the analysis, we also accommodated scenarios with two randomly located pumping wells. Here, the proposed architecture continued to yield superior performance compared to the vanilla DeepONet, with computed error metrics of 6.4×10^{-5} and 4.3×10^{-4} on the training and testing datasets, respectively. These results are shown in the last two rows of Figure 4.6.

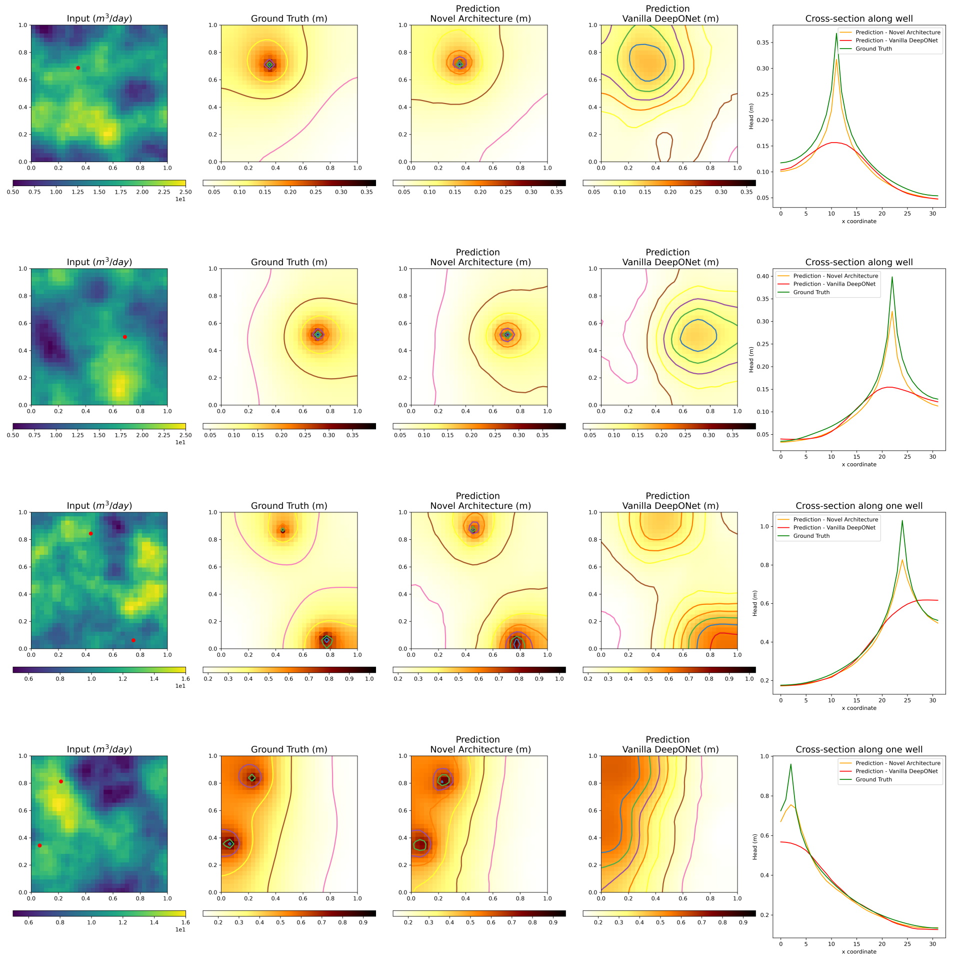


Figure 4.6: Predictions of the hydraulic head for unseen heterogeneous hydraulic conductivity fields (first column) and unseen location of the pumping wells. Predictions with vanilla DeepONet (fourth column) and proposed network architecture (third column) are compared with the target fields (second column) for four representative test samples, with either one or two pumping wells. Input to the branch network is the hydraulic conductivity field and a binary map indicating the location of the well. Input to the trunk network is the coordinates in which the network evaluates the solution.

4.4.3 *E*3: Time-dependent Forward problems

In the time-dependent case, we consider a confined aquifer where the pumping rates at one or more wells change over time. The aim is to predict the hydraulic head response in both one and two-dimensional scenarios. The evaluation of the one-dimensional case is detailed in 4.C. For the two-dimensional case, we first consider a 32×32 grid with a single pumping well. The well operates for 100 days, maintaining a constant rate for 10 days. Our model aims to predict the hydraulic head distribution over the domain in response to this pumping action.

Figure 4.7 demonstrates the model's prediction. The results demonstrate that the model can adapt and successfully handle higher-dimensional inputs and outputs, with an average MSE test error equal to 2.3×10^{-8}. For a detailed illustration of the inputs and outputs in the case of a single point evaluation, please refer to Figure 4.20 in 4.C.

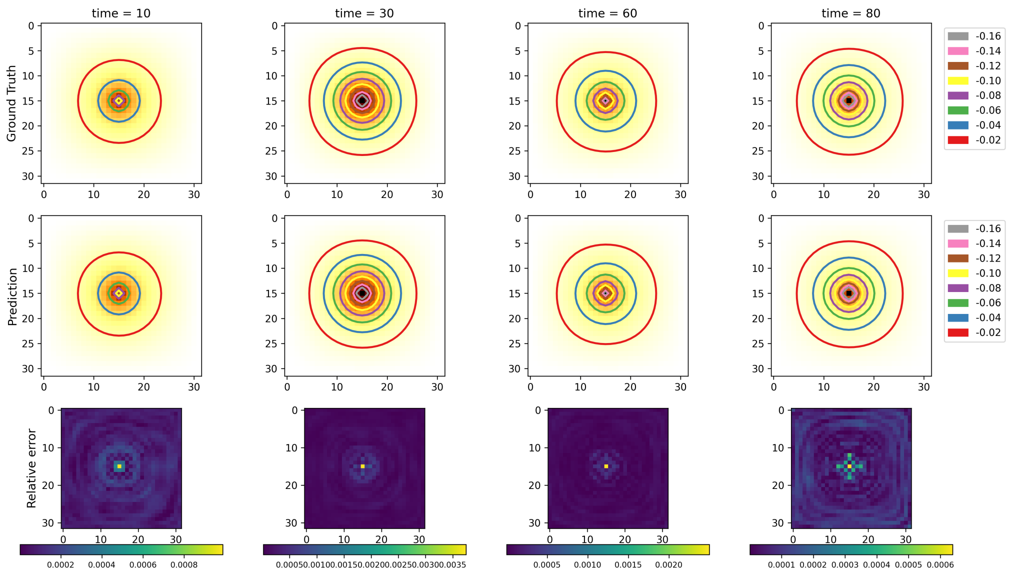


Figure 4.7: This figure depicts the results for a test sample in a two-dimensional scenario with a single pumping well: the input to the model is the time-dependent pumping rate of the well located at the center of the domain. The first row shows the ground truth hydraulic head, while the second row presents the predicted hydraulic head from our model, with each contour line representing a different hydraulic head value. The third row represents the difference between the ground truth and predicted values. The results demonstrate an excellent match between the prediction and ground truth.

Then, we consider a more complex scenario with three pumping wells each operating with a different time series as input. For a more comprehensive overview of the dataset, please consult 4.B and Figure 4.17. The pumping rates are input to the branch, concatenated in a single vector. Input to the trunk are the 2D spatial and the temporal coordinates for the evaluation points. The results shown in Figure 4.8 indicate that the model accurately predicts the response of the hydraulic head over the domain to the operations of the three wells. The average MSE training and testing errors are 2.95×10^{-4} and 3.02×10^{-4} respectively.

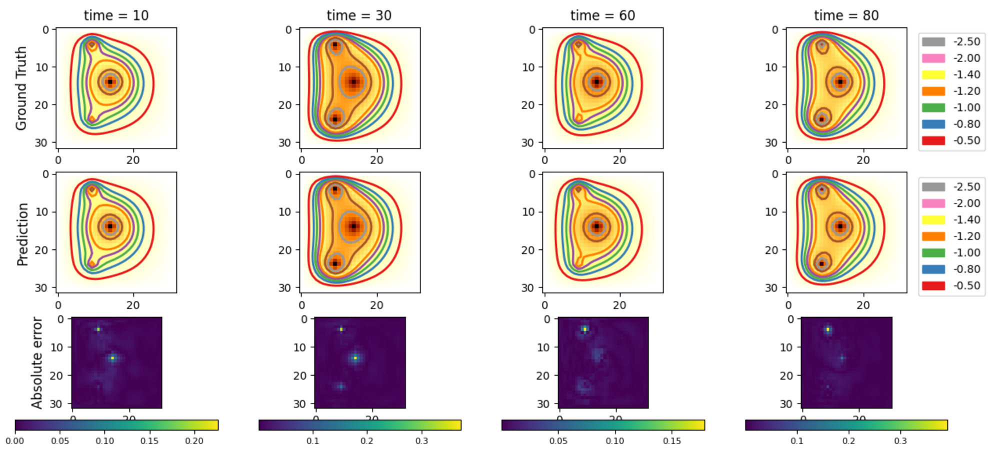


Figure 4.8: This figure illustrates the results for a test sample in a two-dimensional scenario with three pumping wells. The model receives the normalized time-dependent pumping rates of the three wells as its inputs. Similar to Figure 4.7, the first row shows the ground truth hydraulic head, the second row shows the predicted hydraulic head, and the third row highlights the difference between these two. Each column corresponds to a different time step.

4.4.4 $E4$: Nonlinear Forward problem

Many groundwater processes exhibit nonlinear behavior and hence require the use of iterative solvers to obtain the solution of the hydraulic head. In this study, we show that the proposed data-driven method has the capability to resolve the system directly and save on the computational cost required on the conventional iterative solver. The traditional finite-difference form of the groundwater flow equations can be written as $Ah = b$, where h is the vector of head values at the end of the time step, A is the matrix of the coefficients of head and b is a vector of the constant terms [13]. In a nonlinear system, the individual entries in A matrix is a function of the hydraulic head and the system of equations needs to be resolved through a nonlinear outer iteration loop.

This example explores a nonlinear groundwater system modeled by a single-layer confined aquifer, with hydraulic conductivity varying across the domain, Dirichlet boundary conditions, and featuring a consistently-flowing pumping well located in the center of the domain and a head-dependent well that utilizes a superimposed drain approach to emulate nonlinearity, which is fixed in another location in the domain. The reader is referred to 4.B for a broader description of the dataset. The hydraulic head predicted by the neural operator accurately matches the target values in the whole domain for given hydraulic conductivity fields (considered as input to the branch net), unseen during training (Figure 4.4.4). The training process takes 430 seconds for a total of 44100 iterations and the error metrics computed on the training and the testing datasets are equal to 3.1×10^{-5} and 3.9×10^{-5}, respectively.

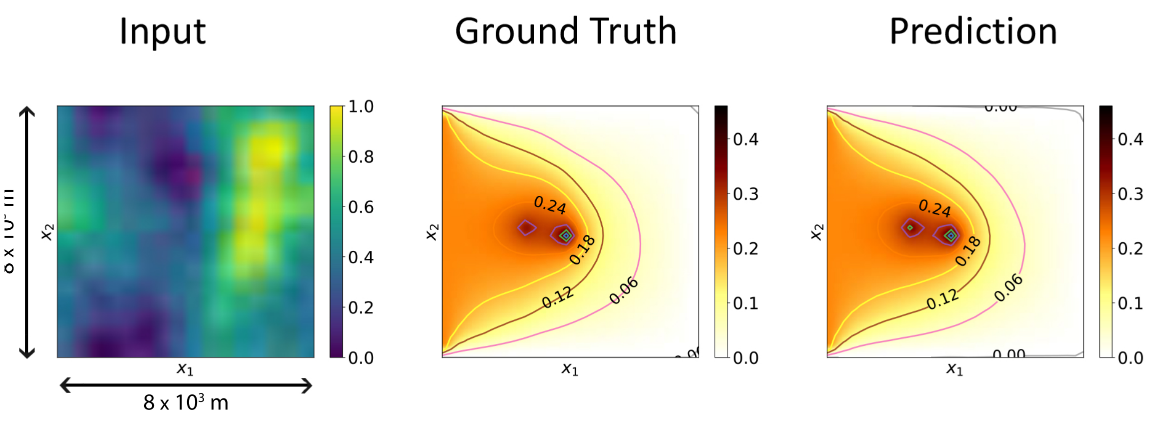


Figure 4.9: Comparison between the ground truth and the prediction of the hydraulic head from DeepONet for unseen heterogeneous conductivity field for the nonlinear problem, **E4**.

4.4.5 $E5$: Inverse problem

This section explores the efficiency of the operator network for solving inverse problems and, more specifically, for underground property characterization. Given that it is impossible to directly observe the whole underground system, the aim of the inverse analysis is to understand the heterogeneous aquifer properties (*i.e.*, hydraulic conductivity field) using sparse observations of the conductivity field and some known information of the hydraulic head. We consider a test problem for which the information on the hydraulic head in the whole domain is available along with sparse observations of hydraulic conductivity. While no practical scenarios exist where the hydraulic head is known universally, this idealized setup serves as a benchmark, enabling us to gauge the optimal accuracy achievable. Figure 4.10 shows a representative test case with the inputs and the prediction obtained from the operator network. The MSE error computed on the test samples are obtained as 1.02×10^{-2} when using 20% of the values of the hydraulic conductivity field (randomly sampled) as observation points. Beyond the comparison between the target reference fields (first column) and the simulated inverse results (fourth column), we also compare the input hydraulic head (second column) and the hydraulic head corresponding to the predicted conductivity as calculated with the traditional solver (third column). The operator network gives trustworthy predictions with accurate and consistent performance across the whole test dataset: the MSE test error is equal to 1.46×10^{-2}, with standard deviation 5.3×10^{-3}.

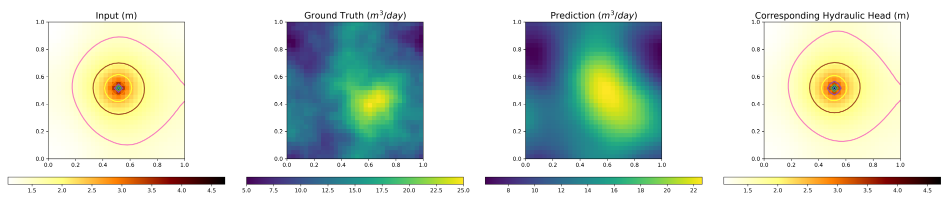


Figure 4.10: Sample test case from the inverse analysis using the vanilla DeepONet. In the first column, we see the input hydraulic head, while the second column illustrates the unknown hydraulic conductivity field. The third column displays the results of the simulated inverse procedure, and the final column shows the hydraulic head corresponding to the predicted conductivity, as calculated using a conventional solver.

The use of the hydraulic conductivity observational data informs the network and enhances its accuracy when compared to the case in which no measurements of hydraulic conductivity are made available. Incorporating 20% of hydraulic conductivity data as supplementary input results in a noticeable enhancement in model accuracy, with a 29.5% reduction in the MSE error for test samples. This improvement becomes more pronounced and equal to 37.6% with 40% of hydraulic conductivity data as extra input. A further improvement can be achieved by informing the network with the observation of the hydraulic heads at different timesteps. Specifically, by encoding the hydraulic head at intervals of 67, 133, and 200 days in the branch net — as opposed to a singular observation at 200 days — the predictions are refined by 4%. Moreover, in the real world, the observations of hydraulic heads are inherently sparse. Therefore, we assessed all 200 test samples, picking randomly selected points to use as input for the hydraulic conductivity. Figure 4.11 presents the results of the same test case of Figure 4.10 but when using only 10% randomly chosen observations of the hydraulic head as input. As expected, the results are less accurate when compared to the case in which the input is the whole field. However, it's worth noting that the model still manages to capture the dominant features of the system. In particular, the predicted values of hydraulic conductivity are not only lower in the corners of the domain, mirroring the ground truth, but also the regions with the highest values match those of the actual system. This is likely due to the distribution of observation points across the domain, which allows the model to grasp the overall trends, even with limited data. Furthermore, in reporting these results it is important to highlight the ill-posed nature of this version of the problem, based upon sparse hydraulic head observations. It is certainly possible that more than one conductivity map could lead to an identical set of sparse water head observations (with the likelihood of this increasing with increased sparsity), so the results presented in Figures 4.11 and 4.12 should be considered in this context.

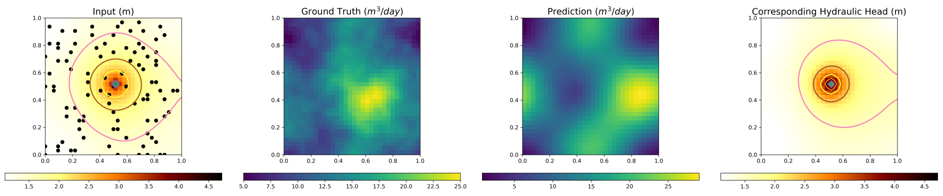


Figure 4.11: This figure presents the same test sample from Figure 4.10, but in this case, only 10% of the hydraulic head is supplied as input. The randomly chosen points are depicted as black dots in the first column.

Figure 4.12 displays the average error (solid line) along with the standard deviation (shaded region) for all test samples across a range of 0 to 100% input hydraulic head. The results improve with an increase in input observation: with less than 4% input, the accuracy is low, but it substantially improves as we add more data.

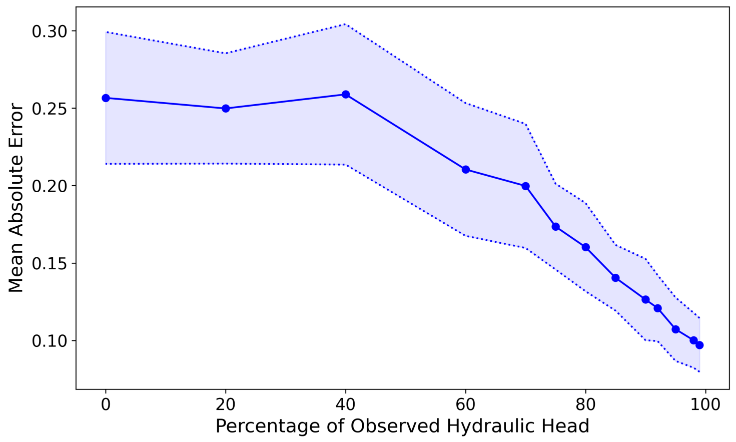


Figure 4.12: Graphical representation of the mean absolute error for different percentages of the known information from the hydraulic head. The solid line denotes the mean and the errors computed across all the test samples for a given number of observations of the hydraulic head shown in the $x-$axis, and the shaded region denotes the standard deviation of the errors.

4.5 Summary and Discussion

This paper presents the DeepONet framework as a surrogate model to efficiently and accurately calculate the state response of a groundwater system. The model is trained and tested in multiple experiments that demonstrate its capability to predict hydraulic head in a heterogeneous aquifer with a fixed pumping well, generalize to unseen pumping well locations and time-dependent pumping rates, characterize aquifer properties (inverse analysis), and deal with nonlinear systems. The proposed model accurately learns both the forward and inverse relations between the spatially varying hydraulic conductivity and the hydraulic head fields very accurately. However, modifications to the original formulation of DeepONet are needed when the pumping well is placed at any location in the domain. To address this, the paper introduces a novel contribution by linking the input of the branch network to the trunk network, allowing the network to accurately predict solutions for unseen well locations and hydraulic conductivity fields. By successfully implementing the neural operator on several examples, we demonstrate the capacity of the network to support a range of tasks that require repetitive forward numerical simulations of the groundwater model. We acknowledge that our current study is limited to single-well and two-wells configurations and we would like to extend our work in the direction of multiple well-surrogate modeling in the future. Moreover, we explored the impact of sparse observations for inverse modeling. While our focus was on the quantity of observations, the strategic placement of these points is a subject for future research. Our experiments provide initial insights into leveraging limited observations for reliable predictions and the influence of data distribution on the model performance. Future research directions should include uncertainty analysis to account for the fact that each set of sparse water head observations can correspond to multiple conductivity maps. Additional avenues to consider might be the use of a multi-fidelity and multi-modality approaches. Real-world data are not only sparse but come from varied sources, each with unique characteristics. Integrating these diverse data sources presents both a significant challenge and a promising research opportunity.

In the long term, such a model will be extended to accommodate more complicated and realistic sub-surface problems. These could include more complex predictions from a wider range of abstraction rates and aquifer system geometry, properties, and boundaries, and the interaction with other surface water abstractions and discharges.

4.A Appendix A: Table of notations

A table of notations is given in Table 4.3.

Notation	Meaning
h	Hydraulic head
K	Hydraulic conductivity
q_s	Volumetric flux of groundwater sources and sinks per unit volume
S_s	Specific storage
t	Time
x_P	Location of the pumping well
x	Spatial and temporal coordinates
$\mathcal{G}$	Nonlinear Operator
$\mathcal{G}_\theta$	Network operator approximator
θ	Learnable parameters
u	Input function
y_j	Evaluation point
m	Number of sensor points
P	Number of evaluation points
N	Number of input functions
N_T	Number of training sample pairs
q	Number of tunable weights at the last hidden layers of the two sub-networks
b	The output vector defined by the last layer of the branch network
t	The output vector defined by the last layer of the trunk network

Table 4.3: Table of notations

4.B Appendix B: Numerical simulation settings for generating the datasets

The labeled high-fidelity datasets for training the data-driven neural operator framework are obtained by solving the governing equation using a finite difference scheme. The parameters and settings used for generating the labeled dataset are described in this section. The datasets are generated within the Python ecosystem and, in particular, using the iMOD Python package for time-dependent and for nonlinear cases [4] and a script adapted from [6] for the remaining test cases.

$E1$**: Forward problem for fixed well location.** Figure 4.13 shows representative plots of input and output pairs from the generated dataset. This problem consists of a single-layer model representing a confined aquifer. The layer is 50 m thick and the extent of the model R is 8×10^3 m. The grid is defined as 32×32. Each cell is 250 m on a side, which corresponds to a typical spatial grid used in practice [5]. The aquifer, whose specific elastic storage S_s is 2×10^{-4} 1/m, is highly heterogeneous with horizontal hydraulic conductivity $k \in [5, 25]$ m/day. We create the hydraulic conductivity distributions using Gaussian random fields with Power Spectrum $P(l) = l^q$ with $q = -4$. The resulting continuous float values are re-scaled in the range $k \in [k_{min}, k_{max}]$. The minimum and maximum values of K, k_{min} and k_{max}, vary for each sample: they are randomly selected integers so that $k_{min} \in [5, 10]$ m/day and $k_{max} \in [20, 25]$ m/day. A pumping well, placed at $(x, y) = (16, 16)$, is extracting water from the sandy aquifer. Abstraction is specified at a constant rate of $Q = 5000$ m^3/d, which corresponds to a typical abstraction value [5]. This generates the target hydraulic head, which is the solution at the time step corresponding to the simulation time $T = 200$ days.

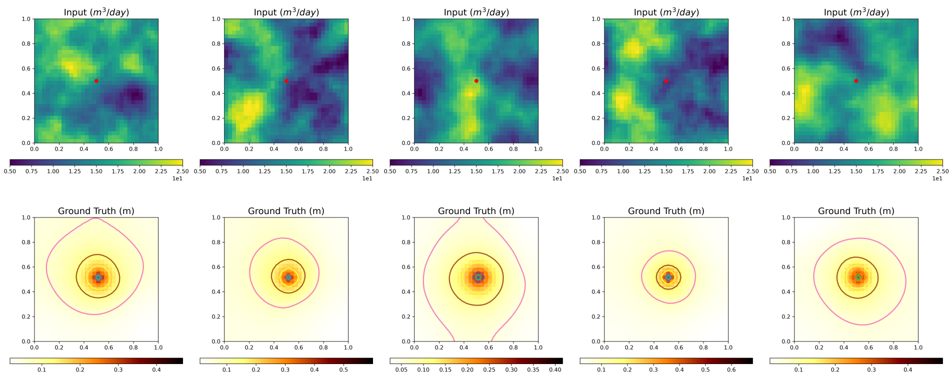


Figure 4.13: Representative input(top row) - output (Bottom row) pairs for $E1$, which considers the forward problem for a fixed well location operating at a constant rate. The well is situated in the center of the domain, extracting water at a fixed rate, which affects the hydraulic head distribution over time. The first row represents model input, specifically the hydraulic conductivity, and the second row illustrates the ground truth for the hydraulic head.

The forward problem with channeled input is similar to the previously discussed forward problem, except the hydraulic head is not randomly generated but developed in a channeled pattern (Figure 4.14). The process involves the creation of certain sinusoidal 'channel'paths which overlaid with a specific value, while the remaining areas are filled with a different value. This approach results in a channeled and binary distribution of hydraulic heads.

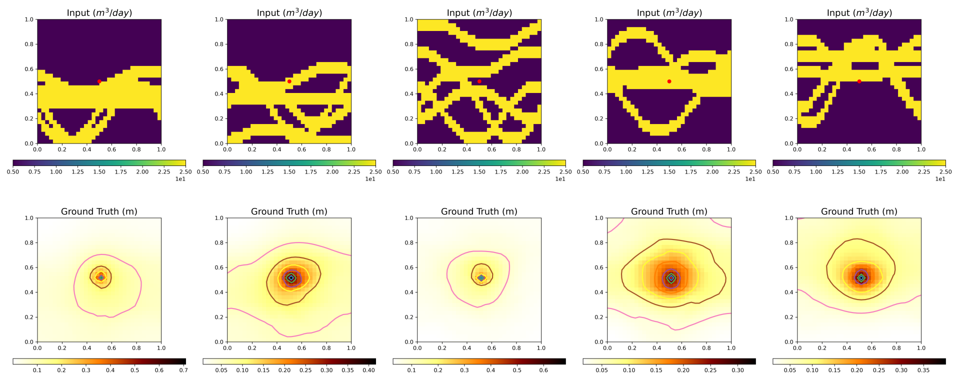


Figure 4.14: Representative input(top row) - output (Bottom row) pairs for $E1$, which considers the forward problem, where the hydraulic head is generated in a channeled pattern. The distribution is binary, with channels overlaying certain paths with $k_1 = 5m/day$, while the remaining areas are filled with $k_2 = 25m/day$. The top row represents model input, specifically the hydraulic conductivity, and the second row illustrates the ground truth for the hydraulic head. Each column is a unique sample from the dataset.

$E2$**: Forward problem for varying well locations.** In this case, we continue to learn the mapping between the hydraulic conductivity field K to the hydraulic head, h, however, the pumping well is randomly placed in the domain and the operator considers the well location, x_P as an additional input in addition to the conductivity field. All other settings are kept the same as discussed above. Figure 4.15 shows representative plots of samples taken from the labeled dataset of this experiment.

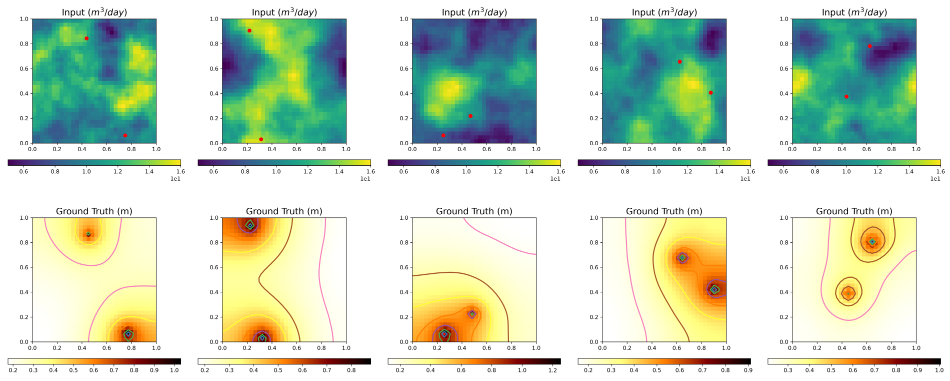


Figure 4.15: Representative input(top row) - output (Bottom row) pairs for $E2$, which considers the forward problem, where two pumping wells are randomly located in the domain. The operator is tasked to learn the mapping from both the well location and the hydraulic conductivity field (first row) to the hydraulic head (second row). Each column is a unique sample from the dataset. The well locations are marked using red dots.

$E3$**: Time-dependent Forward problem.** In the time-dependent cases, the study focuses on a confined aquifer over a 500×500 area with a 100m depth. The aquifer is characterized by a specific elastic storage of 5×10^{-5} and a uniform hydraulic conductivity of 0.5 m/day. Constant-head cells set at 0 m are imposed along all the boundaries following Dirichlet boundary conditions. The time-dependent forward tests encompass three scenarios. The first case is where the model predicts the hydraulic head response to a single pumping well located at the center of the domain, operating for 100 days with a constant rate maintained for 10 days in a single point scenario (see Figure 4.19). The second case broadens this to a 2-dimensional domain, aiming to test the model's adaptability to higher dimensions, with outcomes depicted in Figure 4.16.

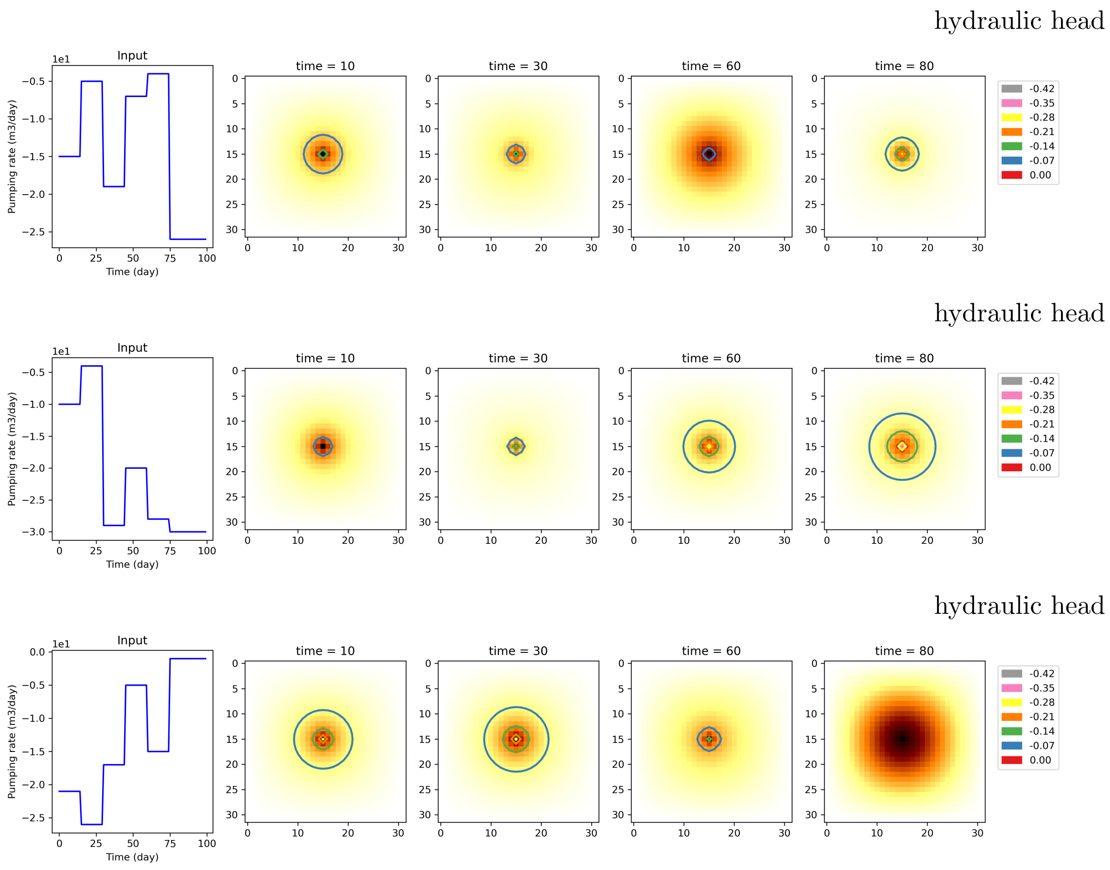


Figure 4.16: Three representative samples (row-wise) for $E3$, which aims to predict the hydraulic head for a single pumping well located at the center of the domain. The well is operated for 100 days with a constant rate that changes every 10th day, as shown in the first column. The first column is the pumping rate, and the other columns represent the solution in the 2D domain at different time steps in days ($t = 10, 30, 60, 80$).

The third and final scenario involves a complex setting with three pumping wells each operating at different pumping rates fluctuating between a minimum of 500 m^3/day and a maximum of 50 m^3/day (Figure 4.17). The location and the number of wells are kept the same across the training and testing datasets.

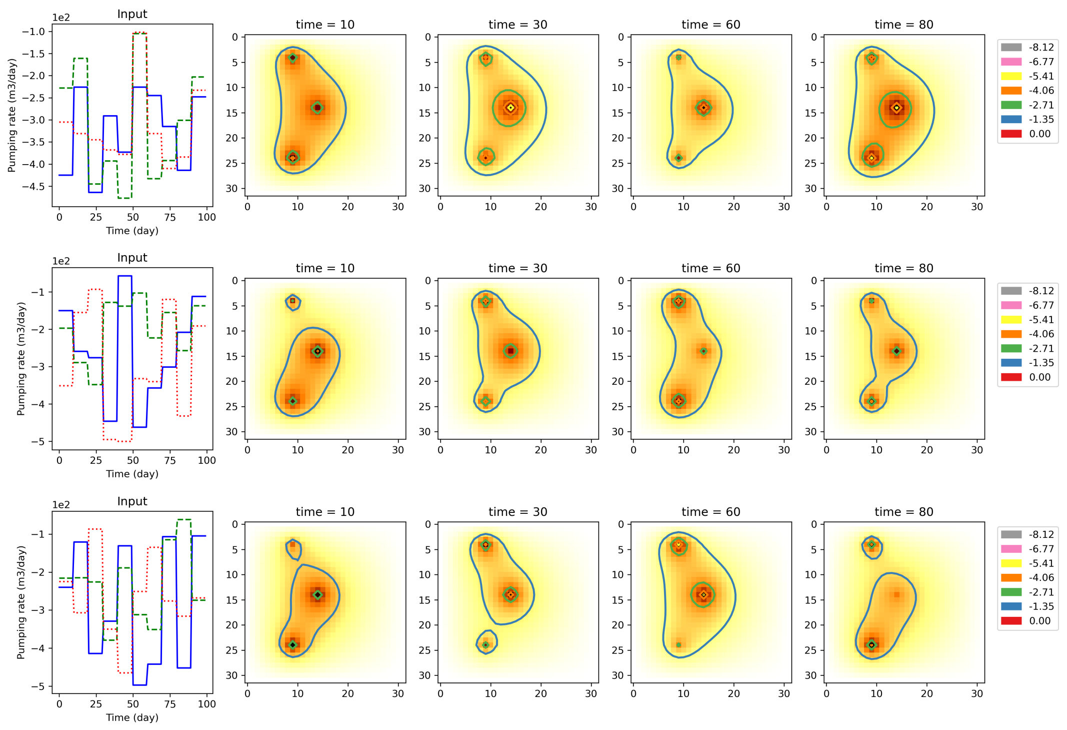


Figure 4.17: The model prediction of the hydraulic head response to three different pumping wells, each operating with different pumping rates. All three pumping rates are presented in the first column, and the following columns illustrate the solutions in the 2D domain at different time steps. Different rows show different dataset samples.

$E4$**: Nonlinear system** The problem consists of a single-layer model representing a confined aquifer and the aim is to map the hydraulic head from the conductivity field. The layer is 60 m thick with a surface level of 0 m and the extent of the model R is 3.2×10^4m. The domain is discretized as 32×32 and each cell is 1000 m on a side. The specific storage of the aquifer $S_s = 1.6 \times 10^{-4}$ [1/m] and its hydraulic conductivity varies spatially for each data sample. The conductivity fields of the heterogeneous aquifer are Gaussian random fields with Power Spectrum $P(l) = l^q$ with $q = -4$, with values re-scaled in the range $k \in [k_{min}, k_{max}]$, where $k_{min} = 5 \times 10^{-4} m/day$ and $k_{max} = 5 \times 10^{-3} m/day$. Figure 4.18 (left) illustrates the setup of the model. The initial head corresponds to the surface level everywhere. We impose Dirichlet boundary conditions with constant-head cells equal to -2 m along the boundary $x = 0$ and equal to the surface levels along the other three sides of the domain. A pumping well is located in the center of the domain, *i.e.*, at $(x, y) = (16, 16)$. The pumping well has specified flow boundaries: the flow is constant in time and equal to 0.5 m^2/d. A head-dependent well, whose flow is calculated as a function of the head, is located at $(x, y) = (10, 16)$. The relationship between the flow and head of this well is reported in Figure 4.18 (Right).

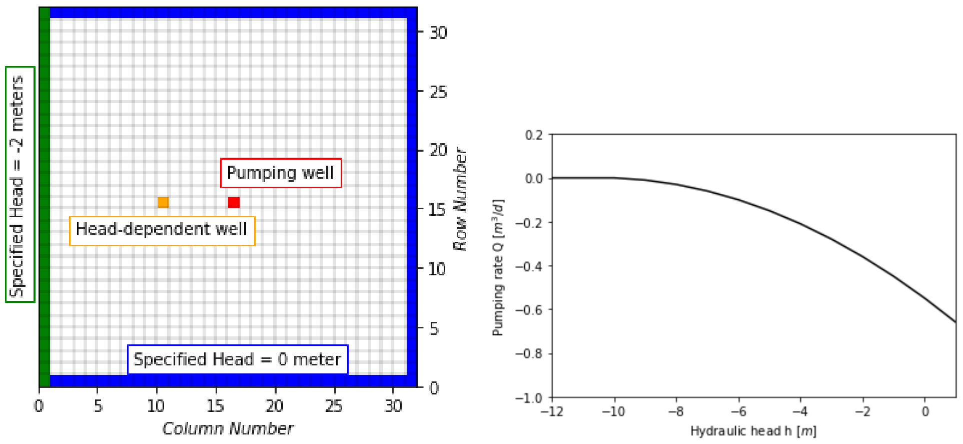


Figure 4.18: Setup for the inverse problem test case ($E4$). Left: boundary conditions and source terms; right: plot demonstrating the relation between flow and head for the head-dependent well.

A simple way to specify this user-defined function is by superposing boundary conditions that are already implemented in MODFLOW. A drain is a type of boundary condition that removes water from the aquifer at a rate, called conductance, proportional to the difference between the head in the aquifer and the drain elevation. If the head in the aquifer is below the drain elevation, the drain is deactivated and the flow is null. In order to build the function between flow and head for the head-dependent well, we decided to place 11 drains in the same cell, one underneath the other. The topmost drain is located at an elevation 0 m, the spacing between the drains is 1 m in the vertical direction and all drains have the same conductance equal to 0.010 m^2/d. The resulting superposition of the 11 separate piecewise functions corresponds to a single function between the flux and the head in the cell. This may be treated as a single head-dependent well which removes water according to the relationship illustrated in Figure 4.18 (Right) with an approximate function found by interpolation:

$$Q(u) = \begin{cases} 0 & \text{if } u \leq -11 \\ -5.0 \cdot 10^{-7}x^5 + 2.0 \cdot 10^{-6}x^4 + 2.4 \cdot 10^{-4}x^3 + 2.9 \cdot 10^{-3}x^2 - 0.11x - 0.66 & \text{if } u > -11 \end{cases} \tag{4.2}$$

The calculations are carried out with iMOD Python for a single stress period of 200 days. The hydraulic head is the output of the calculation, as shown in Figure 4.1 (fourth row).

$E5$: **Inverse problem** The learning goal of this problem is to infer the hydraulic conductivity in the whole domain given the spatial distribution of the hydraulic head resulting from water abstraction. The numerical settings used for the inverse problem are the same as the forward problem, with the only difference being that the pumping well is now located in the center of the domain and that the resultant values of the hydraulic conductivity belong to five distinct classes. This method is utilized to make the inverse problem more feasible and to confine the range of potential solutions. Nevertheless, the model learns the hydraulic conductivity with a regressive approach, which means that the inferred values of hydraulic conductivity are not constrained to belong to those predefined classes. The input and target data of the forward problem discussed above become the target and the input for the inverse problem, respectively. In addition to the hydraulic head calculated from the simulation at time T, corresponding to 200 days, we also provide sparse observations of the target hydraulic conductivity as an additional input.

4.C Appendix C: Additional time-dependent analyses

In this section, we delve further into the exploration of the time-dependent analyses. Specifically, we extend our investigation by presenting results derived from a one-dimensional time-dependent scenario. Additionally, we offer supplementary insights by introducing a distinct plot concerning the two-dimensional spatial and temporal case, as previously discussed in 4.4.3.

4.C.1 Response to a Single Pumping Well at a Single Point

In this scenario, we focus on the hydraulic head response at a single point to a central pumping well operating for 100 days. Figure 4.19 represents the inputs, outputs, and prediction of this case. The average MSE training and testing errors for this case are 5.8×10^{-8} and 6.0×10^{-9} respectively.

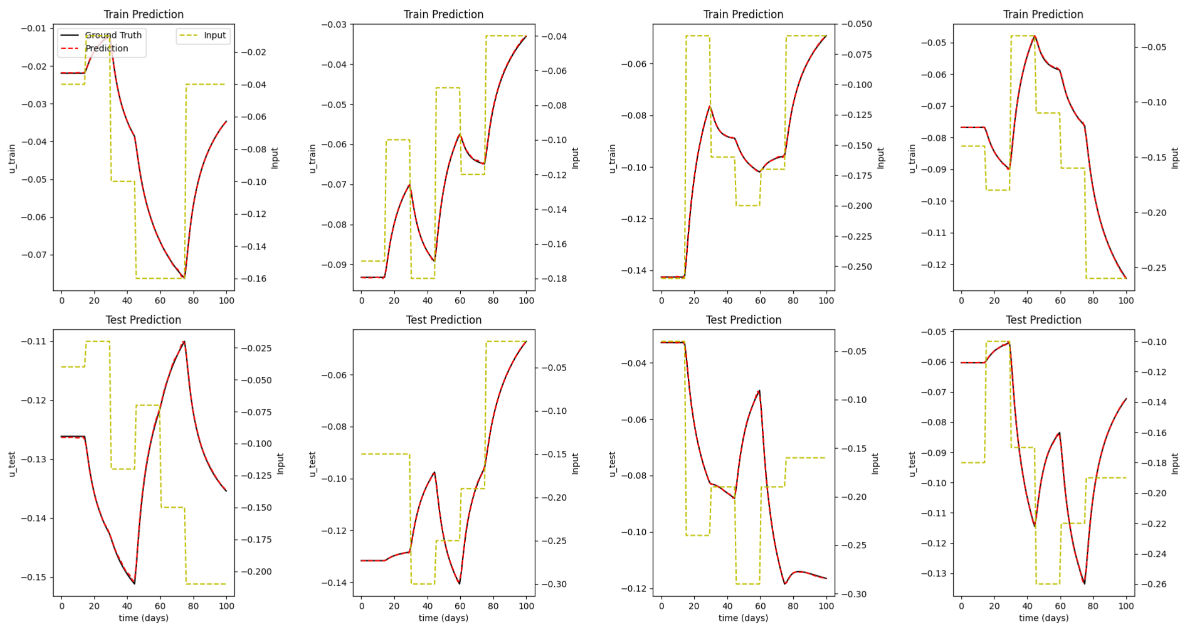


Figure 4.19: Inputs and Outputs for 1D case: point (12,15) of the 2d case.

4.C.2 2D problem: Performance on a Single Point Evaluation

In this subsection, we present additional results of the two-dimensional time-dependent case with a single pumping well (4.4.3) but evaluate the model's performance at a single point (12,15) in the 2D domain. As indicated by the results in Figure 4.16, our model shows an outstanding performance. The predicted values match the ground truth almost perfectly, demonstrating the robustness of our model in accurately predicting the hydraulic head at a specific location in the domain, despite the changes in the pumping rate at the well.

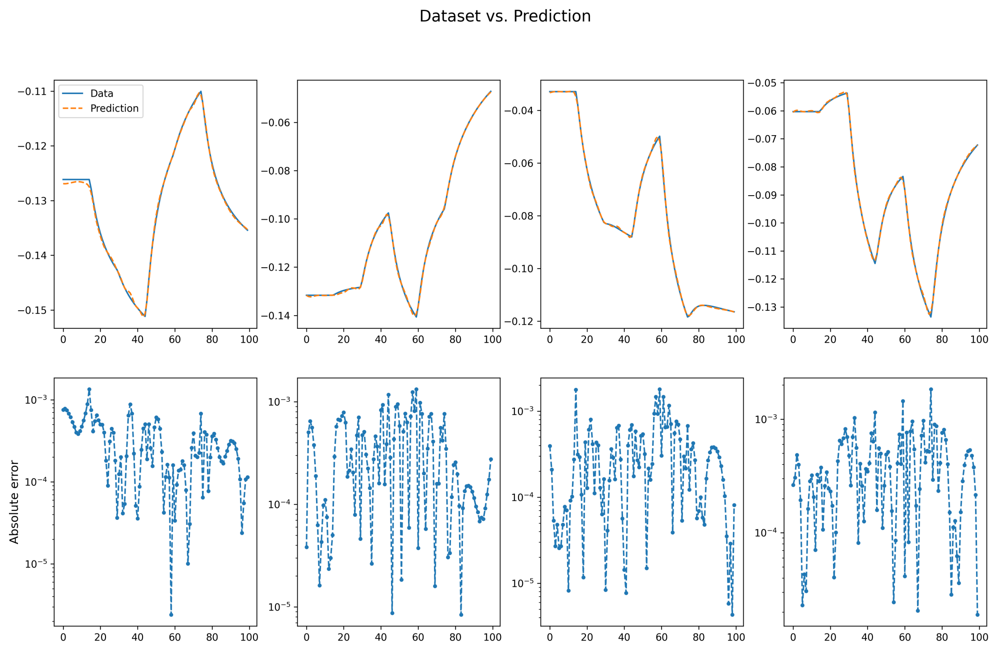


Figure 4.20: Performance of the model evaluated at point (12,15) in the 2D domain.The first row illustrates the comparison between predictions and targets, showcasing hydraulic head in the y-axis and time in the t-axis for four distinct test cases (columns). The second row displays the corresponding absolute error in the y-axis across time (t) for the same four test cases.

4.D Appendix D: Comparing neural network models

In this section, we compare the accuracy of the predicted solution for all the experiments except the time-dependent example ($E3$) obtained using DeepONet and the Fourier neural operator (FNO), when both models are evaluated after training converges. For time-dependent cases, we conduct a separate comparative analysis against Long Short-Term Memory (LSTM) networks. FNO, as introduced by Li et al. [22], leverages the Fast Fourier Transform for the direct parametrization of the integral kernel in Fourier space. We adhere to the FNO implementation for the 2D Darcy Flow problem as detailed in [22].

As depicted in Figure 4.21, the vanilla DeepONet achieves the best predictive accuracy in every case except for the forward problem for varying well locations, indicating the need for architecture modifications for this particular scenario. It's noteworthy that DeepONet, whose architecture is described in 4.3.3, has significantly fewer parameters (approximately 3.84×10^5) compared to FNO (1.19×10^6), which consists of four Fourier layers and Gelu activation. Yet, DeepONet exhibits robust performance across simulations. However, the comparison does not account for tuning effort across the models. It's plausible that with more tuning effort, FNO might present improved results.

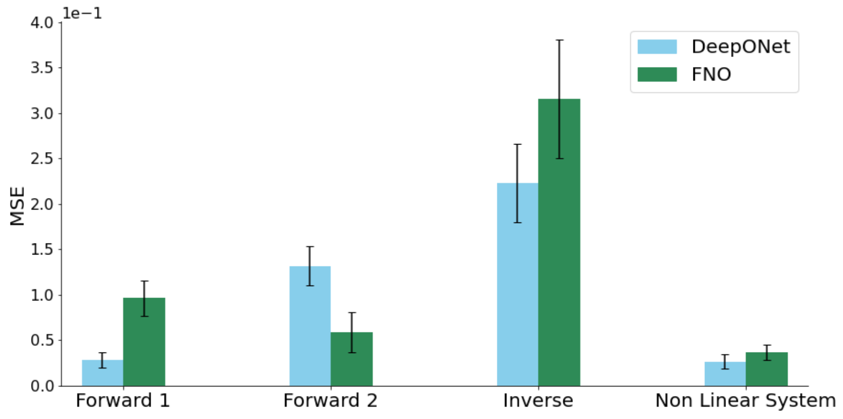


Figure 4.21: Comparison of mean squared error (MSE) for DeepONet and FNO across the time-independent different test cases. Forward 1 refers to the forward problem for fixed well location, while Forward 2 to the forward problem for varying well locations.

For the time-dependent scenarios, we shift our comparison to Long Short-Term Memory (LSTM) networks. In the first case, we consider the hydraulic head response at a single point. DeepONet demonstrates remarkable performance, producing average Mean Squared Error (MSE) training and testing errors of 5.8×10^{-8} and 6.0×10^{-9} respectively, as detailed in 4.C and with 30,800 trainable parameters. In contrast, the LSTM network employed in this case is designed with an architecture featuring an input layer followed by an LSTM layer with 60 units. A TimeDistributed Dense layer is then used to yield the output. This LSTM network displays an MSE error of 2.7×10^{-6} for training and 1.8×10^{-5} for testing, with 14,900 trainable parameters. In the second scenario, we extrapolate our analysis to the 2D domain with a single pumping well. The DeepONet model's objective is to predict the hydraulic head distribution across the domain in response to the time-dependent pumping well, as described in Section 4.4.3. The average MSE training and testing errors in this case are 1.1×10^{-7} and 1.0×10^{-7}, respectively. In comparison, we also employ an LSTM model, whose architecture consists of an input layer, and an LSTM layer with 256 units, followed by a permutation of dimensions and a reshaping step. The final output is produced using a Time Distributed Conv2DTranspose layer. This LSTM model yields an MSE error of 3.34×10^{-5} for training and 3.9×10^{-5} for testing, with 354,292 trainable parameters. In both scenarios, the DeepONet architecture clearly outperforms the LSTM networks in terms of prediction accuracy, establishing its superior efficacy in handling the cases in question.

4.E Appendix E: SVD analysis of training data

The lower accuracy that the vanilla DeepONet shows when approximating the operator in the case in which the source term can appear at any location is traceable to a limitation of the model in dealing with symmetries such as translation, rotation, and stretching. Venturi [35] describes DeepONet as a linear projection-based method, presenting the same inefficiency of singular value decomposition (SVD) when objects translate, rotate, or scale [30]. By showing that the trunk retrieves the modes of the projection, Venturi recommends employing SVD on the training data. This approach can help design the trunk net and its number of outputs during an exploratory phase.

Performing SVD on the output solutions gives the number of modes that are required to capture the variance or energy in the dataset. Each output solution of the labeled dataset, with dimension 32×32, is reshaped into a column vector; all the resulting vectors are then stacked horizontally as columns in the matrix $\mathbf{X}$. Figure 4.22 shows the results of the SVD analysis on the matrix $\mathbf{X}$ of datasets of the first two test cases. When the pumping well is fixed at the center of the domain, the energy contained in the first mode is equal to 85%. Its value decreases remarkably when the pumping well can be placed at any location: the first mode captures just 24% of the energy of the original datasets and the singular values produced by SVD decay slowly. The results suggest that the SVD fails as it doesn't account for the translating nature of the data. We conclude that this is also the reason why vanilla DeepONet cannot accurately predict the solution for unseen well locations.

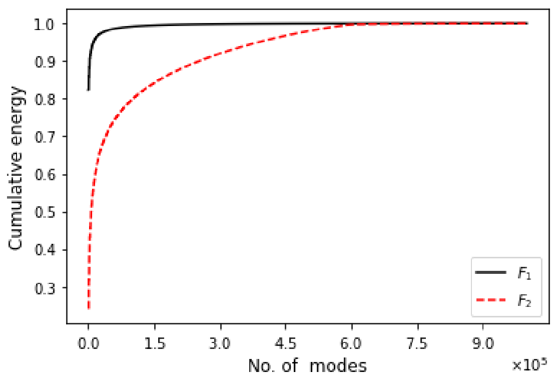


Figure 4.22: SVD on the output functions of the two data-set F_1 of experiment **E1** and F_2 of experiment **E2**: 1) the pumping well is placed in the same location of the domain across the whole training and testing datasets, 2) the pumping well can be placed at any location.

References

[1] Yohai Bar-Sinai et al. “Learning data-driven discretizations for partial differential equations”. In: *Proceedings of the National Academy of Sciences* 116.31 (2019), pp. 15344–15349.

[2] Qianying Cao et al. “Deep neural operators can predict the real-time response of floating offshore structures under irregular waves”. In: *arXiv preprint arXiv:2302.06667* (2023).

[3] Tianping Chen and Hong Chen. “Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems”. In: *IEEE transactions on neural networks* 6 4 (1995), pp. 911–7.

[4] Joeri van Engelen, Joost Delsman, and Huite Bootsma. *Reproducible large-scale groundwater modelling projects using the iMOD Python package.* Tech. rep. Copernicus Meetings, 2022.

[5] Rolf Farrell et al. “Splicing recharge and groundwater flow models in the Environment Agency National Groundwater Modelling System”. In: *Geological Society, London, Special Publications* 408.1 (2017), pp. 55–69.

[6] *Finite Difference Modeling course for groundwater flow.* `https://github.com/Olsthoorn/FDModelBuildingCourseGroundwaterFlow/`. Accessed: 2022-06-22. 2017.

[7] Somdatta Goswami et al. “A physics-informed variational DeepONet for predicting crack path in quasi-brittle materials”. In: *Computer Methods in Applied Mechanics and Engineering* 391 (2022), p. 114587.

[8] Somdatta Goswami et al. “A physics-informed variational DeepONet for predicting the crack path in brittle materials”. In: *arXiv preprint arXiv:2108.06905* (2021).

[9] Somdatta Goswami et al. “Deep transfer operator learning for partial differential equations under conditional shift”. In: *Nature Machine Intelligence* (2022), pp. 1–10.

[10] Somdatta Goswami et al. “Learning stiff chemical kinetics using extended deep neural operators”. In: *arXiv preprint arXiv:2302.12645* (2023).

[11] Somdatta Goswami et al. “Neural operator learning of heterogeneous mechanobiological insults contributing to aortic aneurysms”. In: *Journal of the Royal Society Interface* 19.193 (2022), p. 20220410.

[12] Patrik Simon Hadorn. "Shift-DeepONet: Extending Deep Operator Networks for Discontinuous Output Functions". en. MA thesis. Zurich: ETH Zurich, 2022-03-16. DOI: 10.3929/ethz-b-000539793.

[13] J. D. Hughes et al. "The MODFLOW Application Programming Interface for simulation control and software interoperability". In: *Environmental Modelling & Software* 148 (2022).

[14] Joseph D Hughes et al. "The MODFLOW Application Programming Interface for simulation control and software interoperability". In: *Environmental Modelling & Software* 148 (2022), p. 105257.

[15] Joongoo Jeon and Sung Joong Kim. "FVM Network to reduce computational cost of CFD simulation". In: *arXiv preprint arXiv:2105.03332* (2021).

[16] Pengzhan Jin, Shuai Meng, and Lu Lu. "MIONet: Learning multiple-input operators via tensor product". In: *SIAM Journal on Scientific Computing* 44.6 (2022), A3490–A3514.

[17] Adar Kahana et al. "On the geometry transferability of the hybrid iterative numerical solver for differential equations". In: *Computational Mechanics* (2023), pp. 1–14.

[18] George Em Karniadakis et al. "Physics-informed machine learning". In: *Nature Reviews Physics* 3.6 (2021), pp. 422–440.

[19] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization.* 2014. DOI: 10.48550/ARXIV.1412.6980. URL: https://arxiv.org/abs/1412.6980.

[20] Dmitrii Kochkov et al. "Machine learning–accelerated computational fluid dynamics". In: *Proceedings of the National Academy of Sciences* 118.21 (2021), e2101784118.

[21] Samuel Lanthaler, Siddhartha Mishra, and George E Karniadakis. "Error estimates for deeponets: A deep learning framework in infinite dimensions". In: *Transactions of Mathematics and Its Applications* 6.1 (2022), tnac001.

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

[23] Chensen Lin et al. "Operator learning for predicting multiscale bubble growth dynamics". In: *The Journal of Chemical Physics* 154.10 (2021), p. 104118.

[24] L. Lu et al. "Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators". In: *Nature Machine Intelligence* 3.3 (Mar. 2021. doi: https://doi.org/10.1038/s42256-021-00302-5).

[25] Lu Lu et al. “A comprehensive and fair comparison of two neural operators (with practical extensions) based on FAIR data”. In: *Computer Methods in Applied Mechanics and Engineering* 393 (Apr. 2022), p. 114778.

[26] Marjolein Mens et al. “Dilemmas in developing models for long-term drought risk management: The case of the National Water Model of the Netherlands”. In: *Environmental Modelling & Software* 143 (2021), p. 105100.

[27] Shaoxing Mo et al. “Deep autoregressive neural networks for high-dimensional inverse problems in groundwater contaminant source identification”. In: *Water Resources Research* 55.5 (2019), pp. 3856–3881.

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

[29] Vivek Oommen et al. “Learning two-phase microstructure evolution using neural operators and autoencoder architectures”. In: *arXiv preprint arXiv:2204.07230* (2022).

[30] Thomas Peters. “Data-driven science and engineering: machine learning, dynamical systems, and control”. In: *Contemporary Physics* 60.4 (2019), pp. 320–320. DOI: 10.1080/00107514.2019.1665103. eprint: https://doi.org/10.1080/00107514.2019.1665103. URL: https://doi.org/10.1080/00107514.2019.1665103.

[31] Esteban Samaniego et al. “An energy approach to the solution of partial differential equations in computational mechanics via machine learning: Concepts, implementation and applications”. In: *Computer Methods in Applied Mechanics and Engineering* 362 (2020), p. 112790.

[32] Alexander Y. Sun. “Discovering State-Parameter Mappings in Subsurface Models Using Generative Adversarial Networks”. In: *Geophysical Research Letters* 45.20 (Oct. 2018). DOI: 10.1029/2018gl080404. URL: https://doi.org/10.1029%2F2018gl080404.

[33] Maria Luisa Taccari et al. “Attention U-Net as a surrogate model for groundwater prediction”. In: *Advances in Water Resources* 163 (2022), p. 104169.

[34] Meng Tang, Yimin Liu, and Louis J. Durlofsky. “A deep-learning-based surrogate model for data assimilation in dynamic subsurface flow problems”. In: *Journal of Computational Physics* 413 (July 2020), p. 109456. DOI: 10.1016/j.jcp.2020.109456. URL: https://doi.org/10.1016%2Fj.jcp.2020.109456.

[35] Simone Venturi and Tiernan Casey. “SVD Perspectives for Augmenting DeepONet Flexibility and Interpretability”. In: *arXiv preprint arXiv:2204.12670* (2022).

[36] Sifan Wang, Mohamed Aziz Bhouri, and Paris Perdikaris. “Fast PDE-constrained optimization via self-supervised operator learning”. In: *arXiv preprint arXiv:2110.13297* (2021).

[37] Sifan Wang, Hanwen Wang, and Paris Perdikaris. “Learning the solution operator of parametric partial differential equations with physics-informed DeepONets”. In: *Science advances* 7.40 (2021), eabi8605.

[38] Zhi Zhong, Alexander Y Sun, and Hoonyoung Jeong. “Predicting CO2 plume migration in heterogeneous formations using conditional deep convolutional generative adversarial network”. In: *Water Resources Research* 55.7 (2019), pp. 5830–5851.

[39] Yinhao Zhu and Nicholas Zabaras. “Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification”. In: *Journal of Computational Physics* 366 (2018), pp. 415–447.

Chapter 5

Spatial-Temporal Graph Neural Networks for Groundwater Data

Abstract This paper introduces a novel application of spatial-temporal graph neural networks (ST-GNNs) to predict groundwater levels. Groundwater level prediction is inherently complex, influenced by various hydrological, meteorological, and anthropogenic factors. Traditional prediction models often struggle with the nonlinearity and non-stationary characteristics of groundwater data. Our study leverages the capabilities of ST-GNNs to address these challenges in the Overbetuwe area, Netherlands.

We utilize a comprehensive dataset encompassing 395 groundwater level time series and auxiliary data such as precipitation, evaporation, river stages, and pumping well data. The graph-based framework of our ST-GNN model facilitates the integration of spatial interconnectivity and temporal dynamics, capturing the complex interactions within the groundwater system. Our modified Multivariate Time Graph Neural Network model shows significant improvements over traditional methods, particularly in handling missing data and forecasting future groundwater levels with minimal bias. The model's performance is rigorously evaluated when trained and applied with both synthetic and measured data, demonstrating superior accuracy and robustness in long-term forecasting. The study's findings highlight the potential of ST-GNNs in environmental modeling, offering a significant step forward in predictive modeling of groundwater levels.

5.1 Introduction

The complexity of groundwater level (GWL) prediction and modeling arises from its nonlinearity and sensitivity to various hydrological, meteorological, and anthropogenic influences [17, 10]. Conventional GWL models, such as physical-based and traditional statistical methods, are limited by their high parameterization needs. This requirement makes the models computationally difficult to calibrate, as each parameter must be accurately estimated to reflect real-world conditions. Additionally, the models are challenged by computational intensity and difficulty in capturing the temporal evolution and nonlinearity in GWL data [5, 8, 21, 26]. In this context, deep learning models and, in particular, spatial-temporal graph neural networks (ST-GNNs) offer promising new avenues for accurate and efficient groundwater forecasting. Our study focuses on leveraging ST-GNNs to predict groundwater levels in the Overbetuwe area, Netherlands. The selection of this area is driven by the availability of data [4].

Machine learning methods, particularly data-driven ones, have been successfully applied in groundwater level prediction studies without requiring detailed physical process knowledge [25, 23, 24]. Recent works have observed an evolving landscape in groundwater level forecasting, shifting from conventional shallow networks to deep learning techniques [20]. Wunsch et al compare the effectiveness of shallow neural networks against deep learning models such as Long Short-Term Memory (LSTM) and Convolutional Neural Networks (CNNs). They emphasize their adaptability and efficiency with sparse training data and the potential to outperform with larger datasets [23]. However, these methods focus predominantly on temporal dynamics and often overlook the spatial relationships in GWL data, a gap that ST-GNNs are well-equipped to address. This gap in research is where this paper contributes.

ST-GNNs, rooted in graph theory and neural network methodologies, are particularly effective in datasets where spatial interconnectivity and temporal changes are significant. The ST-GNNs process inputs consisting of multivariate time series data, accompanied by a graph structure which delineates the interconnections between the variables within the multivariate time series. In these networks, spatial correlations among nodes are effectively represented through graph convolution techniques, while the temporal relations among past states are analyzed using recurrent neural network architectures [22, 27, 6, 18, 7, 22, 14, 12, 13]

While ST-GNNs have primarily been used in traffic prediction and skeleton-based action recognition, their application has recently extended to broader scientific fields, including meteorology [16, 15] and seismology [3]. This study harnesses the power of ST-GNNs to predict GWL in the Overbetuwe area, Netherlands. Unlike traditional methods, ST-GNNs adeptly handle unstructured spatial data and capture the complex interplay of spatial and temporal dynamics affecting GWL. The selection of ST-GNNs for this task is motivated by the distinct characteristics of the data and the specific challenges they introduce to modeling. Firstly, the nature of groundwater data, characterized by its complex spatial and temporal interactions, aligns well with the capabilities of spatial-temporal graph modeling. This approach offers the necessary granularity and flexibility to accurately model the intricate relationships within multivariate time series data. Secondly, the data encountered in this study is inherently hybrid, containing both continuous and discrete elements. This heterogeneity requires a modeling solution that can adeptly manage such diverse data types. Furthermore, the complexities of real-world data pose significant challenges to accurate modeling and forecasting of groundwater levels. Notably, the presence of noise and instances of missing data, which span both spatial and temporal dimensions, can obscure patterns and trends, making it difficult to identify precise relationships. These conditions necessitate a modeling strategy capable of discerning correlations between observations at various locations and times, yet the reliance on black-box learning methods is constrained by the limited quantity of data available. Therefore, the deployment of ST-GNN, infused with a degree of prior knowledge, emerges as a requisite strategy. By incorporating control variables like pumping rates and considering multiple aquifers, the proposed ST-GNN model extends the applicability of GNNs in hydrological forecasting beyond current capabilities, addressing challenges such as missing data and ensuring model generalizability.

One recent study that does address spatial relationships in groundwater forecasting is by Bai et al. [1]. In their work, they employ a graph neural network to forecast groundwater dynamics in the southwest area of British Columbia, Canada. Bai et al.'s model demonstrates superior performance compared to baseline models like LSTM [11] and Gated recurrent unit (GRU) [9]. However, there are notable differences between Bai et al.'s approach and our study. While Bai et al. focus on groundwater dynamics in a region predominantly influenced by rainfall and characterized by high seasonality in groundwater level (GWL) fluctuations, our research extends to the Overbetuwe area, Netherlands, which does not exhibit such pronounced seasonal patterns. Our study area is influenced by a broader range of hydrological, meteorological, and anthropogenic factors, including the effects of pumping wells and river stages, contributing to more complex and nuanced groundwater behavior. Moreover, our choices of GNN architectures and predefined adjacency matrix contrast with Bai et al.'s use of Graph WaveNet and a self-adaptive adjacency matrix. Our model is tailored to our unique dataset and the specific challenges it presents, including the integration of control variables such as pumping rates, and the definition of the graph structure to represent the differet hydrological connections. Lastly, a critical advancement in our methodology, compared to Bai et al., is our approach to handling data gaps. Bai et al. exclude wells with gaps larger than one month and interpolate smaller gaps, potentially overlooking valuable information. Conversely, we introduce a masking strategy to track the locations and times of missing values, seamlessly integrating this information into our loss function to avoid training on these gaps, thereby enhancing model accuracy and reliability.

This study goes beyond traditional forecasting by exploring the ST-GNN model's utility in scenario planning and decision-making for drinking water extraction, considering varying weather condition scenarios months in advance. This capability is pivotal for managing water resources sustainably, ensuring that the extraction volumes are optimized to prevent drought conditions. Furthermore, the deep learning model supports rapid computational capabilities. This feature enables the quick generation and simulation of multiple scenarios, making it an invaluable resource for strategic planning and promoting efficient water usage.

The paper is structured as follows: following the introduction, we describe our methodology, including data description, preprocessing, and model architecture in Section 5.2. Section 5.3 presents the results, including an abation study and a comparative analysis with the state-of-the-art traditional numerical model. Finally, we conclude in Section 5.4 with a discussion on the implications of our findings for hydrological forecasting and future research directions.

5.2 Methodology

This section outlines the methodology for predicting groundwater levels using ST-GNNs. It starts with a detailed description of data from the Overbetuwe area in the Netherlands, focusing on hydrological, meteorological, and anthropogenic factors affecting groundwater levels. The preprocessing steps for model training are then presented, followed by an explanation of a graph-based framework designed to capture the complex spatial-temporal relationships in groundwater data. Finally, it discusses training strategies and error metrics to optimize model performance and accurately evaluate its predictive capabilities.

5.2.1 Data Description

The study focuses on the Overbetuwe area in the Netherlands, a polder region approximately 30 km by 10 km in size, flanked by two branches of the Rhine river. The land surface elevation varies from around +10 m NAP (Amsterdam Ordnance Datum) in the east to around +7 m NAP in the west. The shallow subsurface is characterized by a low-permeable phreatic layer, underlain by two aquifers separated by an aquitard. The groundwater is relatively shallow, with the depth to the water table varying between 0.8 and 4.2 m. For a more comprehensive understanding of the hydrological system, the reader is referred to the work of [4].

The study encompasses data from 213 observation wells, also called piezometers, yielding a total of 395 groundwater level time series. Some of these time series originate from the same geographical location (but differing in terms of depth). While each time series begins in a different year, some starting as early as the 1950s, the frequency and availability of data have evolved significantly over time. Initially, the data collection for these time series was on a monthly basis. However, the introduction of automatic loggers at the turn of the new millennium enabled an increase in frequency to daily or even more frequent recordings. Figure **5.1** visually showcases the distribution and types of sensors across the Overbetuwe area.

In addition to the groundwater levels from the observation wells, which represent the primary variable of interest, other types of time series data are recorded as exogenous variables: precipitation, evaporation, river stages, and pumping wells. Of these, only the pumping well serves as the control variable. Daily precipitation data is available from seven measurement stations, and evaporation data from two weather stations. Additionally, river stage measurements are taken every 10 minutes, and drinking water extraction data is gathered daily from 4 stations. The metadata for each of these time series includes location coordinates and depth where applicable.

5.2.2 Data Preprocessing

In addressing the critical challenges of data sparsity and noise inherent in the dataset, the preprocessing strategy lays a foundational stone for the subsequent analysis and modeling phases of this study. The complexity introduced by noisy data and sparse datasets—characterized by

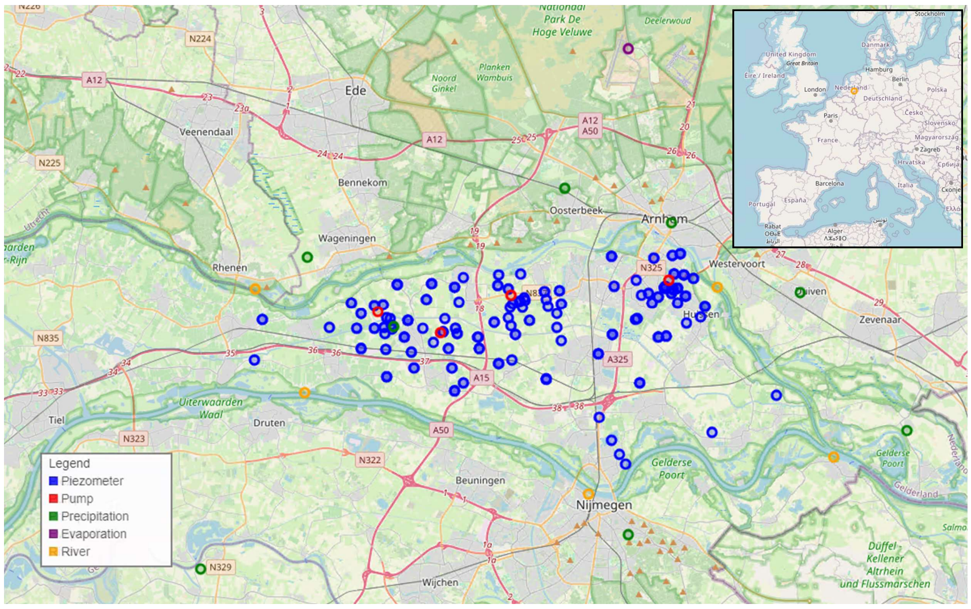


Figure 5.1: Illustration of the sensor network. Different types of sensors are represented by distinct colors, providing an overview of the spatial distribution and categorization of each sensor within the study area. Notably, one evaporation sensor is located outside the depicted map area. Additionally, the precise locations of some observation points may not be distinctly visible, as certain sensors share the same coordinates but are situated at different depths, and others are too closely positioned to be distinguished on the map.

significant gaps and intermittent recordings—necessitate a rigorous approach to data cleaning and selection. Refining the dataset to mitigate these challenges, enhances the reliability of the analysis and establishes a robust basis for the modeling phase. The approach to preprocessing not only addresses the immediate issues of data quality and completeness but also serves as a crucial contribution of this study, illustrating the significance of a well-considered preprocessing phase for robust forecasting results.

The starting point for this analysis is established as the year 2004. This choice rests on two key factors: the comprehensive availability of time series data from observation wells starting from this year, and the significant portion of these data recorded on at least a daily basis. To balance maintaining sufficient data and managing its variability, and considering the relatively slow daily variations, the data frequency is adjusted to a weekly basis, with values averaged.

Out of the 395 available time series, 200 are selected, prioritizing those with the most complete data. Initially, the overall percentage of missing values for the observation wells stands at 29.3%; this selection reduces the overall percentage of missing values to 8.1%. Missing values undergo linear interpolation, and a mask is introduced to track the locations and times of these missing values. This mask proves instrumental in the loss function, ensuring that these points are not used for training. For the river stage data, only five locations with no missing data are retained: Dodewaard, IJsselkop, Nijmegen haven, Grebbe, and Pannerdense Kop. The other types of measurements exhibit no missing data issues.

All data are rescaled within the range (0, 1). The dataset is chronologically divided, with 80% allocated for training starting from 2004-01-04. The remaining 20% is used for both validation and testing, starting from 2018-04-01. Due to limited data availability, there is no split between validation and test sets.

5.2.3 Graph-Based Framework in Groundwater Level Forecasting

In this study, a deep learning graph-based approach is adopted to forecast groundwater levels in the Overbetuwe area. Each measurement, whether from observation wells (the subjects of the forecasting task), precipitation, or other hydrological data points, represents a node in a graph. The values at each time step t are denoted by $\mathbf{z}_t \in \mathbf{R}^{N'}$, where N' is the number of time series of groundwater levels and $z_t[i] \in \mathbb{R}$ represents the specific measurement value at that time step.

Objective The primary goal is to predict future groundwater levels using historical observations within an input window of W time steps, leading up to time step P. This historical data is represented as $\mathbf{X} = \{\mathbf{z}_{t_{P-W+1}}, \mathbf{z}_{t_{P-W+2}}, \ldots, \mathbf{z}_{t_P}\}$. The aim is to forecast a sequence of future values over a forecasting window of Q time steps, denoted as $\mathbf{Y} = \{\mathbf{z}_{t_{P+1}}, \mathbf{z}_{t_{P+2}}, \ldots, \mathbf{z}_{t_{P+Q}}\}$.

Incorporating Auxiliary Data To enhance the model's predictive capability, auxiliary features are integrated including precipitation, evaporation, river stages, and pumping well data, with their number of time series being N''. These features are considered up to and including the forecasting window t_{P+Q}: $\mathbf{X}' = \{\mathbf{s}'_{t_{P-W+1}}, \mathbf{s}'_{t_{P-W+2}}, \ldots, \mathbf{s}'_{t_{P+Q}}\}$, where each $\mathbf{s}'_{t_i} \in \mathbf{R}^{N''}$ contains the auxiliary data at time step t_i. This auxiliary dataset $\mathbf{X}'$ then concatenates with the historical input data $\mathbf{X}$, which only includes observations from time step t_{P-W+1} to t_P. Padding applies to align the different lengths of historical and auxiliary data for concatenation. In practical applications, while exogenous variables such as precipitation and river stages will be forecasted (or modelled), the pumping well data is treated as a controllable variable. The approach constructs a mapping function $f : (\mathbf{X}, \mathbf{X}') \to \mathbf{Y}$, where $\mathbf{X}$ includes the historical data up to time step t_P, and $\mathbf{X}'$ comprises the extended auxiliary data up to t_{P+Q}, with the goal of predicting future values $\mathbf{Y}$.

Graph Definition This study conceptualizes a graph $G = (V, E)$ to represent the hydrological system. V signifies the set of nodes, each corresponding to a distinct hydrological measurement, while E encapsulates the connections or relationships among these measurements. The total number of nodes in the graph N is the sum of two subsets: N' representing groundwater level measurements and N'' comprising other hydrological factors. Forecasting efforts focus on N', utilizing both historical groundwater levels and various exogenous variables.

For each node $v \in V$, corresponding to a specific hydrological measurement or time series, its neighborhood $N(v)$ is defined. This neighborhood consists of other measurements that are hydrologically interconnected or influenced by v. Each piezometer is linked to its three closest counterparts, determined by Euclidean distance, a choice that balances the graph's connectivity without making it too sparse or overly dense, as established through an ablation study presented in 5.3.1. Furthermore, each piezometer establishes connections with all pumping locations, the nearest precipitation and evaporation stations, and the two closest river measurement points. The adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ where $N{=}N'{+}N''$ represents these hydrological connections. In this matrix, an entry A_{ij} assigns a non-zero value if there is a connection between nodes i and j, and zero if no such relationship exists. The assigned values in the adjacency matrix, ranging from 0.1 to 0.5, differentiate the types of hydrological connections, such as those between piezometers, between piezometers and pumping wells, and so on.

5.2.4 Model Architecture

A modified version of the Multivariate Time Graph Neural Network (MTGNN) model, originally described by [22], is adapted in this study. This adaptation omits the graph learning layer, favoring predefined graph structures which we found to yield significantly improved results.

The architecture, visualized in Figure 5.2, demonstrates the sequential processing of data through the model. Input data is first subjected to a starting convolution operation, after which it progresses through multiple graph and temporal convolution modules, designed to capture spatial and temporal dependencies, respectively. These modules are interconnected by skip connections, enhancing the model's ability to preserve information across layers. The graph convolution module operates on spatial relationships by aggregating node information with neighboring nodes, leveraging mix-hop propagation layers for this purpose. Meanwhile, the temporal convolution module employs dilated 1D convolution filters to analyze temporal patterns, utilizing a combination of filter and gating layers to modulate the flow of information. Finally, the output module itself consists of two 1x1 convolution layers, adjusting the channel dimension of the input to meet the desired output dimension.

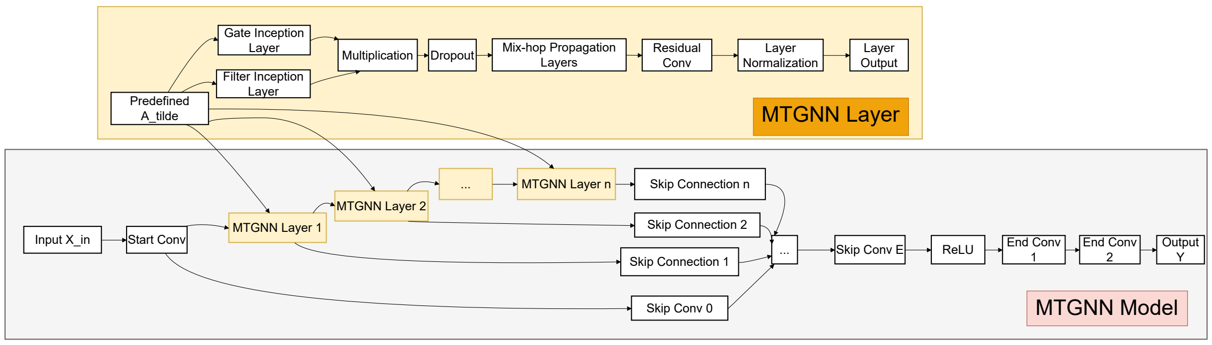


Figure 5.2: The model architecture includes both graph and temporal convolution modules, highlighted by their sequential processing and integration of skip connections to facilitate data flow and information preservation across layers.

For a more detailed description of the architecture, readers are referred to the original article [22], while the subsequent sections of this paper will only briefly present the temporal and spatial modules, highlighting the innovations in this work.

Graph Convolution Module

The graph convolution modules address spatial dependencies by aggregating information from each node with its neighbors. This involves two mix-hop propagation layers, which facilitate the inflow and outflow of information at each node [22]. The mix-hop propagation layer, given by Equation 5.1 and Equation 5.2, handles the spatial information flow across nodes in the network:

$$\mathbf{H}^{(k)} = \beta \mathbf{H}_{in} + (1-\beta)\tilde{\mathbf{A}}\mathbf{H}^{(k-1)}, \tag{5.1}$$

$$\mathbf{H}_{out} = \sum_{k=0}^{K} \mathbf{H}^{(k)}\mathbf{W}^{(k)}. \tag{5.2}$$

In Equation 5.1, $\mathbf{H}^{(k)}$ represents the hidden states at the k-th propagation step, β is a hyper-parameter controlling the retention ratio of the root node's original states, $\mathbf{H}_{in}$ denotes the input hidden states from the previous layer, $\tilde{\mathbf{A}}$ is the normalized adjacency matrix including self-connections, and $\mathbf{H}^{(k-1)}$ are the hidden states from the $(k-1)$-th propagation step. In Equation 5.2, $\mathbf{H}_{out}$ represents the output hidden states of the current layer, K is the depth of propagation, and $\mathbf{W}^{(k)}$ is the parameter matrix acting as a feature selector at each propagation step.

Temporal Convolution Module

The temporal convolution modules focus on capturing the temporal dynamics within the data. These modules utilize standard dilated 1D convolution filters, arranged in two layers: a filter layer followed by a tanh activation function and a gating layer with a sigmoid activation function [22]. The temporal convolution module utilizes dilated 1D convolution filters:

$$\mathbf{z} = \text{concat}(\mathbf{z} \star \mathbf{f}_{1\times 1}, \mathbf{z} \star \mathbf{f}_{1\times 2}), \tag{5.3}$$

where $\mathbf{z}$ is the input 1D sequence, $\mathbf{f}_{1\times k}$ are the convolution filters of varying sizes and $\mathbf{z} \star \mathbf{f}_{1\times k}$ represents the dilated convolution operation. This study deviates from the original model's filter sizes of 1×2, 1×3, 1×6, and 1×7, whose combinations were designed to capture a variety of inherent periods typical of temporal signals. Given our data is measured in weeks without significant seasonal fluctuations, this broad spectrum of filter sizes was not deemed necessary and the choice of the filters is based on an ablation study presented in 5.3.1. Dilated convolution introduces "gaps" in the convolution kernel to extend its coverage on the input feature map without increasing parameters or computation. This efficiently broadens the receptive field, enabling the incorporation of wider input information without escalating the model's parameter count. The outputs of convolution operations across different filter sizes are concatenated by the concat function. Finally, the skip connection layer and output module, which standardize and transform the data for final output, follow the temporal and spatial modules.

5.2.5 Training Approach

Our model harnesses historical observations to forecast future groundwater levels. It adopts a recursive forecasting approach, wherein the groundwater level prediction at time $t + 1$ serves as an input for the subsequent time step $t + 2$, thereby recursively using forecasted outputs as historical inputs when the prediction window exceeds a single time step. This recursive strategy is enhanced by incorporating exogenous variables, which, although they might originate from forecasts of different models or be based on scenarios to be considered, are given as inputs for the next time step without being forecasted themselves.

Furthermore, the training regime involves adjusting the size of the prediction window dynamically, thus tailoring the model for different forecasting scenarios. The training employs an iterative process that gradually extends the prediction window. Specifically, our model utilizes a past window size W of 5 time steps, with each time step representing a week, to incorporate recent observations and a future prediction window F of up to 8 time steps for projecting imminent trends. Through this methodology, the model concurrently optimizes forecasts across the entire prediction horizon during training, ensuring a comprehensive learning process.

The loss function, as defined in Equation 5.4, is designed to accommodate missing data within the time series through a masked mean squared error method. The $\text{mask}_{i,t}$ is a binary indicator where the value is set to 0 when data is missing and 1 otherwise. This binary mask ensures that while the missing points are linearly interpolated and used as inputs, they do not contribute to the loss calculation, thereby preserving the integrity of the model's training process. In this way, the model concentrates on accurately predicting the available data points, while effectively disregarding the segments with missing or unreliable data.

$$\text{Loss} = \sum_{i=1}^{B} \sum_{t=1}^{F} \text{mask}_{i,t} \cdot (y_{i,t} - \hat{y}_{i,t})^2, \tag{5.4}$$

Here, B represents the batch size and F is the length of the future prediction window. The actual observed values are denoted by $y_{i,t}$, and the model's predicted values are represented by $\hat{y}_{i,t}$. The use of the mask in the loss function ensures that the model is trained primarily on the robust data points, providing a reliable performance metric that truly reflects the model's forecasting capabilities.

To quantify the accuracy of our forecasting model, we utilize the Root Mean Square Error (RMSE) as the primary error metrics. The RMSE, detailed in Equation 5.5, quantifies the forecast error's magnitude and indicates the average deviation between the model's predictions and the actual observed values.

$$\text{RMSE} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(y_i - \hat{y}_i)^2}, \tag{5.5}$$

In Equation 5.5, M denotes the number of forecasts, y_i the actual observed values, and $\hat{y}_i$ the values predicted by the model. A lower RMSE value indicates a more accurate model.

5.3 Results

This section presents a rigorous evaluation of the deep learning model's efficacy in long-term groundwater level forecasting. The first part presents an ablation study, analyzing the influence of various model components and parameters on the forecasting results. Subsequently, the model's efficacy is evaluated on real-world data, with performance comparisons drawn against the traditional numerical MODFLOW model.

5.3.1 Ablation Study

This section explores various model configurations and parameters with the goal of assessing the impact of these variables on forecasting accuracy, quantified by the Root Mean Square Error (RMSE). A key focus is the evaluation of spatial and temporal convolution modules, alongside the influence of graph parameters on model performance. In the ablation study, using a historical window size W of 5 time steps, each corresponding to one week, training is confined to forecasts up to only 2 weeks ahead to expedite the processing of numerous model configurations. During testing, the model extends its forecasting to a future prediction window F of 100 weeks.

This approach demonstrates a notable improvement in performance when using synthetic data (see below for details), for which the ST-GNN model secures a baseline RMSE of 8.13, in contrast to RMSE of 16.9 obtained with real data. This underscores the utility of synthetic data in honing forecast accuracy and in refining the model parameters during the ablation study. The increased error observed with real data can likely be attributed to the inherent noise present in such real-world data. Subsequent studies could explore the estimation of this noise and its introduction into synthetic datasets to achieve a more authentic representation of measured data, particularly when such data are insufficient for training deep learning models.

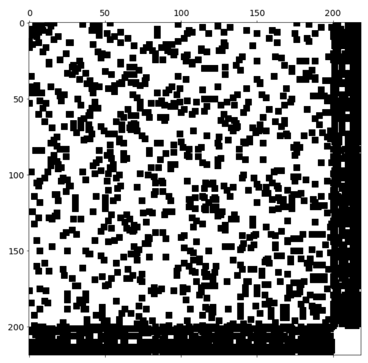
The synthetic dataset is derived from the MODFLOW groundwater model for the Overbetuwe area, a component of the larger MORIA model [19], which represents a collaborative effort involving several organizations, including provinces, waterboards, and drinking water companies. The model leverages a comprehensive range of input data to simulate groundwater dynamics accurately, including subsurface schematization through REGIS II.2 [19]. It accounts for the actual meteorological conditions and known groundwater abstractions during the period in question, facilitating a balanced comparison with the real-world data. Additionally, it incorporates factors like anisotropy, drainage, boundary conditions, initial heads, and storage coefficients. The dataset includes over 800 observation wells and spans from 2008 to 2019, with daily recordings. To ensure a consistent evaluation, the dataset consists of 188 observation wells that have both simulated and real-world data. The unavailability of simulated data for 12 out of the original 200 wells leads to selecting alternative wells with real data, which are not chosen among the real-world observations due to their higher rates of missing data. Although the dataset includes spatial coordinates, depth, and node type as static features, certain aspects like the top and bottom elevations of aquifers and hydraulic conductivity for aquifers are not directly used in model training. Instead, they are considered intrinsic properties learned by the GNN.

Furthermore, the ablation study assesses the performance of different hyperparameters and model configurations, beginning with the examination of the utility of both the temporal and graph convolutional modules. Initially, replacing each dilated 1D convolution layer in the temporal module with an LSTM, followed by a fully connected layer, results in an increased error, raising the RMSE to 34.6. Removing the graph convolution module from the model incurs a pronounced increase in RMSE to 31.0 as well, clearly illustrating the vital contribution of both modules to maintaining accuracy.

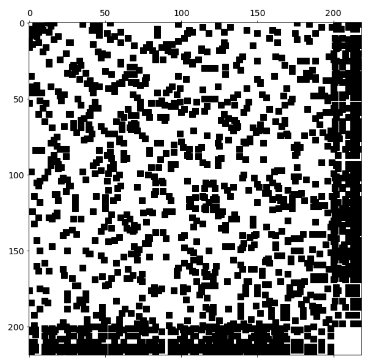
The network graph's definition is also shown to be crucial for performance. The series of adjacency matrices depicted in Figure 5.3 illustrates various connectivity levels among observation wells and exogenous variables. Figure 5.3a illustrates the proposed network configuration, establishing connections for each well to its three nearest counterparts, all pumping sites, the closest precipitation and evaporation monitoring stations, and the two nearest river gauging points. Within the adjacency matrix, the values assigned to these connections range from 0.1 to 0.5, reflecting the varying types of hydrological interactions. Conversely, setting all non-zero values in the matrix to 1 leads to a significant error increase of approximately 112.75%. This change highlights the critical role of detailed connectivity, affirming that this configuration yields a balanced and empirically validated network structure. The matrix's denser lower segments reveal more intense connections with external variables. By contrast, Figure 5.3b, which features only half of the observation wells—randomly selected—connected to exogenous variables, results in a more sparse matrix and an elevated RMSE of 15.0. This underscores the importance of linking all observation wells to exogenous variables. The minimal connectivity scenario in Figure 5.3c, where each well is linked to only its nearest neighbor, results in the sparsest matrix and an increased error of 10.3. Similarly, connecting each well to four of its neighbors results in a higher error of 10.7. Enabling the network to learn the adjacency matrix, as described by Wu et al. [22], resulted in a higher error of 10.90. In contrast, employing the predefined adjacency matrix with the proposed connectivity achieved a 25.4% improvement in RMSE compared to allowing the network to learn the adjacency matrix. This indicates that while the strategy of allowing the network to autonomously learn the adjacency matrix can be beneficial initially, it does not lead to the optimal outcome when compared to the adjustments made to address the deficiency.

The ablation study further evaluates the model's sensitivity to a range of hyperparameters, encompassing those impacting both temporal and spatial convolutional layers, with the aim of optimizing forecast accuracy. In the finalized implementation of the ST-GNN model, the depth of the graph convolution is set to four layers, with kernel sizes selected as 1 and 2. To prevent overfitting, a dropout rate of 50% is implemented. Moreover, the model employs a β value of 0.05 to preserve the original states of the root nodes.

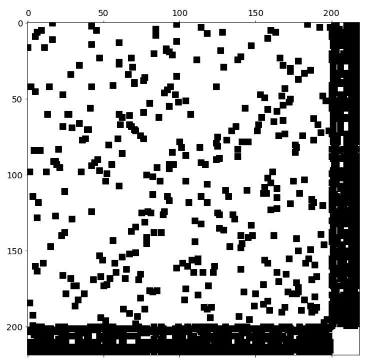
Finally, the impact of the multi-step-ahead function on the model's training efficacy is investigated. Drawing inspiration from the approach taken by [2], the effect of extending the forecast horizon on model performance is examined. The results align with Bentivoglio et al.'s findings, showing that increasing the number of steps ahead consistently improves model accuracy. However, this enhancement comes with higher memory requirements and longer training times. Specifically, the duration needed for each training epoch increases from 2 seconds with a single forecast horizon to 93 seconds for 10 forecast windows, equating to a period of 10 weeks. An optimization analysis of the forecast horizon reveals that the model achieves notable performance within just a three-week timeframe. While further extension of the forecast horizon does yield improved outcomes, the rate of performance gain diminishes, nearly reaching a plateau beyond three weeks. Therefore, a six-week period is chosen for real-world data analysis in the



(a) Adjacency matrix with optimal connectivity



(b) Adjacency matrix with 50% connectivity to exogenous variables



(c) Adjacency matrix with minimal connectivity

Figure 5.3: Comparison of adjacency matrices representing different levels of connectivity in the modeled groundwater network. Figure 5.3a depicts the adjacency matrix corresponding to the best case scenario where each observation well is connected to its three nearest wells based on Euclidean distance. Additionally, connections extend to all pumping stations, the nearest precipitation and evaporation monitors, and the two proximal river measurement sites, explaining the increased density seen in the last 19 rows and columns beyond the 200 observation wells. Figures 5.3b and 5.3c illustrate matrices with reduced connectivity: the former with only half of the observation wells linked to exogenous variables, and the latter with each well connected to only its single nearest counterpart.

following section to strike an optimal balance between enhancing performance and managing the increased memory demands associated with a longer forecast horizon. This strategy ensures peak performance while maintaining computational efficiency.

5.3.2 Real-World Data Application

This section examines the application of the deep learning model to real-world datasets. Figure 5.4 presents a comparison between forecasts for four randomly selected observation wells and actual data that was not seen by the model during training, beginning in April 2018. The solid lines represent the actual measured groundwater levels, while the dashed lines depict the model's forecasts. This figure illustrates that the model closely aligns with the actual observed groundwater levels. Despite being trained to forecast up to 6 weeks ahead, the model demonstrates remarkable extrapolation capabilities for up to 100 weeks, as evidenced by its predictions, which closely follow the trends observed in the actual groundwater measurements without any significant error propagation or accumulation. In contrast to synthetic data, these real-world datasets include missing values. As indicated in the figure, where there is a prediction for a missing window in an observational well, it underscores the model's ability to predict for periods where data are unavailable. This highlights another potential application of the model for data infilling.

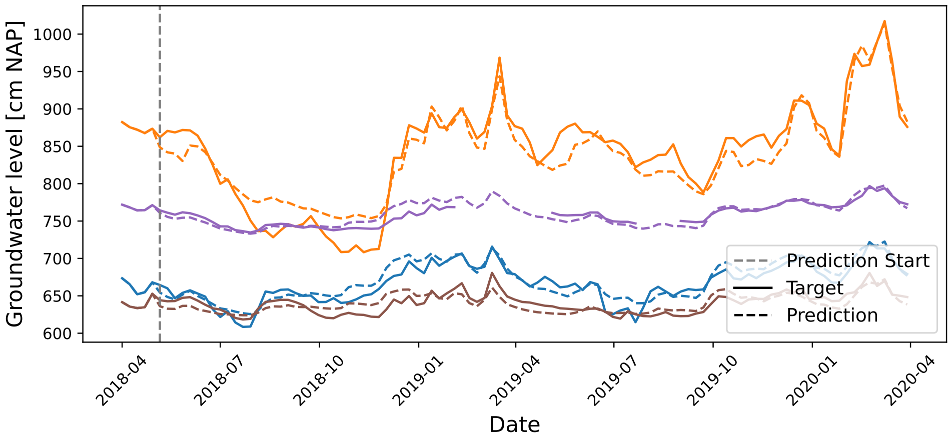


Figure 5.4: Comparative performance of the model's predictions over four randomly selected piezometers. The model utilizes a past window of 5 weeks and extends its forecasts to demonstrate its proficiency in capturing groundwater level trends over a longer horizon. Solid lines correspond to observed data, while dashed lines represent forecasts.

Figure 5.5 showcases two contrasting results, featuring selected wells with very high and very low RMSE values. The green lines depict the best testing outcome, which achieves an RMSE of 4.2; this result closely follows the target trajectory with only minor prediction errors that do not result in error amplification. Conversely, the red line represents a test sample with an RMSE of 53.4. In this scenario, a larger variation in groundwater levels is observed; however, the model still manages to forecast within this wider range. Although the trend is predicted, there is a notable discrepancy in the values towards the end. This sample ranks as the second worst, surpassed only by another with an RMSE of 67.6, which is not depicted here due to its relatively stationary trend. It is noteworthy that these worst two samples originate from the same site but at different depths, hinting at a potential issue within the system, possibly missing some external information. This suggests that the model could also serve as a tool for identifying such anomalies. Despite these findings, no specific pattern or geographical distribution of errors could be determined.

When benchmarked with real data, the traditional MODFLOW model exhibits an RMSE of 25.2, thus showcasing the enhanced precision of the deep learning model in the forecasting task. Figure 5.6 presents a scatter plot comparison of the RMSE values for the ST-GNN and MODFLOW models. In this plot, each data point corresponds to a unique piezometer in the test set, with the inclusion of an equality line serving as a reference to easily discern which model achieves a lower RMSE. Moreover, observation wells featured in Figures 5.4 and 5.5 are marked

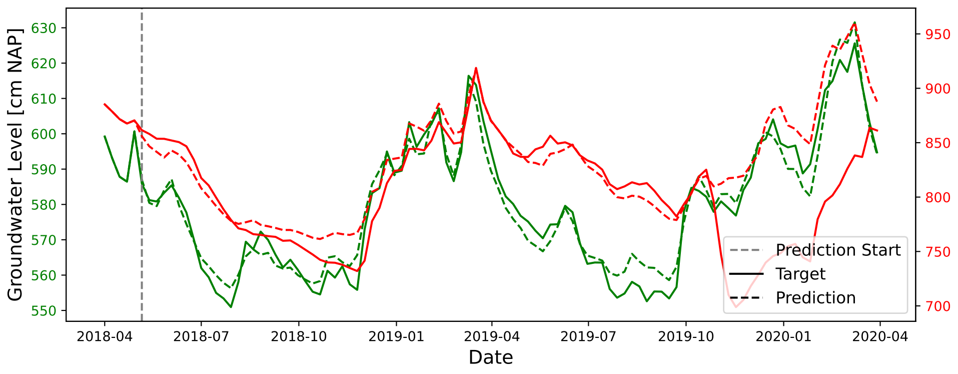


Figure 5.5: Comparative analysis of the model's forecasting accuracy for observation wells, highlighting those with the highest (red) and lowest (green) RMSE. Solid lines indicate observed data, while dashed lines denote forecasts.

with a star, with colors matching their depiction in the plots. Data points situated beneath the equality line indicate superior performance by the ST-GNN model. The aggregation of points below the equality line corroborates the ST-GNN model's consistent outperformance, validating its efficacy in predicting groundwater levels. It is noted that, with the exception of the observation well with the highest error, all other high-RMSE predictions, such as the one visualized in Figure 5.5, do not lie far from the equality line in Figure 5.6.

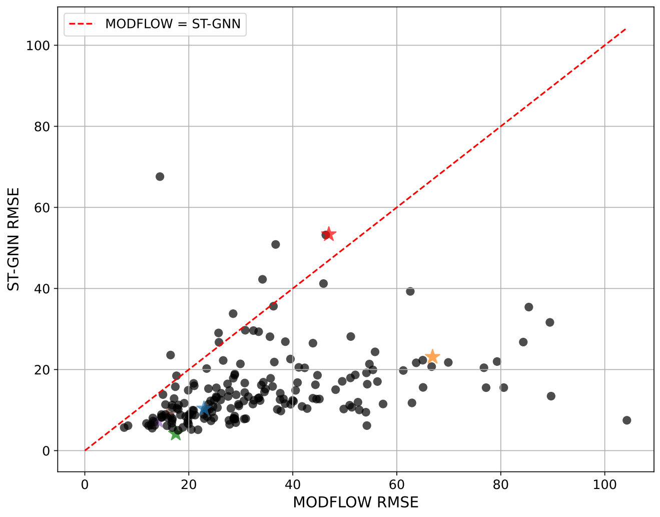


Figure 5.6: RMSE scatter plot comparison between the GNN and MODFLOW models, with the ST-GNN model predominantly achieving lower RMSE values. Stars and corresponding colors highlight the specific observation wells from Figures 5.4 and 5.5.

In conclusion, evaluating the ST-GNN model on real-world data showcases its robustness and reliability. The model yields more accurate forecasts than traditional MODFLOW models, showcasing the profound potential of deep learning in groundwater level forecasting.

5.4 Summary and Conclusions

This research has successfully demonstrated the application of spatial-temporal graph neural networks (ST-GNNs) for predicting groundwater levels in the Overbetuwe area of the Netherlands. The utilization of a comprehensive dataset, including 395 groundwater level time series and auxiliary data such as precipitation, evaporation, river stages, and pumping well data, has enabled the modeling of the spatial interconnectivity and temporal dynamics influencing groundwater systems. The graph-based framework of the ST-GNN model has exhibited remarkable capability in capturing these intricate interactions, resulting in high forecasting accuracy and robustness, despite the presence of noise in the real-world data and instances of missing data.

The performance evaluation of our model against traditional numerical models, such as MODFLOW, underscores the superior accuracy and predictive capabilities of ST-GNNs in long-term forecasting scenarios, particularly when forecasting in specific observation wells. The utility of the ST-GNN model extends beyond mere prediction, showing the potential to play a crucial role in scenario planning and decision-making for drinking water extraction under varying weather conditions. By enabling the simulations for various weather scenarios months in advance, the model aids in determining the optimal volume of water extraction to prevent drought conditions. The deep learning architecture of the model facilitates rapid computation, allowing for the quick execution of numerous probabilistic scenarios. This capability is potentially invaluable for water resource managers, offering a tool for strategic planning and sustainable water usage that is both flexible and efficient.

Future research could focus on enhancing the model's predictive accuracy by integrating additional variables into the model architecture, such as incorporating resistance as weights within the graph's edges. Explicitly including features such as the top and bottom elevations of aquifers, their hydraulic conductivity, and the resistance in aquitards within the dataset could significantly improve the model's performance. Moreover, the scalability of this approach to encompass larger networks of groundwater measurements and the integration of additional exogenous variables present promising avenues for research and application. The development of networks capable of processing multi-modal inputs, including continuous geological information, could further enhance the model's predictive accuracy and utility. Importantly, by incorporating additional continuous variables, it would be interesting to develop a model that can extrapolate and predict also in points that are not observation wells, thereby truly competing with traditional models which currently provide solutions in the whole domain.

In summary, this study contributes significantly to advancing the application of machine learning in hydrology, establishing the potential for ST-GNNs to become a a powerful tool for predictive modeling of groundwater levels.

5.E Appendix A: Table of notations

A table of notations is given in Table 5.1.

Symbol	Description
G	Graph representing the hydrological system
V	Set of nodes in graph G, representing hydrological measurements
E	Set of edges in graph G, representing relationships among measurements
N	Total number of nodes in graph G
N'	Number of time series of groundwater levels
N''	Number of time series of auxiliary hydrological factors
$\mathbf{A}$	Adjacency matrix representing connections in the hydrological network
$\mathbf{z}_t$	Measurement values at time step t
$\mathbf{X}$	Historical data used for training the model
$\mathbf{Y}$	Forecasted future values over a prediction window
W	Past window size in time steps
Q	Forecasting window size in time steps
F	Future prediction window size in time steps
$\mathbf{X}'$	Auxiliary data up to and including the forecasting window
$\mathbf{s}'_{t_i}$	Auxiliary data at time step t_i
$\mathbf{H}^{(k)}$	Hidden states at the k-th propagation step in the graph convolution module
β	Hyperparameter controlling retention ratio in graph convolution module
$\mathbf{H}_{in}$	Input hidden states from the previous layer
$\mathbf{H}_{out}$	Output hidden states of the current layer
$\tilde{\mathbf{A}}$	Normalized adjacency matrix including self-connections
$\mathbf{W}^{(k)}$	Parameter matrix in graph convolution module
$\mathbf{z}$	Input 1D sequence in temporal convolution module
$\mathbf{f}_{1\times k}$	Convolution filters in temporal convolution module

Table 5.1: Table of notations used in the paper.

References

[1] Tao Bai and Pejman Tahmasebi. "Graph neural network for groundwater level forecasting". In: *Journal of Hydrology* 616 (2023), p. 128792. ISSN: 0022-1694. DOI: https://doi.org/10.1016/j.jhydrol.2022.128792. URL: https://www.sciencedirect.com/science/article/pii/S0022169422013622.

[2] R. Bentivoglio et al. "Rapid Spatio-Temporal Flood Modelling via Hydraulics-Based Graph Neural Networks". In: *EGUsphere* 2023 (2023), pp. 1–24. DOI: 10.5194/egusphere-2023-284. URL: https://egusphere.copernicus.org/preprints/2023/egusphere-2023-284/.

[3] Stefan Bloemheuvel et al. "Graph neural networks for multivariate time series regression with application to seismic data". In: *International Journal of Data Science and Analytics* (2022), pp. 1–16.

[4] D A Brakenhoff et al. "Application of time series analysis to estimate drawdown from multiple well fields". In: *Frontiers in earth science* 10 (2022), p. 907609.

[5] Simon Brenner et al. "Process-based modelling to evaluate simulated groundwater levels and frequencies in a Chalk catchment in south-western England". In: *Natural Hazards and Earth System Sciences* 18.2 (2018), pp. 445–461.

[6] D. Cao et al. "Spectral temporal graph neural network for multivariate time-series forecasting". In: *Advances in neural information processing systems*. Vol. 33. 2020, pp. 17766–17778.

[7] C. Chen et al. "Gated Residual Recurrent Graph Neural Networks for Traffic Prediction". In: *AAAI*. 2019.

[8] Junjun Chen et al. "An improved tandem neural network architecture for inverse modeling of multicomponent reactive transport in porous media". In: *Water Resources Research* 57.12 (2021), e2021WR030595.

[9] Kyunghyun Cho et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation". In: *arXiv preprint arXiv:1406.1078* (2014).

[10] RW Dudley et al. "Estimating historical groundwater levels based on relations with hydrologic and meteorological variables in the US glacial aquifer system". In: *Journal of Hydrology* 562 (2018), pp. 530–543.

[11] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[12] L. Huang et al. “Dynamic Relation Discovery and Utilization in Multi-Entity Time Series Forecasting”. In: *arXiv preprint arXiv:2202.10586* (2022).

[13] J. Kan et al. “Sign Language Translation with Hierarchical Spatio-Temporal Graph Neural Network”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2022, pp. 3367–3376.

[14] A. M. Karimi et al. “Spatiotemporal Graph Neural Network for Performance Prediction of Photovoltaic Power Systems”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.17 (2021), pp. 15323–15330. URL: `https://ojs.aaai.org/index.php/AAAI/article/view/17799`.

[15] Ryan Keisler. *Forecasting Global Weather with Graph Neural Networks*. 2022. arXiv: `2202.07575 [physics.ao-ph]`.

[16] Remi Lam et al. “Learning skillful medium-range global weather forecasting”. In: *Science* 382.6677 (2023), pp. 1416–1421. DOI: `10.1126/science.adi2336`. eprint: `https://www.science.org/doi/pdf/10.1126/science.adi2336`. URL: `https://www.science.org/doi/abs/10.1126/science.adi2336`.

[17] S Sahoo et al. “Machine learning algorithms for modeling groundwater level changes in agricultural regions of the US”. In: *Water Resources Research* 53.5 (2017), pp. 3878–3895.

[18] J. Simeunović et al. “Spatio-temporal graph neural networks for multi-site PV power forecasting”. In: *IEEE Transactions on Sustainable Energy* 13.2 (2021), pp. 1210–1220.

[19] Stowa and NHI. *NHI modelcodes en koppelingen*. `https://nhi.nu/documents/47/Documentatie_modelcode.pdf`. Accessed: 01/03/2024. May 2023.

[20] Hai Tao et al. “Groundwater level prediction using machine learning models: A comprehensive review”. In: *Neurocomputing* 489 (2022), pp. 271–308.

[21] Jos R Von Asmuth et al. “Modeling time series of ground water head fluctuations subjected to multiple stresses”. In: *Groundwater* 46.1 (2008), pp. 30–40.

[22] Zonghan Wu et al. “Connecting the dots: Multivariate time series forecasting with graph neural networks”. In: *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 2020, pp. 753–763.

[23] A. Wunsch, T. Liesch, and S. Broda. “Groundwater level forecasting with artificial neural networks: a comparison of long short-term memory (LSTM), convolutional neural net-

works (CNNs), and non-linear autoregressive networks with exogenous input (NARX)". In: *Hydrology and Earth System Sciences* 25.3 (2021), pp. 1671–1687. DOI: `10.5194/hess-25-1671-2021`. URL: `https://hess.copernicus.org/articles/25/1671/2021/`.

[24] Andreas Wunsch, Tanja Liesch, and Stefan Broda. "Deep learning shows declining groundwater levels in Germany until 2100 due to climate change". In: *Nature communications* 13.1 (2022), p. 1221.

[25] Andreas Wunsch, Tanja Liesch, and Stefan Broda. "Forecasting groundwater levels using nonlinear autoregressive networks with exogenous input (NARX)". In: *Journal of Hydrology* 567 (2018), pp. 743–758. ISSN: 0022-1694. DOI: `https://doi.org/10.1016/j.jhydrol.2018.01.045`. URL: `https://www.sciencedirect.com/science/article/pii/S0022169418300556`.

[26] Wenjie Yin et al. "Improved water storage estimates within the North China Plain by assimilating GRACE data into the CABLE model". In: *Journal of Hydrology* 590 (2020), p. 125348.

[27] B. Yu, H. Yin, and Z. Zhu. "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting". In: *arXiv preprint arXiv:1709.04875* (2017).

Chapter 6

Conclusions

This thesis has successfully demonstrated the potential of deep learning in transforming groundwater modeling. The research has shown that machine learning models can provide rapid, accurate predictions and efficiently handle complex hydrological systems. The contributions of this thesis is to provide an evolution of research from steady-state scenarios using synthetic data and computer vision techniques to more complex applications involving sparse data, neural operators, and real-world scenarios with graph neural networks. Throughout this journey, the work has been anchored by a central research question: identifying the most effective neural network-based approach for predicting groundwater flow. Each paper within this thesis contributes uniquely to addressing this question, progressively fulfilling the outlined objectives and pushing the boundaries of current knowledge and methodologies in groundwater prediction.

6.1 Discussion

Paper 1 (Chapter 2): Attention U-Net as a Surrogate Model for Groundwater Prediction

This work introduces a convolutional encoder-decoder network, specifically the Attention U-Net, as a surrogate model for predicting the steady-state response of groundwater systems. This approach represents a significant advancement in groundwater modeling, leveraging the power of machine learning to address complex, nonlinear relationships between inputs (hydraulic conductivity fields and boundary conditions) and outputs (the hydraulic head field) with greater efficiency and reduced computational demands. The adoption of attention mechanisms within the U-Net architecture has been shown to significantly enhance the model's ability to focus on salient regions of the input data, thereby improving approximation accuracy and reducing model uncertainty. This approach yields very accurate results in forward simulations, substantially reducing the computational time required compared to traditional numerical solvers. Another notable strength of this methodology is the efficiency of the training process, which is carried out offline and converges in a relatively short period. Once trained, the model can be deployed for inference without further adjustments or re-training.

Among the limitations of this work, the paper acknowledges that the choice of hyperparameters and the U-Net architecture specifics are based on manual variation rather than systematic optimization. Future work could benefit from a more robust hyperparameter tuning study, possibly incorporating automated methods like grid search or Bayesian optimization to explore a broader parameter space and identify optimal configurations. Moreover, the study's focus on Dirichlet boundary conditions and a single resolution for data samples presents limitations in the model's applicability to more diverse real-world scenarios. Expanding the model to accommodate mixed boundary conditions and varying discretizations could significantly enhance its utility, making it more adaptable to different groundwater systems and the varying quality of available data.

Addressing more complex, dynamic, and uncertain systems, including time-dependent problems, three-dimensional simulations, and coupled transport through porous media, represents a natural and necessary progression of this research. Such extensions would likely require larger training datasets and potentially deeper neural network architectures to capture the increased complexity of these systems adequately. Finally, the suggestion to incorporate prior physical knowledge directly into the learning process, by imposing physics constraints in the loss function, is particularly promising. This method could offer a way to increase inference speed and reduce the amount of data needed for training by leveraging existing knowledge of the underlying physical processes governing groundwater flow.

In conclusion, this paper represents a significant advancement in the application of deep learning for groundwater modeling, demonstrating the potential of the Attention U-Net model as a surrogate for groundwater prediction. Its innovative application of attention mechanisms for focusing on relevant parts of the domain offers detailed and accurate predictions, showcasing significant advancements over traditional approaches. This initial investigation successfully addresses the first two objectives of the thesis: firstly, by developing a basic model that learns the steady-state solution of the governing Partial Differential Equations in a homogeneous domain under constant boundary conditions, and secondly, by progressing to more sophisticated models capable of understanding steady-state solutions in heterogeneous domains with diverse boundary conditions. This foundational work not only achieved these specific objectives but also established a solid basis for further exploration of complex models and their applications in real-world hydrological systems.

Paper 3 (Chapter 3): Understanding the Efficacy of U-Net & Vision Transformer for Groundwater Numerical Modelling

The paper investigates the efficacy of U-Net, U-Net integrated with Vision Transformers (ViT), and Fourier Neural Operator (FNO) models for predicting hydraulic head in groundwater studies. The comparative analysis, grounded in synthetic datasets reflective of the Overbetuwe region's conditions in the Netherlands, reveals critical insights into the performance and applicability of these models, particularly in scenarios characterized by sparse data.

The demonstrated superiority of the U-Net and U-Net + ViT models in handling sparse observation scenarios underscores the practical relevance of these approaches to real-world groundwater modeling, where data scarcity often poses significant challenges. The U-Net model demonstrates proficiency in identifying pumping wells, with only a marginal improvement in performance observed through the integration of Vision Transformers in its bottleneck. The fact that a more sophisticated Vision Transformer architecture does not necessarily lead to improved performance calls for a critical reflection on the balance between model complexity and practical utility. This balance is especially relevant in the field of groundwater modeling, where both the interpretability and computational efficiency of models are of paramount importance, along with predictive accuracy.

The intention to apply this methodology to real-world data from the Overbetuwe region presents a promising direction for future research. This step is crucial for validating the models in more complex and uncertain environments, which are characteristic of actual groundwater systems.

In summary, the paper further enriches the thesis by comparing different neural network architectures in their efficacy for groundwater numerical modeling. This comparative analysis directly addresses the third objective of the thesis, which involves evaluating the effectiveness of the developed models against other methodologies. Additionally, it sheds light on both the potential and limitations of current models, paving the way for future research that is more attuned to the complexities and uncertainties of real-world applications.

Paper 2 (Chapter 4): Developing a Cost-Effective Emulator for Groundwater Flow Modeling Using Deep Neural Operators

This paper presents the DeepONet framework as an efficient surrogate model for groundwater flow prediction. This framework demonstrates its capability to predict hydraulic head in heterogeneous aquifers, transient systems and generalize to unseen conditions, such as changes in well locations and hydraulic conditions. The successful implementation of this neural operator across various experiments, including forward and inverse problems,underscores its potential as a versatile tool for repetitive forward numerical simulations, offering significant improvements over traditional modeling approaches. Moreover, a novel contribution of this work is the modification of the DeepONet architecture, specifically the linking of the branch network to the trunk network. This adjustment enables the model to accurately predict outcomes for unseen well locations and varying hydraulic conductivity fields.

However, the study acknowledges certain limitations and areas for future research. The current focus on single-well and two-wells configurations represents a constrained exploration of the model's capabilities. Expanding the model to encompass multiple well-surrogate modeling is identified as a natural progression for future work, suggesting a need to explore more complex and realistic subsurface problems. This expansion would enhance the model's applicability and relevance to real-world groundwater management challenges. Furthermore, the exploration of sparse observations for inverse modeling in this study opens the door to investigating the strategic placement of observation points, which are the inputs to the trunk network. Understanding how the distribution of these points affects model performance is a critical aspect that needs further exploration. Such investigations could lead to more effective strategies for data collection and model training, thereby improving the reliability and accuracy of predictions. Future research directions also include uncertainty analysis and the integration of multi-fidelity and multi-modality approaches. Recognizing that real-world data are not only sparse but also vary in source and characteristics, the paper points towards the challenge and opportunity of integrating diverse data sources.

In conclusion, this paper advances the narrative of the thesis by introducing a deep neural operator framework for groundwater flow modeling, capable of handling sparse data and diverse problem setups. This development aligned with the thesis's aim to create versatile models for groundwater management, addressing the complexity and diversity inherent in real-world hydrological systems. The contributions of the paper in developing a cost-effective emulator for groundwater flow modeling using Deep Neural Operators directly align with the fourth and fifth objectives of the thesis. These include focusing on inferring solutions from limited and varied point measurements to address the challenges of incomplete or indirect data found in real-world scenarios and extending the models to capture transient solutions of the PDE, thereby incorporating the crucial dimension of time that reflects the dynamic nature of groundwater flow.

Paper 4 (Chapter 5): Spatial-Temporal Graph Neural Networks for Groundwater Data

Leveraging a comprehensive dataset of real-world groundwater level time series, this paper investigates the application of spatial-temporal graph neural networks (ST-GNNs) for groundwater level forecasting. The approach directly aligns with the sixth objective of the thesis, which focuses on employing developed models in real-world case studies to evaluate their predictive accuracy and practical applicability against measured ground truth data. The study demonstrates the ST-GNN model's superior performance in long-term forecasting and its robustness against noise, outperforming traditional LSTM networks and the numerical MODFLOW model.

By leveraging a comprehensive dataset that includes both groundwater level time series and auxiliary data such as precipitation, evaporation, and river stages, the study adeptly models the complex spatial and temporal dynamics that influence groundwater systems. The graph-based approach of the ST-GNN model, with its inherent capacity to encapsulate these multifaceted interactions, demonstrates high forecasting accuracy and robustness, especially in long-term forecasting and specific observation wells scenarios, a notable achievement given the noise and missing data typical of real-world datasets. The model's utility extends to scenario planning and decision-making for water resource management, offering a strategic tool for determining optimal water extraction volumes under various weather conditions. The rapid computation afforded by the deep learning architecture underscores the model's practical value, enabling the execution of numerous probabilistic scenarios with efficiency and agility.

The study not only demonstrates the efficacy of the current model but also paves the way for future enhancements to its predictive capabilities. Future work could be the incorporation of additional variables into the model's architecture, such as using resistance as weights in the graph's edges. Enriching the model with a broader range of exogenous variables and additional features such as aquifer elevations, hydraulic conductivity, and aquitard resistance could deepen the model's understanding of groundwater dynamics, leading to more precise predictions. The prospect of developing networks capable of processing multi-modal inputs, including continuous geological information, could dramatically enhance the model's utility, making it an even more powerful tool for groundwater level prediction.

Moreover, future research could aim to scale the ST-GNN approach to encompass larger networks capable of making predictions beyond mere observation points. This ambition sets the stage for a model that could rival traditional approaches, offering solutions for the entire domain. This advancement would not only broaden the scope of machine learning applications in hydrology but also potentially revolutionize the field by providing more flexible, efficient, and comprehensive tools for groundwater management.

In conclusion, this work culminates the research journey by employing spatial-temporal graph neural networks to predict groundwater levels in a real-world case study. This paper's focus on integrating spatial and temporal dynamics through ST-GNNs directly addressed the sixth objective of the thesis, applying developed models in a practical context and benchmarking their predictions against actual data. The success of ST-GNNs in providing accurate long-term forecasts, especially in handling missing data, underscored the thesis's ultimate goal of developing sophisticated, accurate, and practical tools for groundwater modeling and prediction.

Collectively, these papers weave a coherent narrative that responds to the central research questions of the thesis, while significantly advances the modelling and methodological approaches within the field of groundwater prediction. This thesis demonstrates the transformative potential of deep learning-based models in groundwater management, offering novel insights and methodologies that pave the way for future research.

6.2 Future Work

Moving forward, there is a significant need for the development of scalable and efficient models that can accurately simulate complex processes across a range of scales, from local to global groundwater level systems. A critical step in this direction involves creating comprehensive benchmark datasets for groundwater. These datasets will serve a dual purpose: firstly, they will enable the development and rigorous evaluation of machine learning models tailored for GWL forecasting and uncertainty quantification; secondly, they will establish a standardized foundation that the broader research community can leverage to innovate, test, and validate new modeling approaches and technologies.

Expanding on the groundwork laid by this thesis, future research will also explore the application of these models to broader environmental contexts, such as coupled ground-water and surface-water flow models. This expansion will leverage the experience gained in groundwater prediction to address complex environmental challenges more holistically.

Moreover, the creation of robust tools for quantifying and managing uncertainty in environmental models represents a vital area of future work. By building on advances in AI and machine learning, integrating uncertainty quantification methods into environmental simulations will enhance the predictive power and reliability of these models, contributing to more informed and effective real-world decision-making.

Finally, the methodologies and findings presented in this thesis lay a solid foundation for continued innovation in the field of groundwater modeling. As we look to apply these models to larger and more complex real-world datasets, such as an entire national network of groundwater measurements, the horizon for future research in this field is both exciting and promising. Further research will continue to advance the field, driving forward the capabilities of machine learning in environmental science and hydrology.