Fast Electromagnetic Field Pattern Calculation with Fourier Neural Operators

Nattawat Pornthisan[1] and Stefano Markidis[2]

[1] Chulalongkorn University, Bangkok, Thailand
[2] KTH Royal Institute of Technology, Stockholm, Sweden

Abstract. Calculating the field pattern arising from an array of radiating sources is a central problem in Computational ElectroMagnetics (CEM) and a critical operation for designing and developing antenna systems. Yet, it is a computationally expensive operation when using traditional numerical approaches, including finite-difference in the time and spectral domains. To address this issue, we develop a new data-driven surrogate model for fast and accurate calculation of the field radiation pattern. The method is based on the Fourier Neural Operator (FNO) technique. We show that we achieve a performance improvement of 31x when compared to the performance of the `Meep` CEM solver when running on a desktop laptop CPU at the cost of a small accuracy loss.

Keywords: Computational Electromagnetics · Fourier Neural Operator · Electromagnetic Field Pattern · Dipole Antenna Array

1 Introduction

Computational ElectroMagnetics (CEM) is a discipline at the intersection of applied mathematics, scientific computing, and electromagnetics theory [4] with critical applications to the design and development of electromagnetics systems [3], such as antenna arrays, waveguides, resonators, radar systems, to mention a few applications. At its heart, CEM comprises several numerical techniques for the solutions of Maxwell's equations for determining the electric and magnetic fields in the vacuum or a medium. These numerical approaches range from simple finite-difference schemes (in the time and frequency domains) to the Finite Element Method (FEM), Method of Moments (MoM), to Montecarlo techniques [12].

In this work, we focus on determining the field pattern generated by the several radiating current sources, a central topic in CEM with applications to the design of antenna arrays. In particular, we aim to solve the field pattern problem of finding the fields produced in response to a source at a single frequency ω. This a central problem in CEM and has a wide range of applications. The solution to this problem can be achieved by several numerical methods [12]. However, these numerical approaches are computationally expensive as either they require (i) simulating with a constant-frequency source for a long time so

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_24

that all transient effects, from the source turn-on, vanish or (ii) solving a large linear system with complex-valued terms.

Recently, Machine Learning (ML) data-driven models have been proposed either to substitute or to augment existing traditional techniques [8,1]. Examples of such emerging ML approaches are neural networks. These methods are data-driven and work in a supervised or semi-supervised fashion – the neural networks take as input a series of input data (for instance, the pixel values of an image) and the associated labels, and an optimization process updates the network weights and biases so that given an unseen input, the neural network produces a result prediction. Minimizing the loss function, which expresses the error between the network prediction and the actual label, drives the optimization step. The final result of the training is a neural network (with its weights and biases) that can be used as a replacement for the simulation tool, employed for training input-output pairs. For this reason, these methods are also called *surrogate* methods. These methods are considerably faster than traditional approaches when considering only the prediction step (as the computationally expensive neural network training is performed offline) at the cost of lower accuracy.

One of the most recent and powerful neural network approaches is the Fourier Neural Operator (FNO) [6]. An FNO is a neural network architecture, combining Deep Learning with Fourier analysis. FNOs are designed to efficiently solve Partial Differential Equations (PDE) using a spectral method based on the Fourier transform. The key idea behind FNOs is to represent the solution to a PDE as a superposition of sinusoidal functions of different frequencies, which can be efficiently computed using the Fast Fourier Transform (FFT). The neural network is then trained to learn the mapping between the input data and the corresponding Fourier coefficients of the solution. The key difference between traditional neural networks and neural operators, like FNO, is that the latter are designed to operate on functions directly [5], rather than on discrete data points. This makes them particularly well-suited for problems that involve continuous functions or signals, such as electromagnetic waves, where the inputs and outputs are functions rather than discrete vectors.

In this work, we design and implement FNOs to calculate the electromagnetic radiation pattern in two-dimensional geometry. In particular, we create the dataset and compare the performance of our surrogate model to the `Meep` frequency-domain solver [10]. With the use of the FNO, we can able to predict in a fast way the field pattern at a fraction of the cost of the state-of-the-art `Meep` CEM solver with reasonable accuracy.

2 Methodology

We divide our work into three phases consisting of (i) generating the training of the datasets with state-of-the-art `Meep` CEM solver (ii) designing the FNO architecture, and (iii) training a neural network. That said, we will present both the technical details and simplifying constraints at each stage in the subsections below.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_24

2.1 Simulation Setup and Dataset Generation with `Meep`

Our simulation setup and methodology is inspired by Ref. [2]. First, we generate an initial dataset using the state-of-the-art Finite-Difference Time-Domain (FDTD) `Meep` CEM solver developed at MIT. More specifically, we obtain the field pattern by running the `Meep` frequency-domain solver that combines an FDTD time step with an iterative solver for complex-value linear system. The specific simulation workflow and setup is shown in Fig. 1 and as follows.

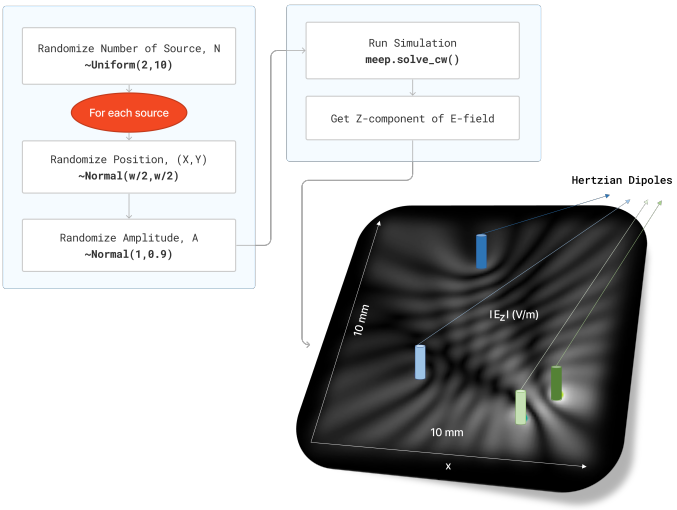

Fig. 1. Setup of `Meep` simulation for preparing the input data set to train and test the network. The generation process is shown as a stochastic process described in the two blue boxes, while an example of simulation result is shown in the lower-right corner.

The simulation region is two-dimensional on the x-y plane. The simulation domain is a 10 mm $\times$ 10mm vacuum area centered at the origin, which will be surrounded by a Perfectly Match Layer (PML) of 1mm thick to enable open boundary conditions. The resolution of the simulation is set at 10 pixels per millimeter, totaling a grid size of 140 $\times$ 140 grid points. The `Meep` time step is calculated automatically to satisfy the Courant–Friedrichs–Lewy (CFL) condition. In this work, the simulation result is the field pattern which is the magnitude of the out-of-plane electric field component, E_z in units of V/m.

To create the initial dataset, we run 2,000 `Meep` simulations with a random number of radiation sources in random locations with random amplitude and obtain the final outcome of our work: the field pattern as the E_z magnitude. More precisely, for each simulation (or item of our dataset), we first select several point sources between two and ten according to a random uniform distribution. Then, for each point source, we uniformly sample their positions within the simulation

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_24

region and their amplitude from a normal distribution with mean and standard deviation equal to 1.0 mA and 0.9 mA, respectively. All the antennas are z-axis aligned Hertzian dipoles oscillating at the same frequency of 200GHz. A few examples of field pattern is then obtained by the frequency domain solver provided by `Meep`. For visualization, we report a few examples of results from the generation process in Fig. 2.

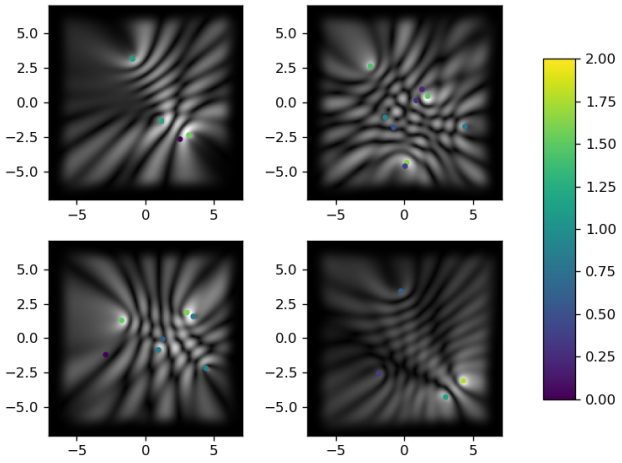

Fig. 2. Example of data generation results from `Meep`. The dots on the images represent a point source, with the amplitude indicated by the color (in mA units). The `Meep`-simulated field pattern for each source distribution is shown in the background (in V/m units). Note that the domain also includes the PML cells.

2.2 Fourier Neural Operator Architecture

To implement our network, we follow the FNO architecture, presented in the seminal paper on FNO [7]. A step-by-step visualization of the architecture can be found in Fig. 3. We implement a neural network consisting of four FNO layers which (i) apply Fast Fourier Transform (F) on the up-projected latent space, (ii) linearly transform the input in the frequency domain (R), and (iii) perform the inverse Fourier (F^{-1}) transform back to a spatial domain. As in Ref. [7], we apply a filter to eliminate the high-frequency: only 16 lower Fourier modes are linearly transformed while the rest are discarded and set to zero. The bias term (W) and activation function (σ) are then added and applied respectively at the end of each Fourier layer.

We split the datasets into training and test sets consisting of 1,760 and 240 pairs, respectively. We then train the FNO for 160 epochs (see Fig. 4) using the Adam optimizer with an L_p loss function [7]. After a few trials, we select

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_24

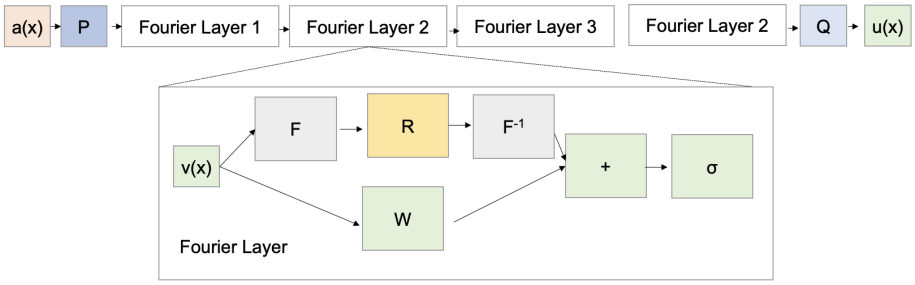

Fig. 3. The FNO architecture [7].

a suitable learning rate of 10^{-3} and a batch size of 128. The best-performing parameters on the validation set are chosen among all epochs for inference. To compare the performance with FNO, we also train a U-Net [11], a convolutional neural network architecture, commonly used for image segmentation tasks. The Python notebooks for generating the datasets with `Meep`, and performing the FNO and U-Net training can be found on the GitHub repository [3].

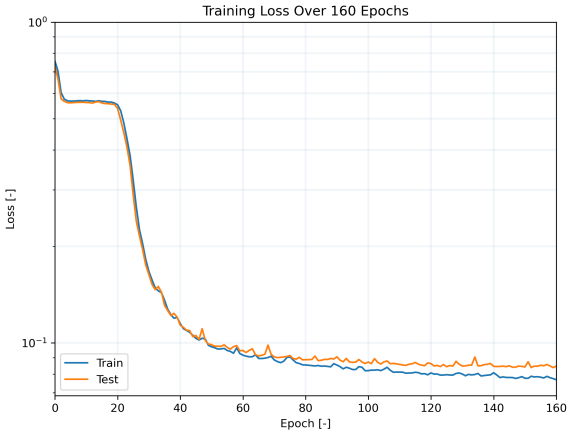

Fig. 4. The loss of the neural network over 160 epochs of optimization on training (blue) and testing (orange) datasets.

[3] `https://github.com/winnaries/fno-fastem`

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_24

3 Results

In this section, we evaluate the FNO results against ground-truth simulation regarding accuracy and performance, and compare with U-Net. In addition, we also present a generalizability of the model over a range of hand-picked input.

The FNO achieves a mean-square-error of 0.172×10^{-10}, which is 14% more accurate than U-Net model whose score is tested to be 0.207×10^{-3}. For the mean-absolute-percentage-error, the FNO achieves a significantly better score of 0.257, which is 44% more than UNet model's score of 0.456. Qualitatively, it is evident that the FNO can accurately learn the relationship among the sources while also accounting for their amplitudes. The waves' crests are visually indistinguishable to the ground truth.

We then measure the execution time of the `Meep` simulation and neural network over 50 inputs on an Ubuntu machine with an AMD Ryzen 7 5800X CPU, and a Nvidia RTX A4000 GPU. The computational performance results are summarized in Table 1. Both methods were given the same input size and evaluated on CPU since `Meep` does not support acceleration on GPU [9]. The FNO is faster than `Meep` by 31 times while maintaining high accuracy. When using the Nvidia GPU, the surrogate network is 467x faster than `Meep` running on the CPU.

Table 1. Average execution time (and its standard deviation) over 50 inputs of `Meep` field pattern solver on CPU and FNO on CPU and GPU. The speedup ratio is calculated with respect to `Meep` on CPU.

	Running Time, ms		
Method	CPU/MEEP	CPU/FNO	GPU/FNO
Mean	934.67	34.525	2.0063
±Std.	64.671	4.4655	0.22788
Speedup		31x	467x

4 Discussion & Conclusions

In this work, we developed a way to efficiently approximate the field pattern of a dipole antenna array on a xy-plane. We implemented FNOs and trained it on datasets generated with the `Meep` CEM solver. The results show that the FNO prediction has relatively small mean-square-error magnitude of 10^{-3}. The FNO is also faster than the `Meep` CEM solver by 31x on CPU for a target region size of 140×140 cells.

By comparing FNO with U-Net results, FNO is far superior to conventional convolutional networks in terms of learning the physical relationship between each given source. The datasets we trained on in this project are generated in

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_24

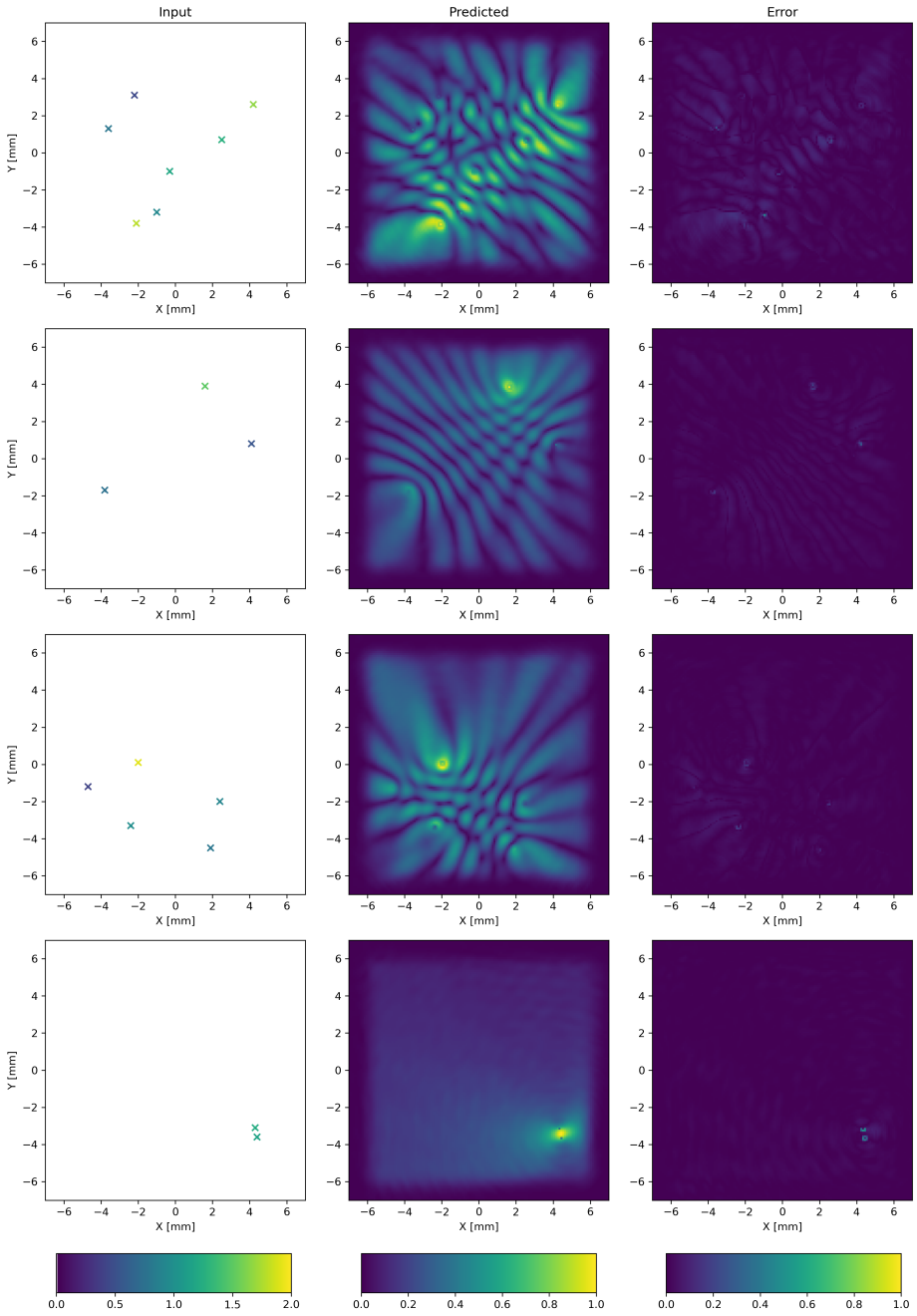

Fig. 5. Example field pattern FNO prediction output (middle) and its corresponding error with respect to the ground-truth (right) given the initial source distribution input (left). Note that the domain also includes the PML cells.

the same process. However, we did not augment the data, which is typically required for training transformation-invariant CNNs.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_24

Although the FNO mean-square error is considerably small, we cannot deny that the FNO is still data-driven and partially depends on the quality of training distribution. That said, it is safer to assume that the results from the neural network are an approximate solution that would eventually need to be validated by a conventional solver. However, the speedup that we can achieve with neural networks is undoubtedly beneficial for some large-scale simulations that might not require an exact solution.

The work done in this project needs additional efforts before we can apply it to solve real-world CEM problems, e.g., antenna arrays. As future work, the FNO architecture can consider the phase difference between each element in an antenna array, the physical relationship in 3D space, or the properties of the designated media. Moreover, for an entirely different problem set, it may be worth investigating how a generative neural network can be used to guide the design of antenna arrays given a desirable field pattern.

References

1. Aguilar, X., Markidis, S.: A deep learning-based particle-in-cell method for plasma simulations. In: 2021 IEEE International Conference on Cluster Computing (CLUSTER). pp. 692–697. IEEE (2021)
2. Baas, M.: Using fastai v2 to approximate electric fields for dipole antenna arrays (jan 2020), `https://rf5.github.io/2020/01/02/ml-emags1.html`
3. Balanis, C.A.: Advanced engineering electromagnetics. John Wiley & Sons (2012)
4. Bondeson, A., Rylander, T., Ingelström, P.: Computational electromagnetics. Springer (2012)
5. Kovachki, N., Li, Z., Liu, B., Azizzadenesheli, K., Bhattacharya, K., Stuart, A., Anandkumar, A.: Neural operator: Learning maps between function spaces. arXiv preprint arXiv:2108.08481 (2021)
6. Li, Z., et al.: Fourier neural operator for parametric partial differential equations. arXiv preprint arXiv:2010.08895 (2020)
7. Li, Z., Kovachki, N.B., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A.M., Anandkumar, A.: Fourier neural operator for parametric partial differential equations. CoRR **abs/2010.08895** (2020), `https://arxiv.org/abs/2010.08895`
8. Markidis, S.: The old and the new: Can physics-informed deep-learning replace traditional linear solvers? Frontiers in big Data p. 92 (2021)
9. MEEP - Documentation (Aug 2022), `https://meep.readthedocs.io/en/latest/FAQ/`
10. Oskooi, A.F., al.: Meep: A flexible free-software package for electromagnetic simulations by the fdtd method. Computer Physics Communications **181**(3), 687–702 (2010)
11. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015. pp. 234–241. Springer (2015)
12. Sadiku, M.N.: Numerical techniques in electromagnetics. CRC press (2000)

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_24