Artificial Neural Networks for the Identification of Partial Differential Equations of Land Surface Schemes in Climate Models

Mikhail A. Krinitskiy,[a,b,*] **Victor M. Stepanenko**[b,c,d] **and Ruslan V. Chernyshev**[b,c]

[a] *Shirshov Institute of Oceanology, Russian Academy of Sciences, 36 Nahimovskiy pr., Moscow, 117997, Russia*
[b] *Research Computing Center, Lomonosov Moscow State University, 1 Leninskie Gory, bld. 4, Moscow, 119234, Russia*
[c] *Faculty of Geography, Lomonosov Moscow State University, GSP-1, 1 Leninskie gory, bld. 1, Moscow, 119234, Russia*
[d] *Moscow Center for Fundamental and Applied Mathematics, GSP-1, 1 Leninskie gory, bld. 1, Moscow, 119234, Russia*

E-mail: krinitsky@sail.msk.ru

Land surface scheme in climate models is a solver for a nonlinear PDE system, which describes thermal conductance and water diffusion in soil. Thermal conductivity λ_T, water diffusivity λ_W and hydraulic conductivity γ coefficients of this system are functions of the solution of the system W and T. For the climate models to accurately represent the Earth system's evolution, one needs to approximate the coefficients or estimate their values empirically. Measuring the coefficients is a complicated in-lab experiment without a chance to cover the full range of environmental conditions. In this work, we propose a data-driven approach for approximating the parameters of the PDE system describing the evolution of soil characteristics. We formulate the coefficients as parametric functions, namely artificial neural networks. We propose training these neural networks with the loss function computed as a discrepancy between the PDE system solution and the measured characteristics W and T. We also propose a scheme inherited from the backpropagation method for calculating the gradients of the loss function w.r.t. network parameters. As a proof-of-concept step, we assessed the capabilities of our approach in three synthetic scenarios: a nonlinear thermal diffusion equation, a nonlinear liquid water W diffusion equation, and Richards equation. We generated realistic initial conditions and simulated synthetic evolutions of W and T that we used as measurements in the networks‘ training procedure for these three scenarios. The results of our study show that our approach provides an opportunity for reconstructing the PDE coefficients of various forms accurately without actual knowledge of their ground truth values.

The 5th International Workshop on Deep Learning in Computational Physics
28-29 June, 2021
Moscow, Russia

[*]Speaker

1. Introduction

The core of a land surface scheme in climate models is a solver for a nonlinear PDE system describing thermal conductance and water diffusion in soil (1, 2). The system includes heat conductivity equation:

$$\rho C \frac{\partial T}{\partial t} = \frac{\partial}{\partial z} \lambda_T \frac{\partial T}{\partial z}, \tag{1}$$

where ρ is soil density, C is thermal capacity, and z is a coordinate directed along gravity downwards. In this study, we do not consider water phase transitions, thus we dropped some of the terms that are irrelevant in this case. The second equation of the system is the one for liquid water content (2) describing vertical transport (diffusion and gravitational infiltration):

$$\frac{\partial W}{\partial t} = \frac{\partial}{\partial z} \left(\lambda_W \frac{\partial W}{\partial z} \right) + \frac{\partial \gamma}{\partial z}, \tag{2}$$

where we also dropped irrelevant terms for the sake of clarity. There are also an equation describing water vapor transport and the dynamics of ice content that we do not consider in this study.

The system of equations above is supplemented by boundary conditions, representing heat and mass balance at boundaries $z = 0, H$ or prescribed time series (Dirichlet conditions). The system includes thermal conductivity $\lambda_T(W)$, liquid water diffusivity $\lambda_W(W)$ and hydraulic conductivity $\gamma(W)$ coefficients that are functions of the solution of the system, i.e., liquid water content W. For the climate models to accurately represent the Earth system's evolution, one needs to identify the equations meaning either approximating the coefficients or assessing their values empirically. Measuring the coefficients is a complicated in-lab experiment. Also, there are many soil types that differ in physical properties, which results in varying dependencies $\lambda_T(W)$, $\lambda_W(W)$ and $\gamma_W(W)$. Thus, there is no chance to exhaustively measure soil characteristics for all soil types covering the full range of environmental conditions. There are known approximate parametric forms of the coefficients [1–3] that, however, lack accuracy and, in turn, need their identification w.r.t. their own parameters.

The objective of this study is to show the way to acquire the coefficients as functions of solutions T and W, given known solution $W(t), T(t)$ or measured evolution of soil column $W_m(t), T_m(t)$.

2. Methods

In this proof-of-concept study, we consider known forms of the PDE parameters λ_T, λ_W and γ shown in Section 2.3 as ground truth. We consider processes described by equations (1,2,3) with known λ_T, λ_W and γ as the ones that generate the measured T and W profiles. We also consider these ground truth coefficients being too expensive to measure in a field or in-lab experiments. Instead, we propose a data-driven approach for approximating these coefficients. In our study, we formulate the approximated coefficients as parametric functions $\lambda_T(T) = F(T, \theta_{\lambda_T})$, $\lambda_W(W) = F(W, \theta_{\lambda_W})$ and $\gamma_W(W) = F(W, \theta_\gamma)$, namely artificial neural networks. Here, θ_{λ_T}, θ_{λ_W} and θ_γ are the networks' coefficients. We formulate the networks the way so they have high enough expressive power to represent a wide range of nonlinear functions. In this study, we model numerically the evolution of soil characteristics T or W with ground truth coeffcients. This way we acquire the evolution we call "measured" or "true" (see top branch of forward computations in Figure 1 marked with blue bold arrows). We then model the same evolution using the coefficients approximated by the neural networks mentioned above (see bottom branch of forward computations in Figure 1 marked with red bold arrows). The parameters of the networks are then optimized by minimizing the discrepancy between the modeled solutions of these two types. In Figure 1, one may see the two numerical solutions of a PDE and the loss function which is shown to be $MSE\left(\frac{\partial X}{\partial t}\right)$, where X is a substitute for temperature T or liquid water content W depending on the equation considered. Note that the loss function is not necessarily mean squared error, but may be any other differentiable loss function.

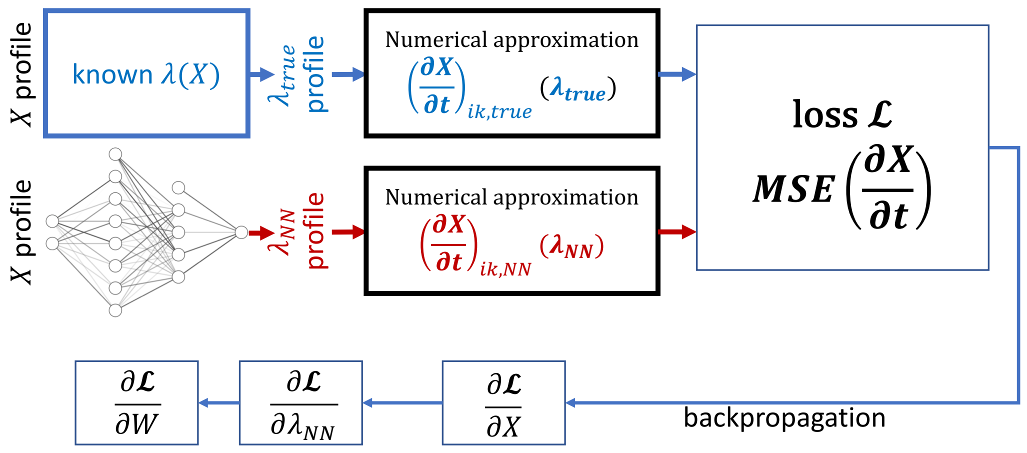
X profile
known $\lambda(X)$
λ_{true} profile
Numerical approximation $\left(\frac{\partial X}{\partial t}\right)_{ik,true}(\lambda_{true})$
X profile
λ_{NN} profile
Numerical approximation $\left(\frac{\partial X}{\partial t}\right)_{ik,NN}(\lambda_{NN})$
loss $\mathcal{L}$
$MSE\left(\frac{\partial X}{\partial t}\right)$
$\frac{\partial \mathcal{L}}{\partial W}$
$\frac{\partial \mathcal{L}}{\partial \lambda_{NN}}$
$\frac{\partial \mathcal{L}}{\partial X}$
backpropagation

Figure 1: The scheme of the approach for the computation of loss function $\mathcal{L}$ gradients w.r.t. parameters of the neural network. Here, the numerical approximations of the partial derivatives $\frac{\partial X}{\partial t}$ are computed using the formulae (4) and (5); X is a substitute for temperature T or liquid water content W; λ_{true} is known form for the parameter to identify; λ_{NN} is the approximation of this parameter computed using the neural network; i enumerates level elements following the spatial coordinate z; k enumerates profiles of a mini-batch in case of mini-batch stochastic optimization procedure.

2.1 Problem setup

In this study, we assess the capabilities of the approach we present in three scenarios. In the **first scenario**, we consider nonlinear thermal conductance (diffusion) equation (1). We assume Dirichlet boundary conditions meaning for each T profile the solution on the boundaries assumed either to be constant or to be represented by a time series. The method we propose, however, may be applied to cases with other b.c.s as well. The task is to find a dependency $\lambda_{NN}(T)$ represented by a neural network, given a solution T of this equation and which minimizes the error of T_{NN} approximation in respect to T, where T_{NN} is a solution of (1) given $\lambda_T(T) = \lambda_{NN}(T)$. In this scenario, we imply the only *a priori* assumption about $\lambda_T(T)$ that the function $\lambda_T(T)$ may be approximated by a neural network with an appropriate quality.

In the **second scenario**, we consider a nonlinear liquid water diffusion equation with Dirichlet b.c.s, similar to the thermal conductance equation presented above:

$$\frac{\partial W}{\partial t} = \frac{\partial}{\partial z}\left(\lambda_W(W)\frac{\partial W}{\partial z}\right). \qquad (3)$$

Here, the assumptions about $\lambda_W(W)$ are similar to the ones about $\lambda_T(T)$ with additional physics-prescribed considerations:

- $W \geq 0$, and $W \leq 1$;
- $\lambda_W(0) = 0$;
- $\frac{\partial \lambda_W}{\partial W} \geq 0$ for any W satisfying the first requirement.

In the **third scenario**, we consider the equation of nonlinear liquid water dynamics in soil, also known as Richards equation (2). The assumptions about $\lambda_W(W)$ here are the same. We also considered the same assumptions about $\gamma_W(W)$.

2.2 Loss function and optimization procedure

In a routine supervised data-driven problem, one needs to present ground truth for a target value. In the case of a soil PDE system, one cannot afford measurements of the full range of coefficients' ground truth values. In contrast to the routine data science approach, we propose training the neural networks with the loss function computed as a discrepancy between a PDE solution with the coefficients approximated by the network, and the measured or modeled evolution of the characteristics W and T.

As mentioned in Section 2.1, we model the "true" evolution as a numerical solution of a PDE with known forms of the coefficients. The numerical solution is often known in discrete points of time and space. For the discrete mesh, we approximate numerically the partial derivatives in equations (1), (2) or (3) using explicit scheme. In case of eq. (1), the derivative at some moment t is given with the following formula:

$$\frac{\partial}{\partial z}\left(\lambda_T(T)\frac{\partial T}{\partial z}\right)_i = \frac{2}{\Delta z_i}\left[\lambda_T(T_{i+1/2})\frac{T_{i+1}-T_i}{\Delta z_{i+1}+\Delta z_i} - \lambda_T(T_{i-1/2})\frac{T_i - T_{i-1}}{\Delta z_i + \Delta z_{i-1}}\right], \qquad (4)$$

where i enumerates level elements following the spatial coordinate z. The approximate derivatives for equation (3) is given with similar formula. In case of complete Richards equation (2), the derivative at some moment t is given with the following formula:

$$\frac{\partial W}{\partial t}_i = \frac{2}{\Delta z_i}\left[\lambda_W(W_{i+1/2})\frac{W_{i+1}-W_i}{\Delta z_{i+1}+\Delta z_i} - \lambda(W_{i-1/2})\frac{W_i - W_{i-1}}{\Delta z_i + \Delta z_{i-1}}\right] + \frac{\gamma(W_{i+1/2}) - \gamma(W_{i-1/2})}{\Delta z_i}, \qquad (5)$$

One may note that these numeric estimates of the right-hand sides of the equations (1), (2) and (3) are linear functions of λ_T, λ_W and γ. Thus, these functions are differentiable w.r.t. the coefficients λ_T, λ_W and γ. Once one can calculate partial deivatives $\frac{\partial T}{\partial \lambda_T}$, $\frac{\partial W}{\partial \lambda_W}$ and $\frac{\partial W}{\partial \gamma}$, there is a way to employ chain rule for calculating gradients of a loss function w.r.t. parameters θ_{λ_T}, θ_{λ_W} and θ_γ. In simple case of diffusion equations (1) or (3), one can compute the gradients of loss function $\mathcal{L}\left(X, \lambda_X, \theta_{\lambda_X}\right)$ w.r.t. neural network parameters θ_{λ_X} the following way:

$$\frac{\partial \mathcal{L}}{\partial \theta_{\lambda_X}} = \frac{\partial \mathcal{L}}{\partial X} \times \frac{\partial X}{\partial \lambda_{X,NN}} \times \frac{\partial \lambda_{X,NN}}{\partial \theta_{\lambda_X}}, \tag{6}$$

where X is a substitute for T or W depending on the equation, $\lambda_{X,NN}$ is the coefficient approximated with neural network $F\left(X, \theta_{\lambda_X}\right)$, θ_{λ_X} are the parameters of the network. The term $\frac{\partial \mathcal{L}}{\partial X}$ is the derivative of $MSE\left(x\right)$ or $MAPE\left(x\right)$ w.r.t. x; the term $\frac{\partial X}{\partial \lambda_{NN,X}}$ is calculated using the derivative of an explicit form (4) or (5) depending on the scenario; the term $\frac{\partial \lambda_{X,NN}}{\partial \theta_{\lambda_X}}$ is calculated following backpropagation scheme common for artificial neural networks. The whole scheme presented in eq. (6) is essentially backpropagation rule for artificial neural networks with additional custom operation, namely numerical integration using explicit finite difference scheme (eq. 4 or 5). Figure 1 illustrates this approach in a simplified manner.

Using the approach described above for calculating the gradients of the loss functions $\mathcal{L}\left(F_{NN,X}\left(X, \theta_X\right)\right)$ w.r.t. parameters θ_X, we optimize it using stochastic gradient optimization methods. We employ Adam optimizer [4] in this study.

As mentioned in Section 2, the loss function represents the discrepancy between the numerical solutions of a PDE (1, 3 or 2) using known coefficients $\lambda_{T,true}$, $\lambda_{W,true}$ and γ_{true} and the ones approximated by neural networks: $\lambda_{T,NN}$, $\lambda_{W,NN}$ and γ_{NN}. Considering the evolutions being partial derivatives $\frac{\partial X}{\partial t}$ integrated numerically, one may note that the integration is a sum of a derivative estimates multiplied by dt in case of explicit Euler integration scheme. Since dt is a constant scalar, the loss functions and their gradients are just scaled by dt compared to those calculated as a discrepancy between partial derivatives $\frac{\partial X_{true}}{\partial t}$ and $\frac{\partial X_{NN}}{\partial t}$. Thus, instead of X evolution errors, in this study, we used linear combination of mean absolute percentage error ($MAPE$) and mean squared error (MSE) of partial derivatives of the solutions X_{true} and X_{NN} as loss function:

$$\mathcal{L} = \alpha_{mse} \times MSE\left(\frac{\partial X_{NN}}{\partial t}, \frac{\partial X_{true}}{\partial t}\right) + \alpha_{mape} \times MAPE\left(\frac{\partial X_{NN}}{\partial t}, \frac{\partial X_{true}}{\partial t}\right), \tag{7}$$

where α_{mse} and α_{mape} are MSE and $MAPE$ loss coefficients correspondingly, and they are the hyperparameters of our method.

In the **second** and **third** scenarios, we also used physics-imposed regularizations that represent physics-prescribed constraints mentioned in Section 2.1. Here are the regularization terms (written for λ_W):

$$\mathcal{L}_{reg,zp} = \alpha_{zp} \times \left|F_{NN,\lambda_W}\left(W = 0\right)\right|, \tag{8}$$

$$\mathcal{L}_{reg,neg} = \alpha_{neg} \times \sum_{F_{NN,\lambda_W}(W)<0} \left|F_{NN,\lambda_W}\left(W\right)\right|, \tag{9}$$

$$\mathcal{L}_{reg,dp} = \alpha_{dp} \times \sum_{\frac{\partial F_{NN,\lambda_W}(W)}{\partial W}<0} \left|\frac{\partial F_{NN,\lambda_W}\left(W\right)}{\partial W}\right|, \tag{10}$$

where $\mathcal{L}_{reg,zp}$ is the zero-point regularization term corresponding to the constraint $\lambda_W(0) = 0$; $\mathcal{L}_{reg,neg}$ corresponds to the constraint $\lambda_W > 0$; the term $\mathcal{L}_{reg,dp}$ is called derivative penalty and corresponds to the constraint $\frac{\partial \lambda_W}{\partial W} \geq 0$; α_{zp}, α_{neg} and α_{dp} are regularization coefficients that are hyperparameters of the method. Note that we added derivative penalty (10) only in the **third** scenario since the tasks regarding the diffusion equations (1,3) were optimized easily without this regularization term. In case of Richards equation (2), we added regularization terms for γ coefficient similar to eqs. (8,9,10).

In all of the three scenarios, we simulated measurement errors by adding zero-centered Gaussian noise to the "true" partial derivatives $\frac{\partial X_{true}}{\partial t}$. On the other hand, the additive noise may be considered data augmentation in terms of standard machine learning framework. The noise we add is spatially correlated and parameterized by correlation radius r_c, so the covariance matrix for level-wise noise samples is parameterized by r_c:

$$\Sigma_{ij} = \sigma^2 e^{-\frac{|z_i - z_j|}{r_c}}, \tag{11}$$

where σ is noise magnitude in uncorrelated case, i.e., the value of diagonal elements of Σ_{ij}. Thus, σ and r_c are hyperparameters that are not supposed to be optimized. Instead, one may study the sensitivity of the method to the magnitude σ and correlation radius r_c to predict the sensitivity to errors in real measurements. One may also de-center the noise simulating systematic measurement errors. In this study, we did not explore the full range of available noise options. However, the option we exploited is reasonable when simulating high-quality instruments‘ errors in an accurately designed field experiment.

2.3 Ground truth options

In the **first scenario**, we model heat diffusion in soil integrating eq. (1) within the framework described above. In this scenario, we define $\lambda_{T,true}$ by the following options:

$$\lambda_{T,true} = a\exp(bT), \tag{12}$$

$$\lambda_{T,true} = a\exp(bT) * \frac{C + \cos(k\pi T)}{C}, \tag{13}$$

$$\begin{cases} \zeta = \frac{1}{2}\left(1 - \tanh\left(\epsilon\left(T - T_{th}\right)\right)\right), \\ \lambda_T(T) = a_1 e^{b_1 T} * \zeta + a_2 e^{b_2 T} * (1 - \zeta), \end{cases} \tag{14}$$

where a, b, C, k, a_1, a_2, b_1, b_2, T_{th} and ϵ are human-defined parameters characterizing the features of λ_T. In particular, ϵ characterizes the smoothness of the thresholding behavior; T_{th} characterizes the location of the threshold. In this study, we set $T_{th} = 5°C$, and tested two cases: $\epsilon = 2$ and $\epsilon = 6$. In Figure 2, we present the diagrams of the ground truth forms of λ_T in the **first scenario**.

In case of liquid water diffusivity $\lambda_{W,true}(W)$ and hydraulic conductivity $\gamma_{true}(W)$ coefficients in the **second** and **third** scenarios, we used widely accepted functions proposed by Mualem [2] and van Genuchten [1]:

$$\lambda_{W,true}(W) = \frac{\gamma_{max}(1-m)}{\alpha m(W_{max} - W_{min})} W^{1/2-1/m}\left[\left(1 - W^{1/m}\right)^{-m} + \left(1 - W^{1/m}\right)^{m} - 2\right] \tag{15}$$

$$\gamma_{W,true}(W) = \gamma_{max} W^{1/2}\left[1 - \left(1 - W^{1/m}\right)^{m}\right]^2, \tag{16}$$

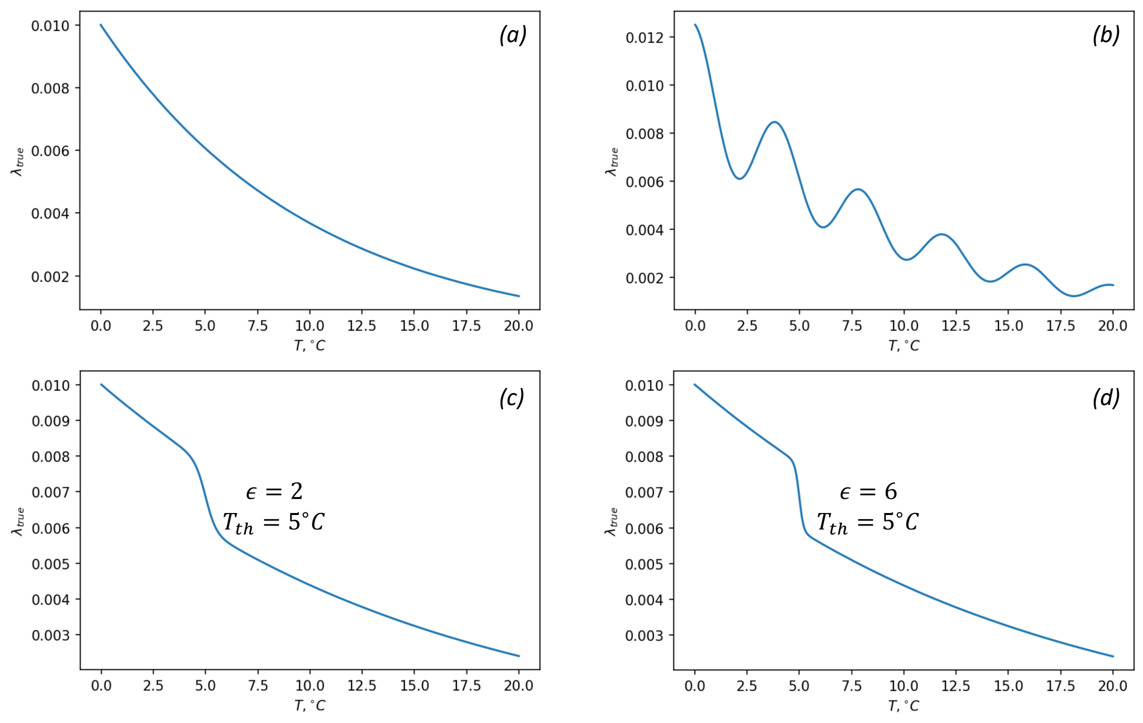

Figure 2: The shape of "true" thermal conductivity $\lambda_{T,true}(T)$: **(a)** in case of eq. (12), **(b)** in case of eq. (13), and **(c,d)** in case of eq. (14): soft and strong threshold correspondingly.

where γ_{max}, m, W_{max}, W_{min} and α are the parameters describing soil type. In this study, we used the following values for the parameters: $\gamma_{max} = 1.56 \times 10^{-4}$; $m = 0.5$, $alpha = 0.5$, $W_{max} - W_{min} = 0.5$. In Figure 3, the above mentioned analytical representations of $\lambda_{W,true}$ and γ_{true} are presented.

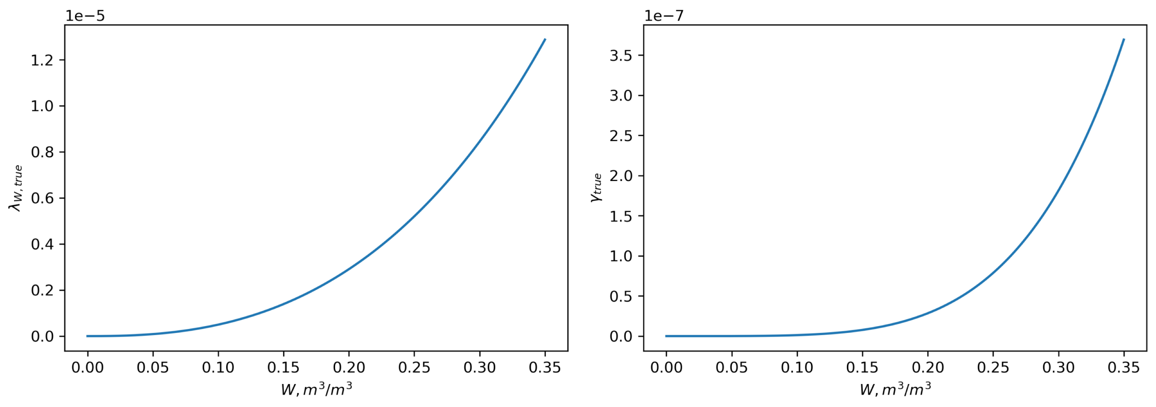

Figure 3: The shapes of "true" **(a)** liquid water diffusivity $\lambda_{W,true}(W)$ and **(b)** hydraulic conductivity γ_{true} in equations (2,3) in the **second** and **third** scenarios.

Worth being noted, that within the presented approach, one does not need to actually integrate the continuous evolution of soil characteristics. In fact, the loss function (7) relies only on partial derivatives $\frac{\partial T}{\partial t}$ and $\frac{\partial W}{\partial t}$ of a time moment. Following stochastic optimization procedure, we generate a batch of T and W profiles simulating various evolution moments. With these profiles, we calculate the partial derivatives using appropriate finite difference scheme (4) or (5). We then compute the loss function in accordance with eqs. (7,8,9,10). Using backpropagation rule (6), we then compute loss function gradients w.r.t. neural network parameters: θ_{λ_T} in the **first scenario**; θ_{λ_W} in the **second scenario**; θ_{λ_W} and θ_γ in the **third scenario**. We then apply stochastic gradient descent step using Adam [4] optimizer.

In this study, we also employ the following techniques for stabilizing the training and faster convergence: gradient clipping by global norm; gradient clipping by individual values; learning rate scheduling using cosine SGDR schedule [5] with simulated annealing and exponential decay of annealing magnitude; and exponential decay of noise magnitude σ (see eq. (11)).

The architecture of the neural networks we use is the same for all of the three scenarios. We use fully-connected artificial neural network (a.k.a. multilayer perceptron) with six layers. The layers are wide enough for the network to have high enough expressive power for approximating functions like (13). While designing the networks, we optimized its structural hyperparameters ensuring its capability of approximating complicated functions by solving simple supervised problem with target values generated by "true" forms like eqs. (12,14,13). In order to improve convergence, we used *Mish* activation function [6] for all layers except the last one, where we used linear activation.

2.4 Quality assessment

We assess the quality of our method with the four quality measures:

- $MAPE\,(\lambda)$, where λ is λ_T in the **first** scenario and λ_W in the **second** and **third** scenarios:
- $MAPE\,(\gamma)$, which is applicable for the **third** scenario only;
- $RMSE\left(\frac{\partial X}{\partial t}\right)$, where X is temperature T in the **first** scenario, and liquid water content W in the **second** and **third** scenarios.

Here $RMSE$ stands for "root mean squared error", and $MAPE\,(x)$ stands for "mean absolute percentage error":

$$RMSE\,(x) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(x_{i,NN} - x_{i,true}\right)^2},$$

$$MAPE\,(x) = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{x_{i,NN} - x_{i,true}}{x_{i,true} + \epsilon}\right|.$$

where x is the substitute for $\frac{\partial X}{\partial t}$, λ or γ; i is a finite element index; ϵ is a very small positive number added for computational safety.

3. Experiments, results and discussion

Following the method and problem setups we presented in Section 2, we performed the optimization of fully-connected neural networks approximating λ_T, λ_W and γ in three abovementioned scenarios. Current code base of the study is written in Python 3.8.5. We used Tensorflow 2.4.0

Scenario	True parameter form	$MAPE\,(\lambda)$	$MAPE\,(\gamma)$	$RMSE\left(\frac{\partial X}{\partial t}\right)$
1-st, eq. (1)	eq. (12) (fig. 2a)	8×10^{-4}	-	4.6×10^{-6}
1-st, eq. (1)	eq. (13) (fig. 2b)	1.6×10^{-3}	-	7×10^{-6}
1-st, eq. (1)	eq. (14); $\epsilon = 2$ (fig. 2c)	1.5×10^{-4}	-	3.6×10^{-4}
1-st, eq. (1)	eq. (14); $\epsilon = 6$ (fig. 2d)	1.75×10^{-4}	-	2.0×10^{-4}
2-nd, eq. (3)	eq. (15) (fig. 3a)	2.3×10^{-2}	-	7.7×10^{-4}
3-rd, eq. (2)	eqs. (16,15) (fig. 3a,b)	4.5×10^{-2}	4.8×10^{-2}	4.7×10^{-6}

Table 1: Quality of the presented approach assessed with measures described in Section 2.4.

library [7] for automatic differentiation and also for creating and optimizing neural networks. Computations were performed on NVIDIA DGX Station equipped with GPU NVIDIA V100 (32GB).

In Table 1, we show the results of our method in all the three scenarios assessed in terms of quality measures described in Section 2.4. In Figures 4,5,6, we also show qualitative results of the approximations $\lambda_{T,NN}$, $\lambda_{W,NN}$ and γ_{NN} in comparison with the true forms $\lambda_{T,true}$, $\lambda_{W,true}$ and γ_{true}. In these diagrams, one may see a clear correspondence between "true" forms of the coefficients and their approximations with neural networks. This correspondence is supported by the quantitative results shown in tab. 1: $MAPE\,(\lambda)$ is usually of order 10^{-2} or better (lower). In case of the **third** scenario, the optimization of γ network $F_{NN,\gamma}\left(W, \theta_\gamma\right)$ lacks convergence out-of-the-box. Most probable cause for that is the difference in orders of magnitude of the coefficients λ_W and γ and in the corresponding terms of Richards equation (2). At the same time, accurate hyperparameters tuning helps stabilizing the training. We found that the values of regularization coefficients in (eqs. 8,9,10) delivers major improvement in the $F_{NN,\gamma}$ training. However, there is still a room for improvement in the convergence of γ network within the **third** scenario.

4. Conclusions

In this work, we presented a novel framework for the identification of partial differential equations meaning approximating their coefficients in case they may be represented as functions of these PDE solutions themselves. We demonstrated the way one may employ differentiable functions in this problem. In our study, we used the functions of a specific class, namely artificial neural networks, known to be capable of approximating a broad range of various dependencies (a.k.a. universal approximators). However, there is no particular reason for limiting the class of functions since the approach we demonstrated in this study works perfectly with any kind of differentiable functions. For example, one may fit widely accepted parametric functions (e.g., [1–3]), thus replacing complicated in-lab measurements of soil characteristics by field measurements of T and W profiles. This kind of experiment setup may provide much higher accuracy and reliable statistcal significance due to the opportunity for automatic measurements.

We demonstrated the capabilities of our approach in the case of PDEs known to describe the evolution of soil characteristics in climate models. Our results show that the method we presented delivers approximations with high accuracy without prior knowledge about the exact form of the

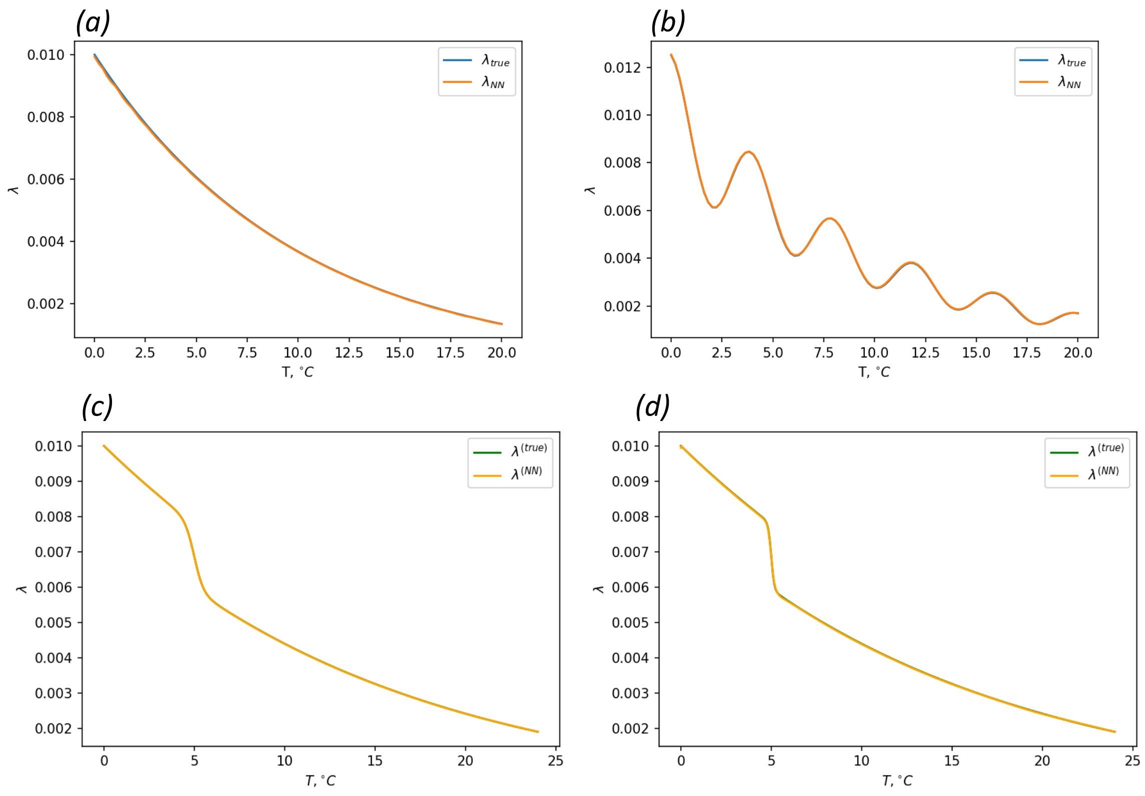

Figure 4: $\lambda_T(T)$ approximation qualitative results within the **first** scenario: **(a)** in case of the true form $\lambda_{T,true}$ given by eq. (12); **(b)** in case of the true form $\lambda_{T,true}$ given by eq. (13); **(c)** in case of the true form $\lambda_{T,true}$ given by eq. (14) with smooth threshold ($\epsilon = 2$); **(d)** in case of the true form $\lambda_{T,true}$ given by eq. (14) with sharp threshold ($\epsilon = 6$).

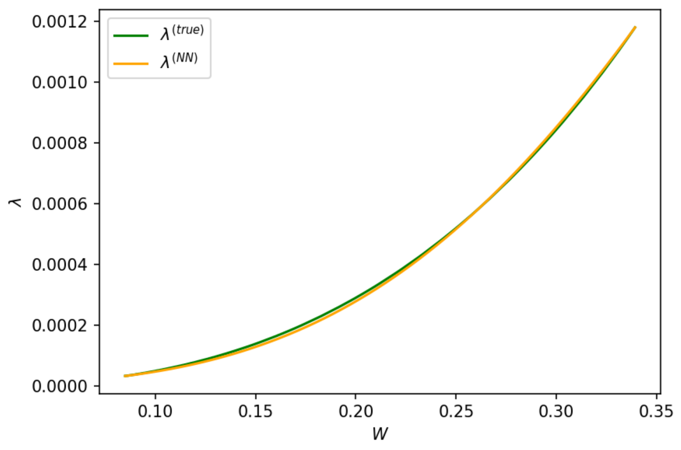

Figure 5: Qualitative results of λ_W approximation within the **second** scenario in case of the true form $\lambda_{W,true}$ given by eq. (15)

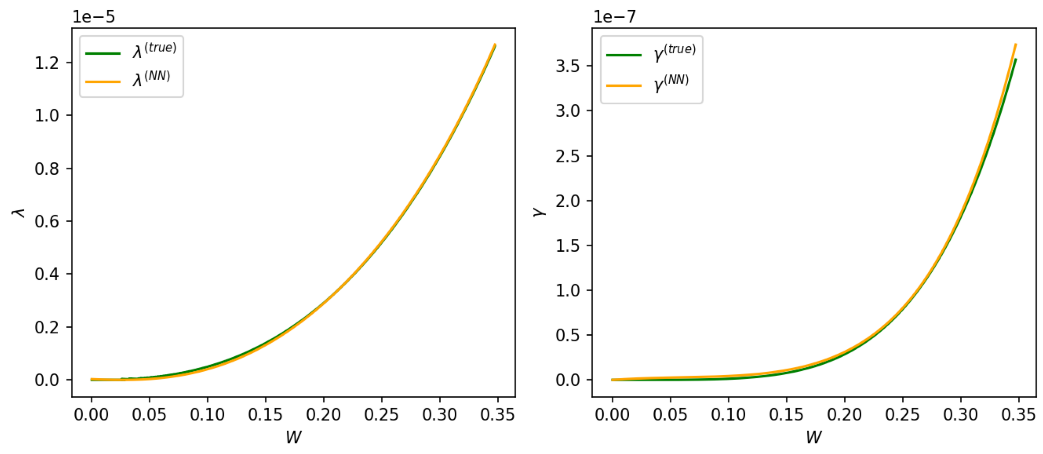

Figure 6: Qualitative results of $\lambda_{W,NN}$ and γ_{NN} approximation within the **third** scenario in case of the true forms $\lambda_{W,true}$ and γ_{true} given by eqs. (15),(16)

identified coefficients. The only assumption about these coefficients is that they may be represented by some differentiable functions of the PDE solution. In contrast to well-known machine learning approach, our method does not require target variable values, meaning coefficients to identify. This is particularly convenient in case the coefficients are too expensive to measure. Instead, one may rely on the measured evolution of the characteristics described by the PDEs, that are usually measurable.

In this work, we found, that in some particular cases, the optimization lacks convergence, thus, there is a room for improvement. Most promising way for it is hyperparameters optimization, preferably focusing on the right balance of regularization coefficients. We also demonstrated that physics-prescribed constraints, imposed in a form of regularization terms, may improve the convergence of the neural networks approximating PDE coefficients.

Our approach is not limited by the land surface scheme of climate models. One may apply the same method in other problems involving PDE systems with funtional coefficients depending on the PDE system solution. The approach we presented in this study may be particularly fruitful when developing parameterizations of turbulent fluxes in the atmosphere, in the ocean or in the boundary layer. However, there always will be a trade-off between the convenience and the accuracy of our method, and the interpretability of the acquired solutions. Also, our approach is essentially based on machine learning framework, thus, the identified coefficients may not be quite accurate in variables range supported insufficiently by training sample.

Acknowledgments

The work was supported by Russian Foundation for Basic Research (grant No. 20-05-00773) and Moscow State University interdisciplinary school “Brain, cognitive systems, artificial intelligence”.

References

[1] M.T. van Genuchten, *A Closed-form Equation for Predicting the Hydraulic Conductivity of Unsaturated Soils*, *Soil Science Society of America Journal* **44** (1980) 892.

[2] Y. Mualem, *A new model for predicting the hydraulic conductivity of unsaturated porous media*, *Water Resources Research* **12** (1976) 513.

[3] J. Côté and J.-M. Konrad, *A generalized thermal conductivity model for soils and construction materials*, *Canadian Geotechnical Journal* **42** (2005) 443.

[4] D.P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 1412.6980.

[5] I. Loshchilov and F. Hutter, *Sgdr: Stochastic gradient descent with warm restarts*, 1608.03983.

[6] D. Misra, *Mish: A self regularized non-monotonic activation function*, 1908.08681.

[7] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro et al., *Tensorflow: Large-scale machine learning on heterogeneous distributed systems*, 1603.04467.