DOI: 10.14529/jsfi220302

A General Neural-Networks-Based Method for Identification of Partial Differential Equations, Implemented on a Novel AI Accelerator

Mikhail A. Krinitskiy[1,2], *Victor M. Stepanenko*[2,3], *Alexey O. Malkhanov*[5], *Mikhail E. Smorkalov*[4,5]

Partial differential equations (PDEs) are pervasive in vast domains of science and engineering. Although there is huge legacy of numerical methods for solving direct and inverse PDE problems, these methods are computationally expensive for many fundamental and real-life applications, demanding supercomputer resources. Moreover, existing methods for PDEs identification assume concrete functional forms for the coefficients to be found, significantly limiting the range of possible solutions. The mentioned circumstances lead to increasing interest in AI-based methods for direct solving and identification of PDEs. In this study, we propose a novel method based on artificial neural networks (ANNs) for the identification of partial differential equations. The method does not require any strong *a priori* assumptions regarding the family of the functions approximating PDE coefficients. It allows one to approximate the coefficients of a PDE based on the observed evolution of PDE direct solution. We demonstrate efficacy and high accuracy of ANN-based method in case of diffusion equation and nonlinear diffusion-advection equation (Richards equation) applied to the simulation of heat and moisture transfer in soil. We demonstrate that the novel method implemented on Ascend platform using the mixed precision floating point operations overperforms the classical gradient descent method in Barzilai–Borwein stabilized modification (BBstab, realized on a conventional central processor), in terms of MAPE (mean absolute percentage error) and RMSE (root mean square error) of approximated coefficients at least an order of magnitude. We also found that ANN-method is much less sensitive to initial guess of parameters compared to BBstab approach. Since the considered equations are of generic form, we anticipate that the proposed ANN-based method can be successfully exploited in other applications. These potential applications include hydrodynamic-type problems, e.g., optimization of turbulence closures, where the assumed reference solutions of PDEs are usually obtained from high-resolution direct Navier-Stokes simulations.

Keywords: partial differential equations, artificial neural networks, machine learning, inverse problems, land surface model, Richards equation, Ascend platform.

Introduction

Partial differential equations (PDE) are pervasive in many domains of science and engineering. Over the past decades, multiple methods have been proposed which allow to solve PDEs numerically, such as finite elements method (FEM), finite differences method (FDM), finite volumes method (FVM), etc. Such methods allow to achieve good accuracy of the numerical solution, but often are computationally expensive, which becomes an important limitation in applications when one needs to perform thousands or even millions of simulations. Such scenarios are common in engineering, e.g., the problem of finding optimal configuration of design parameters may require a large number of forward simulations for individual parameter sets. Another category of problems where traditional numerical methods often require heavy computations are inverse

[1] Shirshov Institute of Oceanology, Russian Academy of Sciences, Moscow, Russian Federation
[2] Lomonosov Moscow State University, Moscow, Russian Federation
[3] Moscow Center of Fundamental and Applied Mathematics, Moscow, Russian Federation
[4] Skolkovo Institute of Science and Technology, Moscow, Russian Federation
[5] Huawei Nizhny Novgorod Research Center, Nizhny Novgorod, Russian Federation

or data assimilation problems where the governing PDEs need to be reconstructed based on the measurement data or data of very-fine-scale simulations recognized as “truth” (e.g., direct numerical simulations in turbulence [35]).

The limitations of classical numerical methods resulted in the increasing interest in AI-based methods for solving and/or identification of PDEs which hold a perspective of effectively addressing the above mentioned issues. These AI-based methods can be split into two major categories: data-driven methods and physics-informed ones. The former category includes methods which use only observation data to implicitly reproduce the physics of natural phenomena, with Fourier neural operator (FNO) [29] and deep operator networks (DeepONet) [32] being some of well-known examples. Physics-informed methods, in contrast, leverage known information about governing physical laws. The most famous and likely most widely used method in this category is physics-informed neural networks (and its flavours) [22, 38–40], developed by research group of professor G. Karniadakis at Brown University.

The application of AI-based methods allowed to achieve notable results in areas like protein structure prediction [4, 25], photonics [44], the solution to Schrödinger equation for fermions [37], quantum transport [45], molecular dynamics [23], climate analytics [28], weather prediction [43], computational fluid dynamics (CFD) [16, 30], solid mechanics [19], Earth radiation belt modeling [10] and others.

While the interest of scientific community in AI-based methods of solving PDEs does increase quickly, most case studies presented so far leverage single precision arithmetic and thus focus on commodity hardware such as CPU and general-purpose computing on graphics processing units (GPGPU), even though specialized deep learning accelerators could be a perfect hardware target for such technologies in theory, due to their higher computational performance. However, the important question which needs to be addressed to evaluate the usefulness of AI accelerators in the field of scientific computing is whether the half precision arithmetic typically supported by those accelerators would allow to achieve acceptable accuracy in the problems of interest. This motivates us to thoroughly consider the question of datatype precision in the current paper where to the best of our knowledge we present the first case study of using specialized AI accelerator for solving the PDE identification problem relevant to land surface modeling field via artificial neural network (ANN) based algorithm.

The paper is organized as follows: in the Section 1, we present the context of the problem of PDE identification and the specifics of the downstream tasks we consider in this study that are: soil heat conductance and soil water content dynamics. We introduce the PDEs that describe these physical processes. These PDEs are subjects to identify in our study. In Section 2, we present the methods for the identification of PDEs we developed in our study. Both classical method (Section 2.3) and the one based on artificial neural networks (ANNs) (Section 2.4) are presented in this section in a unified manner in order to make it easy for a reader to compare the approaches and find the key differences. In Section 2.4 we describe our method based on ANNs in detail, including the physics-guided regularizations, our improved scalars and tensors scaling scheme, and the scheme of random re-weighting of individual elements of loss sums. In Section 3, we present the results we deliver in this study. We compare the accuracy and the performance of classical identification method we reproduce in this study, with the accuracy and performance of the method based on ANNs we propose in this study. In Conclusions section, we summarize the paper and draw the conclusions based on our results.

1. Problem Setup

1.1. General Remarks on the PDE Identification Problem

Let us assume we have a partial differential equation or a system of PDEs supplemented with appropriate initial and boundary conditions:

$$B\mathbf{f} + A\left[\lambda(\mathbf{f})\right]\mathbf{f} = \mathbf{g}, \tag{1}$$

where B and A are differential operators, $\lambda(\mathbf{f})$ is a coefficient of the operator A which is a function of solution $\mathbf{f}$, $\mathbf{g}$ is a given right-hand side (r.h.s.) function of time and space. Below we confine ourselves to operators A which are linear on λ. The inverse problem is formulated as follows: the PDE solution $\mathbf{f}$ is given, B and A are known, $\lambda(\mathbf{f})$ has to be found. For any test function (coefficient) $\lambda^*(\mathbf{f}) = \lambda(\mathbf{f}) + \delta\lambda(\mathbf{f})$, we get a residual of (1), defined as ϵ:

$$B\mathbf{f} + A\left[\lambda(\mathbf{f}) + \delta\lambda(\mathbf{f})\right]\mathbf{f} - \mathbf{g} = \epsilon, \tag{2}$$

$$A\left[\delta\lambda(\mathbf{f})\right]\mathbf{f} = \epsilon. \tag{3}$$

The residual ϵ is zero if $\delta\lambda(\mathbf{f}) = 0$. The opposite is not true in general case, i.e., there might be non-trivial solutions $\delta\lambda(\mathbf{f})$ of equation (3) with ϵ. This means the solution of inverse problem is not unique in general case. The practical consequence of this fact is that minimizing ϵ to zero over a space of $\lambda^*(\mathbf{f})$ functions does not necessarily provide $\delta\lambda(\mathbf{f}) \to 0$. In this study, we impose additional constrains on $\lambda^*(\mathbf{f})$, which ensure the correct solution of the inverse problem, including the monotonicity and matching the known points of $\lambda(\mathbf{f})$ (see Section 2.4.3).

Let us assume now, that we have a solution of (1), $\mathbf{f}$, at a set of nodes in time and space (say, data of fine-scale measurements or simulations with *a priori* correct but computationally expensive mathematical model). Then, the equation (1) can be discretized as follows:

$$\epsilon^*_{h\tau} \doteq B_{h\tau}\mathbf{f}_{h\tau} + A_{h\tau}\left[\lambda(\mathbf{f}_{h\tau}) + \delta\lambda(\mathbf{f}_{h\tau})\right]\mathbf{f}_{h\tau} - \mathbf{g}_{\mathbf{h}\tau} = \epsilon_{h\tau} + O(h^n) + O(\tau^m), \tag{4}$$

where h and τ stand for spatial and temporal spacing of the mesh where the above mentioned data is available, $\mathbf{f}_{h\tau}$, $\mathbf{g}_{h\tau}$, $\epsilon_{h\tau}$ are the projections of $\mathbf{f}$, $\mathbf{g}$, ϵ onto the mesh, n and m are the orders of approximation of A and B by discrete analogues $A_{h\tau}$ and $B_{h\tau}$, respectively. Now, under fixed h and τ, minimizing the norm of residual of (4), $||\epsilon^*_{h\tau}||$, over $\lambda^*(\mathbf{f}_{h\tau})$ does not lead to vanishing of $\epsilon_{h\tau}$ and $\delta\lambda(\mathbf{f}_{h\tau}) \to 0$. This is because the residual of discretized PDE is caused by both the deviation of test coefficient function from the true one *and* discretization errors, so that vanishing $||\epsilon^*_{h\tau}||$ means $\epsilon_{h\tau} \to O(h^n) + O(\tau^m)$. In this study, we minimize the discretization issue by using fine-resolution grids for PDE approximation.

1.2. Equations for Soil Thermodynamics and Hydrodynamics in Weather and Climate Models

The land surface scheme is a necessary compartment of any numerical weather forecast system or the Earth system model. Its key feature is the presence of a number of loosely constrained parameters of PDE coefficients which need to be identified using observation data. This includes the biophysical properties of soil and vegetation responsible for simulation of soil moisture regime and river runoff, *inter alia.* In this paper, we suggest a new ANN-based approach for identification of heat conductance and Richards equations. These equations are standard for land surface schemes, and are used in the land surface scheme jointly developed by the Institute of Numerical Mathematics (INM) RAS and Lomonosov Moscow State University (MSU). This land surface scheme is a module of the national INMCM Earth system model [42] and SLAV numerical weather forecast system [13].

The basic numerical kernel of any land surface scheme is a solver for an equation system governing heat and water transport in soil. This system includes heat equation:

$$\rho c \frac{\partial T}{\partial t} = \frac{\partial}{\partial z}\left(\lambda_T \frac{\partial T}{\partial z}\right) + \rho_d \left(L_i F_i - L_v F_v\right), \tag{5}$$

where T is temperature, ρc is volumetric specific heat of soil, ρ_d is dry soil density, L and F are specific heat and rate of water phase transition (subscript i standing for freezing/melting and v denoting evaporation/condensation processes), t is time, z is a spatial coordinate directed along gravity. Importantly, the heat conductivity coefficient λ_T is a function of liquid water content W (expressed in kg/kg or m^3/m^3), i.e., it depends on the solution of the equation set. The l.h.s. represents the change of soil enthalpy, and terms to the r.h.s. stand for heat conductance and heat release/consumption due to phase changes. An equation for liquid water content is referred to as Richards equation and takes into account vertical transport (diffusion and gravitational infiltration), freezing/melting and evaporation/condensation:

$$\frac{\partial W}{\partial t} = \frac{\partial}{\partial z}\left(\lambda_W \frac{\partial W}{\partial z}\right) + \frac{\partial \gamma}{\partial z} - F_i - F_v - R_{roots} - R_{runoff}, \tag{6}$$

where γ is gravitational water flux, R_{roots} is soil moisture uptake by roots, and R_{runoff} is a sink of water due to horizontal runoff. This equation involves a concept of soil moisture potential (or water retention curve, WRC), Ψ, and hydraulic conductivity, γ, which are dependent on moisture W, defining coefficients λ_W, γ as functions of PDE solution. The liquid water diffusivity $\lambda_W(W)$ and hydraulic conductivity (HC) $\gamma(W)$ are related functions, i.e.:

$$\lambda_W(W) = \gamma(W)\frac{\partial \Psi(W)}{\partial W}. \tag{7}$$

At least 22 semi-empirical forms are proposed for the WRC function [12], fitting different sets of empirical data with different performance. The WRC function explicitly enters the hydraulic conductivity function. For instance, choosing Mualem approach for HC quantification [36], one arrives to:

$$\gamma = \gamma_s \widetilde{W}^{1/2}\left[\int_0^{\widetilde{W}} \frac{d\widetilde{W}'}{\Psi(\widetilde{W}')}\left(\int_0^1 \frac{d\widetilde{W}'}{\Psi(\widetilde{W}')}\right)^{-1}\right]^2, \tag{8}$$

where $\widetilde{W} \doteq (W - W_r)/(W_s - W_r)$ is a degree of soil moisture saturation, subscripts “s” and “r” standing for saturated and residual quantities of liquid water in a soil matrix. Thus, introducing in (8) and (7) $n = 22$ forms of WRC yields 22 possible pairs of functions (λ_W, γ), based on Mualem equation. Given existence of different theoretical approaches (including Mualem’s formulation (8)) to quantify hydraulic conductivity [9, 14, 36], say, m approaches, we get $m*n$ possible functional forms for a pair (λ_W, γ), which is ~ 100 for the current state of the field. This clearly shows that no generally accepted formulations for these two coefficients are available, so that a method, deriving coefficients $\lambda_W(W)$ and $\gamma(W)$ *independently* from data on measurable variables such as W with very general *a priori* assumptions on these functions (such as positive definiteness, monotonicity) would ease the collection of more empirical data on WRC and HC functions and eventually serve the development of unified theory.

The dynamics of water vapor (V, kg/kg) in soil pores in the INM RAS-MSU land surface model is governed by diffusion equation with evaporation/condensation term:

$$\frac{\partial V}{\partial t} = \frac{\partial}{\partial z} \lambda_V \frac{\partial V}{\partial z} + F_v, \tag{9}$$

(λ_V – diffusivity coefficient for the water vapor) while the evolution of ice content I (kg/kg) is defined by phase transitions only:

$$\frac{\partial I}{\partial t} = F_i. \tag{10}$$

The system of equations (5), (6), (9), (10) is supplemented in the land surface model by boundary conditions, representing heat and moisture balance at the top and bottom margins of the soil column:

$$-\lambda_T \frac{\partial T}{\partial z}\bigg|_{z=0} = R_{net} - H_s - LE_s, \tag{11}$$

$$-\lambda_W \frac{\partial W}{\partial z}\bigg|_{z=0} = r_{pr} - E_s, \tag{12}$$

$$V|_{z=0} = V_0(t), \tag{13}$$

$$\frac{\partial T}{\partial z}\bigg|_{z=H} = \frac{\partial W}{\partial z}\bigg|_{z=H} = \frac{\partial V}{\partial z}\bigg|_{z=H} = 0. \tag{14}$$

Here, R_{net} is the net radiation, H_s and LE_s are sensible and latent heat fluxes in the surface air layer, respectively, L – specific heat of evaporation/condensation, r_{pr} – the liquid water flux to the soil from rain or melted snow. The lower boundary conditions mean zero flux. The infiltration flux, γ is non-zero at the lower bound of soil column, and is assumed to contribute the river runoff. The aforementioned boundary conditions are prone to uncertainties induced by quantifying radiation and heat fluxes, both in measurements and theoretical approaches. Thus, in this study, we make use of a fact that at the diurnal scale, the temperature and moisture remain almost constant at sufficiently large depth, and follow sinusoidal pattern at the soil-atmosphere interface (see Problem Specification section).

1.3. Problem Specification

The key feature of the soil model described above is a set of PDE coefficients, which are dependent on PDE solution. These coefficients are routinely not measured. Our objective is to find coefficients as functions of solution, given the solution is known. The temperature T and moisture W can be available either from observations or from reference direct problem solution with *a priori* given coefficients. In order to simplify the task, we neglect water phase transitions, root uptake, horizontal runoff, which leaves us with a set of two equations – (5), (6) – with $F_i = F_v = R_{roots} = R_{runoff} = 0$:

$$\rho c \frac{\partial T}{\partial t} = \frac{\partial}{\partial z}\left(\lambda_T \frac{\partial T}{\partial z}\right), \tag{15}$$

$$\frac{\partial W}{\partial t} = \frac{\partial}{\partial z}\left(\lambda_W \frac{\partial W}{\partial z}\right) + \frac{\partial \gamma}{\partial z}, \tag{16}$$

and the simplified Dirichlet-type boundary conditions are specified as:

$$f|_{z=0} = f_1 + \frac{1}{2}(f_2 - f_1)\left(1 + \sin(\omega t - \pi/2)\right), \tag{17}$$

$$f|_{z=H} = \frac{1}{2}\left(f_1 + f_2\right), \tag{18}$$

where $f = T,\ W,\ W_1 = W_{min},\ W_2 = W_{max}$, and $T_1 = T_{max},\ T_2 = T_{min}$ ($T_{min} = 0$ K, $T_{max} = 5$ K, $W_{min} = 0.2$ kg/kg, $W_{max} = 0.65$ kg/kg), and ω is a frequency of the surface forcing, in this study set as $\omega = 2\pi/T_d$, $T_d = 24$ h. We substituted the conventional flux-type boundary conditions (11)–(14) with (17)–(18) to avoid additional uncertainties associated to parameterization of heat and moisture turbulent fluxes to the atmosphere (whereas Dirichlet conditions are directly measurable). The initial profiles for temperature and moisture are:

$$f|_{t=0} = f_1 + (1 - \exp(-\lambda_{dec} z))(f_2 - f_1), \tag{19}$$

with $f = T,\ W,\ \lambda_{dec} = 3\ \text{m}^{-1}$.

The coefficients $\lambda_T,\ \lambda_W$ and γ are assumed to be the functions of W only, i.e., no explicit dependence on z is imposed owing to the vertical soil structure. In other words, all soil physical parameters (e.g., soil porosity, heat conductivity of solid particles, hydraulic conductivity at saturation, etc.) potentially affecting thermal conductivity and Richards equation coefficients are taken as depth-independent.

Equations (9) and (10) are no longer a part of the problem, due to neglecting the phase transitions. The depth of soil is set as $H = 1$ m, a depth where diurnal oscillations are typically vanishing in real soils. In this study, the direct problem is solved using the reference ("true") functions describing $\lambda_T,\ \lambda_W, \gamma$ using finite difference scheme and thus providing $T_{h\tau},\ W_{h\tau}$ at a fine grid, with number of layers $n_z = 1000$, and timestep $\Delta t = 10\ s$ or $\Delta t = 1\ h$. Then the reference $\lambda_T,\ \lambda_W, \gamma$ are sought given $T_{h\tau},\ W_{h\tau}$ only. The two methods for seeking the PDE coefficients are used:

- method constructed by joining the well-known elements (referred to hereafter as "classical method"): the functional form of coefficients is given by one of options well-established in soil science (Cote-Konrad [11] or Johansen [24] representations of λ_T, and Mualem-van Genuchten [17], Books-Corey [7] or Gardner [15] formulae for $\lambda_W,\ \gamma$), the parameters of these forms are optimized by stabilized Barzilai–Borwein algorithm [6, 8];
- new artificial neural network (ANN)-based method: the functional form of coefficients $\lambda_T,\ \lambda_W,\ \gamma$ is represented by artificial neural network, and the parameters of ANN are optimized.

In the implementation of ANN method, a special focus is put on the impacts of using FP32 (floating point, 32 bits) and FP16 (floating point, 16 bits) precision on solution wall-clock time and accuracy of inverse problem solution using the Ascend platform provided by Huawei.

In both methods of PDE identification, the parameters of analytical forms for $\lambda_T,\ \lambda_W, \gamma$ are sought to minimize a loss function of PDE residual $\epsilon_{h\tau}$ (see eq. (4)), where operators $B_{h\tau}$ and $A_{h\tau}$ represent discretization of heat conduction (15) and Richards (16) equations. The discretization of heat conduction equation and diffusive part of Richards equation is central-difference in space and 1st order implicit in time. The advective part of equation (16) is approximated by 1st order explicit scheme in time and central-differences in space. Thus, the Richards equation is solved by time-splitting by physical processes (terms to the r.h.s. of (16)).

2. Methods

2.1. Reference Solutions

2.1.1. Heat conduction equation

To produce the reference numerical solution of direct heat conduction problem, we use the Cote-Konrad model for coefficient of soil heat conductivity [11] (hereafter denoted as “C-K”):

$$\lambda_T = \lambda_{T,dry} + (\lambda_{T,sat} - \lambda_{T,dry})\mathrm{Ke}, \tag{20}$$

$$\lambda_{T,dry} = \xi 10^{(-\eta * p)}, \tag{21}$$

$$\lambda_{T,sat} = \lambda_w^p \lambda_{sp}^{1-p}, \tag{22}$$

$$\mathrm{Ke} = (K_{te}\widetilde{W})/(1 + (K_{te} - 1)\widetilde{W}), \tag{23}$$

where $\lambda_{T,dry}$ and $\lambda_{T,sat}$ are the values of conductivity coefficient for dry and saturated soil, respectively, Ke is the so-called Kersten number, p is the soil porosity (taken as 0.496, from the measurements in soil of Meteorological Observatory of Moscow State University [27]), λ_w is heat conductivity of water, λ_{sp} is heat conductivity of soil solid particles (taken as 2.5 W m^{-1} K^{-1}), and the soil-type specific constants are set to be representative of silt and clay: $K_{te} = 1.69$, $\eta = 1.8$, $\xi = 1.7$ W m^{-1} K^{-1}, $\rho c = 2.4{\times}10^6$ J m^{-3} K^{-1}. The Cote-Konrad model is a representative of a class of conductivity models based on normalized heat conductivity concept, where Kersten number is used to interpolate between dry and saturated soil state.

2.1.2. Richards equation

To produce reference numerical solutions of Richards equation (16), we involved widely used functions of the soil moisture developed by Mualem [36] and van Genuchten [17] for the liquid water diffusivity $\lambda_W(W)$ and hydraulic conductivity $\gamma(W)$ (hereafter abbreviated as “M-vG”):

$$\lambda_{W,M-vG}(W) = \frac{\gamma_s(1-m)}{\alpha m(W_s - W_r)}\widetilde{W}^{1/2-1/m}\left[\left(1 - \widetilde{W}^{1/m}\right)^{-m} + \left(1 - \widetilde{W}^{1/m}\right)^{m} - 2\right], \tag{24}$$

$$\gamma_{W,M-vG}(W) = \gamma_s\widetilde{W}^{1/2}\left[1 - \left(1 - \widetilde{W}^{1/m}\right)^{m}\right]^2. \tag{25}$$

These functions monotonically increase with liquid moisture content W, and are zero in the origin ($W = W_r$). The parameters in the above formulae are taken from [27]: $m = 0.272$, $\gamma_s = 3.78D_{sec}^{-1}$ ms^{-1} (hydraulic conductivity of saturated soil), $\alpha = 0.7$ m^{-1}, $W_r = 0.149$, $W_s = 0.7$, where $D_{sec} = 86400$ s is a day duration.

2.2. General Approach of the Study

In this paper, we present two methods for PDE identification: classical method and ANN-based approach. We compare them in terms of wall-clock computational time and accuracy. In both methods, the coefficients of PDEs are sought as those minimizing the PDE residual $\epsilon_{h\tau}$, where the $B_{h\tau}$ and $A_{h\tau}$ operators correspond to central differences in space and 1st order scheme in time.

Key Differences in Classical and ANN-based Methods. The methods presented in this study have a similar structure, differing in the components:

- loss function, which is a measure of ϵ_{hT} deviation from zero in classic method and is a more complex metrics in a case of ANN-based method,
- explicit form of coefficients $\lambda_T,\ \lambda_W, \gamma$ as functions of soil moisture W,
- method for the optimization of parameters in the assumed coefficient functions of W.

In the later sections, the detailed description of classical and ANN-based methods highlight these features.

2.3. Classical Method

2.3.1. Loss function

The loss function in classical method is defined as RMSE of ϵ_{hT} computed in time and space, where the boundary mesh nodes are excluded, i.e.:

$$\mathcal{L}_{CM} = \left[\frac{1}{(n_z-1)n_t} \sum_{i=2}^{n_z} \sum_{j=1}^{n_t} \left(\epsilon_{hT,i}^{j} \right)^2 \right]^{1/2}, \tag{26}$$

with i standing for index of depth level, j denoting the time instant, n_t being the number of timesteps (in classical method, $n_t = 24$, corresponding to $\Delta t = 1$ h).

2.3.2. Assumed form of solution

Having data on the measured evolution of physical variables (soil temperature T and moisture W) we *always* do not know the real functional forms of $\lambda_T(W),\ \lambda_W(W),\ \gamma(W)$, so that *any* functional form which is assumed during inverse problem solution differs from that provided by nature. The indirect corroboration of this statement is provided by existence of ~ 100 pairs of explicit functions $\lambda_W(W),\ \gamma(W)$ reported in literature, none of which performs best on all the empirical datasets. At the same time, it is generally accepted that the heat conduction (15) and Richards (16) equations are exact. In the classical method of PDE identification, the *difference* between real and assumed types of PDE coefficients inevitable in real applications is imitated by using the widespread representations of heat conductivity, liquid water diffusivity and hydraulic conductivity which are *different* from Cote-Konrad and Mualem-van Genuchten formulae, used to compute reference solution, respectively. At the same time, using the functional form of $\lambda_T(W),\ \lambda_W(W),\ \gamma$ in PDE identification algorithm which are *the same* to ones used to produce reference direct problem solution allows to check the correctness of the gradient-descent algorithm in classical method as the latter allows to provide exact inverse problem solution in this case.

For solution of inverse temperature conduction problem we use the Johansen parameterization of heat conductivity [24], which involves the normalized heat conductivity concept (20), and the geometric mean approach for conductivity coefficient of saturated soil (22), however, proposes the different (compared to C-K) formulations for dry soil conductivity and Kersten number (hereafter referred to as “Joh”):

$$\lambda_{T,dry} = \frac{C_1 \rho_{dry} + C_2}{\rho_p - C_3 \rho_{dry}}, \tag{27}$$

$$\mathrm{Ke} = \widetilde{W}, \tag{28}$$

where dry soil density ρ_{dry} and mean soil particles density ρ_p are expressed in g cm^{-3}, and $C_1 = 0.135,\ C_2 = 0.0647,\ C_3 = 0.947$ are empirical constants. The second equation in a set above has been suggested for fine soils.

For identification of the Richards equation (16), the Brooks-Corey parameterization [7] (hereafter referred to as “B-C”):

$$\lambda_{W,B-C}(W) = \frac{\gamma_s |\Psi_{max}| b}{W_s - W_r} \widetilde{W}^{b+2}, \tag{29}$$

$$\gamma_{B-C}(W) = \gamma_s \widetilde{W}^{2b+3}, \tag{30}$$

and the Gardner parameterization [15] (hereafter denoted as “Gard”):

$$\lambda_{W,Gard} = \frac{\gamma_s a^{1/c}}{c} \widetilde{W}^{1/2} W^{\frac{c+1}{c}}, \tag{31}$$

$$\gamma_{Gard} = \gamma_s \widetilde{W}^{1/2} W^{\frac{2c+2}{c}}, \tag{32}$$

are involved, $a,\ b,\ c$ are empirical constants to be calibrated, see details in Section 2.3.3.

2.3.3. Optimization method

As the functional forms of $\lambda_T,\ \lambda_W,\ \gamma$ are defined above, the task of PDE identification is reduced to finding optimal values of parameters in formulas (27)–(28), (24)–(25), (29)–(30), (31)–(32). These parameters are sought using gradient descent method in Barzilai–Borwein modification, having the form [6]:

$$\mathbf{x}^{n+1} = \mathbf{x}^n - \Gamma_n \nabla \mathcal{L}_{CM}(\mathbf{x}^n), \tag{33}$$

$$\Gamma_{n,0} = \frac{|(\mathbf{x}^n - \mathbf{x}^{n-1})^T (\nabla \mathcal{L}_{CM}(\mathbf{x}^n) - \nabla \mathcal{L}_{CM}(\mathbf{x}^{n-1}))|}{||(\nabla \mathcal{L}_{CM}(\mathbf{x}^n) - \nabla \mathcal{L}_{CM}(\mathbf{x}^{n-1}))||^2}. \tag{34}$$

Here, $\mathbf{x}^n$ is a vector of parameters at n-th iteration, vertical index T stands for vector transpose. The gradient of the loss function $\nabla \mathcal{L}(\mathbf{x}^n)$ at each iteration is found by finite-differencing, gradient descent rate Γ_n is set to $\Gamma_{n,0}$ in conventional BB method, and in this study we use the stabilized Barzilai–Borwein method [8], which introduces the following limiting of the descent rate:

$$\Gamma_n = \min(\Gamma_{n,0}, \Gamma_n^{stab}), \tag{35}$$

$$\Gamma_n^{stab} = \frac{\Delta}{||\nabla \mathcal{L}_{CM}(\mathbf{x}^n)||}, \tag{36}$$

where Δ is maximal value allowed for $||\mathbf{x}^{n+1} - \mathbf{x}^n||$. This method is referred to as BBstab, and is free from large “jumps” of parameter vector $\mathbf{x}_n$ during iteration process, which are known to happen in conventional BB algorithm, and were observed in the course of Richards equation identification process in our study. As the physical parameters entering $\mathbf{x}$ have different scales, the equation being identified is rewritten in terms of parameters, normalized by respective maximal values, so that $\mathbf{x}$ is sought in a unit hypercube, and $\Delta \ll 1$.

The loss function $\mathcal{L}_{CM}(\mathbf{x})$ is defined as a RMS of finite-difference equation residual at a given set of test coefficients taken over all time and depth combinations of the mesh (equation (26)).

The parameter vector is dependent on a PDE being identified:

- $\mathbf{x} \doteq (p, \lambda_{sp}, W_r, W_s)$ in C-K case of temperature equation (15),
- $\mathbf{x} \doteq (p, \lambda_{sp}, \rho_p, W_r, W_s, \rho_{dry})$ in Joh case of temperature equation,

- $\mathbf{x} \doteq (m, \gamma_s, \alpha, W_r, W_s)$ in M-vG case of Richards equation (16),
- $\mathbf{x} \doteq (b, \gamma_s, \Psi_{max}, W_r, W_s)$ in B-C case of Richards equation,
- $\mathbf{x} \doteq (c, \gamma_s, a, W_r, W_s)$ in Gard case of Richards equation.

The criterion to stop iterations used in our study is satisfaction of any of the two inequalities: $\mathcal{L}_{CM}$ changes negligibly with iteration number or $n > n_{max}$. The first guess $\mathbf{x_0}$ significantly affects the number of iterations needed to converge to true solution. Moreover, given that some of parameters in λ_W and γ vary between soil types orders of magnitude (e.g., γ_s), the initial guess far from optimal value leads to large error in solution. In the numerical experiments presented below, we use the values $\mathbf{x}_0 = 0.5 * \mathbf{x}_{true},\ \mathbf{x}_1 = 0.6 * \mathbf{x}_{true}$ (Barzilai–Borwein method needs first two points in $\mathbf{x}$-space to start iterations by formula (33)–(34)), which ensured the correct convergence. Alternatively, to compute $\mathbf{x}_1$, one may use (33) with some small Γ_0. Other choices of initial guesses (e.g., $\mathbf{x}_0 = 0.1 * \mathbf{x}_{true},\ \mathbf{x}_1 = 0.2 * \mathbf{x}_{true};\ \mathbf{x}_0 = 1.9 * \mathbf{x}_{true},\ \mathbf{x}_1 = 1.8 * \mathbf{x}_{true}$), have been tested and delivered comparable results in accuracy and number of iterations. In addition to numerical experiments, where all parameters listed above were optimized, a set of simulations were conducted, where all but two parameters were fixed to known reference values, and the these two were optimized only. It was found that $n_{max} = 100$ ensures convergence (i.e., negligible change of $\mathcal{L}_{CM}$ with iteration number) of the algorithm for both PDEs studied in this paper and both types of optimization.

2.4. Neural-Network-Based Solution

2.4.1. PDE identification as a differentiable inverse problem

In this study, we consider PDE identification as an inverse problem of regression type (see Problem Specification section). In both cases of simulated evolution and a solution given by regular observations, we refer to the given evolution as the true one. However, when using the true evolution as a reference, one still cannot apply routine data science approach for approximating the coefficients λ_T, λ_W and γ with ANNs. Instead, in case of ANN-based method, we propose to minimize the error of an integrated evolution within the gradient minimization approach given the coefficients λ_T, or λ_W and γ approximated by ANNs that are essentially differentiable parametric functions of potentially unlimited expressive power. More precisely, we propose minimizing PDE residual or, equivalently, the error of the tendencies $\frac{\partial T}{\partial t}$ (in case of heat diffusion equation) and $\frac{\partial W}{\partial t}$ (in case of Richards equation). The tendencies are estimated using finite-differences approximation of the equations (5) and (6). Finite-differencing is a differentiable operation as well as loss function characterizing errors of tendency estimates. Thus, there is a straight-forward way for the computation of the gradients of the loss function w.r.t. parameters of the functions modeling λ_T, or λ_W and γ. In Fig. 1, we present our ANN-based approach in a form of data-flow and operations scheme for Richards equation as a demonstrative example. Due to similarity of the derivatives inference, optimization procedures and regularizations applied in cases of heat diffusion equation and Richards equation, we further use X as a substitute correspondingly for W or T (e.g., see Fig. 1). We also present the formulae considering Richards equation case with λ_W and γ terms further in this section. At the same time, the derivatives inference, the regularizations and optimization procedure are the same for λ_T in case of heat diffusion equation.

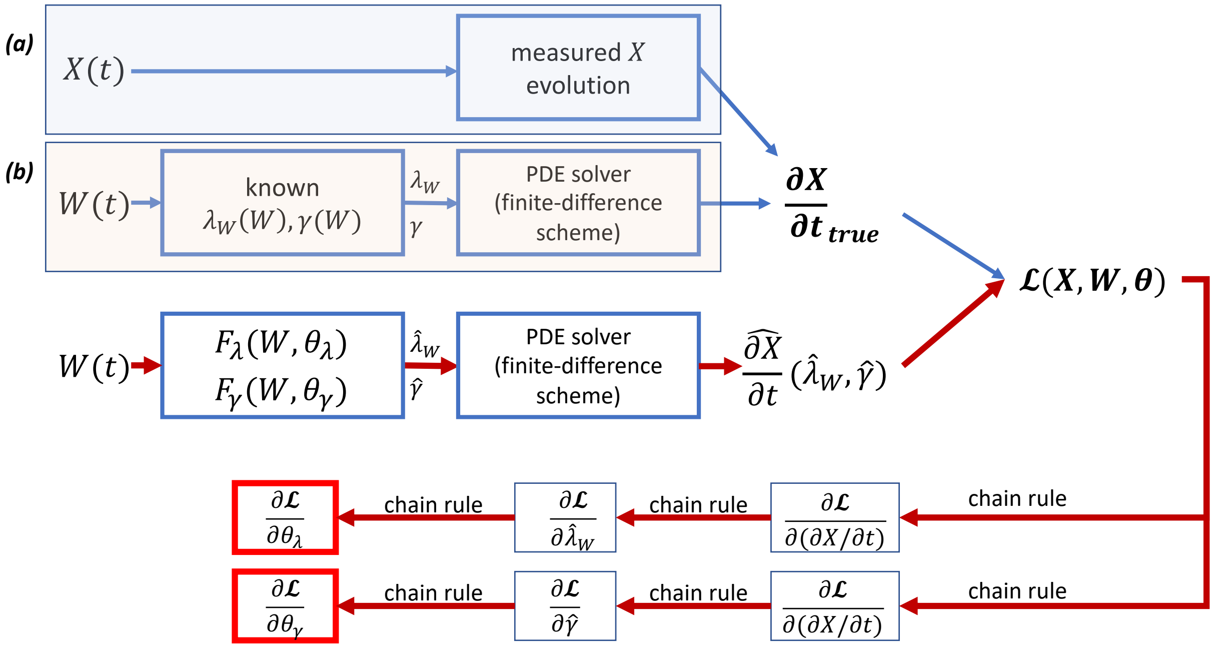

(a)
$X(t)$
measured X evolution
(b)
$W(t)$
known $\lambda_W(W), \gamma(W)$
λ_W
γ
PDE solver (finite-difference scheme)
$\frac{\partial X}{\partial t}_{true}$
$\mathcal{L}(X, W, \theta)$
$W(t)$
$F_\lambda(W, \theta_\lambda)$
$F_\gamma(W, \theta_\gamma)$
$\hat{\lambda}_W$
$\hat{\gamma}$
PDE solver (finite-difference scheme)
$\widehat{\frac{\partial X}{\partial t}}(\hat{\lambda}_W, \hat{\gamma})$
$\frac{\partial \mathcal{L}}{\partial \theta_\lambda}$
chain rule
$\frac{\partial \mathcal{L}}{\partial \hat{\lambda}_W}$
chain rule
$\frac{\partial \mathcal{L}}{\partial(\partial X/\partial t)}$
chain rule
$\frac{\partial \mathcal{L}}{\partial \theta_\gamma}$
chain rule
$\frac{\partial \mathcal{L}}{\partial \hat{\gamma}}$
chain rule
$\frac{\partial \mathcal{L}}{\partial(\partial X/\partial t)}$
chain rule

Figure 1. The PDE identification approach using ANNs F_λ and F_γ for Richards equation (16) in case of either (a) measured evolution, or (b) modeling the reference evolution using known coefficients λ_W and γ. Here X is a substitute for W, the similar algorithm is applied for temperature equation ($X = T$). The bold red arrows indicate the computational graph that involves chain rule (a.k.a. backpropagation) for evaluating the gradients of loss function $\mathcal{L}$ w.r.t. the parameters θ_λ of F_λ and θ_γ of F_γ. In this paper, the case (b) is considered only

2.4.2. Assumed form of solution

In this method, the approximations of the coefficients λ_T, λ_W and γ of the heat conduction (15) and Richards equations (16) are represented by artificial neural networks (ANNs) denoted further as either F_{NN,λ_T}, F_{NN,λ_W} and $F_{NN,\gamma}$ or F_{λ_T}, F_{λ_W} and F_γ (for the sake of clarity). The architecture of the ANNs is the same for all the cases. It is the fully-connected ANNs (a.k.a. multilayer perceptron) with six layers. The layers are wide for the networks to have high enough expressive power for approximating functions like eqs. (20), (24) and (25). We optimized the structural hyperparameters of the networks ensuring the capability of approximating the target functions by solving simple supervised problem with target values generated by “true” forms like eqs. (20), (24) and (25). In order to improve convergence, we used *Mish* activation function [34] for all layers except the last one, where we used linear activation.

2.4.3. Loss function

The basic part of the loss function in this study is a weighted sum of the three components: MAPE of tendency, MSE (mean square error) and MAE (mean absolute error) of PDE residual:

$$\mathcal{L} = \alpha_{mse}\frac{1}{N}\sum \epsilon_{h\tau}^2 + \alpha_{mape}\frac{1}{N}\sum \left| \frac{\widehat{(\partial X/\partial t)}_{h\tau} - (\partial X/\partial t)_{h\tau,true}}{(\partial X/\partial t)_{h\tau,true} + \epsilon} \right| + \alpha_{mae}\frac{1}{N}\sum |\epsilon_{h\tau}|, \quad (37)$$

where N is a number of mesh nodes over which the PDE residual and discrepancy between test and reference tendencies are summed, X stands for T or W, depending on equation being identified, the subscript “$h\tau$” indicates variables discretized on the mesh, the hat $\widehat{(...)}$ denotes values calculated using test PDE coefficients represented by ANNs, and $\epsilon = 10^{-8}$ is a number added for numerical stability. Note that $(\widehat{\partial X/\partial t})_{h\tau} - (\partial X/\partial t)_{h\tau,true} = \epsilon_{h\tau}$.

Zero-Point Regularization. In order to stabilize the ANN training and improve the convergence, we complemented the loss function (37) with the regularization terms informed by physical constraints. First, one may note that dry soil should not conduct liquid water. This is in agreement with the reference solutions for λ_W and γ given by eqs. (24) and (25). Thus, $\lambda_W\left(W=0\right)=0$, and $\gamma\left(W=0\right)=0$. We implemented the respective regularization loss terms as:

$$\mathcal{L}_{reg,zp} = \alpha_{zp,\lambda} \times \left|F_{NN,\lambda_W}\left(W=0\right)\right|, \tag{38}$$

$$\mathcal{L}_{reg,zp} = \alpha_{zp,\gamma} \times \left|F_{NN,\gamma}\left(W=0\right)\right|, \tag{39}$$

where $\alpha_{zp,\lambda}$ and $\alpha_{zp,\gamma}$ are regularization coefficients related to the zero-point values of λ_W and γ approximated by the ANNs $F_{NN,\lambda_W}\left(W\right)$ and $F_{NN,\gamma}\left(W\right)$, correspondingly.

Known Points Regularization. In a real-world application case, one may improve the convergence with a regularization encouraging the ANNs to predict specific known (e.g., measured) values of PDE coefficients for a set of argument values X_{kp}. We call the respective loss term the known-points regularization:

$$\mathcal{L}_{reg,kp} = \alpha_{kp,\lambda} \times \sum_i \left|F_{NN,\lambda_W}\left(W_{kp,i}\right) - \lambda_W\left(W_{kp,i}\right)\right|, \tag{40}$$

$$\mathcal{L}_{reg,kp} = \alpha_{kp,\gamma} \times \sum_i \left|F_{NN,\gamma}\left(W_{kp,i}\right) - \gamma\left(W_{kp,i}\right)\right|, \tag{41}$$

where $\alpha_{kp,\lambda}$ and $\alpha_{kp,\gamma}$ are regularization coefficients related to the values of λ_W and γ correspondingly in a limited set of known points $W_{kp,i}$ approximated by the neural networks $F_{NN,\lambda_W}\left(W\right)$ and $F_{NN,\gamma}\left(W\right)$; i enumerates measured tuples of W_{kp}, λ_W and γ. In our study, the number of known points is 3.

Derivative Penalty. One may also note that the coefficients λ_W and γ are strictly increasing functions, which is a case for all analytical representations (including Mualem-van Genuchten, Brooks-Corey and Gardner) for physical reasons. According to this constraint, we introduce corresponding physics-guided regularization terms that we call derivative penalty:

$$\mathcal{L}_{reg,dp} = \alpha_{dp,\lambda} \times \sum_{\frac{\partial F_{NN,\lambda_W}(W)}{\partial W}<0} \left|\frac{\partial F_{NN,\lambda_W}\left(W\right)}{\partial W}\right|, \tag{42}$$

$$\mathcal{L}_{reg,dp} = \alpha_{dp,\gamma} \times \sum_{\frac{\partial F_{NN,\gamma}(W)}{\partial W}<0} \left|\frac{\partial F_{NN,\gamma}\left(W\right)}{\partial W}\right|. \tag{43}$$

The regularizations presented in eqs. (38)–(43) are “soft”, meaning they do not necessarily deliver the desired properties of the approximations. Thus, one may note that the derivative penalties in eqs. (42) and (43) do not necessarily guarantee strictly ascending ANN-approximations for λ_W and γ. In order to additionally encourage the ascending approximations, we complement the loss function with one more term which encourages the networks to output less value at the left margin of the X range compared to the output at the right margin of X range:

$$\mathcal{L}_{reg,lr} = \left\{ \begin{array}{cl} \alpha_{lr,\lambda} \times \left|F_{NN,\lambda_W}\left(X_{max}\right) - F_{NN,\lambda_W}\left(X_{min}\right)\right| & \text{if } F_{NN,\lambda_W}\left(X_{max}\right) < F_{NN,\lambda_W}\left(X_{min}\right) \\ 0 & \text{otherwise} \end{array} \right. \tag{44}$$

$$\mathcal{L}_{reg,lr} = \left\{ \begin{array}{cl} \alpha_{lr,\gamma} \times \left|F_{NN,\gamma}\left(X_{max}\right) - F_{NN,\gamma}\left(X_{min}\right)\right| & \text{if } F_{NN,\gamma}\left(X_{max}\right) < F_{NN,\gamma}\left(X_{min}\right) \\ 0 & \text{otherwise,} \end{array} \right. \tag{45}$$

where X_{min} and X_{max} are the lower and higher limits for X that are either inferred from measurements or set by a researcher conducting a computational experiment. In case of X being substitute for W, one may set the lower limit to 0 and higher one to the maximum value inferred from data. In case of X being substitute for T, both lower and higher limits may be set to the values inferred from data. Here one needs to pay attention to the way we inject regularization terms into loss function. In particular, one should pay attention to derivative penalty $\mathcal{L}_{reg,dp}$, since gradients of this loss term w.r.t. network parameters includes second-order derivatives $\frac{\partial F_{NN}(W,\theta)}{\partial W \partial \theta}$, which is computationally expensive. In order to save computational power, we do not compute the derivative penalty using the W and T profiles themselves since the number of data points in this case is `batch_size*`n_z. Instead, we compute derivative penalty using additional sample of T and W values distributed uniformly in the range $[T_{min}, T_{max}]$ and $[W_{min}, W_{max}]$ correspondingly. The same applies to the regularization terms $\mathcal{L}_{reg,zp}$, $\mathcal{L}_{reg,kp}$. In this case, the number of data points is managed by a researcher. We set this number to n_z. Here, T_{min} and T_{max} are the maximum and minimum values of temperature in the limited set of measured known data points; W_{min} and W_{max} – the same for liquid water content W. This way, we managed to improve the performance approximately 6-8 times compared to that when using computations with full training set of W and T profiles.

Rescaling Neural Networks Output. ANNs are known to be strongly overparameterized and thus having too high expressive power for representing functions of relatively simple form. In the case of limited support of target value distribution, one may need to restrict the distribution of the output of the network. One way to do so is using a finite-bounds activation function for the output layer, e.g., tanh or sigmoid. These functions are known to saturate which may cause optimization difficulties and low approximation accuracy in some argument ranges. In our study, we use dynamic min-max rescaling of the network output for both λ_W and γ networks:

$$\zeta = F_{NN}\left(X\right), \tag{46}$$

$$\tilde{\zeta} = \zeta_{min,true} + \frac{\zeta - \zeta_{min}}{\zeta_{max} - \zeta_{min}} \times \left(\zeta_{max,true} - \zeta_{min,true}\right), \tag{47}$$

where ζ is the substitute for λ_W or γ; $\tilde{\zeta}$ is the substitute for λ_W or γ rescaled using min-max scheme; $\zeta_{min,true}$ and $\zeta_{max,true}$ are the minimum and maximum values of true λ_W or γ in the $[X_{min}, X_{max}]$ range; ζ_{min} and ζ_{max} are the minimum and maximum values of ANN-based approximations for λ_W or γ in the $[X_{min}, X_{max}]$ range. One may note that the values ζ_{min} and ζ_{max} may not be the ones $F_{NN}\left(X_{min}\right)$ and $F_{NN}\left(X_{max}\right)$ due to inaccurate meeting of the strictly-ascending requirement by the neural network. On the contrary, here we consider once again the known-points values for $\zeta_{min,true}$ and $\zeta_{max,true}$ to be the $\zeta_{X_{min},true}$ and $\zeta_{X_{max},true}$. These values are guaranteed to be minimum and maximum ones due to strictly ascending λ_W and γ.

Initial Guess. There is a well-known issue related to the choice of initial guess for the ANN parameters θ_{λ_T}, θ_{λ_W} and θ_γ of the networks F_{NN,λ_T}, F_{NN,λ_W} and $F_{NN,\gamma}$. One may even choose an initialization that would prevent a network from training. In [33], the importance of proper initialization is demonstrated. In this paper, the authors demonstrated that one needs to initialize deep ANNs with ReLU activations using random variables drawn from a uniform or normal distribution with parameters dependent on layers' sizes. This is done for the variance of activations of hidden layers and the output layer to remain nearly constant, which prevents output and gradients variances from vanishing or exploding. The most successful initialization at the moment is Kaiming He [21]. However, the so-called "gain" parameter in Kaiming He initialization [21] is computed using the size of layers' weight tensors with the assumption of either ReLU of Leaky ReLU nonlinearity. In our study, we use *Mish* activation functions, thus, there may be a need for a derivation of proper gain value. We did not conduct this research. Instead, we assumed that *Mish* has the effect on the distributions of activations similar to ReLU due to their large-scale similarity. Thus, in our study, we used Kaiming He initialization with gain value similar to the one computed with ReLU nonlinearity assumption. The results of our ANN-based method presented in Section 3.2 confirm the validity of this decision.

Optimization of Hyperparameters. In this study, we demonstrate that the balance between the multipliers standing before the loss function terms is crucial for the stability of training, for Richards equation (16) identification. In order to stabilize the training, we explored the space of hyperparameters.

Learning Rate Schedule. In our study, we exploited stochastic gradient-based optimization procedure, which implies iterative increment of network's parameters with adjusted loss gradients multiplied by a small factor namely learning rate. In recent studies, the importance of learning rate scheduling was shown [31]. However, we found that with the stability improvements presented further in this section, there is no need to apply sophisticated learning rate schedules. Thus, we employ exponential decay of learning rate. Typically, the starting leaning rate is 10^{-4}, and the decay rate is 10^{-2}. Thus, to the end of the training, learning rate decreases to 10^{-6} level.

2.4.4. Optimization and implementation details

In case of ANN-based PDE identification of our study, in the numerically-simulated evolution, the number of z-levels is $n_z = 1000$, and the number of time steps is $n_t = 8640$ ($\Delta t = 10\ s$) which corresponds to 24 hours of modeled time. The optimization of ANNs implies stochastic optimization algorithms (in this study, we employ Adam [26]). For this algorithm, to evaluate loss function, we implemented data sampler, which draws `batch_size` time steps from the reference PDE solution dataset.

In our study, we found that Adam optimization is not enough for the stable loss function minimization and for reaching the solution with low uncertainty in case of Richards equation problem. Inverse PDE problems are known for sensitivity of the solution to input data. One more issue compromising the solution stability is the mixed precision computations which requires particular attention in case of small and large loss values and gradient values. There are known approaches of loss scaling and corresponding gradient inverse scaling by NVIDIA [1]. However, we faced the issue of partly implemented method of NVIDIA loss scaling in Tensorflow 1.15, which fails to run in case of provided software and hardware stack (see Section 2.5). In order to achieve stable and converging training of ANNs for λ_W and γ approximations, we implemented several additional approaches described below.

Scaling Scheme for Scalars and Tensors. Optimization of ANNs using mixed precision floating-point calculations is complicated by the narrow range of values represented in FP16. General description of computational issues related to FP16 is presented on NVIDIA web-site. The NVIDIA loss scaling [1] is implemented in detail in Tensorflow 2.x, and is *partially* implemented in Tensorflow 1.15. On Huawei Ascend platform, Tensorflow 1.15 was more matured at the time that the study was conducted, thus, we developed a new scheme for loss scaling which may be characterized as more “gentle” yet efficient. The same scheme may be employed for gradient scaling in a case the tensors are found vanishing.

The main goal of the scheme is to preserve the scale of loss function values close the order of 10^0. To do so, we introduce the following variables:

1. a factor for exponential moving average f_{ema}. We typically set it to 0.1;
2. loss which is subject to scale L. This loss value is the one we compute directly using networks outputs with equations (37)–(45);
3. current loss scale s;
4. scaled loss L_s;
5. $\log_2(L_s)$, denoted as l_2;
6. exponentially moving averaged value of the loss L_{ema};
7. exponentially moving averaged value of $\log_2(L)$, denoted as $l_{2,ema}$;
8. gradients g_s computed using backpropagation implemented in auto-differentiation schemes of Tensorflow or other autodiff frameworks. These gradients are calculated from scaled loss values L_s;
9. inverse-scaled gradients g.

The operation of loss scaling and gradients inverse scaling are linear operations:

$$L_s = L \times s, \tag{48}$$

$$g_s = \nabla_\theta L_s, \tag{49}$$

$$g = g_s / s, \tag{50}$$

where θ is a set of parameters (a.k.a. weights) of ANNs involved in loss function computations, and ∇_θ is a gradient over this parameter set.

The operations for scale s adjustment are the following:

$$L_{ema}^{(\tau)} = L_{ema}^{(\tau-1)} \times (1 - f_{ema}) + L^{(\tau)} \times f_{ema}, \tag{51}$$

$$l_2^{(\tau)} = \log_2 L_{ema}^{(\tau)}, \tag{52}$$

$$l_{2,ema}^{(\tau)} = l_{2,ema}^{(\tau-1)} \times (1 - f_{ema}) + l_2^{(\tau)} \times f_{ema}, \tag{53}$$

$$s^{(\tau)} = 2^{-\lfloor l_{2,ema}^{(\tau)} \rceil}, \tag{54}$$

where $\lfloor x \rceil$ denotes rounding to a nearest integer; τ is the number of iteration of adjustment (typically coincides with the number of optimization iteration). With this scheme, the loss scale remains to be of order 10^0.

With the same scheme, one may apply gradient scaling for its expected absolute value to be of order 10^0. To do so, the only corrections to the scheme presented above are the following: instead of L, one needs to use the scale of $\tilde{L} = \text{mean}\,|g_\theta|$, where g is gradient tensor of a neural network w.r.t. parameters tensor of a particular layer of a network. In case of some other variable of interest A subjected to scaling, the scheme looks the following:

$$\begin{aligned} A_s &= A \times s_A, && (55)\\ \tilde{A}^{(\tau)}_{ema} &= A^{(\tau-1)}_{ema} \times (1 - f_{ema}) + A^{(\tau)} \times f_{ema}, && (56)\\ a^{(\tau)}_2 &= \log_2 A^{(\tau)}_{ema}, && (57)\\ a^{(\tau)}_{2,ema} &= a^{(\tau-1)}_{2,ema} \times (1 - f_{ema}) + a^{(\tau)}_2 \times f_{ema}, && (58)\\ s^{(\tau)}_A &= 2^{-\lfloor l^{(\tau)}_{2,ema} \rceil}, && (59) \end{aligned}$$

where $\tilde{A}$ is sample mean of $|A|$ in case of A being a tensor with rank greater than zero, and $\tilde{A} = A$ in case of A being a scalar. This way, we scale the gradients and values of $(\partial X/\partial t)_{h\tau}$ preventing them from becoming small enough to vanish to FP16-zero level in case of quadratic loss components $\text{MSE}\left[\widehat{(\partial X/\partial t)}_{h\tau}, (\partial X/\partial t)_{h\tau,true}\right]$.

The gradient scaling scheme has an impact on the optimization procedure. First, using this scheme, one introduces noise into stochastic optimization when the scale s is changing, since the change is made in a discrete multiplicative manner (see eq. (54)). Second, when one applies gradient scaling presented in this study, the magnitude of gradients is preserved of order 10^0. In common gradient optimization algorithms, the gradients' magnitudes are assumed to decrease over time, resulting in sampling of the loss landscape with increasing resolution and, thus, finding the solution with increasing accuracy. This behavior is essentially equivalent to learning rate decrease, which does not take place in case gradient scaling applied. Thus, one needs to pay additional attention to setting the learning rate schedule, which may play the role of a factor increasing the resolution of loss landscape sampling.

Reducing the Interaction between Loss Sum Components. There is one more issue of the ANN-based identification of the Richards equation related to smooth nature of γ function. The finite-difference approximation of the gravitational infiltration term in the Richards equation (16) is:

$$\begin{gathered} \left(\frac{\partial W}{\partial t}\right)_{\gamma,i} = \left(\frac{\partial \gamma(W)}{\partial z}\right)_i \approx \frac{\gamma(W_{i+1/2}) - \gamma(W_{i-1/2})}{\Delta z_i},\\ \text{MSE}\left[\widehat{\left(\frac{\partial W}{\partial t}\right)}_{\gamma,h\tau}, \left(\frac{\partial W}{\partial t}\right)_{\gamma,h\tau,true}\right] = \frac{1}{N\Delta z}\sum_{i=1}^{N}\left[\widehat{\left(\frac{\partial W}{\partial t}\right)}_{\gamma,h\tau,i} - \left(\frac{\partial W}{\partial t}\right)_{\gamma,h\tau,true,i}\right]^2,\\ \text{MAE}\left[\widehat{\left(\frac{\partial W}{\partial t}\right)}_{\gamma,h\tau}, \left(\frac{\partial W}{\partial t}\right)_{\gamma,h\tau,true}\right] = \frac{1}{N\Delta z}\sum_{i=1}^{N}\left|\widehat{\left(\frac{\partial W}{\partial t}\right)}_{\gamma,h\tau,i} - \left(\frac{\partial W}{\partial t}\right)_{\gamma,h\tau,true,i}\right|,\\ \text{MAPE}\left[\widehat{\left(\frac{\partial W}{\partial t}\right)}_{\gamma,h\tau}, \left(\frac{\partial W}{\partial t}\right)_{\gamma,h\tau,true}\right] = \frac{1}{N\Delta z}\sum_{i=1}^{N}\left|\frac{\widehat{\left(\frac{\partial W}{\partial t}\right)}_{\gamma,h\tau,i} - \left(\frac{\partial W}{\partial t}\right)_{\gamma,h\tau,true,i}}{\left(\frac{\partial W}{\partial t}\right)_{\gamma,h\tau,true,i}}\right|. \end{gathered} \tag{60}$$

Here, we replaced residual by difference of tendencies, i.e., $\epsilon_{h\tau} = \widehat{(\partial W/\partial t)}_{h\tau} - (\partial W/\partial t)_{h\tau,true}$ and Δz_i with Δz for the uniform z mesh, i enumerates individual components of loss sums. One may note that the consequent terms of MSE, MAE and MAPE sums interact between each other due to smoothness of the γ function (see details in Appendix A). We also found empirically that the gradients of these loss function terms are zero almost everywhere except for small subset of depths including the boundaries at $z = 0$ and $z = H$ (Fig. 2a).

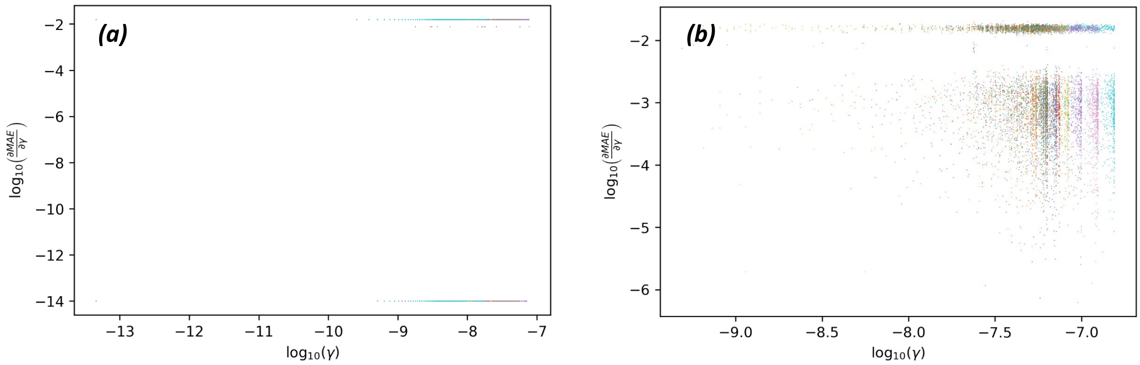


Figure 2. MAE_γ loss gradients w.r.t. γ as individual sum terms vs. γ: (a) in case of standard MAE_γ formulation; (b) in case of random re-weighting of individual MAE_γ sum terms according to eq. (61). The logarithmic scale uses additive log-stability term 10^{-14}

In order to alleviate this issue, we introduce the technique of random re-weighting of loss sum components. We reformulate the loss function terms the following way:

$$
\begin{aligned}
\mathcal{L}_{\text{MSE}} &= \frac{1}{N\Delta z}\sum_{i=1}^{N} \mathcal{W}_{\text{MSE},i} \times \left[\widehat{\left(\frac{\partial W}{\partial t}\right)}_{\gamma,h\tau,i} - \left(\frac{\partial W}{\partial t}\right)_{\gamma,h\tau,true,i}\right]^2, \\
\mathcal{L}_{\text{MAE}} &= \frac{1}{N\Delta z}\sum_{i=1}^{N} \mathcal{W}_{\text{MAE},i} \times \left|\widehat{\left(\frac{\partial W}{\partial t}\right)}_{\gamma,h\tau,i} - \left(\frac{\partial W}{\partial t}\right)_{\gamma,h\tau,true,i}\right|, \\
\mathcal{L}_{\text{MAPE}} &= \frac{1}{N\Delta z}\sum_{i=1}^{N} \mathcal{W}_{\text{MAPE},i} \times \left|\frac{\widehat{\left(\frac{\partial W}{\partial t}\right)}_{\gamma,h\tau,i} - \left(\frac{\partial W}{\partial t}\right)_{\gamma,h\tau,true,i}}{\left(\frac{\partial W}{\partial t}\right)_{\gamma,h\tau,true,i}}\right|.
\end{aligned}
\tag{61}
$$

Here, we generate the weights as Gaussian random variable: $\mathcal{W} \sim \mathcal{N}\left(1, \sigma^2\right)$, where σ may be optimized during hyperparameters optimization stage. In our study, we set $\sigma = 0,1$. As a result, the loss gradients became more informative and are not zero, as shown in Fig. 2b.

2.4.5. Further details of the ANN-based solution for Richards equation

We found that λ_W-network converges much faster compared to the one approximating γ due to the strongly interacting loss sum components in the latter case described in the previous section. Moreover, the λ_W-network converges to a very good approximation with $\text{MAPE}\left(\lambda_W\right) \approx 10^{-2}$. Thus, in this study, we used the approach of consequent training of γ- and λ_W-networks. First, we trained λ_W-network with the γ-network initialized to output zeros. Then, after a fixed training period, we freeze the λ_W-network and optimize the one approximating γ.

2.5. Hardware and Software

Numerical experiments with classical method of PDE identification were performed on MacBook Pro, 2.3 GHz Intel Core i5, 8 Gb RAM, 2133 MHz, LPDDR3.

Computations using ANN-based approach were conducted on Atlas 800 Training Server (Model 9010) with 8-socket Huawei Ascend 910 NPUs, 16 Huawei 64-bit TaiShan CPU cores at 2.4 GHz. Each NPU provides three HCCS links and a maximum bandwidth of 90 Gbit/s. The server supports two CPUs with up to 3.8 GHz frequency, 38.5 MB L3 cache, and three 10.4 GT/s UPI links, up to to 24 DDR4 RDIMMs, 16 GB, 32 GB or 64 GB per DIMM. The capacity of HBM is 32 GB with bandwidth 1228 Gbit/s.

In this study, we developed the software package implementing our ANN-based method using Python 3.5.7 [41] with Tensorflow 1.15 (within the `session.run()` approach and explicit computational graph definition) [2]. We also used numpy [20] and other python modules typical for data science data processing and visualization.

2.6. Metrics for Quality of the Inverse Problem Solution

The values of coefficients λ_W and γ range of orders of magnitude depending on the argument W, as well as corresponding tendency of the solution $\partial W/\partial t$ in Richards equation (16). Thus, in order to represent the accuracy of the inverse problem solution, we use several quality measures: $\mathrm{RMSE}\,(\lambda_X)$; $\mathrm{MAPE}\,(\lambda_X)$; $\mathrm{RMSE}\,(\gamma)$; $\mathrm{MAPE}\,(\gamma)$. We calculate these measures using conventional $\mathrm{RMSE}\,(\cdot)$ and $\mathrm{MAPE}\,(\cdot)$ formulations without any re-weighting. We also use the following error metrics: $\mathrm{RMSE}\,((\partial X/\partial t)_{h\tau})$ which is equivalent to root mean squared $\epsilon_{h\tau}$, and $\mathrm{MAPE}\,((\partial X/\partial t)_{h\tau})$ $(X = W, T)$:

$$\mathcal{Q}_{rmse,\epsilon_{h\tau}} = \mathrm{RMSE}\left[\left(\frac{\partial X}{\partial t}\right)_{h\tau}\right] = \sqrt{\frac{1}{N}\sum_{1}^{N}\epsilon_{h\tau}^2}, \tag{62}$$

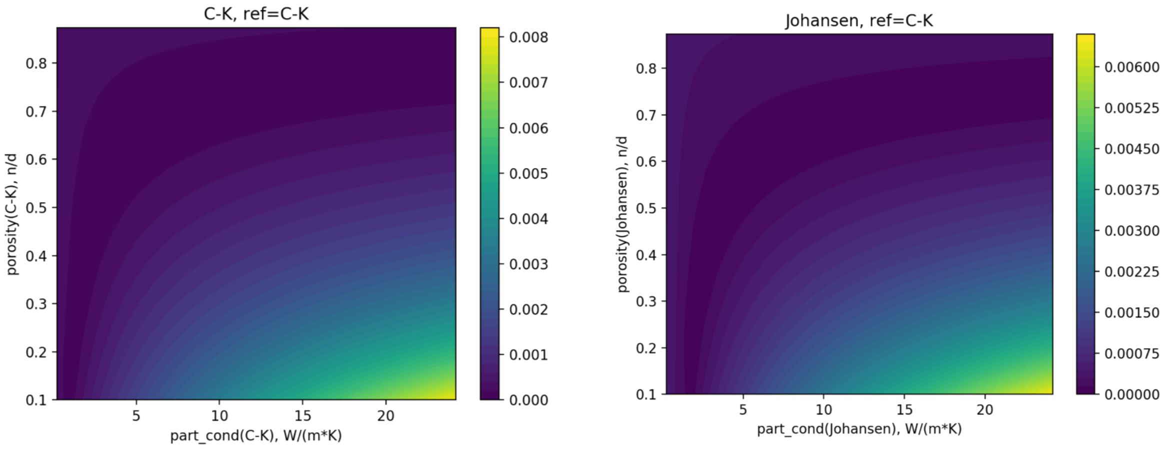
$$\mathcal{Q}_{mape,\epsilon_{h\tau}} = \mathrm{MAPE}\left[\left(\frac{\partial X}{\partial t}\right)_{h\tau}\right] = \frac{1}{N}\sum_{1}^{N}\left|\frac{\epsilon_{h\tau}}{\left(\frac{\partial X}{\partial t}\right)_{h\tau,true} + \epsilon}\right|. \tag{63}$$

3. Results and Discussion

3.1. Stabilized Barzilai–Borwein Method

First, we consider the loss function $\mathcal{L}_{CM}$ of two parameters, where the rest parameters are fixed to reference values, and where $\mathcal{L}_{CM}$ is computed at a regular grid 30×30 (Fig. 3 and 4). The range of parameters covers most of variability reported in soil science. Note the single minimum of loss function for each of three functional forms of test coefficients. The minimum in parameter space of reference C-K and M-vG form delivers exact solution of the inverse problem. However, in a case, where all (from 4 to 6) parameters are optimized, the minimum of $\mathcal{L}_{CM}$ is no more unique (including a scenario of inverse problem solution, where C-K and M-vG forms are used), which is manifested by the fact that the argument of loss minimum found by BBstab method does depend on initial guess (not presented in this paper).

As a result of multiple minima of $\mathcal{L}_{CM}$ in parameter space, optimal coefficient functions, provided by all analytical forms, significantly differ from the reference ones (Fig. 5ab, 6a–c, 7a–c). The thermal conductivity coefficient remains the same order of magnitude over the W range, and the absolute error does the same. Using the Joh formulation provides significantly larger errors compared to a case of using C-K representation. For Richards equation (16), the optimal test coefficients significantly deviate from reference ones as well, with absolute error increasing towards W values close to saturation. Surprisingly, seeking Richards coefficients in M-vG way, does not necessarily provide the smallest MAPE and RMSE respective to "true" coefficients, compared to other scenarios (e.g., compare B-C and M-vG results for γ), which means the choice of initial M-vG parameter values close to exact ones is crucial for solution accuracy.



(a) The case of Cote-Konrad (C-K) formulation

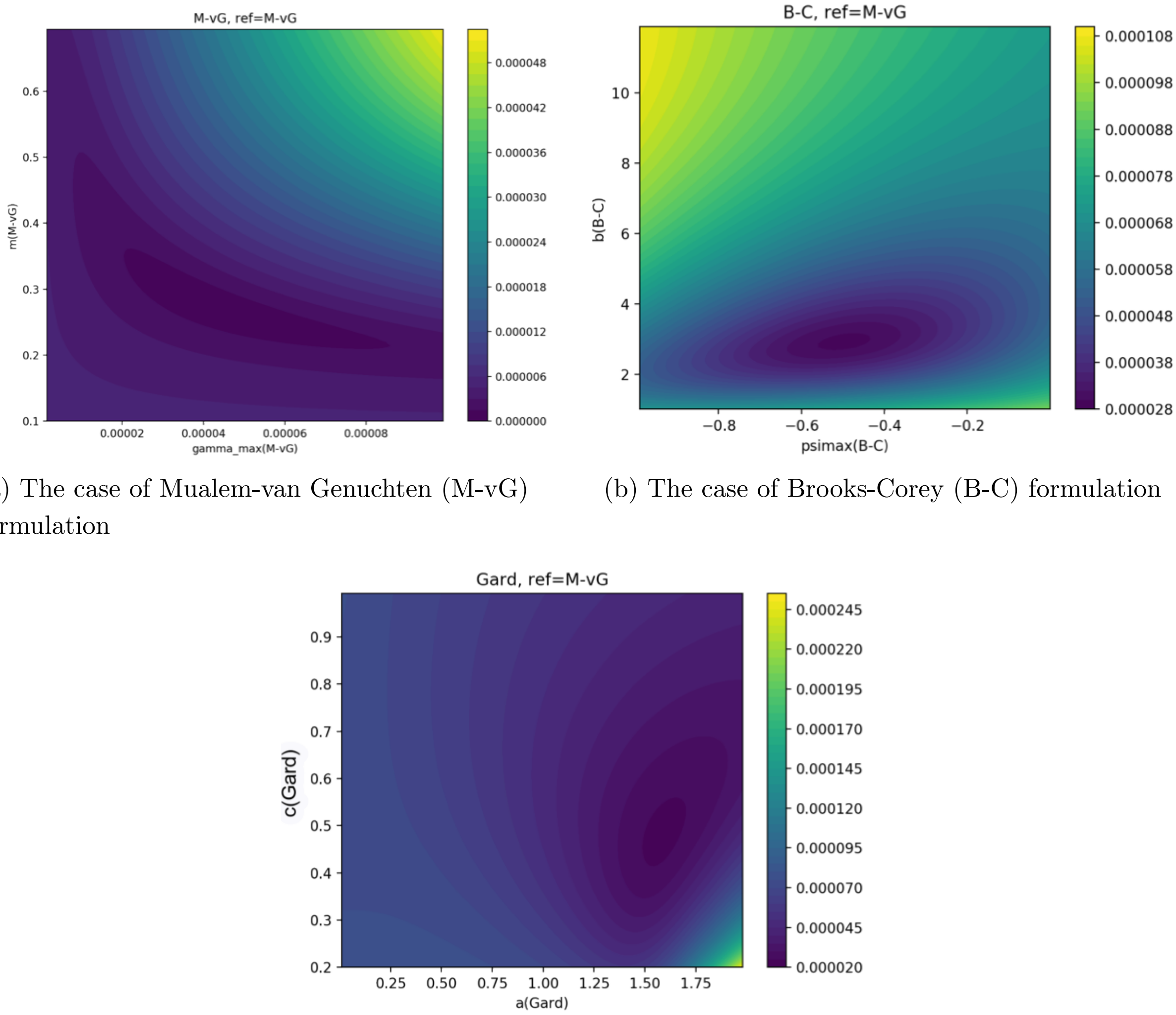
(b) The case of Johansen (Johansen) formulation

Figure 3. Loss function $\mathcal{L}_{CM}$ of two parameters (porosity, p, and heat conductivity of soil particles, λ_{sp}) being optimized in the cases: Cote-Konrad (C-K, left) and Johansen (Johansen, right) formulas for thermal conductivity in test evaluation of thermal conductivity equation (15) residual $\epsilon_{h\tau}$, the realistic reference solution T of PDE being produced using C-K formulation

3.2. Neural-Network Method

Here, we present results of the optimization of ANNs within the framework described in Section 2.4, for the Richards (16) and temperature conductance (15) equations with realistic reference direct problem solution. We use soil column temperature and moisture profiles generated by direct solution of PDEs for 24 hours of model time. This way, we obtained 8640 profiles. For each of these profiles we apply our scheme described in Section 2.4. We use stochastic optimization with `batch_size=128` number of profiles per iteration, each containing $n_z = 1000$ number of z-levels.

We optimized some of the hyperparameters of the method using Optuna framework [3]. To optimize the hyperparameters, we compute quality metrics of the solutions with λ_T in case of heat conductance equation or λ_W, γ in case of Richards equation modeled by ANNs for each set of hyperparameters; the learning rate schedule was set to exponential decay with starting value `1e-4`, decay rate `1e-2` and number of steps `8192`. We found that in this setup, the ANNs optimization can converge with reasonable quality, thus we have the opportunity to assess the quality metrics for each set of hyperparameters. We performed 3500 runs optimizing the following hyperparameters: $\alpha_{mse}, \alpha_{mape}, \alpha_{mae}, \alpha_{zp}, \alpha_{dp}, \alpha_{kp}$. A sample for each of these hyperparameters was generated using log-uniform sampling with the distribution support $[1, 10^{14}]$. As a result, we found the following hyperparameters that let us train our networks reaching appropriate quality $\mathcal{Q}_{rmse,\epsilon_{h\tau}}$ and $\mathcal{Q}_{mape,\epsilon_{h\tau}}$: $\alpha_{mse} = 10^{11}, \alpha_{mape} = 10, \alpha_{mae} = 1, \alpha_{zp} = 10^8, \alpha_{dp} = 10, \alpha_{kp} = 10^8$ (see (37), (38), (41), (42)). With these hyperparameters, we then trained the networks F_{NN,λ_W} and $F_{NN,\gamma}$ in case of Richards equation (16), and F_{NN,λ_T} in case of heat conductance problem (15).



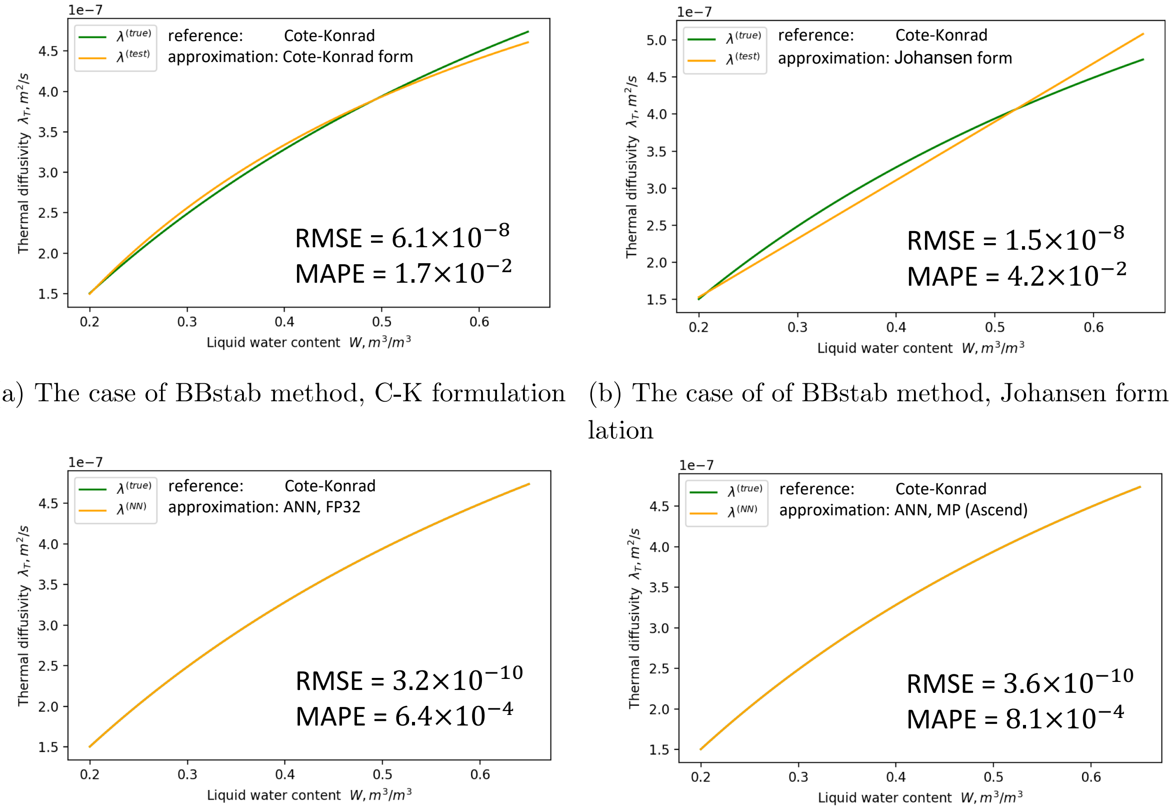
(a) The case of Mualem-van Genuchten (M-vG) formulation

(b) The case of Brooks-Corey (B-C) formulation

(c) The case of Gardner (Gard) formulation

Figure 4. Loss function $\mathcal{L}_{CM}$ of two parameters being optimized in the cases: Mualem-van Genuchten (M-vG, (a), parameters are γ_{max}, m), Brooks-Corey (B-C, (b), parameters are Ψ_{max}, b) and Gardner (Gard, (c), parameters are a, c) formulas for moisture diffusivity and hydraulic conductivity coefficients used in test evaluation of Richards equation (16) residual $\epsilon_{h\tau}$, the realistic reference PDE solution W being produced using M-vG formulation

Important part of our study was focused on computations in mixed precision (MP). We observed that the out-of-box MP settings resulted in the drop of accuracy comparing to single precision, but we managed to stabilize those using the black-list of operations which were prohibited to run in FP16: `"Exp"`, `"Tanh"`, `"TanhGrad"`, `"Div"`, `"Sub"`, `"Pow"`, `"Add"`, `"Mul"`, `"Log"`. With this black-list and the stability improvements described in detail in Section 2.4, we managed to reach the quality of the approximation similar to FP32 computations (see results in Tab. 1 and 2) without negative impact to the computational efficiency. We also tested the computational efficiency (performance) in FP32 and mixed-precision. The quantitative results for heat conductance problem are presented in Tab. 3. Corresponding measures for Richards equation case are presented in Tab. 2 and 4. One may note, that in Tab. 4, we show the performance estimates in iterations per second. Here, one iteration corresponds to 128000 profile elements per second since the number of moisture profiles is exactly 128 per batch (iteration), and number of z-levels is 1000 for ANN-based experiments in this study.



(a) The case of BBstab method, C-K formulation

(b) The case of of BBstab method, Johansen formulation

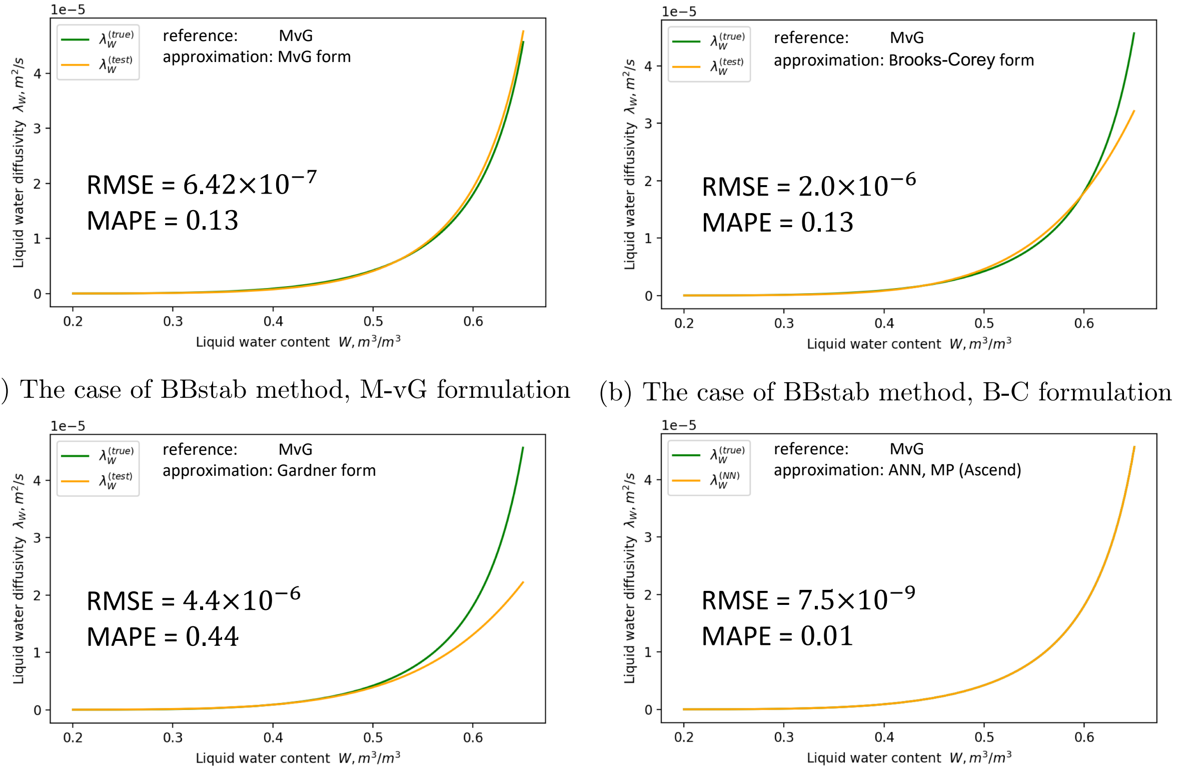
(c) The case of ANN, FP32 precision

(d) The case of ANN, mixed precision (MP)

Figure 5. Thermal diffusivity coefficient λ_T for soil as a function of volumetric moisture content W being approximated (a) in a form given by C-K formulae in test solution of heat conduction equation using BBstab optimization procedure; (b) in a form given by Johansen formula in test solution of heat conduction equation using BBstab optimization procedure; (c) in a form of artificial neural network, FP32 precision computations; (d) in a form of artificial neural network, MP computations employed on Ascend compute platform. Here, we produce the reference solution using the C-K formulation

The results of PDE identification using ANN-based method are shown in Fig. 5cd, 6d, 7d. One may see excellent correspondence between the reference forms of the coefficients and the approximations made by ANNs.

We found that ANNs optimization process may begin to diverge if one continues to run it long enough in case of Richards equation problem. Thus, there is a room for decision at which point to get the snapshots of the networks and corresponding quality measures. In case of real-world application, one cannot compute quality measures for the parameters λ_T, λ_W and γ due to unavailability of their ground truth. Thus, one cannot formulate the stopping rule for ANNs optimization based on $\mathrm{RMSE}\left(\lambda_{\mathrm{W}}\right)$, $\mathrm{MAPE}\left(\lambda_{\mathrm{W}}\right)$, $\mathrm{RMSE}\left(\gamma\right)$ or $\mathrm{MAPE}\left(\gamma\right)$. The only indicators one may use to judge on solution accuracy during optimization process are the measures of discrepancy between true and approximated PDE tendencies $\mathcal{Q}_{rmse,\epsilon_{h\tau}}$ or $\mathcal{Q}_{mape,\epsilon_{h\tau}}$. In our study, we used $\mathcal{Q}_{rmse,\epsilon_{h\tau}}$ as the indicator for the stopping rule, thus, in eqs. (1) and (2) we demonstrate the quality measures corresponding to optimal (minimal) $\mathcal{Q}_{rmse,\epsilon_{h\tau}}$



(a) The case of BBstab method, M-vG formulation (b) The case of BBstab method, B-C formulation

(c) The case of BBstab method, Gard formulation (d) The case of ANN, mixed precision (MP)

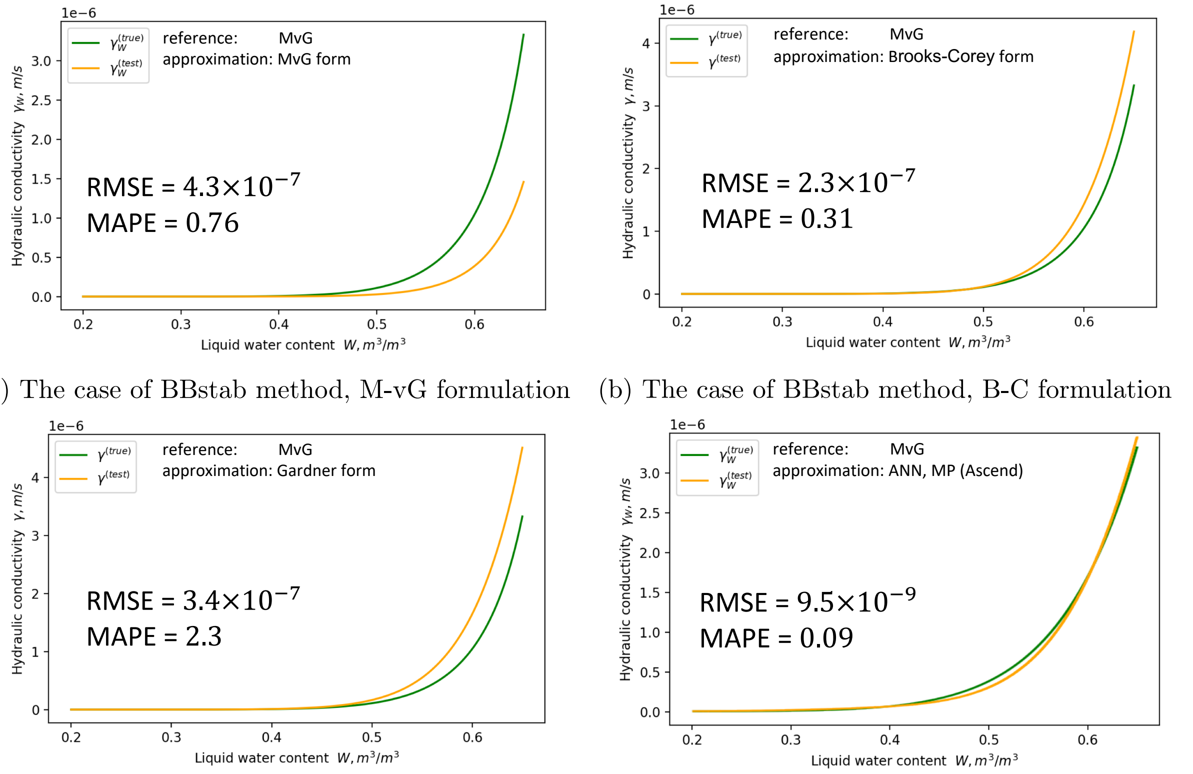
Figure 6. Diffusion coefficient for soil moisture λ_W as a function of volumetric moisture content W being approximated given forms of (a) M-vG, (b) Brooks-Corey (B-C) and (c) Gardner (Gard) formulae for moisture diffusivity λ_W and hydraulic conductivity γ coefficients in test solution of Richards equation (16), BBstab optimization procedure applied. In (d), we present diffusion coefficient for soil moisture being approximated given artificial neural networks for moisture diffusivity λ_W and hydraulic conductivity γ coefficients in test solution of Richards equation (16), MP computations employed on Ascend compute platform. Here, we produce the reference solution using the M-vG formulation

Table 1. Quality measures of the temperature conductance equation (15) identification. The ANN-based solutions are obtained using Ascend compute platform

method;	MAPE $(\lambda_T/\rho c)$	RMSE $(\lambda_T/\rho c)$	$\mathcal{Q}_{mape,\epsilon_{h_\tau}}$	$\mathcal{Q}_{rmse,\epsilon_{h_\tau}}$
FP32/MP	-	m^{2}s^{-1}	-	K s^{-1}
ANN, MP	8.1×10^{-4}	3.6×10^{-10}	7.04×10^{-3}	1.9×10^{-6}
ANN, FP32	6.4×10^{-4}	3.2×10^{-10}	4.1×10^{-4}	5.2×10^{-7}
BBstab(C-K), FP32	1.7×10^{-2}	6.1×10^{-9}	3.6×10^{-2}	1.6×10^{-6}
BBstab(Joh), FP32	4.2×10^{-2}	1.5×10^{-8}	8.7×10^{-2}	4.1×10^{-6}

3.3. Relative Performance of ML-Based Method vs. Classical PDE Identification Method

Tables 1–2 and 3–4 allow the direct comparison of BBstab and ANN methods in terms of error metrics and computational efficiency, respectively. Though being faster in wall-clock time, BBstab method significantly underperforms in accuracy. In MAPE and RMSE for temperature



(a) The case of BBstab method, M-vG formulation

(b) The case of BBstab method, B-C formulation

(c) The case of BBstab method, Gard formulation

(d) The case of ANN, mixed precision (MP)

Figure 7. Hydraulic conductivity coefficient for soil moisture γ as a function of volumetric moisture content W being approximated given forms of (a) M-vG, (b) Brooks-Corey (B-C) and (c) Gardner (Gard) formulae for moisture diffusivity λ_W and hydraulic conductivity γ coefficients in test solution of Richards equation (16), BBstab optimization procedure applied. In (d), we present hydraulic conductivity coefficient for soil moisture being approximated given artificial neural networks for moisture diffusivity λ_W and hydraulic conductivity γ coefficients in test solution of Richards equation (16), MP computations employed on Ascend compute platform. Here, we produce the reference solution using the M-vG formulation

Table 2. Quality measures of the Richards equation (16) identification. The ANN-based solutions are obtained using Ascend compute platform

method;	MAPE (λ_W)	MAPE (γ)	RMSE (λ_W)	RMSE (γ)	$\mathcal{Q}_{mape,\epsilon_{h\tau}}$	$\mathcal{Q}_{rmse,\epsilon_{h\tau}}$
FP32/MP	-	-	m^{2}s^{-1}	m s^{-1}	-	s^{-1}
ANN MP	1.01×10^{-2}	8.9×10^{-2}	7.5×10^{-9}	9.5×10^{-9}	5.8×10^{-1}	6.1×10^{-8}
BBstab(M-vG) FP32	1.3×10^{-1}	7.6×10^{-1}	6.4×10^{-7}	4.3×10^{-7}	1.2×10^{0}	1.3×10^{-6}
BBstab(B-C) FP32	1.3×10^{-1}	3.1×10^{-1}	2.0×10^{-6}	2.3×10^{-7}	1.2×10^{0}	1.2×10^{-6}
BBstab(Gard) FP32	4.4×10^{-1}	2.3×10^{0}	4.4×10^{-6}	3.4×10^{-7}	8.1×10^{-1}	3.1×10^{-6}

conductivity, liquid water diffusivity, and hydraulic conductivity, BBstab demonstrates error values 1–3 orders of magnitude higher than those obtained with ANN method.

Table 3. Performance estimates of the temperature conductance equation (15) identification. The ANN-based solutions were obtained using Ascend compute platform

method	FP32/MP	perf., it/sec	T_c, iter	T_c, w.time
ANN	MP	8.68	11800	2086 s.
BBstab(C-K)	FP32	9.1×10^{-2}	100	1096 s.
BBstab(Joh)	FP32	9.1×10^{-2}	100	1053 s.

Table 4. Performance estimates of the Richards equation (16) identification. The ANN-based solutions were obtained using Ascend compute platform

method	FP32/MP	perf., it/sec	T_c, iter	T_c, w.time
ANN	MP	9.1	15507	2500 s.
BBstab(M-vG)	FP32	1×10^{-1}	100	994 s.
BBstab(B-C)	FP32	1.2×10^{-1}	100	843 s.
BBstab(Gard)	FP32	1.1×10^{-1}	100	939 s.

3.4. Relation to Results of Other Groups

An idea to reconstruct the Richards equation from measured data on soil moisture has got development during recent years. In [5], the ANN with monotonicity constraints were used to approximate coefficients of the Richards equation of the same form as used in our study. The reference solutions were produced by HYDRUS-1D model with Mualem-van Genuchten formulation for PDE coefficients. The results of inverse problem solution are thoroughly analyzed from the point of soil physics, however, no comparison with the baseline PDE identification method (like BBstab in our case) is provided and the visual inspection of figures in the cited paper allows to conclude that in our study the ANN-based algorithm produces solution with much higher accuracy. In [18], the ANNs are involved to recover the linear Richards equation containing additional terms of higher-order derivatives. In contrast, our approach sticks to classic nonlinear form of Richards equation, having solid physical basis. In both mentioned papers, the input data for identification problem was synthetic, produced by direct problem solution with parameters and/or boundary conditions, instructed from real measurements; the same strategy applied in our work.

Conclusions

In this study, we propose a novel method based on ANNs for the identification of partial differential equations. We demonstrated its efficacy and high accuracy in a case of diffusion equation applied to the problem of heat conduction in soil, and nonlinear diffusion-advection equation (Richards equation) applied to the soil moisture simulation. We also propose physics-guided and other types of regularizations for the stabilization of neural networks training. We found that in case of the identification of advection term, one may face the issue of strong interaction between loss sums elements in MSE, MAE and MAPE, which may cause uninformative gradients and, in turn, poor identification quality and high uncertainty in approximated PDE coefficient. To alleviate this issue, we elaborated random re-weighting of the individual components of these terms' sums, which resulted in informative gradients, stabilized training and good approximation of the PDE coefficient. The usage of novel AI Ascend platform allows to significantly speedup implementation of ANN-based algorithm.

Regarding the quality of the PDE identification, we draw the following conclusions:

- novel method based on ANNs for the identification of PDE coefficients describing heat and moisture transport in soil implemented on Ascend platform using mixed precision floating point operations overperforms the classical gradient descent method in Barzilai–Borwein stabilized (BBstab) modification, in terms of MAPE and/or RMSE at least an order of magnitude;
- BBstab method requires good initial guess of parameters being optimized to converge to true solution, whereas, for ANN method, the sensitivity of training results to initial guess is much less limiting the accuracy of solution.

The ANN-based method we developed for PDE identification may be used in research areas other than soil thermodynamics. The Richards equation is a diffusion-advection type equation, with highly nonlinear advective term. We expect applicability of suggested approach to hydrodynamic-type problems, e.g., developing turbulence closures, where the reference solutions of PDEs are usually obtained from high-resolution direct Navier-Stokes simulations. Moreover, our method of ANN-based identification is not limited to the cases where a PDE solver is implemented in a form of finite-difference approximation, it can be used in conjunction with any other differentiable numerical PDE solvers.

Acknowledgements

The work is partially supported by the Russian Ministry of Science and Higher Education, project No. 075-15-2019-1621 (the general statement of the inverse problem), by Research and Educational School of Moscow State University "Brain, cognitive systems, artificial intelligence" (development of the neural-network based algorithm), and by the Russian Science Foundation, grant no. 21-71-30003 (creation and application of new high-performance algorithms for the development of Earth system models, using novel computing architectures).

References

1. Mixed precision training, `http://docs.nvidia.com/deeplearning/frameworks/mixed-precision-training/index.html`

2. Abadi, M., Agarwal, A., Barham, P., *et al.*: TensorFlow: Large-scale machine learning on heterogeneous systems (2015), `http://tensorflow.org/`

3. Akiba, T., Sano, S., Yanase, T., *et al.*: Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD International Conference

on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019. pp. 2623–2631. ACM (2019). https://doi.org/10.1145/3292500.3330701

4. Baek, M., DiMaio, F., Anishchenko, I., *et al.*: Accurate prediction of protein structures and interactions using a three-track neural network. Science 373(6557), 871–876 (2021). https://doi.org/10.1126/science.abj8754

5. Bandai, T., Ghezzehei, T.A.: Physics-Informed Neural Networks With Monotonicity Constraints for Richardson-Richards Equation: Estimation of Constitutive Relationships and Soil Water Flux Density From Volumetric Water Content Measurements. Water Resources Research 57(2) (feb 2021). https://doi.org/10.1029/2020WR027642

6. Barzilai, J., Borwein, J.M.: Two-Point Step Size Gradient Methods. IMA Journal of Numerical Analysis 8(1), 141–148 (1988). https://doi.org/10.1093/imanum/8.1.141

7. Brooks, R., Corey, A.: Hydraulic Properties of Porous Media. Tech. rep., Colorado State University, Fort Collins (1964)

8. Burdakov, O., Dai, Y.H., Huang, N.: Stabilized Barzilai-Borwein method. Journal of Computational Mathematics 37(6), 916–936 (2019). https://doi.org/10.4208/jcm.1911-m2019-0171

9. Burdine, N.: Relative Permeability Calculations From Pore Size Distribution Data. Journal of Petroleum Technology 5(03), 71–78 (mar 1953). https://doi.org/10.2118/225-G

10. Camporeale, E., Wilkie, G.J., Drozdov, A., Bortnik, J.: Machine-learning based discovery of missing physical processes in radiation belt modeling (2021)

11. Côté, J., Konrad, J.M.: A generalized thermal conductivity model for soils and construction materials. Canadian Geotechnical Journal 42(2), 443–458 (apr 2005). https://doi.org/10.1139/t04-106

12. Du, C.: Comparison of the performance of 22 models describing soil water retention curves from saturation to oven dryness. Vadose Zone Journal 19(1) (jan 2020). https://doi.org/10.1002/vzj2.20072

13. Fadeev, R.Y., Ushakov, K.V., Tolstykh, M.A., Ibrayev, R.A.: Design and development of the SLAV-INMIO-CICE coupled model for seasonal prediction and climate research. Russian Journal of Numerical Analysis and Mathematical Modelling 33(6), 333–340 (dec 2018). https://doi.org/10.1515/rnam-2018-0028

14. Fuentes, C., Chávez, C., Brambila, F.: Relating Hydraulic Conductivity Curve to Soil-Water Retention Curve Using a Fractal Model. Mathematics 8(12), 2201 (dec 2020). https://doi.org/10.3390/math8122201

15. Gardner, W.R.: Field Measurement of Soil Water Diffusivity. Soil Science Society of America Journal 34(5), 832 (1970). https://doi.org/10.2136/sssaj1970.03615995003400050045x

16. Gasmi, C.F., Tchelepi, H.: Physics informed deep learning for flow and transport in porous media (2021)

17. van Genuchten, M.T.: A Closed-form Equation for Predicting the Hydraulic Conductivity of Unsaturated Soils. Soil Science Society of America Journal 44(5), 892–898 (sep 1980). https://doi.org/10.2136/sssaj1980.03615995004400050002x

18. Ghorbani, A., Sadeghi, M., Jones, S.B.: Towards new soil water flow equations using physics-constrained machine learning. Vadose Zone Journal 20(4) (jul 2021). https://doi.org/10.1002/vzj2.20136

19. Haghighat, E., Raissi, M., Moure, A., *et al.*: A deep learning framework for solution and discovery in solid mechanics: linear elasticity. CoRR abs/2003.02751 (2020), https://arxiv.org/abs/2003.02751

20. Harris, C.R., Millman, K.J., van der Walt, S.J., *et al.*: Array programming with NumPy. Nature 585(7825), 357–362 (Sep 2020). https://doi.org/10.1038/s41586-020-2649-2

21. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (December 2015), https://www.cv-foundation.org/openaccess/content_iccv_2015/html/He_Delving_Deep_into_ICCV_2015_paper.html

22. Jagtap, A., Karniadakis, G.: Extended Physics-Informed Neural Networks (XPINNs): A Generalized Space-Time Domain Decomposition Based Deep Learning Framework for Nonlinear Partial Differential Equations. Communications in Computational Physics 28, 2002–2041 (11 2020). https://doi.org/10.4208/cicp.OA-2020-0164

23. Jia, W., Wang, H., Chen, M., *et al.*: Pushing the limit of molecular dynamics with ab initio accuracy to 100 million atoms with machine learning. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. SC '20, IEEE Press (2020). https://doi.org/10.5555/3433701.3433707

24. Johansen, O.: Thermal conductivity of soils. Ph.D. thesis, University of Trondheim (1975)

25. Jumper, J., Evans, R., Pritzel, A., *et al.*: Highly accurate protein structure prediction with alphafold. Nature (2021), https://doi.org/10.1038/s41586-021-03819-2

26. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Bengio, Y., LeCun, Y. (eds.) 3rd Int. Conf. on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015), http://arxiv.org/abs/1412.6980

27. Kokoreva, A.A., Dembovetskiy, A.V., Ezhelev, Z.S., *et al.*: Simulating water transport in porous media of urban soil. IOP Conference Series: Earth and Environmental Science 862(1), 012042 (oct 2021). https://doi.org/10.1088/1755-1315/862/1/012042

28. Kurth, T., Treichler, S., Romero, J., *et al.*: Exascale deep learning for climate analytics. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis, SC 2018, Dallas, TX, USA, November 11-16, 2018. pp. 51:1–51:12. IEEE / ACM (2018). https://doi.org/10.5555/3291656.3291724

29. Li, Z., Kovachki, N.B., Azizzadenesheli, K., *et al.*: Fourier neural operator for parametric partial differential equations. CoRR abs/2010.08895 (2020), https://arxiv.org/abs/2010.08895

30. Li, Z., Kovachki, N.B., Azizzadenesheli, K., *et al.*: Fourier neural operator for parametric partial differential equations. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net (2021), `https://openreview.net/forum?id=c8P9NQVtmnO`

31. Loshchilov, I., Hutter, F.: SGDR: stochastic gradient descent with warm restarts. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net (2017), `https://openreview.net/forum?id=Skq89Scxx`

32. Lu, L., Jin, P., Karniadakis, G.E.: DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. CoRR abs/1910.03193 (2019), `http://arxiv.org/abs/1910.03193`

33. Mishkin, D., Matas, J.: All you need is a good init. In: 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings (2016), `http://arxiv.org/abs/1511.06422`

34. Misra, D.: Mish: A self regularized non-monotonic activation function (2020), `http://arxiv.org/abs/1908.08681`

35. Mortikov, E.V., Glazunov, A.V., Lykosov, V.N.: Numerical study of plane Couette flow: turbulence statistics and the structure of pressure–strain correlations. Russian Journal of Numerical Analysis and Mathematical Modelling 34(2), 119–132 (apr 2019). `https://doi.org/10.1515/rnam-2019-0010`

36. Mualem, Y.: A new model for predicting the hydraulic conductivity of unsaturated porous media. Water Resources Research 12(3), 513–522 (jun 1976). `https://doi.org/10.1029/WR012i003p00513`

37. Pfau, D., Spencer, J.S., Matthews, A.G.D.G., Foulkes, W.M.C.: Ab initio solution of the many-electron Schrödinger equation with deep neural networks. Physical Review Research 2(3) (Sep 2020). `https://doi.org/10.1103/physrevresearch.2.033429`

38. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics 378, 686–707 (2019). `https://doi.org/https://doi.org/10.1016/j.jcp.2018.10.045`

39. Sitzmann, V., Martel, J.N.P., Bergman, A.W., *et al.*: Implicit neural representations with periodic activation functions. CoRR abs/2006.09661 (2020), `https://arxiv.org/abs/2006.09661`

40. Tancik, M., Srinivasan, P.P., Mildenhall, B., *et al.*: Fourier features let networks learn high frequency functions in low dimensional domains. CoRR abs/2006.10739 (2020), `https://arxiv.org/abs/2006.10739`

41. Van Rossum, G., Drake, F.L.: Python 3 Reference Manual. CreateSpace, Scotts Valley, CA (2009)

42. Volodin, E.M., Gritsun, A.S.: Simulation of Possible Future Climate Changes in the 21st Century in the INM-CM5 Climate Model. Izvestiya, Atmospheric and Oceanic Physics 56(3), 218–228 (may 2020). https://doi.org/10.1134/S0001433820030123

43. Weyn, J.A., Durran, D.R., Caruana, R.: Improving data-driven global weather prediction using deep convolutional neural networks on a cubed sphere. Journal of Advances in Modeling Earth Systems 12(9) (Sep 2020). https://doi.org/10.1029/2020ms002109

44. Wiecha, P.R., Arbouet, A., Girard, C., Muskens, O.L.: Deep learning in nano-photonics: inverse design and beyond. Photonics Research 9(5), B182 (Apr 2021). https://doi.org/10.1364/prj.415960

45. Ziogas, A.N., Ben-Nun, T., Fernández, G.I., *et al.*: A data-centric approach to extreme-scale ab initio dissipative quantum transport simulations. Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (Nov 2019). https://doi.org/10.1145/3295500.3357156

Appendix A. On the Interaction of the Loss Sum Elements in Approximated Advection Term

In Section 2.4.4, we stated that in Richards equation case, there is an issue of loss sum elements strongly interacting with each other resulting in uninformative gradients. In particular, this feature takes place when one considers the advective term $\frac{\partial\gamma}{\partial z}$ of the r.h.s. of Richards equation within our approximation approach. In this section, we present the derivation of the gradients of loss sums in eq. (60) w.r.t. γ in order to demonstrate either their reduction to zero or to come constant independent of ground truth, in case of smooth function $\gamma\left(W\right)$ and fine z-grid.

Let us consider the loss function of a tendency $\frac{\partial W}{\partial t}$ concerning the term $\frac{\partial\gamma}{\partial z}$:

$$\mathcal{L} = \mathrm{Err}\left(\left.\frac{\partial W}{\partial t}\right|_{\gamma}\right) = \mathrm{Err}\left(\frac{\partial\gamma}{\partial z}\right), \tag{64}$$

where Err is either MSE, MAE or MAPE.

A.1. MSE Term

First, we consider MSE $\left(\frac{\partial\gamma}{\partial z}\right)$ (see the details for MAE and MAPE further).

$$\mathcal{L} = \sum_{i=1}^{N}\left(\left.\frac{\partial\gamma}{\partial z}\right|_{NN,i} - \left.\frac{\partial\gamma}{\partial z}\right|_{true,i}\right)^2, \tag{65}$$

where i enumerates z-elements of a soil profile.

In our scheme, the r.h.s. of Richards equation is approximated using finite differences, thus, one may express the loss sum the following way:

$$\mathcal{L} = \sum_{i=1}^{N}\left(\left.\frac{\Delta\gamma}{\Delta z}\right|_{NN,i} - \left.\frac{\Delta\gamma}{\Delta z}\right|_{true,i}\right)^2, \tag{66}$$

In our method, z-grid is uniform, thus, all the Δz_i are equal: $\Delta z_i = \delta_z, \forall i$. Let us denote $\mathcal{L}'$ the finite-difference approximated loss term multiplied by δ_z^2: $\mathcal{L}' = \delta_z^2 \mathcal{L}$. Then this term is expressed the following way:

$$\mathcal{L}' = \sum_{i=1}^{N} \left(\Delta\gamma|_{NN,i} - \Delta\gamma|_{true,i} \right)^2, \tag{67}$$

Let us denote $\nu = \gamma_{NN}$, and $\tau = \gamma_{true}$ for the convenience and brevity. Consider the expansion of the sum in eq. (67):

$$\begin{aligned}\mathcal{L}' =& \sum_{i=2}^{N} \left(\nu_i - \nu_{i-1} - \tau_i + \tau_{i-1}\right)^2 = \\ =& \ldots + \left(\nu_j - \nu_{j-1} - \tau_j + \tau_{j-1}\right)^2 + \left(\nu_{j+1} - \nu_j - \tau_{j+1} + \tau_j\right)^2 + \ldots = \\ =& \ldots + \nu_j^2 + \nu_{j-1}^2 + \tau_j^2 + \tau_{j-1}^2 - \\ & - 2\nu_j\nu_{j-1} - 2\nu_j\tau_j + 2\nu_j\tau_{j-1} + 2\nu_{j-1}\tau_j - 2\nu_{j-1}\tau_{j-1} - 2\tau_j\tau_{j-1} + \\ & + \nu_{j+1}^2 + \nu_j^2 + \tau_{j+1}^2 + \tau_j^2 - \\ & - 2\nu_{j+1}\nu_j - 2\nu_{j+1}\tau_{j+1} + 2\nu_{j+1}\tau_j + 2\nu_j\tau_{j+1} - 2\nu_j\tau_j - 2\tau_{j+1}\tau_j + \ldots\end{aligned} \tag{68}$$

Here, j is an index of some arbitrary sum element in the expansion. When one would like to compute the gradients of the loss term $\mathcal{L}$ w.r.t. parameters θ_γ of the neural network F_γ approximating γ coefficient, the chain rule is applied in the following form:

$$\frac{\partial \mathcal{L}}{\partial \theta_\gamma} = \frac{1}{\delta_z^2}\frac{\partial \mathcal{L}'}{\partial \theta_\gamma} = \frac{1}{\delta_z^2}\frac{\partial \mathcal{L}'}{\partial \gamma}\frac{\partial \gamma}{\partial \theta_\gamma} = \frac{1}{\delta_z^2}\sum_{i=1}^{N}\frac{\partial \mathcal{L}'}{\partial \nu_i}\frac{\partial \nu_i}{\partial \theta_\gamma}.$$

Since the most of the expanded sum terms in eq. (68) are not dependent of ν_j, the finite-difference approximated gradient of the loss $\mathcal{L}'$ w.r.t. γ for an arbitrary j-th element ν_j is expressed the following way:

$$\begin{aligned}\left.\frac{\partial \mathcal{L}'}{\partial \gamma}\right|_j =& \frac{\partial \mathcal{L}'}{\partial \nu_j} = \\ =& \frac{\partial}{\partial \nu_j}\left(2\nu_j^2 - 2\nu_j\nu_{j-1} - 2\nu_j\tau_j + 2\nu_j\tau_{j-1} - 2\nu_j\nu_{j+1} + 2\nu_j\tau_{j+1} - 2\nu_j\tau_j\right) = \\ =& 2\nu_j^2 + 2\nu_j\left(-\nu_{j-1} - \tau_j + \tau_{j-1} - \nu_{j+1} + \tau_{j+1} - \tau_j\right).\end{aligned} \tag{69}$$

Since the function γ is supposed to be smooth, and also the z grid has 1000 levels with fine resolution, one may observe the following approximate equalities:

$$\begin{aligned}\nu_{j+1} + \nu_{j-1} \approx 2\nu_j, \\ \tau_{j-1} + \tau_{j+1} \approx 2\tau_j.\end{aligned} \tag{70}$$

Thus, the gradient in eq. (69) is transformed the following way:

$$\frac{\partial \mathcal{L}'}{\partial \nu_j} = \frac{\partial}{\partial \nu_j}\left(2\nu_j^2 + 2\nu_j\left(-2\nu_j + 2\tau_i - 2\tau_j\right)\right) \approx -4\nu_j. \tag{71}$$

Thus, in case of smooth γ function and fine z-grid, one may clearly see that the gradients of MSE loss of the advective term do not depend on external data γ_{true}, and, thus, are uninformative.

A.2. MAE Term

In this section, we consider MAE $\left(\frac{\partial\gamma}{\partial z}\right)$.

$$\mathcal{L} = \sum_{i=1}^{N} \left| \left.\frac{\partial\gamma}{\partial z}\right|_{NN,i} - \left.\frac{\partial\gamma}{\partial z}\right|_{true,i} \right|, \tag{72}$$

where i enumerates z-elements of a soil profile. Similar to the derivation of MSE term, we use ν and τ notation for γ_{NN} and γ_{true} correspondingly. We also use the following notation here: $\mathcal{L}' = \delta_z\mathcal{L}$. Thus, the gradient of the MAE loss term is expressed the following way:

$$\frac{\partial\mathcal{L}}{\partial\theta_\gamma} = \frac{1}{\delta_z}\frac{\partial\mathcal{L}'}{\partial\theta_\gamma} = \frac{1}{\delta_z}\frac{\partial\mathcal{L}'}{\partial\gamma}\frac{\partial\gamma}{\partial\theta_\gamma} = \frac{1}{\delta_z}\sum_{i=1}^{N}\frac{\partial\mathcal{L}'}{\partial\nu_i}\frac{\partial\nu_i}{\partial\theta_\gamma}.$$

Let us consider the gradient of the expansion of the $\mathcal{L}'$ term w.r.t. some arbitrary sum elements ν_j in case of finite-difference estimated derivatives $\frac{\partial\gamma}{\partial z}$:

$$\begin{aligned}\frac{\partial\mathcal{L}'}{\partial\nu_j} =& \frac{\partial}{\partial\nu_j}\sum_{i=2}^{N}|\nu_i - \nu_{i-1} - \tau_i + \tau_{i-1}| = \\ =& \frac{\partial}{\partial\nu_j}\left(\ldots + s_i\left(\nu_j - \nu_{j-1} - \tau_j + \tau_{j-1}\right) + s_{i+1}\left(\nu_{j+1} - \nu_j - \tau_{j+1} + \tau_j\right) + \ldots\right),\end{aligned}$$

where $s_j = 1$ in case $(\nu_j - \nu_{j-1} - \tau_j + \tau_{j-1}) \geq 0$, and $s_j = -1$ otherwise. Since ν and τ are smooth, and z-levels are small, one may consider the case $s_j = s_{j+1}$ most frequent compared to the case $s_j = -s_{j+1}$. In case $s_j = s_{j+1}$:

$$\left.\frac{\partial\mathcal{L}'}{\partial\nu_j}\right|_{s_j=s_{j+1}} = s_i\frac{\partial}{\partial\nu_j}\left(\nu_j - \nu_{j-1} - \tau_j + \tau_{j-1} + \nu_{j+1} - \nu_j - \tau_{j+1} + \tau_j\right) = 0,$$

which is an uninformative gradient.

In case $s_j = -s_{j+1}$:

$$\left.\frac{\partial\mathcal{L}'}{\partial\nu_j}\right|_{s_j=-s_{j+1}} = s_j\frac{\partial}{\partial\nu_j}\left(\nu_j - \nu_{j-1} - \tau_j + \tau_{j-1} - \nu_{j+1} + \nu_j + \tau_{j+1} - \tau_j\right) = 2s_j,$$

which is an uninformative gradient since it is not dependent of "true" values that are τ in this notation. Thus, in this section, we demonstrated that in case of MAE loss term, its gradients w.r.t. $\gamma_{NN,j}$ are uninformative constants - either zero or $\pm 2\delta_z$. This is in agreement with Fig. 2, where one may observe that the gradients with added 10^{-14} stability value are either 10^{-14} or some constant.

A.3. MAPE Term

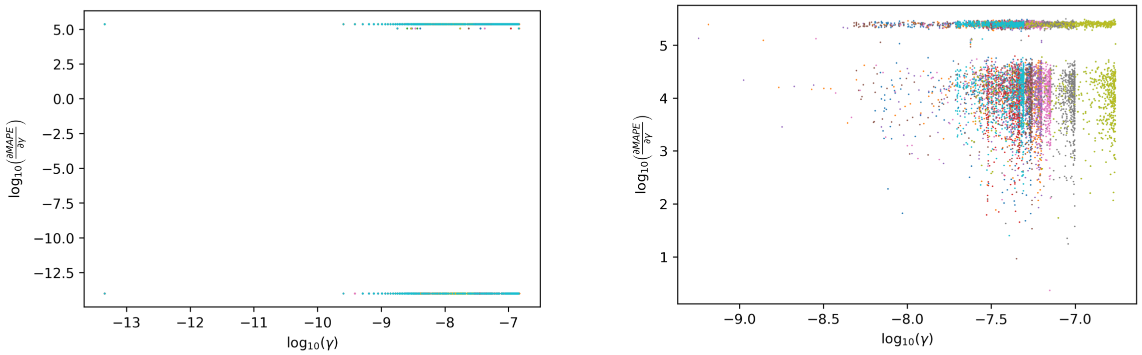
In this section, we consider MAPE $\left(\frac{\partial\gamma}{\partial z}\right)$.

$$\mathcal{L} = \sum_{i=1}^{N} \left| \frac{\left.\frac{\partial\gamma}{\partial z}\right|_{NN,i} - \left.\frac{\partial\gamma}{\partial z}\right|_{true,i}}{\left.\frac{\partial\gamma}{\partial z}\right|_{true,i}} \right|, \tag{73}$$

One may notice that τ is smooth, thus, in case of finite-difference approximation of the derivatives in this loss term, $\Delta\tau_j \approx \Delta\tau_{j+1}$. The derivation of the gradients in this section is similar to the previous section A.2 with the only reservation of denominator $\Delta\tau_j$, thus, the gradients may be expressed the following way:

$$\begin{aligned} \left.\frac{\partial \mathcal{L}'}{\partial \nu_j}\right|_{s_j = s_{j+1}} &= 0, \\ \left.\frac{\partial \mathcal{L}'}{\partial \nu_j}\right|_{s_j = -s_{j+1}} &= \frac{2s_j}{\Delta\tau_j}. \end{aligned} \tag{74}$$

Here one may note once again that the case $s_i = s_{j+1}$ is much more frequent compared to the case $s_i = -s_{j+1}$ due to smoothness of γ coefficient and $F_{NN,\gamma}$ neural network. Thus, major gradient values are zeros which is uninformative. This is in agreement with Fig. 8, where one may observe that the gradients with added 10^{-14} stability value are mostly 10^{-14} with some rare exceptions. In Fig. 8b, one may also observe that random re-weighting of individual MAPE sum elements according to eq. (61) made the gradients informative.



(a) The case of routine MAPE$_\gamma$ formulation

(b) The case of random re-weighting of individual MAPE$_\gamma$ sum terms

Figure 8. MAPE$_\gamma$ loss gradients w.r.t. γ as individual sum terms vs. γ: (a) in case of routine MAPE$_\gamma$ formulation; (b) in case of random re-weighting of individual MAPE$_\gamma$ sum terms according to eq. (61). Here we present it in logarithmic scale with additive log-stability term 10^{-14}