# On the Importance of the Mathematical Formulation to Get PINNs Working

Brahim El Mokhtari, Cédric Chauviere, Pierre Bonnet

# On The Importance of The Mathematical Formulation to Get PINNs Working

Brahim EL MOKHTARI, Cédric CHAUVIERE, Pierre BONNET

***Abstract*—Physics-informed neural networks (PINNs) are a powerful approach that combines deep learning with physical principles to solve complex problems. However, like any method, they do have some drawbacks. The first one is hyperparameter sensitivity such as learning rates, network architectures and activation functions. Many researchers have devoted their time and energy to design efficient neural network models by searching optimal hyperparameters. In this article, we follow another path by showing that the mathematical formulation of the problem to be solved, has a critical influence on the performance of the model. Electrostatic examples illustrate this.**

***Index Terms*—PINN, Poisson Equation, meshless method.**

## I. Introduction

SINCE the seminal work of Raissi et al. in 2017 [1], [2], later published in a merged version [3], PINNs have attracted the attention of an increasing number of researchers over the last few years. Yet, there are still many obstacles that need to be addressed before PINNs get competitive in comparison with well established numerical methods such as finite difference or finite elements methods, for example. Numerical stability is probably the main issue. Indeed, PINNs involve solving linear and nonlinear partial differential equations (PDEs) numerically, which can lead to stability issues. In the context of machine learning models, stability issues usually means that the learning process does not proceed as expected. In this case, the optimisation procedure required to compute the parameters of the model converges to a local minima that lead to a wrong solution [4]. Careful consideration of numerical methods is essential for accurate and reliable results when PINNs are used [5], [6], since they may lead to solutions that look good but are still highly inaccurate. This is a major difference compared with standard numerical methods that tend to diverge roughly when things go wrong. Training PINNs can also be computationally expensive, especially for large-scale problems or complex physics simulations and this can limit their applicability in real-time or resource-constrained environments. Despite these drawbacks, the use of PINNs continues to advance, and researchers are actively working on addressing these challenges to enhance their practicality and effectiveness. In the jungle of newly proposed PINNs methods to treat these issues, Lawal et al. [7] conveniently classify them into three families: extended PINNs [8]–[10], hybrid PINNs [11], [12] and minimized loss techniques [13], [14].

Brahim El Mokhtari and Pierre Bonnet are with the Université Clermont Auvergne, Clermont Auvergne INP, CNRS, Institut Pascal, F-63000 Clermont-Ferrand, France (e-mail: Brahim.el_mokhtari@uca.fr; pierre.bonnet@uca.fr).

Cédric Chauvière is with the Université Clermont Auvergne, Clermont Auvergne INP, CNRS, Laboratoire de Mathématiques, F-63000 Clermont-Ferrand, France (e-mail: cedric.chauviere@uca.fr).

In spite of the drawbacks cited above, PINNs come with several advantages that make them a valuable tool in various scientific and engineering applications. The possibility to embed prior knowledge is one of the main benefits of PINNs. Indeed, they can seamlessly integrate existing physical knowledge (i.e. experimental data points) and partial differential equations into a single learning process. Since neural networks have the capacity to model highly complex and nonlinear relationships, this flexibility enables PINNs to capture intricate patterns and behaviors in physical systems that may be challenging for traditional methods. This is why they are well-suited for handling multi scale problems [15] where phenomena occur at different spatial or temporal scales. For the same reason, they have the potential to automatically discover hidden patterns or relationships within the data. This can lead to new insights and discoveries in scientific research, particularly in cases where the underlying physics may not be fully understood [16]. Furthermore, when the learning procedure is performed with all the proper rules (i.e. with training, validation and test data), PINNs can exhibit resilience to noisy or incomplete data. The incorporation of physics-based constraints [17] helps the model generalize well, even in the presence of uncertainties or measurement errors. Since most of the computations required by the training procedure consist in matrix-matrix products, training neural networks can make effective use of GPU cards and be parallelized efficiently, making it feasible to handle large-scale problems. This scalability is advantageous for applications in various fields such as antennas and propagation [18], fluid dynamics [19], structural mechanics [20] or materials science [21], to quote a few. Last but not least, as PINNs is a meshless methods, the possibility to get a solution without having to mesh the computational domain is a great advantage over traditional numerical techniques. This also comes with a disarmingly simple way to get a solution at points that do not belong to the training points, without the recourse to complex interpolation. Overall, the advantages of PINNs make them a valuable tool for solving challenging problems in science and engineering, providing a bridge between data-driven approaches and physical principles.

Therefore, it is not surprising that in computational electromagnetic, PINNs have been utilized in several ways. It has been employed both in the frequency domain [22], [23] and in the time domain [18], [24] to deal with complex electromagnetic phenomena. These phenomena involve intricate interactions between electromagnetic fields and various components, and PINNs can capture these interactions effectively. PINNs can also predict the electromagnetic performance of systems [25] by simulating their behavior under different operating conditions. The same idea applied to electromagnetic

compatibility (EMC) could help engineers identify potential problems early in the design phase and take necessary measures to mitigate them. Not only PINNs can be used for direct problems, but also in electromagnetic inverse problems by training a neural network to infer material properties [23] or object properties from measured electromagnetic field data for example [26]. Despite this tremendous promise of PINNs application, one should not think that the technology is yet advanced enough to solve any type of problem. This is very well explained in the context of fluid simulation in [27], but generalizes to many other fields, including electromagnetism. Although many articles describe solving Poisson or Laplace equations, the success of their method is mainly due to the fact that their problem is formulated in a very specific way [28]–[31]. In all articles cited, the charges inside the domain are not treated under the form of interior boundaries conditions, as is often done in finite difference methods. More specifically, the only boundaries conditions are the outer ones of the computational domain. We found that in order to get PINNs working for the Poisson or Laplace equation, it was necessary to have no inside boundaries conditions. Note that numerical resolution of the Poisson (or Laplace) equation is often an important first step in solving cable EMC problems by calculating their Per-Unit-Length parameters [32], [33].

In this article, we strive to compute, by mean of standard PINNs, the electrostatic potential in a given computational domain regardless the boundary condition and the geometry. We show that the formulation used to model the problem has a dramatic influence on the solution provided by a trained PINN. For each formulation a centred finite difference solution was computed as reference solution to compare with the PINNs solution using the same formulations.

The paper is structured as follows. In the first section, we introduce three possible mathematical formulations of the electrostatic potential model in a general computational domain. In the second section, our PINN solver is introduced to serve as benchmark solver for our problem. Then, we apply it to each formulation and we show that only one of them lead to the right solution. In section V, using the right formulation, we solve a complex problem that would be challenging to tackle with standard method due to the intricate shape of the charge distribution. Finally, in section VI, we draw some conclusions.

## II. Possible formulations of the problem

To make matters simple, assume that we want to compute the electrostatic potential $V$ generated by any distribution of charges into a two-dimension domain denoted by $\Omega = [0,1] \times [0,1]$ (see Fig. 1) with the condition $V = 0$ at infinity (i.e. outside of $\Omega$). The procedure described above can easily generalize to more complex problems. Let $\Gamma_1$ and $\Gamma_2$ denote two rectangular area supporting surface charge distribution $\rho_s$. In the case where $\Gamma_1$ (respectively $\Gamma_2$) is a conductive plate, this distribution reduces to linear charge distribution $\rho_l$ on $\partial\Gamma_1$ (respectively $\partial\Gamma_2$).

Two equivalent mathematical formulations can be used to calculate the electric potential $V$ over $\Omega$. The first one imposes a constant electric potential $V_0$ over $\Gamma_1$ and $-V_0$ over $\Gamma_2$ :

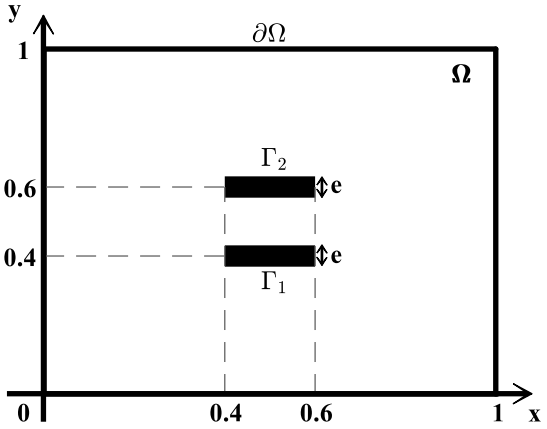


Fig. 1. Representation of the studied problem

$$\begin{cases} \Delta V(x,y) = 0, & (x,y) \in \Omega \\ V(x,y) = 0, & (x,y) \in \partial\Omega \\ V(x,y) = V_0, & (x,y) \in \Gamma_1 \\ V(x,y) = -V_0, & (x,y) \in \Gamma_2 \end{cases} \tag{1}$$

In this configuration, $\Gamma_1$ and $\Gamma_2$ are supposed to be perfectly electric conductors since $V$ is equipotential on the two rectangular area. Obviously, $\Gamma_1$ and $\Gamma_2$ define a planar capacitor in this case. Since there always exist a linear electric charge density $\rho_l(x,y)$ for $(x,y) \in \partial\Gamma_1 \cup \partial\Gamma_2$ which induces an equivalent electric potential, the problem can be reformulated as :

$$\begin{cases} \Delta V(x,y) = -\dfrac{\rho_l(x,y)}{\epsilon_0} \mathbb{1}_{\partial\Gamma_1 \cup \partial\Gamma_2}(x,y), (x,y) \in \Omega \\ V(x,y) = 0, & (x,y) \in \partial\Omega \end{cases} \tag{2}$$

where $\epsilon_0$ is the electric permittivity in vacuum.

Although traditional numerical method, such as finite difference, have no difficulties solving this problem using either formulation, PINNs fail to train properly. Such behaviour is well documented in the literature and maybe sometimes cured using adaptive loss balance, extensive hyper-parameter tuning or experimentation with different neural network architectures. It turned out that none of these patches lead to a correct solution (see Fig. 2 and Fig. 5 for typical solutions of a PINN that failed to train using either formulation and Fig. 4 for a correct solution).

On the other hand, the following mathematical formulation below, was not plagued with PINN failure mode, even for very simple structures of neural networks:

$$\begin{cases} \Delta V(x,y) = -\dfrac{\rho_s(x,y)}{\epsilon_0} \mathbb{1}_{\Gamma_1 \cup \Gamma_2}(x,y), (x,y) \in \Omega \\ V(x,y) = 0, & (x,y) \in \partial\Omega \end{cases} \tag{3}$$

Note that this formulation is very similar to the previous one, with the exception of the rectangular plates, where the electric charges distributions $\rho_s$ are now set over surfaces $\Gamma_1 \cup \Gamma_2$ instead of lines $\partial\Gamma_1 \cup \partial\Gamma_2$. Here we no longer consider perfectly electric conductors. Contrary to formulation (1), formulation (3) has a great advantage of not requiring imposing boundaries conditions at the plates, alleviating the

need to treat loss balancing. This advantage is even more glaring when there are numerous electrically charged surfaces in the computational domain. If one want to recover the solution to equivalent problem (1) or (2), then two neural networks may be built using formulation (3) : one with slightly deflated plates capacitors $(\Gamma_1 \backslash \delta\Gamma_1) \cup (\Gamma_2 \backslash \delta\Gamma_2)$, giving the neural network model $NN1(x, y)$; and one with the original plates capacitor $\Gamma_1 \cup \Gamma_2$, giving the neural network model $NN2(x, y)$. Since we consider linear problems, according to the principle of superposition, one can recover the solution sought as $NN(x, y) = NN2(x, y) - NN1(x, y)$.

In the section IV, we'll present numerical results with PINNs using the formulation (3).

## III. The PINN solver

Although generating points for PINNs usually involves a combination of known data points (e.g., from experimental measurements or simulations) and randomly sampled points within the domain of interest, here we only use the latest. Accordingly, the data-driven loss that ensures that the neural network solution matches the known data points and penalizes the difference between the predicted solution at specific points and the corresponding measured values is not taken into account. Here, only the Physics-Informed loss that enforces the neural network solution to satisfy the governing equations of the problem is used. It is formulated by taking the differential equations (e.g., Poisson/Laplace equations) and substituting the neural network solution and its derivatives into them. The residuals of these equations represent the discrepancy between the predicted solution and the true solution. Minimizing these residuals ensures that the neural network solution satisfies the physics of the problem. Here, two type of points are generated: points along the boundaries of the domain to enforce boundary conditions and points within the interior of the domain to capture the behavior of the system away from the boundaries. This is a great advantage of PINNs solver as they belong to meshless methods.

Let denote by $NN_\theta(x, y) \approx V(x, y)$ a neural network parameterised by $\theta$ that approximates problem (3). The parameters of the model are obtained by minimization of the loss function $\mathcal{L}_{PINN}$, i.e.

$$\theta^* = \arg\min_\theta(\mathcal{L}_{PINN}), \tag{4}$$

The PINN loss consists of two parts: $\mathcal{L}_{BC}$ for the boundaries conditions on $\partial\Omega$ and $\mathcal{L}_{PDE}$ for the PDE. They are balanced by a parameter $\alpha \in \mathbb{R}^{+*}$ according to

$$\mathcal{L}_{PINN} = \mathcal{L}_{BC} + \alpha\mathcal{L}_{PDE}, \tag{5}$$

where

$$\mathcal{L}_{PDE} = \frac{1}{N_{PDE}} \sum_i \left( \nabla NN_\theta(x_i, y_i) + \frac{\rho(x_i, y_i)}{\epsilon_0} \right)^2, \tag{6}$$

for $(x_i, y_i) \in \Omega$; and

$$\mathcal{L}_{BC} = \frac{1}{N_{BC}} \sum_i \left( NN_\theta(x_i, y_i) \right)^2, \tag{7}$$

for $(x_i, y_i) \in \partial\Omega$.

Note that the standard formulations (1) would require additional loss terms for the boundary conditions on $\partial\Gamma_1$ and $\partial\Gamma_2$. As a consequence, the balance of the different terms accounting to the total loss, would also be trickier.

## IV. Numerical results using PINNs

### A. Building a reference solution.

Since there is no exact solution to the proposed problem we need to build a reference solution. This solution is computed with a centred finite difference method using a fine mesh and employing the same mathematical formulation as for PINNs. To ensure that it can be used as a reference solution, we conduct a convergence study for all three formulations. Four distinct uniform grids on $\Omega = [0, 1] \times [0, 1]$ with $100 \times 100$, $200 \times 200$, $400 \times 400$, and $800 \times 800$ points are used to compute solutions. For all of these grids, let $\tilde{V}^{(k)}$ $k \in \{1, 2, 3, 4\}$ be the discrete respective solutions. The mean square error (MSE) between $\tilde{V}^{(k)}$ $k \in \{1, 2, 3\}$ and $\tilde{V}^{(4)}$ is computed in order to conduct the convergence analysis. Results are shown in Table I for all three formulations. We see that the MSE decreases with mesh refinement up to $10^{-6}$. Therefore the solution on the $800 \times 800$ mesh can serve as reference solution to asses the accuracy of the PINNs result.

TABLE I
CONVERGENCE STUDY OF THE FINITE DIFFERENCE SOLUTIONS

| | $100 \times 100$ vs $800 \times 800$ | $200 \times 200$ vs $800 \times 800$ | $400 \times 400$ vs $800 \times 800$ |
|---|---|---|---|
| MSE formulation (1) | $3.908 \times 10^{-4}$ | $7.482 \times 10^{-5}$ | $8.531 \times 10^{-6}$ |
| MSE formulation (2) | $3.435 \times 10^{-5}$ | $6.293 \times 10^{-6}$ | $7.004 \times 10^{-7}$ |
| MSE formulation (3) | $1.199 \times 10^{-4}$ | $2.211 \times 10^{-5}$ | $2.465 \times 10^{-6}$ |

### B. Using formulation (1) or (2)

Whereas extensive trials with different neural network architecture, hyper-parameters search, regularization techniques and loss balance were carried out, to the best of our knowledge, none of them led to an acceptable neural network model when the formulation (1) or (2) were used. Typical failed solution with equations (1) is shown in Fig. 2 for $V_0 = +1V$.

In Fig. 3, we see that the loss drops quickly to about $0.3$ and stays around this value, even when the number of epochs is set at a large value.

When formulation (2) is used, although the solution looks plausible (see Fig. 5), the maximum point wise absolute error reaches unacceptably high values. This can be seen in Fig. 6, where the color bar indicates errors up to $0.63$ around the plates.

A closer look on the absolute error between the failed PINN solution and a finite difference solutions (see Fig. 6 and Fig. 4) reveals that the major part of the error is located around the plates. However, we found that stacking more points around the plates did not cure the problem. This observation led us to treat interior boundary conditions in a different way, as described by equations (3).

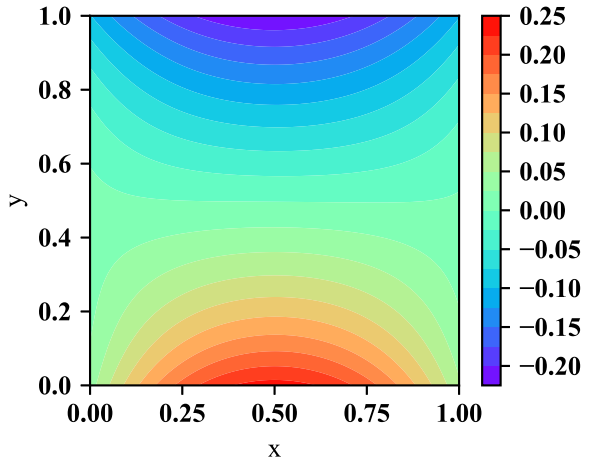


Fig. 2. Plot of the wrong electric potential obtained with PINN and formulation (1).

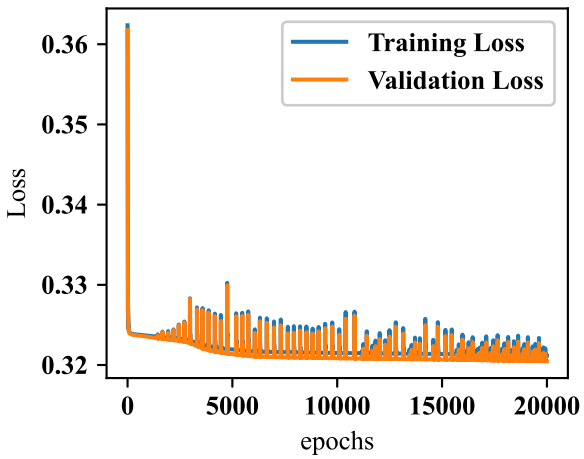


Fig. 3. Plot of the training loss and validation loss with formulation (1).

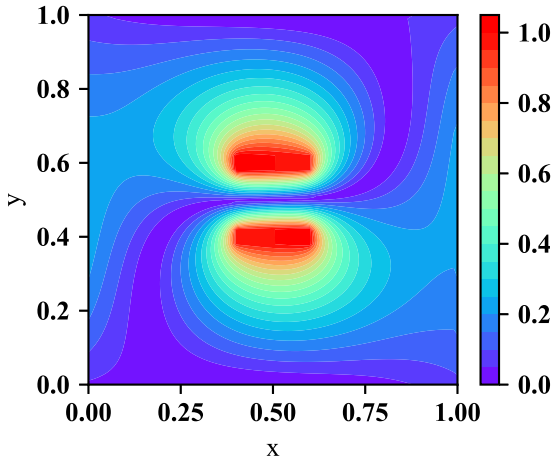


Fig. 4. Point wise absolute error between PINN and finite difference result obtained with formulation (1).

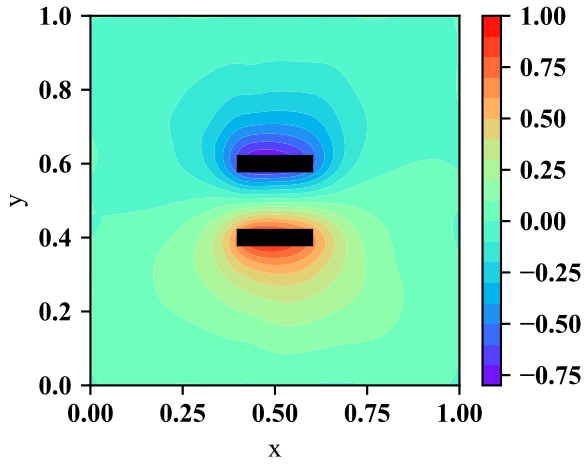


Fig. 5. Electric potential using the PINN with formulation (2).

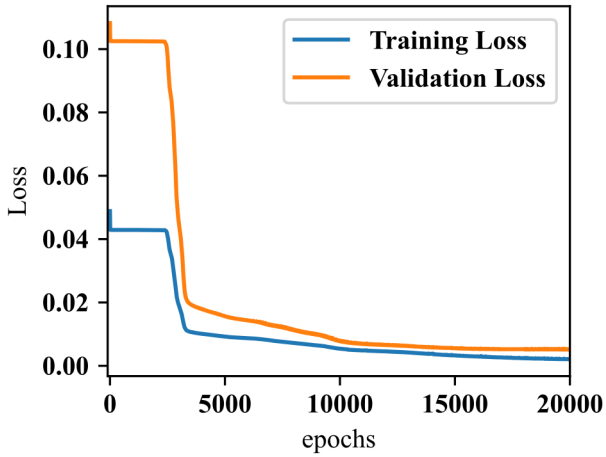


Fig. 6. Point wise absolute error of the electric field between PINN result and the finite difference result with formulation (2).

### C. *Using formulation* (3)

When the formulation (3) was used, even for an extremely simple neural network with three dense layers and 32 neurons in each layer, the learning process went smoothly and led to a good PINN solution. In this case, the training loss goes down to $1.5 \times 10^{-3}$ ($5 \times 10^{-3}$ for the validation loss), see Fig. 7. Here, at the beginning of the training, we notice a plateau phenomenon which refers to a situation where the loss function shows very little change. This can happen when the optimizer encounters a flat region (a "plateau") in the loss landscape, making it difficult for the model to find directions for improvement. Hopefully, in our case, the optimizer which is based on the stochastic gradient descent escape from the plateau.

The model accounted for a total of 2241 parameters and the hyperbolic tangent was used as activation function in each layers, excepted for the last one that was left as a linear layer. The learning rate was kept constant during the learning process and taken equal to $10^{-3}$. The parameter $\alpha$ to balance the two parts of the loss is set such that the boundary loss is weighted

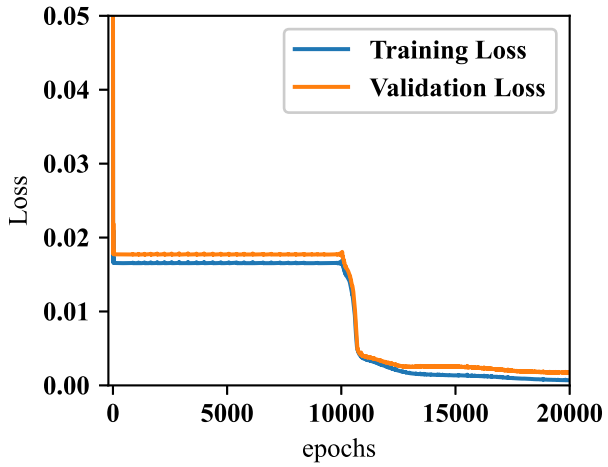


Fig. 7. Plot of the training loss and validation loss with formulation (3).

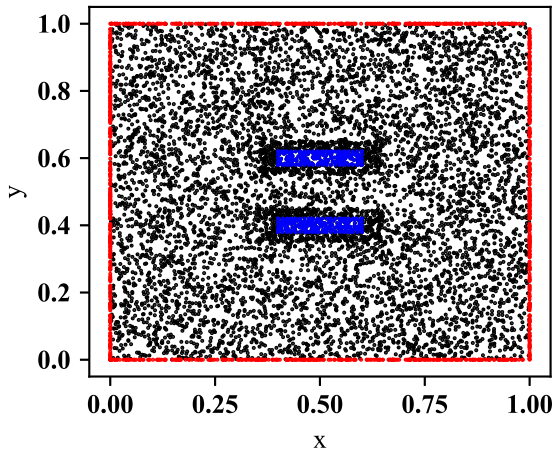


Fig. 8. PINNs training collocation point randomly drawn into the domain.

500 times larger than the PDE loss. The parameters of the neural network are computed using the Adam [34] optimiser.

The numerical experiments are carried out with three different meshes built by generating random points on $\Omega = [0,1] \times [0,1]$. The points are generated with the uniform probability low on $[0,1] \times [0,1]$ and extra point are also randomly stacked around the plates to ensure a better resolution in that part of the domain. A total of 10000, 40000 and 160000 points serve to build the three PINNs models. Amongst them, 400 points (respectively 1600 and 6400) are specifically set on $\partial\Omega$ (red points) and 100 are set on $\partial\Gamma_1 \cup \partial\Gamma_2$ (contour of the blue points), see Fig. 8. In all cases, 80% of points are training points and the remaining 20% are validation points. Results for the point wise absolute error are shown on Fig. 11-13. We see that the maximum point wise absolute error decreases slowly as the mesh is refined (from 0.063 for 10000 points mesh to 0.036 for 160000 points mesh). The mean square error between the finite difference solution (which can be considered as an exact solution) and the PINN solution is equal to $6.43 \times 10^{-4}$ for the 10000 points mesh, $2.16 \times 10^{-4}$ for the 40000 point mesh and $2.21 \times 10^{-4}$ for the 160000 points mesh. Those results show that increasing the number of training point from 40000 to 160000 points does not bring any accuracy improvement when the mean square error is considered.

If we look at the pointwise error between the formulation (1)-(2) and the formulation (3) , by comparing the Fig. 4 and Fig. 6 with the Fig. 11-13 that the maximum error is no longer located around the plates, and it has been reduced by a factor of around 10.

Having successfully tested this formulation on a simple example, we investigate its potential on a more complex problem in the next section.

## V. Formulation (3) applied to a complex problem

We now extend the use of the PINN method described by formulation (3) to solve the Poisson equation with a complex charge distribution represented in Fig. 14. The charge distribution was built such that it can generate an electric

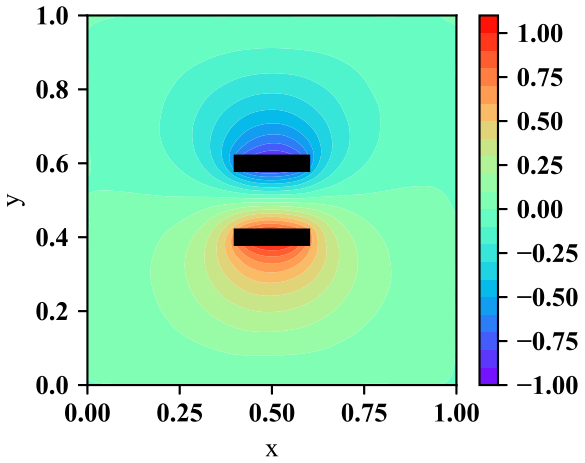


Fig. 9. Plot of the electric potential obtained with PINN and formulation (3).

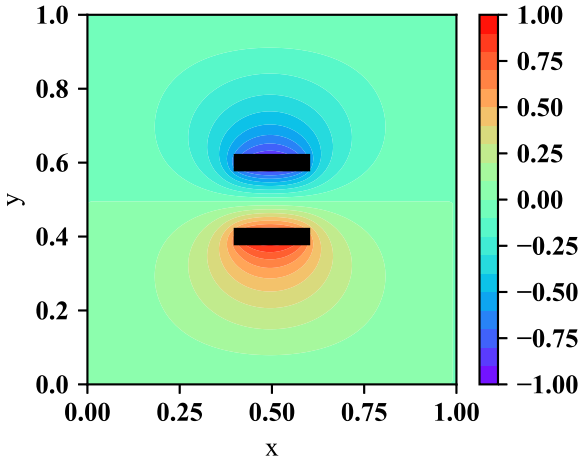


Fig. 10. Plot of the electric potential obtained with the finite difference method.

potential representing the ”EMC” letters (see Fig. 15). This

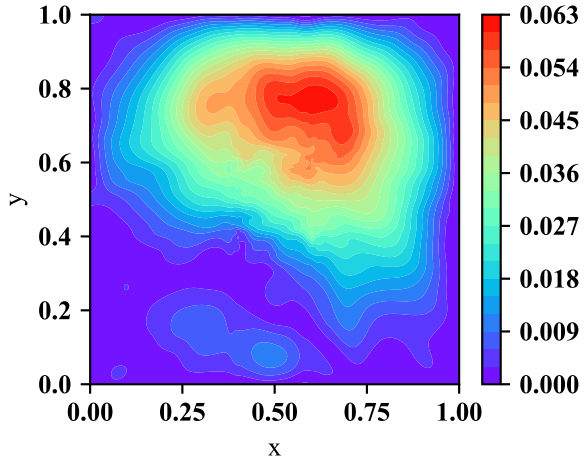


Fig. 11. Pointwise absolute error of the electric field (PINN model trained with 10000 points).

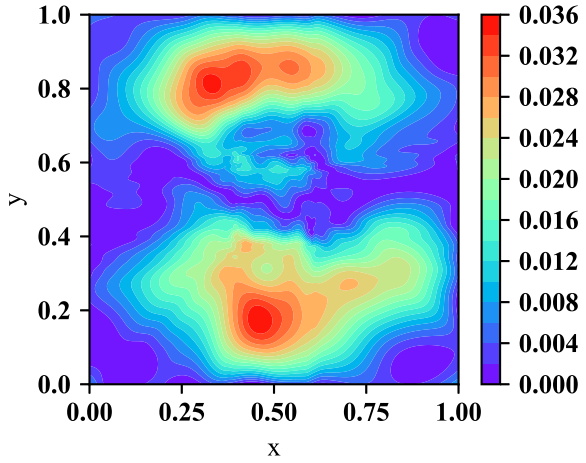


Fig. 13. Pointwise absolute error of the electric field (PINN model trained with 160000 points).

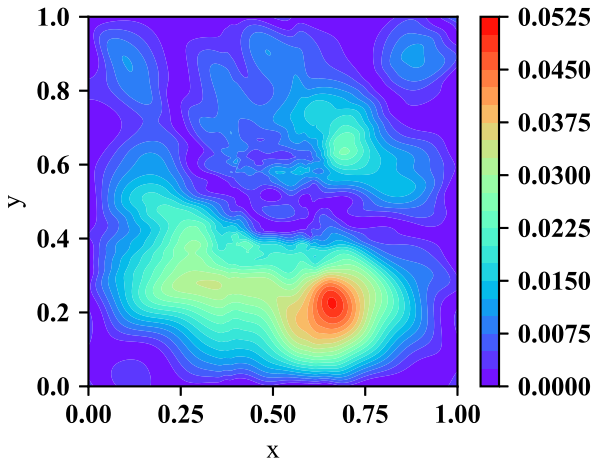


Fig. 12. Pointwise absolute error of the electric field (PINN model trained with 40000 points).

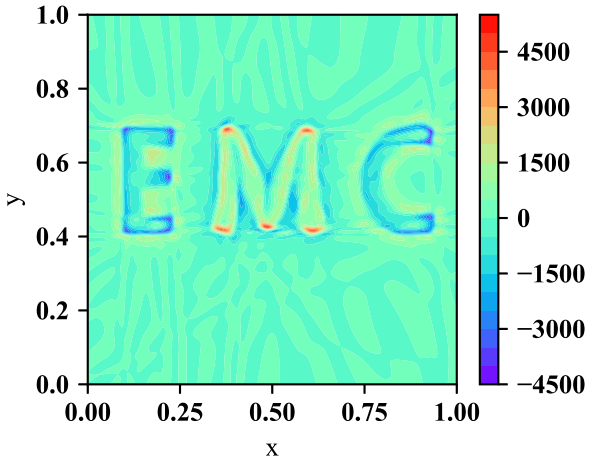


Fig. 14. Electric charge distribution normalised by vacuum electrical permittivity $\epsilon_0$.

charge distribution was generated on a 500x500 uniform grid. It is this grid that we'll use for both the reference solution (using the finite difference method) and the PINN grid. This problem is particularly interesting, since the magnitude of the charge distribution over the domain is now quite large compared to the previous example (see values of the color bar in Fig. 14).

Comparing the PINN results of Fig. 16 with the finite difference reference solution of Fig. 17, we clearly see that the learning process does not fall into a failure mode. To be more precise, the pointwise absolute error does not exceed $1.2 \times 10^{-1}$ over the entire computational domain $\Omega$, as can be seen in Fig. 17. It should be noted that, as for the previous example, we simply use a basic neural network made of dense layers. We have not carried out extensive search for optimized hyperparameters since the purpose of the article was simply to assess the importance of the mathematical formulation to ensure that the model does not converge to a wrong solution.

Two points should be emphasized for the results presented in

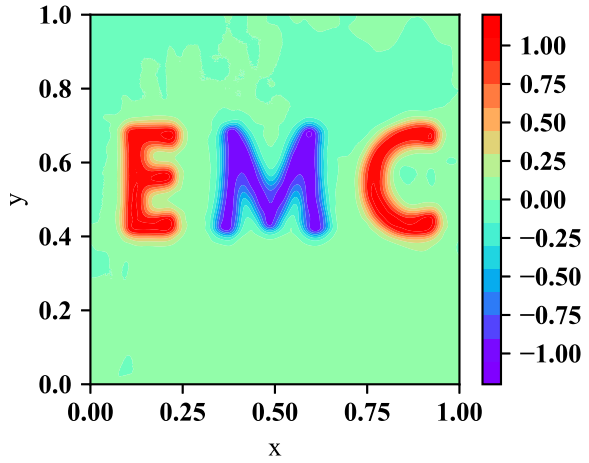


Fig. 15. Electric potential using PINN for the charge distribution of Fig. 14.

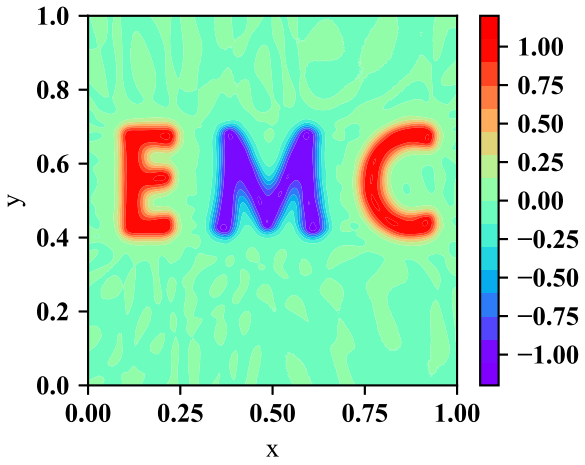


Fig. 16. Electric potential using the finite difference method for the charge distribution of Fig. 14.

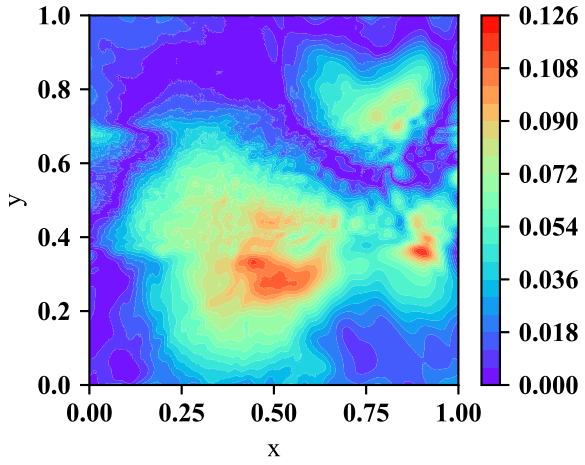


Fig. 17. Pointwise absolute error of the electric field between PINN result and the finite difference result.

this section. First, the solution provided by the finite-difference method is plagued with some errors, due to the intrinsic approximations of the scheme, particularly the staircasing discretization of the structure's geometry. Secondly, this last point could be alleviated by refining the mesh, but at the cost of a significant computational overhead. The meshless nature of PINN avoids this drawback. This property, combined with the ability of neural networks to handle nonlinear problems, will undoubtedly be key factors in the success of PINNs applied to the numerical simulation of large-scale high-complexity nonlinear EMC problems [35], [36].

## VI. Conclusion

PINNs are a still a new paradigm to solve partial differential equations and, sometimes, they fail to converge to the right solution. When this occurs, one is usually tempted to cure the problem by optimizing the hyperparameters of the neural network, or by using more complex layers such as convolutional layers or long short-term memory layers. However, this does not always successfully treats the problem.

In this article we endeavored to demonstrate the importance of the mathematical formulation to get PINNs working. This is illustrated on Poisson's equation by first showing that a formulation which does not pose a problem with the finite difference method gets PINNs reluctant to converge to the right solution. We then propose an alternative formulation for which PINNs converge without having to resort to complex neural networks or advanced hyperparameters optimization. Numerical examples show that the proposed formulation is also able to solve problems with complex charge distribution. Taking advantage of the proposed stable formulation, next step will be dedicated to improve the accuracy of PINNs for EMC applications.

## References

[1] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics informed deep learning (part I): Data-driven solutions of nonlinear partial differential equations," *arXiv:1711.10561 [cs, math, stat]*, Nov. 2017. arXiv: 1711.10561.

[2] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations," *arXiv:1711.10566 [cs, math, stat]*, Nov. 2017. arXiv: 1711.10566.

[3] 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," *Journal of Computational Physics*, vol. 378, Feb. 2019.

[4] 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* (M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds.), vol. 34, Curran Associates, Inc., 2021.

[5] S. Wang, X. Yu, and P. Perdikaris, "When and why PINNs fail to train: A neural tangent kernel perspective," *Journal of Computational Physics*, vol. 449, Jan. 2022.

[6] S. Wang, S. Sankaran, and P. Perdikaris, "Respecting causality for training physics-informed neural networks," *Computer Methods in Applied Mechanics and Engineering*, vol. 421, 2024.

[7] Z. K. Lawal, H. Yassin, D. T. C. Lai, and A. Che Idris, "Physics-informed neural network (pinn) evolution and beyond: a systematic literature review and bibliometric analysis," *Big Data and Cognitive Computing*, vol. 6, no. 4, 2022.

[8] 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," *Computer Methods in Applied Mechanics and Engineering*, vol. 365, June 2020.

[9] G. Pang, M. D'Elia, M. Parks, and G. Karniadakis, "npinns: Nonlocal physics-informed neural networks for a parametrized nonlocal universal laplacian operator. algorithms and applications," *Journal of Computational Physics*, vol. 422, 2020.

[10] M. Rafiq, G. Rafiq, and G. S. Choi, "DSFA-PINN: Deep Spectral Feature Aggregation Physics Informed Neural Network," *IEEE Access*, vol. 10, 2022.

[11] Z. Fang, "A High-Efficient Hybrid Physics-Informed Neural Networks Based on Convolutional Neural Network," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[12] W. Chen, Q. Wang, J. S. Hesthaven, and C. Zhang, "Physics-informed machine learning for reduced-order modeling of nonlinear problems," *Journal of Computational Physics*, vol. 446, 2021.

[13] A. F. Psaros, K. Kawaguchi, and G. E. Karniadakis, "Meta-learning pinn loss functions," *Journal of Computational Physics*, vol. 458, 2022.

[14] Z. Xiang, W. Peng, X. Liu, and W. Yao, "Self-adaptive loss balanced physics-informed neural networks," *Neurocomputing*, vol. 496, 2022.

[15] R. Li, E. Lee, and T. Luo, "Physics-informed neural networks for solving multiscale mode-resolved phonon boltzmann transport equation," *Materials Today Physics*, vol. 19, 2021.

[16] Z. Chen, Y. Liu, and H. Sun, "Physics-informed learning of governing equations from scarce data," *Nature Communications*, vol. 12, Oct. 2021.

[17] A. Jagtap, E. Kharazmi, and G. Karniadakis, "Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 365, 06 2020.
[18] O. Noakoasteen, S. Wang, Z. Peng, and C. Christodoulou, "Physics-informed deep neural networks for transient electromagnetic analysis," *IEEE Open Journal of Antennas and Propagation*, vol. 1, 2020.
[19] S. Cai, Z. Mao, Z. Wang, M. Yin, and G. E. Karniadakis, "Physics-informed neural networks (pinns) for fluid mechanics: A review," *Acta Mech. Sin.*, no. 37, 2021.
[20] J.-H. Bastek and D. M. Kochmann, "Physics-informed neural networks for shell structures," *European Journal of Mechanics - A/Solids*, vol. 97, 2023.
[21] E. Zhang, M. Dao, G. E. Karniadakis, and S. Suresh, "Analyses of internal structures and defects in materials using physics-informed neural networks," *Science Advances*, vol. 8, no. 7, 2022.
[22] K. Fujita, "Impedance modeling of accelerator beams with discontinuous charge density using scattered-field physics-informed neural networks," *IEICE Electronics Express*, vol. 20, no. 3, 2023.
[23] Z. Fang and J. Zhan, "Deep physical informed neural networks for metamaterial design," *IEEE Access*, vol. 8, 2020.
[24] S. Qi and C. D. Sarris, "Hybrid physics-informed neural network for the wave equation with unconditionally stable time-stepping," *IEEE Antennas and Wireless Propagation Letters*, 2024.
[25] M. Baldan, P. Di Barba, and D. A. Lowther, "Physics-informed neural networks for inverse electromagnetic problems," *IEEE Transactions on Magnetics*, vol. 59, no. 5, 2023.
[26] Y.-D. Hu, X.-H. Wang, H. Zhou, L. Wang, and B.-Z. Wang, "A more general electromagnetic inverse scattering method based on physics-informed neural network," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, 2023.
[27] L. A. Chuang, Pi-Yueh; Barba, "Experience report of physics-informed neural networks in fluid simulations: Pitfalls and frustration.," in *Proc. of the 21st SCIPY conference*, 2022.
[28] S. Markidis, "The old and the new: Can physics-informed deep-learning replace traditional linear solvers?," *Front Big Data.*, vol. 4, 2021.
[29] O. S. Hafezianzade F, Biagooi M, "Physics informed neural network for charged particles surrounded by conductive boundaries.," *Sci Rep.*, vol. 13(1), 2023.
[30] L. Cheng, E. A. Illarramendi, G. Bogopolsky, M. Bauerheim, and B. Cuenot, "Using neural networks to solve the 2d poisson equation for electric field computation in plasma fluid simulations," 2021.
[31] W. Tang, T. Shan, X. Dang, M. Li, F. Yang, S. Xu, and J. Wu, "Study on a poisson's equation solver based on deep learning technique," in *2017 IEEE Electrical Design of Advanced Packaging and Systems Symposium (EDAPS)*, 2017.
[32] C. R. Paul, *Analysis of Multiconductor Transmission Lines*. Wiley-IEEE Press, 2nd ed., 2007.
[33] C. Jullien, P. Besnier, M. Dunand, and I. Junqua, "Advanced modeling of crosstalk between an unshielded twisted pair cable and an unshielded wire above a ground plane," *IEEE Transactions on Electromagnetic Compatibility*, vol. 55, no. 1, 2013.
[34] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, (San Diega, CA, USA), 2015.
[35] C. Yang, T. Wendt, M. De Stefano, M. Kopf, C. M. Becker, S. Grivet-Talocia, and C. Schuster, "Analysis and optimization of nonlinear diode grids for shielding of enclosures with apertures," *IEEE Transactions on Electromagnetic Compatibility*, vol. 63, no. 6, 2021.
[36] T. Wendt, C. Yang, H. D. Brüns, S. Grivet-Talocia, and C. Schuster, "A macromodeling-based hybrid method for the computation of transient electromagnetic fields scattered by nonlinearly loaded metal structures," *IEEE Transactions on Electromagnetic Compatibility*, vol. 62, no. 4, 2020.

PLACE PHOTO HERE

**BRAHIM EL MOKHTARI** received a master's degree in Nuclear physics and radiation technology in 2017 from Mohammed V University, Rabat, Morocco, and first year master's degree in electromagnetic compatibility in 2022 from Clermont Auvergne University where he is currently working toward the Ph.D. degree from Institut Pascal. His current research interests include Computational physics application, artificial intelligence, time domain electromagnetic source identification.

PLACE PHOTO HERE

**CEDRIC CHAUVIERE** received a Ph.D. degree in mechanical engineering from the École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, in 2001. He conducted postdoctoral research at the Department of Applied Mathematics, Brown University, Providence, RI, USA, from 2002 to 2003. He is currently the Maître de Conférences (Associate Professor) of Mathematics with the University of Clermont Auvergne, ClermontFerrand, France. His current research interests include artificial intelligence, uncertainty quantification, electromagnetic source identification, and numerical methods for stochastic problems.

PLACE PHOTO HERE

**PIERRE BONNET** received the Diploma Engineer degree in physics from Polytech Clermont Ferrand, Clermont-Ferrand, France, in 1994, and the M.Sc. and Ph.D. degrees in electromagnetism from Blaise Pascal University, Aubière, France, in 1994 and 1998, respectively. From 1999 to 2008, he was an Assistant Professor with the Department of Physics and the Institut Pascal, Blaise Pascal University. He is currently a Full Professor with Clermont Auvergne University, Clermont-Ferrand, where he is also the Head of the Electromagnetic Compatibility Group. His research interests include electromagnetic with an emphasis on electromagnetic compatibility/electromagnetic interference problems, electromagnetic shielding, reverberating electromagnetic environment, time reversal, source identification, and stochastic modeling.