PHYSICS-INFORMED NEURAL NETWORKS AS SURROGATE MODELS IN SOLID MECHANICS

JAYA MANIDEEP REBBAGONDLA *

Abstract.
This article reviews the theoretical solid mechanics background and the corresponding practical implementation methodology of physics-informed neural networks (PINN) as surrogate models. Initially, the formulation of a given solid mechanics problem in two dimensional domain with boundary conditions into a mathematical format using the material constitutive properties is reviewed. The relevance of these material constitutive relationships as residual loss terms to drive the learning of the neural network parameters in the right direction, is discussed. Proceeding further, the details of the machine learning architecture for building the PINN are provided and several performance characteristics are analyzed. The optimization of the looping part of the code to cut down the runtime are also discussed. These neural network architectures are further augmented by using different sampling techniques and using a more conservative formulation of the boundary conditions of the solid mechanics problems. Finally a summary of the leanings from this case study and possible future directions of exploration are provided.

Key words. Physics Informed Neural Networks, Artificial Neural Networks, Solid Mechanics, Surrogate models, Linear Elasticity

1. Introduction. In the life of a modern day product designer, computational tools that provide the services of Computer Aided Designing and Computer Aided Evaluation are becoming ubiquitous and playing a significant role in the product design life cycle. In a typical product design cycle, the shape of the designs are initially generated keeping the aesthetics in mind, but with minimum supervision on their strength and performance aspect. These initially generated designs are updated, based on their computational and experimental evaluations by the specialists in respective fields, to optimize their performance. This cycle between the design and the analysis team continues until a satisfactory performance and aesthetic factors are achieved. But in the real world scenarios, these design update iterations might consume significant time and result in huge costs. Because of this reason, instead of completely ignoring the performance aspects till the design reaches the analyst, it would be advantageous to provide the designer with real time design feedback computational tools, even if they are of low fidelity. Given this motivation and with the explosive growth in the computational power at the hands of the designers, there is a significant amount of ongoing research in Surrogate models [3] [4]. In a typical surrogate model formulated using a machine learning architecture, the process of training involves the minimization of a specially formulated loss function. This loss function can either be based on the available output reference data at hand (the process of which is known as supervised learning) or can be learned from the physics of the problem - PINN. In physics informed models, the loss function is formulated using both the sparse data sampling and the physics of the problem. This type of formulation of Physics based models was first proposed by Raissi et al.[5] and later applied to solve PDEs over a wide range of domains, like in Solid Mechanics [1], Compressible flows [8] heat transfer [6]. Gaining popularity for these type of system formulations recently is majorly enabled by the recent advent in the ease of implementation of automatic differentiation through versatile code packages and also by the explosive growth in the availability of computational power.

*Massachusetts Institute of Technology (manideep@mit.edu, http://digitalstructures.mit.edu/page/about#rebbagondla).

Aimed with this background information in Physics informed models, we will proceed further to review the material I prepared as part of my learning process of these models. We will be majorly focusing on the linear elastic solid mechanics problems in two dimensional domain. Practically, Finite Element Analysis (FEA) is a major tool for structural analysis but in our case, to prevent any undesirable artifacts in the solutions formed because of the approximations in FEA, we will stick to the analytical solutions for comparison. For this purpose, we will initially review the material constitutive relationships and then discuss the solutions to linear elastic solid mechanics problems in some specially crafted problem set. Next, using the linear elastic material properties and relationships and the sparely sampled data from the known analytical solution, we will proceed to formulate the Physics Informed machine learning model and the corresponding loss function. This loss function includes terms based on the residuals of its physics based equations and the residuals of comparison of its results with the sparsely sampled data set. Equipped with this, understanding we will proceed to analyze various augmentations of the proposed model and their effects on the results. Especially we will look at the effects of using different sizes of the neural networks, learning rates, the strategy of sampling the analytical solution domain for results comparison etc., In addition to this, we will also consider the hard boundary conditions strategy to make sure that the required characteristics of the solutions are maintained at the boundaries. Finally, based on the results form this study future directions of explorations are also proposed.

. Rest of the paper carries out the description of these techniques as follows. The mathematical formulations of the solid mechanics problem using SIMP material modeling and the related optimization problem are discussed in Section 2. The nuances of MMA optimization algorithm and relevance of Lagrangian duality are discussed in Section **??**. The details of the results obtained from these implementations are discussed in Section **??** and the concluding remarks and future goals are presented in Section **??**.

2. Elasticity equations, boundary conditions and problem formulation.

$$\nabla.\sigma + f = 0$$

$$\implies \sigma_{xx,x} + \sigma_{xy,y} + f_x = 0 \tag{2.1}$$

$$\implies \sigma_{xy,x} + \sigma_{yy,y} + f_y = 0 \tag{2.2}$$

$$\varepsilon = (\nabla u + (\nabla u)^T)/2$$

$$\implies \varepsilon_{xx} = \frac{\partial u_x}{\partial x} \tag{2.3}$$

$$\implies \varepsilon_{yy} = \frac{\partial u_y}{\partial y} \tag{2.4}$$

$$\implies \varepsilon_{xy} = \frac{1}{2}\Big[\frac{\partial u_y}{\partial x} + \frac{\partial u_x}{\partial y}\Big] \tag{2.5}$$

$$\sigma = C\varepsilon$$

$$\implies \sigma_{xx} = (\lambda + 2\mu)\varepsilon_{xx} + \lambda\varepsilon_{yy} \tag{2.6}$$

$$\implies \sigma_{yy} = (\lambda + 2\mu)\varepsilon_{yy} + \lambda\varepsilon_{xx} \tag{2.7}$$

$$\implies \sigma_{xy} = 2\mu\varepsilon_{xy} \tag{2.8}$$

In this section we will review the methodology of formulating the linear elastic solid mechanics problem and methodology of dealing with the boundary conditions.

The behaviour of linear elastic materials in a 2 dimensional domain is given by the equations eq. (2.1 - 2.9). Where the definitions of the symbols are as shown below:

$\sigma_{xx}, \sigma_{yy}, \sigma_{xy}$	- Components of the stress tensor in 2D
$\varepsilon_{xx}, \varepsilon_{yy}, \varepsilon_{xy}$	- Components of the strain tensor in 2D
u_x, u_y	- Components of the displacement vector in 2D
f_x, f_y	- Components of the body force vector in 2D
λ, μ	- Material constitutive properties

For a given set of boundary conditions, these differential equations in the 2D domain are used to evaluate the required parameters. In practical applications, the domain is discretized into finite elements and these variables are approximated using interpolating functions and nodal values. But these type of discretizations might introduce some undesirable artifacts and skew the machine learning algorithms in the wrong direction. So in this project, we will be using a crafted problem statement to which a solution already exists.This problem statement is picked from Ref. [3] The domain of this problem is represented as shown in Img. 1

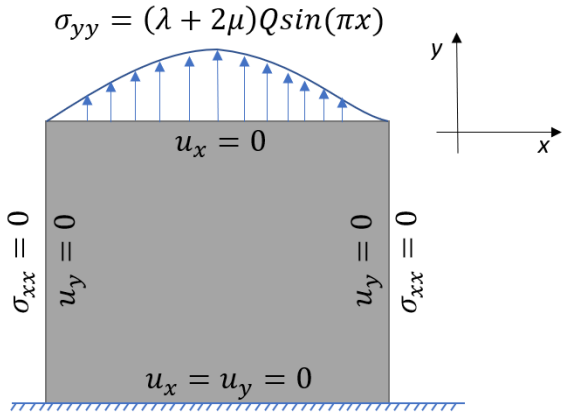

FIG. 1. *2D problem domain with the boundary conditions*

The 2D domain here is made of a unit sided square with the boundary conditions as indicated in the Img. 1.The body forces exerted per unit area of the domain are given by 2.9.

$$
\begin{aligned}
f_x =& \lambda(4\pi^2 \cos(2\pi x)\sin(\pi y) - \pi\cos(\pi x)Qy^3) \\
&+ \mu(9\pi^2 \cos(2\pi x)\sin(\pi y) - \pi\cos(\pi x)Qy^3) \\
f_y =& \lambda(-3\sin(\pi x)Qy^2 + 2\pi^2 \sin(2\pi x)\cos(\pi y)) + \mu(-6\sin(\pi x)Qy^2 \\
&+ 2\pi^2 \sin(2\pi x)\cos(\pi x) + \pi^2 \sin(\pi x)Qy^4/4)
\end{aligned} \tag{2.9}
$$

The analytical solution to this problem statement is given by:

$$u_x(x,y) = \cos(2\pi x)\sin(\pi y) \tag{2.10}$$

$$u_y(x,y) = \frac{\sin(\pi x) Q y^4}{4}$$

(2.11)

Where we assume the parameter values: $\lambda = 1, \mu = 0.5, Q = 4$.With these equations, we will be able to derive the fields of displacement, stress and strain throughout the problem domain. The mappings related to this are provided in Img. **??**. Now that we have a complete picture of the required fields and their relationships across the domain, lets proceed with the discussion on the architecture of the Physics Informed Neural Network.

3. Basic Machine Learning framework. As we have seen before, the field variables in this problem statement are: $\sigma_{xx}, \sigma_{yy}, \sigma_{xy}, u_x, u_y$. While the strain parameters $\varepsilon_{xx}, \varepsilon_{yy}, \varepsilon_{xy}$ can be obtained from the previously indicated equations. There are several possible variations in choosing inputs and outputs of the network. A first option can be to take the location variables as input x, y and provide the displacement results u_x, u_y. Remaining stress and strain parameters are obtained from these results. But the issue with this formulation is that the displacements are related to the stress values through a double derivative. The derivative of this double derivative term will be required for the machine learning parameter optimizations. So we take a step back and assume that the outputs of the network are; $\sigma_{xx}, \sigma_{yy}, \sigma_{xy}, u_x, u_y$. This way we need not have any double derivative terms. Another advantage of having the location co-ordinates as inputs is that the derivatives in the equations can be easily obtained from the backward passes in Neural networks. So in summary, we choose a single Neural Network which takes in two inputs u_x, u_y and provides 5 outputs $\sigma_{xx}, \sigma_{yy}, \sigma_{xy}, u_x, u_y$ as shown in Img. 2

Now that we have created out machine learning network, lets look at the loss function formulation. Generally the loss function is given by the comparison of the data points as shown in eq. 3.1. But in case of Physics Informed Neural Networks, we have the additional terms of loss as shown in 3.2. Each of these terms should ideally be zero but when represented using a neural network, they do have some variations which will be reduced during the optimization iterations to find the parameters of the neural network. Given these equations, the total loss is given by the sum of two loss terms.

$$\begin{aligned} \Gamma_{comp} =& |u_x - u_x*| + |u_y - u_y*| + |\sigma_{xx} - \sigma_{xx}*| \\ &+ |\sigma_{yy} - \sigma_{yy}*| + |\sigma_{xy} - \sigma_{xy}*| \end{aligned} \tag{3.1}$$

$$\begin{aligned} \Gamma_{phys} =& |\sigma_{xx,x} + \sigma_{xy,y} + f_x| + |\sigma_{xy,x} + \sigma_{yy,y} + f_y| \\ &+ |(\lambda + 2\mu)\varepsilon_{xx} + \lambda\varepsilon_{yy} - \sigma_{xx}| + |(\lambda + 2\mu)\varepsilon_{yy} + \lambda\varepsilon_{xx} - \sigma_{yy}| \\ &+ |2\mu\varepsilon_{xy} - \sigma_{xy}| \end{aligned} \tag{3.2}$$

$$\Gamma_{tot} = \Gamma_{comp} + \Gamma_{phys} \tag{3.3}$$

It can be observed that there are derivative terms in these residuals $\sigma_{yy,y}, \sigma_{xy,y}$, $\sigma_{xy,x}, \sigma_{xx,x}$. These terms are obtained using automatic differentiation packages of Julia.

4. Results and discussion. The machine learning network and the loss function are modelled as shown in the previous section. Inorder to access the sampling

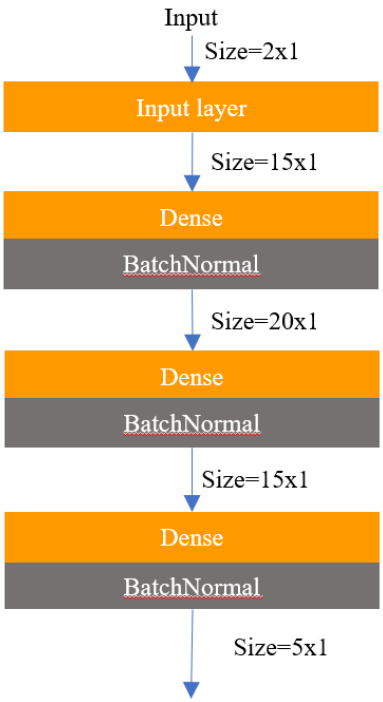

FIG. 2. *Reference problem*

points, the domain is sampled at 400 points uniformly across the domain as shown in Fig. 3.

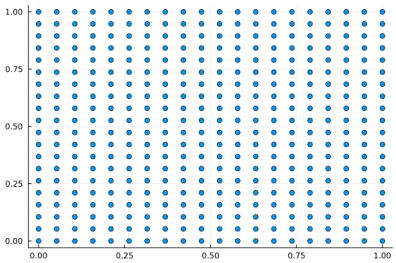

FIG. 3. *Reference problem*

This network is trained on the 400 sampling points and the corresponding physics loss terms and after training for 1000 epochs, the loss convergence is as shown in Img. img:bscconv, though there is convergence, the value of the error is still high. We can look at some sample field comparisons as shown in Img 5

As shown in the images, though the predictions are similar, there are several regions of error where the stress and the displacement terms differ widely from the sampled actual field.An increase in the learning rate resulted in very high fluctuations in the values (order of 10^{40} and a decrease is reducing the convergence rate. So I abandoned the idea of tweaking the learning rate and moving forward with other modifications. A change in the network size to increase the intermediate layer size

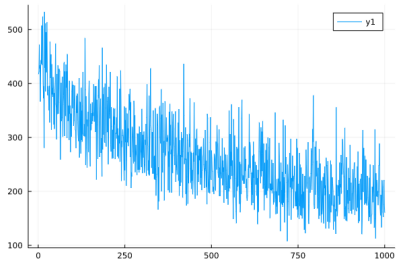

FIG. 4. *Reference problem*

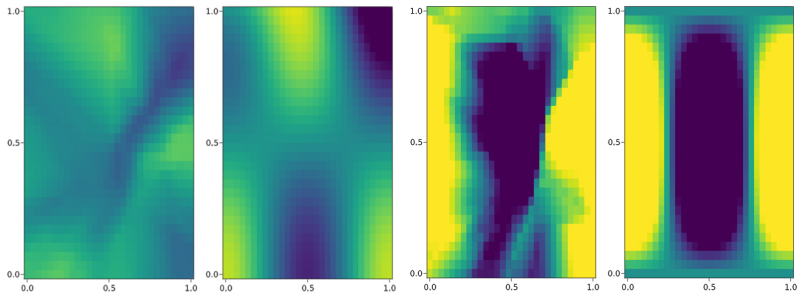

FIG. 5. *From left prediction σ_{xy}, true σ_{xy}, predicted u_x, true u_x*

to 30 still resulted in a high loss value of ≈ 100. the following are the obtained in fig:6.Additionally for these results, the number of sampling points are increased to 900. This results in a better representation of the fields as shown in the images. but the error is still high.

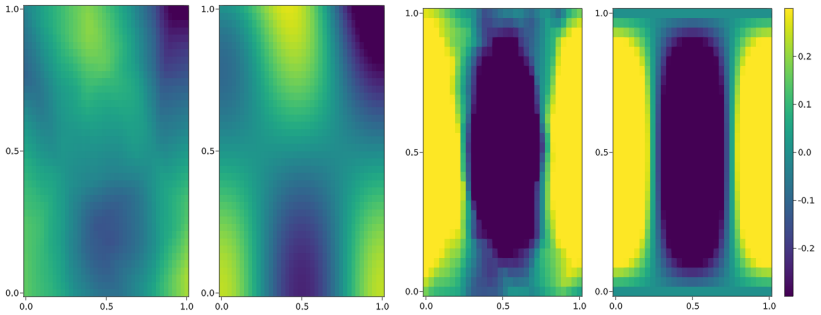

FIG. 6. *From left prediction σ_{xy}, true σ_{xy}, predicted u_x, true u_x*

4.1. Enforcing hard boundary conditions. As part of the next modification, I looked at modifying the results from the neural network, We know that the analytical equations always satisfy the boundary conditions shown in Img. 1 but that is not the case for the predicted results. So as per the ref. **??** we can strictly impose the boundary conditions using the following formula:

$$N_{ihard}(x,y) = N_i(x,y) * D_i(x,y) + s_i(x,y) \tag{4.1}$$

Where $N_{ihard}(x, y)$ is the resulting function with the hard boundary conditions enforced. $N_i(x, y)$ is the prediction from the existing neural network.$D_i(x, y)$ is the shortest distance from the corresponding boundary, $s_i(x, y)$ is the corresponding boundary condition value. The same is implemented in Julia and the results are shown in Fig 7 and 8

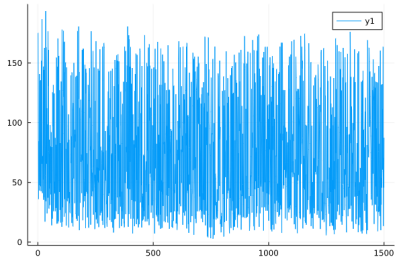

FIG. 7. *Reference problem*

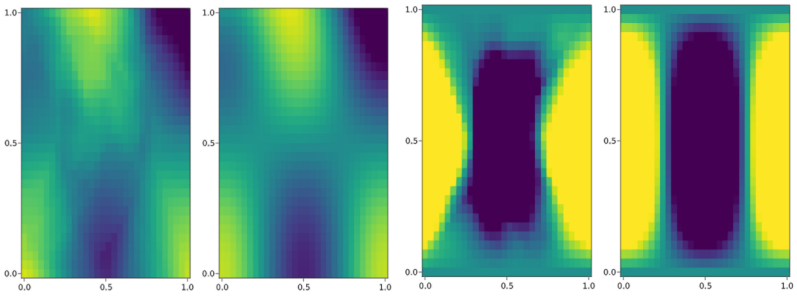

FIG. 8. *From left prediction σ_{xy}, true σ_{xy}, predicted u_x, true u_x*

Next, I tried to modify the sampling method of the domain from a uniform to a random manner as shown in Fig. 9. But this resulted in a highly diverging loss to Inf. This sampling technique forces the model to learn the distribution in harder way, since there is almost no same x or y for any of the points, which is not the case for uniform sampling.

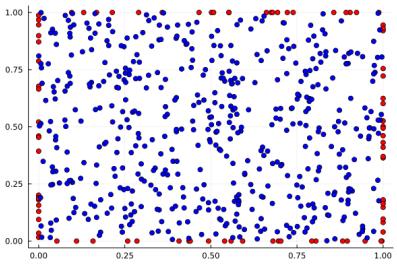

FIG. 9. *Reference problem*

4.2. Coding efficiency:. Inorder to reduce the run time of the code, inplace modifications of vectors are used as much as possible. But there was a major runtime difference when all the variables are maintained as Float32 type. This gave us a

reduction from time consumption=35min to time consumption=20. These results need more analysis and model tuning to become better at approximating the original fields.

5. Learning, further plans, conclusions. This work on Physics informed neural networks is really interesting to me, and since now that I got a basic version of it implemented and running, moving forward I would like to further analyze even complex machine learning architectures. This reference [7], highlights the issues with the existing PINN framework and propose upgrades that can significantly improve the performance. I would like implement them for the current architecture. Additionally, I would like to use a separate network for each of the outputs to understand how it effects the results. Another possible direction of exploration is the weighted sum of each of the loss function terms instead of unweighted sum. This might help in guiding the gradient descent in the better direction. Finally I would like to incorporate the domain dependency of the results using a similar technique as shown in [2]. As for the current implementations, my learnings are that, the learning rate significantly affects the results and need to be tuned properly, as the n-dimensional loss surface complexity increases, with increase in the number of variable parameters of the network, it gets difficult for the loss to converge to a specific value. Once all these learnings and proposals are implemented the random sampling technique would ideally be the best way to assess the overfitting property of the model.

REFERENCES

[1] Y. L. Chengping Rao, Hao Sun, *Physics-Informed Deep Learning for Computational Elastodynamics without Labeled Data*, Journal of Engineering Mechanics, https://doi.org/10.1061/(ASCE)EM.1943-7889.0001947, 2021.

[2] A. M. H. G. R. J. E. Haghighat, M. Raissi, *A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics*, Computer Methods in Applied Mechanics and Engineering, https://doi.org/10.1016/j.cma.2021.113741, 2021.

[3] A. M. H. G. R. J. Ehsan Haghighat, Maziar Raissi, *A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics*, Computer Methods in Applied Mechanics and Engineering, https://doi.org/10.1016/j.cma.2021.113741, 2021.

[4] P. O. R. K. Johannes G. Hoffer, Bernhard C. Geiger, *Mesh-Free Surrogate Models for Structural Mechanic FEM Simulation: A Comparative Study of Approaches*, Applied Sciences, https://www.mdpi.com/2076-3417/11/20/9411, 2019.

[5] G. K. M. Raissi, P. Perdikaris, *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*, Journal of Computational Physics, https://doi.org/10.1016/j.jcp.2018.10.045, 2019.

[6] S. W. P. P. G. E. K. Shengze Cai, Zhicheng Wang, *Physics-Informed Neural Networks for Heat Transfer Problems*, journal of Heat Transfer, https://doi.org/10.1115/1.4050542, 2021.

[7] P. P. Sifan Wang, Xinling Yu, *When and why PINNs fail to train: A neural tangent kernel perspective*, Journal of Computational Physics, https://doi.org/10.1016/j.jcp.2021.110768, 2022.

[8] G. E. K. Zhiping Mao, Ameya D. Jagtap, *Physics-informed neural networks for high-speed flows*, Computer Methods in Applied Mechanics and Engineering, https://doi.org/10.1016/j.cma.2019.112789, 2020.