XLI CILAMCE
XLI Ibero-Latin American Congress on Computational Methods in Engineering
16-19 November, 2020 FOZ DO IGUAÇU PARANÁ - BRAZIL

Physics–Informed Neural Networks for the Factored Eikonal Equation

Rômulo M. Silva[1], Alvaro L. G. A. Coutinho[1]

[1]*High Performance Computing Center and Civil Engineering, COPPE/Federal University of Rio de Janeiro*
Av. Horácio Macedo, 2030, Room I248, 21941-598, Rio de Janeiro, Brazil
romulo.silva@nacad.ufrj.br, alvaro@nacad.ufrj.br

Abstract. The Eikonal equation often appears in problems including, but not limited to, geometric optics, shortest path problems, image segmentation, seismic and medical imaging. While there are efficient and stable techniques for solving the Eikonal equation for regular or arbitrary geometries in several dimensions, it remains a big challenge to solve inverse problems governed by this equation, especially when it comes to uncertainty quantification. As an alternative to classical methods for solving forward and inverse problems governed by PDEs, the Physics-Informed Neural Networks (PINNs) have shown accuracy and versatility for solving problems in fluid dynamics, inference of hydraulic conductivity, velocity inversion, phase separation, and others, being also able to quantify uncertainty in these problems. Here we study PINNs for solving the forward and inverse probabilistic Factored Eikonal Equation. We solve the probabilistic inverse problem using variational inference and the Flipout technique. Our results for a benchmark problem show excellent accuracy.

Keywords: Physics-Informed Neural Networks, Scientific Machine Learning, Factored Eikonal Equation, Inverse Problems, Uncertainty Quantification

1 Introduction

The Eikonal equation is a first-order nonlinear equation present in many fields. It describes phenomena like wave propagation for acoustic and elastic media, as well as electromagnetic media [1]. Therefore, the Eikonal equation plays an important role in problems like geometric optics, shortest path problems, image segmentation, seismic and medical imaging. The Eikonal equation is given by,

$$\|\boldsymbol{\nabla}\phi(\mathbf{x})\|_2 = \frac{1}{v(\mathbf{x})}, \forall \mathbf{x} \in \Omega. \tag{1}$$

where $\Omega \subset \mathbb{R}^{n_{sd}}$, is the domain, $n_{sd} = 1, 2, 3$ is the number of space dimensions, $\mathbf{x} = \{x, y, z\}$ is the position vector, and $\|.\|$ stands for the L^2-norm. Solving this equation for $\phi(\mathbf{x})$ for a given velocity field, $v(\mathbf{x})$, and proper boundary conditions become feasible employing numerical methods as, for example, for cartesian grids, the Fast Marching Method [2], and the Fast Sweeping Method [3], or, for unstructured grids [4]. A more complicated task is to find a velocity $v(\mathbf{x})$ that satisfies eq. (1) and a given a set of scattered measures of $\phi(\mathbf{x})$. It falls in the class of inverse problems, which are extremely ill-posed and have no unique solution. Therefore, there is a need for new techniques for solving both forward and inverse problems governed by the Eikonal equation, particularly when it comes to uncertainty quantification. However, when point singularities are present, as, in seismic imaging, the Factored Eikonal Equation (FEE for short) is preferable, as pointed out in [5].

Recently, scientific machine learning methods, and particularly the Physics-Informed Neural Networks (PINNs) [6–8] have shown considerable success for solving forward and inverse problems, also including uncertainty quantification. PINNs have been used for solving problems governed by the Cahn-Hilliard, Allen-Cahn, and Navier-Stokes equations [9, 10], solving stochastic PDEs [11], learning pressure and velocity fields for fluid mechanics problems [12], and solving forward [13, 14] and inverse [15] problems governed by the Eikonal equation. The present study contributes to expanding the field, tackling direct and inverse problems governed by the FEE. The remainder of this paper is organized as follows. In Section 2, we briefly present the forward and inverse problems of interest. In the sequel, Section 3 introduces the PINNs for the direct and probabilistic inverse problems governed

by the FEE. Section 4 shows our numerical results for both the forward and inverse problems. The paper ends with a summary of our main findings.

2 Forward and Inverse Factored Eikonal Equation Problems

For some specific situations, the Eikonal equation solution can be factored as $\phi(\mathbf{x_s}, \mathbf{x_r}) = \tau(\mathbf{x_s}, \mathbf{x_r})\xi(\mathbf{x_s}, \mathbf{x_r})$, where $\mathbf{x_s}$ is the source position, $\mathbf{x_r}$ is the receiver position, which can be any point that lies in the domain Ω. For the particular scenario where a point source is applied as boundary condition, $\xi(\mathbf{x_s}, \mathbf{x_r}) = \|\mathbf{x_s} - \mathbf{x_r}\|_2$ represents the solution of equation (1) for the case where a point source is applied in a domain such as $v(\mathbf{x}) = v(\mathbf{x_r}) = 1$. Substituting the factorization of ϕ in (1), the resulting equation,

$$\tau^2 \|\boldsymbol{\nabla}_{\mathbf{x_r}} \xi\|_2^2 + 2\tau\xi \boldsymbol{\nabla}_{\mathbf{x_r}} \tau \cdot \boldsymbol{\nabla}_{\mathbf{x_r}} \xi + \xi^2 \|\boldsymbol{\nabla}_{\mathbf{x_r}} \tau\|_2^2 = \frac{1}{v(\mathbf{x_r})^2} \tag{2}$$

is the FEE. Here, $\boldsymbol{\nabla}_{\mathbf{x_r}}$ is the gradient operator with respect to the receiver position. This formulation is advantageous in situations where $\phi(\mathbf{x})$ has point source singularities, which can be well captured by $\xi(\mathbf{x_s}, \mathbf{x_r})$, while $\tau(\mathbf{x_s}, \mathbf{x_r})$ acts as a smooth correction at the neighborhood of the point source singularities [5]. In the forward problem, equation (2) is only solved for τ which has as boundary condition $\tau(\mathbf{x_s}, \mathbf{x_r} = \mathbf{x_s}) = \frac{1}{v(\mathbf{x_s})}$. In the inverse problem, given the source position $\mathbf{x_s}$ and a set of (few) scattered measurements of $\phi(\mathbf{x})$ (here called ϕ_{data}), one seeks to find a set of parameters θ_ϕ, for representing the solution of the state variable ϕ, and θ_v, for representing the velocity v, that satisfies the PDE and the scattered measurements of $\phi(\mathbf{x})$. However, since the inverse problem is ill-posed and have no unique solution, it is reasonable to pose it as a probabilistic one. Thus, instead of doing a Maximum a Posteriori (MAP) estimate of $\theta_\mathbf{v}$, we can use the Bayesian framework and try to compute $\theta_\mathbf{v}$'s posterior probability distribution $P(\theta_\mathbf{v}|\phi_{data})$ as,

$$P(\theta_\mathbf{v}|\phi_{data}) \propto P(\phi_{data}|\theta_\mathbf{v})P(\theta_\mathbf{v}) \tag{3}$$

where $P(\phi_{data}|\theta_\mathbf{v})$ is the likelihood function, $P(\mathbf{v})$ is the parameters' prior, and $P(\phi_{data})$ is the evidence. We can approximate $P(\theta_\mathbf{v}|\phi_{data})$ by $q_{\theta_\mathbf{v}}$, using Variational Bayesian Inference methods [16], by minimizing,

$$\mathcal{F}(\theta_\mathbf{v}) = \mathbb{E}\left[logP(\phi_{data}|\theta_\mathbf{v})\right] - D_{KL}(q_{\theta_\mathbf{v}}, P(\theta_\mathbf{v}|\phi_{data})) \tag{4}$$

which is known as the variational lower bound or the evidence lower bound (ELBO). The term $\mathbb{E}\left[logP(\phi_{data}|\theta_\mathbf{v})\right]$ is the expected negative log-likelihood function and $D_{KL}(q_{\theta_\mathbf{v}}, P(\theta_\mathbf{v}|\phi_{data}))$ is the Kullback-Leibler divergence which measures the discrepancy between the true posterior distribution and its approximation.

3 Physics-Informed Neural Networks in a Nutshell

Let $\Phi(\hat{\mathbf{x}})$ with $\hat{\mathbf{x}} \in \mathbb{R}^{N_{in}}, \Phi \in \mathbb{R}^{N_{out}}$ be a state variable and $\mathcal{N}$ a non-linear differential operator such that,

$$\mathcal{N}(\hat{\mathbf{x}}; \Phi; ...) = 0. \tag{5}$$

One can choose to approximate the state variable by a convenient representation $\hat{\Phi}$ which could be, among many possible options, a Neural Network (NN, for short), $\mathcal{H}(\hat{\mathbf{x}}) : \mathbb{R}^{N_{in}} \rightarrow \mathbb{R}^{N_{out}}$, which falls within the class of Universal Approximators [17]. By introducing this approximation in eq. (5), it generates a residual $\mathcal{R}(\hat{\mathbf{x}})$, that is,

$$\mathcal{R}(\hat{\mathbf{x}}) = \mathcal{N}(\hat{\mathbf{x}}; \hat{\Phi}; ...). \tag{6}$$

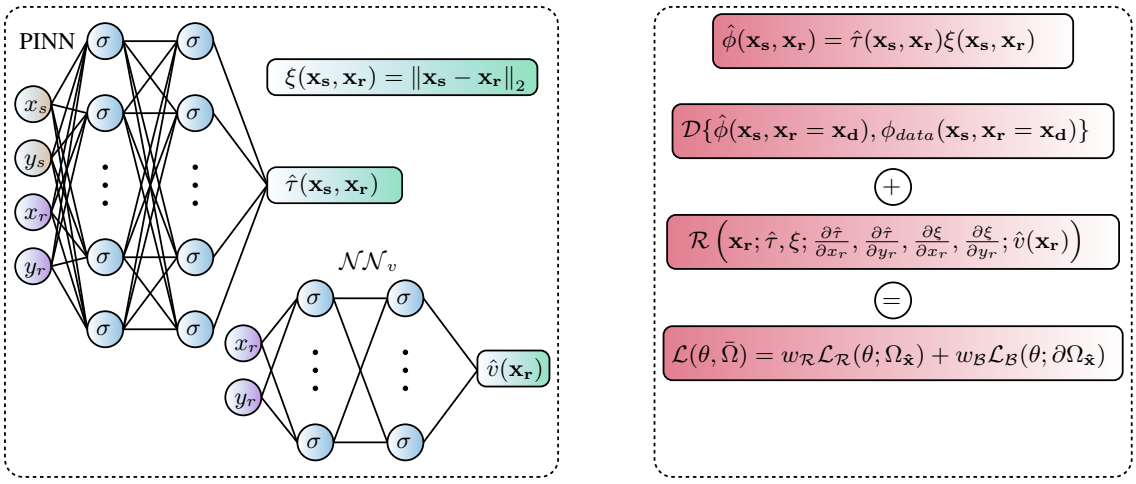
By assuming $\hat{\Phi} = \mathcal{H}(\hat{\mathbf{x}})$, one can impose the NN to satisfy the PDE residual $\mathcal{R}(\hat{\mathbf{x}})$, the boundary and experimental data, $\mathcal{B}(\hat{\Phi}, \hat{\mathbf{x}})$, by composing a loss function as,

$$\mathcal{L}(\theta, \bar{\Omega}) = w_{\mathcal{R}}\mathcal{L}_{\mathcal{R}}(\theta; \Omega_{\hat{\mathbf{x}}}) + w_{\mathcal{B}}\mathcal{L}_{\mathcal{B}}(\theta; \partial\Omega_{\hat{\mathbf{x}}}) \tag{7}$$

where $\bar{\Omega} = \Omega_{\hat{\mathbf{x}}} \cup \partial\Omega_{\hat{\mathbf{x}}}$, $\Omega_{\hat{\mathbf{x}}}$ denotes the set of residual (collocation) points, $\hat{\mathbf{x}}$, $\partial\Omega_{\hat{\mathbf{x}}}$ indicates boundary and experimental points, and $w_{\mathcal{R}}$ and $w_{\mathcal{B}}$ are the weights. The terms $\mathcal{L}_{\mathcal{R}}(\theta; \Omega_{\hat{\mathbf{x}}})$ and $\mathcal{L}_{\mathcal{B}}(\theta; \partial\Omega_{\hat{\mathbf{x}}})$ are given by,

$$\mathcal{L}_{\mathcal{R}}(\theta; \Omega_{\mathbf{x}}) = \mathcal{D}\{\mathcal{R}(\hat{\mathbf{x}}), 0\}, \text{ and } \mathcal{L}_{\mathcal{B}}(\theta; \partial\Omega_{\hat{\mathbf{x}}}) = \mathcal{D}\{\mathcal{B}(\hat{\Phi}, \hat{\mathbf{x}}), \Phi_{data}\} \tag{8}$$

where $\mathcal{D}\{\cdot, \cdot\}$ is an operator used to measure the discrepancy between two quantities, whose suitable choices could be the L^2, L^1 norms, and the Huber Loss [18]. It is also worth to remind that equations (5) and (6) also encapsulate the velocity, which can be represented as trainable parameters or even by an NN, facilitating the process of parameter estimation using PINNs. For the sake of generalization in FEE problems with multiple point sources, we choose to use the same input scheme for our PINN as used in EikoNet [13] where the NN receives not only the receiver position but also the source's position. Hence, a unique PINN can solve multi-shot problems like the ones that arise in geophysics. Consequently, the PINN input vector is a concatenation of $\mathbf{x_s}$ and $\mathbf{x_r}$ and it approximates the field $\tau(\mathbf{x_s}, \mathbf{x_r})$ by $\hat{\tau}(\mathbf{x_s}, \mathbf{x_r})$ represented in Figure 1 for the 2D case. For inversion purposes, we use a NN to predict the velocity ($\mathcal{NN}_v$ for short), which is also represented in Fig. 1. For solving typical deterministic forward and inverse problems, both NNs can be composed of simple, fully connected hidden layers written in terms of weights and biases represented by θ (said the network parameters). Here, we call θ_ϕ the parameters of the PINN that solves the FEE, and θ_v the parameters of $\mathcal{NN}_v$. Figure 1 also shows the relationships between the two NNs and each of the main components of the loss function. It is worth to remind that Automatic Differentiation [19] computes the NNs derivatives with respect to the input coordinates.



(a) PINNs for $\hat{\tau}$ and $\hat{v}$ (b) Loss function composition

Figure 1. Factored Eikonal Equation PINN Scheme.

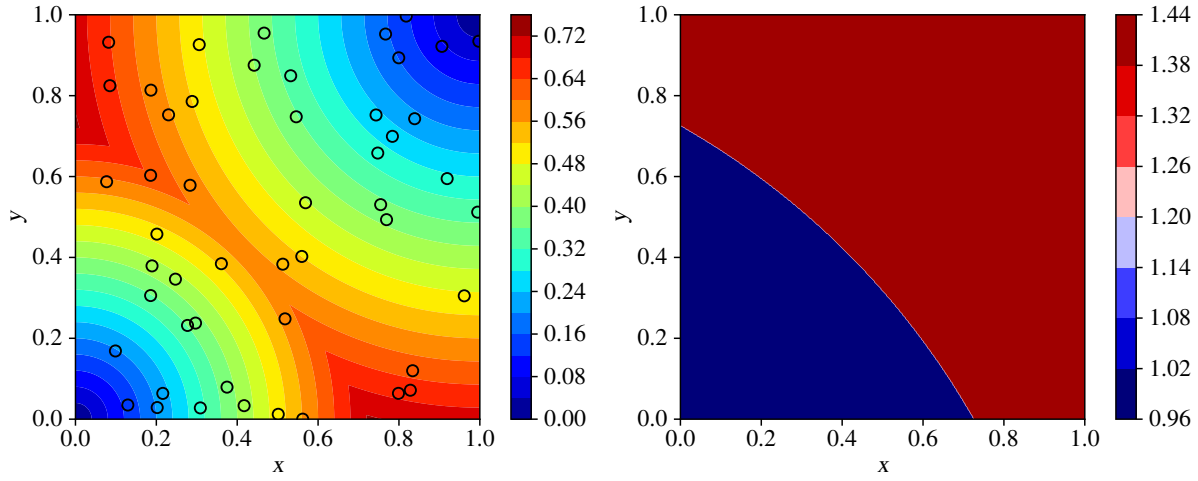
Therefore, to solve the forward problem (2), given the source position $\mathbf{x_s}$ and the velocity $v(\mathbf{x_r})$, we need to find a set of parameters θ_ϕ that minimizes eq. (7), *i.e.* with these parameters the NN should satisfy the FEE. The minimization of the loss function is easily achievable by using methods such as ADAM, RMSProp, Newton-CG, and LBFGS. For inversion purposes, the velocity is parameterized by $\mathcal{NN}_v$, and it is straightforward to think that we should somehow treat the network parameters θ_v as random variables to quantify its uncertainties and analyze how it propagates to $\hat{v}(\mathbf{x_r})$. To treat $\mathcal{NN}_v$ as a Bayesian NN, we substitute its regular dense hidden layers with Dense-Flipout layers. These layers use the Flipout estimator [20], which applies a Monte Carlo approach to the posterior distribution by integrating over $\theta_\mathbf{v}$. Flipout has the additional advantage of showing low variance levels if compared to other methods. We use the implementation of the Dense-Flipout layers in TensorFlow Probability [21]. Since in $\mathcal{NN}_v$ all the information regarding the velocity is encapsulated into the residual part of the loss function, it is not necessary to directly compute the expected negative log-likelihood function shown in eq. (4), instead, the Kullback-Leibler divergence term is computed by Tensorflow Probability and added into the loss function (8) with a scale factor $\gamma_{KL} = 10^{-8}$.

4 Results and Discussion

Here we solve a mixed problem [15] whose exact solutions for both forward (eq. (9)) and inverse (eq. (10)) problems are given and shown in Figure 2.

$$\phi(x,y) = min\left(\sqrt{x^2+y^2}, 0.7\sqrt{(x-1)^2+(y-1)^2}\right) \tag{9}$$

$$v(x,y) = \begin{cases} 1.0 \text{ if } \sqrt{x^2+y^2} < \sqrt{(x-1)^2+(y-1)^2} \\ \frac{1.0}{0.7} \text{ otherwise} \end{cases} \tag{10}$$



(a) Exact solution, $\phi(\mathbf{x})$

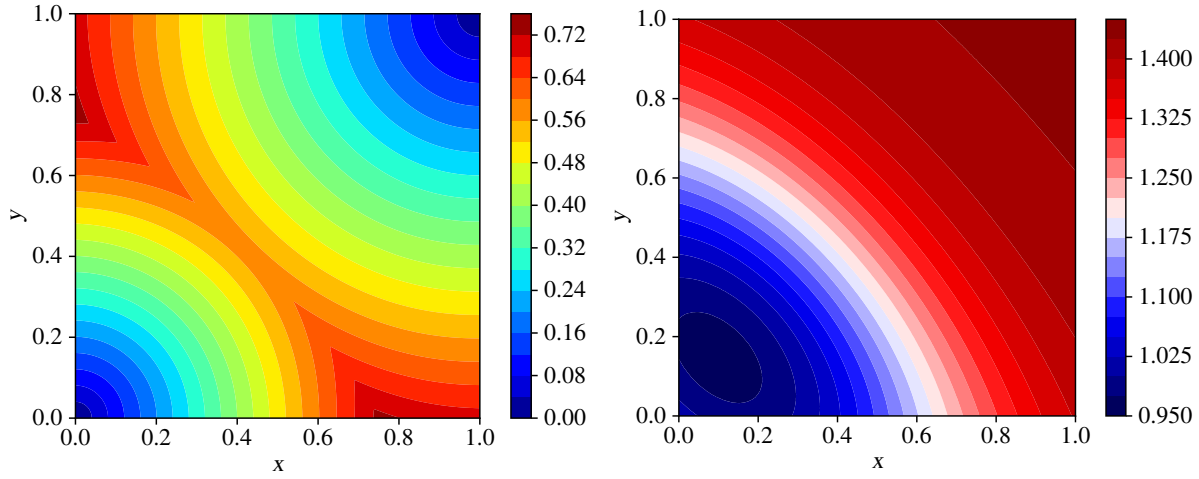
(b) Ground truth velocity, $v(\mathbf{x})$

Figure 2. Exact solution and velocity field for the mixed problem in [15]. The circles indicate the positions of the synthetic measurements.

We need a metric to measure eq. (8). We choose the L^2 norm. The NN used to predict the velocity model is composed of four probabilistic layers using the Flipout [20] technique with ReLU activation, followed by a linear layer. The NN that predicts the travel time field is composed by five fully connected layers with 20 neurons each using as activation function the exponentially scaled modified Bessel function of order 1, also followed by a linear layer in which we apply the $abs(\cdot)$ function to allow only positive values since our factorization is multiplicative. We use a set of 50 scattered travel time measurements randomly selected inside the domain. As in [15] we parametrize the NN output that predicts the velocity as $\bar{v}(\mathbf{x}) = V_{max}\sigma_{sigmoid}(\hat{v}(\mathbf{x_r}))$ with $V_{max} = \frac{1.0}{0.7}$, and add a Total Variation [22] term to the loss function. The total variation term is necessary to overcome the excessive noise that appears at the beginning of the training process due to the use of probabilistic layers for uncertainty quantification instead of initializing and training multiple NNs as in [15]. As shown in [23] the choice of the weights $w_{\mathcal{R}}$ and $w_{\mathcal{B}}$ in eq. (7) has a considerable influence on the PINNs convergence. It is recommended to reinforce the NN to first satisfy the boundary conditions and experimental data to accelerate convergence, which means that we should have $w_{\mathcal{R}}, w_{\mathcal{B}} > 0$ with $w_{\mathcal{R}} < w_{\mathcal{B}}$. Here we choose to make $w_{\mathcal{R}} = 1$ while $w_{\mathcal{B}} = 25$.

The problem in eq. (9) can be decomposed in two different problems with point sources located at $\mathbf{x_{s_1}} = (0,0)$ and $\mathbf{x_{s_2}} = (1,1)$. Their respective solutions are given by $\phi_1(\mathbf{x})$ and $\phi_2(\mathbf{x})$, both approximated by the same PINN, where the final solution is given by $\hat{\phi}(\mathbf{x}) = min(\phi_1(\mathbf{x}), \phi_2(\mathbf{x}))$, since the FEE solution gives the minimum time for a wavefront to travel from the point sources to any receiver inside the domain. Figure 3a shows the forward problem solution, which due to the simple velocity field, achieves a R^2 score of 0.996 when compared to the exact solution. Therefore, given only the sources' positions and the velocity, we can solve the forward problem with high accuracy. We train the NNs for 50,000 epochs using the ADAM optimizer with a learning rate of 10^{-3}. At each training epoch, we randomly select 1200 collocation points along with the spatial domain for evaluating the PDE residual. Figure 3b shows the mean velocity field obtained by generating 500 samples after the training

process. Even without inserting noise into the synthetic measurements, it is worth trying to quantify uncertainty, mainly due to the presence of the discontinuity in the ground truth velocity field, which is challenging to capture and represent. We can see in Fig. 4a the higher values of the standard deviation of the velocity field appears close to the discontinuity.



(a) Predicted travel time, $\hat{\phi}(\mathbf{x})$. $R^2 = 0.966$

(b) Mean velocity field. $R^2 = 0.768$

Figure 3. Solution of the forward problem (a) and the mean solution of the inverse probabilistic problem (b).

We limit the output of the velocity prediction to be $0.0 < \hat{v}(\mathbf{x}) \leq \frac{1.0}{0.7}$. Consequently, the region in which the velocity should assume its maximum value ends up with the smallest values of variance, as we can see in Fig. 4a. That is associated with the fact the predictions tend to collapse into one of the defined bounds. Since the minimum admissible value is not specified to be equal to 1.0, as in the ground truth velocity field, it is reasonable to expect that we have considerably higher levels of variance in the regions in which the velocity should be equal to 1.0. As expected, it is possible to notice that the higher levels of variance are close to the velocity discontinuity, which is a difficult feature to capture. For a better interpretation of the uncertainty, we show in Fig. 4b the Confidence Index (CI for short) map [24]. It simply consists of a point-wise normalization of the standard deviation to the interval $[0.0, 1.0]$, in which the lower values indicate a less reliable prediction, and the higher values indicate a strongly reliable prediction. Here we can notice the same behavior as in Fig. 4a, where both indicate that the less reliable region is close to the velocity discontinuity. Thus, the Confidence Index is given by,

$$CI = \frac{\sigma_{max} - \sigma_{\mathbf{x}}}{\sigma_{max} - \sigma_{min}} \tag{11}$$

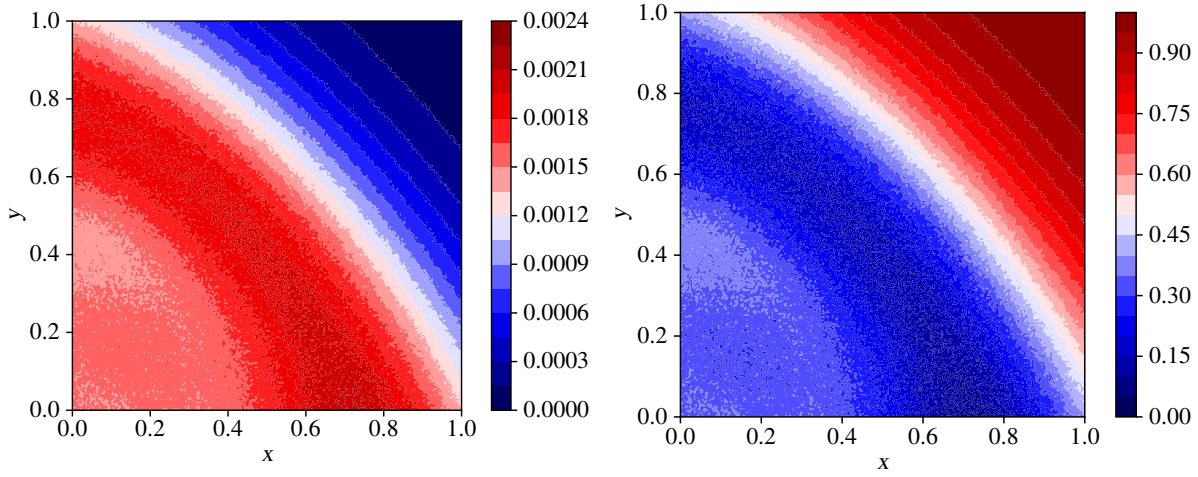
where σ_{max} and σ_{min} are the maximum and minimum standard deviation values in the whole domain, and $\sigma_{\mathbf{x}}$ is the standard deviation evaluated at the spatial coordinate $\mathbf{x}$.

For monitoring the training process, at each iteration, we choose one random snapshot of the velocity model and compare it with the ground truth by using the R^2 score as the metric, as shown in Fig. 5a. We can see in Fig. 5b the evolution of the loss function. After 20,000 epochs, we achieve reasonably good results, but we still training until the completion of 50,000 training epochs.

Once the PINN is trained, it is possible to generate an arbitrary number of samples at any spatial coordinates and quantify the uncertainty using any metric, which could be simple point-wise histograms, as in Figs. 6a and 6b, or even maps as the Confidence Index as presented before.

5 Conclusions

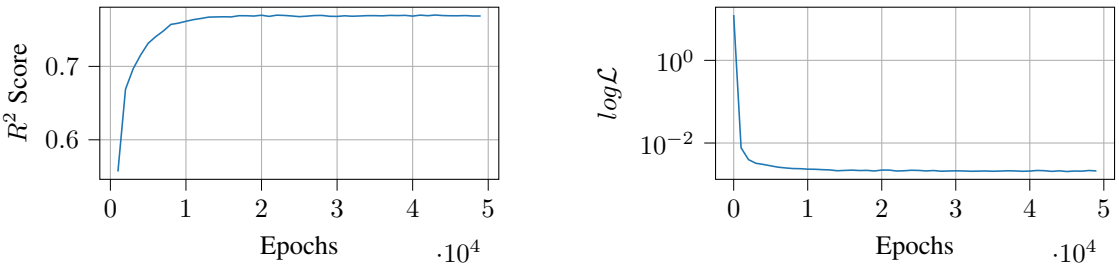
Our results suggest that the FEE is indeed the best alternative for solving point-source Eikonal problems using PINNs. By using the same input scheme (source and receiver position) for the PINN as in [13], our solution scheme becomes a powerful tool for solving multi-shot or multi-source problems. Besides, we achieve for the forward problem excellent results ($R^2 = 0.996$) with simple, fully connected NNs with a small number of neurons, differently from the networks in [13], which used a larger number of neurons in its dense blocks. We also easily



(a) Standard Deviation

(b) Confidence Index

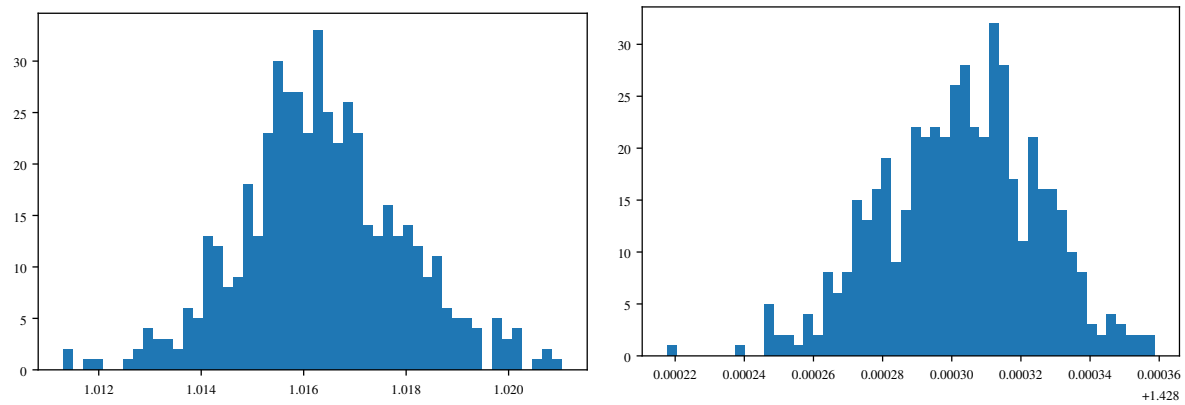
Figure 4. Standard deviation (a) and Confidence Index for the inverse probabilistic problem (b).



(a) Evolution of the R^2 score

(b) Evolution of the Loss Function

Figure 5. Monitoring PINN training: R^2 score (a) and Loss function (b) for the inverse probabilistic problem.



(a) Histogram for $\mathbf{x} = (0, 0)$

(b) Histogram for $\mathbf{x} = (1, 1)$

Figure 6. Uncertainty Quantification: Velocity histograms at chosen locations.

extend the PINNs for solving probabilistic inverse problems efficiently thanks to the implementation of the Dense-Flipout layers in TensorFlow Probability. We observe in the training that the loss function after 10,000 epochs is around 0.002 and the $R^2 > 0.76$, indicating a good quality in the results. As a last remark, we choose our activation function for the direct and probabilistic inverse problems by trial and error, which is a cumbersome task. Automatic hyperparameter tuning contributes to alleviating this burden, and we plan to explore it shortly.

Acknowledgements. This work is partially supported by CNPq, FAPERJ, ANP, and Petrobras.

Authorship statement. The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work, and that all material that has been herein included as part of the present paper is either the property (and authorship) of the authors, or has the permission of the owners to be included here.

References

[1] Debnath, L., 2012. *First-Order Nonlinear Equations and Their Applications*, pp. 227–256. Birkhäuser Boston, Boston.

[2] Stanley Osher, R. F. a., 2003. *Level Set Methods and Dynamic Implicit Surfaces*. Applied Mathematical Sciences 153. Springer-Verlag New York, 1 edition.

[3] Zhao, H., 2004. A fast sweeping method for eikonal equations. *Mathematics of Computation*, vol. 74, n. 250, pp. 603–628.

[4] Elias, R. N., Martins, M. A., & Coutinho, A. L., 2007. Simple finite element-based computation of distance functions in unstructured grids. *International journal for numerical methods in engineering*, vol. 72, n. 9, pp. 1095–1110.

[5] Fomel, S., Luo, S., & Zhao, H., 2009. Fast sweeping method for the factored eikonal equation. *Journal of Computational Physics*, vol. 228, n. 17, pp. 6440 – 6455.

[6] Raissi, M., Perdikaris, P., & Karniadakis, G. E., 2017a. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561*, vol. .

[7] Raissi, M., Perdikaris, P., & Karniadakis, G. E., 2017b. Physics informed deep learning (part ii): Data-driven discovery of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10566*, vol. .

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

[9] Wight, C. L. & Zhao, J., 2020. Solving allen-cahn and cahn-hilliard equations using the adaptive physics informed neural networks. *arXiv preprint arXiv:2007.04542*, vol. .

[10] Yin, M., Zheng, X., Humphrey, J. D., & Karniadakis, G. E., 2020. Non-invasive inference of thrombus material properties with physics-informed neural networks. *arXiv preprint arXiv:2005.11380*, vol. .

[11] Yang, L., Treichler, S., Kurth, T., Fischer, K., Barajas-Solano, D., Romero, J., Churavy, V., Tartakovsky, A., Houston, M., Prabhat, & Karniadakis, G., 2019. Highly-scalable, physics-informed gans for learning solutions of stochastic pdes. *arXiv preprint arXiv:1910.13444*, vol. .

[12] Raissi, M., Yazdani, A., & Karniadakis, G. E., 2020. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, vol. 367, n. 6481, pp. 1026–1030.

[13] Smith, J. D., Azizzadenesheli, K., & Ross, Z. E., 2020. EikoNet: Solving the Eikonal equation with Deep Neural Networks. *arXiv preprint arXiv:2004.00361*, vol. .

[14] bin Waheed, U., Haghighat, E., Alkhalifah, T., Song, C., & Hao, Q., 2020. Eikonal solution using physics-informed neural networks. *arXiv preprint arXiv:2007.08330*, vol. .

[15] Sahli Costabal, F., Yang, Y., Perdikaris, P., Hurtado, D. E., & Kuhl, E., 2020. Physics-informed neural networks for cardiac activation mapping. *Frontiers in Physics*, vol. 8, pp. 42.

[16] Seeger, M. & Wipf, D., 2010. Variational bayesian inference techniques. *IEEE Signal Processing Magazine*, vol. 27, n. 6, pp. 81–91.

[17] Sifaoui, A., Abdelkrim, A., & Benrejeb, M., 2008. On the use of neural network as a universal approximator. *Int. J. Sci. Tech. Control Comput. Eng*, vol. 2, pp. 386–399.

[18] Huber, P. J., 1964. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, vol. 35, n. 1, pp. 73–101.

[19] Baydin, A. G., Pearlmutter, B. A., Radul, A. A., & Siskind, J. M., 2017. Automatic differentiation in machine learning: a survey. *The Journal of Machine Learning Research*, vol. 18, n. 1, pp. 5595–5637.

[20] Wen, Y., Vicol, P., Ba, J., Tran, D., & Grosse, R., 2018. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. *arXiv preprint arXiv:1803.04386*, vol. .

[21] Dillon, J. V., Langmore, I., Tran, D., Brevdo, E., Vasudevan, S., Moore, D., Patton, B., Alemi, A., Hoffman, M., & Saurous, R. A., 2017. Tensorflow distributions. *arXiv preprint arXiv:1711.10604*, vol. .

[22] Rudin, L. I., Osher, S., & Fatemi, E., 1992. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, vol. 60, n. 1-4, pp. 259–268.

[23] Shin, Y., Darbon, J., & Karniadakis, G. E., 2020. On the convergence and generalization of physics informed neural networks. *arXiv preprint arXiv:2004.01806*, vol. .

[24] Li, Y. & Sun, J., 2016. 3d magnetization inversion using fuzzy c-means clustering with application to geology differentiation. *GEOPHYSICS*, vol. 81, n. 5, pp. J61–J78.