Neural Network–Based Surrogate Models Applied to Fluid–Structure Interaction Problems

Conference Paper · June 2022

DOI: 10.23967/wccm-apcom.2022.080

4 authors:

Daniel Andrés Arcones
Bundesanstalt für Materialforschung und -prüfung
1 PUBLICATION **0** CITATIONS
SEE PROFILE

Rishith Ellath Meethal
Technische Universität München
10 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Birgit Obst
Siemens
16 PUBLICATIONS **50** CITATIONS
SEE PROFILE

Roland Wüchner
Technische Universität Braunschweig
216 PUBLICATIONS **5,299** CITATIONS
SEE PROFILE

15th World Congress on Computational Mechanics (WCCM-XV)
8th Asian Pacific Congress on Computational Mechanics (APCOM-VIII)
Virtual Congress: 31 July – 5 August 2022
S. Koshizuka (Ed.)

NEURAL NETWORK-BASED SURROGATE MODELS APPLIED TO FLUID-STRUCTURE INTERACTION PROBLEMS

DANIEL ANDRÉS ARCONES[1], RISHITH E. MEETHAL[2], BIRGIT OBST[2] AND ROLAND WÜCHNER[1]

[1] Institute of Structural Analysis, Technische Universität Braunschweig
Beethovenstraße 51, 38106 Braunschweig, Germany
{daniel.andres, r.wuechner}@tu-braunschweig.de, https://www.tu-braunschweig.de/isd

[2] Technology, Siemens AG
Otto-Hahn-Ring 6, 81739 Munich, Germany
{rishith.ellath_meethal, birgit.obst}@siemens.com

Key words: Data-driven Surrogate Models, Co-Simulation, Neural Networks, Fluid-Structure Interaction

Abstract. *Traditional computational methods face significant challenges with ever-increasing complexity in the problems of engineering interest. One category of problems that suffer from this phenomenon is those where Fluid-Structure Interaction (FSI) is present. One set of problems that suffer from this phenomenon is those where Fluid-Structure Interaction (FSI) is present. FSI simulations are traditionally time-consuming and computationally extremely expensive. Potential alternatives rely on using a surrogate model to substitute one or more systems involved. A promising approach employs artificial neural networks as the basis for such a surrogate model combined with strong physics simulations based on finite element methods (FEM). This approach requires the seamless integration of AI algorithms and packages into the simulation workflow. Such an example is the NeuralNetworkApplication developed in KratosMultiphysics. The routines related to the neural networks are executed through an interface with the Keras API. Mok's benchmark is chosen as the study case to test the capacity of the previous method applied to FSI problems. Two cases in which one of the systems is substituted by a neural network-based surrogate model are analyzed. Strong and weak coupling scenarios are considered. The results present improvements in simulation time without sacrificing accuracy, especially when compared with the original benchmark. This contribution discusses the influence of the original data and network architecture on the simulation outcome and different considerations for generating surrogate models for FSI.*

1 INTRODUCTION

Nowadays, the interest in coupled multiphysics simulations is higher than ever. The Fluid-Structure Interaction (FSI) problems appear in a broad range of key technologies, from aircraft to wind turbines or bridges. Traditional approaches for analyzing and designing such systems are greatly limited by their complexity, often relying on physical tests. Recent advances in computation have allowed the solution of these problems through simulations, creating new possibilities for faster and cheaper design. Nevertheless, coupled multiphysics problems still impose some of the most significant challenges when simulating engineering systems [1]. Therefore, extensive efforts have been put into developing methods that alleviate the problems from multiphysics simulations, such as the convergence of the coupling schemes and

their computational performance.

FSI simulations are traditionally time-consuming and computationally extremely expensive. Additionally, there are convergence or stability problems in many cases. Plenty of research has been devoted to accelerating and enhancing their performance [2]. However, there is usually a trade-off between speed and accuracy, which may limit their use in specific applications. One attractive prospect solution for some of these limitations is the implementation of surrogate models that substitute or complement any of the subsystems. We propose here the use of neural networks to generate such surrogate models.

Neural networks (NN) make up a subfield of supervised machine learning algorithms that have experienced great success in recent years due to their efficiency and versatility. Being a data-driven approach, they avoid the necessity of a complex physical model, allowing predictions of results based on observations. These properties make them promising candidates as the basis of surrogate models. Neural networks were first proposed in the 40s [3] and began to get traction with the discovery of the back-propagation algorithm in the 70s [4]. However, it was not until the 90s that their first applications in structural engineering were considered. At the start, Valunchene and Sun [5] trained neural networks on simple benchmarks such as prismatic beams, simple truss structures, and plates. In the following years, the applications in structural engineering expanded [6, 7], but their implementation was not generalized due to the lack of computational access. The most significant advances were made in the field of material and model parametrization from real data [8].

In parallel, there has been extensive research in the field of fluid mechanics due to the potential of avoiding expensive simulations. However, it was not until the last years that the advances in machine learning and the computational power of modern computers led to an explosion of research in the field. For example, in fluid mechanics, simulation results are being predicted from visual data [9, 10]. In solid mechanics, neural networks have been extensively used for applications like structural optimization [11]. The recently implemented Physics-Informed Neural Networks (PINN) [12, 13] is providing promising results in simulating physical systems with the help of neural networks by integrating governing equations in the training loop. The physics guided machine learning models like the one's introduced by Rishith et al. in [14, 15] are also getting attention as different versions of informed machine learning. Additionally, as shown in [16], different solutions based on neural networks have been proposed for a variety of dynamical systems.

However, up to now, there have been very few applications of neural networks to coupled FSI simulations. Totounferoush et al. [17] recently proposed the use of predictors based on neural networks as surrogate models in FSI simulations, where both the fluid and the structure are substituted. Other surrogate models of structures for FSI problems were additionally presented in [18].

The main contribution of this work is the analysis of neural network-based surrogate models to FSI problems. To this end, Mok's [19] benchmark is evaluated. Mok's benchmark is a well-known study case to test the capacity of different methods applied to FSI problems. In the first instance, the structural model in the example is substituted by a neural network-based surrogate model trained on known data while retaining the original fluid model. A second case where a surrogate substitutes the fluid model, keeping the structural one untouched, is evaluated. In both cases, the neural network predicts the total response of the system it substitutes. Strong and weak coupling scenarios are considered. The implementation is done in the multiphysics simulation framework KratosMultiphysics [20] (a.k.a Kratos), which uses Keras API [21] as a background for machine learning operations. In the following Section 2, the methodology for the implementation is detailed. Then, Section 3 tests the surrogate models on Mok's benchmark and discusses the results, concluding in Section 4.

2 METHODOLOGY

The approach for the generation of the neural network-based surrogate models consists in three steps: data generation, model training and surrogate model implementation (see Figure 1). The inclusion of NeuralNetworkApplication in Kratos' framework allows the integration of each of these steps in the traditional workflow of FSI simulations.

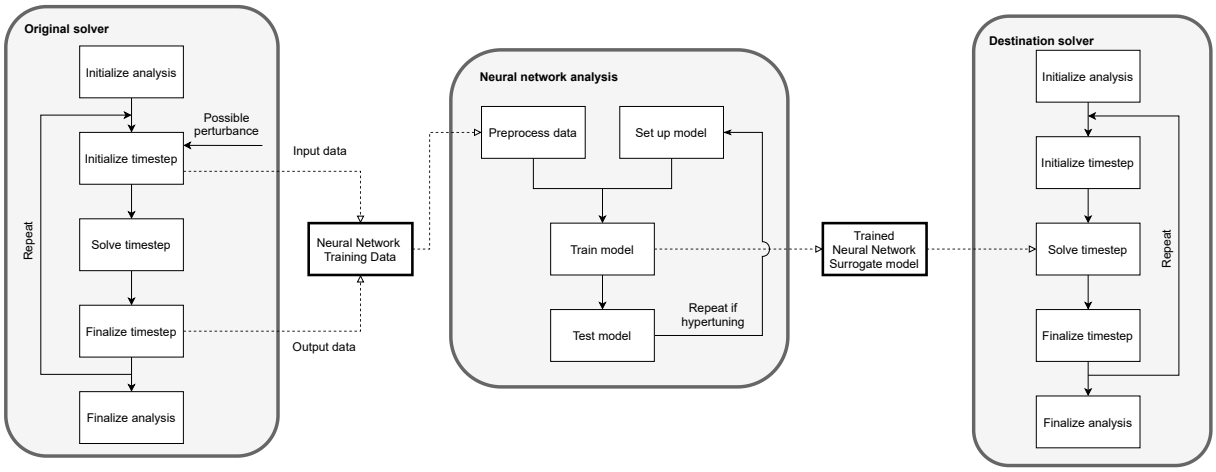

Figure 1: NeuralNetworkApplication's workflow.

2.1 Data generation

The first step in the development of a supervised surrogate model is the generation of the training datasets. These datasets can be obtained either from external sources (e.g. sensors) or generated from computational models. Taking advantage of the capacities of NeuralNetworkApplication, the second approach is chosen for this project. In this case, the input and output data are generated from the original benchmark, which employs an FSI simulation under strong coupling [22]. The results at the nodes of the model are saved at each timestep of the FEM simulation in the "model part". The model part defined in Kratos is a container for the geometrical information of the model, including the mesh and the variables present at the nodes. This data is later retrieved from the model part and stored separately, distinguishing between input and output data. It will be preprocessed and later fed to the neural network.

2.2 Model training

Once the training datasets are generated, the next step is the set-up and training of the model, whose architecture must be decided.

2.2.1 Neural network architecture

There exist numerous amount of options for the neural network's architecture. In this specific case, several requirements must be met. First, the neural network must be able to predict temporal data sequences, as the problem is a dynamic one. Additionally, the tendency between timesteps must be conserved, as the system is physical and conserves some inertia. Moreover, the results of adjacent nodes must also

be correlated, as the continuity of the model prevents discontinuities in the solution fields. Finally, the surrogate model will solve a regression problem; therefore, the output must be adaptable to the objective solution space. In conclusion, the neural network architecture must preserve the spatial and temporal properties of the system, proposing a solution for a given set of inputs.

The proposed architecture that fulfills these requirements is composed of convolutional and recurrent layers with a linear dense output. Convolutional Neural Networks (CNN) [23] use an assortment of weighted filters applied over ordered tensor grids. The training is performed on the weights of the filters, which in turn capture the relationship between different elements. As convolutional layers increase the dimensionality of the tensors, a pooling layer must be added at their output to collapse them. Two pooling strategies were considered: max-pooling and average-pooling. Max-pooling layers favor the extreme values, while average-pooling ones disfavor them. Average-pooling is chosen for our cases as it typically results in more stable models. The CNN block keeps the spatial information of the model.

Alternatively, the recurrent layers are used for recording temporal information. Long-Short Term Memory (LSTM) [24] layers are employed in this work. These layers receive the output of a selected number of previous timesteps as input and generate the next value. The number of timesteps considered is referred to as lookback. Combining convolutional with LSTM layers in a CNNLSTM architecture allows contemplating both the temporal and spatial relations of the model. Immediately after these two blocks, a final block composed of dense layers with ReLU activation functions is added to provide the model with generalization capabilities. A last dense layer with linear activation and the same dimension as the output objective space predicts the output for regression. The precise architectures for each case are specified in their respective sections.

2.2.2 Training

The training scheme consists of dataset pre-processing, prediction, loss evaluation, and backpropagation. As pre-processing, input and output data are normalized, and their constant values are suppressed from the dataset. Additionally, the output of previous timesteps is combined with the input to make up the lookback. This dataset is passed through the model that predicts the output. The output is compared with the expected actual value, and a mean-squared error (MSE) loss is evaluated. Then, the weights are updated through backpropagation using an Adam optimizer with a learning rate of 0.001 and decay of 1E-5. This process is repeated until convergence. This model is trained on a single temporal series close to the expected one from the coupled problem; therefore, setting a testing procedure is unnecessary. In addition, finer tuning of the hyperparameters could be achieved, but it is not consisted in this project.

2.3 Surrogate model implementation

Once the neural network is trained, it can be used as a surrogate model. This model will substitute either the structure or the fluid system in the coupled co-simulation problem. The coupling can follow weak or strong schemes. Through the NeuralNetworkApplication solver, the model is seamlessly integrated into the solution loop. First, the input variables are retrieved from the model part. Next, they are preprocessed with the same parameters as used for the model's training and passed to the neural network. The network then predicts the output results, which are further processed to invert the transformations performed in the training phase to the output dataset. Finally, the predicted results are newly assigned to the model part. The respective coupling and mapping operations are performed before and after the predictions.

3 RESULTS

The neural network-based surrogate model can be applied to Mok's benchmark in several ways. In this project, two configurations are implemented: one with the surrogate model substituting the structural one and one substituting the fluid one. Both models differ mainly on the input and output variables, with some modifications in the neural network architecture.

3.1 Original benchmark

Mok's [19] benchmark is composed of a flexible wall in a channel flow, which is fixed at its base and is displaced by the force of the flow. The fluid is modeled using an ALE formulation, where the displacement of the membrane is reflected in the fluid as a modification of its boundary conditions. This mesh displacement has an effect on the velocity and pressure results of the fluid, especially near the interface. This is a 2D FSI simulation between the convergent flow and the flexible wall. The problem is strongly coupled, as the similar order of magnitude of the densities of structure and fluid implies large interactions between them. The first reference solutions are provided by Mok (2001) [19] and are further refined by Valdés (2007) [22]. Kratos' solution is closer to Valdés and is used as the basis for the neural network approach.

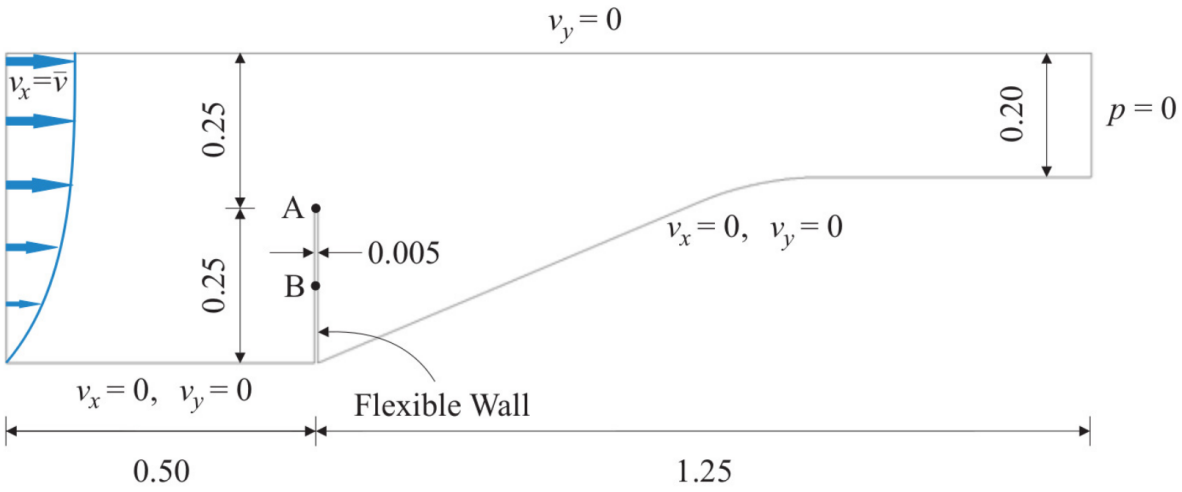

Figure 2: Mok's benchmark of flexible wall structure is in a convergent fluid channel

The geometry of the system can be seen in Figure 2. The general properties are the following:

- The inlet velocity has a parabolic profile with $v(y,t) = 4\bar{v}y(1-y)$, where $\bar{v} = \frac{0.06067}{2}\left(1-\cos\frac{\pi t}{10}\right)$ in the ramp-up phase (the first 10 seconds) and $\bar{v} = 0.06067$ otherwise.
- The fluid domain is considered with a Newtonian constitutive law, with a density $\rho = 956$ kg/m^3 and a kinematic viscosity $\nu = 0.145$ m^2/s as characteristic parameters.
- A linear elastic plane stress constitutive law is considered in the structure domain.
- The structure parameters are density $\rho = 1500$ kg/m^3, Young's modulus $E = 2.6 \times 10^6$ Pa and Poisson's ratio $\nu = 0.45$.
- The timestep is 0.1 seconds, running the simulation for a total time of 20.0 seconds.

The listed properties are used for each of the cases studied in this work. The results are compared for two points at the interface. Point A is located at the very top of the structure, while point B lies in the middle. The study cases are tested with two different couplings: weak or explicit coupling and strong or implicit coupling. Both of them apply a Gauss-Seidel scheme. The surface of the structure is defined as the interface for the exchange of information. The reaction on the fluid is synchronized with the point load on the structure. The displacement of the structure is likewise coupled with the mesh displacement of the fluid. Both solvers' timescales are identical. The meshes of fluid and structure are almost coincident at the interface. Each fluid node has a coincident node from the structure at the same position. The structure has one extra node at the top edge with no corresponding fluid one. The mapping is performed with the nearest neighbors search, which keeps the node alignment.

3.2 Structural surrogate

The first approach tested is the substitution of the structure by a neural network-based surrogate model. In the coupling sequence, the forces on the interface from the fluid model are mapped to the structural model. Then, the neural network solver reads the loads at the nodes of the model as input, preprocesses them, uses them to generate predictions of the displacements, and stores those predictions at the nodes. Then, the new coupling step maps back the node displacements of the structure model part to the fluid mesh displacements.

The training data for the network is generated from the original benchmark with strong coupling. In this case, the input data are the point loads at the nodes, and the output data are their displacements. As the surrogate model will substitute the structure, the values are gathered from the structural model. The next steps are pre-treating the datasets, setting up the neural network model, and training it.

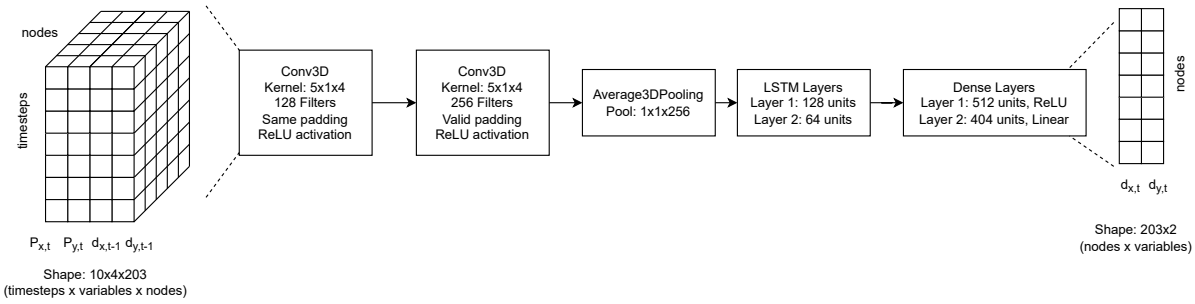

Figure 3: Neural network architecture for strong coupling.

The architecture of the neural network chosen is shown in Figure 3. First, the initial input data are the loads in X and Y direction for each node, in total a 2D-tensor of shape (203, 2) at each of the 200 data points. The output is the displacements at every node, again a 2D-tensor of shape (203, 2) at each of the 200 data points. The displacements are added to the input as lookback in the preprocessing step, retaining the load observed at each earlier timestep. The initial input tensor has, in total, shape (10, 4, 203), having a training input dataset with shape (200, 10, 4, 203). At each of the 10 lookback timesteps and at each of the nodes, the variables are $[P_{X,t}, P_{Y,t}, d_{X,t-1}, d_{Y,t-1}]$. The loads P_t originate from the current timestep, and the displacements d_{t-1} from the previous one. The shape of the input tensor implies using a 3D-CNNLSTM architecture for the neural network. The output tensor is composed of the displacements d_t. The neural network is trained as explained in Section 2.

The results for weak and strong coupling are presented in Figures 4 and 5. In every case, the displacement plots are very close to each other. As expected, the displacement error in the strong coupling case is smaller than in the weak one. The greatest differences come from the pressure plots. As it can be observed, the pressure obtained with weak coupling and the structural surrogate is lower than the original, significantly at the top of the structure. In particular, the maximum pressure at top is a 14% lower than the one from the benchmark. Additionally, the model with strong coupling and the structural surrogate introduces significant instabilities in the pressure measurements. This comes from deformations of the mesh that would not be physically possible with the structural membrane, and appear due to the inaccuracies generated by the neural network. These instabilities are more prominent after the ramp-up phase, which can come from the LSTM layers of the model. LSTM layers are suitable for transient and oscillatory signals, but introduce errors in cases where the input and the output vary relatively little. These errors are accumulated with time, leading to an eventual loss of convergence.

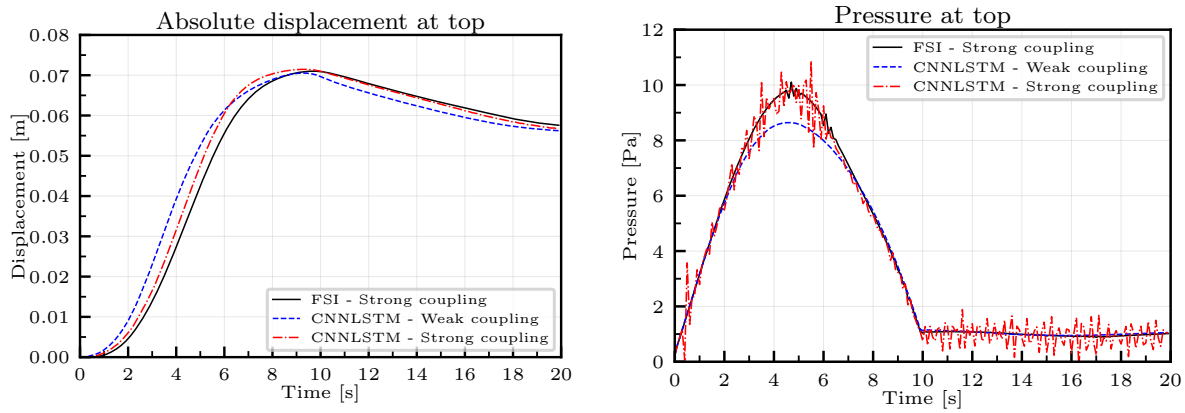

Figure 4: Results at top, structural surrogate.

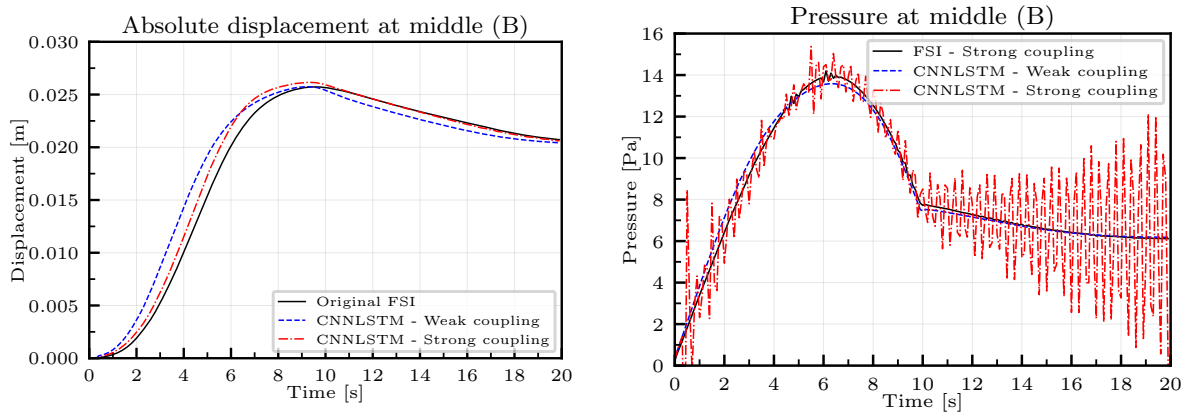

Figure 5: Results at middle, structural surrogate.

3.3 Fluid surrogate

The second alternative consists of substituting the fluid with the surrogate model. This approach is very promising, as most of the computational resources used in solving Mok's benchmark are employed in solving the fluid system. The process is analogous to the substitution of the structural system. The fluid system has two relevant boundary conditions that vary in every timestep: the velocity at the inlet and the displacement at the interface. Its values at the nodes are gathered, preprocessed, and used as input for the neural network. The surrogate model predicts the reaction forces at the interfaces, mapping them afterward to the structural model. Finally, the structural model is solved, and the displacements at the interface are mapped back to the fluid mesh. The velocity inlet is updated in every timestep as specified.

As in the previous case, the first step is generating the training datasets. To this effect, the horizontal component of the velocity at the inlet and the displacement of the interface from the original benchmark are saved as training datasets. The output dataset will be the reaction force at the interface, which is also collected from the initial case. These datasets are used to train the neural network.

The chosen neural network architecture is similar to the one from the structural surrogate case. As shown in Figure 6, the architecture is a 2D-CNNLSTM, combining a 2D-convolutional layer with LSTM layers. In contrast to the structural case, the input data comes from two different model parts, with a different number of nodes and different variables. Therefore, it is not reasonable to group the variables by nodes, leading to a 2D-convolutional layer instead of 3D. The input tensor has a shape (10,854), with the total training dataset shape being (200,10,854).

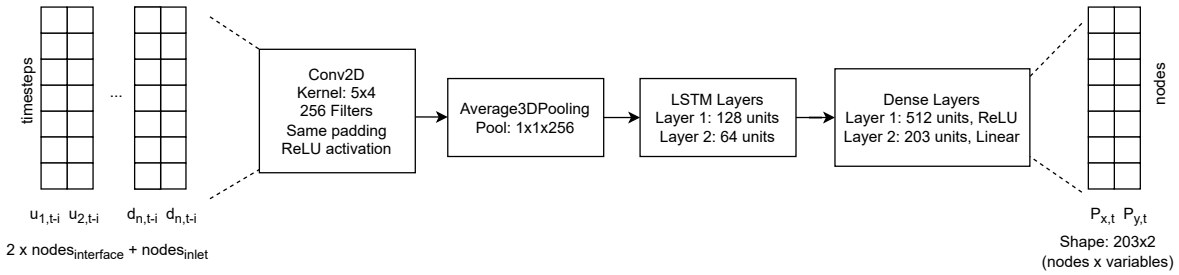

Figure 6: Neural network architecture for fluid surrogate model

The results can be observed in Figure 7 and 8 for points A and B respectively. Weak and strong coupling schemes are analyzed. The obtained results are satisfactory, as the structural displacements in every case are close to the values of the benchmark. In the ramp-up phase, the strong coupling results stick closer to the benchmark ones, while its stabilization is faster than the original. The weak coupling results are the complete opposite, with an oscillatory behaviour in the stabilization phase. The solution loop is run only once per timestep with weak coupling, which amplifies any inaccuracy produced by the neural network. In this case, a variation of the loads causes oscillations in the system, giving results to the observed pattern. The predicted point load results are in this case very remarkable. For point A, they differ greatly from the benchmark values. This is a result of the inaccuracies in the displacement, as the point loads at the top point are affected by the rotation of the structure. This conclusion is supported by the point load plots for point B, which show that the loads are predicted accurately in every case. There is no significant difference between the point loads in weak and strong coupling, as they are a direct

result of the predictions of the neural network. The variations in the displacements from both cases are not sufficiently large to produce a great disturbance in the predictions. It is of great significance that this case stabilizes around the benchmark solution and does not give signs of a loss of convergence.

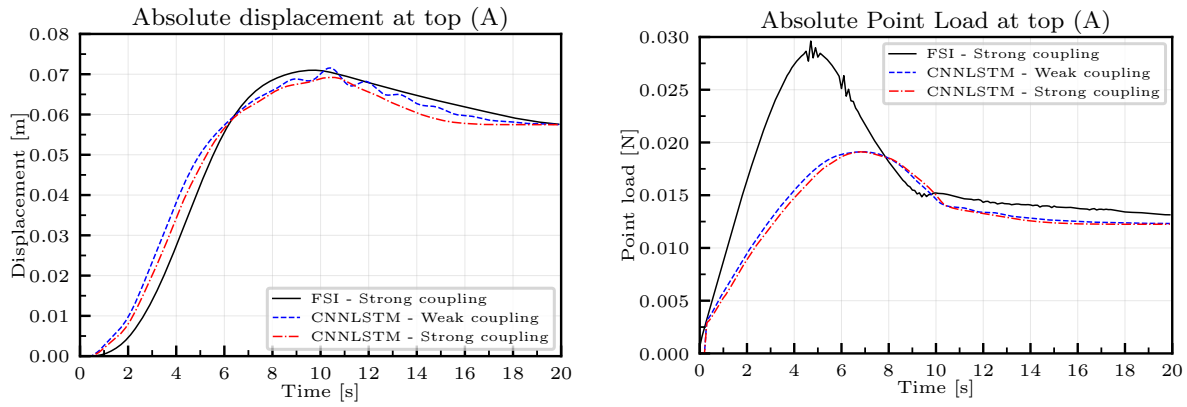

Figure 7: Results at top, fluid surrogate.

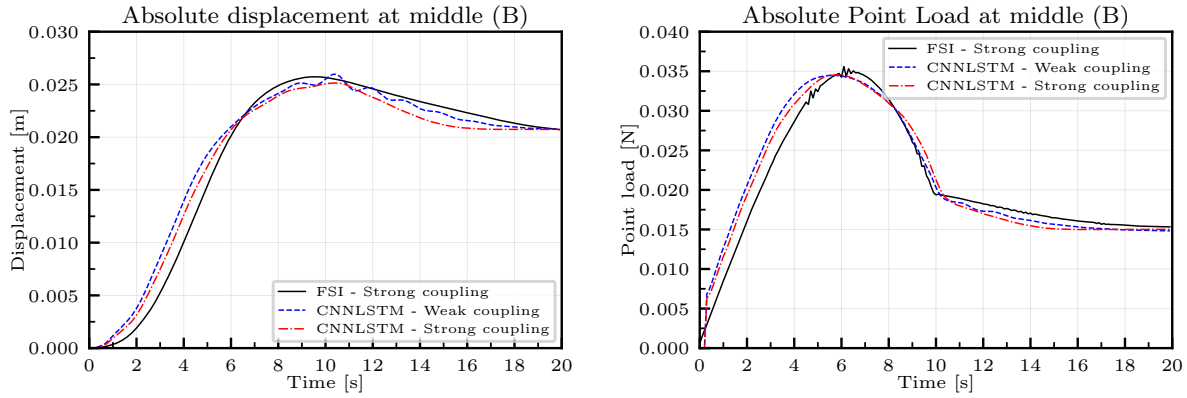

Figure 8: Results at middle, fluid surrogate.

3.4 Comparison

There are two main considerations to compare the different models: accuracy of the displacements and time of simulation. Additionally, there are relevant differences between weak and strong coupling schemes. The displacement results for weak coupling can be observed in Figure 9. The surrogate models are generally accurate in every case. The response of the model with a structural surrogate is closer to the original one, with the fluid surrogate presenting undesirable oscillations. It must be noted that both cases are significantly more accurate than using a weak coupling scheme. This comes from the fact the training data is generated from the strong coupling simulation of the original benchmark, which produces neural networks that predict results closer to the strong coupling ones.

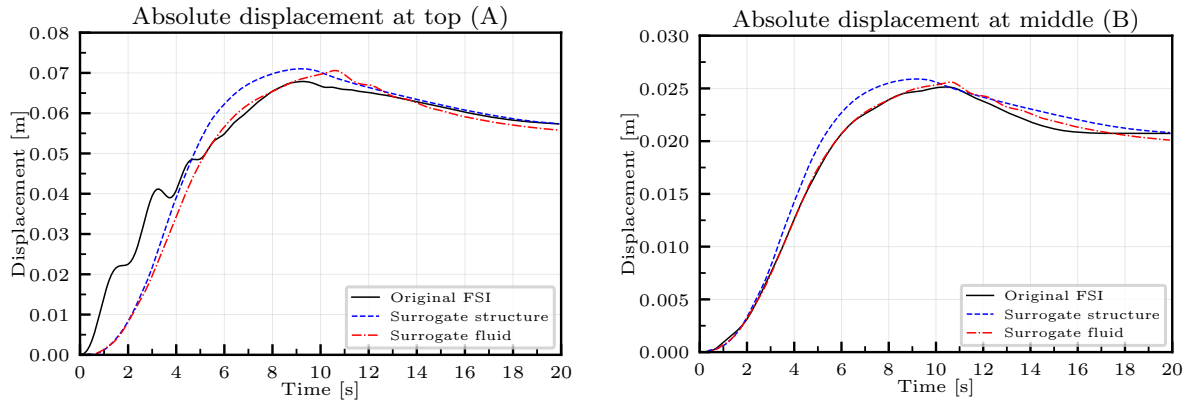

Figure 9: Results comparison with weak coupling.

The results obtained from strong coupling simulations are analogous to those in weak coupling. Higher accuracy and stability are assured by using the strong coupling scheme, which may be required for some applications. However, as observed in the point load plots from Section 3.2, the structural surrogate model may lose the convergence after some time.

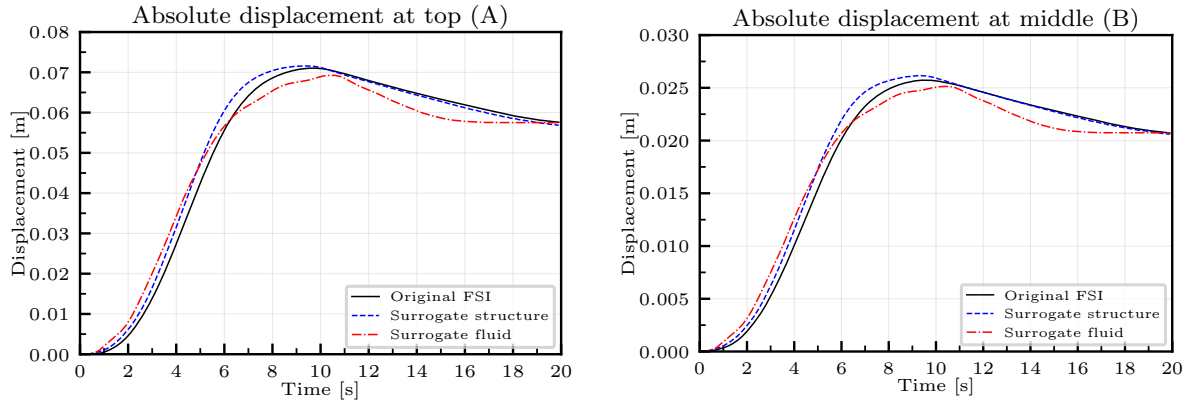

Figure 10: Results comparison with strong coupling.

The different simulation times obtained with the surrogate models are collected in Table 1. As observed, the surrogate structure does not speed up the simulation significantly. However, the weak coupling simulation times are faster than the original full simulation with a strong coupling one, with a negligible loss in accuracy. This approach can be used for generating accurate models that need to be reused, running the strong coupling simulation once and using the surrogate later, especially with a weak coupling scheme. The structural surrogate in strong coupling results in a less stable simulation and therefore requires more iterations of the coupling scheme, leading to longer simulation times. Nevertheless, the greatest improvements in simulation time come from the fluid surrogate in both weak and strong coupling. The fluid simulation is 55.2% faster than the original one in weak coupling and 75.4 % in strong coupling. The difference in time between weak and strong coupling is not significant, as the system does

not require many iterations in the solving loops of the strong coupling. For near-real-time or iterative applications, the substitution of the system with a fluid surrogate is advised.

Table 1: Simulation times for different configurations on Mok's benchmark.

Model	Time weak coupling simulation [s]	Time strong coupling simulation [s]
Full FSI	90.34	181.42
Surrogate structure 3DCNNLSTM	104.43	280.03
Surrogate fluid 2DCNNLSTM	40.45	44.65

4 CONCLUSIONS AND OUTLOOK

We presented the implementation of neural network-based surrogate models applied to FSI problems, namely Mok's benchmark. These models replace the structural or fluid solvers, showing significant advantages with respect to their original counterparts. A structural surrogate trained with the results of the strongly coupled simulation can be used in weak coupling to achieve enhanced results in a shorter simulation time than the original model. Alternatively, a fluid surrogate accelerates the simulation time up to 75% with respect to the original case without increasing the error significantly, which allows its use in time-sensitive applications. These findings create a reference frame for future implementations. The neural network-based surrogate is proved to be suitable for increasingly complex cases, accommodating state-of-the-art model architectures to face their new challenges.

As proposal for future research, the modification of the data generation would have a great impact on the generalization of the use of this surrogate-model approach. The aforementioned methodology uses data generated from a strongly-coupled FSI simulation for training the networks, which in many situations can be undesirable or impractical. An alternative would be the use of data obtained from the individual simulation to be substituted. For example, a simulation of the structure could be run with several load patterns and then the structural surrogate model would be trained with the dataset obtained from it. A fluid surrogate could be generated analogously. The generation of the training datasets in such a manner would completely avoid the need for a coupled simulation with the full models. Nevertheless, this data generation process would introduce new sources of error that would lead to a potential loss of stability and accuracy of the surrogate models that must be analyzed.

References

[1] Brown, D. L. et al. Applied mathematics at U.S. Department of Energy: Past, present and a view to the future. *U.S. Department of Energy Reports* (2008). URL `https://www.osti.gov/biblio/944335`.

[2] Bungartz, H.-J. and Schäfer, M. *Fluid-Structure Interaction*, Springer Berlin Heidelberg, Vol. 53 (2006).

[3] McCulloch, W. S. and Pittis, W. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics* (1943) **5**:115–133.

[4] Werbos, P. J. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. Harvard University (1975).

[5] Vanluchene, R. D. and Sun, R. Neural networks in structural engineering. *Computer-Aided Civil and Infrastructure Engineering* (1990) **5**:207–215.

[6] Flood, I. and Kartam, N. Neural networks in civil engineering. I: Principles and understanding. *Journal of Computing in Civil Engineering* (1994) **8**:131–148.

[7] Flood, I. and Kartam, N. Neural networks in civil engineering. II: Systems and application. *Journal of Computing in Civil Engineering* (1994) **8**:149–162.

[8] Waszczyszyn, Z. *Advances of soft computing in engineering*. Springer Verlag, Vol. 512 (2010).

[9] Bhatnagar, S. et al. Prediction of aerodynamic fields using convolutional neural networks. *Computational Mechanics* (2019) **64**:525–545.

[10] Guo, X., Li, W. and Iorio, F. Convolutional neural networks for steady flow approximation. *22nd ACM SIGKDD International Conference* (2016). 481–490.

[11] Hoyer, S., Sohl-Dickstein, J. and Greydanus, S. Neural reparameterization improves structural optimization. *arXiv*(2019). URL `http://arxiv.org/abs/1909.04240`.

[12] Haghighat, E., Raissi, M., Moure, A., Gomez, H. and Juanes, R. A deep learning framework for solution and discovery in solid mechanics. *arXiv* (2020). URL `https://arxiv.org/pdf/2003.02751.pdf`

[13] Raissi, M., Perdikaris, P. and Karniadakis, G. E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics* (2019) **378**:686–707.

[14] Meethal, R. E. and Kondamadugula, L. S. P. R. Generalized physics-informed machine learning for numerically solved transient physical systems. *AAAI MLPS* (2021).

[15] Meethal, R., Obst, B., Khalil, M., Ghantasala, A., Kodakkal, A., Bletzinger, K. & Wüchner, R. Finite Element Method-enhanced Neural Network for Forward and Inverse Problems. (2022)

[16] Brunton, S. L. and Kuty, J. N. *Data-driven Science and Engineering*. Cambridge University Press (2019).

[17] Totounferoush. A., Schumacher, A. and Schulte, M. Partitioned deep learning of fluid-structure interaction. *arXiv* (2021). URL `http://arxiv.org/abs/2105.06785`.

[18] Meethal. R. E., Ghantasala, A., Bucher, P., Wüchner, R. and Heinrich, C. Co-simulation of multiphysics problems with data driven surrogate models. *ECCOMAS Congress* (2020).

[19] Mok, D. P. Partitionierte lösungsasätze in der strukturdynamik un der fluid-struktur-interaktion. *PhD. Thesis* (2001).

[20] Dadvand, P., Rossi, R. and Oñate, E. An object-oriented environment for developing finite element codes for multi-disciplinary applications. *Archives of computational methods in engineering* (2010) **17**:253–297.

[21] Chollet, F. et al. Keras (2015). URL `https://keras.io/`.

[22] Valdés Vázquez, J. G. Nonlinear analysis of orthotropic membrane and shell structures including fluid-structure interaction. *PhD. Thesis* (2007).

[23] Goodfellow, I., Bengio, Y. and Courville, A. *Deep Learning*. Massachusetts Institute of Technology (2016).

[24] Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Computation* (1997) **9**:1735–1780.