Comparing Classical Ordinary Differential Equation and Neural Network Models for Reduced-Order Single-Cell Electrophysiology

Yan Barbosa Werneck[1] **, Bernardo Martins**[1] **, Rodrigo Weber**[1]

[1] Programa de Pós-Graduação em Modelagem Computacional (PPGMC)
Universidade Federal de Juiz de Fora (UFJF) – Juiz de Fora, Brazil

yanbwerneck@outlook.com

Abstract. *Electrophysiology modeling is key for non-invasive diagnostics and understanding heart and brain function. Traditional models use ODEs, from detailed ion channel dynamics to reduced-order phenomenological models. We compare a fast reduced-order model with data-driven and physics-informed neural networks as efficient alternatives to numerical solutions. Using the FitzHugh-Nagumo model, we trained networks with numerical data and model physics, employing architecture optimization, adaptive point density, and time-domain splitting. Inference via TensorRT achieved up to 1.8× speedup over optimized CUDA solvers with minimal accuracy loss. These results showcase neural networks as viable emulators when complexity is controlled.*

1. Introduction

Electrophysiology malfunctions, leading to heart and brain diseases, are among the leading causes of mortality worldwide, contributing to millions of deaths annually. Early detection through preventive examinations is essential for mitigating their impact by identifying risk factors before they develop into severe conditions. Digital twin technology has emerged as a promising tool for enhancing risk assessment by enabling virtual simulations of complex physiological systems [Coorey 2022, Sel 2024]. These digital twins integrate large-scale heart simulations by combining conduction models based on partial differential equations (PDEs) with action potential (AP) electrophysiology models governed by ordinary differential equations (ODEs). However, solving these models numerically is computationally expensive, particularly when accounting for parameter uncertainties [Campos et al. 2020] and inter-individual variability, which limits their feasibility for real-time applications and large-scale exploratory studies.

To address these challenges, this work investigates neural network-based surrogate models as a means to accelerate electrophysiology simulations. Recent advancements in deep learning, particularly Physics-Informed Neural Networks (PINNs), have shown promise in replacing full-scale numerical solvers with neural network surrogates, potentially reducing computational costs while preserving model dynamics and predictive capabilities [Qian et al. 2022, Raissi et al. 2017]. Our study develops and benchmarks three neural network architectures—data-driven neural networks (DDNNs), PINNs, and iterative neural networks (ITNNs)—optimized for approximating action potential models using the FitzHugh-Nagumo framework. These architectures are specifically tailored to capture the essential dynamics of AP propagation while significantly reducing computational overhead. We focus on scalability and GPU-acceleration, utilizing NVIDIA's

TensorRT to optimize inference performance, which facilitates high-throughput processing and broad hardware compatibility.

A key advantage of our surrogate models is their differentiable nature, which enables rapid simulations and enhances capabilities for sensitivity analysis, optimization, and uncertainty quantification. By integrating these differentiable surrogates into modern differentiable programming frameworks , they can be seamlessly applied to gradient-based algorithms for parameter inference, model-based control, and end-to-end training. This approach offers a pathway to making digital twins viable for real-time decision-making and large-scale predictive modeling, with applications spanning both clinical and research settings. Through careful optimization and leveraging cutting-edge AI hardware, we aim to enhance the practical applicability of neural network-based surrogates in modeling complex electrophysiological systems.

2. Methods

2.1. FitzHugh-Nagumo Model and Efficient Numerical Solution on CUDA

This work replaces the numerical solution of the FitzHugh-Nagumo (FHN) model with neural networks. The FHN model, a reduced representation of action potential dynamics, is governed by two coupled ODEs:

$$\frac{du}{dt} = c \cdot \left(u - \frac{u^3}{3} - v \right) + I_{iapp}, \quad \frac{dv}{dt} = \frac{u + a - b \cdot v}{\tau}, \tag{1}$$

where u represents the membrane potential, v the recovery variable, and a, b, c, and τ are model parameters. The model captures key features of action potential dynamics, such as excitability, threshold behavior, and recovery [FitzHugh 1961], presenting a complex nonlinear solution space.

We considered three levels of parametrization for the FHN model, labeled Problems A, B, and C:

- **Problem A:** Single continous trajectory $u, v(t)$.
- **Problem B:** Parametrizes the initial conditions (u_0, v_0).
- **Problem C:** Introduces I_{iapp} that modulates the frequency of oscillations.

Training datasets contain solutions for uniformly sampled parameters, solved using an optimized CUDA-based Euler method. GPU parallelism ensures state-of-the-art performance that serve as a baseline for comparison with the emulators.

2.2. Neural Network Surrogates and Training

We use parameterized multi-layer perceptrons (MLPs) to approximate the FHN model solutions, with multiple architectures. Three classes of models are employed: Data-Driven Neural Networks (DDNN), Physics-Informed Neural Networks (PINN), and Iterative Neural Networks (ITNN). DDNNs fit numerical solutions by minimizing the mean-squared error (MSE) loss:

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

PINNs incorporate the FHN equations directly into the loss function using automatic differentiation, enforcing physics consistency:

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

Boundary conditions are penalized with L_B, and a combined loss balances data, physics, and boundary terms:

$$L_{\text{PINN}} = \lambda_D L_D + \lambda_P L_{\text{physics}} + \lambda_B L_B. \tag{3}$$

ITNNs learn discrete update rules, advancing the solution from (U_t, W_t) at time t to $(U_{t+\Delta t}, W_{t+\Delta t})$ at time $t + \Delta t$, with an MSE loss:

$$L_{IT} = \frac{1}{N_{batch}} \sum_{i=1}^{N_{batch}} \Big(\hat{U}, \hat{W}_{itnn}(U_t^i, W_t^i, \phi_i) - U^i, W_{t+\Delta t}^i \Big)^2. \tag{4}$$

By iterating these update rules, trajectories from $t = 0$ to $t = T$ are approximated.

2.3. Experimental Design

We evaluate the computational cost of the trained neural networks against that of the numerical solver, for this a optimized CUDA implementation of the Euler solver is compared with optimized neural networks accelerated using NVIDIA's TensorRT.

Models are optimized by exploring various architecture and activation function combinations, including rectangle, funnel, bottleneck, bowtie, and diamond shapes. These structures are chosen for their common uses in feature extraction, generative models, and autoencoders. TensorRT accelerates matrix multiplications in fully connected layers, significantly reducing computational costs. A set of 800 candidate architectures is sampled with a balanced distribution of model sizes and paired with appropriate input/output layers and loss functions. Models are trained in parallel on a multi-GPU cluster using MPI, and inference time is measured after compiling with the TensorRT framework.

Experiments were conducted on an NVIDIA RTX 4070 GPU and Intel i5-12400F CPU. Inference time is measured for both the numerical and neural approaches, with minimal overhead from data transfers. Neural network batch size is tuned for optimal throughput, while the Euler solver uses a time step of $\Delta t = 0.1\,\text{ms}$, and neural networks are tested with a sampling frequency of $\delta t = 1\,\text{ms}$.

3. Results and Discussions

PINNs showed clear benefits in data-scarce scenarios, outperforming DDNNs when data was insufficient (Fig. 1), as they leveraged the physics constraint to push the model closer to the true solution. However, when data was abundant, both models converged similarly, and enforcing physics constraints became detrimental. This was due to increased computational cost (training time doubled) and slower convergence from a more complex loss landscape. Thus, it is more efficient to generate a complete training set from numerical simulations than to incorporate ODEs directly into training.

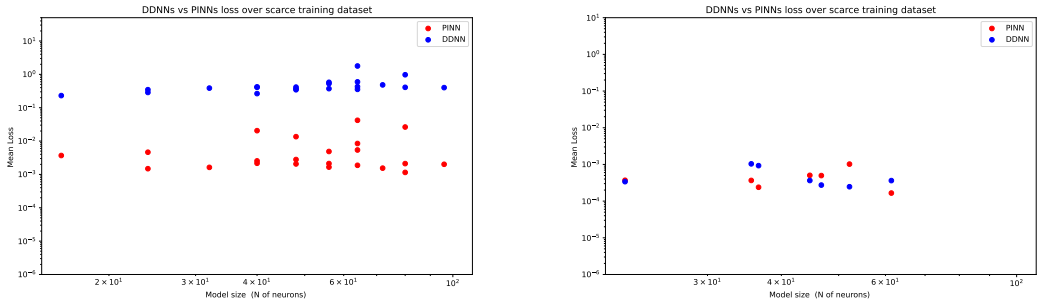

Figure 1. PINNs and DDNNs trained on a scarce dataset with 10 samples (top) and an abundant dataset with 1000 samples (bottom). PINNs achieve consistently lower validation mean error in the data-scarce scenario, whereas both methods perform similarly when data is abundant. A T-test value of 7.42 confirms a significant difference in favor of PINNs in the scarce setting, while 0.33 indicates no statistical difference in the abundant case

When ample data was available, DDNNs could approximate the ODE solution, with performance varying depending on the architecture. No single architecture consistently outperformed others, highlighting the importance of model optimization. Smaller models exhibited higher variance in accuracy, while larger models showed more stable performance. For smaller models, architecture shape significantly influenced performance, while for larger models, neuron count became the dominant factor. This trend indicates that as model capacity increases, optimization should focus on neuron count and fine-tuning activation functions, rather than altering the architecture itself.

Furthermore, as the problem complexity increased (e.g., moving from Problem A to Problem B), larger models were required to maintain accuracy, highlighting the need for increased model expressiveness in complex scenarios. In particular, simpler architectures struggled more when parameterizing initial conditions than when adding model parameters, as learning a few features of the solution was easier than modeling an entire set of shifted solutions.

The inference stage of the models reveals the trade-off between accuracy and inference speed. Notably, a quasi linear relation between neuron count and inference time was observed, shown in 4. Considering that only the smallest models were faster than the numerical solution, and the high-variability observed for small model accuracy, it follows that architecture optimization was crucial for finding models that were fast enough yet reasonably accurate.

4. Conclusion

This study investigates the use of neural networks as differentiable surrogates to traditional numerical solver of action potentials ODE models such as the FHN. Our results suggest that the most efficient way to train an ODE surrogate is to simply using the numerical solve to generate a lot of training data and using purely data driven approach, rather than trying to learn directly from the ODEs with physics informed learning.

Several techniques were employed to optimize training and reduce inference times, such as architecture optimization to balance model size and performance, time-domain splitting to reduce complexity, and increasing cloud point density in high-error

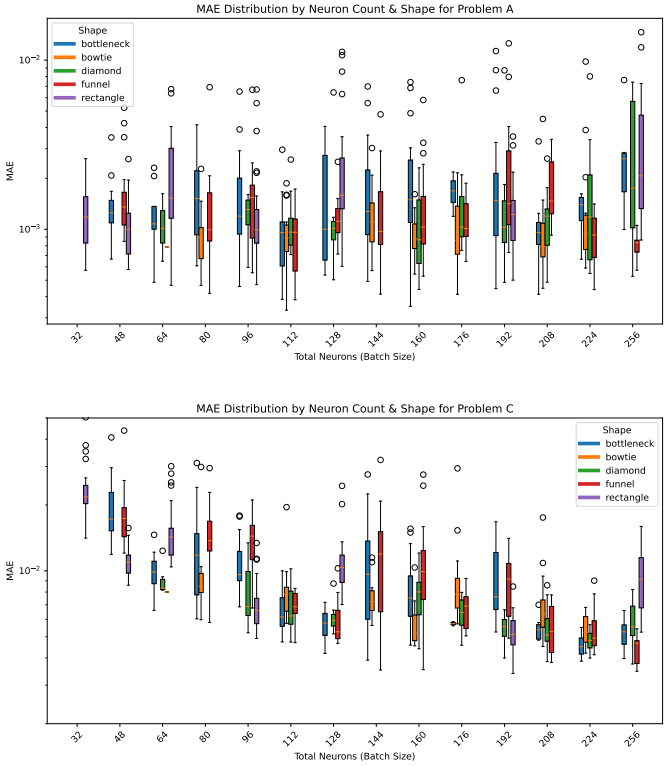

Figure 2. Set of 800 models trained for Problem A (top) and C (bottom). Results in terms of MAE, models are separated by shape and number of neurons.

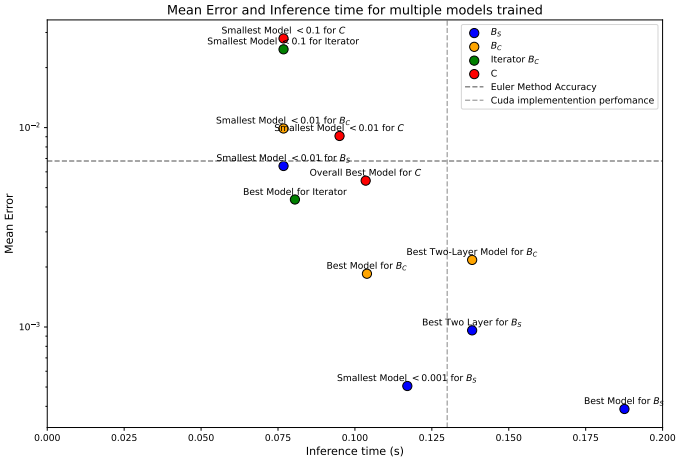

Figure 3. Model error and inference performance for multiple models trained during this work. The plot shows models with the best trade-off between accuracy and inference speed for each problem. Numerical solution for the model using the Euler method and $dt = 0.1$ is shown for reference with dotted lines.

regions. Further optimization is done by running the models with TensorRT, which leverages tensor cores to accelerated matrix calculations, this yields achieving speeds up of up to 1.8× in relation to the numerical solver. Our results shown that this level of optimization

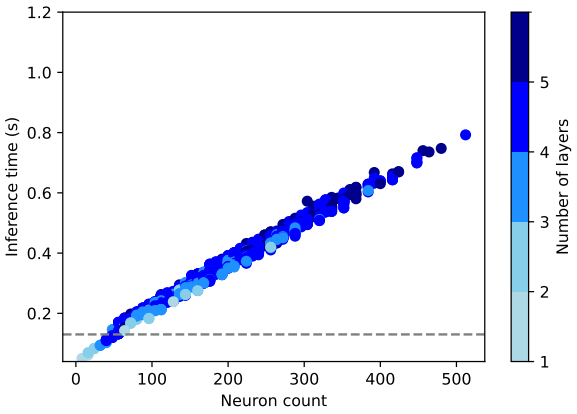

Figure 4. Inference time for the reference set for multiple models with various architectures. Results shows an increase in cost as neuron count increases. Dotted line shows the Euler implementation results for the same set.

is required in order to produce emulators more efficient than the numerical solution.

In conclusion, this work demonstrates that both PINNs and data-driven models can serve as effective surrogates for excitable cell models like FHN and more complex ones like the Hodgkin-Huxley model. The resulting model are fully differentiable and can be easily integrated into various sensitivity analysis, optimization and uncertainty quantification pipelines. Although computational advantages diminish with increased model complexity, strategic model design and hardware acceleration can mitigate these issues.

References

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

Coorey, Glen, F. G. A. F. D. F. S. V. J. (2022). The health digital twin to tackle cardiovascular disease—a review of an emerging interdisciplinary field. *npj Digital Medicine*, 5(1):6.

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

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

Raissi, M., Perdikaris, P., and Karniadakis, G. E. (2017). Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561*.

Sel, K., O. D. Z. F. (2024). Building digital twins for cardiovascular health: From principles to clinical impact. *Journal of the American Heart Association*, 13:e031981.