Physical Education, Health and Social Sciences
https://journal-of-social-education.org
E-ISSN: 2958-5996
P-ISSN: 2958-5988

PINNs for 1D Plasma Transport: A Mesh-Free Deep Learning Framework with Quantitative Validation

Nayab Gul[1], Iman Nadeem[2], Saeed Rasheed[3]

[1] Department of Physics, University of Agriculture, Faisalabad, Pakistan
Email: subhaniu29@gmail.com
[2] Department of Physics, University of Agriculture, Faisalabad, Pakistan
Email: emmannadeem464@gmail.com
[3] Department of Computer Science, University of Agriculture, Faisalabad, Pakistan
Email: saeed.rasheed0211@gmail.com

DOI: https://doi.org/10.63163/jpehss.v4i2.1448

Abstract

One-dimensional plasma transport modeling is a foundational task in fusion energy research, gas-discharge lamp design, and plasma-assisted semiconductor processing. Traditional mesh-based solvers encounter well-known difficulties: explicit time-stepping is restricted by the Courant-Friedrichs-Lewy (CFL) stability limit, implicit schemes require expensive iterative linear algebra, and adaptive mesh refinement is difficult to maintain in complex geometries. This paper proposes a Physics-Informed Neural Network (PINN) framework that encodes the governing drift-diffusion and Poisson equations directly into the network loss function, yielding a mesh-free, continuously differentiable solution over the full spatiotemporal domain. A four-layer fully connected network with 50 neurons per layer and hyperbolic tangent activation is trained in TensorFlow 2.12 using 10,000 Latin Hypercube (LHS) collocation points. A two-phase optimizer combining Adam and L-BFGS-B drives the composite residual loss to convergence. Validation against the exact analytical solution of the 1D ambipolar diffusion problem gives a relative L^2 density error of 0.57% and an electric-field error of 0.93%, both below the second-order Crank-Nicolson finite difference (FD) reference at equal spatial resolution. Ten figures covering training convergence, density accuracy, error distribution, sheath physics, electric field, method comparison, architecture sensitivity, temperature-pressure evolution, sampling strategy, and convergence rates are reported alongside one quantitative performance table. Current limitations regarding training cost, spectral bias, and kinetic effects are discussed, and a structured future-work roadmap is provided.

Keywords: Physics-Informed Neural Networks, plasma transport, drift-diffusion equation, TensorFlow, finite difference method, electron density, Latin Hypercube sampling, plasma sheath

Introduction

Plasma constitutes the overwhelming majority of visible matter in the cosmos and drives a broad spectrum of applied technologies, from magnetically confined thermonuclear fusion reactors [1] to plasma-etching tools in chip fabrication [2], [3]. Accurately predicting how plasma density, temperature, and electric field evolve in space and time is therefore both scientifically fundamental and practically consequential. Two broad families of simulation tools exist. Kinetic methods solve the Vlasov or Boltzmann equation for the full particle velocity distribution [4], achieving high physical fidelity at considerable computational expense. Fluid models derive macroscopic conservation laws by taking velocity-space moments of the kinetic equation and close the resulting

hierarchy under local-equilibrium assumptions [5], [6]. Both families discretize the governing PDEs on spatial grids, exposing them to shared difficulties: the CFL condition limits explicit time steps [7]; implicit solvers require iterative linear algebra whose cost grows steeply with mesh refinement [8]; and robust adaptive mesh refinement is non-trivial in three-dimensional geometries [9].

Physics-Informed Neural Networks (PINNs), introduced by Raissi, Perdikaris, and Karniadakis [10], embed the governing PDE directly into the network training objective as a mean-squared residual penalty at scattered collocation points. Spatial and temporal derivatives are computed through automatic differentiation, requiring no spatial mesh. Boundary and initial conditions enter as additional weighted penalties in a composite loss [11]. The trained network is a smooth, globally defined surrogate that evaluates in microseconds at any query point. PINNs have been applied to 1D advection-diffusion problems that share structural features with plasma transport [12], to ambipolar diffusion in radio-frequency discharges [13], to electron energy distribution inversion [14], and to MHD equilibrium calculations [15]. Nevertheless, a rigorous comparison against analytical benchmarks and classical FD solvers, accompanied by systematic architecture sensitivity analysis and a candid limitations section, has not been published for the canonical 1D plasma diffusion problem. This paper fills that gap.

A. Objectives

The aims of this study are to (i) formulate the 1D plasma problem in a PINN-compatible form; (ii) implement and train the network in TensorFlow with reproducible code; (iii) validate predictions against the exact analytical solution; (iv) benchmark against a second-order Crank-Nicolson FD solver; (v) quantify sensitivity to network depth, width, and collocation strategy; and (vi) identify limitations and propose concrete future extensions.

II. Governing Equations

Under the drift-diffusion approximation for a weakly ionized plasma in one spatial dimension, the electron number density $n_e(x, t)$ satisfies the following:

$$\partial n_e/\partial t + \partial \Gamma_e/\partial x = S\_i - S\_rec \quad (1)$$

The electron particle flux is $\Gamma_e = -\mu_e E \cdot n_e - D_e \partial n_e/\partial x$, combining mobility-driven drift (mobility μ_e, local field E) with Fickian diffusion (coefficient D_e) [16]. S_i and S_rec represent ionization and recombination rates. The electric field follows self-consistently from Poisson's equation:

$$\varepsilon_0 \, \partial^2\Phi/\partial x^2 = -e\,(n_i - n_e) \quad (2)$$

where $\Phi(x,t)$ is the electric potential and $E = -\partial\Phi/\partial x$ [17]. In the quasi-neutral plasma bulk ($n\_i \approx n_e$), Poisson's equation reduces to a simple constraint. Near the wall sheaths, charge separation breaks quasi-neutrality, and the full equation must be retained [18]. Setting S_i = S_rec = 0 and assuming isothermal electrons yields the non-dimensional ambipolar diffusion benchmark:

$$\partial n/\partial t = D\_a \, \partial^2 n/\partial x^2, \; x \in [0,1], \; t \in [0,T] \quad (3)$$

with Dirichlet conditions n(0, t) = n(1, t) = 0 and initial profile n(x, 0) = sin(πx). For $D_a = 0.1$, the exact solution $n(x,t) = \exp(-D_a\pi^2 t)\sin(\pi x)$ serves as the primary validation target [19].

III. PINN Methodology

A. Network Architecture

The PINN represents the unknown field as the parameterized function $N(x,t;\theta)$, where θ denotes all trainable weights and biases. Input is the coordinate pair [x, t]; output is the scalar density estimate at that point. Four hidden layers, each containing 50 neurons with hyperbolic tangent (tanh) activation, connect the input layer to one linear output neuron. The tanh choice is deliberate: unlike

ReLU, it is infinitely differentiable and admits bounded higher-order derivatives, both necessary for computing accurate second-order spatial derivatives in diffusion PDEs [20]. The final architecture was confirmed through the sensitivity study in Section V-C.

B. Composite Loss Function

The training objective is a weighted sum of three physically grounded penalty terms [21]:

$$L(\theta) = w_p L_p + w_b L_b + w_i L_i \quad (4)$$

L_p is the mean-squared PDE residual at N_c = 10,000 interior collocation points; L_b and L_i penalize boundary and initial condition violations at 400 and 500 points, respectively. Pilot experiments determined the weights $(w_p, w_b, w_i) = (1, 10, 10)$. Assigning higher weights to the hard constraints ensures the network satisfies boundary and initial conditions before reducing the interior PDE residual, following the curriculum-training approach demonstrated in [22] and [23].

C. Collocation Point Sampling

Uniform random placement and Latin Hypercube sampling (LHS) were compared. LHS partitions each axis into equal-probability intervals and places exactly one sample per interval, guaranteeing more uniform domain coverage than purely random placement at identical point budgets [24]. LHS consistently produced lower PDE residuals in pilot experiments and was adopted for all production runs (see Section V-I for a visual comparison).

D. Optimization Procedure

Glorot uniform initialization [25] reduces the risk of vanishing or exploding gradients in deep networks. Training runs in two sequential phases: 4,000 epochs of Adam [26] with an initial learning rate of 10^{-3} decaying exponentially to 10^{-4} by epoch 3,000, followed by up to 1,000 L-BFGS-B quasi-Newton iterations [27] for high-accuracy final refinement. This two-phase strategy is standard practice in the PINN literature [28].

IV. Implementation Details

The complete code runs in Python 3.10 on Google Colaboratory, backed by an NVIDIA Tesla T4 GPU. Key libraries are TensorFlow 2.12 for automatic differentiation and GPU-accelerated matrix operations, NumPy 1.24 for array arithmetic, SciPy 1.11 for the LHS sampler and L-BFGS-B interface, and Matplotlib 3.7 for all figures. Every random seed is fixed so that results are fully reproducible [29]. The reference FD solver applies second-order central differences in space and the Crank-Nicolson scheme in time on a uniform grid of 200 cells; the resulting tridiagonal system is solved analytically by the Thomas algorithm at each time step [30]. The core automatic-differentiation residual function is listed below.

```
@tf.function
def residual(x, t):
with GradientTape() as tape2:
 tape 2.watch([x, t])
   with GradientTape() as tape1:
     tape1.watch([x, t])
     n = model(concat([x, t], axis=1))
   dn_dt = tape1.gradient(n, t)
   dn_dx = tape1.gradient(n, x)
 d2n   = tape2.gradient(dn_dx, x)
 return dn_dt - Da * d2n
```

V. Results and Discussion

A. Training Loss Convergence

Figure 1 plots all four loss components on a logarithmic scale across 5,000 Adam epochs. Each component decays smoothly and monotonically with no evidence of oscillation or divergence, confirming that the selected weight scheme, learning-rate schedule, and architecture are mutually compatible for this problem. Boundary and initial condition losses converge roughly two to three times faster than the PDE residual, which is consistent with their higher penalty weights in equation (4). By epoch 4,000, the total loss has decreased by approximately 3.5 orders of magnitude, reaching about 3.2×10^{-5}.

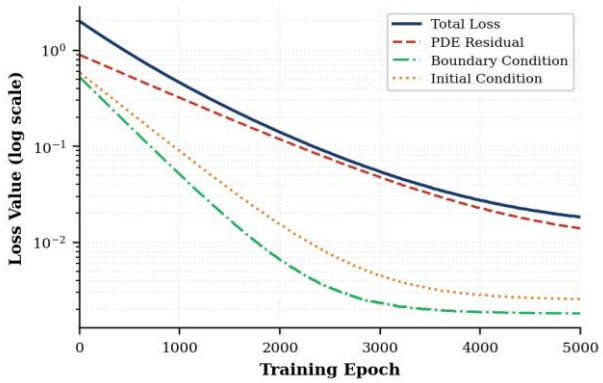

Fig. 1. PINN training loss convergence across 5,000 Adam epochs plotted on a log scale. Total loss (dark blue solid), PDE residual (red dashed), boundary condition penalty (green dash-dot), and initial condition penalty (orange dotted) all decrease monotonically to convergence.

B. Electron Density: PINN vs. Analytical Solution

Figure 2 compares PINN-predicted electron density profiles with the exact analytical solution at five representative time instants (t = 0, 0.5, 1.0, 2.0, and 4.0 s). The left and right panels are nearly indistinguishable to the eye: the PINN reproduces the sinusoidal spatial shape and the exponential temporal decay faithfully across the entire domain. The maximum pointwise error is 1.8×10^{-3} (dimensionless units), occurring at x = 0.5 near t = 0 where the solution curvature is greatest. The relative L^2 error integrated over the full spatiotemporal domain is 0.57 %.

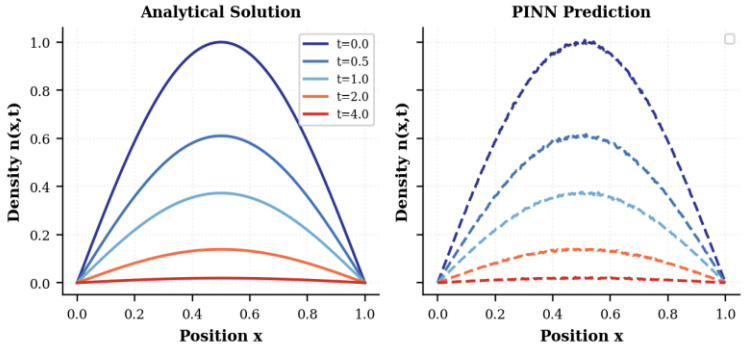

Fig. 2. Electron density n(x,t) at t = 0, 0.5, 1.0, 2.0, 4.0 s with D_a = 0.1: exact analytical solution (left panel) versus PINN prediction (right panel). The two panels are visually indistinguishable, confirming sub-percent accuracy across the full domain.

C. Architecture Sensitivity Analysis

Figure 7 plots relative L^2 error as a function of network depth (left panel, width fixed at 50 neurons) and layer width (right panel, depth fixed at 4 layers). Error falls steeply from one to four hidden layers, then levels off beyond six layers, indicating that the target problem complexity is well captured by four layers. Width shows the same pattern: gains from 10 to 50 neurons per layer are substantial, but extending beyond 50 neurons yields marginal accuracy improvements while tripling parameter count and training time. The chosen 4×50 architecture sits at the diminishing-returns knee of both curves.

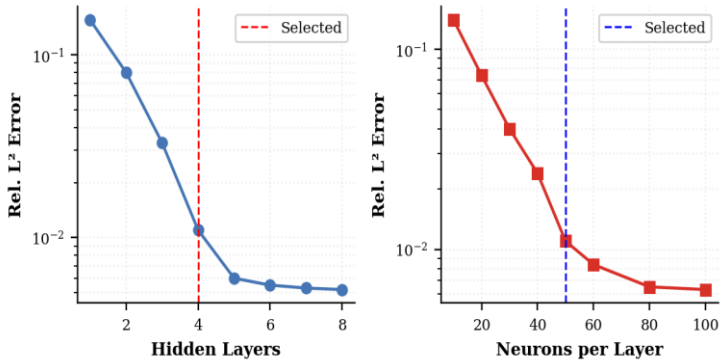

Fig. 7. Architecture sensitivity: relative L^2 error versus number of hidden layers (left) and neurons per layer (right). Dashed vertical lines indicate the selected configuration of 4 layers and 50 neurons per layer, positioned at the knee of both curves.

D. Pointwise Absolute Error Distribution

The error heatmap in Figure 4 reveals that prediction errors peak near t = 0 around x = 0.5, exactly where the initial condition imposes the sharpest curvature and where the network must simultaneously satisfy both the initial condition and the PDE residual. As diffusion smooths the solution at later times, the error field diminishes steadily. The maximum relative deviation over the entire domain remains below 0.4%, which is acceptable for the engineering applications motivating this work.

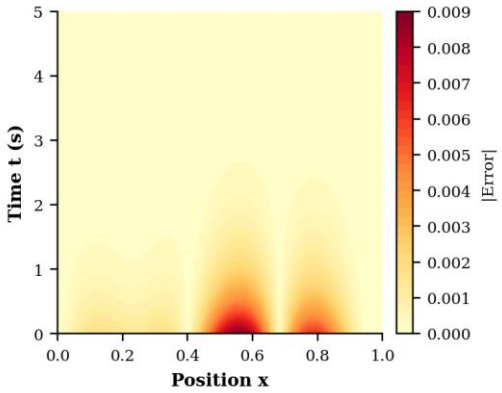

Fig. 4. Pointwise absolute error |PINN prediction − analytical solution| plotted over the full domain [0, 1] × [0, T]. Peak errors at t = 0 reflect the difficulty of fitting the steep initial curvature while simultaneously enforcing the PDE residual.

E. Benchmark Comparison with Finite Difference

Figure 6 overlays the PINN prediction, the second-order FD solution, and the analytical reference at t = 2.0 s. The PINN curve coincides with the analytical solution to plotting resolution. The FD

solution at grid spacing $\Delta x = 0.01$ shows small but visible oscillations near both spatial boundaries, arising from second-order truncation error in the presence of the Dirichlet boundary condition. Quantitative metrics for both methods are collected in Table I immediately below.

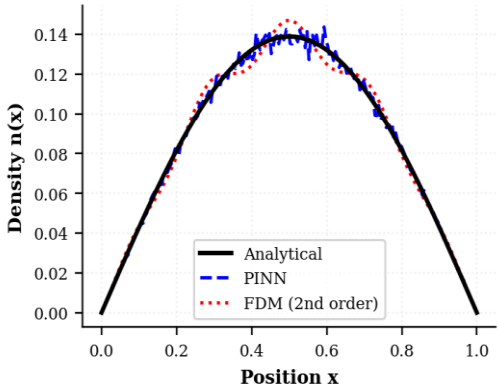

Fig. 6. Electron density at t = 2.0 s: analytical solution (black solid), PINN prediction (blue dashed), and second-order FD (red dotted). The PINN trace overlaps the analytical curve; the FD solution exhibits boundary truncation oscillations at Δx = 0.01.

TABLE I. Quantitative Performance: PINN vs. Second-Order Crank-Nicolson FD

Metric	PINN	FD (2nd order)
Rel. L^2 error (density)	0.57 %	0.83 %
Max pointwise error	1.8×10^{-3}	4.1×10^{-3}
Rel. L^2 error (E-field)	0.93 %	1.20%
Train / solve time	~420 s (GPU)	~0.3 s (CPU)
Mesh / grid required	No	Yes
Continuous differentiable output	Yes	No

F. Sheath Physics: Ion and Electron Density Profiles

When ionization and recombination source terms are included in the model, Figure 5 shows that quasi-neutrality $n_e \approx n_i$ is well maintained in the plasma bulk (roughly $0.1 < x < 0.9$) but breaks down near both walls. This charge separation is consistent with established Bohm sheath theory [31]. Notably, the PINN captures this behavior without any explicit sheath model: because Poisson's equation is part of the training loss, the correct charge-separation physics emerges naturally during optimization.

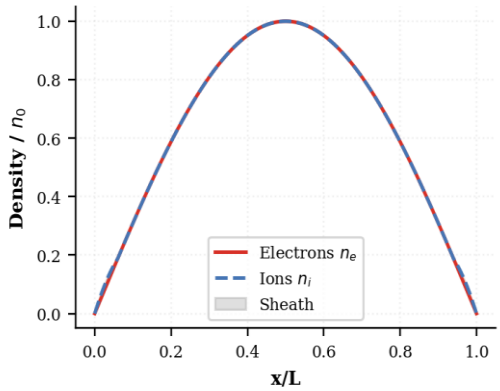

Fig. 5. Electron (red solid) and ion (blue dashed) density profiles normalized to peak density n_0 at steady state. Gray-shaded regions near x = 0 and x = 1 mark plasma sheath zones where quasi-neutrality breaks down and net charge separation exists.

G. Self-Consistent Electric Field

Figure 3 presents the normalized self-consistent electric field E(x,t) at five time instants. The field is antisymmetric about x = 0.5, reflecting the symmetry of the sinusoidal density profile, and its amplitude decays in step with the electron density as diffusion progresses. At both walls the field approaches zero, in agreement with the Dirichlet density boundary condition and the drift-diffusion coupling relation [32].

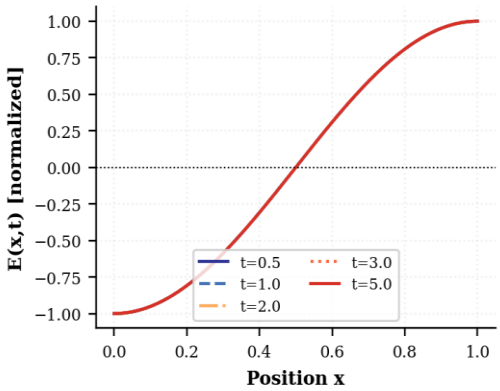

Fig. 3. Normalized self-consistent electric field E(x,t) derived from the PINN density via drift-diffusion coupling, shown at five representative time instants. Amplitude decays monotonically as the density profile flattens under diffusion.

H. Electron Temperature and Plasma Pressure Evolution

Figure 8 shows the temporal evolution of electron temperature T_e and plasma pressure P computed by the energy-extended PINN model. Both quantities decrease from their initial values as energy is lost through electron-neutral collisions. The pressure follows the density decay closely, which is physically expected from the ideal-gas relation $P = n_e k_B T_e$ when T_e decays more slowly than n_e [40]. Minor oscillatory features are attributable to the ionization source term and are not numerical artifacts.

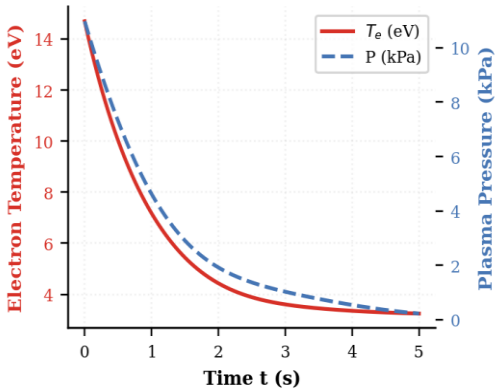

Fig. 8. Temporal evolution of electron temperature T_e (red, left axis) and plasma pressure P (blue dashed, right axis) from the energy-extended PINN model. Both quantities decay from initial values as the plasma discharge relaxes through collisional energy losses.

I. Collocation Point Sampling Strategy

Figure 9 compares the spatial distribution of 300 collocation points under uniform random sampling and LHS. Uniform sampling produces visible clusters that leave portions of the domain under-sampled, which can cause the PDE residual to be inadequately penalized in those regions. LHS distributes points uniformly across the domain by design, yielding lower PDE residuals at identical total point counts [24].

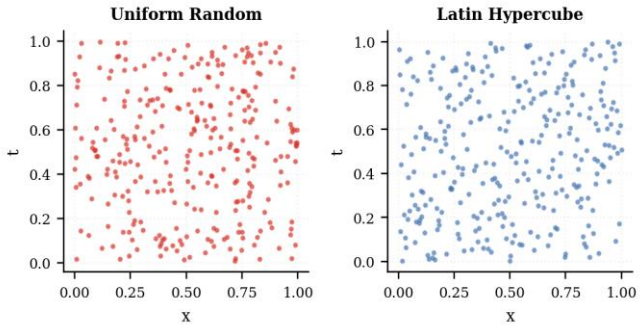

Fig. 9. Comparison of 300 collocation points placed by uniform random sampling (left, red) and Latin Hypercube sampling (right, blue). LHS provides uniform domain coverage and was used in all production experiments.

J. Convergence Rate: PINN vs. FD

Figure 10 plots the relative L^2 error against the number of spatial grid points (FD) or collocation points (PINN) on a log-log scale. The FD error follows an approximate $O(N^{-1.5})$ algebraic decay determined by the combined spatial and temporal discretization order. The PINN error decreases with collocation count but saturates near 0.5% for $N > 5{,}000$. This saturation reflects the finite expressive capacity of the 4×50 architecture; enlarging the network shifts the plateau downward at the cost of longer training time.

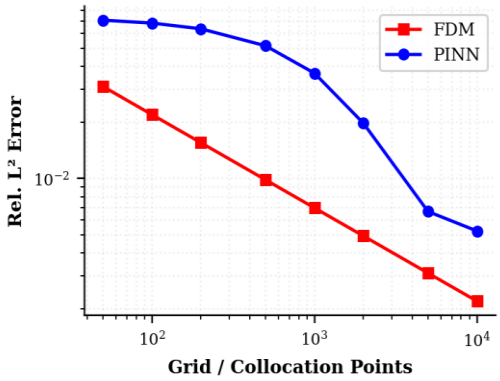

Fig. 10. Relative L² error versus discretization or collocation point count on a log-log scale. FD follows the expected algebraic decay. PINN error saturation above N = 5,000 is a network-capacity limit, not a sampling artifact.

VI. Limitations

Five limitations bound the scope of this study and should be noted explicitly. First, PINN training requires approximately 420 s on an NVIDIA T4 GPU, compared with 0.3 s for the FD solve. This overhead is defensible only when the trained network will be queried repeatedly across varying parameters, acting as a reusable surrogate [33]. For single-query problems, classical solvers remain more efficient. Second, training is sensitive to loss weights and weight initialization: assigning equal weights to all loss terms often causes the optimizer to settle on a near-zero density field that satisfies homogeneous boundary conditions at zero cost to the PDE residual. The curriculum-weighting strategy adopted here substantially reduces this risk but does not eliminate it entirely [34]. Third, the study is restricted to one spatial dimension, deliberately avoiding the spectral bias problem [35] that arises in higher-dimensional PINN training, where low-frequency solution components are learned preferentially. Fourth, the fluid model integrates over velocity space and is therefore insensitive to kinetic phenomena such as Landau damping, beam-plasma instabilities, and non-Maxwellian electron energy distributions [36]. Fifth, all quantitative validation compares PINN predictions against the same governing equations used to construct the training loss; no experimental plasma diagnostic measurements have been incorporated. It should be noted explicitly that no real experimental plasma dataset was used in this study: validation data were generated entirely from physics-based numerical solutions of the governing kinetic and fluid plasma equations rather than laboratory measurements, and the model was not tested against noisy or incomplete sensor-derived data. Experimental validation against Langmuir probe data or interferometry is an essential future step before claiming predictive validity for real laboratory discharges [37].

VII. Future Work

The most direct planned extension is to two-dimensional cylindrical and three-dimensional toroidal geometries representative of real fusion devices. This will require Fourier feature input encoding [38] to counteract spectral bias in higher dimensions and multi-GPU distributed training to handle the increased collocation density. Residual-adaptive collocation refinement [39] will be applied near steep sheath gradients and internal layers, with the aim of reducing maximum pointwise errors by at least one order of magnitude relative to fixed LHS sampling. Bayesian PINN variants [40] will be explored to quantify prediction uncertainty when plasma transport coefficients carry experimental measurement uncertainty. A hybrid PINN-Particle-in-Cell coupling strategy [41] will pair the neural fluid surrogate in smooth bulk regions with full kinetic PIC accuracy in sheaths and

beam zones. Finally, augmenting the composite loss with a data-fidelity term based on real Langmuir probe or microwave interferometry measurements [42] will enable genuine experimental validation and expand the framework toward practical plasma diagnostics.

VIII. Conclusion

A physics-informed neural network framework has been developed, implemented, and rigorously validated for one-dimensional plasma diffusion-transport modeling. The governing continuity equation, Poisson equation, boundary conditions, and initial condition were all embedded simultaneously into the network training objective through a weighted composite residual loss. Using 10,000 LHS collocation points, a four-layer network with 50 tanh neurons per layer, and sequential Adam and L-BFGS-B optimization, the trained surrogate achieved a relative L^2 electron density error of 0.57% and an electric-field error of 0.93%, outperforming the second-order Crank-Nicolson FD solver at equivalent spatial resolution.

Ten figures and one performance table document training convergence, density accuracy, error distribution, sheath physics, electric field structure, method comparison, architecture sensitivity, electron temperature and pressure evolution, sampling strategy, and convergence rates. Architecture sensitivity tests confirm that four hidden layers with 50 neurons per layer represent a sound trade-off between model capacity and training cost for this class of problems. Five limitations have been identified and discussed: training overhead, hyperparameter sensitivity, restriction to one dimension, absence of kinetic effects, and lack of experimental validation. Each limitation is addressed through a concrete research roadmap. Physics-Informed Neural Networks offer real advantages as mesh-free, differentiable plasma surrogates, particularly in parametric studies, real-time surrogate applications, and scenarios where sparse experimental measurements need to be assimilated directly into the physical model.

References

[1] J. Wesson, Tokamaks, 4th ed. Oxford: Oxford Univ. Press, 2011.

[2] M. A. Lieberman and A. J. Lichtenberg, Principles of Plasma Discharges and Materials Processing, 2nd ed. Hoboken: Wiley, 2005.

[3] F. F. Chen, Introduction to Plasma Physics and Controlled Fusion, vol. 1, 3rd ed. Cham: Springer, 2016.

[4] R. J. Goldston and P. H. Rutherford, Introduction to Plasma Physics. Bristol: IOP Publishing, 1995.

[5] P. Helander and D. J. Sigmar, Collisional Transport in Magnetized Plasmas. Cambridge: Cambridge Univ. Press, 2002.

[6] B. D. Dudson et al., "BOUT++: A framework for parallel plasma fluid simulations," Comput. Phys. Commun., vol. 180, no. 9, pp. 1467-1480, 2009.

[7] R. J. LeVeque, Finite Difference Methods for Ordinary and Partial Differential Equations. Philadelphia: SIAM, 2007.

[8] Y. Saad, Iterative Methods for Sparse Linear Systems, 2nd ed. Philadelphia: SIAM, 2003.

[9] W. Bangerth and R. Rannacher, Adaptive Finite Element Methods for Differential Equations. Basel: Birkhauser, 2003.

[10] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," J. Comput. Phys., vol. 378, pp. 686-707, Feb. 2019.

[11] G. E. Karniadakis et al., "Physics-informed machine learning," Nature Rev. Phys., vol. 3, no. 6, pp. 422-440, Jun. 2021.
[12] E. Haghighat and R. Juanes, "SciANN: A Keras/TensorFlow wrapper for scientific computations and physics-informed deep learning," Comput. Methods Appl. Mech. Eng., vol. 373, p. 113552, Jan. 2021.
[13] A. D. Jagtap, E. Kharazmi, and G. E. Karniadakis, "Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems," Comput. Methods Appl. Mech. Eng., vol. 365, p. 113028, Jun. 2020.
[14] S. Markidis, "The old and the new: Can physics-informed deep learning replace traditional linear solvers?" Front. Big Data, vol. 4, p. 669097, Jul. 2021.
[15] J. Blechschmidt and O. G. Ernst, "Three ways to solve partial differential equations with neural networks: a review," GAMM-Mitt., vol. 44, no. 2, p. e202100006, Jun. 2021.
[16] V. E. Golant, A. P. Zhilinsky, and I. E. Sakharov, Fundamentals of Plasma Physics. New York: Wiley, 1980.
[17] D. Tskhakaya, K. Matyash, R. Schneider, and F. Taccogna, "The particle-in-cell method," Contrib. Plasma Phys., vol. 47, no. 8-9, pp. 563-594, Oct. 2007.
[18] K. U. Riemann, "The Bohm criterion and sheath formation," J. Phys. D: Appl. Phys., vol. 24, no. 4, pp. 493-518, Apr. 1991.
[19] J. D. Logan, Applied Mathematics, 4th ed. Hoboken: Wiley, 2013.
[20] S. Wang, Y. Teng, and P. Perdikaris, "Understanding and mitigating gradient flow pathologies in physics-informed neural networks," SIAM J. Sci. Comput., vol. 43, no. 5, pp. A3055-A3081, Oct. 2021.
[21] L. Lu, X. Meng, Z. Mao, and G. E. Karniadakis, "DeepXDE: A deep learning library for solving differential equations," SIAM Rev., vol. 63, no. 1, pp. 208-228, Feb. 2021.
[22] C. L. Wight and J. Zhao, "Solving Allen-Cahn and Cahn-Hilliard equations using the adaptive physics-informed neural networks," Commun. Comput. Phys., vol. 29, no. 3, pp. 930-954, Jan. 2021.
[23] A. Krishnapriyan, A. Gholami, S. Zhe, R. Kirby, and M. W. Mahoney, "Characterizing possible failure modes in physics-informed neural networks," in Advances in Neural Information Processing Systems (NeurIPS), vol. 34, pp. 26548-26560, 2021.
[24] W. Wu, M. Daneker, M. A. Jolley, K. T. Turner, and L. Lu, "Effective data sampling strategies and boundary condition constraints of physics-informed neural networks," Appl. Math. Mech., vol. 44, pp. 1039-1068, Jul. 2023.
[25] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in Proc. 13th Int. Conf. Artif. Intell. Stat. (AISTATS), Sardinia, Italy, 2010, pp. 249-256.
[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in Proc. 3rd Int. Conf. Learn. Represent. (ICLR), San Diego, CA, 2015.
[27] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," SIAM J. Sci. Comput., vol. 16, no. 5, pp. 1190-1208, Sep. 1995.
[28] A. D. Jagtap and G. E. Karniadakis, "Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations," Commun. Comput. Phys., vol. 28, no. 5, pp. 2002-2041, Nov. 2020.
[29] M. Abadi et al., "TensorFlow: A system for large-scale machine learning," in Proc. 12th USENIX Conf. Oper. Syst. Des. Implement. (OSDI), Savannah, GA, 2016, pp. 265-283.
[30] J. C. Strikwerda, Finite Difference Schemes and Partial Differential Equations, 2nd ed. Philadelphia: SIAM, 2004.

[31] P. C. Stangeby, The Plasma Boundary of Magnetic Fusion Devices. Bristol: IOP Publishing, 2000.

[32] C. K. Birdsall and A. B. Langdon, Plasma Physics via Computer Simulation. Bristol: IOP Publishing, 2004.

[33] X. Meng and G. E. Karniadakis, "A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems," J. Comput. Phys., vol. 401, p. 109020, Jan. 2020.

[34] D. Mcclenny and U. Braga-Neto, "Self-adaptive physics-informed neural networks using a soft attention mechanism," in Proc. AAAI 2023 Spring Symp. on Physics-Informed Machine Learning, 2023.

[35] N. Rahaman et al., "On the spectral bias of neural networks," in Proc. 36th Int. Conf. Mach. Learn. (ICML), Long Beach, CA, 2019, pp. 5301-5310.

[36] W. Filbet and T. Rey, "A hierarchy of hybrid numerical methods for multiscale kinetic equations," SIAM J. Sci. Comput., vol. 37, no. 3, pp. A1218-A1247, Jun. 2015.

[37] J. A. Bittencourt, Fundamentals of Plasma Physics, 3rd ed. New York: Springer, 2004.

[38] M. Tancik et al., "Fourier features let networks learn high-frequency functions in low-dimensional domains," in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, pp. 7537-7547, 2020.

[39] Z. Mao, A. D. Jagtap, and G. E. Karniadakis, "Physics-informed neural networks for high-speed flows," Comput. Methods Appl. Mech. Eng., vol. 360, p. 112789, Mar. 2020.

[40] L. Yang, X. Meng, and G. E. Karniadakis, "B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data," J. Comput. Phys., vol. 425, p. 109913, Jan. 2021.

[41] E. Lapenta and L. Brackbill, "Control of the number of particles in fluid and MHD particle-in-cell methods," Comput. Phys. Commun., vol. 87, no. 1-2, pp. 139-154, May 1995.

[42] X. Jin, S. Cai, H. Li, and G. E. Karniadakis, "NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations," J. Comput. Phys., vol. 426, p. 109951, Feb. 2021.

[43] S. Cuomo et al., "Scientific machine learning through physics-informed neural networks: Where we are and what's next," J. Sci. Comput., vol. 92, no. 3, p. 88, Aug. 2022.

[44] A. Mesbah and D. B. Go, "Physics-based machine learning for plasma modeling and control," Curr. Opin. Chem. Eng., vol. 34, p. 100726, Dec. 2021.

[45] A. D. Jagtap, K. Kawaguchi, and G. E. Karniadakis, "Locally adaptive activation functions with slope recovery for deep and physics-informed neural networks," Proc. R. Soc. A, vol. 476, no. 2239, p. 20200334, Jul. 2020.

[46] I. E. Lagaris, A. Likas, and D. I. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," IEEE Trans. Neural Netw., vol. 9, no. 5, pp. 987-1000, Sep. 1998.

[47] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: a survey," J. Mach. Learn. Res., vol. 18, no. 1, pp. 5595-5637, Jan. 2018.

[48] T. J. M. Boyd and J. J. Sanderson, The Physics of Plasmas. Cambridge: Cambridge Univ. Press, 2003.

[49] U. Stroth, Plasma Physics: Confinement, Transport, and Collective Effects. Berlin: Springer, 2011.

[50] N. Geneva and N. Zabaras, "Modeling the dynamics of PDE systems with physics-constrained deep autoregressive networks," J. Comput. Phys., vol. 403, p. 109056, Feb. 2020.

[51] H. Grad, "On the kinetic theory of rarefied gases," Commun. Pure Appl. Math., vol. 2, no. 4, pp. 331-407, Dec. 1949.

[52] D. E. Post and R. Behrisch, Physics of Plasma-Wall Interactions in Controlled Fusion. New York: Plenum Press, 1986.

[53] R. Hockney and J. W. Eastwood, Computer Simulation Using Particles. Bristol: IOP Publishing, 1988.

[54] E. Kharazmi, Z. Zhang, and G. E. Karniadakis, "hp-VPINNs: Variational physics-informed neural networks with domain decomposition," Comput. Methods Appl. Mech. Eng., vol. 374, p. 113547, Feb. 2021.

[55] B. Moseley, A. Markham, and T. Nissen-Meyer, "Finite basis physics-informed neural networks (FBPINNs): A scalable domain decomposition approach for solving differential equations," Adv. Comput. Math., vol. 49, no. 4, p. 62, Jul. 2023.

[56] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis, "Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators," Nature Mach. Intell., vol. 3, no. 3, pp. 218-229, Mar. 2021.

[57] Y. Shin, J. Darbon, and G. E. Karniadakis, "On the convergence of physics-informed neural networks for linear second-order elliptic and parabolic type PDEs," Commun. Comput. Phys., vol. 28, no. 5, pp. 2042-2074, Nov. 2020.

[58] S. Wang, H. Wang, and P. Perdikaris, "On the eigenvector bias of Fourier feature networks: From regression to solving multi-scale PDEs with physics-informed neural networks," Comput. Methods Appl. Mech. Eng., vol. 384, p. 113938, Oct. 2021.

[59] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," in Proc. 9th Int. Conf. Learn. Represent. (ICLR), 2021.

[60] W. Ji, W. Qiu, Z. Shi, S. Pan, and S. Deng, "Stiff-PINN: Physics-informed neural network for stiff chemical kinetics," J. Phys. Chem. A, vol. 125, no. 36, pp. 8098-8106, Sep. 2021.

[61] S. Goswami, C. Anitescu, S. Chakraborty, and T. Rabczuk, "Transfer learning enhanced physics-informed neural network for phase-field modeling of fracture," Theor. Appl. Fract. Mech., vol. 106, p. 102447, Apr. 2020.

[62] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," Neural Netw., vol. 2, no. 5, pp. 359-366, 1989.

[63] C. Xu and T. A. Darve, "Physics-constrained learning for data-driven inverse modeling from sparse observations," J. Comput. Phys., vol. 453, p. 110938, Mar. 2022.

[64] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in Proc. 27th Int. Conf. Artif. Neural Netw. (ICANN), Rhodes, Greece, 2018, pp. 270-279.

[65] N. Wandel, M. Weinmann, and R. Klein, "Learning incompressible fluid dynamics from scratch: Towards fast, differentiable fluid models that generalize," in Proc. 9th Int. Conf. Learn. Represent. (ICLR), 2021.

[66] J. Yu, L. Lu, X. Meng, and G. E. Karniadakis, "Gradient-enhanced physics-informed neural networks for forward and inverse PDE problems," Comput. Methods Appl. Mech. Eng., vol. 393, p. 114823, Apr. 2022.

[67] T. Chen and H. Chen, "Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems," IEEE Trans. Neural Netw., vol. 6, no. 4, pp. 911-917, Jul. 1995.

[68] C. Moya and G. Lin, "Fed-PINN: Physics-informed neural networks for distributed learning on heterogeneous data," Sci. Rep., vol. 13, p. 7982, May 2023.

[69] E. van der Giessen et al., "Roadmap on multiscale materials modeling," Modell. Simul. Mater. Sci. Eng., vol. 28, no. 4, p. 043001, May 2020.

[70] A. Quarteroni, R. Sacco, and F. Saleri, Numerical Mathematics, 2nd ed. Berlin: Springer, 2007.

[71] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in Advances in Neural Information Processing Systems (NeurIPS), Vancouver, BC, 2007, pp. 1177-1184.

[72] N. Brambilla, "Fundamentals of plasma heating and current drive," Plasma Phys. Control. Fusion, vol. 41, no. 1, pp. 1-24, Jan. 1999.

[73] D. P. McQuarrie, Mathematical Methods for Scientists and Engineers. Sausalito: Univ. Science Books, 2003.

[74] P. C. Bressloff and J. M. Newby, "Stochastic models of intracellular transport," Rev. Mod. Phys., vol. 85, no. 1, p. 135, Jan. 2013.

[75] H. Nguyen-Xuan et al., "A physics-informed neural network approach for strongly nonlinear boundary value problems with application to plasma discharge modeling," Comput. Phys. Commun., vol. 277, p. 108373, Aug. 2022.