Asociación Argentina
de Mecánica Computacional

Mecánica Computacional Vol XL, págs. 681-687 (artículo completo)
F.A. Avid, L.C. Bessone, P. Gamazo, J.J. Penco, M.A. Pucheta, M.A. Storti (Eds.)
Concordia, 6-9 Noviembre 2023

AUTOMATING A FEM SOLUTION DATABASE GENERATION AND NEURAL NETWORK LEARNING FOR SOLID MECHANICS PROBLEMS

L.C. Agorio[a], M.C. Vanzulli[b], B. Bazzano[c] and Jorge M. Pérez Zerpa[c]

[a]*Instituto de Ingeniería Eléctrica, Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay*

[b]*Instituto de Ingeniería Mecánica y Producción Industrial, Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay*

[c]*Instituto de Estructuras y Transporte, Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay*

Keywords: Artificial Neural Networks, Finite Element Method, Solid Mechanics, Surrogate-models.

Abstract. Solving finite element problems can be computationally expensive, particularly in nonlinear solid mechanics. This challenge emerges in applications such as biomechanics or manufacturing process design, where the same problems may need to be solved in real time for different configurations or input data. In this paper, we combine Finite Element Method (FEM) and Artificial Neural Networks (ANN) to improve the speed and efficiency of solvers in solid mechanics problems. We developed a pipeline to generate databases of FEM solutions by interacting with an in-house Open Source Software for non-linear Analysis of Structures (ONSAS). We used these databases to train a neural network that takes geometrical and material properties as inputs, and as an output, predicts the displacements solution of the mechanical problems. Our experiments showed that the proposed approach was effective, achieving low losses on both the training and test datasets. We present validation examples where the ANN was capable of matching the analytic solutions with great accuracy.Moreover, we solved more complex problems with different geometries, boundary conditions, and materials, considering large strain deformations. One advantage of our implementation is its simplicity and scalability, allowing us to easily solve a wide range of mechanical problems.Additionally, the use of ANN provides faster computation times than the traditional solvers using the FEM method. Although this study presents promising results, we also discuss the limitations of the proposed approach and potential directions for future work.

1 INTRODUCTION

In this article, we show the results obtained using a surrogate model for simple solid mechanics problems. Our main goal is to conclude whether this approximate model can provide precise solutions using less computation time than the Finite Element Method (FEM) (Martínez-Martínez et al., 2017; Böhringer et al., 2023). To achieve this, we trained a neural network (NN) using a dataset of FEM solutions for compression/extension mechanical problems.

There is an increasing interest in using these types of models in a wide range of applications, such as in biological tissues modeling (Pellicer-Valero et al., 2020). In this work, we repeatedly used a FEM solver to generate a large dataset of results, which was used to train the NN. The NN was designed to learn from the FEM solutions and predict the displacement field at particular nodes (Yang et al., 2022). Surrogate models can be used for structural assessment by tracking geometric features along time (Zhu et al., 2023) and for solving material identification problems (Steuben et al., 2015). Moreover, existing methods such as Physics-Informed Neural Networks leverage the underlying dynamics to train the model (Raissi et al., 2017).

In this work, we present the methodology used to develop the NN, the performance results of the model compared to the FEM, and potential future directions. By developing faster surrogate models, our research line has the potential to significantly improve the efficiency of solving mechanical problems. This has broader implications for the engineering design process.

2 METHODOLOGY

In this section, we describe the pipeline we developed to generate the dataset of FEM solutions and train the NN. We used a simple solid mechanical models with the `ONSAS.m` FEM solver, which we automated with a bash script. The resulting dataset was preprocessed and used to train a multi-layer perceptron (MLP) implemented in PyTorch.

2.1 Uniaxial Load Model

This model consisted of a uniaxial mechanical problem in which a load was applied to a three-dimensional prism, causing it to deform in three dimensional. The model is uniaxial in the sense that the applied load is along the x-axis. We used the `ONSAS.m` FEM solver to simulate the deformation and generate a dataset of solutions. We chose for its simplicity and as a starting point for more complex models.

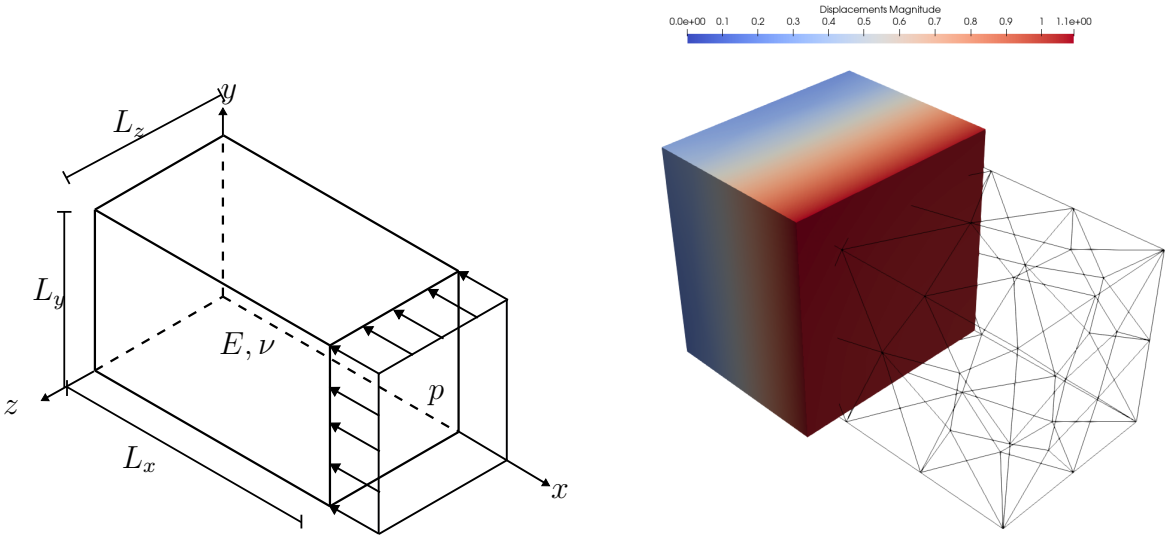
The simulated uniaxial problem can be seen in Figure 1. Loads are applied to the prism at the right end, causing it to deform. The prism is constrained to maintain the lefmost face (Figure 1) in the y-z plane, and the origin fixed, deforming itself in the three directions due to the uniaxial load.

2.2 Composed Cantilever Model

The composed cantilever model consists of two prisms made of different materials, denoted as 1 and 2, for which the elastic moduli and Poisson ratios are E_1, ν_1 and E_2, ν_2, respectively. Prism 1 is fixed by its face at the $x = 0$ plane. A load of $(-p_x, t_y, t_z)$ is applied at the plane located at $x = Lx$ along the x, y, and z axes, respectively as shown in Figure 2.

2.3 Bash Script

To automate the FEM solver, we developed a bash script that generated the input files for the `ONSAS.m` (Pérez Zerpa et al., 2018) solver and executed it repeatedly to generate a large



(a) Reference configuration sketch.

(b) System with an applied load.

Figure 1: Uniaxial load problem simulated and solved in `ONSAS.m`.

dataset of solutions. The script also handled the output files, extracting the relevant data and formatting into a `.csv` it for further processing.

The variables swept by the script were the length of the prism (Lx, ranging from 1 to 3), the applied load (p, ranging from 0.1 to 3), and the elastic modulus of the material (E, ranging from 1 to 4). Ly and Lz were fixed at 1, and the Poisson ratio (ν) was fixed at 0.3. We processed the solver's output to obtain the displacement of the center point of the prism's face in the three dimensions. The script generated a total of 18,900 data points, which we used to train the NN.

For the composed cantilever model, we swept the variables including the length of the prism (Lx, ranging from 1 to 2), the applied load (p, ranging from 0.05 to 0.14), and the elastic moduli of each material (E_1 and E_2, ranging from 1.4 to 2.4). Fixed values were $Ly = 1$, $Lz = 0.5$, and $\nu_1 = \nu_2 = 0.3$. We processed the output of the solver to obtain the displacement of the center point of the prism's face in the three dimensions. The script generated a total of 13,310 data points, which were used to train the NN.

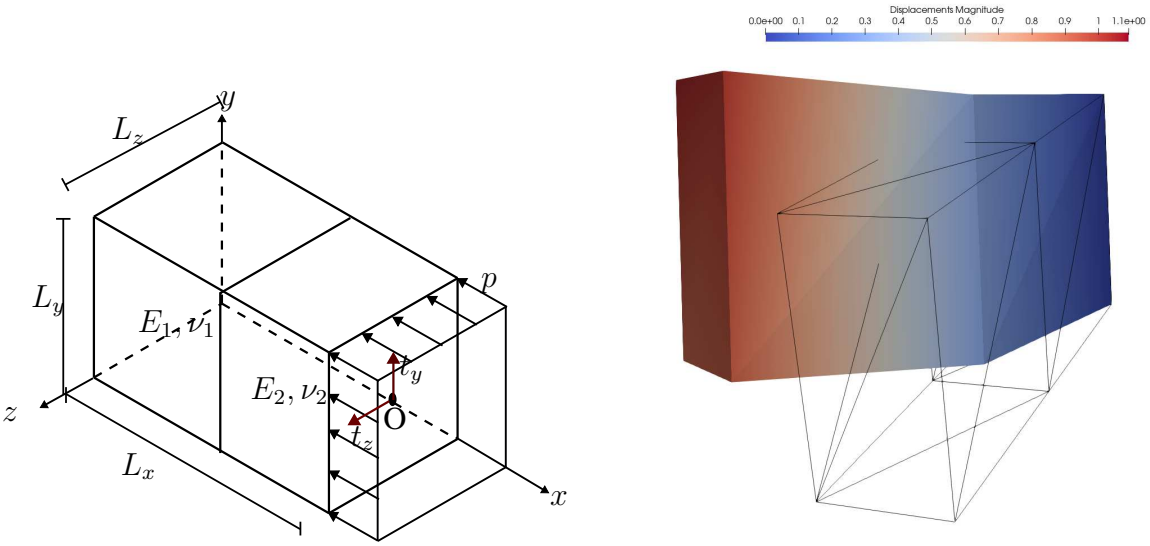
2.4 Neural Network

We implemented an MLP in PyTorch to learn from the FEM solutions and solve similar mechanical problems more quickly. The MLP has two hidden layers with 20 and 10 neurons, respectively, and a ReLU activation function. The output layer has three neurons, corresponding to the three dimensions of the displacement of the center point of the prism's face. We used Adam optimizer with a learning rate of 0.001 through 100 epochs.

In the uniaxial case, we used a 50/50 split for the training and validation sets, and 200 test samples using the analytic solution. For the composed cantilever case, we used 1000 samples for both the training and validation sets.

We computed two different loss functions: the mean squared error (MSE) and the relative MSE. The MSE is the most common loss function for regression problems and is defined as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{1}$$



(a) Reference configuration sketch.

(b) System with an applied load.

Figure 2: Composed cantilever problem solved with `ONSAS.m`.

where y_i is the true value and $\hat{y}_i$ is the predicted value. The relative MSE is defined as:

$$RMSE = \frac{1}{n}\sum_{i=1}^{n}\frac{||y_i - \hat{y}_i||_2^2}{||y_i||_2^2} \tag{2}$$

The relative MSE is more useful for comparing the performance of different models, since it is normalized by the true value.

2.5 Analytic Solution

To compare the performance of our NN to the FEM, we computed the analytic solution to the uniaxial problem. The analytic solution is given by:

$$\boldsymbol{u} = (\alpha - 1)L_x\boldsymbol{e}_1 + (\beta - 1)\frac{L_y}{2}\boldsymbol{e}_2 + (\beta - 1)\frac{L_z}{2}\boldsymbol{e}_3 \tag{3}$$

where $\boldsymbol{u}$ is the displacement of the center point of the prism's face, L_x, L_y and L_z are the dimensions of the solid as it is shown in Figure 2a. We obtained the values of α and β by solving the following pair of non-linear equations:

$$\alpha\left(\mu - \frac{\mu}{\alpha^2} + \frac{K\beta^2}{\alpha}(\beta^2\alpha - 1)\right) = -p \tag{4}$$

$$\beta\left(\mu - \frac{\mu}{\beta^2} + K(\alpha^2\beta^2 - \alpha)\right) = 0 \tag{5}$$

where p is the applied load, $K = \frac{E}{3(1-2\nu)}$ is the Bulk modulus and $\mu = \frac{E}{2(1+\nu)}$ is the second Lamé parameter, E is the elastic modulus and ν is the Poission's ratio.

We used the analytic solution to generate a dataset, in order to compare the performance of the neural network to the FEM. For doing so we used lhs to generate a Latin Hypercube Sampling (Stein, 1987) of the input variables, and then computed the analytic solution for each of the 200 points in the sample.

3 RESULTS

In this section, we present the results of our experiments. We compared the performance of our neural network to the FEM, and evaluated its effectiveness on a uniaxial problem and a composed cantilever problem. All the scripts used to generate the results are publicly available in the corresponding GitHub repository [1].

3.1 Evaluation for Uniaxial Compression

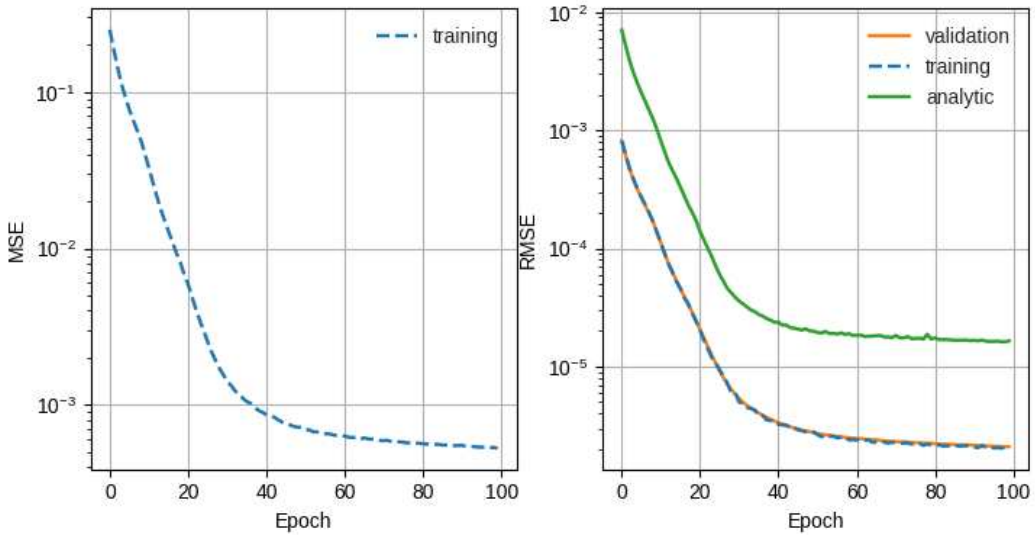


Figure 3: Training, validation and test loss of the neural network on the uniaxial compression/extension dataset.

Figure 3 shows the training, validation, and test losses of our neural network on the uniaxial compression/extension dataset. The final training loss for MSE is 5.29×10^{-4} and the RMSE is 2×10^{-6}. The final validation loss RMSE is 2×10^{-6}, and the final analytic loss RMSE is 1.7×10^{-5}.

We also evaluated the ability of our model to extrapolate beyond the parameters in the training set. When we measured the RMSE extrapolation error for $(Lx, E, p) = (4.00, 0.50, 3.50)$, we found it to be 6.7×10^{-2}, while the MSE extrapolation error was $9.5 \times 10^{-1}\ m^2$. Since the variation range for the training parameters was $(Lx, E, p) = (1 \ldots 3, 1 \ldots 4, .1 \ldots 3)$, these results show that our neural network was able to extrapolate to a point that was not in the training set, with an error of 6.7 % of the true value.

3.2 Evaluation on Cantilever Model

For the case of the composed cantilever model, the training, validation and test loss is shown in Figure 4. As we can see, the training loss decreases rapidly during the first few epochs and then levels off.

Figure 4 shows the training, validation, and test losses of our neural network on the cantilever model. The final training loss for MSE is 2.3×10^{-3} and the RMSE is 2.4×10^{-4}. The final validation loss RMSE is 2.5×10^{-4}, and the final analytic loss RMSE is 2.9×10^{-4}.

[1] https://github.com/leopoldoagorio/solid-mechanics-ML

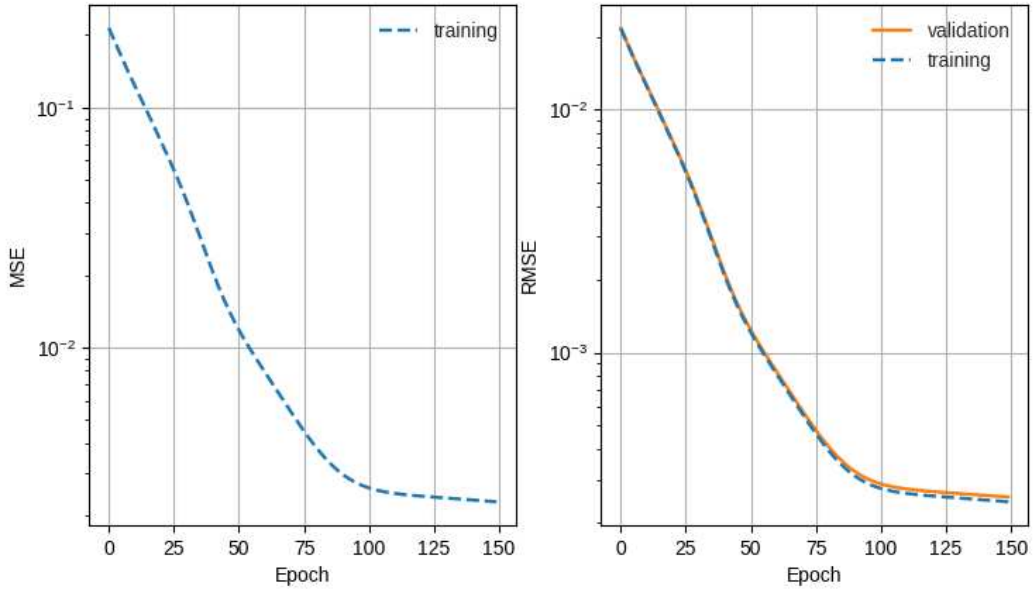


Figure 4: Training, validation and test loss of the neural network on the composed cantilever dataset.

We also evaluated the ability of our model to extrapolate beyond the parameters in the training set. When we measured the RMSE extrapolation error for $(Lx, E1, E2, p) = (1.50, 1.3, 1.3, 0.10)$, we found it to be 0.21. Since the variation range for the training parameters was $(Lx, E1, E2, p) = (1 \ldots 2, 1.4 \ldots 2.4, 1.4 \ldots 2.4, 0.05 \ldots 0.14)$, these results show that our neural network was able to extrapolate to a point that was not in the training set, with an error of 21% of the true value.

4 CONCLUSION

In this work, we presented a pipeline for generating a dataset of FEM solutions for simple solid mechanical problems. The dataset was used to train a neural network to predict displacements at control nodes. The results showed that the proposed approach was effective, with the neural network achieving low losses on both the training and test datasets. The network closely matched the theoretical curve for uniaxial compression, demonstrating the accuracy of the predictions. Additionally, the evaluation on a cantilever model indicated the potential for extending this approach to more complex mechanical problems.

One advantage of this implementation is its simplicity and scalability. By starting with a relatively simple mechanical problem, we were able to develop a pipeline that can be easily scaled to handle more complex problems. Furthermore, the use of a neural network allows for faster computation times compared to traditional FEM methods. This has the potential to significantly reduce computational costs in numerical processes that rely on FEM simulations, such as material identification algorithms for tissue disease diagnostics and stress CAD for manufacturing. However, it should be noted that predictions for material properties outside the training space may lead to higher error rates, as shown in the results.

In conclusion, we have demonstrated the effectiveness of using a neural network to solve simple mechanical problems. By developing a faster surrogate model, we have shown the potential

to significantly improve the efficiency of solving such problems, with broader implications for the engineering design process. Future work includes scaling the proposed approach to more complex mechanical problems and evaluating its effectiveness on a larger dataset.

REFERENCES

Böhringer P., Sommer D., Haase T., Barteczko M., Sprave J., Stoll M., Karadogan C., Koch D., Middendorf P., and Liewald M. A strategy to train machine learning material models for finite element simulations on data acquirable from physical experiments. *Computer Methods in Applied Mechanics and Engineering*, 406:115894, 2023.

Martínez-Martínez F., Rupérez-Moreno M.J., Martínez-Sober M., Solves-Llorens J.A., Lorente D., Serrano-López A., Martínez-Sanchis S., Monserrat C., and Martín-Guerrero J.D. A finite element-based machine learning approach for modeling the mechanical behavior of the breast tissues under compression in real-time. *Computers in biology and medicine*, 90:116–124, 2017.

Pellicer-Valero O.J., Rupérez M.J., Martínez-Sanchis S., and Martín-Guerrero J.D. Real-time biomechanical modeling of the liver using machine learning models trained on finite element method simulations. *Expert Systems with Applications*, 143:113083, 2020.

Pérez Zerpa J.M., Vanzulli Pena M.C., Villié A., Bazzano B.J., Viera Sosa J., Forets M., and Battini J.M. ONSAS.m: an Open Nonlinear Structural Analysis Solver for GNU-Octave/Matlab. 2018.

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

Stein M. Large sample properties of simulations using latin hypercube sampling. *Technometrics*, 29(2):143–151, 1987.

Steuben J., Michopoulos J., Iliopoulos A., and Turner C. Inverse characterization of composite materials via surrogate modeling. *Composite Structures*, 132:694–708, 2015.

Yang R., Wang S., Wu X., Liu T., and Liu X. Using lightweight convolutional neural network to track vibration displacement in rotating body video. *Mechanical Systems and Signal Processing*, 177:109137, 2022. ISSN 0888-3270. doi:https://doi.org/10.1016/j.ymssp.2022.109137.

Zhu Y., Wang S., Liu T., Liu C., and Liu X. A visual measurement method of structural body vibration displacement combined with image deblurring. *Measurement*, 211:112598, 2023.