Improved Training of Physics-informed Neural Networks using Energy-Based priors: A Study on Electrical Impedance Tomography

Akarsh Pokkunuru
Department of Computer Science
University of North Carolina at Charlotte
apokkunu@uncc.edu

Amirmohammad Rooshenas
Department of Computer Science
University of Illinois at Chicago
pedram@uic.edu

Thilo Strauss
Bosch ETAS Research
thilum@gmx.de

Anuj Abhishek
Department of Mathematics and Statistics
University of North Carolina at Charlotte
anuj.abhishek@uncc.edu

Taufiquar Khan
Department of Mathematics and Statistics
University of North Carolina at Charlotte
tkhan13@uncc.edu

Abstract

Physics-informed neural networks (PINNs) are attracting significant attention for solving partial differential equation (PDE) based inverse problems, including electrical impedance tomography (EIT). EIT is non-linear and especially its inverse problem is highly ill-posed. Therefore, successful training of PINN is extremely sensitive to interplay between different loss terms and hyper-parameters, including the learning rate. In this work, we propose a Bayesian approach through data-driven energy-based model (EBM) as a prior, to improve the overall accuracy and quality of tomographic reconstruction. In particular, the EBM is trained over the possible solutions of the PDEs with different boundary conditions. By imparting such prior onto physics-based training, PINN convergence is expedited by more than ten times faster to the PDE's solution. Evaluation outcome shows that our proposed method is more robust for solving the EIT problem.

1 Introduction

Physics-informed neural networks (PINNs) [8] parameterize the solutions of partial differential equations (PDE) by training neural networks to minimize a residual PDE and associated boundary conditions (BCs), in order to predict scalar valued solutions for any given point inside the problem's domain. In this work, we augment unsupervised PINNs with a joint representation using a Bayesian approach. We train a data-driven prior over joint solutions of PDEs on the entire domain and boundary points. This prior relates the predictions of PINNs via explicit joint representation and encourages learning a coherent and valid solution. More importantly, we focus on the non-linear and ill-posed electrical impedance tomography (EIT) inverse problem and experimentally show that using our data-driven Bayesian approach results in accurate, fast, and more robust training algorithms.

Machine Learning and the Physical Sciences workshop, NeurIPS 2022.

The EIT inverse problem is inspired by applications in paradigms such as tomographic imaging and geophysical ground water flow imaging. The goal is to reconstruct unknown electrical conductivities σ of any body $\Omega \subset \mathbb{R}^d$ with $d \in \{2, 3\}$ from measurements of finite electrical potential differences of neighboring surface electrodes. An elliptical PDE and its BCs govern the distribution of either electric potential u in its forward or σ in its inverse problem respectively. These are shown as follows:

$$-\nabla \cdot (\sigma \nabla u) = 0 \quad \text{on } \Omega \tag{1}$$

$$\sigma \left(\frac{\partial u}{\partial n} \right) = g \quad \text{on } \partial\Omega \text{ (Neumann BC)}, \tag{2}$$

$$u = f \quad \text{on } \partial\Omega \text{ (Dirichlet BC)} \tag{3}$$

An EIT experiment involves applying an electrical current g on the surface $\partial\Omega$ of the region Ω to be imaged, which then produces a current density given in Eq. 2. Here, n is a unit normal vector w.r.t u associated with Ω at its boundary. This current also induces u inside the body, whose surface value f given in Eq. 3 can be measured. Thus by repeating such experiments, we obtain the Neumann-to-Dirichlet (NtD) operator which finally aids in imaging. In this work, we solve a simplified version of EIT known as the semi-inverse problem, where we recover σ given the measurements of u in the interior of the medium. It should be noted that a similar formulation was studied in [1]. In our training paradigm, we simulate u in interior of the medium for any underlying σ and solve for the forward problem by training a PINN named u-Net, that can predict the value of the function $u(x)$ at any given point $x \in \Omega$. One can skip this step and directly use the ground truth u-data however, we use predictions from u-Net to simulate noisier forward solution for more robustness during inverse map construction. Thus, once we have access to pre-trained u-Net, we will subsequently train another PINN namely σ-Net, to finally recover $\sigma(x)$ for any $x \in \Omega$ from point-wise measurements of a function $u(x)$ that satisfies Eq. 1, 2 and, 3. In order to train σ-Net, we first define a functional form of Eqs. 1 and 2 as follows:

$$\mathcal{L}^d_{\text{PDE}} = \nabla \cdot \left(\sigma_d \, \nabla u_d \right) \quad \forall d \in \Omega \tag{4}$$

$$\mathcal{L}^b_{\text{BC}} = \left[\sigma_b \frac{\partial u_b}{\partial n_b} \right] \quad \forall b \in \partial\Omega_b, \quad \mathcal{L}^e_{\text{BC}} = \left[\sigma_e \frac{\partial u_e}{\partial n_e} - g_e \right] \quad \forall e \in \partial\Omega_e. \tag{5}$$

Note that, we use Neumann BC from Eq. 2 twice, once on all boundary points $d\Omega_b$ except electrodes and separately on electrodes $d\Omega_e$. We can then write our combined objective function as follows:

$$\mathcal{L}_\theta = \frac{\alpha}{\Omega} \sum_{d \in \{\Omega\}} (\mathcal{L}^d_{\text{PDE}})^2 + \frac{\beta}{M} \sum_{m \in \text{top}_M \, \mathcal{L}_{\text{PDE}}} |\mathcal{L}_{\text{PDE}}| + \frac{\gamma}{|\partial\Omega_b|} \sum_{b \in \partial\Omega_b} |\mathcal{L}^b_{\text{BC}}| + \frac{\delta}{|\partial\Omega_e|} \sum_{e \in \partial\Omega_e} \mathcal{L}^e_{\text{BC}} + \mathcal{R}(x), \tag{6}$$

where α, β, and γ and δ control the contribution of each term to the overall loss. Additionally, we enforce strong regularization in $\mathcal{R}(x)$ to encourage valid σ predictions (see Appendix Eq. 8 for full equation).

2 Energy Based Priors

As our main contribution, we propose a data-driven energy-based model (EBM) [7] as a prior to improve PINN training. Our energy-based prior is defined over the conductivity distribution σ as: $(p(\boldsymbol{\sigma}) \propto \exp(-E_\phi(\boldsymbol{\sigma}))$. Although several techniques have been proposed in literature for training EBMs including contrastive divergence [2], noise-contrastive estimation [4], score matching [6], and denoising score matching [13], we found denoising score matching (DSM) to be more stable, less compute intensive and ultimately generate more realistic σ solutions in our setting. DSM trains the energy function such that its vector field $(\nabla_{\boldsymbol{\sigma}_m} \log p_\phi(\boldsymbol{\sigma}_m))$ matches the vector field of the underlying $\boldsymbol{\sigma}$ distribution $p(\boldsymbol{\sigma}_m)$, which is approximated by perturbing the empirical data distribution with Gaussian noise of different intensities. See [11], [12] for more detail. We jointly estimate a noise conditional energy function $E_\phi(\boldsymbol{\sigma}, \mu)$ for all noise-perturbed data distributions conditioned on various noise scales $\mu \in [1 \dots L]$, satisfying $\mu_1 > \mu_2 > \cdots > \mu_L$. In our work, we chose $L = 20$ linearly spaced noise scales between $\mu_i \in [2, 0.01]$. Essentially, the training is to minimize the following objective:

$$\mathcal{L}(\phi; \mu_i) = \frac{1}{L} \sum_{i=1}^{L} \lambda(\mu_i) \left[\frac{1}{2} \mathbb{E}_{p(\sigma)} \, \mathbb{E}_{\mathbf{z} \sim \mathcal{N}(0,I)} \left\| \nabla_\sigma E_\phi(\hat{\boldsymbol{\sigma}}, \mu_i) - \frac{\mathbf{z}}{\mu_i} \right\|_2^2 \right], \tag{7}$$

where $\lambda(\mu_i) > 0$ is a coefficient function chosen as $\lambda(\mu) = \mu^2$ and finally, $\hat{\boldsymbol{\sigma}} = \boldsymbol{\sigma} + \mu_i \mathbf{z}$ is the noise perturbed version of conductivity distribution $\hat{\boldsymbol{\sigma}} \sim \mathcal{N}(\boldsymbol{\sigma}, \mu_i^2 \mathbf{I})$. In order to use E_ϕ as a prior, in contrast to the standard DSM training that trains a score network ($S(.) = -\nabla_z E$), we directly train the energy network. See Appendix A.3 for training details of E_ϕ, some generated $\boldsymbol{\sigma}$ and closest training data point shown in Fig. 4.

Upon successful training of the E_ϕ using Eq. 7, the energy function $E^*_\phi(\boldsymbol{\sigma}, \mu)$ will assign lower energy values to predicted σ that provide valid solutions and vice-versa for unlikely assignments that violates Eq. 1 greatly. Moreover, L_θ from Eq. 6 can be interpreted as a residual of violation in Eq. 1, noted as r for simplicity. Assume $\mathbf{r}$ (residual of entire domain) follow a multi-variate Gaussian distribution with a zero mean and diagonal covariance matrix. Therefore, maximizing the likelihood of $p(\mathbf{r}|\boldsymbol{\sigma}; \theta_\sigma)$ results in the similar optimization as minimizing $\sum_{d \in \Omega} (\mathcal{L}^d_{\text{PDE}})^2$ with respect to θ_σ (parameters of σ-Net). Now, we can define our Bayesian approach but assuming $\boldsymbol{\sigma}$ follows the prior distribution $p(\boldsymbol{\sigma})$: $\max_{\theta_\sigma} \log p(\mathbf{r}|\boldsymbol{\sigma}) + \log p(\boldsymbol{\sigma})$, where $\boldsymbol{\sigma}$ is parameterized by θ_σ via σ-Net. We interpret $p(\boldsymbol{\sigma})$ as possible solutions to the PDE defined by Eq. 1 for different boundary conditions. Now we can rewrite the final σ-Net training objective as: $\mathcal{L}_S = \frac{1}{K} \sum_{k=1}^{K} \mathcal{L}_{\theta_k} + \mathcal{R}(x) - \kappa E^*_\phi(\boldsymbol{\sigma}, \mu)$, where κ is the weight of the prior in the overall loss. The main advantage of the our framework is the generalizability of $E^*_\phi(\boldsymbol{\sigma}, \mu)$ term. Given that a large-scale training sample generation scheme is available to train the data-driven EB prior, one can easily train any suitable EBM model and can plug it into any solvable PDE and obtain highly accurate solutions.

3 Experiments

Our EIT data simulation setup primarily consists of phantom generation and forward solution construction via finite element solver. We initially construct various discretized solutions of σ with randomly chosen anomaly configuration and shape on a 2D mesh-grid of size 128×128 to obtain phantoms $\Omega_{1 \ldots Z}$. The target σ values are chosen randomly between $\in [3, 15]$ for either $1, 2$ or 3 anomalies per mesh. We additionally smooth our solutions following [3] using a Gaussian low-pass filter of size 200 and standard deviation 3. We then generate 6512 such smoothed phantoms for EB prior training and 1628 for testing which are standardized to $[0, 1]$ interval by dividing all samples using the maximum conductivity σ value obtained from training set. We additionally generate a few hand-crafted phantoms to train the u-Net and σ-Net for forward and inverse problems. These phantom configurations are specifically designed to be more challenging by setting higher shape deformity, larger variance in σ values for each anomaly and placing them in challenging locations inside mesh ro ensure robustness of our framework. Note that there are no phantoms in the EBM training set which resemble these hand-crafted samples.

3.1 Expedited Semi-Inverse Problem Evaluation

We now present the main results of our proposed framework. As discussed in prior sections, at each step of σ-Net training procedure, the current prediction $\hat{\sigma}$ is fed to the trained EB prior $E^*_\phi(\boldsymbol{\sigma}, \mu_L)$ to obtain a scalar energy value. This energy value provides useful supervision to σ-Net and essentially expedites the convergence of PINNs. This phenomenon can be viewed in Fig. 1. Here, each sub-figure

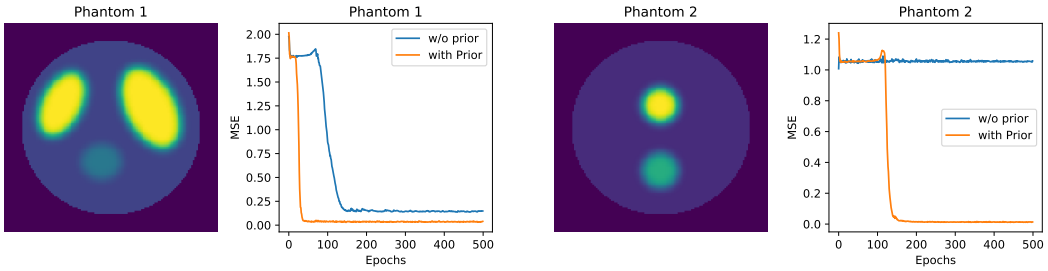


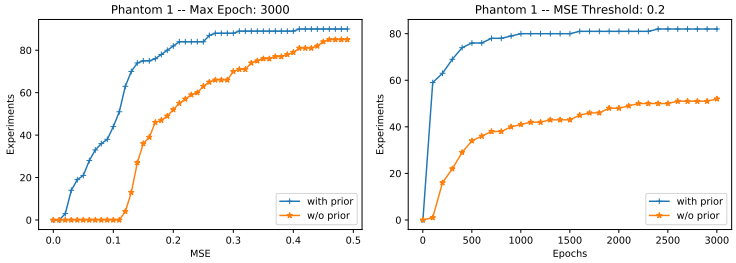
Figure 1: PDE solution convergence and accuracy analysis via σ MSE for various phantoms.

represents the learning curve of σ-Net trained to recover conductivity of phantoms 1, 2 with and without the inclusion of EB prior. We use mean squared error (MSE) as the choice of evaluation

metric. Evidently, the PINNs with EB prior converge much faster within the first few 100 epochs than the training without prior. In case of phantom 2, the convergence is more than ten times faster while also aiding the PINN to avoid getting stuck in a local minima. (See Appendix Fig. 3 showing the quality of reconstruction for other phantoms.)

3.2 Sensitivity analysis via random parameter search

We present a sensitivity analysis study for various parameters in Eq. 6. Since PINNs are extremely sensitive to the balance of interplay among different hyperparameters [14], the training procedure essentially becomes an optimization problem where one has to tune the right settings in order to converge to a highly accurate solution or fall into a failed trivial solution. The problem of choosing parameters is even more challenging in EIT as it is highly non-linear and ill-posed. We thus study the effect of changing the strength of various loss weighting penalties. More precisely, we restrict the search space to α, β, γ and δ which are main terms involving the PDE and its BC, while solving the semi-inverse problem. We then configure a limited valid set of weights for $\alpha \in [0.01, 0.05, 0.1, 0.5, 1]$, $\beta \in [0.01, 0.05, 0.1, 0.5, 1]$, $\gamma \in [0.01, 0.1, 0.5, 1.0, 2.0]$ and finally $\delta \in [0.001, 0.01, 0.1, 0.5, 1.0, 2.0]$ and randomly choose a single weight from each set to perform training. We conduct 200 of such random search runs on phantom 1, with and without the EB prior. The results of this study can be seen in Fig. 2.



(a) (b)

Figure 2: Sensitivity analysis of σ-Net while randomly searching the optimal weights in Eq. 6.

Here, Fig. 2a indicates the number of random experimental runs that were successful to reach a set upper bound of 0.5 MSE after training for 3000 epochs. We infer that the PINN training success rate with the EB prior is much higher. Additionally, a large number of experiments reach a MSE of 0.1 or lower in comparison to experiments without prior, translating to high accuracy rates when using EBM. In Fig. 2b, we set a strict MSE threshold of 0.2 and check for number of experiments that reach this threshold by the end of all 3000 training epochs. Clearly, we can see strong evidence that PINNs with EB prior has much higher success rate in producing highly accurate σ reconstructions.

3.3 Comparison with baselines

We now compare the performance of our framework with some baselines; Bar&Sochen [1], Deep Galerkin Methods (DGM) [10] by using phantoms from Fig. 1. We train the baselines and our

Table 1: Evaluation of PINNs augmented with EB priors across various metrics and frameworks

Phantom	Metric	DGM	DGM With EB prior	Bar&Sochen	Bar&Sochen With EB prior	Our method	Our method With Prior
Ω_1	MSE↓	3.26 ± 0.028	0.13 ± 0.002	2.57 ± 0.041	0.16 ± 0.026	0.13 ± 0.001	$\mathbf{0.03 \pm 0.001}$
	PSNR↑	8.84 ± 0.038	22.77 ± 0.079	9.89 ± 0.068	22.02 ± 0.645	22.79 ± 0.034	$\mathbf{29.94 \pm 0.146}$
	MDE↓	2.29 ± 0.024	0.36 ± 0.003	1.84 ± 0.028	0.23 ± 0.002	0.35 ± 0.002	$\mathbf{0.12 \pm 0.014}$
Ω_2	MSE↓	2.00 ± 0.013	0.08 ± 0.002	1.64 ± 0.019	0.2 ± 0.036	0.07 ± 0.001	$\mathbf{0.01 \pm 0.0001}$
	PSNR↑	15.04 ± 0.028	29.07 ± 0.099	15.92 ± 0.052	25.23 ± 0.779	29.68 ± 0.034	$\mathbf{38.45 \pm 0.052}$
	MDE↓	1.47 ± 0.013	0.26 ± 0.004	1.1 ± 0.02	0.19 ± 0.001	0.25 ± 0.003	$\mathbf{0.12 \pm 0.011}$

model while using the same hyperparameters (see Appendix Table 3), with and without the EB prior. We report scores on MSE, peak signal-to-noise ratio (PSNR) and mean difference error (MDE is error in means of σ over all $(x, y) \in \Omega$ and $d\Omega$) to evaluate the quality of predicted σ solution. All

experiments are repeated 10 times and the averaged metrics along with standard deviation errors on each metric are presented. The main observation from Table 1 is that, we can greatly improve accuracy of PINNs by incorporating EB priors. Lastly, we can easily plug the prior into any PINN-based framework to greatly improve their performances, showcasing our framework's generalizability.

4 Conclusion

Physics-informed neural networks are an important category of data-driven PDE solvers. However, for more complicated problems PINNs are not stable and robust. In this work, we look at the EIT problem and show that we can improve the stability and robustness of PINNs training via a Bayesian approach. We describe a data-driven prior using energy-based methods, which can easily be used with the other loss term to robustly train PINNs. In the EIT setting, our experimental result also show that PINNs converge faster and also to a more accurate solution when trained with prior.

Impact Statement

Physics-informed neural networks (PINNs) have many applications in scientific problems including the medical imaging. Our Bayesian framework expedites training of PINNs and makes them more robust and stable, which in-turn can facilitate the training of PINNs for a broader use. However, we use a data-driven prior which may not easily obtainable for every application. In our work, we discuss training the prior for electrical impedance tomography, which itself is an important problem in medical tomographic imaging.

References

[1] Leah Bar and Nir Sochen. Strong solutions for pde-based tomography by unsupervised learning. *SIAM Journal on Imaging Sciences*, 14(1):128–155, 2021.

[2] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. *Advances in Neural Information Processing Systems*, 32, 2019.

[3] Lawrence C Evans. *Partial differential equations*, volume 19. American Mathematical Soc., 2010.

[4] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[6] Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

[7] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

[8] Maziar Raissi, Paris Perdikaris, and George 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.

[9] Samuli Siltanen, Jennifer Mueller, and David Isaacson. An implementation of the reconstruction algorithm of a nachman for the 2d inverse conductivity problem. *Inverse Problems*, 16(3):681, 2000.

[10] Justin Sirignano and Konstantinos Spiliopoulos. Dgm: A deep learning algorithm for solving partial differential equations. 2017.

[11] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

[12] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020.

[13] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

[14] Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021.

[15] Yuxin Wu and Kaiming He. Group normalization, 2018.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [N/A]
- Did you include the license to the code and datasets? [N/A]
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

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)? [Yes] We did not share code but plan to in the future. All our experiments are reproducible, see Appendix.
   (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)? [Yes]
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? [Yes]
   (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]

A Appendix

The model architectures and training implementation details for u-Net, σ-Net and EBM are shared here.

A.1 Semi-Inverse Implementation Details

We present some additional phantoms in Fig. 3 that were used to train the σ-Net PINN and their corresponding predicted solutions in row 2.

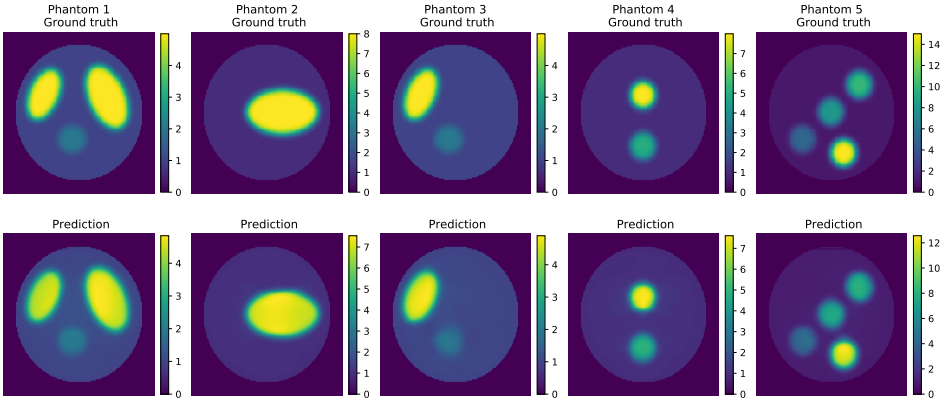


Figure 3: Comparison of ground truth σ and predictions from PINN with EB prior.

As for implementation and reproducible details, we train σ-Net such that it learns the conductivity distribution over all given mesh points to solve the semi-inverse problem. σ-Net incorporates the main PDE and the Nuemann BCs seen in Eq. 6 as a part of its training objective in order learn the conductivity inside Ω. However, the problem is known to be illposed and thus needs strong regularizers to improve the quality of reconstructions in conjunction with $\mathcal{L}_\theta$. For instance, we regulate the norm of gradients $\nabla_{x,y}\sigma$ inside Ω to promote sparse edges in predictions, and we penalize any conductivity prediction of less the one (= conductivity of vacuum) using $\mathcal{L}^h_{\text{hinge}} = \max(0, 1-\sigma_h) \quad \forall h \in \{\Omega \cup \partial\Omega\}$. The combined σ-Net training objective assimilating the elliptical PDE, its BC, and all the aforementioned mentioned regularizers is given as follows:

$$\mathcal{L}_{\theta_{\sigma\text{-Net}}} = \mathcal{L}_\theta + \frac{1}{|\partial\Omega|}\sum_{b\in\partial\Omega_b}|\sigma_b - \sigma^*_{\partial\Omega_b}| + \frac{\tau}{|\Omega|}\sum_{d\in\Omega}|\nabla_{x,y}\sigma_d| + \frac{\upsilon}{|\Omega\cup\partial\Omega|}\sum_{h\in\{\Omega\cup\partial\Omega\}}\mathcal{L}^h_{\text{hinge}} + \zeta\left\|w_\sigma\right\|^2, \tag{8}$$

The common loss weighting penalties used by both the u-Net, σ-Net models are: $\alpha = 0.05$, $\beta = 0.05$, $M = 40$, $\delta = 0.1$ and $\zeta = 1e^{-6}$. The Nuemann loss penalty $\gamma = 0.1$ and Dirichlet loss penalty $\epsilon = 100$ are specific to the forward problem only. While, $\gamma = 1$, $\tau = 0.01$, $\nu = 10$ and lastly $\kappa = 0.0001$ are specific inverse problem. Additionally, both models use the same multi-layer perceptron (MLP) architecture with residual connections [14] and consist of 4 hidden layers with $\tanh$ activation and 64 neurons each. A single output neuron with no activation aids in predicting the solutions for given mesh points. We train both these models with a batch size of 1000 using ADAM optimizer for 3000 epochs with an initial learning rate of 0.005 and decay it exponentially with a rate of 0.9 over intervals of every 200 epochs. We use NVIDIA Titan RTX GPU for all our training purposes and the training time for forward and semi-inverse problems separately take around takes ~ 8 minutes for 3000 epochs.

A.2 Forward Problem Implementation and Results

For our forward problem setup, we assume that Neumann condition is imposed on the entire boundary where the function g in Eq. 2 is given as a trigonometric pattern [9] as follows:

$$g = \frac{1}{\sqrt{2\pi}}\cos\left(\eta\omega + \psi\right), \quad n \in \mathbb{Z} \tag{9}$$

where ω is the angle along $d\Omega$, η and ψ are the current frequency, phase respectively. We use one current pattern where $n = 1$ and $\psi = 0$ to train u-Net for learning forward solution, which will aid σ-Net in learning inverse problem, as indicated in previous sections.

We now show Table 2 with the evaluation results of u-Net for the forward problem on the considered phantoms which was trained using the following loss function:

$$\mathcal{L}_{\theta_{u\text{-Net}}} = \mathcal{L}_\theta + \frac{\epsilon}{|\partial\Omega_e|} \sum_{e \in \{\partial\Omega_e\}} |u_e - f_e| + \zeta \left\| w_u \right\|^2 \tag{10}$$

The second term of Eq. 10 enforces Dirichlet BC in Eq. 3 only on electrodes while the last term controls the weights of u-Net using L_2 regularization on θ_u. We study the quality of forward problem solutions from u-Net while adding random uniform noise to the 16 electrode measurements of various levels. As seen in these experiments, the quality degradation is obvious due to increasing levels of noise. To counter this effect, we tune the penalty weights α, β, γ, δ on individual loss terms in Eq. 10 for increasing noise levels to improve the quality of reconstructions. The weights on Dirichlet BC $\epsilon = 100$, L_2 model parameter regularization $\zeta = 1e-6$ are fixed for all experiments and phantoms. Additionally, we use the same parameter set for all phantoms while only tuning them for a given noise level, which showcases the robustness of our method.

Table 2: Performance evaluation of u-Net for solving Forward problem with various noise levels

Noise Std	Metric	Phantom 1	Phantom 2	Phantom 3	Phantom 4	Phantom 5
		Parameters: $\alpha = 0.05$, $\beta = 0.05$, $\gamma = 0.1$, $\delta = 0.1$				
0	MSE↓	0.00046	0.00091	0.00027	0.00058	0.00036
	PSNR↑	39.41	33.40	46.90	47.02	40.51
		Parameters: $\alpha = 0.1$, $\beta = 0.1$, $\gamma = 0.1$, $\delta = 0.1$				
0.1	MSE↓	0.0090	0.070	0.02076	0.0208	0.026
	PSNR↑	26.51	14.55	28.13	31.48	21.91
		Parameters: $\alpha = 0.05$, $\beta = 0.05$, $\gamma = 1$, $\delta = 0.1$				
0.25	MSE↓	0.0426	0.1665	2.667	1.722	0.039
	PSNR↑	16.70	13.854	7.04	12.30	20.22
		Parameters: $\alpha = 0.1$, $\beta = 0.05$, $\gamma = 1$, $\delta = 0.1$				
0.5	MSE↓	1.041	2.25	10.59	2.46	4.70
	PSNR↑	5.89	0.52	1.05	3.01	0.587

A.3 EB prior implementation and Evaluation

Implementation We train our EB prior on σ solutions by perturbing them with Gaussian noise of 20 noise scales $\mu_i \in [2, 0.01]$ in order to learn by DSM training. We create a deep convolutional neural network with multiple residual [5] connections inspired by the architectures in [11] and [12]. In particular, our model consists of a single 3×3 convolution layer followed by series of 16 convolutional residual and residual-downsampling blocks in the form of an encoder block of typical convolution encoders. Each of these blocks consist of convolutional and Group normalization [15] layers with a group size 32. We then learn the input features by gradually down-sampling them to smaller resolutions. Upon feature map reduction at the end of the last convolution layer, we introduce a single hidden dense layer of 256 units to jointly learn the compressed σ features along with the noise scales, followed by a final dense layer which outputs scalar energy values. We use ELU non-linearity for all our EBM layers except the last energy layer which has no activation. We then train this EBM model to denoise the noise perturbed σ solutions via the objective seen in Eq. 7 for 2000 epochs, with a batch size of 64, using ADAM optimizer with a fixed learning rate of 0.0001. The training time takes around 15 hours while using a NVIDIA Titan RTX GPU.

Evaluation After the EBM $E^*_\phi(\boldsymbol{\sigma}, \mu)$ is trained, we can generate samples by using annealed Langevin dynamics (LD) [11] sampling. We start from a fixed prior distribution such as uniform noise and initially run LD for 100 steps with a step size of s_{n_1}, using first μ_1 noise scale and draw samples from $E_\phi(., \mu_1)$ by adding Gaussian noise. Next we draw samples from $E_\phi(., \mu_2)$ by reducing the step size and refine the samples. We continue running LD until all the noise scales are used to sample and we finally arrive at the final step size $s_{n_L} = a\mu_i^2/\mu_L^2$, where $a = 0.0002$ in our inference run. During the final step of last noise scale μ_L, we perform 1 step of gradient descent instead of LD to obtain an better denoised version of the σ generations. In Fig. 4, we display some of these curated σ solutions along with their nearest neighbors in the training set which are obtained by calculating the L_2 distance between all training set examples.

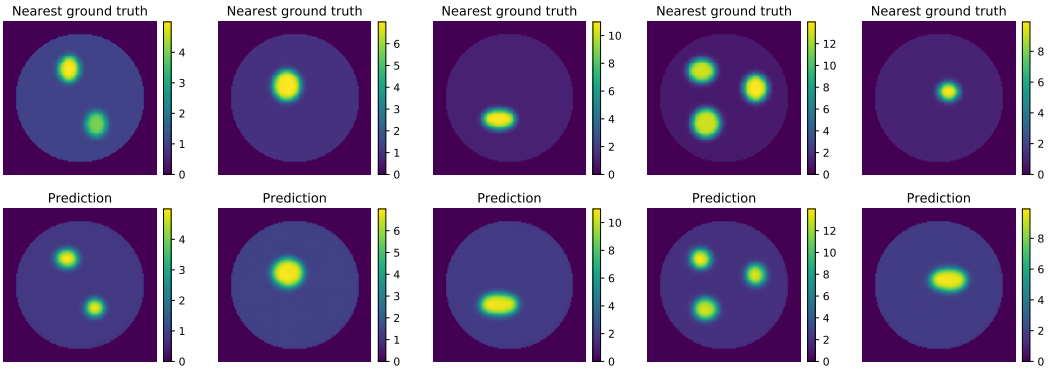


Figure 4: Comparison of training set samples to nearest generated σ using annealed Langevin dynamics.

A.4 Hypeparameter selection during baseline comparison studies

We present a full list of hyperparameters that are used to reproduce Table!1.

Table 3: List of common hyperparameters among various baselines

Baseline	α	β	M	γ	δ	ϵ	ζ	τ	ν	κ
DGM	1	0	0	1	1	1	0	0	0	0
Bar&Sochen	0.01	0.01	40	1	1	1	1e-8	0.01	0	0
Ours	0.05	0.05	40	1	0.1	100	1e-6	0.01	10	0.0001

A.5 Learning rate convergence

We aim to show through this study that, the learning rate vastly effects PINN's solution accuracy and this inherent instability can alleviated easily by introducing the EB priors. For this study, we keep all the parameters of our proposed method shown in Table 3 fixed and vary only the learning rate during σ-Net training with Eq. 8. We choose 0.0001, 0.001, 0.01 and 0.1 learning rates to study their effects on the semi-inverse PINN training with and without the EB prior. Note that, we still decay the all these chosen learning rates using exponential decay with rate 0.9 over every 200 epochs. The results of learning rate studies are presented in Fig. 5. As seen in this figure, the true σ value for the ellipsoid anomalies is 5 while the smaller circular anomaly in the bottom-middle has $\sigma = 2$. We can evidently see that the PINN without EB prior fails to converge to an accurate solution with regards to row three. More importantly, the maximum σ values for row one and two's color-bars indicate the failure to reach an accurate conductivity prediction.

Clearly, the PINN without EB prior never matches the solution accurately, under all learning rates. More adversely, the PINN outputs a trivial solution without our EB prior while using a learning rate that is either too low or too high respectively. On the other hand, PINNs augmented with EB prior always produce accurate solutions of slightly varying degree of MSE as seen in row three of the figure. More importantly, the training session with extremely large learning rate 0.1 still manages to produce a reconstruction although the accuracy is lower. This proves the robustness of our framework and instigates clear evidence that PINN training is being improved by a very large margin while incorporating EB priors. Other advantages such as, burden of choosing the optimal learning rate is also alleviated.

A.6 Semi-Inverse With Noisy Data

In the final set of experiments, we test the robustness of our framework while learning from noisy data. Initially, we add uniform noise of various standard deviation levels such as $0.1, 0.25, 0.5$, to the 16 boundary measurements. Then the u-Net learns the forward problem under these noisy settings and learns to approximate a noisy-forward solution (see Appendix A.2 for full evaluation of u-Net under noisy settings). While moving to the semi-inverse problem, we use data originating

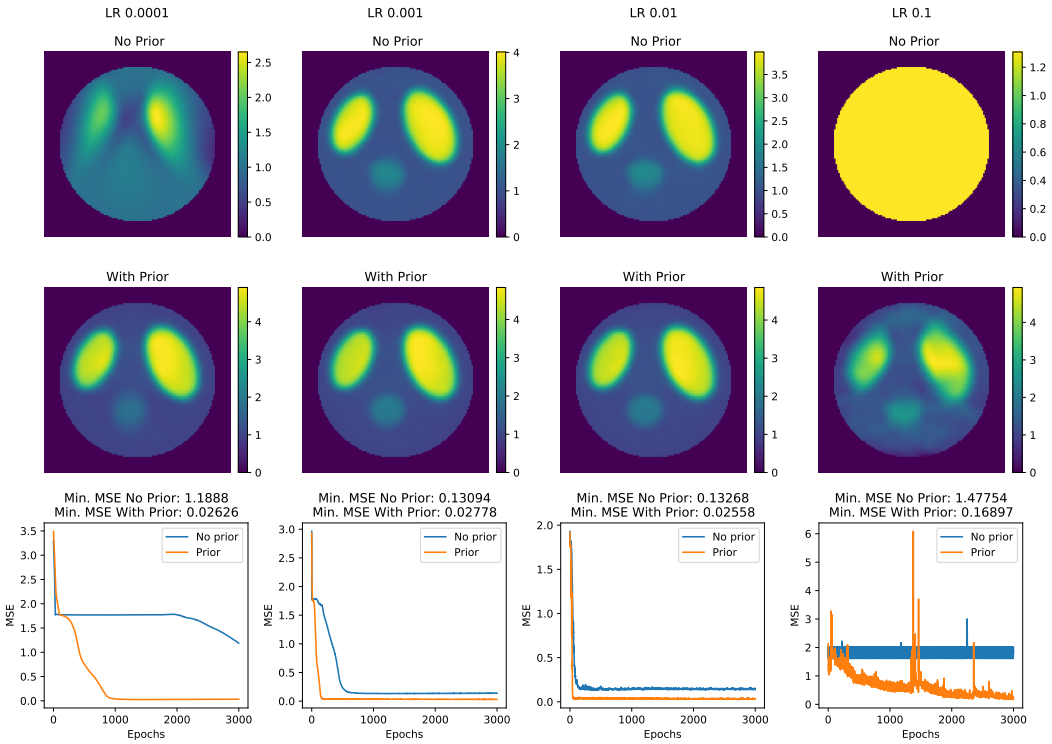


Figure 5: Training robustness of our proposed work with EB prior for various learning rates on Phantom 1.

from the noisy u-Net predictions and evaluate σ-Net performance and test its ability to withstand measurement noise with and without the EB prior. The evaluation results can be seen in Table 4. The trend of performance degradation is visible while the noise addition increases, more noticeable in some phantoms that others. Despite these noisy conditions, incorporating the EB prior during PINN training greatly improves the σ predictions and proves robustness of the method.

Table 4: Performance evaluation of σ-Net using noisy measurements

Noise	**Metric**	**Phantom 1**		**Phantom 2**		**Phantom 3**		**Phantom 4**		**Phantom 5**	
		w/o Prior	w/ Prior	w/o Prior	w/ Prior	w/o Prior	w/ Prior	w/o Prior	w/ Prior	w/o Prior	w/ Prior
0	MSE↓	0.13	**0.03**	0.4	**0.06**	0.04	**0.01**	0.07	**0.01**	0.57	**0.22**
	PSNR↑	22.79	**29.94**	22.04	**30.58**	28.20	**35.46**	29.68	**38.45**	25.95	**30.17**
	MDE↓	0.35	**0.12**	0.43	**0.1**	0.25	**0.11**	0.25	**0.12**	0.44	**0.08**
0.1	MSE↓	0.094	**0.03**	3.38	**1.64**	**0.02**	0.03	2.84	**0.01**	0.49	**0.11**
	PSNR↑	24.25	**29.91**	12.77	**15.91**	**31.50**	29.72	13.53	**36.78**	26.67	**33.09**
	MDE↓	0.29	**0.16**	1.09	**0.52**	0.21	**0.21**	0.02	**0.17**	0.42	**0.23**
0.25	MSE↓	0.17	**0.02**	0.58	**0.08**	0.03	**0.03**	0.09	**0.03**	0.67	**0.31**
	PSNR↑	21.55	**31.20**	20.40	**29.10**	28.95	**29.29**	28.46	**33.86**	25.24	**28.56**
	MDE↓	0.37	**0.23**	0.50	**0.29**	0.24	**0.16**	0.26	**0.17**	0.49	**0.37**
0.5	MSE↓	0.13	**0.06**	0.37	**0.13**	0.61	**0.57**	0.09	**0.02**	5.22	**4.39**
	PSNR↑	22.96	**26.05**	22.42	**26.84**	16.15	**16.41**	28.69	**35.43**	16.35	**17.09**
	MDE↓	0.34	**0.13**	0.42	**0.13**	0.14	**0.22**	0.25	**0.17**	0.90	**0.34**

This concludes the appendix section.