Time-Domain 3D Electromagnetic Fields Estimation Based on Physics-Informed Deep Learning Framework

1st Huifan Zhang
School of Information Science and Technology
ShanghaiTech University
Shanghai, China
zhanghf@shanghaitech.edu.cn

2nd Yun Hu
School of Information Science and Technology
ShanghaiTech University
Shanghai, China
huyun@shanghaitech.edu.cn

3rd Pingqiang Zhou
School of Information Science and Technology
ShanghaiTech University
Shanghai, China
zhoupq@shanghaitech.edu.cn

***Abstract*—Electromagnetic simulation is important and time-consuming in RF/microwave circuit design. Physics-informed deep learning is a promising method to learn a family of parametric partial differential equations. In this work, we propose a physics-informed deep learning framework to estimate time-domain 3D electromagnetic fields. Our method leverages physics-informed loss functions to model Maxwell's equations which govern electromagnetic fields. Our post-trained model produces accurate results with over $200\times$ speedup over the FDTD simulation. We reduce the mean square error by at least 14% and 15%, with respect to purely data-driven learning and the Fourier operator learning method FNO. In order to optimize data and physical loss simultaneously, we introduce a self-adaptive scaling factors updating algorithm, which has 8.4% less error than the loss balancing method ReLoBRaLo. Our codes are open-source at this link.**

***Index Terms*—Electromagnetic simulation, RF/microwave circuits, Physics-informed machine learning**

I. Introduction

Electromagnetic (EM) simulation is necessary in circuit simulation to analyze EM properties of electronics on multiple platforms, including RF/microwave circuits (such as microstrip filters, antenna arrays, and transmission lines), interconnect systems on both PCB [1] and heterogeneous 3D ICs [2], and also silicon photonic ICs [3]. EM simulation uses computational electromagnetic (CEM) methods to model EM fields [4], [5] by solving Maxwell's equations. Typical CEM methods include the methods of moments (MoM) [6], finite element methods (FEM) [7], finite difference time domain methods (FDTD) [8]. To meet the high performance requirements and strict operation conditions, such as frequency and input power, the design process of RF/microwave circuits is computationally intensive [9], [10]. As the number of geometric parameters increases, the design space grows exponentially. Consequently, hundreds even thousands runs of simulation are involved to fine-tune these parameters.

Artificial intelligence is promising to leverage past simulation data and reduce the runs of simulation required. Recently, deep learning (DL) architectures are used to learn the field pattern of known parametric combinations [11]. Then, EM fields of the rest unknown cases can be estimated fast and accurately by post-trained models. In [12], a convolutional neural networks (CNN) -based model is utilized to estimate magnetic fields based on given permittivity distributions and excitation source maps. Shibata et al. [13] propose a convolutional autoencoder model to rapidly compute S-parameters by using image patterns as input features.

Generally speaking, these approaches develop purely data-driven models, requiring a large amount of training data, which undermines the advantage of deep learning. Additionally, while these models may fit well to observed data, their outputs often fail to adhere to physical laws during extrapolation, resulting in a lack of robustness and generality. Physics-informed neural networks (PINNs) offer an alternative approach by incorporating prior physics knowledge into machine learning. It can be accomplished in two ways: a) imposing the partial differential equations (PDEs), boundary conditions and initial conditions which govern and constrain the physical systems into forms of loss functions in DL, and b) designing special physics-inspired neural network structures, such as Fourier Neural Operator (FNO) [14] and DeepONet [15]. In [16], Khan et al. present a PINN-based method for calculating electrostatic and magnetostatic 2D box fields. Pornthisan and Markidis apply FNO [14] to approximate the field patterns of a 2D dipole antenna array [5]. However, these aforementioned works either target a 2D electronic module or simply predict EM fields of a single timestep. A physics-informed DL solution for estimating complete 3D EM fields of multiple timesteps still remains unexplored.

In this paper, we introduce a physics-informed deep learning framework to estimate temporal EM fields with diverse geometric parameters. It is designed to learn a family of parameterized Maxwell's PDEs on microstrip RF/microwave circuits. Our main contributions are as follows:

- *For the first time*, a PINN method utilizing physics-informed loss functions and a multi-channel CNN-based neural network backbone is proposed to solve time-

This work was supported by the NSFC under award 62074100.

978-3-9826741-0-0/DATE25/© 2025 EDAA

domain 3D EM fields on parametric RF/microwave circuits, achieving a $200\times$ speedup compared to the FDTD method and a $400\times$ speedup over the FNO.

- Compared to the FNO, our method reduces the error by at least 15% and with 40% smaller neural network model (as measured by the number of parameters in the model). Compared to the purely data-driven methodology under the same network backbone, our physics-informed learning method has 14% less error.
- We implement an algorithm to adaptively update loss scaling factors of data loss and physical loss. Compared to the loss balancing method ReLoBRaLo [17], our self-adaptive loss balancing algorithms reduce data error by 8.4%.

II. Related Works

A. EM simulation using FDTD method

We choose the FDTD method as a representative of traditional CEM methods to generate training and test data, because it is widely used in modeling EM problems of rectangular shape [18]. Meanwhile, it provides and stores complete EM fields data in a matrix form, which is convenient to be constructed as a dataset for neural networks.

Given a time-harmonic current source $\mathbf{J}(t)$, and in the absence of magnetic current sources, the EM fields produced by the source adheres to Maxwell's equations:

$$\nabla \times \mathbf{E}(t) = -\mu_r \mu_0 \frac{\partial \mathbf{H}(t)}{\partial t}, \nabla \times \mathbf{H}(t) = \mathbf{J} + \epsilon_r \epsilon_0 \frac{\partial \mathbf{E}(t)}{\partial t} \quad (1)$$

where $\mathbf{E}(t)$ and $\mathbf{H}$(t) are temporal 3D electric and magnetic fields at time t, which can be decomposed as $\mathbf{E}(t) = \hat{\mathbf{a_x}}\mathbf{E}_x(t) + \hat{\mathbf{a_y}}\mathbf{E}_y(t) + \hat{\mathbf{a_z}}\mathbf{E}_z(t)$, $\mathbf{H}(t) = \hat{\mathbf{a_x}}\mathbf{H}_x(t) + \hat{\mathbf{a_y}}\mathbf{H}_y(t) + \hat{\mathbf{a_z}}\mathbf{H}_z(t)$ in Cartesian coordinates del. ϵ_0 and ϵ_r are the vacuum and relative electric permittivity, μ_0 and μ_r are the vacuum and relative magnetic permeability, and $\mathbf{J}$ is the electric current source. The FDTD method defines the EM fields on a stair-like mesh grid and updates them recursively in the time domain. We implement the FDTD method using the open-source CEM simulation toolkit MindElec[1].

B. Loss balancing for PINNs

PDEs, boundary conditions, and initial conditions can be introduced into the PINNs by adding multiple loss functions. Therefore, the training process of PINNs can be viewed as a multi-objective learning task. In such circumstances, different physical and data losses each possess their own local minima [19], [20]. When global gradient descent methods, such as Adam or Stochastic Gradient Descent (SGD), are used to optimize combined loss functions with fixed scaling factors, PINNs may fail to learn from data if too much weight is placed on the physical loss term. Conversely, if the scaling factor is too low, it can lead to overfitting, similar to purely data-driven methods. Hence, balancing the multi-objective losses during the training of PINNs is crucial for achieving fast convergence, reaching global minima, and improving test accuracy.

Wang et al. [21] investigate a fundamental issue in PINNs related to the failure of back-propagated gradients, which is caused by imbalanced gradients, and they propose an adaptive learning rate annealing algorithm. The scaling factor of loss function term i at optimization step t is calculated by:

$$\hat{\lambda}_i = \frac{\max_{\theta_t}\{|\nabla_\theta \mathcal{L}_r(\theta_t)|\}}{\overline{|\lambda_i \nabla_\theta \mathcal{L}_i(\theta_t)|}}, i \in \{1, \ldots, k\} \quad (2)$$

$$\lambda_i = (1-\alpha)\lambda_i + \alpha\hat{\lambda}_i \quad (3)$$

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}_r(\theta_t) - \eta \sum_{i=1}^{M} \lambda_i \nabla_\theta \mathcal{L}_i(\theta_t) \quad (4)$$

where θ_t denotes the network parameters at epoch t, and $\nabla_\theta \mathcal{L}_i$ denotes the gradients of loss function $\mathcal{L}_i$. η and α are hyperparameters. However, the calculation based on detailed layer-wise gradient statistics is difficult to implement on complex network backbones.

Adaptive loss balancing methods proposed in the computer vision scenario offer a different perspective to solve the identical issue in PINNs. Heydari et al. [22] propose SoftAdapt, a Softmax-inspired algorithm for adaptive multi-objective loss weighting. The scaling factor λ_i is normalized by:

$$\lambda_i = \frac{\exp\left(\mathcal{T}(\mathcal{L}_i - \max(\mathcal{L}_i))\right)}{\sum_{j=1}^{k} \exp\left(\mathcal{T}(\mathcal{L}_j - \max(\mathcal{L}_j))\right) + \epsilon}, i \in \{1, \ldots, k\} \quad (5)$$

where $\mathcal{T}$ is the hyperparameter temperature, and $\epsilon = 10^{-8}$ for numerical stability.

In [17], Bischof and Kraus integrate the concepts of the aforementioned SoftAdapt [21], GradNorm [23], and learning rate annealing algorithm in [21] to form a novel scheme termed Relative Loss Balancing with Random Lookback (ReLoBRaLo):

$$\lambda_i^{bal}(t, t') = m \cdot \frac{\exp\left(\frac{\mathcal{L}_i(t)}{\mathcal{T}\mathcal{L}_i(t')}\right)}{\sum_{j=1}^{m} \exp\left(\frac{\mathcal{L}_j(t)}{\mathcal{T}\mathcal{L}_j(t')}\right)}, i \in \{1, \ldots, m\} \quad (6)$$

$$\lambda_i^{hist}(t) = \rho\lambda_i(t-1) + (1-\rho)\lambda_i^{bal}(t, 0)) \quad (7)$$

$$\lambda_i(t) = \alpha\lambda_i^{hist} + (1-\alpha)\lambda_i^{bal}(t, t-1) \quad (8)$$

where α is the exponential decay rate, ρ is a Bernoulli random variable and $\mathbb{E}[\rho]$ should be chosen close to 1. $\lambda_i^{bal}(t, t')$ calculates the scaling factor based on the relative improvements of each loss function term from step t' to current step t. However, this method has several drawbacks:

- The weights and biases in neural networks are typically initialized using normal or uniform distributions, which can result in a relatively low residual loss of the physical term. As a result, looking back to the very beginning of training is not particularly meaningful.
- When the decline in losses reaches a plateau, the expression $\exp\left(\mathcal{L}i(t)/\mathcal{T}\mathcal{L}i(t')\right)$ can produce an extremely large value, potentially leading to overflow.

To solve the above problems, we present a self-adaptive loss balancing method which achieves better accuracy and aviods risky overflow.

[1] https://gitee.com/mindspore/mindscience/tree/master/MindElec

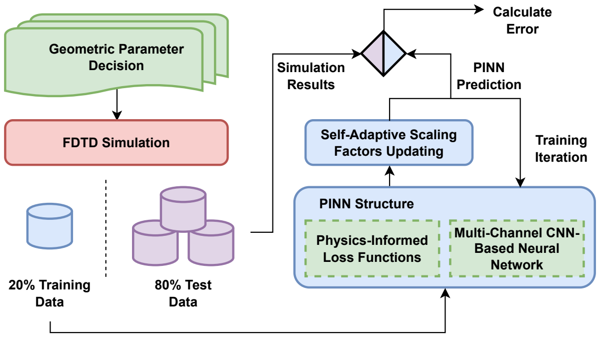
Calculate Error
Geometric Parameter Decision
Simulation Results
PINN Prediction
FDTD Simulation
Self-Adaptive Scaling Factors Updating
Training Iteration
PINN Structure
Physics-Informed Loss Functions
Multi-Channel CNN-Based Neural Network
20% Training Data
80% Test Data

Fig. 1. Workflow of the proposed PINN framework.

III. The Proposed PINN Framework

The overall workflow of our proposed PINN framework is shown in Fig. 1. Firstly, we completely sample all possible combinations in the geometric parameters range. Then, FDTD simulations are conducted to build the EM fields dataset for these samples. We use only 20% of them for training and the remaining 80% for test to prove that our PINN can be trained with a small amount of data. In the neural network structure section, we propose a multi-channel CNN-based backbone to learn EM fields evolution and their relations with geometric floorplan. Physics knowledge is integrated into the PINN as loss functions. Further more, we design a novel self-adaptive balancing algorithm to update scaling factors of different loss functions.

A. Physics-informed loss functions

Suppose $u(\Omega_i;\theta)$ is a neural network that approximates the ground truth EM fields $EM_t(\Omega)$ where θ represents all the trainable parameters. Here, Ω represents the 3D time-varying discrete domain of dimension (T, z, h, w) where T represents the number of total timesteps. z, h, and w are the thickness, height, and width of fields in spatial domain. The uniform timestep is Δ_t and the mesh grid unit size is $(\Delta_x, \Delta_y, \Delta_z)$.

To leverage both data-driven and physics-informed learning, the loss function is defined as:

$$L_{total} = \lambda_d * L_d(\Omega;\theta) + \lambda_p * L_p(\Omega;\theta) \tag{9}$$

$$L_d(\Omega;\theta) = \frac{1}{N}\sum_{i=1}^{N} ||u(\Omega_i;\theta) - EM_t(\Omega_i)||_2^2 \tag{10}$$

$$L_p(\Omega;\theta) = \frac{1}{N}\sum_{i=1}^{N} ||f(u(\Omega_i;\theta))||_2^2 \tag{11}$$

where L_d and L_p represents data loss and physical loss, respectively, and $f(\cdot)$ is the discrete approximation of Maxwell's equations in residual form. For example, the residual form Lenz's law function of $\mathbf{E}_z, \mathbf{H}_x, \mathbf{H}_y$ can be expressed as:

$$f(\mathbf{E}_z, \mathbf{H}_x, \mathbf{H}_y) = \mathbf{E}_z + \epsilon\frac{\partial \mathbf{E}_z}{\partial t} - \left(\frac{\partial \mathbf{H}_y}{\partial x} - \frac{\partial \mathbf{H}_x}{\partial y}\right) \tag{12}$$

The residual form Ampère's law function of $\mathbf{E}_x, \mathbf{E}_y, \mathbf{H}_z$ is:

$$f(\mathbf{E}_x, \mathbf{E}_y, \mathbf{H}_z) = -\sigma_m\mathbf{H}_z - \mu\frac{\partial \mathbf{H}_z}{\partial t} - \left(\frac{\partial \mathbf{E}_y}{\partial x} - \frac{\partial \mathbf{E}_x}{\partial y}\right) \tag{13}$$

In our proposed framework, EM fields are represented discretely in 4D arrays. The location-relevant and time-relevant derivatives can be discretized in align with the Yee's grid arrangement in the FDTD simulation [24]. Furthermore, the deep learning platform supports the automatic calculation of gradients using the second-order accurate central differences approach.

In summary, both data-driven loss and physics-informed loss functions are implemented, with the latter structured to align with the discrete update equations used in the FDTD method, suggesting that the ideal physical loss should approach zero.

B. The multi-channel CNN-based neural network backbone

Inspired by SimVP [25], a simple video prediction model built entirely upon CNN, we propose a CNN-based neural network as the backbone of PINN. As shown in Fig. 2, it consists of an encoder, a translator, and a multi-channel decoder.

The encoder extracts input features of shape (C_{in}, h, w) where h and w equal the height and width of circuit floorplan mesh (same as aforementioned EM field dimensions). C_{in} represents the number of decisive features considered, including geometry structures, permittivity, permeability, and source current density distribution. The translator unfolds the encoded features along the time dimension and learns spatio-temporal evolution. Finally, the decoder restore the hidden features back to the field domain. The output of network are temporal EM fields of shape (T, C_{em}, z, h, w). C_{em} is the number of EM field components, and the rest are clarified in the previous section III.A. It is worth noting that, to reduce the complexity of model, the decoder part is time-independent, which means temporal information is completely learned by the translator.

Encoder. The encoder follows the typical architecture of U-Net [26]. Each encoder module consists of a 2×2 max pooling layer for down-sampling, a double convolution layer (Conv2d + BatchNorm + Conv2d), and a ReLU activation layer. The channels of each module are twice those of the previous module. Spatial features after different layers are skip connected with the decoder.

Translator. The translator formulates hidden features whose shape is $(T \times C_{hid}, h_{hid}, w_{hid})$. N_t inception blocks are stacked in U-shape, each of them contains 4 group convolution blocks with different kernel sizes.

Decoder. As the reverse of the encoder, the decoder utilizes transposed convolution to which convolutes on the concatenation of encoder bypass and previous translator tensors. The channels of each convolution is $C_{hid} \times C_{em}$ grouped by C_{em} to prune the connections between different field components and reduce convolutional kernel sizes.

C. Self-adaptive scaling factors balancing algorithm with periodic lookback

We propose a novel self-adaptive balancing algorithm that updates scaling factors of multiple data-driven and physics-informed terms in the loss functions:

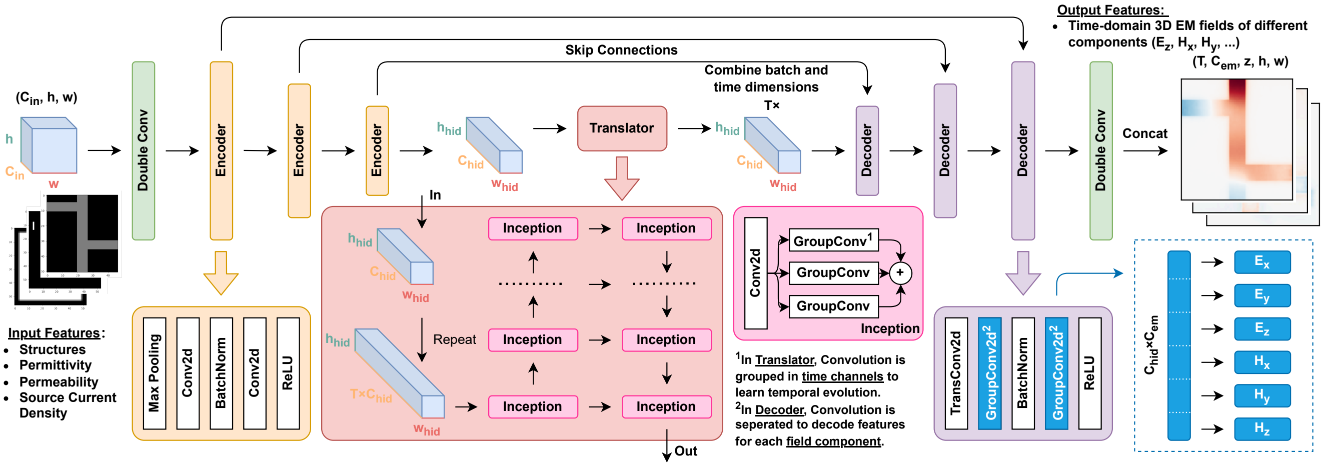

Fig. 2. The multi-channel CNN-based neural network backbone which consists of an encoder, a translator, and a multi-channel decoder. Input features include geometry structures, permittivity, permeability, and source current density. Output features are different EM field components covering T timesteps.

- The process starts with learning from data, then adjusts the physics-informed factor based on the peak and periodic loss improvements during this process.
- Unlike ReLoBRaLo, we look back to the maximum loss during the first data-driven process rather than initial loss, in order to prevent the model from converging too early to a local minimum of the physics-informed loss.
- The balance factor is computed using a softmax function to ensure that the sum of the ratios remains bounded. Similar to SoftAdapt, the difference between time steps is calculated by subtraction rather than division.

The details of the algorithm implementation are presented in Algorithm 1. During the first data-driven learning process, the maximum physical loss serves as the reference for assessing long-term loss reduction. In the second physics-informed learning phase, the history scaling factor, λ_i^{hist}, is determined by both the maximum-to-current factor, $\lambda_i^{bal}(t, t_{max})$, and the periodic-lookback factor $\lambda_i^{bal}(t, t-M)$. The hyperparameter β controls the weighting between long-term and short-term improvements. When $\beta = 1$, only the difference relative to the epoch with the maximum physical loss, L_i^{max}, is taken into account.

IV. Experiment Results

A microstrip is an integrated transmission line structure composed of a strip conductor and a ground plane, separated by a dielectric substrate. It is one of the most fundamental elements in RF/microwave circuits [27]. To demonstrate the generality of our approach in modeling different circuits, we choose the microstrip bandpass filter (MBPF) and inverted F-shape antenna (IFA) as examples. Their floorplan designs are shown in Fig. 3. We define the geometric variables as $L1$, $L2$, and D for MBPF and $L1$, $L2$, $L3$ and $L4$ for IFA. The ranges of these parameters are provided in Table I. The sample step for both cases is limited to one unit. The total dataset sizes are 200 for the MBPF and 256 for the IFA. In the time dimension, we set $T = 25$.

Algorithm 1 Self-adaptive scaling factors balancing algorithms

```
Require: Temperature $\mathcal{T}$, Data-driven loss function $L_d$, Physics-informed loss function $L_p$, lookback period $M$, Total epochs $t_{total}$, weight to the past scaling $\alpha$, weight to the maximum lookback $\beta$, numerical stability factor $\epsilon = 10^{-8}$
Ensure: Loss scaling factors for each loss function item $\lambda_i$ at epoch $t$
    Data-driven initial learning process:
while $t_{max} - t <= M$ do
    if $L_p > L_{pmax}$ then
        $L_{pmax} = L_p$
        $\mathcal{T}_{max} = t$
    end if
    $t = t + 1$
    $\lambda_d = 1,\ \lambda_p = 0$
end while
    Physics-informed learning process:
while $t < t_{total}$ do
    for $i$ in loss function terms do
        $\lambda_i^{bal}(t, t') = \frac{\exp\left((\mathcal{L}_i(t) - \max(\mathcal{L}_i(t')))/\mathcal{T}\right)}{\sum_{j=1}^{k} \exp((\mathcal{L}_i(t) - \max(\mathcal{L}_i(t')))/\mathcal{T}) + \epsilon}$, $i \in \{1, \ldots, k\}$
        $\lambda_i^{hist}(t) = \beta\lambda_i^{bal}(t, t_{max}) + (1-\beta)\lambda_i^{bal}(t, t-M)$
        $\lambda_i(t) = \alpha\lambda_i(t-1) + (1-\alpha)\lambda_i^{hist}$
        return $\lambda_i(t)$
    end for
end while
```

TABLE I
Parameters Range of MBPF (Left) and IFA (Right)

Symbol	Min	Max
$L1$	5 units	14 units
$L2$	35 units	44 units
D	5 units	6 units

Symbol	Min	Max
$L1$	10 units	13 units
$L2$	24 units	27 units
$L3$	40 units	43 units
$L4$	20 units	23 units

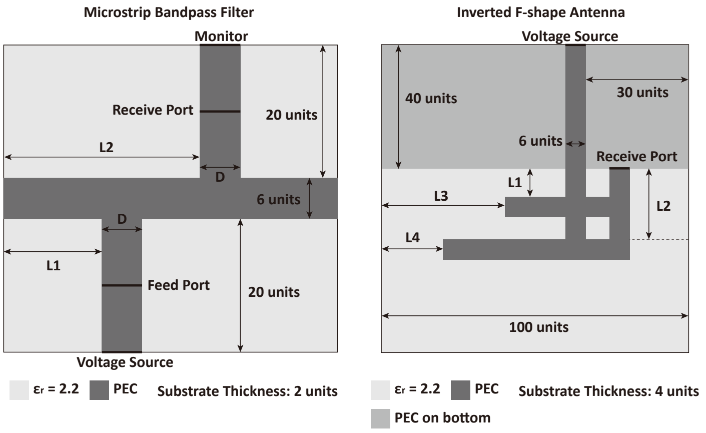

Fig. 3. Parametric floorplan of the microstrip bandpass filter and inverted F-shape antenna.

A. FDTD settings

The 3D mesh grid unit size set in FDTD simulations is $(0.4064, 0.4233, 0.265)$ mm for MBPF and $(0.4, 0.4, 0.262)$ mm for IFA. The total modeling object size is set to be $(51, 47, 2)$ units and $(100, 100, 4)$ units, respectively. For unifying the initial condition, an identical Gaussian waveform $\exp\left(-\left((t-4.5\tau)/\tau\right)^2\right)$ is applied on the feed port where $\tau = 0.5/f_{max} = 0.5/20\,\text{GHz} = 2.5 \times 10^{-11}$ s is the maximum Nyquist sampling interval. In the time domain, sequences of length 25 with an interval of Δ_t are sampled. The time interval is estimated based on CFL condition $\Delta_t = (1/c)/\sqrt{1/\Delta_x^2 + 1/\Delta_y^2 + 1/\Delta_z^2} * \text{CFL number}$.

B. Training settings

FNO is a well recognized state-of-art physics-informed ML-based architecture [5], [28]–[30] for learning parametric PDEs. However, the FNO structure is designed for image-to-image regression. In order to learn the time-domain EM fields, we need to combine FNO with recurrent structures, such as RNN, Long Short-Term Memory (LSTM) [31], or Gate Recurrent Unit (GRU) [32]. The recurrent unit in GRU can adaptively capture both short-term and long-term dependencies while being more computationally efficient than the LSTM cell. In this paper, we implement the FNO-GRU as a representative for existing physics-informed learning methods for comparison with our proposed approach.

We implement, train, and test all the FNO-GRU, UNet, and our PINN models on PyTorch 2.2.2. All the experiments are conducted on a cluster with 4 Intel Xeon CPU E5-269 CPUs and 8 NVIDIA GeForce GTX 1080 GPUs. During the training process, weights and bias in convolutional and fully connected layers are initialized using Kaiming uniform distribution. The Adam optimizer with a learning rate of 10^{-3} is employed. Each model is trained for 10,000 epochs, with a batchsize of 32.

C. Test settings

We use mean square error (MSE) as metrics to evaluate data-driven and physics-informed loss as shown in Eq. 10 and Eq. 11. The data loss is calculated by error between results from FDTD and estimated by the model, and the physical loss is determined by the error between two sides of Maxwell's equations. Because the source waveform excited has only E_z element, TM^z polarization is assumed.

D. Loss balancing algorithm settings

We implement a vanilla fixed scaling factor method and ReLoBRaLo to compare with our proposed scaling factors balancing algorithm. For fair comparison, the hyperparameters are optimized with reference to Table VII in [17]. Results comparison of different methods are shown in the next section, and the final choices of hyperparameters are shown in Table II.

TABLE II
FINAL CHOICES OF HYPERPARAMETERS

Symbol	Hyperparameter	ReLoBRLo	Our proposed
$\mathbb{E}[\rho]$	Expected saudade	0.99	-
$\mathcal{T}$	Temperature	10^{-2}	10^{-2}
α	Past scaling weight	0.99	0.99
β	Periodic lookback weight	-	0.1

E. Results

In Table III, we compare the number of parameters in the model, data and physical error by different models of the example MBPF. Since physics-informed loss function is not employed in the FNO method, that item is omitted. The test error of the example IFA is shown in Table IV. The network structures are identical, with the only difference being output channels, which vary due to different substrate thicknesses.

The FNO-GRU model contains a 4-layer FNO backbone to map between input features and initial fields, and another 5-layer FNO connected by GRU to learn the temporal evolution. The vanilla UNet architecture encompasses only an encoder and a decoder, excluding a translator. EM fields at different timesteps are learned in the multi-channel decoder where channel number is the same as timesteps number. The symbol λ within the table denotes the fixed scaling factor for the physical loss term. A value of $\lambda = 0$ implies purely data-driven learning.

First, we illustrate our results of the MBPF test case. An example of the EM field components at different timesteps is shown in Fig.4. Both estimated and FDTD simulation results and their pixel-wise absolute error are shown.

Training Resources. The number of our proposed model's parameters is 42% fewer than that of FNO-GRU and only 6% omore the UNet structure. The training time of FNO-GRU costs 22.2s per epoch, while our proposed structure takes only 7.05s per epoch, reducing 68% of training time.

Inference Time. In the MBPF example, our FDTD simulator costs approximately 2.29s for a single simulation. The FNO-GRU models takes 4.86s due to the time-consuming Fast Fourier Transformation and the recurrent structure. Our PINN model only spends 0.01s, which corresponds to a speedup of over 200× compared to the FDTD method and over 400× compared to the FNO-GRU .

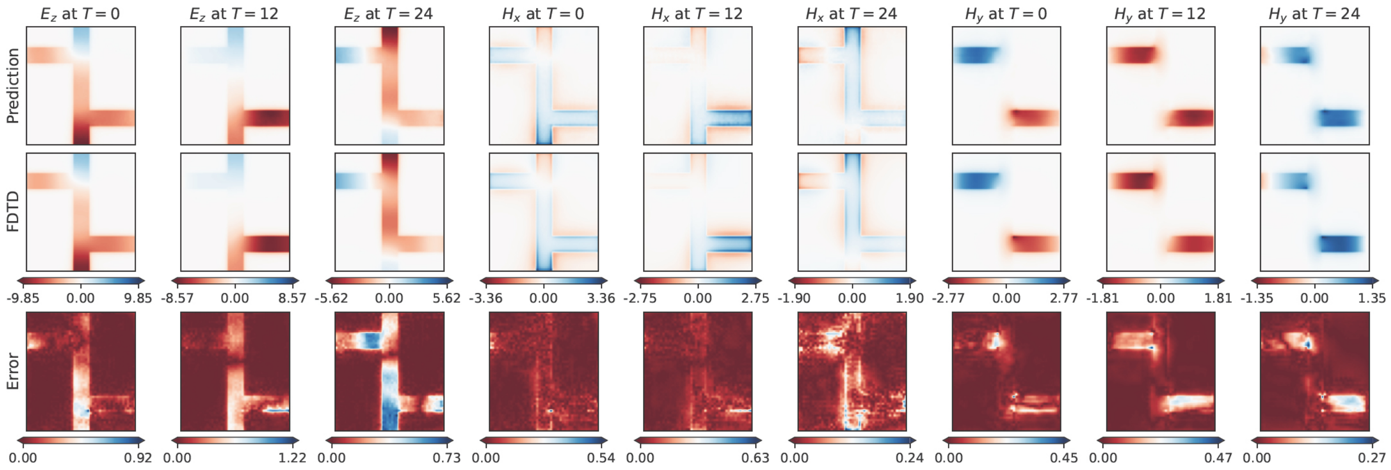

Fig. 4. Different components of the EM fields estimated by our proposed PINN and generated by FDTD simulation with comparison (MBPF case).

Error. For the MBPF test data, the MSE of FNO-GRU is 0.0331 and of UNet is 0.0843, our proposed multi-channel CNN-based structure reduces it to 0.0327 *with purely data-driven learning*, proving the priority of our neural network backbone. With the inclusion of physics-informed loss functions, data loss on the test set initially decreases alongside the physical loss, as the model becomes more physically consistent. However, as the weight coefficients continue to increase, the model begins to underfit the known data, leading to a subsequent rise in data error. When $\lambda_p = 0.01$, the minimum data error is acquired, which accomplishes 15% reduction to the FNO-GRU. In the IFA example, that ratio is 71%.

Our proposed self-adaptive scaling factors updating algorithm achieves a balance between data and physics loss. Compared with the ReLoBRaLo method, our algorithm reduces data error by 8.4% in the MBPF example and 55% in the IFA example.

TABLE III
COMPARISON OF DIFFERENT MODELS' PARAMETERS NUMBER, TEST DATA ERROR AND PHYSICAL ERROR ON THE EXAMPLE MBPF

Model	#Params(M)	Loss Balancing	Test Error	
			Data	Physics
FNO-GRU	56.79	-	0.0331	-
UNet	30.89	$\lambda_p = 0$	0.0843	0.0344
		$\lambda_p = 0.01$	0.0731	0.0290
		$\lambda_p = 0.1$	0.0713	0.0124
Proposed	32.81	$\lambda_p = 0$ (data-driven)	0.0327	0.00816
		$\lambda_p = 1 \times 10^{-3}$	0.0313	0.00779
		$\lambda_p = 1 \times 10^{-2}$	**0.0281**	0.00673
		$\lambda_p = 1 \times 10^{-1}$	0.0310	0.00422
		$\lambda_p = 1$	0.0354	**0.00215**
		ReLoBRaLo	0.0321	0.00377
		Self-Adapt	**0.0294**	**0.00368**

TABLE IV
COMPARISON OF DIFFERENT MODELS' TEST DATA ERROR AND PHYSICAL ERROR ON THE EXAMPLE IFA

Model	Loss Balancing	Test Error	
		Data	Physics
FNO-GRU	-	0.00823	-
Proposed	$\lambda_p = 0$ (data-driven)	0.00354	0.90711
	$\lambda_p = 1 \times 10^{-5}$	0.00371	0.30321
	$\lambda_p = 1 \times 10^{-4}$	**0.00294**	0.05998
	$\lambda_p = 1 \times 10^{-3}$	0.00328	0.01684
	$\lambda_p = 1 \times 10^{-2}$	0.00647	**0.00913**
	ReLoBRaLo	0.00527	0.00107
	Self-Adapt	**0.00237**	**1.048×10^{-5}**

V. CONCLUSION

Current efforts to address 3D EM fields modeling problems using physics-informed DL either concentrate on individual 2D field components or restricts to only step-to-step prediction. In this paper, we propose a PINN framework with physics-informed loss functions and a multi-channel CNN-based neural network backbone. With our novel self-adaptive scaling factors balancing algorithm, a unified minimum of both data and physical loss is ensured. Experiments conducted on two microstrip RF/microwave circuits demonstrate that our approach enables faster modeling compared to the FDTD simulation. It also prevails in accuracy among purely data-driven learning and the FNO method. The results reveal a promising application in agile RF/microwave circuits design with fewer simulations required.

REFERENCES

[1] M. V. Ugryumova and W. H. Schilders, "On passivity of the super node algorithm for em modeling of interconnect systems," in *2010 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2010, pp. 471–476.

[2] S. Zhang, Z. Li, H. Zhou, R. Li, S. Wang, K.-W. Paik, and P. He, "Challenges and recent prospectives of 3d heterogeneous integration," *E-Prime-Advances in Electrical Engineering, Electronics and Energy*, vol. 2, p. 100052, 2022.

[3] J. Youn, L. Ramini, Z. Lu, A. Alam, J. Pond, M. Fiorentino, and R. G. Beausoleil, "Multiphysics design and simulation methodology for dense wdm silicon photonics," in *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2023, pp. 1–2.

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

[5] N. Pornthisan and S. Markidis, "Fast electromagnetic field pattern calculation with fourier neural operators," in *International Conference on Computational Science*, 2023, pp. 247–255.

[6] R. F. Harrington, *Field computation by moment methods*. Wiley-IEEE Press, 1993.

[7] J.-M. Jin, *The finite element method in electromagnetics*. John Wiley & Sons, 2015.

[8] K. S. Kunz and R. J. Luebbers, *The finite difference time domain method for electromagnetics*. CRC press, 1993.

[9] W. Schoenmaker, P. Meuris, C. Strohm, and C. Tischendorf, "Holistic coupled field and circuit simulation," in *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2016, pp. 307–312.

[10] K. Bittner, H. G. Brachtendorf, W. Schoenmaker, and P. Reynier, "Coupled circuit/em simulation for radio frequency circuits," in *2017 54th Annual Design Automation Conference (DAC)*, 2017, pp. 1–6.

[11] L. Jiang, H. Yao, H. Zhang, and Y. Qin, "Machine learning based computational electromagnetic analysis for electromagnetic compatibility," in *IEEE International Conference on Computational Electromagnetics (ICCEM)*, 2018, pp. 1–2.

[12] S. Qi, Y. Wang, Y. Li, X. Wu, Q. Ren, and Y. Ren, "Two-dimensional electromagnetic solver based on deep learning technique," *IEEE Journal on Multiscale and Multiphysics Computational Techniques*, vol. 5, pp. 83–88, 2020.

[13] R. Shibata, M. Ohira, and Z. Ma, "A novel convolutional-autoencoder based surrogate model for fast s-parameter calculation of planar bpfs," in *IEEE/MTT-S International Microwave Symposium-IMS*, 2022, pp. 498–501.

[14] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," *arXiv preprint arXiv:2010.08895*, 2020.

[15] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis, "Learning nonlinear operators via deeponet based on the universal approximation theorem of operators," *Nature machine intelligence*, vol. 3, no. 3, pp. 218–229, 2021.

[16] A. Khan and D. A. Lowther, "Physics informed neural networks for electromagnetic analysis," *IEEE Transactions on Magnetics*, vol. 58, no. 9, pp. 1–4, 2022.

[17] R. Bischof and M. Kraus, "Multi-objective loss balancing for physics-informed deep learning," *arXiv preprint arXiv:2110.09813*, 2021.

[18] X. Xuan, M. Yang, and Z. Jiang, "Extraction of multimode s-parameters by using a hybrid fdtd/svd technique," *IEEE Transactions on Antennas and Propagation*, vol. 67, no. 3, pp. 2012–2016, 2018.

[19] M. Elhamod, J. Bu, C. Singh, M. Redell, A. Ghosh, V. Podolskiy, W.-C. Lee, and A. Karpatne, "Cophy-pgnn: Learning physics-guided neural networks with competing loss functions for solving eigenvalue problems," *ACM Transactions on Intelligent Systems and Technology*, vol. 13, no. 6, pp. 1–23, 2022.

[20] Z. Xiang, W. Peng, X. Zheng, X. Zhao, and W. Yao, "Self-adaptive loss balanced physics-informed neural networks for the incompressible navier-stokes equations," *arXiv preprint arXiv:2104.06217*, 2021.

[21] S. Wang, Y. Teng, and P. Perdikaris, "Understanding and mitigating gradient flow pathologies in physics-informed neural networks," *SIAM Journal on Scientific Computing*, vol. 43, no. 5, pp. A3055–A3081, 2021.

[22] A. A. Heydari, C. A. Thompson, and A. Mehmood, "Softadapt: Techniques for adaptive loss weighting of neural networks with multi-part loss functions," *arXiv preprint arXiv:1912.12355*, 2019.

[23] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, "Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks," in *International conference on machine learning*, 2018, pp. 794–803.

[24] J. B. Schneider, "Understanding the finite-difference time-domain method," 2010.

[25] Z. Gao, C. Tan, L. Wu, and S. Z. Li, "Simvp: Simpler yet better video prediction," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 3170–3180.

[26] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer Assisted Intervention*, 2015, pp. 234–241.

[27] O. B. Kobe, J. Chuma, R. Jamisola Jr, and M. Chose, "A review on quality factor enhanced on-chip microwave planar resonators," *Engineering Science and Technology, an International Journal*, vol. 20, no. 2, pp. 460–466, 2017.

[28] J. Gu, Z. Gao, C. Feng, H. Zhu, R. Chen, D. Boning, and D. Pan, "Neurolight: A physics-agnostic neural operator enabling parametric photonic device simulation," *Advances in Neural Information Processing Systems*, vol. 35, pp. 14 623–14 636, 2022.

[29] M. A. Rahman, Z. E. Ross, and K. Azizzadenesheli, "U-no: U-shaped neural operators," *arXiv preprint arXiv:2204.11127*, 2022.

[30] A. Tran, A. Mathews, L. Xie, and C. S. Ong, "Factorized fourier neural operators," *arXiv preprint arXiv:2111.13802*, 2021.

[31] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[32] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.