Institute of Parallel and Distributed Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Bachelorarbeit

Application and Extension of a Super Resolution Physics-Informed Convolutional Neural Network to Groundwater Modelling

Manuel Hirche

Course of Study:	Informatik
Examiner:	Prof. Dr. Miriam Schulte
Supervisor:	Julia Pelzer, M.Sc.
Commenced:	November 1, 2022
Completed:	March 30, 2023

Abstract

The computational effort of a simulation can be reduced by running simulations on a coarse grid and interpolating to a fine one. This interpolation can be done using data-driven neural networks, called super-resolution. To minimize the need to perform expensive simulations to create the datasets required for training, physics-informed neural networks (PINNs) add a physical error term to the learning process. In this work, we extend and apply a super-resolution PINN approach to a groundwater simulation with heat pumps. We extend an existing network model for flow velocity and pressure to include temperature and permeability of the soil and derive a corresponding error term. The model is trained on a data set from a simulation with different pressures and permeabilities. The results are compared with a data-driven network and bicubic interpolation. We find that both neural networks significantly outperform bicubic interpolation, whereas the PINN approach achieves slightly better results than the data-driven network.

Kurzfassung

Der Rechenaufwand einer Simulation kann verringert werden, indem Simulationen auf einem groben Gitter durchgeführt und auf ein feines interpoliert werden. Diese Interpolation kann mithilfe von datengetriebenen neuronalen Netzen durchgeführt werden, auch Super-Resolution genannt. Um möglichst wenige teure Simulationen zur Erstellung des Datensatzes durchführen zu müssen, wird das Netzwerk um einen physikalischen Fehlerterm ergänzt (PINN). In dieser Arbeit wird ein Super-Resolution-PINN-Ansatz auf eine Grundwassersimulation mit Wärmepumpen angewendet und erweitert. Dazu haben wir ein bestehendes Netzwerkmodell für Strömungsgeschwindigkeit und Druck auf Temperatur und Durchlässigkeit des Bodens erweitert, sowie einen entsprechenden Fehlerterm aufgestellt. Das Modell wird auf einem Datensatz aus einer Simulation mit unterschiedlichen Drücken und Durchlässigkeiten trainiert. Die Ergebnisse werden mit einem datengetriebenen Netzwerk und bikubischer Interpolation verglichen. Wir stellen fest, dass beide neuronalen Netze die bikubische Interpolation deutlich übertreffen. Der PINN-Ansatz erziehlt dabei geringfügig bessere Resultate als das datengetriebenen Netzwerk.

Contents

List of Figures

1 Introduction

With the progression of climate change, the demand for alternative energy sources with net-zero greenhouse gas emissions is becoming increasingly important [MVP+18]. The heating and cooling of buildings account for a significant portion of global energy consumption [GSS+15]. Geothermal energy through groundwater heat pumps (GWHP) is an important renewable source for building climate control [RBO+19]. GWHPs extract groundwater, transfer thermal energy to or from the water, and pump it back into the ground, resulting in changes in groundwater temperature around the injection point. As the number of GWHPs continues to rise, it is increasingly important to study and monitor their impact on the groundwater and their interference with each other [GMG+20].

High-fidelity groundwater simulations have been developed to determine the effects of GWHPs and provide a tool for planning heat pump locations and operational constraints, such as in the GEO-KW project [ZBD+22]. However, the large computational requirements of high-resolution fluid simulations make optimization on a large scale, e.g. whole cities, challenging.

Instead of simulating on a fine grid, high-resolution results can also be obtained by simulating first on a coarse grid and then interpolating the result back to the fine resolution (Figure 1.1). This approach can significantly reduce computational costs, but it requires an efficient interpolation function that approximates the fine-grid solution as closely as possible. The challenge lies in finding such an interpolation function that is both computationally cheap and accurately approximates the fine-grid solution.

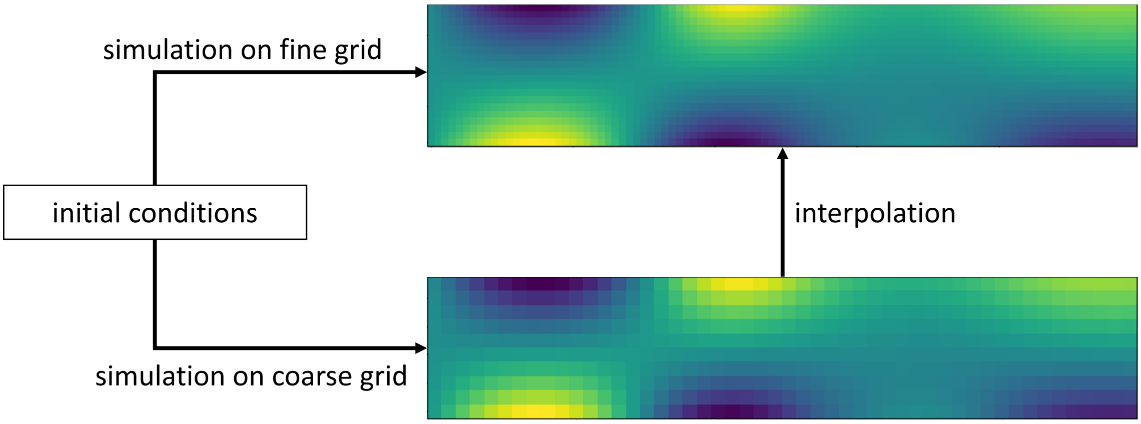


Figure 1.1: We can reduce the computational cost by first simulating on a coarse grid and then interpolating the results back to a high resolution instead of simulating directly on a fine grid.

In recent years, deep learning-based methods were very successful in finding such interpolation functions, see, e.g., [FFT20; FFT21; FSD+20; MFR+20; SGHK20; YOH+22]. This machine learning driven interpolation is called super-resolution. However, the problem remains, that a large amount of high-resolution simulation data is required to train those deep neural networks.

To address the challenge of learning with limited data, physics-informed neural networks (PINNs) have emerged as a promising approach. Instead of only learning from data samples like a classical neural network, PINNs integrate domain knowledge in the form of partial differential equations into the learning process. See [RPK17a; RPK17b; RPK19] for an introduction to PINNs.

The objective of this thesis is to assess the effectiveness of a super-resolution PINN approach for groundwater modeling with heat pumps. Specifically, we apply a super-resolution PINN to a two-dimensional groundwater flow system, including a single heat pump used to cool a building, as illustrated in Figure 1.2. We are considering a two-dimensional horizontal cut through the ground at the height of a heat pump outlet. The dataset used in this thesis is generated from a groundwater simulation performed in Pflotran [HLM14]. Our work builds upon the PINN model of Gao et al. [GSW21], as it has demonstrated promising results and is straightforward to extend to additional variables. To evaluate the performance of our model, we compare its results with those obtained using bicubic interpolation and the same model trained solely with a data loss.

Our contributions towards this objective are the following. We extend an existing network model for flow velocity and pressure to include temperature and permeability of the soil. We derive a suitable physics loss to train the network model as a PINN. We test the physics loss, implement the model and train it on a dataset with different pressures and permeabilities. We compare the results with a data-driven network and bicubic interpolation.

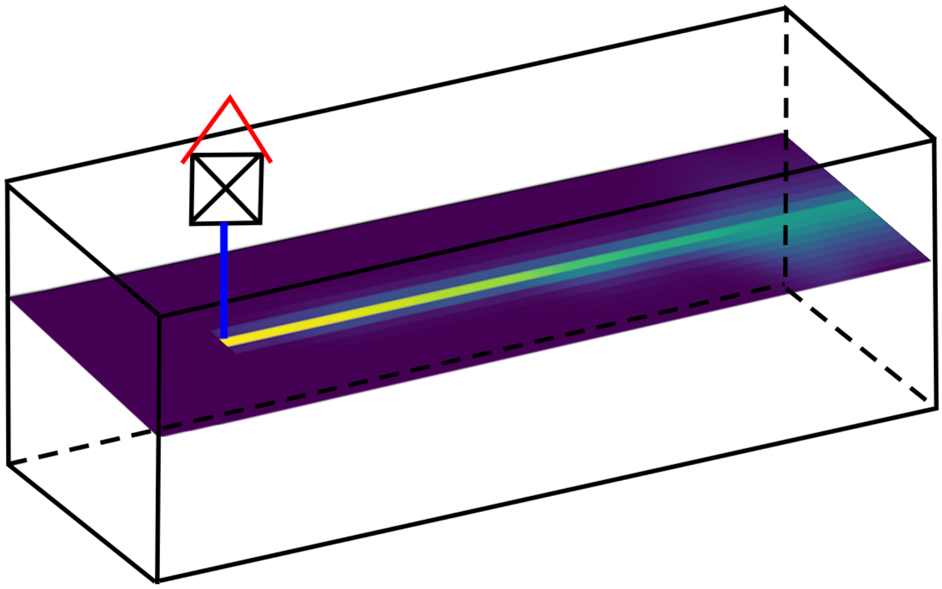


Figure 1.2: Thermal plume that results from a groundwater heat pump that extracts cold water, cools a building through heat exchange, and injects the warmed water back into the ground.

Chapter 2 reviews the relevant literature for the thesis, followed by Chapter 3 providing a detailed introduction to the concepts of fluid dynamics, convolutional neural networks, and PINNs. Furthermore, here we describe the model of Gao et al. which serves as the basis for the thesis. In Chapter 4, an introduction to the groundwater model and the dataset is provided followed by a discussion of the derivation of the physics loss and the extended network architecture. The evaluation of the network trained on the groundwater dataset is elaborated in Chapter 5 and concluded with the study results summarized in Chapter 6.

2 Related Work

Interpolation is a fundamental problem in numerical analysis. A common method being polynomial interpolation [DR08]. Bicubic interpolation is an extension of one-dimensional cubic interpolation to two dimensions. It has found widespread use in computer graphics, image processing, and other scientific fields [Rus95]. It is often used as a benchmark for other interpolation methods, see, e.g., [BGL+21; GSW21; JEA+20; KRM22; RRL+22]. Therefore, we will also compare our results to bicubic interpolation.

The interpolation problem requires the interpolant function to match the supporting points exactly. When we relax this constraint to give the best fit according to some optimization criteria, not all supporting points have to lie on the curve. This opens the door to powerful machine-learning techniques. In the field of computer vision, the recovery of high-resolution images from low-resolution ones is called super-resolution. In recent years deep learning based methods have become very successful at this task, see, e.g., [LSK+17; LTH+17; SCH+16]. Their success inspired the application of these methods to a broad range of fluid dynamics problems like near-surface temperature distribution in urban areas [YOH+22], climate models [SGHK20], blood flow [FSD+20], computer graphics [XFCT18], or turbulent flow [FFT20; FFT21; MFR+20].

Besides super-resolution, machine learning methods have many uses in fluid mechanics, of which Brunton et al. give a comprehensive overview [BNK20]. Deep learning based methods for computational fluid dynamics all have in common that the many high-resolution labels that are required for training are computationally expensive to obtain. To overcome this problem, Raissi et al. introduced PINNs as a method for the data-driven solution and discovery of partial differential equations [RPK17a; RPK17b; RPK19]. Instead of only learning from data samples like a classical neural network, PINNs incorporate differential equations into the learning process in order to require less training data. Since their introduction, PINNs with various architectures have been successfully applied for the data-driven solution of many fluid dynamics problems. Based on a fully connected neural network, Cai et al. [CWF+21] inferred the flow over an espresso cup, Kashefi et al. [KM22] and Jin et al. [JCLK21] consider incompressible flow, and Arzani et al. [AWD21] apply a PINN to blood flow. Sun et al. [SGPW20] use a fully connected network based PINN to solve the system of continuity and Navier–Stokes equation for incompressible flow without training data. To reconstruct a flow field from sparse measurement data, Molnar et al. [MG22] use a PINN based on a fully connected neural network by incorporating the measurement model into the loss function. A different network architecture, i.e., a generative adversarial network, is used by Subramaniam et al. [SWB+20] on turbulent flow. Directly related to our work is the paper by Leiteritz et al. [LDSP22], in which they used an autoencoder-based PINN as a surrogate model to predict the thermal plumes of groundwater heat pumps from a given subsurface velocity field. In contrast to their work, we also have the thermal plume and other variables as input but on a coarse resolution.

PINNs were also successfully applied for the super-resolution of various fluid dynamics problems. Kelshaw et al. [KRM22] and Zayats et al. [ZZYZ22] used a convolutional neural network based super-resolution PINN on turbulent flow. Bode et al. [BGL+21] applied a generative adversarial network (GAN) based PINN on turbulent flow. Also with a GAN based PINN, Li et al. [LM22] did super-resolution on multiphase flow. Ren et al. [RRL+22] used a long short term memory (LSTM) convolutional PINN on spatio-temporal problems. Jiang et al. [JEA+20] also developed a PINN for a spatio-temporal setting. Arora [Aro22] applied a PINN to computational solid mechanics to study mechanical behaviour of materials. They used a network architecture with a separate individual neural network on each interpolated variable. Gao et al. [GSW21] used a similar architecture, but with a simpler structure of the individual sub-networks, on cardiovascular blood flow and achieved good results, even without high-resolution labels. The successful application of PINNs to super-resolution in fluid dynamics lead to this work that we base on the network model of Gao et al. [GSW21].

3 Theoretical Background

First, we will summarize the basic concepts of fluid dynamics and explore the equations of motion that govern fluid behavior. Next, we give an introduction to (convolutional) neural networks for super-resolution. Finally, we introduce physics-informed neural networks and the network model of Gao et al. which is used as the basis for our work.

3.1 Fluid Dynamics

For the main part of this section, we follow the work of Landau et al. [LL87], which gives a thorough introduction to fluid dynamics. Fluids can be described by three physical quantities: fluid velocity, and two thermodynamic quantities. The latter are commonly pressure and density, but they can be interchanged with other thermodynamic quantities, e.g., temperature, because they are related through so-called equations of state.

The three variables are linked through three equations, namely the conservation of mass, momentum, and energy. We first look at the conservation of mass for a general fluid. Depending on the effects that are to be modeled, certain simplifying assumptions are made. A common assumption is that the fluid is incompressible.

The conservation of mass in a fluid is expressed by the so-called equation of continuity. If we consider some volume of space, it states that the mass inside the volume equates to the mass that is moving in and out of the volume through its surface. Let ρ be the fluid density and $\mathbf{v}$ its velocity. Then the equation of continuity reads

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0. \tag{3.1}$$

3.1.1 Incompressible Fluids

If the density ρ is treated as a constant, we call the fluid incompressible. In this case, the equation of continuity simplifies to

$$\nabla \cdot \mathbf{v} = 0. \tag{3.2}$$

Through the conservation of momentum, we get the equation of motion for a fluid linking its velocity to pressure p. As a consequence of internal friction and heat conduction, energy is dissipated in a moving fluid, which must be considered. This internal friction in a fluid is quantified by viscosity. If we regard the viscosity as constant, we arrive at the form of the Navier-Stokes equation

$$\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\mathbf{v} = -\frac{1}{\rho}\nabla p + \frac{\mu}{\rho} \triangle \mathbf{v} \tag{3.3}$$

with the constant viscosity coefficient μ, which is often used for fluid models. Such fluids are called Newtonian.

The equation of continuity and the conservation of momentum equation gives us two equations for the three unknowns. Therefore a third equation is needed. In the case of an incompressible fluid, the simplifying condition of constant density is often used as a third equation. This implicitly requires a constant temperature since, for fluids with non-uniform temperature, the density would vary. In that case, an equation for the conservation of energy is required. Since we are interested in the temperature distribution in the ground, our model needs this equation. We look at this equation in the context of porous media.

3.1.2 Flow in Porous Media

Previously, we have looked at fluids that flow freely and unobstructed. To arrive at a groundwater flow model, we have to take into account that the water moves through a porous medium. In hydrology, this is called an aquifer. The aquifer is a geologic formation that allows water to move through it. The portion of the aquifer that is not solid matter is the pore space occupied by water or air.

For a comprehensive introduction to groundwater flow, see [Bea79]. Our simulation dataset comes from a simulation performed in Pflotran [HLM14]. The following equations are therefore in the form used by Pflotran [HLLM12; LHL+20].

In an aquifer, the equation of continuity has the same form as (3.1) with the addition of the porosity φ and saturation s in the time derivative

$$\frac{\partial}{\partial t}(\varphi s \rho) + \nabla \cdot (\rho \mathbf{q}) = Q_{\mathrm{w}}. \tag{3.4}$$

Porosity denotes the percentage of pore space in the aquifer. Saturation measures the percentage of pore space that is filled with water. Q_{w} is a source term. Our previous flow velocity $\mathbf{v}$ is now called Darcy velocity $\mathbf{q}$. This is due to Darcy's law which we will introduce shortly. The Darcy velocity is related to the flow velocity by the porosity through $\mathbf{v} = \mathbf{q}/\varphi$.

Inherent to the flow in an aquifer is the notion of permeability k. It expresses how well the fluid can flow at a certain location. We will only consider isotropic aquifers, i.e., the permeability is the same in every direction. The conservation of momentum, linking fluid velocity and pressure, now takes the form of Darcy's law

$$\mathbf{q} = -\frac{k k_r}{\mu} \nabla (p - \rho g z). \tag{3.5}$$

It was first determined experimentally but has since also been derived analytically from the Navier-Stokes equation [Whi86]. Here, $k_{\mathrm{r}} = k_{\mathrm{r}}(s)$ denotes the relative permeability, which is a function of the saturation s. The quantity $\rho g z$ is called the elevation head and represents the influence of gravity on the fluid with the gravity g and $z = (0, 0, 1)^{\top}$.

As stated earlier, in the case of non-uniform temperature, we need the equation for the conservation of energy. It accounts for the processes of internal friction due to viscosity and the transfer of heat throughout the fluid that occurs because of the non-uniform temperature. This leads to the equation for the conservation of energy

$$\frac{\partial}{\partial t}(\varphi s \rho U + (1-\varphi)\rho_r c_p T) + \nabla \cdot (\rho \mathbf{q} H - \kappa \nabla T) = Q_\mathrm{e} \tag{3.6}$$

where U denotes the internal energy and H is the enthalpy, which are two thermodynamic quantities that measure the energy in a thermodynamic system. They are, as, e.g., density, given by equations of state. Moreover, ρ_r is the rock density, c_p denotes the heat capacity, κ the thermal conductivity, and Q_e is again a source term.

3.2 Neural Networks

Comprehensive overviews of neural networks are available in [HTF09] and [Bis95]. A neural network is a function $f : \mathbb{R}^{h_0} \to \mathbb{R}^{h_L}$, where each layer k is h_k-dimensional. We define an L-layer neural network recursively as follows.

$$\text{1-Layer: } f_\beta(x) = W_1 x + b_1 \tag{3.7a}$$

$$\text{2-Layer: } f_\beta(x) = W_2 \sigma(W_1 x + b_1) + b_2 \tag{3.7b}$$

$$\text{L-Layer: } f_\beta(x) = W_L \sigma(\dots \sigma(W_1 x + b_1) \dots) + b_L \tag{3.7c}$$

The network has the parameters $\beta = (W_{1:L}, b_{1:L})$, where $W_k \in \mathbb{R}^{h_k \times h_{k-1}}$ and $b_k \in \mathbb{R}^{h_k}$. The W_k are called weights, and the b_k biases. In our case, the first layer dimension h_0 equals the coarse grid resolution and h_L the fine grid resolution, respectively.

After the first layer, the output is put elementwise through a so-called activation function σ. The role of the activation function is to make the model sufficiently complex to approximate nonlinear functions. Desired properties of this function are non-linearity and differentiability almost everywhere. A commonly used activation function is Rectified Linear Unit (ReLU) [DSC22]

$$\sigma(x) = \max\{0, x\}. \tag{3.8}$$

Looking at layer k, we have the calculations

$$z^{(k)} = W_k x^{(k-1)} + b_k, \quad x^{(k)} = \sigma(z^{(k)}) \tag{3.9}$$

with an intermediate result z.

We call this result $x^{(k)}$ a neuron. A single neuron in this layer performs a scalar product of the whole input x with weights before a bias is added and the activation function is applied. Together with the other neurons, this forms a network in which each neuron is connected to the output of all neurons from the previous layers.

3.2.1 Convolutional Neural Networks

In fully connected layers, each neuron is connected to all neurons of the previous layers. In a convolutional layer, each neuron only looks at a smaller region of the previous layer. The idea behind this comes from computer vision and tries to exploit the two following properties. First, the features of objects are spatially invariant, e.g., a cat in the top left of an image is still a cat if it is in the lower right corner. Second, the information at a specific image location is more related to its neighborhood rather than to distant points [KSH12]. Those ideas also translate to the domain of partial differential equations.

A neuron in a convolutional layer computes the scalar product of a so-called filter kernel w_i with the input region flattened to a 1D vector

$$z^{(k)} = w_i^\top x + b_k, \quad x^{(k)} = \sigma(z^{(k)}). \tag{3.10}$$

The region is slid over the whole input, which gives a plane of output values, each calculated with the same filter kernel. Repeating this with different filter kernels makes up the convolutional layer, see Figure 3.1. Hyperparameters of this layer, i.e., parameters that are not trained by the network, are the size of the filter kernel, the stride, and the layer size. The stride denotes the step size used when moving the filter.

3.2.2 Super Resolution with Neural Networks

Our neural network f_β (3.7c) is determined by its parameters β. The challenge lies in finding suitable values for these parameters. This process is called training which can be formulated as an optimization problem.

Let $D = \{(\text{coarse}_i, \text{fine}_i)\}_{i=1}^n$ denote the simulation data set consisting of matching simulation results on coarse and fine grids. For a given coarse and corresponding fine result $(\text{coarse}_i, \text{fine}_i)$, we first calculate the output of the network for the input coarse_i. This is called forward propagation. Then we define a loss function $\ell(\beta)$ that we want to minimize. In our case, that is the least squares loss

$$\ell(\beta) = \|f_\beta(\text{coarse}_i) - \text{fine}_i\|_2^2. \tag{3.11}$$

The optimization is done by gradient descent

$$\beta_{i+1} = \beta_i - \epsilon \frac{\partial \ell}{\partial \beta} \tag{3.12}$$

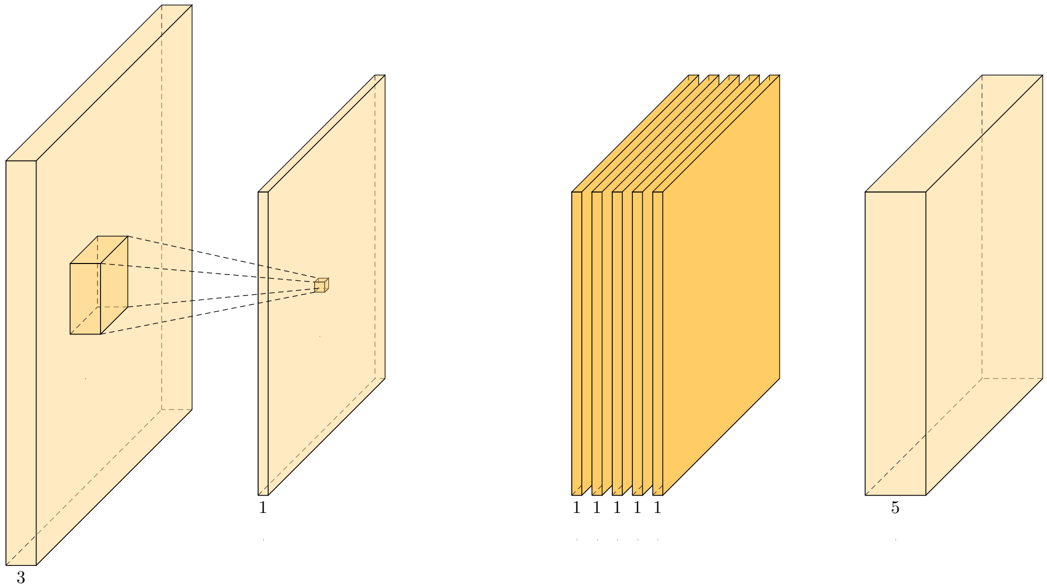


Figure 3.1: Sliding the region over the input makes up a plane of neurons that share a filter kernel. Repeating this calculation for different filter kernels makes up a convolutional layer. How often the calculation is repeated determines its size. In this example the layer size is 5.

with the learning rate ϵ as step size. The gradients are calculated recursively through the network by an algorithm called backpropagation.

Taking the gradient over the whole training set every iteration, which is the sum over the gradients of the single training samples, is computationally too heavy for large datasets. Instead, a randomly selected subset of the training set D_i, called minibatch, is used to calculate an average gradient. This is called stochastic gradient descent. A commonly used stochastic gradient descent algorithm is ADAM by Kingma and Ba [KB15].

3.3 Physics-Informed Neural Networks

Physics-informed neural networks (PINNs) were introduced by Raissi et al. as a method for the data-driven solution and discovery of partial differential equations that require less training data [RPK17a; RPK17b; RPK19]. Instead of learning only from data samples like a classical neural network, PINNs incorporate domain knowledge in form of partial differential equations into the learning process.

Consider a nonlinear partial differential equation in the form

$$u_t + \mathcal{N}[u;\lambda] = 0, \quad x \in \Omega, t \in [0,T] \tag{3.13}$$

with the solution $u(t,x)$ and the operator $\mathcal{N}[\cdot;\lambda]$ parameterized by λ on a region $\Omega \subset \mathbb{R}^d$.

We approximate the solution $u(t,x)$ through a neural network. Let $g(t,x) = u_t + \mathcal{N}[u;\lambda]$ be the left-hand side of equation (3.13). This now-called physics-informed neural network $g(t,x)$ can be derived from $u(t,x)$ through automatic differentiation and shares the same parameters with $u(t,x)$.

Let $\{(t_i^u, x_i^u, u_i)\}_{i=1}^{N_u}$ denote the initial and boundary training data for $u(t,x)$ and $\{(t_i^g, x_i^g)\}_{i=1}^{N_g}$ collocation points for $g(t,x)$. Then the shared parameters can be learned with the combined mean squared error loss

$$\mathrm{MSE} = \mathrm{MSE}_u + \mathrm{MSE}_g \tag{3.14}$$

with

$$MSE_u = \frac{1}{N_u} \sum_{i=1}^{N_u} \| u(t_i^u, x_i^u) - u_i \|_2^2 \tag{3.15}$$

and

$$MSE_f = \frac{1}{N_g} \sum_{i=1}^{N_g} \| g(t_i^g, x_i^g) \|_2^2 . \tag{3.16}$$

In the case of super-resolution, we have our dataset of corresponding low and high-resolution samples $D = \{(\mathrm{coarse}_i, \mathrm{fine}_i)\}_{i=1}^{n}$. We can then train our super-resolution network f_β (that corresponds to u) on the combined loss [GSW21]

$$MSE = \frac{1}{n} \sum_{i=1}^{n} \| f_\beta(\mathrm{coarse}_i) - \mathrm{fine}_i \|_2^2 + \frac{1}{n} \sum_{i=1}^{n} \| g(\mathrm{coarse}_i) \|_2^2 . \tag{3.17}$$

3.3.1 Super Resolution Model from Gao et al.

Our work builds on the work of Gao et al. [GSW21]. They applied their super-resolution PINN model to cardiovascular blood flow. Even without high-resolution data, with additional noise on the input data, and simultaneous inference of unknowns like boundary conditions, they achieved good results. We describe their model here since our model is an extension of their convolutional neural network. We limit our description to the super-resolution part of the model since we do not consider inference.

Gao et al. consider the flow described by the equation of continuity (3.2) and the Navier-Stokes equation (3.3). The input to the model is a two-dimensional flow described by the flow velocity in the two dimensions and pressure. Figure 3.2 shows the network. In the first step, the three variables are interpolated by bicubic interpolation to the target resolution. After that, every variable has a separate path leading to three distinct outputs. Note that every path gets all three variables as input but is specialized to one output variable. The paths are composed of four convolutional layers. Each layer performs the convolution with a 5x5 filter kernel. The input of the convolutional layers is padded by 2, and the stride is 1, so every layer has the same dimensions as the previous layer. ReLU is used as the activation function. The first convolution is performed with 16 filter kernels, the second with 32, and the third again with 16. The last convolution has depth 1, which is the output. Note that these are distinct paths without connections. With data-driven learning, they could be trained independently. However, they are coupled by the physics loss in the PINN setting.

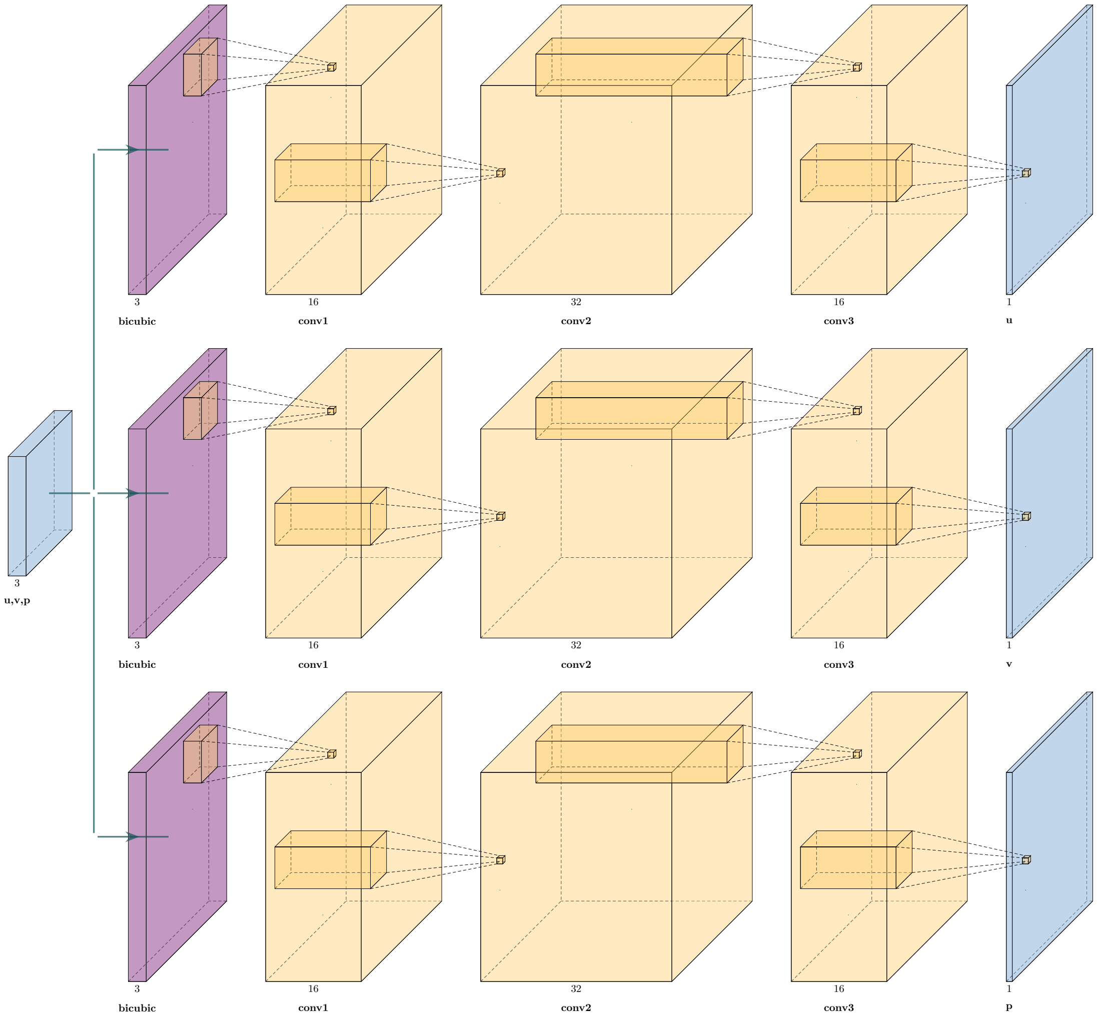


Figure 3.2: Architecture of the super-resolution convolutional neural network by Gao et al [GSW21]. The input to the model is a two-dimensional flow described by the flow velocity and pressure. In the first step (purple), the three variables are interpolated by bicubic interpolation to the target resolution. Four convolutional layers follow on every path leading to distinct outputs for every variable.

4 PINN for SR in Groundwater Modeling

In this section, we present the development of a super-resolution physics-informed convolutional neural network designed for the heat pump scenario described in the introduction. We are considering a two-dimensional horizontal cut through the ground at the height of a heat pump outlet, which pumps the warmed water back into the ground. Figure 4.1 shows an example of a thermal plume that develops at that height.

First, we introduce the dataset and the physics model that we use to derive the physics loss function. Second, we evaluate the performance of the physics loss by testing how well it fits the dataset. Finally, we describe the neural network architecture.

4.1 Dataset and Groundwater Model

The training dataset comes from a groundwater simulation performed with the subsurface flow and reactive transport code Pflotran [HLM14]. It consists of 100 samples that are generated by simulating the combination of 10 different permeabilities and 10 pressure boundary conditions. The pressure boundary conditions are gradients with different magnitudes as the one shown in Figure 4.2. The permeabilities are sampled from Perlin noise [Per02]. Three of the samples are shown in Figure 4.3.

We only want to consider a steady-state temperature distribution in the ground, therefore the simulation is run for five years to reach that. The considered region is 80 m by 1280 m large. For consistency reasons, we denote the short edge as the x direction and the long edge as the y direction. The two components of the Darcy flow are denoted by u in x-direction and v in y-direction. So the Darcy flow is $\mathbf{q} = (q_u, q_v)^\top$.

Our dataset consists of five sets of the above-described 100 samples that were simulated on different resolutions increasing by powers of 2. Starting with the coarsest resolution of 4 in x-direction and 64 in y-direction. The finest resolution is 64 by 1024. We did most of our analysis and development on the 8 by 128 and 16 by 256 resolutions. Those were the first ones available and lie in the middle of all the resolutions.

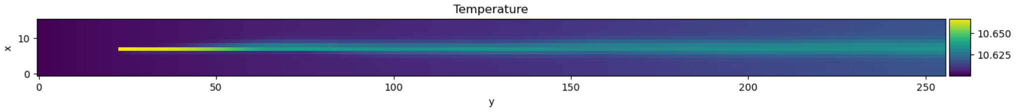


Figure 4.1: Thermal plume that develops at a two-dimensional horizontal cut through the ground at the height of a heat pump outlet.

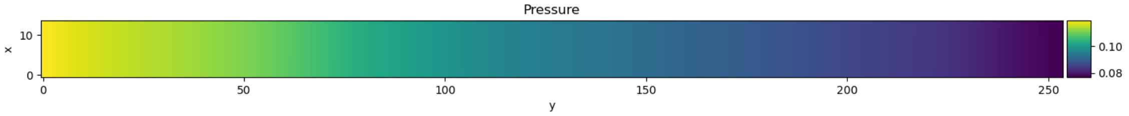


Figure 4.2: Pressure gradient that is used in different magnitudes as an initial boundary condition for our simulation.

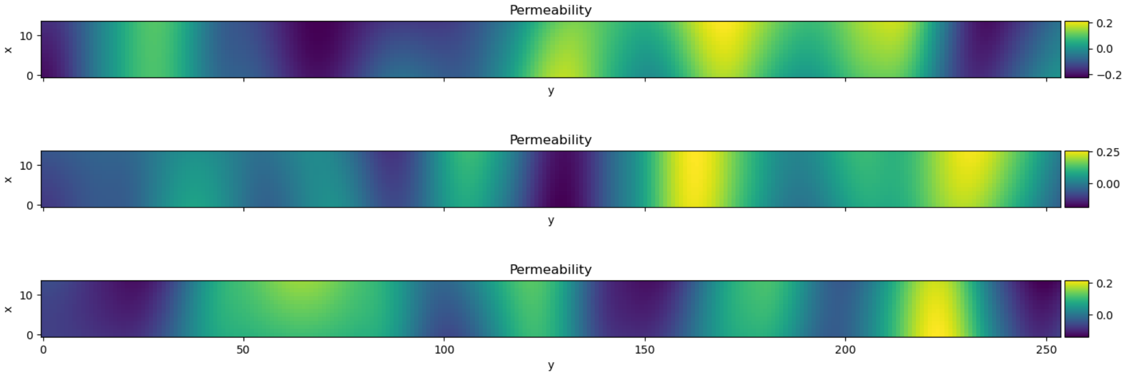


Figure 4.3: Examples of permeabilities sampled from Perlin noise that are used to generate our dataset. Note that it might look like the noise favors one direction. This is not the case and comes from the region being very small in one direction compared to the other.

The simulation is carried out in the Thermal-Hydrologic mode (TH Mode). This mode solves the equation of continuity (3.4), darcy velocity (3.5), and conservation of energy (3.6) fully coupled. Because of the steady state, our model uses the time-invariant forms of those equations. In our case, the ground is fully saturated, so the saturation s is set to fully saturated ($s = 1.0$), and therefore, the relative permeability k_r is also one. Since we only consider a two-dimensional horizontal plane, the gravitational head $\rho g z$ has no component in that direction and vanishes. The resulting equations of our model are

$$\nabla \cdot (\rho \mathbf{q}) = Q_\mathrm{w}, \tag{4.1a}$$

$$\mathbf{q} = -\frac{k}{\mu} \nabla p, \tag{4.1b}$$

$$\nabla \cdot (\rho \mathbf{q} H - \kappa \nabla T) = Q_\mathrm{e} \tag{4.1c}$$

where $\mathbf{q} = (q_u, q_v)^\top$ denotes the darcy velocity, T the temperature, p the pressure, and k the permeability. These are the five variables we are going to interpolate. Q_w and Q_e are source terms. The remaining variables are given by equations of state. These are the viscosity $\mu(T, p)$, density $\rho(T, p)$, and enthalpy $H(T, p)$, and thermal conductivity $\kappa(T)$.

The Pflotran documentation [HLLM12; LHL+20] makes no note of the used equations of state. We found in the source code [LHL+23] that equations from [Ver67] are used. For our work, we directly used the equations from the Pflotran source code.

4.2 Physics Loss

In order to train our model as a PINN, we derive a physics loss function based on the physical model described in Section 4.1.

Let $D = \{(\text{coarse}_i, \text{fine}_i)\}_{i=1}^{n}$ denote the simulation data set consisting of matching simulation results on coarse and fine grids. From the residuals of (4.1a, 4.1b, and 4.1c), we derive the following loss terms

$$MSE_{\text{Continuity}} = \frac{1}{n}\sum_{i=1}^{n} \|\nabla \cdot \mathbf{q} - Q_w\|_2^2, \tag{4.2}$$

$$MSE_{\text{Darcy u}} = \frac{1}{n}\sum_{i=1}^{n} \|(\mathbf{q} + \frac{k}{\mu}\nabla p)_x\|_2^2, \tag{4.3}$$

$$MSE_{\text{Darcy v}} = \frac{1}{n}\sum_{i=1}^{n} \|(\mathbf{q} + \frac{k}{\mu}\nabla p)_y\|_2^2, \tag{4.4}$$

$$MSE_{\text{Energy}} = \frac{1}{n}\sum_{i=1}^{n} \|\nabla \cdot (\eta \mathbf{q} H - \kappa \nabla T) - Q_e\|_2^2, \tag{4.5}$$

where the interpolated variables are given by the output of our neural network $[\mathbf{q}, T, p, k] = f_\beta(\text{coarse}_i)$ and the others through equations of state.

Together with the data loss

$$MSE_{\text{D}} = \frac{1}{n}\sum_{i=1}^{n} \|f_\beta(\text{coarse}_i) - \text{fine}_i\|_2^2 \tag{4.6}$$

we train our network on a weighted sum of those loss functions. The combined loss function with weights w_i reads

$$\text{Loss} = \sum_{i \in \{\text{D},\text{Continuity},\text{Darcy u},\text{Darcy v},\text{Energy}\}} w_i MSE_i. \tag{4.7}$$

4.3 Evaluation of the Physics Loss

Since the simulation results are a numerical solution of the PDEs, the residual vanishes for them. So we can check our error terms by inserting the simulation data and observing the residual. We do this on three different versions of the dataset in increasing complexity to spot different problems. All derivatives are discretized using symmetric difference quotients. We present only exemplary samples from the dataset since the results are similar on the whole dataset.

4.3.1 Uniform Permeability Without Heat Pump

We first look at the simplest case, i.e., with the heat pump turned off and with uniform permeability. The residuals of (4.1a), (4.1b), and (4.1c) are shown in Figure 4.4. The Darcy residual in the u-direction (second from the top) shows the best results with a residual in the order of 10^{-12}. The

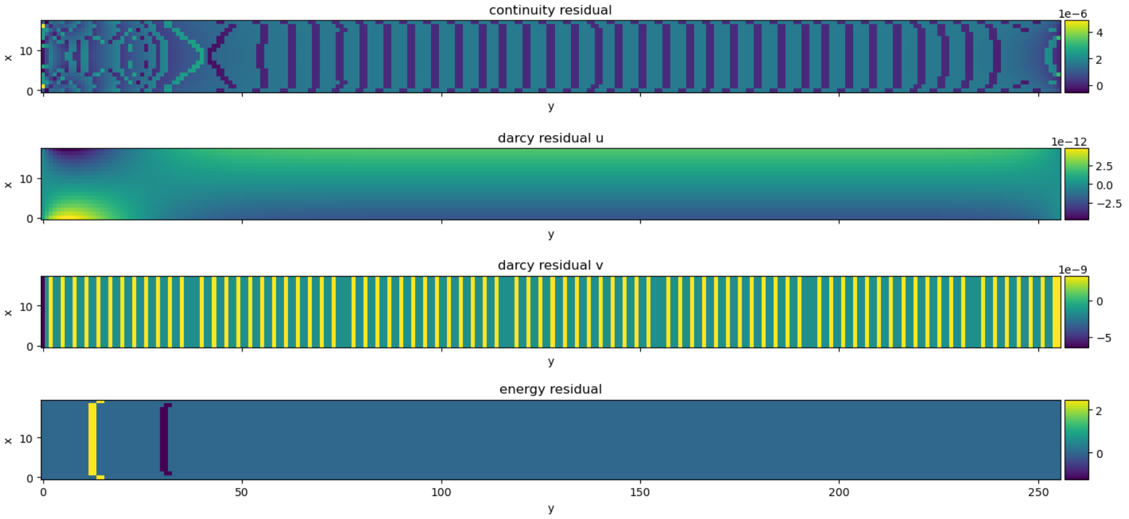


Figure 4.4: Residuals of (4.1) on a data sample with uniform permeability and without a heat pump.

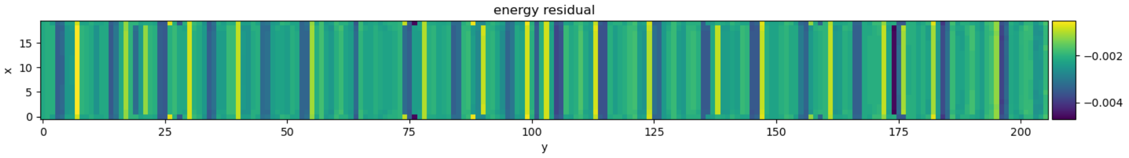


Figure 4.5: Residual of (4.1c) on a data sample with uniform permeability and without a heat pump. The starting region ($y < 50$) was cut away.

continuity and darcy residual in the v direction are also small. The energy residual, however, shows two large artifacts in the order of 10^0. When looking at the area with those artifacts cut away (Figure 4.5), we are in the order of 10^{-3}.

4.3.2 Non-Uniform Permeability Without Heat Pump

Next, we consider a non-uniform permeability sampled from Perlin noise. The heat pump is still turned off. The results are shown in figure 4.6. The continuity and Darcy velocity residuals are within the magnitude of single-point machine precision. All three show patterns that are not directly related to the permeability. In the case of the Darcy velocity residual, it looks like the patterns overlay Gaussian white noise. The energy residual now has patterns of large values (10^0) in the whole region.

4.3.3 Non-Uniform Permeability With Heat Pump

Finally, we consider the case that is also the one we perform super-resolution on. The permeability is sampled from Perlin noise, and a heat pump is present. The results, shown in Figure 4.7, are similar to the case without the heat pump, except for one major difference. Here we are looking at

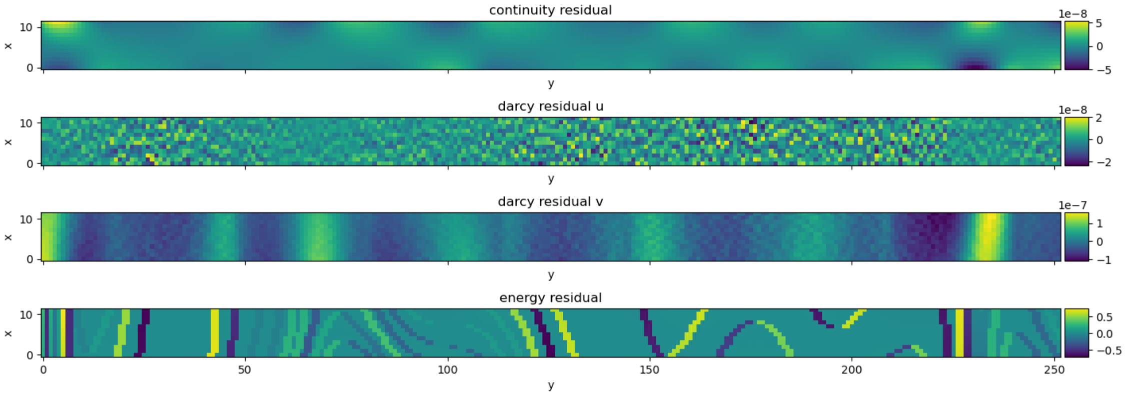


Figure 4.6: Residuals of (4.1) on a data sample with non-uniform permeability and without a heat pump.

the residuals without the source terms Q_e and Q_w, so we expect the source terms to show in the continuity and energy residuals. As can be observed, artifacts at the heat pump locations indeed overshadow the rest of the residual. However, there is also a source artifact in the Darcy residual in the v-direction and possibly in the u- direction, which is hard to see there. Those artifacts should not exist since the Darcy residual has no source terms. We ignore those extra artifacts since their values are in the same order of magnitude.

Our physics model does not fit the dataset perfectly. It is unclear if the observed patterns in the residuals and the high energy residual are caused by a mismatch between our model to the simulation model or the inner workings of the simulation. We saw that they are also present on simpler versions of the dataset.

Also, the structure of the source terms is unknown. According to the Pflotran documentation, the source terms have the form

$$Q_\mathrm{w} = q_\mathrm{M}\delta(\boldsymbol{r} - \boldsymbol{r}_{ss}) \tag{4.8}$$

where q_M is a mass rate and $\boldsymbol{r}_{ss}$ the location of the source term. The Dirac delta δ is defined to have an integral of one over the whole region, therefore we tried to fit a Gaussian distribution to the observed source term. However, we could not get them to match.

We describe our remedy to those problems in the next section.

4.4 Extension of the PINN Model

We extend the super-resolution model of Gao et al. [GSW21] described in 3.3.1 to apply it to our groundwater scenario. In addition to flow velocity in u and v-direction and pressure, we now also have temperature and permeability as inputs that we want to interpolate. The model already has separate paths for flow velocity and pressure, so the addition of temperature and permeability is straightforward. To handle the two additional variables, we simply introduce two new paths. The resulting model is shown in Figure 4.8. Every path in the model now has five inputs: flow velocity, pressure, temperature, and permeability.

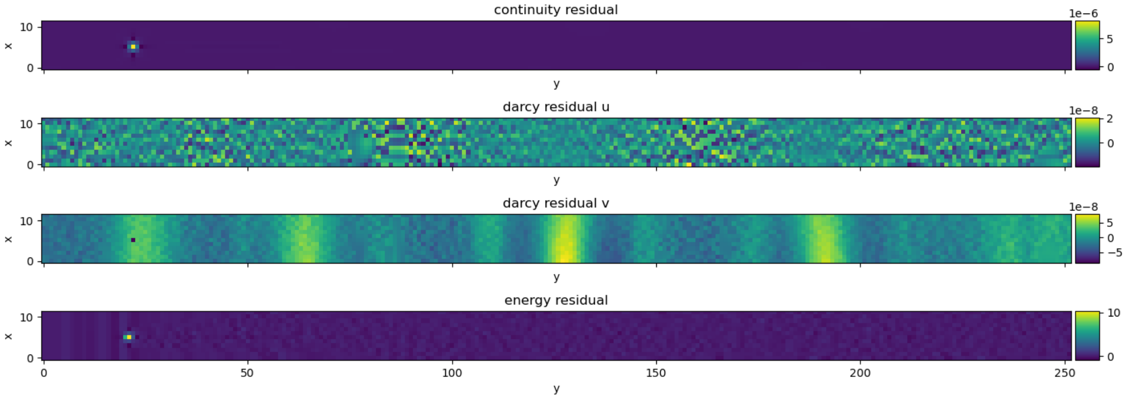


Figure 4.7: Residuals of (4.1) on a data sample with non-uniform permeability and with a heat pump.

We implemented our model with the Pytorch library [PGM+19]. To improve the neural network's learning, we normalized the input data set so that every variable has a mean of zero and a standard deviation of one [HQZ+55]. For our physics loss to work, we denormalized the network's output in every iteration before calculating the physics loss. The data loss is computed with the normalized output.

As described in Section 4.3, we could not fit the loss perfectly to the simulation dataset. We tried to remedy the two main problems, i.e., the high energy residual and the unknown shape of the input terms, with the following measures. On every training iteration, we calculate the continuity and energy residual also on the fine-resolution label. This gives us the input terms. We also scale the energy loss by a factor of 10^{-8} to be in the ballpark of the other residuals.

In [RRL+22] Ren et al. describe that imposing boundary conditions on the output of the neural network helps the learning process. For Dirichlet boundary conditions, they do this by padding the output with the boundary values before calculating the physics loss. We follow their implementation by interpolating the input to a resolution one smaller than the fine resolution and then imposing the boundary of the fine resolution into the output.

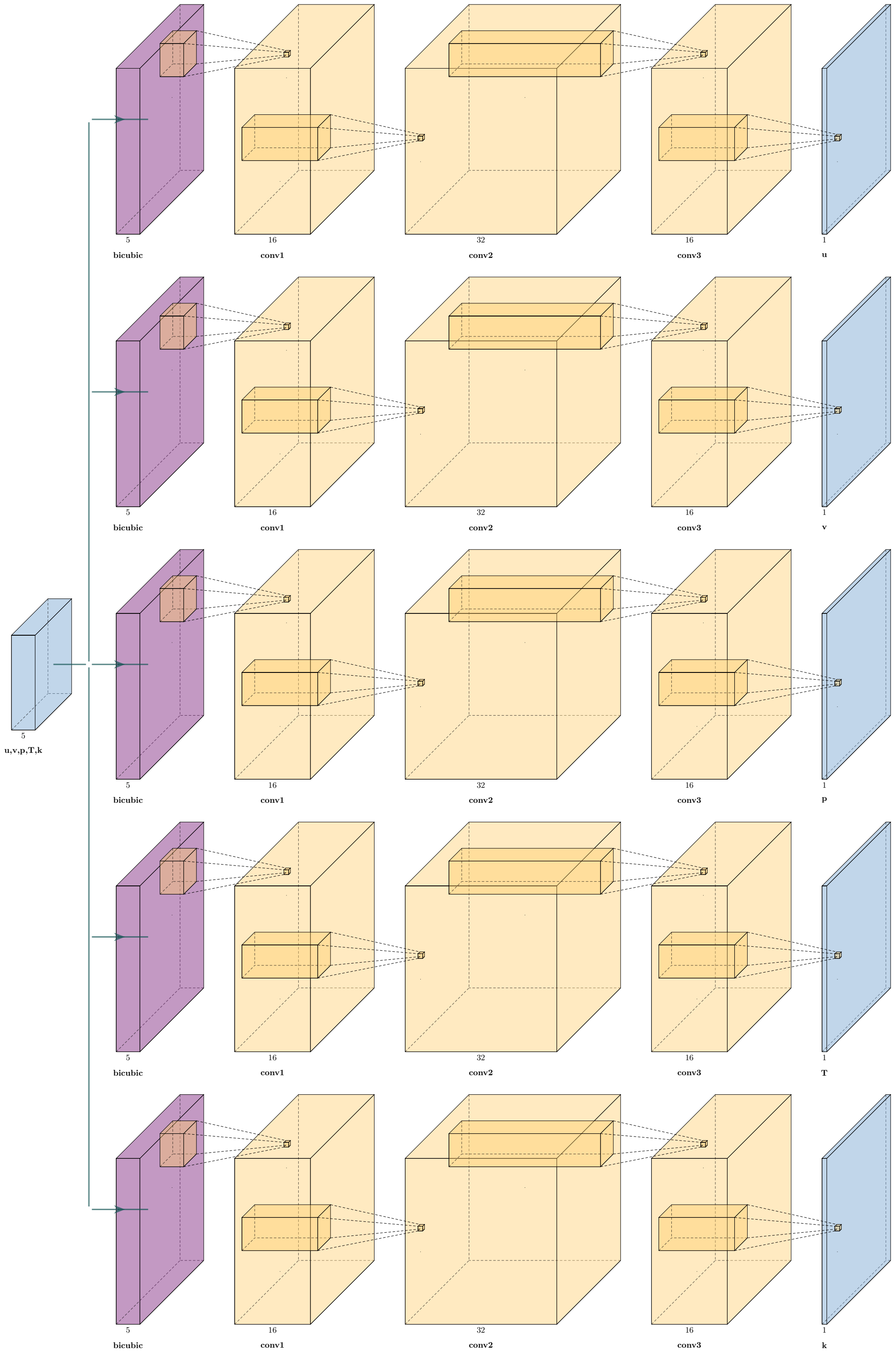


Figure 4.8: Extended model of our proposed network architecture for super-resolution in the groundwater scenario which has two additional input variables and paths in comparison to the architecture by Gao et al. (Figure 3.2).

5 Model Training and Evaluation

In this chapter, we elaborate on how our PINN was trained on the groundwater dataset. First, we describe some details about the training process. Then the trained model is compared to bicubic interpolation and a version of the model that is trained only with a data loss in order to assess the impact of the physics loss. Finally, the performance is evaluated on different resolution increases.

5.1 Training Details

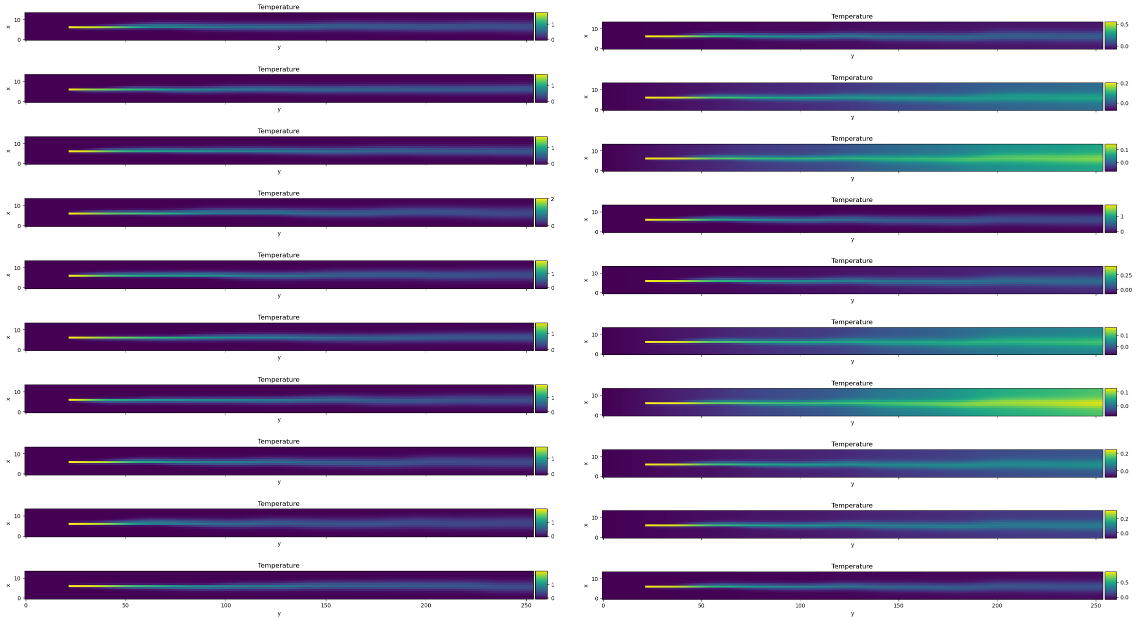
We split our data set into 70 training, 20 validation, and 10 test samples. To mitigate overfitting, we examined the dataset in order to find ten samples that would be most difficult for our network to predict, i.e., that the network has not seen before. For this, we looked at ten samples with constant pressure and varying permeability and ten samples where the pressure is varying and the permeability is constant. As Figure 5.1 shows, the samples with a constant permeability vary considerably more than the ones with constant pressure. Therefore the pressure has a greater impact on the developed thermal plume than the permeability. So we selected ten samples with a pressure distribution that the network will not see during training as the test set. The validation samples were chosen randomly among the remaining samples.

The model is trained for 5000 epochs with a batch size of 20 using the ADAM gradient descent algorithm [KB15]. During training, the model with the best score on the validation set is kept as the final model. With the validation set, we determined the weight of the physics loss compared to the data loss of 10^{-8}.

5.2 Comparison to a Data-Driven Network

We compare our trained PINN with bicubic interpolation and with a version of the neural network that was trained only with the data loss. For this evaluation, we used the 8 by 128 resolution as coarse data and the 16 by 256 resolution as fine data. Figure 5.2 shows the results on the ten test samples after training. Compared to bicubic interpolation we see a substantial improvement in the prediction accuracy for both training methods. Our model trained as a PINN with a physics loss performs slightly better than the model trained only with a data loss.

Figures 5.3, 5.4, and 5.5 show the result of one of the test samples for temperature and Darcy velocity in both directions. The neural network super-resolution approach is able to model sharp edges much better than the smoothing bicubic interpolation. This leads to substantially better results for the interpolated temperature. On other variables that are inherently smoother, the bicubic approach is on par or even better than the neural network.



(a) Constant pressure

(b) Constant permeability

Figure 5.1: Temperature plumes for constant pressure and varying permeability (a) and varying pressure with constant permeability (b).

5.3 Evaluation on Different Resolution Increases

We also trained the model on different combinations of coarse and fine resolutions from our dataset to see how the error changes for increasing interpolation factors. We used the finest resolution in our dataset (64 by 1024) as the target resolution and the other resolutions as coarse input. Figure 5.6 shows the mean of the total MSE on the test samples for bicubic interpolation and our trained PINN. The temperature field for the most extreme case, i.e., from 4 by 64 to 64 by 1024, is shown in Figure 5.7.

In the tests with multiple resolutions, it is interesting to see, that the error for the PINN network increases much faster than the bicubic error. Since the input to the first convolutional layer of our network is also the bicubic upscaled version of the input, the PINN error should in theory not overtake the bicubic error. In our tests with the interpolation from 4 by 64 to 64 by 1024, the network seems to overcompensate for the smooth input with negative values. On smaller increases in resolution, the first interpolation step seems to work well. But as the bicubic interpolation error increases this input to the convolutional layers gets smoother and hinders the ability of the network to preserve sharp edges.

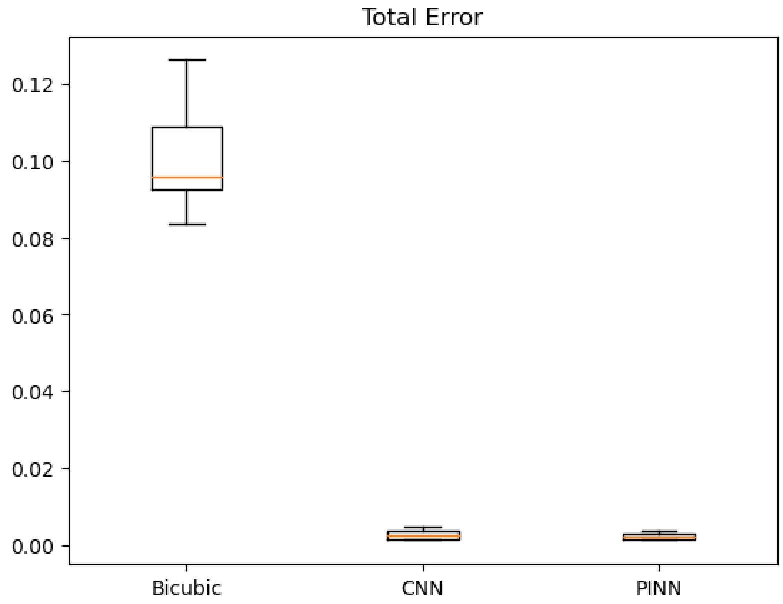


Figure 5.2: Mean squared error on the test samples for a resolution increase from 8 by 128 to 16 by 256. The test samples are selected as described in Section 5.1. The left result is from bicubic interpolation. The middle result is our model trained with only a data loss. On the right is our model trained as a PINN, i.e., with a data and physics loss.

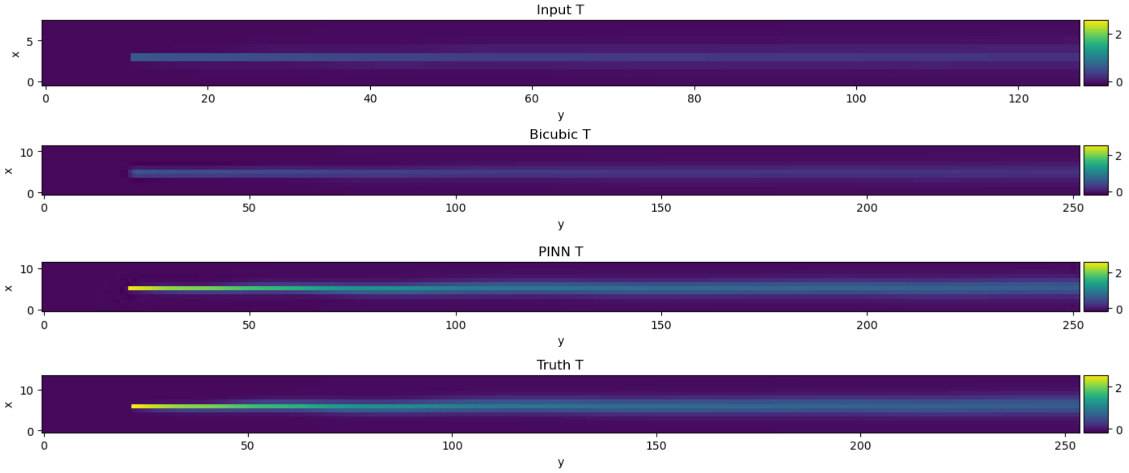


Figure 5.3: Interpolation results for temperature on a test sample for a resolution increase from 8 by 128 to 16 by 256. From top to bottom: The coarse input, bicubic interpolation, the output of our PINN, and the fine resolution ground truth. The values are normalized.

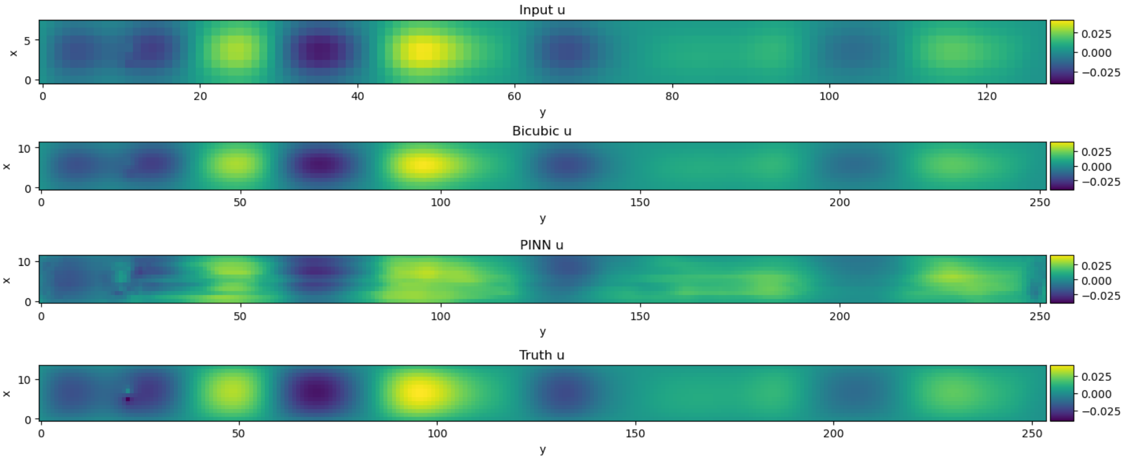


Figure 5.4: Interpolation results for Darcy velocity in u-direction on a test sample for a resolution increase from 8 by 128 to 16 by 256. From top to bottom: The coarse input, bicubic interpolation, the output of our PINN, and the fine resolution ground truth. The values are normalized.

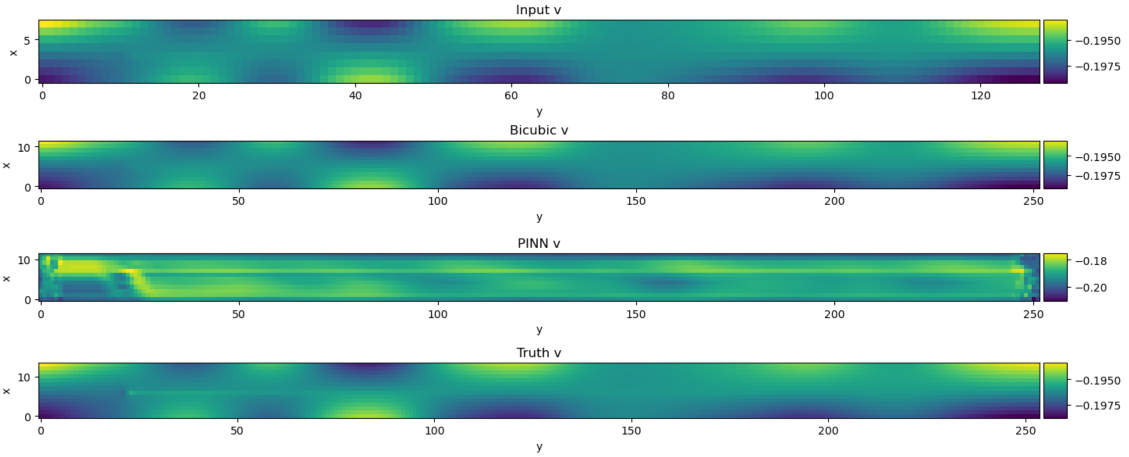


Figure 5.5: Interpolation results for Darcy velocity in v-direction on a test sample for a resolution increase from 8 by 128 to 16 by 256. From top to bottom: The coarse input, bicubic interpolation, the output of our PINN, and the fine resolution ground truth. The values are normalized.

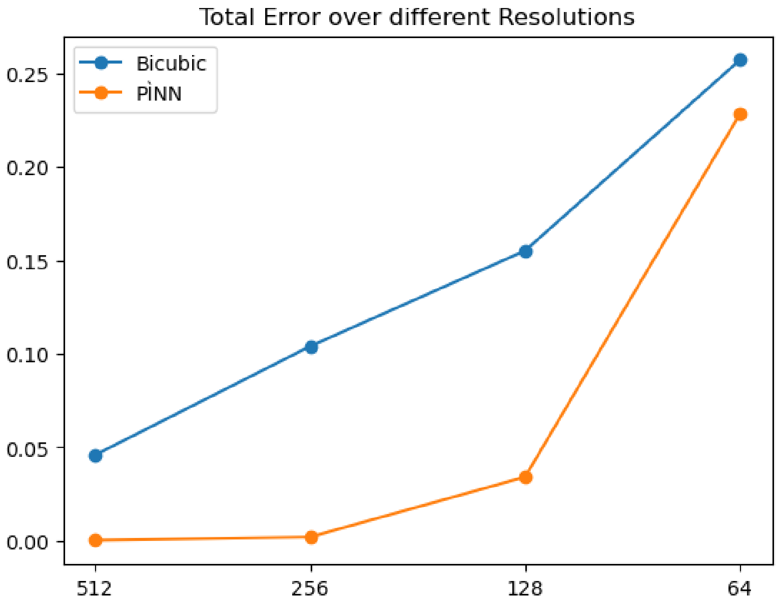


Figure 5.6: Mean of the total MSE on the test samples for bicubic interpolation and our trained PINN. Starting from an interpolation from 32 by 512 to 64 by 1024 up to an interpolation from 4 by 64 to 64 by 1024.

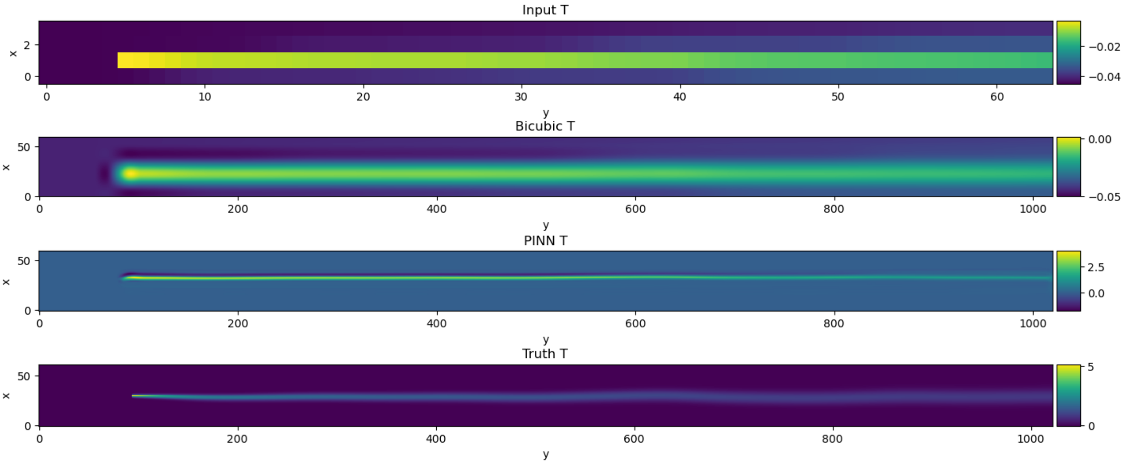


Figure 5.7: Interpolation results for temperature on a test sample for a resolution increase from 4 by 64 to 64 by 1024. From top to bottom: The coarse input, bicubic interpolation, the output of our PINN, and the fine resolution ground truth. The values are normalized.

6 Conclusion and Outlook

In this work, we applied a super-resolution PINN to a two-dimensional groundwater flow system, including a single heat pump used to cool a building. We extended the PINN model from Gao et al. [GSW21] which demonstrated promising results for cardiovascular blood flow to include water temperature and soil permeability. Our model was trained on a dataset generated from a groundwater simulation performed in Pflotran [HLM14].

After training our network on the groundwater simulation data, we see a substantial improvement in the prediction accuracy compared to bicubic interpolation. However, with the physics-trained network, we do not see significant improvements compared to a purely data-trained network like in [Aro22; BGL+21; GSW21; JEA+20; KRM22; RRL+22; ZZYZ22]. The only slightly better results of the PINN and the strong scaling we had to impose on the physics loss to achieve those results suggest that our physics loss acts more like a regularization instead of making up for unavailable training data. Notably, Li et al. [LM22], who considered a super-resolution PINN for multiphase flow, came to a similar conclusion.

Our physics loss does not fit the simulation data perfectly. We tried to mitigate the effects, but this mismatch is most likely the cause for our PINN model not performing better than the purely data-trained model. In our eyes, this gives the most potential for improvements. We tried to fit our model to an existing dataset. For future work, we suggest starting with a dataset and model that fit perfectly.

For more complex physical models, Krishnapriyan et al. [KGZ+21] found that the PINN approach can introduce problems into the learning process, hindering performance. They also provide solutions to address these problems. Since the groundwater model, especially with the equations of state, arguably counts as more complex, there might be a benefit in looking at these methods.

In our tests with multiple resolution increases, we see that the first bicubic interpolation step of our network architecture works well on smaller increases in resolution but seems to be a drawback for larger increases. Other network architectures like autoencoders [LDSP22], general adversarial networks [BODE20212617, Li], or networks with a pixel shuffle [SCH+16] might therefore lead to better results.

A logical direction for future work is furthermore the extension to three-dimensional interpolation and the addition of a temporal component. Since the output of PINNs is not required to fit the underlying physical model perfectly, it could also be interesting to investigate a coupled approach, similar to, e.g., [OVMC20], where the output of a super-resolution PINN is used as the initial value for a simulation to get a performance increase and convergence guarantees.

Bibliography

[Aro22] R. Arora. "PhySRNet: Physics informed super-resolution network for application in computational solid mechanics". In: *2022 IEEE/ACM International Workshop on Artificial Intelligence and Machine Learning for Scientific Applications (AI4S)*. 2022, pp. 13–18. DOI: 10.1109/AI4S56813.2022.00008 (cit. on pp. 16, 39).

[AWD21] A. Arzani, J.-X. Wang, R. M. D'Souza. "Uncovering near-wall blood flow from sparse data with physics-informed neural networks". In: *Physics of Fluids* 33.7 (2021), p. 071905. DOI: 10.1063/5.0055600 (cit. on p. 15).

[Bea79] J. Bear. *Hydraulics of groundwater*. McGraw-Hill, New York, 1979. ISBN: 0070041709 (cit. on p. 18).

[BGL+21] M. Bode, M. Gauding, Z. Lian, D. Denker, M. Davidovic, K. Kleinheinz, J. Jitsev, H. Pitsch. "Using physics-informed enhanced super-resolution generative adversarial networks for subfilter modeling in turbulent reactive flows". In: *Proceedings of the Combustion Institute* 38.2 (2021), pp. 2617–2625. ISSN: 1540-7489. DOI: https://doi.org/10.1016/j.proci.2020.06.022 (cit. on pp. 15, 16, 39).

[Bis95] C. M. Bishop. *Neural Networks for Pattern Recognition*. USA: Oxford University Press, Inc., 1995. ISBN: 0198538642 (cit. on p. 19).

[BNK20] S. L. Brunton, B. R. Noack, P. Koumoutsakos. "Machine Learning for Fluid Mechanics". In: *Annual Review of Fluid Mechanics* 52.1 (2020), pp. 477–508. DOI: 10.1146/annurev-fluid-010719-060214 (cit. on p. 15).

[CWF+21] S. Cai, Z. Wang, F. Fuest, Y. J. Jeon, C. Gray, G. E. Karniadakis. "Flow over an espresso cup: inferring 3-D velocity and pressure fields from tomographic background oriented Schlieren via physics-informed neural networks". In: *Journal of Fluid Mechanics* 915 (2021), A102. DOI: 10.1017/jfm.2021.135 (cit. on p. 15).

[DR08] W. Dahmen, A. Reusken. *Numerik für Ingenieure und Naturwissenschaftler*. Springer Berlin, Heidelberg, 2008. ISBN: 978-3-540-76493-9. DOI: https://doi.org/10.1007/978-3-540-76493-9 (cit. on p. 15).

[DSC22] S. R. Dubey, S. K. Singh, B. B. Chaudhuri. "Activation functions in deep learning: A comprehensive survey and benchmark". In: *Neurocomputing* 503 (2022), pp. 92–108. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2022.06.111 (cit. on p. 19).

[FFT20] K. Fukami, K. Fukagata, K. Taira. "Assessment of supervised machine learning methods for fluid flows". In: *Theoretical and Computational Fluid Dynamics* 34 (Aug. 2020). DOI: 10.1007/s00162-020-00518-y (cit. on pp. 12, 15).

[FFT21] K. Fukami, K. Fukagata, K. Taira. "Machine-learning-based spatio-temporal super resolution reconstruction of turbulent flows". In: *Journal of Fluid Mechanics* 909 (2021), A9 (cit. on pp. 12, 15).

[FSD+20] E. Ferdian, A. Suinesiaputra, D. J. Dubowitz, D. Zhao, A. Wang, B. Cowan, A. A. Young. "4DFlowNet: Super-Resolution 4D Flow MRI Using Deep Learning and Computational Fluid Dynamics". In: *Frontiers in Physics* 8 (2020). ISSN: 2296-424X. DOI: 10.3389/fphy.2020.00138 (cit. on pp. 12, 15).

[GMG+20] A. García-Gil, M. Mejías Moreno, E. Garrido Schneider, M. Á. Marazuela, C. Abesser, J. Mateo Lázaro, J. Á. Sánchez Navarro. "Nested Shallow Geothermal Systems". In: *Sustainability* 12.12 (2020). ISSN: 2071-1050. DOI: 10.3390/su12125152 (cit. on p. 11).

[GSS+15] N. Gerhardt, F. Sandau, A. Scholz, H. Hahn, P. Schumacher, C. Sager, F. Bergk, C. Kämper, J. Kräck, U. Lambrecht, O. Antoni, J. Hilpert, K. Merkel, T. Müller. *Interaktion EE-Strom, Wärme und Verkehr. Analyse der Interaktion zwischen den Sektoren Strom, Wärme/Kälte und Verkehr in Deutschland in Hinblick auf steigende Anteile fluktuierender Erneuerbarer Energien im Strombereich unter Berücksichtigung der europäischen Entwicklung Ableitung von optimalen strukturellen Entwicklungspfaden für den Verkehrs-und Wärmesektor*. 2015 (cit. on p. 11).

[GSW21] H. Gao, L. Sun, J.-X. Wang. "Super-resolution and denoising of fluid flow using physics-informed convolutional neural networks without high-resolution labels". In: *Physics of Fluids* 33.7 (July 2021), p. 073603. DOI: 10.1063/5.0054312 (cit. on pp. 12, 15, 16, 22, 23, 29, 39).

[HLLM12] G. Hammond, P. Lichtner, C. Lu, R. Mills. "PFLOTRAN: Reactive Flow and Transport Code for Use on Laptops to Leadership-Class Supercomputers". In: *Groundwater Reactive Transport Models* (Jan. 2012). DOI: 10.2174/978160805306311201010141 (cit. on pp. 18, 26).

[HLM14] G. E. Hammond, P. C. Lichtner, R. T. Mills. "Evaluating the performance of parallel subsurface simulators: An illustrative example with PFLOTRAN". In: *Water Resources Research* 50 (2014), pp. 208–228. DOI: 10.1002/2012WR013483 (cit. on pp. 12, 18, 25, 39).

[HQZ+55] L. Huang, J. Qin, Y. Zhou, F. Zhu, L. Liu, L. Shao. "Normalization Techniques in Training DNNs: Methodology, Analysis and Application". In: *IEEE Transactions on Pattern Analysis I& Machine Intelligence* 01 (Feb. 5555), pp. 1–20. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2023.3250241 (cit. on p. 30).

[HTF09] T. Hastie, R. Tibshirani, J. Friedman. *The Elements of Statistical Learning*. Second Edition. Springer New York, NY, 2009. ISBN: 978-0-387-84858-7. DOI: https://doi.org/10.1007/978-0-387-84858-7 (cit. on p. 19).

[JCLK21] X. Jin, S. Cai, H. Li, G. E. Karniadakis. "NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations". In: *Journal of Computational Physics* 426 (2021), p. 109951. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2020.109951 (cit. on p. 15).

[JEA+20] C. " Jiang, S. Esmaeilzadeh, K. Azizzadenesheli, K. Kashinath, M. Mustafa, H. A. Tchelepi, P. Marcus, Prabhat, A. Anandkumar. "MeshfreeFlowNet: A Physics-Constrained Deep Continuous Space-Time Super-Resolution Framework". In: *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. SC '20. Atlanta, Georgia: IEEE Press, 2020. ISBN: 9781728199986 (cit. on pp. 15, 16, 39).

[KB15] D. P. Kingma, J. Ba. “Adam: A Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Y. Bengio, Y. LeCun. 2015. URL: http://arxiv.org/abs/1412.6980 (cit. on pp. 21, 33).

[KGZ+21] A. S. Krishnapriyan, A. Gholami, S. Zhe, R. M. Kirby, M. W. Mahoney. “Characterizing possible failure modes in physics-informed neural networks”. In: *CoRR* abs/2109.01050 (2021). arXiv: 2109.01050. URL: https://arxiv.org/abs/2109.01050 (cit. on p. 39).

[KM22] A. Kashefi, T. Mukerji. “Physics-informed PointNet: A deep learning solver for steady-state incompressible flows and thermal fields on multiple sets of irregular geometries”. In: *Journal of Computational Physics* 468 (2022), p. 111510. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2022.111510 (cit. on p. 15).

[KRM22] D. Kelshaw, G. Rigas, L. Magri. *Physics-Informed CNNs for Super-Resolution of Sparse Observations on Dynamical Systems*. Oct. 2022 (cit. on pp. 15, 16, 39).

[KSH12] A. Krizhevsky, I. Sutskever, G. E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira, C. Burges, L. Bottou, K. Weinberger. Vol. 25. Curran Associates, Inc., 2012 (cit. on p. 20).

[LDSP22] R. Leiteritz, K. Davis, M. Schulte, D. Pflüger. *A Deep Learning Approach for Thermal Plume Prediction of Groundwater Heat Pumps*. Mar. 2022 (cit. on pp. 15, 39).

[LHL+20] P. C. Lichtner, G. E. Hammond, C. Lu, S. Karra, G. Bisht, B. Andre, R. T. Mills, J. Kumar, J. M. Frederick. *PFLOTRAN User Manual*. Tech. rep. http://documentation.pflotran.org. 2020 (cit. on pp. 18, 26).

[LHL+23] P. C. Lichtner, G. E. Hammond, C. Lu, S. Karra, G. Bisht, B. Andre, R. T. Mills, J. Kumar, J. M. Frederick. *PFLOTRAN Source Code*. Tech. rep. https://bitbucket.org/pflotran/pflotran/wiki/Home. 2023 (cit. on p. 26).

[LL87] L. Landau, E. Lifshitz. *Fluid Mechanics*. Second Edition. Pergamon, 1987. ISBN: 978-0-08-033933-7. DOI: https://doi.org/10.1016/B978-0-08-033933-7.50009-X (cit. on p. 17).

[LM22] M. Li, C. McComb. “Using Physics-Informed Generative Adversarial Networks to Perform Super-Resolution for Multiphase Fluid Simulations”. In: *Journal of Computing and Information Science in Engineering* 22.4 (Feb. 2022). 044501. ISSN: 1530-9827. DOI: 10.1115/1.4053671. eprint: https://asmedigitalcollection.asme.org/computingengineering/article-pdf/22/4/044501/6850327/jcise_22_4_044501.pdf (cit. on pp. 16, 39).

[LSK+17] B. Lim, S. Son, H. Kim, S. Nah, K. M. Lee. “Enhanced Deep Residual Networks for Single Image Super-Resolution”. In: July 2017, pp. 1132–1140. DOI: 10.1109/CVPRW.2017.151 (cit. on p. 15).

[LTH+17] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. “Photo-realistic single image super-resolution using a generative adversarial network”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4681–4690 (cit. on p. 15).

[MFR+20] R. Maulik, K. Fukami, N. Ramachandra, K. Fukagata, K. Taira. “Probabilistic neural networks for fluid flow surrogate modeling and data recovery”. In: *Physical Review Fluids* 5 (Oct. 2020). DOI: 10.1103/PhysRevFluids.5.104401 (cit. on pp. 12, 15).

[MG22] J. P. Molnar, S. J. Grauer. “Flow field tomography with uncertainty quantification using a Bayesian physics-informed neural network”. In: *Measurement Science and Technology* 33.6 (Mar. 2022), p. 065305. DOI: 10.1088/1361-6501/ac5437 (cit. on p. 15).

[MVP+18] Masson-Delmotte, P. Z. V., H.-O. Pörtner, D. Roberts, P. S. J. Skea, A. Pirani, W. Moufouma-Okia, C. Péan, R. Pidcock, S. Connors, J. Matthews, Y. Chen, X. Zhou, M. Gomis, E. Lonnoy, T. Maycock, M. Tignor, T. W. (eds.) “Global Warming of 1.5°C.An IPCC Special Report on the impacts of global warming of 1.5°C above pre-industrial levels and related global greenhouse gas emission pathways, in the context of strengthening the global response to the threat of climate change, sustainable development, and efforts to eradicate poverty”. In: *IPCC* (2018) (cit. on p. 11).

[OVMC20] O. Obiols-Sales, A. Vishnu, N. Malaya, A. Chandramowliswharan. “CFDNet”. In: *Proceedings of the 34th ACM International Conference on Supercomputing*. ACM, June 2020. DOI: 10.1145/3392717.3392772 (cit. on p. 39).

[Per02] K. Perlin. “Improving Noise”. In: *ACM Trans. Graph.* 21.3 (July 2002), pp. 681–682. ISSN: 0730-0301. DOI: 10.1145/566654.566636 (cit. on p. 25).

[PGM+19] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. 2019. arXiv: 1912.01703 [cs.LG] (cit. on p. 30).

[RBO+19] O. Ruhnau, S. Bannik, S. Otten, A. Praktiknjo, M. Robinius. “Direct or indirect electrification? A review of heat generation and road transport decarbonisation scenarios for Germany 2050”. In: *Energy* 166 (2019), pp. 989–999. ISSN: 0360-5442. DOI: https://doi.org/10.1016/j.energy.2018.10.114 (cit. on p. 11).

[RPK17a] M. Raissi, P. Perdikaris, G. E. Karniadakis. “Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations”. In: *arXiv preprint arXiv:1711.10561* (2017) (cit. on pp. 12, 15, 21).

[RPK17b] M. Raissi, P. Perdikaris, G. E. Karniadakis. “Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations”. In: *arXiv preprint arXiv:1711.10566* (2017) (cit. on pp. 12, 15, 21).

[RPK19] M. Raissi, P. Perdikaris, G. E. Karniadakis. “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational Physics* 378 (2019), pp. 686–707 (cit. on pp. 12, 15, 21).

[RRL+22] P. Ren, C. Rao, Y. Liu, Z. Ma, Q. Wang, J. Wang, H. Sun. “Physics-informed Deep Super-resolution for Spatiotemporal Data”. In: *ArXiv* abs/2208.01462 (2022) (cit. on pp. 15, 16, 30, 39).

[Rus95] W. S. Russell. “Polynomial interpolation schemes for internal derivative distributions on structured grids”. In: *Applied Numerical Mathematics* 17.2 (1995), pp. 129–171. ISSN: 0168-9274. DOI: https://doi.org/10.1016/0168-9274(95)00014-L (cit. on p. 15).

[SCH+16] W. Shi, J. Caballero, F. Huszar, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, Z. Wang. “Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, June 2016, pp. 1874–1883. DOI: 10.1109/CVPR.2016.207 (cit. on pp. 15, 39).

[SGHK20] K. Stengel, A. Glaws, D. Hettinger, R. King. “Adversarial super-resolution of climatological wind and solar data”. In: *Proceedings of the National Academy of Sciences* 117 (July 2020), p. 201918964. DOI: 10.1073/pnas.1918964117 (cit. on pp. 12, 15).

[SGPW20] L. Sun, H. Gao, S. Pan, J.-X. Wang. “Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data”. In: *Computer Methods in Applied Mechanics and Engineering* 361 (2020), p. 112732. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2019.112732 (cit. on p. 15).

[SWB+20] A. Subramaniam, M. L. Wong, R. Borker, S. Nimmagadda, S. K. Lele. “Turbulence Enrichment using Physics-informed Generative Adversarial Networks”. In: *ArXiv* abs/2003.01907 (2020) (cit. on p. 15).

[Ver67] D. Verein Deutscher Ingenieure. *International Formulation Committee of the 6th International Conference on the Properties of Steam*. Tech. rep. 1967 (cit. on p. 26).

[Whi86] S. Whitaker. “Flow in Porous Media I: A Theoretical Derivation of Darcy’s Law”. In: *Transport in Porous Media* 1 (Mar. 1986), pp. 3–25. DOI: 10.1007/BF01036523 (cit. on p. 18).

[XFCT18] Y. Xie, E. Franz, M. Chu, N. Thuerey. “tempoGAN: A Temporally Coherent, Volumetric GAN for Super-resolution Fluid Flow”. In: *ACM Transactions on Graphics (TOG)* 37.4 (2018), p. 95 (cit. on p. 15).

[YOH+22] Y. Yasuda, R. Onishi, Y. Hirokawa, D. Kolomenskiy, D. Sugiyama. “Super-resolution of near-surface temperature utilizing physical quantities for real-time prediction of urban micrometeorology”. In: *Building and Environment* 209 (2022), p. 108597. ISSN: 0360-1323. DOI: https://doi.org/10.1016/j.buildenv.2021.108597 (cit. on pp. 12, 15).

[ZBD+22] K. Zosseder, F. Böttcher, K. Davis, C. Haas, S. Halilovic, T. Hamacher, H. Heller, L. Odersky, V. Pauw, T. Schramm, S. M. and. *Schlussbericht zum Verbundprojekt GEO-KW: Kopplung des geothermischen Speicherpotenzials mit den wechselnden Anforderungen des urbanen Energiebedarfs zur effizienten Nutzung der regenerativen Energiequelle Grundwasser für die dezentrale Kälte- und Wärmebereitstellung in der Stadt*. de. Tech. rep. Bundesministerium für Wirtschaft und Klimaschutz, 2022. DOI: 10.14459/2022md1692003 (cit. on p. 11).

[ZZYZ22] M. Zayats, M. Zimoń, K. Yeo, S. Zhuk. *Super Resolution for Turbulent Flows in 2D: Stabilized Physics Informed Neural Networks*. Apr. 2022 (cit. on pp. 16, 39).

Declaration

I hereby declare that the work presented in this thesis is entirely my own and that I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

place, date, signature