FAST COMPUTATION OF LUBRICATED CONTACTS: A PHYSICS-INFORMED DEEP LEARNING APPROACH

Faras Brumand-Poor[1]*, Niklas Bauer[1], Nils Plückhahn[1], Katharina Schmitz[1]

[1] *RWTH Aachen University – Institute for Fluid Power Drives and Systems (ifas)*

* Corresponding author: Tel.: +49 241 80 47743; E-mail address: faras.brumand@ifas.rwth-aachen.de

ABSTRACT

The frictional behavior of pneumatic seals determines functionality in various fluid power switching applications. Understanding the complex relationship between friction and component properties is challenging. Experimental descriptions are often infeasible due to time and cost constraints. Therefore, an elastohydrodynamic lubrication (EHL) simulation model, the ifas-DDS, was implemented to calculate the friction in translational pneumatic seals [1]. While the EHL simulation provides an accurate solution to the underlying partial differential equations, it is computationally expensive. An approach to obtain an accelerated solution is the use of neural networks (NN). However, their main disadvantage is that they do not necessarily embed the physical mechanism underlying a particular dataset. In recent years, a new form of NN has emerged, the physics-informed neural network (PINN), which is imposed with physical laws, increasing the algorithm's accuracy. This approach offers several possibilities, e.g., extrapolation and more robust training. In this paper, a variation of the Reynolds equation, implemented in the ifas-DDS, is solved with a PINN. The complete hydrodynamic PINN (HD-PINN) framework is presented and compared to the ifas-DDS afterward. The results show the possibility of the HD-PINN framework for modeling the EHL with high speed, less to no accuracy loss, and minimal tuning effort.

Keywords: Tribology, Physics-informed neural networks; Elastohydrodynamic simulation; Pneumatic sealing; Physics-informed machine learning

1. INTRODUCTION

Lubricated contacts are part of many fluid power components. They occur e.g., in seals of hydraulic cylinders and pneumatic valves or functionally critical contacts to contacts in pumps. Since they often have a crucial influence on the performance and efficiency of the respective components, it is important to comprehensively understand all phenomena within these contacts by physically motivated models. However, obtaining an analytical solution for the complex phenomena in lubricated contacts is often impossible without making several simplifications. Furthermore, characterizing this through experiments is often unfeasible due to their time-consuming and expensive nature.

Consequently, the behavior of lubricated contacts is modeled using an elastohydrodynamic lubrication (EHL) simulation, representing the acting phenomena through the Reynolds equation, and defining the pressure distribution within the tribological contact. One example of a validated EHL simulation model is the ifas-DDS, which was implemented at ifas to calculate the frictional behavior of reciprocating pneumatic seals. This is achieved by solving the Reynolds equation to obtain the hydrodynamic pressure within the contact and a FEM calculation for the seal's deformation. The EHL yields good agreement with experimental data, shown in **Figure 1**, but requires a long computation time for a rather short-term process, due to the complexity of the partial differential equations and their numerical solution [2]. A straightforward method to reduce

computation time is to enhance computational power. However, this may not always be feasible and may even become more challenging as the simulation grows in complexity. This is especially true when a component with multiple contacts or even a system consisting of multiple of these components is to be considered. In that case, the computation time of current EHL simulation models makes these simulations almost impossible to apply.

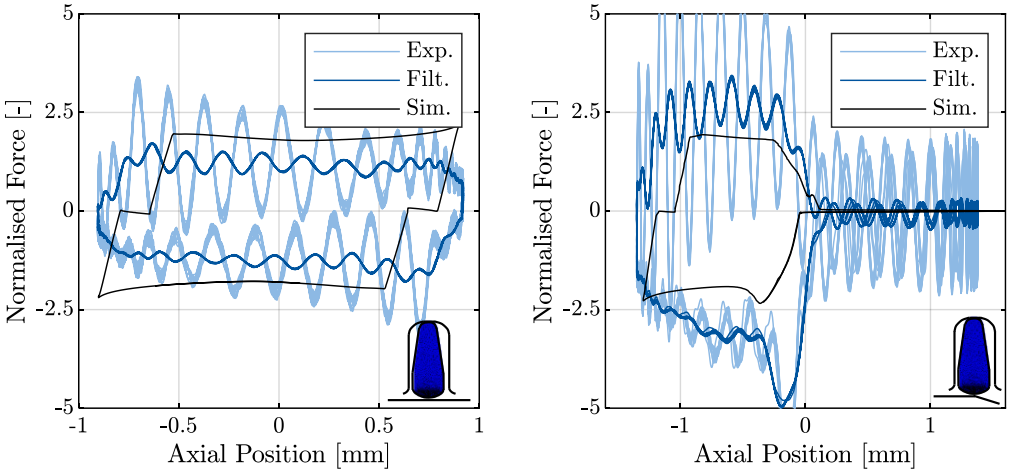

Figure 1: Comparison of measured (filtered and unfiltered) and simulated friction force. Left: Without moving over the control edge. Right: Moving over the control edge. [2]

A novel approach to accelerate the computational process of simulations has emerged in recent years: the physics-informed neural network (PINN). Traditional neural networks require a shorter computation time regarding most iterative solver of distributed parameter simulations. However, they lack an understanding of the physical laws governing a specific dataset. In typical regression tasks, the objective of a neural network is to minimize the deviation between the predicted output and the true output. However, tuning the network's parameters to achieve this goal may only be valid within the provided data range. In contrast, PINNs incorporate the underlying physical laws of the investigated simulation, providing a more universal and accelerated solution for complex simulation models.

This contribution introduces a framework for developing and parametrizing a PINN to speed up the computation of pressure distribution in EHL models. For this, the framework's various components are described and the advantages regarding a successful implementation of PINNs are highlighted. Firstly, the hydrodynamic lubrication (HL) is presented in section 2. Succeeding, an overview of the literature regarding PINNs, their application for HL, and their main drawbacks is provided. The fourth section of this paper deals with the whole framework. It describes the implemented solutions for the common issues encountered in PINNs described in the previous section. In section 5 the accuracy of the PINN is compared to the rigid ifas-DDS. In this contribution the deformation of the seal is not considered, therefore resulting in a purely HL model. Eventually, the results are summarized in section 6.

2. HYDRODYNAMIC LUBRICATION

EHL simulations are employed to calculate friction, leakage, and wear characteristics within lubricated contacts. These simulations account for the interplay between the lubricant and contacting surfaces, involving the computation of surface deformations within the contact area and the subsequent build-up of hydrodynamic pressure.
The ifas-DDS is a simulation model used to describe the interaction between a seal and the adjacent

counterface. It takes into consideration the presence of lubricating fluid between the seal and counterface. The deformation of the seal is determined through the utilization of the finite element software Abaqus. The HL phenomena are described by the Reynolds equation and integrated in Abaqus through user subroutines.

In the scope of this study, the primary emphasis is on solving the Reynolds equation and developing the PINN framework. Consequently, the deformation of the two contact partners is not taken into consideration. Typically, a seal undergoes deformation under load, leading to modifications in the HL behavior. To validate the presented PINN, the simulation model ifas-DDS without deformation is employed, herein referred to as rigid ifas-DDS. The ifas-DDS in its original form encompasses an extended Reynolds equation using flow factors Φ^{τ} and Φ^{p}, as described by Patir and Cheng, which allows for the consideration of surface topography [1], and an implementation of the Jakobsson-Floberg-Olsson cavitation model [3] introducing the parameter θ.

The complete Reynolds equation used in the ifas-DDS extending the originally derived equation from Osbourne Reynolds in 1886 is given as [4]:

$$\frac{v}{2}\frac{\partial}{\partial x}\Big((1-\theta)\rho h + (1-\theta)\rho R_q \Phi^{\tau}\Big) - \frac{1}{12\eta}\frac{\partial}{\partial x}\Big(\Phi^{p}\rho h^3(1-\theta)\frac{\partial p}{\partial x}\Big) + \frac{\partial}{\partial t}\big((1-\theta)\rho h\big) = 0 \quad (1)$$

3. PHYSICS-INFORMED NEURAL NETWORKS

As mentioned in the preceding section, the Reynolds equation is typically solved using finite volume, element, or difference methods. Recently, machine learning models have found success in the field of tribology [5,6]. Notable examples include the application of autoencoders and convolutional neural networks for fault detection in tribological systems. Particularly noteworthy is the work of Hess and Shang, who developed a convolutional neural network to determine the elastohydrodynamic pressure distribution in journal bearings [7]. Machine learning, especially neural networks as universal function approximators, has demonstrated significant potential, not only for data modeling but also for solving partial differential equations.

The mathematical foundation, demonstrating that a feed-forward neural network with at least one hidden layer can approximate any continuous function with arbitrary accuracy, was laid by Cybenko in 1989 [8]. In the same year, Hornik extended this to Borel measurable functions [9]. A year later, Hyuk, one of the pioneers in the field of physics-informed neural networks, investigated the ability to solve differential equations with neural networks [10]. While Hyuk did not use the term "physics-informed," the idea and motivation behind his work align with the principles of PINNs. The loss function for the neural network under investigation was expanded to incorporate the underlying differential equation.

However, this approach was relatively overlooked but saw a resurgence in interest due to progress in available computational resources, more advanced ML models, and efficient gradient calculation techniques by automatic differentiation.

This revival of physics-inspired machine learning was led by Owhadi, as one of the first in 2014, embedding prior knowledge about the desired solution. He suggested reformulating the PDE solution problem as a Bayesian inference problem and presented the potential of enhancing the algorithm with prior knowledge [11]. Raissi et al. developed a probabilistic ML algorithm for

general linear equations using the Gaussian process [12,13]. The Gaussian process is tailored to the investigated integro-differential or partial differential equation. This approach was extended by Raissi et al. for nonlinear partial differential equations [14,15].
A significant milestone for the physics-informed machine learning community was the introduction of physics-informed neural networks [16], which Cuomo et al. describe as a mesh-free technique for solving PDEs by transforming the problem of directly solving the underlying equation into a loss function optimization problem [17].
Raissi introduced PINNs as a novel data-driven solver class in a two-part publication in 2017 and later in a merged article in 2019, PINNs aim to concurrently minimize the deviation of known values such as boundary and initial conditions (supervised losses), and the PDE residual (unsupervised), within the investigated computational space [18,19,20].

The first published work on PINNs solving the Reynolds equation was by Almqvist in 2021, focusing on the fundamental application of PINNs for the Reynolds equation [21]. More advanced algorithms were developed by Zhao et al. and Li et al., dealing with the 2D Reynolds equation for linear sliders and gas bearings, respectively [22, 23]. Most recently, Rom became the first to develop PINNs for solving the stationary Reynolds equation with the Jakobsson-Floberg-Olsson (JFO) cavitation model [24]. Building on this success, Cheng et al. implemented a PINN to solve the Reynolds equation with either the JFO or the Swift-Stieber (SS) cavitation model, applying three different multi-task learning methods to balance the various loss terms [25]. It's worth noting that most of these publications primarily focus on the PINN itself rather than on a framework for developing PINNs to solve HL tasks. Therefore, a significant amount of manual work is still required.

In **Figure 2** an exemplary hybrid (physics-informed and data-based) PINN is shown [26]. The four key elements are the neural network, the automatic differentiation, the losses, and the optimizer. As already described, the loss contains more than the difference between actual and predicted output. The residual loss term of the PDE is determined by the computation of partial derivatives using automatic differentiation [27]. Using the chain rule, the derivatives of the outputs are computed with respect to the network inputs, weights, and biases. Typically, the training of PINNs is based on the L2-norm (mean squared error, MSE) computed on uniformly sampled collocation points within the computational space [26]. In this work, the residual loss of the Reynolds equation is computed for uniformly distributed positions of the seal and the adjacent counterface.

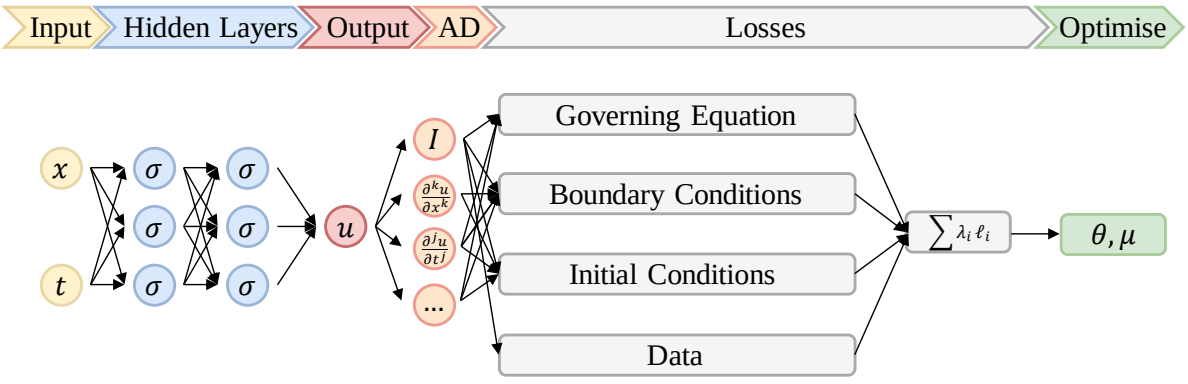

Figure 2: Schematic illustration of a physics-informed neural network. Adapted from [26].

In **Figure 2** the four different types of loss terms are illustrated: The loss based on the governing equation (e.g., Reynolds equation), on the boundary conditions (e.g., pressure at the boundaries of the investigated geometry), on the initial conditions (e.g., the initial pressure distribution over the whole geometry) and eventually on the available data (provided by the e.g., ifas-DDS).

The traditional PINN consists of the three first loss terms, adding the data loss results in a hybrid PINN, which is not considered in this work, to investigate the potential of the physics-informed idea. These losses are minimized jointly and therefore represent a multi-objective optimization. This class of optimization deals with concurrently minimizing a set of more than one, potentially conflicting objective [28]. Optimization problems concerning engineering tasks generally require trade-offs to synchronously fulfill all terms to the desired degree [29]. The desired solution for multi-objective optimization problems can be represented as a set of Pareto optima, which defines a feasible solution, where one objective cannot be minimized without worsening another term [30]. A multi-objective optimization is independent of balancing the different terms [31]. However, the non-convex nature of the physical solution space in multi-objective optimization poses challenges for finding the globally optimal solution through gradient-based optimization [26]. Moreover, the gradients in multi-objective optimization of physical systems can vary greatly. Most optimizer adjusts the network parameters using an explicit scheme (the updated parameter is based on the current parameter) and can face instability if the optimization steps are too high. For classical neural networks, the problem of different magnitudes in the loss gradient occurred rarely, leading to the predominant use of explicit schemes instead of implicit ones (considering the updated parameter in the update itself) to speed up the training procedure. However, this problem occurs in PINNs and generally requires a reduction of the loss update in each step. Unfortunately, a small magnitude can trap the optimizer in a local minimum. Therefore, a loss balancing algorithm is implemented for the hydrodynamic PINN (HD-PINN) framework and presented in section 4.3.

The training of PINNs involves a great number of design options (hyperparameters), e.g., network width, network depth, and learning rate. As mentioned, PINNs solve multi-objective optimization, which introduces additional hyperparameters to the whole set due to the need for loss balancing. The high number of dimensions of these parameters makes it challenging to find the optimal set. Therefore, finding the optimal hyperparameter set, yielding the best performance, becomes an iterative process. Depending on the specific problem, a single training routine can be time-consuming, and infeasible for a high number of iterations. Basic approaches such as manual or grid search suffer from the curse of dimensionality and might be inefficient [32]. This problem of PINNs is addressed by enhancing the HD-PINN framework with a sophisticated and automated hyperparameter search and presented in section 4.2

In summary, PINNs offer a promising method for solving HL simulation models by optimizing their parameters based on several physics-informed objectives. The following section introduces the HD-PINN framework, which includes the investigation of the differential equation, automated hyperparameter search, and loss balancing.

4. METHODOLOGY

In this section, the HD-PINN framework is presented. First, the investigated Reynolds equation is derived, and the PINN input and output are described. Then, the loss calculations for the residual loss term of the Reynolds equation and the pressure boundaries are explained briefly. The subsequent part deals with the efficient and robust parameter tuning of the framework. The third part of this section presents an algorithm for the improvement of the training due to loss balancing.

4.1. PINN for Solving the Reynolds Equation

As stated in the previous section, a handful of researchers have investigated the PINN for solving the Reynolds equation. The researchers focused primarily on the PINN itself and therefore still had

to manually tune a great number of design options e.g., layer width or depth.
In this work, the emphasis is on creating an automated HD-PINN framework designed to address the common issues and limitations associated with PINNs. It is initially validated using a simplified version of the Reynolds equation but can be easily extended to the complete mathematical expression with the transient term and the JFO cavitation model. The underlying assumptions are as follows:

- Time dependencies are neglected ($\frac{\partial}{\partial t} = 0$)
- Cavitation is not considered ($\theta = 1$)
- The surface is ideally smooth ($\Phi^{\tau} = 0, \Phi^{p} = 1$)

Therefore, the investigated Reynolds equation holds:

$$\frac{v}{2}\frac{\partial}{\partial x}\rho h - \frac{1}{12\eta}\frac{\partial}{\partial x}\left(\rho h^3 \frac{\partial p}{\partial x}\right) = 0 \tag{2}$$

The PINN obtains the dynamic viscosity η and density of the fluid ρ, the height h, and the velocity of the counter surface v as input and determines the pressure p as output. The equation above contains partial differentials, which are required to obtain the physics-informed loss term for the residual of the Reynolds equation. These values are obtained by automatic differentiation, which allows for accurate and efficient function derivation [33].
Furthermore, the boundary loss terms, which are Dirichlet boundary conditions for the Reynolds equation, consist of two values for the left and right boundary respectively. These are compared to the boundary pressure predicted by the PINN and embedded as an MSE term. For PDEs including Neumann boundary conditions, automatic differentiation can be used to efficiently determine the desired loss function. The physics-informed Reynolds loss is implemented as the following:

$$\ell_{Rey} = \mathrm{MSE}\left(\frac{v}{2}\frac{\partial}{\partial x}\rho h - \frac{1}{12\eta}\frac{\partial}{\partial x}\left(\rho h^3 \frac{\partial p}{\partial x}\right), 0\right) \tag{3}$$

The MSE of the Reynolds equation compared to zero is calculated as the loss and strongly resembles the residual term of common EHD simulations. For the boundary loss the following function is implemented:

$$\ell_{\mathrm{BC},l,r} = \mathrm{MSE}\left(p_{HD-PINN,l,r}, p_{b,l,r}\right) \tag{4}$$

In summary, the PINN loss comprises three terms that do not depend on any specific data provided by simulation of experiments. The rigid ifas-DDS primarily serves to acquire a pressure distribution for validating the PINN. In the next sections, two main features of the HD-PINN framework are presented. These solve the previously explained common drawbacks occurring in the application of PINNs. Eventually, the whole training process is described.

4.2. Hyperparameter Optimization

Compared to traditional neural networks, PINNs exhibit even more hyperparameters, especially due to the already-mentioned loss balancing. Basic approaches for finding optimal parameters often

suffer from the curse of dimensionality and are therefore not feasible for the HD-PINN framework.

Bayesian optimization [34] offers a solution to this dilemma and has already been successfully integrated into PINNs [32]. Bayesian optimization aims to obtain an optimal set of hyperparameters with as few training procedures as possible by approximating an unknown loss function (PINN losses) with a probabilistic surrogate model (Gaussian Process). After evaluating one set of hyperparameters, the next set is chosen based on an acquisition function (expected improvement), which aims to achieve a trade-off between exploration and exploitation by comparing the current best set to the next chosen set.
The HD-PINN framework incorporates Bayesian optimization to determine the optimal hyperparameters, including the number of layers, layer width, and four parameters relevant to the loss balancing task: learning rate, decay rate, temperature factor, and saudade value. These parameters will be introduced in the following section.

4.3. Loss Balancing

The Bayesian optimizer aims to find the optimal set of hyperparameters for the given task. For a given set of hyperparameters, their performance is analyzed by training the PINN for a defined number of epochs and eventually evaluating the final sum of losses. During one epoch, the PINN determines the pressure distribution based on the provided inputs for a pre-defined number of collocation points (ten in this work). With the pressure distribution, the residual loss of the Reynolds equation and the loss for the boundary values are determined.
The loss is further utilized and differentiated with automatic differentiation to calculate the changes in the network parameters, including weights and biases. The optimization of these network parameters is achieved using the adaptive moment estimation (ADAM) algorithm, a first-order gradient-based explicit optimization method for stochastic objective functions [35]. ADAM is widely recognized as one of the state-of-the-art optimizers for deep neural networks [36,37]. It is computationally efficient with little memory requirements, capable of solving problems of large scale, and has been successfully implemented in PINN applications [38].

As stated in section 3, explicit optimizers might face problems due to the heavily varying loss term magnitudes. Therefore, a loss balancing scheme according to [26] is added to the HD-PINN framework. Bischof proposed an algorithm based on combining three already existing methods and adding one novel feature. The four features of the Relative Loss Balancing with Random Lookback (ReLoBRaLo) are as follows:

1. The sum of scalings is bound by a softmax function, which is adjusted by the so-called temperature factor $\mathcal{T}$ [39].
2. The learning progress is considered by dividing the loss at the current iteration by the loss of the last iteration [40].
3. Inspired by the Learning Rate Annealing, the scalings are set to higher values initially and decay exponentially over the iterations with the decay factor α. This leads to the utilization of loss statistics from more than one training step [41].
4. A random lookback (defined by the saudade value ρ_s) is embedded in the exponential decay and randomly decides the consider the previous steps' loss statistics or look back to the beginning of the training to compute the scalings [26].

This novel balancing algorithm adds a significant performance enhancement to the HD-PINN framework by tackling the loss scaling problems. **Figure 3** illustrates the training

progress of the complete HD-PINN framework. The Bayesian optimizer selects parameters, which are used for the initialisation of the PINN and the network parameter optimizer. Afterward, the actual pressure prediction is done by the PINN and the loss is computed. If the training is not finished, the loss is used to update the PINN's weights and biases, and a new pressure distribution is predicted. This loop continues until a certain number of iterations is reached. Afterward, the parameter determined by the Bayesian optimizer is updated.
In the next section, the whole framework is validated for two different scenarios and an illustration of the training progress is provided.

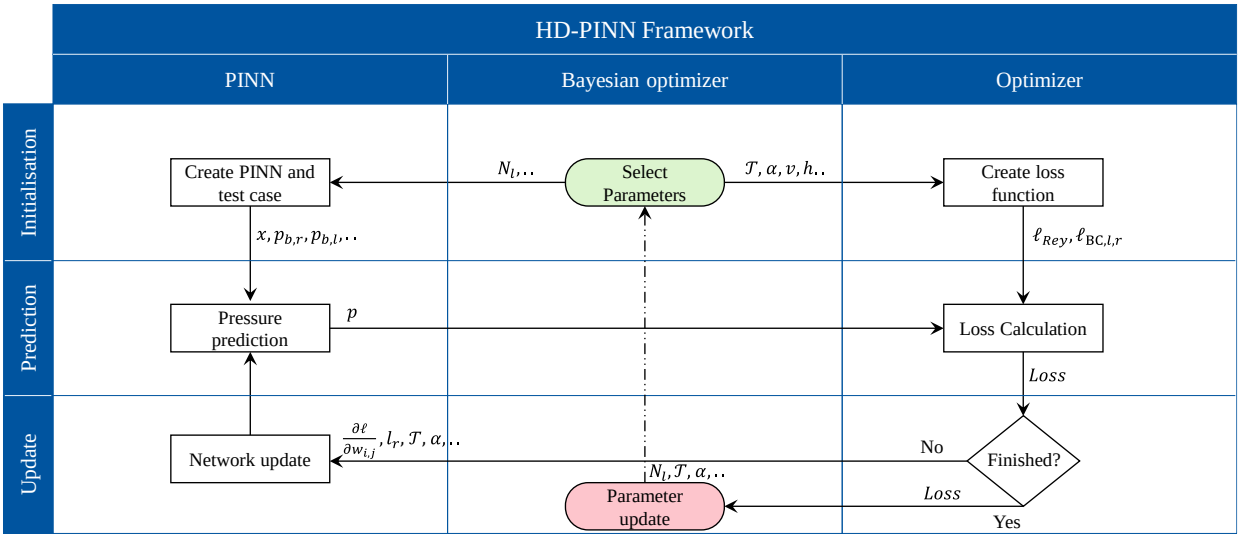

Figure 3: The HD-PINN framework and its training progress.

5. RESULTS AND DISCUSSION

The HD-PINN is tested for two test scenarios, shown in **Figure 4**, a convergent ($h = [1, 0.5]$), and a divergent ($h = [0.5, 1]$), height profile. The convergent case resembles Almqvist's work for a linear slider with dimensionless film thickness [26]. However, in comparison to that work, the pressure boundary is not set to zero (left pressure for the convergent gap $p_{b,l} = 0.2$, for the divergent gap $p_{b,l} = 0.7$ and right pressure for the convergent gap $p_{b,r} = 0.7$, for the divergent gap $p_{b,r} = 0.4$).

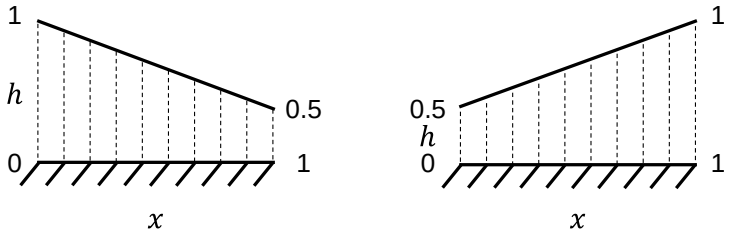

Figure 4: Left: Convergent height profile Right: Divergent height profile.

The PINN is tuned by the Bayesian optimizer for 30 trials and eventually, the best-estimated parameters are used for the actual training process (200 training iterations). The validation data is obtained by the rigid ifas-DDS. It is noteworthy that the convergent height profile required the maximum number of iterations (1000) to obtain the residues set in the ifas-DDS.

First, the convergent height profile is presented. In **Figure 5** the pressure distribution and the loss progression are illustrated. The HD-PINN predicts the pressure with high accuracy, with a maximum deviation of around 1 %. The residual loss of the Reynolds equation and the boundary loss decrease drastically over 200 epochs, with minor fluctuations in the boundary term and none in

the Reynolds term.
For the divergent case, an even higher accuracy of the pressure distribution (deviation of less than 1 %) can be observed in **Figure 6**. Similar to the first case, the Reynolds-based loss remains stable without any fluctuations, while the boundary loss term exhibits more volatility.

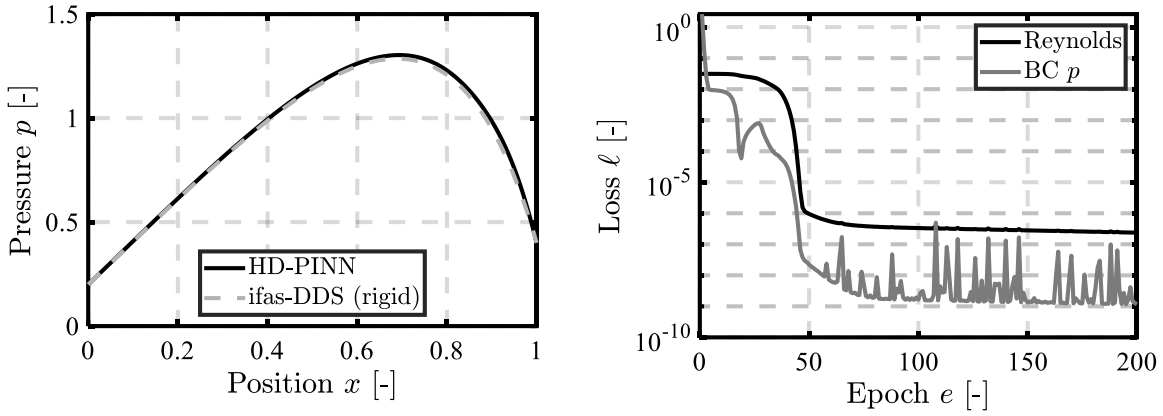

Figure 5: Left: Pressure distribution of HD-PINN vs. rigid ifas-DDS for the convergent gap. Right: The loss trajectories for the convergent gap.

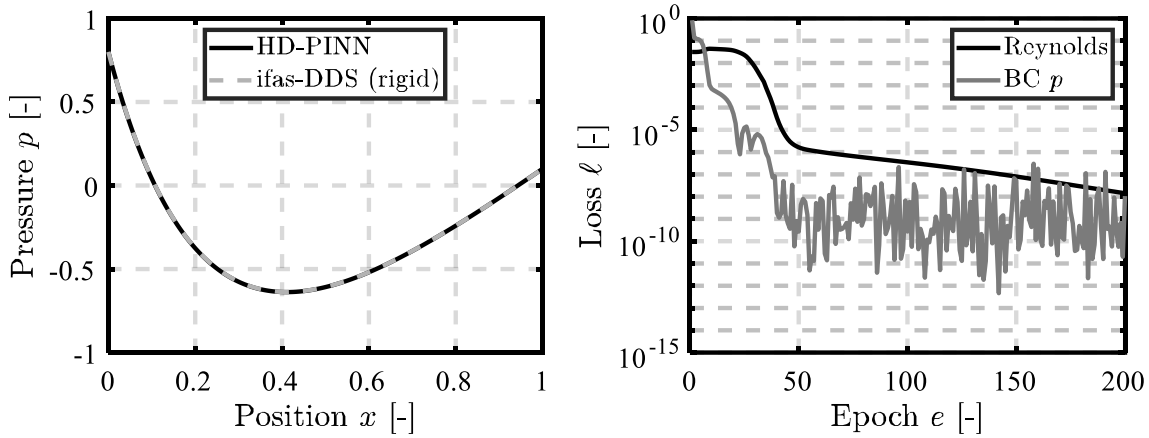

Figure 6: Left: Pressure distribution of HD-PINN vs. rigid ifas-DDS for the divergent gap. Right: The loss trajectories for the divergent gap.

To obtain a visual representation of the learning process of the PINN four different pressure distributions after different epochs (1, 10, 30, 40) are illustrated in **Figure 7**. Like the boundary loss trajectories, the pressure boundary values are met first and already precisely predicted after ten epochs. An accurate prediction is achieved after 40 epochs, which illustrates the fast-learning ability of the PINNs and suggests decreasing the whole 200 epochs to reduce the training time in future research.

6. SUMMARY AND CONCLUSION

This contribution introduces an automated HD-PINN framework for predicting pressure build-up in sealing contacts within a housing, described by the Reynolds equation. It starts by presenting the concept of physics-informed neural networks and their advantages. Then, it addresses the common drawbacks of PINNs and describes various components implemented in this work to mitigate these issues.
The incorporation of the Bayesian optimizer and the loss balancing algorithm reduces the amount of manual tuning of the hyperparameters and scaling the loss terms.

The validation of the framework, utilizing a variant of the Reynolds boundary equation with a fixed height profile demonstrates the PINNs' ability to model lubricated contacts with high speed and, notably, with minimal accuracy loss compared to traditional HL simulations.
Future steps involve integrating the missing terms of the Reynolds equation to obtain an accelerated solution for the complete rigid ifas-DDS, which encompasses transient behavior and cavitation. Additionally, an extended set of inputs will be defined to characterize the investigated geometry, broadening the PINN's range of validity to accommodate various geometries rather than being specific to just one. Furthermore, a second model will be developed to account for sealing deformation and thereby address the entire EHL in the ifas-DDS.

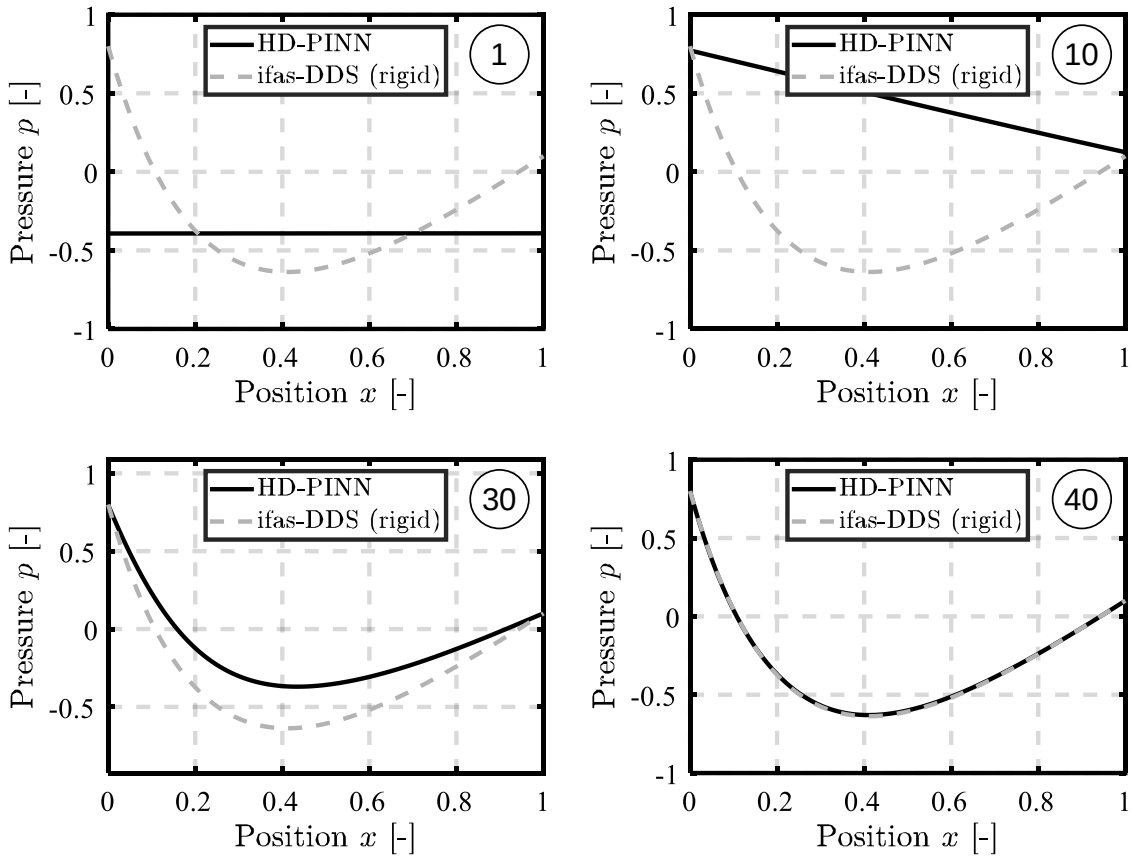

Figure 7: Pressure distribution of the HD-PINN after 1, 10, 30, and 40 epochs for the divergent gap.

ACKNOWLEDGMENTS

The authors thank the Research Association for Fluid Power of the German Engineering Federation VDMA for its financial support. Special gratitude is expressed to the participating companies and their representatives in the accompanying industrial committee for their advisory and technical support.

NOMENCLATURE

e	Epoch	-
h	Dimensionless height profile	-
ℓ	Loss	-
l_r	Learning Rate	-
N_l	Number of hidden network layers	-

t	Dimensionless time	-
p	Dimensionless pressure	-
$p_{b,l,r}$	Dimensionless left and right boundary pressure	-
R_q	Root of the mean squared roughness	-
v	Velocity of the counter surface	-
x	Dimensionless position	-
α	Decay	-
η	Dimensionless dynamic viscosity	-
θ	Dimensionless cavitation	-
ρ	Dimensionless density	-
ρ_s	Saudade value	-
Φ^p	Flow factor	-
Φ^τ	Shear flow factor	-
$\mathcal{T}$	Temperature factor	-
$\frac{\partial}{\partial x}$	Partial derivative regarding the position and time	-
$\frac{\partial}{\partial t}$	Partial derivative regarding the time	-

REFERENCES

[1] Bauer, N., et. al. (2023) Elastohydrodynamic Simulation of Pneumatic Sealing Friction Considering 3D Surface Topography. Chem Eng & Technol 46 (1).

[2] Bauer, N., et. al. (2022) Experimental determination and EHL simulation of transient friction of pneumatic seals in spool valves. International Sealing Conference: 21st ISC.

[3] Angerhausen, J., et al. (2019) Simulation and experimental validation of translational hydraulic seal wear. Tribology International 134 296-307

[4] Bauer, N., et. al. (2021) Strategies for Implementing the Jakobsson-Floberg-Olsson Cavitation Model in EHL Simulations of Translational Seals. Journal of Fluid Power, 22(2), 199-232

[5] Marian M., Tremmel S. (2021) Current Trends and Applications of Machine Learning in Tribology—A Review. Lubricants 9(9), 86

[6] Paturi. U., Palakurthy S., Reddy N. (2023) The Role of Machine Learning in Tribology: A Systematic Review. Archives of Computational Methods in Engineering 30, 1345-1397

[7] Hess N., Shang L. (2022) Development of a Machine Learning Model for Elastohydrodynamic Pressure Prediction in Journal Bearings. J. Tribol 144(8)

[8] Cybenko G. (1989) Approximation by Superpositions of a Sigmoidal Function. Math. Control Signals Systems 2, 303-314

[9] Hornik K., Stinchcombe M., White H. (1989) Multilayer feedforward networks are universal approximators. Neural Networks, 2(5) 359-366

[10] Lee H., Kang S. (1990) Neural algorithm for solving differential equations. Journal of Computational Physics 91(1) 110-131

[11] Owhadi H. (2014) Bayesian Numerical Homogenization. arXiv: 1406.6668v2

[12] Raissi M., Perdikaris P., Karniadakis G. (2016) Inferring solutions of differential equations using noisy multi-fidelity data. arXiv: 1607.04805v1

[13] Raissi M., Karniadakis G. (2017) Machine Learning of Linear Differential Equations using Gaussian Processes. arXiv: 1701.02440v1

[14] Raissi M., Perdikaris P., Karniadakis G. (2017) Numerical Gaussian Processes for Time-dependent and Non-linear Partial Differential Equations. arXiv: 1703.10230v1

[15] Raissi M., Karniadakis G. (2017) Hidden Physics Models: Machine Learning of Nonlinear Partial Differential Equations. arXiv: 1708.00588v2

[16] Blechschmidt J., Ernst O. (2021) Three Ways to Solve Partial Differential Equations with Neural Networks -- A Review. arXiv:2102.11802v2
[17] Cuomo S., et al. (2022) Scientific Machine Learning Through Physics-Informed Neural Networks: Where we are and What's Next. Journal of Scientific Computing 92(3)
[18] Raissi M., Perdikaris P., Karniadakis G. (2017) Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations. arXiv: 1711.10561v1
[19] Raissi M., Perdikaris P., Karniadakis G. (2017) Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations. arXiv: 1711.10566v1
[20] Raissi M., Perdikaris P., Karniadakis G. (2019) Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics 378(1), 686-707
[21] Almqvist A. (2021) Fundamentals of Physics-Informed Neural Networks Applied to Solve the Reynolds Boundary Value Problem. Lubricants 9(8)
[22] Zhao Y., Guo L., Wong P. (2022) Application of physics-informed neural network in the analysis of hydrodynamic lubrication. Friction, 11, 1253–1264
[23] Li L. et. al. (2022) ReF-nets: Physics-informed neural network for Reynolds equation of gas bearing. Computer Methods in Applied Mechanics and Engineering, 391
[24] Rom M. (2023) Physics-informed neural networks for the Reynolds equation with cavitation modeling. Tribology International 179: https://doi.org/10.1016/j.triboint.2022.108141
[25] Cheng Y., et. al. (2023) HL-nets: Physics-informed neural networks for hydrodynamic lubrication with cavitation. Tribology International 188: https://doi.org/10.1016/j.triboint.2023.108871
[26] Bischof R., Kraus M. (2022) Multi-Objective Loss Balancing for Physics-Informed Deep Learning. arXiv: 2110.09813v2
[27] Cai S., et. al. (2021) Physics-informed neural networks (PINNs) for fluid mechanics: A review. arXiv:2105.09506v1
[28] Caruana R. (1997) Multitask Learning. Machine Learning 28, 41-75
[29] Chang K.-H. (2015) Chapter 17 – design optimzation. E-Design Ed. Academic Press, 907-1000
[30] Sener O., Koltun V. (2019) Multi-Task Learning as Multi-Objective Optimization. arXiv:1810.04650v2
[31] Heydari A., Thompson C., Mehmood A. (2019) SoftAdapt: Techniques for Adaptive Loss Weighting of Neural Networks with Multi-Part Loss Functions. arXiv:1909.04630v1
[32] Escapil-Inchauspé P., Ruz G. (2023) Hyper-parameter tuning of physics-informed neural networks: Application to Helmholtz problems. arXiv:2205.06704v2
[33] Baydin A., et. al. (2018) Automatic Differentiation in Machine Learning: a Survey. Journal of Machine Learning Research 18(153):1−43
[34] Močkus J. (2023) On bayesian methods for seeking the extremum. Optimization Techniques IFIP Technical Conference Novosibirsk: https://doi.org/10.1007/3-540-07165-2_55
[35] Kingma D., Ba J. (2014) Adam: A Method for Stochastic Optimization. arXiv:1412.6980v9
[36] Schmidt R., Schneider F., Henning P. (2021) Descending through a Crowded Valley - Benchmarking Deep Learning Optimizers. arXiv:2007.01547v6
[37] Reyad M., Sarhan A., Arafa M. (2023) A modified Adam algorithm for deep neural network optimization. Neural Comput & Applic 35, 17095–17112
[38] Singh S., et. al. (2023) Adam Optimization of Burger's Equation Using Physics-Informed Neural Networks. International Conference on Advancement in Computation & Computer Technologies
[39] Rajeswaran A., et. al. (2019) Meta-Learning with Implicit Gradients. arXiv:1912.12355v1
[40] Chen Z. et. al. (2018) GradNorm: Gradient Normalization for Adaptive Loss Balancing in Deep Multitask Networks. arXiv:1711.02257v4
[41] Wang S., Teng Y., Perdikaris P. (2020) Understanding and mitigating gradient pathologies in physics-informed neural networks. arXiv:2001.04536v1