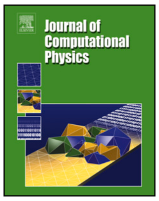
Journal of Computational Physics 516 (2024) 113299

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp



On Physics-Informed Neural Networks training for coupled hydro-poromechanical problems

Caterina Millevoi [a,*], Nicolò Spiezia [b], Massimiliano Ferronato [a]

[a] Department of Civil, Environmental and Architectural Engineering, University of Padova, Italy
[b] M3E Srl, Padova, Italy

ARTICLE INFO

Keywords:
Physics-Informed Neural Networks
Coupled hydro-poromechanics
Data-driven scientific computing
Model architecture
Sensor-driven training

ABSTRACT

The robust and efficient numerical solution of coupled hydro-poromechanical problems is of paramount importance in many application fields, in particular in geomechanics and biomechanics. Even though the solution by means of the Finite Elements Method (FEM) still remains the preferred option, Physics-Informed Neural Networks (PINNs) are rapidly gaining attention as a powerful and promising approach. By involving the residual of the governing PDEs as a constraint in the training, PINNs combine a physics-driven approach together with a data-driven one, leveraging on deep learning techniques.
This work focuses on coupled hydro-poromechanical processes, where a PINN-based approach is implemented and investigated in classical benchmarks. An analysis of the influence of the hyper-parameter selection has been performed to study the model sensitivity to PINN architecture and identify the most appropriate one. To take advantage of PINN ability to integrate data and reduce the complexity of the solution of the coupled hydro-poromechanical problem, a "sensor-driven" training is proposed where data are provided at locations inside the domain even to solve a forward problem. The goal is to assess and validate the approach, thus contributing to the foundation of this method in coupled hydro-poromechanics.

1. Introduction

Coupled hydro-poromechanics describes important processes occurring in many different fields, in particular in sectors of geomechanics, hydrogeology, and biomechanics. Timely challenges, such as gas, water, CO2, hydrogen injection and/or withdrawal from deep reservoirs, or the interaction between biological tissues and fluid flow, can be accurately described only by considering a fully coupled formulation. In essence, hydro-poromechanics consists of the simultaneous action of fluid flow and solid deformation in fully or partially saturated porous media. The native theory has been developed starting from K. Terzaghi's one-dimensional consolidation in 1925 [1] and the three-dimensional extension by M. A. Biot in 1941 [2], with successive thermodynamically robust formulations available nowadays [3,4]. Nevertheless, the numerical solution of hydro-poromechanical partial differential equations (PDEs) is still an active subject of research, as they typically involve multi-physics and multi-scale systems, and efficiency and accuracy are challenging tasks. Many numerical strategies, based on either Finite Element (FE) and Finite Difference (FD) methods, have been applied, also in the case of complex materials and high dimensional problems, e.g. [5–7].

* Corresponding author.
E-mail address: caterina.millevoi@unipd.it (C. Millevoi).

https://doi.org/10.1016/j.jcp.2024.113299
Received 10 March 2022; Received in revised form 29 February 2024; Accepted 20 July 2024

Available online 25 July 2024

Though the above mentioned numerical methods remain the preferred choice as solvers for this system of PDEs, in recent years Physics-Informed Neural Networks (PINNs) gained an increasing attention [8–11]. The combination of Machine Learning (ML) with physical modeling has been identified as one of the most promising and challenging approaches in the last years [12,13], since it could result in a potentially significant breakthrough for efficiency, accuracy, and generalization capability of numerical methods for PDE solution. The peculiarity of PINNs is that they allow to incorporate information from the physics in addition to data by means of the use of the governing PDEs. Indeed, the so-called loss function that has to be minimized contains not only data and prediction mismatch (as in traditional ML), but also the residual of the governing equations. Once a PINN has been trained, it can provide an almost real-time solution [14–17], thus drastically improving tasks such as sensitivity analysis, model calibration, uncertainty quantification, parametric design, optimization, and so on. The nature of PINNs makes them suitable for different classes of problems, which can be classified into two main groups: i) feedforward solution of differential equations (direct problem), ii) parameter identification (inverse problem). In the first class, PINNs are used as a pure PDE solver, where the user only knows the governing equations alongside with the initial and boundary conditions. By distinction, in the second class some parameters of the PDE are assumed to be unknown, but it is assumed that some data inside the domain are available. A hybrid approach, where PDE solution is performed leveraging also on known data inside the domain, may also be possible. This strategy, similar to a data assimilation approach, makes PINNs more attractive from a practical point of view, and will deserve particular attention in this paper.

The application of PINNs, in their two classical approaches, has been already investigated in many fields, such as fluid mechanics [18–20], solid mechanics [21,22], heat diffusion [23], acoustic [24], additive manufacturing [25], health science [26–28]. Also in the field of porous materials, the interest in PINNs has grown rapidly in recent years, focusing in particular on aspects related to fluid diffusion and transport [29–36]. However, the PINN training turned out to be a slow and tricky activity, especially for multi-physics problems with coupled governing equations, due to the multi-objective optimization problem resulting from the multiple-term loss function [37]. For these reasons, the use of PINNs for the solution of the coupled hydro-poromechanical problem still presents several issues that deserve attention [38,39]. For example, to fix this problem, a stress-split sequential training is proposed in [40] and later applied taking the temperature into account, too [41].

This work focuses on the development of a PINN-based model for hydro-poromechanical applications, in particular governed by Biot's poroelasticity equations, discussing the principal aspects needed for an efficient and robust formulation. Initially, we investigate the role of PINN architecture and hyper-parameter selection in the construction of an effective model. Subsequently, to take advantage of PINN ability to integrate data and reduce the complexity of the solution of the coupled hydro-poromechanical problem, a so-called "sensor-driven" hybrid approach is proposed, where data are provided at locations inside the domain even to solve a forward problem. In other words, the idea is to exploit the possible availability of sensor-measured data in practical applications, so as to accelerate the PINN convergence to the problem solution and compensate for possible uncertainties in the problem formulation.

In order to check carefully the accuracy of the proposed approach, the paper focuses on well-known benchmark problems, which have been used to assess the numerical implementation and performance. This work aims to contribute to future applications of PINNs to challenging real-world problems, in particular those where monitored data recorded by sensors are available.

The paper is organized as follows. Section 2 contains a brief overview of the PINN methodology while Section 3 describes the general Biot's equations of poroelasticity and how PINN can be applied to such problem. Section 4 presents a discussion on the identification of an optimal hyper-parameter selection for PINN architecture, with the analysis focusing on two specific well-known benchmarks, namely the one-dimensional Terzaghi's consolidation and two-dimensional Mandel's problem. Then, in Section 5 the identified optimal layouts are used to simulate the solution of a problem where recorded data are available from potential sensors. PINN is first used as a PDE solver for the forward problem, and then the advantages arising from the integration of recorded measurements, even including possible noise effects and perturbation in the PDE parameters, are investigated. Finally, the PINN solution in a more realistic heterogeneous setting of a consolidation problem in the sensor-driven framework is described. In conclusion, Section 6 summarizes the outcome of the work and introduces the possible future developments.

2. Physics-Informed Neural Networks

PINNs are particular Neural Networks (NNs) that allow to incorporate information from the physics in addition to data by means of the governing PDEs [8]. The NN structure consists of many neurons (or nodes) distributed in different layers. Neurons in adjacent layers are connected each other and every connection is associated to a weight. Each neuron transfers to the following layer a linear combination of the signals received from the previous layer with some weights and biases, to which the activation function ϕ is applied.

Consider a NN with L layers. Let n_i be the number of neurons of the i-th layer and $\mathbf{x}^{(i)} \in \mathbb{R}^{n_{i-1}}$ be the input to layer i, for $i = 1, \dots, L$. We define the function $\boldsymbol{\Sigma}^{(i)} : \mathbb{R}^{n_{i-1}} \to \mathbb{R}^{n_i}$:

$$\boldsymbol{\Sigma}^{(i)}(\mathbf{x}^{(i)}) = \phi^{(i)}(\mathbf{W}^{(i)}\mathbf{x}^{(i)} + \mathbf{b}^{(i)}), \tag{1}$$

where $\mathbf{W}^{(i)} \in \mathbb{R}^{n_i \times n_{i-1}}$, $\mathbf{b}^{(i)} \in \mathbb{R}^{n_i}$, $\phi^{(i)}$ are weights, biases and the activation function of the i-th layer, respectively. For time dependent problems, the input of the first layer, $\mathbf{x}^{(1)} \in \mathbb{R}^{n_0}$, reads $\mathbf{x}^{(1)} = (\mathbf{x}, t)$ and $n_0 = n + 1$, with $\mathbf{x} \in \Omega \subset \mathbb{R}^n$ and $n = 1, 2, 3$ the spatial dimension. Then, the neural network $\hat{\mathbf{u}}(\mathbf{x}, t) : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L}$ is built as the composition of functions:

$$\hat{\mathbf{u}}(\mathbf{x}, t) = \boldsymbol{\Sigma}^{(L)} \circ \boldsymbol{\Sigma}^{(L-1)} \circ \cdots \circ \boldsymbol{\Sigma}^{(1)}(\mathbf{x}, t). \tag{2}$$

Assume that a general PDE holds true in $\Omega \times [0, +\infty)$:

$$\mathbf{u}_t + \mathcal{N}[\mathbf{u}] = 0, \quad \mathbf{x} \in \Omega \subset \mathbb{R}^n, \quad t \geq 0, \tag{3}$$

where $\mathcal{N}[\cdot]$ is a non linear differential operator in space and the subscript $\cdot_t$ indicates the derivative with respect to time. The solution $\mathbf{u}(\mathbf{x},t) : \Omega \times [0,+\infty) \to \mathbb{R}^m$, with m the output space size, is approximated by a neural network $\hat{\mathbf{u}}(\mathbf{x},t)$, whose training is performed by minimizing a cost function that includes not only the errors with respect to data, but also the residual of equation (3), thus constraining the model to both fit the data and comply with the expected physics. Furthermore, while data approximation is a point-wise learning mechanism, the residual entails also local knowledge of the solution, given by physical processes and dynamics involving partial derivatives [42].

Let us denote by $\mathcal{N}\mathcal{N}$ the function set of all the neural networks $\hat{\mathbf{u}}(\mathbf{x},t)$ defined in $\Omega \times [0,+\infty)$. Using the Mean Squared Error as loss measure, the training of the model aims at minimizing the functional $\mathcal{L} : \mathcal{N}\mathcal{N} \to [0,\infty)$:

$$\mathcal{L}(\hat{\mathbf{u}}) = w_d \frac{1}{N_d} \sum_{i=1}^{N_d} \|\hat{\mathbf{u}}(\mathbf{x}_d^i, t_d^i) - \mathbf{u}^i\|_2^2 + w_c \frac{1}{N_c} \sum_{i=1}^{N_c} \|\hat{\mathbf{u}}_t(\mathbf{x}_c^i, t_c^i) + \mathcal{N}[\hat{\mathbf{u}}(\mathbf{x}_c^i, t_c^i)]\|_2^2, \tag{4}$$

with $\{\mathbf{u}^i\}_{i=1}^{N_d}$ the set of N_d training data for $\hat{\mathbf{u}}(\mathbf{x},t)$, located at the data points $\{\mathbf{x}_d^i, t_d^i\}_{i=1}^{N_d}$; $\{\mathbf{x}_c^i, t_c^i\}_{i=1}^{N_c}$ the set of N_c collocation points for the residual computation; w_d and w_c proper weights to balance the two contributions in (4). With respect to standard neural networks, PINNs usually allow to obtain reasonable approximations with smaller datasets, as the governing physical laws are added as constraints. Furthermore, the residual contribution acts as a regularization term and increases the generalization ability of the neural network, thus allowing to deal with noisy data and outliers without a significant lack of robustness. In particular, PINN addresses two major challenges of deep learning application in Earth system science, as identified in [12]: (i) physical consistency, thanks to the residual term of the loss function providing a physical constraint, and (ii) limited labels, as the information given by the PDE balances the lack possibly coming from data, since Earth measurements are often sparse and noisy.

Different libraries have been recently developed to implement the PINN construction, e.g. [43–46]. The model considered in this work has been implemented using SciANN (Scientific Computational with Artificial Neural Networks), a recent TensorFlow and Keras wrapper specific to scientific computations with PINN [47]. This Application Programming Interface (API) makes it possible to build in a quite simple way the PINN structure with a readable code and to use automatic differentiation to compute at machine precision the derivatives of the neural network approximation of the solution $\mathbf{u}$. Originally designed to optimize the Gradient Descent algorithm in the NN training [48], automatic differentiation turned out to be extremely useful in PINN implementation. The possibility of using automatic differentiation has been inherited by SciANN from TensorFlow and Keras, resulting in faster and more robust codes, as it avoids the numerical discretization and the related rounding error propagation. This is particularly attractive when dealing with noisy data, usually resulting from real measurements. This library also supports running computations on a variety of devices, including CPUs and GPUs.

3. PINN for hydro-poromechanics

We focus on Biot's poroelasticity equations, governing the interaction between a granular material and the fluid filling its pores [2]. The set of PDEs consists of a stress equilibrium equation coupled with a fluid flow equation, which result from a conservation law of linear momentum and mass, respectively. Incorporating Terzaghi's effective stress principle, which links the grain forces to the fluid pore pressure, the equilibrium equation for an isotropic poroelastic medium and the continuity equation for the fluid mass balance can be written as:

$$\mu \Delta \mathbf{u} + (\lambda + \mu) \nabla \mathrm{div} \mathbf{u} = \alpha \nabla p + \mathbf{b}, \tag{5}$$

$$-\mathrm{div}(\boldsymbol{\kappa} \nabla p) + \frac{\partial}{\partial t}(\phi \beta p + \alpha \mathrm{div} \mathbf{u}) = f, \tag{6}$$

where $\mathbf{u}$ is the medium displacement, p is the fluid pore pressure, $\mathbf{b}$ is the body force, and f is a flow source or sink. Equation (6) has been obtained by coupling the continuity of pore fluid with Darcy's law:

$$\boldsymbol{\kappa}^{-1} \mathbf{v} + \nabla p = 0, \tag{7}$$

where $\mathbf{v}$ is Darcy's velocity. The material parameters are Lamé's moduli λ and μ, the Biot coefficient α, Darcy's conductivity tensor $\boldsymbol{\kappa}$, the medium porosity ϕ, and the fluid compressibility β. The Darcy's conductivity tensor is given by $\boldsymbol{\kappa} = \boldsymbol{k}/\gamma_w$ with $\boldsymbol{k}$ the hydraulic conductivity tensor and γ_w the fluid specific weight. As usual, ∇ and Δ denote the gradient and the Laplacian operator, respectively.

Let $\Omega \subset \mathbb{R}^n$ be the domain of the coupled partial differential system in (5)-(6) and Γ its boundary. The problem of finding the unknowns $\mathbf{u}$ and p is well-posed if proper boundary:

$$\begin{cases} \mathbf{u}(\mathbf{x},t) = \mathbf{u}_D(\mathbf{x},t), & \forall \mathbf{x} \in \Gamma_u, t > 0, \\ \sigma(\mathbf{x},t)\mathbf{n}(\mathbf{x}) = \mathbf{t}_N(\mathbf{x},t), & \forall \mathbf{x} \in \Gamma_\sigma, t > 0, \\ p(\mathbf{x},t) = p_D(\mathbf{x},t), & \forall \mathbf{x} \in \Gamma_p, t > 0, \\ \mathbf{v}(\mathbf{x},t) \cdot \mathbf{n}(\mathbf{x}) = q_N(\mathbf{x},t), & \forall \mathbf{x} \in \Gamma_q, t > 0, \end{cases} \tag{8}$$

and initial conditions:

$$\begin{cases} \mathbf{u}(\mathbf{x},0)=\mathbf{u}_0(\mathbf{x}), & \forall \mathbf{x}\in\Omega\cup\Gamma, \\ p(\mathbf{x},0)=p_0(\mathbf{x}), & \forall \mathbf{x}\in\Omega\cup\Gamma, \end{cases} \tag{9}$$

apply. In equations (8) and (9), $\Gamma_u\cup\Gamma_\sigma=\Gamma_p\cup\Gamma_q=\Gamma$, with $\Gamma_u\cap\Gamma_\sigma=\Gamma_p\cap\Gamma_q=\emptyset$, $\boldsymbol{\sigma}$ is the total stress tensor, and $\mathbf{n}$ is the outer normal to Γ, while the right-hand side functions $\mathbf{u}_D$, $\mathbf{t}_N$, p_D, q_N, $\mathbf{u}_0$, and p_0 are known.

In our problem, a neural network for each unknown is set up, i.e., one for the pressure, $\hat{p}(\mathbf{x},t)$, and one for every component of the displacement, $\hat{u}_j(\mathbf{x},t)$ with $j=1,n$. This is motivated by results obtained in [21], which suggest using many different NNs with single output instead of a unique one with multiple outputs, even if entailing a higher number of parameters. The use of distinct networks is more suitable, as the outputs of the problem have different physical natures. Denoting by $\hat{\mathbf{u}}$ the vector with components $\hat{u}_j$, the loss function reads:

$$\mathcal{L}(\hat{\mathbf{u}},\hat{p})=\mathcal{L}_d+\sum_{j=1}^{n}\mathcal{L}_{equ,j}+\mathcal{L}_{cont}+\sum_{j=1}^{n}\mathcal{L}_{\Gamma_u,j}+\sum_{j=1}^{n}\mathcal{L}_{\Gamma_\sigma,j}+\mathcal{L}_{\Gamma_p}+\mathcal{L}_{\Gamma_q}+\mathcal{L}_{IC}. \tag{10}$$

The terms in the loss function are defined as follows. The loss on training data $\mathcal{L}_d$ is:

$$\mathcal{L}_d=\sum_{j=1}^{n} w_{d,u,j}\frac{1}{N_d}\sum_{i=1}^{N_d}\|\hat{u}_j(\mathbf{x}_d^i,t_d^i)-u_j^i\|_2^2+w_{d,p}\frac{1}{N_d}\sum_{i=1}^{N_d}\|\hat{p}(\mathbf{x}_d^i,t_d^i)-p^i\|_2^2, \tag{11}$$

where $\{\mathbf{x}_d^i,t_d^i\}_{i=1}^{N_d}$ are the points where data are imposed. The weights $w_{d,\cdot}$ are selected to ensure the dimensional consistency of equation (11) and so as to normalize the set of data $\{p^i,\mathbf{u}^i\}_{i=1}^{N_d}$ between 0 and 1. Also the terms associated to the residual and boundary conditions must be multiplied by suitable weights $w_\cdot$, so that each contribution is comparable to others in the global loss function (10). This prevents from a term to largely prevail over the others in the minimization process and generally is a good practice for neural networks training, too. In practice, scaled neural networks $\hat{p}_s$ and $\hat{\mathbf{u}}_s$ are built to fit the scaled data, while the neural network approximations $\hat{p}$ and $\hat{\mathbf{u}}$ are obtained by the inverse denormalization. The contributions $\mathcal{L}_{equ,j}$ and $\mathcal{L}_{cont}$:

$$\begin{aligned} \mathcal{L}_{equ,j}&=w_{equ,j}\frac{1}{N_c}\sum_{i=1}^{N_c}\|\mu(\Delta\hat{\mathbf{u}})_j(\mathbf{x}_c^i,t_c^i)+(\lambda+\mu)(\nabla\mathrm{div}\hat{\mathbf{u}})_j(\mathbf{x}_c^i,t_c^i)-\alpha(\nabla\hat{p})_j(\mathbf{x}_c^i,t_c^i)+\mathbf{b}_j(\mathbf{x}_c^i,t_c^i)\|_2^2, \\ \mathcal{L}_{cont}&=w_{cont}\frac{1}{N_c}\sum_{i=1}^{N_c}\|-\mathrm{div}(\boldsymbol{\kappa}\nabla\hat{p})(\mathbf{x}_c^i,t_c^i)+\frac{\partial}{\partial t}(\phi\beta\hat{p}(\mathbf{x}_c^i,t_c^i)+\alpha\mathrm{div}\hat{\mathbf{u}}(\mathbf{x}_c^i,t_c^i))-f(\mathbf{x}_c^i,t_c^i)\|_2^2, \end{aligned} \tag{12}$$

are the loss terms arising from the residual of the governing physical equations on the set of collocation points $\{\mathbf{x}_c^i,t_c^i\}_{i=1}^{N_c}$, whereas the remaining terms in (10) are needed to impose the boundary conditions (8):

$$\begin{aligned} \mathcal{L}_{\Gamma_u,j}&=w_{\Gamma_u,j}\frac{1}{N_{\Gamma_u}}\sum_{i=1}^{N_{\Gamma_u}}\|\hat{u}_j(\mathbf{x}_{\Gamma_u}^i,t_{\Gamma_u}^i)-u_{D,j}(\mathbf{x}_{\Gamma_u}^i,t_{\Gamma_u}^i)\|_2^2, \\ \mathcal{L}_{\Gamma_\sigma,j}&=w_{\Gamma_\sigma,j}\frac{1}{N_{\Gamma_\sigma}}\sum_{i=1}^{N_{\Gamma_\sigma}}\|(\hat{\boldsymbol{\sigma}}(\mathbf{x}_{\Gamma_\sigma}^i,t_{\Gamma_\sigma}^i)\mathbf{n}(\mathbf{x}_{\Gamma_\sigma}^i))_j-\mathbf{t}_{N,j}(\mathbf{x}_{\Gamma_\sigma}^i,t_{\Gamma_\sigma}^i)\|_2^2, \\ \mathcal{L}_{\Gamma_p}&=w_{\Gamma_p}\frac{1}{N_{\Gamma_p}}\sum_{i=1}^{N_{\Gamma_p}}\|\hat{p}(\mathbf{x}_{\Gamma_p}^i,t_{\Gamma_p}^i)-p_D(\mathbf{x}_{\Gamma_p}^i,t_{\Gamma_p}^i)\|_2^2, \\ \mathcal{L}_{\Gamma_q}&=w_{\Gamma_q}\frac{1}{N_{\Gamma_q}}\sum_{i=1}^{N_{\Gamma_q}}\|\boldsymbol{\kappa}\nabla\hat{p}(\mathbf{x}_{\Gamma_q}^i,t_{\Gamma_q}^i)\cdot\mathbf{n}(\mathbf{x}_{\Gamma_q}^i)-q_N(\mathbf{x}_{\Gamma_q}^i,t_{\Gamma_q}^i)\|_2^2, \end{aligned} \tag{13}$$

and the initial conditions (9):

$$\mathcal{L}_{IC}=\sum_{j=1}^{n}w_{IC,u,j}\frac{1}{N_{IC}}\sum_{i=1}^{N_{IC}}\|\hat{u}_j(\mathbf{x}^i,0)-u_{0,j}(\mathbf{x}^i)\|_2^2+w_{IC,p}\frac{1}{N_{IC}}\sum_{i=1}^{N_{IC}}\|\hat{p}(\mathbf{x}^i,0)-p_0(\mathbf{x}^i)\|_2^2. \tag{14}$$

Here, $\hat{\boldsymbol{\sigma}}=\boldsymbol{\sigma}(\hat{\mathbf{u}})$ indicates the approximation of the total stress, that is defined by the material constitutive law, $\{\mathbf{x}_{\Gamma_u}^i,t_{\Gamma_u}^i\}_{i=1}^{N_{\Gamma_u}}$, $\{\mathbf{x}_{\Gamma_\sigma}^i,t_{\Gamma_\sigma}^i\}_{i=1}^{N_{\Gamma_\sigma}}$, $\{\mathbf{x}_{\Gamma_p}^i,t_{\Gamma_p}^i\}_{i=1}^{N_{\Gamma_p}}$ and $\{\mathbf{x}_{\Gamma_q}^i,t_{\Gamma_q}^i\}_{i=1}^{N_{\Gamma_q}}$ are points on the boundary, and $\{\mathbf{x}^i,0\}_{i=1}^{N_{IC}}$ are points in the domain Ω at $t=0$. The presented PINN formulation is used in the next sections to solve two classical benchmarks and a heterogeneous test case in coupled hydro-poromechanics.

Table 1
Hyper-parameter set of values.

Hyper-parameter	Description	Set
Layers	Number of hidden layers	{4,8,12}
Neurons	Number of neurons per layer	{20,40}
Act fun	Activation function of each neuron in the hidden layers	{tanh, elu}

4. Identification of PINN architecture

In this section, we investigate the optimal construction of PINNs for two classical coupled problems, namely Terzaghi's problem (1D) and Mandel's problem (2D), for which analytical solutions are available. In particular, we analyze the optimal choice of the hyper-parameters and evaluate the accuracy of the PINN model used as a forward solver for the governing differential problem.

It is well-known that an established procedure to choose the architecture of a neural network does not exist, but it mainly remains at the modeler's experience. A general rule of thumb is the deeper the network, the higher the ability to capture complex links, but of course the slower the training time. Many factors can come into play, influencing in different ways the difficulty of the procedure. Some of them are specifically problem-dependent, e.g., the number of data and collocation points, or the hyper-parameters regulating the training, such as batch size and number of epochs. However, other factors basically depend on the class of problem and the structure of the governing equations. Driven by these motivations, we extensively analyze the founding elements of the architecture of PINNs for coupled hydro-poromechanics, with the aim at investigating its robustness and accuracy with respect to the hyper-parameter set selection.

Among the hyper-parameters needed to set up a PINN model, the most influential ones are:

- number of layers,
- number of neurons per layer,
- type of activation function.

For each item, we consider the set of possible entries reported in Table 1 with the effect of all possible combinations.

In this work, we assume for the sake of simplicity that all hidden layers have the same number of neurons and the same activation function. For PINN models, it is preferable to use differentiable activation functions, because the neural networks have to approximate a smooth PDE solution and are automatically differentiated to compute the loss terms (12). For this reason, along with tanh, we use the ELU activation function, defined as:

$$ELU(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha(e^x - 1) & \text{if } x < 0 \end{cases}, \tag{15}$$

with $\alpha = 1$, instead of the more common Rectified Linear Unit (ReLU) function. In the following sections, we present the results obtained for a one-dimensional and a two-dimensional formulation of the problem, comparing the accuracy of the approximations for the different combinations of hyper-parameters.

A full factorial combination of the different values for the number of layers, neurons and the activation function type has been considered, with the networks obtained by each architecture properly trained. The quality of every realization is evaluated by computing the weighted L2-norm of the difference between the NN prediction, $\hat{\mathbf{y}}$, and the analytical solution, $\mathbf{y}$:

$$E(\hat{\mathbf{y}}, \mathbf{y}) = \frac{\|\hat{\mathbf{y}} - \mathbf{y}\|_2}{\|\mathbf{y}\|_2} \tag{16}$$

The use of such weighted norm allows to deal with dimensionless and comparable errors. Then, the population of $E(\hat{\mathbf{y}}, \mathbf{y})$ data is statistically processed in order to identify the impact of the hyper-parameters, their mutual influence, the robustness of the proposed PINN architecture with respect to their variation, and the setup providing the most accurate outcome.

4.1. 1D: Terzaghi's problem

Terzaghi's problem consists of a poroelastic fluid-saturated column with a constant loading P_L applied instantaneously on top at time $t = 0$, as shown schematically in Fig. 1. The column has height L and at the basement there are zero flux and null displacement. Only through the upper boundary free drainage is allowed. In a one-dimensional configuration and assuming the z-axis positive downward (Fig. 1), equations (5) and (6) read:

$$(\lambda + 2\mu)\frac{\partial^2 u}{\partial z^2} = \alpha \frac{\partial p}{\partial z}, \tag{17}$$

$$-\kappa \frac{\partial^2 p}{\partial z^2} + \frac{\partial}{\partial t}\left(\phi\beta p + \alpha \frac{\partial u}{\partial z}\right) = 0, \tag{18}$$

with the following boundary conditions:

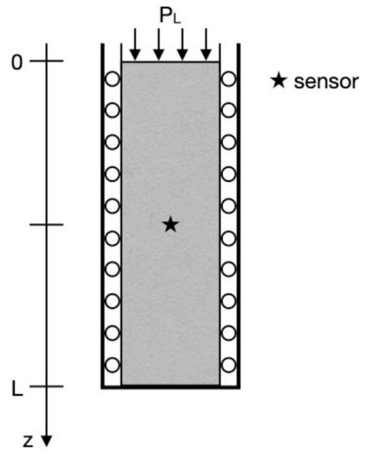


Fig. 1. Sketch of the setup for Terzaghi's problem with indication of sensor location.

Table 2
Material parameters definition.

Parameter	Name	Unit	Definition
M	Biot modulus	MPa	$M = [\phi\beta + (\alpha - \phi)c_{br}]^{-1}$
K_u	Undrained bulk modulus	MPa	$K_u = \lambda + 2\mu/3 + \alpha^2 M$
c_M	Vertical uniaxial compressibility	MPa^{-1}	$c_M = [\lambda + 2\mu]^{-1}$
c	Consolidation coefficient	m^2/s	$c = k/[\gamma_w(M^{-1} + \alpha^2 c_M)]$
B	Skempton's coefficient	–	$B = \alpha M/K_u$
ν_u	Undrained Poisson's ratio	–	$\nu_u = [3\nu + \alpha B(1-2\nu)]/[3 - \alpha B(1-2\nu)]$

$$\begin{aligned} &p(0,t) = 0, && (\lambda + 2\mu)\frac{\partial u}{\partial z}(0,t) = -P_L, && z = 0, \\ &\frac{\partial p}{\partial z}(L,t) = 0, && u(L,t) = 0, && z = L. \end{aligned} \tag{19}$$

The initial overpressure $p_0(z)$ and displacement $u_0(z)$ caused by the instant load P_L read [4–7]:

$$\begin{aligned} p_0(z) &= \begin{cases} 0 & z = 0 \\ \dfrac{\alpha M}{K_u + 4\mu/3} P_L & \text{otherwise} \end{cases}, \\ u_0(z) &= \frac{1}{K_u + 4\mu/3} P_L (L - z), \end{aligned} \tag{20}$$

and the analytical solutions are:

$$\begin{aligned} p(z,t) &= \frac{4}{\pi} p_0 \sum_{m=0}^{\infty} \frac{1}{2m+1} \exp\left[-\left(\frac{(2m+1)\pi}{2L}\right)^2 ct\right] \sin\left[\frac{(2m+1)\pi z}{2L}\right], \\ u(z,t) &= c_M p_0 \left\{ (L - z) - \frac{8L}{\pi^2} \sum_{m=0}^{\infty} \frac{1}{(2m+1)^2} \exp\left[-\left(\frac{(2m+1)\pi}{2L}\right)^2 ct\right] \cos\left[\frac{(2m+1)\pi z}{2L}\right] \right\} + u_0. \end{aligned} \tag{21}$$

The definition of the material parameters arising in (20) and (21) is summarized in Table 2, where c_{br} denotes the solid grain compressibility, ν the drained Poisson ratio, λ and μ the Lamé constants. All material parameter values are reported in Table 3. The poroelastic medium consists of a homogeneous sandy material (λ_s, μ_s and k_s) with $L = 15$ m and $P_L = 10^{-2}$ MPa.

In the 1D formulation (17)-(18), the unknown functions are the fluid pore pressure p and the vertical displacement u. As stated in Section 2, they are approximated by two distinct neural networks, with the loss function built following (10). The terms of the loss function give their contribution on different portions of the space-time domain $[0, L] \times [0, +\infty)$. For this reason, training points have to be properly distributed to cover the domain completely. They are generated with DataGeneratorXT of SciANN, which builds a grid with a number of points distributed in both the domain and the boundaries, so as to guarantee that each loss term is proportionally sampled and the optimizer performs better.

The numerical time domain is bounded at $T = 1000$ s, which represents a value where steady state conditions are practically achieved for the selected material parameters. First of all, we notice that a uniform distribution of training points in time is not effective, since significant oscillations of the initial boundary bands can be observed (Fig. 2). Such sharp oscillations are substantially smoothed generating a logarithmic random sample in time. This is motivated by the shape of the analytical solution (21) that

Table 3
Material properties.

Parameter	Name	Value
λ_s	Lamé constant (sand)	40 MPa
μ_s	Lamé constant (sand)	40 MPa
λ_c	Lamé constant (clay)	4 MPa
μ_c	Lamé constant (clay)	4 MPa
α	Biot coefficient	1.0
ϕ	Medium porosity	0.375
β	Fluid compressibility	$4.4 \cdot 10^{-4}$ MPa^{-1}
k_s	Hydraulic conductivity (sand)	10^{-5} m/s
k_c	Hydraulic conductivity (clay)	10^{-7} m/s
γ_w	Fluid specific weight	$9.81 \cdot 10^{-3}$ MN/m^3
ν	Drained Poisson ratio	0.3
c_{br}	Solid grain compressibility	0 MPa^{-1}

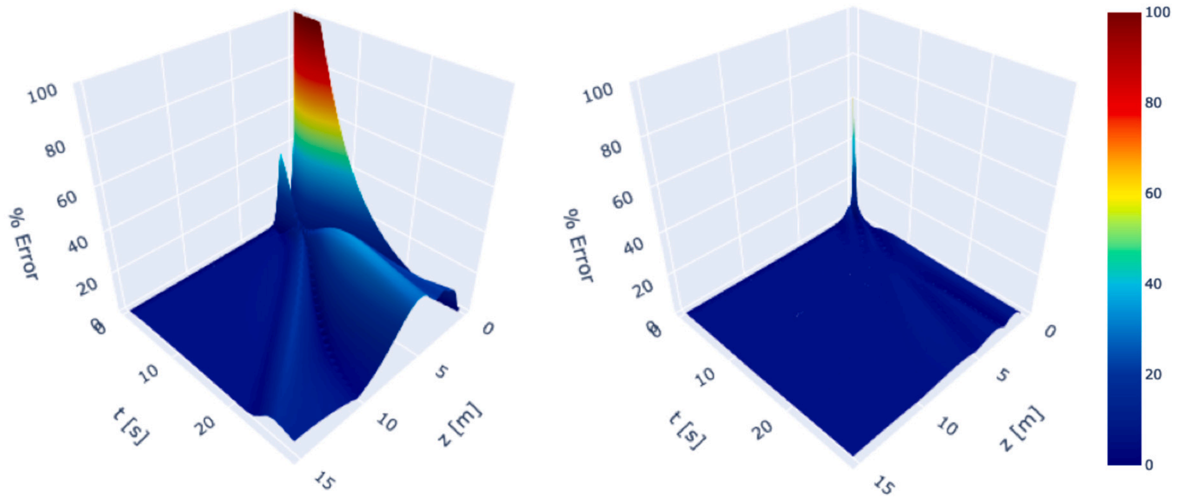


Fig. 2. Terzaghi's problem: relative L2-norm pressure error (equation (16)) with a uniform (left) and logarithmic (right) training point distribution in time.

is exponentially decaying in time. This behavior is common for every coupled hydro-poromechanical problem, independently of application dimension and the specific boundary conditions.

The total number N_d of training data is set to 3000, half of which is equally distributed between boundary and initial conditions. This is an empirical choice balancing the need of limiting the computational burden of the training processes and the size of the approximation errors. Therefore, we suppose to have pressure and displacement data $\{p^i, u^i\}_{i=1}^{N_d}$ obtained by evaluation of the analytical solutions (21) over training points $\{z_d^i, t_d^i\}_{i=1}^{N_d}$ spread all over the domain. The points located inside the domain are used as collocation points, while the ones on the space-time boundaries allow to evaluate (19) and (20), thus resulting in $N_c = 1500$, $N_{IC} = 750$, and $N_{BC} = 750$ split into top and bottom (Fig. 3).

We consider all the possible 144 architectures generated by varying the number of layers in $\{4, 8, 12\}$, the number of neurons in $\{20, 40\}$ and the type of the activation function, either tanh or ELU, for both networks (Table 1). The number of epochs is set to 5000 with a batch size of 500. Note that the size of the batches has to be quite big so as to contain points over the boundaries too. This is necessary because the gradient updates have to consider also information for the boundary terms of the loss function (10). An early stopping is added to quit the training if the loss function does not improve for 500 epochs. Adam algorithm [49] is used with a learning rate constantly equal to 0.001 until the 2000th epoch and then exponentially decreasing to 0.0001 until the last epoch [50]. The weights of both the neural networks are initialized with Glorot normal initializer of Keras.

As mentioned before, to improve the training procedure all data have been normalized between 0 and 1 by Scikit-learn Min-Max scaler. First, the scaled networks $\hat{p}_s$ and $\hat{u}_s$ have been trained over these scaled data, and then denormalized to obtain $\hat{p}$ and $\hat{u}$:

$$\begin{aligned} \hat{p} &= \hat{p}_s p_{sc} + p_{min}, \\ \hat{u} &= \hat{u}_s u_{sc} + u_{min}, \end{aligned} \tag{22}$$

where $p_{sc} = p_{max} - p_{min}$, $u_{sc} = u_{max} - u_{min}$, with p_{min}, p_{max} and u_{min}, u_{max} the minimum and maximum pressure and displacement values, respectively. Also, each term of the loss function is multiplied by an appropriate weighting factor to balance their role in the training process. In this case, the weights have been set as follows: $w_{equ,z} = w_{\Gamma_p} = p_{sc}$, $w_{cont} = w_{\Gamma_u,z} = u_{sc}$, $w_{\Gamma_\sigma,z} = P_L$, $w_{\Gamma_q} = 100p_{sc}$, $w_{IC,u,z} = \frac{1}{K_u + 4\mu/3} P_L L$, and $w_{IC,p} = \frac{\alpha M}{K_u + 4\mu/3} P_L$. The choice is also motivated by the different unit of measure of the loss function

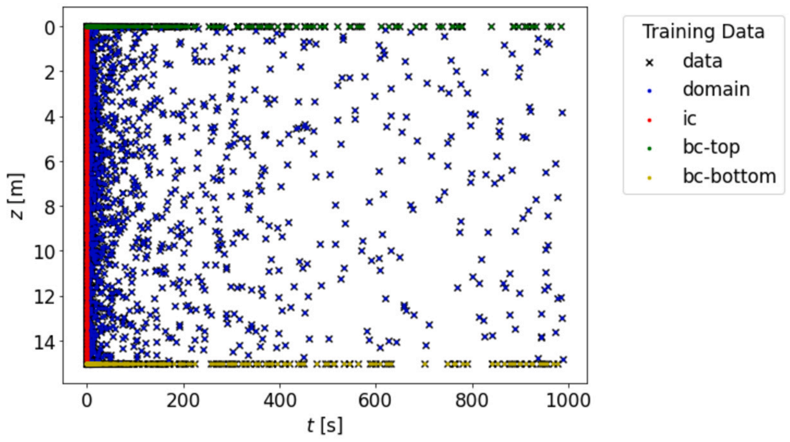


Fig. 3. Terzaghi's problem: training point distribution in the space-time domain. Blue ones are collocation points, while green, red and black ones are used to impose initial and boundary conditions at $z = 0$ and $z = L$, respectively. Pressure and displacement data are given on them all (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.).

Table 4
Response optimization results.

	Layers p	Layers u	Neurons p	Neurons u	Act fun p	Act fun u
Terzaghi	10	12	40	20	elu	elu
Mandel	4	8	20	20	tanh	tanh

contributions, see eqs. (17) to (20). Notice that w_{Γ_q} is magnified 100 times with respect to other similar weights to better enforce boundary conditions at $z = L$. The seed generating the random points is the same in all simulations in order to have reproducible and comparable results.

To evaluate the accuracy, the trained networks are computed over a uniform grid with 1501×1001 points in the $z - t$ domain $[0, 15]$ m $\times [0, 1000]$ s. The errors are evaluated by numerically computing the dimensionless weighted L^2-norm (16) over such a uniform grid and plotted in Fig. 4 for all 144 architectures. Mean value and standard deviation of the errors are equal to $\mu_p = 3.291 \times 10^{-2}$ and $\sigma_p = 1.719 \times 10^{-2}$ for p, $\mu_u = 5.622 \times 10^{-2}$ and $\sigma_u = 3.327 \times 10^{-2}$ for u. Overall, we can observe that the PINN accuracy appears to be quite satisfactory for almost any of the investigated architectures. The displacement errors are larger than the pressure ones on average, with a similar statistical distribution. It is interesting to note that for both p and u the smallest errors occur in architecture indices between 100 and 120, which correspond to the use of ELU activation function, 12 layers and 20 neurons for u. These plots highlight the robustness of the PINN method, that for a large number of combinations gives similar and satisfactory error values. Outliers are quite few and even in the worse cases the errors are at most around 12%. The full factorial combination of the values in Table 1 has been done with three repetitions for each architecture, obtained using three different random seeds. Graphs in Fig. 4 preserve the same allocation of the errors for all the different seeds, so that it is possible to conclude that the choice of the seed has a negligible impact.

Fig. 5 contains the main effect plots of the mean errors obtained with the different hyper-parameters. They show that the strongest impact on the PINN accuracy is given by the choice of the activation function for $\hat{u}$ and the number of its neurons, with ELU and 20 neurons being more effective. The choices for the $\hat{p}$ architecture are not so significant for the accuracy in the displacement approximation. Conversely, for pressure approximation ELU is again more appropriate. It is also possible to conclude that a suitable value for the number of layers of both the NNs is between 8 and 12.

Finally, a classical response optimization process [51] applied to both p- and u-error provides the architecture in Table 4 as the best one. The response optimization is ideal for optimizing and exploring deployed predictive data mining models. It performs a discrete search in the independent variable space, so as to enrich the plumbed space of the hyper-parameters, until a set of independent values are discovered for which the model yield minimizes the responses. Note that the optimal values for $\hat{u}$ are ELU activation function, 12 layers and 20 neurons, as confirmed also from Fig. 4 and 5.

4.2. 2D: Mandel's problem

Mandel's problem simulates the consolidation of an infinitely long poroelastic slab, which is sandwiched between two rigid, impermeable and frictionless plates. The poroelastic fluid-saturated slab is homogeneous and isotropic and both plates are suddenly squeezed at time $t = 0$ by a constant compressive vertical load per unit length $2F$ (Fig. 6). Let $2a$ and $2b$ be the slab sizes. The left and right boundaries ($x = \pm a$) are stress-free, drained and always at ambient pressure ($p = 0$), while top and bottom boundaries ($z = \pm b$) have a prescribed stress and no flux. As the domain is symmetric, only the highlighted quarter of the $x - z$ plane in Fig. 6 is considered.

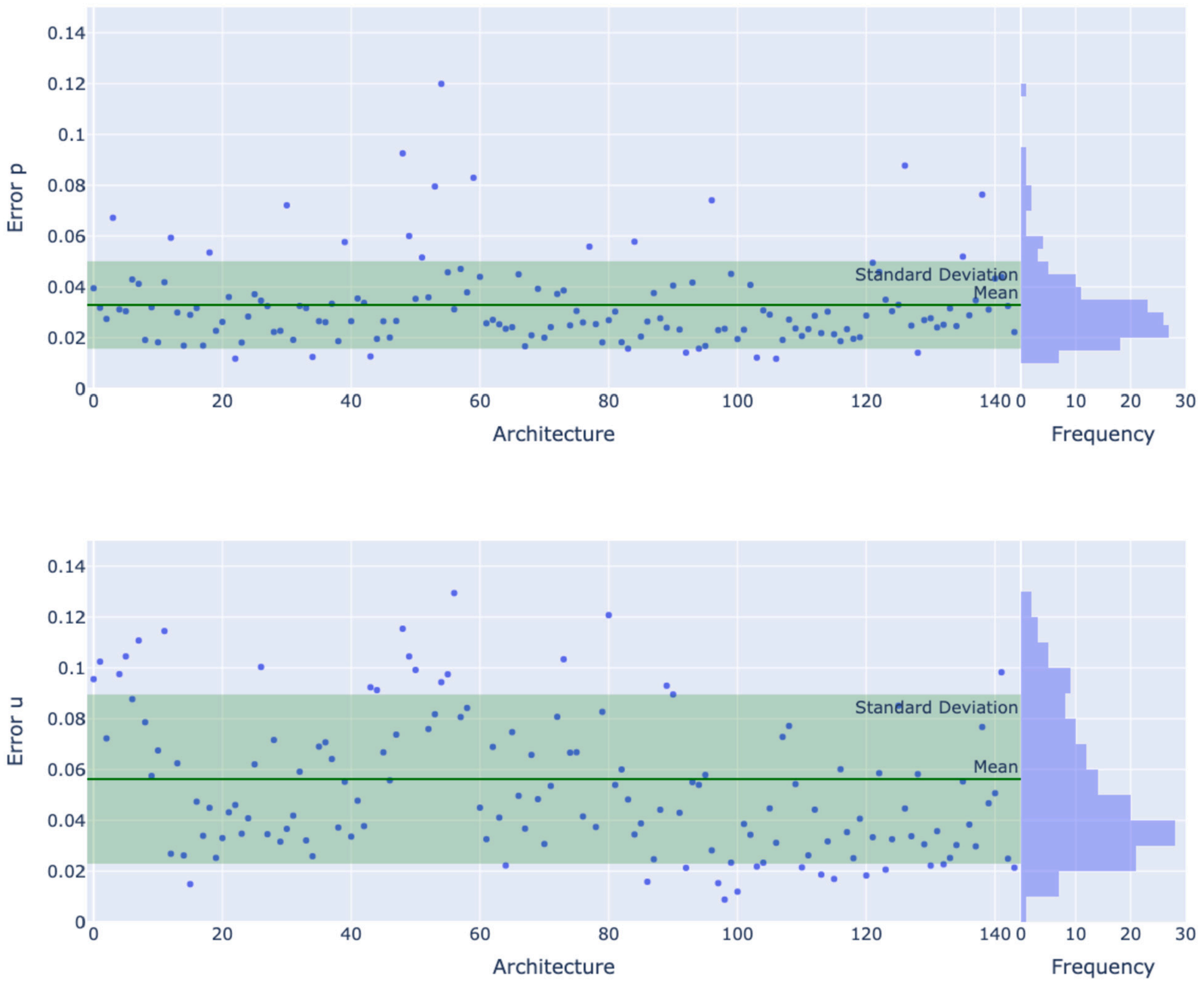


Fig. 4. Terzaghi's problem: errors in the pressure (top) and displacement (bottom) reconstruction for the different architectures. On the rightmost frames, the error statistical distribution is provided.

Equations (5) and (6) in two dimensions for Mandel's configuration take the form:

$$
\begin{aligned}
&(\lambda+2\mu)\frac{\partial^2 u_x}{\partial x^2}+\mu\frac{\partial^2 u_x}{\partial z^2}+(\lambda+\mu)\frac{\partial^2 u_z}{\partial x\partial z}=\alpha\frac{\partial p}{\partial x},\\
&\mu\frac{\partial^2 u_z}{\partial x^2}+(\lambda+2\mu)\frac{\partial^2 u_z}{\partial z^2}+(\lambda+\mu)\frac{\partial^2 u_x}{\partial x\partial z}=\alpha\frac{\partial p}{\partial z},
\end{aligned}
\tag{23}
$$

$$
-\kappa\left(\frac{\partial^2 p}{\partial x^2}+\frac{\partial^2 p}{\partial z^2}\right)+\frac{\partial}{\partial t}\left(\phi\beta p+\alpha\left(\frac{\partial u_x}{\partial x}+\frac{\partial u_z}{\partial z}\right)\right)=0,
\tag{24}
$$

where the Darcy conductivity $\boldsymbol{\kappa}$ is assumed to be the identity tensor $\mathbf{1}$ multiplied by the scalar $\kappa = k/\gamma_w$. The boundary conditions read:

$$
\begin{array}{llll}
\dfrac{\partial p}{\partial x}(0,z,t)=0, & u_x(0,z,t)=0, & \dfrac{\partial u_z}{\partial x}(0,z,t)=0, & x=0,\\
p(a,z,t)=0, & \dfrac{\partial u_x}{\partial x}(a,z,t)=\dfrac{F\nu}{2\mu a}, & \dfrac{\partial u_z}{\partial x}(a,z,t)=0, & x=a,\\
\dfrac{\partial p}{\partial z}(x,0,t)=0, & \dfrac{\partial u_x}{\partial z}(x,0,t)=0, & u_z(x,0,t)=0, & z=0,\\
\dfrac{\partial p}{\partial z}(x,b,t)=0, & \dfrac{\partial u_x}{\partial z}(x,b,t)=0, & \dfrac{\partial u_z}{\partial z}(x,b,t)=-\dfrac{F(1-\nu)}{2\mu a}, & z=b.
\end{array}
\tag{25}
$$

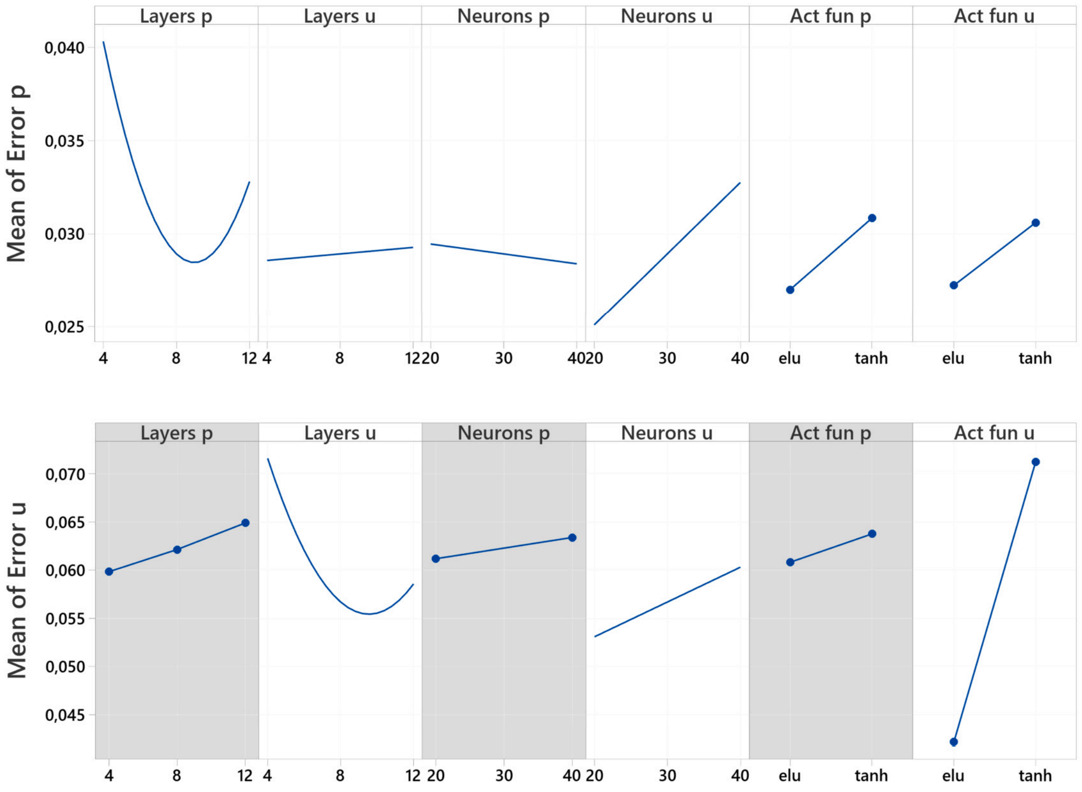


Fig. 5. Terzaghi's problem: main effect plots of the sensitivity analysis on all hyper-parameters for the pressure (top) and displacement (bottom) approximations. A gray background denotes a term not statistically significant in the model.

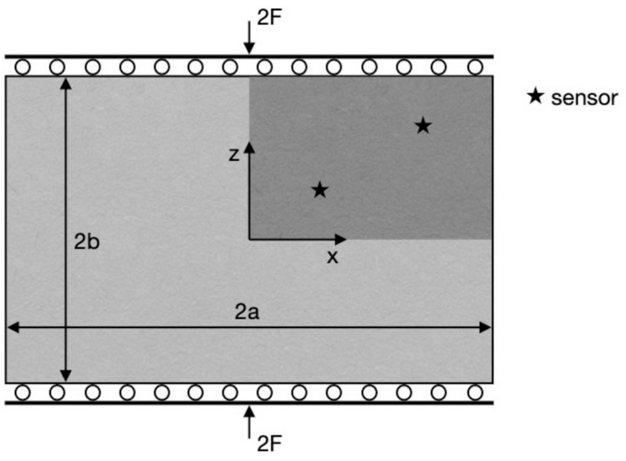


Fig. 6. Sketch of the setup for Mandel's problem with indication of sensor locations, only in the portion of the domain considered for symmetry reasons.

The instantaneous load application at $t = 0$ causes the initial overpressure $p_0(x,z)$ and horizontal and vertical displacements $u_{x,0}(x,z)$ and $u_{z,0}(x,z)$ [6,7]:

$$p_0(x,z) = \begin{cases} 0 & x = a \\ \dfrac{1}{3a} B(1+\nu_u)F & \text{otherwise} \end{cases},$$

$$u_{x,0}(x,z) = \frac{F\nu_u}{2\mu}\frac{x}{a}, \tag{26}$$

$$u_{z,0}(x,z) = -\frac{F(1-\nu_u)}{2\mu}\frac{z}{a}.$$

The analytical solution reads [7]:

$$\begin{aligned}
p(x,z,t) &= 2p_0 \sum_{n=1}^{\infty} \frac{\sin\alpha_n}{\alpha_n - \sin\alpha_n\cos\alpha_n}\left(\cos\frac{\alpha_n x}{a} - \cos\alpha_n\right)\exp\left(-\frac{\alpha_n^2 ct}{a^2}\right),\\
u_x(x,z,t) &= \left[\frac{F\nu}{2\mu a} - \frac{F\nu_u}{\mu a}\sum_{n=1}^{\infty}\frac{\sin\alpha_n\cos\alpha_n}{\alpha_n - \sin\alpha_n\cos\alpha_n}\exp\left(-\frac{\alpha_n^2 ct}{a^2}\right)\right]x\\
&\quad + \frac{F}{\mu}\sum_{n=1}^{\infty}\frac{\cos\alpha_n}{\alpha_n - \sin\alpha_n\cos\alpha_n}\sin\frac{\alpha_n x}{a}\exp\left(-\frac{\alpha_n^2 ct}{a^2}\right),\\
u_z(x,z,t) &= \left[-\frac{F(1-\nu)}{2\mu a} + \frac{F(1-\nu_u)}{\mu a}\sum_{n=1}^{\infty}\frac{\sin\alpha_n\cos\alpha_n}{\alpha_n - \sin\alpha_n\cos\alpha_n}\exp\left(-\frac{\alpha_n^2 ct}{a^2}\right)\right]z,
\end{aligned} \tag{27}$$

with α_n the positive roots of the non-linear equation:

$$\tan\alpha_n = -\frac{\nu - 1}{\nu_u - \nu}\alpha_n. \tag{28}$$

See Table 2 and 3 for the material parameter definitions and values, with $\lambda = \lambda_s$, $\mu = \mu_s$ and $k = k_s$. We choose $a = b = 1$ m and the force F equal to 10^{-2} MN/m.

In Mandel's problem there are three unknown functions, i.e., fluid pressure $p(x,z,t)$, horizontal displacement $u_x(x,z,t)$, and vertical displacement $u_z(x,z,t)$. Each one is approximated by a neural network with the loss function built as in (10), so as to honor both the governing equations (23)-(24) and the auxiliary conditions in (25)-(26). The problem domain in space and time is $\Omega = [0,a]\times[0,b]\times[0,+\infty)$. Given the material parameter values of Table 3, the domain is limited along the time direction to $T = 10$ s. Recalling the observations made for Terzaghi's problem, we empirically select $N_d = 6000$ data points using SciANN DataGeneratorXYT to build the random samples $\{x_d^i, z_d^i, t_d^i\}_{i=1}^{N_d}$ with a logarithmic scale in time. The procedure is forced to produce half training points inside the domain and the remaining ones equally distributed along the boundaries. In particular, 375 points are set on each side of the space domain boundary and 1500 at $t = 0$.

The same hyper-parameter domain as Terzaghi's problem (Table 1) is considered for the sensitivity analysis. We use 5000 epochs with a batch size set equal to 1000 and the learning rate of Adam optimization algorithm to 0.001 for the first 2000 epochs, then exponentially decreasing to 0.0001 until the last epoch. As for Terzaghi's problem, the weights of the networks are initialized with Glorot normal initialization and the training data are scaled with the min-max scaler. Then, three NNs, namely $\hat{p}_s$, $\hat{u}_{x,s}$, and $\hat{u}_{z,s}$, are built to fit the scaled data and other three networks, $\hat{p}$, $\hat{u}_x$, and $\hat{u}_z$, are derived as in (22). The values for the remaining weights are: $w_{equ,x} = w_{equ,z} = w_{\Gamma_p} = w_{\Gamma_q} = p_{sc}$, $w_{cont} = w_{\Gamma_u,x} = u_{x,sc}$, $w_{\Gamma_u,z} = u_{z,sc}$, $w_{\Gamma_\sigma,x} = w_{\Gamma_\sigma,z} = \frac{F}{2\mu}$, $w_{IC,u,x} = \frac{F\nu_u}{2\mu}$, $w_{IC,u,z} = -\frac{F(1-\nu_u)}{2\mu}$, and $w_{IC,p} = \frac{B(1+\nu_u)F}{3a}$.

After training, the network accuracy is evaluated on an equally spaced grid in $[0,1]$ m $\times [0,1]$ m $\times [0,10]$ s of $101\times101\times201$ points. The L^2-errors of equation (16) are plotted in Fig. 7. As for Terzaghi's problem, the L^2-errors of equation (16) denote quite a limited spread, providing evidence again of the PINN robustness. It appears also that displacements are very marginally affected by the hyper-parameter selection, with the worse outliers characterized by an error around 7%. By distinction, there exists a marked difference for the fluid pressure, where the realizations which correspond to the tanh activation function for displacements (from no. 1 to no. 72) exhibit a more consistent behavior than the remaining ones. Means and standard deviations of the errors are $\mu_p = 4.671\times10^{-2}$, $\mu_{u_x} = 4.653\times10^{-2}$, $\mu_{u_z} = 5.131\times10^{-2}$, and $\sigma_p = 1.314\times10^{-2}$, $\sigma_{u_x} = 6.522\times10^{-3}$, $\sigma_{u_z} = 6.268\times10^{-3}$, i.e., the same order of magnitude as Terzaghi's problem. As for the 1D case, Fig. 7 provides evidence of the robustness of the method, since around 70% of the possible architectures leads to errors inside tight $\mu_\cdot \pm \sigma_\cdot$ intervals.

According to the main effects, the activation function of the displacements is the hyper-parameter mostly affecting the overall network behavior. In this case, it appears preferable to set it at tanh. The other hyper-parameter showing an appreciable influence on the network accuracy is the number of layers for both displacement and pressure, the latter exclusively for vertical displacement errors. The response optimization process gives the architecture provided in Table 4.

5. PINN application with a sensor-driven condition

The quality and effectiveness of the PINN architectures identified in the previous section are tested on a more challenging situation, with the aim at applying a PINN strategy to something closer to a real-world application. First, we use the PINN model as a standard forward PDE solver with no other information but the initial and boundary conditions. Second, we add a few pieces of information as training data, arising from a synthetic "sensor" installed at some fixed locations to monitor the evolution of the process. This is quite a common situation that may occur in real-world applications, where data can be recorded at some points of the space-time domain. Third, to better replicate the condition driven by measured data, we add some noise to the synthetic data. The key idea here is to show that even a few training data points from a very limited subset of the domain can substantially improve the computational efficiency of the numerical procedure for minimizing the loss function in the forward PINN solution of a PDE system. At the same

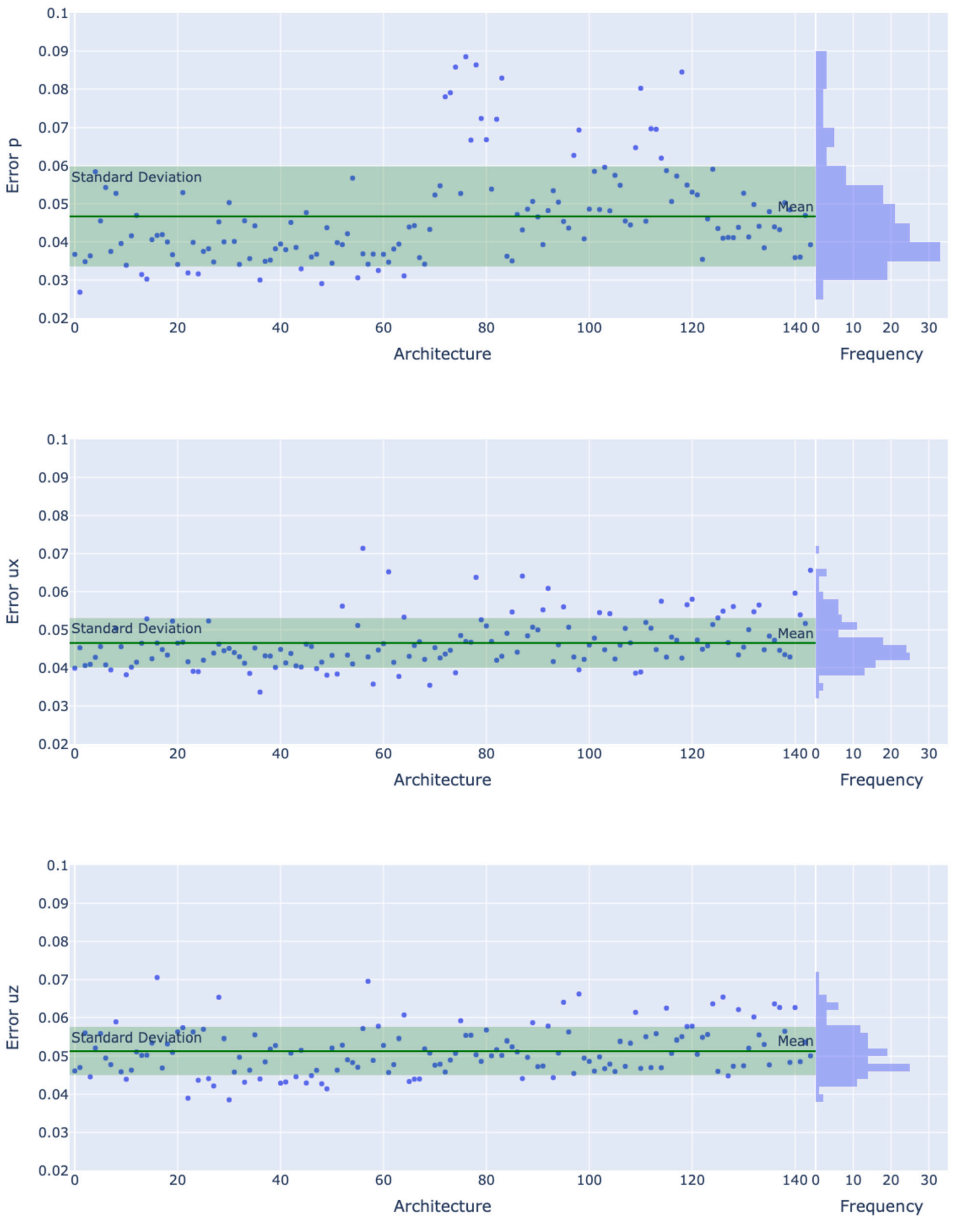


Fig. 7. Mandel's problem: errors in the pressure (top), horizontal (middle) and vertical displacement (bottom) reconstruction for the different architectures. On the rightmost frames, the error statistical distribution is provided.

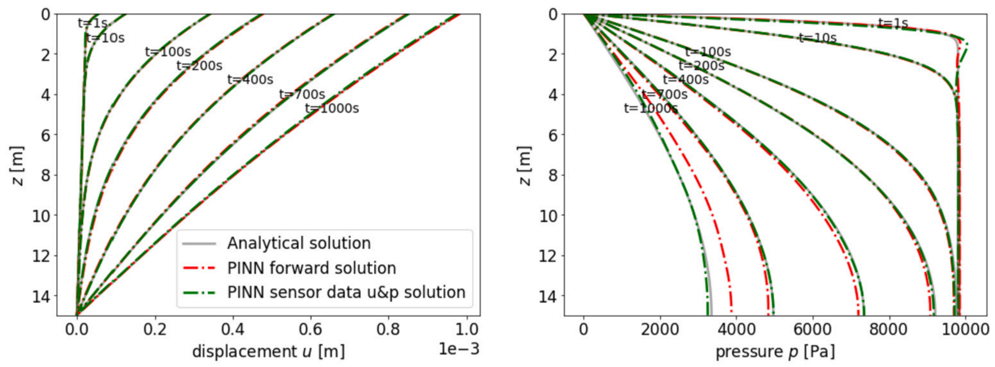


Fig. 8. Terzaghi's problem: comparison between analytical and PINN solution using the model as a forward PDE solver.

time, exploiting the available, and even noisy, data can significantly improve the quality of the PINN prediction. We denote the applications investigated in this section as "sensor-driven" simulations, in the sense that we assume that data within the domain are available only where a physical sensor is installed.

The procedure is here applied to artificially-created synthetic examples, but the objective is to assess the PINN performance in view of its application to a real case, where monitoring data describing the evolution of the physical process are available at some specified locations. Both the one-dimensional and the two-dimensional configurations of Section 4 are tested and compared with a strictly forward solution.

5.1. 1D: Terzaghi's problem

Following the outcome of Table 4, we build the PINN model by using a pressure network with 10 layers, 40 neurons and ELU activation function, while the displacement network has 12 layers, 20 neurons and ELU activation function. This PINN model is used to solve the classical 1D Terzaghi's consolidation problem in the setting of Fig. 1 with the initial and boundary conditions of equations (19)-(20). Since we assume to use PINN as a forward PDE solver with no additional information available, the number of collocation points has to be significantly increased with respect to the analysis carried out in Section 4. In particular, to achieve a sufficient accuracy we needed $N_c = 20000$ collocation points inside the domain where the PDE residual is evaluated, $N_{IC} = 6000$ initial points and $N_{BC} = 9000$ points uniformly distributed in time along the z-boundary. The training is performed with Adam algorithm for 30000 epochs, with a batch size equal to 1000 and a learning rate constantly equal to 0.001 for the first 2000 epochs and then with an exponential decay to 0.0001 until the 5000th epoch and to 0.00001 until the end. The loss function is the same as in (10), but the weights vary following a Neural Tangent Kernel (NTK) guided gradient descent. This is chosen because it has been verified that this method generally improves convergence [52]. All the other parameters have been chosen the same as in Section 4.1, with no early stopping. The accuracy of the outcome is satisfactory (Fig. 8) with relative L^2-errors for pressure and displacement at about 2% and 0.7%, respectively.

Now, we assume that a sensor is located at $z = 7$ m (Fig. 1) collecting pressure and displacement values in time. Exploiting the data coming from one single point in the physical domain, the number of collocation, initial and boundary points used for the PDE residuals and the auxiliary conditions can be reduced to $N_c = 10000$, $N_{IC} = 3000$, and $N_{BC} = 4000$, with no other data points inside the domain, except for $N_d = 3000$ points at $z = 7$ m (Fig. 9a). Pressure and displacement data at the sensor location are synthetically obtained from the analytical solution (21). In order to better reproduce a realistic setting, noise has been also added to the exact values by introducing an error with a Gaussian distribution, zero mean and a standard deviation equal to 5% of exact values (Fig. 9b), similarly to what is done in [8].

The training is carried out with the same parameters as before, but now it can be stopped after 10000 epochs. The trends of the loss functions provided in Fig. 10 clearly show the improvement allowed by the introduction of one "sensor" point only in space with respect to a simple forward solution in terms of both speed of convergence and accuracy. The mean of the L^2-norms of the errors on the validation set of three runs is about 0.5% for the predicted pressure and 0.2% for the vertical displacement (Table 5). Notice that also in case of noisy data we have an improvement both in the accuracy and in the convergence speed, hence in the overall computational load.

We considered also the case where only partial measurements are available, i.e. only pressure points are used within the domain at $z = 7$ m with no data for displacements. This is usually closer to a real-world hydro-poromechanical problem, since in many applications, for example in subsurface engineering, it is generally easier and less expensive to monitor the fluid pressure in time with respect to deep displacements. In these last cases the accuracy obviously decreases, but the approximation errors still remain acceptable from an engineering application point of view, as it can be noticed in Table 5, with the related loss functions still efficiently minimized (Fig. 10a).

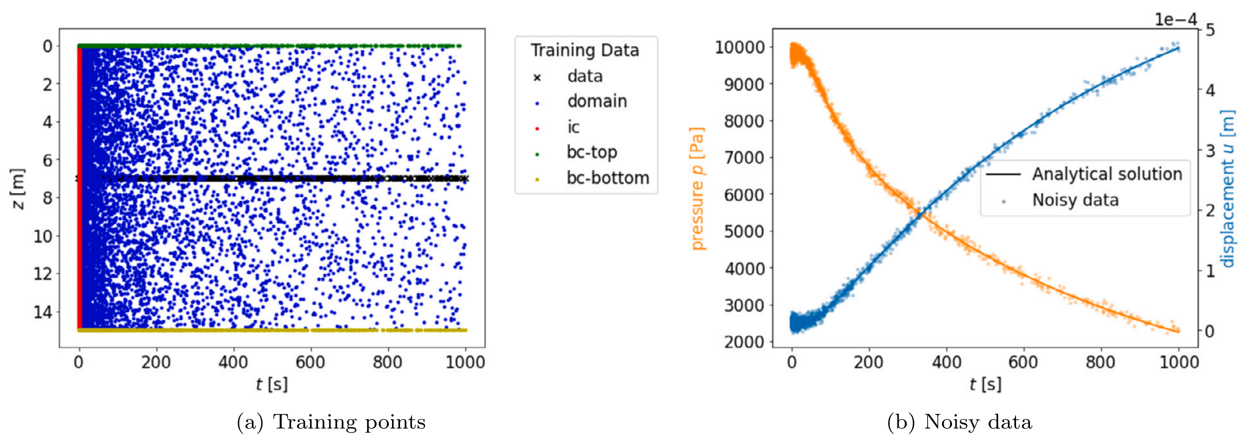


Fig. 9. Terzaghi's problem: (a) training points in the "sensor-driven" simulation in Terzaghi's problem with data given only at $z = 7$ m; (b) 5% noise is added to the training sensor data to better reproduce a realistic setting.

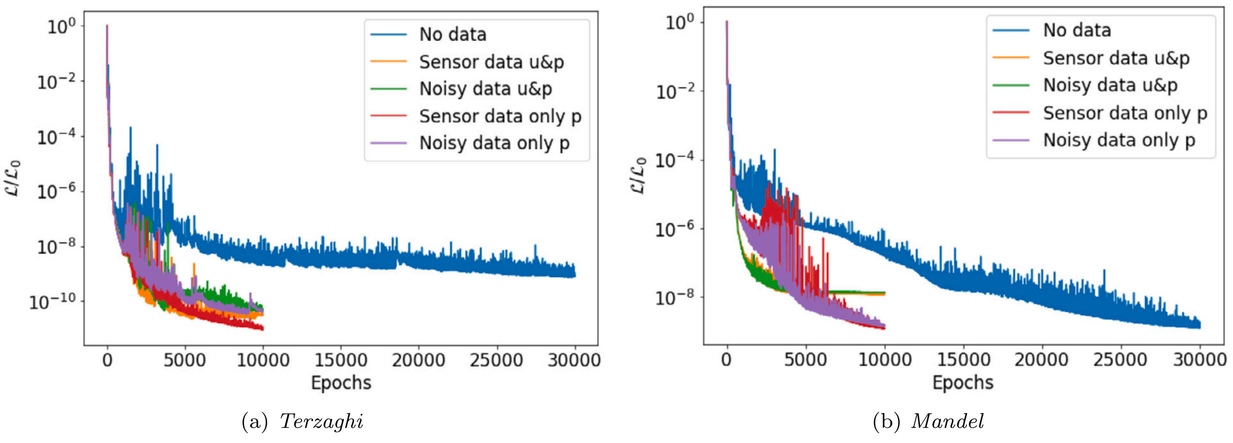


Fig. 10. Comparison between the losses relative to the initial value $\mathcal{L}_0$ of the forward and "sensor condition" frameworks.

Table 5
Errors for the different model applications. The mean errors computed from three runs and different random seeds are provided.

	Terzaghi 1 sensor at $z = 7$ m		Mandel 2 sensors at $(x, z) = (0.3, 0.3)$ m, $(0.7, 0.7)$ m		
	Error p	Error u	Error p	Error ux	Error uz
No data	2.279×10^{-2}	7.205×10^{-3}	2.655×10^{-1}	1.921×10^{-1}	1.268×10^{-1}
Sensor data u&p	4.891×10^{-3}	2.731×10^{-3}	4.311×10^{-2}	3.521×10^{-2}	5.571×10^{-2}
Noisy data u&p	1.556×10^{-2}	3.106×10^{-3}	2.177×10^{-2}	3.518×10^{-2}	6.799×10^{-2}
Sensor data only p	6.654×10^{-3}	1.314×10^{-2}	2.595×10^{-2}	1.446×10^{-1}	1.265×10^{-1}
Noisy data only p	1.411×10^{-2}	1.051×10^{-2}	4.327×10^{-2}	1.437×10^{-1}	1.275×10^{-1}

5.2. 2D: Mandel's problem

The same analysis as in Section 5.1 is applied to Mandel's problem. We consider the PINN architecture defined by the hyper-parameters in Table 4, using for fluid pore pressure 4 layers with 20 neurons and tanh activation, and for horizontal and vertical displacement 8 layers with 20 neurons and tanh activation. We refer to the problem setting of Fig. 6 and carry out an initial test with no other data but the initial and boundary conditions (25)-(26). The networks are trained over $N_c = 40000$ collocation points for the PDE residuals, $N_{IC} = 8000$ initial points, and $N_{BC} = 12000$ points equally distributed along the four spatial boundaries. The loss function is built as stated before with the NTK weighting. The training is performed for 30000 epochs with Adam algorithm, a 0.001 learning rate constant until the 2000th epoch, with an exponential decay to 0.0001 for 3000 epochs and then to 0.00001 until the end. The size of the batch has been chosen equal to 2000, while the other choices follow those of the previous Section 4.2 without early stopping.

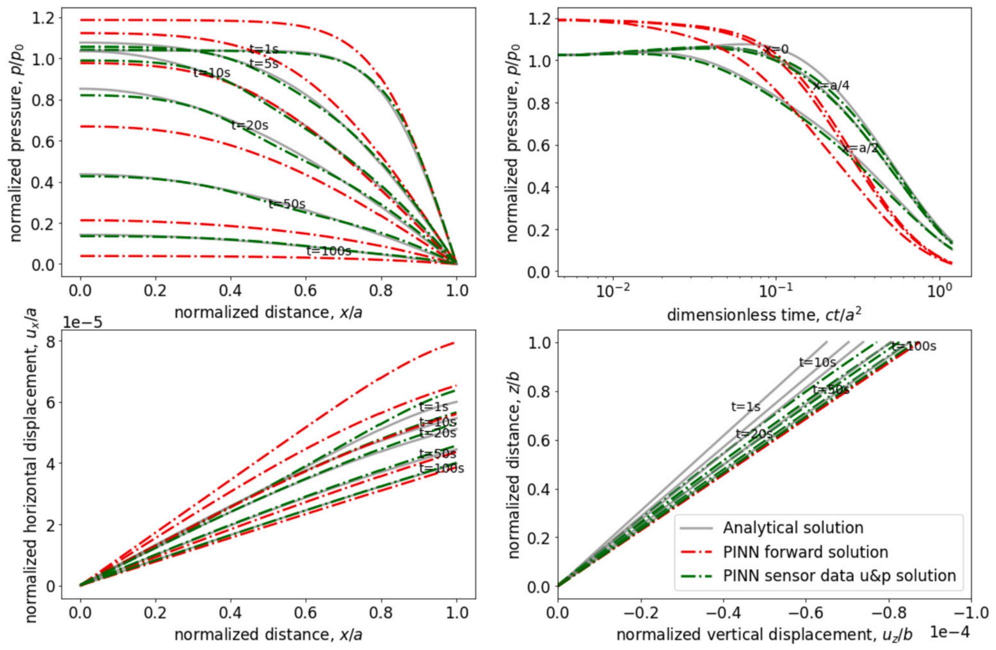


Fig. 11. Mandel's problem: comparison between analytical and PINN solution using the model as a forward PDE solver.

Fig. 11 shows a comparison between the PINN approximation and analytical solution. The outcome is only fairly satisfactory, with discrepancies that can be appreciated for both the pressure and the displacement solution. In this case, the relative L^2-error norm is around 26% for pressure, 14% for horizontal displacement and 13% for vertical displacement.

If we add a few pieces of information, the training process can be substantially improved. We suppose that two sensors are located at $(x, z) = (0.3, 0.3)$ m and $(x, z) = (0.7, 0.7)$ m (Fig. 6) collecting pressure and displacement values in time. As before, the synthetic measurements in time taken from the analytical solution (27) at the sensor locations are also perturbed by some noise with the same characteristics as for Terzaghi's problem. The networks are now trained over $N_c = 20000$, $N_{IC} = 6000$, and $N_{BC} = 8000$ points, while $N_d = 6000$ training data points split into 3000 data points are collected at each location in time. As already observed in Terzaghi's application, the introduction of a "sensor-driven" condition significantly improves both the accuracy and the convergence speed. The loss minimization process requires only 10000 epochs with the behavior shown in Fig. 10b, while the relative L^2-errors decrease to 4.3%, 3.5% and 5.6% for pressure, horizontal and vertical displacement, respectively (Table 5).

As done before, we considered also the case where only pressure measurements are available. A summary of the errors is reported in Table 5. In Mandel's case, the errors are larger with respect to Terzaghi's case, but the advantage of the "sensor-driven" framework is still evident (Fig. 10b and Fig. 11). In particular, in this example, the precision reached by the forward solution is not fully satisfactory. Assuming that the information is available both for the pressure and displacements field, the results are again quite accurate. As expected, in the case where only pressure data are available, the accuracy for pressure remains quite good, but the errors for the approximations of the displacements are higher.

5.3. *Sensor-driven condition with perturbed material parameters*

When dealing with real-world applications, the knowledge of the problem is always affected by a number of uncertainties. In the field of poromechanics, the most significant source of uncertainty is often the hydro-geological characterization of the porous medium, because of the difficulty intrinsically connected to the accurate measurement of material properties in the subsurface and their natural variability in space. Prior assumptions in the problem formulation can be derived from laboratory tests, e.g., [53], or in-situ direct or indirect observations with specialized measurement equipment, e.g., [54,55]. However, most of the measurement techniques are representative of the medium only at a very local level, hence they are naturally affected by some inaccuracies in the estimate of the actual material properties. Moreover, in a practical situation the general behavior of the physical process is known, but many other different phenomena, which can be hardly quantified, may affect the outcome in an unpredictable way. The proposed *sensor-driven* framework can also help in addressing this issue, since the integration of some data measured in the site can guide to the actual solution of the problem and help to account for marginal effects due to several minor dynamics that are not explicitly accounted for in the governing model equations. In this context, the solution is not forced to strictly satisfy the assumed main process, but can balance it with the actual measurements and with other natural occurrences.

We investigate this situation in Mandel's problem, where we simulate a scenario in which the prior knowledge of the geomechanical and hydraulic properties of the medium is affected by an error. Suppose that the available material parameters $\lambda = \mu = 40$ MPa and $k = 10^{-5}$ m/s are inaccurate and that their true value is $\lambda^* = \mu^* = 50$ MPa and $k^* = 10^{-4}$ m/s, i.e., the mechanical properties are

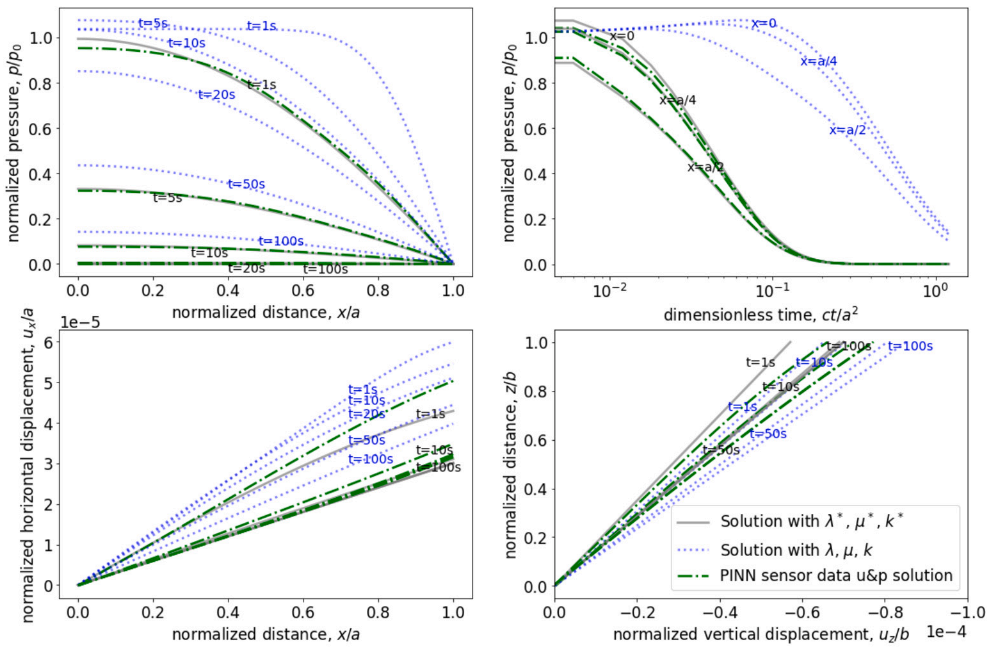


Fig. 12. Mandel's problem: test case with inconsistent PDE parameters and sensor data.

Table 6
Errors of the PINN solution of Mandel's problem for the case of sensor-driven condition with perturbed parameters. The mean errors computed from three runs and different random seeds are provided.

	Mandel - perturbed parameters 2 sensors at $(x,z) = (0.3, 0.3)$ m, $(0.7, 0.7)$ m		
	Error p	Error ux	Error uz
Sensor data u&p	5.415×10^{-2}	8.405×10^{-2}	8.096×10^{-2}
Sensor data only p	7.934×10^{-2}	2.765×10^{-1}	2.615×10^{-1}

affected by a 20% underestimate and the hydraulic conductivity is incorrect by one order of magnitude. Hence, the sensor data for pressure and displacements, $\{p^i, u_x^i, u_z^i\}_{i=1}^{N_d}$, correspond to the values obtained from the solution (27) at $(0.3, 0.3)$ m and $(0.7, 0.7)$ m with λ^*, μ^* and k^*, which is not consistent with the prior available information used to compute the residual of the governing equations with λ, μ, and k. We build $\hat{p}$, $\hat{u}_x$, and $\hat{u}_z$ with the architecture defined in Table 4 and we train them by minimizing the misfit with available sensor data and the residual of (23)-(24), along with boundary and initial conditions (25)-(26), with inaccurate parameters. The same training conditions as Section 5.2 apply.

Fig. 12 compares the solution obtained with PINNs, the true solution corresponding to λ^*, μ^* and k^*, and the solution corresponding to the incorrect parameters λ, μ and k. Table 6 reports the error with sensor data available for either both pressure and displacements, or pressure only. The results show that even if the PDE residual introduced in the loss function arises from an inaccurate estimate of the true material parameters, its combination with available sensor data allows the PINN-model to converge to the actual solution, with an error and convergence time comparable to the sensor-driven case with consistent parameters. It is worth emphasizing that this outcome would not have been possible with any forward PDE solver. By distinction, the use of a PINN-based model appears to be very flexible in adjusting the prediction so as to respect the underlying physics learned from the governing equation and at the same time fitting the actual available observations.

5.4. Heterogeneous consolidation problem

Finally, the PINN application with sensor-driven condition is investigated in a more complex and realistic setting. We simulate the consolidation of a heterogeneous medium, consisting of clay and sand layers, subjected to a surface distributed load. This stratified subsurface setting is a representation of the actual shallow subsurface conditions in sedimentary basins and is a first step towards the use of PINN-based models in a realistic configuration with real sensor data.

We consider the 2D domain sketched in Fig. 13, with a clay layer lying on top of a sandy formation. The constant uniformly distributed load P_H is applied on the ground surface. The lower and upper layers have thickness H_1 and H_2, respectively. The heterogeneity is modeled by setting λ, μ, and k as the following jump functions:

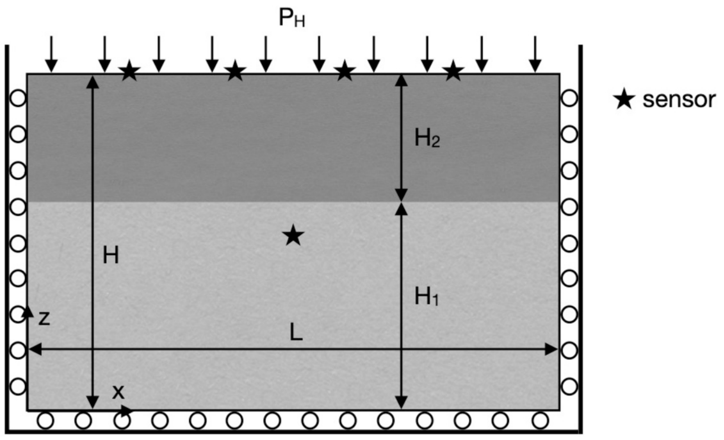


Fig. 13. Sketch of the setup for 2D consolidation in a heterogeneous setting with indication of sensor locations.

$$\lambda(x,z)=\begin{cases}\lambda_s & \forall (x,z)\in[0,L]\times[0,H_1]\\ \lambda_c & \forall (x,z)\in[0,L]\times]H_1,H]\end{cases},\qquad k(x,z)=\begin{cases}k_s & \forall (x,z)\in[0,L]\times[0,H_1]\\ k_c & \forall (x,z)\in[0,L]\times]H_1,H]\end{cases},\tag{29}$$

and $\mu(x,z)=\lambda(x,z)$. In other words, the collocation points can be marked as belonging to the sandy or clayey layer, so as to identify which value of the material parameters has to be used in order to compute the corresponding residual. The full space-time domain is $\Omega=[0,L]\times[0,H]\times[0,+\infty)$. The initial conditions read:

$$p_0(x,z)=\begin{cases}0 & z=H\\ P_H & \text{otherwise}\end{cases},\qquad u_{x,0}(x,z)\ =0,\qquad u_{z,0}(x,z)=0,\tag{30}$$

and the boundary conditions are:

$$\begin{aligned}
&\frac{\partial p}{\partial x}(0,z,t)=0, && u_x(0,z,t)=0, && \frac{\partial u_z}{\partial x}(0,z,t)=0, && x=0,\\
&\frac{\partial p}{\partial x}(L,z,t)=0, && u_x(L,z,t)=0, && \frac{\partial u_z}{\partial x}(L,z,t)=0, && x=L,\\
&\frac{\partial p}{\partial z}(x,0,t)=0, && \frac{\partial u_x}{\partial z}(x,0,t)=0, && u_z(x,0,t)=0, && z=0,\\
&p(x,H,t)=0, && \frac{\partial u_x}{\partial z}(x,H,t)=0, && (\lambda_c+2\mu_c)\frac{\partial u_z}{\partial z}(x,H,t)=-P_H, && z=H.
\end{aligned}\tag{31}$$

We set the external load $P_H=90$ kPa, and the domain dimensions $L=100$ m and $H=30$ m, with $H_1=18$ m and $H_2=12$ m the heights of the lower and upper layers, respectively. All the other parameters are chosen as indicated in Table 3. The numerical final time instant is $T=218160$ s. Since the analytical solution for this problem is not available in close form, we use as reference solution the one obtained by solving numerically the problem at hand with a very fine space and time discretization. In particular, we used the stabilized Mixed Finite Element approach presented in [56] with bilinear elements for the displacement representation and the lowest order Raviart-Thomas space for fluid pressure and velocity. This scheme was used for its stability and accuracy, already verified in a number of real-world test cases. The numerical reference solution has been used to extract the sensor data as well.

The sensor locations, highlighted in Fig. 13, are selected considering the potential data availability in a real-world setting. Since nowadays the ground surface motion can be easily detected with a great accuracy over a large number of points in space and time by satellite measurements, e.g., Interferometric Synthetic Aperture Radar (InSAR) maps and Continuous Global Positioning System (CGPS) stations, we consider displacement data available from four top surface points uniformly distributed over the domain ($x=20,40,60,80$ m). By distinction, data from the subsurface are usually much more difficult to detect, so we assume to have only one sensor in the position $(x,z)=(50,15)$ m, measuring both pressure and vertical displacement in time. For instance, these data can be realistically provided by a single borehole instrumented with both a piezometer and an extensometer.

We build $\hat{p}$, $\hat{u}_x$, and $\hat{u}_z$ with the architecture defined in Table 4 for Terzaghi's test problem. The training is carried out by minimizing the misfit with available sensor data and the residual on the governing equations, the boundary and the initial conditions at $N_c=10000$ collocation points randomly distributed over the domain Ω, $N_{IC}=3000$, $N_{BC}=4000$. For each sensor, we provide 25 data computed with the numerical reference solution, so $N_d=125$. A logarithmic distribution is selected, as usual, in time. The same training technical choices as in Section 5.2 are used. The outcome of our sensor-driven PINN-based solution of Biot's model is compared to the reference numerical result in Fig. 14, while Table 7 displays the approximation errors (16) for each of the unknown variables, p, u_x, u_z. The results show that, also in this more realistic test case, the PINN-based model is able to correctly learn the physics generating the reference solution behavior and the heterogeneity of the material properties. In particular, the errors are quantitatively comparable with those obtained in the theoretical benchmarks characterized by an analytical solution. The present outcome is particularly encouraging in view of the possible application of the proposed approach with real-world sensor data.

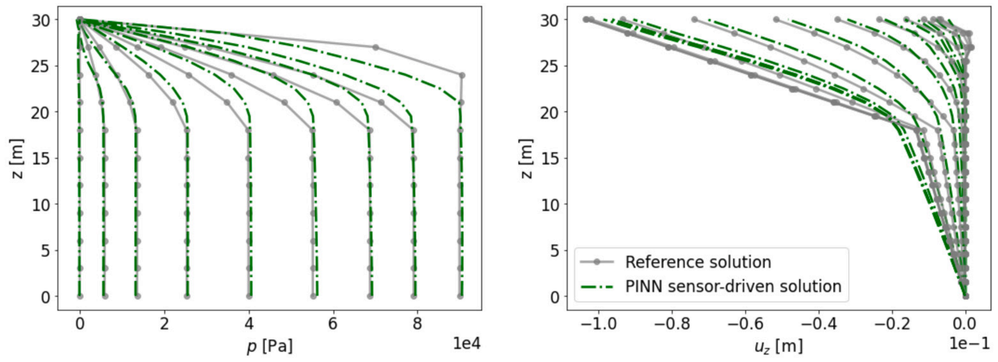


Fig. 14. Heterogeneous consolidation problem: comparison between the reference and the PINN-based sensor-driven solution.

Table 7
Errors of the PINN-based sensor-driven solution for the heterogeneous consolidation problem.

Heterogeneous consolidation		
Error p	Error ux	Error uz
3.466×10^{-2}	1.231×10^{-2}	1.696×10^{-1}

6. Conclusions

The implementation of PINN models for problems governed by Biot's equations of hydro-poromechanics has been investigated. By this approach, prior knowledge (theoretical, empirical, mathematical, observational, etc.) of a problem is used to enhance the overall modeling procedure. However, the construction of effective neural networks for an accurate representation of the physical process depends on the selection of a number of hyper-parameters and is often quite expensive, ultimately remaining on the modeler's experience rather than on robust indications. For this reason, we have carried out an extensive experimentation to analyze the structure of effective architectures for the accurate PINN training in poromechanics. The analysis shows that some hyper-parameters are more influential than others in controlling the PINN model accuracy. Such a knowledge can limit the usually time-demanding empirical process required to build the approximating networks, driving the user into the design of NN architectures for the problem of interest. On the other side, PINN implementation on coupled problems suffers from well-known drawbacks as the high computational cost and the difficulties related to the multi-objective loss minimization. These limitations make the method non-competitive with most traditional and well-established PDE solvers based on discretization methods. However, PINNs have the advantage of allowing for an automatic data integration in the PDE solution stage. In this work, we introduced a "sensor-driven" framework, with the purpose of both accelerating the convergence of the minimization process and increasing the accuracy of the method. The numerical experiments on synthetic test cases show that integrating very few samples in space can substantially improve the PINN performance as a forward PDE solver. On summary, the results that follow are worth summarizing.

- The most significant hyper-parameter is the activation function of the neural networks approximating displacements. Care must be taken in its setting, restricting the choices to tanh and ELU.
- The NNs for poromechanical applications do not seem to require a very complex structure to achieve approximation relative errors lower than 10^{-2}. Twelve layers or less, along with no more than 40 neurons per layer, are enough.
- The magnitude of the approximation errors appears to be acceptable ($< 10\%$) for a wide range of hyper-parameter combinations, providing evidence of a good matching between analytical and PINN solutions.
- The analysis emphasizes the robustness of the PINN approach, also due to the presence of the PDE residuals in the loss function acting as regularization terms. This means that there is a high chance to reach good accuracy once identified only the most significant hyper-parameters.
- The proposed "sensor-driven" architecture has resulted in improvements with respect to a pure forward solution in terms of both computational training time and model accuracy.
- The application of the analysis to more practical cases, such as assuming the availability of a limited number of observations, has provided promising results, since it has led to accurate approximations from data given only on very few (1 or 2) fixed locations.
- The proposed framework has showcased its effective applicability in cases affected by uncertainties in the problem formulation and heterogeneous realistic settings. This appears to be a good starting point for the assimilation of data in real-world problems' modeling, and will be the object of specific future investigations.

The PINN approach appears to be very promising for the generation of surrogate models, which could be of support in planning and decision-making processes by taking advantage of the machine learning nature of the method. Moreover, the capability of storing knowledge by training a NN on a problem and then applying it to a similar one, called transfer learning, can be advantageous in case of limited physics understanding or short time changes. This can yield cheap and fast generalizations of the models, representing another attractive aspect to be investigated in the next future. PINN application in a heterogeneous context could be further investigated and an ad hoc implementation, such as a cusp-capturing PINN [57], could improve the capacity to model heterogeneities. In the near future, the goal is to apply the presented PINN approach to real-world applications, where materials properties are not homogeneous and data are available from real sensors installation.

CRediT authorship contribution statement

Caterina Millevoi: Conceptualization, Investigation, Methodology, Writing – original draft. **Nicolò Spiezia:** Investigation, Writing – review & editing, Conceptualization. **Massimiliano Ferronato:** Conceptualization, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: co-author Nicolò Spiezia is currently employed in the private company M3E S.r.l.

Data availability

Data will be made available on request.

References

[1] K. Terzaghi, Erdbaumechanik auf bodenphysikalischer Grundlage, F. Deuticke, Leipzig U. Wien, 1925.
[2] M.A. Biot, General theory of three-dimensional consolidation, J. Appl. Phys. 12 (1941) 155–164, https://doi.org/10.1063/1.1712886.
[3] O. Coussy, Mechanics of Porous Continua, Wiley, Chichester, 1995.
[4] H.F. Wang, Theory of Linear Poroelasticity with Applications to Geomechanics and Hydrogeology, Princeton University Press, 2000, http://www.jstor.org/stable/j.ctt1jktrr4.
[5] S.J. Wang, K.C. Hsu, The application of the first-order second-moment method to analyze poroelastic problems in heterogeneous porous media, J. Hydrol. 369 (1) (2009) 209–221, https://doi.org/10.1016/j.jhydrol.2009.02.049.
[6] M. Ferronato, N. Castelletto, G. Gambolati, A fully coupled 3-D mixed finite element model of Biot consolidation, J. Comput. Phys. 229 (2010) 4813–4830, https://doi.org/10.1016/j.jcp.2010.03.018.
[7] N. Castelletto, J. White, H. Tchelepi, Accuracy and convergence properties of the fixed-stress iterative solution of two-way coupled poromechanics, Int. J. Numer. Anal. Methods Geomech. 39 (2015) 1593–1618, https://doi.org/10.1002/nag.2400.
[8] 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, J. Comput. Phys. 378 (2019) 686–707.
[9] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics informed deep learning (Part I): data-driven solutions, Nonlinear Partial Differ. Equ. (2017), arXiv:1711.10561.
[10] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics informed deep learning (Part II): data-driven discovery, Nonlinear Partial Differ. Equ. (2017), arXiv:1711.10566.
[11] D. Zhang, L. Lu, L. Guo, G.E. Karniadakis, Quantifying total uncertainty in physics-informed neural networks for solving forward and inverse stochastic problems, J. Comput. Phys. 397 (2019) 108850.
[12] M. Reichstein, G. Camps-Valls, B. Stevens, et al., Deep learning and process understanding for data-driven Earth system science, Nature 566 (2019) 195–204, https://doi.org/10.1038/s41586-019-0912-1.
[13] Y-Kevrekidis Karniadakis, G. L-Lu, Physics-informed machine learning, Nat. Rev. Phys. (2021) 1–19, https://doi.org/10.1038/s42254-021-00314-5.
[14] A.D. Jagtap, K. Kawaguchi, G.E. Karniadakis, Adaptive activation functions accelerate convergence in deep and physics-informed neural networks, J. Comput. Phys. 404 (2020) 109136, https://doi.org/10.1016/j.jcp.2019.109136.
[15] Y. Shin, J. Darbon, G.E. Karniadakis, On the convergence of physics informed neural networks for linear second-order elliptic and parabolic type PDEs, Commun. Comput. Phys. 28 (5) (2020) 2042–2074, https://doi.org/10.4208/cicp.oa-2020-0193.
[16] S. Wang, Y. Teng, P. Perdikaris, Understanding and mitigating gradient pathologies in physics-informed neural networks, arXiv:2001.04536, 2020.
[17] O. Fuks, H. Tchelepi, Limitations of Physics Informed Machine Learning for Nonlinear Two-Phase Transport in Porous Media, preprint, 07 2020.
[18] S. Cai, Z. Mao, Z. Wang, M. Yin, G.E. Karniadakis, Physics-informed neural networks (PINNs) for fluid mechanics: a review, Acta Mech. Sin. (2022) 1–12.
[19] X. Yang, S. Zafar, J.-X. Wang, H. Xiao, Predictive large-eddy-simulation wall modeling via physics-informed neural networks, Phys. Rev. Fluids 4 (3) (2019) 034602.
[20] Q. Lou, X. Meng, G.E. Karniadakis, Physics-informed neural networks for solving forward and inverse flow problems via the Boltzmann-BGK formulation, J. Comput. Phys. 447 (2021) 110676.
[21] E. Haghighat, M. Raissi, A. Moure, H. Gomez, R. Juanes, A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics, Comput. Methods Appl. Mech. Eng. 379 (2021) 113741, https://doi.org/10.1016/j.cma.2021.113741, https://www.sciencedirect.com/science/article/pii/S0045782521000773.
[22] A. Dourado, F.A. Viana, Physics-informed neural networks for missing physics estimation in cumulative damage models: a case study in corrosion fatigue, J. Comput. Inf. Sci. Eng. 20 (6) (2020).
[23] S. Cai, Z. Wang, S. Wang, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks for heat transfer problems, J. Heat Transf. 143 (6) (2021).
[24] C. Song, T. Alkhalifah, U.B. Waheed, Solving the frequency-domain acoustic VTI wave equation using physics-informed neural networks, Geophys. J. Int. 225 (2) (2021) 846–859.
[25] Q. Zhu, Z. Liu, J. Yan, Machine learning for metal additive manufacturing: predicting temperature and melt pool fluid dynamics using physics-informed neural networks, Comput. Mech. 67 (2) (2021) 619–635.
[26] F. Sahli Costabal, Y. Yang, P. Perdikaris, D.E. Hurtado, E. Kuhl, Physics-informed neural networks for cardiac activation mapping, Front. Phys. 8 (2020) 42.
[27] M. Yin, X. Zheng, J.D. Humphrey, G.E. Karniadakis, Non-invasive inference of thrombus material properties with physics-informed neural networks, Comput. Methods Appl. Mech. Eng. 375 (2021) 113603.
[28] G. Kissas, Y. Yang, E. Hwuang, W.R. Witschey, J.A. Detre, P. Perdikaris, Machine learning in cardiovascular flows modeling: predicting arterial blood pressure from non-invasive 4D flow MRI data using physics-informed neural networks, Comput. Methods Appl. Mech. Eng. 358 (2020) 112623.

[29] Q. He, D. Barajas-Solano, G. Tartakovsky, A.M. Tartakovsky, Physics-informed neural networks for multiphysics data assimilation with application to subsurface transport, Adv. Water Resour. 141 (2020) 103610.
[30] M.M. Almajid, M.O. Abu-Al-Saud, Prediction of porous media fluid flow using physics informed neural networks, J. Pet. Sci. Eng. 208 (2022) 109205.
[31] Z. Zhang, A physics-informed deep convolutional neural network for simulating and predicting transient Darcy flows in heterogeneous reservoirs without labeled data, J. Pet. Sci. Eng. (2022) 110179.
[32] C.G. Fraces, H. Tchelepi, Physics Informed Deep Learning for Flow and Transport in Porous, Media, 2021, arXiv:2104.02629.
[33] M. Yang, J.T. Foster, hp-variational physics-informed neural networks for nonlinear two-phase transport in porous media, J. Mach. Learn. Model. Comput. 2 (2) (2021).
[34] A.M. Tartakovsky, C.O. Marrero, P. Perdikaris, G.D. Tartakovsky, D. Barajas-Solano, Physics-informed deep neural networks for learning parameters and constitutive relationships in subsurface flow problems, Water Resour. Res. 56 (5) (2020) e2019WR026731.
[35] K. Wang, Y. Chen, M. Mehana, N. Lubbers, K.C. Bennett, Q. Kang, H.S. Viswanathan, T.C. Germann, A physics-informed and hierarchically regularized data-driven model for predicting fluid flow through porous media, J. Comput. Phys. 443 (2021) 110526.
[36] Y.W. Bekele, Physics-informed deep learning for one-dimensional consolidation, J. Rock Mech. Geotechn. Eng. 13 (2) (2021) 420–430, https://doi.org/10.1016/j.jrmge.2020.09.005, https://www.sciencedirect.com/science/article/pii/S1674775520301384.
[37] R. Santoso, D. Degen, M. Cacace, F. Wellmann, State-of-the-art physics-based machine learning for hydro-mechanical simulation in geothermal applications, 2022.
[38] Y.W. Bekele, Physics-informed deep learning for flow and deformation in poroelastic media, arXiv preprint, arXiv:2010.15426, 2020, arXiv:2010.15426.
[39] T. Kadeethum, T.M. Jørgensen, H.M. Nick, Physics-informed neural networks for solving inverse problems of nonlinear Biot's equations: batch training, in: 54th US Rock Mechanics/Geomechanics Symposium, OnePetro, 2020.
[40] E. Haghighat, D. Amini, R. Juanes, Physics-informed neural network simulation of multiphase poroelasticity using stress-split sequential training, arXiv:2110.03049, 2021.
[41] D. Amini, E. Haghighat, R. Juanes, Physics-informed neural network solution of thermo-hydro-mechanical (THM) processes in porous media, https://doi.org/10.48550/ARXIV.2203.01514, https://arxiv.org/abs/2203.01514, 2022.
[42] T. Kadeethum, T.M. Jørgensen, H.M.N. Hamidreza, Physics-informed neural networks for solving nonlinear diffusivity and Biot's equations, PLoS ONE 15 (5) (2020) 1–28, https://doi.org/10.1371/journal.pone.0232683.
[43] O. Hennigh, S. Narasimhan, M.A. Nabian, A. Subramaniam, K. Tangsali, M. Rietmann, J. del Aguila Ferrandis, W. Byeon, Z. Fang, S. Choudhry, NVIDIA SimNet™: an AI-accelerated multi-physics simulation framework, 2020, arXiv:2012.07938.
[44] L. Lu, X. Meng, Z. Mao, G.E. Karniadakis, DeepXDE: a deep learning library for solving differential equations, SIAM Rev. 63 (1) (2021) 208–228, https://doi.org/10.1137/19m1274067.
[45] A. Koryagin, R. Khudorozkov, S. Tsimfer, PyDEns: a python framework for solving differential equations with neural networks, arXiv:1909.11544, 2019.
[46] F. Chen, D. Sondak, P. Protopapas, M. Mattheakis, S. Liu, D. Agarwal, M.D. Giovanni, NeuroDiffEq: a Python package for solving differential equations with neural networks, J. Open Sour. Softw. 5 (46) (2020) 1931, https://doi.org/10.21105/joss.01931.
[47] E. Haghighat, R. Juanes, SciANN: a Keras/TensorFlow wrapper for scientific computations and physics-informed deep learning using artificial neural networks, Comput. Methods Appl. Mech. Eng. 373 (2021) 113552, https://doi.org/10.1016/j.cma.2020.113552, https://www.sciencedirect.com/science/article/pii/S0045782520307374.
[48] A.G. Baydin, B.A. Pearlmutter, A.A. Radul, Automatic differentiation in machine learning: a survey, CoRR, arXiv:1502.05767 [abs], 2015.
[49] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, arXiv:1412.6980, 2017.
[50] J. Brownlee, Deep Learning with Python: Develop Deep Learning Models on Theano and TensorFlow Using Keras, Machine Learning Mastery, 2016.
[51] G.E.P. Box, K.B. Wilson, On the experimental attainment of optimum conditions, J. R. Stat. Soc., Ser. B, Methodol. 13 (1) (1951) 1–38, https://doi.org/10.1111/j.2517-6161.1951.tb00067.x.
[52] S. Wang, X. Yu, P. Perdikaris, When and why PINNs fail to train: a neural tangent kernel perspective, J. Comput. Phys. (2021) 110768, https://doi.org/10.1016/j.jcp.2021.110768, https://www.sciencedirect.com/science/article/pii/S002199912100663X.
[53] S. Mantica, S.-K. Nguyen, V. G, G. Musso, B. M, G. F, Implementation of an elasto-viscoplastic constitutive law in Abaqus/Standard for an improved characterization of rock materials, in: Science in the Age of Experience, Boston, 2016.
[54] M. Ferronato, G. Gambolati, P. Teatini, N. Castelletto, C. Janna, II cycle compressibility from satellite measurements, Geotechnique 63 (2013) 479–486, https://doi.org/10.1680/geot.11.P.149.
[55] L. Gazzola, M. Ferronato, P. Teatini, C. Zoccarato, A. Corradi, M.C. Dacome, S. Mantica, Reducing uncertainty on land subsidence modeling prediction by a sequential data-integration approach. application to the arlua off-shore reservoir in Italy, Geomech. Energy Environ. 33 (2023) 100434, https://doi.org/10.1016/j.gete.2023.100434.
[56] M. Frigo, N. Castelletto, M. Ferronato, J.A. White, Efficient solvers for hybridized three-field mixed finite element coupled poromechanics, Comput. Math. Appl. 91 (2021) 36–52, https://doi.org/10.1016/j.camwa.2020.07.010.
[57] Y.-H. Tseng, T.-S. Lin, W.-F. Hu, M.-C. Lai, A cusp-capturing PINN for elliptic interface problems, J. Comput. Phys. 491 (2023) 112359, https://doi.org/10.1016/j.jcp.2023.112359, https://www.sciencedirect.com/science/article/pii/S0021999123004540.