Original scientific paper

UDK 517.938:004.738.5
DOI 10.7251/COMEN2301043M

NEURAL NETWORKS FOR SOLVING HUXLEY'S EQUATION

Bogdan Milićević[1,2*], Miloš Ivanović[2,3], Boban Stojanović[2,3], Nenad Filipović[1,2]

[1] Faculty of Engineering Sciences, University of Kragujevac, Kragujevac, Serbia
[2] Research and Development Center for Bioengineering, Kragujevac, Serbia
[3] Faculty of Science and Mathematics, University of Kragujevac, Kragujevac, Serbia
[*] Corresponding author: bogdan.milicevic@uni.kg.ac.rs

Abstract: Biophysical muscle models, also known as Huxley-type models, are appropriate for simulating non-uniform and unsteady contractions. Large-scale simulations can be more challenging to use because this type of model can be computationally intensive. The method of characteristics is typically used to solve Huxley's muscle equation, which describes the distribution of connected myosin heads to the actin-binding sites. Once this equation is solved, we can determine the generated force and the stiffness of the muscle fibers, which may then be employed in the macro-level simulations of finite element analysis. In our paper, we developed a physics-informed surrogate model that functions similarly to the original Huxley muscle model but uses a lot less computational resources in order to enable more effective use of the Huxley muscle model.

Keywords: physics-informed neural networks, numerical analysis, machine learning, Huxley's muscle model.

1. INTRODUCTION

To analyze muscle behavior via in silico analysis we model biophysical processes on multiple spatial and temporal scales. We perform multi-scale simulation in which continuum muscle mechanics is modeled using the finite element method and material characteristics of muscle at the microscopic scale are defined by Huxley's muscle contraction model [1]. During transient finite element simulation, we use Huxley's model to calculate stress and instantaneous stiffness, given the muscle activation, stretch, and other material parameters and properties. These finite element simulations can be quite computationally intensive. The most time-consuming part of these simulations are calculations carried out at the microscale. To lower the computational requirements of the simulations, we create a computationally more efficient surrogate model to replace the real Huxley muscle model. We solved the Huxley equation, using physics-informed neural networks, to acquire the distribution of attached myosin heads to the actin-binding sites.

2. METHODS

Huxley considered the dynamics of the filaments within muscle and the probability of establishing connections (cross-bridges) of myosin heads to actin filaments inside sarcomeres. The $n(x,t)$ function describes the rate of connections between myosin heads and actin filaments, as a function of the position of the nearest available actin-binding site relative to the equilibrium position of the myosin head x:

$$\frac{\partial n(x,t)}{\partial t} - v\frac{\partial n(x,t)}{\partial t} = [1 - n(x,t)]f(x,a) - n(x,t)g(x) \quad (1)$$

where $f(x,a)$ and $g(x)$ represent the attachment and detachment rates of cross-bridges respectively, v is the velocity of filaments sliding, positive in the direction of contraction, and a is muscle activation given as a function of time. Once the $n(x,t)$ values are acquired we can calculate generated force F within the muscle fiber and also stiffness K using the equations:

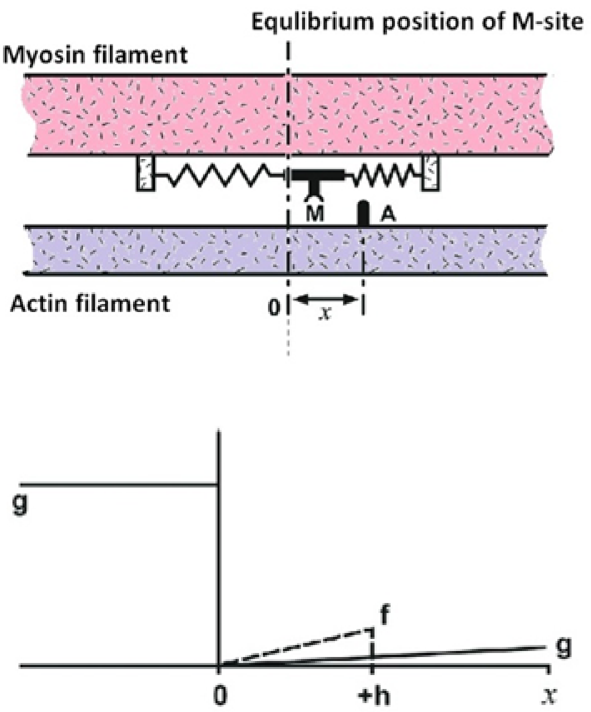
$$F(t) = k \sum_{-\infty}^{\infty} n(x,t)x \ dx$$

and $$K(t) = k \sum_{-\infty}^{\infty} n(x,t) \ dx \qquad (2)$$

where k is the stiffness of cross-bridges. Stress and stress derivatives can be calculated as:

and $$\sigma_m = F\frac{\sigma_{iso}}{F_{iso}}, \quad \frac{\partial \sigma_m}{\partial e} = \lambda L_0 K \frac{\sigma_{iso}}{F_{iso}} \qquad (3)$$

where F_{iso} is maximal force achieved during isometric conditions, σ_{iso} maximal stress achieved during isometric conditions, L_0 *the* initial length of sarcomere and λ is stretch. Calculated stresses and stress derivatives can be further used at the macro-level during finite element analysis.



a)

b)

Figure 1. *Schematic of the Huxley's muscle model*

Physics-informed neural networks (PINNs) are trained to solve supervised learning tasks while respecting any given law of physics described by general nonlinear partial differential equations [2]. These neural networks form a new class of data-efficient universal function approximators that naturally encode any underlying physical laws as prior information [2]. The major innovation with PINN is the introduction of a residual network that encodes the governing physics equations, takes the output of a deep-learning network, called a surrogate, and calculates a residual value [3]. The basic formulation of the PINN training does not require labeled data, results from other simulations or experimental data, and is unsupervised. PINNs only require the evaluation of the residual function. Providing simulation data or experimental data for training the network in a supervised manner is also possible and necessary in some cases, especially inverse problems. The supervised approach is often used for solving ill-defined problems when for instance we lack boundary conditions or an Equation of State to close a system of equations. Once a PINN is trained, the inference from the trained PINN can be used to replace traditional numerical solvers in scientific computing [3]. PINNs are a gridless method because any point in the domain can be taken as input without requiring the definition of a mesh. Moreover, the trained PINN network can be used for predicting the values on simulation grids of different resolutions without the need of being retrained [3]. PINNs can also be used for time-dependent problems. Since time is represented as any other variable, it's possible to have a prediction of output at the specified time without solving for previous time steps. For these reasons, the computational cost does not scale with the number of grid points like many traditional computational methods. PINN has been employed for predicting the solutions for the Burgers' equation, the Navier–Stokes equations, and the Schrodinger equation [4]. In this study, we focused on the basic PINNs and solving PDE without relying on other simulations to assist the training. The residual of the differential equation is minimized by training the neural network. PINNs calculate differential operators on graphs using automatic differentiation. To implement PINN and incorporate the equation (1), we used SciANN [5], a high-level artificial neural networks API, written in Python using Keras and TensorFlow backends. SciANN is designed to abstract neural network construction for scientific computations and solution and discovery of partial differential equations (PDE) using the physics-informed neural networks [5].

For the isometric case of the muscle contraction, we used a standard approach for physics-informed neural networks, without data collected from simulations. In this case, inputs to the network consist only of time t and x, since there is no change in stretch and activation of the muscles, and the network predicts the n value. To solve the isotonic case of muscle contraction, we also provided data to the neural network in a standard supervised manner. The data were collected from multi-scale finite element simulation with the original Huxley muscle model built-in. Our neural network, or a surrogate model, receives current and previous stretch, muscle activation, time, and x, and based on these values predicts the n value. During training the residual of Huxley's muscle differential equation, residual of initial condition and error between true and predicted n values are minimized. After the training is done, the neural network is loaded in multi-scale finite element simulation and used instead of the original Huxley muscle model.

3. RESULTS

To solve the isometric muscle contraction case, using the *SciANN* framework we constructed a neural network with 8 hidden layers, each containing 20 neurons with a hyperbolic tangent activation function. The network is trained by minimizing the residual derived from equation (1) and by providing initial conditions to solve the equation. We used Adam optimizer [6] with a learning rate of 10^{-4} and batch size of 512, with a total number of 10000 epochs. We also used the neural tangent kernel (*NTK*) method to get the adaptive weights, balancing the difference between the number of collocation points, used to minimize the residual of PDE, and the number of points used to minimize the residual of the initial condition. We generated a data grid consisting of x values in the range of $-20.8\ nm \leq x \leq 62.4\ nm$ and t values in the range $0s \leq t \leq 0.5s$. These generated points were used to train the network. Once the network was trained we calculated generated stress based on n values and we compared the solutions obtained by the method of characteristics and *PINN*. We can see that the PINN produces very similar results as the method of characteristics.

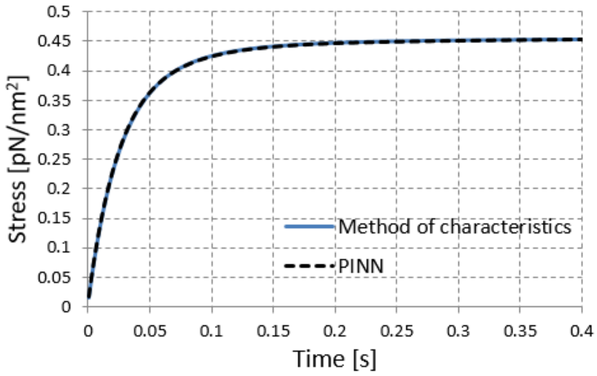


Figure 2. *Isometric contraction case*

To solve the isotonic muscle contraction case, we first ran 10 numerical experiments with random muscle activation functions, such that the muscles contracted and then returned to the initial positions. These finite element simulations consist of one element as shown in Fig.3. The model is constrained in all directions in point A, it can slide up and down at point C, and at points, D and B the model is free to move. We collected x values, time, activation, and stretch values (input data), and also the produced n values (output data) from these numerical experiments. Then, we fed this data to the neural network, and we also provided the encoded Huxley equation with initial conditions to the network. Using the *SciANN* framework we constructed a neural network with 8 hidden layers each containing 200 neurons with a hyperbolic tangent activation function. Adam optimizer with a learning rate of 10^{-4} and batch size of 16384, with a total number of 30000 epochs. We also used the neural tangent kernel (*NTK*) method. Once the training was finished we integrated the network as a material model into the finite element analysis software. At the macro level, finite elements provide the neural network with muscle activation, time, and current stretch, based on these values and x values the network predicts the n values. Using n values we calculated the stress and stress derivatives and compared the values in the case of the method of characteristics and *PINN* (Fig. 4). The isotonic case is more complicated since the muscle activation varies, and the velocity of the contraction is non-zero, so we didn't achieve as precise results as with the isometric case but there is a similarity between stresses obtained by the method of characteristics and PINN.

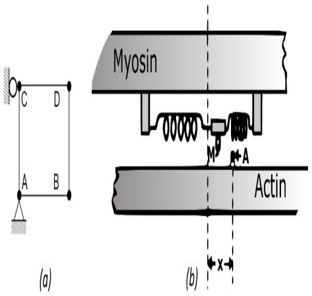


Figure 3. *Finite element model*

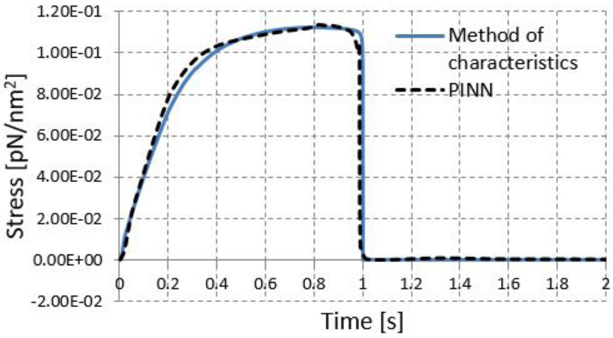


Figure 4. Isotonic contraction case

Finally, we measured the execution time of the multi-scale finite element simulations with PINN and with methods of characteristics at the micro-level. It can be seen in Fig.5 that we achieved 10 times speed-up with PINN compared to the method of characteristics.

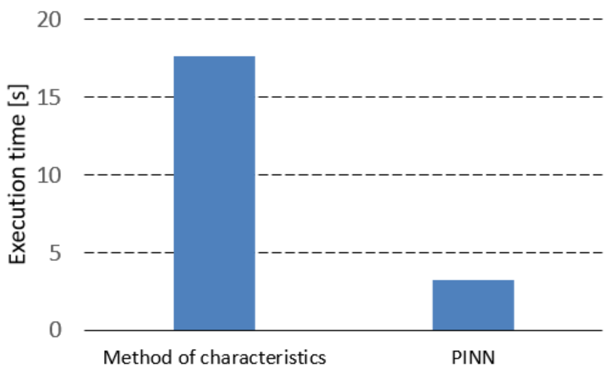


Figure 5. *Execution time*

4. CONCLUSIONS

In this paper, we presented the surrogate model of the Huxley muscle model based on physics-informed neural networks. We collected data from multi-scale finite element simulation and trained the network to produce probabilities of attachment of the myosin heads to actin-binding sites. Based on these probabilities we calculated stress and instantaneous stiffness. Once the neural network was trained, the surrogate was built into a finite element solver, as a material model, and we compared the stresses obtained from the original and the surrogate model. We showed that the surrogate model produces similar outputs as the original model, concluding that the surrogate model has the potential to replace the original model within finite element simulations. Our future research will focus more on isotonic contractions, and if we achieve higher precision using one finite element we will use the surrogate model in the large-scale models such as the left ventricle model.

5. REFERENCES

[1] B. Stojanovic, M. Svicevic, A. Kaplarevic-Malisic, R. J. Gilbert, S. M. Mijailovich, "Multi-scale striated muscle contraction model linking sarcomere length-dependent cross-bridge kinetics to macroscopic deformation," in *Journal of Computational Science*, Elsevier, Volume 39, 2020, January

[2] Raissi M., Perdikaris P., and Karniadakis G. E., Physics Informed Deep Learning (Part I): Data-Driven Solutions of Nonlinear Partial Differential Equations. New York City, NY: arXiv preprint arXiv:1711.10561. 2017a

[3] Markidis Stefano, The Old and the New: Can Physics-Informed Deep-Learning Replace Traditional Linear Solvers, Frontiers in Big Data, Vol. 4, ISSN 2624-909X 2021. https://doi.org/10.3389/fdata.2021.669097

[4] M. Raissi, P. Perdikaris, 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. 378 686–707. 2019.

[5] Haghighat E. and Juanes R., Sciann: A Keras Wrapper for Scientific Computations and Physics-Informed Deep Learning Using Artificial Neural Networks. New York City, NY: arXiv preprint arXiv:2005.08803. 2020.

[6] Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., & Han, J. On the Variance of the Adaptive Learning Rate and Beyond. ArXiv, abs/1908.03265. 2020.

НЕУРОНСКЕ МРЕЖЕ ЗА РЕШАВАЊЕ ХАКСЛИЈЕВЕ ЈЕДНАЧИНЕ

Сажетак Биофизички модели мишића, или модели Хакслијевог типа, су погодни за симулирање неуниформних и нестабилних контракција. Симулација са овим типом модела могу бити веома рачунски захтевне. Метод карактеристика се обично користи за решавање Хакслијеве једначине, која опсије дистрибуцију закачених миозинских глава за актин. Након решавања Хакслијеве једначине можемо одредити генерисану силу и крутост у мишићним влакнима, што даље може бити коришћено у анализи методом коначних елемената. У нашем раду развили смо сурогат модел, базиран на неуронским мрежама информисаним физичким законима, које уместо методе крактеристика рачунски ефикасније решавају Хакслијеву једначину.

Кључне ријечи: неуронске мреже информисане физичким заковнима, нумеричка анализа, машинско учење, Хакслијев модел мишића

Paper received: 24 August 2022 **Paper accepted: 12 March 2023**