Neural Network Models as Surrogates of Classical Ordinary Differential Equations for Reduced-Order Single-Cell Electrophysiology

Yan B Werneck[1], Bernardo M Rocha[1,2], Rafael Sachetto Oliveiraa[3], Rodrigo W dos Santos[1,2],

[1]Computational Modeling Graduate Program - Federal University of Juiz de Fora, Juiz de Fora, Brazil
[2]Department of Computer Science - Federal University of Juiz de Fora, Juiz de Fora, Brazil
[3]Department of Computer Science - Federal University Federal de São João del-Rei, Juiz de Fora, Brazil

Abstract

Electrophysiology modeling is essential for non-invasive diagnostics and for advancing our understanding of heart and brain function. Traditional approaches rely on systems of ordinary differential equations (ODEs), ranging from detailed ion channel dynamics to reduced-order models. In this study, we investigate neural networks as differentiable surrogate models for electrophysiology simulations, comparing data-driven and physics-informed approaches using the FitzHugh–Nagumo model. Networks are trained with numerical solutions, incorporating system dynamics through tailored loss functions. We introduce extensive architecture optimization and adaptive training strategies to enhance performance. Inference accelerated with NVIDIA TensorRT achieves up to 1.8× speedup over optimized CUDA solvers with minimal accuracy degradation. These results demonstrate the potential of neural networks as scalable, differentiable emulators for reduced-order electrophysiology models, enabling their integration into real-time, AI-accelerated digital twin frameworks for clinical and research applications.

1. Introduction

In 1952, Hodgkin and Huxley introduced a foundational model of neuronal action potentials, providing a basis for decades of electrophysiology research. Their framework has been extensively adapted for cardiac modeling, crucial for understanding and managing heart diseases, which remain a leading global cause of mortality.

Digital twins, virtual simulations based on patient-specific data and biophysical models, have transformative potential for personalized cardiac care [1–3]. These twins enable early risk assessment and treatment planning but face computational challenges due to the high cost of solving biophysical models, especially under uncertainty [4]. This limits their scalability in real-time clinical workflows.

Neural networks may offer an efficient alternative, approximating differential equation solutions through deep architectures [5]. Physics-Informed Neural Networks (PINNs) further enhance this by embedding physical laws into training, reducing data dependency and ensuring accuracy [6]. This study employs three kinds of neural networks for solving the FitzHugh–Nagumo model: Data-Driven Neural Networks (DDNNs), PINNs, and Iterative Neural Networks (ITNNs). Using GPU acceleration and TensorRT for optimization, these differentiable surrogates combine accuracy and efficiency, enabling scalable digital twin applications for cardiac care.

2. Methods

To accelerate electrophysiology simulations, we replace traditional numerical solutions of the FitzHugh–Nagumo (FHN) [7] model with differentiable neural network surrogates. While simple, the FHN model captures key features of action potentials such as excitability and recovery. It is governed by two coupled ODEs:

$$\frac{du}{dt} = c\left(u - \frac{u^3}{3} - w\right) + I_{\text{iapp}}, \tag{1}$$

$$\frac{dw}{dt} = \frac{u + a - bw}{\tau}. \tag{2}$$

Here, u is the membrane potential, w the recovery variable, while a, b, c, and τ are model parameters. We train surrogates for the following three problem settings:

- Problem A: Outputs $u, w(t)$ from a single time input t.
- Problem B: Parameters for initial conditions also included: $u, w(t, u_0, w_0)$.
- Problem C: Adds parameterization of I_{iapp}, which modulates oscillation frequency: $u, w(t, u_0, w_0, I_{\text{iapp}})$.

This model is used to generate separate training/validation datasets to train the surrogates using a numerical solution via the Euler method ($= 0.01$ ms), evenly exploring the parameter space. After training, surrogate

speeds are benchmarked against a highly optimized GPU implementation of the Euler solver using $= 0.1$ ms, the highest possible time step size for the error to be less than 10^{-2}, on the same hardware (RTX 4070 GPU, Intel i5-12400F CPU). Training is done with 1000, 10000, and 100000 samples for each problem respectively, while inference time and validation error are measured for solving 100,000 different samples over 50 ms.

We employ three distinct types of neural networks that share the same underlying MLP structure but learn with different loss functions. Multiple architectures are tested by parameterizing MLP generation with properties such as number of layers, neurons, and activation functions per layer. Resulting architectures are classified by shape depending on the relation of layer sizes, labeling them square, rectangle, bottleneck, funnel, or bowtie-shaped networks. Every possible architecture is mapped and a set of 800 candidates is selected to evenly represent various model sizes and shapes.

Data-Driven Neural Networks. Data-Driven Neural Networks (DDNNs) are regression-based neural models that learn to approximate the solutions of differential equations by directly fitting numerical data obtained from an Euler solver $\hat{U}, \hat{W}$. In this approach, the network is trained to minimize the discrepancy between its predictions and the generated solutions. The training loss is the mean-squared error (MSE) computed over a training set containing numerical solutions:

$$L_D = \frac{1}{N_{\text{set}}} \sum_{i=1}^{N_{\text{set}}} \frac{1}{2} \Big(\hat{U}, \hat{W}(t^i, \phi^i) - U, W_{\text{solver}}(t^i, \phi^i) \Big)^2.$$

Physics-Informed Neural Networks. PINNs integrate the governing FHN differential equations directly into the loss function via automatic differentiation (AD), exploiting the fully differentiable nature of our surrogate models. In a forward pass, the network maps inputs (t, ϕ) to predictions $(\hat{U}, \hat{W})$; a subsequent backward pass computes the time derivatives $\frac{\partial \hat{U}}{\partial t}$ and $\frac{\partial \hat{W}}{\partial t}$ using the chain rule. These derivatives are compared against the expected dynamics, encapsulated in the residual function $f(t, \phi)$ that includes FHN Equations (1) and (2), resulting in the physics-based loss:

$$L_{\text{physics}} = \frac{1}{N_{\text{b}}} \sum_{i=1}^{N_{\text{b}}} \Big(\frac{\partial \hat{U}, \hat{W}(t^i, \phi^i)}{\partial t} - f_U, f_W(t^i, \phi^i) \Big)^2,$$

where N_b is the batch size. To further enforce initial or boundary conditions, an additional loss term is defined:

$$L_B = \frac{1}{N_{\text{b}}} \sum_{i=1}^{N_{\text{b}}} \Big\| \hat{U}(t=0, \phi^i) - U_0^i,\ \hat{W}(t=0, \phi^i) - W_0^i \Big\|^2.$$

When available, a data-fitting term L_D may also be incorporated. The combined PINN loss is then expressed as the weighted sum of each term. We use weight 1 for data and the PINN interior constraint, and weight 100 for the boundary.

Iterative Neural Networks. In contrast to the DDNN and PINN approaches, which learn the entire temporal trajectory in one pass, ITNNs learn a discrete "update rule" that advances the system state from (U_t, W_t) at time t to $(U_{t+\Delta t}, W_{t+\Delta t})$ at time $t + \Delta t$. The MSE loss then uses numerical data to enforce the update rule:

$$L_{\text{I}} = \frac{1}{N_{\text{b}}} \sum_{i=1}^{N_{\text{b}}} \Big(\hat{U}_{\text{itnn}}, \hat{W}_{\text{itnn}}(U_t^i, W_t^i, \phi_i) - U_{t+\Delta t}^i, W_{t+\Delta t}^i \Big)^2. \tag{3}$$

By iteratively applying this learned update, the entire trajectory from $t = 0$ to $t = T$ can be approximated. While larger time-steps (Δt) accelerate inference, they require the network to model more complex transitions; conversely, smaller Δt values simplify individual steps at the cost of increased iterations. For a fair benchmark, we sample the PINNs and DDNNs networks once per time unit (i.e., $\Delta t = 1$) and we use the same interval for the iterative network discrete step, allowing for the same number of network passes per solution.

All models are trained with the Adam optimizer (initial learning rate 1×10^{-3}, reduce-on-plateau schedule). After training the models are compiled using the Tensor RT engine in order to leverage tensor cores during inference, and them are bench-marked.

3. Results and Discussions

In general, after optimization, all three approaches employed, DDNNs, PINNs, and ITNNs, yielded at least some good surrogates where NNs managed to learn almost the entire solution space, with only minor error regions concentrated around bifurcation points, as shown in Figure 1. However, each class has its own caveats: PINNs were only justifiable in data-scarce scenarios, outperforming DDNNs when data is insufficient but providing no positive effect when data is sufficient; ITNNs performed worse than the continuous models, especially in stiff parts of the solutions, requiring extensive optimization to produce models where

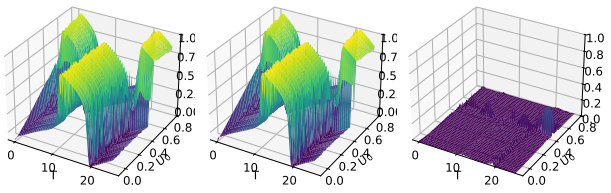

Figure 1. Example of NN accuracy: true solution (left) emulator prediction (center) solutions, and error(right) for a random surrogate of Problem B.

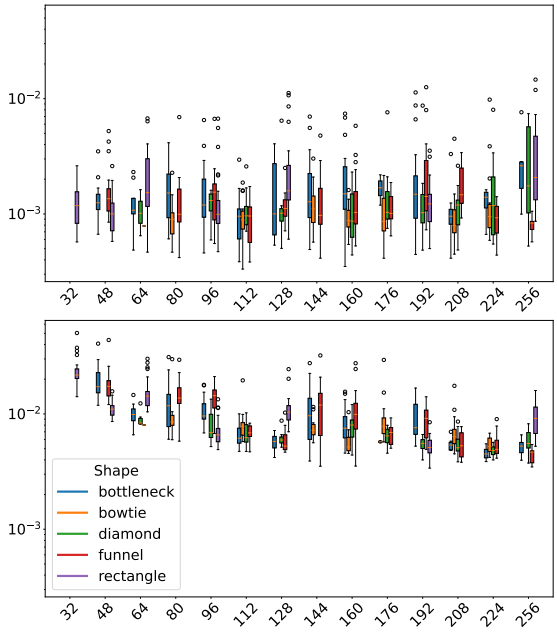

Figure 2. MAE for 800 models trained for Problems A (top) and C (bottom), grouped by architecture shape and neuron count (x axis).

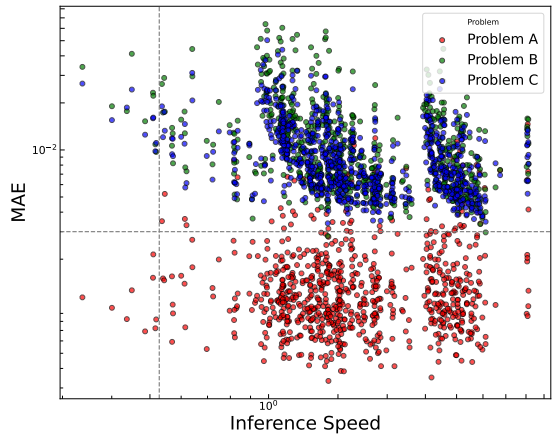

Figure 3. Model error vs. inference speed (time to evaluate 100000 samples in seconds) for models across problems. Euler results shown dashed.

successive iterations did not diverge; and finally, while purely data-driven models (DDNNs) had the best accuracy results, they required large training sets.

PINNs only improved surrogate accuracy when the training data was insufficient to describe the solution well, where the data-driven constraints stagnate with the model far from the actual solution (mean error larger than 0.1). Further optimizing the physics constraint then pushes the model much closer to the true solution, decreasing the error by up to a 100× factor. However, when abundant data is available, both models converge similarly, rendering the physics constraint redundant. In such cases, enforcing physics constraints can hinder training in two primary ways: increasing computational cost per training iteration due to additional derivative computations (we observed an increase in training time of around 2×), and slowing convergence by introducing a more complex and difficult-to-optimize loss landscape [8, 9], resulting in less efficient training. Thus, it follows that it is more beneficial to run the numerical simulation enough times to produce a complete training set than to directly incorporate the ODEs into training. The amount of data required for this effect varies: about 100 points are sufficient for the one-dimensional Problem A, but depending on the range of the additional parameters, this increases proportionally.

ITNNs generally perform worse than continuous models. While they can achieve low single-iteration errors, recurrent evaluation often leads to two failure modes (Fig. 4): (I) small errors accumulate over iterations, causing a temporal shift from the true solution, and (II) high errors or parameter-space anomalies push the model into unstable equilibria. Even with added data for high-error regions, few surrogates avoided these issues, and successful time-series solutions were not consistently linked to surrogate size.

Therefore, when enough data is available, the DDNN models represent a better choice, and such is the case when building ODE solution surrogates. The cost associated with generating a larger set is offset by the better performance of DDNNs and the lower training time relative to PINNs. Compared to ITNNs, they are simply more robust, yielding sufficiently accurate solutions in most cases, while retaining the key differentiability attribute. However, depending on the architecture, the DDNNs are able to approximate the ODE solution with different degrees of accuracy. Surrogate accuracy varied greatly, even for models with the same neuron count and number of layers

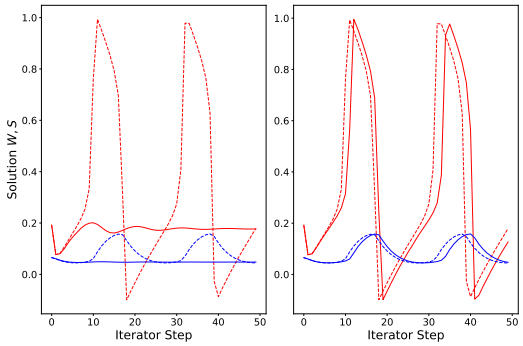

Figure 4. Failure modes of ITNN models: unstable equilibrium (left) and growing temporal shift (right). Numerical solution shown dashed.

as shown in Figure 2, with architecture size limiting accuracy only up to a certain point that changes depending on problem complexity. Increasing surrogate size beyond that point has little effect, and up to 10× variation in accuracy is observed for same-size architectures, with no single activation function sequence or shape achieving consistent advantage. Thus, model optimization played a crucial role in identifying efficient models.

Systematic evaluation of architectures reveals that most models are slower than numerical schemes, with inference speed primarily influenced by neuron count rather than by the number of layers, constraining effective surrogates to two- or three-layer models with up to 256 neurons. While neuron count has a direct impact on inference speed, its relationship with accuracy is more nuanced; larger models exhibit lower accuracy variance and benefit less from architecture optimization. Statistical analysis shows that for smaller models (< 120 neurons), architecture shape significantly affects performance, as indicated by higher F-values (19.6 vs. 4.00 for larger models). However, this influence diminishes with increasing model size, where neuron count becomes dominant (F-values of 42.8 and 34.5 for small and large models, respectively). Variance analysis supports this trend: smaller models exhibit higher extra-group variance in mean error (1.46×10^{-5} vs. 1.06×10^{-6}), while larger models see converging variances (3.17×10^{-7} vs. 2.08×10^{-7}). Thus, optimization efforts are better spent focusing on activation function choice.

4. Conclusions

Our study evaluated neural network surrogates for the FitzHugh–Nagumo model, comparing Data-Driven Neural Networks, Physics-Informed Neural Networks, and Iterative Neural Networks across varying complexities. DDNNs excel in data-rich scenarios, demonstrating robustness and lower training costs, outperforming PINNs and ITNNs in most cases. ITNNs often struggle with error accumulation, particularly in long-term predictions, while PINNs perform best in data-scarce scenarios due to their integration of physical constraints. As shown in Figure 3, small, accurate models work well for simpler problems but fail as complexity increases, particularly for Problems B and C. The shift from Problem A to B, which introduces initial condition parameterization, presents more challenges than the shift from B to C, which adds a model parameter. This suggests that learning variations in features (e.g., wave shape, length) is easier than capturing shifted solutions. Additionally, architectural design, such as neuron count and activation functions, significantly influences accuracy, especially in smaller models. With hardware acceleration provided by TensorRT, these smaller neural network surrogates can even surpass numerical solvers in speed while remaining differentiable. These findings highlight the trade-offs between data availability, computational cost, and model design, providing insights into how neural surrogates can complement traditional numerical methods in complex simulations.

Acknowledgments

The authors would like to express their thanks to Minas Gerais State Research Support Foundation (FAPEMIG), "Coordenação de Aperfeiçoamento de Emprego de Nível Superior" (CAPES), "Conselho Nacional de Desenvolvimento Científico e Tecnológico" (CNPq) Proc. 423278/2021-5 and 310722/2021-7; "Empresa Brasileira de Serviços Hospitalares" (Ebserh), National Council for Scientific and Technological Development (CNPq), SINAPAD Santos-Dumont, and Federal University of Juiz

References

[1] Coorey Glen FGAFDFSVJ. The health digital twin to tackle cardiovascular disease—a review of an emerging interdisciplinary field. npj Digital Medicine 2022;5(1):6.

[2] Sel K. ODZF. Building digital twins for cardiovascular health: From principles to clinical impact. Journal of the American Heart Association 2024;13:e031981.

[3] De Lepper A, Buck C. From evidence-based medicine to digital twin technology for predicting ventricular tachycardia in ischaemic cardiomyopathy. Journal of the Royal Society Interface 2022;19(188):20220317.

[4] Campos J, Sundnes J, Dos Santos R, Rocha B. Uncertainty quantification and sensitivity analysis of left ventricular function during the full cardiac cycle. Philosophical Transactions of the Royal Society A 2020;378(2173):20190381.

[5] Qian, Sun Y, Zheng Wang L, Huang Y. Accelerating sparse deep neural network inference using gpu tensor cores. In 2022 IEEE High Performance Extreme Computing Conference (HPEC). IEEE, 2022; 1–6.

[6] Raissi M, Perdikaris P, Karniadakis GE. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. arXiv preprint arXiv171110561 2017;URL https://arxiv.org/abs/1711.10561.

[7] FitzHugh R. Impulses and physiological states in theoretical models of nerve membrane. Biophysical journal 1961; 1(6):445–466.

[8] Rathore V, Cai W, Wu H. Mitigating the ill-conditioning of physics-informed neural networks. arXiv preprint arXiv240201868 2024;.

[9] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Communications of the ACM 2012;60(6):84–90.

Address for correspondence:

Yan Barbosa Werneck
Rua São Mateus 872 Juiz de Fora, Brazil
yanbwerneck@outlook.com