A Novel Numerical Method for Agro-Hydrological Modeling of Water Infiltration in Soil

Zeyuan Song, Zheyu Jiang
School of Chemical Engineering
Oklahoma State University
Stillwater, OK 74074
{taekwon.song,zjiang}@okstate.edu

Abstract

Root-zone soil moisture monitoring is essential for sensor-based precision agriculture, smart irrigation, and agricultural drought prevention. Modeling the spatiotemporal water flow dynamics in porous media such as soil is typically achieved by solving an agro-hydrological model, the most important of which being the Richards equation. To overcome the computational challenges associated with solving this highly nonlinear partial differential equation (PDE), in this paper, we present a novel solution algorithm, which we name as the DRW (Data-driven global Random Walk) algorithm, that holistically integrates adaptive L-scheme, neural networks, and global random walk in a finite volume discretization framework. We thoroughly discuss the need and benefits of introducing these components to achieve synergistic improvements in solution accuracy and numerical stability. Furthermore, we show that the DRW algorithm can accurately solve the mixed-form n-dimensional Richards equation with guaranteed convergence under reasonable assumptions. Through several illustrative examples, we demonstrate that the DRW algorithm not only achieves superior accuracy, but also better preserves the underlying physics and mass conservation of the Richards equation compared to state-of-the-art solution algorithms and commercial solver. Furthermore, the data-driven feature of our DRW algorithm makes it amenable for integrating with soil sensing technologies for root-zone soil moisture monitoring leveraging real-time soil sensor measurements. Overall, the proposed data-driven numerical method is expected to be a generalizable framework for modeling a wide range of applications, including fluid transport in fibrous porous materials, liquid extraction from saturated granular materials, water and hydrocarbon flow in petroleum reservoirs, etc.

Keywords: Soil moisture, precision agriculture, Richards equation, global random walk, neural network

1 Introduction

With increasing demand for food and the resulting food-energy-water nexus challenges from the growing population, there is an increasing interest among chemical engineering researchers, especially those in process systems engineering (PSE), to design the next-generation food and agricultural systems that are sustainable, resource-efficient, and resilient. In particular, the development of new PSE tools that provide "sustainable engineering solutions for food and water" is prominently featured by the National Academies as one of the "key research priorities and new directions in chemical engineering" [1].

Along this line, sensor-based digital agriculture for sustainable and efficient use of water by leveraging real-time soil monitoring is essential for improving agricultural production and crop

productivity, providing basis for precision irrigation and agriculture, preventing leaching of agro-chemicals and soil nutrients into groundwater, and predicting agricultural droughts [2]. Recent studies reveal that adjusting irrigation activities based on root-zone soil moisture information can reduce irrigation water consumption by 40-60% [3] and increase farmer's revenue by 20-60% [4]. Modeling the spatiotemporal behavior of root zone soil moisture from precipitation and surface soil moisture data is typically achieved by solving an agro-hydrological model that describes water movement through unsaturated soils. Nowadays, most existing agro-hydrological models are based on the Richards equation [5], which captures irrigation, precipitation, evapotranspiration, runoff, and drainage dynamics in soil:

$$\begin{aligned} &\partial_t \theta(\psi) + \nabla \cdot \mathbf{q} = -S(\psi), \\ &\mathbf{q} = -K(\theta(\psi))\nabla(\psi + z), \end{aligned} \tag{1}$$

where ψ stands for pressure head (in, e.g., m), $\mathbf{q}$ represents the water flux (in, e.g., $\mathrm{m}^3/\mathrm{m}^2 \cdot \mathrm{s}$), S is the sink term associated with root water uptake (in, e.g., s^{-1}), θ denotes the soil moisture content (in, e.g., $\mathrm{m}^3/\mathrm{m}^3$), K is unsaturated hydraulic water conductivity (in, e.g., m/s), $t \in [0, T]$ denotes the time (in, e.g., s), and z corresponds to the vertical depth (in, e.g., m). The Richards equation is a nonlinear convection-diffusion equation [6], in which the convection term is due to gravity and the diffusive term comes from Darcy's law [7]. For unsaturated flow, both θ and K are highly nonlinear functions of pressure head ψ and soil properties, making Equation (1) challenging to solve. Specifically, $\theta(\psi)$ and $K(\psi)$ (or $K(\theta)$, depending on the model) are commonly referred to as the water retention curve (WRC) and hydraulic conductivity function (HCF), respectively. Several empirical models have been developed for WRC and HCF for major soil types, among which some of the widely adopted ones are summarized in Table 1.

Model	HCF ($K(\psi)$ or $K(\theta)$)	WRC ($\theta(\psi)$)
Haverkamp [8]	$K_s \frac{A}{A+\lvert\psi\rvert^\gamma}$	$\theta_r + \frac{\alpha(\theta_s-\theta_r)}{\alpha+\lvert\psi\rvert^\beta}$
Mualem-van Genuchten [9, 10]	$K_s\sqrt{\frac{\theta-\theta_r}{\theta_s-\theta_r}}\left\{1-\left[1-\left(\frac{\theta-\theta_r}{\theta_s-\theta_r}\right)^{\frac{l}{l-1}}\right]^{\frac{l-1}{l}}\right\}^2$	$\theta_r + \frac{\theta_s-\theta_r}{\left[1+(\alpha\lvert\psi\rvert)^n\right]^{\frac{n-1}{n}}}$
Gardner [11]	$K_s e^{\alpha\psi}$	$\theta_r + (\theta_s - \theta_r)e^{\alpha\psi}$

Table 1: Some of the widely used HCF and WRC models. In these models, A, γ, α, β, n, θ_s, and θ_r are soil-specific parameters.

Due to the high nonlinearity of WRC and HCF, analytical solutions to the Richards equation do not exist in general [12]. Thus, the Richards equation is typically solved by numerically, which almost always requires some form of discretization. Consider the discretized version of Equation (1), whose control volume $V \subset \mathbb{R}^d$ ($d = 1, 2, 3$) is dicretized into N small cells $V_1, \ldots, V_N$. Using implicit Euler method on the time domain with a time step size of Δt, the discretized Richards equation at time step $m = 0, 1, \ldots, \lceil \frac{T}{\Delta t} \rceil - 1$ can be expressed as:

$$\begin{cases} \theta(\psi_i^{m+1}) - \theta(\psi_i^m) - \Delta t \nabla \cdot \left[K\left(\theta(\psi_i^{m+1})\right) \nabla\left(\psi_i^{m+1} + z\right) \right] + S(\psi_i^{m+1}) = 0, \\ \text{Dirichlet boundary condition: } \psi_j(\cdot) = 0 \quad \text{for all } V_j \subset \partial V, \\ \text{Initial condition: } \psi(0, \cdot) = \psi_0(\cdot), \end{cases} \tag{2}$$

where ψ_i^m is the pressure head in cell V_i and time step m, and $\psi_0(\cdot)$ denotes the initial condition at $t = 0$.

The performance of a numerical PDE solver depends theoretically on the well-posedness of the PDE [13], which is an essential property that certifies the accuracy and reliability of numerical solutions to the PDE. A PDE is said to be well-posed if its weak solution exists, is unique, and depends continuously on the problem's initial conditions [13, 14]. The weak solution of Equation (2) is formally defined as:

Definition 1.1. *Given $\psi_i^m \in H_0^1(V)$, if for any $v \in H_0^1(V)$ and $v(T,\cdot) = 0$,*

$$\left\langle \left(\theta(\psi^{m+1}) - \theta(\psi^m)\right), v\right\rangle_{V_i} + \Delta t \left\langle K\left(\theta(\psi^{m+1})\right)\nabla(\psi^{m+1}+z), \nabla v\right\rangle_{V_i} + \left\langle S(\psi^{m+1}), v\right\rangle_{V_i} = 0 \tag{3}$$

holds, then ψ_i^{m+1} is a weak solution of the discretized Richards equation.

Note that in Definition 1.1, an inner product $\langle\cdot,\cdot\rangle_{V_i} : L^2\left([0,T], H_0^1(V)\right) \to L^2\left([0,T], H_0^1(V)\right)$ is defined as $\langle f, g\rangle_{V_i} \coloneqq \int_{V_i} fg\,\mathrm{d}V$. When V_i is sufficiently small, $\int_{V_i} fg\,\mathrm{d}V = (fg)_i \mathrm{vol}(V_i)$, in which $(fg)_i$ denotes the value of fg evaluated at cell V_i, and $\|f\|^2 \coloneqq \langle f, f\rangle_{V_i} = (f^2)_i \mathrm{vol}(V_i)$. In particular, we point out that Equation (3) is originated from Equation (2) as:

$$\begin{aligned}
\langle \nabla\cdot[K\left(\Theta(\psi^{m+1})\right)\nabla\left(\psi^{m+1}+z\right)], v\rangle_{V_i} &= \int_{V_i} \nabla\cdot\left[K\left(\Theta(\psi^{m+1})\right)\nabla\left(\psi^{m+1}+z\right)\right] v\,\mathrm{d}V \\
&= \int_{\partial V_i} K\left(\Theta(\psi^{m+1})\right)\nabla\left(\psi^{m+1}+z\right) v\,\mathrm{d}S - \int_{V_i} K\left(\Theta(\psi^{m+1})\right)\nabla\left(\psi^{m+1}+z\right)\nabla v\,\mathrm{d}V \\
&= -\langle K\left(\Theta(\psi^{m+1})\right)\nabla(\psi^{m+1}+z), \nabla v\rangle_{V_i},
\end{aligned}$$

in which we use integration by parts for higher dimensions and the fact that the surface integral over ∂V_i is 0. We remark that, in the context of the Richards equation, the existence and uniqueness of its weak solution have been rigorously established and carefully studied [15, 16, 17], setting up the theoretical foundation for developing an efficient solution algorithm to solve the discretized Richards equation numerically.

2 Literature Review

Among existing solution algorithms for the Richards equation, methods based on finite difference and finite element discretizations [18, 19] have been studied and implemented the most [20]. However, these methods often face challenges when handling large-scale problems and suffer from instability issues such as oscillations [21]. Recently, Ireson et al. [22] used the method of lines to convert the 1-D Richards equation into an ordinary differential equation (ODE), which was then solved by finite difference method. Despite these advancements, finite difference- and finite element-based methods generally require high mesh resolution to satisfy the local equilibrium condition [23, 24, 25]. Furthermore, they tend to fail to preserve global mass balance [26] and other important underlying physical relations connecting soil moisture, pressure head, and water flux [27, 28], which further deteriorates solution accuracy.

Meanwhile, finite volume discretization method (FVM) has demonstrated promising potential in achieving high solution accuracy and preserving the mass conservation when solving the Richards equation [29]. For example, Lai and Ogden [30] obtained a family of mass-conservative finite volume predictor-corrector solutions for the 1-D Richards equation. A second-order accurate monotone FVM was also proposed by Misiats and Lipnikov [16] for solving the 1-D Richards equation. Several other FVM-based numerical methods have also been developed to solve 1-D and 2-D Richards equations [31, 32, 33]. However, like finite difference- and finite element-based methods, conventional

FVM typically converts the discretized Richards equation into a large, stiff matrix equation, which can be challenging to solve.

Thus, instead of following the standard practice of converting the discretized equations into a matrix equation, methods such as the linearization scheme attempt to solve the discretized Richards equations iteratively. A few notable linearization scheme implementations include Bergamaschi and Putti [34] and Pop et al. [35]. In static L-scheme, a fixed linearization parameter L is used for all time steps and discretized cells. However, choosing an appropriate static value of L is not straightforward, as soil moisture content and pressure head exhibit strong spatiotemporal patterns. To addresss this drawback, Mitra and Pop [36] modified the static L-scheme for solving 1-D nonlinear diffusion equations by allowing L to be adaptive with respect to space and time. More recently, Albuja and Avila [37] modified the Newton-type scheme and proposed a new linearization scheme for solving the 1-D Richards equation with guaranteed global convergence. Nevertheless, some of the existing adaptive L-schemes have been reported to suffer from numerical oscillations [38]. And so far, no adaptive L-scheme has been established for solving the 3-D Richards equation.

On the other hand, as a promising approach to enhance the accuracy and stability of numerical algorithms, stochastic methods such as the random walk model have been introduced and incorporated in discretization-based methods. In the random walk model, particles, in this case water molecules, can move to their neighboring discretized cells following certain probabilities. Furthermore, random walks are memoryless, meaning that the current movements of particles are independent of their past movements. These properties makes the random walk model attractive for solving various transport problems [39], including the Richards equation (e.g., [40]). Having said that, in conventional random walk model, only one particle can move at a time. To overcome this, the global random walk (GRW) model was recently proposed [41, 42], allowing all particles to move and/or stay at the same time. The GRW model has been successfully implemented to model groundwater transport [41, 43], diffusion process [44, 43], and 1-D and 2-D Richards equation [42]. Furthermore, the GRW model has been recently coupled with the L-scheme for the first time [42] using a fixed L. Specifically, a 1-D GRW scheme for the Richards equation was proposed by Suciu et al.[42] as a randomization method that approximated the solution of the finite difference scheme by the distribution of random walkers on a regular lattice. It was shown that the ensemble average of the 1-D GRW scheme is the finite difference scheme. Meanwhile, to implement the 2-D GRW scheme [42], the term $a/\mathcal{N}$ was fixed to 1 (where in [42], a is a unit conversion factor and $\mathcal{N}$ is the total number of particles). This resulted in a deterministic GRW scheme that was identical to the finite-differnce scheme. Thus, both GRW schemes are derived from the finite difference method by incorporating the relation that the pressure head in each discretized cell at any time step is proportional to the number of particles in that cell and time step.

3 Motivation and Scope of Our Approach

While the original GRW scheme has a solid theoretical foundation and clear connection with finite difference scheme [41], when it comes to a realistic simulation setting and actual implementation, due to practical limitations such as truncation errors and constrained computing power, the pressure head and number of particles solutions obtained by an actual computer may not always be strictly proportional to each other. In fact, when experimenting the GRW scheme, we observe that different choices of linearization parameter L, total iteration count S, and total number of particles N lead to slightly different solutions, among which the proportionality relation no longer strictly follows. To systematically capture and address this discrepancy between GRW theory and implementation, we propose to relax the proportionality relation between the pressure head and number of particles

solutions and approximate the actual relation using a data-driven approach. The resulting new solution algorithm, which we named as the data-driven global random walk (DRW) algorithm, was recently developed to solve 1-D and 3-D Richards equations [27, 28]. This preliminary prototype achieves remarkable performance in terms of solution accuracy and the ability to capture underlying physics. In particular, we find that the use of neural networks to characterize the relation between pressure head and the number of water molecules observed in a realistic computational setting significantly enhances the performance of GRW scheme during implementation. Despite these promising preliminary results, many of the technical details and rigorous analyses were not discussed in [27, 28] due to space limits. Therefore, in this article, we fill in these missing gaps and introduce several innovative approaches to further improve the numerical accuracy and computational efficiency of our DRW algorithmic framework. Specifically, we will formally present our DRW algorithm for solving any d-dimensional Richards equation, provide rigorous theoretical justification of the convergence behavior of our DRW algorithm, and conduct comprehensive case studies and in-depth analyses of 1- through 3-D benchmark problems. Some of the new contributions to the previously established DRW framework are highlighted as follows:

- We formalize our prior works [27, 28] and present a rigorous, generalized framework for any d-dimensional ($d = 1, 2, 3$) Richards equation, and demonstrate that the new DRW algorithm can be versatilely adopted to modeling different realistic scenarios (e.g., layered soil, actual precipitation).
- We introduce a "coarse-to-fine" approach to enhance the solution accuracy of our DRW algorithm without requiring a large amount of high-accuracy, fine-mesh training data. We demonstrate that this coarse-to-fine approach maintains a good balance between computational efficiency and solution accuracy.
- We show that, by synergistically integrating our novel adaptive L-scheme, global random walk, and neural network, our new DRW algorithmic framework significantly enhances the ability of FVM discretization in preserving the underlying physical relationships and mass conservation associated with the Richards equation.

Apart from the motivations above, we remark that our DRW framework is attractive also because it provides new opportunities for integrating physics-based modeling (i.e., the Richards equation) with online in situ soil moisture measurements for sensor-driven soil monitoring and precision agriculture applications. Commercial soil moisture sensors, whether directly or indirectly measure soil moisture content, are not perfect and instrumental errors will always be present in their measurements. Furthermore, environmental factors (e.g., wind, temperature, evapotranspiration) and imperfect installation and maintenance will bring additional uncertainties to online soil sensing data [45]. Thus, the data-driven approach in our DRW algorithm makes it amenable to leverage uncertainty-embedded dataset produced by in situ soil sensors in field environment for predictive modeling of root-zone soil moisture profiles.

We organize the subsequent sections of the paper as follows. In Section 4, we derive the FVM-based adaptive L-scheme formulation and prove its global convergence. Then, in Section 5, we incorporate the adaptive L-scheme formulation in the DRW algorithm. To compare the performance of our DRW algorithm with state-of-the-art solvers, we conduct case studies and discuss key findings in Section 6. Finally, we summarize the results and discuss future directions in Section 8, followed by disclosing the computational tools, methods, codes used in conducting the case studies in Section 9.

4 Adaptive L-Scheme of Discretized Richards Equation

In this section, we will formally introduce the adaptive L-scheme formulation of the Richards equation discretized by FVM. We will also derive sufficient conditions for the linearization parameter to ensure convergence. Furthermore, we will analyze the convergence behavior of the resulting sequence of solutions $\{\psi_i^{m+1,s}\}_s$, where s is the iteration count.

4.1 Adaptive L-scheme for the Richards Equation

We present the key steps involved in the discretization of the Richards equation using FVM, followed by introducing adaptive L-scheme to solve the discretized equation iteratively. First, integrating both sides of Equation (1) over V gives:

$$\int_V \left[\partial_t \theta(\psi) + S(\psi)\right] \mathrm{d}V = \int_V \nabla \cdot \left[K(\theta)\nabla(\psi + z)\right] \mathrm{d}V. \tag{4}$$

Then, we apply the divergence theorem to Equation (4), which converts the volume integral on the RHS into a surface integral:

$$\left[\partial_t \theta(\hat{\psi}) + S(\hat{\psi})\right]_{\hat{\psi}\in V} \mathrm{vol}(V) = \oint_{S_V} K(\theta)\nabla(\psi + z) \cdot \mathbf{n} \, \mathrm{d}S_V, \tag{5}$$

where $\mathrm{vol}(V)$ is the volume of V, S_V is the surface of V and $\mathbf{n}$ is the outward pointing unit normal to the boundary ∂V. The common surface shared by cell V_i and cell V_j is denoted as $\omega_{i,j}$. With this, we can rewrite the operator $K(\cdot)\nabla(\cdot)$ and the outward pointing unit normal vector $\mathbf{n}$ on $\omega_{i,j}$ as $\left[K(\cdot)\nabla(\cdot)\right]_{\omega_{i,j}}$ and $\mathbf{n}_{\omega_{i,j}}$, respectively. After FVM discretization, we obtain the discretized version of Equation (5) as:

$$\partial_t \theta_i \mathrm{vol}(V_i) + S(\psi_i)\mathrm{vol}(V_i) = \sum_{j\in\mathcal{N}_i} \left[K(\theta)\nabla(\psi + z)\right]_{\omega_{i,j}} \cdot \mathbf{n}_{\omega_{i,j}} A_{\omega_{i,j}} \qquad \forall i = 1, \ldots, N, \tag{6}$$

where $\partial_t \theta_i$ refers to the time derivative $\partial_t \theta(\psi_i)$ in cell V_i, $\mathcal{N}_i$ denotes the index set of all the neighboring cells sharing a common surface with V_i, and $A_{\omega_{i,j}}$ is the area of surface $\omega_{i,j}$.

In static L-scheme, for every cell V_i and time step $m+1$, one would add the term $L(\psi_i^{m+1,s+1} - \psi_i^{m+1,s})$ to either side of Equation (6), so that the Richards equation can be solved in an iterative manner and the pressure head solution ψ_i^m is the fixed-point solution of this iterative procedure. Since L is a static constant, a trial-and-error procedure is typically required to obtain an appropriate L value that avoids convergence issues. Not only is this search procedure tedious to implement, the solutions obtained are also less accurate most of the time as we will show in Section 6.1. Thus, inspired by previous works [36, 37], we proposed a novel adaptive L-scheme that replaces the static L with $L_i^{m+1,s}$, which adjusts itself for each specific discretized cell, time step, and iteration count. We then introduce the term $L_i^{m+1,s}(\psi_i^{m+1,s+1} - \psi_i^{m+1,s})$ to the LHS of Equation (6), which leads to:

$$\begin{aligned} \psi_i^{m+1,s+1} =& \psi_i^{m+1,s} + \frac{1}{L_i^{m+1,s}} \sum_{j\in\mathcal{N}_i} \left[K(\theta)\nabla(\psi + z)\right]_{\omega_{i,j}}^{m+1,s} \cdot \mathbf{n}_{\omega_{i,j}} A_{\omega_{i,j}} \\ & - \frac{1}{L_i^{m+1,s}} \left[\partial_t \theta_i^{m+1,s} + S(\psi_i^{m+1,s})\right] \mathrm{vol}(V_i), \end{aligned} \tag{7}$$

By discretizing $\partial_t \theta_i^{m+1,s}$ using implicit Euler scheme as $\frac{\theta(\psi_i^{m+1,s})-\theta(\psi_i^m)}{\Delta t}$, we can obtain the adaptive L-scheme of the FVM-discretized Richards equation:

$$\begin{aligned}\psi_i^{m+1,s+1} = \psi_i^{m+1,s} &+ \frac{1}{L_i^{m+1,s}} \sum_{j\in\mathcal{N}_i} K_{\omega_{i,j}}^{m+1,s} \frac{(\psi+z)_j^{m+1,s} - (\psi+z)_i^{m+1,s}}{\mathrm{dist}(V_j,V_i)} \mathbf{e}\cdot\mathbf{n}_{\omega_{i,j}} A_{\omega_{i,j}} \\ &- \frac{1}{L_i^{m+1,s}} \left[\frac{\theta(\psi_i^{m+1,s}) - \theta(\psi_i^m)}{\Delta t} + S(\psi_i^{m+1,s}) \right] \mathrm{vol}(V_i),\end{aligned} \tag{8}$$

where $\mathbf{e} = (1,1,1)$ for the standard 3-D Cartesian coordinate system, and $\mathrm{dist}(\cdot,\cdot)$ represents the Euclidean distance function.

4.2 Choice of Adaptive Linearization Parameter

In adaptive L-scheme, one needs to automatically select an appropriate linearization parameter. We observe that $L_i^{m+1,s}$ needs to be sufficiently large because otherwise, the RHS of Equation (8) could approach infinity, which affects the convergence of adaptive L-scheme. In addition, the choice of $L_i^{m+1,s}$ can impact the accuracy of solutions. Considering these aspects, we propose a new way of selecting appropriate linearization parameter. First, to prevent $L_i^{m+1,s}$ from being too close to 0, we impose a user-specified global lower bound L_0:

$$L_i^{m+1,s} \geq L_0.$$

In addition, we must ensure that $\frac{||\psi_i^{m+1,s+1}-\psi_i^{m+1,s}||}{||\psi_i^{m+1,s}||}$ is no greater than a prespecified tolerance ρ. In other words, we have:

$$\frac{||\psi_i^{m+1,s+1} - \psi_i^{m+1,s}||}{||\psi_i^{m+1,s}||} = \frac{\frac{1}{L_i^{m+1,s}}||g_i^{m+1,s}||}{||\psi_i^{m+1,s}||} \leq \rho \qquad \forall s = 1,\ldots,S,$$

where S is the user-specified total number of iterations for convergence, ρ should be no less than the overall tolerance of convergence ϵ (to be discussed in Section 4.3), and:

$$\begin{aligned}g_i^{m+1,s} = &\sum_{j\in\mathcal{N}_i} K_{\omega_{i,j}}^{m+1,s} \frac{(\psi+z)_j^{m+1,s} - (\psi+z)_i^{m+1,s}}{\mathrm{dist}(V_j,V_i)} \mathbf{e}\cdot\mathbf{n}_{\omega_{i,j}} A_{\omega_{i,j}} \\ &- \frac{\theta(\psi_i^{m+1,s}) - \theta(\psi_i^m)}{\Delta t}\mathrm{vol}(V_i) - S(\psi_i^{m+1,s})\mathrm{vol}(V_i).\end{aligned}$$

This implies that:

$$L_i^{m+1,s} \geq \frac{(1+\rho)||g_i^{m+1,s}||}{\rho||\psi_i^{m+1,s}||} \qquad \forall s = 1,\ldots,S, \tag{9}$$

whose RHS can be explicitly determined from the results of the previous iteration. In actual implementation, we choose $L_i^{m+1,s}$ based on:

$$L_i^{m+1,s} = \max\left\{ L_0, \frac{(1+\rho)||g_i^{m+1,s}||}{\rho||\psi_i^{m+1,s}||} \right\} \qquad \forall s = 1,\ldots,S. \tag{10}$$

Equation (10) gives an automated way to choose the linearization parameter. Meanwhile, we can monitor the sensitivity of solutions obtained by adaptive L-scheme and make sure that the

solutions do not change drastically with respect to small perturbations. To do this, following Zarba [46] and Celia and Zarba [47], we explicitly write down Equation (8) for all discretized cells in the form of a matrix equation:

$$\mathbf{A}\mathbf{x}^{m+1,s} = \mathbf{b}, \tag{11}$$

where the ith element of vector $\mathbf{x}^{m+1,s}$ is $x_i^{m+1,s} = \psi_i^{m+1,s+1} - \psi_i^{m+1,s}$, which corresponds to cell V_i. Here, it is worth mentioning that Equation (11) is not used for solving Equations (8) as it is an explicit scheme. Rather, it is used for analyzing the properties of the scheme after $x_i^{m+1,s}$ solutions are obtained by solving Equation (8). For example, to evaluate the choice of $L_i^{m+1,s}$, we can calculate the condition number of $\mathbf{A}$ based on the solutions obtained from the chosen $L_i^{m+1,s}$. If the condition number is larger than a user-specified threshold, we will update L_0 in Equation (10) so that the condition number drops below the threshold. For 1-D problems, Zarba [46] showed that $\mathbf{A}$ is a $N \times N$ asymmetric tridiagonal matrix. In this case, the condition number of $\mathbf{A}$ can be determined by calculating its eigenvalues. On the other hand, for 2-D and 3-D problems, $\mathbf{A}$ is a rectangular matrix, so that singular value decomposition will be used to determine its condition number.

4.3 Convergence Analysis of Adaptive L-Scheme

Here, we study the the convergence behavior of our adaptive L-scheme. From Bergamaschi and Putti [34], as $L_i^{m+1,s}$ approaches $\dot{\theta}(\psi_i^{m+1,s})$, our adaptive L-scheme essentially becomes the Newton's scheme, which exhibits quadratic convergence. In general, the convergence of our adaptive L-scheme formulation of Equation (8) is characterized in Theorem 4.1. To show this, the idea is to leverage Definition 1.1 and find $\psi_i^{m+1,s+1} \in H_0^1(V)$ given ψ_i^m, $\psi_i^{m+1,s} \in H_0^1(V)$ such that:

$$\begin{aligned}\left\langle \theta(\psi^{m+1,s+1}) - \theta(\psi^m), v \right\rangle_{V_i} + \Delta t L_i^{m+1,s} \left\langle \psi_i^{m+1,s+1} - \psi_i^{m+1,s}, v \right\rangle_{V_i} + \left\langle S(\psi^{m+1,s+1}), v \right\rangle_{V_i} \\ = -\Delta t \left\langle K\left(\theta(\psi^{m+1})\right) \nabla(\psi^{m+1,s+1} + z), \nabla v \right\rangle_{V_i}\end{aligned} \tag{12}$$

holds for any $v \in H_0^1(V)$. We remark that, unlike previous proofs [36, 37] that are based on several restrictive assumptions, our proof of convergence follows a distinct approach that is intuitive and flexible, as it does not involve any additional assumptions other than the two observations below.

Theorem 4.1. *$\{\psi_i^{m+1,s}\}_s$ converges to $\psi_i^{m+1} \in H_0^1(V)$ for $m = 0, 1, \ldots, \lceil \frac{T}{\Delta t} \rceil - 1$ and $i = 1, \ldots, N$.*

Proof. First, we state two general observations for water infiltration in soil:

Observation 1: There exists a scaling factor $0 < \gamma_{s,0} < \infty$ such that $||\psi_i^{m+1,s+1} - \psi_i^{m+1}|| < \gamma_{s,0}||\psi_i^{m+1,s} - \psi_i^{m+1}||$. In other words, $||\psi_i^{m+1,s+1} - \psi_i^{m+1}|| < +\infty$. This observation has been previously proven by [36] in the context of solving nonlinear diffusion problems using another numerical scheme.

Observation 2: $\dot{\theta}(\psi) = \frac{\mathrm{d}\theta}{\mathrm{d}\psi}|_{\psi_i^{m+1,s}} > 0$, which is valid in most WRC models (see Table 1). Similarly, $\dot{S}(\psi) = \frac{\mathrm{d}S}{\mathrm{d}\theta}\frac{\mathrm{d}\theta}{\mathrm{d}\psi}|_{\psi_i^{m+1,s}} \geq 0$ in the region between the start and optimal root water extraction.

First, we subtract Equation (3) from Equation (12) and obtain:

$$\begin{aligned}&\left\langle \theta(\psi^{m+1,s+1}) - \theta(\psi^{m+1}), v \right\rangle_{V_i} + \Delta t L_i^{m+1,s} \left\langle \psi_i^{m+1,s+1} - \psi_i^{m+1,s}, v \right\rangle_{V_i} \\ &+ \left\langle S(\psi^{m+1,s+1}) - S(\psi^{m+1}), v \right\rangle_{V_i} = -\Delta t \left\langle K\left(\theta(\psi^{m+1})\right) \nabla(\psi^{m+1,s+1} - \psi^{m+1}), \nabla v \right\rangle_{V_i}.\end{aligned} \tag{13}$$

Since $\psi_i^{m+1,s+1} - \psi_i^{m+1,s} = (\psi_i^{m+1,s+1} - \psi_i^{m+1}) - (\psi_i^{m+1,s} - \psi_i^{m+1})$, Equation (13) can be rewritten as:

$$\begin{aligned}\Delta t L_i^{m+1,s}\Big\langle \psi_i^{m+1,s} - \psi_i^{m+1}, v\Big\rangle_{V_i} &= \big\langle \theta(\psi^{m+1,s+1}) - \theta(\psi^{m+1}), v\big\rangle_{V_i} + \big\langle S(\psi^{m+1,s+1}) - S(\psi^{m+1}), v\big\rangle_{V_i} \\ \Delta t L_i^{m+1,s}\Big\langle \psi_i^{m+1,s+1} - \psi_i^{m+1}, v\Big\rangle_{V_i} &+ \Delta t \big\langle K\left(\theta(\psi^{m+1})\right)\nabla(\psi^{m+1,s+1} - \psi^{m+1}), \nabla v\big\rangle_{V_i}.\end{aligned} \tag{14}$$

Let $v = \psi_i^{m+1,s+1} - \psi_i^{m+1}$, then from Observation 1, the LHS of Equation (14) can be bounded by:

$$\begin{aligned}\Delta t L_i^{m+1,s}\Big\langle \psi_i^{m+1,s} - \psi_i^{m+1}, v\Big\rangle_{V_i} &= \Delta t L_i^{m+1,s}(\psi_i^{m+1,s} - \psi_i^{m+1})(\psi_i^{m+1,s+1} - \psi_i^{m+1})\mathrm{vol}(V_i) \\ &< \gamma_{s,0}\Delta t L_i^{m+1,s}(\psi_i^{m+1,s} - \psi_i^{m+1})^2\mathrm{vol}(V_i) \\ &= \gamma_{s,0}\Delta t L_i^{m+1,s}||\psi_i^{m+1,s} - \psi_i^{m+1}||^2.\end{aligned} \tag{15}$$

Similarly, for the third term on the RHS of Equation (14), we have:

$$\Delta t L_i^{m+1,s}\Big\langle \psi_i^{m+1,s} - \psi_i^{m+1}, v\Big\rangle_{V_i} = \Delta t L_i^{m+1,s}||\psi_i^{m+1,s+1} - \psi_i^{m+1}||^2. \tag{16}$$

By the mean value theorem, the first and second terms on the RHS of Equation (14) can be written as:

$$\begin{aligned}\big\langle \theta(\psi^{m+1,s+1}) - \theta(\psi^{m+1}), v\big\rangle_{V_i} &= \dot\theta(\xi_\theta)\big\langle \psi^{m+1,s+1} - \psi^{m+1}, v\big\rangle_{V_i} = \dot\theta(\xi_\theta)||\psi_i^{m+1,s+1} - \psi_i^{m+1}||^2; \\ \big\langle S(\psi^{m+1,s+1}) - S(\psi^{m+1}), v\big\rangle_{V_i} &= \dot S(\xi_S)||\psi_i^{m+1,s+1} - \psi_i^{m+1}||^2;\end{aligned} \tag{17}$$

for $\xi_\theta, \xi_S \in (\psi_i^{m+1,s+1}, \psi_i^{m+1})$.

Lastly, for the last term on the RHS of Equation (14), we have:

$$\Delta t\big\langle K\left(\theta(\psi^{m+1})\right)\nabla(\psi^{m+1,s+1} - \psi^{m+1}), \nabla v\big\rangle_{V_i} = \Delta t K\left(\theta(\psi_i^{m+1})\right)||\nabla\psi_i^{m+1,s+1} - \nabla\psi_i^{m+1}||^2. \tag{18}$$

Combining Equations (15) through (18) leads to:

$$\begin{aligned}||\psi_i^{m+1,s+1} - \psi_i^{m+1}|| &< \sqrt{\frac{\gamma_{s,0}L_i^{m+1,s}\Delta t}{L_i^{m+1,s}\Delta t + \dot\theta(\xi_\theta) + \dot S(\xi_S)}}||\psi_i^{m+1,s} - \psi_i^{m+1}|| \\ &= \sqrt{\gamma_{s,0}\left(1 - \frac{\dot\theta(\xi_\theta) + \dot S(\xi_S)}{L_i^{m+1,s}\Delta t + \dot\theta(\xi_\theta) + \dot S(\xi_S)}\right)}||\psi_i^{m+1,s} - \psi_i^{m+1}|| \\ &= \sqrt{\gamma_{s,0}(1 - a_s)}||\psi_i^{m+1,s} - \psi_i^{m+1}||,\end{aligned} \tag{19}$$

where $a_s \coloneqq \frac{\dot\theta(\xi_\theta) + \dot S(\xi_S)}{L_i^{m+1,s}\Delta t + \dot\theta(\xi_\theta) + \dot S(\xi_S)} \in (0,1)$ and is a constant for a given iteration s. Let $\gamma_{s,1} \coloneqq \sqrt{\gamma_{s,0}(1-a_s)}$ and replace $\gamma_{s,0}$ in Observation 1 by $\gamma_{s,1}$. One can repeat the derivations above to obtain $||\psi_i^{m+1,s+1} - \psi_i^{m+1}|| < \sqrt{\gamma_{s,1}(1-a_s)}||\psi_i^{m+1,s} - \psi_i^{m+1}||$. From here, we can further define $\gamma_{s,2} = \sqrt{\gamma_{s,1}(1-a_s)}$, and so on. Overall, this recursion leads to:

$$\gamma_{s,n} = \gamma_{s,0}^{\frac{n}{2}}(1-a_s)^{\frac{n(n+1)}{4}},$$

which, for large enough n, becomes strictly less than 1. We denote such a $\gamma_{s,n} < 1$ as the scaling factor for iteration s, γ_s, such that $||\psi_i^{m+1,s+1} - \psi_i^{m+1}|| < \gamma_s ||\psi_i^{m+1,s} - \psi_i^{m+1}||$.

With this, one can show that for a given tolerance $\epsilon > 0$, there exists $S \in \mathbb{N}^+$ such that:

$$||\psi_i^{m+1,s+1} - \psi_i^{m+1}|| < \gamma_s ||\psi_i^{m+1,s} - \psi_i^{m+1}|| < \gamma_s \gamma_{s-1} ||\psi_i^{m+1,s-1} - \psi_i^{m+1}|| < \cdots$$
$$< \prod_{k=1}^{s} \gamma_k ||\psi_i^{m+1,1} - \psi_i^{m+1}|| < \left(\max_{1 \le k \le s} \gamma_k \right)^s ||\psi_i^{m+1,1} - \psi_i^{m+1}|| < \epsilon \quad \forall s \ge S,$$

which is consistent with the convergence criterion. This completes the proof. □

5 Data-driven Global Random Walk (DRW) Algorithm

Once the adaptive L-scheme for the FVM-discretized Richards equation is established, we incorporate it in our DRW algorithm to enhance the solution accuracy and mass conservation performance. Let $n_i^{m,s}$ be the number of water molecules in cell V_i at iteration step s and time step m, and $\delta n_{i,j}^{m,s}$ be the number of particles moving from the cell V_i to the neighboring cell $j \in \mathcal{N}_i$. The global random walk (GRW) incorporates the following generalized relation for d-dimensional case ($d = 1, 2, 3$) [42]:

$$n_i^{m+1,s} = \delta n_{i,i}^{m+1,s} + \sum_{j \in \mathcal{N}_i} \delta n_{j,i}^{m+1,s}. \tag{20}$$

For instance, for the 1-D case, we have $n_i^{m+1,s} = \delta n_{i,i}^{m+1,s} + \delta n_{i+1,i}^{m+1,s} + \delta n_{i-1,i}^{m+1,s}$ for every $i = 1, \ldots, N$ [42] (see Appendix A for explicit formulations for 1-D through 3-D cases). Thus, as long as the relationship between pressure head $\psi_i^{m,s}$ and the number of particles $n_i^{m,s}$ is established, Equation (20) can be incorporated into our adaptive L-scheme formulation of Equation (8) to solve the Richards equation iteratively. As pointed out earlier, the original GRW scheme [42] is derived from the finite difference scheme using the linear relation that $\psi_i^{m,s}$ is proportional to $n_i^{m,s}$. Nevertheless, in actual implementation of the numerical scheme, we observe deviation of this linear relation due to numerical errors and computational constraints in realistic simulation settings. Furthermore, with modern in situ soil sensing technologies, one has access to a large volume of real-time sensor data that are subject to various sources of measurement uncertainties. Considering these practical issues, we propose to relax the linear relation used to derive the GRW scheme [42] and introduce data-driven techniques to the GRW framework to enhance its adaptability, robustness and accuracy.

Therefore, in this work, we extend the original GRW framework by introducing a data-driven approach to directly learn the observed relation between $\psi_i^{m,s}$ and $n_i^{m,s}$ under the influence of numerical errors and realistic simulation settings. The resulting data-driven global random walk (DRW) algorithm uses one neural network denoted as $\hat{f}_{\text{NN}}$ to learn the map $n_i^{m,s} \to \psi_i^{m,s}$ and another neural network denoted as $\hat{f}_{\text{NN}}^{-1}$ to learn the inverse map $\psi_i^{m,s} \to n_i^{m,s}$, both of which are subject to numerical errors during implementation. And two separate steps, namely neural network training and solution process, are involved in the DRW algorithm.

5.1 Neural Network Training

A neural network is capable of approximating any function provided that it contains enough neurons [48, 49]. During neural network training step, for each cell V_i and time step m, we obtain $n_i^{m,S}$ solution from our in-house GRW solver, which is built upon the original GRW framework established by Suciu et al. [42, 50] and incorporates the linear relation between number of particles and pressure

head into the static L-scheme of the FVM-discretized Richards equation. Here, S is the user-specified total iteration number. The corresponding $\psi_i^{m,S}$ solutions are obtained separately from the static L-scheme version of Equation (8), which is an explicit scheme and does not involve any particles. The resulting set of solution pairs, $\left\{\left(\psi_i^{m,S}, n_i^{m,S}\right)\right\}_{i,m}$, form a set of original "reference solutions". In actual implementation, we obtain multiple sets of original reference solutions by selecting multiple total iteration numbers $(S_1, \ldots, S_p)$ and linearization parameters $(L_1, \ldots, L_r)$ that cover their ranges expected during the actual solution process. These sets of original reference solutions, which are $\left\{\left(\psi_i^{m,S_1}, n_i^{m,S_1}\right)|_{L_1}\right\}_{i,m}, \ldots, \left\{\left(\psi_i^{m,S_p}, n_i^{m,S_p}\right)|_{L_r}\right\}_{i,m}$, are combined to form a larger set to perform data augmentation. If this set is already very large (as in the case of the 1-D benchmark problem to be discussed in Section 6.1), then we may randomly select a subset of reference solutions to perform data augmentation. It turns out that this random selection does not affect the accuracy of our DRW algorithm but accelerates neural network training.

Next, to apply data augmentation, we introduce Gaussian noise $Z_q \sim \mathcal{N}(0, \sigma_q^2)$ with different variances $\sigma_1^2, \ldots, \sigma_Q^2$ to each and every element in the reference solution set obtained previously. After data augmentation, the resulting expanded set of reference solutions, $\left\{(\psi_i^{m,S_1} + Z_p, n_i^{m,S_1} + Z_q)|_{L_1}\right\}_{i,m,q}, \ldots, \left\{(\psi_i^{m,S_p} + Z_q, n_i^{m,S_p} + Z_q)|_{L_r}\right\}_{i,m,q}$, is denoted as $\mathcal{S}$ and will be used for neural network training. This data augmentation step not only increases the size of the training dataset, but also reflects the characteristics of actual soil sensing data, which are subject to various measurement uncertainties due to instrumental error, environmental uncertainties, and imperfect installation and maintenance. In Section 6.1, we will show that introducing Gaussian noise can greatly reduce the biases of reference solutions and enhance generalization performance [51], thereby significantly improving the accuracy of numerical solutions.

We remark that, depending on the problem settings, the desired choices of optimal optimizer, number of hidden layers, and activation functions can vary. Based on our extensive research and hyperparameter tuning, we find that a simple three-layer neural network with 256 neurons in each layer achieves the best performance for most 1-D through 3-D problems compared to other more complex neural network architectures (e.g., LSTM). Also, we find that stochastic gradient decent (SGD) optimizer often outperforms others (e.g., Adam or RMSProp). This simple neural network structure makes its training much less computationally expensive compared to state-of-the-art neural PDE solvers [52, 53].

5.2 Solution Process

When neural network training is complete, the trained $\hat{f}_{\text{NN}}^{-1}$ and $\hat{f}_{\text{NN}}$ can then be incorporated into the adaptive L-scheme of Equation (8) to derive the following DRW formulation of the FVM-discretized Richards equation:

$$n_i^{m+1,s+1} = n_i^{m+1,s} + \frac{1}{L_i^{m+1,s}} \sum_{j \in \mathcal{N}_i} K_{\omega_{i,j}}^{m+1,s} \mathbf{e} \cdot \mathbf{n}_{\omega_{i,j}} \frac{n_j^{m+1,s} - n_i^{m+1,s}}{\text{dist}(V_j, V_i)} A_{\omega_{i,j}} + \hat{f}_{\text{NN}}^{-1}(J), \tag{21}$$

where $J = \frac{1}{L_i^{m+1,s}} \sum_{j \in \mathcal{N}_i} K_{\omega_{i,j}}^{m+1,s} \mathbf{e} \cdot \mathbf{n}_{\omega_{i,j}} \frac{z_j - z_i}{\text{dist}(V_j, V_i)} A_{\omega_{i,j}} - \frac{1}{L_i^{m+1,s}} \left(\frac{\theta_i^{m+1,s} - \theta_i^m}{\Delta t} + S(\psi_i^{m+1,s}) \right) \text{vol}(V_i)$. To solve Equation (21), we will adopt a similar strategy as in Equation (10) to adaptively select the linearization parameter $L_i^{m+1,s}$. Additional relations needed to be coupled with Equation (21) to solve for the particle distribution $n_i^{m+1,s+1}$ are given in Appendix A, and the resulting explicit formulations for 1-D through 3-D are provided in Appendix B.

To start the solution process, we assign a large number of particles (e.g., 10^{10}) per unit of pressure head as specified in the initial and boundary conditions. This gives the initial distribution

of particles in the control volume at $m = 0$. Next, for each new time step $m + 1$, by leveraging the trained neural networks $\hat{f}_{\text{NN}}^{-1}$ and $\hat{f}_{\text{NN}}$, the distribution of these particles can be determined by simultaneously solving Equation (21) along with additional relations in Appendices A and B. Note that the iterative usage of $\hat{f}_{\text{NN}}$ is implicitly implied in the DRW scheme, as the term J in Equation (21) contains $\psi_i^{m+1,s}$ that must be evaluated by applying $\hat{f}_{\text{NN}}$ on $n_i^{m+1,s}$. Overall, this iterative solution process produces a sequence $\{n_i^{m+1,s}\}$ for time step $m + 1$ and cell V_i. To monitor the convergence of $\{n_i^{m+1,s}\}$, we define the relative error RE_s as:

$$\text{RE}_s \coloneqq \left| \frac{n_i^{m+1,s+1} - n_i^{m+1,s}}{n_i^{m+1,s+1}} \right|. \tag{22}$$

Once RE_s is below a user-specified tolerance tol (typically in the order of 10^{-6}), we declare convergence of $\{n_i^{m+1,s}\}_s$ to n_i^{m+1}. With this, we can determine the converged ψ_i^{m+1} using $\hat{f}_{\text{NN}}$, followed by obtaining other physical quantities such as soil moisture content θ_i^{m+1} and $\mathbf{q}_i^{m+1}$ from the WRC and HCF models (Table 1) and Equation (1). The entire solution process then repeats itself in the next time step until $m = \lceil \frac{T}{\Delta t} \rceil - 1$.

Furthermore, it is worth mentioning that, when neural network training for a specific problem setting (e.g., boundary condition and initial condition) is complete, the trained neural networks can be saved as a pretrained model. As we encounter a new problem setting, the pretrained model provides a strong starting point that can be quickly trained with a small number of epochs (typically no more than 100) before it can be deployed to solve the new problem. The use of pretrained model is a well-established technique in machine/deep learning for leveraging knowledge learned from (large) datasets, reducing the need for extensive training data and computation, and enabling faster deployment and improved performance in new tasks through fine-tuning.

5.3 Convergence of DRW Algorithm

To show the convergence of our DRW algorithm, we extend Theorem 4.1 and investigate the convergence behavior of stochastic gradient descent (SGD) for neural network realizations of $\hat{f}_{\text{NN}}$ and $\hat{f}_{\text{NN}}^{-1}$. Overall, a similar convergence property for the sequence $\{n_i^{m+1,s}\}_s$ can be established as:

Theorem 5.1. *$\{n_i^{m+1,s}\}_s$ converges to n_i^{m+1} for $m = 0, 1, \ldots, \lceil \frac{T}{\Delta t} \rceil - 1$ and $i = 1, \ldots, N$.*

Proof. See Appendix C for the complete proof. □

6 Case Studies

We have now completed the formulation of our DRW-based solution algorithm for the Richards equation. In this section, we systematically validate our DRW numerical framework on selected 1-D through 3-D benchmark problems modified from the literature [19, 54, 55, 56, 45]. Specifically, we extensively study the 1-D benchmark problem of Celia et al. [19] to demonstrate the need and benefits of adaptive L-scheme, data-driven global random walk, and data augmentation. Also, using this problem as a benchmark, we demonstrate the accuracy of our solution algorithm with respect to state-of-the-art solvers. In the 1-D layered soil case study proposed by Berardi et al. [56], we show that our solution algorithm is capable of handling discontinuities in soil properties and modeling the infiltration process through the interface of two different soils. In the 2-D case study adopted from Gąsiorowski and Kolerski [54], we show that our DRW algorithm can better satisfy mass balance embedded in the Richards equation. In the 3-D case study adopted from Tracy [55] in which an

analytical solution to the Richards equation exists, we show that our DRW algorithm produces much more accurate solutions compared to GRW solvers. Finally, we study a 3-D problem adopted from Orouskhani et al. [45] featuring an actual center-pivot system and validate the accuracy and robustness of our DRW algorithm in modeling real-world precipitation and irrigation scenarios for a long period of time.

6.1 A 1-D Benchmark Problem

Here, we study the 1-D benchmark problem over a 40 cm deep soil presented by Celia et al. [19]. The HCF and WRC adopt the model of Haverkamp et al. [8] (see Table 1). And the parameters are listed in Table 6.1. The initial condition is given by $\psi(z, 0) = -61.5$ cm, whereas the two boundary conditions are $\psi(40 \text{ cm}, t) = -20.7$ cm, $\psi(0, t) = -61.5$ cm, respectively [8]. This benchmark problem ignores the sink term.

Soil-specific Parameters	Values	Units
Saturated hydraulic conductivity, K_s	0.00944	cm/s
Saturated soil moisture content, θ_s	0.287	–
Residual soil moisture content, θ_r	0.075	–
α in Haverkamp's model	1.611×10^6	cm
A in Haverkamp's model	1.175×10^6	cm
β in Haverkamp's model	3.96	–
γ in Haverkamp's model	4.74	–
Total time, T	360	s

Table 2: soil-specific parameters and their values used in the 1-D case study of Celia et al. [19] based on the empirical model developed by Haverkamp et al. [8].

Through this 1-D illustrative example, we will highlight the benefits of (a) adopting an adaptive L-scheme as opposed to static L-scheme, (b) implementing the DRW algorithm as opposed to the GRW algorithm, and (c) integrating the adaptive L-scheme with DRW in a holistic numerical framework.

6.1.1 The Need for Adaptive L-Scheme

To illustrate how adaptive L-scheme improves convergence and accuracy of conventional linearization schemes, we compare the pressure head solution profiles at $t = T = 360$ seconds obtained by different linearization parameters after (a) $S = 500$ iterations and (b) $S = 10,000$ iterations. We adopt a spatial grid containing 101 mesh points ($\Delta z = 0.4$ cm) and a temporal grid of $\Delta t = 1$ second. As shown in Figure 1, when using static L-scheme, the choice of linearization parameter and the total number of iterations can impact the solution accuracy and algorithm stability significantly. For example, when the linearization parameter is too small (e.g., $L = 0.5$ for this problem), the stability of the linearization scheme can be severely affected (as illustrated by the zigzag pressure head profile towards $z = 40$ cm). Another key observation is that, increasing the total number of iterations sometimes deteriorates solution accuracy of static L-scheme. These observations pose a practical challenge to use static L-scheme for solving the Richards equation, especially when the ground truth solutions are absent, as identifying the optimal linearization parameter and total number of iterations that would yield accurate solutions will not be possible without referring to ground truth solutions.

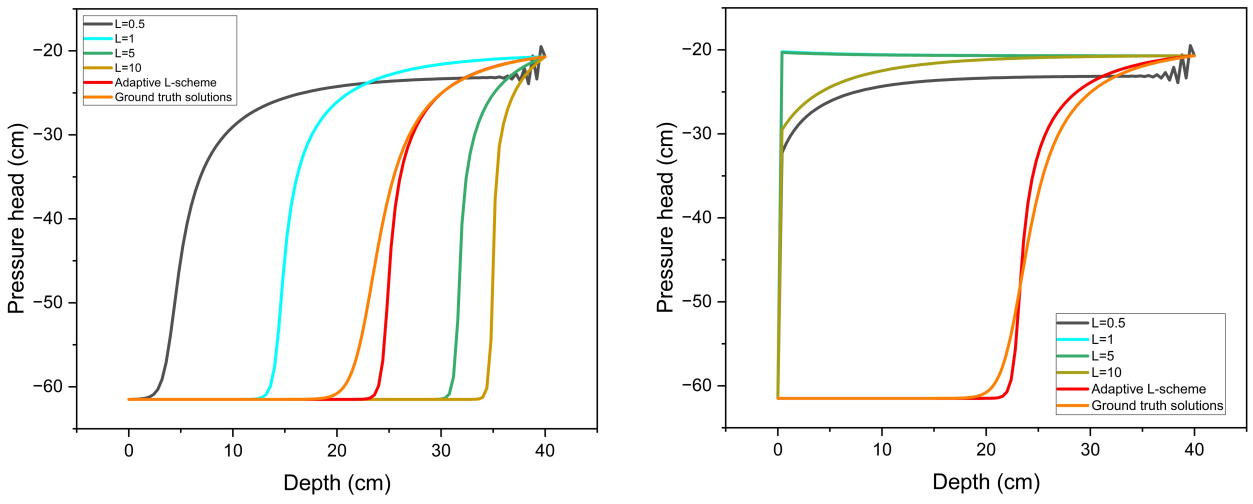

Figure 1: Comparison of pressure head solution profiles after (a) $S = 500$ and (b) $S = 10000$ iterations for the 1-D benchmark problem [19] using static and our adaptive L-schemes (Equation (8)). The solutions obtained from Celia et al. [19] based on very fine space and time steps are marked as the "ground truth solutions".

Unlike static L-scheme, by implementing the adaptive L-scheme of Equation (8) (note that DRW algorithm is not yet introduced), we observe that the pressure head solutions are close to the ground truth solutions even when only a limited number of iterations ($S = 500$) is used. Furthermore, as the number of iterations increases, the solution accuracy actually improves. This makes adaptive L-scheme a robust and reliable numerical scheme that produce solutions that are close to true solutions in one shot. Also, it is worth noting that our adaptive L-scheme successfully bypasses the singularity issue as $L_i^{m+1,s}$ approaches to 0 and correctly calculates the pressure head solutions for $z \in [0, 20 \text{ cm}]$ where $\theta(\psi)$ becomes small.

6.1.2 The Need for Data-driven Global Random Walk

To generate the reference solutions, we consider a coarse spatial discretization containing 40 cells (i.e., grid size $\Delta z = 1$ cm) and solve for $T = 40$ seconds. The time step size Δt is determined using the heuristic formula in [42]. The pressure head solutions are obtained using the finite difference method developed by Celia et al. [19]. On the other hand, the number of particles solutions are separately obtained from our in-house GRW solver derived from coupling the original GRW framework [42, 50] with the static L-scheme of FVM-discretized Richards equation under five different static linearization parameter values $L = 0.5, 1, 5, 10$ and 15 and 328 different total iteration counts $S = 60$, 120, ..., up to $19,680$. To initialize the GRW solver by expressing the initial and boundary conditions in terms of the number of particles, we multiply these conditions, originally represented in terms of pressure head, by a factor of 10^{10} particles per cm of pressure head.

As discussed earlier, the original GRW scheme [42, 50] was derived from the finite difference scheme using the relation that $\psi_i^{m,s}$ is proportional to $n_i^{m,s}$. To validate this relation under realistic computational settings, in Figure 2, we plot the reference solution subset containing $1,640$ randomly selected number of particles reference solutions at the last time step with different cells, total iteration counts, and linearization parameters against their corresponding pressure head solutions prior to performing data augmentation. It is clear from Figure 2 that, while the magnitude of pressure head shows roughly an increasing trend with respect to the number of particles present in

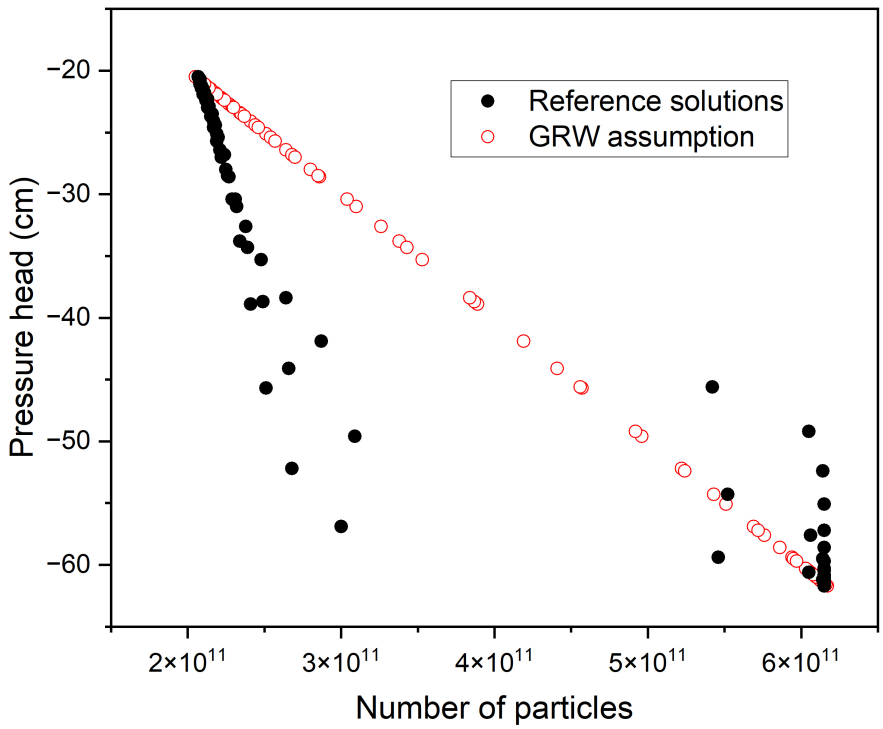

Figure 2: The relationships between 1,640 randomly selected number of particles reference solutions and their corresponding pressure head solutions, which are obtained by two distinct approaches. The resulting nonlinearity present in the randomly selected subset of reference solutions highlights the influence of numerical errors and realistic simulation settings on the actual implementation of the GRW scheme.

a cell, the relationship observed between the number of particles and pressure head deviates from the linear relationship embedded in the original GRW framework. Interestingly, we do observe close-to-linear trends in specific regions of pressure head values (i.e., between -20 and -30 cm, and between -50 and -60 cm). By examining Figure 1, one can see that these two regions correspond to the two ends of the spatial domain. Indeed, towards both ends, $\frac{\partial \psi}{\partial z}$ is close to 0, indicating that the advection term in Equation (1), $\nabla \cdot (K \nabla z) = \frac{\partial K}{\partial z} = \frac{\partial K}{\partial \psi} \frac{\partial \psi}{\partial z}$, vanishes. Thus, the Richards equation essentially becomes the diffusion equation, in which the proportionality assumption between pressure head and the number of particles is shown to be valid [44]. However, for the rest of the region where pressure head changes rapidly with respect to depth, this linear relation can be relaxed by data-driven methods to effectively learn the actual relationship observed between pressure head and the number of particles.

6.1.3 Improving DRW Algorithm Performance via Data Augmentation

To implement our DRW algorithm, we use two fully-connected neural networks, each containing 3 hidden layers and 256 neurons in each layer. We use the leaky ReLU activation function [57] and stochastic gradient descent (SGD) optimizer with a learning rate of 0.001. We perform data augmentation on the 1,640 randomly selected reference solutions to increase dataset size. We use our in-house GRW solver, which is built upon the original GRW framework [42, 50] and coupled with static L-scheme of the FVM-discretized Richards equation, to obtain the number of particles

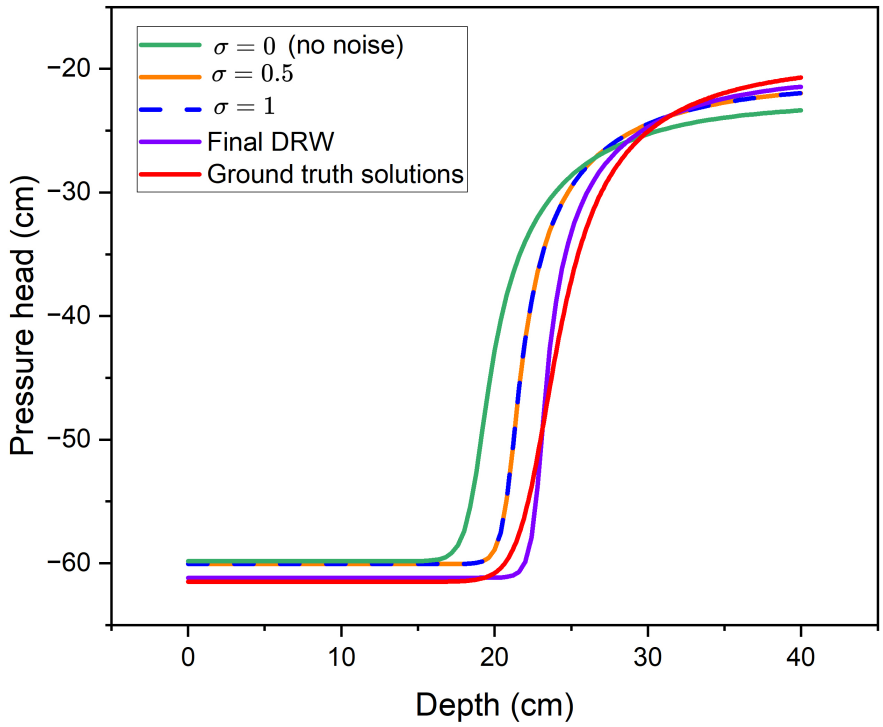

Figure 3: Pressure head profiles synthesized using different numerical methods and different training data. Specifically, curves named $\sigma = 0, 0.5, 1$ are calculated from our DRW algorithm, in which only 1,640 reference solutions (with Gaussian noise added for $\sigma = 0, 0.5, 1$) are used for neural network training. The purple curve is the final DRW pressure head profile obtained by considering the full 17,104 data points for neural network training through data augmentation.

solutions. We then make multiple copies of the pressure head solutions and append each copy to the number of particles solutions obtained from our GRW solver. Finally, we add zero-mean Gaussian noises with standard deviation varying from 0.1 to 0.5 to these augmented reference solutions. Overall, this leads to a total of 17,104 reference solutions for neural network training and validation. Note that, as previously discussed, the original and augmented reference solutions are generated using a coarse grid ($\Delta z = 1$ cm). Thus, they can be obtained relatively efficiently. On the other hand, in the solution step, we will use a more refined grid containing 101 mesh points ($\Delta z = 0.4$ cm).

To understand the impact of data augmentation to the solution quality of our DRW algorithm, we examine several cases as illustrated in Figure 3. First, we observe that, compared to directly using the original reference solutions for neural network training, simply introducing Gaussian noise to the reference solutions can significantly improve the solution accuracy of our DRW algorithm. Note that these augmented reference solutions come from from a small set of original reference solutions generated from a coarse grid. This "coarse-to-fine" approach can therefore enhance the solution accuracy of our DRW algorithm without requiring a large amount of high-accuracy, fine-mesh training data. Furthermore, when augmented reference solutions are used for training, only 100 additional epochs are needed to retrain the neural networks that are already trained using the original reference solutions . Second, we notice that that there is almost no difference in final pressure head solution profile when Gaussian noises of different magnitudes are directly added to the original reference solutions without augmenting them together. Third, increasing the size of training data (from 1,640 to 17,104) via data augmentation of original reference solutions is an

effective way to improve the solution accuracy of our DRW algorithm, as the pressure head profile matches very well with the ground truth solution.

6.1.4 Incorporating Adaptive L-Scheme in GRW Framework

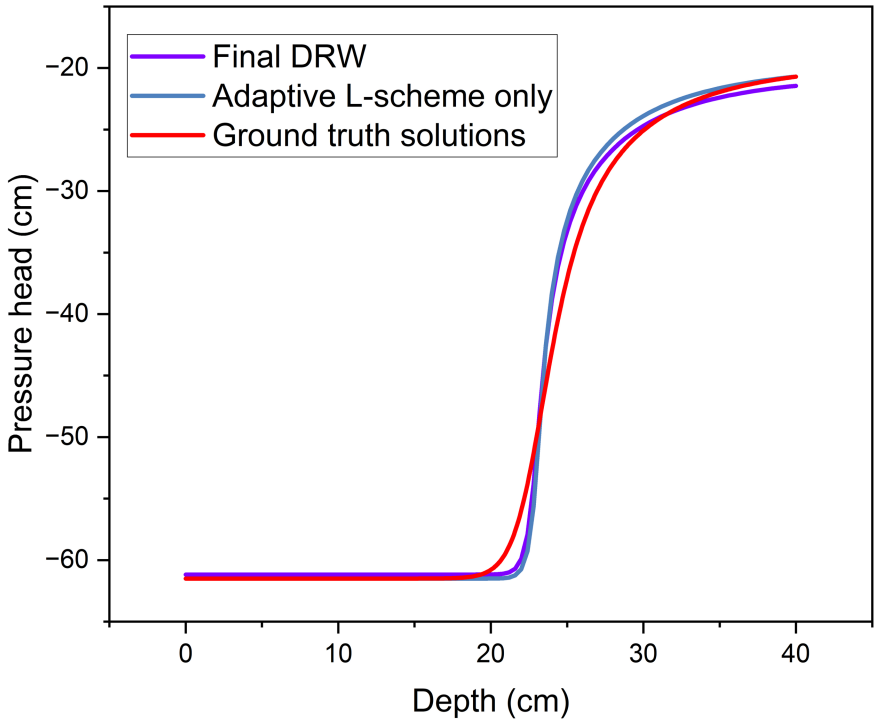

Figure 4: Comparison of pressure head solution profiles produced from adaptive L-scheme itself (Equation (8)) and from the integrated DRW framework coupled with adaptive L-scheme (Equation (21)).

From Figure 4, it is clear that incorporating adaptive L-scheme in the DRW framework synergistically improves the overall solution accuracy of the Richards equation, especially in the region where pressure head changes rapidly with respect to depth (i.e., between $z = 20$ to 30 cm). On the other hand, we observe slight discrepancy in pressure head solution close to $z = 40$ cm when comparing our DRW algorithm with ground-truth solutions, whereas the solution produced by adaptive L-scheme alone matches perfectly with ground-truth solution at $z = 40$ cm, which corresponds to one of the boundary conditions. Similar discrepancies have been observed in Figure 3 as well. We believe that this is due to the fact that $\hat{f}_{\text{NN}}$ and $\hat{f}_{\text{NN}}^{-1}$ only approximate f and f^{-1}, respectively, and the resulting induced error causes discrepancies in pressure head solutions even at the boundaries. To overcome this limitation, one way is to increase the size of the augmented reference solutions for neural network training. This is supported by the observations in Figure 3, in which the pressure head solutions get closer to the ground truth solutions at $z = 40$ cm as data augmentation takes place. Another approach is to switch from DRW (Equation (21)) to adaptive L-scheme only (Equation (8)) when solving for the boundary conditions. We leave this research for future work.

6.1.5 Comparison with Benchmark Solvers

We compare our DRW solver with benchmark algorithms based on computational performance and solution accuracy under two scenarios. In Scenario 1, we set the error tolerance tol to be 3.2×10^{-5},

whereas in Scenario 2, we set the total number of iterations $S = 500$. In terms of computational efficiency, we use two metrics. The first metric is the relative error RE defined in Equation (22), which measures the speed of convergence. The second metric is the condition number of matrix $\mathbf{A}$ defined in Equation (11), which measures the sensitivity of linearization scheme subject to small perturbations. These results are summarized in Tables 3 through 6.

Algorithm	Average number of iterations needed (Average final RE reached)	
	Static L-scheme	Adaptive L-scheme
GRW	736 (3.1999×10^{-5})	1 (9.2873×10^{-6})
DRW	2 (3.1975×10^{-5})	1 (8.5403×10^{-6})

Table 3: Comparison of average number of iterations across all discretized cells and time steps needed to reach the specified tol (Scenario 1) for GRW and DRW algorithms that implement static or adaptive L-scheme. In static L-scheme, we use the optimal linearization parameter of 3.5 identified by trial-and-error process.

Algorithm	Average final RE reached	
	Static L-scheme	Adaptive L-scheme
GRW	4.1130×10^{-5}	4.8824×10^{-7}
DRW	3.9287×10^{-5}	5.2006×10^{-7}

Table 4: Comparison of average RE after 500 iterations (Scenario 2) across all discretized cells and time steps for GRW and DRW algorithms that implement static or adaptive L-scheme. We also the optimal linearization parameter of 3.5 when implementing the static L-scheme.

Algorithm	Average condition number of $\mathbf{A}$ obtained from [46] (Scenario 1)	
	Static L-scheme	Adaptive L-scheme
GRW	1.7666	1.0064
DRW	1.7472	1.0074

Table 5: Comparison of average condition number under Scenario 1 across all time steps (as Equation (11) already considers all discretized cells) for GRW and DRW algorithms that implement static or adaptive L-scheme.

Algorithm	Average condition number of $\mathbf{A}$ obtained from [46] (Scenario 2)	
	Static L-scheme	Adaptive L-scheme
GRW	1.7206	1.0064
DRW	1.7137	1.0074

Table 6: Comparison of average condition number under Scenario 2 across all time steps for GRW and DRW algorithms that implement static or adaptive L-scheme.

From Tables 3 and 4, it is clear that, compared to using static L-scheme, implementing adaptive L-scheme can greatly accelerate convergence for both in GRW and DRW solvers. From Tables 5

and Table 6, we see that implementing adaptive L-scheme also significantly improves the stability of GRW and DRW based solvers, as matrix $\mathbf{A}$ is well-conditioned. Both observations suggest that adaptive L-scheme outperforms static L-scheme in enhancing convergence behavior of both GRW and DRW algorithms and is a powerful iterative procedure to solve the Richards equation.

In terms of accuracy, we also consider two metrics. The first metric is the discrepancy against ground truth solutions of Celia et al. [19]. The comparison results are illustrated in Figure 5. And the second metric is the solver's performance in preserving the mass (moisture) balance, which is quantified by the mass balance measure MB defined in [19]:

$$\text{MB} = \frac{\text{total additional mass in the domain}}{\text{total water flux into the domain}}. \tag{23}$$

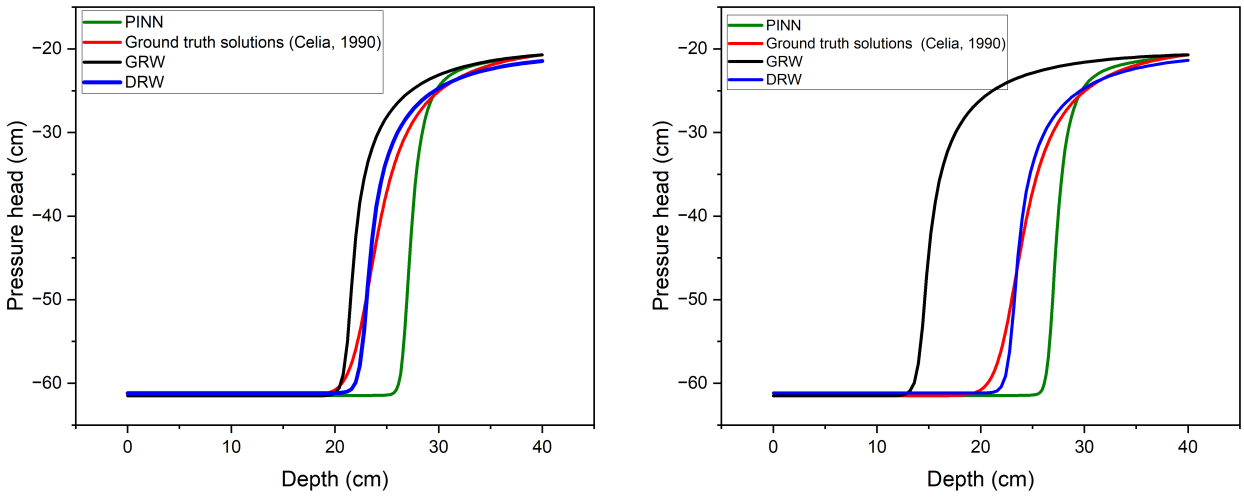

Figure 5: Pressure head profiles at $t = T = 360$ sec obtained by different algorithms under (left) Scenario 1, and (right) Scenario 2. Our DRW algorithm incorporates adaptive L-scheme, while our GRW algorithm adopts the static L-scheme of the FVM-discretized Richards equation ($L = 3$ for Scenario 1 and $L = 1$ for Scenario 2, identified through trial-and-error experimentation). Note that physics-informed neural network (PINN) method is not an iterative method, thus the solution profile is the same under both scenarios.

In Figure 5, we compare the pressure head profiles obtained from our DRW algorithm (which implements adaptive L-scheme), the GRW algorithm, and a state-of-the-art physics-informed neural network (PINN) method based on the work of Bandai and Ghezzehei [58], against the ground truth solution [19]. Clearly, as more iterations are used, both GRW and DRW solutions move closer to ground truth solutions. However, in both scenarios, compared with the DRW solutions, PINN and GRW solutions are further apart from ground truth solutions.

In our earlier work [28], we illustrated the DRW algorithm's ability to accurately capture underlying physics of water flow dynamics in soil, including the relationships among pressure head, soil moisture, and water flux. Extending these observations, we compare the mass conservation performance among various algorithms.

As summarized in Table 7, among all numerical methods studied, in both Scenarios 1 and 2, our DRW algorithm achieves the highest MB values while using the coarsest time step, which is more computationally efficient than using finer time steps. Similarly, Table 8 shows that, compared to other numerical methods, our DRW algorithm achieves the highest MB values when fixing Δt to be

Method used	Scenario	Δt (sec)	MB
GRW algorithm	1	16.79	96.46%
DRW algorithm	1	18.68	**98.92%**
GRW algorithm	2	16.87	62.12%
DRW algorithm	2	18.20	**99.28%**
Celia et al. [19]	N/A	10	95.00%

Table 7: MB results of different numerical methods. Note that here, Δt is determined for each method by the heuristic formula of Equation 5 in [42].

Method used	Scenario	MB ($\Delta t = 10$ sec)
GRW algorithm	1	98.00%
DRW algorithm	1	**99.47%**
GRW algorithm	2	80.37%
DRW algorithm	2	**99.85%**
Celia et al. [19]	N/A	95.00%

Table 8: MB results of different numerical methods, in which a common $\Delta t = 10$ sec is used for all numerical methods.

the same. In addition, given more iterations, the MB value in all iterative methods will improve. This is also consistent with how the accuracy of the pressure head profiles improves with increasing iterations when comparing Figure 5a with 5b.

6.1.6 Remark on Computational Efficiency

Although our DRW framework does involve neural network training which will take some additional time, there are several well-established strategies widely used in the machine/deep learning community to reduce the overall computational time and costs. For example, as previously discussed, one can leverage the previously trained neural network from a different problem setting as a good starting point to train with new dataset for the new problem setting in just a small number of epochs. To see this, we run the 1-D benchmark problem of [19] in a Dell Precision 7920 Tower equipped with Intel Xeon Gold 6246R CPU and NVIDIA Quadro RTX 6000 GPU (with 24GB GGDR6 memory). The DRW algorithm is implemented in Python 3.10.5. The total computational time for solving the Celia problem from scratch with $S = 500$ is 110.89 sec, in which the neural network training step costs 64.96 sec. On the other hand, when using a pretrained model, the total computational time is reduced by 53% down to 51.90 sec. Meanwhile, the computational time for a direct solver is 18.36 sec. While our DRW algorithm still takes more time than the direct solver, it is still an attractive numerical framework as: 1) it gives more accurate solutions; 2) its data-driven nature makes it suitable for seamless integration between physics-based modeling and in situ soil sensing technologies; 3) for large-scale and/or more complex problem settings, the neural network training time will become less significant compared to the actual solution time; and 4) our DRW algorithm consumes less computational time compared to many neural PDE solvers [53, 52].

6.2 A 1-D Layered Soil Benchmark Problem

To investigate how robust our DRW algorithm is in handling realistic problems, we study the classic Hills' problem [59] that involves the 1-D water infiltration into two layers of very dry soil, each having a depth of 30 cm. The top layer (layer 1) corresponds to Berino loamy fine sand and the bottom layer (layer 2) corresponds to Gledale clay loam. The WRC and HCF follow the Mualem-van Genutchen model. The soil-specific parameters are extracted from [59] and are listed in Table 9. This benchmark problem also ignores the sink term.

As pointed out by Berardi et al. [56], the dry condition is the most challenging physical case to model from a numerical point of view. The presence of discontinuous interface across the two soil layers presents yet another complication to this problem. And we simulate the problem up to a total time of $T = 7.5$ minutes. For neural network training, we generate a total of 30,500 reference solutions by our in-house GRW solver using the optimal static linearization parameter of 5.

Soil	θ_r	θ_s	α	n	K_s
Berino loamy fine sand	0.029	0.366	0.028	2.239	541.0
Gledale clay loam	0.106	0.469	0.010	1.395	13.10

Table 9: Soil-specific parameters and constants used in the layered soil problem of Hills et al. [59].

Figure 6 illustrates the soil moisture profile at three different times obtained using our DRW algorithm, the GRW solver, and the Transversal Method of Lines (TMOL) [56] which is considered the current state-of-the-art algorithm for this problem. All three approaches adopt the same discretized temporal ($\Delta t = 1$ second) and spatial steps ($\Delta z = 1$ cm). We set the $\text{RE}_s = 10^{-5}$ as the common stopping criteria for every cell and time step. From Figure 6, we observe that our DRW algorithm is capable of successfully simulating this challenging problem with discontinuities in soil properties at the interface. The soil moisture solutions obtained by our DRW algorithm are also consistent with existing solvers. In fact, compared to the GRW solver, the solutions produced by our DRW algorithm are closer to the state-of-the-art TMOL solutions.

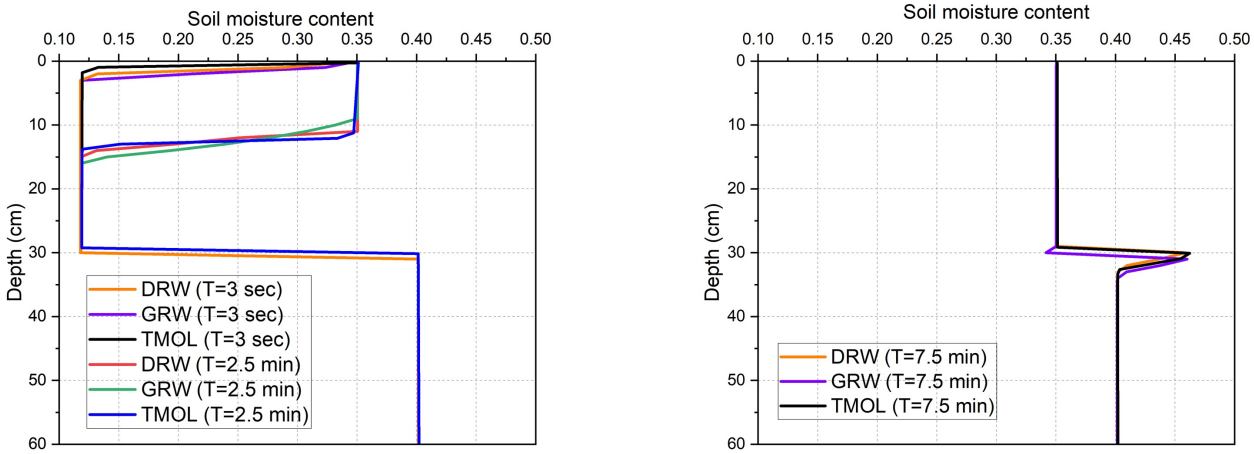

Figure 6: Comparison of soil moisture content profile obtained different methods with $\Delta z = 1$ cm under (left) DRW, GRW and TMOL at $t = T = 3$ sec and $t = T = 2.5$ min and (right) DRW, GRW and TMOL at $t = T = 7.5$ min. Note that TMOL by Berardi et al. [56] is not an iterative method. GRW and DRW are implemented for 500 iterations at every time step.

6.3 A 2-D Benchmark Problem

In the second example, we study the 2-D Richards equation for an infiltration process in a $1\text{m} \times 1\text{m}$ loam soil field [54]. The spatial steps in both horizontal (Δx) and vertical (Δz) directions are set to be 0.02 m. And the time step used for this comparison study is $\Delta t = 10$ seconds. The Mualem-van Genuchten model (see Table 1) was used in this case study. And the soil-specific parameters, given by Carsel and Parrish [60], are listed in Table 10. This problem also ignores the sink term.

Property	Symbol	Value	Units
Saturated hydraulic conductivity	K_s	2.89×10^{-6}	m/s
Saturated water content	θ_s	0.43	–
Residual water content	θ_r	0.078	–
van Genuchten Constant	α	3.6	m^{-1}
van Genuchten Constant	n	1.56	–
Total time	T	1.26×10^4	s

Table 10: Soil-specific parameters and constants used in 2-D case study.

The initial and boundary conditions of this case study are given by:

$$\text{Initial condition: } \psi(x, z, t = 0 \text{ s}) = \begin{cases} 0\text{m}, & x \in [0.46, 0.54]\text{m},\ z = 0\text{m}, \\ -10\text{m}, & \text{otherwise.} \end{cases}$$

Boundary condition: $\psi(x \in [0.46, 0.54]\text{m}, z = 0, t) = 0\text{m}$, no slip conditions for other boundaries.

Note that the initial and boundary conditions are symmetric along $x = 0.5$m. We first obtain 1,734 original reference solutions for neural network training from our in-house developed 2-D GRW solver derived from incorporating the original GRW framework [42, 50] in static L-scheme of the FVM-discretized Richards equation and uses a spatial step of 0.05 m under two different static linearization parameters $L = 0.5$ and 1 and two total iteration counts $S = 250$ and 500, while excluding any NaN values. Then, we apply data augmentation by adding Gaussian noises with σ^2 values ranging from 0.1 to 0.5 to generate a total of 26,010 reference solutions (which also contain the original reference solutions). These reference solutions are used to train the two neural networks for our DRW algorithm. Each neural network contains 2 hidden layers and 25 neurons in each layer. ReLU activation function is adopted in each layer, and each neural network is trained by Levenberg-Marquardt optimization for 1,000 epochs. We set the total iteration number to be $S = 500$, at which the relative error calculated using Equation (22) for our DRW algorithm and the GRW solver are given by 3.675×10^{-6} and 1.094×10^{-5}, respectively. This indicates that our DRW algorithm achieves faster convergence per iteration than the GRW solvers. The total computational time for our DRW algorithm to run from scratch with $S = 500$ is 570.54 sec, whereas our GRW solver takes 688.14 sec under the same S. When using a pretrained model, the total computational time for our DRW algorithm is reduced to 198.19 sec.

Meanwhile, we also simulate this 2-D problem using HYDRUS software [61] and compare the pressure head results at $t = T = 1.26 \times 10^4$ sec with our DRW algorithm and the GRW solver (optimal linearization parameter identified to be 0.5 by trial-and-error). From Figure 7, we can draw two observations. First, the pressure head solution profiles for both GRW and DRW algorithms appear to be symmetric along $x = 0.5$ m, whereas HYDRUS 2D shows a clear asymmetric profile. As pointed out earlier, since the initial and boundary conditions are symmetric along $x = 0.5$ m,

symmetry in the pressure head solutions is expected. This suggests that both GRW and DRW based solvers can capture some degree of underlying physics of the original problem. Second, despite the assymetric behavior in pressure head profile, the size of isolines for the HYDRUS 2D simulation result is more similar to our DRW solution than to the GRW solver solution. This observation is also consistent with the information presented in Figure 9a. In fact, both observations can be carried over to the soil moisture profile as well, as shown in Figures 8 and 9b.

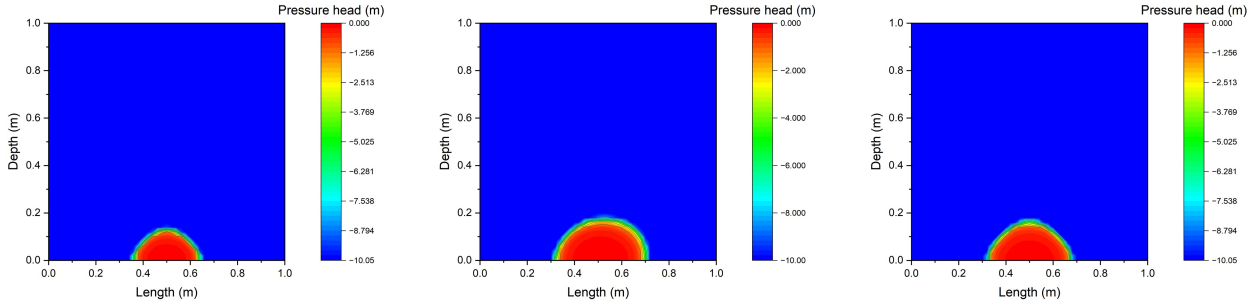

Figure 7: Pressure head solution profile obtained from three numerical methods: (left) GRW solver (linearization parameter = 0.5); (middle) HYDRUS 2D software; (right) our DRW algorithm.

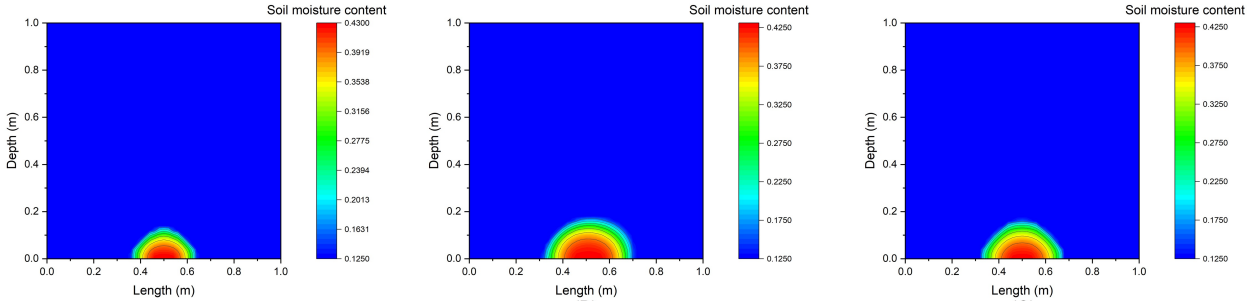

Figure 8: Soil moisture solution profile obtained from three numerical methods: (left) GRW solver (linearization parameter = 0.5); (middle) HYDRUS 2D software; (right) our DRW algorithm.

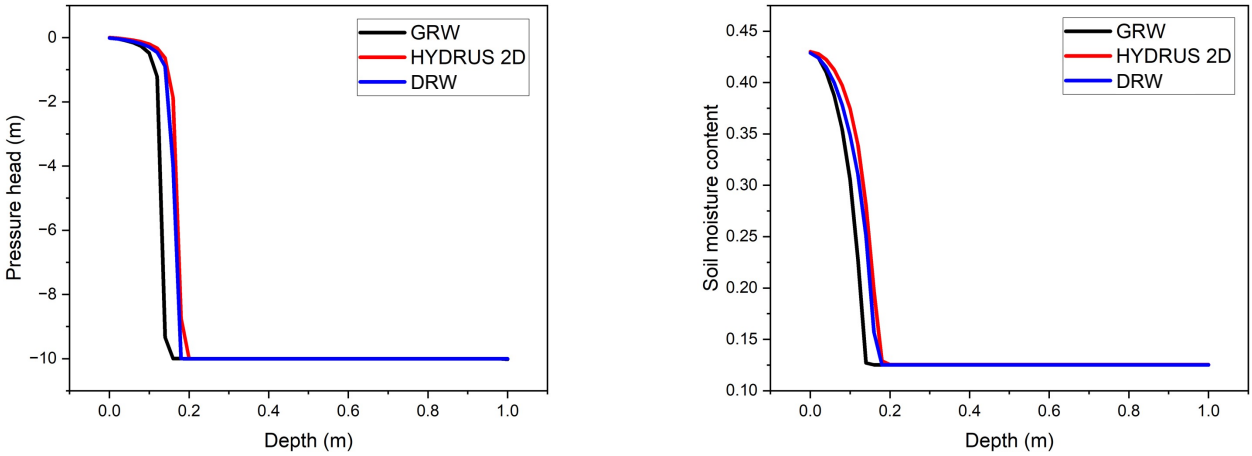

Figure 9: Cross-sectional view ($x = 0.5$m) of: (left) the pressure head profile; (right) soil moisture profile.

On the other hand, when comparing the water flux results, we see from Figure 10 that the GRW

solution no longer preserves the symmetry of water flux profile along the horizontal direction. In other words, among the three numerical methods considered in this case study, our DRW algorithm achieves the best performance in terms of preserving the symmetry implied by the problem. This result is also consistent with the mass conservation calculations using Equation (23), as our DRW algorithm achieves significantly higher MB value compared to other benchmark solver (see Table 11).

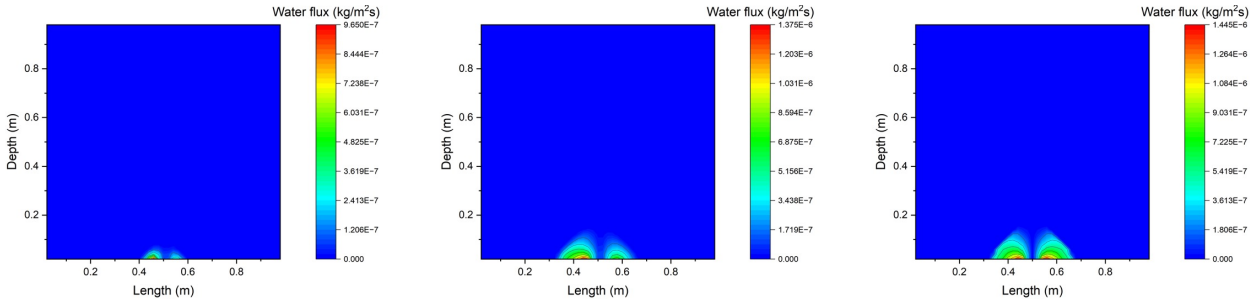

Figure 10: Magnitude of water flux along the horizontal (x-axis) direction for three numerical solvers: (left) GRW solver (linearization parameter = 0.5); (middle) HYDRUS 2D software; (right) our DRW algorithm. Note that, along the horizontal direction, the water flux is negative in $[0, 0.5]$ m and positive in $(0.5, 1]$ m.

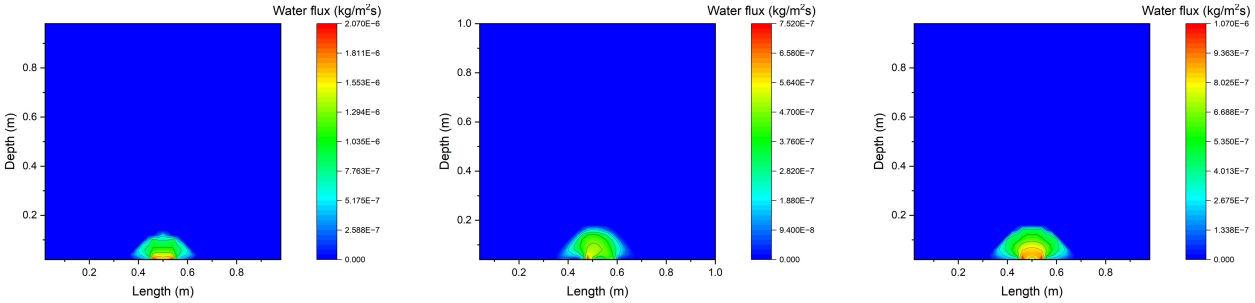

Figure 11: Magnitude of water flux along the vertical (z-axis) direction for three numerical solvers: (left) GRW solver (linearization parameter = 0.5); (middle) HYDRUS 2D software; (right) our DRW algorithm.

Method	MB ($\Delta t = 10$ sec)
GRW algorithm	63.93%
HYDRUS 2D simulation	62.45%
DRW algorithm	**73.07%**

Table 11: MB results of three methods at $x = 0.5$ m.

6.4 A 3-D Benchmark Problem with Analytical Solutions

Lastly, we consider a 3-D water infiltration example where the analytical solution exists [55]. In this example, V is a 3-D cuboid $[0, a] \times [0, b] \times [0, c]$. The hydraulic conductivity function follows the Gardner's model [11] (see Table 1). The initial condition is given by:

$$\psi(x, y, z, t = 0) = h_r,$$

where h_r is a constant. And the boundary condition is given by:

$$\psi(x, y, z = c, t) = \frac{1}{\alpha} \ln \left[\exp(\alpha h_r) + \overline{h}_0 \sin \frac{\pi x}{a} \sin \frac{\pi y}{b} \right],$$

where $\overline{h}_0 = 1 - \exp(\alpha h_r)$. Ignoring the sink term, the pressure head solution for this problem was derived in [55] as:

$$\psi = \frac{1}{\alpha} \ln \left\{ \exp(\alpha h_r) + \overline{h}_0 \sin \frac{\pi x}{a} \sin \frac{\pi y}{b} \exp\left(\frac{\alpha(c-z)}{2} \right) \left[\frac{\sinh \beta z}{\sinh \beta c} + \frac{2}{zd} \sum_{k=1}^{\infty} (-1)^k \frac{\lambda_k}{\gamma} \sin(\lambda_k z) \exp(-rt) \right] \right\}, \tag{25}$$

where $d = \frac{\alpha(\theta_s - \theta_r)}{K_s}$, $\lambda_k = \frac{k\pi}{c}$, $\gamma = \frac{\lambda_k^2 + \beta^2}{c}$ and $\beta = \sqrt{\frac{\alpha^2}{4} + (\frac{\pi}{a})^2 + (\frac{\pi}{b})^2}$.

The infinite series in Equation (6.4) is convergent by alternating series test, and we consider only the first 1000 terms of this series. Note from Equation (6.4) that the analytical solution depends only on the saturated (θ_s) and residual soil moisture content (θ_s). And the Mualem-van Genuchten correlation [9, 10] tabulated in Table 1 was used for the water retention curve $\theta(\psi)$. The constants and parameters used in this case study are listed in Table 12.

Property	Symbol	Value	Units
Saturated hydraulic conductivity	K_s	1.1	m/s
Saturated soil moisture	θ_s	0.5	–
Residual soil moisture	θ_r	0	–
Parameter in Gardner's model	α	0.1	m^{-1}
Parameter in intial and boundary conditions	h_r	−15.24	m
Length of V	a	2	m
Width of V	b	2	m
Depth of V	c	2	m
Total time	T	86,400	sec

Table 12: Soil-specific parameters and constants used in the 3-D case study.

Our goal is to compare the accuracy of our DRW algorithm with GRW solvers using this analytical solution as the benchmark. We use our own in-house 3-D GRW solver [27], which is built upon the original GRW framework [42, 50] and implements the static L-scheme of the FVM-discretized 3-D Richards equation, to obtain 1,734 original reference solutions using a coarse grid of $\Delta x = \Delta y = \Delta z = 0.4$ m under two static linearization parameters $L = 0.5$ and 1 and five total iteration counts $S = 100$, 200, ..., 500, while excluding any NaN values. Then, data augmentation is applied by introducing Gaussian noise, resulting in a total of $8,820$ data points (which include the original reference solutions) for neural network training. For both GRW and DRW algorithms, we set the tolerance to be 10^{-9}, which can be achieved in less than 500 iterations for each time step. The total computational time for our DRW algorithm to run from scratch is 14.33 hours, whereas our GRW solver takes 12.78 hours.

We examine and compare the pressure head solutions at $z = 0.5$ and 1m, which are shown in Figure 12 and 13, respectively. We quantify the differences between the numerical solution and the analytical solution by $\frac{\psi_{\text{analytical}} - \psi_{\text{numerical}}}{\psi_{\text{analytical}}}$. From the relative difference heat map of Figure 12c,e and 13c,e, we observe that, first, the magnitude of relative error of our DRW based solver is significantly lower than that of GRW based solver. And second, the largest relative error of our DRW pressure

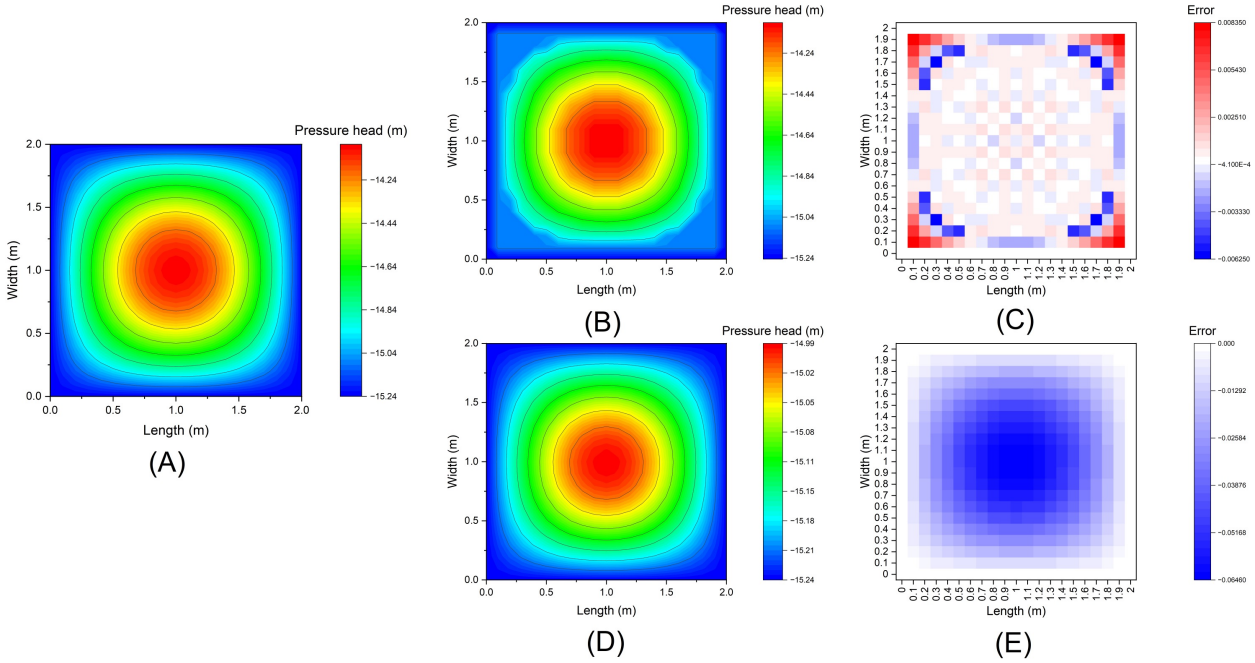

Figure 12: Pressure head solution at $z = 0.5$m of different methods: (A) analytical solution, (B) DRW algorithm, (C) the relative error between analytical and DRW based solutions, (D) GRW solver (optimal $L = 0.5$ identified by trial-and-error) and (E) the relative error between analytical solution and GRW based solutions.

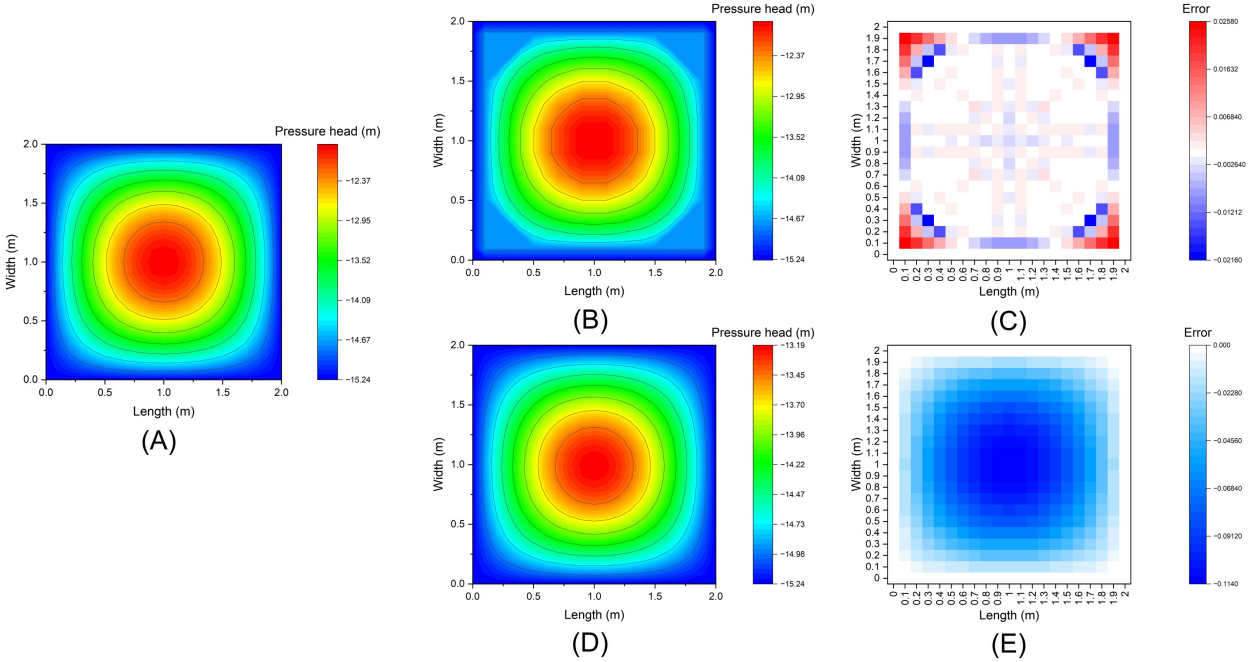

Figure 13: Pressure head solution at $z = 1.0$ m of different methods: (A) analytical solution, (B) DRW algorithm, (C) the relative error between analytical and DRW based solutions, (D) GRW solver ($L = 0.5$) and (E) the relative error between analytical solution and GRW based solutions.

head solution occurs around the four corners of the x–y domain, whereas the largest relative error of GRW solution occurs in the center of the x–y domain. Furthermore, in each cell, the relative error of GRW-based pressure head solution is always non-positive, whereas that of DRW based pressure head solution can be positive or negative.

Here, we provide some justifications to these observations. First, for conventional GRW solver that embeds the static L-scheme formulation, we observe from Equations (8) that:

$$\psi_{\text{analytical}} - \psi_{\text{numerical}} \propto \left\{ \sum_{j \in \mathcal{N}_i} \left[K(\psi) \nabla (\psi + z) \right]_{\omega_{i,j}}^{m+1,s} \cdot \mathbf{n}_{\omega_{i,j}} A_{\omega_{i,j}} - \partial_t \theta_i^{m+1} \text{vol}(V_i) \right\},$$

for any s, discretized cell V_i, and discretized time step m. Since the hydraulic conductivity function is positive and symmetric along $x = 1$ m and $y = 1$ m, and $\nabla \psi \big|_{\omega^+ := [0,1] \times [0,1] \times z} = -\nabla \psi \big|_{\omega^- := [1,2] \times [1,2] \times z}$, we have $\sum_{j \in \mathcal{N}_i} \left[K(\theta(\psi)) \nabla (\psi + z) \right]_{\omega_{i,j}}^{m+1,s} \cdot \mathbf{n}_{\omega_{i,j}} A_{\omega_{i,j}} > 0$. Meanwhile, $\partial_t \theta_i^{m+1}(\psi) \text{vol}(V_i)$ is typically small due to the slow dynamics of water infiltration in soil and the fact that $\text{vol}(V_i)$ is small. Thus, we have $\psi_{\text{analytical}} - \psi_{\text{numerical}} > 0$ for GRW solution, which explains the negativity of the relative error. On the other hand, for our DRW algorithm, the use of neural networks to approximate f and f^{-1} complicates the behavior (including the sign) of the relative error.

Regarding the distribution of the magnitude of relative error in the GRW solver, since hydraulic conductivity function is an increasing function of ψ, and ψ is at its maximum at the center of the x–y plain, it is expected that $\sum_{j \in \mathcal{N}_i} \left[K(\psi) \nabla (\psi + z) \right]_{\omega_{i,j}}^{m+1,s} \cdot \mathbf{n}_{\omega_{i,j}} A_{\omega_{i,j}}$, and hence the relative error, is maximized at and around the center of the x-y plane. However, for the DRW based pressure head solution, we suspect that the higher relative error at the four corners may be attributed to the slight decrease in accuracy of neural networks in approximating f and f^{-1} near the boundaries of domain.

Finally, we evaluate the MSE by summing the individual MSE values over all cells at $z = 0.5$ m and $z = 1$ m. For $z = 0.5$ m, MSE_{DRW} and MSE_{GRW} are calculated to be 8.044×10^{-4} and 0.1972, respectively. For $z = 1$ m, MSE_{DRW} and MSE_{GRW} are 6.736×10^{-3} and 0.5052, respectively. This indicates that the MSE of the GRW based pressure head solution is typically 1 to 2 orders of magnitude higher than our DRW based solution.

7 A Realistic Case Study

Finally, we consider a real-world case study adopted from Orouskhani et al. [45], where infiltration, irrigation, and root water extraction take place in circular agricultural field, equipped with a center-pivot irrigation system with a radius of 50 m, located at Lethbridge, Alberta. Soil moisture sensors are inserted at a depth of 25 cm across 20 different locations in this field to collect soil moisture data every 30 min from June 19 to August 13, 2019. To validate our DRW algorithm in solving 3-D real-world applications, we select one of the 20 locations where the Mualem-van Genuchten WRC and HCF model parameters are identified and given in [45]. We consider a cylindrical control volume V with a radius of 0.1 m and depth of 25 cm. We discretize V into 6, 40 and 22 nodes in the radial, azimuthal and axial directions, respectively. The time step size Δt is determined using the heuristic formula in [42]. Thus, we reformulate Equation (21) in cylindrical coordinate system as:

$$n_i^{m+1,s+1} = n_i^{m+1,s} + \frac{1}{L_i^{m+1,s}} \sum_{j \in \mathcal{N}_i} K_{\omega_{i,j}}^{m+1,s} \hat{\mathbf{e}}_j \cdot \mathbf{n}_{\omega_{i,j}} \frac{n_j^{m+1,s} - n_i^{m+1,s}}{\text{dist}(V_j, V_i)} A_{\omega_{i,j}} + f^{-1}(J),$$

where $\hat{\mathbf{e}}_j = (1, \frac{1}{r_j^2}, 1)^T$ and

$$\begin{aligned} J =& \frac{1}{L_i^{m+1,s}} \sum_{j \in \mathcal{N}_i} K_{\omega_{i,j}}^{m+1,s} \hat{\mathbf{e}}_j \cdot \mathbf{n}_{\omega_{i,j}} \frac{z_j - z_i}{\text{dist}(V_j, V_i)} A_{\omega_{i,j}} - \frac{1}{L_i^{m+1,s}} \frac{\theta_i^{m+1,s} - \theta_i^m}{\Delta t} \text{vol}(V_i) \\ & - \frac{1}{L_i^{m+1,s}} S(\psi_i^{m+1,s}) \text{vol}(V_i). \end{aligned} \tag{26}$$

Here, the sink term in S follows the Feddes model [62]:

$$S = \sigma(\psi) S_{\max}, \tag{27}$$

where $S_{\max}$ is the maximum possible root extraction rate and σ denotes a dimensionless water stress reduction factor (see [63] for the detailed formulation).

The boundary conditions are given by:

$$\begin{aligned} &\frac{\partial \psi(r, \omega, z)}{\partial r} = 0 \quad \text{at } r = 0\text{m}, \\ &\frac{\partial \psi(r, \omega, z)}{\partial r} = 0 \quad \text{at } r = 0.1\text{m}, \\ &\frac{\partial \psi(r, \omega, z)}{\partial z} = 0 \quad \text{at } z = 0 \text{ cm}, \\ &\frac{\partial \psi(r, \omega, z)}{\partial z} = -1 - \frac{u_{\text{irr}}}{K(\psi)} \quad \text{at } z = 25 \text{ cm}, \\ &\psi(r, \omega = 0, z) = \psi(r, \omega = 2\pi, z), \end{aligned}$$

where u_{irr} is the irrigation rate (in m/s). And the initial condition is simply:

$$\psi(x, y, z, t = 0) = h_r,$$

where h_r is the starting pressure head recording.

Note that the boundary conditions are time-dependent due to u_{irr}. This poses a potential computational challenge as the neural networks typically need to be retrained whenever the initial or boundary conditions change [64, 65]. To overcome this practical challenge, we adopt a new approach of training the two neural networks with 3,000 epochs based on the boundary conditions for June 19, 2019 (no irrigation) when data collection began. Then, the trained weights within these two neural networks serve as the starting point for retraining when a new set of boundary conditions is adopted. This way, only 500 epochs are sufficient to retrain the neural networks. For each set of boundary conditions, we obtain the training set containing 84,480 reference solutions. In addition, the dataset provided by [45], after performing data augmentation by introducing Gaussian noises, is also included in our training dataset. Each neural netowrk, which has 5 hidden layers with 256 neurons in each layer, is trained using SGD optimizer with a learning rate of 0.001. We set the stopping criterion to be $\text{RE}_s = 10^{-9}$, which can be achieved well within 500 iterations.

For this problem, we simulate the pressure head from 1:00 am on June 19, 2019 to 5:00 pm on July 28, 2019. As mentioned in [45], there are two irrigation instances between this time frame, one is on July 4 (the 15th day, 1.81 mm) and the other is on July 18 (the 30th day, 1.58 mm). Figure 14 shows the pressure head solution profile obtained by our DRW algorithms compared to the experimental measurements provided by Orouskhani et al. [45] over the course of 35 days. We observe that, most of the time, the our DRW solutions match with the experimental measurements very well. The only major mismatches between experimental measurements and DRW solutions occur on the 15th

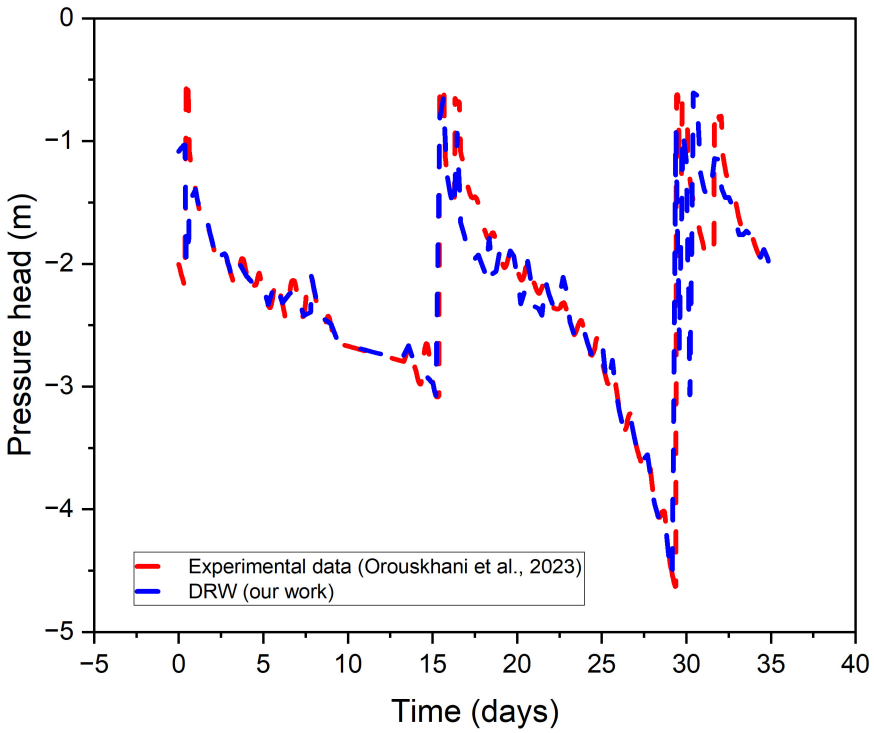

Figure 14: Comparison of pressure head profile at $z = 25$ cm in a selected 0.1-m radius region (averaged for all $6 \times 40 = 240$ cells at $z = 25$ cm) in the field.

day and the 30th day, which correspond to the time when the irrigation takes place. We believe both mismatches are a result of our simplifying assumption regarding the irrigation schedule. Due to the limited information we have about the exact irrigation schedule and intensity, we have to assume that both irrigation instances happened over the course of the entire day. Thus, we simply divide the irrigation amount by 86,400 seconds to obtain u_{irr}. However, in reality, the irrigation may finish within 24 hours, thus causing the mismatches. With more accurate u_{irr} model, our DRW algorithm is expected to produce highly accurate solutions that closely match with experimental measurements at all times. This makes our DRW algorithm an accurate and scalable numerical framework to solve Richards equation over a long period of time.

8 Conclusion

In this work, we present a novel data-driven global random walk (GRW) algorithm named DRW to accurately and efficiently solve d-dimensional Richards equation ($d = 1, 2, 3$). Our DRW algorithm, which is built upon the original GRW framework established by Suciu et al. [42, 50], adopts an adaptive L-scheme based on the FVM discretization of the Richards equation, which significantly improves convergence and stability of the solution process. To account for the numerical errors observed during actual implementation due to computational constraints and realistic simulation settings, we relax the linear relation between pressure head and number of particles solutions implied in the original GRW framework and introduce a data-driven approach to first learn the forward and inverse maps between the solutions using two neural networks, followed by integrating the trained neural networks with the numerical scheme to achieve synergistic improvement in solution accuracy. Furthermore, we also discuss effective ways, such as the "coarse-to-fine" approach, to perform data augmentation to facilitate neural nework training using just a small number of low-fidelity reference

solutions as training set. Overall, these innovative techniques work together seamlessly to improve the convergence and accuracy of DRW algorithm in solving the Richards equation. Indeed, via several 1-D through 3-D case studies that span across benchmark problems and real-world applications, we demonstrate that, compared to state-of-the-art numerical solvers, our DRW algorithm not only achieves significantly improved accuracy and convergence, but also better preserves the overall mass balance and conservation laws while being computationally efficient to implement. Moreover, the proposed data-driven numerical method is expected to be a generalizable computational framework for modeling a wide range of chemical engineering applications, including fractional diffusion [66], fluid transport in fibrous porous materials [67, 68, 69], liquid extraction from saturated granular materials [70], water and hydrocarbon flow in petroleum reservoirs [71], and so on.

9 Open Research

We use the open-source platform SimPEG [72] and the code at `https://simpegdocs.appspot.com/content/examples/20-published/plot_richards_celia1990.html` to implement the finite difference method by Celia et al. [19] for the 1-D case study. The ground truth solutions in Figures 1, 3, 4 and 5 are extracted from Celia et al. [19]. The GRW solutions used in the 1-D through 3-D case studies for neural network training and comparison purposes are obtained from our in-house GRW solvers, which are built upon the original GRW framework established by Suciu et al. [42, 50] and implement the static L-scheme of the FVM-discretized Richards equation. The MATLAB source code for the GRW solver developed by Suciu et al. [42] is available in the GitHub repository [50]. The Python codes for our 1-D through 3-D GRW and DRW solvers as well as the original and augmented reference solution datasets used in our benchmark case studies are available in GitHub repository [73]. We use HYDRUS 2D/3D Pro software (version 2.04.0580) to generate Figures 7b through 11b in the 2-D case study. For better illustration purposes, all figures are produced using OriginPro 2023b.

Acknowledgments

We acknowledge financial support from the startup fund of College of Engineering, Architecture, and Technology at Oklahoma State University.

A Formulations of $\delta n^{m+1,s}$ for Equation (20)

In Appendix A, we explicitly write down the expression of $\delta n^{m+1,s}$ for 1-D through 3-D cases.

The 1-D case:

For a discretized cell in 1-D, Equation (20) can be explicitly written as:

$$n_i^{m+1,s} = \delta n_{i+1,i}^{m+1,s} + \delta n_{i-1,i}^{m+1,s} + \delta n_{i,i}^{m+1,s}.$$

The $\delta n^{m+1,s}$ expression for the 1-D case has been explicitly derived in [41, 42] as:

$$\overline{\delta n_{i,i}^{m+1,s}} = \left[1 - \left(r_{i-\frac{1}{2}}^{m+1,s} + r_{i+\frac{1}{2}}^{m+1,s}\right)\right] \overline{n_i^{m+1,s}},$$
$$\overline{\delta n_{i\pm\frac{1}{2},i}^{m+1,s}} = r_{i\pm\frac{1}{2}}^{m+1,s} \overline{n_i^{m+1,s}},$$

where $r_{i\pm\frac{1}{2}}^{m+1,s} = K\left(\theta(\psi_{i\pm\frac{1}{2}}^{m+1,s})\right)\frac{\Delta t}{L_{i\pm\frac{1}{2}}^{m+1,s}\Delta z^2}$. In actual implementation, we use the unaveraged relation thereafter.

The 2-D case:

A discretized cell in 2-D is represented using the notation (i,j). With this, Equation (20) can be explicitly written as:

$$n_{i,j}^{m+1,s} = \delta n_{(i+1,j),(i,j)}^{m+1,s} + \delta n_{(i-1,j),(i,j)}^{m+1,s} + \delta n_{(i,j+1),(i,j)}^{m+1,s} + \delta n_{(i,j-1),(i,j)}^{m+1,s} + \delta n_{(i,j),(i,j)}^{m+1,s}.$$

The $\delta n^{m+1,s}$ expression for the 2-D case has been explicitly derived in [42] as:

$$\begin{aligned}
\overline{\delta n_{(i,j),(i,j)}^{m+1,s}} &= \left[1 - \left(r_{i-\frac{1}{2},j}^{m+1,s} + r_{i+\frac{1}{2},j}^{m+1,s} + r_{i,j-\frac{1}{2}}^{m+1,s} + r_{i,j+\frac{1}{2}}^{m+1,s}\right)\right]\overline{n_{i,j}^{m+1,s}},\\
\overline{\delta n_{(i\pm\frac{1}{2},j),(i,j)}^{m+1,s}} &= r_{i\pm\frac{1}{2},j}^{m+1,s}\overline{n_{i,j}^{m+1,s}},\\
\overline{\delta n_{(i,j\pm\frac{1}{2}),(i,j)}^{m+1,s}} &= r_{i,j\pm\frac{1}{2}}^{m+1,s}\overline{n_{i,j}^{m+1,s}},
\end{aligned}$$

where:

$$\begin{aligned}
r_{i\pm\frac{1}{2},j}^{m+1,s} &= \frac{K\left(\theta\left(\psi_{i\pm\frac{1}{2},j}^{m+1,s}\right)\right)\Delta t}{(\Delta x)^2 L_{i\pm\frac{1}{2},j}^{m+1,s}},\\
r_{i,j\pm\frac{1}{2}}^{m+1,s} &= \frac{K\left(\theta\left(\psi_{i,j\pm\frac{1}{2}}^{m+1,s}\right)\right)\Delta t}{(\Delta z)^2 L_{i,j\pm\frac{1}{2}}^{m+1,s}}.
\end{aligned}$$

The 3-D case:

A discretized cell in 3-D is represented using the notation (i,j,k). With this, Equation (20) can be explicitly written as:

$$\begin{aligned}
n_{i,j,k}^{m+1,s} =& \delta n_{(i+1,j,k),(i,j,k)}^{m+1,s} + \delta n_{(i-1,j,k),(i,j,k)}^{m+1,s} + \delta n_{(i,j+1,k),(i,j,k)}^{m+1,s}\\
&+ \delta n_{(i,j-1,k),(i,j,k)}^{m+1,s} + \delta n_{(i,j,k+1),(i,j,k)}^{m+1,s} + \delta n_{(i,j,k-1),(i,j,k)}^{m+1,s} + \delta n_{(i,j,k),(i,j,k)}^{m+1,s}.
\end{aligned}$$

$\delta n^{m+1,s}$ must satisfy the mean condition:

$$\begin{aligned}
\overline{\delta n_{(i,j,k),(i,j,k)}^{m+1,s}} &= \left[1 - \left(r_{i+\frac{1}{2},j,k}^{m+1,s} + r_{i-\frac{1}{2},j,k}^{m+1,s} + r_{i,j+\frac{1}{2},k}^{m+1,s} + r_{i,j-\frac{1}{2},k}^{m+1,s} + r_{i,j,k+\frac{1}{2}}^{m+1,s} + r_{i,j,k-\frac{1}{2}}^{m+1,s}\right)\right]\overline{n_{i,j,k}^{m+1,s}},\\
\overline{\delta n_{(i\pm\frac{1}{2},j,k),(i,j,k)}^{m+1,s}} &= r_{i\pm\frac{1}{2},j,k}^{m+1,s}\overline{n_{i\pm1,j,k}^{m+1,s}},\\
\overline{\delta n_{(i,j\pm\frac{1}{2},k),(i,j,k)}^{m+1,s}} &= r_{i,j\pm\frac{1}{2},k}^{m+1,s}\overline{n_{i,j\pm1,k}^{m+1,s}},\\
\overline{\delta n_{(i,j,k\pm\frac{1}{2}),(i,j,k)}^{m+1,s}} &= r_{i,j,k\pm\frac{1}{2}}^{m+1,s}\overline{n_{i,j,k\pm1}^{m+1,s}},
\end{aligned}$$

where:

$$r^{m+1,s}_{i\pm\frac{1}{2},j,k} = \frac{K\left(\theta\left(\psi^{m+1,s}_{i\pm\frac{1}{2},j,k}\right)\right)\Delta t}{(\Delta x)^2 L^{m+1,s}_{i\pm\frac{1}{2},j,k}},$$

$$r^{m+1,s}_{i,j\pm\frac{1}{2},k} = \frac{K\left(\theta\left(\psi^{m+1,s}_{i,j\pm\frac{1}{2},k}\right)\right)\Delta t}{(\Delta y)^2 L^{m+1,s}_{i,j\pm\frac{1}{2},k}},$$

$$r^{m+1,s}_{i,j,k\pm\frac{1}{2}} = \frac{K\left(\theta\left(\psi^{m+1,s}_{i,j,k\pm\frac{1}{2}}\right)\right)\Delta t}{(\Delta z)^2 L^{m+1,s}_{i,j,k\pm\frac{1}{2}}}.$$

B Detailed Numerical Scheme Formulations

The detailed formulation of Equation (21) derived by incorporating the relations in Appendix A are provided as follows:

The 1-D case:

$$\begin{aligned} n_i^{m+1,s+1} = &\left[1 - \left(r^{m+1,s}_{i+\frac{1}{2}} + r^{m+1,s}_{i-\frac{1}{2}}\right)\right] n_i^{m+1,s} + r^{m+1,s}_{i+\frac{1}{2}} n^{m+1,s}_{i+1} + r^{m+1,s}_{i-\frac{1}{2}} n^{m+1,s}_{i-1} \\ &+ \left\lfloor \hat{f}^{-1}_{\mathrm{NN}}\left[\left(r^{m+1,s}_{i+\frac{1}{2}} - r^{m+1,s}_{i-\frac{1}{2}}\right)\Delta z - \frac{\theta(\psi_i^{m+1,s}) - \theta(\psi_i^m)}{L_i^{m+1,s}} - \frac{S(\psi_i^{m+1,s})}{L_i^{m+1,s}}\right]\right\rfloor, \end{aligned}$$

where $\lfloor\cdot\rfloor$ is the floor function.

The 2-D case:

$$\begin{aligned} n_{i,j}^{m+1,s+1} = &\left[1 - \left(r^{m+1,s}_{i+\frac{1}{2},j} + r^{m+1,s}_{i-\frac{1}{2},j} + r^{m+1,s}_{i,j+\frac{1}{2}} + r^{m+1,s}_{i,j-\frac{1}{2}}\right)\right] n_{i,j}^{m+1,s} \\ &+ r^{m+1,s}_{i+\frac{1}{2},j} n^{m+1,s}_{i+1,j} + r^{m+1,s}_{i-\frac{1}{2},j} n^{m+1,s}_{i-1,j} + r^{m+1,s}_{i,j+\frac{1}{2}} n^{m+1,s}_{i,j+1} + r^{m+1,s}_{i,j-\frac{1}{2}} n^{m+1,s}_{i,j-1} \\ &+ \left\lfloor \hat{f}^{-1}_{\mathrm{NN}}\left[\left(r^{m+1,s}_{i,j+\frac{1}{2}} - r^{m+1,s}_{i,j-\frac{1}{2}}\right)\Delta z - \frac{\theta(\psi_{i,j}^{m+1,s}) - \theta(\psi_{i,j}^m)}{L_{i,j}^{m+1,s}} - \frac{S(\psi_{i,j}^{m+1,s})}{L_{i,j}^{m+1,s}}\right]\right\rfloor. \end{aligned}$$

The 3-D case:

$$\begin{aligned} n_{i,j,k}^{m+1,s+1} = &\left[1 - \left(r^{m+1,s}_{i+\frac{1}{2},j,k} + r^{m+1,s}_{i-\frac{1}{2},j,k} + r^{m+1,s}_{i,j+\frac{1}{2},k} + r^{m+1,s}_{i,j-\frac{1}{2},k} + r^{m+1,s}_{i,j,k+\frac{1}{2}} + r^{m+1,s}_{i,j,k-\frac{1}{2}}\right)\right] n_{i,j,k}^{m+1,s} \\ &+ r^{m+1,s}_{i+\frac{1}{2},j,k} n^{m+1,s}_{i+1,j,k} + r^{m+1,s}_{i-\frac{1}{2},j,k} n^{m+1,s}_{i-1,j,k} + r^{m+1,s}_{i,j+\frac{1}{2},k} n^{m+1,s}_{i,j+1,k} \\ &+ r^{m+1,s}_{i,j-\frac{1}{2},k} n^{m+1,s}_{i,j-1,k} + r^{m+1,s}_{i,j,k+\frac{1}{2}} n^{m+1,s}_{i,j,k+1} + r^{m+1,s}_{i,j,k-\frac{1}{2}} n^{m+1,s}_{i,j,k-1} \\ &+ \left\lfloor \hat{f}^{-1}_{\mathrm{NN}}\left[\left(r^{m+1,s}_{i,j,k+\frac{1}{2}} - r^{m+1,s}_{i,j,k-\frac{1}{2}}\right)\Delta z - \frac{\theta(\psi_{i,j,k}^{m+1,s}) - \theta(\psi_{i,j,k}^m)}{L_{i,j,k}^{m+1,s}} - \frac{S(\psi_{i,j,k}^{m+1,s})}{L_{i,j,k}^{m+1,s}}\right]\right\rfloor. \end{aligned}$$

C Proof of Theorem 5.1

To prove Theorem 5.1, we first need to introduce the following preliminary assumptions and results from [74] and [75].

Assumption 1: The objective function f is L-smooth.

Assumption 2: There exists a Polish probability space $(Z, \mathcal{Z}, \pi^Z)$ and $\eta \geq 0$ such that one of the following conditions holds:

(a) There exists a function $H : \mathbb{R}^d \times Z \to \mathbb{R}^d$ such that for any $x \in \mathbb{R}^d$,

$$\int_Z H(x,z)d\pi^Z(z) = \nabla f(x), \qquad \int_Z \|H(x,z) - \nabla f(x)\|^2 d\pi^Z(z) \leq \eta.$$

(b) There exists a function $\tilde{f} : \mathbb{R}^d \times Z \to \mathbb{R}$ such that for all $z \in Z$, $\tilde{f}(\cdot, z) \in C^1(\mathbb{R}^d, \mathbb{R})$ is L-smooth. Furthermore, there exists $x^* \in \mathbb{R}^d$ such that, for any $x \in \mathbb{R}^d$,

$$\int_Z \tilde{f}(x,z)d\pi^Z(z) = f(x), \quad \int_Z \nabla\tilde{f}(x,z)d\pi^Z(z) = \nabla f(x), \quad \int_Z \|\nabla\tilde{f}(x^*,z)\|^2 d\pi^Z(z) \leq \eta.$$

In this case, we define $H = \nabla\tilde{f}$.

Assumption 3: There exists $M \geq 0$ such that for any $x, y \in \mathbb{R}^d$,

$$\|\Sigma(x)^{1/2} - \Sigma(y)^{1/2}\| \leq M\|x - y\|.$$

Assumption 4: One of the following conditions holds:

(a) For Assumption 2(a): f is convex, *i.e.*, for any $x, y \in \mathbb{R}^d$,

$$\langle \nabla f(x) - \nabla f(y), x - y\rangle \geq 0,$$

and there exists a minimizer $x^* \in \arg\min_{x\in\mathbb{R}^d} f$.

(b) For Assumption 2(b): For all $z \in Z$, $\tilde{f}(\cdot, z)$ is convex, and there exists a minimizer $x^* \in \arg\min_{x\in\mathbb{R}^d} f$.

Under Assumptions 1 and 2, we introduce the sequence $\{X_n\}_{n\in\mathbb{N}}$ starting from $X_0 \in \mathbb{R}^d$ corresponding to SGD with non-increasing step sizes for any $n \in \mathbb{N}$ by:

$$X_{n+1} = X_n - \gamma(n+1)^{-\alpha}H(X_n, Z_{n+1}),$$

where $\gamma > 0$, $\alpha \in [0, 1]$, and $\{Z_n\}_{n\in\mathbb{N}}$ is a sequence of independent random variables on a probability space $(\Omega, \mathcal{F}, P)$ valued in $(Z, \mathcal{Z})$ such that for any $n \in \mathbb{N}$, Z_n is distributed according to π^Z. As [74] pointed out, the solution of the following SDE is a continuous counterpart of $\{X_n\}_{n\in\mathbb{N}}$:

$$\mathrm{d}\mathbf{X_t} = -(\gamma + t)^{-\alpha}\nabla f(\mathbf{X_t})\,\mathrm{d}t + \gamma(\gamma + t)^{-2\alpha}\Sigma(\mathbf{X_t})^{1/2}\,\mathrm{d}\mathbf{B_t},$$

where $\gamma_\alpha = \gamma^{1/(1-\alpha)}$ and $(B_t)_{t\geq 0}$ is a d-dimensional Brownian motion.

Given these preliminaries, we now leverage two established results as lemmas:

Lemma C.1 (Theorem 6 of [74]). *Let $\alpha, \gamma \in (0, 1)$, for $f \in C^2(\mathbb{R}^d, \mathbb{R})$, there exists $C \geq 0$ such that for any $T \geq 1$,*

$$\mathbb{E}[f(\mathbf{X_T})] - \min_{x\in\mathbb{R}^d} f \leq C\frac{(1 + \log(T))^2}{T^{\alpha(1-\alpha)}}.$$

Lemma C.2 (Equation 35 of [75]). *Suppose $\tilde{f}$ with an at most polynomially growing derivative is the "true" function learned by the neural network. Let $\kappa > 0$ be the polynomial growth rate, there exists $D \geq 0$ such that*

$$\|\tilde{f}(x) - \tilde{f}(y)\| \leq D\left(1 + \|x\|^{\kappa+2} + \|y\|^{\kappa+2}\right)\|x - y\|$$

holds.

With Lemmas C.1 and C.2, we are now ready to give the proof for Theorem 5.1 which accounts for the convergence of SGD:

Proof. To start, we have:

$$\begin{aligned}\|n_i^{m+1,s+1} - n_i^{m+1,s}\| &= \mathbb{E}\left[\|\hat{f}_{\text{NN}}^{-1}(\psi_i^{m+1,s+1}, \mathbf{X_T}) - \hat{f}_{\text{NN}}^{-1}(\psi_i^{m+1,s}, \mathbf{X_T})\|\right] \\ &\leq \mathbb{E}\left[\|\hat{f}_{\text{NN}}^{-1}(\psi_i^{m+1,s+1}, \mathbf{X_T}) - \tilde{f}(\psi_i^{m+1,s+1})\|\right] + \|\tilde{f}(\psi_i^{m+1,s+1}) - \tilde{f}(\psi_i^{m+1,s})\| \\ &\quad + \mathbb{E}\left[\|\tilde{f}(\psi_i^{m+1,s}) - \hat{f}_{\text{NN}}^{-1}(\psi_i^{m+1,s}, \mathbf{X_T})\|\right],\end{aligned}$$

where $\mathbf{X_T}$ is the weights of $\hat{f}_{\text{NN}}^{-1}$ optimized by SGD optimizer and $\tilde{f}$ is the true function learned by $\hat{f}_{\text{NN}}^{-1}$. Note that $f = \frac{1}{N}\|\hat{f}_{\text{NN}}^{-1} - \tilde{f}\|$ and there exists $\varepsilon > 0$, $\min_{x\in\mathbb{R}^d} f \leq \frac{\varepsilon}{6}$. Utilizing Lemma C.1, we have:

$$\mathbb{E}\left[\|\hat{f}_{\text{NN}}^{-1}(\psi_i^{m+1,s+1}, \mathbf{X_T}) - \tilde{f}(\psi_i^{m+1,s+1})\|\right] \leq CN\frac{(1+\log(T))^2}{T^{\alpha(1-\alpha)}} + \frac{\varepsilon}{6}$$

and

$$\mathbb{E}\left[\|\hat{f}_{\text{NN}}^{-1}(\psi_i^{m+1,s}, \mathbf{X_T}) - \tilde{f}(\psi_i^{m+1,s})\|\right] \leq CN\frac{(1+\log(T))^2}{T^{\alpha(1-\alpha)}} + \frac{\varepsilon}{6}.$$

Once the neural network $\hat{f}_{\text{NN}}^{-1}$ is trained for sufficient large epochs, for $\varepsilon > 0$, it follows that $\frac{(1+\log(T))^2}{T^{\alpha(1-\alpha)}} \leq \frac{\varepsilon}{6CN}$.

Next, for the term $\|\tilde{f}(\psi_i^{m+1,s+1}) - \tilde{f}(\psi_i^{m+1,s})\|$, it can be bounded using Lemma C.2 and the result $\|\psi_i^{m+1,s+1} - \psi_i^{m+1,s}\| \leq \frac{\varepsilon}{3D\left(1+\|\psi_i^{m+1,s+1}\|^{\kappa+2}+\|\psi_i^{m+1,s}\|^{\kappa+2}\right)}$ obtained from Theorem 4.1:

$$\begin{aligned}\|\tilde{f}(\psi_i^{m+1,s+1}) - \tilde{f}(\psi_i^{m+1,s})\| &\leq D\left(1 + \|\psi_i^{m+1,s+1}\|^{\kappa+2} + \|\psi_i^{m+1,s}\|^{\kappa+2}\right)\|\psi_i^{m+1,s+1} - \psi_i^{m+1,s}\| \\ &\leq \frac{\varepsilon}{3}.\end{aligned}$$

Therefore, it follows that

$$\|n_i^{m+1,s+1} - n_i^{m+1,s}\| \leq \frac{\varepsilon}{3} + \frac{\varepsilon}{3} + \frac{\varepsilon}{3} = \varepsilon,$$

which completes the proof. □

References

[1] National Academy of Engineering and National Academies of Sciences, Engineering, and Medicine, New Directions for Chemical Engineering, The National Academies Press, Washington, DC, 2022.

[2] E. Babaeian, M. Sadeghi, S. B. Jones, C. Montzka, H. Vereecken, M. Tuller, Ground, proximal, and satellite remote sensing of soil moisture, Reviews of Geophysics 57 (2) (2019) 530–616.

[3] D. Spelman, K.-D. Kinzli, T. Kunberger, Calibration of the 10HS soil moisture sensor for Southwest Florida agricultural soils, Journal of Irrigation and Drainage Engineering 139 (12) (2013) 965–971. doi:10.1061/(ASCE)IR.1943-4774.0000647.

[4] M. Saavoss, J. Majsztrik, B. Belayneh, J. Lea-Cox, E. Lichtenberg, Yield, quality and profitability of sensor-controlled irrigation: a case study of snapdragon (Antirrhinum majus L.) production, Irrigation Science 34 (5) (2016) 409–420.

[5] L. A. Richards, Capillary conduction of liquids through porous mediums, Physics 1 (5) (1931) 318–333.

[6] J.-G. Caputo, Y. A. Stepanyants, Front solutions of Richards' equation, Transport in Porous Media 74 (1) (2008) 1–20.

[7] R. E. Smith, K. Smettem, P. Broadbridge, D. Woolhiser, Infiltration Theory for Hydrologic Applications, American Geophysical Union, 2002.

[8] R. Haverkamp, M. Vauclin, J. Touma, P. Wierenga, G. Vachaud, A comparison of numerical simulation models for one-dimensional infiltration, Soil Science Society of America Journal 41 (2) (1977) 285–294.

[9] Y. Mualem, A new model for predicting the hydraulic conductivity of unsaturated porous media, Water Resources Research 12 (3) (1976) 513–522.

[10] M. T. Van Genuchten, A closed-form equation for predicting the hydraulic conductivity of unsaturated soils, Soil Science Society of America Journal 44 (5) (1980) 892–898.

[11] W. Gardner, Some steady-state solutions of the unsaturated moisture flow equation with application to evaporation from a water table, Soil Science 85 (4) (1958) 228–232.

[12] M. W. Farthing, F. L. Ogden, Numerical solution of Richards' equation: A review of advances and challenges, Soil Science Society of America Journal 81 (6) (2017) 1257–1269.

[13] V. S. Sizikov, et al., Well-posed, ill-posed, and intermediate problems with applications, De Gruyter, 2011.

[14] L. C. Evans, Partial differential equations, Vol. 19, American Mathematical Soc., 2010.

[15] W. Merz, P. Rybka, Strong solutions to the Richards equation in the unsaturated zone, Journal of Mathematical Analysis and Applications 371 (2) (2010) 741–749.

[16] O. Misiats, K. Lipnikov, Second-order accurate monotone finite volume scheme for Richards' equation, Journal of Computational Physics 239 (2013) 123–137.

[17] N. Abdellatif, C. Bernardi, M. Touihri, D. Yakoubi, A priori error analysis of the implicit Euler, spectral discretization of a nonlinear equation for a flow in a partially saturated porous media, Advances in Pure and Applied Mathematics 9 (1) (2018) 1–27.

[18] P. R. Day, J. N. Luthin, A numerical solution of the differential equation of flow for a vertical drainage problem, Soil Science Society of America Journal 20 (4) (1956) 443–447.

[19] M. A. Celia, E. T. Bouloutas, R. L. Zarba, A general mass-conservative numerical solution for the unsaturated flow equation, Water Resources Research 26 (7) (1990) 1483–1496.

[20] M. W. Farthing, F. L. Ogden, Numerical solution of Richards' equation: A review of advances and challenges, Soil Science Society of America journal 81 (6) (2017) 1257–1269.

[21] B. Belfort, A. Younes, M. Fahs, F. Lehmann, On equivalent hydraulic conductivity for oscillation-free solutions of Richard's equation, Journal of Hydrology 505 (2013) 202–217.

[22] A. M. Ireson, R. J. Spiteri, M. P. Clark, S. A. Mathias, A simple, efficient, mass-conservative approach to solving Richards' equation (openre, v1. 0), Geoscientific Model Development 16 (2) (2023) 659–677.

[23] D. Or, P. Lehmann, S. Assouline, Natural length scales define the range of applicability of the Richards equation for capillary flows, Water Resources Research 51 (9) (2015) 7130–7144.

[24] K. Roth, Scaling of water flow through porous media and soils, European Journal of Soil Science 59 (1) (2008) 125–130.

[25] H.-J. Vogel, O. Ippisch, Estimation of a critical spatial discretization limit for solving Richards' equation at large scales, Vadose Zone Journal 7 (1) (2008) 112–114.

[26] K. Rathfelder, L. M. Abriola, Mass conservative numerical solutions of the head-based Richards equation, Water Resources Research 30 (9) (1994) 2579–2586.

[27] Z. Song, Z. Jiang, A data-driven modeling approach for water flow dynamics in soil, in: A. C. Kokossis, M. C. Georgiadis, E. Pistikopoulos (Eds.), 33rd European Symposium on Computer Aided Process Engineering, Vol. 52 of Computer Aided Chemical Engineering, Elsevier, 2023, pp. 819–824.

[28] Z. Song, Z. Jiang, A data-driven random walk approach for solving water flow dynamics in soil systems, in: Proceedings of Foundations of Computer-Aided Process Operations and Chemical Process Control (FOCAPO/CPC), 2023, pp. 1–6.

[29] R. Eymard, M. Gutnic, D. Hilhorst, The finite volume method for Richards equation, Computational Geosciences 3 (3) (1999) 259–294.

[30] W. Lai, F. L. Ogden, A mass-conservative finite volume predictor–corrector solution of the 1D Richards' equation, Journal of Hydrology 523 (2015) 119–127.

[31] S. Bassetto, C. Cancès, G. Enchéry, Q.-H. Tran, On several numerical strategies to solve Richards' equation in heterogeneous media with finite volumes, Computational Geosciences 26 (5) (2022) 1297–1322.

[32] D. Caviedes-Voullième, P. Garcı, J. Murillo, et al., Verification, conservation, stability and efficiency of a finite volume method for the 1D Richards equation, Journal of Hydrology 480 (2013) 69–84.

[33] G. Manzini, S. Ferraris, Mass-conservative finite volume methods on 2-D unstructured grids for the Richards' equation, Advances in Water Resources 27 (12) (2004) 1199–1215.

[34] L. Bergamaschi, M. Putti, Mixed finite elements and Newton-type linearizations for the solution of Richards' equation, International Journal for Numerical Methods in Engineering 45 (8) (1999) 1025–1046.

[35] I. S. Pop, F. Radu, P. Knabner, Mixed finite elements for the Richards' equation: linearization procedure, Journal of Computational and Applied Mathematics 168 (1-2) (2004) 365–373.

[36] K. Mitra, I. S. Pop, A modified L-scheme to solve nonlinear diffusion problems, Computers & Mathematics with Applications 77 (6) (2019) 1722–1738.

[37] G. Albuja, A. I. Ávila, A family of new globally convergent linearization schemes for solving Richards' equation, Applied Numerical Mathematics 159 (2021) 281–296.

[38] V. Casulli, P. Zanolli, A nested Newton-type algorithm for finite volume methods solving Richards' equation in mixed form, SIAM Journal on Scientific Computing 32 (4) (2010) 2255–2273.

[39] V. Zaburdaev, S. Denisov, J. Klafter, Lévy walks, Reviews of Modern Physics 87 (2) (2015) 483.

[40] A. Zoia, M.-C. Néel, A. Cortis, Continuous-time random-walk model of transport in variably saturated heterogeneous porous media, Physical Review E 81 (3) (2010) 031104.

[41] N. Suciu, Diffusion in Random Fields: Applications to Transport in Groundwater, Springer, 2019.

[42] N. Suciu, D. Illiano, A. Prechtel, F. A. Radu, Global random walk solvers for fully coupled flow and transport in saturated/unsaturated porous media, Advances in Water Resources 152 (2021) 103935.

[43] N. Suciu, C. Vamoş, I. Turcu, C. Pop, L. Ciortea, Global random walk modelling of transport in complex systems, Computing and Visualization in Science 12 (2) (2009) 77–85.

[44] C. Vamos, N. Suciu, H. Vereecken, O. Nitzsche, H. Hardelauf, Global random walk simulations of diffusion, in: Scientific Computing, Validated Numerics, Interval Methods, Springer, 2001, pp. 343–354.

[45] E. Orouskhani, S. Sahoo, B. Agyeman, S. Bo, J. Liu, Impact of sensor placement in soil water estimation: a real-case study, Irrigation Science 41 (3) (2023) 395–411.

[46] R. Zarba, A Numerical Investigation of Unsaturated Flow, Massachusetts Institute of Technology, Department of Civil Engineering, Cambridge, MA, 1988.

[47] M. A. Celia, R. Zarba, A comparative study of numerical solutions for unsaturated flow, in: S. N. Atluri, G. Yagawa (Eds.), Computational Mechanics '88, Springer Berlin Heidelberg, Berlin, Heidelberg, 1988, pp. 1659–1662.

[48] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Networks 4 (2) (1991) 251–257.

[49] A. Pinkus, Approximation theory of the MLP model in neural networks, Acta Numerica 8 (1999) 143–195. doi:10.1017/S0962492900002919.

[50] N. Suciu, D. Illiano, A. Prechtel, F. A. Radu, Richards equation (2021). doi:10.5281/zenodo.4709693.
URL https://github.com/PMFlow/RichardsEquation

[51] I. B. V. Da Silva, P. J. Adeodato, PCA and Gaussian noise in MLP neural network training improve generalization in problems with small and unbalanced data sets, in: The 2011 International Joint Conference on Neural Networks, IEEE, 2011, pp. 2664–2669.

[52] J. Brandstetter, D. E. Worrall, M. Welling, Message passing neural PDE solvers, CoRR abs/2202.03376 (2022). arXiv:2202.03376.
URL https://arxiv.org/abs/2202.03376

[53] P. Y. Lu, S. Kim, M. Soljačić, Extracting interpretable physical parameters from spatiotemporal systems using unsupervised learning, Phys. Rev. X 10 (2020) 031056.
URL https://link.aps.org/doi/10.1103/PhysRevX.10.031056

[54] D. Gąsiorowski, T. Kolerski, Numerical solution of the two-dimensional Richards equation using alternate splitting methods for dimensional decomposition, Water 12 (6) (2020) 1780.

[55] F. T. Tracy, Clean two-and three-dimensional analytical solutions of Richards' equation for testing numerical solvers, Water Resources Research 42 (8) (2006).

[56] M. Berardi, F. Difonzo, M. Vurro, L. Lopez, The 1D Richards' equation in two layered soils: a filippov approach to treat discontinuities, Advances in Water Resources 115 (2018) 264–272.

[57] A. L. Maas, A. Y. Hannun, A. Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in: Proc. ICML, Vol. 30, Atlanta, Georgia, USA, 2013, p. 3.

[58] T. Bandai, T. A. Ghezzehei, 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) (2021) e2020WR027642.

[59] R. G. Hills, I. Porro, D. B. Hudson, P. J. Wierenga, Modeling one-dimensional infiltration into very dry soils: 1. model development and evaluation, Water Resources Research 25 (6) (1989) 1259–1269.

[60] R. F. Carsel, R. S. Parrish, Developing joint probability distributions of soil water retention characteristics, Water Resources Research 24 (5) (1988) 755–769.

[61] J. Šimůnek, M. T. Van Genuchten, M. Šejna, Recent developments and applications of the HYDRUS computer software packages, Vadose Zone Journal 15 (7) (2016).

[62] R. Feddes, H. Zaradny, Model for simulating soil-water content considering evapotranspiration — comments, Journal of Hydrology 37 (3) (1978) 393–397.

[63] B. T. Agyeman, S. Bo, S. R. Sahoo, X. Yin, J. Liu, S. L. Shah, Soil moisture map construction using microwave remote sensors and sequential data assimilation (2020). arXiv:2010.07037.
URL https://arxiv.org/abs/2010.07037

[64] R. Mattey, S. Ghosh, A novel sequential method to train physics informed neural networks for Allen Cahn and Cahn Hilliard equations, Computer Methods in Applied Mechanics and Engineering 390 (2022) 114474. doi:10.1016/j.cma.2021.114474.
URL http://dx.doi.org/10.1016/j.cma.2021.114474

[65] R. Brecht, L. Bakels, A. Bihlo, A. Stohl, Improving trajectory calculations by flexpart 10.4+ using single-image super-resolution, Geoscientific Model Development 16 (8) (2023) 2181–2192.

[66] E. Gerolymatou, I. Vardoulakis, R. Hilfer, Modelling infiltration by means of a nonlinear fractional diffusion model, Journal of Physics D: Applied Physics 39 (18) (2006) 4104.

[67] Y. Jabbari, E. Tsotsas, C. Kirsch, A. Kharaghani, Determination of the moisture transport coefficient from pore network simulations of spontaneous imbibition in capillary porous media, Chemical Engineering Science 207 (2019) 600–610.

[68] A. Ashari, H. Vahedi Tafreshi, A two-scale modeling of motion-induced fluid release from thin fibrous porous media, Chemical Engineering Science 64 (9) (2009) 2067–2075.

[69] A. H. Tavangarrad, B. Mohebbi, C. Qin, S. M. Hassanizadeh, R. Rosati, J. Claussen, B. Blümich, Continuum-scale modeling of water infiltration into a stack of two thin fibrous layers and their inter-layer space, Chemical Engineering Science 207 (2019) 769–779.

[70] M. Khammar, Y. Xu, Continuous liquid extraction from saturated granular materials, Chemical Engineering Science 173 (2017) 390–401.

[71] Soledad Fioroni, A. E. Larreteguy, G. B. Savioli, An OpenFOAM Application for Solving the Black Oil Problem, Mathematical Models and Computer Simulations 13 (5) (2021) 907–918.

[72] R. Cockett, S. Kang, L. J. Heagy, A. Pidlisecky, D. W. Oldenburg, SimPEG: An open source framework for simulation and gradient based parameter estimation in geophysical applications, Computers & Geosciences (2015).

[73] Z. Song, Z. Jiang, AI4Soil (2024).
URL `https://github.com/taekwonzysong/AI4Soil`

[74] X. Fontaine, V. D. Bortoli, A. Durmus, Convergence rates and approximation results for SGD and its continuous-time counterpart, in: M. Belkin, S. Kpotufe (Eds.), Proceedings of Thirty Fourth Conference on Learning Theory, Vol. 134 of Proceedings of Machine Learning Research, PMLR, 2021, pp. 1965–2058.

[75] J. Berner, D. Elbrächter, P. Grohs, A. Jentzen, Towards a regularity theory for ReLU networks – chain rule and global error estimates, in: 2019 13th International conference on Sampling Theory and Applications (SampTA), 2019, pp. 1–5. `doi:10.1109/SampTA45681.2019.9031005`.