Imperial College London
Department of Earth Science and Engineering
MSc in Applied Computational Science and Engineering

Independent Research Project
Final Report

Deep Learning for Solving PDEs

by
Nikolas Vornehm

Ncv19@imperial.ac.uk
GitHub login: acse-ncv19

Supervisors:
Dr Daniel Busby (TOTAL)
Dr Adriana Paluszny (Imperial College London)

August 2020

GitHub repository: https://github.com/acse-2019/irp-acse-ncv19

Abstract

A trend of solving physical problems using Deep Neural Networks rose within the last few years [2], [4], [5], [7], [9] and [10]. In this work we start investigating this approach for transport through porous media problems which commonly appear in oil & gas applications. Particularly to perform hydrocarbon production forecasts. Reservoir simulations involve solving two major problems: the forward problem consisting in modelling the evolution of a multiphase-phase flow in a subsurface domain; and the inverse problem also called history matching: consisting in tuning unknown reservoir parameters such as permeability or porosity to match observed well production data. The difficulty is to obtain an approximated solution to this problem using exclusively sparse observation data obtained from the injection and production wells [5]. Even though the oil & gas industry has been developing several methods to overcome this issue [11][12][13][14] it is still not computationally feasible to obtain satisfactory results for history matching. In this paper we present a Deep Neural Network method to approximate the solution to this problem. We present a method that produces synthetic permeability and porosity data that is not equal but nearly equivalent to the real permeability and porosity and yields a satisfying approximation of the real two-phase flow. Major advantages of this method compared to classical numerical simulations are mesh independency; derivatives with respect to reservoir parameters are computed at machine precision; it is computationally efficient.

Keywords: *Physics Informed Neural Networks, Permeability-Inversion, two-phase transport, Partial Differential Equations*

GitHub repository: https://github.com/acse-2019/irp-acse-ncv19

Table of Contents

GitHub repository: https://github.com/acse-2019/irp-acse-ncv19

Notation

α	*phase index (water, oil)*	
Γ	*boundary in space and time*	
g	*gravitational acceleration*	$[m\ s^{-1}]$
θ	*neural network weights and biases*	
K	*absolute permeability*	$[m^2]$
k_r	*relative permeability*	$[-]$
L_{index}	*loss function*	$[-]$
L	*length of domain in x-direction*	$[m]$
λ	*calibration parameter (domain specific)*	$[-]$
m	*mobility*	$[kg^{-1}\ ms]$
μ	*dynamic viscosity*	$[kg\ m^{-1}s^{-1}]$
N_{index}	*number of possible entries for "index"*	$[-]$
Ω	*spatial domain*	$[-]$
Pe	*Péclet number*	$[-]$
p	*pressure*	$[kg\ m^{-1}s^{-2}]$
Ψ	*potential*	$[kg\ m^{-1}s^{-2}]$
$\boldsymbol{q}$	*volumetric flow vector*	$[m^3s^{-1}]$
res	*residual*	$[-]$
ρ	*density*	$[kg\ m^{-3}]$
S	*saturation*	$[-]$

t	*time*	$[s]$
t_D	*dimensionless time*	$[-]$
$\boldsymbol{u}$	*velocity vector*	$[m\ s^{-1}]$
ϕ	*porosity*	$[-]$
x	*horizontal coordinate*	$[m]$
x_D	*dimensionless horizontal coordinate*	$[-]$
z	*vertical coordinate*	$[m]$

Introduction

In the last decade machine learning has experienced an immense growth due to the copious amounts of data and computing resources available in present days. Game changing results have been achieved across various scientific fields such as image recognition [15] and natural language processing [16]. However, when solving complex physical and engineering problems data is usually restricted or very expensive to obtain and often forces decision-making under lack of information. This drives the research towards data driven methods for analyzing such problems under sparse observation [4]. Especially the development of Physics Informed Neural Networks demonstrated the ability of neural networks to approximate physics described by Partial Differential Equation with respect to the few datapoints representing sparse observations [2], [4], [5], [7], [9] and [10].

Predicting the performance of an oil reservoir is a complex process that includes optimization and inverse problem solving, additionally to solving the governing Partial Differential Equations (PDEs) [5]. The high level of subsurface heterogeneity at multiple scales has a major influence on the fluid flow within an oil reservoir [6]. Optimizing the subsurface properties at feasible computational cost utilizing classical numerical simulators often results in scaling down accuracy to be able to work at feasible computational efficiency [5]. Classical numerical simulators rely on mesh-based discretization methods such as Finite Difference or Finite Elements [7]. However, no discretization is necessary when solving a PDEs using Physics Informed Neural Networks (PINNs). Instead, a PDE is being approximated by adding a PDE constrain in the loss function of a PINN [7]. This mesh independency is a major benefit which allows to work at low computational cost without scaling down accuracy. Additionally, PINNs mostly rely on learning based on the PDE constrain, which allows to produce feasible results even with small amounts of training data available [2].

The purpose if this thesis is to investigate the ability of PINNs to perform two-phase subsurface flow modelling by producing both, synthetic field data (permeability and porosity) and the solution to the PDE describing the subsurface flow. The method presented aims to produce an approximation at low computational cost. The main concept is the production of synthetic field data that yields a substantial approximation of the original subsurface flow.

The investigation starts with testing a one-dimensional PDE-regularization method using a dimensionless PDE to approximate an analytical solution. This method is then used to approximate a homogeneous simulation result. To be able to PDE-regularize the neural network on simulation data, the dimensionless equation needs to be calibrated by adding calibration parameters to the PDE. This calibration method is then applied to approximate the flow in a heterogenous domain. Once the flow is approximated the permeability and porosity still remain unknown. Using a method called Physics Constrained Learning we can then invert for the permeabilities and porosities. The inversion result is what we call synthetic field data. This synthetic field data can then be used to run a new simulation whose results will be an approximation of the original simulation results. The synthetic field data represents a “close to equivalent” permeability and porosity field to the original domain.

Literature Review

Related Research

Raissi et al. [4] introduce the *physics informed neural network* as a data-driven approach for solving Partial Differential Equations and physics informed surrogate-modelling. The paper provides empirical proof through supervised learning and PDE regularization for solving physical boundary value problems. They achieve to solve two main classes of problems: data-driven solution and data-driven discovery of partial differential equations. The major breakthrough is the data efficacy of the algorithms to reproduce high accuracy simulation data. Some of the generally non-linear PDEs describing the physics of the showcased problems also include shocks which the neural network also proofs to approximate well. The paper showcases PINN approximations to boundary value problems described by the following PDEs: Burgers Equation, Allen-Cahn Equation, incompressible Naiver-Stokes Equation, Schrödinger's Equation and Kortewegde-Vries Equation [4].

Fuks et al. [2] have done a study on the ability of neural networks to solve for two-phase flow in porous media. More specifically, they use the analytical Buckley-Leverett solution to provide test data. The model itself is trained with boundary data and regularized using the dimensionless version of the mass conservation law in porous media. They tested several flux functions (fractional flow) and came to the conclusion that neural networks fail to provide satisfying results when the loss function includes the non-linear hyperbolic conservation equation. Particularly this is the case when applying a non-convex flux function that involves shocks and mixed waves (shocks and rarefactions). However, they found that the neural network can accurately predict the solution when utilizing a parabolic form of the conservation equation. This is achieved by complementing the PDE regularization with a small amount of diffusion.

Another study on solving the Buckley-Leverett solution using neural networks was done in [8]. They also discover that neural networks have difficulties on providing satisfying solutions to this problem. Here they suggest two ways to achieve satisfying results using neural networks. The first suggestion is to overfit a neural network to solve a Buckley-Leverett solution and then apply these weights and biases for transfer learning. By retraining the last few layers of the neural network only, a solution can be approximated without having to re-learn the general shape of two-phase displacement. The second approach suggested is a Generative Adversarial Network (GAN) which consists of a discriminator network that differentiates physical solutions from fake solutions, a generator network that tries to trick the discriminator and a posterior network for the variational approximation of the true data. Additionally, they also discovered that adding a diffusion term to the PDE regularization accelerates the training by a factor of two and provides more accurate results.

A very recent approach for the use of physics informed neural networks was introduced in [9]. The paper introduces a combinatory implementation of a numerical simulation and a neural network to invert the permeability values of an injector-producer problem. This new method is called "*physics constrained learning*". As opposed to the classical "*penalty method*" (classical PINN) which uses a PDE regularization term in the loss function, physics constrained learning (PCL) uses a neural network to predict the field values of interests (permeability) and uses those predictions to perform a simulation. The PCL loss is then optimized with regards to the difference in water saturation of the simulation results obtained with the predicted permeability and the real water saturation. According to the paper, this method enjoys faster convergence with regards to the number of epochs and also with

regards to computing time, even though one optimization iteration takes more time due to the need of performing a simulation at every iteration.

Automatic Differentiation

Automatic differentiation is one method to obtain derivatives of expressions computationally. It is a hybrid approach combining numerical differentiation and symbolic differentiation. It keeps track of the numerical values of the derivative of an expression. By using the limited set of fundamental operations for which derivatives are known (i.e. $(x^2)' = 2x$) and combining them using the chain rule to obtain the differentiations of the original expression. This method allows to find derivatives at machine precision within a timeframe comparable to the time it takes to compute derivatives manually [8]. All code written for the purpose of this thesis is written in Julia (Version 1.3.1) using the ADCME library which is compiled with a TensorFlow1 (TF1) backend. TF1 enables the user to create and compile computational graphs which allow to obtain the derivatives of interest. Once the graph is set up it is used throughout the entire optimization process. It computes both, the derivatives to update the neural network's weights and biases during optimization and the derivatives for the PDE regularization through automatic differentiation. This static approach reduces the computational cost significantly [8].

Problem Specification

Qualitative problem description

In this thesis we investigate the problem of two-phase flow transport through porous media. We assume to have a subsurface domain with an injector well that injects a so-called wetting fluid (denoted with the subscript *w*) (source) and a producer well that extracts fluid(s) from the subsurface (sink).

Governing equations

The governing equations of the two-phase flow in porous media can be derived for each phase from the mass conservation law and the momentum conservation law [19].

The conservation of mass in porous media yields the following equation for each phase:

$$\phi \rho_\alpha \frac{dS_\alpha}{dt} + \nabla \cdot (\rho_\alpha \boldsymbol{u}_\alpha) = \rho_\alpha q_\alpha, \qquad \alpha = water, oil \tag{1}$$

The sum of the saturations satisfies:

$$\sum_{\alpha} S_{\alpha} = 1 \tag{2}$$

The conservation of momentum is described through the multi-phase version of Darcy's law:

$$\boldsymbol{u}_{\alpha} = -\frac{K k_{r\alpha}(S_{\alpha})}{\mu_{\alpha}} (\nabla p_{\alpha} - g \rho_{\alpha} \nabla z), \qquad \alpha = water, oil \tag{3}$$

[1][19]

All signatures can be found in the notation section.

Dimensionless Equation

The Methods used in the following investigations make use of the dimensionless form of the mass conservation for water (equation 1). To nondimensionalize this equation we can first introduce a non-convex flux function (fractional flow):

$$f_w(S_w) = \frac{m_w}{m_w + m_o} \tag{4}$$

with

$$\boldsymbol{u}_w = \boldsymbol{u}_{tot} * f_w(S_w) \tag{5}$$

[2][19]

So that the mass balance equation (equation 1) becomes:

$$\phi \frac{dS_w}{dt} + \nabla \cdot (f_w(S_w) * \boldsymbol{u}_{tot}) = q_w \tag{6}$$

Reducing this to a single dimension yields:

$$\phi \frac{dS_w}{dt} + u_{tot} * \frac{d}{dx}\big(f_w(S_w)\big) = 0 \tag{7}$$

Assuming constant velocity we can obtain the dimensionless variables

$$x_D = \frac{x}{L} \tag{8}$$

and

$$t_D = \int_0^t \frac{u_{tot}(K) * dt}{\phi * L} \tag{9}$$

This yields the following dimensionless equation

$$\frac{d}{dt_D}(S_w) + \frac{d}{dx_D}\left(f_w(S_w)\right) = 0 \tag{10}$$

[2]

Data

The Data used to train the neural networks presented in this thesis is either produced analytically or produced using an open source simulation code written in Julia available online @ https://lidongzh.github.io/FwiFlow.jl/dev/tutorials/flow/#footnote-pcl.

Analytical Buckley-Leverett solution

The Buckley-Leverett solution is a simplified one-dimensional analytical solution to the above introduced PDE (equation 1) that describes the displacement of the shock in water saturation.

Based on the fractional flow theory, the Buckley Leverett solution assumes the following:

- linear and horizontal flow
- water and oil are incompressible and immiscible
- homogeneous porosity and permeability
- total flow rate is constant over time and space
- gravity and capillary pressure are negligible
- fractional flow is only a function of water saturation

The last assumption yields the following flux function

$$f_w(S_w) = \frac{{S_w}^2}{{S_w}^2 + \frac{(1-S_w)^2}{\frac{\mu_o}{\mu_w}}} \tag{11}$$

[3].

Simulation

The simulation code simulates a 2D injector-producer-problem in porous media and is based on the governing equations previously introduced. It uses a non-linear implicit timestep discretization to solve the mass balance equation:

$$\phi(S_w^{n+1} - S_w^n) - \nabla \cdot (m_w(S_w^{n+1})K\nabla\Psi_w^n)\Delta t = \left(q_w^n + q_o^n \frac{m_w(S_w^{n+1})}{m_o(S_w^{n+1})}\right)\Delta t \quad \textbf{(12)}$$

[1]

where

$$m_\alpha = \frac{k_{r\alpha}}{\mu_\alpha} \quad \textbf{(13)}$$

[19]

and

$$\Psi_\alpha = p_\alpha - g\rho_\alpha z \quad \textbf{(14)}$$

With

$$k_{r\alpha} = (S_\alpha)^2 \quad \textbf{(15)}$$

From equation 7 we can derive the capillary potential as

$$\Psi_c = (\rho_w - \rho_o)gz \quad \textbf{(16)}$$

[1]

GitHub repository: https://github.com/acse-2019/irp-acse-ncv19

Neural Network for 1D dimensionless Mass Balance

Methodology

The dimensionless mass balance equation is solely dependent on water saturation. Therefore, the neural network only needs to predict water saturation by taking the spatial coordinate and the time as inputs. The initial condition and the boundary condition are assumed to be known as:

$$S_w(t_D = 0, x_D) = s_{w_0} = 0, \qquad \forall x_D \tag{17}$$

$$S_w(t_D, x_D) = s_{w_b}, \qquad x_D = 0, \quad t_D > 0 \tag{18}$$

These conditions will serve as training data and can be formulated as a classical loss function of the form:

$$L_\Gamma = \frac{1}{N_\Gamma} \sum_{\Gamma} \left| S_{w,i,real} - S_{w,i,NN} \right|^2, \qquad i \in \Gamma \tag{19}$$

Only the predictions at the time and space boundaries are used for this loss function (equation 19). In order to avoid overfitting the initial and boundary data, we use a PDE regularization term. The predictions fed into the PDE regularization can be obtained by inputting a finite set of colocation points (a set of pairs containing time and a space-coordinate) into the neural network. The neural network output will then be used to solve the PDE [4]. In this case the PDE is the dimensionless mass balance equation for the water phase (equation 10). As suggested in [2] and [8] we will add a diffusion term to the PDE in order for the neural network to converge. The regularization term now looks like the following:

$$\frac{d}{dt_D}(S_w) + \frac{d}{dx_D}\left(f_w(S_w)\right) - \frac{1}{Pe}\frac{d^2}{{dx_D}^2}(S_w) = res \cong 0 \tag{20}$$

The square of the residual of this equation is the regularization loss function to be minimized:

$$L_{PDE} = \frac{1}{N_{coloc}} \sum_{coloc} |res_{coloc}|^2 \tag{21}$$

The neural network itself is designed after the architecture suggested by Raissi et al [4]: Eight hidden layers each with 20 neurons, $tanh$ activation function and Xavier initialised weights [18].

GitHub repository: https://github.com/acse-2019/irp-acse-ncv19

Approximating Buckley-Leverett

Using the Buckley-Leverett solution we used a total of 101 spatial locations for $x_D \in [0,1]$ and 100 timesteps for $t_D \in]0,1]$ (Buckley-Leverett has no solution at $t = 0$ if $sw_0 = 0$). The initial and boundary conditions serve as training data and the remaining data serves as validation data. In addition to the training data we used a total of 10,100 colocation points for the PDE regularization.

The neural network converges fast and yields satisfying results after only 3000 Epochs with a relative error ($\mathcal{L}_2$-error) of $\sim 5.88\%$. The result is observable in figure 1. The smooth edges at the shock in the predicted water saturation result from the diffusion term [2]. For this example, we used a Péclet number of $Pe = 1/(2.5 * 10^{-3})$.

Figure 2 shows the result of the same neural network trained with the same data, same number of colocation points and the same number of epochs, but without diffusion term in the loss function. No further convergence was observable. It is obvious that the result is not satisfying. The relative error is $\sim 24.67\%$. This shows the importance of a parabolic loss function as pointed out in [2].

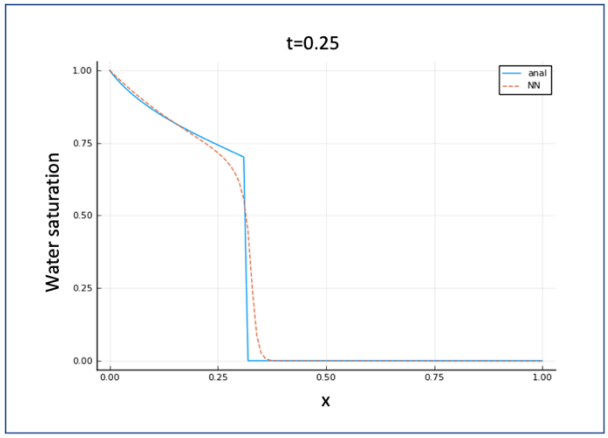

*Figure 1: Water saturation of 1-dimensional two phase flow (Buckley Leverette solution) of oil and water. Blue line is the analytical solution, dashed line is the solution provided by the Physics Informed Neural Network trained for 3000 Epochs. 10.000 colocation points. Loss function includes diffusion term: 1/Pe = 2.5 * 10e-3. Relative error: 0.05884980938350867*

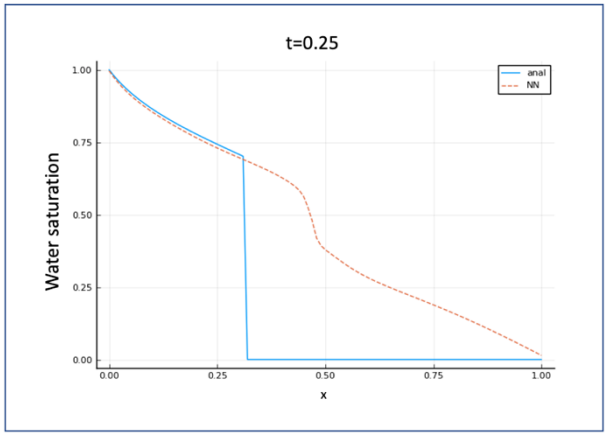

Figure 2: Water saturation of 1-dimensional two phase flow (Buckley Leverette solution) of oil and water. Blue line is the analytical solution, dashed line is the solution provided by the Physics Informed Neural Network trained for 3000 Epochs. 10.000 colocation points. Loss function not including diffusion term. Relative error: 0.24671658907705124

Approximating Numerical Simulation Data

We trained the same neural network using 1-dimensional simulation data. The data used for this test case has been produced using the simulation code introduced in the "**Data**" chapter. Since this simulation code solves the two-phase flow in porous media for two dimensions [1], we simply reduced the vertical dimension to a single cell. As a result, the simulation neglects gravity and capillary pressure (equation 16) [1].

The one-dimensional physical domain simulated has a horizontal length of 3000 meters. The initial water saturation is zero throughout the entire domain and the simulation runs for a total of 230 days. The simulation is spatially discretized into 100 cells. Each cell has a horizontal (and vertical) length of 30 meters. The time is discretized into 2300 timesteps. Each timesteps has a duration of 0.1 days. The

permeability is set to 20 millidarcy and the porosity is 25%. Both are homogeneous throughout the entire domain.

The neural network regularizes using the dimensionless mass balance equation as before (equation 20). When training the neural network to solve for the Buckley-Leverett solution we utilized a flux function based on the assumptions made for Buckley-Leverett (equation 4). To train the neural network to approximate the simulation data we need to switch to the original flux function (equation 4).

Known Homogeneous Subsurface Domain

As explained in "**Methodology**" the neural network input is the x-coordinate and the time. Now that we are working with the dimensionless equation, we need to rescale these parameters (equation 8 and 9) to obtain the dimensionless variables x_D and t_D. Since this is a homogeneous case, we can assume the velocity to be constant and compute by dividing the discharge with the cross-sectional flow area (cell size). This is a forward problem. Therefore, we only need initial and injection data to approximate it (no production data).

The results of the neural network prediction are shown in figure 3. This network has been trained for a total of 3000 epochs on the ADCME built-in BFGS optimizer using a total of 5151 colocation points. No further convergence was observable. The relative error is $\sim 2.04\%$.

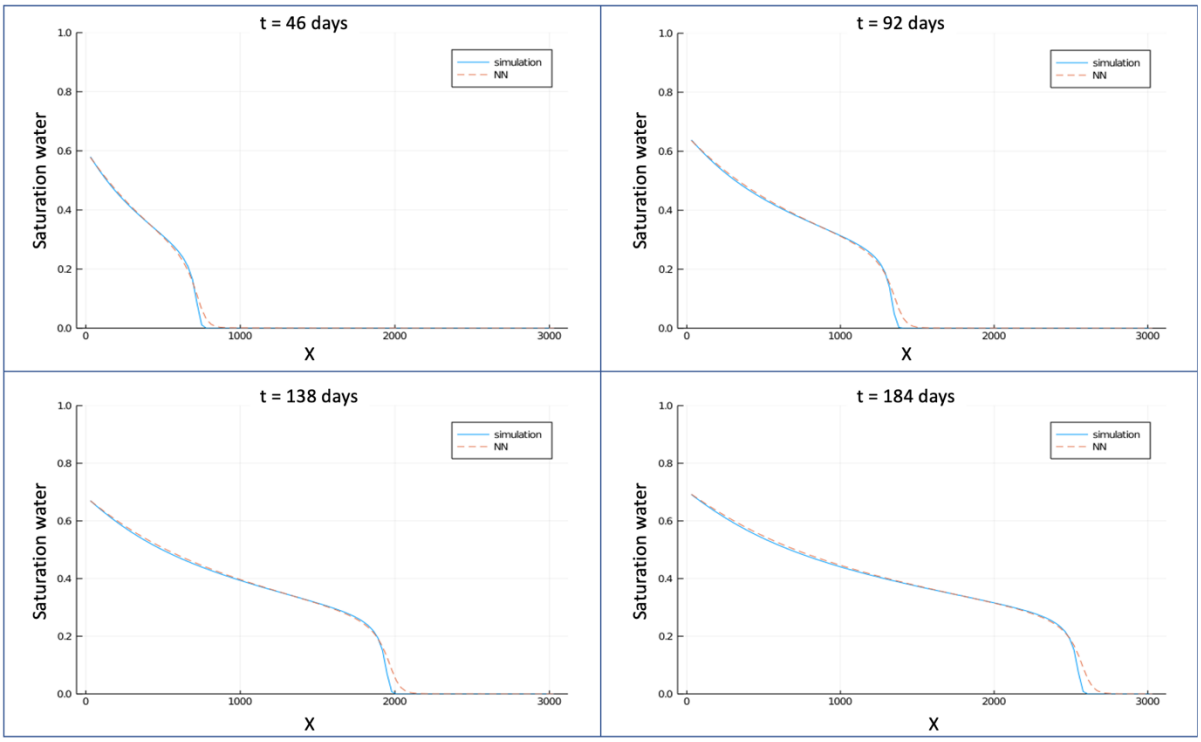

Figure 3: Result of neural network trained on simulation data. PDE regularization is dimensionless. Only initial condition and injection data is used for direct training. Training for 3000 epochs on ADCME built-in BFGS optimizer. 5151 colocation points. Relative error: 0.02048324264276119.

Unknown Homogeneous Subsurface Domain

If we assume that we are unaware of the permeability and porosity, we cannot compute the dimensionless time t_D. To be able to achieve satisfying results we suggest a method to calibrate the dimensionless equation in the PDE regularization term (equation 20) ([4] introduces a similar calibration method to solve for the incompressible Navier-Stokes equation). As a result, the new regularization term adapts the following form:

$$\lambda_1 \frac{d}{dt_D}(S_w) + \lambda_2 \frac{d}{dx_D}\left(f_w(S_w)\right) - \lambda_3 \frac{1}{Pe}\frac{d^2}{{dx_D}^2}(S_w) = 0 \qquad \textbf{(22)}$$

Assuming:

$$min(t_d) = 0; \quad max(t_D) = 1 \qquad \textbf{(23)}$$

Up until now we only used initial data and injection data for training and a set of colocation points for the PDE regularization (forward problem). Now that we are calibrating the PDE the problem has become an inverse problem. To be able to solve this inverse problem we need to train the neural network with additional output data (production data). This process is now called history matching. Once history matching has been performed successfully the model can be used to make future prediction.

In order to calibrate the PDE systematically we have run a number of tests to find out which λ to optimize for and which not. While each λ can be denoted as a TF1 variable inside the computational graph and will, therefore, be optimized, it can be denoted as a TF1 constant to avoid optimization and keep at the same value [17]. All λ are initialised with a value of 1.0. Table 1 shows the results for all possible combinations of λ. All combinations are trained for 3000 epochs with the ADCME built-in BFGS optimizer using 5151 colocation points.

OPTIMIZED $\boldsymbol{\lambda}$	LOSS	RELATIVE ERROR	$\boldsymbol{\lambda_1}$- $\boldsymbol{Value}$	$\boldsymbol{\lambda_2}$- $\boldsymbol{Value}$	$\boldsymbol{\lambda_3}$- $\boldsymbol{Value}$
$\boldsymbol{None}$	0.0512102	0.5569488	1.0	1.0	1.0
$\boldsymbol{\lambda_1}$	0.0020243	0.4396369	4.3331711	1.0	1.0
$\boldsymbol{\lambda_2}$	0.0001329	0.6646989	1.0	−0.0021189	1.0
$\boldsymbol{\lambda_3}$	0.0528604	0.5123455	1.0	1.0	0.2515622
$\boldsymbol{\lambda_1, \lambda_2}$	0.0001171	0.0183592	2.3563000	1.0398141	1.0
$\boldsymbol{\lambda_1, \lambda_3}$	0.0019885	0.3257937	3.5054264	1.0	1.007316
$\boldsymbol{\lambda_2, \lambda_3}$	0.0017585	0.8972965	1.0	−0.0008040	0.0875564
$\boldsymbol{\lambda_1, \lambda_2, \lambda_3}$	0.0000012	0.3956476	−0.0000013	0.0000005	−0.0000003

Table 1: Results of grid search to find best calibration method. Best result is marked with red colour. Training for 3000 epochs on ADCME built-in BFGS optimizer.

The best performing combination of calibration parameters is by far the optimization of λ_1 and λ_2. Its results yield a relative error of $\sim 1.83\%$. In the actual mass balance equation with dimensions

(equation 1) the first term is influenced by the porosity and the second term is influenced by the permeability. Therefore, we can make the conclusion that:

$$\lambda_1 \simeq f(\phi) \tag{24}$$

and

$$\lambda_2 \simeq f(K) \tag{25}$$

In a homogeneous subsurface domain, it is possible to describe the entire porosity field with a single value and the entire permeability field with another single value. Therefore, it is sufficient to calibrate the PDE using only two calibration parameters.

We did some further training on this combination of calibration parameters using two different optimizers. First, we trained the neural network for a total 5000 epochs on the TF1 Adam optimizer and afterwards for another 10.000 epochs on the ADCME built-in BFGS optimizer. The relative error scales down to $\sim 1.31\%$. The result can be observed in figure 4.

To improve results even further we tried to use transfer learning as suggested in [8]. To do so we overfitted a neural network on the Buckley-Leverett solution using the entire set of colocation points for both, training data and for the PDE regularization. However, this did not accelerate training and the relative error did not change significantly.

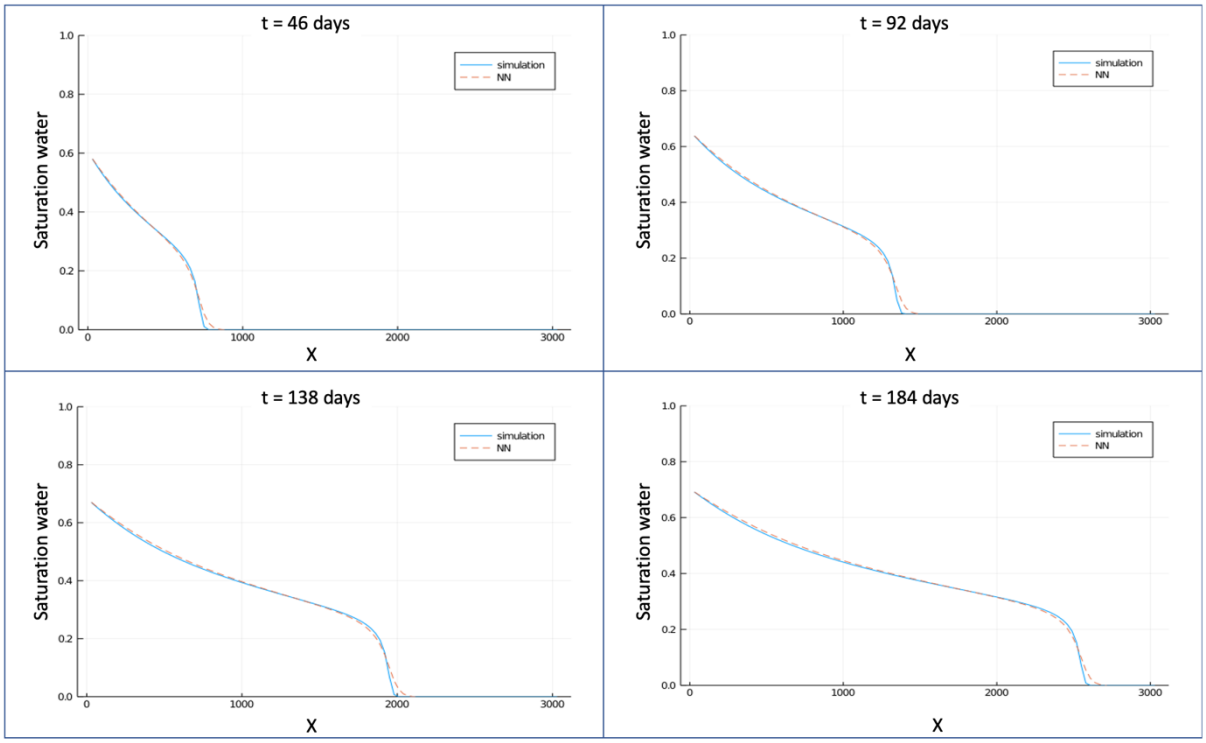

Figure 4: Neural Network prediction of homogeneous simulation data. Dimensionless water mass balance equation used for regularization. Calibration parameters: λ_1 calibrating saturation term and λ_2 calibrating flux-function. Total training for 5000 epochs on TF1 Adam optimizer and 10.000 epochs on ADCME built-in BFGS optimizer. Relative error: 0.013193972980944503.

Homogeneous Approximation of Heterogeneous Subsurface Domain

In a heterogeneous subsurface domain, we cannot make the assumption that the velocity is constant. This means that we cannot compute the dimensionless time. Therefore, we will use the same PDE regularization method calibrating for the parameters λ_1 and λ_2. As mentioned in the previous chapter we can only approximate for a homogeneous domain when using two calibration parameters. When using this same method on a heterogeneous domain we are approximating its subsurface flow by finding the best fitting homogeneous flow.

The training and test data for the heterogeneous case is also produced with the same simulation code [1] as the data for the homogeneous case. All input variables of the simulation are identical to the homogeneous simulation except for the porosity and the permeability. The porosity and the permeability are both heterogeneous and share a linear relationship (figure 5).

Again, we trained the neural network for 5000 epochs on the TF1 Adam optimizer and 10,000 epochs on the ADCME built-in BFGS optimizer. The total relative error is $\sim 9.62\%$. The results are observable in figure 6. As expected, the error is much larger than in the homogeneous case. Also, the neural network prediction is visibly much smoother than the plotted validation data as it is only the best fitting homogeneous flow rather than a heterogeneous approximation.

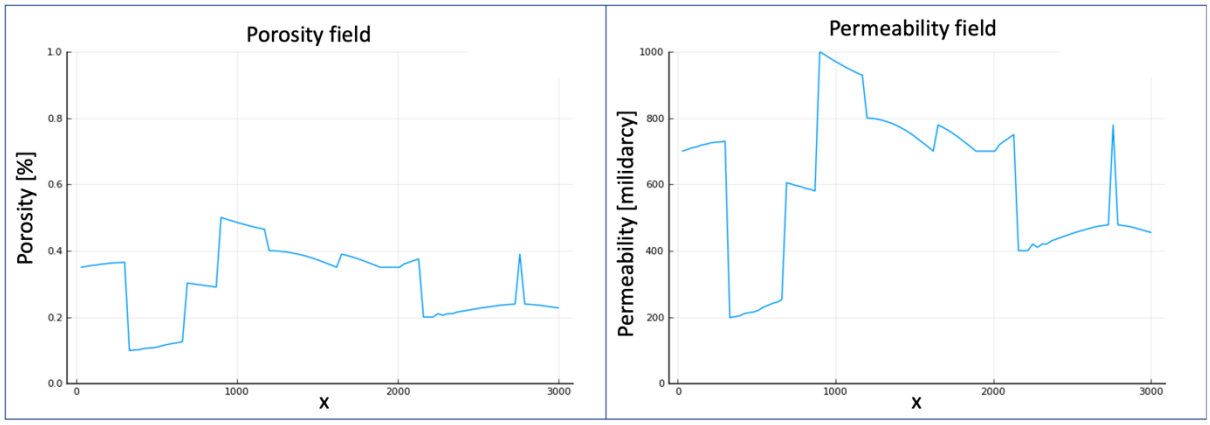

Figure 5: Left: porosity field; Right: permeability field

Heterogeneous Subsurface Domain

Another option is to discretise the permeability and porosity field. This is done by turning λ_1 and λ_2 into vectors. The vector's size is equal to the number of spatial colocation points at each timestep. By optimizing the values inside the λ-vectors a synthetic heterogeneity is being produced.

Using this method, we trained the neural network for 5000 epochs on the TF1 Adam optimizer and 10,000 epochs on the ADCME built-in BFGS optimizer. The total relative error is $\sim 20.88\%$. No further convergence has been observed. The results in figure 7 show that this method is not particularly successful and produces and error that is worse than the one produced by the best fitting homogeneous approximation.

GitHub repository: https://github.com/acse-2019/irp-acse-ncv19

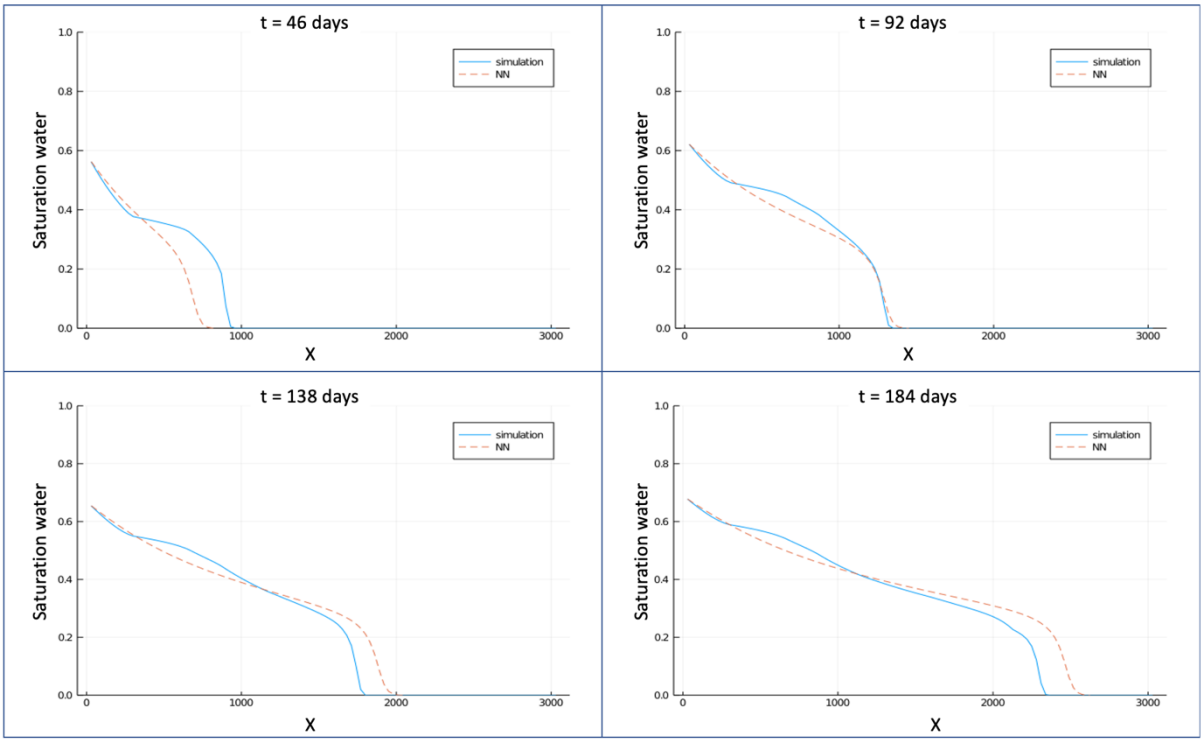

Figure 6: Neural Network prediction of heterogeneous simulation data. Dimensionless water mass balance equation used for regularization. Calibration parameters: $\lambda_1$calibrating saturation term and $\lambda_2$calibrating flux-function. Total training for 5000 epochs on TF1 Adam optimizer and 10.000 epochs on ADCME built-in BFGS optimizer. Relative error: 0.09623160382748334.

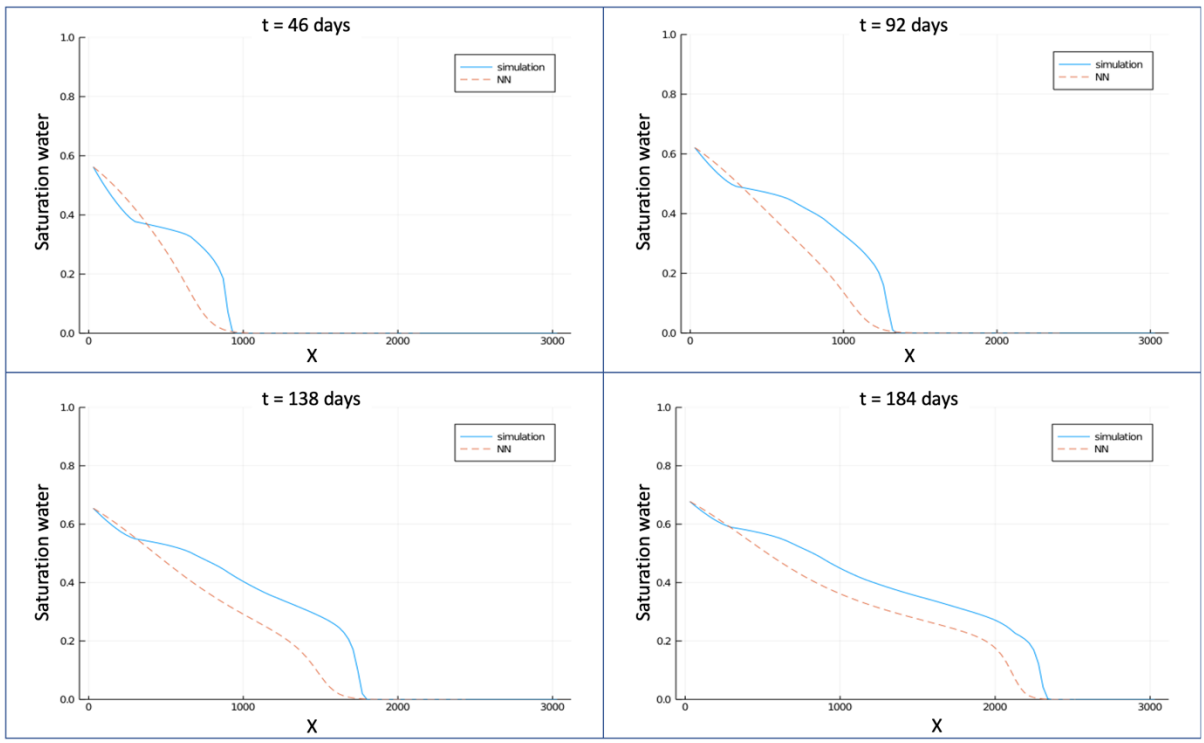

Figure 7: Neural Network prediction of heterogeneous simulation data. Dimensionless water mass balance equation used for regularization. Calibration vectors: $\lambda_1$calibrating saturation term and $\lambda_2$calibrating flux-function discretized into 51 points. Total training for 5000 epochs on TF1 Adam optimizer and 10.000 epochs on ADCME built-in BFGS optimizer. Relative error: 0.20884981153335658.

Predicting Permeability and Porosity

To produce actual permeability and porosity values we can use a method introduced in [9]. It is called Physics Constrained Learning (PCL). In this case PCL takes initial values for the porosity and permeability and then performs a simulation that produces saturation data (same simulator used as before). With this saturation data a loss function can be formulated that computes the squared difference between the saturation data produced with the initial field data (permeability and porosity) and the saturation data of the neural network prediction from the previous chapter.

$$L_{\Omega} = \sum_{\Omega} \left| S_{w,i,K\_pred} - S_{w,i,NN} \right|^2, \qquad i \in \Omega \tag{26}$$

This loss function can be minimized by optimizing the initial values for permeability and porosity. The optimized permeability and porosity are called synthetic field data. This method converges within few iterations and is, therefore, fairly efficient even though every iteration requires a new simulation [9].

To get a final measure of accuracy we can run the original simulation and a simulation that uses the synthetic field data and compute a relative error between the both. Since this is an inverse problem that requires injection and production data to be solved, it would be useful to know how it performs for future prediction. To do this we can simply run both simulations until reaching steady state and again compute a relative error. This relative error would be a test error.

Inverting Best Fitting Homogeneous Flow

The data for the loss function is going to be the data predicted using the single value λ method. When inverting permeabilities for the best fitting homogeneous flow the result is going to be close to homogeneous, as well. For the optimization process we used the TF1 Adam optimizer for 100 iteration. The final relative error computed as described in the previous paragraph is $\sim 8.86\%$ which is slightly better than without producing synthetic field data. If we run the simulations until reaching steady state, we even get a lower relative error of $\sim 5.24\%$ regarding the entire simulation time. The synthetic field data produced by the inversion can be observed in figure 9 and the results for saturation in figure 10 and 11.

GitHub repository: https://github.com/acse-2019/irp-acse-ncv19

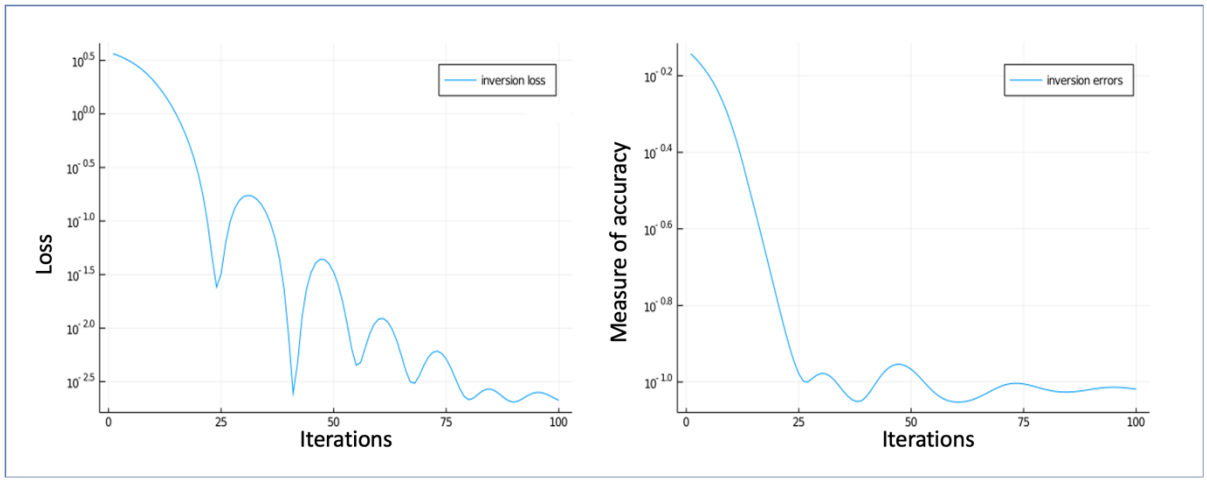

Figure 8: Left: Optimization loss over iteration. Right: Relative error between simulation with real permeability and porosity and simulation with inverted permeability and porosity. Final relative error: 0.08863445644824573.

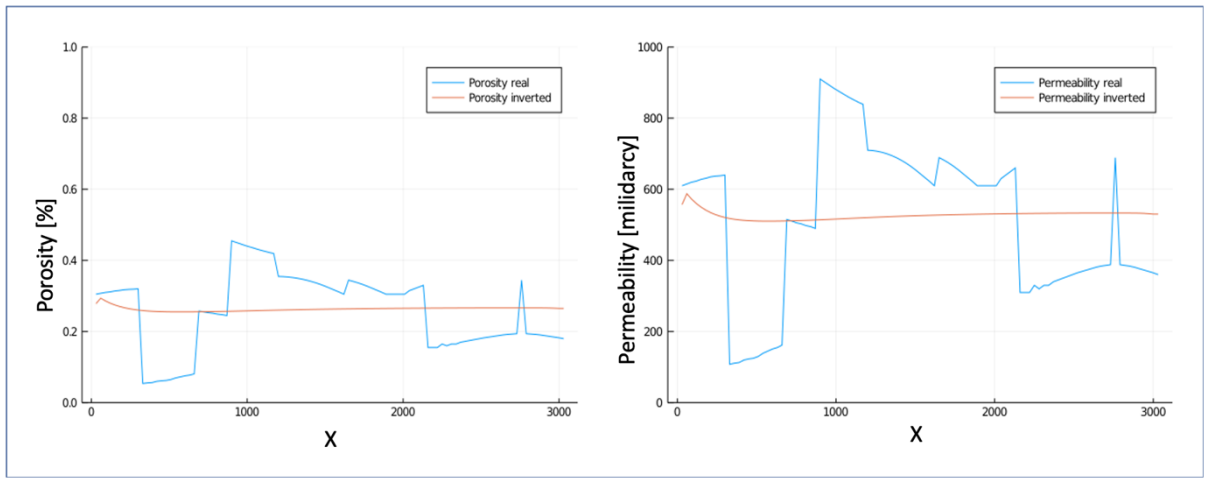

Figure 9: Synthetic field data.

GitHub repository: https://github.com/acse-2019/irp-acse-ncv19

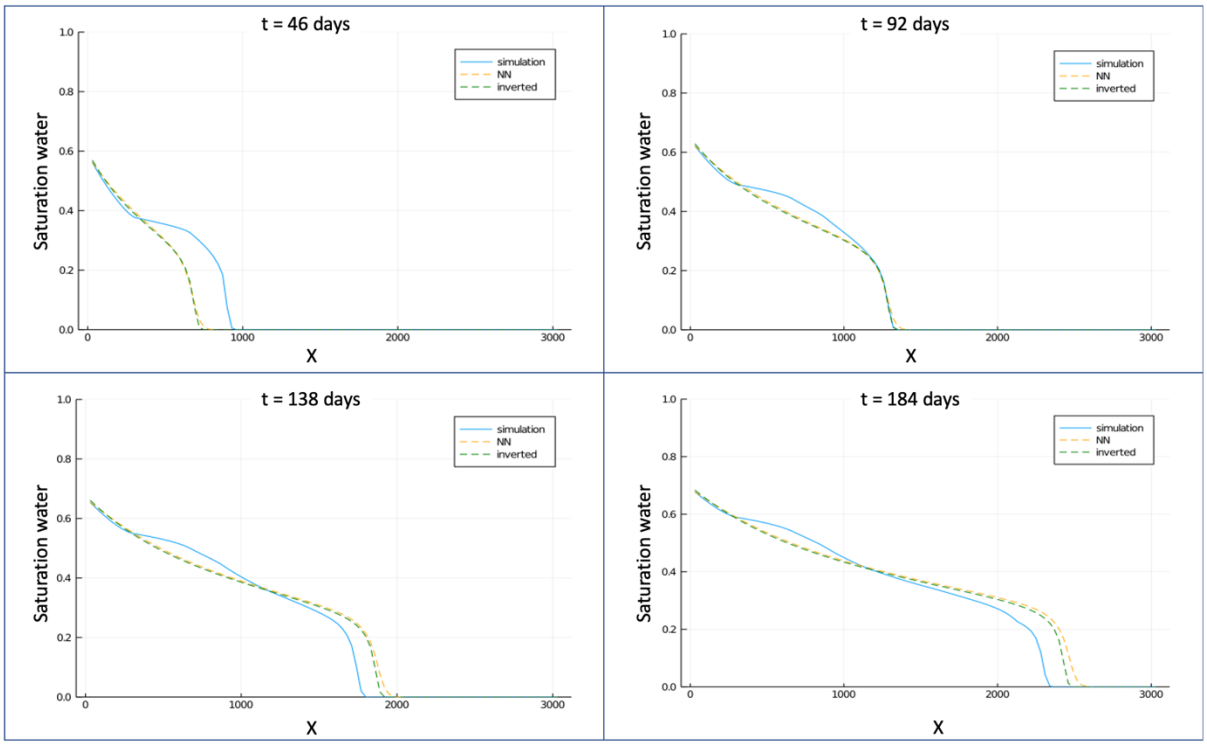

Figure 10: Blue: Two phase flow in porous media simulation in heterogeneous domain; Orange: Neural network prediction of simulation using single value lambda calibration; Green: simulation of two-phase flow in porous media with inverted permeability and porosity based on neural network prediction. Relative error between simulation with synthetic (inverted) field data and original simulation: 0.08863445644824573.

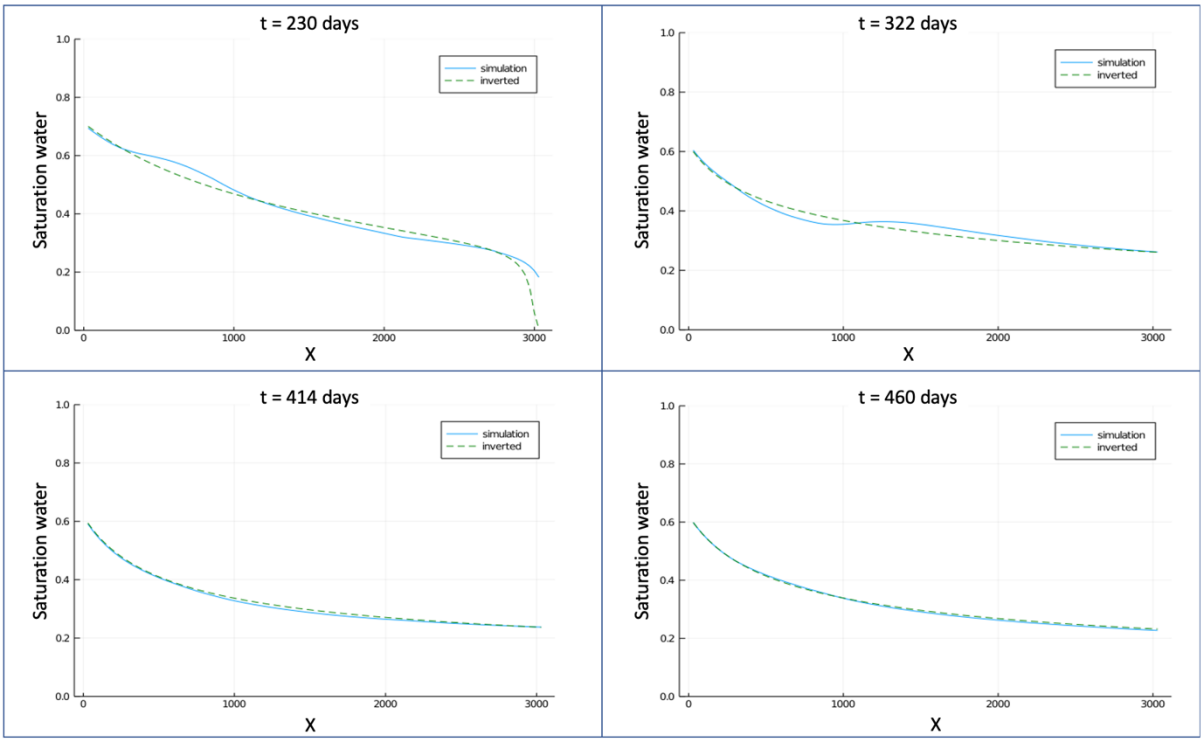

Figure 11: Blue: Two phase flow in porous media simulation in heterogeneous domain until reaching steady state; Green: simulation of two-phase flow in porous media with inverted permeability and porosity based on neural network prediction. Relative error between simulation with synthetic (inverted) field data and original simulation: 0.05247430801073294.

Conclusion

This combination of methods manages to successfully approximate a two-phase subsurface flow within an unknown permeability and porosity field at low computational cost. Once the inversion problem is solved it also proves to be able to make future predictions with a lower but similar error. Since the approximation is only a homogeneous approximation there are limits to its accuracy. Some further research could be done by increasing the number calibration parameters as shown in the "**Heterogeneous Subsurface Domain**"-chapter. However, the results show that neural networks have a potential of solving such complex inverse problems without any knowledge of the subsurface domain.

GitHub repository: https://github.com/acse-2019/irp-acse-ncv19

Bibliography

[1] LIDONGZH.GITHUB.IO. (2020). FLOW INVERSION · FWIFLOW. [ONLINE] AVAILABLE AT: < https://lidongzh.github.io/FwiFlow.jl/dev/tutorials/flow/> [ACCESSED 25 JUNE 2020].

[2] FUKS, O.; TCHELEPI, H. (2020). LIMITATIONS OF PHYSICS INFORMED MACHINE LEARNING FOR NONOLINEAR TWO-PHASE TRANSPORT IN POROUS MEDIA. STANDFORD: BEGELL HOUSE, PP.1-10

[3] AMOOIE, M. (2016). ANALYSIS OF THE BUCKLEY-LEVERETT SOLUTION AND COMPARISON WITH NUMERICAL SIMULATION. COLUMBUS: THE OHIO STATE UNIVERSITY, PP.1-3

[4] RAISSI, M.; PERDIKARIS, P.; KARNIADAKIS, G. (2017). PHYSICS INFORMED DEEP LEARNING (PART 2): DATA-DRIVEN SOLUTIONS OF NONLINEAR PARTIAL DIFFERENTIAL EQUATIONS. PROVIDENCE: JOURNAL OF COMPUTATIONAL PHYSICS, PP.1-10

[5] GENEVA, N.; ZABARAS N. (2019). MODELING THE DYNAMICS OF PDE SYSTEMS WITH PHYSICS-CONSTRAINED DEEP AUTO-REGRESSIVE NETWORKS. NOTRE DAME: JOURNAL OF COMPUTATIONAL PHYSICS, PP.2-3

[6] LIE, K.; MALLISON, B. (2015). MATHEMATICAL MODELS FOR OIL RESERVOIR SIMULATION. OSLO: SINTEF DIGITAL, P.1

[7] LU, L.; MENG, X.; MAO, Z.; KARNIADAKIS, G. (2020). DEEPXDE: A DEEP LEARNING LIBRARY FOR SOLVING DIFFERENTIAL EQUATIONS. PROVIDENCE: ARXIV, P.1

[8] FRACES, C.; PAPAIOANNOU, A.; TCHELPI, H. (2020). PHYSICS INFORMED DEEP NEURAL NETWORKS FOR TRANSPORT IN POROUS MEDIA. BUCKLEY LEVERETT PROBLEM. STANFORD: ARXIV, PP.1-16

[9] XU, K.; DARVE, E. (2020). PHYSICS CONSTRAINED LEARNING FOR DATA-DRIVEN INVERSE MODELING FROM SPARSE OBSERVATION. STANFORD: ARXIV, PP.1-16

[10] LU, L.; JIN, P.; KARNIADAKIS, G. (2020). DEEPONET: LEARNING NONLINEAR OPERATORS FOR IDENTIFYING DIFFERENTIAL EQUATIONS BASED ON THE UNIVERSAL APPROXIMATION THEOREM OF OPERATORS. PROVIDENCE: ARXIV, P.1

[11] KLIE, H.; FLOREZ, H. (2018). DATA-DRIVEN MODELING OF FRACTURED SHALE RESERVOIRS. BARCELONA: 16TH EUROPEAN CONFERENCE ON THE MATHEMATICS OF OIL RECOVERY, P.1

[12] BATYCKY, R.; THIELE, M.; BLUNT, M. (1997). A STREAMLINE-BASED RESERVOIR SIMULATION OF THE HOUSE MOUNTAIN WATERFLOOD. STANFORD: STANFORD CENTRE FOR EARTH RESOURCE FORECASTING, P.1

[13] DARTS. (2019) DELFT ADVANCED RESEARCH TERRA SIMULATOR

[14] DE HOOP, S.; VOSKOV, D.; VOSSEPOEL, F.; JUNG A (2018). QUANTIFICATION OF COARSENING EFFECTS ON RESPONSE UNCERTAINTY IN RESERVOIR SIMULATION. STANFORD: STANFORD CENTRE FOR EARTH RESOURCE FORECASTING, P.1

GitHub repository: https://github.com/acse-2019/irp-acse-ncv19

[15] KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. (2012). IMAGENET CLASSIFICATION WITH DEEP CONVOLUTIONAL NEURAL NETWORKS. TORONTO: NIPS 2012, PP, 1097-1105

[16] LECUN, Y.; BENGIO, Y.; HINTON, G. (2015). DEEP LEARNING. TORONTO: NATURE, P.521, 436-444

[17] TENSORFLOW (N.D.). TF.TENSOR | TENSORFLOW CORE V1.15.0. [ONLINE] AVAILABLE AT: <https://www.tensorflow.org/versions/r1.15/api_docs/python/tf/Tensor> [ACCESSED 26 AUG. 2020].

[18] GLOROT, X.; BENGIO, Y.; (2010). UNDERSTANDING THE DIFFICULTY OF TRAINING DEEP FEEDFORWARD NEURAL NETWORKS. IN INTERNATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE AND STATISTICS, PP.249-256

[19] CHEN, H.; SUN, S.; (2020). A NEW PHYSICS-PRESERVING IMPES SCHEME FOR INCOMPRESSIBLE AND IMMISCIBLE TWO-PHASE FLOW IN HETEROGENEOUS POROUS MEDIA. ELSEVIER: FUJIAN, P.3