Development of Data-driven Constitutive Models: Applications in the Finite Element Simulation of Hyperelastic Materials

Eduardo S. Carvalho[1], João P.S. Ferreira[1], Marco P.L. Parente[1]

[1]*Dept. of Mechanical Engineering, Faculdade de Engenharia da Universidade do Porto*
Rua Dr Roberto Frias, 4200, Porto, Portugal
eduardocarvalho@live.com.pt, joaof@fe.up.pt, mparente@fe.up.pt

Abstract. When subjected to large deformations, hyperelastic materials present a highly nonlinear behaviour, which makes their constitutive description complex and computationally expensive. Surrogate models can replace these traditional and costly models and overcome some computational limitations by learning directly from acquired data. Currently, the usage of surrogate models in commercial finite element (FE) software, such as Abaqus, is nonexistent. In this work, surrogate models, able to describe the constitutive behaviour of different materials, were developed and were then incorporated into Abaqus, using a user defined material, programmed in Fortran Language. Artificial neural networks were trained to predict the isochoric part of the Cauchy stress tensor and the spatial elasticity tensor from the existing data. With the parameters of the trained neural networks, a user defined material was developed. The present method was validated using classical benchmark problems, and the results obtained using the developed constitutive models were compared with the ones obtained with the conventional approach. The correctness of the obtained results highlights the possibility of using data-driven constitutive models to describe the behaviour of hyperelastic materials. The proposed approach can be a viable alternative, avoiding the need to express a given material constitutive equation directly.

Keywords: Machine Learning, Surrogate Model, Constitutive Modelling, Hyperelasticity, Finite Element Method

1 Introduction

The finite element (FE) method is a powerful tool used to simulate and test many engineering phenomena and components. It requires the proper definition of the domain of interest of the problem (geometry), of the applied loads and boundary conditions and of the mechanical properties of the materials in analysis. The behaviour of the considered materials is described by constitutive equations formulated mathematically and parametrized by experimental or micro-mechanically generated data [1]. However, the process of obtaining the constitutive equations for a given material is still challenging, particularly for hyperelastic materials, such as soft tissues and polymers, which are extremely nonlinear and undergo large deformations, making their constitutive modelling expensive in terms of time and computational resources [2, 3].

Standardly, to obtain such a constitutive model, it is necessary to delineate general characteristic behaviours, establish an appropriate theoretical framework, identify specific functional forms of the constitutive relation, calculate the values of the material parameters and evaluate the predictive capability of the final constitutive relation [4].

This approach requires a high level of expertise, particularly in the first three steps and was the golden standard for many years. To explore possible alternatives to this classical constitutive modelling approach, where expert-constructed constitutive equations are used, data-driven constitutive modelling is gaining more interest in recent years mainly because computer performance has improved, and more tools are available to facilitate the use of machine learning (ML) techniques.

Data-driven constitutive modelling might overcome some challenges posed by the classical approaches by providing a flexible form that can be determined directly from experimental data. One of the first works that employed data-driven constitutive modelling was the ones from Ghaboussi et al. [5] and Wu et al. [6], where a neural network (NN) was trained with experimental data to obtain stresses from strains. The material model used

in the FE method updates the stresses based on the current stress–strain state and strain increment and calculates the material stiffness matrix for the constitutive relation. The work from Ghaboussi et al. did not accomplish the second function, which led Hashash et al. [7] to propose an explicit formulation of the material stiffness matrix for NN constitutive material models. However, these works did not apply the FE method using the trained surrogates for the constitutive modelling. Afterwards, many authors proposed different data-driven constitutive models for hyperelastic materials [1, 8–11] and even extended them to anisotropic hyperelastic materials [12–14]. However, the trained models were not used in any FE commercial software. The surrogates developed were sometimes used for in-house FE method codes.

Finite element software packages usually have the behaviours of the most common materials implemented, but in the field of bioengineering, it is often necessary to define a given material of interest. To accomplish this, it is necessary to create a subroutine that computes, at each integration point, the stress state $\boldsymbol{\sigma}$ and a consistent tangent stiffness matrix $\mathbf{c}$ for a given deformation gradient $\mathbf{F}$. Conventionally, to accomplish these, a given material model, existing in the literature is used, which defines the strain energy function in a specific form. With the strain energy function, it is possible to express mathematically the constitutive equations of the material in a Fortran subroutine, named UMAT. However, this can be a complex task due to the large number of derivatives and tensorial algebra, needed to define the constitutive equations. Therefore, to overcome this difficulty, surrogate models were developed in this work, to describe the constitutive behaviour of materials. Afterwards, a used defied material model, in Fortran language, was developed to define the given material, making the surrogates compatible to be used by any FE commercial software. The framework used is illustrated in Fig. 1.

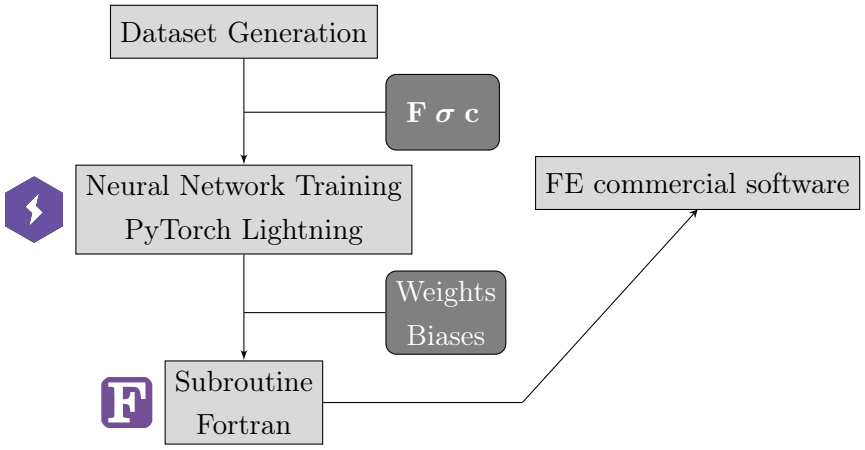


Figure 1. Framework.

2 Methodology

The models already developed by experts obey physical-based principles relevant to soft tissues and include knowledge about mechanical behaviour and underlying microstructure properties. Therefore, they were used to obtain data assuming different material models proposed in the literature, which were then used to train surrogate models that output the stresses and the spatial elasticity tensor from a given deformation gradient. More specifically, fully connected NNs were trained for two material models: the Neo-Hookean model and the Holzapfel-Gasser-Ogden (HGO) model. Afterwards, the surrogate models were used to create a Fortran subroutine, which can be used to define a material of interest in FE software, without the need to directly express its constitutive equations.

It is important to mention that all the results that are going to be presented are valid for any system of units that is consistent. As an example, if the material parameter used for the Neo-Hookean model was $C_{10} = 2$ and if the unit considered for the material parameter is MPa, all the stresses shown are in MPa, displacements in mm and forces in N.

2.1 Data Generation

Data-driven constitutive modelling of materials requires large amounts of data to train the models. In this study, a large dataset was generated, where it was important to obtain deformation gradients representative of multiple deformations. Therefore, arbitrary deformation gradients were obtained as follows:

$$\mathbf{F} = \prod_{}^{m} ({}^{e}_{\theta}\mathbf{R}_m \; \mathbf{F}_m \; {}^{e}_{\theta}\mathbf{R}_m^{\mathrm{T}}), \quad \text{with} \begin{cases} m = \{uni, bi, ss\} \\ \theta \in [0°, 180°] \\ e = \{Ox, y, z\} \end{cases} \tag{1}$$

where $\mathbf{F}_m$ are the deformation gradients representative of three homogeneous loading cases: uniaxial load ($m = uni$), biaxial load ($m = bi$) and simple shear load ($m = ss$). For each value of m there is an associated rotation matrix ${}^{e}_{\theta}\mathbf{R}_m$, defined by an axis of rotation e and an angle θ.

To obtain the isochoric part of $\boldsymbol{\sigma}$ and $\mathbf{c}$ for each $\mathbf{F}$ generated, the strain energy functions of the materials and their respective material parameters were considered known. Only the independent components of each tensor were considered, i.e., 6 components for $\boldsymbol{\sigma}$ and 21 components for $\mathbf{c}$ due to the major symmetries existing in hyperelastic materials.

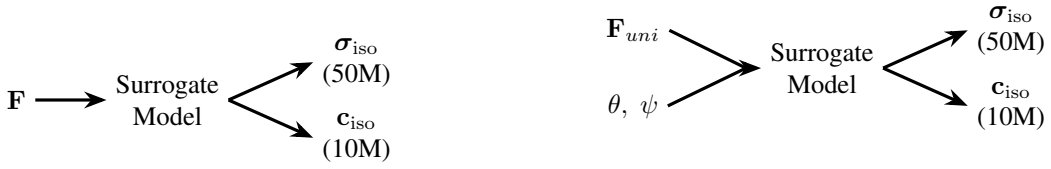
For the Neo-Hookean material model, eq. (1) was used to obtain arbitrary deformation gradients. The dataset had 50M pairs of $\mathbf{F}$ and $\boldsymbol{\sigma}$ and 10M pairs of $\mathbf{F}$ and $\mathbf{c}$. For the HGO material model, only uniaxial loading cases were considered to simplify the problem. In this material model, two angles were also obtained randomly for each $\mathbf{F}$: the azimuthal angle θ that can range from 0 to 2π rads and the polar angle ψ that can range from 0 to $\pi/2$ rads. The angles could range in increments of $\pi/4$ rads, and they define the preferred direction of the material in spherical coordinates, which can then be used to get the direction in Cartesian coordinates.

For the HGO material model, the dataset had 50M pairs of $(\mathbf{F}, \theta, \psi)$ and $\boldsymbol{\sigma}$ and 10M pairs of $(\mathbf{F}, \theta, \psi)$ and $\mathbf{c}$.

The generated data were then divided into three subsets: 80% for training, 10% for validation and 10% to test the model.

2.2 Surrogate Model Architecture

For the Neo-Hookean material model, the NN had 9 inputs that correspond to the 9 components of $\mathbf{F}$, whereas, for the HGO material model, the NN had 11 inputs, correspondent to the 9 components of $\mathbf{F}$ plus the two angles that define the preferred direction. The NN to predict the stress state had 6 outputs, correspondent to the 6 independent components of $\boldsymbol{\sigma}$ and the NN to predict the tangent stiffness matrix had 21 outputs, correspondent to the 21 independent components of $\mathbf{c}$. The architecture of the NNs is shown in Fig. 2.



(a) Neo-Hooke (b) HGO

Figure 2. NNs Architechture.

For these NNs to learn complex and non-linear relationships between inputs and outputs, it was necessary to use an activation function, namely the rectified linear function (ReLU), to introduce non-linearity into the network.

Some hyperparameters of the NNs were tuned in an attempt to search for the best performance possible. To perform the hyperparameter optimization, an automatic search optimization software called Optuna was used [15], which uses a variant of the Bayesian optimization called tree-structured Parzen [16]. For a cost-effective search, it was used a strategy, called pruning, to stop unpromising trials [15].

The hyperparameters tuned with Optuna were the number of hidden layers, the corresponding number of neurons and the batch size. To tune them, an optimization process was created to minimize an objective function based on the validation loss. The objective function takes the hyperparameters as inputs and, for each trial, the values for the hyperparameters are dynamically generated based on the `suggest` application programming interface (API).

The possible values for the hyperparameters, defined with the `suggest` API, are shown in Table 1, where the number of layers is an integer value chosen within a specified interval, the number of neurons and the batch size

are also integers, but chosen from a given list. The objective function runs every trial, and it gives the evaluation metric, which is the validation loss in this case.

Table 1. Hyperparameters and its possible values

Hyperparameter	Type	Values
Number of layers	Integer	[2,4]
Number of neurons	Categorical	4 or 8 or 16 or 32 or 64 or 128 or 256
Batch size	Categorical	16 or 32 or 64 or 128 or 256

In this analysis, only a study was carried out with 100 trials and applied to a random 1 % of the training set in order to speed up the tuning process. Then, with the values for these hyperparameters defined, the learning rate was defined. To tune the learning rate, a learning rate finder available in PyTorch Lightning was used. It does a small run with a learning rate that increases after each processed batch and then the corresponding loss is logged, which gives guidance for the optimal learning rate to use in training.

2.3 Training

The models only learn the desired behaviour after training the NN, where the weights and biases are adjusted to obtain the relation between the isochoric components of $\boldsymbol{\sigma}$ or $\mathbf{c}$ and $\mathbf{F}$ in the training data.

For the training, we used PyTorch Lightning, an open-source Python library that is a PyTorch wrapper and handles data loading, distributed training, and logging [17], and the adaptive moment estimation (ADAM) optimization [18] to adjust the parameters of the NN by minimizing the mean-squared error.

The training was taken in batches, where a sample of a predefined size was randomly sampled from the training data in each batch. After the completeness of this process for all batches, an epoch of the training stage is finished and the process is repeated for the following epochs. The maximum number of epochs was not defined, and the training process ended when no further decrease in the validation loss was observed.

3 Results and Discussion

3.1 Block with Pressure

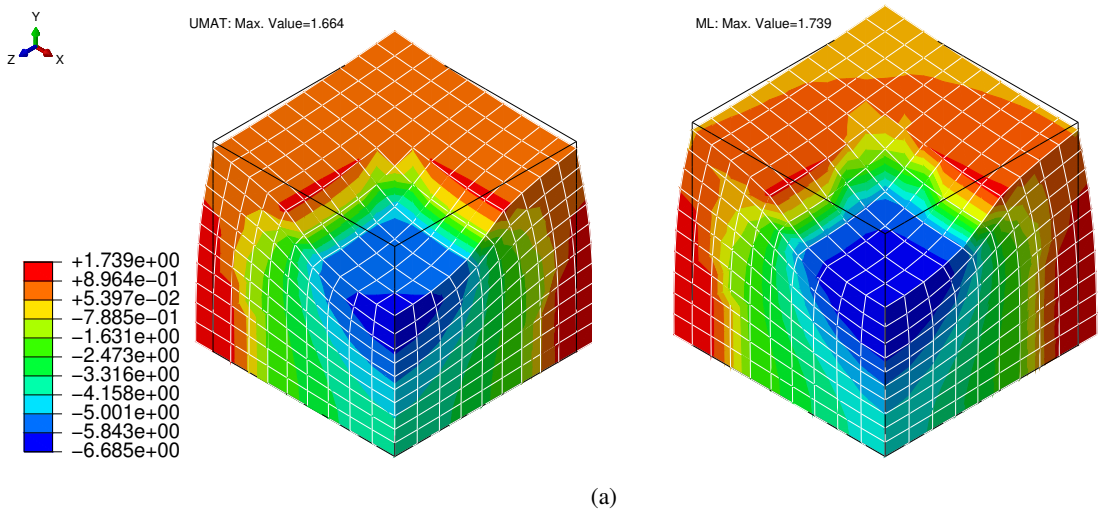
In this problem, a block is subjected to a distributed pressure load on part of its upper surface. The displacements are fixed in the x- and z-directions on the upper surface and fixed in the y-direction at the bottom. The symmetry of the block was considered, which implied some additional Dirichlet boundary conditions: fixed displacements in the x-direction due to symmetry in the x-plane and fixed displacements in the z-direction due to symmetry in z-plane. The block is loaded partially in the upper surface with a constant pressure load p, leading to a mixed Dirichlet-Neumann boundary condition [19]. A mesh of 10 x 10 x 10 hexahedral elements with 8 linear nodes was used, and a pressure load ($p = 6$) was applied.

The results obtained with the conventional approach (UMAT) and the proposed one (ML) are shown in Fig. 3. To compare the results, it was calculated the absolute and relative differences between all the components of the Cauchy stress tensor. The differences are displayed in Fig. 4 with a box-and-whisker plot to show the variation of the errors for all the integration points and for each time increment.

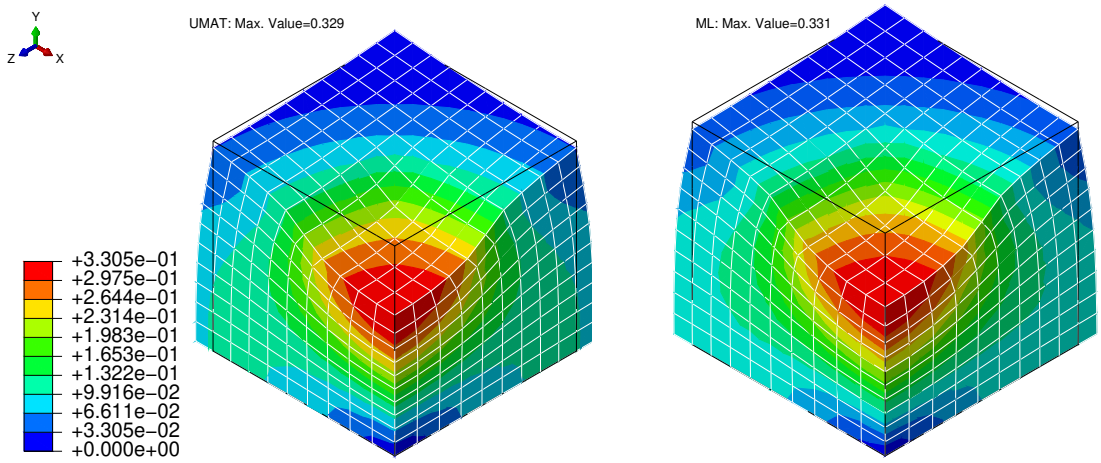
It is possible to observe that the contour plots shown have really similar colour maps, and the maximum value obtained with the two approaches has a small relative error. Additionally, it is also possible to see that the absolute difference between the displayed variables is also minimal for the most part of the block.

Regarding the box-and-whisker plots of relative error, it is notorious that there are some high values. A possible reason for such high values is that at the beginning of the analysis, the stress values are small, which means that even for a small absolute error, the relative error is high. This hypothesis is supported by the box-and-whisker plot of the absolute error that shows that, in fact, the absolute error has a small value and by the decreasing trend of the relative error as the number of increments increases.

Another relevant aspect to compare is the convergence of the problem with the two approaches. The proposed framework required 33 total iterations and 22 seconds, instead of 21 iterations and 8 seconds required with the conventional approach. Since the tangent stiffness matrix was obtained with a ML model, it was expected that the

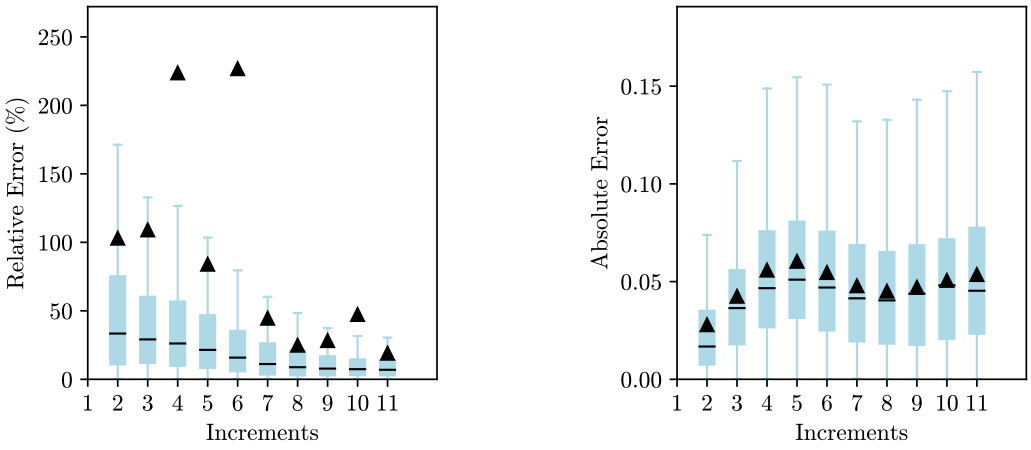


(a)



(b)

Figure 3. Deformed configurations and contour plots of (a) maximum principal stress and (b) displacement magnitude for the block under pressure. Conventional UMAT (left) and ML approach (right).



(a) (b)

Figure 4. Box-and-whisker plot for the (a) relative error and for the (b) absolute error between all the components of $\boldsymbol{\sigma}$ for all integration points. The box extends from the first to the third quartile, the black line represents the median, the whiskers extend from the edges of the box and show the range of the data and the black arrow represents the average.

proposed framework would require more iterations and more CPU time when compared to an analysis that used the analytically calculated tangent stiffness matrix.

3.2 One Element Anisotropic Cube

In this example, we considered a cube with one hexahedral element with 8 linear nodes, stretched in the x direction with different fibre orientations. Different directions for the fibres were considered for a tensile load, and an example with the fibres in the x-direction but subjected to compression was also analysed. The results are shown in Fig. 5.

The HGO material model is much more complex than the Neo-Hookean model because, with a different fibre direction, the material behaves completely differently. Additionally, the HGO model states that if the fibres are in compression, they do not contribute to the strain energy function and the material is regarded as an isotropic material, which is another complexity that the NN was able to learn, as it was proved with the uniaxial compressive load and for the fibres in the y-direction.

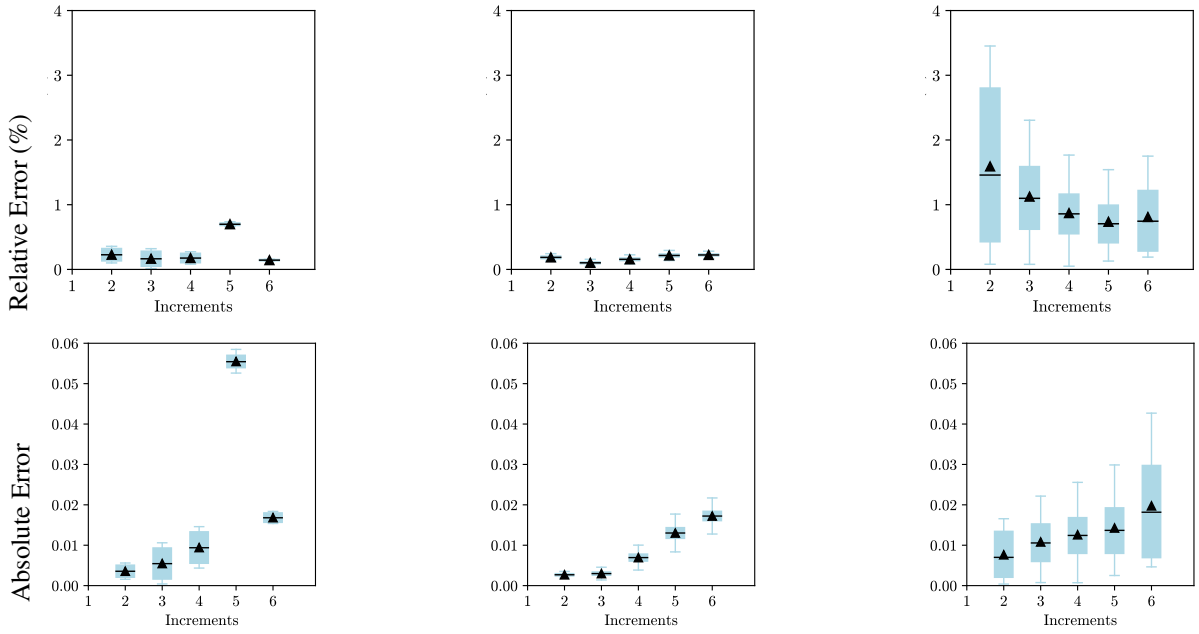


Figure 5. Box-and-whisker plot for the relative and absolute errors between all the components of $\boldsymbol{\sigma}$ for all integration points with fibres in (a) x-direction, (b) y-direction and (c) x-direction (compression). The box extends from the first to the third quartile, the black line represents the median, the whiskers extend from the edges of the box and show the range of the data and the black arrow represents the average.

4 Conclusions

Data-driven constitutive modelling is gaining more interest in recent years, and there are already some different approaches proposed in the literature. However, most of them use the invariants of the Cauchy-Green deformation tensors as inputs of the NN and the strain energy and its derivatives as outputs. These solutions provide more flexibility but still require some mathematical treatment to compute the stress and the tangent stiffness matrix required by the UMAT. In this work, the UMAT has a model to predict the stresses and a model to predict the spatial elasticity tensor, and there is no need to express the mathematical equations that describe the different models used. Only the volumetric part is calculated with mathematical equations because they depend on the boundary conditions of the problem in analysis. Therefore, in this work, there is no need to derive equations to describe the behaviour of the material in analysis and it is only necessary to train a surrogate model and pass the parameters of the trained model to Fortran.

In general, the numerical examples analysed showed that the ML-based approach was able to model the constitutive behaviour of hyperelastic materials since the results obtained were similar to the ones obtained with a

conventional UMAT. The analyses took longer to converge, which was expected. However, all the analyses were completed, requiring only more iterations and time to do so when compared with a standard UMAT.

For future work, it would be interesting to consider other material models and test this framework with more complex examples. It would also be interesting to consider different inputs and outputs for the NNs.

Authorship statement. The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work, and that all material that has been herein included as part of the present paper is either the property (and authorship) of the authors, or has the permission of the owners to be included here.

References

[1] K. A. Kalina, L. Linden, J. Brummund, P. Metsch, and M. Kästner. Automated constitutive modeling of isotropic hyperelasticity based on artificial neural networks. *Computational Mechanics*, vol. 69, n. 1, pp. 213–232, 2022.

[2] M. Liu, L. Liang, and W. Sun. A generic physics-informed neural network-based constitutive model for soft biological tissues. *Computer Methods in Applied Mechanics and Engineering*, vol. 372, pp. 113402, 2020.

[3] S. S. Sajjadinia, B. Carpentieri, D. Shriram, and G. A. Holzapfel. Multi-fidelity surrogate modeling through hybrid machine learning for biomechanical and finite element analysis of soft tissues. *Computers in Biology and Medicine*, vol. 148, pp. 105699, 2022.

[4] J. Humphrey and S. L. O'Rourke. *An Introduction to Biomechanics: Solids and Fluids, Analysis and Design*. Springer New York, NY, 2015.

[5] J. Ghaboussi, J. H. Garrett, and X. Wu. Knowledge-based modeling of material behavior with neural networks. *Journal of Engineering Mechanics*, vol. 117, n. 1, pp. 132–153, 1991.

[6] X. Wu, J. G. J. H, and J. Ghaboussi. Representation of material behavior: neural network-based models. *1990 IJCNN International Joint Conference on Neural Networks*, vol. 1, pp. 229–234, 1990.

[7] Y. M. A. Hashash, S. Jung, and J. Ghaboussi. Numerical implementation of a neural network based material model in finite element analysis. *International Journal for Numerical Methods in Engineering*, vol. 59, n. 7, pp. 989–1005, 2004.

[8] K. Linka, M. Hillgärtner, K. P. Abdolazizi, R. C. Aydin, M. Itskov, and C. J. Cyron. Constitutive artificial neural networks: A fast and general approach to predictive data-driven constitutive modeling by deep learning. *Journal of Computational Physics*, vol. 429, pp. 110010, 2021.

[9] H. Dal, F. A. Denli, A. K. Açan, and M. Kaliske. Data-driven hyperelasticity, part i: A canonical isotropic formulation for rubberlike materials. *Journal of the Mechanics and Physics of Solids*, vol. 179, pp. 105381, 2023.

[10] F. As'ad, P. Avery, and C. Farhat. A mechanics-informed artificial neural network approach in data-driven constitutive modeling. *International Journal for Numerical Methods in Engineering*, vol. 123, n. 12, pp. 2738–2759, 2022.

[11] V. Tac, K. Linka, F. Sahli-Costabal, E. Kuhl, and A. B. Tepole. Benchmarks for physics-informed data-driven hyperelasticity, 2023.

[12] K. Linka and E. Kuhl. A new family of constitutive artificial neural networks towards automated model discovery. *Computer Methods in Applied Mechanics and Engineering*, vol. 403, pp. 115731, 2023.

[13] V. Tac, V. D. Sree, M. K. Rausch, and A. B. Tepole. Data-driven modeling of the mechanical behavior of anisotropic soft biological tissue. *Engineering with Computers*, vol. 38, n. 5, pp. 4167–4182, 2022.

[14] J. N. Fuhg, N. Bouklas, and R. E. Jones. Learning hyperelastic anisotropy from data via a tensor basis neural network. *Journal of the Mechanics and Physics of Solids*, vol. 168, pp. 105022, 2022.

[15] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. Optuna: A next-generation hyperparameter optimization framework, 2019.

[16] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. *Advances in Neural Information Processing Systems*, vol. 24, pp. 2546 – 2554, 2011.

[17] W. Falcon and T. P. L. team. Pytorch lightning, 2023.

[18] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017.

[19] J. Schröder, T. Wick, S. Reese, P. Wriggers, R. Müller, S. Kollmannsberger, M. Kästner, A. Schwarz, M. Igelbüscher, N. Viebahn, H. R. Bayat, S. Wulfinghoff, K. Mang, E. Rank, T. Bog, D. D'Angella, M. Elhaddad, P. Hennig, A. Düster, W. Garhuom, S. Hubrich, M. Walloth, W. Wollner, C. Kuhn, and T. Heister. A selection of benchmark problems in solid mechanics and applied mathematics. *Archives of Computational Methods in Engineering*, vol. 28, n. 2, pp. 713–751, 2021.