Using physics-informed regularization to improve extrapolation capabilities of neural networks

David Davini
University of California, Los Angeles

Bhargav Samineni
New Jersey Institute of Technology

Benjamin Thomas
Louisiana State University

Amelia Huong Tran
Mount Holyoke College

Cherlin Zhu
Johns Hopkins University

Kyung Ha
University of California, Los Angeles

Ganesh Dasika
Advanced Micro Devices, Inc.

Laurent White
Advanced Micro Devices, Inc.
laurent.white@amd.com

Abstract

Neural-network-based surrogate models, which replace (parts of) a physics-based simulator, are attractive for their efficiency, yet they suffer from a lack of extrapolation capability. Focusing on the wave equation, we investigate the use of several physics-based regularization terms in the loss function as a way to increase the extrapolation accuracy, together with assessing the impact of a term that conditions the neural network to weakly satisfy the boundary conditions. These regularization terms do not require any labeled data. By gradually incorporating the regularization terms while training, we achieve a more than $5\times$ reduction in extrapolation error compared to a baseline (i.e., physics-less) neural network that is trained with the same set of labeled data. We map out future research directions, and provide some insights about leveraging the trained neural-network state for devising sampling strategies.

1 Introduction

Scientific computing has benefited from decades of progress in numerical algorithms and advances in hardware performance [1]. This evolution enters a new chapter as large-scale scientific computing applications face many challenges related to the slowdown of Moore's law [7], the pursuit of extreme parallelism [5], and the quest for higher energy efficiency [4]. Assuming the use of existing computing technology, high-performance computers of prohibitive dimension and cost would be required to meet society's requirements for more accurate and reliable computational models. While the development of new computing devices will address some of these challenges [14], a step change in numerical algorithms may be necessary. Inspired by rapid progress in applying deep learning for cognitive tasks [6], there is increasing hope and expectation that neural networks (NN) can be leveraged to improve the speed, accuracy, and/or energy efficiency of scientific computations. This endeavor is a subset of the broader *AI for Science* (or *Scientific Machine Learning*) discipline [13].

NN-based surrogate models, whereby (part of) a physics-based simulation is replaced by a neural network prediction, are attractive because their time-to-solution can sometimes be orders of magnitude lower than physics-based algorithms [3]. NN-based surrogate models also tend to be more flexible

Fourth Workshop on Machine Learning and the Physical Sciences (NeurIPS 2021).

and to handle nonlinear transformations better than more traditional model-reduction techniques, such as the proper orthogonal decomposition method [8]. Not surprisingly, neural networks are especially effective at predicting solutions in problem configurations for which the neural network was trained (i.e., neural networks interpolate well) [2]. Issues arise, and mitigation strategies need to be implemented, when neural networks are used to make predictions based on input data that lie outside of the range of the dataset used for optimizing the network parameters (i.e., neural networks extrapolate poorly). Mitigation strategies such as training on datasets of larger range and retraining on the fly when encountering new input data increase the training cost, thereby making this cost less amortizable and decreasing the value proposition of using neural networks in the first place.

Recognizing the value of physics-based modeling in scientific computing [15], the emerging field of physics-informed neural networks seeks to optimize the NN parameters subject to constraints imposed by the physics of the problem [11]. Here, we adopt the method in which the constraint takes the form of a regularization term in the objective function. That regularization term, which is derived from equations underlying the problem, imposes a penalty on the NN parameters and steers the neural network to behave in closer agreement to the physics. In particular, we investigate the use of multiple types of physics-informed regularization terms for the acoustic wave equation and show that by adding the proper regularization terms, the extrapolation error decreases by more than $5\times$ compared to using a physics-less NN.

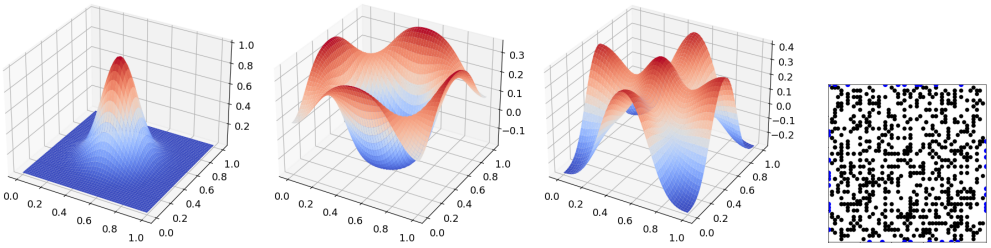
2 Physics-based regularization for the wave equation

The wave equation is at the core of many scientific applications, such as earthquake modeling, acoustics, hydrocarbon exploration, and stealth aircraft design. Traditional numerical methods are effective at solving the wave equation, yet they remain expensive, especially when considering the requirement of solving the equation over an entire domain while typically only needing time series at a few locations. It is conceivable that a well-trained neural network, only taking a spatial location and a time stamp as inputs, would be very advantageous from a time-to-solution perspective. While there has been some research on using NNs to approximate wave propagation [10, 9], very limited work has focused on using physics-informed NNs (PINNs) to improve the extrapolation accuracy in the presence of boundaries and by conditioning the NN with boundary conditions, which is the focus of this section, together with an investigation of various physics-based regularization terms.

We consider the two-dimensional acoustic wave equation

$$\frac{\partial p}{\partial t} + \kappa \nabla \cdot \mathbf{v} = 0,$$
$$\frac{\partial \mathbf{v}}{\partial t} + \frac{1}{\rho} \nabla p = 0,$$

along with reflecting boundary conditions $\mathbf{v} \cdot \hat{n} = 0$, where p and $\mathbf{v}$ are pressure and velocity, respectively, and $\hat{n}$ is the normal to the boundary. We assume unit values for the bulk modulus of compressibility, κ, and density, ρ, in our experiments. We solve the wave equation as a system of first-order partial differential equations in our simulator based on the discontinuous Galerkin method, which is used for generating training data for $(\mathbf{x}, t) \in [0, 1]^2 \times [0, 1]$. Our intent is to train, and to adequately regularize, a neural network based on simulation data acquired every 10^{th} time step for $t \in [0, 1]$ (time step is 0.001) and to assess the ability of the NN to accurately predict the solution for $t \in (1, 2]$. Instead of using all simulation samples in space for a given time snapshot, we randomly select 1% of the discretization points in $(0, 1)^2$ and 10% of the points along the domain boundary, as illustrated in Figure 1. The test set is built from sampling 5% of simulation data in $(0, 1)^2$ every 10^{th} time step. We consider a fixed network architecture (dense NN with five 100-wide hidden layers and $\tanh$ activation) in all experiments presented below. All NNs are trained with the same set of labeled data for $t \leq 1$, as described above. Other training parameters are provided in Section A.1. Differences in performance stem from whether regularization is introduced or not, and its type. The loss function consists of the mean square error (MSE) between the simulation output $(p, \mathbf{v})$ and the NN prediction of these state variables, $(\hat{p}, \hat{\mathbf{v}})$, for each input $(\mathbf{x}, t)$. For all regularization-based training procedures, the loss function is augmented with a physics-based penalty term, E_r, which can take various forms as presented in Table 1. We also add a term, E_b, that conditions the neural network to satisfy the boundary conditions: $\mathbf{v} \cdot \hat{n} = 0$. Note that this conditioning is only applied when using the first-order system as regularizer, which is consistent with the underlying mathematical model.



(a) Simulation (t = 0.0) (b) Simulation (t = 0.4) (c) Simulation (t = 1.0) (d) Sampling

Figure 1: (a-c) Simulation snapshots and (d) example of data collection for t=0.56, where black and blue markers denote interior and boundary data samples, respectively.

The total loss function L can be written as:

$$L = MSE + \lambda_r E_r + \lambda_b E_b, \tag{1}$$

where λ_r and λ_b allow for varying the strength of regularization and conditioning, which are gradually introduced in later stages of training. Interpolation and extrapolation errors are reported in Table 1. As

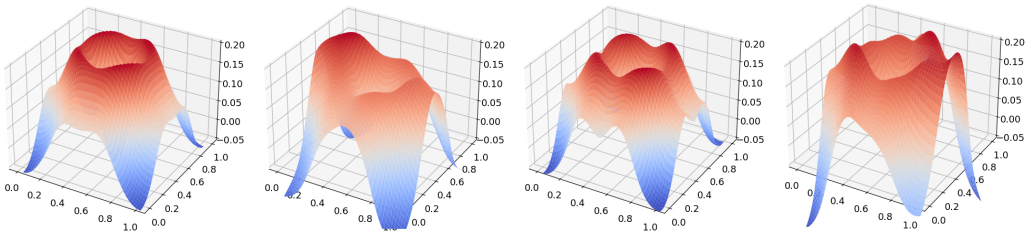
Table 1: Regularization type and prediction errors (RMS of difference between prediction and 5% of simulation data, randomly sampled in space every 10^{th} step). Interpolation error for $t \in [0, 1]$. Extrapolation error for $t \in (1, 2]$. Average of 5 runs (standard deviation between parentheses).

Name	E_r	E_b	**Interp. error**	**Extrap. error**
Baseline NN	None	None	1.8×10^{-3} (43%)	1.8×10^{-1} (16%)
PINN 1st	$\lvert\hat{p}_t + \nabla \cdot \hat{\mathbf{v}}\rvert + \lVert\hat{\mathbf{v}}_t + \nabla\hat{p}\rVert$	$\lVert\hat{\mathbf{v}}_n\rVert$	7.3×10^{-3} (9%)	3.1×10^{-2} (35%)
PINN 2nd	$\lvert\hat{p}_{tt} - \nabla^2\hat{p}\rvert$	None	1.8×10^{-2} (7%)	1.9×10^{-1} (25%)

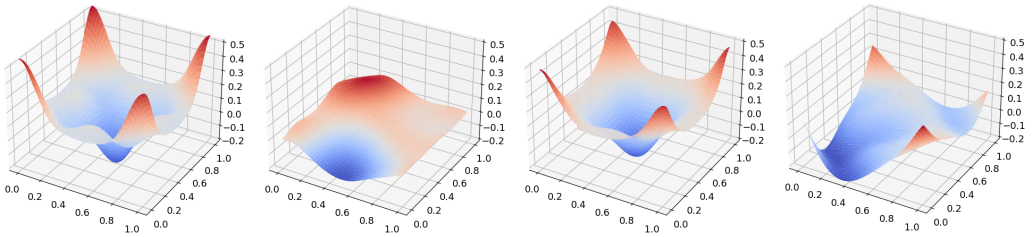
expected, the baseline NN (without any regularization) makes accurate predictions in the interpolation domain, exhibiting the ability to reproduce the solution of the wave equation as it is given in the form of labeled data. The extrapolation error is much higher, which is also visible in Figure 2. Adding first-order regularization and boundary conditioning decreases the extrapolation error by more than $5\times$, to the detriment of a slight (though acceptable) loss of accuracy in the interpolation region. This compromise could be mitigated by using a larger NN (not tried). We note the inability of the second-order regularizer to improve the extrapolation accuracy. In an attempt to isolate the effect of regularization on prediction accuracy, we adopted the same training strategy for all neural networks (except for the regularization term). To that effect, we believe that some of the errors in Table 1 could be reduced by training differently and/or by using different neural architectures, all of which is ongoing work.

3 Future work

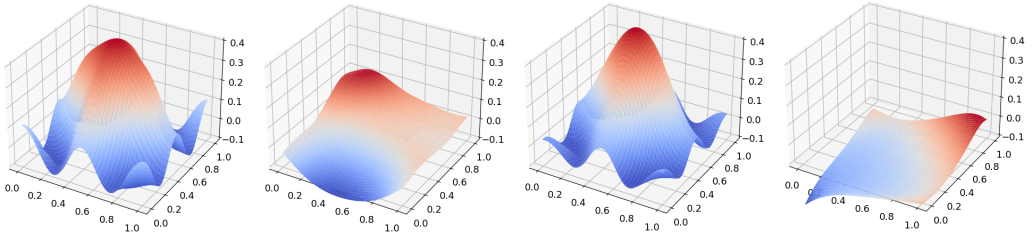
An appropriate physics-based regularization forces a shift in the NN's predictive power from the interpolation region to the extrapolation region. While it is clear that the shift occurs through changing the neural network weights, the nature of the change is not clear. Insights in this area could make neural network extrapolation more accessible and deliberate. Most of the complex behavior of a NN's activation functions occurs around the y-axis, that is, in the active region. Inputs far outside of the active region are saturated and their activation is insensitive to small changes, making the NN less effective at modeling complex behavior. We propose to use saturation as a measure of a NN's ability to properly extrapolate. We simply define saturation of a neuron $\mu_n(z)$ through an indicator function: $\mu_n = 1$ if $|z| > k$ and $\mu_n = 0$ if $|z| \leq k$, for a given activation input z and threshold k (other metrics can be used [12]). Layerwise saturation is the average of neuronal saturations and lies between 0 and 1. We consider a simple two-dimensional paraboloid $x^2 + y^2$ as the target function to extrapolate and train NNs with labeled data gathered on $[-1, 1]^2$. One NN is trained without



(a) Simulation (t = 1.2) (b) Baseline NN (t = 1.2) (c) PINN 1st (t = 1.2) (d) PINN 2nd (t = 1.2)



(e) Simulation (t = 1.6) (f) Baseline NN (t = 1.6) (g) PINN 1st (t = 1.6) (h) PINN 2nd (t = 1.6)



(i) Simulation (t = 2.0) (j) Baseline NN (t = 2.0) (k) PINN 1st (t = 2.0) (l) PINN 2nd (t = 2.0)

Figure 2: Comparison of extrapolation performance between simulation (left column) and predictions from neural networks trained with different regularization strategies (as described in Table 1).

regularization (baseline NN), whereas training of the second NN (PINN) is regularized via penalizing the variance of second-order derivatives on $[-2, 2]^2$ (in addition to using labeled data on $[-1, 1]^2$). Layerwise saturation as a function of space is shown in Figure 3. In particular, we show the difference in layer saturation between the PINN and the baseline NN, with negative values indicating areas where the PINN is less saturated (and more effective) than the baseline NN. The maps of saturation difference in the first two layers deserves special attention, as they confirm the PINN's behavior observed for the wave extrapolation problem. Specifically, the tendency of the PINN to relax the accuracy requirement in the interpolation region so as to be able to increase the accuracy in the extrapolation region. This behavior, which appears to be target-function-agnostic, could serve as the basis for building a posteriori error estimates or devising more efficient sampling strategies.

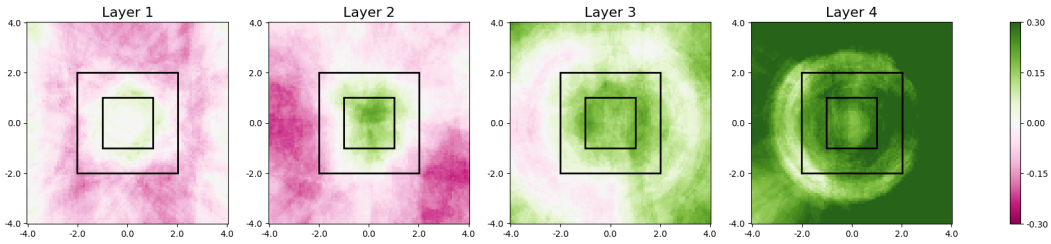


Figure 3: Difference between PINN saturation and baseline NN saturation, for each layer (average of five trained models).

4 Summary

Preliminary results demonstrate the importance of not only incorporating physics-based regularization that is consistent with the mathematical model being used to generated labeled data, but to also incorporate consistent boundary conditioning. It is unclear, however, how to translate those requirements for field or lab labeled data. We propose to use and to continue investigating input saturation as an indirect measure of a NN's extrapolative power as surrogate models. The difference in saturation between a physics-less NN and a PINN is consistent with the shift in experimental errors, which strengthens the case that saturation-based metrics could be developed into a posteriori *and* function-agnostic error estimates, and used to devise smarter sampling strategies.

5 Broader impact

Neural networks as surrogate models in the physical sciences continue to make progress, with potential sizable impact in our ability to make much faster and more energy-efficient, even if less accurate, predictions of physical processes. Areas such as sensitivity analysis (where parameter sweep is used) and optimization (where the forward model is run many times and high accuracy of the forward model is not always required) could see large benefits if used carefully. Our work leverages past work on physics-informed neural networks and brings additional insight into the type of regularization that is most effective. While focusing on the wave equation, our findings should be transferable to other equations. Saturation-based a posteriori error analysis should be further developed and could be used for devising sampling strategies to reduce extrapolation errors in target regions. Since this metric is function-agnostic, its impact could be broader than what we presented.

Acknowledgments and Disclosure of Funding

This work was conducted during the Research in Industrial Projects for Students (RIPS) program, hosted at the University of California (Los Angeles)'s Institute for Pure and Applied Mathematics, from June 21 to August 20, and under AMD's mentorship. The first five authors were undergraduate students at the time of the summer program and contributed equally.

References

[1] P. Bauer, A. Thorpe, and G. Brunet. The quiet revolution of numerical weather prediction. *Nature*, 525:47–55, 2015.

[2] F. Brockherde, L. Vogt, L. Li, M. E. Tuckerman, K. Burke, and K.-R. Müller. Bypassing the kohn-sham equations with machine learning. *Nature Communication*, 8(872), 2017.

[3] W. Jia, H. Wang, M. Chen, D. Lu, L. Lin, R. Car, W. E, and L. Zhang. Pushing the limit of molecular dynamics with ab initio accuracy to 100 million atoms with machine learning. In *SC '20: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–14, 2020.

[4] S. Kamil, J. Shalf, and E. Strohmaier. Power efficiency in high performance computing. In *2008 IEEE International Symposium on Parallel and Distributed Processing*, pages 1–8, 2008.

[5] C. Kato, Y. Yamade, K. Nagano, K. Kumahata, K. Minami, and T. Nishikawa. Toward realization of numerical towing-tank tests by wall-resolved large eddy simulation based on 32 billion grid finite-element computation. In *SC '20: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–13, 2020.

[6] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[7] C. E. Leiserson, N. C. Thompson, J. S. Emer, B. C. Kuszmaul, B. W. Lampson, D. Sanchez, and T. B. Schardl. There's plenty of room at the top: what will drive computer performance after moore's law? *Science*, 368, 2020.

[8] R. Maulik, B. Lusch, and P. Balaprakash. Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders. *Physics of Fluids*, 33(3):037106, 2021.

[9] B. Moseley, A. Markham, and T. Nissen-Meyer. Solving the wave equation with physics-informed deep learning. *arXiv*, 2006.11894, 2020.

[10] B. Moseley, T. Nissen-Meyer, and A. Markham. Deep learning for fast simulation of seismic waves in complex media. *Solid Earth*, 11:1527–1549, 2020.

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

[12] A. Rakitianskaia and A. Engelbrecht. Measuring saturation in neural networks. In *2015 IEEE Symposium Series on Computational Intelligence*, pages 1423–1430, 2015.

[13] R. Stevens, V. Taylor, J. Nichols, A. B. MacCabe, K. Yelick, and D. Brown. AI for Science. Technical report.

[14] N. C. Thompson and S. Spanuth. The decline of computers as a general purpose technology. *Communications of the ACM*, 64:64–72, 2021.

[15] K. E. Willcox, O. Ghattas, and P. Heimbach. The imperative of physics-based modeling and inverse theory in computational science. *Nature Computational Science*, 1:166–168, 2021.

A Appendix

A.1 Neural network training for the wave problem

All neural networks for the wave problem were trained with the Adam optimizer for 2500 epochs and a batch size of 512. The learning rate $\alpha(\epsilon)$ depends on the epoch number, ϵ, as follows:

$$\alpha(\epsilon) = \begin{cases} \alpha_0 & \text{if } \epsilon < \mathcal{E}_0 \\ (\alpha_0 - \alpha_1)\frac{\epsilon - \mathcal{E}_0}{\mathcal{E}_1 - \mathcal{E}_0} + \alpha_1 & \text{if } \mathcal{E}_0 \leq \epsilon \leq \mathcal{E}_1 \\ \alpha_1 & \text{if } \mathcal{E}_1 < \epsilon \end{cases},$$

where $\alpha_0 = 1.0 \times 10^{-3}$ and $\alpha_1 = 5.0 \times 10^{-4}$, and $\mathcal{E}_0 = 400$, $\mathcal{E}_1 = 2250$. The regularization parameters, λ_r and λ_b in Eq. (1), take on values according to the following schedule:

$$\lambda_{r,b} = \begin{cases} \lambda_0 & \text{if } \epsilon < \mathcal{E}_0 \\ (\lambda_0 - \lambda_1)\frac{\epsilon - \mathcal{E}_0}{\mathcal{E}_1 - \mathcal{E}_0} + \lambda_1 & \text{if } \mathcal{E}_0 \leq \epsilon \leq \mathcal{E}_1 \\ \lambda_1 & \text{if } \mathcal{E}_1 < \epsilon \end{cases},$$

where $\lambda_0 = 0.0$ and $\lambda_1 = 0.055$.

Forward model runs (to produce training data) and training of the neural networks were all conducted on a workstation-class 24-core CPU.