Plug-and-Play Latent Diffusion for Electromagnetic Inverse Scattering with Application to Brain Imaging

Rui Guo, *Member, IEEE*, Yi Zhang, *Member, IEEE*, Yhonatan Kvich, *Graduate Student Member, IEEE*, Tianyao Huang, *Member, IEEE*, Maokun Li, *Fellow, IEEE*, and Yonina C. Eldar, *Fellow, IEEE*

***Abstract*—Electromagnetic (EM) imaging is an important tool for non-invasive sensing with low-cost and portable devices. One emerging application is EM stroke imaging, which enables early diagnosis and continuous monitoring of brain strokes. Quantitative imaging is achieved by solving an inverse scattering problem (ISP) that reconstructs permittivity and conductivity maps from measurements. In general, the reconstruction accuracy is limited by its inherent nonlinearity and ill-posedness. Existing methods, including learning-free and learning-based approaches, fail to either incorporate complicated prior distributions or provide theoretical guarantees, posing difficulties in balancing interpretability, distortion error, and reliability. To overcome these limitations, we propose a posterior sampling method based on latent diffusion for quantitative EM brain imaging, adapted from a generative plug-and-play (PnP) posterior sampling framework. Our approach allows to flexibly integrate prior knowledge into physics-based inversion without requiring paired measurement-label datasets. We first learn the prior distribution of targets from an unlabeled dataset, and then incorporate the learned prior into posterior sampling. In particular, we train a latent diffusion model on permittivity and conductivity maps to capture their prior distribution. Then, given measurements and the forward model describing EM wave physics, we perform posterior sampling by alternating between two samplers that respectively enforce the likelihood and prior distributions. Finally, reliable reconstruction is obtained through minimum mean squared error (MMSE) estimation based on the samples. Experimental results on brain imaging demonstrate that our approach achieves state-of-the-art performance in reconstruction accuracy and structural similarity while maintaining high measurement fidelity.**

***Index Terms*—inverse scattering problems, brain imaging, posterior sampling, latent diffusion, plug-and-play.**

I. Introduction

Electromagnetic (EM) imaging is a crucial sensing modality in diverse fields due to its non-ionizing nature, material penetration capability, and cost-effectiveness [1]–[7]. Its application scenario includes security [3], [4], non-destructive evaluation [8], [9], and biomedical diagnostics [5]–[7]. Recently, it has emerged as a promising tool for early diagnosis and long-term monitoring of strokes [5], [10]. Stroke occurs when blood flow to the brain is disrupted by a blockage (ischemic stroke) or a rupture (hemorrhagic stroke). Rapid identification of stroke types and long-term monitoring are essential for improving outcomes [11]. Traditional imaging modalities—computed tomography (CT), positron emission tomography (PET), and magnetic resonance imaging (MRI)—have limitations: CT and PET involve ionizing radiation, and MRI lacks portability, limiting its use for bedside monitoring.

The physical foundation of EM stroke imaging is based on the fact that different types of strokes exhibit distinct electrical properties, such as permittivity and conductivity, compared to normal tissue. As EM waves penetrate the head, they are altered by the spatial variations in electrical property. The externally measured EM fields thus carry information about the internal electrical properties.

One way to retrieve the permittivity and conductivity is to formulate an EM inverse scattering problem (ISP) [12], which quantitatively solves for these parameters according to Maxwell's equations, given the temporal and spatial information of excitation sources and measurements. The resulting ISP is highly nonlinear due to multiple scattering among heterogeneous tissues and the skull, and ill-posed due to the limited degrees of freedom of the scattered field [13]. This leads to unstable solutions sensitive to measurement noises.

Classical model-based methods formulate the ISP as a regularized optimization or a Bayesian inference problem. Gauss-Newton [14], conjugate gradient [15], and alternating direction method of multiplier (ADMM) [16], are then used to iteratively update the unknowns. These approaches face limitations of non-convex landscapes. When initial guesses deviate significantly from the true solutions, local minima can trap the optimization process. In Bayesian inversion, sampling methods such as Markov chain Monte Carlo (MCMC) are used to sample from assumed distributions, but they are usually more computationally expensive than optimization approaches. Furthermore, both methods use oversimplified priors such as handcrafted regularizers (ℓ_2 norm or total variation (TV)) or simple distributions [14]–[19], which cannot capture realistic structures in complex materials. To better constrain target shapes, handcrafted reparameterization basis, such as geometry parameters [7], radial basis [20] and wavelets [21], have been proposed. However, these techniques are tailored to

This research was supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant No. 101000967), by the Israel Science Foundation (grant No. 3805/21, 536/22) within the Israel Precision Medicine Partnership (IPMP) program, and by the Manya Igel Centre for Biomedical Engineering and Signal Processing. (*Corresponding author: Rui Guo*)

Rui Guo, Yi Zhang, Yhonatan Kvich, and Yonina C. Eldar are with Faculty of Math and Computer Science, Weizmann Institute of Science, Rehovot 7610001, Israel.

Tianyao Huang is with the School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China.

Maokun Li is with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China.

specific scenarios and lack flexibility for broader applications.

Recently, deep learning methods were suggested to address ISPs [2], [22]–[26]. A straightforward approach is to learn the mapping between measurements and property maps [23]. However, purely data-driven models suffer from several limitations. Their black-box nature lacks physical interpretability, making them less suitable for security/clinical/industrial applications where reliability is crucial. Additionally, these methods exhibit setup rigidity—they require fixed measurement configurations during both training and inference, including transmitter/receiver geometries, excitation waveforms, sensor frequency responses, and radiation patterns [27]. This imposes strict consistency requirements on the measurement system, which are often impractical due to varying constraints in real world scenarios. Consequently, any modification to the measurement setup necessitates time-consuming dataset regeneration and neural network re-training, which involves thousands to millions of computationally intensive full-wave simulations.

To enhance interpretability and generalizability of data-driven approaches, many works integrate physical models with the deep learning process [2], [24], [25]. These hybrid methods are known to lead to more accurate networks while requiring fewer learned parameters compared to purely data-driven techniques [28], [29]. One branch of this integration is *physics-embedded deep learning* [26], where algorithms incorporate physical principles in different parts of the data-driven workflow, such as designing inputs and labels [30]–[32], loss functions [33], [34], and architectures of neural networks [35], [36]. Nevertheless, these methods still inherit the setup rigidity limitations from typical data-driven methods. Another promising branch is the *plug-and-play (PnP)* approach based on learned priors, which integrates neural network-expressed prior knowledge into classical physics-driven frameworks [37], [38]. Although this method has slower online execution, it offers high flexibility by learning priors from image-domain data (e.g., property maps) without being tied to specific measurement setups. The computational cost for training is substantially reduced by eliminating repeated full-wave simulations for each new configuration – the most time-consuming part in dataset preparation and *physics-embedding deep learning*. Moreover, since the imaging process remains primarily physics-driven, it generally guarantees high data fidelity, which is a critical criterion for EM imaging [24].

Representative PnP methods for EM imaging, which use neural network parameters to represent electrical property maps, include deep image prior (DIP) [39], [40] and generative model reparameterization (GMR) [38], [41], [42]. In DIP, the output of neural networks are considered property maps and connected to the forward modeling function. The neural network parameters are optimized through minimizing the data residual between measurements and forward simulations. Neural network architectures act as implicit regularizers during optimization. In GMR, pre-trained generative neural networks that embed complex prior knowledge are used to reparameterize the property maps. Instead of optimizing the whole network in DIP, the unknowns in GMR are only latent variables of the pre-trained generative models. For example, in [40], the authors apply DIP using an untrained U-net for electrical impedance tomography to produce conductivity maps with sharp boundaries. In [41], GMR is used to encode electrical properties into the latent space of a variational autoencoder (VAE), which yields clear structural features and improves resolution in microwave brain imaging. Both DIP and GMR treat the inverse problems as deterministic optimization problems, which may get trapped in local minima.

Another limitation of DIP and GMR is that they are maximum a posteriori (MAP)-based methods that reconstruct a single property map maximizing a posterior probability. This may produce overconfident estimates when the posterior distribution is multimodal. Using a minimum mean squared error (MMSE) metric can avoid overconfident solutions by computing the posterior expectation, which is usually approximated by the sample average of the posterior distribution. Advances in diffusion models enable posterior sampling with complex priors and tractable computational costs [43]–[46]. However, most existing works are developed for linear problems, and many of them lack theoretical guarantees. Recently, relying on the generative PnP and denoising priors, Bouman and Buzzard proposed an asymptotically exact method for posterior sampling based on iterative sampling [47]. This approach is extend to diffusion PnP (DPnP) posterior sampling for general inverse problems with a convergence guarantee [48], [49]. DPnP alternatively performs a likelihood step that generates samples consistent with the measurements, and a denoising diffusion sampler that enforces the prior constraint in the image pixel domain. This method was tested on vision problems, such as phase retrieval, quantized sensing, and super resolution [48], but was not considered to solve ISPs. Due to the drastically varying sensitivities of the measurement with respect to pixels of the electrical property map, directly applying pixel-based DPnP (P-DPnP) to ISP has intolerably slow convergence, which limits the practical applicability of this method to the physics-based inversion domain where the computational cost of forward modeling is very high.

To address the aforementioned challenges, we propose latent-DPnP (L-DPnP), where we aim to compute the MMSE estimate of the ISP under a Bayesian setting and achieve posterior sampling in a customized latent space that encodes domain-specific electrical parameter correlations. To the best of our knowledge, this is the first ISP posterior sampling PnP framework with learned priors. Our contributions are summarized as follows:

- The proposed L-DPnP framework enables posterior sampling for a wide range of computational imaging problems, by addressing the slow convergence limitation of P-DPnP caused by significantly varying sensitivities of unknowns.
- We apply the L-DPnP framework to quantitative EM inversion, which improves reconstruction accuracy by incorporating learned prior knowledge and by avoiding overconfident solutions in MAP estimation, and naturally provides uncertainty estimates.
- Through brain strokes imaging, we show that the proposed approach achieves superior reconstruction accuracy while maintaining high data fidelity comparable to other

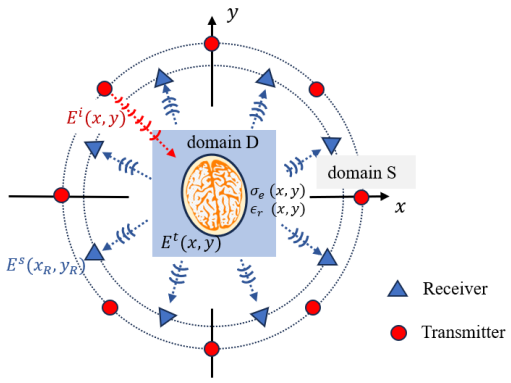

Fig. 1. Illustration of the EM imaging scenario in a 2-D setup. The transmitters sequentially transmit incident fields E^i into the investigation domain D that is characterized by permittivity and conductivity. The transmitters and receivers are located in domain S, where there is no permittivity or conductivity inhomogeneities.

state-of-the-art baselines.

Preliminary results were presented in a conference paper [50]. Herein, we introduce the derivation of L-DPnP, evaluate the method under broad testing scenarios, compare it against a wide range of existing techniques, provide a detailed explanation of its advantages, and describe implementation details not covered previously. Results in this paper show high robustness to noise and reconstruction accuracy, indicating its reliability for brain imaging.

The paper is organized as follows. Section II introduces preliminary knowledge about the EM wave physics and formulate the ISP. Section III establishes the L-DPnP framework and Section IV introduces detailed posterior sampling algorithms. In Section V, we validate our approach under two toy scenarios and apply it to brain imaging, comparing the results with existing methods. Section VI discusses the advantages and limitations of L-DPnP, as well as future directions. Section VII concludes the work.

II. Problem Formulation

In this section, we begin by presenting the measurement setup in our scenario and introducing the wave physics involved in EM imaging. We then formulate our inverse scattering problem that needs to be solved for imaging.

A. Measurement setup and wave physics

We consider a 2-D measurement scenario shown in Fig. 1, which corresponds to a non-contact sensing scenario where the antennas are situated in air or an impedance-matched medium. The target of interest, for example, human head, is located inside D, surrounded by N_T transmitters and N_R receivers placed in domain S. Domain S is homogeneous and shares the same electrical properties as the background of D.

EM wave propagation is governed by the Maxwell's equations. The transmitters illuminate the entire domain with *incident field* $E^i(x,y)$, where (x,y) denotes the coordinate of the 2-D space. When the target is illuminated, the spatially varying electrical properties—relative permittivity $\epsilon_r(x,y)$ and conductivity $\sigma_e(x,y)$—alters the field distribution, resulting in a *total field* $E^t(x,y)$. The receivers collect the *scattered fields* $E^s(x_R,y_R)$ at location (x_R,y_R), which are uniquely determined by the $E^t(x,y)$, $\epsilon_r(x,y)$ and $\sigma_e(x,y)$. The scattered fields collected under all transmissions constitute the observation data used for imaging.

We next describe this scattering process in matrix forms for a single transmission at one frequency. We first discretize domain D into $N_x \times N_y$ subunits, resulting in a discrete representation of relative permittivity and conductivity, i.e., $\boldsymbol{\epsilon_r} \in \mathbb{R}^{N_x \times N_y}$ and $\boldsymbol{\sigma_e} \in \mathbb{R}^{N_x \times N_y}$, respectively. Denote the index of transmitter, receiver, and frequency points to be p, q, and g, respectively, with $p = 1, ..., N_T$, $q = 1, ..., N_R$, and $g = 1, ..., N_f$. For the transmitter p at transmitting frequency f_g, the *incident field*, *total field* and *scattered field* are denoted by $\mathbf{E^i}_{p,g} \in \mathbb{C}^{N_x N_y}$, $\mathbf{E^t}_{p,g} \in \mathbb{C}^{N_x N_y}$ and $\mathbf{E^s}_{p,g} \in \mathbb{C}^{N_R}$, respectively. Their relations are given by [12], [51]

$$\mathbf{E^t}_{p,g} = \mathbf{E^i}_{p,g} + \mathbf{G}_D \boldsymbol{\chi}(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}) \mathbf{E^t}_{p,g}, \tag{1}$$

$$\mathbf{E^s}_{p,g} = \mathbf{G}_S \boldsymbol{\chi}(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}) \mathbf{E^t}_{p,g}, \tag{2}$$

where $\mathbf{G}_D \in \mathbb{C}^{N_x N_y \times N_x N_y}$ is the Green's function related to the specific background material and frequency but not related to receiver locations, and $\mathbf{G}_S \in \mathbb{C}^{N_R \times N_x N_y}$ is the Green's function matrix that propagates $\mathbf{E^t}_{p,g}$ to the receivers, and is related to the background material, frequency, and receiver locations. Finally, $\boldsymbol{\chi}(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}) \in \mathbb{C}^{N_x N_y \times N_x N_y}$ is called the contrast matrix, which is a diagonal matrix constructed from $\boldsymbol{\epsilon_r}$ and $\boldsymbol{\sigma_e}$, with the diagonal given by

$$\operatorname{diag}\big(\boldsymbol{\chi}(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e})\big) = \operatorname{vec}\left\{ \frac{\boldsymbol{\epsilon_r}}{\epsilon_{r,b}} - \mathrm{j}\frac{\boldsymbol{\sigma_e}}{2\pi f_g \epsilon_0 \epsilon_{r,b}} - 1 \right\} \in \mathbb{C}^{N_x N_y}. \tag{3}$$

Here $\epsilon_{r,b} \in \mathbb{C}$ is the complex relative permittivity of a known background, ϵ_0 is the permittivity in vacuum, $\operatorname{vec}\{\cdot\}$ is the vectorization operation, and $\mathrm{j} = \sqrt{-1}$ denotes the imaginary unit. Details about constructing $\mathbf{G}_D$, $\mathbf{G}_S$ and $\mathbf{E^i}_{p,g}$ and computing (1) and (2) can be found in [12].

For compact representation, we eliminate $\mathbf{E^t}_{p,g}$ by expressing it from (1) and substituting it into (2), yielding

$$\mathbf{E^s}_{p,g} = \mathbf{G}_S \boldsymbol{\chi}(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}) \big(\mathbf{I} - \mathbf{G}_D \boldsymbol{\chi}(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e})\big)^{-1} \mathbf{E^i}_{p,g}, \tag{4}$$

from which we can see that the relationship between $\mathbf{E^s}_{p,g}$ and $\boldsymbol{\chi}(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e})$ is nonlinear. We then define a nonlinear operator $F_{p,g}(\cdot)$ that maps $\boldsymbol{\epsilon_r}$ and $\boldsymbol{\sigma_e}$ to $\mathbf{E^s}_{p,g}$, that is,

$$F_{p,g}(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}) = \mathbf{G}_S \boldsymbol{\chi}(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}) \big(\mathbf{I} - \mathbf{G}_D \boldsymbol{\chi}(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e})\big)^{-1} \mathbf{E^i}_{p,g}. \tag{5}$$

During measurement, each transmitter sequentially illuminates domain D, and for every transmission, the resulting scattered fields are measured at multiple frequencies. This process yields an observation set comprising all transmitter-receiver-frequency pairs. By aggregating the scattered field data together, we have the observation vector $\mathbf{d}_{obs} = \{\{\mathbf{E}^s_{p,g}\}_{p=1}^{N_T}\}_{g=1}^{N_f} \in \mathbb{C}^{N_T N_R N_f}$. Similarly, we define a nonlinear operator $F(\cdot)$ that simultaneously models the scattered field generated by $\boldsymbol{\epsilon_r}$ and $\boldsymbol{\sigma_e}$ for all transmitters, receivers, and frequencies, that is, $F(\cdot) = \{\{F_{p,g}(\cdot)\}_{p=1}^{N_T}\}_{g=1}^{N_f}$, where $F(\cdot)$ is

first-order differentiable. Consequently, the measurement can be modeled as

$$\mathbf{d}_{obs} = F(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}) + \mathbf{n}_{obs}, \tag{6}$$

where $\mathbf{n}_{obs}$ denotes measurement noise.

B. Inverse scattering problem

We wish to solve for $\boldsymbol{\epsilon_r}$ and $\boldsymbol{\sigma_e}$ in (6) given $\mathbf{d}_{obs}$, from which the physical conditions of human tissues can be inferred. The inverse problem is severely ill-posed. This lead to numerous pairs of $\boldsymbol{\epsilon_r}$ and $\boldsymbol{\sigma_e}$ that approximately reproduce $\mathbf{d}_{obs}$ [13], whereas only a limited number of them correspond to the material in our imaging scenario. Therefore, incorporating prior knowledge is essential to eliminate solutions that do not align with real-world feasibility.

We adopt a probabilistic setting that describes the original deterministic ISP as a Bayesian inverse problem. In this setting, $\mathbf{d}_{obs}$, $\boldsymbol{\epsilon_r}$, and $\boldsymbol{\sigma_e}$ are treated as random variables, and prior distributions on $\boldsymbol{\epsilon_r}$ and $\boldsymbol{\sigma_e}$ serve to constrain the solution space.

Under this Bayesian setting, our goal is to estimate the optimal $\boldsymbol{\epsilon_r}$ and $\boldsymbol{\sigma_e}$ from the observed data $\mathbf{d}_{obs}$. The MMSE estimator is known to be optimal with respect to the ℓ_2 loss, which is defined as:

$$\begin{aligned}(\hat{\boldsymbol{\epsilon}}_r, \hat{\boldsymbol{\sigma}}_e)_{\text{MMSE}} &= \arg\min_{(\hat{\boldsymbol{\epsilon}}_r, \hat{\boldsymbol{\sigma}}_e)} \mathbb{E}\left[\|(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}) - (\hat{\boldsymbol{\epsilon}}_r, \hat{\boldsymbol{\sigma}}_e)\|^2\right] \\ &= \mathbb{E}\left[(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e})|\mathbf{d}_{obs}\right].\end{aligned} \tag{7}$$

The expectation in (7) is not analytically tractable. We therefore proceed to approximate it by taking the average of all samples from the posterior $p(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}|\mathbf{d}_{obs})$:

$$(\hat{\boldsymbol{\epsilon}}_r, \hat{\boldsymbol{\sigma}}_e)_{\text{MMSE}} \approx \frac{1}{M}\sum_{j=1}^{M}(\hat{\boldsymbol{\epsilon}}_r^{(j)}, \hat{\boldsymbol{\sigma}}_e^{(j)}), \tag{8}$$

where $(\hat{\boldsymbol{\epsilon}}_r^{(j)}, \hat{\boldsymbol{\sigma}}_e^{(j)}) \sim p(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}|\mathbf{d}_{obs})$ denotes the paired samples drawn from $p(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}|\mathbf{d}_{obs})$, indexed by j, with the number of all samples being M.

In the reminder of the paper, we are interested in obtaining these samples from the posterior $p(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}|\mathbf{d}_{obs})$ proportional to

$$p(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}|\mathbf{d}_{obs}) \propto p(\mathbf{d}_{obs}|\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}) \cdot p(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}), \tag{9}$$

where $p(\mathbf{d}_{obs}|\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e})$ is the likelihood described by the forward model (6), and $p(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e})$ is the prior distribution encoding any known statistical structure or physical constraints of the material properties. A key challenge arises from the fact that $p(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e})$ is not available in closed-form. Hence, we learn it from datasets. We next specifically address how to learn the prior and then perform posterior sampling using it.

III. L-DPnP framework

This section presents the L-DPnP framework that samples the posterior with the learned prior, which is rooted in P-DPnP [48], [49] but allows faster convergence and improved accuracy.

An overview of our framework is presented in Fig. 2, which consists of a pre-training stage and an inversion stage.

In the pre-training stage, given a dataset of paired samples $(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e})$, we train an autoencoder to obtain an encoder $\mathcal{E}$ and a decoder $\mathcal{G}$. The encoder $\mathcal{E}$ is used to map all $(\boldsymbol{\epsilon}_r, \boldsymbol{\sigma}_e)$ samples into corresponding latent representations $\mathbf{z}$. Based on the collection of $\mathbf{z}$ samples from the entire dataset, a latent diffusion model $\mathcal{S}$ is trained, which can output samples from the prior distribution of $\mathbf{z} \sim p(\mathbf{z})$.

In the inversion stage, given observed measurement data $\mathbf{d}_{obs}$, we independently draw M posterior samples in the latent space using an iterative sampling procedure. In each iteration, there is a likelihood step that promotes measurement fitness, and a prior step that encourages agreement with the learned prior. Specifically, the prior step is achieved by denoising the output of the likelihood step using the latent diffusion model $\mathcal{S}$. After obtaining all latent samples, we decode them using $\mathcal{D}$ to reconstruct the corresponding $(\boldsymbol{\epsilon}_r, \boldsymbol{\sigma}_e)$ pairs. The MMSE estimate is approximated by the average of all $(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e})$ pairs.

We begin by introducing how to jointly parameterize $\boldsymbol{\epsilon_r}$ and conductivity $\boldsymbol{\sigma_e}$ so that the unknowns jointly describe structures and materials. Based on this representation, the MMSE estimation problem is reformulated in the latent space. We then present an efficient method for estimating $\boldsymbol{\epsilon_r}$ and $\boldsymbol{\sigma_e}$ under this formulation.

A. Latent representation

Representing $\boldsymbol{\epsilon_r}$ and $\boldsymbol{\sigma_e}$ with discrete image pixels results in high dimensionality, redundant details, varying sensitivity to observations, and loss of intrinsic material relationships. As we explain in Section VI-A, this leads to high degrees of freedom and slow convergence in the solution.

To overcome this issue, we suggest learning the parameterization from datasets, that is, projecting $\boldsymbol{\epsilon_r}$ and $\boldsymbol{\sigma_e}$ onto the latent space of an autoencoder. This approach allows to represent the joint distribution of $\boldsymbol{\epsilon_r}$ and $\boldsymbol{\sigma_e}$ in a compact low dimensional space, empowering the parameterization to have material-specific characteristics [41].

We train a convolutional autoencoder on a dataset of $(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e})$ image pairs. The autoencoder contains an encoder and decoder, denoted as $\mathcal{E}$ and $\mathcal{G}$, respectively. The encoder learns to map the input $(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e})$ image pair to a low-dimensional latent variable $\mathbf{z}$, that is,

$$\mathbf{z} = \mathcal{E}(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}) \in \mathbb{R}^{N_u \times N_v} \tag{10}$$

with $N_u < N_x$ and $N_v < N_y$. The decoder learns to reconstruct the original pair from $\mathbf{z}$, that is,

$$(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}) = \mathcal{G}(\mathbf{z}). \tag{11}$$

The training of the autoencoder can be achieved by a self-supervised manner, that is, minimizing the reconstruction error between its output and input. The network structure and training details are given in Appendix A and B, respectively.

B. Inverse problem in the latent space

This subsection shows how the MMSE estimation (7) is achieved with latent the representation. Based on (8) and (11),

$$(\hat{\boldsymbol{\epsilon}}_r, \hat{\boldsymbol{\sigma}}_e)_{\text{MMSE}} \approx \frac{1}{M}\sum_{j=1}^{M}\mathcal{G}(\hat{\mathbf{z}}^{(j)}), \tag{12}$$

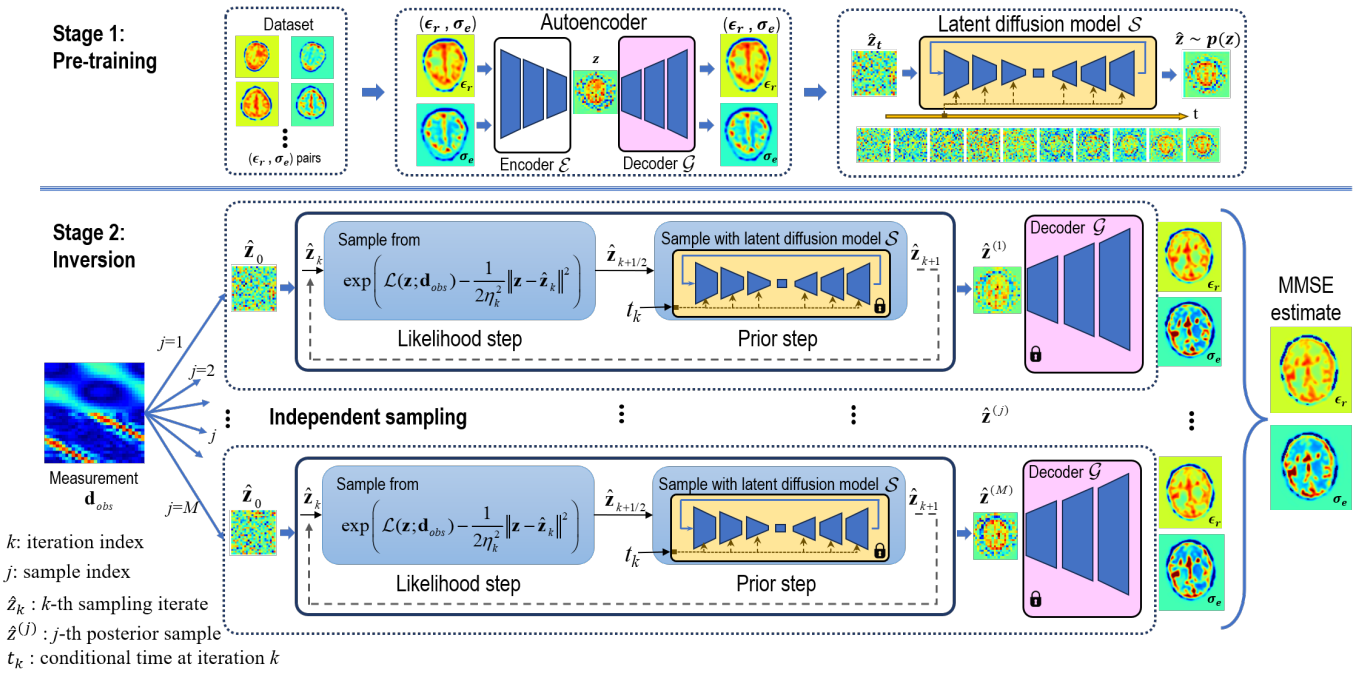

Fig. 2. Overview of the L-DPnP framework. Stage 1 (Pre-training): An autoencoder is trained on paired $(\boldsymbol{\epsilon}_r, \boldsymbol{\sigma}_e)$ to obtain a latent representation $\mathbf{z}$. A latent diffusion model is then trained on the $\mathbf{z}$-space to learn the prior distribution $p(\mathbf{z})$. Stage 2 (Inversion): Given observed measurement data $\mathbf{d}_{obs}$, posterior samples in the latent space are iteratively drawn by alternating between a likelihood step (enforcing data consistency) and a prior step (guided by the diffusion model). The decoded samples are averaged to approximate the MMSE estimate of $(\boldsymbol{\epsilon}_r, \boldsymbol{\sigma}_e)$.

where $\hat{\mathbf{z}}^{(j)}$ is the corresponding latent variable describing $(\hat{\boldsymbol{\epsilon}}_{\boldsymbol{r}}^{(j)}, \hat{\boldsymbol{\sigma}}_{\boldsymbol{e}}^{(j)})$. Next, we explain how to generate $\hat{\mathbf{z}}^{(j)}$ given measurement $\mathbf{d}_{obs}$.

The posterior of $\mathbf{z}$ given $\mathbf{d}_{obs}$ is proportional to

$$p(\mathbf{z}|\mathbf{d}_{obs}) \propto p(\mathbf{d}_{obs}|\mathbf{z}) \cdot p(\mathbf{z}), \tag{13}$$

where the prior $p(\mathbf{z})$ is induced by $p(\boldsymbol{\epsilon}_{\boldsymbol{r}}, \boldsymbol{\sigma}_{\boldsymbol{e}})$ and the encoder $\mathcal{E}$ (10). The likelihood $p(\mathbf{d}_{obs}|\mathbf{z})$ is derived from the following forward process equivalent to our original forward model:

$$\mathbf{d}_{obs} = F_{\mathcal{G}}(\mathbf{z}) + \mathbf{n}_{obs}, \tag{14}$$

where $F_{\mathcal{G}}(\cdot) = F(\mathcal{G}(\cdot)) : \mathbb{R}^{N_u \times N_v} \to \mathbb{C}^{N_T N_R N_f}$ denotes a nonlinear map composed with the pre-trained decoder $\mathcal{G}$ (11). Consequently, assuming the measurement noise is Gaussian with variance σ^2, the likelihood $p(\mathbf{d}_{obs}|\mathbf{z})$ is defined as

$$p(\mathbf{d}_{obs}|\mathbf{z}) \propto \exp(\mathcal{L}(\mathbf{z}; \mathbf{d}_{obs})) \tag{15}$$

with

$$\mathcal{L}(\mathbf{z}; \mathbf{d}_{obs}) = -\frac{1}{2\sigma^2}\|\mathbf{d}_{obs} - F_{\mathcal{G}}(\mathbf{z})\|^2. \tag{16}$$

We can therefore transform our problem to sampling the latent variable $\hat{\mathbf{z}}^{(j)}$ from the posterior (13) in terms of $F_{\mathcal{G}}(\cdot)$ and $\mathbf{z}$, after which the MMSE estimation (8) are computed based on (12).

The choice of $p(\mathbf{z})$ affects the algorithm used to estimate $\mathbf{z}$. For instance, when $p(\mathbf{z})$ is uniform or Gaussian, the posterior $p(\mathbf{z}|\mathbf{d}_{obs})$ becomes Gaussian. This allows to solve $\mathbf{z}$ through deterministic optimization by maximizing the posterior, as studied in [41]. However, in this work, we consider a more general setting where $p(\mathbf{z})$ may have multiple peaks or be arbitrarily shaped, which makes closed-form or optimization-based solutions infeasible.

C. Posterior sampling

We next focus on drawing samples from the posterior distribution in (13) using the likelihood (15) and the prior $p(\mathbf{z})$. This subsection assumes that $p(\mathbf{z})$ is known; the method for obtaining it is provided in Section IV-B. In the following, we describe the process for drawing a single sample and omit the superscript (j) when there is no ambiguity.

Standard sampling methods, such as Metropolis-Hastings algorithm [52], which requires a function proportional to the probability density, or Langevin algorithm [53], which relies on the log-density gradient, cannot be applied to sample from (13), because we have no closed-form expression for its density, its gradient, or any equivalent unnormalized function.

A result developed in [47] shows that a Markov chain (MC) constructed by alternately sampling from two proximal distributions—one derived from the prior and the other from the likelihood—converges to a stationary distribution that approximates the posterior. More specifically, the two proximal distributions of $p(\mathbf{d}_{obs}|\mathbf{z})$ and $p(\mathbf{z})$ are given by

$$q_0(\mathbf{z}|\mathbf{v}; \eta) \propto \exp\left(\mathcal{L}(\mathbf{z}; \mathbf{d}_{obs}) - \frac{1}{2\eta^2}\|\mathbf{z} - \mathbf{v}\|^2\right), \tag{17}$$

$$q_1(\mathbf{z}|\mathbf{v}; \eta) \propto p(\mathbf{z}) \exp\left(-\frac{1}{2\eta^2}\|\mathbf{z} - \mathbf{v}\|^2\right), \tag{18}$$

where $\mathbf{v}$ is a random vector and η is a parameter of the proximal distribution. When the new state of the MC is generated by sampling (17) and (18) conditioned by the previous state,

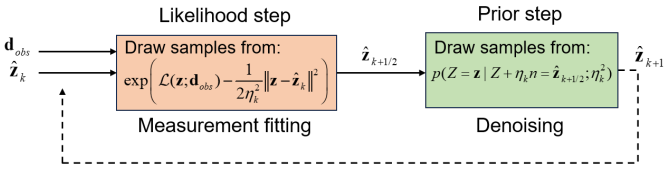

Fig. 3. One iteration of the iterative sampling approach. Given the iterate $\hat{\mathbf{z}}_k$ and the measurement $\mathbf{d}_{obs}$, the likelihood step samples an intermediate sample $\hat{\mathbf{z}}_{k+\frac{1}{2}}$ that promotes the measurement fitness. The prior step then generate a sample $\hat{\mathbf{z}}_{k+1}$ consistency with the prior, which can be viewed as a denoising step.

the distribution of the samples are close to the posterior as the chain is infinitely long and $\eta \to 0$. This result is important as it transforms the intractable task of posterior sampling into two simpler, tractable sampling problems.

Consequently, the samples of the MC chain are obtained by an iterative approach. Given an iterate $\hat{\mathbf{z}}_k$ at the k-th iteration, a new sample $\hat{\mathbf{z}}_{k+1}$ is generated by sequentially drawing from two distributions:

1) **Likelihood step:**

$$\hat{\mathbf{z}}_{k+\frac{1}{2}} \sim q_0(\mathbf{z}|\hat{\mathbf{z}}_k;\eta_k), \tag{19}$$

2) **Prior step:**

$$\hat{\mathbf{z}}_{k+1} \sim q_1(\mathbf{z}|\hat{\mathbf{z}}_{k+\frac{1}{2}};\eta_k), \tag{20}$$

where η_k is an annealing parameter in iterations. The likelihood step and prior step respectively promote consistency with the measurements and the prior knowledge.

In Fig. 3, we show a block diagram of the iterative approach at each iteration, explained in detail below. Given the measurement $\mathbf{d}_{obs}$ and the iterate $\hat{\mathbf{z}}_k$, the likelihood step (19) generates a sample $\hat{\mathbf{z}}_{k+\frac{1}{2}}$ that promotes measurement fitness $\mathcal{L}(\mathbf{z};\mathbf{d}_{obs})$ (16). This step can be viewed as sampling from an *approximate* posterior distribution where the prior $p(\mathbf{z})$ is replaced by a Gaussian distribution concentrated around the iterate $\hat{\mathbf{z}}_k$. Since $q_0(\mathbf{z}|\hat{\mathbf{z}}_k;\eta_k)$ is analytical, classical sampling algorithms can be employed to draw samples from it, as detailed in Section IV-A.

After $\hat{\mathbf{z}}_{k+\frac{1}{2}}$ is obtained, the prior step (20) draws $\hat{\mathbf{z}}_{k+1} \sim q_1(\mathbf{z}|\hat{\mathbf{z}}_{k+\frac{1}{2}};\eta_k)$. This step can be rigorously characterized by the denoising processing using a diffusion model [48]. To see this, one can consider the second term of (20), $\exp\left(-\frac{1}{2\eta_k^2}\|\mathbf{z}-\hat{\mathbf{z}}_{k+\frac{1}{2}}\|^2\right)$, as a likelihood that is Gaussian,

$$p(\hat{\mathbf{z}}_{k+\frac{1}{2}}|\mathbf{z}) \propto \exp\left(-\frac{1}{2\eta_k^2}\|\hat{\mathbf{z}}_{k+\frac{1}{2}}-\mathbf{z}\|^2\right). \tag{21}$$

Consequently, $q_1(\mathbf{z}|\hat{\mathbf{z}}_{k+\frac{1}{2}};\eta_k)$ is proportional to

$$p(\mathbf{z})p(\hat{\mathbf{z}}_{k+\frac{1}{2}}|\mathbf{z}) \propto p(Z=\mathbf{z}|Z+\eta_k\mathbf{n}=\hat{\mathbf{z}}_{k+\frac{1}{2}};\eta_k^2), \tag{22}$$

with $\mathbf{n}\sim\mathcal{N}(0,\mathbf{I})$ where $\mathcal{N}$ denotes the Gaussian distribution. The expression (22), derived from Bayes' rule, reformulates the sampling from (20) as a denoising problem. Specifically, the input $\hat{\mathbf{z}}_{k+\frac{1}{2}}$ is interpreted as a noisy sample corrupted with Gaussian noise of variance η_k^2, and the prior step acts as a denoiser that takes this noisy input and produces a clean estimate. Its relationship with diffusion models will be further explained in Section IV-B and IV-C.

We now summarize the L-DPnP framework introduced so far. Given a dataset of paired $\boldsymbol{\epsilon_r}$ and $\boldsymbol{\sigma_e}$ maps, we first train an autoencoder to obtain an encoder $\mathcal{E}$ and a decoder $\mathcal{G}$, and a latent diffusion model describing $p(\mathbf{z})$ (details in Section IV-B). Next, we draw M posterior samples in the latent space, that is, $\hat{\mathbf{z}}^{(j)}$ $(j=1,\ldots,M)$, by repeatedly sampling. Each sample is obtained by running the likelihood step and prior step for N_k (pre-defined) iterations using $\mathcal{G}$ and the measurement $\mathbf{d}_{obs}$, as detailed in Section IV. Finally, the estimates of permittivity and conductivity is computed as the average of the M samples according to (12). The entire procedure is outlined in Algorithm 1.

Algorithm 1 L-DPnP framework

Require: Dataset of $(\boldsymbol{\epsilon_r},\boldsymbol{\sigma_e})$ pairs.
Pre-training: Train an autoencoder on $(\boldsymbol{\epsilon_r},\boldsymbol{\sigma_e})$ pairs to obtain encoder $\mathcal{E}$ and decoder $\mathcal{G}$. Train a latent diffusion model $\mathcal{S}$ on all $\mathbf{z}=\mathcal{E}(\boldsymbol{\epsilon_r},\boldsymbol{\sigma_e})$ maps.
Input: Measurement data $\mathbf{d}_{obs}$, number of samples M, number of iterations N_k.
for $j=1$ to M **do**
  for $k=0$ to N_k-1 **do**
    Likelihood step (19):
    Draw $\hat{\mathbf{z}}_{k+\frac{1}{2}} \sim \exp\left(\mathcal{L}(\mathbf{z};\mathbf{d}_{obs})-\frac{1}{2\eta_k^2}\|\mathbf{z}-\hat{\mathbf{z}}_k\|^2\right)$,
    Prior step (22):
    Draw $\hat{\mathbf{z}}_{k+1} \sim p(Z=\mathbf{z}|Z+\eta_k n=\hat{\mathbf{z}}_{k+\frac{1}{2}};\eta_k^2)$,
  end for
  Save final sample: $\hat{\mathbf{z}}^{(j)}=\hat{\mathbf{z}}_{N_k}$,
end for
MMSE estimation: $(\hat{\boldsymbol{\epsilon}}_{\boldsymbol{r}},\hat{\boldsymbol{\sigma}}_{\boldsymbol{e}})_{\text{MMSE}} \approx \frac{1}{M}\sum_{j=1}^{M}\mathcal{G}(\hat{\mathbf{z}}^{(j)})$,
Output: $(\hat{\boldsymbol{\epsilon}}_{\boldsymbol{r}},\hat{\boldsymbol{\sigma}}_{\boldsymbol{e}})_{\text{MMSE}}$.

IV. Implementation of L-DPnP

This section presents the algorithm for drawing a single posterior sample, as summarized in Algorithm 2. We begin by presenting the numerical sampling algorithms for the likelihood step. Next, we introduce the learning of $p(\mathbf{z})$ using a diffusion model, which provides the foundation for implementing the prior step. Finally, we establish the connection between the diffusion model and the prior step, and detail the sampling procedure.

A. Numerical sampling in the likelihood step

The likelihood step is achieved using the Langevin dynamics algorithm [53], which transforms the sampling problem into solving a stochastic differential equation (SDE). Specifically, the SDE for sampling the likelihood (19) is given by:

$$\mathrm{d}\mathbf{z}_\tau = -\nabla\mathcal{L}(\mathbf{z}_\tau;\mathbf{d}_{\text{obs}})\,\mathrm{d}\tau + \frac{1}{\eta_k^2}(\mathbf{z}_\tau-\hat{\mathbf{z}}_k)\,\mathrm{d}\tau + \sqrt{2}\,\mathrm{d}\mathbf{w}_\tau, \tag{23}$$

Algorithm 2 L-DPnP posterior sampling (single sample)

Require: Pre-trained decoder $\mathcal{G}(\cdot)$, pre-trained score model $S(\cdot,\cdot)$, forward model $F_{\mathcal{G}}(\cdot)$.
Input: Measurements $\mathbf{d}_{obs}$, drift coefficient $f(t)$, diffusion coefficient $g(t)$, and transition probability variance $\beta^2(t)$.
Hyperparameters: The number of alternating the likelihood-prior step loops N_k, number of iteration N_τ in the likelihood step, number of iteration N_t in the prior step, annealing schedule η_k, and the discrete time inteval γ and δ in the likelihood step and the prior step.
Initialization: $\hat{\mathbf{z}}_0$.
Sampling:
for $k = 0$ **to** $N_k - 1$ **do**
  Likelihood step: Draw samples from (19) :
  $\mathbf{z}[0] = \hat{\mathbf{z}}_k$,
  $r = e^{-\gamma/\eta_k^2}$,
  for $n = 0$ to $N_\tau - 1$ **do**
    $\mathbf{n} \sim \mathcal{N}(0, \mathbf{I})$,
    $\mathbf{z}[n+1] = \eta_k^2(1-r)\nabla_{\mathbf{z}[n]}\mathcal{L}(\mathbf{z}[n]; \mathbf{d}_{obs}) + r\mathbf{z}[n]$
      $+ (1-r)\hat{\mathbf{z}}_k + \eta_k\sqrt{1-r^2}\mathbf{n}$,
  end for
  $\hat{\mathbf{z}}_{k+\frac{1}{2}} = \mathbf{z}[N_\tau]$.
  Prior step: Denoising with (22):
  $t_{N_t} = \beta^{-1}(\eta_k)$,
  $\delta = \frac{t_{N_t} - \varepsilon_t}{N_t}$,
  $\mathbf{z}[N_t] = \hat{\mathbf{z}}_{k+\frac{1}{2}}$,
  for $n = N_t$ to 1 **do**
    $\mathbf{n} \sim \mathcal{N}(0, \mathbf{I})$,
    $t_n = t_{N_t} - (N_t - n)\delta$,
    $\mathbf{z}[n-1] = \mathbf{z}[n] + \big(g^2(t)\mathcal{S}(\mathbf{z}[n], t_n) - f(t_n)\mathbf{z}[n]\big)\delta$
      $+ g(t_n)\sqrt{\delta}\mathbf{n}$,
  end for
  $\hat{\mathbf{z}}_{k+1} = \mathbf{z}[0]$.
end for
Output: $\hat{\mathbf{z}} = \hat{\mathbf{z}}_{N_k}$.

where τ is the time notation, and $\mathbf{w}_\tau$ represents a standard Wiener process. Following [48], we apply the exponential integrator to discretize (23) since the linear drift $\frac{1}{\eta_k^2}(\mathbf{z}_\tau - \hat{\mathbf{z}}_k)$ may be large. In the discrete domain, we represent the time as $\tau_n = n\gamma$, where $n = 0, \ldots, N_\tau$ with γ being the time interval, and the corresponding discrete variable $\mathbf{z}_\tau$ is denoted as $\mathbf{z}[n]$. Based on (23), the sample at $\tau_{n+1} = (n+1)\gamma$ is computed by

$$\begin{aligned}\mathbf{z}[n+1] = \eta_k^2(1-r)\nabla_{\mathbf{z}[n]}\mathcal{L}(\mathbf{z}[n]; \mathbf{d}_{obs}) + r\mathbf{z}[n] \\ + (1-r)\hat{\mathbf{z}}_k + \eta_k\sqrt{(1-r^2)}\mathbf{n},\end{aligned} \tag{24}$$

where $r = e^{-\gamma/\eta_k^2}$ and $\mathbf{n} \sim \mathcal{N}(0, \mathbf{I})$. We initialize $\mathbf{z}[0] = \hat{\mathbf{z}}_k$, and after N_τ iterations of the update, we obtain $\mathbf{z}[N_\tau]$. We treat this result as $\hat{\mathbf{z}}_{k+\frac{1}{2}}$, that is, $\hat{\mathbf{z}}_{k+\frac{1}{2}} = \mathbf{z}[N_\tau]$.

B. Diffusion models: preparation for the prior step

In Section III-C, we have seen that the prior step works as a denoiser that generates a clean sample given noisy input. This process can be rigorously described by the sampling (reverse) process of a diffusion model [48]. We next introduce how a diffusion model learns the prior $p(\mathbf{z})$.

Diffusion models estimate the prior distribution of a signal by training neural networks to reverse a gradual noising process. This technique achieves state-of-the-art performance in various generative tasks [54]–[57]. It can be modeled by a perturbation process and a reverse process.

In the perturbation process, the datapoints are perturbed with a stochastic process over time $t \in [0, 1]$, according to the following SDE

$$\mathrm{d}\mathbf{z}_t = f(t)\mathbf{z}_t\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}_t, \tag{25}$$

where $\mathbf{z}_t$ is the trajectory of random variables in a stochastic process, $f(t)$ and $g(t)$ is the drift and diffusion coefficient of the SDE, respectively, and $\mathbf{w}_t$ is a standard Wiener process. At $t = 0$, $\mathbf{z}_0 \equiv \mathbf{z}$, and distribution $p_0(\mathbf{z})$ represents the prior $p(\mathbf{z})$: $p_0(\mathbf{z}) \equiv p(\mathbf{z})$. The SDE gradually transforms $\mathbf{z}_0$ into noise, with transition distribution

$$p_{0t}(\mathbf{z}_t|\mathbf{z}_0) = \mathcal{N}(\mathbf{z}_t|\alpha(t)\mathbf{z}_0, \beta^2(t)\mathbf{I}). \tag{26}$$

Here $\alpha(t)$ and $\beta(t)$ can be derived analytically from $f(t)$ and $g(t)$ [58].

The reverse process, used for sampling, transforms a noisy sample $\mathbf{z}_t$ back to a data sample $\mathbf{z}_0 \sim p_0(\mathbf{z})$ by solving the reverse-time SDE:

$$\mathrm{d}\mathbf{z}_t = \big(f(t)\mathbf{z}_t - g^2(t)\nabla_{\mathbf{z}_t}\log p_t(\mathbf{z}_t)\big)\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}_t, \tag{27}$$

where $\mathrm{d}t$ and $\bar{\mathbf{w}}_t$ respectively denotes the time differential and a standard Wiener process running in reverse time. In particular, the score function denoted by $\nabla_{\mathbf{z}_t}\log p_t(\mathbf{z}_t)$ is approximated training a time-dependent score model $\mathcal{S}(\mathbf{z}_t, t)$ using denoising score matching [59] on samples $\{\mathbf{z}_0^{(i)}\}_{i=1}^{N_{\text{data}}} \sim p(\mathbf{z})$:

$$\begin{aligned}\min \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} \mathbb{E}_{t\sim\mathcal{U}[0,1]}\mathbb{E}_{\mathbf{z}_t^{(i)}\sim p_{0t}(\mathbf{z}_t^{(i)}|\mathbf{z}_0^{(i)})} \\ \Big[\|\mathcal{S}(\mathbf{z}_t^{(i)}, t) - \nabla_{\mathbf{z}_t^{(i)}}\log p_{0t}(\mathbf{z}_t^{(i)}|\mathbf{z}_0^{(i)})\|_2^2\Big],\end{aligned} \tag{28}$$

where superscript (i) indexes the training samples, $\mathcal{U}[0,1]$ denotes the uniform distribution over the time interval $[0,1]$, and $p_{0t}(\mathbf{z}_t|\mathbf{z}_0)$ is the transition distribution defined in (26). In practice, the clean training datapoint $\mathbf{z}_0$ is generated by the encoder $\mathcal{E}$ (10) and the noisy data $\mathbf{z}_t$ are generated according to the transition distribution (26).

After training, we obtain $\mathcal{S}(\mathbf{z}_t, t) \approx \nabla_{\mathbf{z}_t}\log p_t(\mathbf{z}_t)$. Plugging this into (27) and initializing the following SDE from $\mathbf{z}_T$ at time $T \in [0,1]$ enables the generation of samples consistent with $p(\mathbf{z})$:

$$\mathrm{d}\mathbf{z}_t = \big(f(t)\mathbf{z}_t - g^2(t)\mathcal{S}(\mathbf{z}_t, t)\big)\,\mathrm{d}t + g(t)\,\mathrm{d}\bar{\mathbf{w}}_t, \tag{29}$$

where T and $\mathbf{z}_T$ denote the starting time and initial condition of the process, respectively.

C. Numerical sampling in the prior step

We now establish the connection between (29) and the prior step (22). Recall that the prior step denoises the noisy input $\hat{\mathbf{z}}_{k+\frac{1}{2}}$ with variance η_k^2 to produce a clean estimate. The likelihood of generating $\hat{\mathbf{z}}_{k+\frac{1}{2}}$ from $\hat{\mathbf{z}}_k$, given in (21), has the same form as the transition distribution (26). If we solve (29) with $\mathbf{z}_T = \hat{\mathbf{z}}_{k+\frac{1}{2}}$, $\beta(T) = \eta_k$, and $\alpha(t) \equiv 1$, where $\alpha(t)$ and $\beta(t)$ are determined by $f(t)$ and $g(t)$, the solution process becomes equivalent to sampling from (22). An option of $f(t)$ and $g(t)$ is provided in Appendix B6.

The reverse-time SDE (29) is numerically solved by the Euler-Maruyama approach [60]. We denote the sample at discrete time points $t_n = n\delta + \varepsilon_t$ as $\mathbf{z}[n]$, where $n = N_t, \ldots, 0$, δ is the discrete time interval, and ε_t is a small positive number. By using the finite difference, the sample at $t_{n-1} = (n-1)\delta + \varepsilon_t$ is computed by

$$\begin{aligned}\mathbf{z}[n-1] = \mathbf{z}[n] + \big(g^2(t_n)\mathcal{S}(\mathbf{z}[n], t_n) - f(t_n)\mathbf{z}[n]\big)\delta \\ + g(t_n)\sqrt{\delta}\mathbf{n},\end{aligned} \tag{30}$$

where $\mathbf{n} \sim \mathcal{N}(0, \mathbf{I})$. We initialize $\mathbf{z}[N_t] = \hat{\mathbf{z}}_{k+\frac{1}{2}}$ and $t_{N_t} = \beta^{-1}(\eta_k)$. After N_t iterations of the update, we obtain $\mathbf{z}[0]$. The sample $\hat{\mathbf{z}}_{k+1}$ is taken as $\hat{\mathbf{z}}_{k+1} = \mathbf{z}[0]$.

V. Experiments

A. Datasets

We test our method on three datasets that have diverging characteristics, derived from MNIST [61], Fashion-MNIST [62], and ATLAS [63]. The first two datasets serve as algorithmic validation benchmarks [23], [30], [35]. The measurement environment in these two scenarios resembles those in industrial non-destructive evaluation or radar-based imaging.

Reconstructing MNIST digits can be challenging in microwave because the cavities (hollow or enclosed regions) within the digits act as resonators, which leads to strong multiple scattering phenomena that increase the nonlinearity of the inverse problem. Fashion-MNIST further extends the evaluation to inhomogeneous targets, allowing us to assess the model's ability to recover spatially varying materials.

Finally, we test our method on dataset derived from ATLAS (a clinical MRI dataset of anatomical brain images from stroke patients), which represents our target application in brain imaging. Accurate reconstruction in this setting is particularly challenging: brain tissues are highly inhomogeneous, exhibit strong attenuation of electromagnetic waves, and are enclosed by the skull, which reflects most of the incident energy. As a result, the scattered signals contain limited information about the interior, necessitating effective use of the measurement for high-fidelity reconstruction.

1) Property map generation: **MNIST** & **Fashion-MNIST**: the original images of size 28×28 are upscaled to $N_x \times N_y = 64 \times 64$. The pixel values are normalized to the range 1–2, which corresponds to the relative permittivity of the targets. The size of domain D is 0.30 m $\times$ 0.30 m. Conductivity is not considered in these two datasets, therefore the contrast matrix $\boldsymbol{\chi}$ is real-valued. The background relative permittivity is $\epsilon_{r,b} = 1$, corresponding to the air. The number of training samples is 60K.

ATLAS: The MRI images are segmented into five categories: skull, white matter, gray matter, cerebrospinal fluid (CSF), and the remaining parts including background and other tissues [41]. We synthesize strokes using morphological irregularities with different electrical properties. We employ data augmentation, including scaling, rotation, flipping, and translation, to simulate a broader diversity of head shapes and poses. Based on [41] and [10], we assign each tissue type and stroke with specific permittivity and conductivity values. Note that while the original image resolution in ATLAS is 1 mm$\times$ 1mm, we interpolate it onto a 3 mm $\times$ 3mm mesh, with $N_x \times N_y = 96 \times 96$, to accommodate the resolution limits of microwaves and ensure a reasonable runtime for EM simulations. The size of domain D is 0.28 m $\times$ 0.28 m. The complex relative permittivity of the background medium is $\epsilon_{r,b} = 44 - 17.9\text{j}$, corresponding to a matching medium. The number of training samples is 160K.

2) Synthetic measurement setup: **MNIST** & **Fashion-MNIST**: The number of transmitters and receivers is set to $N_T = 16$, and $N_R = 32$, respectively. The simulation frequency points are 1 GHz and 3 GHz. Both the transmitting and receiving antennas are uniformly distributed on a circle centered at the center of domain D, with a radius of 2 m. The incident fields are generated by the 2-D point source model. For each target, the simulated scattered field, denoted as$\mathbf{d}^{\mathbf{s}}$, are corrupted with $nl = 4\%$ Gaussian noise to produce the synthetic observation data:

$$\mathbf{d}_{obs} = \mathbf{d}^{\mathbf{s}} + nl \cdot std(\text{Real}(\mathbf{d}^{\mathbf{s}})) \cdot \mathbf{n}_1 + \text{j} \cdot std(\text{Imag}(\mathbf{d}^{\mathbf{s}})) \cdot \mathbf{n}_2, \tag{31}$$

where $std(\cdot)$ means computing the standard deviation, Real$(\cdot)$ and Imag$(\cdot)$ means taking the real and imaginary part, respectively, and $\mathbf{n}_1, \mathbf{n}_2 \sim \mathcal{N}(0, \mathbf{I})$.

ATLAS: We set $N_T = 16$, and $N_R = 32$. The simulation frequency points are 0.6 GHz and 1 GHz. The transmitting and receiving antennas are uniformly distributed on the circle with a radius of 0.14 m. Incident fields are generated by the 2-D point source model. For each brain slice, the synthetic observation data are corrupted with $nl = 20\%$ Gaussian noise according to (31), corresponding to a reasonable noise level in real cases.

B. Methods for comparison

This subsection introduces other popular methods we use for comparison. They can be grouped into three categories, i.e., standard learning-free techniques, model-based deep learning, and learning-based PnP. For each category, we provide two comparison baselines. All relevant neural network architectures and implementation details can be found in Appendix A and B, respectively.

1) Standard learning-free techniques: The first baseline is an iterative optimization method that minimizes the data fidelity term together with an ℓ_2 norm of the spatial gradient of permittivity and conductive maps. This approach, denoted as "Occam", was proposed in geophysical EM inversion to produce smooth reconstructions [64]. The second baseline is the iterative reconstruction method that employs total variation (TV) regularization using ADMM, denoted as "TV-ADMM" [16], [65].

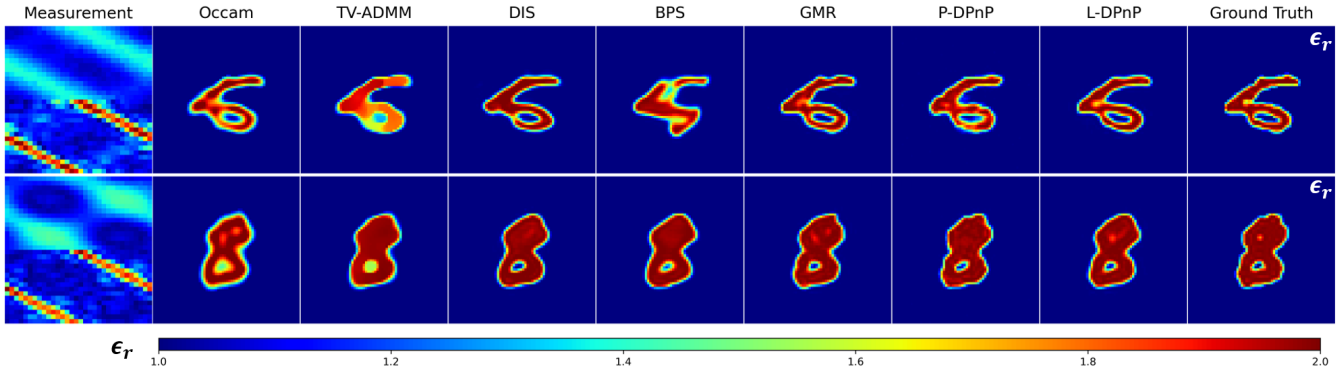

Fig. 4. Results of permittivity (ϵ_r) reconstruction for MNIST targets. Only the amplitude of complex electrical fields are visualized in the measurement maps, with the upper half presenting the scattered field at 1 GHz, and the lower half presenting the scattered field at 3 GHz.

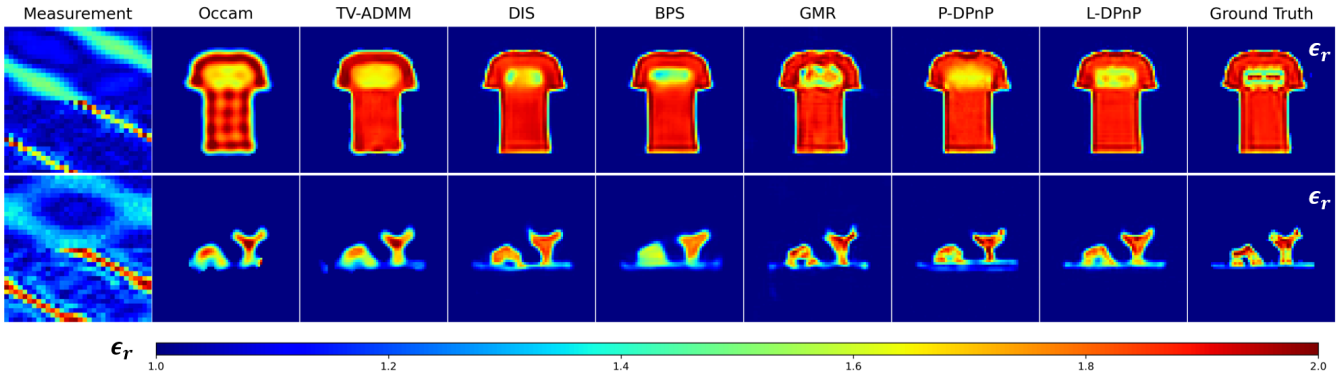

Fig. 5. Results of permittivity (ϵ_r) reconstruction for Fashion-MNIST targets. Only the amplitude of complex electrical fields are visualized in the measurement maps, with the upper half presenting the scattered field at 1 GHz, and the lower half presenting the scattered field at 3 GHz.

2) Model-based deep learning: The first baseline, called "DIS", uses the difference between measured total fields and simulated total fields – computed with a guessed permittivity map [35] – as the input of a convolutional neural network to predict the true permittivity map. It is noted that when the initial guess is set to the background, this approach is equivalent to the method proposed in [66]. The second baseline, "BPS" [30], feeds a post-processed map after back-projection [12] into a convolutional neural network to predict the permittivity map.

3) Learning-based PnP: The first baseline is based on generative model reparameterization, namely "GMR", where the permittivity and conductivity map is projected to latent variables that are solved via deterministic optimization using gradient descent methods. The idea was first proposed in [38] and applied to microwave imaging in [41]. The second baseline is the application of the P-DPnP [48] to our problem. We mention that directly applying P-DPnP without masking highly sensitive regions may lead to failure due to the challenges discussed in Section VI-A. The results presented in ATLAS examples are obtained by performing the likelihood step with an appropriate mask.

C. Results

1) Visual results: The inversion results of different algorithms tested on the three datasets are shown in Fig. 4, Fig. 5, and Fig. 6, respectively. Due to high computational cost, we take the average of five samples, that is, $M = 5$ in (8), as the MMSE estimate. It can be seen that our algorithm can recover both lossless and lossy targets with the finest details and the least artifacts, for example, the enclosed regions in Fig. 4, the inhomogeneous material in Fig. 5, and the skull, brain tissues, and stroke regions in Fig. 6.

We see that the learning-free methods cannot handle multi-scale details: the images have either over-smooth boundaries or insufficient inhomogeneity. Purely data-driven methods tend to generate unrealistic structures particularly when the target has complex features, especially under noisy environment (see DIS for brain imaging). GMR may be trapped in local minima, as illustrated in the second examples in Fig. 6. P-DPnP tends to produce images with low saturation and less detailed structures, indicating less accurate reconstruction of the electrical properties. In comparison, our method yields reconstructions that most closely match the ground truth. For brain imaging, Fig. 6 shows that other comparison methods may cause false alarms or fail to detect strokes while our approach reliably reconstruct property maps indicating the existence of strokes.

2) Performance metric: We report the performance using three metrics: RMSE in the measurement domain, RMSE in the reconstruction domain, and SSIM in the reconstruction domain. These metrics evaluate inversion quality from different yet complementary perspectives. RMSE in the mea-

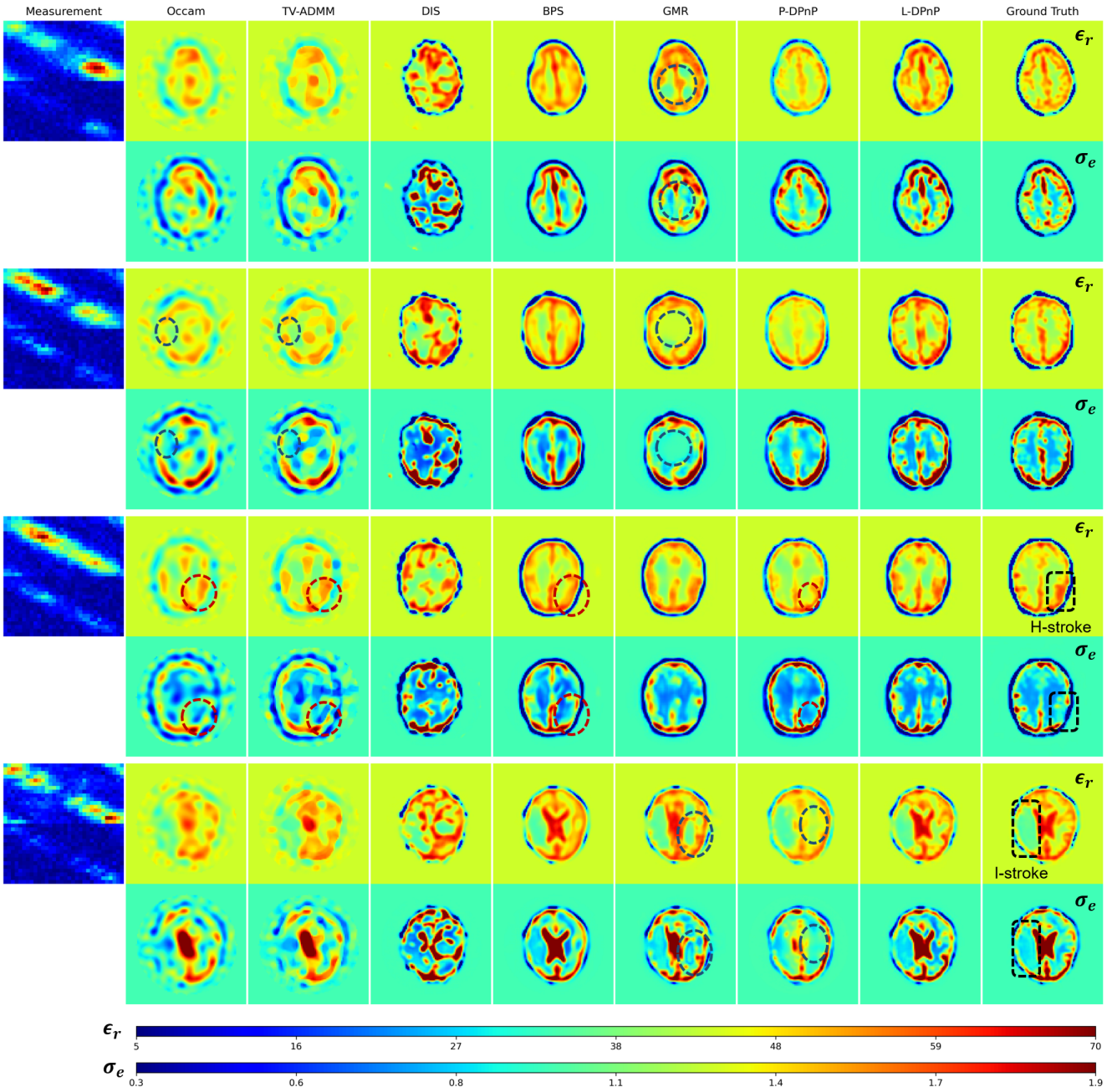

Fig. 6. Results of permittivity (ϵ_r) and conductivity (σ_e) reconstruction for brains. From top to bottom, two normal cases, one case with hemorrhagic stroke (h-stroke), and one case with ischemic stroke (i-stroke) are shown. The h-stroke (red color in ϵ_r, and turquoise color in σ_e) is characterized by higher permittivity and conductivity than brain tissues. The i-stroke (turquoise color in both ϵ_r and turquoise σ_e) is characterized by lower permittivity but higher conductivity than tissues. DIS cannot reflect meaningful results under high noise level. In other comparison methods, we point out some false alarms in the normal cases (dark blue circles) and missed detection (dark red circles) .

surement domain quantifies how well the simulated waves match the actual measurements, which is critical because a visually pleasing permittivity map might still be physically implausible if it fails to accurately reproduce the measurement data. In practical detection scenarios, measurement fitness often serves as the primary quantitative criterion before the ground truth is precisely known. However, relying solely on RMSE in the measurement domain cannot fully assess imaging performance; for example, a low measurement-domain RMSE does not necessarily guarantee a meaningful reconstruction (as observed in learning-free methods). We further use RMSE in the reconstruction domain to quantify recovered electrical property that determines the material composition, and SSIM in the reconstruction domain to indicate how realistic of the reconstruction in terms of structures. Together, achieving low RMSE in the measurement domain, along with low RMSE and high SSIM in the reconstruction domain, indicates a reliable reconstruction that is physically plausible, quantitatively accurate, and structurally realistic.

The comparisons of different methods are summarized in

TABLE I
QUANTITATIVE EVALUATION OF SEVEN METHODS ON THREE DATASETS (EACH 200 SAMPLES). **BOLD**: BEST, UNDERLINE: SECOND BEST.

Dataset	Metric	Method						
		Occam	TV-ADMM	DIS	BPS	GMR	P-DPnP	L-DPnP
MNIST	RMSE (Measurement)↓	**0.053**	0.060	0.070	0.096	0.055	0.058	0.059
	RMSE (Reconstruction)↓	0.040	0.053	0.033	0.046	0.029	0.033	**0.027**
	SSIM (Reconstruction)↑	0.976	0.964	0.983	0.973	0.984	0.983	**0.987**
Fashion-MNIST	RMSE (Measurement) ↓	0.054	**0.054**	0.079	0.085	0.055	0.057	0.062
	RMSE (Reconstruction)↓	0.056	0.053	0.043	0.050	0.043	0.045	**0.039**
	SSIM (Reconstruction)↑	0.854	0.881	0.908	0.887	0.899	0.903	**0.913**
ATLAS	RMSE (Measurement)↓	0.180	**0.165**	0.405	0.248	0.245	0.222	0.192
	RMSE (Reconstruction)↓	0.129	0.128	0.156	0.086	0.089	0.099	**0.075**
	SSIM (Reconstruction)↑	0.695	0.711	0.814	0.910	0.897	0.886	**0.922**

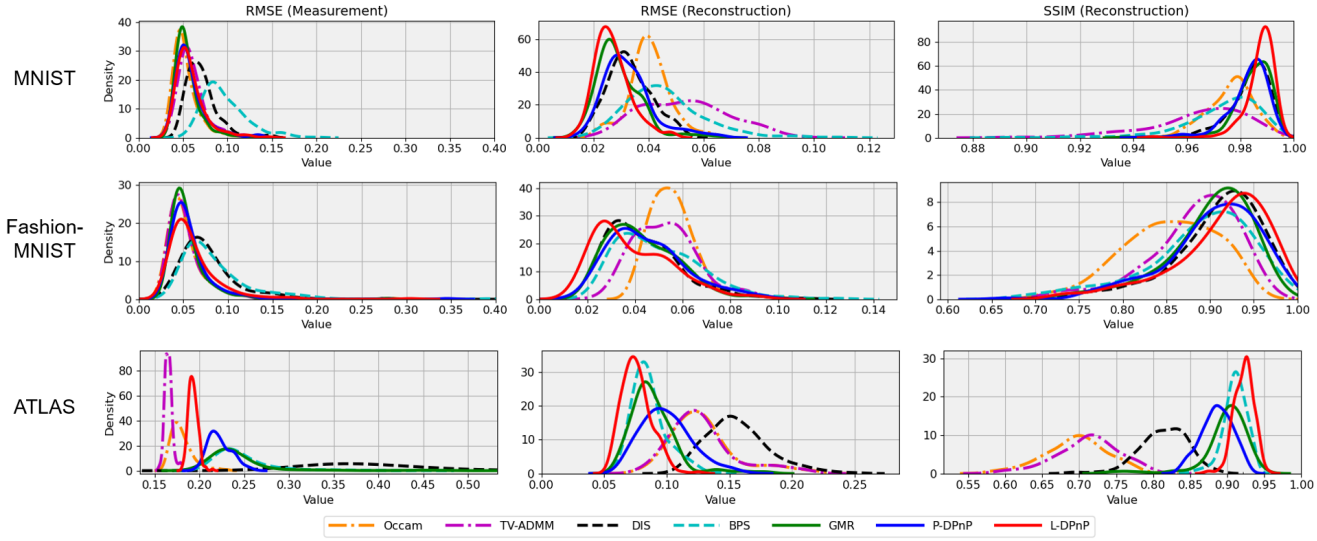

Fig. 7. Statistics of RMSE in the measurement domain, RMSE and SSIM in the reconstruction domain for different methods tested on the three datasets.

Tab. I. We tested 200 cases for each dataset due to the high computational complexity of the problem. We observe that physics-driven, learning-free methods yield the lowest measurement RMSE, however, their reconstruction RMSE and SSIM are unsatisfactory due to the absence of prior knowledge. In particular, when imaging the brain with high noises, the physics-driven methods achieve much lower measurement RMSE than the preset noise level, which is a sign of overfitting to noise. Model-based deep learning methods achieve improved reconstruction RMSE and SSIM compared to learning-free methods, but they exhibit higher measurement RMSE, which raises concerns about reliability. PnP methods offer a favorable balance between measurement and reconstruction fidelity. Specifically, our method achieves the highest reconstruction quality while maintaining measurement consistency near the noise level. The statistics of the metrics in Fig. 7 show that our methods exhibit small RMSE and SSIM variance, indicating high inversion reliability.

3) Imaging uncertainty: Compared to deterministic optimization methods, sampling methods enable uncertainty quantification. As an additional demonstration, we present the standard deviation map corresponding to the reconstructed results from previous examples, see Fig. 8. Regions with high standard deviation consistently correspond to areas exhibiting large reconstruction errors.

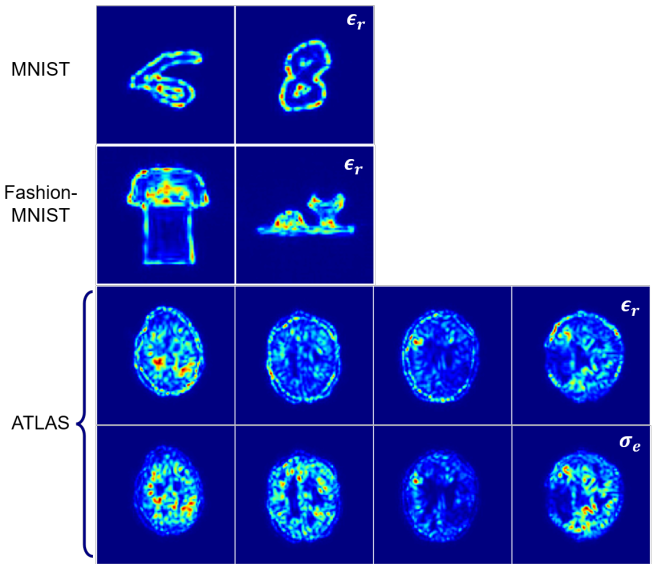

Fig. 8. Standard deviation of the reconstructed permittivity (ϵ_r) and conductivity (σ_e) corresponding to previous examples.

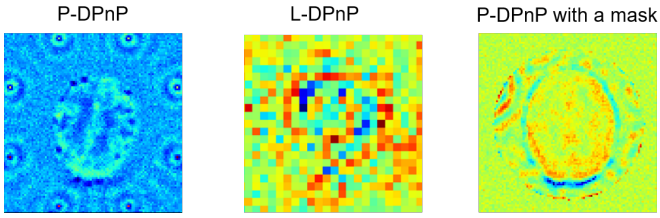

Fig. 9. Output of the likelihood step using P-DPnP [48] and L-DPnP. The first map shows the output of the likelihood step in P-DPnP, which exhibits large values near high-sensitivity regions. The circular areas indicate the receivers' location. The second map shows the output of the likelihood step in L-DPnP, which is less structured thanks to the rescale of sensitivity in the latent space. The third map shows that masking out the receivers mitigates sensitivity variation, but is still less balanced and appears ripples due to the sensitivity difference of EM waves within a wavelength.

VI. Discussions

A. Benefits of the latent space

L-DPnP offers three advantages over P-DPnP. First, by operating in the latent space rather than pixel space, the latent diffusion model achieves higher efficiency in both training and sampling [67]: eliminating imperceptible pixel-level details reduces data dimensionality and accelerates convergence in training, while the lower-dimensional latent space enables faster sample generation.

Second, sampling in the latent space allows to generate more physically grounded targets than in the pixel space, which supports reliable interpretation. This is because the autoencoder jointly maps structural and material information into a coherent latent representation. By sampling latent codes within this manifold and decoding them, the permittivity and conductivity maps preserve realistic spatial–physical patterns. In contrast, pixel-wise representation of permittivity and conductivity treats each image pixel independently, which increase the risk of producing physically implausible material.

Most importantly, L-DPnP overcomes slow convergence of P-DPnP, thanks to more scalable sensitivities in the latent space than in the pixel space. Recall that (24) is computed based on the gradients that indicate the sensitivity of unknown parameters to measurements. The pixel-based discretization, which is based on spatial coordinates, causes the sensitivity maps to have large variations, because (1) the wave amplitude decays exponentially with distance, (2) the wave's phase and amplitude change at different rates within each wavelength, and (3) the wave attenuation differs spatially with the material's conductivity. Consequently, the output of likelihood step in the pixel space, as shown in the first map of Fig. 9, tends to exhibit large values near high-sensitivity regions. The P-DPnP algorithm preferentially updates high-sensitivity regions while low-sensitivity regions are updated more slowly. Considering the large computational cost of our forward problem, this slow-converging sampling process becomes impractical in real-world applications. In our experiments, we cannot get an acceptable sample within a reasonable running time (up to $N_k = 200$, around 6 hours).

In contrast, in L-DPnP, the encoder and decoder act like regularizers that normalize latent variable sensitivity by using fewer parameters to represent only essential features. This produces a less structured and more balanced gradient map, as shown in the second map of Fig. 9. Empirically, the output of the likelihood step in latent space does not have large variances, therefore the latent variables can update at similar rates during sampling. It is relatively easy to adjust a proper stepsize in the likelihood step to achieve fast convergence ($N_k = 20$, around 25 minutes).

To mitigate the effect of varying sensitivity, we apply the mask on the gradient map to remove the high sensitivity regions near receivers in this paper. However, as shown in the third column of Fig. 9, the result in the likelihood step is still less balanced due to varying sensitivity across pixels. This adjustment accelerates convergence compared to sampling without the mask, however, our experiments show that the reconstruction accuracy is still not as accurate as L-DPnP with the same N_k.

B. Limitations and future work

L-DPnP faces higher computational challenges than deterministic optimization such as GMR [41]. The majority of the computational cost is incurred in the likelihood step, which requires a total of $N_k \times N_\tau \times M$ gradient evaluations to generate an MMSE estimate. For denser computational mesh, the time and memory demands can become prohibitive. To address this, future work should explore distributed parallelization of forward solvers and the use of surrogate or reduced-order forward models to enable more efficient gradient computations.

The prior step, while benefiting from diffusion-based denoising, also incurs cost proportional to the number of diffusion timesteps. With recent advances in diffusion models, integrating fast sampling schemes such as DDIM [68] or DPM-Solver [69] could further accelerate convergence.

Extending L-DPnP to 3-D EM imaging [41] is an important next step, as real-world targets often exhibit complex 3-D structures that cannot be accurately captured by 2-D approximations. However, this extension remains challenging since it requires designing scalable network architectures for learning the prior and faster 3-D solvers that can handle the increased memory and sampling time. Beyond EM, applying L-DPnP to the ISP of other imaging modalities such as ultrasound or electrical impedance tomography [70], [71] offers the opportunity of super-resolution and uncertainty analysis in these domains. Moreover, L-DPnP for multi-modality joint imaging [72]–[74], where the latent diffusion model could fuse information from different physical properties, may yield more robust and complementary reconstructions in the future.

VII. Conclusion

We propose a new posterior sampling approach to solve highly nonlinear and ill-posed EM ISP based on latent diffusion. This approach contains a the likelihood step that promotes measurement fitness and a prior step that enforces prior knowledge in the latent space. It does not require paired training data and can flexibly adapt to heterogeneous measurement setups. The framework is modular, which enables the flexible integration of physical models into the generative models to leverage prior knowledge to improve reconstruction performance. The samples yield low quantitative error and

high-fidelity structures. After sampling, our approach enables MMSE estimation and uncertainty quantification, which improves the reliability of EM imaging. Experimental results demonstrate that our approach achieves state-of-the-art performance in terms of reconstruction fitness and structural similarity with high measurement fidelity, outperforming existing physics-driven and learning-based methods.

APPENDIX

For notational consistency, we use $\mathbf{X}$ and $\mathbf{X}_{out}$ to denote the input and output of neural networks that operate in the pixel-based imaging (including Occam, TV-ADMM, DIS, BPS, P-DPnP), and $\mathbf{Z}$ and $\mathbf{Z}_{out}$ to denote the input and output of neural networks operating in the latent space (including GMR and DPnP). The exact physical meanings and construction details of these inputs and outputs in different neural networks are provided in Appendix B.

A. Neural network architectures

Notations for basic neural network operation:

- $\text{Conv2D}_{C_{out}}$: 2D convolution with a kernel size of 3×3, stride 1, and C_{out} output channels.
- $\text{Conv2D}_{C_{out},/2}$: 2D convolution with stride 2 for downsampling, producing C_{out} output channels.
- $\text{Conv2D}_{C_{out},\times 2}$: First applies upsampling with a scale factor of 2, followed by 2D convolution producing C_{out} output channels.
- $\text{GroupNorm}_{C_{out}/2}$: Group normalization with $C_{out}/2$ groups.
- SiLU: Sigmoid Linear Unit activation function.

1) Autoencoder for latent representation: The architecture's diagram is shown in Fig. 10, which is modified from [41]. We apply a VAE as the autoencoder, suggested by [67]. The encoding and decoding blocks are built with fully convolutional layers. Different from [41], where latent variables are 1D vectors, our latent map is 2D to adapt to latent diffusion. Denote the input and output tensor as $\mathbf{X}, \mathbf{X}_{out} \in \mathbb{R}^{B\times C\times H\times W}$, respectively, where B, C, H, and W is the batch size, input channel number, height and width, respectively. A mean tensor $\mathbf{Z}_\mu$ and logarithmic variance tensor $\mathbf{Z}_{var}$ is obtained by going through cascaded encoding blocks $\text{E-Block}^{(i)}_{C_{in}\to C_{out}}$, where C_{in} and C_{out} represents the number of input and output channels of the block. Following the reparameterization trick [75] of training a VAE, a latent sample $\mathbf{Z}$ is generated according to the mean and variance tensor. Then $\mathbf{Z}$ is decoded back by cascaded decoding blocks $\text{D-Block}^{(i)}_{C_{in}\to C_{out}}$, yielding the output $\mathbf{X}_{out}$. This process is summarized in Architecture 1.

As suggested by [67], we utilize less aggressive downsampling in the autoencoder to attain a high decoding resolution. While [67] applied the vector quantized VAE (VQ-VAE) [76] to avoid posterior collapse, our VAE adopts a small penalty on the KL divergence of the evidence lower bound (ELBO) [75], [77] to address this problem. This autoencoder is expected to generate high-resolution images and have a more complete latent space than a classical autoencoder. After it is trained, the encoder $\mathcal{E}$ is used to generate datasets for training the score model, and the decoder is integrated into the forward model (6) to form $F_{\mathcal{G}}(\cdot)$.

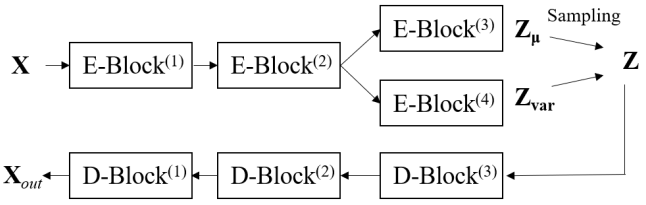

Fig. 10. Diagram of the autoencoder's architecture.

Architecture 1 Autoencoder for latent representation

Input: $\mathbf{X} \in \mathbb{R}^{B\times C\times H\times W}$.
$\mathbf{X}_1 = \text{E-block}^{(2)}_{16\to 64}(\text{E-block}^{(1)}_{C\to 16}(\mathbf{X}))$,
$\mathbf{Z}_\mu = \text{E-block}^{(3)}_{64\to 1}(\mathbf{X}_1)$,
$\mathbf{Z}_{var} = \text{E-block}^{(4)}_{64\to 1}(\mathbf{X}_1)$,
$\mathbf{n} \sim (0, \mathbf{I})$,
$\mathbf{Z} = \mathbf{Z}_\mu + \exp(0.5\mathbf{Z}_{var}) \cdot \mathbf{n}$,
$\mathbf{X}_2 = \text{D-Block}^{(3)}_{1\to 64}((\mathbf{Z}))$,
$\mathbf{X}_{out} = \text{D-block}^{(1)}_{16\to C}(\text{D-block}^{(2)}_{64\to 16}(\mathbf{X}_2))$.
Output: $\mathbf{X}_{out} \in \mathbb{R}^{B\times C\times H\times W}$.

The $\left\{\text{E-Block}^{(i)}_{C_{in}\to C_{out}}\right\}_{i=1,2}$ ($C_{out} = 16$ for $i = 1$ and $C_{out} = 64$ for $i = 2$) are similarly constructed. Denote the input and output of the blocks as $\mathbf{X}^{block}_{in}$ and $\mathbf{X}^{block}_{out}$, respectively[1]. Their structures are presented in Architecture 2. Similarly, architectures of $\left\{\text{E-Block}^{(i)}_{64\to 1}\right\}_{i=3,4}$ are shown in Architecture 3.

Architecture 2 $\left\{\text{E-Block}^{(i)}_{C_{in}\to C_{out}}\right\}_{i=1,2}$

Input: $\mathbf{X}^{block}_{in} \in \mathbb{R}^{B\times C_{in}\times H\times W}$.
$\mathbf{X}_1 = \text{SiLU}(\text{GroupNorm}_{C_{out}/2}(\text{Conv2D}_{C_{out}}(\mathbf{X}^{block}_{in})))$,
$\mathbf{X}_2 = \text{SiLU}(\text{GroupNorm}_{C_{out}/2}(\text{Conv2D}_{C_{out}}(\mathbf{X}_1))$,
$\mathbf{X}_2 = \text{SiLU}(\text{GroupNorm}_{C_{out}/2}(\text{Conv2D}_{C_{out}}(\mathbf{X}_2))$,
$\mathbf{X}^{\text{res}}_1 = \mathbf{X}_1 + 0.1\mathbf{X}_2$,
$\mathbf{D}_2 = \text{SiLU}(\text{Conv2D}_{C_{out}}(\text{SiLU}(\text{Conv2D}_{C_{out}}(\mathbf{X}^{\text{res}}_1))))$,
$\mathbf{D}_2 = \text{Conv2D}_{C_{out},/2}(\mathbf{D}_2)$,
$\mathbf{X}^{block}_{out} = \mathbf{D}_1 + 0.1\mathbf{D}_2$.
Output: $\mathbf{X}^{block}_{out} \in \mathbb{R}^{B\times C_{out}\times H/2\times W/2}$.

Architecture 3 $\left\{\text{E-Block}^{(i)}_{64\to 1}\right\}_{i=3,4}$

Input: $\mathbf{X}^{block}_{in} \in \mathbb{R}^{B\times C_{in}\times H\times W}$.
$\mathbf{X}_1 = \text{SiLU}((\text{Conv2D}_{32}(\mathbf{X}^{block}_{in})))$,
$\mathbf{X}^{block}_{out} = \text{Conv2D}_1(\text{SiLU}((\text{Conv2D}_{16}(\mathbf{X}_1))))$.
Output: $\mathbf{X}^{block}_{out} \in \mathbb{R}^{B\times C_{out}\times H\times W}$.

The architectures of the decoder blocks $\text{D-Block}^{(3)}_{1\to 64}$ and $\left\{\text{D-Block}^{(i)}_{C_{in}\to C_{out}}\right\}_{i=1,2}$ ($C_{out} = C$ for $i = 1$ and $C_{out} = 16$ for $i = 2$) are presented in Architecture 4 and 5, respectively.

[1]The intermediate variable symbols are local to each network block. Identical symbols in different blocks do not refer to the same variable and should be interpreted according to context.

Architecture 4 D-Block$^{(3)}_{1\to 64}$

Input: $\mathbf{X}_{in}^{block} \in \mathbb{R}^{B\times 1\times H\times W}$
$\mathbf{X}_1 = \text{SiLU}((\text{Conv2D}_{16}(\mathbf{X}_{in}^{block})))$,
$\mathbf{X}_{out}^{block} = \text{Conv2D}_{64}(\text{SiLU}((\text{Conv2D}_{32}(\mathbf{X}_1))))$.
Output: $\mathbf{X}_{out}^{block} \in \mathbb{R}^{B\times 64\times H\times W}$.

Architecture 5 $\left\{\text{D-Block}^{(i)}_{C_{in}\to C_{out}}\right\}_{i=1,2}$

Input: $\mathbf{X}_{in}^{block} \in \mathbb{R}^{B\times C_{in}\times H\times W}$.
$\mathbf{X}_1 = \text{SiLU}(\text{GroupNorm}_{C_{out}}(\text{Conv2D}_{2C_{out}}(\mathbf{X}_{in}^{block})))$,
$\mathbf{X}_2 = \text{SiLU}(\text{GroupNorm}_{C_{out}}(\text{Conv2D}_{2C_{out}}(\mathbf{X}_2)))$,
$\mathbf{X}_2 = \text{SiLU}(\text{GroupNorm}_{C_{out}}(\text{Conv2D}_{2C_{out}}(\mathbf{X}_2)))$,
$\mathbf{X}_1^{\text{res}} = \mathbf{X}_1 + 0.1\mathbf{X}_2$,
$\mathbf{U}_1 = \text{Conv2D}_{C_{out},\times 2}(\mathbf{X}_1^{\text{res}})$,
$\mathbf{U}_2 = \text{SiLU}(\text{Conv2D}_{C_{out}}(\text{SiLU}(\text{Conv2D}_{C_{out},\times 2}(\mathbf{X}_1^{\text{res}}))))$,
$\mathbf{U}_2 = \text{Conv2D}_{C_{out}}(\mathbf{U}_2)$,
$\mathbf{X}_{out}^{block} = \mathbf{U}_1 + 0.1\mathbf{U}_2$.
Output: $\mathbf{X}_{out}^{block} \in \mathbb{R}^{B\times C_{out}\times 2H\times 2W}$.

2) Score model for L-DPnP: The latent diffusion model's architecture is based on U-Net. Time embedding is used to handle the temporal dynamics of the diffusion process. After it is trained, we obtain the score model $\mathcal{S}$ in (29).

Denote the input and output tensor as $\mathbf{Z}, \mathbf{Z}_{out} \in \mathbb{R}^{B\times 1\times H\times W}$, where B, H, and W is the batch size, height and width, respectively. The modified U-net is composed of cascaded down-sampling blocks DS-Block$^{(i)}_{C_{in}\to C_{out}}$, up-sampling blocks US-Block$^{(i)}_{C_{in}\to C_{out}}$, and output decoding block D-Block$^{o}_{32\to 1}$, conditioned by the time-embedding vector $\mathbf{V}(t) \in \mathbb{R}^{B\times 512}$. The architecture's diagram is shown in Fig. 11, with details in Architecture 6.

In Architecture 6, time embedding is achieved by Gaussian Fourier projection. Given a batch of time values $t \in \mathbb{R}^B$, the projection generates high-dimensional features $\mathbf{V}_p(t) \in \mathbb{R}^{B\times 512}$ using fixed (unlearnable) random frequencies $\mathbf{W} \in \mathbb{R}^{256}$. $t\mathbf{W}$ is the element-wise multiplication (broadcasted across the batch dimension). The final time embedding $\mathbf{V}(t)$ is obtained after a fully connected layer $\text{Dense}_{512\to 512}$, whose input and output dimension are both 512.

Architecture 6 Score model for L-DPnP

Input: $\mathbf{Z} \in \mathbb{R}^{B\times 1\times H\times W}$, $t \in \mathbb{R}^B$.
$\mathbf{V}_p(t) = \text{concat}[\sin(2\pi t\mathbf{W}), \cos(2\pi t\mathbf{W})]$,
$\mathbf{V}(t) = \text{SiLU}(\text{Dense}_{512\to 512}(\mathbf{V}_p(t)))$,
$\mathbf{Z}_1^{skip}, \mathbf{Z}_1 = \text{DS-Block}^{(1)}_{C\to 32}(\mathbf{Z}, \mathbf{V}(t))$,
$\mathbf{Z}_2^{skip}, \mathbf{Z}_2 = \text{DS-Block}^{(2)}_{32\to 64}(\mathbf{Z}_2, \mathbf{V}(t))$,
$\mathbf{Z}_3^{skip}, \mathbf{Z}_3 = \text{DS-Block}^{(3)}_{64\to 128}(\mathbf{Z}_3, \mathbf{V}(t))$,
$\mathbf{Z}_3 = \text{US-Block}^{(3)}_{128\to 128}(\mathbf{Z}_3, \mathbf{Z}_3^{skip}, \mathbf{V}(t))$,
$\mathbf{Z}_2 = \text{US-Block}^{(2)}_{256\to 64}(\mathbf{Z}_3, \mathbf{Z}_2^{skip}, \mathbf{V}(t))$,
$\mathbf{Z}_1 = \text{US-Block}^{(1)}_{128\to 32}(\mathbf{Z}_2, \mathbf{Z}_1^{skip}, \mathbf{V}(t))$,
$\mathbf{Z}_{out} = \text{D-Block}^{o}_{64\to 1}(\mathbf{Z}_1)/\beta(t)$.
Output: $\mathbf{Z}_{out} \in \mathbb{R}^{B\times 1\times H\times W}$.

The DS-Block$^{(i)}_{C_{in}\to C_{out}}$ ($i = 1, 2, 3$, $C_{out} = 32, 64, 128$, respectively) are constructed according to Architecture 7. The

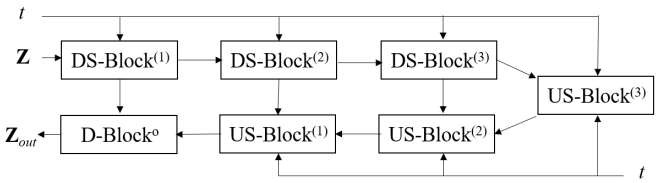

Fig. 11. Diagram of the score model's architecture.

US-Block$^{(i)}_{C_{in}\to C_{out}}$ ($i = 1, 2, 3$, $C_{out} = 32, 64, 128$, respectively) are constructed according to Architecture 9. Finally, the output decoding block D-Block$^{o}_{64\to 1}$ generates 1-channel output.

Architecture 7 DS-Block$^{(i)}_{C_{in}\to C_{out}}$

Input: $\mathbf{Z}_{in}^{block} \in \mathbb{R}^{B\times C_{in}\times H\times W}$, $\mathbf{V}(t) \in \mathbb{R}^{B\times 512}$.
$\mathbf{Z}_1 = \text{SiLU}(\text{GroupNorm}_{C_{out}/2}(\text{Conv2D}_{C_{out}}(\mathbf{Z}_{in}^{block})))$,
$\mathbf{Z}_1 = \text{SiLU}(\text{GroupNorm}_{C_{out}/2}(\text{Conv2D}_{C_{out}}(\mathbf{Z}_1)))$,
$\mathbf{Z}_1 = \mathbf{Z}_1 + \text{Dense}_{512\to C_{out}}(\mathbf{V}(t))$,
$\mathbf{Z}_1 = \text{SiLU}(\text{GroupNorm}_{C_{out}/2}(\text{Conv2D}_{C_{out}}(\mathbf{Z}_1)))$,
$\mathbf{Z}_{out}^{block,skip} = \text{SiLU}(\text{GroupNorm}_{C_{out}/2}(\text{Conv2D}_{C_{out}}(\mathbf{Z}_1)))$,
$\mathbf{Z}_{out}^{block} = \text{SiLU}((\text{Conv2D}_{C_{out},/2}(\mathbf{Z}_{out}^{block})))$.
Output: $\mathbf{Z}_{out}^{block,skip} \in \mathbb{R}^{B\times C_{out}\times H\times W}$, $\mathbf{Z}_{out}^{block} \in \mathbb{R}^{B\times C_{out}\times H/2\times W/2}$.

Architecture 8 US-Block$^{(i)}_{C_{in}\to C_{out}}$

Input: $\mathbf{Z}_{in}^{block} \in \mathbb{R}^{B\times C_{in}\times H\times W}$, $\mathbf{X}_{in}^{block,skip} \in \mathbb{R}^{B\times C_{in}\times 2H\times 2W}$, $\mathbf{V}(t) \in \mathbb{R}^{B\times 512}$.
$\mathbf{Z}_1 = \text{SiLU}(\text{GroupNorm}_{C_{out}}(\text{Conv2D}_{2C_{out}}(\mathbf{Z}_{in}^{block})))$,
$\mathbf{Z}_1 = \text{SiLU}(\text{GroupNorm}_{C_{out}}(\text{Conv2D}_{2C_{out}}(\mathbf{Z}_1)))$,
$\mathbf{Z}_1 = \mathbf{Z}_1 + \text{Dense}_{512\to 2C_{out}}(\mathbf{V}(t))$,
$\mathbf{Z}_1 = \text{SiLU}(\text{GroupNorm}_{C_{out}}(\text{Conv2D}_{2C_{out}}(\mathbf{Z}_1)))$,
$\mathbf{Z}_1 = \text{SiLU}(\text{GroupNorm}_{C_{out}}(\text{Conv2D}_{2C_{out}}(\mathbf{Z}_1)))$,
$\mathbf{Z}_1 = \text{SiLU}((\text{Conv2D}_{C_{out},\times 2}(\mathbf{Z}_1)))$,
$\mathbf{Z}_{out}^{block} = \text{concat}[\mathbf{Z}_1, \mathbf{Z}_{in}^{block,skip}]$.
Output: $\mathbf{Z}_{out}^{block} \in \mathbb{R}^{B\times 2C_{out}\times 2H\times 2W}$.

Architecture 9 D-Block$^{o}_{64\to 1}$

Input: $\mathbf{Z}_{in}^{block} \in \mathbb{R}^{B\times 1\times H\times W}$.
$\mathbf{Z}_1 = \text{SiLU}((\text{Conv2D}_{64}(\mathbf{Z}_{in}^{block})))$,
$\mathbf{Z}_{out}^{block} = \text{Conv2D}_1(\text{SiLU}((\text{Conv2D}_{64}(\mathbf{Z}_1))))$.
Output: $\mathbf{Z}_{out}^{block} \in \mathbb{R}^{B\times 1\times H\times W}$.

3) Neural network for baselines: The architecture is used for DIS and BPS. Denote the input tensor as $\mathbf{X} \in \mathbb{R}^{B\times C\times H\times W}$ and the output tensor as $\mathbf{X}_{out} \in \mathbb{R}^{B\times U\times H\times W}$. The network is cascaded by encoding blocks E-Block$^{(i)}_{C_{in}\to C_{out}}$ and decoding blocks D-Block$^{(i)}_{C_{in}\to C_{out}}$, whose operations are the same as those in Architecture 2 and 5, respectively. The whole architecture is in Architecture 10.

4) Score model for P-DPnP: The architecture is modified from Architecture 6, considering that the dimensions of input and output channels are different in the pixel

Architecture 10 Autoencoder for baselines

Input: $\mathbf{X} \in \mathbb{R}^{B\times C\times H\times W}$.
$\mathbf{X}_1 = \text{E-Block}^{(2)}_{16\to 32}(\text{E-Block}^{(1)}_{C\to 16}(\mathbf{X}))$,
$\mathbf{X}_1 = \text{E-Block}^{(4)}_{64\to 128}(\text{E-Block}^{(3)}_{32\to 64}(\mathbf{X}_1))$,
$\mathbf{X}_1 = \text{D-Block}^{(3)}_{128\to 64}(\text{D-Block}^{(4)}_{128\to 128}(\mathbf{X}_1))$,
$\mathbf{X}_1 = \text{D-Block}^{(1)}_{32\to 16}(\text{D-Block}^{(2)}_{64\to 32}(\mathbf{X}_1))$,
$\mathbf{X}_{out} = \text{D-Block}^{(0)}_{16\to U}(\mathbf{X}_1)$.
Output: $\mathbf{X}_{out} \in \mathbb{R}^{B\times U\times H\times W}$.

and latent diffusion. Denote the input and output tensor as $\mathbf{X}, \mathbf{X}_{out} \in \mathbb{R}^{B\times C\times H\times W}$. The blocks $\text{DS-Block}^{(i)}_{C_{in}\to C_{out}}$ and $\text{US-Block}^{(i)}_{C_{in}\to C_{out}}$ ($i = 1, 2, 3, 4$, $C_{out} = 32, 64, 128, 256$, respectively) are cascaded as in Architecture 11. Operations inside $\text{DS-Block}^{(i)}_{C_{in}\to C_{out}}$ and $\text{US-Block}^{(i)}_{C_{in}\to C_{out}}$ can be found in Architecture 7 and 9, respectively.

Architecture 11 Score model for P-DPnP

Input: $\mathbf{X} \in \mathbb{R}^{B\times C\times H\times W}$, $t \in \mathbb{R}^B$.
$\mathbf{V}_p(t) = \text{concat}[\sin(2\pi t\mathbf{W}), \cos(2\pi t\mathbf{W})]$,
$\mathbf{V}(t) = \text{SiLU}(\text{Dense}_{512\to 512}(\mathbf{V}_p(t)))$,
$\mathbf{X}_1^{skip}, \mathbf{X}_1 = \text{DS-Block}^{(1)}_{C\to 32}(\mathbf{X}, \mathbf{V}(t))$,
$\mathbf{X}_2^{skip}, \mathbf{X}_2 = \text{DS-Block}^{(2)}_{32\to 64}(\mathbf{X}_2, \mathbf{V}(t))$,
$\mathbf{X}_3^{skip}, \mathbf{X}_3 = \text{DS-Block}^{(3)}_{64\to 128}(\mathbf{X}_3, \mathbf{V}(t))$,
$\mathbf{X}_4^{skip}, \mathbf{X}_4 = \text{DS-Block}^{(3)}_{128\to 256}(\mathbf{X}_3, \mathbf{V}(t))$,
$\mathbf{X}_4 = \text{US-Block}^{(4)}_{256\to 256}(\mathbf{X}_4, \mathbf{X}_4^{skip}, \mathbf{V}(t))$,
$\mathbf{X}_3 = \text{US-Block}^{(3)}_{512\to 128}(\mathbf{X}_4, \mathbf{X}_3^{skip}, \mathbf{V}(t))$,
$\mathbf{X}_2 = \text{US-Block}^{(2)}_{256\to 64}(\mathbf{X}_3, \mathbf{X}_2^{skip}, \mathbf{V}(t))$,
$\mathbf{X}_1 = \text{US-Block}^{(1)}_{128\to 32}(\mathbf{X}_2, \mathbf{X}_1^{skip}, \mathbf{V}(t))$,
$\mathbf{X}_{out} = \text{D-Block}^{o}_{64\to C}(\mathbf{X}_1)/\beta(t)$.
Output: $\mathbf{Z}_{out} \in \mathbb{R}^{B\times C\times H\times W}$.

B. Implementation details

Our codes are written in PyTorch. All gradients are computed using the built-in automatic differentiation. In the following, we use $\boldsymbol{\chi} \in \mathbb{C}^{N_x\times N_y}$ as a shorthand notation for the reshaped contrast vector $\text{diag}(\boldsymbol{\chi}(\boldsymbol{\epsilon_r}, \boldsymbol{\sigma_e}))$.

1) Occam inversion: Occam inversion generates smooth boundary targets by penalizing the ℓ_2 norm of spatial gradients. In our loss function, the unknown parameter is $\boldsymbol{\chi}$, and the data fidelity term is normalized by the ℓ_2 norm of measurements. For the MNIST and Fashion-MNIST experiments, we set the smoothness regularization coefficient to 0.3 and performed 400 iterations of L-BFGS [78] (learning rate = 0.05). In the ATLAS example, we apply the multiplicative regularization [70] and run 400 iterations of ADAM [79] (learning rate = 0.01). The regularization coefficient in each iteration is multiplied by measurement RMSE in each iteration, with an initial coefficient being 20.

2) TV-ADMM: TV penalizes the ℓ_1 norm of spatial gradients. In our loss function, the unknown parameter is $\boldsymbol{\chi}$, with the data fidelity term being normalized by the ℓ_2 norm of measurements. In the MNIST and Fashion-MNIST examples, we set the TV coefficient to 5 and perform 20 ADMM iterations. Within each ADMM iteration, the quadratic loss function containing data fidelity is minimized by 20 steps of L-BFGS (learning rate = 0.1). In the ATLAS example, the TV coefficient is set to 1.5, and we also run 20 iterations of ADMM iterations. Within each ADMM iteration, the quadratic loss associated with data fidelity is minimized by 20 steps of ADAM (learning rate = 0.005).

3) DIS: The neural network architecture is presented in Appendix A3. It takes modified measurements as input $\mathbf{X} \in \mathbb{R}^{B\times 2N_f\times N_x\times N_y}$ and electrical property maps as output $\mathbf{X}_{out} \in \mathbb{R}^{B\times U\times N_x\times N_y}$, respectively. In training, the measurements are simulated from permittivity maps using (6), the former being inputs and the latter being labels. The training loss is the mean squared error between labels and predictions. The dataset for training is preprocessed as follows. We first reshape the measurements to $B \times 2N_f \times N_T \times N_R$, where the channels are concatenated by the real and imaginary parts at different frequency points. Then, bilinear interpolation is performed on the reshaped measurement tensor to produce the input tensor of size $B \times 2N_f \times N_x \times N_y$. The output is $\boldsymbol{\chi}$. For the MNIST and Fashion-MNIST experiments, $N_f = 2$, $N_x = N_y = 64$, and the output has one channel $U = 1$ representing the real part of $\boldsymbol{\chi}$. The model is trained for 20 epochs using a batch size of 256 and a exponentially decaying learning rate with the initial value of 0.0005 and a multiplicative factor of 0.99. In the ATLAS example, $N_f = 2$, $N_x = N_y = 96$, and the output has two channels $U = 2$ that represent the real and imaginary part of $\boldsymbol{\chi}$. The model is trained for 70 epochs using a batch size of 256 and a exponentially decaying learning rate with the initial value of 0.0005 and a multiplicative factor of 0.99. The ATLAS example takes more epochs because the inverse mapping from measurements to properties is more difficult to learn due to the complicated multiple scattering effects.

4) BPS: We apply the multiple-frequency back-projection [36] – a classical linear imaging method that approximates the total field with the incident field – to generate input data from the measurement. The neural network architecture is presented in Appendix A3. It takes back-projection results as input, $\mathbf{X} \in \mathbb{R}^{B\times C\times N_x\times N_y}$, and electrical property maps as output $\mathbf{X}_{out} \in \mathbb{R}^{B\times U\times N_x\times N_y}$, respectively. The training loss is the mean squared error between the predicted and true electrical property maps. For the MNIST and Fashion-MNIST experiments, we set $N_x = N_y = 64$ and $C = U = 1$ (for real-valued property maps). The model is trained for 100 epochs using a batch size of 256 and an exponentially decaying learning rate (initial value 0.0005, and multiplicative factor 0.99). For the ATLAS experiment, $N_x = N_y = 96$ and $C = U = 2$ (for complex-valued property maps). Since the dataset is larger than MNIST and Fashion-MNIST, we train for 70 epochs. Other settings are the same as those in the MNIST and Fashion-MNIST experiments.

5) GMR: It solves for the latent representation of the electrical properties via deterministic optimization. This process consists of two main steps: first, training an encoding model to learn the latent representation; and second, optimizing the latent variables to minimize the data fidelity term.

We use the neural network architecture detailed in Appendix A1 to reparameterize the property maps. The input and output

has the same shape, $\mathbf{X}, \mathbf{X}_{out} \in \mathbb{R}^{B\times C\times N_x\times N_y}$. Here, the channels represent the real and imaginary components of the complex maps, with each channel normalized independently between -1 and 1. The training loss is the ELBO that includes a small KL divergence term [75], [80]. All models are trained for 400 epochs using a batch size of 256 and an exponentially decaying learning rate with a decaying multiplicative factor 0.99. For the MNIST and Fashion-MNIST experiments, $N_x = N_y = 64$ and $C = 1$ (for real-valued property maps), with a KL divergence coefficient of 0.02, the initial learning rate of 0.0008. For the ATLAS experiment, $N_x = N_y = 96$ and $C = 2$ (for complex-valued property maps), with a KL divergence coefficient of 0.03, and the initial learning rate of 0.0005.

After the autoencoder is trained, we integrate the decoder into the optimization [41] and solves for the latent variables using ADAM. The loss function consists of a data fidelity term normalized by the ℓ_2 norm of measurements and a regularization term that penalizes the ℓ_2 norm of the latent variables. The maximum step number of ADAM is set to 500. For the MNIST and Fashion-MNIST experiments, the learning rate is fixed at 0.08, and the initial latent variables are drawn from a standard Gaussian distribution. The regularization coefficient is 0.005. For the ATLAS experiment, we adopt a cosine annealing learning rate, with a starting value of 0.8, minimum learning rate of 0.1, and a full cycle iteration number 1000. The initial latent variables are set to the encoded latent variable of the average property map computed from the entire training dataset, and the regularization coefficient is 0.08.

6) P-DPnP: The neural network's architecture is detailed in Appendix A4. The input and output has the same shape, $\mathbf{X}, \mathbf{X}_{out} \in \mathbb{R}^{B\times C\times N_x\times N_y}$. The channels represent the real and imaginary components of the complex maps, with each channel normalized independently between -1 and 1. As suggested by [46], [54], we set $f(t) = 0$ and $g(t) = \sigma_d^t$ in (25). In this case, the variance of the transition distribution is

$$\beta^2(t) = \frac{1}{2\log\sigma_d}(\sigma_d^{2t} - 1). \tag{32}$$

We set $\sigma_d = 20$. All models for different experiments are trained for 400 epochs using a batch size of 256 and an exponentially decaying learning rate (initial value 0.00008, and multiplicative factor 0.99). For the MNIST and Fashion-MNIST experiments, $C = 1$, and $N_x = N_y = 64$. For the ATLAS experiment, $C = 2$, and $N_x = N_y = 96$.

Recall that we define the following parameters: the number of the two-step loops N_k, the number of iterations N_τ in the likelihood step, the number of iterations N_t in the prior step, and the number of MMSE samples M. For all experiments, we set $N_k = 20$, $N_\tau = 120$, $N_t = 500$, and $M = 5$ (due to high computational cost). To balance the scales between the gradient and noise terms, we introduce a weighting coefficient α_n to the gradient term in (24). The update equation becomes

$$\begin{aligned}\boldsymbol{\chi}[n+1] = \alpha_n\eta_k^2(1-r)\nabla_{\boldsymbol{\chi}[n]}\mathcal{L}(\boldsymbol{\chi}[n], \mathbf{d}_{obs}) + r\boldsymbol{\chi}[n] \\ + (1-r)\hat{\boldsymbol{\chi}}_k + \eta_k\sqrt{(1-r^2)}\mathbf{n},\end{aligned} \tag{33}$$

with

$$\alpha_n = \alpha_0\frac{\|\boldsymbol{\chi}[n]\|_2^2}{\eta_k^2(\|\nabla_{\boldsymbol{\chi}[n]}\mathcal{L}(\boldsymbol{\chi}[n], \mathbf{d}_{obs})\|_2^2 + 0.001)}, \tag{34}$$

The initial image $\boldsymbol{\chi}[0]$ is drawn from a Gaussian distribution with a mean equal to the homogeneous background. We set $\gamma = 0.015\eta_k^2$ in $r = e^{-\gamma/\eta_k^2}$. For the prior step, the initial time is computed according to (32) and $t_{N_t-1} = \beta^{-1}(\eta_k)$:

$$t_{N_t-1} = \left.\frac{\log\left(1 + 2\,\eta_k\,\log\sigma_d\right)}{2\,\log\sigma_d}\right|_{\sigma_d=20}. \tag{35}$$

In addition, we set $\varepsilon_t = 0.001$ when computing the time interval $\delta = \frac{t_{N_t}-\varepsilon_t}{N_t}$.

For the MNIST and Fashion-MNIST experiments, we set $\alpha_0 = 0.3$, $\eta_k = 0.4$ for $k = 0, ..., 4$; and for $k = 5, ..., 19$, η_k decays uniformly on a logarithmic scale from 0.4 to 0.1. In addition, since the transmitters and receivers are outside the imaging domain D, we do not apply masks in the likelihood step. For the ATLAS experiment, we set $\alpha_0 = 1$, and η_k decays uniformly on a logarithmic scale from 0.2 to 0.02 for $k = 0, ..., 19$. In each likelihood step (36), the gradients in regions near the sensors are set to zero by applying a circular mask with a radius of 0.12m.

7) L-DPnP: The neural network architecture of the autoencoder and score model is presented in Appendix A1 and A2, respectively. The setup for training the autoencoder is detailed in Appendix B5. The input and output of the score model in latent space is $\mathbf{X}, \mathbf{X}_{out} \in \mathbb{R}^{B\times 1\times N_u\times N_v}$, respectively. The settings for training the score model is the same as those in Appendix B6. For the MNIST and Fashion-MNIST experiments, $N_u = N_v = 16$. For the ATLAS experiment, $N_u = N_v = 24$.

We set $N_k = 20$, $N_t = 500$, $N_\tau = 120$, and $M = 5$. The update equation for the likelihood step is

$$\begin{aligned}\mathbf{z}[n+1] = \alpha_n\eta_k^2(1-r)\nabla_{\mathbf{z}[n]}\mathcal{L}(\mathbf{z}[n], \mathbf{d}_{obs}) + r\mathbf{x}[n] \\ + (1-r)\hat{\mathbf{z}}_k + \eta_k\sqrt{(1-r^2)}\mathbf{n}.\end{aligned} \tag{36}$$

with

$$\alpha_n = \alpha_0\frac{\|\mathbf{z}[n]\|_2^2}{\eta_k^2(\|\nabla_{\mathbf{z}[n]}\mathcal{L}(\mathbf{z}[n], \mathbf{d}_{obs})\|_2^2 + 0.001)}, \tag{37}$$

where $\mathbf{z}$ denotes the latent domain, and the initial $\mathbf{z}[0]$ is drawn from the Gaussian distribution. For all experiments, we set $\alpha_0 = 1$.

For the MNIST and Fashion-MNIST experiments, $\eta_k = 0.4$ for $k = 0, ..., 4$; and for $k = 5, ..., 19$, η_k decays uniformly on a logarithmic scale from 0.4 to 0.1. For the ATLAS experiment, η_k decays uniformly on a logarithmic scale from 1 to 0.03 for $k = 0, ..., 19$. Other settings are kept the same with those in Appendix B6.

REFERENCES

[1] M. Pastorino, *Microwave imaging*. John Wiley & Sons, 2010.

[2] M. Li and M. Salucci, *Applications of Deep Learning in Electromagnetics: Teaching Maxwell's Equations to Machines*. IET, 2023.

[3] K. Tan, X. Chen, S. Wu, and G. Fang, "Efficient frequency scaling algorithm for short-range 3-D holographic imaging based on a scanning MIMO array," *IEEE Transactions on Microwave Theory and Techniques*, vol. 68, no. 9, pp. 3885–3897, 2020.

[4] X. Zhang and X. Ye, "Solving strongly nonlinear inverse scattering problems for mixture of dielectric and PEC with novel contraction T-matrix equation," *IEEE Transactions on Microwave Theory and Techniques*, 2025.

[5] M. Salucci, A. Gelmini, J. Vrba, I. Merunka, G. Oliveri, and P. Rocca, "Instantaneous brain stroke classification and localization from real scattering data," *Microwave and Optical Technology Letters*, vol. 61, no. 3, pp. 805–808, 2019.

[6] M. T. Bevacqua, S. Di Meo, L. Crocco, T. Isernia, and M. Pasian, "Millimeter-waves breast cancer imaging via inverse scattering techniques," *IEEE Journal of Electromagnetics, RF and Microwaves in Medicine and Biology*, 2021.

[7] F. Zardi, L. Tosi, M. Salucci, and A. Massa, "A physics-driven AI approach for microwave imaging of breast tumors," *IEEE Transactions on Antennas and Propagation*, 2025.

[8] A. Massa, M. Pastorino, A. Rosani, and M. Benedetti, "A microwave imaging method for NDE/NDT based on the SMW technique for the electromagnetic field prediction," *IEEE Transactions on Instrumentation and Measurement*, vol. 55, no. 1, pp. 240–247, 2006.

[9] R. Zoughi, *Microwave non-destructive testing and evaluation principles*, vol. 4. Springer Science & Business Media, 2000.

[10] A. Hamidipour, T. Henriksson, M. Hopfer, R. Planas, and S. Semenov, "Electromagnetic tomography for brain imaging and stroke diagnostics: Progress towards clinical application," in *Emerging electromagnetic technologies for brain diseases diagnostics, monitoring and therapy*, pp. 59–86, Springer, 2018.

[11] H.-Y. Wey, V. R. Desai, and T. Q. Duong, "A review of current imaging methods used in stroke research," *Neurological research*, vol. 35, no. 10, pp. 1092–1102, 2013.

[12] X. Chen, *Computational methods for electromagnetic inverse scattering*. John Wiley & Sons, 2018.

[13] O. M. Bucci and G. Franceschetti, "On the degrees of freedom of scattered fields," *IEEE transactions on Antennas and Propagation*, vol. 37, no. 7, pp. 918–926, 1989.

[14] A. Abubakar, T. M. Habashy, G. Pan, and M.-K. Li, "Application of the multiplicative regularized Gauss–Newton algorithm for three-dimensional microwave imaging," *IEEE Transactions on Antennas and Propagation*, vol. 60, no. 5, pp. 2431–2441, 2012.

[15] A. Abubakar, P. van den Berg, and J. Mallorqui, "Imaging of biomedical data using a multiplicative regularized contrast source inversion method," *IEEE Transactions on Microwave Theory and Techniques*, vol. 50, no. 7, pp. 1761–1771, 2002.

[16] J. Liu, H. Zhou, L. Chen, and T. Ouyang, "Alternating direction method of multiplier for solving electromagnetic inverse scattering problems," *International Journal of Microwave and Wireless Technologies*, vol. 12, no. 8, pp. 790–796, 2020.

[17] Q. Shen, J. Chen, X. Wu, Y. Huang, Z. Han, Q. Shen, J. Chen, X. Wu, Y. Huang, and Z. Han, "Bayesian inversion and sampling methods," *Statistical Inversion of Electromagnetic Logging Data*, pp. 17–29, 2021.

[18] X. Zhang, A. Lomas, M. Zhou, Y. Zheng, and A. Curtis, "3-D bayesian variational full waveform inversion," *Geophysical Journal International*, vol. 234, no. 1, pp. 546–561, 2023.

[19] J. Zhou, D. Husmeier, H. Gao, C. Yin, C. Qiu, X. Jing, Y. Qi, and W. Liu, "Bayesian inversion of frequency-domain airborne em data with spatial correlation prior information," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 62, pp. 1–16, 2024.

[20] M. Li, A. Abubakar, and T. M. Habashy, "A three-dimensional model-based inversion algorithm using radial basis functions for microwave data," *IEEE Transactions on Antennas and Propagation*, vol. 60, no. 7, pp. 3361–3372, 2012.

[21] L. Guo and A. M. Abbosh, "Microwave imaging of nonsparse domains using Born iterative method with wavelet transform and block sparse Bayesian learning," *IEEE Transactions on Antennas and Propagation*, vol. 63, no. 11, pp. 4877–4888, 2015.

[22] X. Chen, Z. Wei, L. Maokun, P. Rocca, *et al.*, "A review of deep learning approaches for inverse scattering problems (invited review)," *Electromagnetic Waves*, vol. 167, pp. 67–81, 2020.

[23] L. Li, L. G. Wang, F. L. Teixeira, C. Liu, A. Nehorai, and T. J. Cui, "DeepNIS: Deep neural network for nonlinear electromagnetic inverse scattering," *IEEE Transactions on Antennas and Propagation*, vol. 67, no. 3, pp. 1819–1825, 2018.

[24] M. Li, R. Guo, K. Zhang, Z. Lin, F. Yang, S. Xu, X. Chen, A. Massa, and A. Abubakar, "Machine learning in electromagnetics with applications to biomedical imaging: A review," *IEEE Antennas and Propagation Magazine*, vol. 63, no. 3, pp. 39–51, 2021.

[25] J. C. Ye, Y. C. Eldar, and M. Unser, *Deep Learning for Biomedical Image Reconstruction*. Cambridge University Press, 2023.

[26] R. Guo, T. Huang, M. Li, H. Zhang, and Y. C. Eldar, "Physics-embedded machine learning for electromagnetic data imaging: Examining three types of data-driven imaging methods," *IEEE Signal Processing Magazine*, vol. 40, no. 2, pp. 18–31, 2023.

[27] J.-M. Geffrin, P. Sabouroux, and C. Eyraud, "Free space experimental scattering database continuation: experimental set-up and measurement precision," *Inverse Problems*, vol. 21, no. 6, p. S117, 2005.

[28] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Processing Magazine*, vol. 38, no. 2, pp. 18–44, 2021.

[29] N. Shlezinger, Y. C. Eldar, and S. P. Boyd, "Model-based deep learning: On the intersection of deep learning and optimization," *IEEE Access*, vol. 10, pp. 115384–115398, 2022.

[30] Z. Wei and X. Chen, "Deep-learning schemes for full-wave nonlinear inverse scattering problems," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 4, pp. 1849–1860, 2018.

[31] X. Ye, Y. Bai, R. Song, K. Xu, and J. An, "An inhomogeneous background imaging method based on generative adversarial network," *IEEE Transactions on Microwave Theory and Techniques*, vol. 68, no. 11, pp. 4684–4693, 2020.

[32] N. Cohen, Y. Kvich, R. Guo, and Y. C. Eldar, "Deep unfolding of full waveform inversion for quantitative ultrasound imaging," in *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, IEEE, 2025.

[33] Y. Jin, Q. Shen, X. Wu, J. Chen, and Y. Huang, "A physics-driven deep-learning network for solving nonlinear inverse problems," *Petrophysics*, vol. 61, no. 01, pp. 86–98, 2020.

[34] L. Bar and N. Sochen, "Strong solutions for PDE-based tomography by unsupervised learning," *SIAM Journal on Imaging Sciences*, vol. 14, no. 1, pp. 128–155, 2021.

[35] T. Sharon and Y. C. Eldar, "Real-time model-based quantitative ultrasound and radar," *IEEE Transactions on Computational Imaging*, vol. 10, pp. 1175–1190, 2024.

[36] R. Guo, Z. Lin, T. Shan, X. Song, M. Li, F. Yang, S. Xu, and A. Abubakar, "Physics embedded deep neural network for solving full-wave inverse scattering problems," *IEEE Transactions on Antennas and Propagation*, vol. 70, no. 8, pp. 6148–6159, 2021.

[37] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-play priors for model based reconstruction," in *2013 IEEE Global Conference on Signal and Information Processing*, pp. 945–948, IEEE, 2013.

[38] A. Bora, A. Jalal, E. Price, and A. G. Dimakis, "Compressed sensing using generative models," in *International Conference on Machine Learning*, pp. 537–546, PMLR, 2017.

[39] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9446–9454, 2018.

[40] D. Liu, J. Wang, Q. Shan, D. Smyl, J. Deng, and J. Du, "DeepEIT: Deep image prior enabled electrical impedance tomography," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 8, pp. 9627–9638, 2023.

[41] R. Guo, Z. Lin, J. Xin, M. Li, F. Yang, S. Xu, and A. Abubakar, "Three dimensional microwave data inversion in feature space for stroke imaging," *IEEE Transactions on Medical Imaging*, vol. 43, no. 4, pp. 1365–1376, 2024.

[42] A. Khorashadizadeh, V. Khorashadizadeh, S. Eskandari, G. A. E. Vandenbosch, and I. Dokmanić, "Deep injective prior for inverse scattering," *IEEE Transactions on Antennas and Propagation*, vol. 71, no. 11, pp. 8894–8906, 2023.

[43] Z. Kadkhodaie and E. Simoncelli, "Stochastic solutions for linear inverse problems using the prior implicit in a denoiser," *Advances in Neural Information Processing Systems*, vol. 34, pp. 13242–13254, 2021.

[44] H. Chung, J. Kim, M. T. Mccann, M. L. Klasky, and J. C. Ye, "Diffusion posterior sampling for general noisy inverse problems," in *The Eleventh International Conference on Learning Representations*, 2023.

[45] B. Kawar, M. Elad, S. Ermon, and J. Song, "Denoising diffusion restoration models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 23593–23606, 2022.

[46] Y. Song, L. Shen, L. Xing, and S. Ermon, "Solving inverse problems in medical imaging with score-based generative models," in *International Conference on Learning Representations*, 2022.

[47] C. A. Bouman and G. T. Buzzard, "Generative plug and play: Posterior sampling for inverse problems," in *2023 59th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 1–7, 2023.

[48] X. Xu and Y. Chi, "Provably robust score-based diffusion posterior sampling for plug-and-play image reconstruction," *Advances in Neural Information Processing Systems*, 2024.

[49] Z. Wu, Y. Sun, Y. Chen, B. Zhang, Y. Yue, and K. Bouman, "Principled probabilistic imaging using diffusion models as plug-and-play priors," *Advances in Neural Information Processing Systems*, vol. 37, pp. 118389–118427, 2024.

[50] R. Guo, Y. Kvich, Y. Zhang, T. Huang, M. Li, and C. Y. Eldar, "Posterior sampling with latent diffusion for microwave brain imaging," in *47th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, p. to be appeared, 2025.

[51] J.-M. Jin, *Theory and computation of electromagnetic fields*. John Wiley & Sons, 2015.

[52] C. P. Robert, G. Casella, C. P. Robert, and G. Casella, "The metropolis—hastings algorithm," *Monte Carlo statistical methods*, pp. 267–320, 2004.

[53] G. O. Roberts and J. S. Rosenthal, "Optimal scaling of discrete approximations to Langevin diffusions," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 60, no. 1, pp. 255–268, 1998.

[54] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *International Conference on Learning Representations*, 2021.

[55] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in Neural Information Processing Systems*, vol. 33, pp. 6840–6851, 2020.

[56] C. He, Y. Shen, C. Fang, F. Xiao, L. Tang, Y. Zhang, W. Zuo, Z. Guo, and X. Li, "Diffusion models in low-level vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[57] B. B. Moser, A. S. Shanbhag, F. Raue, S. Frolov, S. Palacio, and A. Dengel, "Diffusion models, image super-resolution, and everything: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[58] S. Särkkä and A. Solin, *Applied stochastic differential equations*, vol. 10. Cambridge University Press, 2019.

[59] P. Vincent, "A connection between score matching and denoising autoencoders," *Neural Computation*, vol. 23, no. 7, pp. 1661–1674, 2011.

[60] P. E. Kloeden, E. Platen, P. E. Kloeden, and E. Platen, *Stochastic differential equations*. Springer, 1992.

[61] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[62] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[63] S.-L. Liew, J. M. Anglin, N. W. Banks, M. Sondag, K. L. Ito, H. Kim, J. Chan, J. Ito, C. Jung, N. Khoshab, *et al.*, "A large, open source dataset of stroke anatomical brain images and manual lesion segmentations," *Scientific Data*, vol. 5, no. 1, pp. 1–11, 2018.

[64] S. C. Constable, R. L. Parker, and C. G. Constable, "Occam's inversion: A practical algorithm for generating smooth models from electromagnetic sounding data," *Geophysics*, vol. 52, no. 3, pp. 289–300, 1987.

[65] H. S. Aghamiry, A. Gholami, and S. Operto, "ADMM-based multiparameter wavefield reconstruction inversion in VTI acoustic media with TV regularization," *Geophysical Journal International*, vol. 219, no. 2, pp. 1316–1333, 2019.

[66] L. Li, L. G. Wang, F. L. Teixeira, C. Liu, A. Nehorai, and T. J. Cui, "DeepNIS: Deep neural network for nonlinear electromagnetic inverse scattering," *IEEE Transactions on Antennas and Propagation*, vol. 67, no. 3, pp. 1819–1825, 2019.

[67] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

[68] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," *arXiv preprint arXiv:2010.02502*, 2020.

[69] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu, "DPM-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps," *Advances in Neural Information Processing Systems*, vol. 35, pp. 5775–5787, 2022.

[70] K. Zhang, M. Li, F. Yang, S. Xu, and A. Abubakar, "Three-dimensional electrical impedance tomography with multiplicative regularization," *IEEE Transactions on Biomedical Engineering*, vol. 66, no. 9, pp. 2470–2480, 2019.

[71] Y.-B. Li, J. Wang, C. Su, W.-J. Lin, X.-M. Wang, and Y. Luo, "Quantitative ultrasound brain imaging with multiscale deconvolutional waveform inversion," *Chinese Physics B*, vol. 32, no. 1, p. 014303, 2023.

[72] J. Ma, Y. Deng, X. Li, R. Guo, H. Zhou, and M. Li, "Recent advances in machine learning-enhanced joint inversion of seismic and electromagnetic data," *Surveys in Geophysics*, pp. 1–29, 2024.

[73] R. Guo, H. Zhou, X. Wei, Z. Lin, M. Li, Y. C. Eldar, F. Yang, S. Xu, and A. Abubakar, "Deep joint inversion of multiple geophysical data with U-net reparameterization," *Geophysics*, vol. 90, no. 3, pp. WA61–WA75, 2024.

[74] X. Song, M. Li, F. Yang, S. Xu, and A. Abubakar, "Three-dimensional joint inversion of EM and acoustic data based on contrast source inversion," *IEEE Journal on Multiscale and Multiphysics Computational Techniques*, vol. 5, pp. 28–36, 2020.

[75] D. P. Kingma and M. Welling, "Auto-Encoding variational Bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[76] A. Razavi, A. Van den Oord, and O. Vinyals, "Generating diverse high-fidelity images with VQ-VAE-2," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[77] Y. Pu, Z. Gan, R. Henao, X. Yuan, C. Li, A. Stevens, and L. Carin, "Variational autoencoder for deep learning of images, labels and captions," *Advances in Neural Information Processing Systems*, vol. 29, pp. 2352–2360, 2016.

[78] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1, pp. 503–528, 1989.

[79] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[80] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.