Nanoscale

Inverse deep learning methods and benchmarks for artificial electromagnetic material design

Journal:	*Nanoscale*
Manuscript ID	NR-ART-12-2021-008346.R1
Article Type:	Paper
Date Submitted by the Author:	11-Feb-2022
Complete List of Authors:	Ren, Simiao; Duke University, Electrical and Computer Engineering Mehendra, Ashwin; Duke University, Electrical and Computer Engineering Khatib, Omar; Duke University, Electrical and Computer Engineering Deng, Yang; Duke University, Electrical and Computer Engineering Padilla, Willie; Duke University, Department of Electrical and Computer Engineering Malof, Jordan; Duke University, Electrical and Computer Engineering

Nanoscale

ARTICLE TYPE

Cite this: DOI: 00.0000/xxxxxxxxxx

Inverse deep learning methods and benchmarks for artificial electromagnetic material design†

Simiao Ren [a‡], Ashwin Mahendra [a‡], Omar Khatib[a], Yang Deng [a], Willie J. Padilla,[a] and Jordan M. Malof [a]

Received Date
Accepted Date

DOI: 00.0000/xxxxxxxxxx

In this work we investigate the use of deep inverse models (DIMs) for designing artificial electromagnetic materials (AEMs) - such as metamaterials, photonic crystals, and plasmonics - to achieve some desired scattering properties (e.g., transmission or reflection spectrum). DIMs are deep neural networks (i.e., deep learning models) that are specially-designed to solve ill-posed inverse problems. There has recently been tremendous growth in the use of DIMs for solving AEM design problems however there has been little comparison of these approaches to examine their absolute and relative performance capabilities. In this work we compare eight state-of-the-art DIMs on three unique AEM design problems, including two models that are novel to the AEM community. Our results indicate that DIMs can rapidly produce accurate designs to achieve a custom desired scattering on all three problems. Although no single model always performs best, the Neural-Adjoint approach achieves the best overall performance across all problem settings. As a final contribution we show that not all AEM design problems are ill-posed, and in such cases a conventional deep neural network can perform better than DIMs. We recommend that a deep neural network is always employed as a simple baseline approach when studying or solving AEM design problems. We publish python code for our AEM simulators and our DIMs to enable easy replication of our results, and benchmarking of new DIMs by the AEM community.

1 Introduction

In this work we consider the problem of designing artificial electromagnetic materials (AEMs), such as metamaterials, photonic crystals, and plasmonics. The goal of AEM design is to find the geometric structure, material composition, or other features of an AEM - denoted g - that will produce a desired electromagnetic (EM) response (e.g., a specific transmission or absorption spectrum), denoted s here[1–3]. This is a widely-studied problem involving a rich body of research, and a variety of effective methods[4]. Here we focus on emerging methods involving deep inverse models (DIMs), which have recently been found highly effective for solving AEM design problems[3,5–8].

DIMs are data-driven methods, and therefore assume access to a dataset of design-scattering pairs, $D = \{(g_n, s_n)\}_{n=1}^{N}$, which are obtained by evaluating the so-called "forward model" of the AEM system[3] at specific values of g. The output of the forward model, denoted $s = f(g)$, is usually estimated via theoretical results or computational electromagnetic simulations (CEMS). DIMs then use D to infer, or learn, an *inverse* model, denoted $g = f^{-1}(s)$, that maps from a desired scattering directly to an AEM design that will produce the desired scattering. This process of learning f^{-1} is sometimes referred to as "training", and D is referred to as the training dataset. Learning the inverse model is essentially a regression problem where s comprises the independent variables, and g comprises the dependent variable that we wish to predict. Therefore conventional regression models, such as deep neural networks (DNNs), can be employed to infer f^{-1}. A substantial body of recent work has investigated DNNs using deep neural networks (DNNs) to approximate the forward model, f, yielding impressive results[3,7,9–11]. This success is often attributed to DNN's ability to approximate complex and highly non-linear functions. Despite this capability, and its associated success however, DNNs can produce poor results if the problem is *ill-posed*[3,12], as is often the case when approximating f^{-1}.

Ill-posed problems are those that violate one of the three following Hadamard conditions[13]: (i) existence; (ii) uniqueness; and (iii) smoothness. In principle, inverse AEM problems can violate any of the three Hadamard conditions, however, most recent attention has been given to violations of condition (ii), which is sometimes called "one-to-manyness", or non-uniqueness. We re-

‡ *These authors contributed equally*

[a] *Department of Electrical and Computer Engineering, Duke University, Box 90291, Durham, NC 27708, USA. E-mail: jordan.malof@duke.edu*

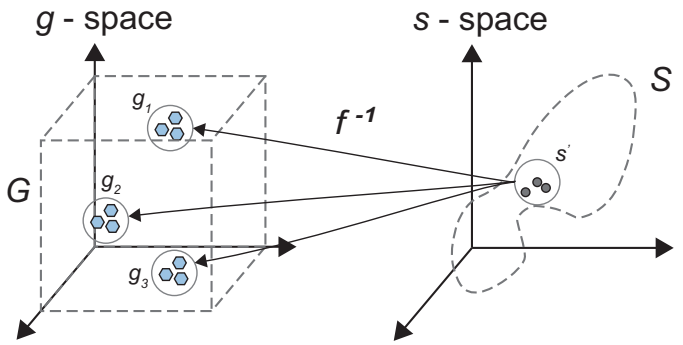

Fig. 1 Illustration of non-uniqueness. There are multiple unique designs (left) that yield a similar scattering (right). When trying to learn the inverse model (i.e., $g = f^{-1}(s)$), the input is s and the model attempts to predict a valid value of g. However, conventional regression models assume there is only one solution, and often fail when confronted with datasets where there are multiple valid solutions for some input values, as shown here.

fer the reader to other sources for a more thorough treatment of the other conditions[3]. In the context of AEM design, non-uniqueness arises when there exist multiple values of g that all yield similar AEM scattering (i.e., values of s), as illustrated in Fig. 1. This is problematic for conventional DNNs because they can only produce a single output for each input, making it impossible to simultaneously predict two g values for a given s. Furthermore, the training procedure for DNNs does not account for non-uniqueness. As a result, when DNNs are trained on datasets that exhibit non-uniqueness, they can learn to make highly inaccurate predictions.

In recent years a large number of specially-designed models have been proposed, or adapted from the machine learning community, in order to overcome non-uniqueness in inverse AEM problems[14–23]. We refer to these approaches as DIMs, and they can be taxonomized into the following broad categories based upon their modeling strategy: probabilistic (GAN-based[14,15],VAE-based[16,17]), deterministic (e.g., tandem[18,19]), and iterative (e.g., neural-adjoint[20,21,24], genetic algorithms[22,23]). Each of these these model classes has been employed successfully for solving AEM inverse problems. We will describe DIMs in greater detail in Sec. 3, as well as recent work employing them for inverse AEM problems.

1.1 Challenges with benchmarking and evaluation

Despite the success and rapid growth of DIMs in AEM research over the past several years, there has been little replication and benchmarking (e.g., comparisons of different DIMs on the same AEM problems), making it difficult to determine the extent to which real methodological progress is being made over time. This also makes it challenging for researchers and practitioners in the AEM community to choose methods that are best-suited to solve their problems. One fundamental reason for the absence of benchmarking may be the difficulty of reproducing the computational simulations from a previous study, which often requires substantial AEM expertise, and computation time. Furthermore, publications may provide an insufficient level of detail to accurately reproduce simulations.

One potential solution to this problem, which has become common in the machine learning community, is for authors to release their datasets and trained models. As noted in recent reviews[3,7], and quantified in a recent study[25], this is rarely done in the AEM literature. One notable exception to this trend is a recent study study that published a benchmark of three AEM problems for data-driven *forward* modeling (i.e., approximating $f(g)$). However, to our knowledge there has been no systematic comparison and analysis of DIMs for AEM problems.

Another challenge with the study of DIMs is the selection of an appropriate inverse problem. Few studies reported whether the problem under consideration truly exhibits non-uniqueness, or compared to a conventional DNN. These are important considerations since most DIMs are built upon the assumption that non-uniqueness is present in the data. If there is no non-uniqueness, then any performance differences between inverse models must be due to other factors that are unrelated to ill-posedness. Furthermore, simpler models such as conventional DNNs may achieve more accurate results in these cases. In this work we will show that not all inverse problems exhibit non-uniqueness, and in such cases it may be sufficient to simply use a conventional DNN.

One final challenge with the study of DIMs, especially within the context of AEMs, is scoring. One advantage of many modern DIMs is that they can propose multiple solutions for a given input (i.e., value of s). Despite this capability, nearly all studies in AEM only evaluate the accuracy of the first proposed solution. Furthermore, in practice it is often possible, or desirable, to evaluate the efficacy of several designs (e.g., via numerical simulation) and adopt the best one. Recent machine learning research has shown that modern DIMs can achieve substantially better results if several proposed solutions are considered, and that the best DIM depends on how many proposed solutions can be considered[20]. In this work we will adopt a scoring metric that accounts for this capability.

1.2 Contributions of this work

(i) The first public and accessible benchmark of inverse AEM problems. In this work we develop a benchmark dataset comprising three unique inverse AEM problems: nanophotonic shell, graphene multi-layer stack, and an all-dielectric metamaterial array. These benchmark problems were adopted from recently-published research across different AEM sub-fields, helping to ensure their relevance and significance to the broader AEM community. To support replication, we publish our benchmark datasets and documentation online. We also publish fast and easily-accessible forward simulators for each benchmark problem, which is crucial to enable the benchmarking of DIMs (see Sec. 2).

(ii) The first systematic comparison of deep inverse models for AEM tasks. Using our benchmark resources, we perform a systematic comparison of eight state-of-the-art DIMs.* One of these

* During review of this paper, we found Ma et al.[26] concurrently benchmarked three DIMs on two AEM problems, with an focus on robustness and diversity in addition

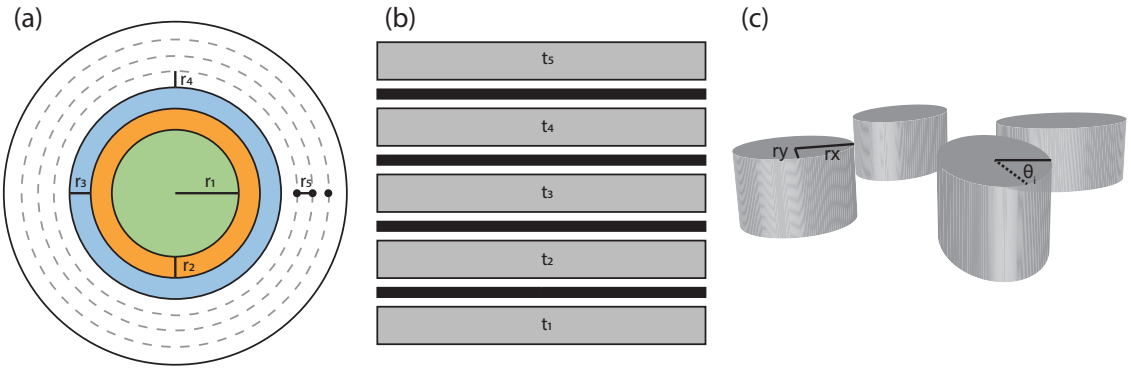

Fig. 2 Depiction of the three artificial electromagnetic materials used for inverse design test. (a) concentric plasmonic layers composing the TiO_2-Silica multi-layer Shell, (b) the multiple layers of alternating silicon nitride and graphene composing the Graphene-Si_3N_4 2D Multi-layer Stack, (c) 2×2 unit-cell of all-dielectric metasurface resonators composing the .

techniques, the conditional invertible neural network (cINN) is novel in the context of AEM research. Rather than evaluating their accuracy using a single proposed solution, following Ren et al.[20] we evaluate how each model performs as a function of the number of proposed solutions that we are permitted to evaluate (including only one solution) . We assert that this is a more practically relevant measure of performance for inverse AEM tasks (see Sec. 2.1). We find that model performance improves as more solutions are permitted, and that the relative performance of many DIMs depends upon the number of proposals that they are permitted to make.

(iii) Identifying and addressing non-uniqueness in inverse AEM Problems. Inverse problems are not guaranteed to exhibit non-uniqueness, which has several important implications when solving inverse AEM problems. We discuss these implications and make several recommendations. In particular, we suggest that researchers always include a conventional DNN as a baseline model when solving inverse problems, and we propose a measure for the level of non-uniqueness present in a given task.

The remainder of this work is organized as follows: Section 2 introduces the design of our benchmark, including the benchmark problems, performance metrics, and data handling; Section 3 describes the eight DIMs employed in our benchmark comparison; Section 4 describes our experiments; Section 5 discusses the results of our experiment; Section 6 discusses non-uniqueness in AEM inverse problems; and Section 7 presents our final conclusions and future work.

2 The inverse AEM benchmark

The objective of our benchmark is to establish a shared set of problems on which the AEM community can compare DIMs, and thereby measure research progress more reliably. To achieve this goal, we chose three initial problems to include in our benchmark, and we share resources to maximize the accessibility of the benchmark.

to the usual accuracy.

2.1 Problem formulation and error metrics

In data-driven inverse modeling we assume access to some dataset, $D = \{(g_n, s_n)\}_{n=1}^{N}$, comprising N pairs of input and output sampled from the forward model of our system (i.e., AEM system in our case). Each pair is generated by first sampling some input, $g \in G$, where G refers to some designer-chosen domain. In AEM problems G is often a hypercube as illustrated in Fig. 1, and values are sampled uniformly[1,21,24]. We assume that D is partitioned into three disjoint subsets: $D = D_{tr} \cup D_{val} \cup D_{te}$. D_{tr} is used to infer the parameters of the DIM (i.e., train the model), while D_{val} is used to monitor the progress of training and stop it at an appropriate time (e.g., before over fitting). D_{te} is the testing dataset, and the goal of inverse modeling is to use D_{tr} and D_{val} to learn a model of the form $\hat{g} = \hat{f}^{-1}(s,z)$ that produces accurate designs for all $s \in D_{te}$. In other words, the designs inferred by $\hat{f}$ should yield scattering that is similar to the desired input scattering. Many DIMs can produce multiple solutions for the same s, and the variable z controls which of these solutions is output by the inverse model.

The error of DIMs is typically measured using re-simulation error, given by $\mathscr{L}(s, \hat{s}(z)))$, where $\mathscr{L}$ refers to some measure of the error between s and $\hat{s}(z) = f(\hat{g}(z))$ (e.g., mean squared error (MSE)). We assume that we are permitted to evaluate T proposed inverse solutions by passing them through the forward model (e.g., a computational simulator) so that we can compute their re-simulation error and then take the solution with the lowest error. Then the objective of DIMs is to minimize the expected (i.e., average) error over the testing dataset, given T solution proposals. A sample estimator for this metric is given by[20]:

$$\hat{r}_T = \frac{1}{|D_{te}|} \sum_{s \in D_{te}} [\min_{i \in [1,T]} \mathscr{L}(\hat{s}(z_i), s)] \quad (1)$$

where Z_i is a set of z values that is indexed by the variable i. Note that, at $T = 1$, the re-simulation error is equivalent to mean squared re-simulation error, a widely-used metric in the AEM literature. Therefore r_T can be seen as a generalization of the conventional single-solution error measure used in most studies. Furthermore, because r_T essentially returns the minimum-error solution among all T solutions, it is a monotonically decreasing

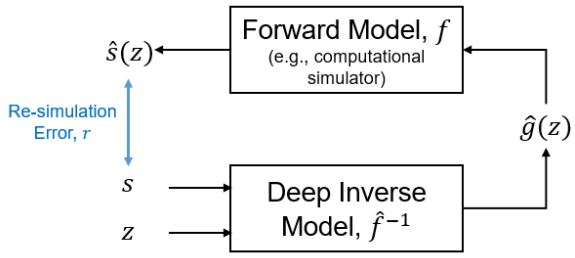

Fig. 3 Illustration of re-simulation error. Note that the variable z is only utilized by models that propose multiple solutions.

function of T (i.e., it can decrease or remain unchanged, but not increase in value).

In our benchmark study, the goal of the DIMs will be to learn an inverse function that minimizes $\hat{r}_T$. The number of solution proposals that can be tolerated may vary across different applications, and therefore we consider a variety of T values in our experiments to provide a richer characterization of the expected performance for each DIM. For our benchmark, we propose to use MSE as the loss function, $\mathscr{L}$, since it is widely used for AEM problems[21,27–30]. It is also well-behaved and well-defined for all values of s, unlike the mean-relative-error – another metric sometimes used in the AEM literature (e.g., Peurifoy et al.[24], Chen et al.[1]). MRE has the limitation that it grows exponentially as the value of $s \to 0$, and becomes infinity when $s = 0$.

2.2 Benchmark problems and selection criteria

The three AEM problems that we selected for inclusion in our benchmark are presented in Table 1, along with key details. These problems were chosen based upon several criteria to maximize the relevance of our benchmark to the AEM community. The first criterion was that each benchmark task had been studied in a recent AEM publication. A second criterion was representativeness; we deliberately chose benchmark tasks that originate in different sub-fields within AEM research. By choosing problems in this way we also help ensure that any conclusions obtained on the benchmark are more likely to generalize across AEM research. Finally, we chose problems of varying input and output dimensionality, since dimensionality is an influential factor in the performance and behavior of DIMs. Next we describe major details about each benchmark problem.

TiO_2-Silica Multi-layer Shell (Shell). The geometry of a multi-layer dielectric spherical nanoparticle of alternating, tunable thickness TiO_2 and silica shells are optimized to produce target scattering-cross section spectra – shown in Fig. 2 (a). Peurifoy et al[31] describes this problem and implements an analytical Matlab simulator replicated in python for this work. Adjustable thickness TiO_2 and silica shells parameterize the geometry of the nanosphere. We consider the 8-parameter version of this problem, where spectra are discretized by 201 uniformly spaced outputs between the wavelengths of 400-800nm.

Graphene-Si_3N_4 2D Multi-layer Stack (Stack). The geometry of a multi-layer stack of alternating graphene and Si_3N_4 dielectric layers, as depicted in Fig. 2 (b), is optimized to produce a target absorption spectra under an incident beam of s-polarized light. Chen et al.[1] describes this problem and implements an analytical transfer matrix simulator. Paired graphene-Si_3N_4 subunits of infinite width and adjustable Si_3N_4 thickness parameterize the geometry of each stack. We consider the 5-parameter version of this problem, in which spectra are discretized by 256 uniformly spaced outputs between the wavelengths of 240-2000nm. Given the design specification of this problem, it is possible that certain permutations of the stack layers leads to identical scattering. As a result, it is possible for two different design specifications to yield very similar scattering (i.e., non-uniqueness).

All-dielectric metasurface supercell (ADM). This problem was originally described and published in[21]. The ADM task was selected because it possesses several features: (1) It has 14-dimensional geometry inputs, as shown in Table 1, which is greater than many AEM studies found in the literature. The higher dimensional input results in greater complexity. Furthermore, this problem exhibits translational invariance, meaning that spatially translating the metasurface results in a different design specification, but without actually changing the underlying metasurface configuration or its scattering properties. As a result it is *guaranteed* that some unique design settings will yield the exact same scattering properties, guaranteeing that the inverse modeling problem will exhibit non-uniqueness. (2) The scattering response in this dataset is the absorptivity spectrum with 2000 frequency points and many sharp peaks that are traditionally challenging to fit. (3) This is the only dataset that is generated from full-wave simulation software. Each supercell, as shown in Fig 2, consists of four *SiC* elliptical resonators. The geometry parameters of one supercell are: height h (identical for all resonators), periodicity p, x-axis and y-axis radii r_x, r_y, and each elliptical resonator is free to rotate and described by θ. The absorptivity spectra are discretized by 2000 uniform frequency points from 100-500THz.

2.3 Neural surrogate simulators to enable replication.

One challenge with benchmarking DIMs is that it requires access to a computationally fast implementation of the true forward function, f; this is needed to compute the re-simulation error (Eq. 1) for each DIM. Many AEM problems, such as the ADM problem here, rely upon computational simulators to evaluate f, which is time-consuming to setup, difficult to replicate across studies, and computationally slow. This is a major obstacle to evaluating DIMs on many modern AEM problems that rely upon computational simulation. In this work we propose a general strategy overcome this problem, first suggested in Ren et al.,[20], that involves training a data-driven surrogate model for the simulator (e.g., a deep neural network), $\hat{f}$, and then treating this surrogate as the true simulator for experimentation with DIMs. A neural network surrogate is computationally fast that can be readily shared with other researchers for accurate replication and future benchmarking. For the ADM problem here, we adopt a surrogate simulator that was developed in Ren et al.[20], which comprises an ensemble of neural networks trained on a large dataset from the original simulator. This surrogate model is computationally fast, and highly accurate

Dataset name	Stack	Shell	ADM
Description	2D Multi-layer Stack	Multi-Layer Shell	All-Dielectric Metasurface
Geometry Dimension, $\|g\|$	5	8	14
Spectra Dimension, $\|s\|$	256	201	2,000
Training set size, $\|D_{tr}\|$	40,000	40,000	8,000
Validation set size, $\|D_{val}\|$	10,000	10,000	2,000
Test set size, $\|D_{te}\|$	500	500	500
Source publication	Chen et al.[1]	Peurifoy et al.[24]	Deng et al.[21]

Table 1 Basic statistics about benchmark dataset used

($6e-5$ with respect to the true simulator).

2.4 Benchmark resources.

The (Python) code base for our benchmark task is maintained at the following remote repository: `https://github.com/BensonRen/AEM_DIM_Bench` and can be easily downloaded. The code base includes all model architecture and code, and is open source under MIT license. We will maintain our code base through a remote repository at github. This will allow users to post comments or concerns about the code, as well as build upon the code repository.

3 Benchmark deep inverse models

In this section we describe the eight deep learning models that are included in our benchmarking experiments in Sec. 5, which are listed in Table 2. We focus here on describing the motivation for including each of the models, as well as some of their important characteristics, which are also provided in Table 2. However, the technical details of each model are reported in the Supplementary materials, and we publish software implementations of each model with this paper(see Sec. 2.4).

3.1 Overview of the Inverse Models

The eight benchmark models were included for somewhat different reasons. The Neural Network (NN) model in Table 2 represents a conventional feedforward DNN and serves as an important baseline approach. Unlike all the other models in the benchmark, the NN is not designed to address non-uniqueness. Consequently it is simpler to train and use than the other models. Furthermore, and as our results suggest in Sec. 5, it also tends to yield superior performance compared to the other benchmark models if the problem under consideration does not actually exhibit non-uniqueness. In Sec. 6 we leverage these properties of the NN to analyze the level of non-uniqueness in our benchmark problems.

The remaining benchmark models are specially-designed to address non-uniqueness in regression problems. Five of these models are included because they have been employed to solve inverse AEM problems in prior publications: the Tandem (TD), Neural-Adjoint (NA), Genetic Algorithm (GA), Variational Auto-Encoder (VAE), and the Mixture Density Network (MDN). We provide the associated AEM references for each model in Table 2. The two remaining models are the Invertible Neural Network (INN) and the Conditional INN (cINN). These models have recently been found to achieve state-of-the-art performance for solving general data-driven inverse problems[59–61], however they have not yet been explored specifically for AEM problems. In this work we explore the use of INNs and cINNs to solve inverse AEM problems for the first time.

While we believe that our benchmark models encompass a large proportion of the existing work on DIMs within the AEM community, it *is not* intended to encompass all work. We refer the reader to recent review articles for a more comprehensive treatment of the topic[3,6,7,10]. It is our hope however that AEM community will utilize our benchmark dataset and results to rigorously evaluate important additional models from the AEM literature in future work.

3.2 Key inverse model properties for AEM applications

Here we describe two key properties of DIMs with respect to AEM applications. Table 2 classifies each of our benchmark models according to these two properties. The first important property is whether a DIM produces multiple solutions. As we find in our experiments in Sec. 5, models that are permitted to propose several solutions often also find progressively better solutions (i.e., r_T reduces as T grows). This capability also makes it possible for the designer to consider several viable solutions that may have somewhat different scattering properties, and choose the one that is best-suited for the application. It is important to note that each additional model proposal that is considered requires an evaluation of the true forward model, f, which imposes a (usually modest) trade-off between design quality and computation time.

A second important property of some DIMs is that they rely on an iterative process for inferring each inverse solution. Most DIMs attempt to learn a direct mapping from s to g, and therefore inference of a single solution proposal is computationally efficient. In contrast, iterative methods usually make an initial set of guesses for the inverse solution, denoted Z_0. A search for superior solutions is then performed based upon Z_0, and the results are used to update Z_0. This process is then repeated for some fixed number of iterations, or until the quality of the solutions (e.g., estimated resimulation error) no longer improves. As we find in Sec. 5 iterative methods can often achieve the superior accuracy, although the computation required to infer solutions can be substantially larger than other methods. It is worth noting however, that this additional computation time is usually only a small fraction of the time for other processes, such as training the DIMs or evaluating

Model	Multi-solution	Iterative	Applications to AEM problems
Conventional Deep Neural Network (NN)	×	×	Chen et al.[1], Tahersima et al.[32] Zhang et al.[33], Akashi et al.[34]
Tandem (TD)	×	×	Liu et al.*[18], Ma et al.[19], Gao et al.[35], Hou et al.[36] So et al.[37] Long et al.[38], He et al.[39], Xu et al.[40] Ashalley et al.[41], Mall et al.[42], Pilozzi et al.[43] Phan et al.[44], Singh et al.[45], Malkiel et al.[46]
Genetic Algorithm (GA)	✓	✓	Zhang et al.*[47], Johnson et al.[48] Forestiere et al.[49], Li et al.[50]
Neural Adjoint (NA)	✓	✓	Deng et al.*[21], Peurifoy et al.[24] Asano et al.[51], Miyatake et al.[52]
Variational Auto-encoder (VAE)	✓	×	Ma et al.*[16,53], Qiu et al.[17] , Kudyshev et al.[54,55] Shi et al.[56], Liu et al.[23], Kiarashinejad et al.[30]
Invertible Neural Network (INN)	✓	×	-
Conditional Invertible Neural Network (cINN)	✓	×	-
Mixture Density Network (MDN)	✓	×	Unni et al.*[57,58]

Table 2 Summary of benchmark models. * indicates that we adopted the implementation presented in the published work with minimal change in our benchmark. More details about the implementation can be found in the supplementary.

f using computational simulators.

4 Experimental design

The fundamental goal of this work is to establish a rigorous comparison of the above-mentioned deep inverse models applied to different problems in metamaterial design. The overall design of our experiment is as follows: (1) Collect data: we generate the training data using code as section 2 illustrated; (2) Train the model as described in section 3 on the training set; (3) Evaluate each model's ability to reproduce the target spectra s_{gt} in the test set for every dataset, through re-simulation of the 200 predicted candidate geometry solutions $\hat{g}$.

4.1 Model optimization and training

For each dataset-DIM pair, we use 24 GPU-hours (specification of hardware used can be found in the appendix Section 6) for network training and hyper-parameter optimization (e.g., width and depth of models, learning rates, regularization strength, model-specific parameters like number of Gaussian for MDN and VAE, KL-divergence coefficient for VAE, MSE coefficient and padding dimension for INN and cINN, mutation rate of the GA etc.. We used a uniform grid of several hyper-parameters dimensions, adjusting the grid range and granularity from heuristics). This was sufficient to evaluate more than 80 hyperparameters settings for each model-dataset pair. We empirically found this number sufficient to achieve diminishing performance returns for each model. During this optimization process, we trained all models on the same training dataset, D_{tr}, for each problem. All model (all implemented in Pytorch[62]) are trained using Adam[63] with batch-norm layer and a learning rate scheduler that reduces on plateau of training loss. We used batch size of 1024 and 300 epochs (by which time all models reached convergence).

To evaluate the quality of a particular hyperparameter setting during the optimization process we measured $r_{T=1}$ on D_{val} and ultimately chose the model for final testing that achieved the lowest validation error. The final estimate of error for each model-task pair was evaluated by computing r_T on D_{te}. The final size and run time of each model-dataset pair can be found in the appendix.

5 Results and discussion

Our experimental results are presented in Fig. 4(a-c), where we plot re-simulation error (r_T, for $T \in [1,200]$) for each of the three benchmark tasks. The results indicate that an error of approximately 10^{-3} (or much lower) can be achieved for all of the benchmark tasks. The achievable error does vary significantly across the three tasks however; for example, the DIMs reach 10^{-7} on the Stack problem, and only $\sim 10^{-3}$ for the Shell problem. In Fig. 4(e-g) we plot a sample target spectrum (solid line) and the solutions produced by the top-three-performing models (dashed lines) for each task. These visualizations illustrate the high complexity of the target spectra, as well as the levels of accuracy that can be achieved by the DIMs. We believe that these levels of error are sufficient to support a variety of different AEM research and development applications, and therefore demonstrate the overall effectiveness of DIMs across a variety of AEM problems.

A major objective of these experiments is to compare the performance of state-of-the-art DIMs. This is challenging because the relative performance of the DIMs depends strongly upon the number of permissible solution proposals (i.e., value of T). This is because the error of multi-solution DIMs (see Table 2) often reduces significantly as T increases, whereas single-solution models do not. The results therefore suggest that multi-solution DIMs offer substantial value if multiple solutions can be considered during design. Furthermore, as we will show, the best-performing model and its achievable performance depend strongly upon T. This is an important finding because most existing AEM research only measures the performance of inverse models for $T = 1$, and no existing studies evaluate $T > 1$ in the fashion we have here. Since it often requires thousands of simulations to train DIMs, we assert that it should usually be possible to consider many solutions, i.e.,

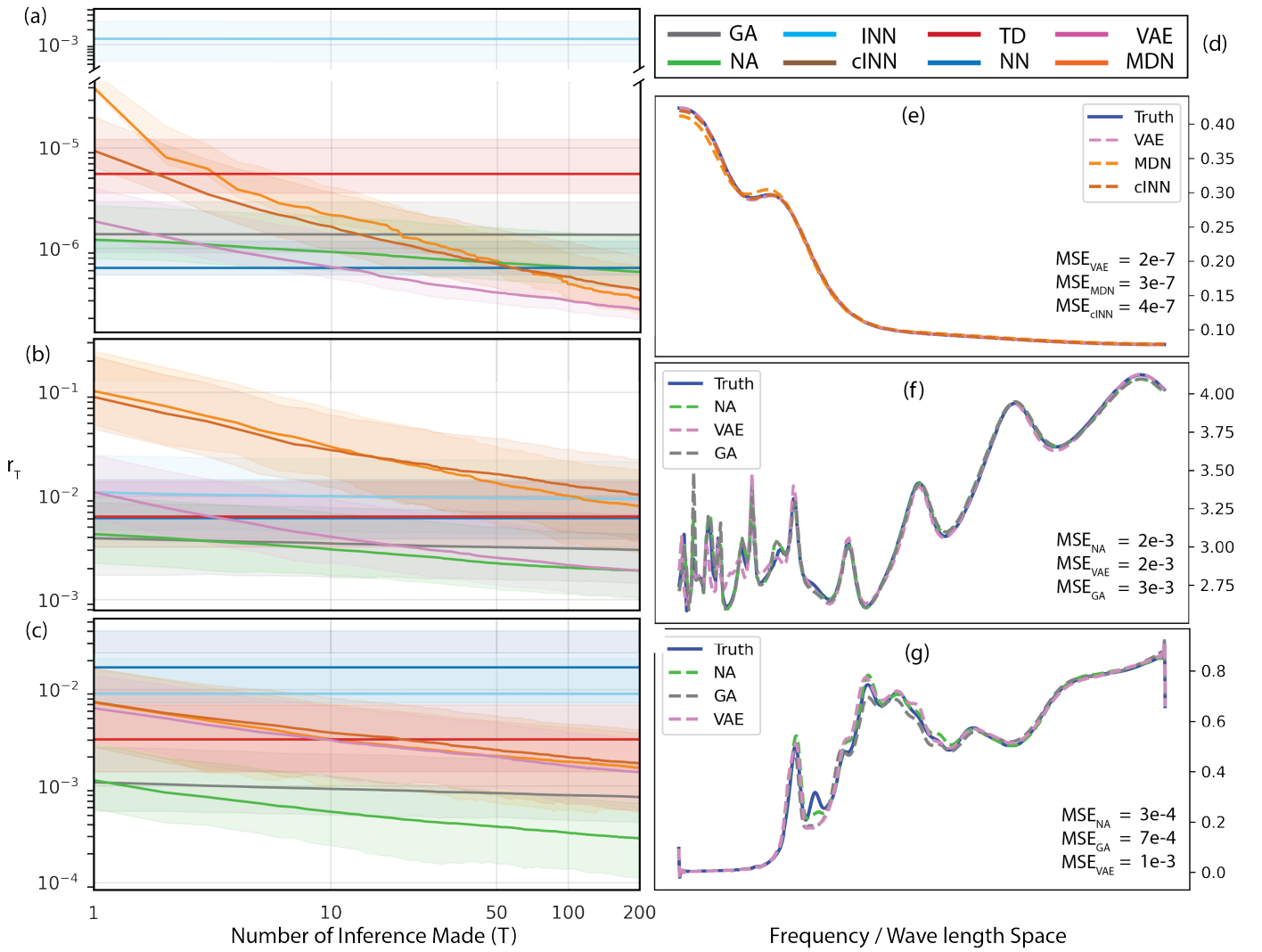

Fig. 4 Resimulation error for eight different inverse approaches for three benchmark data sets as T increases, (a) Graphene-Si_3N_4 2D Multi-layer Stack, (b) TiO_2-Silica Multi-layer Shell, and (c) All-Dielectric Super-unit Cell. Highlighted regions encapsulate 75^{th} and 25^{th} percentiles of resimulation error. $T = 1$ performance ranked on left vertical axis, $T = 200$ performance ranked on right vertical axis. (d) Legend of the plot (a-c). (e-g) The example spectra plot for T=200 for the top-3 methods shown from figure (a-c). Each spectra plotted is approximately (differ by less than 10%) the MSE of that T=200 performance, labelled using text box at right bottom corner. The ground truth target spectra is plotted with solid blue line.

$T >> 1$, making this an important performance measure.

Given the dependency of performance on T, it is difficult to make general statements about the best-performing DIMs. To simplify this analysis we discuss the performance of the DIMs under two more specific scenarios that are especially relevant to AEM applications: when $T = 1$ and $T \to \infty$, respectively. In the last section we discuss the computation time required for each of the DIMs, and its implications for selecting an appropriate DIM.

5.1 Performance when $T = 1$

The vast majority of existing AEM literature evaluates DIMs using $r_{T=1}$ (i.e., mean-squared re-simulation error) making it an important scenario to consider in our benchmark. In Table 3 we present $r_{T=1}$ for each DIM on our three benchmark tasks. Overall, the NA and GA methods achieve the lowest error, and achieve similar results on all three tasks. They achieve the lowest error on the Shell and ADM tasks, however they are outperformed by the NN model on the Stack task. In Sec. 6 we show that the Shell problem does not exhibit significant non-uniqueness, making it suitable for conventional regression models, and making it disadvantageous to use DIMs.

Given these results, the best-performing DIMs on our benchmark are iterative (see Table 2), and their performance advantages come at the cost of somewhat greater computation time compared to other DIMs. However, we note that the additional computation time is small for the tasks here: e.g., 60 seconds, or less (see Sec. 5.3).

5.2 Performance when $T \to \infty$.

In this subsection we consider the relative performance of DIMs when they are permitted to make a large number of solution proposals. It is computationally intensive to evaluate their asymp-

	Iterative		Deterministic		probabilistic			
T=1	NA	GA	NN	TD	INN	cINN	MDN	VAE
Stack	1.22e-6	1.39e-6	**6.37e-7**	4.37e-6	1.30e-3	9.38e-6	3.95e-5	1.86e-6
Shell	3.60e-3	**3.91e-3**	6.12e-3	7.13e-3	1.08e-2	9.03e-2	1.02e-1	1.09e-2
ADM	1.16e-3	**1.10e-3**	1.72e-2	1.66e-3	9.08e-3	7.45e-3	7.34e-3	6.42e-3
T=200	NA	GA	NN	TD	INN	cINN	MDN	VAE
Stack	5.83e-7	1.36e-6	-	-	1.29e-3	3.78e-7	3.03e-7	**2.46e-7**
Shell	**1.58e-3**	3.02e-3	-	-	9.36e-3	1.03e-2	8.05e-3	1.91e-3
ADM	**3.00e-4**	7.73e-4	-	-	9.05e-3	1.73e-3	1.55e-3	1.39e-3

Table 3 Table of $T = 1$ re-simulation error over different DL inverse techniques on each benchmarking task. The best performing methods are in bold. For the deterministic ones, since they do not have the ability to generate multiple solutions, the T=200 accuracy is the same as T=1 and therefore omitted.

totic performance (as $T \to \infty$) for all experiments, and therefore we use $T = 200$ as an approximation.

We argue that this is an important performance measure for inverse AEM problems because it closely reflects the goals of AEM researchers in practice. Most often researchers want the best-performing design, and will evaluate numerous candidate designs in pursuit of this goal. When using DIMs in particular, the quantity of data needed to train the models is substantially larger than $T = 200$, implying that evaluating $T = 200$ solutions (or more) with a simulator will often be feasible. Furthermore, and as we show here, doing so often yields substantially-better designs. Therefore we believe this performance measure is of particular interest to the AEM community. To our knowledge we are the first to propose this measure for evaluating the performance of inverse solvers on AEM problems.

In Table 3 we also present $r_{T=200}$ for each DIM on our three benchmark tasks. As expected, the error of most multi-solution DIMs reduces substantially as more solution proposals are permitted. In this scenario the NA method achieves the lowest error on the Shell and ADM problems. The next best-performing models are the GA and the MDN. Note that the results in Fig. 4 represent the *average* error across 500 targets (i.e., values of s), which reliably decreases as a function of T; the value of r_T for a single target will not always decrease with the addition of each solution proposal, but it is guaranteed not to increase (see Fig. A of the supplementary materials).

Regarding the Stack task, all of the models achieve very low error rates ($< 10^{-5}$). As discussed later in Sec. 5.1, the Stack problem does not exhibit significant non-uniqueness, suggesting that superior performance does not imply superior ability to address non-uniqueness. In this case many DIMs still achieve reductions in error as they make more proposals because they are making slight improvements in their estimates of the *same* inverse solution, rather than a superior unique solution. As a result, the improvements in accuracy are very small as T increases (note the logarithmic error scale).

5.3 Model computation times

Another consideration when utilizing DIMs is their computation time: specifically their training and design-inference time. The precise computation time of a DIM will depend strongly upon the particular task, model size (number of free parameters), and hardware being utilized; a full discussion of these dynamics is beyond the scope of this work. Therefore we present benchmark timings of each DIM on our three benchmark tasks, using a common hardware configuration (a single Nvidia RTX 3090 graphics processing unit). These measures therefore provide rough estimates of computation time that can be expected in many real-world settings for each model. The results of our benchmark timings are provided in Table 4.

In general the MDN, cINN, and INN models require the longest time to train, while the NN, VAE, GA and NA tend to require the least amount of time. The training times also vary across tasks, with the ADM task requiring substantially less time than the others. This is likely due to the smaller quantity of training data available compared to the other problems. Regarding inference time, the NA and GA are substantially slower than the other models, due to their iterative inference procedure. These models tend to achieve the best performance and therefore this imposes a trade-off between design quality and computation time. However, we note that the inference time of the models is relatively small compared to their training time. Furthermore, the time required for computational simulations is usually (though not always) much greater than the time required for training and (especially) solution inference with DIMs. Ultimately the relevance of these factors, and the most appropriate trade-offs, will be problem dependent.

6 Identifying and addressing non-uniqueness in inverse AEM problems

In the AEM literature DIMs are often evaluated, or compared, on an inverse problem without verifying whether the problem truly exhibits non-uniqueness. One good reason for this may be that there is no general test for determining whether a problem exhibits non-uniqueness. However, in general there is no guarantee that an inverse problem will be ill-posed, in which case DIMs may not offer any advantages over conventional regression models. As we show, conventional models may achieve superior performance in these cases. Furthermore, any performance differences between two DIMs on a well-posed problem cannot be caused

Model	Stack		Shell		ADM	
	Train	Eval	Train	Eval	Train	Eval
NN	655	0.0035	590	0.0030	50	0.0036
TD	428	0.0029	856	0.0027	198	0.0030
GA	202	24	519	26	72	25
NA	202	1.4	519	6.65	72	5.21
VAE	320	0.0023	363	0.0026	50	0.0027
INN	1843	0.0079	2919	0.015	487	0.0124
cINN	541	0.013	1131	0.019	152	0.0096
MDN	350	0.015	434	0.003	144	0.010

Table 4 Model training and evaluation time (unit of seconds) of each individual models. Note that training time is dependent of the dataset size and is nearly a constant given hyper-parameter, dataset size and same computation resources (in appendix). The evaluation time is averaged over getting 200 solutions (proposals) without taking IO time into account.

by differences in their ability to address non-uniqueness, making such performance comparisons (between DIMs) potentially misleading.

To mitigate these risks, we propose that a conventional NN always be employed as a baseline approach when solving inverse problems. This ensures that a suitable model is included in case the problem is actually well-posed, and provides a baseline performance to ensure that there is some advantage to using DIMs. Furthermore, one can treat the relative performance of a state-of-the-art DIM and conventional NN as a hypothesis test for the presence of non-uniqueness. If the DIM achieves superior performance, it implies that its modeling assumption of non-uniqueness is (likely) more accurate for the problem than the NN's assumptions of uniqueness. More precisely, we propose the following measure of non-uniqueness:

$$\gamma = \frac{r^{NA}}{r^{NN}}. \tag{2}$$

Here r^{NN} and r^{NA} are the re-simulation errors of a conventional NN and the neural-adjoint DIM, when $T = 1$. We suggest the NN and the NA because they represent state-of-the-art deep conventional and inverse models, respectively. We use $T = 1$ to avoid giving the NA an advantage by evaluating multiple of its solution proposals, while the NN can only submit one solution; in principle, if the problem is non-unique then the NA will yield superior performance even when $T = 1$. Finally, it is also important to ensure that the NA and NN are similarly-sized models (i.e., have similar number of free parameters), since model size can often impact performance.

6.1 Non-uniqueness in the AEM benchmark tasks

On top right corner of Fig 5 we present the value of γ for each of our three benchmark tasks. The measure suggests that non-uniqueness is smallest in the Stack dataset, and largest in the ADM dataset. In the Stack dataset the NN model achieves nearly *twice* the accuracy of the NA method, which is the best-performing DIM at $T = 1$ for this problem. This suggests that for some inverse problems, a simple regression model can indeed substantially outperform DIMs (when $T = 1$), and therefore it can be costly exclude them when solving inverse problems. Because the NN outperforms the DIMs, it suggests that non-uniqueness is not a major obstacle to solving this problem. This is corroborated by the extremely low error of $6.34e-7$ achieved by the NN. From Fig. 4(e-g) this level of error is qualitatively low, providing a near-perfect match with the ground truth target spectra. This also demonstrates that γ is not simply a measure of problem difficulty either, because the overall accuracy of the inverse models (r_T) and γ provide different rank-ordering of the benchmark tasks.

Crucially, these results also imply that care must be taken when interpreting the performance differences between DIMs on the Stack problem. Because there is relatively little non-uniqueness, differences in performance are largely due to factors that are unrelated to addressing non-uniqueness. Although the Stack problem is well-posed, we do observe that as T grows, the performance of the DIMs improves, and the DIMs do eventually outperform the conventional NN. We hypothesize that this occurs because the DIMs are obtaining marginally more accurate approximations of the *same* inverse solution found at $T = 1$ by the NN, rather than identifying superior unique solutions. In the Appendix we provide further evidence that this is indeed the case.

6.2 Visualizing non-uniqueness in the AEM data

To corroborate the γ measure and its implications, we also present visual evidence of the non-uniqueness present in each benchmark task. In Fig. 5(d-e) we randomly-sample a spectrum from each training dataset, s_1 (red), and then identify the four most similar spectra in the training dataset (D_{tr}), in terms of Euclidean distance (gray). We refer to this set of five total spectra as S_1. In Fig. 5(a-c), we present a scatter plot of all designs in D_{tr} along with the designs corresponding to S_1 shown in red. We cannot directly scatter plot any of the designs because their dimensionality is greater than three, and therefore we use the UMAP[64] approach to reduce their dimensionality to two for visualization.

For problems with a single unique solution, we expect that highly similar spectra will *tend* to have designs that are also similar. This is the case for the Stack problem in Fig. 5(a), while the points become increasingly distant for the Shell problem, and then again more distant for the ADM problem. We repeat this process with four additional randomly selected spectra, $s_i, i \in 2,3,4,5$ and plot the designs corresponding in each cluster with a different color. The pattern of non-uniqueness among the three tasks holds for these additional spectra. These visualizations provide additional evidence in support of the γ measure. We leave further theoretical analysis of γ for future work.

7 Conclusions

Recently deep inverse models have been found successful for solving inverse AEM problems (e.g., material design). This has led to the rapid proliferation of different models for performing inverse design, but relatively little rigorous testing or comparison among them. In this work we present the first benchmark performance comparisons of state-of-the-art deep inverse models (DIMs) on in-

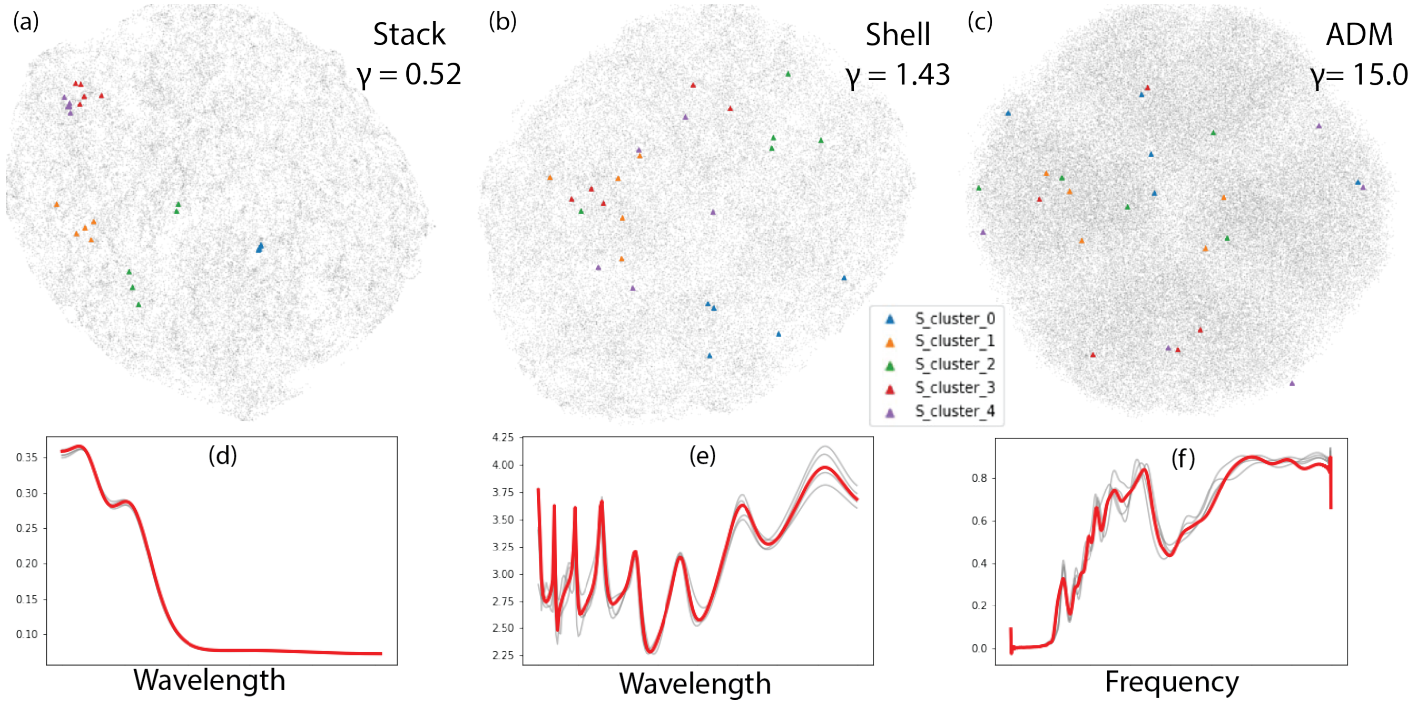

Fig. 5 Visualizing one-to-many of the benchmark datasets. (a-c) are the scatter plot of all geometry points in Stack, Shell and ADM in background in UMAP space (2d). On top of that, geometries that have similar spectra clusters (shown in (d-f)) are color-coded and plotted in triangles. There are 5 randomly selected clusters being plotted here. The more cluster the same color points are, the less one-to-many (in general) is the dataset.

verse AEM problems. We selected three inverse AEM problems to include in the benchmark, which were chosen carefully to be relevant and diverse. We then evaluated the performance of eight different DIMs on each of our three benchmark tasks. Six of these DIMs were were employed in recent AEM studies, while two of them (the INN and cINN) are introduced to the AEM community for the first time in this study.

Recent inverse AEM studies typically measure the accuracy of DIMs based upon the first solution that they propose for a target AEM scattering. However, many recent DIMs can propose multiple solutions for a given target scattering and in this work we evaluated the performance of DIMs as a function of the number of solution proposals that they were permitted to make, denoted T. Therefore, in our benchmark we evaluated DIM performance as a function of T, for $T \in [1, 200]$ This measure is much richer than the conventional performance measure (equivalent to $T = 1$), and we believe it better reflects how DIMs are often used in practice=.

Benchmarking results. The results of our benchmark indicate that DIMs can achieve error levels ranging from nearly $10e-3$ for the Shell problem, to $10e-7$ on the Stack problem. We believe this level of error (illustrated in Fig. 4(e-g)) is sufficient for a variety of applications, and demonstrates the overall effectiveness of DIMs for solving inverse AEM design problems. We find that design quality improves substantially if we are allowed to evaluate multiple solutions (i.e., $T >> 1$) from multi-solution DIMs (see Table 2). Although the performance of DIMs varies by (i) task and (ii) the value T being considered, we find that iterative methods such as the Neural-Adjoint and Genetic Algorithm tend to achieve the best performance. This performance advantage comes at greater computational cost during solution inference compared to other DIMs, however, these costs will usually be small compared to other computational costs.

Publication of resources for future benchmarking. To make our experiments easily replicable, we publish code and documentation for our benchmark datasets and DIMs. Importantly, we also publish fast and easily-usable simulators, which are necessary to benchmark DIMs. It is our hope that researchers will utilize these resources to rigorously evaluate new DIMs, as well as build upon our benchmark by adding new models and tasks.

Conflicts of interest

Authors report no conflict of interest.

Acknowledgements

We acknowledge funding from the Department of Energy under U.S. Department of Energy (DOE) (DESC0014372).

Notes and references

1 Y. Chen, J. Zhu, Y. Xie, N. Feng and Q. H. Liu, *Nanoscale*, 2019, **11**, 9749–9755.
2 I. Staude and J. Schilling, *Nature Photonics*, 2017, **11**, 274–284.
3 O. Khatib, S. Ren, J. Malof and W. J. Padilla, *Advanced Functional Materials*, 2021, 2101748.
4 J. Herskovits, *Advances in Structural Optimization*, Springer Netherlands, Dordrecht, 1995.
5 L. Huang, L. Xu and A. E. Miroshnichenko, *Advances and Applications in Deep Learning*, 2020, 65.

6 P. R. Wiecha, A. Arbouet, C. Girard and O. L. Muskens, *Photonics Research*, 2021, **9**, B182–B200.

7 J. Jiang, M. Chen and J. A. Fan, *Nature Reviews Materials*, 2020, 1–22.

8 W. Ma, Z. Liu, Z. A. Kudyshev, A. Boltasseva, W. Cai and Y. Liu, *Nature Photonics*, 2021, **15**, 77–90.

9 W. Ma, Z. Liu, Z. A. Kudyshev, A. Boltasseva, W. Cai and Y. Liu, *Nature Photonics*, 2020, 1–14.

10 S. So, T. Badloe, J. Noh, J. Rho, J. Rho and J. Bravo-Abad, *Nanophotonics*, 2020, **9**, 1041–1057.

11 L. Huang, L. Xu and A. E. Miroshnichenko, *Advances and Applications in Deep Learning*, IntechOpen, 2020.

12 J. L. Mueller and S. Siltanen, *Linear and nonlinear inverse problems with practical applications*, SIAM, 2012.

13 J. Hadamard, *Princeton University Bulletin*, 1902, 49–52.

14 Z. Liu, D. Zhu, S. P. Rodrigues, K. T. Lee and W. Cai, *Nano Letters*, 2018, **18**, 6570–6576.

15 S. So and J. Rho, *Nanophotonics*, 2019, **8**, 1255–1261.

16 W. Ma, F. Cheng, Y. Xu, Q. Wen and Y. Liu, *Advanced Materials*, 2019, **31**, 1901111.

17 T. Qiu, X. Shi, J. Wang, Y. Li, S. Qu, Q. Cheng, T. Cui and S. Sui, *Advanced Science*, 2019, **6**,.

18 D. Liu, Y. Tan, E. Khoram and Z. Yu, *ACS Photonics*, 2018, **5**, 1365–1369.

19 W. Ma, F. Cheng and Y. Liu, *ACS Nano*, 2018, **12**, 6326–6334.

20 S. Ren, W. J. Padilla and J. M. Malof, Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual, 2020.

21 Y. Deng, S. Ren, K. Fan, J. M. Malof and W. J. Padilla, *Optics Express*, 2021, **29**, 7526.

22 M. R. da Silva, C. d. L. Nóbrega, P. H. d. F. Silva and A. G. D'Assunção, *Microwave and Optical Technology Letters*, 2014, **56**, 827–831.

23 Z. Liu, L. Raju, D. Zhu and W. Cai, *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2020, **10**, 126–135.

24 J. Peurifoy, Y. Shen, L. Jing, Y. Yang, F. Cano-Renteria, B. G. DeLacy, J. D. Joannopoulos, M. Tegmark and M. Soljačić, *Science Advances*, 2018, **4**, 1–8.

25 Y. Deng, J. Dong, S. Ren, O. Khatib, M. Soltani, V. Tarokh, W. Padilla and J. Malof, 2021.

26 T. Ma, M. Tobah, H. Wang and L. J. Guo, *Opto-Electronic Science*, 2022, **1**, 210012–1.

27 Z. Hou, T. Tang, J. Shen, C. Li and F. Li, *Nanoscale Research Letters*, 2020, **15**,.

28 Y. Li, Y. Xu, M. Jiang, B. Li, T. Han, C. Chi, F. Lin, B. Shen, X. Zhu, L. Lai *et al.*, *Physical review letters*, 2019, **123**, 213902.

29 C. C. Nadell, B. Huang, J. M. Malof and W. J. Padilla, *Optics Express*, 2019, **27**, 27523.

30 Y. Kiarashinejad, S. Abdollahramezani and A. Adibi, *npj Computational Materials*, 2020, **6**, 1–12.

31 J. Peurifoy, Y. Shen, L. Jing, Y. Yang, F. Cano-Renteria, B. G. DeLacy, J. D. Joannopoulos, M. Tegmark and M. Soljačić, *Science Advances*, 2018, **4**, eaar4206.

32 M. H. Tahersima, K. Kojima, T. Koike-Akino, D. Jha, B. Wang, C. Lin and K. Parsons, *Scientific Reports*, 2019, **9**, 1–9.

33 T. Zhang, J. Wang, Q. Liu, J. Zhou, J. Dai, X. Han, Y. Zhou and K. Xu, *Photonics Research*, 2019, **7**, 368.

34 N. Akashi, M. Toma and K. Kajikawa, *Applied Physics Express*, 2020, **13**,.

35 L. Gao, X. Li, D. Liu, L. Wang and Z. Yu, *Advanced Materials*, 2019, **31**, 1905467.

36 Z. Hou, T. Tang, J. Shen, C. Li and F. Li, *Nanoscale Research Letters*, 2020, **15**,.

37 S. So, J. Mun and J. Rho, *ACS Applied Materials and Interfaces*, 2019, **11**, 24264–24268.

38 Y. Long, J. Ren, Y. Li and H. Chen, *Applied Physics Letters*, 2019, **114**,.

39 J. He, C. He, C. Zheng, Q. Wang and J. Ye, *Nanoscale*, 2019, **11**, 17444–17459.

40 L. Xu, M. Rahmani, Y. Ma, D. A. Smirnova, K. Z. Kamali, F. Deng, Y. K. Chiang, L. Huang, H. Zhang, S. Gould, D. N. Neshev and A. E. Miroshnichenko, *Advanced Photonics*, 2020, **2**, 1.

41 E. Ashalley, K. Acheampong, L. Vázquez, P. Yu, A. Neogi, A. O. Govorov and Z. Wang, *Photonics Research*, 2020.

42 A. Mall, A. Patil, D. Tamboli, A. Sethi and A. Kumar, *Journal of Physics D: Applied Physics*, 2020, **53**, 49LT01.

43 L. Pilozzi, F. A. Farrelly, G. Marcucci and C. Conti, *Communications Physics*, 2018, **1**, 1–7.

44 A. D. Phan, C. V. Nguyen, P. T. Linh, T. V. Huynh, V. D. Lam, A.-T. Le and K. Wakabayashi, *Crystals*, 2020, **10**, 125.

45 R. Singh, A. Agarwal and B. W. Anthony, *Optics Express*, 2020, **28**, 27893.

46 I. Malkiel, M. Mrejen, A. Nagler, U. Arieli, L. Wolf and H. Suchowski, *Light: Science and Applications*, 2018, **7**,.

47 T. Zhang, Q. Liu, Y. Dan, S. Yu, X. Han, J. Dai and K. Xu, *Optics Express*, 2020, **28**, 18899.

48 J. Johnson and V. Rahmat-Samii, *IEEE Antennas and Propagation Magazine*, 1997, **39**, 7–21.

49 C. Forestiere, A. J. Pasquale, A. Capretti, G. Miano, A. Tamburrino, S. Y. Lee, B. M. Reinhard and L. D. Negro, *Nano Letters*, 2012, **12**, 2037–2044.

50 C.-J. Li, Y.-C. Fang and M.-C. Cheng, *Optics Express*, 2009, **17**, 10177–10188.

51 T. Asano and S. Noda, *Optics Express*, 2018, **26**, 32704.

52 Y. Miyatake, N. Sekine, K. Toprasertpong, S. Takagi and M. Takenaka, *Japanese Journal of Applied Physics*, 2020, **59**, SGGE09.

53 W. Ma and Y. Liu, *Science China: Physics, Mechanics and Astronomy*, 2020, **63**,.

54 Z. A. Kudyshev, A. V. Kildishev, V. M. Shalaev and A. Boltasseva, *Applied Physics Reviews*, 2020, **7**, 021407.

55 Z. A. Kudyshev, A. V. Kildishev, V. M. Shalaev and A. Boltasseva, *Nanophotonics*, 2020, **1**,.

56 X. Shi, T. Qiu, J. Wang, X. Zhao and S. Qu, *Journal of Physics*

D: Applied Physics, 2020, **53**, 275105.

57 R. Unni, K. Yao, X. Han, M. Zhou and Y. Zheng, *Nanophotonics*, 2021.

58 R. Unni, K. Yao and Y. Zheng, *ACS Photonics*, 2020, **7**, 2703–2712.

59 L. Ardizzone, J. Kruse, C. Rother and U. Köthe, International Conference on Learning Representations, 2019.

60 J. Kruse, L. Ardizzone, C. Rother and U. Köthe, Workshop on Invertible Neural Networks and Normalizing Flows, International Conference on Machine Learning, 2019.

61 *Invertible Neural Nets and Normalizing Flows*, `https://invertibleworkshop.github.io/`.

62 A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai and S. Chintala, *Advances in Neural Information Processing Systems 32*, Curran Associates, Inc., 2019, pp. 8024–8035.

63 D. P. Kingma and J. Ba, *arXiv preprint arXiv:1412.6980*, 2014.

64 L. McInnes, J. Healy, N. Saul and L. Grossberger, *The Journal of Open Source Software*, 2018, **3**, 861.