Uncertainty quantification methods for ML-based surrogate models of scientific applications

Kishore Basu
University of Toronto

Judy Hao
Emory University

Delphine Hintz
Bethany Lutheran College

Dev Shah
University of Manchester

Aaron Palmer
University of California, Los Angeles

Gurpreet Singh Hora
Advanced Micro Devices, Inc.

Darian Nwankwo
Advanced Micro Devices, Inc.

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

Abstract

In recent years, there has been growing interest in using machine-learning algorithms to assist classical numerical methods for scientific computations, as this data-driven approach could reduce computational cost. While faster execution is attractive, accuracy should be preserved. Perhaps more importantly, our ability to identify when a given machine-learning surrogate is not reliable should make their application more robust. We aim to quantify the uncertainty of predictions through the application of Bayesian and ensemble methods. We apply these methods to approximate a paraboloid and then the solution to the wave equation with both standard neural networks and physics-informed neural networks. We demonstrate that the embedding of physics information in neural networks reduces the model uncertainty while improving the accuracy. Between the two uncertainty quantification methods, our results show that the Bayesian neural networks render overconfident results while model outputs from a well-constructed ensemble are appropriately conservative.

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 [8], the pursuit of extreme parallelism [6], and the quest for higher energy efficiency [5]. Assuming the use of existing computing technology, high-performance computers of prohibitive specifications 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 [12], a step change in numerical algorithms may be necessary. Inspired by rapid progress in applying deep learning for cognitive tasks

Machine Learning and the Physical Sciences workshop, NeurIPS 2022.

[7], 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 [11].

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 [4]. NN-based surrogate models also tend to be more flexible and to handle nonlinear transformations better than more traditional model-reduction techniques, such as the proper orthogonal decomposition method [9]. 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). One mitigation strategy consists in retraining the network on the fly when the network's predictions become unreliable, which calls for developing methods to quantify the uncertainty around these predictions. If successful, such uncertainty-quantification methods would enable the automation of complex workflows that combine physics-based models and neural-network surrogates used for acceleration.

We focus on two techniques to quantify the model uncertainties: Bayesian neural networks and ensembles. Bayesian neural networks (BNNs) differ from their classical counterparts by training probability distributions for the network parameters instead of deterministic values [13]. Given a unique input, BNN inferences (predictions) will vary. The spread among these predictions indicates the model's uncertainty around the mean, which is typically driven by insufficient data. Ensemble methods consist in training different networks on the same data and in using the spread across these networks' predictions to assess the uncertainty around the mean prediction. A simple steady-state two-dimensional problem is first considered as a way to clearly expose the methodology. We then turn to the wave equation. We seek to quantify the uncertainty of predictions obtained from standard neural networks and from physics-informed neural networks (PINNs)[10], using Bayesian and ensemble methods.

2 Steady-state two-dimensional problem

We consider the two-dimensional paraboloid $f = x^2 + y^2$ defined for $(x, y) \in \Omega = [-2, 2] \times [-2, 2]$. We further consider a split of this domain into an interpolation region $\Omega_i = [-1, 1] \times [-1, 1]$ and an extrapolation region Ω_e in the remainder of the domain, that is, $\Omega_e = \Omega \setminus \Omega_i$. Labeled data is only generated in Ω_i where 2500 points are randomly sampled in space. We seek to investigate the performance of several neural-network methodologies in the extrapolation region, with a focus on each methodology's ability in quantifying the uncertainty. When training physics-informed neural networks, an additional 7500 (x, y) locations are randomly selected in Ω_e to compute the physics-based regularization term that arises in the loss function. In the case of the paraboloid, we choose a penalization term of the form $\mathcal{V}(\hat{f}_{xx}) + \mathcal{V}(\hat{f}_{xy}) + \mathcal{V}(\hat{f}_{yy})$, where $\mathcal{V}$ is the variance and $\hat{f}$ is the network's prediction. This penalization term consists in weakly enforcing the second derivatives of the NN's prediction $\hat{f}$ with respect to the input (x, y) to be zero. We expect, as will be shown, the PINN to outperform the standard NN in the extrapolation region. To assess the uncertainty in Ω_e, we construct Bayesian versions of these two networks, respectively called BNN and BPINN. We also consider ensembles of these neural networks: an ensemble of (standard) NNs and an ensemble of PINNs. These four approaches are summarized in Table 1, where they are contrasted by their architecture and training procedure. Notably, training physics-informed neural networks takes longer because the learning rate is smaller, but also because more data is used (labeled data *and* physical locations where penalization is enforced). A smaller learning rate is required to accommodate the added complexity incurred by adding the physics-informed regularization term. Networks in the ensembles differ by their architecture, namely the number of hidden layers and their width, while the training procedure is the same for all ensemble members.

We estimate the uncertainty around predictions by performing 30 inferences in the Bayesian approach and by constructing 30-member ensembles (and so, performing 30 inferences). We experimentally observed that the standard deviation of predictions exhibited very little change beyond 30 inferences in both the Bayesian and the ensemble approaches. Figure 1 summarizes the results obtained by performing these 30 inferences to approximate the paraboloid on Ω. We note the outperformance of

Table 1: Architecture and training parameters of the neural networks considered for the paraboloid. Training is based on the Adam optimizer. Each member is created by randomly selecting a depth between 3 and 5, and then randomly selecting a width that is fixed for all hidden layers.

Method	**Architecture**	**Training**
BNN	depth=3, width=5	500 epochs, 0.01
BPINN	depth=3, width=5	15,000 epochs, lr = 0.001
NN Ensemble	depth $\in [3, 5]$, width $\in [10, 20]$	500 epochs, lr = 0.01
PINN Ensemble	depth $\in [3, 5]$, width $\in [10, 20]$	5000 epochs, lr = 0.001

physics-informed networks in the extrapolation region, where they have much lower errors compared with their standard counterparts. The PINN ensemble, however, performs relatively worse than the Bayesian PINN, which may be due to the inclusion of poorly-performing networks in the ensemble compared to a single BPINN that is trained until it reaches the desired loss. An additional step could consist in removing these anomalous networks from the ensemble. However, the ensemble diversity also translates to a higher standard deviation (row 3), and more importantly, to a more conservative error-to-standard deviation ratio (see row 4). The ensemble's standard deviation is a more reliable indicator of the error than the Bayesian PINN's standard deviation. The same observation is made when comparing the Bayesian NN and the NN ensemble, with the latter being more conservative in its assessment of the model uncertainty (the Bayesian approach's standard deviation underestimates the error). Of particular note is the NN ensemble's capacity in predicting the area of the domain where the model prediction is reliable, namely within Ω_i, where standard deviation is close to zero.

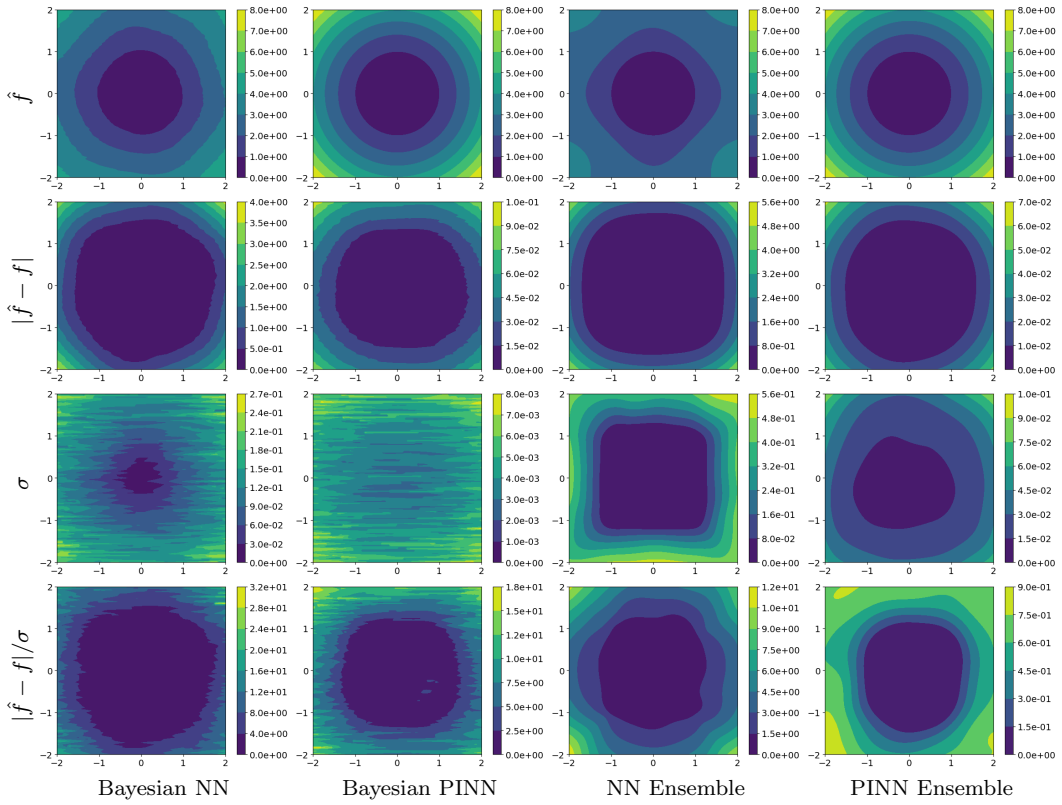

Figure 1: For each neural-network approach (represented by columns), row 1 represents the prediction $\hat{f}$ as obtained from the mean of all 30 inferences, row 2 represents the error against the true function ($f = x^2 + y^2$), row 3 represents the standard deviation σ from the mean across the 30 inferences, and row 4 represents the ratio of the error to standard deviation (row 2 divided by row 3).

3 Wave equation

The wave equation is at the core of many scientific applications, such as earthquake modeling, acoustics, hydrocarbon exploration, and stealth aircraft design. While effective, traditional numerical methods (e.g., finite elements) remain expensive and neural-network-based approximations could be attractive. Here, we develop physics-informed neural networks for the two-dimensional wave equation, as described by [3], and assess the uncertainty in the extrapolation region using Bayesian and ensemble approaches. Training data is gathered from the numerical simulator in the interpolation region ($t \leq 1$) and we define the extrapolation region by $t > 1$. The data-sampling and training strategy follows [3]. In all cases, we use five hidden layers of width 100 and train for 15,000 epochs with a learning rate of 0.001 (Adam optimizer). Given the difficulty in training PINNs for the wave equation, we were able to successfully train only 10 networks. Training procedures only differed by the random initialization of the network parameters, whereas member architectures were identical. In Figure 2, we compare the BPINN and PINN ensemble approaches in their ability to quantify the model uncertainty and assess whether the uncertainty is a reliable indicator of the actual error. The ground truth is obtained by running a numerical simulation of the wave equation based on a high-order discontinuous-Galerkin method. We start by contrasting, at different times, the predictions from a PINN and from a BPINN (one inference at each time) to illustrate the noisy nature of the BPINN's prediction (the mean of all 30 predictions is much smoother). The last two rows of Figure 2 show the error-to-standard deviation ratios at the same times, as an attempt to determine whether each approach is either conservative or overconfident in their error estimation. We note that, contrary to the paraboloid problem, the ensemble method is overconfident, which likely stems from the lack of significant-enough variation across members of the ensemble. The overconfidence is particularly notable in the interpolation region or slightly beyond ($t \leq 1.25$), which could be caused by overfitting. This result highlights the importance of constructing an appropriate ensemble, that is, an ensemble featuring enough architectural variation across members.

4 Summary

We explored two strategies to quantify the uncertainty of NN-based surrogates of physics problems, namely a Bayesian approach and an ensemble approach. Between the two uncertainty-quantification methods, our results show that the Bayesian neural networks render overconfident results while model outputs from a well-constructed ensemble are appropriately conservative. To be usable in scientific applications, these methods must be equipped with a mechanism to interpret, and act on, the uncertainty estimate. This effort is ongoing and we encourage the scientific computing community to move in that direction.

5 Broader impact

Machine-learning-based surrogate models are increasingly considered to accelerate parts of scientific simulations. We recognize that these ML algorithms can significantly speed up simulations and also make them more energy-efficient. We also stress that these approximations should not hamper the overall simulation accuracy more than is acceptable. This requirement drives the need for embedding *efficient* uncertainty-quantification methods while deploying those surrogate models. By failing to do so, we run the risk of limited adoption by a community that is used to relying on rigorous a priori and a posteriori error-estimation techniques. In addition, any uncertainty-quantification technique should be evaluated not only based on its capacity to deliver an appropriate metric, but also on how efficiently it can run on current and future computing resources. While Bayesian methods seem to be less able to provide conservative error estimates, we note that they are more efficient from a computational and data-storage perspective (only one model needs to be trained). This complex landscape warrants more research.

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 19, 2022, and under the mentorship and guidance of Advanced Micro Devices,

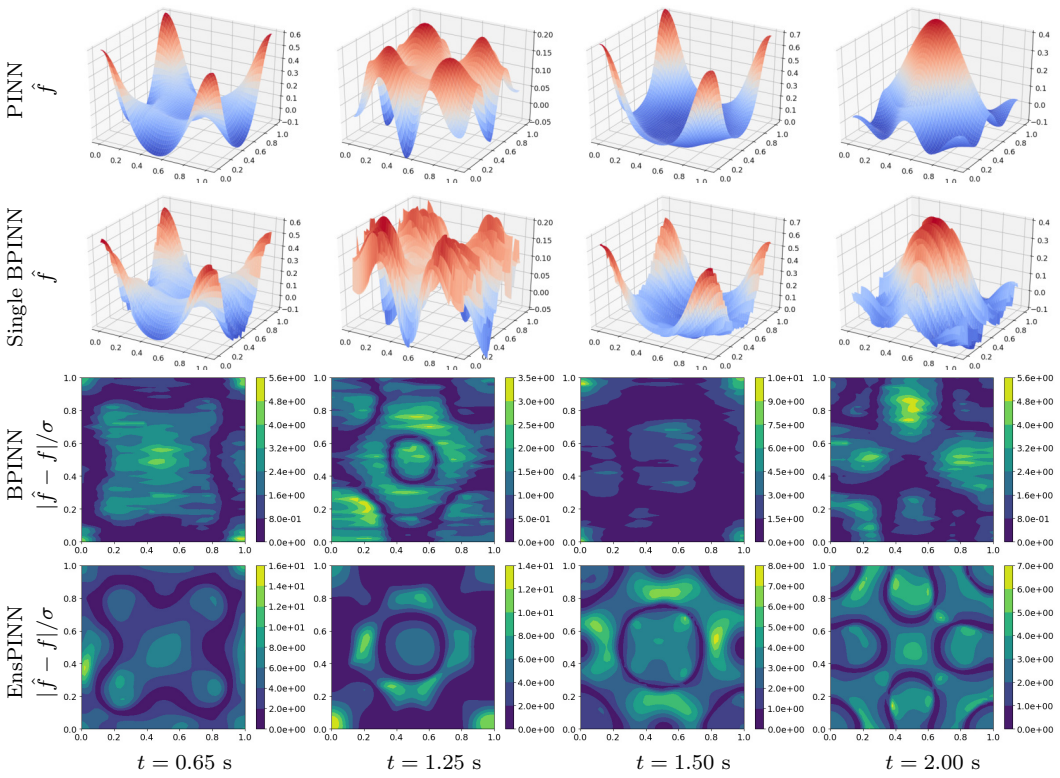

Figure 2: Uncertainty quantification for the wave equation as predicted by a Bayesian PINN (30 inferences) and a 10-member PINN ensemble. Row 1 represents the prediction from a deterministic PINN [3] at different times (labeled data only available for $t \leq 1$). Row 2 shows the BPINN predictions at the same times. Only one inference is computed at each time instead of 30 to illustrate the noisy result and the network's capacity to capture the uncertainty (the mean would be smoother). Row 3 and 4 show the error-to-standard deviation ratios for the BPINN and PINN ensemble approaches, respectively.

Inc. The first four 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] D. Davini, B. Samineni, B. Thomas, A. H. Tran, C. Zhu, K. Ha, G. Dasika, and L. White. Using physics-informed regularization to improve extrapolation capabilities of neural networks. In *NeurIPS Workshop on Machine Learning and the Physical Sciences*, 2021.

[4] 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.

[5] 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.

[6] 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.

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

[8] 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.

[9] 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.

[10] 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.

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

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

[13] L. Yang, X. Meng, and G. E. Karniadakis. B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data. *Journal of Computational Physics*, 425:109913, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] We plan to release the code in the future.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] The focus of the work is on algorithms tested on problems that are fairly small.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]