Physics-informed Supervised Residual Learning for 2D Electromagnetic Forward Modeling

Tao Shan, Xiaqian Song, Rui Guo, Maokun Li, Senior Member, IEEE, Fan Yang, Fellow, IEEE, Shenheng Xu, Member, IEEE

Abstract—In this paper, we propose the physics-informed supervised residual learning (PISR) which is a general framework for electromagnetic forward modeling based on deep learning. PISR is designed to solve a system of linear matrix equations and not limited to a specific electromagnetic problem. Stationary and non-stationary iterative PISR neural networks (SIPISRLNN and NSIPISRLNN) are designed based on the mathematical connection between residual neural network (ResNet) and stationary or non-stationary iterative methods. With convolutional neural network mapping residuals and modifications, SIPISRLNN and NSIPISRLNN can improve the approximated solutions via iterative process, which mimics the procedure of solving linear equations by stationary and non-stationary iterative methods. The generalities of SIPISRLNN and NSIPISRLNN are validated by solving different types of volume integral equations (VIEs) with non-lossy and lossy scatterers. In numerical results of non-lossy scatterers, the mean squared errors (MSEs) of SIPISRLNN and NSIPISRLNN finally converge below $3.152 \times 10^{-4}$ and $4.8925 \times 10^{-7}$; in results of lossy scatterers, the MSEs converge below $1.2775 \times 10^{-4}$ and $1.567 \times 10^{-7}$. The generalization abilities of SIPISRLNN and NSIPISRLNN are verified by testing them on the data sets of contrast shapes and different incident frequencies that are unseen during the training process.

Keywords—Physics-informed Supervised Residual Learning, Volume Integral Equations, Residual Neural Network, Stationary and non-stationary Iterative Method.

I. INTRODUCTION

Computational electromagnetics (CEM) focuses on numerical algorithms of solving Maxwell’s equations via computer techniques. Electromagnetic computational algorithms have been widely applied in scientific simulation, engineering design, and information processing. Typical electromagnetic computational algorithms include finite difference method (FDM), finite element method (FEM) and Method of Moments (MoM). The common approaches of these methods are to solve Maxwell’s equations by discretizing and converting them into linear matrix equations. Such linear matrix equations usually have a large number of unknowns which consume lots of computing resources to solve. Many application scenarios require real-time and accurate computational electromagnetic simulations, such as biomedical imaging and target detection. Computational efficiency has been a long-standing challenge in developing electromagnetic computational methods.

One method of acceleration is to reduce the computational complexity based on the physical laws, such as conjugate gradient-fast Fourier transform, adaptive integral method, multilevel fast multipole algorithm, etc. Another method is to divide the whole computation into offline and online parts, such as reduced basis method, characteristic basis function and model-order reduction method. The online computation can be accelerated by pre-computing partial models in the offline process. Similarly, machine learning (ML) techniques are applied to accelerate the online computation via offline training process. Artificial neural network, self-organizing map, radial basis function network are incorporated into FEM and FDTD in order to improve the computational efficiency. ML techniques are also applied into the microwave circuit aided design, antenna optimizations, microwave imaging, nondestructive testing, direction of arrival estimation, etc.

With the development of high-performance computing platform, deep learning (DL) is ushering in leapfrog development and has been widely applied in image, video and speech processing. DL promotes the development of fast computational simulations in the field of physics modeling. Pixel-based and object-based deep learning methods are applied into intuitive physics by learning the relationships between objects and predicting the future physical states. DL techniques are applied to accelerate the computational fluid, such as smoke simulation, steady fluid prediction, super resolution fluid flow, etc. In order to improve the generalization ability, DL techniques aim to learn the exact physical laws instead of approximating inner law from the data set, such as Hamiltonian neural network motivated by Hamiltonian mechanics. Symplectic recurrent neural network with Symplectic integration and Lagrangian neural network for simulating Lagrangians. DL are also applied to accelerate the electromagnetic computational simulations by replacing the parts of high computational complexity.

In electromagnetic inverse scattering problems, DL-based methods can improve the resolution of inversions and speed up the computing time. Applications of DL in the antenna engineering have been reported, such as phase synthesis of reflectarrays, coding programmable metasurfaces and metasurface imager.

Recently, various works have been reported to investigate the relationship between DL and partial differential equations (PDEs). DL can help reduce the curse of dimensionality in the numerical algorithms of PDEs, such as Schrödinger...
equation [46], Navier-stokes equation [47], Burger’s equation [48], Poisson’s equation [49], Heat equation [50], Advection-diffusion equation [51], etc. To improve the performance and generalization ability, DL are combined with traditional numerical algorithms of PDEs, such as variational method [52], multi-grid method [53], hierarchical matrices [54], weak solutions [55] and solution ansatz [56]. PDEs can be embedded into the loss of neural network and then solved via the automatic differentiation of DL, such as physics-informed neural network [57], DeepXDE [58]. Deep neural networks can be interpreted as the PDEs and PDEs’ numerical algorithms [59]. Such interpretations can further guide the design of deep neural networks. In [60] similarities between the differential operators and convolutions are investigated. The parabolic or hyperbolic equations are embedded into the deep neural networks [61]. Deep neural networks are designed based on the multi-grid method [62] and runge-kutta method [63].

In this paper, we propose the physics-informed supervised residual learning (PISRL) as a general framework for 2D electromagnetic forward modeling. PISRL is proposed based on the similarities between ResNet and stationary or non-stationary iterative methods. Then, stationary and non-stationary iterative PISRL neural networks (SIPISRLNN and NSIPISRLNN) are proposed by embedding stationary and non-stationary iterative methods into ResNets. This work is also an example that deep neural network is designed and interpreted from the perspective of traditional numerical algorithms. SIPISRLNN and NSIPISRLNN mimic the iterative process of non-stationary and stationary iterative methods with the same calculation of residuals and addition of modifications [64]. The only difference is that convolutional neural networks (CNNs) map between residuals and modifications of the solutions of linear matrix equations in SIPISRLNN and NSIPISRLNN. It is noted that PISRL can be easily expanded to solve different electromagnetic problems because it aims to solve a system of linear matrix equations instead of a specific electromagnetic problem. The generalities of SIPISRLNN and NSIPISRLNN are validated by solving volume integral equations of non-lossy and lossy scatterers. Trained by the data set generated by MoM, both SIPISRLNN and NSIPISRLNN can achieve good computing precision. In numerical results of non-lossy scatterers, the MSEs of SIPISRLNN and NSIPISRLNN finally converge below $3.152 \times 10^{-4}$ and $4.8925 \times 10^{-7}$; in results of lossy scatterers, the MSEs converge below $1.2775 \times 10^{-4}$ and $1.567 \times 10^{-7}$.

Furthermore, the generalization abilities of SIPISRLNN and NSIPISRLNN are verified on the data sets of contrast shapes and incident frequencies that are unseen in the training procedure.

This paper is organized as follows. Section II investigates the mathematical connections between ResNet and iterative methods, then proposes and analyzes SIPISRLNN and NSIPISRLNN. In Section III, we introduce the volume integral equations and the solving methods including MoM, SIPISRLNN and NSIPISRLNN. In Section IV, the generalities of SIPISRLNN and NSIPISRLNN are verified by solving VIEs of non-lossy and lossy scatterers, and the generalization abilities are validated on the data sets of contrast shapes and incident frequencies that are unseen in the training process.

Fig. 1. Schematics of residual blocks. (a): the general residual block; (b): the residual block with identity mapping; (c): the proposed residual learning block.

Observations and further discussions are summarized in Section V.

II. PHYSICS-INFORMED SUPERVISED RESIDUAL LEARNING

A. Stationary and Nonstationary Iteration Method for Solving Linear Equation System

In electromagnetic forward modeling problems, Maxwell’s equations with the boundary conditions will be converted into a linear equation system:

$$A x = b$$  \hspace{1cm} (1)

If Eq. (1) is difficult to solve, we can replace $A$ with the approximated one $A^a$ to make Eq. (1) easy to solve:

$$A^a x = b$$  \hspace{1cm} (2)

Let $x^a$ and $x^*$ denote the approximated and true solution respectively, then the difference between $x^a$ and $x^*$ satisfies the following relationship:

$$A(x^* - x^a) = b - A x^a$$  \hspace{1cm} (3)

With $\Delta x$ denoting the error $x^* - x^a$, Eq. (3) can be written as:

$$A \Delta x = b - A x^a$$  \hspace{1cm} (4)

If the error $\Delta x$ meets the stop criterion, $x^a$ can be accepted as the solution of Eq. (1). Otherwise, it needs to be modified by adding $\Delta x$:

$$x^{a}_{modified} = x^a + \Delta x$$  \hspace{1cm} (5)

The $\Delta x$ can be obtained by solving Eq. (4). If Eq. (4) is hard to solve, then the approximated $\Delta x^a$ can be obtained by solving the approximated one:

$$A^a \Delta x^a = b - A x^a$$  \hspace{1cm} (6)

The modified approximated solution can be written as:

$$x^{a}_{modified} = x^a + A^{-1} (b - A x^a)$$  \hspace{1cm} (7)

If $x^{a}_{modified}$ still cannot meet the stop criterion, the above process can be repeated to obtain the satisfied solutions.
Assuming the above process is repeated \( K \) times, a sequence of approximation solutions can be obtained:

\[
x_i^1, x_i^2, x_i^3, \ldots, x_i^K
\]

The update equation of \( x_i^K \) can be written as:

\[
x_{i+1}^a = x_i^a + A^{-1}(b - Ax_i^a) \tag{9}
\]

If we use an operator \( L \) to replace \( A^{-1} \) in \( Eq. (9) \), the equation can be written as:

\[
x_{i+1}^a = x_i^a + \sum_{j=1}^{K-1} L(b - Ax_i^a, A^{-1}) \tag{10}
\]

By recursively adding the former \( K - 1 \)-th iteration, we can get

\[
x_K^a = x_1^a + \sum_{j=1}^{K-1} L(b - Ax_j^a, A^{-1}) \tag{11}
\]

Eq. (10) and Eq. (11) can be regarded as the stationary iteration scheme. Note that if the array \( A \) is different at each iteration step, Eq. (10) and Eq. (11) can be transformed as the non-stationary iteration scheme:

\[
x_{i+1}^a = x_i^a + L(b - Ax_i^a, A^{-1}) \tag{12}
\]

\[
x_K^a = x_1^a + \sum_{j=1}^{K-1} L(b - Ax_j^a, A^{-1}) \tag{13}
\]

Two commonly used iteration schemes are Richardson iteration scheme and Jacobi iteration scheme. They can be written in the format of Eq. (10).

1) Richardson Iteration Scheme: The Richardson iteration scheme can be written as:

\[
x_{i+1}^h = x_i^h + \omega(b - Ax_i^h) \tag{14}
\]

where \( \omega \) is above zero.

2) Jacobi Iteration Scheme: In the Jacobi iteration scheme, the array \( A \) is split into diagonal part \( D \), upper triangle part \( L \) and lower triangle part \( U \):

\[
A = D - L - U \tag{15}
\]

Then the Jacobi iteration scheme can be written as:

\[
x_{i+1}^h = D^{-1}(L + U)x_i^h + D^{-1}b \tag{16}
\]

It can be written as:

\[
x_{i+1}^h = D^{-1}(L + U)x_i^h + D^{-1}b
= D^{-1}(L + U + D - D)x_i^h + D^{-1}b
= x_i^h + D^{-1}(b - Ax_i^h) \tag{17}
\]

B. Physics-informed Supervised Residual Learning Blocks

Residual neural networks (ResNets) have proven effective network architectures and have wide applications in image processing [1]. ResNets are stacked structures composed of a family of modular blocks and these modular blocks are named as residual blocks [1]. ResNets can be easily extended to have extremely deep structures with modular blocks. Works on the interpretation of ResNet have been reported. Single block of ResNet can be formulated as the forward Euler discretization of the ordinary differential equations (ODEs) [65], [66]. The stability and reversibility of the ResNet are also studied from the perspective of ODEs [67]. ResNet is also interpreted as a transport equation in the control problem [68] and dynamical systems [69]. Figure 1(a) shows the general structure of the residual block which can be expressed as [1]:

\[
y_k = h(x_k) + F(x_k, W_k) \tag{18}
\]

\[
x_{k+1} = N(y_k) \tag{19}
\]

where \( x_k \) and \( x_{k+1} \) are input and output of the \( k \)-th residual block; \( h \) denotes the linear projection; \( N \) denotes the nonlinear activation; \( F \) denotes the transform function of the residual and \( W_k \) is the parameter set of \( F \). In Eq. (18), if we regard \( h(x_k) \) and \( y_k \) as the predicted and desired output, then \( F(x_k, W_k) = h(x_k) - y_k \) will be the residual of \( h(x_k) \) and \( y_k \).

In [70], by taking \( h \) and \( N \) as the identity mappings, residual neural network can achieve better performance with improved generalization ability. Then Eq. (18) and Eq. (19) can be re-written as:

\[
x_{k+1} = x_k + F(x_k, W_k) \tag{20}
\]

Figure 1(b) shows the structure of a residual block with identity mappings.

Note that the residual block with identity mappings Eq. (20) is similar to the non-stationary iteration scheme Eq. (12). Motivated by such observation, we propose the physics-informed supervised residual learning (PISRL) blocks by embedding the non-stationary iteration scheme into the residual block. The PISRL block can be expressed mathematically as:

\[
x_{k+1} = x_k + F(b - Ax_k, W_k) \tag{21}
\]

Figure 1(c) illustrates the structure of the proposed PISRL block. With the input \( x_k \), the residual is calculated first by:

\[
R_k = b - Ax_k \tag{22}
\]

where the \( A \) and \( b \) come from the linear equation system Eq. (1). Then the calculated residual \( R_k \) goes through the convolution neural network (CNN) and the structure of CNN is determined by the complexity of the linear equation system. The CNN takes the current residual \( R_k \) as input and then outputs the corresponding modification \( \Delta x_k \) that is added to \( x_k \).

If the CNN of PISRL block is the same in every step, then the PISRL block can be regarded as embedding the stationary iteration scheme Eq. (19):

\[
x_{k+1} = x_k + F(b - Ax_k, W) \tag{23}
\]

C. Stationary Iterative Physics-informed Supervised Residual Learning Neural Network

Based on the PISRL blocks embedding the stationary iteration scheme, we propose one general architecture of physics-informed supervised residual learning neural network (PISRLNN): stationary iterative physics-informed supervised residual learning neural network (SIPISRLNN).
Fig. 2. Schematics of stationary iterative physics-informed supervised residual learning neural network. The number of channels is denoted in the pictures.

Figure 2 shows the schematics of SIPISRLNN. CNNs of PISRL blocks in each iteration have the same structure and share the same parameter set. In each iteration, the mapping between the input and corresponding output is totally different and the shared CNN needs to make reliable predictions. Therefore, the CNN should have powerful learning ability. In this paper, U-net [71] is adopted as the CNN structure in the PISRL blocks, as illustrated in Figure 2.

U-net is an effective network with good learning ability and has achieved good performance in image segmentation [71]. U-net uses $3 \times 3$ convolutional kernels, $2 \times 2$ max pooling, bilinear upsampling, Tanh nonlinear activation and batch normalization throughout the architecture. U-net has a symmetric path consisting of a contracting path and an expansive path [71]. In the contracting path, U-net extracts the feature maps by $2 \times 2$ max pooling the input and then upsamples the concatenation of high-level and low-level feature maps by bilinear upsampling in the expansive path. In the last two layers, batch normalization is not included because the last two layers aims to reduce the channels of feature maps.

Note that the input and output of U-net have the same size due to the fully convolutional neural network structure. The input of the U-net is the concatenation of the residual $b - A x_k$ and the previous output $x_k$. $x_k$ functions as the hidden state of recurrent neural networks (RNNs). $x_k$ can offer more information and the process of iterations. The channels of input and output are denoted as $2c$ and $c$ depends on the linear equation systems, for example, $c$ will be 1 in the real-valued linear equations and $c$ will be 2 in the complex-valued linear equations.

D. Non-stationary Iterative Physics-informed Supervised Residual Learning Neural Network

With the PISRL blocks embedding the non-stationary iteration scheme, another general architecture of physics-informed supervised residual learning neural network is proposed: non-stationary iterative physics-informed supervised residual learning neural network (NSIPISRLNN).

Figure 3 shows the schematics of non-stationary PISRLNN and it can be regarded as the unrolled structure of the SIPISRLNN. The NSIPISRLNN have 7 blocks with the same structure. The CNNs in each PISRL block have the same structure but different parameter sets. In each iteration, independent CNNs can make independent mappings of different
pairs of residuals and modifications. Thus, the CNNs need no complicated structures. In this paper, the CNN consists of five stacked modules of $3 \times 3$ convolutional kernels, batch normalization and Tanh nonlinear activation, as shown in Figure 3. The output has the same size of the input due to the fully convolutional operations. The channels $c$ of input and output depend on the electrical properties of dielectric scatter in free space and will be 1 in real-valued ones and 2 in complex-valued ones.

E. Analysis of Physics-informed Supervised Residual Learning Neural Network

In the SIPISRLNN, the formulation of one block can be written as:

$$x_{L+1}^0 = x_1^0 + \mathcal{F}(b - A x_1^0, W)$$

By summing all output of the former $L-1$ iterations, Eq. (24) can be written as:

$$x_L^0 = x_1^0 + \sum_{i=1}^{L-1} \mathcal{F}(b - A x_i^0, W)$$

If the loss function $\mathcal{E}$ is defined as the mean squared error (MSE), then $\mathcal{E}$ can be written as:

$$\mathcal{E} = \frac{1}{N} ||x_L^0 - x^*||_p^2$$

$$= \frac{1}{N} ||x_1^0 + \sum_{i=1}^{L-1} \mathcal{F}(b - A x_i^0, W) - x^*||_p^2$$

where the $N$ is the number of elements in the $x^*$. Then, based on the chain rule, the back propagation of $\mathcal{E}$ with respect to the $l$-th input can be written as [70]:

$$\frac{\partial \mathcal{E}}{\partial x_l} = \frac{\partial \mathcal{E}}{\partial x_L} \frac{\partial x_L}{\partial x_l} = \frac{\partial \mathcal{E}}{\partial x_L} (1 + \sum_{i=1}^{L-1} \mathcal{F}(b - A x_i^0, W))$$

Eq. (27) shows that the gradient of $\mathcal{E}$ with respect to $x_i$ is unlikely to be zero because $1 + \sum_{i=1}^{L-1} \mathcal{F}(b - A x_i^0, W)$ cannot be always zero [70]. This can prevent the problem of gradient vanishing. Similarly, we can obtain the same conclusions of NSIPISRLNN, of which the block can be written mathematically:

$$x_L^0 = x_1^0 + \sum_{i=1}^{L-1} \mathcal{F}(b - A x_i^0, W)$$

III. VOLUME INTEGRAL EQUATION

Volume integral equation (VIE) describes the scattered field of the dielectric scatter in free space $D$, as shown in Figure 2. The electrical properties of dielectric scatterers only change along the lateral axes. Assuming the permeability of scatterers as vacuum permittivity $\mu_0$, the permittivity can be written as:

$$\varepsilon(r) = \varepsilon_0 \varepsilon_r(r) - j \frac{\sigma(r)}{\omega}$$

where $\varepsilon_0$, $\varepsilon_r(r)$, $\sigma(r)$, $\omega$, $r$ are vacuum permittivity, relative permittivity, conductivity, angular frequency and the position vector in free space $D$ respectively.

The total electric field $E^{tot}(r)$ will be excited when the dielectric scatterers are illuminated by the incident field $E^{inc}(r)$.

Taking transverse magnetic (TM) mode into account, the relationship between $E^{tot}(r)$ and $E^{inc}(r)$ satisfies the electric field integral equation (EFIE):

$$E^{tot}(r) = E^{inc}(r) + k_0^2 \int_D G_D(r, r') \chi(r') E^{tot}(r') \, dr' \quad r \in D$$

where $k_0$ is the wavenumber: $k_0^2 = \omega \mu_0 \varepsilon_0$, $G_D$ is Green’s function in 2D free space:

$$G_D(r, r') = \frac{1}{4j} H^{(2)}_0(k_0|r - r'|)$$

and $\chi(r)$ is the contrast:

$$\chi(r) = \frac{\varepsilon(r) - \varepsilon_0}{\varepsilon_0} = \varepsilon_r(r) - 1 - j \frac{\sigma(r)}{\omega \varepsilon_0}$$

The incident field will induce the current on the scatterers and the induced current can be described as:

$$J(r) = \chi(r) E^{tot}(r)$$

Then the scattered field is radiated from the induced current, which can be calculated by:

$$E^{sca}(r') = \int_D G_S(r', r'') \chi(r') E^{tot}(r'') \, dr' \quad r' \in S$$

where $r''$ is the distance between the scatterers and the receivers.

A. Method of Moments

To determine the scattered field, Eq. (34) needs to be solved numerically. Method of moments (MoM) is applied to solve Eq. (34) in this work. The domain $D$ is discretized with $M \times N$ subdomains. By applying the pulse basis function and delta function, the EFIE Eq. (34) can be discretized as:

$$E^{tot}(r_p) = E^{inc}(r_p) + k_0^2 \sum_{q=1}^{M \times N} \chi(q) E^{tot}(q) \int_{D_q} H^{(2)}_0(k_0|r_p - q'|) \, dq'$$

where $p$ and $q$ are the index of the subdomain. By simplifying the notations, Eq. (35) can be written as:

$$E^{tot}_p = E^{inc}_p + \left(-j k_0^2 \right) \sum_{q=1}^{M \times N} \chi(q) E^{tot}_q \int_{D_q} H^{(2)}_0(k_0|r_p - q'|) \, dq'$$

Fig. 4. The model setup for volume integral equation. Green triangles and yellow squares denote the receivers and transmitters.
For each subdomain, the Eq. (36) is satisfied and then $M \times N$ equations can be obtained. These equations can be written in the format of matrix equations:

$$E^{tot} = E^{inc} + G_D \cdot \chi \cdot E^{tot}$$  \hspace{1cm} (37)

Then Eq. (37) can be converted as a linear equation system:

$$(I - G_D \cdot \chi) \cdot E^{tot} = E^{inc}$$  \hspace{1cm} (38)

where $G_D$ is a $MN \times MN$ array and can be determined by:

$$G_{D, pq} = -\frac{jk^2}{4} \int_{D_q} H_0^{(2)}(k_b|r_p - r'_q|) \, dr'_q$$

$$= \begin{cases} \frac{j k_b a_p H_1^{(2)}(k_b a_p) - 2 j}{2}, & p = q \\ \frac{j k_b a_p H_1^{(2)}(k_b a_p)}{2}, & else \end{cases}$$  \hspace{1cm} (39)

where $a_p$ is the radius of the equivalent circle with the same area $S_p$, which can be calculated as:

$$a_p = \frac{\sqrt{S_p}}{\pi}$$  \hspace{1cm} (40)

B. SIPISRLNN

The SIPISRLNN is applied to solve Eq. (38) that is the discretized form of the EFIE (Eq. (34)). By solving Eq. (38) based on the stationary iteration method, according to Eq. (40), the iteration equation can be written as:

$$E_{k+1}^{tot} = E_k^{tot} + \mathcal{L}(E^{inc} - (I - G_D \cdot \chi) \cdot E_k^{tot}, (I - G_D \cdot \chi)^a k^{-1})$$  \hspace{1cm} (41)

where $(I - G_D \cdot \chi)^a k$ is the approximation array of $(I - G_D \cdot \chi)$ and $\mathcal{L}$ is the first-order non-stationary iteration method. Then similar to Eq. (23), the iteration equation of the $k$-th stationary PISRL block can be written as:

$$E_{k+1}^{tot} = E_k^{tot} + \mathcal{F}(E^{inc} - (I - G_D \cdot \chi) \cdot E_k^{tot}, E_k^{tot}, \mathcal{W})$$  \hspace{1cm} (42)

where $\mathcal{F}$ denotes U-Net and $\mathcal{W}$ is the parameters of U-Net. Note that the $\mathcal{W}$ is the same and shared in all iterations due to the stationary iteration scheme. The $k$-th residual in the $k$-th PISRL block can be defined as:

$$\mathbb{R}_k = E^{inc} - (I - G_D \cdot \chi) \cdot E_k^{tot}$$  \hspace{1cm} (43)

The Green’s function $G_D$ is assumed the same in this paper due to the uniform background. $\chi$ and $E^{inc}$ are different in different numerical samples. $G_D, \chi$ and $E^{inc}$ are included in the SIPISRLNN to calculate the $\mathbb{R}_k$ in Eq. (43). The objective function of the SIPISRLNN is the mean squared error (MSE), which can be written as:

$$\mathcal{E} = \frac{1}{N} \| E_{k+1}^{tot} - E_k^{tot} \|_F^2$$  \hspace{1cm} (44)

where $E_{k+1}^{tot}$ is the total field calculated by SIPISRLNN, $E_k^{tot}$ is the total field calculated by MoM, $\| \cdot \|_F$ is the $F$ norm of a matrix and $N$ is the number of elements in a matrix.
270°, 315°, as shown in Figure 5. The contrast of non-lossy scatterers in the domain $D$ and can be derived from Eq. (32):

$$\chi(r) = \frac{\varepsilon(r) - \varepsilon_0}{\varepsilon_0} = \varepsilon_r(r) - 1$$ \hspace{1cm} (48)

The domain $D$ is assumed to have three cylinders which have random positions and radii. The real parts of the three cylinders' contrast vary from 0 to 1, 1 to 2 and 0 to 2 with their imaginary parts are 0, as shown in Figure 6.

Both SIPISRLNN and NSIPISRLNN are implemented in Pytorch and computed on one Nvidia V100 GPU. The parameters of SIPISRLNN and NSIPISRLNN are optimized by Adam [72]. The learning rate is initialized as 0.002 and decayed by $\times 0.8$ every 20 epochs. We use MoM to generate 40000 data samples for SIPISRLNN and NSIPISRLNN of which 32000 are for training and 8000 for testing.

1) SIPISRLNN: SIPISRLNN is trained to solve the VIEs of non-lossy scatterers and SIPISRLNN is assumed to have three iterations. Figure 7 shows the convergence curve of MSE and MSE finally achieves below $3.152 \times 10^{-4}$. The training MSE and testing MSE agree well with each other and little overfitting exists. U-Net of SIPISRLNN is the same and shared in all iterations. That means U-Net needs to learn the various and complicated mappings between residuals and modifications in all iterations. Thus, the optimization of U-Net is not stable enough and it can be observed that the convergence curve of MSE fluctuates several times during the training process.

Figure 8 illustrates the comparisons of total field computed by SIPISRLNN ($E_{\text{tot}}^{SIPISRLNN}$) and MoM ($E_{\text{tot}}^{MoM}$) and they are randomly selected from the testing data set. $E_{\text{tot}}^{SIPISRLNN}$ and $E_{\text{tot}}^{MoM}$ are in a good agreement and only some points in the absolute error distribution are not good enough. For better interpretation of SIPISRLNN, we print and show the updated total fields in each iteration of SIPISRLNN, as shown in Figure 9. With the incident field $E_{\text{inc}}$ as input, the difference between the updated $E_{\text{tot}}^{SIPISRLNN}$ and $E_{\text{tot}}^{MoM}$ becomes smaller and the corresponding MSE also decreases. Figure 10 shows the histogram of mean absolute error (MAE) of $E_{\text{tot}}^{SIPISRLNN}$ in the training and testing data sets. The MAE’s mean and standard deviation (std) of the training and testing data sets agree well with each other. This observation
Fig. 10. Non-lossy scatterers: histograms of mean absolute error of $E_{\text{tot}}^{\text{SIPISRLNN}}$. (a) and (b) are real and imaginary parts of $E_{\text{tot}}^{\text{SIPISRLNN}}$ in the training data set; (c) and (d) are real and imaginary parts of $E_{\text{tot}}^{\text{SIPISRLNN}}$ in the testing data set.

Fig. 11. Non-lossy scatterers: MSE convergence curve of NSIPISRLNN.

Fig. 12. Non-lossy scatterers: comparisons of total field computed by NSIPISRLNN and MoM. From left to right: contrast $\chi$, input of NSIPISRLNN (iteration initial value, $E_{\text{inc}}$), total field computed by MoM $E_{\text{tot}}^{\text{MoM}}$, total field computed by NSIPISRLNN $E_{\text{tot}}^{\text{NSIPISRLNN}}$, and their absolute error distribution. The first row is the real parts and the second is the imaginary parts.

Fig. 13. Non-lossy scatterers: updated total fields in each iteration computed by NSIPISRLNN. From left to right: contrast $\chi$, updated total field computed by MoM $E_{\text{tot}}^{\text{MoM}}$, input of NSIPISRLNN (initial value of iteration, $E_{\text{inc}}$), total fields computed by NSIPISRLNN $E_{\text{tot}}^{\text{NSIPISRLNN}}$ in the first, third, fifth, sixth and seventh iteration. The first row is the real parts and the second is the imaginary parts.

is consistent with the MSE convergence curve, as shown in Figure 7.

2) NSIPISRLNN: NSIPISRLNN consists of seven PISRL blocks and can be regarded as the unrolled structure of SIPISRLNN. The convergence curve of MSE during the training process is shown in Figure 11. The training MSE agrees well with testing MSE. Both of them decrease steadily and converge below $4.8925 \times 10^{-7}$. Compared to the converged MSE $3.152 \times 10^{-4}$ of SIPISRLNN, NSIPISRLNN have better computational precision. NSIPISRLNN has more iterations (7 PISRL blocks) but simpler structure of CNN. It still indicates that it is more reasonable to use an independent CNN to map residuals and modifications in each iteration, and better precision can be achieved.

Figure 12 shows three examples of total fields solved by NSIPISRLNN, which are randomly chosen from the testing data set. Total fields computed by MoM and NSIPISRLNN agree well with each other and the corresponding absolute error is at a very small level. Figure 13 illustrates the updated total field in the iterative process with incident field as input. The updated total field in the first iteration is the coarse-grained approximation of true total field, then it improves and its MSE decreases with the increase of iterations. Figure 14 offers a whole overview of NSIPISRLNN’s performance by showing the MAE histogram of training and testing data sets. The mean and std are in a good agreement and it verifies the...
Fig. 14. Non-lossy scatterers: histograms of mean absolute error of $E_{\text{NSIPISRLNN}}$. (a) and (b) are real and imaginary parts of $E_{\text{NSIPISRLNN}}$ in the training data set; (c) and (d) are real and imaginary parts of $E_{\text{NSIPISRLNN}}$ in the testing data set.

Fig. 15. Examples of the generalization validation data set for unseen contrast shapes.

MSE convergence curve, as shown in the Figure 11. Small values of std reveal that the range of MAE is small and concentrated. The computational precision of NSIPISRLNN is better and more stable.

3) Generalization Ability on Contrast Shape: The generalization ability of SIPISRLNN and NSIPISRLNN on unseen contrast shapes is considered. There are 8 types of unseen contrast shapes, including polygon, six-pointed star, five-pointed star, heart shape and the letters T, H, U, EE, as shown in Figure 15. We generate 40 samples for each type of unseen contrast shapes and the range of contrast is in $[0, 2]$. The generalization validation data set has 320 data samples ($40 \times 8$). Figure 16 shows the MAE histograms of total field computed by SIPISRLNN and NSIPISRLNN on the contrast generalization validation data set. It can be observed that the MAEs of contrast generalization validation data set are at the same level of error as ones of training and testing data sets. The generalization abilities of SIPISRLNN and NSIPISRLNN on unseen contrast shapes are verified. Figure 17 and Figure 18 show the randomly chosen results of SIPISRLNN and NSIPISRLNN on the data set of unseen contrast shapes.

Fig. 16. Non-lossy scatterers: MAE histograms of SIPISRLNN’s and NSIPISRLNN’s results on generalization validation data set for unseen contrast shapes. (a) and (b) are histograms of real parts and imaginary parts of total fields computed by SIPISRLNN. (c) and (d) are histograms of real parts and imaginary parts of total fields computed by NSIPISRLNN.

Fig. 17. Non-lossy scatterers: results of SIPISRLNN on the unseen contrast shape generalization validation data set.

Fig. 18. Non-lossy scatterers: results of NSIPISRLNN on the unseen contrast shape generalization validation data set.

4) Generalization Ability on Incident Frequency: The generalization ability of SIPISRL and NSIPISRL on the frequencies of incident field is also considered. Here, we consider 12 frequencies of incident field: 2GHz, 2.3GHz, 2.6GHz, 2.9GHz, 3.1GHz, 3.2GHz, 3.4GHz, 3.7GHz, 4GHz, 4.5GHz, 5GHz. For each incident frequency, we generate 320 samples as the data set and the shapes and values of contrast are the same as the contrast generalization validation data set, as shown in Figure 15. Figure 19 shows the SIPISRLNN’s and NSIPISRLNN’s MSEs of different incident frequencies. Both SIPISRLNN and NSIPISRLNN behaved well from 2GHz to 3.2GHz with stable MSEs. When the incident frequency is higher than 3.2GHz, obvious increases in MSE is observed. It indicates that SIPISRLNN and NSIPISRLNN has good generalization ability on lower frequencies compared to the training incident frequency. Figure 20 and Figure 21 give detailed results of SIPISRLNN and NSIPISRLNN with respect to different incident frequencies.

B. Lossy scatterers

In this section, lossy scatterers in domain $D$ are taken into account and the contrast can be calculated using Eq. (32). There are four cylinders in domain $D$ and they have random positions and radii. The ranges of real parts and imaginary parts of contrast can refer to Table I. Figure 22 shows six examples of contrast distributions. The frequency of incident
field is also fixed at 3GHz and the directions of incident field are randomly selected in $[0^\circ, 90^\circ, 180^\circ, 270^\circ]$, as shown in Figure 23.

SIPISRLNN and NSIPISRLNN are implemented in Pytorch and the computing platform is one Nvidia V100 GPU. Adam [72] is the optimization algorithm of SIPISRLNN and NSIPISRLNN. The learning rate is initialized as 0.002 and multiplied by 0.8 every 20 epochs. MoM is used to generate 40000 data samples for SIPISRLNN and NSIPISRLNN of which 80% are for training and 20% for testing. Note that Eq. (38) can be regarded as a different type of equations compared to non-lossy ones due to the imaginary parts of contrast.

**1) SIPISRLNN:** SIPISRLNN has three iterations, that is, a stationary iterative PSIRL block is iterated three times. Figure 24 shows the convergence curve of SIPISRLNN’s MSE and MSE finally converges below $1.2775 \times 10^{-4}$. The training and testing MSEs are in a good agreement. The beginning of the training process is not very stable, which can be observed from the curve of MSE.

**2) NSIPISRLNN:** NSIPISRLNN have seven iterations and each iteration have an independent PISRL blocks. The con-
Fig. 21. Non-lossy scatterers: results of NSIPISRLNN on the frequency generalization validation data set.

Fig. 22. Examples of lossy scatterers: the first row is the real part and the second is the imaginary part.

Fig. 23. Incident field of lossy scatterers: the first row is the real part and the second is the imaginary part; from left to right: the incident angle is $[0^\circ, 90^\circ, 180^\circ, 270^\circ]$.

Fig. 24. Lossy scatterers: MSE convergence curve of SIPISRLNN.

Fig. 25. Lossy scatterers: comparisons of total field computed by SIPISRLNN and MoM. From left to right: contrast $\chi$, input of SIPISRLNN (iteration initial value, $E^{\text{inc}}$), total field computed by MoM $E^{\text{tot}}_{\text{MoM}}$, total field computed by SIPISRLNN $E^{\text{tot}}_{\text{SIPISRLNN}}$, and their absolute error distribution. The first row is the real parts and the second is the imaginary parts.

Fig. 29 shows three randomly chosen results of NSIPISRLNN. The discrepancy between total fields computed by NSIPISRLNN and MoM is small, and the absolute error is also at a small level. Updated total fields during the iterative process are shown in Figure 30. The updated total fields are improved gradually with the increase of iterations. To evaluate the SIPISRLNN from a global perspective, MAEs of both training and testing data sets are charted as histograms, as

The convergence of MSE is shown in Figure 28 and the final MSE is below $1.567 \times 10^{-7}$. The training MSE is a little smaller than the testing MSE but they are still in a good agreement. The final MSE of SIPISRLNN is $1.2775 \times 10^{-4}$ and NSIPISRLNN has better performance. In the numerical results of both lossy and non-lossy scatterers, NSIPISRLNN has better computing precisions which indicates the non-stationary scheme is more suitable for PISRL.
Fig. 26. Lossy scatterers: updated total fields in each iteration computed by SIPISRLNN. From left to right: contrast $\chi$, total field computed by MoM $E_{tot}^{MoM}$, input of SIPISRLNN (iteration initial value, $E_{inc}^{SIPISRLNN}$), total fields computed by SIPISRLNN $E_{tot}^{SIPISRLNN}$ in the first iteration, second iteration and third iteration. The first row is the real parts and the second is the imaginary parts.

Fig. 27. Lossy scatterers: histograms of mean absolute error of $E_{tot}^{SIPISRLNN}$. (a) and (b) are real and imaginary parts of $E_{tot}^{SIPISRLNN}$ in the training data set; (c) and (d) are real and imaginary parts of $E_{tot}^{SIPISRLNN}$ in the testing data set.

Fig. 28. Lossy scatterers: MSE convergence curve of NSIPISRLNN.

Fig. 29. Lossy scatterers: comparisons of total field computed by NSIPISRLNN and MoM. From left to right: contrast $\chi$, input of NSIPISRLNN (iteration initial value, $E_{inc}^{NSIPISRLNN}$), total field computed by MoM $E_{tot}^{MoM}$, total field computed by NSIPISRLNN $E_{tot}^{NSIPISRLNN}$, and their absolute error distribution. The first row is the real parts and the second is the imaginary parts.

3) Generalization Ability on Contrast Shape: In the numerical results of lossy scatterers, we verify the generalization ability of SIPISRLNN and NSIPISRLNN on contrast shape that are unseen in the training process. Eight types of unseen contrast shapes are taken into account, including polygon, six-pointed star, five-pointed star, heart shape and the letters T, H, U, EE, as shown in Figure 32. The real parts and imaginary parts are independently selected in [0,2] and [-2,0] respectively. For each type of shape, 40 data samples are generated and there are 320 samples in the generalization validation data set. The MAE histograms of SIPISRLNN’s and NSIPISRLNN’s results on the contrast generalization validation data set are plotted in Figure 33. It can be observed that the means and stds of generalization data set are higher than the training and testing ones. The generalization ability of SIPISRLNN on the unseen contrast shapes is not good and NSIPISRLNN still maintains good generalization ability on the unseen contrast shapes. Figure 34 and Figure 35 illustrate the randomly chosen results of SIPISRLNN and NSIPISRLNN on the data set of unseen contrast shapes.

4) Generalization Ability on Incident Frequency: The generalization abilities of SIPISRLNN and NSIPISRLNN are also...
Fig. 30. Lossy scatterers: updated total fields in each iteration computed by NSIPISRLNN. From left to right: contrast $\chi$, total field computed by MoM $E_{\text{MoM}}^{\text{tot}}$, input of NSIPISRLNN (initial value of iteration, $E_{\text{inc}}^{\text{NSIPISRLNN}}$), total fields computed by NSIPISRLNN $E_{\text{NSIPISRLNN}}^{\text{tot}}$ in the first, third, fifth, sixth and seventh iteration. The first row is the real parts and the second is the imaginary parts.

Fig. 31. Lossy scatterers: histograms of mean absolute error of $E_{\text{NSIPISRLNN}}^{\text{tot}}$. (a) and (b) are real and imaginary parts of $E_{\text{NSIPISRLNN}}^{\text{tot}}$ in the training data set; (c) and (d) are real and imaginary parts of $E_{\text{NSIPISRLNN}}^{\text{tot}}$ in the testing data set.

Fig. 32. Examples of generalization validation data set for unseen contrast shapes.

considered in the numerical results of lossy scatterers. In this section, we consider 12 frequencies of incident field: 2GHz, 2.3GHz, 2.6GHz, 2.9GHz, 3.1GHz, 3.2GHz, 3.4GHz, 3.7GHz, 4GHz, 4.5GHz, 5GHz. 320 samples are generated by MoM for each incident frequency. The values and shapes of contrast are kept the same as the contrast generalization validation data set for every incident frequency, as shown in Figure 32. Figure 36 shows the SIPISRLNN’s and NSIPISRLNN’s MSE of different incident frequencies. SIPISRLNN maintains good levels of error from 2.8GHz to 3.4GHz but behaves not well in the other incident frequencies. The generalization ability of SIPISRLNN in the lossy numerical results is limited compared to the SIPISRLNN’s in the non-lossy numerical results. NSIPISRLNN have good computing precisions from 2.0GHz to 3.4GHz and the generalization ability of NSIPISRLNN is verified. Detailed results of SIPISRLNN and NSIPISRLNN are shown in Figure 37 and Figure 38 respectively.

NSIPISRLNN shows good generalization ability on both
Fig. 35. Lossy scatterers: results of NSIPISRLNN on the unseen contrast shape generalization validation data set.

Fig. 36. Lossy scatterers: MSE of SIPISRLNN and NSIPISRLNN at different frequency.

contrast shapes and incident frequencies that unseen during the training process in numerical cases of non-lossy and lossy scatterers. SIPISRLNN only shows good generalization ability in numerical cases of non-lossy scatterers. Furthermore, numerical results also verify that NSIPISRLNN have better computing precisions than SIPIRLNN.

V. Conclusion

In this paper, we propose the physics-informed residual learning as a general framework of 2D electromagnetic forward modeling. PISRL is designed by embedding the iterative methods for solving linear matrix equations. PISRL aims to solve a system of linear matrix equations instead of a specific electromagnetic problem. Therefore, PISRL can be easily expanded to solve different electromagnetic problems which are finally converted into a system of linear matrix equations. In this paper, the generality of PISRL is verified by solving VIEs of non-lossy and lossy scatterers. SIPISRLNN and NSIPISRLNN are proposed by embedding stationary and non-stationary iterative methods. In solving VIEs of non-lossy scatterers, the MSEs of SIPIRLNN and NSIPISRLNN can achieve below $3.152 \times 10^{-4}$ and $4.8925 \times 10^{-7}$ and in solving VIEs of lossy scatterers, the MSEs can be below $1.2775 \times 10^{-4}$ and $1.567 \times 10^{-7}$. The generalization abilities of SIPISRLNN and NSIPISRLNN are also validated on the data sets of unseen contrast shapes and incident frequencies. Numerical results indicate that NSIPISRLNN has better computing precisions and better generalization abilities due to the independent CNN at each iteration. This work is also an example to design and interpret deep neural networks from the perspective of traditional numerical algorithms. This indicates a great possibility to build fast electromagnetic solvers with strong generalization abilities by combining traditional electromagnetic computational methods with deep learning techniques.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
[2] J.-M. Jin, Theory and computation of electromagnetic fields. John Wiley & Sons, 2011.
[3] E. F. Knott, J. F. Schaeffer, and M. T. Tulley, Radar cross section. SciTech Publishing, 2004.
[4] D. Poljak, Advanced modeling in computational electromagnetic compatibility. John Wiley & Sons, 2007.
[5] J.-M. Jin and D. J. Riley, Finite element analysis of antennas and arrays. Wiley Online Library, 2009.
Fig. 38. Lossy scatterers: results of NSIPISRLNN on the frequency generalization validation data set.

[6] N. K. Nikolova, “Microwave biomedical imaging,” Wiley Encyclopedia of Electrical and Electronics Engineering, pp. 1–22, 1999.

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

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

[9] W. C. Chew, M. S. Tong, and B. Hu, Integral Equation Methods for Electromagnetic and Elastic Waves. Morgan and Claypool Publishers, 2007.

[10] T. Sarkar, E. Arvas, and S. Rao, “Application of fft and the conjugate gradient method for the solution of electromagnetic radiation from electrically large and small conducting bodies,” IEEE Transactions on Antennas and Propagation, vol. 34, no. 5, pp. 635–640, 1986.

[11] E. Bleszynski, M. Bleszynski, and T. Jaroszewicz, “Aim: Adaptive integral method for solving large-scale electromagnetic scattering and radiation problems,” Radio Science, vol. 31, no. 5, pp. 1225–1251, 1996.

[12] V. Rokhlin, “Rapid solution of integral equations of classical potential theory,” Journal of computational physics, vol. 60, no. 2, pp. 187–207, 1985.

[13] X. Dang, M. Li, F. Yang, and S. Xu, “Quasi-periodic array modeling using reduced basis method,” IEEE Antennas and Wireless Propagation Letters, vol. 16, pp. 825–828, 2017.

[14] V. Prakash, and R. Mittra, “Characteristic basis function method: A new technique for efficient solution of method of moments matrix equations,” Microwave and Optical Technology Letters, vol. 36, no. 2, pp. 95–100, 2003.

[15] W. H. Schilders, H. A. Van der Vorst, and J. Rommes, Model order reduction: theory, research aspects and applications. Springer, 2008, vol. 13.

[16] R. Chedid and N. Najjar, “Automatic finite-element mesh generation using artificial neural networks-part i: Prediction of mesh density,” IEEE Transactions on Magnetics, vol. 32, no. 5, pp. 5173–5178, 1996.

[17] D. Lowther and W. Mai, “On automatic mesh generation using kohonon maps,” IEEE transactions on magnetics, vol. 34, no. 5, pp. 3391–3394, 1998.

[18] Y. Zou, G. Lei, K. Shao, Y. Guo, J. Zhu, and X. Chen, “Hybrid approach of radial basis function and finite element method for electromagnetic problems,” IEEE Transactions on Magnetics, vol. 51, no. 3, pp. 1–4, 2015.

[19] C. Wu, A. Navarro, and J. Litva, “Combination of finite impulse response neural network technique with fddt method for simulation of electromagnetic problems,” Electronics letters, vol. 32, no. 12, pp. 1112–1113, 1996.

[20] F. Wang and Q.-J. Zhang, “Knowledge-based neural models for microwave design,” IEEE Transactions on Microwave Theory and Techniques, vol. 45, no. 12, pp. 2333–2343, 1997.

[21] R. L. Haupt, “Antenna design with a mixed integer genetic algorithm,” IEEE Transactions on Antennas and Propagation, vol. 55, no. 3, pp. 577–582, 2007.

[22] K.-C. Lee, “A neural-network-based model for 2d microwave imaging of cylinders,” International Journal of RF and Microwave Computer-Aided Engineering: Co-sponsored by the Center for Advanced Manufacturing and Packaging of Microwave, Optical, and Digital Electronics (CAMode) at the University of Colorado at Boulder, vol. 14, no. 5, pp. 398–403, 2004.

[23] A. Massa, A. Boni, and M. Donelli, “A classification approach based on svm for electromagnetic subsurface sensing,” IEEE Transactions on Geoscience and Remote Sensing, vol. 43, no. 9, pp. 2084–2090, 2005.

[24] A. H. El Zooghby, C. G. Christodoulou, and M. Georgiopoulos, “A neural network-based smart antenna for multiple source tracking,” IEEE Transactions on Antennas and Propagation, vol. 48, no. 5, pp. 768–776, 2000.

[25] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” nature, vol. 521, no. 7553, pp. 436, 2015.

[26] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” arXiv preprint arXiv:1409.1556, 2014.

[27] G. Ciaparrone, F. L. Sánchez, S. Tabik, L. Troiano, R. Tagliaferri, and F. Herrera, “Deep learning in video multi-object tracking: A survey,” Neurocomputing, vol. 381, pp. 61–88, 2020.

[28] V. Hinton, L. Deng, D. Yu, G. E. Dahl, A. . Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath et al., “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” IEEE Signal processing magazine, vol. 29, no. 6, pp. 82–97, 2012.

[29] W. Li, S. Azimi, A. Leonardis, and M. Fritz, “To fall or not to fall: A visual approach to physical stability prediction,” arXiv preprint arXiv:1604.00066, 2016.

[30] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel, “Neural relational inference for interacting systems,” in International Conference on Machine Learning. PMLR, 2018, pp. 2688–2697.

[31] R. Girdhar, L. Gustafson, A. Adcock, and L. van der Maaten, “Forward prediction for physical reasoning,” arXiv preprint arXiv:2006.10734, 2020.

[32] J. Tompson, K. Schlachter, P. Sprechmann, and K. Perlin, “Accelerating eulerian fluid simulation with convolutional networks,” in Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, 2017, pp. 3424–3433.

[33] X. Guo, W. Li, and F. Iorio, “Convolutional neural networks for steady flow approximation,” in Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD ’16. New York, NY , USA: ACM, 2016, pp. 481–490.

[34] Y. Xie, E. Franz, M. Chu, and N. Thurey, “tempogan: A temporally coherent, volumetric gan for super-resolution fluid flow,” ACM Transactions on Graphics (TOG), vol. 37, no. 4, pp. 1–15, 2018.

[35] S. Greidanus, M. Ozrama, and J. Yosinski, “Hamiltonian neural networks,” arXiv preprint arXiv:1906.01563, 2019.

[36] Z. Chen, J. Zhang, M. Arjovsky, and L. Bottou, “Symplectic recurrent neural networks,” arXiv preprint arXiv:1909.13334, 2019.

[37] M. Cranmer, S. Greidanus, S. Hoyer, P. Battaglia, D. Spergel, and S. Ho, “Lagrangian neural networks,” arXiv preprint arXiv:2003.04650, 2020.

[38] H. M. Yao and L. Jiang, “Machine-learning-based pml for the fdtd method,” IEEE Antennas and Wireless Propagation Letters, vol. 18, no. 1, pp. 192–196, 2018.

[39] J.-J. Sun, S. Sun, Y. P. Chen, L. Jiang, and J. Hu, “Machine-learning-based hybrid method for the multilevel fast multipole algorithm,” IEEE Antennas and Wireless Propagation Letters, vol. 19, no. 12, pp. 2177–2181, 2020.

[40] Z. Wei and X. Chen, “Deep-learning schemes for full-wave nonlinear inverse scattering problems,” IEEE Transactions on Geoscience and Remote Sensing, 2018.
[41] Z. Wei, D. Liu, and X. Chen, “Dominant-current deep learning scheme for electrical impedance tomography,” IEEE Transactions on Biomedical Engineering, vol. 66, no. 9, pp. 2546–2555, 2019.

[42] T. Shan, M. Li, S. Xu, and F. Yang, “Synthesis of reflectarray based on deep learning technique,” in 2018 Cross Strait Quad-Regional Radio Science and Wireless Technology Conference (CSQRWC), 2018, pp. 1–2.

[43] T. Shan, X. Pan, M. Li, S. Xu, and F. Yang, “Coding programmable metasurfaces based on deep learning techniques,” IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 10, no. 1, pp. 114–125, 2020.

[44] L. Li, H. Ruan, C. Liu, Y. Li, Y. Shuang, A. Ali, C.-W. Qiu, and T. J. Cui, “Machine-learning reprogrammable metasurface imager,” Nature Communications, vol. 10, no. 1, pp. 1–8, 2019.

[45] J. Han, A. Jentzen, and E. Weinan, “Solving high-dimensional partial differential equations using deep learning,” Proceedings of the National Academy of Sciences, vol. 115, no. 34, pp. 8505–8510, 2018.

[46] J. Hermann, Z. Schätzle, and F. Noé, “Deep-neural-network solution of the electronic schrödinger equation,” Nature Chemistry, vol. 12, no. 10, pp. 891–897, 2020.

[47] M. Raissi, A. Yazdani, and G. E. Karniadakis, “Hidden fluid mechanics: A navier-stokes informed deep learning framework for assimilating flow visualization data,” arXiv preprint arXiv:1808.04327, 2018.

[48] J. Sirignano and K. Spiliopoulos, “Dgm: A deep learning algorithm for solving partial differential equations,” Journal of computational physics, vol. 375, pp. 1339–1364, 2018.

[49] T. Shan, W. Tang, X. Dang, M. Li, F. Yang, S. Xu, and J. Wu, “Study on a fast solver for poisson’s equation based on deep learning technique,” IEEE Transactions on Antennas and Propagation, vol. 68, no. 9, pp. 6725–6733, 2020.

[50] R. Sharma, A. B. Farimani, J. Gomes, P. Eastman, and V. Pande, “Weakly-supervised deep learning of sea transport via physics informed loss,” arXiv preprint arXiv:1807.11374, 2018.

[51] G. Pang, L. Lu, and G. E. Karniadakis, “fpinns: Fractional physics-informed neural networks,” SIAM Journal on Scientific Computing, vol. 41, no. 4, pp. A2603–A2626, 2019.

[52] E. Weinan and B. Yu, “The deep ritz method: a deep learning-based numerical algorithm for solving variational problems,” Communications in Mathematics and Statistics, vol. 6, no. 1, pp. 1–12, 2018.

[53] D. Greenfeld, M. Galun, R. Basri, I. Yavneh, and R. Kimmel, “Learning to optimize multigrid pde solvers,” in International Conference on Machine Learning. PMLR, 2019, pp. 2415–2423.

[54] Y. Fan, L. Lin, L. Ying, and L. Zepeda-Nunez, “A multiscale neural network based on hierarchical matrices,” 2019.

[55] Y. Zhang, G. Bao, X. Ye, and H. Zhou, “Weak adversarial networks for high-dimensional partial differential equations,” Journal of Computational Physics, vol. 411, p. 109409, 2020.

[56] R. S. Beidokhti and A. Malek, “Solving initial-boundary value problems for systems of partial differential equations using neural networks and optimization techniques,” Journal of the Franklin Institute, vol. 346, no. 9, pp. 898–913, 2009.

[57] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations,” Journal of Computational Physics, vol. 378, pp. 668–707, 2019.

[58] L. Lu, X. Meng, Z. Mao, and G. E. Karniadakis, “Deepxde: A deep learning library for solving differential equations,” SIAM Review, vol. 63, no. 1, pp. 208–228, 2021.

[59] Y. Lu, A. Zhong, Q. Li, and B. Dong, “Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations,” in International Conference on Machine Learning. PMLR, 2018, pp. 3276–3285.

[60] Z. Long, Y. Lu, X. Ma, and B. Dong, “Pde-net: Learning pdes from data,” in International Conference on Machine Learning. PMLR, 2018, pp. 3208–3216.

[61] L. Ruthotto and E. Haber, “Deep neural networks motivated by partial differential equations,” Journal of Mathematical Imaging and Vision, pp. 1–13, 2019.

[62] J. He and J. Xu, “Mgnet: A unified framework of multigrid and convolutional neural network,” Science China mathematics, vol. 62, no. 7, pp. 1331–1354, 2019.

[63] M. Zhu, B. Chang, and C. Fu, “Convolutional neural networks combined with runge-kutta methods,” 2019.

[64] O. Axelsson, Iterative solution methods. Cambridge university press, 1996.

[65] E. Weinan, “A proposal on machine learning via dynamical systems,” Communications in Mathematics and Statistics, vol. 5, no. 1, pp. 1–11, 2017.