Qiang Ren
Yinpeng Wang
Yongzhong Li
Shutong Qi

Sophisticated Electromagnetic Forward Scattering Solver via Deep Learning

Springer

Sophisticated Electromagnetic Forward Scattering Solver via Deep Learning

Qiang Ren · Yinpeng Wang · Yongzhong Li · Shutong Qi

Sophisticated Electromagnetic Forward Scattering Solver via Deep Learning

Springer

Qiang Ren
School of Electronics and Information Engineering
Beihang University
Beijing, China

Yinpeng Wang
School of Electronics and Information Engineering
Beihang University
Beijing, China

Yongzhong Li
Edward S. Rogers Sr. Department of Electrical and Computer Engineering
University of Toronto
Toronto, ON, Canada

Shutong Qi
Edward S. Rogers Sr. Department of Electrical and Computer Engineering
University of Toronto
Toronto, ON, Canada

ISBN 978-981-16-6260-7 ISBN 978-981-16-6261-4 (eBook)
https://doi.org/10.1007/978-981-16-6261-4

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

To my parents Changxing Ren and Wenping Xu

—Qiang Ren

To my parents, to our Editor without whose never-failing sympathy and encouragement, this book would have never been finished.

—Yinpeng Wang

To my beloved family, mentors, friends who support me along the life journey. Without their accompany I could never achieve so far. Thanks to all who make the book possible.

—Yongzhong Li

To my parents, to my respected advisor, Qiang Ren and my co-authors, Yinpeng Wang and Yongzhong Li, whose dedication makes the book an amazing one.

—Shutong Qi

Preface

Electromagnetic (EM) scattering is a widely seen physical phenomenon, which results from the interaction between EM wave and an isolate object or set of objects (termed as scatterer hereafter). Numerical modeling of either static or dynamical process of scattering finds numerous applications in physical and engineering science, with practical impact on wave physics, geophysical sensing, stealth aircraft designs, biomedical imaging and so on. Conventional wisdom in helping formulate the modeling process is originated from applied mathematics view, from which a set of computational tools, such as finite element method, method of moment and finite difference method, were developed to solve integral or differential equations derived from the first principle physics of system under interest (e.g., Maxwell equations). However, all these methods face a same dilemma when they come into real computational scenarios/applications. In this case, the governing equations of problem at hand are generally discretized into a complicated matrix system which is often in high-dimensional space and characterized by millions of unknowns. Conventional tools might provide feasibility to solve, but the computational efficiency lags behind. Hours or even days of calculation prevent us from handling large-scale EM problems or applications requiring real-time response.

Deeping learning then comes at the right time into the stage of scientific computing. The direct idea behind is data-driven modeling, where the traditional numerical tools for problem solving only serve to provide sufficient data (not necessarily a complete set of solutions) from which deep learning algorithms learn the underlying dynamics and try to generalize. In essence, the function relation in any forward EM modeling problems with multiple variables can be faithfully approximated via network training, rather than time-consuming numerical integration or differentiation. At an intuitive level, this ability of function approximation originates from layers of networks consisting of up to millions of trainable weights followed by nonlinear activation. For EM scattering modeling, the training is supervised by the prepared dataset and it may take a few hours to reach the convergent stability, but as long as being well-trained, the deep learning algorithms excel in forward analysis with orders of magnitude acceleration compared to the conventional numerical methods.

Deep learning appears as a transformative method with astonishing performance in scientific computing, modeling and data analysis, but it is never an obscure and prohibitive method to be applied in different research areas where we mainly focus on EM scattering here. Instead, to get a grasp of the knowledge behind, all you need to add on top of your domain expertise (e.g., math, physics, chemistry, biology) is a systematical view into consecutive parts of data-driven modeling. **This book is here to provide comprehensive insights and practical guidance for you to build up a sophistical EM scattering solver using deep learning approach, to help you put deep learning into your daily research practice**. The intended audience includes anyone who is interested in applying machine learning into research of computational physics, especially for forward EM wave modeling.

Throughout the book, we start from the introduction to the EM wave scattering. The formulation of physics behind and conventional wisdom in problem solving are articulate. The second chapter exposes readers with the basic concepts of deep learning and principles for constructing a proper network configuration. The training and testing techniques are given in detail so that readers can expect to get hands-on experience with practical philosophy of how can machine learning be integrated into the EM research. In the following chapter, a closer look is taken into another important part of deep learning research, which is the preparation of data. The numerical method (FDFD) which we relied on to generate dataset is formally introduced in this chapter with proper depth. Practical techniques including the geometry generator and format for physical quantity are also covered. Up to this chapter, techniques behind the topics of this book are provided completely.

Then, we demonstrate the original and concrete experimental results in the following two chapters to showcase how the complete procedure can be done to build a 2D/3D EM scattering solver via deep learning. Readers are expected to enjoy a holistic view of problem solving in data-driven scheme and are encouraged to transfer the basic ideas behind into their own domain of research.

Applying deep learning into the modeling of EM wave propagation or wider range of physics is gradually going from a spark of research practice into mainstream trend of study, and it now has been called widely as scientific machine learning where mathematician, physicist, computer scientist, etc., work together to crackle down complex multiscale and multiphysics problem, across science, engineering and medicine. In this background, the authors believe all the readers will benefit from this book.

Beijing, China — Qiang Ren
Beijing, China — Yinpeng Wang
Toronto, Canada — Yongzhong Li
Toronto, Canada — Shutong Qi

Acknowledgements

The authors would like to thank collaborators Prof. Lei Kang, Prof. Sawyer D. Campbell, Prof. Douglas H. Werner and Prof. Pingjuan L. Werner from Penn State University and Prof. Yi Ren from Xidian University. The authors would like to extend the warm gratitude to Mr. Nianru Wang in Delft University and Ms. Xuan Wu in Beihang University, and funding resource from National Natural Science Foundation of China. Part of the results listed in this book has been published in several conferences and journals, and the authors would like to thank the publishers for their support.

Contents

Abbreviations

CNN	Convolutional Neural Network
DL	Deep Learning
ELU	Exponential Linear Unit
EM	Electromagnetic
FCNN	Fully Connected Neural Network
FD/BD/CD	Forward/Backward/Central Difference
FDFD	Finite Difference Frequency Domain
FEM	Finite Element Method
GAN	Generative Adversarial Network
MoM	Method of Moment
MSE	Mean Square Error
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
PEC	Perfect Electric Conductor
PReLU	Parametric Rectified Linear Unit
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
TE	Transversal Electric
TEM	Transversal Electromagnetic
TM	Transversal Magnetic

Notations

$\mathbf{E}$	Electric field intensity
$\mathbf{H}$	Magnetic field intensity
$\mathbf{D}$	Electric flux density
$\mathbf{B}$	Magnetic flux density
$\mathbf{J}$	Electric current density
ρ	Electric charge density
Q	Quantity of electric charge
$\mathbf{k}$	Wave vector
ω	Angular frequency
$\mathbf{r}$	Position vector
ε	Permittivity
μ	Permeability
σ	Electric conductivity
χ	Polarizability
$\mathbf{K}$	Surface current density
η	Wave impedance
$J_n(\cdot), j_n(\cdot)$	Bessel function, spherical Bessel function
$H_n^{(2)}(\cdot)$	Hankel function
$P_n(\cdot)$	Legendre polynomial
$\mathbf{W}$,$\mathbf{b}$	Weight matrix, bias vector
$\mathbf{A}$	Magnetic vector potential
$\mathbf{F}$	Electric vector potential
λ	Wavelength
h	Planck constant
n,N	Refractive index, complex refractive index
c	Light velocity
ν	Photon frequency
E	Photon energy
α	Learning rate
β	Decay rate
θ	Network parameters

∇	Nabla operator
Δ	Laplace operator
$\mathfrak{Re}, \mathcal{I}m$	Real part, imaginary part

Chapter 1
Introduction to Electromagnetic Problems

Before discussing deep learning techniques in electromagnetics, the rudiments of the theory behind them will be introduced first. Both the forward and inverse electromagnetic scattering problems are based on Maxwell's equations. In this section, the basic field laws, constitutive relations, and boundary conditions will be reviewed first. It is intended that the reader has a basic understanding of electromagnetic theory. An in-depth introduction can be found in [1–12]. After that, the chapter will be oriented to the concepts and models in the forward problems which are important to understand the rest part of this book. Lastly, the traditional electromagnetic algorithms mentioned in the following chapters will be discussed, which are used to generate data or verify the correctness of the simulation results.

1.1 Fundamentals of Electromagnetic Theory

1.1.1 Maxwell's Equations

In electromagnetics, the relations between the electric/magnetic fields, charges and currents are governed by physical laws which are the famous Maxwell's equations [1]. They are described by a set of coupled partial differential equations, which are the modified Ampère's law (1.1), Faraday's law (1.2), Gauss's law (1.3), and Gauss's law for magnetism (1.4). The differential form of the Maxwell's equations can be written as

$$\nabla \times \mathbf{H} = \frac{\partial \mathbf{D}}{\partial t} + \mathbf{J}, \tag{1.1}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}, \tag{1.2}$$

Q. Ren et al., *Sophisticated Electromagnetic Forward Scattering Solver via Deep Learning*, https://doi.org/10.1007/978-981-16-6261-4_1

Table 1.1 The quantities in Maxwell's equation

Symbols	Quantities	Units
H	Magnetic field intensity	V/m
B	Magnetic flux density	Wb/m^2
E	Electric field intensity	A/m
D	Electric flux density	C/m^2
J	Electric current density	A/m^2
ρ	Electric charge density	C/m^3

$$\nabla \cdot \mathbf{D} = \rho, \tag{1.3}$$

$$\nabla \cdot \mathbf{B} = 0. \tag{1.4}$$

The field quantities are functions of space and time, that is, $\mathbf{H} = \mathbf{H}(\mathbf{r}, t)$. The symbols are defined in Table 1.1.

In addition to the four Maxwell's equations, there is another equation named the continuity Eq. (1.5) that relates the electric current density and the electric charge density.

$$\nabla \cdot \mathbf{J} = -\frac{\partial \rho}{\partial t} \tag{1.5}$$

The differential form of Maxwell's equations describes the relations of the physical quantities at a point. It requires that the field vectors be single valued, bounded, continuous functions of space and time, and present continuous derivatives. Although with strict restricts, the differential form is more widely used in practical application.

The integral form of Maxwell's Eqs. (1.6–1.9) describes the relations of the physical quantities over an extended region. The equations can be written as

$$\oint \mathbf{H} \cdot \mathbf{dl} = -\frac{d}{dt} \iint \mathbf{D} \cdot \mathbf{dS} + I, \tag{1.6}$$

$$\oint \mathbf{E} \cdot \mathbf{dl} = -\frac{d}{dt} \iint \mathbf{B} \cdot \mathbf{dS}, \tag{1.7}$$

$$\iint \mathbf{D} \cdot \mathbf{dS} = Q, \tag{1.8}$$

$$\iint \mathbf{B} \cdot \mathbf{dS} = 0. \tag{1.9}$$

where Q is the electric charge contained inside the closed surface while I is the electric current encompassed in the closed path. Similarly, the continuity equation can be written as (1.10)

$$\oiint \mathbf{J} \cdot \mathbf{dS} = -\frac{dQ}{dt}. \tag{1.10}$$

Although the field quantities and their derivatives do not need to possess continuous distributions, it requires that the region to be solved possesses complete symmetry. In practical situations, the integral form has limited applications.

In many systems, the time variations are of sinusoidal form which are referred to as time-harmonic. In this scenario the transient electromagnetic field can be present easily through their complex forms [4]. For a transient electromagnetic vector $\mathbf{A}(\mathbf{r}, t)$ and its complex form $\dot{\mathbf{A}}(\mathbf{r})$ we have

$$\mathbf{A}(\mathbf{r}, t) = \mathrm{Re}\{\dot{\mathbf{A}}(r)e^{j\omega t}\} = \mathbf{A}\cos(\omega t - \mathbf{k} \cdot \mathbf{r}), \tag{1.11}$$

where

$$\dot{\mathbf{A}}(r) = \mathbf{A}e^{-j\mathbf{k}\cdot\mathbf{r}}. \tag{1.12}$$

In (1.12), $\boldsymbol{A}, \boldsymbol{k}, \omega, \boldsymbol{r}$ indicate the amplitude, the wave vector, the angular frequency, and the position vector, respectively. It can be proved that if the real part of a complex field satisfies Maxwell's equations, the complex field itself also satisfies the equations. Therefore, Maxwell's equations and the continuity equation can be rewritten as

$$\nabla \times \dot{\mathbf{H}} = j\omega\dot{\mathbf{D}} + \dot{\mathbf{J}}, \tag{1.13}$$

$$\nabla \times \dot{\mathbf{E}} = -j\omega\dot{\mathbf{B}}, \tag{1.14}$$

$$\nabla \cdot \dot{\mathbf{D}} = \dot{\rho}, \tag{1.15}$$

$$\nabla \cdot \dot{\mathbf{B}} = 0, \tag{1.16}$$

$$\nabla \cdot \dot{\mathbf{J}} = -j\omega\dot{\rho}, \tag{1.17}$$

where $[\dot{\cdot}]$ indicates the complex form of the physical quantities. In the case of time harmonic field, the partial derivative operator $\partial/\partial t$ can be replaced by $j\omega$, which greatly simplifies the calculation.

1.1.2 Constitutive Relationship

In addition to Maxwell's equations and the continuity equation, we require extra equations to specify the characteristics of the media where the field exits. We express

$\mathbf{D}, \mathbf{B}, \mathbf{J}$ in terms of $\mathbf{E}, \mathbf{H}$. Equations of the general form

$$\mathbf{D} = \mathbf{D}(\mathbf{E}, \mathbf{H}), \tag{1.18}$$

$$\mathbf{B} = \mathbf{B}(\mathbf{E}, \mathbf{H}), \tag{1.19}$$

$$\mathbf{J} = \mathbf{J}(\mathbf{E}, \mathbf{H}), \tag{1.20}$$

are called constitutive relationships. In free space, the relationships take the simple forms [1]

$$\mathbf{D} = \varepsilon_0 \mathbf{E}, \tag{1.21}$$

$$\mathbf{B} = \mu_0 \mathbf{B}, \tag{1.22}$$

$$\mathbf{J} = \mathbf{0}, \tag{1.23}$$

where ε_0 is the permittivity of vacuum and μ_0 is the permeability of vacuum.

Under certain conditions, the constitutive relationships can take simple forms. For linear isotropic medium, the constitutive equations are as

$$\mathbf{D} = \varepsilon \mathbf{E}, \tag{1.24}$$

$$\mathbf{B} = \mu \mathbf{H}, \tag{1.25}$$

$$\mathbf{J} = \sigma \mathbf{E}, \tag{1.26}$$

where ε is called the permittivity of the medium and μ is called the permeability of the medium. The parameter σ is called the conductivity of the medium.

Matter is often divided into different categories according to its values of ε, μ and σ. In our studies, we mainly discuss perfect electric dielectrics and metals. For perfect electric dielectrics, they satisfy

$$\varepsilon = \varepsilon_0 \varepsilon_r = \varepsilon_0 (1 + \chi_e), \tag{1.27}$$

$$\mu = \mu_0, \tag{1.28}$$

where ε_r, $\varepsilon_1, \mu_1, \sigma_1$ are called the relative permittivity and the polarizability. For perfect dielectrics, ε_r and χ_e are positive real numbers.

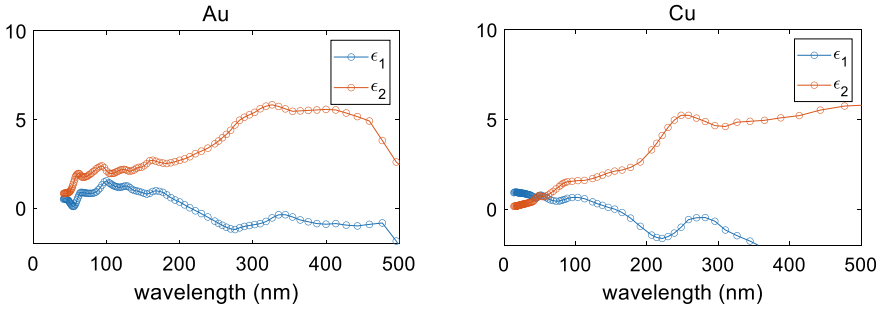


Fig. 1.1 Permittivity of two metals in Chap. 5. Au and Cu are the chemical symbols of gold and copper, respectively. ϵ_1 and ϵ_2 are the real and imaginary part of the permittivity

For metals, the permittivity is much more intricate. The parameter ε_r is generally a complex number where the real part represents the polarization, and the imaginary part represents the absorption. We usually use the free-electron gas model (also known as the Drude model) [13, 14] of metal to deduce the expression of the permittivity[1]

$$\varepsilon_r(\omega) = 1 - \frac{\omega_p^2}{\omega(\omega - j\omega_c)}, \tag{1.29}$$

where ω_p is the plasma resonance frequency and ω_c is the plasma collision frequency. The Drude model can describe the electromagnetic characteristics of metals accurately when the frequency is not higher than visible light. The fitting results are in good agreement with the experimental data. However, for electromagnetic wave in ultraviolet band, Lorentz-Drude model should be introduced to modify the model because of the forbidden transition. A simple modification is to replace "1" in (1.29) with ε_∞. It should be noted that the above correction is only for a specific narrow band (about dozens of nanometers). Figure 1.1 shows the permittivity of two metals [15, 16] selected in Chap. 5 of this book at the incident wavelength below 500 nm.

1.1.3 Boundary Condition

The differential form of Maxwell's equations requires that the field vectors be single valued, bounded, continuous functions of space and time, and present continuous derivatives. However, as the media shows discontinuities in electromagnetic properties, the field vectors are also discontinuous. In fact, the field on both sides of the

[1] In physics, we have $\varepsilon_r(\omega) = 1 - \frac{\omega_p^2}{\omega(\omega + i\omega_c)}$. In electromagnetic engineering, we usually replace i with $-j$.

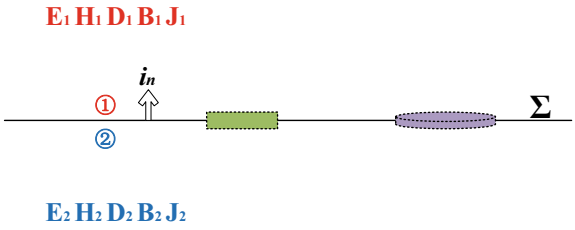


Fig. 1.2 The boundary conditions at the interface. The green rectangle represents the Stokes contour integral, and the purple cylinder represents the Gauss surface integral

interface is governed by the boundary conditions. Due to the universality of the integral form of the field laws, we can utilize them along with the Divergence theorem and Stokes theorem in vector analysis to obtain the boundary conditions. There are two tangential boundary conditions and three normal ones shown as (1.30) to (1.34).

$$\mathbf{i_n} \times (\mathbf{E_1} - \mathbf{E_2}) = \mathbf{0}, \tag{1.30}$$

$$\mathbf{i_n} \times (\mathbf{H_1} - \mathbf{H_2}) = \mathbf{K}, \tag{1.31}$$

$$\mathbf{i_n} \cdot (\mathbf{D_1} - \mathbf{D_2}) = \eta, \tag{1.32}$$

$$\mathbf{i_n} \cdot (\mathbf{B_1} - \mathbf{B_2}) = 0, \tag{1.33}$$

$$\mathbf{i_n} \cdot (\mathbf{J_1} - \mathbf{J_2}) = -\nabla_\Sigma \cdot \mathbf{K} - \frac{\partial \eta}{\partial t}. \tag{1.34}$$

Here, we suppose $\mathbf{i_n}$ is the unit normal vector pointing from region 2 to region 1, As shown in Fig. 1.2, ① and ② on both sides of the boundary are two different media, in which the field quantities are $\mathbf{E_1}$, $\mathbf{H_1}$, $\mathbf{D_1}$, $\mathbf{B_1}$, $\mathbf{J_1}$ and $\mathbf{E_2}$, $\mathbf{H_2}$, $\mathbf{D_2}$, $\mathbf{B_2}$, $\mathbf{J_2}$. The electromagnetic parameters are ε_1, μ_1, σ_1 and ε_2, μ_2, σ_2. $\mathbf{K}$, η, ∇_Σ are the surface electric current, the surface electric charge, and the surface divergence, respectively.

By adding boundary conditions to Maxwell equations and constitutive relationships, we can solve complex boundary value problems.

1.1.4 Plane Wave

The sources of the electromagnetic scattering problems discussed in this book are plane waves [17], which can be expressed as

$$\mathbf{A} = \mathbf{A_0} e^{-j\mathbf{k}\cdot\mathbf{r}}, \tag{1.35}$$

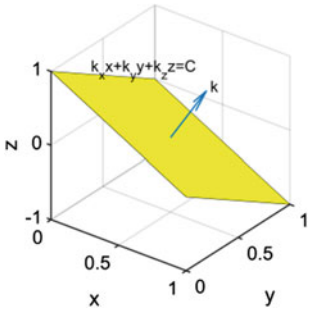


Fig. 1.3 The equiphase surface and the wave vector

where $\mathbf{A_0}$, $\mathbf{k}$, $\mathbf{r}$ are the initial field vector, the wave vector, and the radius vector, respectively. The so-called plane wave refers to the wave whose equiphase surface is a plane, namely

$$\mathbf{k} \cdot \mathbf{r} = k_x x + k_y y + k_z z = C. \tag{1.36}$$

Figure 1.3 shows the vertical relationship between the equiphase surface and the wave vector.

The plane wave can be divided into TEM wave, TE wave and TM wave according to whether there are electromagnetic field components in the propagation direction. TEM wave refers to the electromagnetic wave with neither electric field nor magnetic field in the propagation direction; TE wave refers to the wave with only magnetic field but no electric field in the propagation direction while TM wave with only electric field but no magnetic field. In Fig. 1.4a–c shows the relationship between **E**, **H** and **k** in TEM wave, TE wave and TM wave.

For TE wave, only the component of the magnetic field in the propagation direction is needed to obtain all the components of the electromagnetic field. Taking TE wave propagating in z direction as an example, assuming that H_z is known, then we have

$$E_x = -\frac{j\omega\mu_0}{k_c^2}\frac{\partial H_z}{\partial y}, \tag{1.37}$$

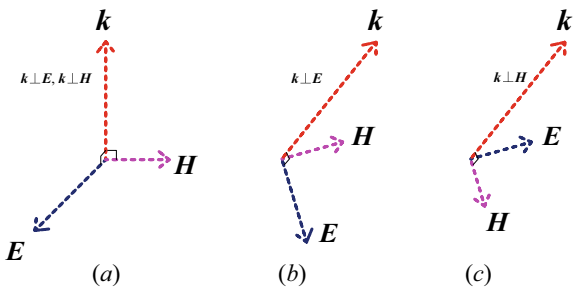


Fig. 1.4 TEM wave, TE wave and TM wave

$$E_y = \frac{j\omega\mu_0}{k_c^2}\frac{\partial H_z}{\partial x}, \tag{1.38}$$

$$H_x = \mp\frac{j\beta}{k_c^2}\frac{\partial H_z}{\partial x}, \tag{1.39}$$

$$H_y = \mp\frac{j\beta}{k_c^2}\frac{\partial H_z}{\partial y}. \tag{1.40}$$

For TM wave propagating along z direction, assuming E_z is known, then we have

$$E_x = \mp\frac{j\beta}{k_c^2}\frac{\partial E_z}{\partial x}, \tag{1.41}$$

$$E_y = \mp\frac{j\beta}{k_c^2}\frac{\partial E_z}{\partial y}, \tag{1.42}$$

$$H_x = \frac{j\omega\mu_0}{k_c^2}\frac{\partial E_z}{\partial y}, \tag{1.43}$$

$$H_y = -\frac{j\omega\mu_0}{k_c^2}\frac{\partial E_z}{\partial x}. \tag{1.44}$$

1.2 Forward Scattering Model

In a broad sense, electromagnetic forward problems generally refer to the problem of solving electromagnetic field with known source type and position, and the distribution of constitutive parameters. Typical electromagnetic forward problems may include many aspects. For example, in electrostatic field, the distribution of electric potential or electric field is solved when the distribution of electric charge density and permittivity is given; in static magnetic field, the distribution of magnetic field is solved when the electric current density and permeability are given. In electromagnetic scattering problems, the scattered field and total field are solved when the incident field and the scatterer are given.

As this book mainly discusses the solution of electrodynamic scattering problem, we will focus on the basic principle of the scattering in this part.

As shown in Fig. 1.5, when a group of sources (including point source, plane wave, etc.) encounter obstacles (scatterers), they will have electromagnetic interaction with the obstacles. Based on Huygens' principle, obstacles can be treated as secondary sources to generate electromagnetic radiation. Assuming that the fields without obstacles are $\mathbf{E}^{\mathbf{i}}$, $\mathbf{H}^{\mathbf{i}}$, which are defined as the incident electric field and magnetic field. $\mathbf{E}$, $\mathbf{H}$ are the total field, which are defined as the field actually existing in space when there are obstacles. The difference between the total field and the incident field is

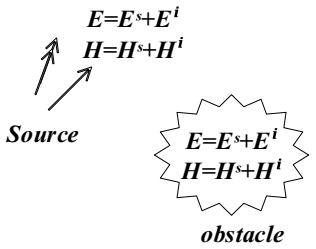


Fig. 1.5 The scattering model. $\mathbf{E}/\mathbf{H}$, $\mathbf{E}^{\mathbf{s}}/\mathbf{H}^{\mathbf{s}}$, $\mathbf{E}^{\mathbf{i}}/\mathbf{H}^{\mathbf{i}}$ are the total electric/magnetic field, the scattering electric/magnetic field and the incident electric/magnetic field, respectively

defined as the scattering field.

$$\mathbf{E}^{\mathbf{s}} = \mathbf{E} - \mathbf{E}^{\mathbf{i}}, \tag{1.45}$$

$$\mathbf{H}^{\mathbf{s}} = \mathbf{H} - \mathbf{H}^{\mathbf{i}}. \tag{1.46}$$

The source of the scattering field is the conduction current, polarization current, and magnetization current excited by the incident field. Outside the obstacle, the scattering field is passive. We choose S to be the surface that is infinitely close to the obstacle but does not coincide, and apply the equivalent principle to the inside and outside of S [4]. Figure 1.6 shows the two problems *a* and *b*. In problem *a*, the scattering field distributes inside and outside S. In problem *b*, the total field distributes inside and outside S. Therefore, the following equivalent problem can be established inside and outside S. There is only scattering field outside S, and the original medium and total field inside S. The original problem and equivalent problem are shown in Fig. 1.7a, b. In order to support these electromagnetic fields, there should be equivalent surface electromagnetic current on S. The current can be obtained by tangential boundary conditions

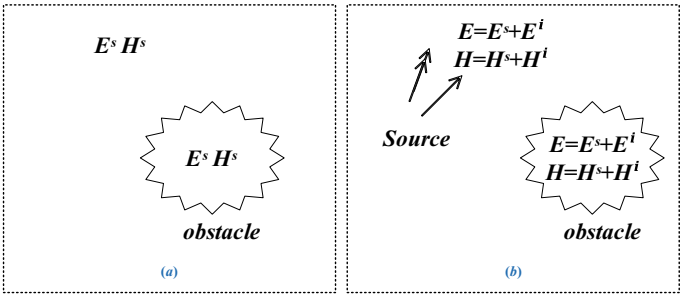


Fig. 1.6 Problem a and b. **a** The scattering field distributes inside and outside S. **b** The total field distributes inside and outside S

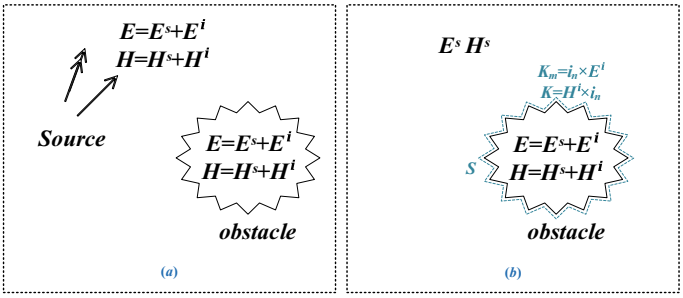


Fig. 1.7 Equivalence principle. **a** Origin problem. **b** Equivalent problem

$$\mathbf{K} = \mathbf{i_n} \times (\mathbf{H^s} - \mathbf{H}). \tag{1.47}$$

$$\mathbf{K_m} = (\mathbf{E^s} - \mathbf{E}) \times \mathbf{i_n}. \tag{1.48}$$

According to the relation between the incident field, scattering field, and total field, we can simplify (1.47) and (1.48) by

$$\mathbf{K} = -\mathbf{i_n} \times \mathbf{H^i}, \tag{1.49}$$

$$\mathbf{K_m} = -\mathbf{E^i} \times \mathbf{i_n}. \tag{1.50}$$

Since the scatterers are distributed in the limited space and the external radiation is towards the unbounded space, it is necessary to impose a boundary condition at infinity to make the scattering field continuously propagate to the outside world. This boundary condition is called the radiation boundary condition. In the three-dimensional problem, the radiation boundary conditions are

$$\lim_{r\to\infty} r(\mathbf{E^s} + \mathbf{i_r} \times \eta_0 \mathbf{H^s}) = 0, \tag{1.51}$$

$$\lim_{r\to\infty} r(\mathbf{H^s} - \mathbf{i_r} \times \frac{\mathbf{E^s}}{\eta_0}) = 0, \tag{1.52}$$

where

$$\eta_0 = \sqrt{\frac{\mu_0}{\varepsilon_0}}. \tag{1.53}$$

Here, η_0 is the wave impedance in vacuum, the unit of which is ohm.

1.3 Forward Modeling Algorithm

The forward problem in electromagnetic scattering refers to the process of solving the scattering field and total field with the known incident field and scatterer. After decades of development, the research of forward modeling is relative mature and a variety of methods have been conceived [18–20]. These methods can be generally divided into analytical methods and numerical methods. Analytical methods are applicable to the regular and symmetric regions. Although the solution is accurate, it is seldom used to solve practical electromagnetic problems. For most of the forward problems, numerical methods are more widely applied. Here, we will briefly introduce these two kinds of methods.

1.3.1 Analytical Methods

Common analytical methods [21–24] include the image method, electric axis method, variable separation method [25], Fourier transform method, series expansion method, etc. Due to space limitations, we mainly introduce the series expansion method, which is applied in Chap. 3 to verify the correctness of the forward program in Chap. 3. An in-depth introduction to other analytical methods can be found in [1–7].

The core of the series expansion is to expand a function into a series of superposition of orthogonal function families. Such expansion is often required to satisfy the mathematical form of boundary conditions so that it is easier to find the corresponding solution. Taking the most common Fourier series expansion as an example, its basis functions are orthogonal sine and cosine functions. For any function $f(x)$ satisfying Dirichlet condition, it can be expressed by Fourier series.

$$f(x) = \sum_{n=1}^{+\infty} (a_n \cos nx + b_n \sin nx) \tag{1.54}$$

Let the period of $f(x)$ be T and the Fourier coefficient can be expressed as

$$a_n = \frac{2}{T} \int_0^T f(x) \cos \frac{2\pi nx}{T} dx, \tag{1.55}$$

$$b_n = \frac{2}{T} \int_0^T f(x) \sin \frac{2\pi nx}{T} dx. \tag{1.56}$$

In two-dimensional scattering problems, we usually use Bessel functions [26–28] to decompose the incident plane wave to satisfy the corresponding boundary conditions. In Chap. 3, we verify the correctness of the FDFD program by calculating the

scattering of TM_z waves by an infinite cylinder [29, 30]. In three-dimensional scattering problems, we usually employ the spherical Bessel function [31] to decompose the incident plane waves. In Chap. 3, we also verify the correctness of the 3D-FDFD program in calculating the scattering of TM_z waves by a sphere. The Bessel functions and spherical Bessel functions of the order 0–4 are shown in Fig. 1.8.

1. 2D Cases

In our 2D cases, assuming that the incident plane wave propagates along the $+x$ direction, the electric field can be expressed as

$$E_z^i = E_0 \sum_{n=-\infty}^{+\infty} j^{-n} J_n(k\rho) e^{jn\varphi}, \tag{1.57}$$

where $J_n(k\rho)$ indicates a series of cylindrical standing waves. Here, we only consider the scattering of perfect electric conductors and perfect dielectrics. The scatterer shown in Fig. 1.9 is a perfect electric conductor cylinder placed at $x = 0, y = 0$ with radius a.

According to the equivalent principle, the scattering field can be regarded as outward traveling waves with the electromagnetic current on the cylinder as the source. Therefore, we have

$$E_z^s = E_0 \sum_{n=-\infty}^{+\infty} j^{-n} a_n H_n^{(2)}(k\rho) e^{jn\varphi}, \tag{1.58}$$

where $H_n^{(2)}(k\rho)$ indicates a series of outward traveling waves. The total field can be expressed as

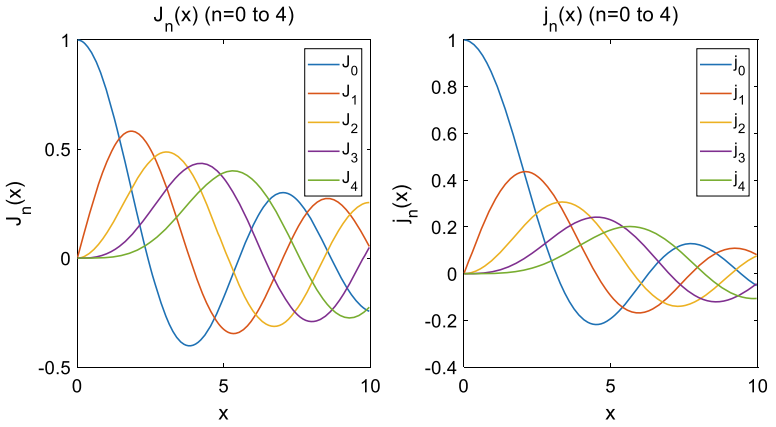


Fig. 1.8 The Bessel functions $J_n(x)$ and spherical Bessel functions $j_n(x)$ of the order 0–4

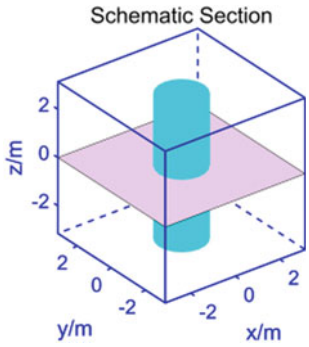


Fig. 1.9 A PEC cylinder placed at the origin

$$E_z = E_z^i + E_z^s. \tag{1.59}$$

For the PEC cylinder, it satisfies the boundary condition that the tangential electric field is zero,

$$E_z(a) = 0. \tag{1.60}$$

Therefore, a_n can be obtained as

$$a_n = \frac{-J_n(ka)}{H_n^{(2)}(ka)}. \tag{1.61}$$

The total field can be calculated through (1.60)

$$E_z = E_0 \sum_{n=-\infty}^{+\infty} j^{-n}\left[J_n(k\rho) - \frac{J_n(ka)}{H_n^{(2)}(ka)} H_n^{(2)}(k\rho)\right] e^{jn\varphi}. \tag{1.62}$$

Here, the distribution of the incident electric field, scattering electric field, total electric field and surface current of the PEC are shown in Fig. 1.10.

The scatterer shown in Fig. 1.11 is a perfect dielectric cylinder placed at $x = 0$, $y = 0$ with radius a. The relative permittivity is ε_r and the permeability is μ_r.

As the electric and magnetic fields satisfy Maxwell's equations, we have

$$\mathbf{H} = -\frac{1}{j\omega\mu}\nabla \times \mathbf{E} = \frac{1}{j\omega\mu}\left(\frac{E_z^i}{\rho} + \frac{\partial E_z^i}{\partial \rho}\right)\mathbf{i}_\varphi. \tag{1.63}$$

Therefore, we can get the two components of the incident magnetic field

$$H_\varphi^i = \frac{1}{j\omega\mu\rho} \cdot \frac{\partial \rho E_z}{\partial \rho} = \frac{1}{j\omega\mu}\left(\frac{E_z^i}{\rho} + \frac{\partial E_z^i}{\partial \rho}\right), \tag{1.64}$$

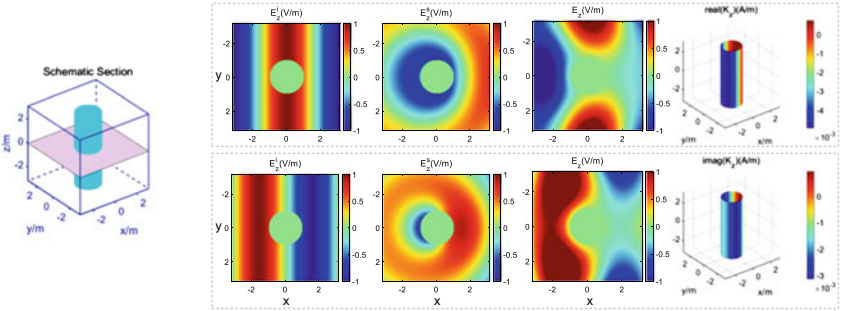
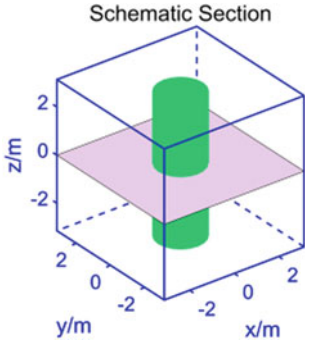


Fig. 1.10 Distribution of the incident electric field, scattering electric field, total electric field and surface current of the PEC cylinder

Fig. 1.11 A perfect electric dielectric cylinder placed at the origin



$$H_\rho^i = -\frac{1}{j\omega\mu\rho}\cdot\frac{\partial E_z}{\partial\varphi} = -\frac{E_0}{j\omega\mu\rho}\sum_{n=-\infty}^{+\infty} n\cdot j^{-n+1}J_n(k\rho)e^{jn\varphi}. \tag{1.65}$$

Since there is no surface charge or surface current on the interface, the tangential components of the electric and magnetic field must be continuous.

$$\mathbf{i_\rho}\times(\mathbf{E^i}+\mathbf{E^s}-\mathbf{E^t})=\mathbf{0}, \tag{1.66}$$

$$\mathbf{i_\rho}\times(\mathbf{H^i}+\mathbf{H^s}-\mathbf{H^t})=\mathbf{0}. \tag{1.67}$$

Namely

$$E_z^i+E_z^s=E_z^t, \tag{1.68}$$

$$H_z^i+H_z^s=H_z^t. \tag{1.69}$$

Since E_z^s is an outward traveling wave, it can be expressed as

$$E_z^s = E_0 \sum_{n=-\infty}^{+\infty} j^{-n} a_n H_n^{(2)}(k\rho) e^{jn\varphi}. \tag{1.70}$$

As the field value is limited at the origin, E_z^t can be expressed as

$$E_z^t = E_0 \sum_{n=-\infty}^{+\infty} j^{-n} b_n J_n(k_1\rho) e^{jn\varphi}. \tag{1.71}$$

Similarly, the magnetic field can be expressed as

$$H_\varphi^s = \frac{E_0}{j\omega\mu} \sum_{n=-\infty}^{+\infty} a_n j^{-n} \left(\frac{1}{\rho} H_n^{(2)}(k\rho) + k H_n^{(2)\prime}(k\rho) \right) e^{jn\varphi}, \tag{1.72}$$

$$H_\varphi^t = \frac{E_0}{j\omega\mu} \sum_{n=-\infty}^{+\infty} j^{-n} \left(\frac{1}{\rho} J_n(k\rho) + k J_n'(k\rho) \right) e^{jn\varphi}, \tag{1.73}$$

$$H_\varphi^t = \frac{E_0}{j\omega\mu} \sum_{n=-\infty}^{+\infty} b_n j^{-n} \left(\frac{1}{\rho} J_n(k_1\rho) + k J_n'(k_1\rho) \right) e^{jn\varphi}. \tag{1.74}$$

Applying the boundary condition at $\rho = a$, we can get

$$J_n(ka) + a_n H_n^{(2)}(ka) = b_n J_n(k_1 a), \tag{1.75}$$

$$\begin{aligned} &\frac{1}{\mu} \left(\frac{1}{a} J_n(ka) + k J_n'(ka) \right) + \frac{a_n}{\mu} \left(\frac{1}{a} H_n^{(2)}(ka) + k H_n^{(2)\prime}(ka) \right) \\ &\quad = \frac{b_n}{\mu} \left(\frac{1}{a} J_n(k_1 a) + k_1 J_n'(k_1 a) \right). \end{aligned} \tag{1.76}$$

As both media are perfect electric dielectrics, the permeability satisfies $\mu = \mu_1 = \mu_0$. We have

$$\sqrt{\varepsilon} J_n'(ka) + \sqrt{\varepsilon} a_n H_n^{(2)\prime}(ka) = \sqrt{\varepsilon_1} b_n J_n'(k_1 a). \tag{1.77}$$

Therefore,

$$a_n = \frac{\sqrt{\varepsilon_1} J_n(ka) J_n'(k_1 a) - \sqrt{\varepsilon} J_n(k_1 a) J_n'(ka)}{\sqrt{\varepsilon} J_n(k_1 a) H_n^{(2)\prime}(ka) - \sqrt{\varepsilon_1} J_n'(k_1 a) H_n^{(2)}(ka)}, \tag{1.78}$$

$$b_n = \frac{J_n(ka) + a_n H_n^{(2)}(ka)}{J_n(k_1 a)}. \tag{1.79}$$

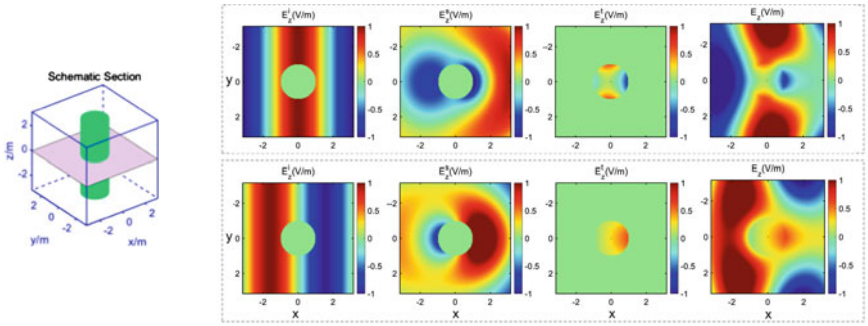


Fig. 1.12 Distribution of the incident electric field, scattering electric field, transmission electric field and total electric field of the perfect dielectric cylinder

Figure 1.12 shows the incident electric field, scattering electric field, transmission electric field and total field in z direction.

2. 3D Cases

In our 3D cases, assuming that the incident plane wave propagates along $+z$ direction and polarizes along $+x$ direction, the electric field can be expressed as

$$E_z = E_0 e^{jkz} = E_0 \sum_{n=0}^{+\infty} j^n (2n+1) j_n(r) P_n(\cos\theta), \tag{1.80}$$

where $j_n(k\rho)$ indicates a series of spherical standing waves. Like the 2D case, we only consider the scattering of perfect electric conductors and perfect dielectrics. The scatterers shown in Fig. 1.13 are perfect electric conductor sphere and perfect dielectric sphere. The sphere with radius a is placed at the origin.

The perfect conducting sphere satisfies the boundary condition that the tangential electric field is zero,

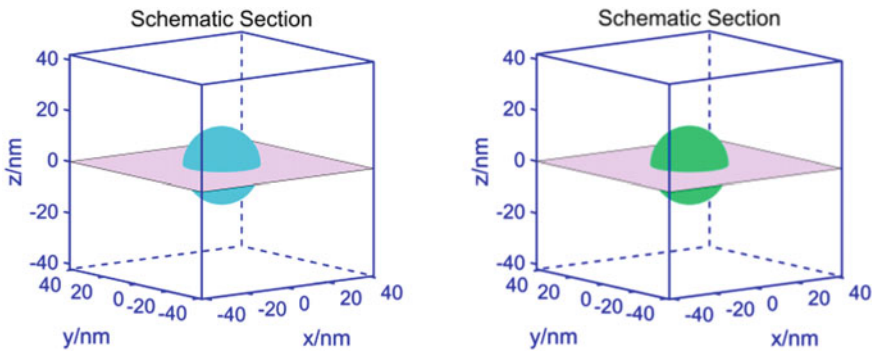


Fig. 1.13 Schematic section of the perfect conducting sphere and perfect dielectric sphere

$$E_\varphi(a) = 0, \tag{1.81}$$

$$E_\theta(a) = 0. \tag{1.82}$$

The perfect dielectric sphere satisfies the boundary condition that the tangential electromagnetic field is continuous,

$$E_\varphi^i(a) + E_\varphi^s(a) = E_\varphi^t(a), \tag{1.83}$$

$$H_\varphi^i(a) + H_\varphi^s(a) = H_\varphi^t(a), \tag{1.84}$$

$$E_\theta^i(a) + E_\theta^s(a) = E_\theta^t(a), \tag{1.85}$$

$$H_\theta^i(a) + H_\theta^s(a) = H_\theta^t(a). \tag{1.86}$$

The magnetic vector potential A_2 and the electric vector potential F_r are as follows,

$$A_r = E_0 \frac{\cos\varphi}{\omega} \sum_{n=1}^{\infty} \left(a_n \widehat{J}_n(kr) + b_n H_n^{(2)}(kr) \right) P_n^1(\cos\theta), \tag{1.87}$$

$$F_r = E_0 \frac{\sin\varphi}{\omega\eta} \sum_{n=1}^{\infty} \left(a_n \widehat{J}_n(kr) + c_n H_n^{(2)}(kr) \right) P_n^1(\cos\theta), \tag{1.88}$$

where

$$a_n = j^{-n} \frac{2n+1}{n(n+1)}. \tag{1.89}$$

When solving the electromagnetic field, the boundary condition is applied to obtain b_n and c_n first. Then the magnetic vector potential A_2 and the electric vector potential F_r are solved with the known coefficient a_n, b_n and c_n. At last, the electromagnetic field can be acquired through the vector potential. Due to the limitation of space, it will not be repeated here.

1.3.2 Numerical Methods

As mentioned above, when faced with complex geometry or non-uniform structure, analytical methods may not work anymore. Therefore, the numerical method has to be applied. According to the applicable electric size, we can divide the numerical methods into high-frequency methods and low-frequency methods. High-frequency

methods include geometrical optics method (GO) [32, 33], physical optics method (PO) [33–35], geometric theory of diffraction (GTD) [36–38], physical theory of diffraction (PTD) [39], etc. These methods have relative high calculation speed but low accuracy. Therefore, they are mainly used to solve the large-scale problems. For small scale forward problems, low-frequency methods are more widely used to guarantee the high accuracy.

Common low frequency methods include differential equation-based methods and integral equation-based methods. Differential equation (DE) based methods are mainly used to handle fine structures and inhomogeneous media. The coefficient matrixes in the methods are sparse matrixes, which require less memory. However, the accuracy of the differential equation methods is low, and sophisticated boundary conditions, such as perfect matching layers (PML) [40], are needed. Generally speaking, these methods include the finite element method (FEM) [41], finite difference method (FDM) [42], finite volume method (FVM) [43], Moment of Methods [44], etc.

The integral equation methods can gain a high accuracy with relatively low spatial sampling density. Besides, they do not need any sophisticated absorbing boundary. However, the main drawback of the methods is the high calculation load resulting from the dense impedance matrix. The integral equation methods can be further divided into the surface integral equation methods (SIEM) and volume integral equation methods (VIEM). Based on the integrand physical quantity, the integral equation methods can also be grouped into the electric field integral equation (EFIE), magnetic field integral equation (MFIE), current integral equation (CFIE) and so on. For traditional computational electromagnetics algorithms, we mainly concentrate on the most easily comprehended finite difference method, which is adopted in Chap. 3 to generate training data.

The finite difference method (FDM) was first come up by A. Thom in 1920s to solve nonlinear hydrodynamic equations. Since then, it has been introduced into other fields and become widely used in scientific calculation. The FDM approximates a differential equation by a finite different equation.

Given the function $f(x)$, the secant is applied to approach the tangent so as to obtain its derivative at a certain point x_0. The difference schemes can be divided into forward difference (1.90), backward difference (1.91) and central difference (1.92) [20]. Figure 1.14 shows the three difference schemes.

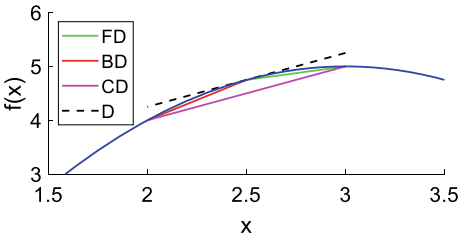


Fig. 1.14 Difference schemes. The FD, BD, CD and D are the forward difference, backward difference, central difference and derivative, respectively

$$f'(x_0) = \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x}, \tag{1.90}$$

$$f'(x_0) = \frac{f(x_0) - f(x_0 - \Delta x)}{\Delta x}, \tag{1.91}$$

$$f'(x_0) = \frac{f(x_0 + \Delta x) - f(x_0 - \Delta x)}{2\Delta x}. \tag{1.92}$$

It can be concluded that the central difference method can more accurately approximate the derivative. By using the approximation of the central difference twice, we can obtain the second order derivative.

$$f''(x_0) = \frac{f(x_0 + \Delta x) + f(x_0 - \Delta x) - 2f(x_0)}{(\Delta x)^2} \tag{1.93}$$

The wave equation in forward problem can be derived from the above formulas. In order to simplify the expression, we define

$$\Phi(i, j) = \Phi(i\Delta x, j\Delta t), \tag{1.94}$$

where $\Phi(i\Delta x, j\Delta t)$ represents the field value at $x = i\Delta x, t = j\Delta t$. The position of each point is shown in Fig. 1.15.

The wave equation to be solved is

$$u^2 \frac{\partial^2 \Phi}{\partial x^2} = \frac{\partial^2 \Phi}{\partial t^2}. \tag{1.95}$$

Suppose that the aspect ratio r is

$$r = \left(\frac{u\Delta t}{\Delta x}\right)^2. \tag{1.96}$$

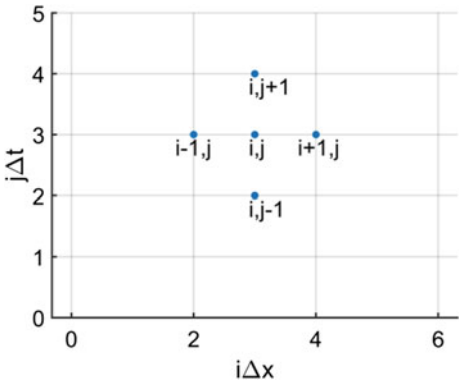


Fig. 1.15 The points in the difference scheme

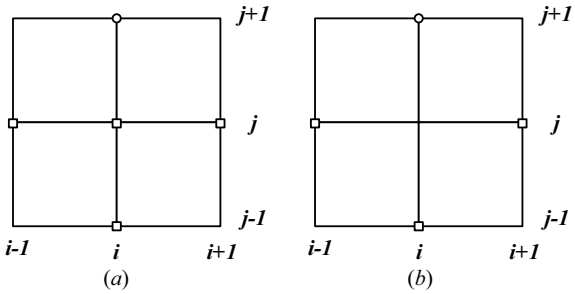


Fig. 1.16 Computational molecule when **a** $r < 1$ and **b** $r = 1$

For explicit methods, to ensure convergence, it is required that $r \leq 1$. We have

$$\Phi(i, j+1) = 2(1-r)\Phi(i, j) + r[\Phi(i+1, j) + \Phi(i-1, j)] - \Phi(i, j-1). \tag{1.97}$$

Figure 1.16 shows the computational molecule when $r < 1$ and $r = 1$ respectively.

By using the algorithm, it is possible to gain the field quantity at each point at any time.

Furthermore, the FDM can be divided into FDTD [45] and FDFD [46]. Since this book mainly discusses how to use deep learning technology to solve electromagnetic problems in the frequency domain, FDFD is opted as the traditional method to generate training data. For relevant contents, please refer to Chap. 3.

1.4 Summary

This chapter introduces the basic electromagnetic problems, scattering models and the traditional algorithms used in the following chapters. Starting from Maxwell's equations, we introduce the field laws, constitutive relationships, boundary conditions and basic concepts of plane waves. Next, the electromagnetic scattering model and equivalent principle are discussed. Finally, this chapter also expounds the fundamental principle of the difference algorithm in generating training data and the series expansion method in verifying the correctness of the difference program. The basic electromagnetic concepts described in this chapter also pave the way for the later introduction of the deep learning process.

References

1. Balanis CA (2012) Advanced engineering electromagnetics. Wiley, New Jersey
2. Bladel JV (2007) Electromagnetic fields. Wiley-Interscience, New York
3. Griffiths DJ (2017) Introduction to electrodynamics. Pearson, Boston

4. Harrington RF (2001) Time-harmonic electromagnetic fields. Wiley-IEEE Press, Piscataway
5. Jackson JD (1962) Classical electrodynamics. Wiley, New York
6. Jin JM (2015) Theory and computation of electromagnetic fields. Wiley-IEEE Press, Piscataway
7. Kong JA (1986) Electromagnetic wave theory. Wiley, New Jersey
8. Landau LD, Lifshitz EM (1980) The classical theory of fields. Butterworth-Heinemann, Oxford
9. Landau LD, Lifshitz EM, Pitaevskii LP (1980) Elelctrodynamics of continuous media. Butterworth-Heinemann, Oxford
10. Purcell EM (2013) Electricity and magnetism. Cambridge University Press, Cambridge
11. Schwartz M (1987) Principles of electrodynamics. Dover Publications, New York
12. Zangwill A (2012) Modern electrodynamics. Cambridge University Press, Cambridge
13. Ashcroft NW, Mermin ND (1976) Solid state physics. Cengage Learning, Stamford
14. Kittel C (2004) Introduction to solid state physics. Wiley, New Jersey
15. Rumble J (2020) CRC handbook of chemistry and physics. CRC Press, New York
16. Speight JG (2005) Lange's handbook of chemistry. McGraw-Hill Education, New York
17. Chew WC (1999) Waves and fields in inhomogeneous media. Wiley-IEEE Press, Piscataway
18. Mittra R (2014) Computational electromagnetics: recent advances and engineering applications. Springer, Berlin
19. Taflove A (2005) Computational electrodynamics: the finite-difference time-domain method. Artech House Publishers, London
20. Sadiku MNO (2018) Computational electromagnetics with MATLAB. CRC Press, New York
21. Evans G, Blackledge J, Yardley P (1999) Analytic methods for partial differential equations. Springer, Berlin
22. Henner V, Belozerova T, Nepomnyashchy A (2019) Partial differential equations: analytical methods and applications. CRC Press, New York
23. Morse PM, Feshbach H (1953) Methods of theoretical physics. McGraw-Hill, New York
24. Rylander T, Ingelström P, Bondeson A (2013) Computational electromagnetics (Texts in applied mathematics (51)). Springer, Berlin
25. Cain G, Meyer GH (2006) Separation of variables for partial differential equations: an Eigenfunction approach. CRC Press, New York
26. Jahnke E, Emde F (1945) Tables of functions. Dover, New York
27. Kreyszig E (2010) Advanced engineering mathematics. Wiley, New Jersey
28. Watson GN (1948) A treatise on the theory of Bessel functions. Cambridge University Press, Cambridge
29. Richmond JH (1965) Scattering by a dielectric cylinder of arbitrary cross section shape. IEEE Trans Antennas Propag 13(3):334–341
30. Wait JR (1959) Electromagnetic radiation from cylindrical structures. Pergamon, New York
31. Zhang S (1996) Computation of special functions. Wiley-Interscience, Piscataway
32. Lin PD (2014) New computation methods for geometrical optics. Springer, New York
33. Meng X, Guo LX, Dong CL, Jiao YC (2019) GO/PO method for the terahertz scattering computation of objects with multiple small-scale grooves. IEEE Access 7:40738–40745
34. Asvestas JS (1980) The physical optics method in electromagnetic scattering. Math Phys 21:290–299
35. Gutiérrez-Meana J, Martínez-Lorenzo JA, Las-Heras F (2011) High frequency techniques: the physical optics approximation and the modified equivalent current approximation (MECA). IntechOpen, London
36. Keller JB (1962) Geometrical theory of diffraction. J Opt Soc of Am 3(52):116–130
37. Pathak PH, Kouyoumjian RG (1974) A uniform geometrical theory of diffraction for an edge in a perfectly conducting surface. Proc IEEE 62(11):1448–1461
38. Tiberio R, Kouyoumjian RG (1979) A uniform GTD solution for the diffraction by strips illuminated at grazing incidence. Radio Sci 14:933–941
39. Paknys R (2016) Applied frequency-domain electromagnetics. Wiley, New Jersey, pp 317–334
40. Shin W, Fan S (2012) Choice of the perfectly matched layer boundary condition for frequency-domain Maxwell's equations solvers. J Comput Phys 23(1):3406–3431

41. Jin JM (2014) The finite element method in electromagnetics. Wiley-IEEE Press, Piscataway
42. Porsching TA (1980) Numerical solution of partial differential equations: finite difference methods (G. D. Smith). SIAM Rev 22(3):376
43. LeVeque RJ (2002) Finite volume methods for hyperbolic problems. Cambridge University Press, Cambridge
44. Harrington RF (1993) Field computation by moment methods. Wiley-IEEE Press, Piscataway
45. Yee KS (1966) Numerical solution of initial boundary value problem involving Maxwell's equations in isotropic media. IEEE Trans Antennas Propag 14(3):302–307
46. Champagne NJ, Berryman JG, Buettner HM (2001) FDFD: a 3D finite-difference frequency-domain code for electromagnetic induction tomography. J Comput Phys 170(2):830–848

Chapter 2
Basic Principles of Unveiling Electromagnetic Problems Based on Deep Learning

In last chapter, the basic electromagnetic theory behind the work has been introduced. In this chapter, it will concentrate on how to use DL method to solve electromagnetics related problems. At the beginning, the fundamental knowledge of DL basics and related physical background are discussed. Technical details in data acquisition, neural network training, performance testing is comprehensively reviewed then. After that, the validation process will also be illustrated. In the later chapter, more concrete examples of how these methods could be applied to solve specific problems will be exhibited.

2.1 Deep Learning Basics

2.1.1 Forward Pass

Feedforward neural networks, also known as multilayer perceptron, are the foundation of many deep learning frameworks, giving birth to a wide range of related applications. It basically forms a mapping $y = f(x, \theta)$ from input x to the output y, where θ is the trainable weights of neural network, representing the best function approximation of underlying mapping relation. The information goes from its input flow directly to the output with no feedback connections cut in. Neural networks with feedforward structure can be complicated with multilayer structure since intermediate computing function f may consists of different function units f^1, f^2, $f^3 \cdots f^n$ with vector-valued input and output, and these functions are named as the nth layers of the neural network. Specifically, the first layer is called input layer, while the last layer refers to output layer. Generally, multiple layers connect the input and output layers into a chain, to form the constituting mapping principles $f(x, \theta) = f^1(f^2(\cdots f^n(x)))$, they are also called hidden layers since no external information is given to the chain calculation. Such kind of neural network structure prove to be very efficient in

Q. Ren et al., *Sophisticated Electromagnetic Forward Scattering Solver via Deep Learning*, https://doi.org/10.1007/978-981-16-6261-4_2

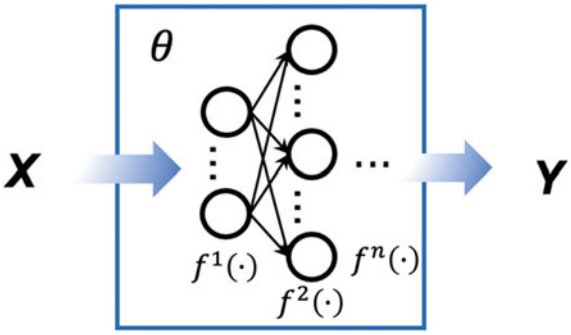


Fig. 2.1 The feedforward neural network

extracting meaningful features from large amounts of data [1]. The forward pass is illustrated in Fig. 2.1.

2.1.2 Backward Propagation

The training of neural networks starts from backpropagation procedure, a simple mechanism based on the chain rule of multivariate derivatives. In general, y, the output of feedforward neural network continues to produce a scalar loss function $L(y, y^*)$ to be optimized. The backpropagation happens when the loss function is optimized to update all the trainable weights of neural networks in order to obtain the desired output y^*. The mainstream deep learning computing platform formulates the whole neural network into a computational graph, where each node represents a variable with forms of scalar, matrix and tensor, while edges refer to the operation within the neural network, such as plus, multiply, square, etc. Computational graph helps algorithms to retrieve the partial derivatives of the loss function with respect to all the trainable parameters, such as the weights θ_n and bias b_n in the nth layers of neural network. Then, gradient-based optimization algorithms are applied to adjust these parameters accordingly. The detailed procedure will be properly illustrated in the following section.

2.2 Deep Learning Configuration

In the last few years, deep learning (DL) technology has been widely recognized as a supreme method to learn complex representation from large quantities of data with multiple levels of abstraction [2]. With worldwide efforts, this technology has push forward the frontiers of task mainly in artificial intelligence such as robotics [3], computer vision [4], as well as nature language processing [5], etc. Meanwhile, deep learning with its inherent advantage in performing mappings between inputs and

outputs has been successfully merged with other fields of research such as physics [6], chemistry [7], materials [8], and engineering [9]. Conventionally, the discovery of underlying mechanisms for these areas are merely accessible through rigorous deduction and complicated analysis which require both domain expertise and time-consuming calculations. Without directly solving the complex governing equations, deep learning helps to expedite the analysis by simplifying it into a weights-learning process for a dynamical network system. These parametric networks mainly help in mimicking the underlying mechanisms of any given systems using tons of trainable parameters. It could build up an end-product-driven engineering process where mathematical rigor is temporarily compromised with strong problem-solving ability.

It is commonly accepted that regression and classification are two major tasks performed by deep learning methods and are now playing an important role in deceiving nontrivial patterns between inputs and outputs. In most cases, the physical mechanism beneath the electromagnetic (EM) problems are represented by partial differential equations (PDEs) which are dynamical enough to quantitatively relate different initial values with their physical responses, if exist. This further reveals the fact that even under same mapping relation, the output of the system can be quite distinct and is hard to fall into a certain category, caused by different initial inputs. Therefore, DL-enabled regression instead of classification is the mainstream to solve EM problems. The revelation of underlying physics of EM problems can be simplified into network learning process for $y = f(x)$, where x denotes the initial setup, y represents the response space (see Fig. 2.2).

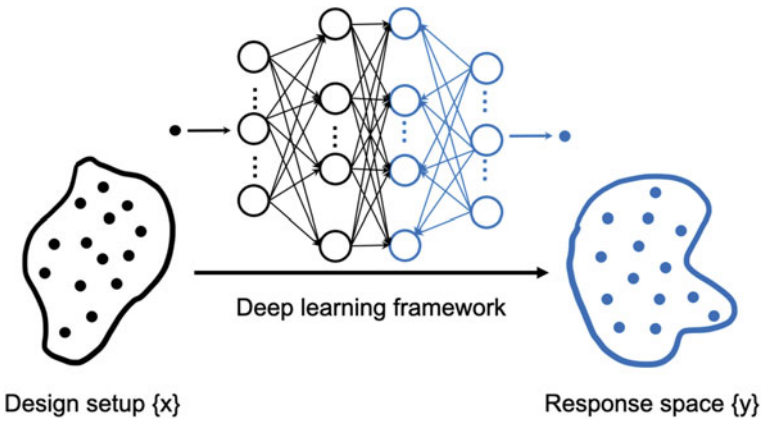


Fig. 2.2 A schematic represents the mapping relation in solving EM problems with deep learning approach

2.2.1 Categories of Input and Output Data Structures

In general, EM problems have diverse input and output data structures due to a broad range of targeted physical quantities such as field prediction [10–12], spectrum analysis [13, 14], device design [15–17] and optimization [18–20]. There are two general data structures that are highly suitable to define the initial input value and physical response, one is the image data defined by two-dimensional (2D) or three-dimensional (3D) matrix, and the other is the discrete data defined by one dimensional (1D) row or column vector. A subclassification can be made to 1D data, the first subclass is parameters to define the EM system, while the second one is the ordered sequence for a physical quantity such as time, wavelength, etc.

Different mapping patterns within distinct data structures are the explicit forms of physical modeling, the result of statistical derivation from the very basic physics rationale. It is noted that the mappings relation between two types of data structures consequently determine the dataset of reference, network architecture, as well as training method. In the following chapter, several mapping patterns within related EM problems are extensively articulated (Fig. 2.3).

Discrete to Discrete: Typically, the basic geometric features of EM or photonic devices with periodic structure can be parameterized into a vector of design variables such as width, height, thickness, and angle, etc. Same strategy can be also applied to other properties of devices, including optical properties (permittivity,

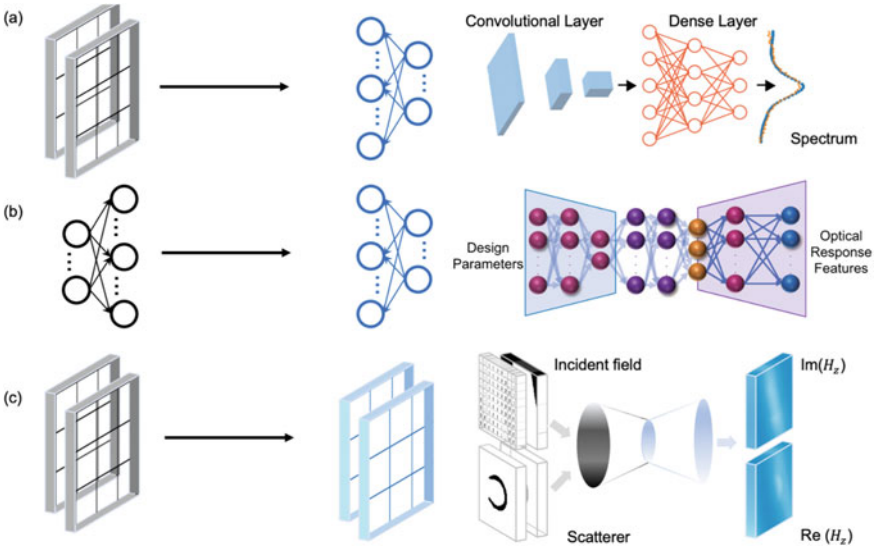


Fig. 2.3 **a** Schematic of image to discrete mapping. *Source* Li et al. [13]. Reproduced with permission of American Physical Society. **b** Schematic of discrete to discrete mapping. *Source* Yashar et al. [21]. **c** Schematic of image to image mapping. *Source* Li et al. [23]

permeability), illumination settings (wavelength, direction of incident wave, polarization), the inherent properties of constituent component (crystallization fraction) [21]. Meanwhile, spectra responses or port signals obtained from sampling points with fixed interval follow the same form of discretized representation. In this case, the discrete-to-discrete mapping patterns are well established either for forward analysis or inverse designs, the input and output data structures are thus facile to formulated as 1-D vectors.

Image to Discrete: Compared with discretized parameters, image is able to directly depict input with high dimensional features. This contributes to the representation of highly complex geometry which is difficult to be parameterized by a set of design variables. By adding the channels of image input, more information could be incorporated into input pairs such as illumination settings or optical properties with complex-valued form. Image to discrete mapping patterns are proposed to describe a specific range of phenomena such as resonant mode analysis [22] or spectra calculation of complex nanostructures [23]. In this case, input takes forms of multidimensional matrix, while output is formulated as 1-D vector.

Image to Image: The physical responses for all kinds of EM problems are not simply limited to discrete data type, an exception is the field distribution or electric polarization density. If the input and output are both in 2D matrix form, image to image mapping patterns are facile to establish. This data structure has an advantage on its physical interpretability and adaptability since 2D matrix is the natural representation of device structure, field pattern, etc. Many research has shown that this type of data format is highly suitable for per-pixel prediction of near-field responses [23]. Noted that the input and output matrixes do not need to have the same size or dimension, this endows the degree of freedom to tailor the input–output pipeline designs for specific applications.

These are three representative data structures for input and output schemes. With the rapidly growing trend of research activities in this area, other types of data structures appear to be very efficient in tackling specific EM-related problems. An example is the graph data structure which is used for EM behavior prediction of distributed circuits [24]. In this case, nodes are attributed with circuit parameters of each circuit components, while edges represent the EM coupling effects between these components. Clearly, different data structures should conform well to the problems of interests in the first place, without doubt, they are also required to further fit in with specific deep learning model.

2.2.2 Constructions of Deep Learning Model

In the previous section, we have introduced several types of input and output data formats that help to represent physical properties of EM problems. To further realize the mappings between the initial input and desired solution, data-dependent neural network architecture should be carefully selected. To date, various neural network architecture with complex inner structures have been proposed to solve all kinds of

problems [2], their sizes vary from single-digit to hundreds of hidden network layers. As a result, it is very important to select proper DL-model which exhibits the best performance on learning the function mapping between input and output.

In view of the fixed data formats for each type of DL-models to process, the mapping patterns derived from input and output data structures of EM problem can be considered as a first criterion to evaluate the applicability of certain DL-model. Beyond that the tuning of network structure regarding the complexity of specific problems is another essential step towards better learning outcome. In the following section, we will dive briefly into several mainstream DL-models and establish their correlation with the EM problems at hand.

Fully connected : As previously discussed, the fundamental unit for deep neural network is the neuron. In the fully connected neural network, neurons in one layer are connected to all the neurons in the next layer, this way of connection between layers is called fully connected (Fig. 2.4). Notably, a fully connected neural network consists of a series of fully connected layers, each layer can have different numbers of neurons, depending on the how complicated for neural network to solve the problem. Mathematically, set $x_i \epsilon \mathbb{R}^m$ as the ith input of the fully connected layer, while $y_i \epsilon \mathbb{R}^n$ as the ith output, the mapping of x to y can be listed as

$$y_i = f\left(\sum_i w_i x_i + b\right), \tag{2.1}$$

where w_i denotes weights for ith connection for x_i and y_i, b is the bias set to improve network learning, $f(x)$ is the nonlinear activation function. Clearly, the fully-connected connection fits well with the discrete data, especially for a number

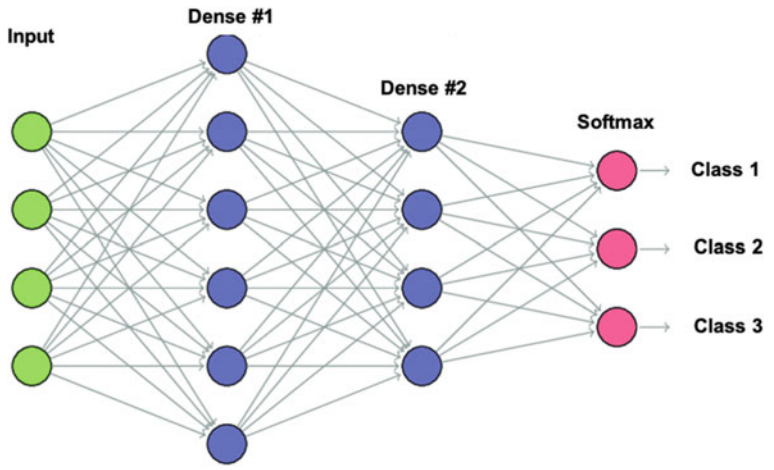


Fig. 2.4 Schematic of fully connected neural network

of device variables or sampled responses appeared in the EM modeling which have no explicit spatial and sequential correlation [25, 26].

Convolutional Neural Network: Unlike the fully connected neural network architecture, a convolution layer appeared as so-called feature map is formulated by locally weighted summarizations of the previous layer (Fig. 2.5). For a single weighted calculation, dot production between one local batch and convolution filter is performed, followed by a nonlinear activation function [27–29]. The calculation will start from neighboring batches with a fixed step size to go through all the elements for input feature map, the output of each calculation then forms into the next feature map. Generally, convolution filter is smaller than the size of feature map, it incorporates a matrix of trainable weights which are shared by two locally connected layers, allowing the network to go deeper without the burden to store tons of parameters. With the help of these features, convolutional neural network is very efficient in processing data structures with large sizes and higher dimension, such as 2-D or 3-D image. That is to say, convolution layer could be adopted if specific electromagnetics system can be naturally represented as 2-D or 3-D data structure. For instance, the DL-models for image-to-image mapping patterns may be purely convolutional, while for the case of image to discrete, the DL-models are more likely to consists of both convolution layers and fully connected layers.

Recurrent Neural Network: RNN (see Fig. 2.6) has been widely adopted in the deep learning community as an effective tool to process ordered sequence, such as speech and language, it uses hidden state h^{t-1} to store the information from previous time steps, elements x^t, $t \in [0, size(x)]$ in the input sequence x feed into the network one by one at every given time t, updating hidden state with h^t and to produce the output element y^t, the governing equations can be written as

$$h^t = f\left(W_h h^{t-1} + W_x x^t + b_h\right), \tag{2.2}$$

$$y^t = g\left(W_y h^t + b_y\right), \tag{2.3}$$

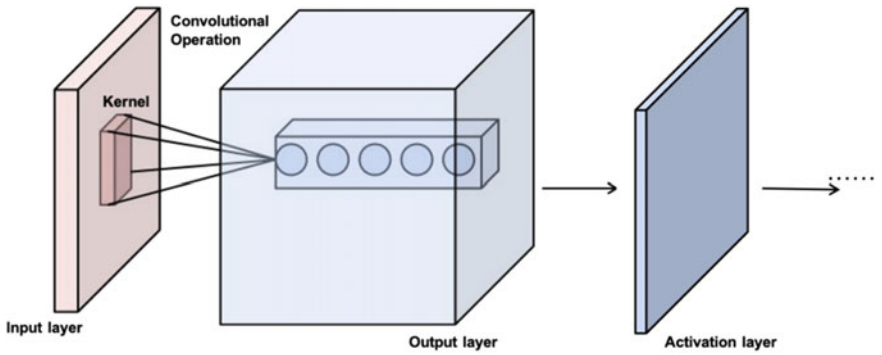


Fig. 2.5 Schematic of convolutional neural network

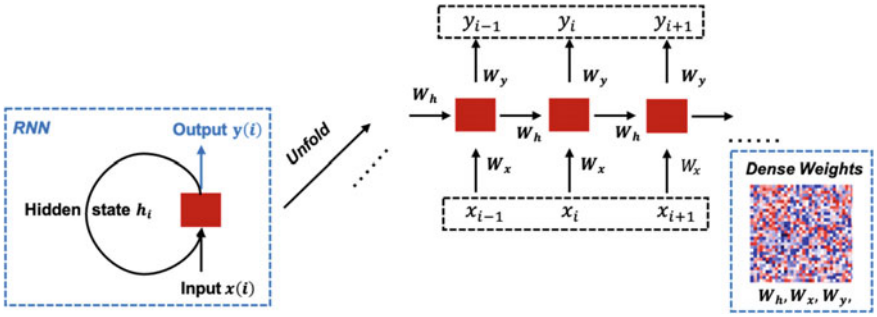


Fig. 2.6 Schematic of recurrent neural network

where W_h, W_x, W_y are dense weight matrices for which the network needs to learn during training, b_h and b_y are the bias terms for hidden state h^t and the output y^t, respectively, while $f(x)$, $g(x)$ is the nonlinear activation functions. For a number of problems in the domain of multi-physics modeling, the solution of a dynamical system at a given time not only depends on the excitation source at that time, but also the hidden state of the system at previous time step. An example is the heat conduction problem [30], the current temperature distribution for a given surface is the result of a combination effect of its previous temperature distribution and heat source at that time. The same goes to the electromagnetics problems where wave equations are intensively involved. In fact, the physical modeling of electromagnetic wave phenomena in the time domain has proved to be very similar to the implementation of deep recurrent neural networks (RNNs) [31].

Generative Adversarial Networks: Generative adversarial networks (GAN) was first brought up by *Ian Goodfellow* in 2014 [32]. This type of network structure aims to provide estimation for the generative models by constructing two sub-models into an adversarial process. These two models are named as generative model G and discriminative model D, each of them is constructed by specific type of neural networks with target objective functions, all in the same training procedure. For the generator, the main goal is to generate structures that are similar to the training sets in order to fool the discriminator, thus the differentiable objective function can be formulated as

$$\min_G \mathbb{E}_{z\sim p_z(z)}\big[\log(1 - D(G(z)))\big], \tag{2.4}$$

where $p_z(z)$ is prior distribution of noise input z. The discriminator is generally constructed as a 0/1 classifier, trying to differentiate the 'fake data' generated by G and the 'real data x' collected from the training dataset $p_{data}(x)$. In this case, the objective function has the following explicit form as

$$\min_D \mathbb{E}_{x\sim p_{data}(x)}\big[\log(D(x))\big] + \mathbb{E}_{z\sim p_z(z)}\big[\log(1 - D(G(z)))\big]. \tag{2.5}$$

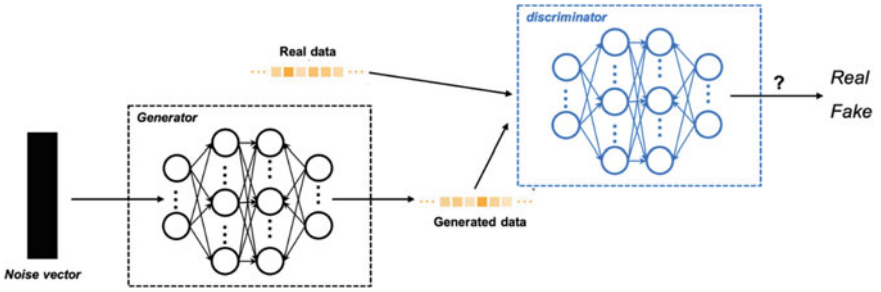


Fig. 2.7 Schematic of generative adversarial neural network

Intuitively, it is not hard to find that the adversarial process is inherently a zero-sum game, where one agent's gain is another agent's loss. Gradient-based rule is applied to the two objective functions during the network training until the equilibrium state is reached, where G is able to mimic the distribution of training dataset and outputs 1/2 everywhere [32]. Recently, GAN has been widely used in the area of inverse EM problem either for device designs [33, 34] and imaging [35] due to its strong capability to generate patterns, the schematic diagram (see Fig. 2.7) for GAN is listed as below.

2.2.3 Training of Proposed Deep Learning Model

In the previous section, we discuss the data structure and data-dependent neural network architecture in detail. This section aims to provide the knowledge to properly train the DL-model. As we have discussed before, supervised paradigm is the most common form to solve EM problem with a deep learning fashion. In this case, every instance in the dataset used for network training contains a pair of input object and desired output pattern, the training can be further summarized as a process to adjust network internal weights with the help of reference dataset. Once fully trained, the deep neural network is able to output the correct solution for an unseen initial input.

2.2.3.1 Loss Functions

There are a number of training details that could affect the final learning outcome, the first one is the definition of loss function for neural network. Usually, in the case of EM related problem, loss function is defined by the difference between the predictions of neural network and real physical response. As a matter of fact, supervised deep learning is inherent an optimization problem, for which an objective function needs to be clearly defined in the first place. In this case, the objective of network training is to minimize the loss function. Since the most of electromagnetics or photonic

problem involves with regression task, L_1 loss and L_2 loss are two frequently used loss functions derivate from the norm of a matrix. These two functions are defined as

$$L_1\, loss = \sum_{i=1}^{n} \left| y_i^{pre} - y_i \right|, \tag{2.6}$$

$$L_2\, loss = \sum_{i=1}^{n} \left| y_i^{pre} - y_i \right|^2, \tag{2.7}$$

where y_i denotes the ith desired response in the training data, y_i^{pre} represents the ith neural network prediction, and n is the number of total elements in the response space. Empirically, L_2 loss is more efficient for convolutional neural network training [36], the time takes for the convergence of neural network is less than L_1 loss. However, L_2 loss suffers from the instance that is apparently deviated from the normal error range. Noted that in some cases, the measurement of similarity between network prediction and ground truth data can be nontrivial, e.g., the output follows a specific probability distributions [37]. In this case, the loss function has distinct from compared with L_1 and L_2 loss and should be properly designed.

2.2.3.2 Optimizers

Optimizer for the training of neural network is actually the optimization algorithm that is used to minimize the value of loss function $L(\theta)$ by adjusting the internal weights $\theta \epsilon \mathbb{R}^d$ of neural network. The optimizer will introduce an extra tuning parameter called learning rate η to the DL-model, this parameter defines the step size to update for each training iteration $\theta = \theta - \eta \nabla_\theta L(\theta)$. The non-trainable feature of η means it needs to be set manually with proper value, this requires trial and error, as well as empirical knowledge of the training of deep neural network.

There are generally two types of optimizers that are frequently used to train the network, all based on the same idea of gradient descent $\nabla_\theta L(\theta)$. The key difference between them lies in whether the learning rate is self-adaptive as the training proceeds. The selection of different optimizers is problem-based. In general, optimizers with adaptive learning rates are very efficient in dealing with complex deep neural network, where the optimization function is highly non-convex posed, *Adam* optimizer [38] is one of this kind which has overall better performance.

2.2.3.3 Weight Initialization

Weight initialization can be quite trivial when the structure of neural network is very simple, i.e., fully connected neural network with a few hidden layers, this can be seen in very basic EM problems such as calculation of scattering spectrum for photonic

device with simple geometric feature. In this case, all weights at any layers can be randomly initialized. However, as the problems become more complicated, such as the field prediction for complex nanostructure, the number of hidden layers with tons of neurons expands greatly followed by nonlinear activation function. In this case, it is more likely to fall into the situation when problems like gradient exploding and gradient vanishing rise, either is detrimental for network training. The triggering factor for these problems is the weights are initialized either too large or too small. Here, we introduce *Xavier* initialization [39] to prevent these problems. For neural network that is mainly based on ReLU nonlinearity, He initialization [40] could also give out a good performance.

2.3 Validation and Evaluation

For any DL-model that aims to mimic the complex underlying mechanism for solving EM problems, it is almost impossible to get the satisfied result after the first round of training. This is because non-trainable parameters, such as the size of training dataset, numbers of hidden layer, all account for the final performance of DL-model and these parameters require manual setting. Thus, it can be an iterative process to get the DL-model properly trained. To solve this problem, validation is put forward to help analyze and expedite the fine-tuning process. It provides rightful methodology to tune the afore-mentioned non-trainable parameters. The experiment results obtained from network validation can help us improve the deep learning system.

2.3.1 Network Validation

The same as network training, validation requires a separated dataset that is supposed to be independent and identical distributed to the training dataset. The common practice is to divide the original dataset into training set, validation set and test set. The ratio of partitions varies with the size of dataset. For example, if the size of dataset is around ten thousand or smaller, greater percentage of validation set and test set, such as 60%: 20%: 20%, promises to give the result of validation smaller variance, which is more valid to be a reference for parameter-tunning. However, if the dataset has millions of instances, smaller percentage of validation set and test set is sufficient to refine and test the models, since enough instances have been incorporated to represent most of the variance in data, 99.5%: 0.25%: 0.25% is a suggested ratio of partition. It is noted that, for general EM problems, the dataset used for deep learning is constructed either from tons of computer simulations or lab scale experiments, both methods can be very time-consuming. In all these cases, the scale of dataset should keep in the range between 1000 and 100,000, smaller dataset may lead to problems of overfitting.

2.3.2 Hyperparameter Tuning

2.3.2.1 Manually Tuning

In most cases, we tune the neural network on the validation set in order to maximize the working efficiency and most importantly, minimize the error rate for the specific tasks. Since deep learning algorithms generally consists of many hyperparameters which set to control of the behaviors of algorithms, the tuning process is all about searching for a set of hyperparameters that gives the neural network better performance.

Most of the hyperparameters are non-trainable during the training process, which means the hyperparameters should be assigned with certain values and stay the same for an entire round of training. There are two commonly used methods to select these hyperparameters, the first one is selecting manually based on the knowledge of how these hyperparameters will affect the quality of deep learning algorithms, the second one refers to selecting automatically with the help of optimization algorithms.

Deep learning community nowadays has established a systematical guidance about how different hyperparameters could impact the effective network capacity of the proposed deep learning algorithms. In general, this capacity is required to match with the level of specific task. For EM problems, models with large capacity are hard to train and are likely to overfit the problems, while relatively small capacity may find it difficult to learn the governing equations, that is to say, as the scales of the problems grow, e.g., 3D computational domain over 2D, electro-dynamical system verse electro-static problem, the effective capacity of the models should be accordingly enlarged. In these cases, hyperparameters such as the number of hidden units, filter size, weight decay rate and dropout rate constrain the model's capacity and are the ones need to be tuned properly.

After discussing how these hyperparameters could affect the effective capacity of the proposed neural network, there is a need to find any quantity that is able to represent it. Notably, test error and total loss can be used to diagnose the network training, these two figures of merits directly give out the outcome of network training and can be really helpful for manually tuning of hyperparameters.

2.3.2.2 Automatic Tuning

Manually tuning the hyperparameters requires expert-level experiences and is inherent an iterative process, it can be really time-consuming given the fact that most of the problems in electromagnetics do not have a baseline for evaluating DL-based methods, that is to say, there is no reference as a good starting point for parameters tuning. Most importantly, one of the reasons why the research community has a strong interest of applying DL methods to the traditional EM problems is that this approach could achieve a certain level of self-consistency, which is good for problem solving and device optimization.

Nowadays, optimization algorithms are introduced to automatic hyperparameters tuning. Basically, choosing rightful parameters in order to minimize the loss value and decrease the error rate can be seen as a multi-objective optimization problem. A good practice for network tuning is Bayesian optimization [41]. Unlike other kinds of optimization, in Bayesian optimization, instead of using local gradient for approximations, the algorithm takes full advantages of the previous evaluation of optimized function $f(x_n)$, $\{x_n \in \mathcal{X}\}_{n=1}^{N}$ to decide where in $\mathcal{X}$ to next evaluate. In our cases, $\mathcal{X}$ is the set of observation behaviors with each x_n represents the union of hyperparameters at observation n, while N is the number of observations. Particularly, $f(x_n)$ follows the Gaussian process (GP) prior ($y_n \sim \mathcal{N}(f(x_n), v)$) due to its flexibility and tractability. The key ideal for Bayesian optimization is its way to determine the next observation point (x_{n+1}, y_{n+1}), the acquisition function $\alpha(x_n)$ is brought up to deal with the problem. There are in general two types of strategy to define the acquisition function, one is Expected Improvement (EI) criterion and the other is GP upper confidence bounds (GP-UCB). Owing to the fact that the former one works well in the minimization problems and do not incorporate terms with tunable parameter, EI criterion is more frequently used, with its explicit form listed as

$$\gamma(x_n) = \frac{f(x_n) - \mu(x_n)}{\sigma(x_n)}, \tag{2.8}$$

$$\alpha_{EI}(x_n) = \sigma(x_n)\big[\gamma(x_n)\Phi(\gamma(x_n)) + \phi(\gamma(x_n))\big], \tag{2.9}$$

where $\mu(x)$ is the predictive mean function and $\sigma(x)$ is the predictive variance function, $\Phi(\cdot)$ is the cumulative distribution function of the standard normal, $\phi(\cdot)$ is the probability distribution function of the standard normal. Therefore, next observation point $x_{n+1} = argmax_{x_n}\alpha_{EI}(x_n)$. It is noted that the best union of hyperparameters x_{best} can be derived from $argmin_{x_n} f(x_n)$. Very recently, several studies related to EM problems have integrated this method for better tuning of hyperparameters [13, 42], mostly for convolutional neural network, their numerical results justify the improved performance induced by Bayesian optimization.

2.3.3 Reference Test Dataset

Most of the implementations of DL-based approach on EM problem rely on tons of data which is generated by computer simulation rather than collected directly from lab experiments. This is because, data collected from the real-world experiment is often insufficient for model to approximate the underlying physical law. In general, the majority of computational task in the domain of electromagnetics can be solved accurately with the help of existing numerical solver [43, 44], giving full degree of freedom to choose self-defined geometrical model for problem in hand, thus it is preferred to use large amount of, noise-free simulated data to train the neural network.

However, simply applying a self-generated dataset for both training and testing casts the doubts on the generalizability of purposed deep learning model, in this case, the neural network might perform badly outside of self-generated datapoint or be less robust to the random noise in real word experiments, clearly, these features undermine the further application for universal methods of design and analysis. This problem aroused when neural network itself being forced to approximate the limited scenario for a small set of datapoints.

In order to avoid the bad generalizability of neural network, it is better to gather more data that gives both training and testing datasets richer set of features. Besides, a more generalized reference test dataset could be referred for further validations. This testing fashion is now widely used in the AI community as a common standard for comparison of performance between different DL models. Here are some open-source datasets which can be used as standard reference when EM problem involves certain kind of geometrical representation either for forward analysis or inverse design.

The MNIST dataset: MNIST database first collected by Yann LeCun, contain 28 by 28 hand-written images of number characters from 0 to 9, it has a training dataset of 60,000 examples and a test dataset of 10,000 examples [45]. This dataset is now widely used in the machine learning community as basic benchmark test. The comprehensive description and download link can be referred to http://yann.lecun.com/exdb/mnist/. Here, some examples of the MNIST dataset are displayed in Fig. 2.8.

The good sides of MNIST dataset lie in its simple format with rich hidden patterns. The images in this dataset are well divided into different groups of numbers with proper label, and each set of images is black and white which can be easily binarized to generate many kinds of geometries with distinct features, this is especially useful when we want to enhance the generalizability of machine learning model. Besides, these geometries can be post-processed to further possess specific physical properties, such as relative permittivity.

The 2D shape structure dataset: Neural networks are famous for the ability to learn nonlinear mapping relation between inputs and outputs, but at the same time, using this method may encounter the problem of overfitting, this happens with a consequence that they cannot generate accurate result outside of training samples. However, this feature is difficult to detect during the training phase, especially for the self-generated dataset, for which dataset may overlap or easily be interpolated. An efficient way to test the generalize ability of proposed neural network architecture is to adopt more complicated test dataset. For the electromagnetic problem concerned with geometrical properties, 2D shape dataset with rich and complex geometries [46] could be obtained for performance validation and case analysis. This dataset involves over 1200 real life objects in 70 shape classes which can be easily projected to 2D Cartesian mesh, the comprehensive description and download link can be referred to the website http://ubee.enseeiht.fr/ShapesDataset. Here, examples of the 2D shape structure dataset are displayed in Fig. 2.9.

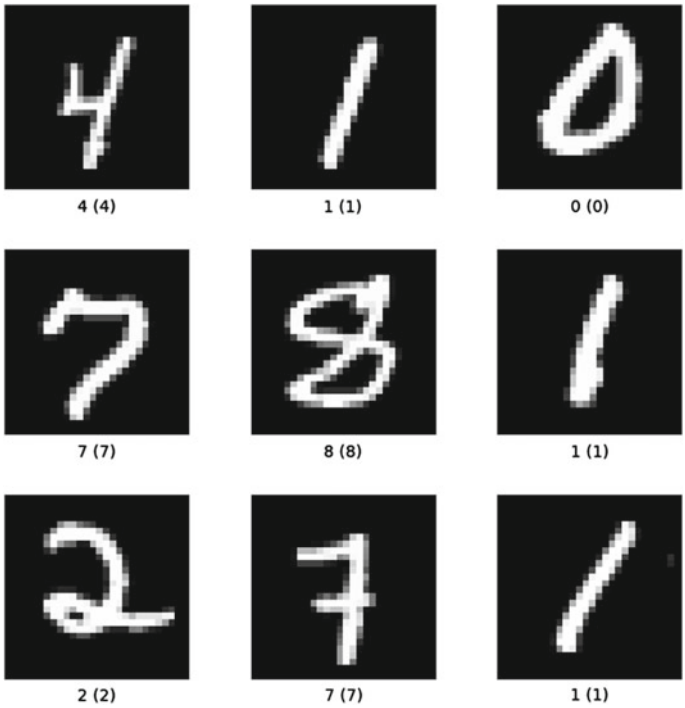


Fig. 2.8 Examples of MNIST dataset

2.3.4 Performance Metric

Speed up rate: Compared with conventional analytical or numerical based approaches for solving computational electromagnetics problems, deep learning method with its inherent working scheme and ever-fast parallel computing platform excels at the speed of computation. Therefore, the speed up rate is a pivotal metric for evaluating the performance of deep learning method. It is noted that although the online training of deep neural networks is indeed time-consuming for most of cases, the offline prediction enforced by neural networks is extremely fast since no numerical iterations are involved compared with traditional methods (FDFD, FEM), thus the time consumption of the latter phases is mainly used for comparison. Another detail that should be remind is that to perform the comparison experiment, same set of computing facilities (GPU, CPU, RAM…) should be adopted, or the result remains untrustworthy. To the best of our knowledge, the computational acceleration rate of deep learning method can reach up to two to four orders of magnitude for either 2D or 3D computing domain [10–12, 23].

Accuracy: Compared with analytical or semi-analytical methods, deep learning suffers from a certain degree of degradation of accuracy mainly because this method which is actually based on statistical learning [47] still have limitations on

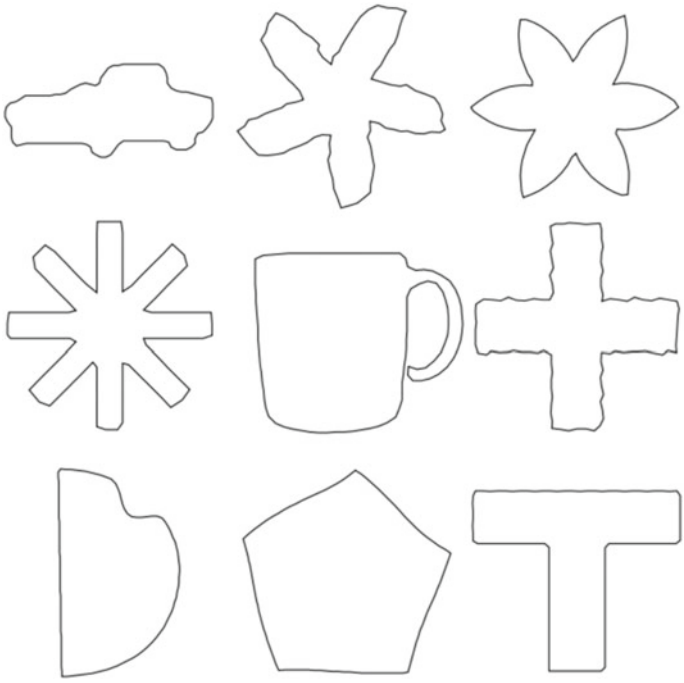


Fig. 2.9 Examples of the 2D shape structure dataset

numerically mimicking different physical scenarios with extremely high complexity. Therefore, the computational accuracy is another important criterion of evaluating proposed neural networks especially for the forward analysis. This metric can be easily evaluated using relative error between the reference results and the neural network prediction. In most cases, mean square error (MSE) is used to denote relative error, which is defined as

$$Error = \frac{1}{N}\sum_{k=1}^{N}\left(Y_k - \hat{Y}_k\right)^2. \tag{2.10}$$

It is also noted that complex output of physical response should be taken care of in terms of real part and imaginary part.

$$Error = \frac{1}{N^2}\sum_{k=1}^{N}\frac{\left(Y_k^r - Y_k^{r\prime}\right)^2 + \left(Y_k^i - Y_k^{i\prime}\right)^2}{Y_k^{r\,2} + Y_k^{i\,2}} \tag{2.11}$$

Generalization ability: It is of high necessity to make sure that the DL method performs analysis by mimicking the underlying physical law, rather than simply overfitting between the training samples. This performance metric is referred to as

generalization ability. A neural network with good generalization ability is able to not only perform interpolation between known training data samples, but also capable of making extrapolation on the test dataset. One of the representative measures to verify this ability is to introduce open-source datasets for validation and testing. The testing accuracy on these datasets may reflect in what extent the DL method could generalize to more complicated physical scenarios.

Others: Besides, for specific problems, such as the inverse design of nanophononics devices, there are some other design criterions to satisfy [48, 49]. Without doubts, all these metrics should be properly considered to evaluate the performance of developed neural networks.

2.4 Summary

In this chapter, we systematically discuss how DL method could be used to solve electromagnetics related problems. Fundamental knowledge of DL basics and related physical background are also discussed. Technical details in data acquisition, neural network training, performance testing is comprehensively reviewed. In the later chapter, we shall see more concrete examples of how these methods could be applied to solve specific problems.

References

1. Schmidhuber J (2015) Deep learning in neural networks: an overview. Neural Netw 61:85–117
2. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
3. Thuruthel TG, Shih B, Laschi C, Tolley MT (2019) Soft robot perception using embedded soft sensors and recurrent neural networks. Sci Robot 4(26)
4. Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L (2009) Imagenet: a large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. IEEE, pp 248–255
5. Chowdhary K (2020) Natural language processing. In: Fundamentals of artificial intelligence. Springer, Berlin, pp 603–649
6. Cong I, Choi S, Lukin MD (2019) Quantum convolutional neural networks. Nat Phys 15(12):1273–1278
7. Schütt K, Gastegger M, Tkatchenko A, Müller K-R, Maurer RJ (2019) Unifying machine learning and quantum chemistry with a deep neural network for molecular wavefunctions. Nat Commun 10(1):1–10
8. Xie T, Grossman JC (2018) Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. Phys Rev Lett 120(14):145301
9. Silver D, Huang A, Maddison CJ, Guez A, Sifre L, Van Den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M (2016) Mastering the game of go with deep neural networks and tree search. Nature 529(7587):484–489
10. Qi S, Wang Y, Li Y, Wu X, Ren Q, Ren Y (2020) 2D electromagnetic solver based on deep learning technique. IEEE J Multiscale Multiphys Comput Tech 5:83–88

11. Shan T, Dang X, Li M, Yang F, Xu S, Wu J (2018) Study on a 3D Poisson's equation solver based on deep learning technique. In: 2018 IEEE international conference on computational electromagnetics (ICCEM). IEEE, pp 1–3
12. Wiecha PR, Muskens OL (2019) Deep learning meets nanophotonics: a generalized accurate predictor for near fields and far fields of arbitrary 3D nanostructures. Nano Lett 20(1):329–338
13. Li Y, Xu Y, Jiang M, Li B, Han T, Chi C, Lin F, Shen B, Zhu X, Lai L (2019) Self-learning perfect optical chirality via a deep neural network. Phys Rev Lett 123(21):213902
14. Peurifoy J, Shen Y, Jing L, Yang Y, Cano-Renteria F, DeLacy BG, Joannopoulos JD, Tegmark M, Soljačić M (2018) Nanophotonic particle simulation and inverse design using artificial neural networks. Sci Adv 4(6):eaar4206
15. Liu D, Tan Y, Khoram E, Yu Z (2018) Training deep neural networks for the inverse design of nanophotonic structures. ACS Photonics 5(4):1365–1369
16. Liu Z, Zhu D, Rodrigues SP, Lee K-T, Cai W (2018) Generative model for the inverse design of metasurfaces. Nano Lett 18(10):6570–6576
17. Malkiel I, Mrejen M, Nagler A, Arieli U, Wolf L, Suchowski H (2018) Plasmonic nanostructure design and characterization via deep learning. Light Sci Appl 7(1):1–8
18. Campbell SD, Sell D, Jenkins RP, Whiting EB, Fan JA, Werner DH (2019) Review of numerical optimization techniques for meta-device design. Opt Mater Express 9(4):1842–1863
19. Jiang J, Fan JA (2019) Global optimization of dielectric metasurfaces using a physics-driven neural network. Nano Lett 19(8):5366–5372
20. Kudyshev ZA, Kildishev AV, Shalaev VM, Boltasseva A (2020) Machine-learning-assisted metasurface design for high-efficiency thermal emitter optimization. Appl Phys Rev 7(2):021407
21. Kiarashinejad Y, Abdollahramezani S, Adibi A (2020) Deep learning approach based on dimensionality reduction for designing electromagnetic nanostructures. npj Comput Mater 6(1):1–12
22. Barth C, Becker C (2018) Machine learning classification for field distributions of photonic modes. Commun Phys 1(1):1–11
23. Li Y, Wang Y, Qi S, Ren Q, Kang L, Campbell SD, Werner PL, Werner DH (2020) Predicting scattering from complex nano-structures via deep learning. IEEE Access 8:139983–139993
24. Zhang G, He H, Katabi D (2019) Circuit-GNN: graph neural networks for distributed circuit design. In: International conference on machine learning, pp 7364–7373
25. Nadell CC, Huang B, Malof JM, Padilla WJ (2019) Deep learning for accelerated all-dielectric metasurface design. Opt Express 27(20):27523–27535
26. Qu Y, Jing L, Shen Y, Qiu M, Soljacic M (2019) Migrating knowledge between physical scenarios based on artificial neural networks. ACS Photonics 6(5):1168–1174
27. Glorot X, Bordes A, Bengio Y (2011) Deep sparse rectifier neural networks. In: Proceedings of the fourteenth international conference on artificial intelligence and statistics, 2011. JMLR workshop and conference proceedings, pp 315–323
28. Maas AL, Hannun AY, Ng AY (2013) Rectifier nonlinearities improve neural network acoustic models. In: Proceedings icml, vol 1. Citeseer, p 3
29. Xu B, Wang N, Chen T, Li M (2015) Empirical evaluation of rectified activations in convolutional network. arXiv preprint arXiv:150500853
30. Özışık MN (1989) Boundary value problems of heat conduction. Courier Corporation, USA
31. Hughes TW, Williamson IA, Minkov M, Fan S (2019) Wave physics as an analog recurrent neural network. Sci Adv 5(12):eaay6946
32. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Advances in neural information processing systems, pp 2672–2680
33. Jiang J, Sell D, Hoyer S, Hickey J, Yang J, Fan JA (2019) Free-form diffractive metagrating design based on generative adversarial networks. ACS Nano 13(8):8872–8878
34. So S, Rho J (2019) Designing nanophotonic structures using conditional deep convolutional generative adversarial networks. Nanophotonics 8(7):1255–1261

35. Ye X, Bai Y, Song R, Xu K, An J (2020) An inhomogeneous background imaging method based on generative adversarial network. IEEE Trans Microwave Theor Tech 68(11):4684–4693
36. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: towards real-time object detection with region proposal networks. arXiv preprint arXiv:150601497
37. Hinton G, Vinyals O, Dean J (2015) Distilling the knowledge in a neural network. arXiv preprint arXiv:150302531
38. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:14126980
39. Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics, pp 249–256
40. He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision, pp 1026–1034
41. Snoek J, Larochelle H, Adams RP (2012) Practical Bayesian optimization of machine learning algorithms. In: Advances in neural information processing systems, pp 2951–2959
42. Khan A, Ghorbanian V, Lowther D (2019) Deep learning for magnetic field estimation. IEEE Trans Magn 55(6):1–4
43. Taflove A, Hagness SC (2000) Computational electromagnetics: the finite-difference time-domain method. Artech House, USA
44. Shin W, Fan S (2012) Choice of the perfectly matched layer boundary condition for frequency-domain Maxwell's equations solvers. J Comput Phys 231(8):3406–3431
45. Deng L (2012) The mnist database of handwritten digit images for machine learning research [best of the web]. IEEE Signal Process Mag 29(6):141–142
46. Carlier A, Leonard K, Hahmann S, Morin G, Collins M (2016) The 2d shape structure dataset: a user annotated open access database. Comput Graph 58:23–30
47. Vapnik V (2013) The nature of statistical learning theory. Springer Science & Business Media, Berlin
48. Jiang J, Chen M, Fan JA (2020) Deep neural networks for the evaluation and design of photonic devices. Nat Rev Mater 1–22
49. Ma W, Liu Z, Kudyshev ZA, Boltasseva A, Cai W, Liu Y (2020) Deep learning for the design of photonic structures. Nat Photonics 1–14

Chapter 3
Building Database

One of the most critical components in machine learning projects is the database. In the training process, it will provide information to the network, so that the network can adapt the inner parameter automatically according to the information. In the testing process, data will indicate the degree of network training. So, appropriate database is of great importance to a deep learning problem. In some cases, public database, such as 'cifar' ([1] https://www.cs.toronto.edu/~kriz/cifar.html) for computer vision and 'Big Bad NLP Database' ([2] https://datasets.quantumstat.com) for natural language process, will release the burden of data for these problems. But in the area of computational physics, there is no such public database available. So, it is necessary to build a database for training and test.

In this work, an end-to-end CNN is built to predict the EM field. The input data are pictures that present the information of scatters and the characteristics of incident wave. While the outputs are the real and imaginary part of the predicted EM field. So, the unique database for this work will contain the information for training the network and the ground truth for testing the result.

Building the dataset for training EM-net, a traditional computational electromagnetics (CEM) program is needed to generate the real data of EM-field. They will function as the ground truth in the training and testing process. Besides, this traditional method should be wrapped by a random geometry generator to fulfill the diversity and universality of the data. After generating a large amount of data, the input data and the ground truth should be encapsulated into a specific format so that they can be read by the network.

In the following, the traditional CEM method used in this work is specified and how the random geometry generator work is also clarified. Then, the format of dataset is stated. At last, we conclude all the work in building the dataset.

Q. Ren et al., *Sophisticated Electromagnetic Forward Scattering Solver via Deep Learning*, https://doi.org/10.1007/978-981-16-6261-4_3

3.1 FDFD Method

Traditional computational technique is used to produce the ground truth in our work. The classical computational electromagnetic methods include Finite-Difference Time-Domain (FDTD) method, which was originally proposed by Kane S. Yee in the seminal paper he published in 1966 [3], Finite-Difference Frequency-Domain (FDFD) method, which is also based on Finite-Difference approximations of the derivative operators in the differential equation being solved, method of moments (MoM) [4] and finite element method (FEM). Among these methods, finite-difference frequency-domain (FDFD) is selected to calculate the EM field scattered by random scatters under plane wave illumination. Though FDFD is not so popular as FDTD, they are complementary tools for analyzing EM phenomena and FDFD is more suitable for frequency domain analysis and treating with dispersive materials.

Maxwell's equations illustrate how electric field and magnetic field interact and govern electromagnetics (EM) phenomena. In the time domain, they are

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}, \tag{3.1}$$

$$\nabla \times \mathbf{H} = \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t}, \tag{3.2}$$

$$\nabla \cdot \mathbf{D} = \rho, \tag{3.3}$$

$$\nabla \cdot \mathbf{B} = 0. \tag{3.4}$$

where $\mathbf{E}$ and $\mathbf{H}$ are the electric and magnetic fields (abbreviation for E-fields and H-fields in the following); $\mathbf{J}$ is the electric current source densities; According to the constitutive equations, $\mathbf{D} = \varepsilon\mathbf{E}$ and $\mathbf{B} = \mu\mathbf{H}$, ε and μ are the electric permittivity and magnetic permeability.

For a specific problem, E-fields and H-fields are functions of position $\mathbf{r}$ and time t. In this way, Faraday's law of induction and Ampere's circuital law are

$$\nabla \times \mathbf{E}(\mathbf{r}, t) = -\partial_t \mathbf{B}(\mathbf{r}, t), \tag{3.5}$$

$$\nabla \times \mathbf{H}(\mathbf{r}, t) = \partial_t \mathbf{D}(\mathbf{r}, t) + \mathbf{J}(\mathbf{r}, t). \tag{3.6}$$

The frequency-domain Maxwell's equations can be derived from these time-domain Maxwell's equations. For a given ω, by assuming a time dependence of $e^{j\omega t}$ in each time-dependent quantity $\mathcal{F}(\mathbf{r}, t)$ to have $\mathcal{F}(\mathbf{r}, t) = \mathbf{F}(\mathbf{r}, \omega)e^{j\omega t}$ and then using the constitutive equations $\mathbf{D} = \varepsilon\mathbf{E}$ and $\mathbf{B} = \mu\mathbf{H}$ to replace $\mathbf{D}$ and $\mathbf{B}$. In the frequency domain, the differential expression of Faraday's law of induction and Ampere's circuital law are

$$\nabla \times \mathbf{E}(\mathbf{r}, \omega) = -j\omega\mu(\mathbf{r}, \omega)\mathbf{H}(\mathbf{r}, \omega), \tag{3.7}$$

$$\nabla \times \mathbf{H}(\mathbf{r}, \omega) = j\omega\varepsilon(\mathbf{r}, \omega)\mathbf{E}(\mathbf{r}, \omega) + \mathbf{J}(\mathbf{r}, \omega). \tag{3.8}$$

When using FDFD method to solve the frequency-domain Maxwell's Eqs. (3.7–3.8) for the E- and H-fields for given current source densities $\mathbf{J}$, there are a few different combinations of equations to choose. First, it is intuitive to think of solving Eq. (3.7–3.8) directly for the E- and H-fields simultaneously. This choice is equivalent to solve

$$\begin{bmatrix} -j\omega\varepsilon & \nabla\times \\ \nabla\times & j\omega\mu \end{bmatrix} \begin{bmatrix} \mathbf{E} \\ \mathbf{H} \end{bmatrix} = \begin{bmatrix} \mathbf{J} \\ \mathbf{0} \end{bmatrix}. \tag{3.9}$$

An alternative is to eliminate either the E- or H-field from Eq. (3.3) to obtain

$$\nabla \times \mu^{-1}\nabla \times \mathbf{E} - \omega^2\varepsilon\mathbf{E} = -j\omega\mathbf{J}, \tag{3.10}$$

or

$$\nabla \times \varepsilon^{-1}\nabla \times \mathbf{H} - \omega^2\mu\mathbf{H} = \nabla \times \varepsilon^{-1}\mathbf{J}. \tag{3.11}$$

In this way, only $\mathbf{E}$-fields or $\mathbf{H}$-fields exist in one equation. When solving the EM-field through this formulation, the size of the solution vector is halved from that of Eq. (3.4), which benefits the solving process further. Once either Eq. (3.10) or Eq. (3.11) is solved for one field, the other field can be easily recovered by simple substitution of the solved field into Eq. (3.1).

In nanophotonic systems, solving Eq. (3.10) is a better preference since it avoids huge differences in the order of magnitude in the equation. In nanophotonics, the second term of the left-hand side of Eq. (3.10) is usually much smaller than the first term, i.e., $\left|\omega^2\varepsilon\mathbf{E}\right| \ll \left|\nabla \times \mu^{-1}\nabla \times \mathbf{E}\right|$, because nanophotonic objects are much smaller than a wavelength. Therefore, the operator of Eq. (3.10) can be well approximated by $\nabla \times \mu^{-1}\nabla\times$, because $\mu = \mu_0$ holds for most materials used in nanophotonic devices. In addition, this operator is Hermitian positive-semidefinite, which is very favorable to numerical solvers. Although the second term of the left-hand side of Eq. (3.11) is also ignorable for the same reason, the operator of the first term is neither Hermitian nor positive-semidefinite, because metals are lossy medium, which have complex ε and its real part is negative.

When written in Cartesian coordinate system, the frequency-domain Maxwell's equations (3.7–3.8) are

$$\partial_y E_z - \partial_z E_y = -j\omega\mu H_x, \tag{3.12}$$

$$\partial_z E_x - \partial_x E_z = -j\omega\mu H_y, \tag{3.13}$$

$$\partial_x E_y - \partial_y E_x = -j\omega\mu H_z, \tag{3.14}$$

and

$$\partial_y H_z - \partial_z H_y = j\omega\varepsilon E_x + J_x, \tag{3.15}$$

$$\partial_z H_x - \partial_x H_z = j\omega\varepsilon E_y + J_y, \tag{3.16}$$

$$\partial_x H_y - \partial_y H_x = j\omega\varepsilon E_z + J_z. \tag{3.17}$$

To solve Eqs. (3.12) to (3.17) numerically by the finite-difference method, each derivative can be approximated by a ratio between finite differences as

$$\frac{E_z^{i,j+1,k} - E_z^{i,j,k}}{\Delta_y^j} - \frac{E_y^{i,j,k+1} - E_y^{i,j,k}}{\Delta_z^k} = -j\omega\mu_x^{i,j,k} H_x^{i,j,k}, \tag{3.18}$$

$$\frac{E_x^{i,j,k+1} - E_x^{i,j,k}}{\Delta_z^k} - \frac{E_z^{i+1,j,k} - E_z^{i,j,k}}{\Delta_x^i} = -j\omega\mu_y^{i,j,k} H_y^{i,j,k}, \tag{3.19}$$

$$\frac{E_y^{i+1,j,k} - E_y^{i,j,k}}{\Delta_x^i} - \frac{E_x^{i,j+1,k} - E_x^{i,j,k}}{\Delta_y^j} = -j\omega\mu_z^{i,j,k} H_z^{i,j,k}, \tag{3.20}$$

and

$$\frac{H_z^{i,j,k} - H_z^{i,j-1,k}}{\tilde{\Delta}_y^j} - \frac{H_y^{i,j,k} - E_y^{i,j,k-1}}{\tilde{\Delta}_z^k} = j\omega\varepsilon_x^{i,j,k} E_x^{i,j,k} + J_x^{i,j,k}, \tag{3.21}$$

$$\frac{H_x^{i,j,k} - E_x^{i,j,k-1}}{\tilde{\Delta}_z^k} - \frac{H_z^{i,j,k} - H_z^{i-1,j,k}}{\tilde{\Delta}_x^i} = j\omega\varepsilon_y^{i,j,k} E_x^{i,j,k} + J_y^{i,j,k}, \tag{3.22}$$

$$\frac{H_y^{i,j,k} - H_y^{i-1,j,k}}{\tilde{\Delta}_x^i} - \frac{H_x^{i,j,k} - H_x^{i,j-1,k}}{\tilde{\Delta}_y^j} = j\omega\varepsilon_z^{i,j,k} E_z^{i,j,k} + J_z^{i,j,k}, \tag{3.23}$$

where $\varepsilon_w^{i,j,k}$ and $\mu_w^{i,j,k}$ are the permittivity and permeability of the cell, respectively, and $\Delta_w^l = \left(\Delta_w^l + \Delta_w^{l-1}\right)/2$, $(w = x, y, z)$.

The differential equations from all cells are collected to build the system equations,

$$\mathbf{C}_e\mathbf{e} = -j\omega\mathbf{D}_\mu\mathbf{h}, \tag{3.24}$$

$$\mathbf{C}_h\mathbf{h} = j\omega\mathbf{D}_\varepsilon\mathbf{e} + \mathbf{j}, \tag{3.25}$$

where

$$\mathbf{e} = \left[\cdots E_x^{i,j,k} E_y^{i,j,k} E_z^{i,j,k} \cdots\right]^T, \tag{3.26}$$

$$\mathbf{h} = \left[\cdots H_x^{i,j,k} H_y^{i,j,k} H_z^{i,j,k} \cdots\right]^T, \tag{3.27}$$

$$\mathbf{m} = \left[\cdots M_x^{i,j,k} M_y^{i,j,k} M_z^{i,j,k} \cdots\right]^T, \tag{3.28}$$

$$\mathbf{j} = \left[\cdots J_x^{i,j,k} J_y^{i,j,k} J_z^{i,j,k} \cdots\right]^T, \tag{3.29}$$

are column vectors representing the relevant fields and source; $\mathbf{C}_e$ and $\mathbf{C}_h$ are matrices for the curl operators on $\mathbf{E}$ and $\mathbf{H}$, respectively. $\mathbf{D}_\mu$ and $\mathbf{D}_\varepsilon$ are diagonal matrices for the permeability and permittivity.

After eliminating $\mathbf{e}$, the system of equations with single field unknown $\mathbf{h}$ is as below

$$\left(\mathbf{C}_h - \omega^2 \mathbf{D}_\varepsilon \mathbf{C}_e^{-1} \mathbf{D}_\mu\right)\mathbf{h} = \mathbf{j}, \tag{3.30}$$

which is simply a system of linear equation

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{3.31}$$

where $\mathbf{A}$ represents the operator, $\mathbf{x}$ represents the E-field we solve for, and $\mathbf{b}$ is a column vector determined by a given electric current source density.

Methods used to solve Eq. (3.31) can be divided into two categories: direct methods and iterative methods. Direct methods factorize $\mathbf{A}$ into a few (typically two or three) factors with which $\mathbf{A}^{-1}\mathbf{b}$ can be calculated efficiently, and they produce a solution in a fixed number of steps. Depending on the structures and properties of $\mathbf{A}$, different factorization methods such as the Cholesky, LU, LDM^T, LDL^T factorizations are used [5].

The other category of methods are iterative methods, which produce an approximate solution at each iteration step until the solution converges sufficiently close to the exact solution [6]. More specifically, suppose that $\mathbf{x}_m$ is the approximate solution produced at the mth iteration step. Then iterative methods continue the process until the residual vector

$$\mathbf{r}_m = \mathbf{b} - \mathbf{A}\mathbf{x}_m \tag{3.32}$$

satisfies $\|\mathbf{r}_m\| / \|\mathbf{b}\| < \tau$, where $\|\cdot\|$ is a norm of a column vector and τ is a user-defined small positive number; typically, the 2-norm is used as the norm, but some iterative methods, e.g., the conjugate gradient method, use different norms such as the $\mathbf{A}$-norm, and in practice $\tau = 10^{-6}$ is sufficiently small for accurate solutions. Iterative methods do not guarantee convergence within a fixed number of iteration steps, but they often produce accurate solutions using less computational resources than direct methods.

The matrix $\mathbf{A}$ of Eq. (3.32) constructed by the finite-difference method is typically very large (often with more than 10 million rows and columns for 3D problems) but extremely sparse (with at most 13 nonzero elements per row). For such an extremely large and sparse matrix, iterative methods are usually preferred to direct methods, because direct methods require too much computer memory.

Among many kinds of iterative methods, we use Krylov subspace methods [7], which are known as one of the most efficient class of iterative methods. The Krylov subspace of dimension m generated by $\mathbf{A}$ and $\mathbf{r}_0$ is

$$\mathcal{K}_m(\mathbf{A}, \mathbf{r}_0) = span\left\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \mathbf{A}^2\mathbf{r}_0, \ldots, \mathbf{A}^{m-1}\mathbf{r}_0\right\}, \tag{3.33}$$

where $\mathbf{r}_0$ is the initial residual vector of Eq. (3.33) for an initial guess solution x_0. Krylov subspace methods find the "best" mth approximate solution $\mathbf{x}_m$ of Eq. (3.32) in the space $|x_0 + \mathcal{K}_m(\mathbf{A}, \mathbf{r}_0)|$. Each Krylov subspace method is distinguished from others by its own criterion for determining the best approximate solution. In general, however, all Krylov subspace methods find better and better approximate solutions as m increases because the search space becomes larger and larger, i.e., $|x_0 + \mathcal{K}_m(\mathbf{A}, \mathbf{r}_0) \subseteq x_0 + \mathcal{K}_{m+1}(\mathbf{A}, \mathbf{r}_0)|$.

Like direct methods, there are Krylov subspace methods specialized for matrices with specific structures and properties, such as real-symmetric, complex-Hermitian, or positive-definite matrices. Unfortunately, our matrix constructed from Maxwell's equations does not satisfy any of these properties: it is complex, nonsymmetric, and indefinite. Therefore, we need to rely on Krylov subspace methods that can handle the most generic matrices.

When producing the EM-field dataset, Krylov subspace methods for generic matrices are used. It uses 2-norm in testing $\|\mathbf{r}_m\|/\|\mathbf{b}\| < \tau$. We use $x_0 = 0$ as an initial guess to create subspaces of Eq. (3.17).

Except from applying an appropriate method to solve the matrix equation, it is also important to choose the perfectly matched layer (PML) for the FDFD method. Dr. Shin elaborated this process in his paper [8]. Since the choice of PML is not so relevant to the topic of this article, it will not be discussed in detail.

3.2 Random Geometry Generator

After establishing the algorithm to calculate the EM-field, the next step is to determine the configuration of scatters. In order to promise good generalization ability of the network, the dataset used to train the model should be various. Besides, to meet practical needs, the shape of scatters should be common. Therefore, the scatters include circles, ellipses, polygons with different sizes, and their combinations in the 2-D experiments and various spheres, spheroids in the 3-D experiments. In this part, the approach for generation of these scatters is illustrated.

3.2.1 2-D Geometry Generator

In the 2-D experiment, the domain area is a square with the edge length of 128 nm shown in Fig. 3.1.

It is straightforward to think of how to generate circles and ellipses through program. According to the definition, given the center point and radius, a circle can be generated directly. In our program, the radius of the circle ranges from 13 to 28 nm while the center of the circle ranges from (32, 32) nm to (96, 96) nm, which is shown in Fig. 3.2.

Some of the examples are shown in Fig. 3.3.

In the same way, having semimajor axis, eccentricity and rotation angle, ellipses can be generated. In the generation program, the semimajor axis of the ellipse is ranging from 19 to 26 nm with the eccentricity ranging from 0.65 to 0.95 while the center of the circle ranges from (32, 32) nm to (96, 96) nm. The rotation angel is ranging from 0 to π. How the ellipse is generated is shown in Fig. 3.4.

Here are some examples of generated ellipses as illustrated in Fig. 3.5.

Unlike generating circles or ellipses, generating polygons through program is not so intuitive. To realize the generation of random polygons, we need to know the concept of cross product. The following is the logic to generate random polygon scatters.

Firstly, determine the number of edges $\boldsymbol{N}$ of the polygons. Secondly, determine the vertex of the polygon. Set a center point $\boldsymbol{O}$ in the domain area and generate $\boldsymbol{N}$ chords and $\boldsymbol{N}$ angles randomly. Sort these $\boldsymbol{N}$ angles and assign each angle a chord. In this way, $\boldsymbol{N}$ vertexes in the computational domain are set in clockwise or counterclockwise order. Define them as $P_1, P_2, P_3, \ldots, P_N$. Thirdly, traverse every

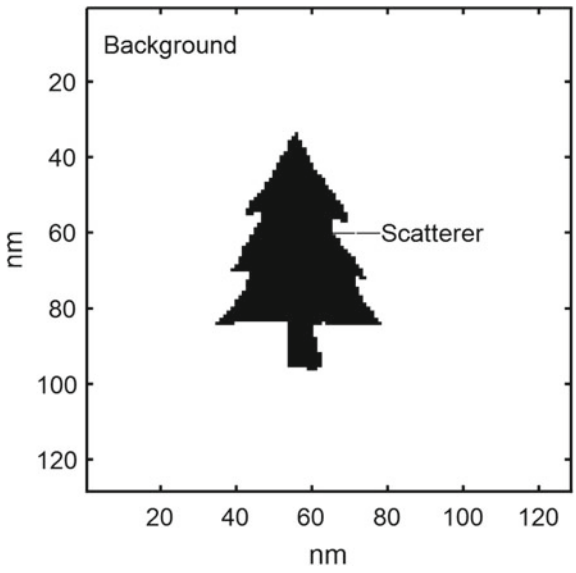


Fig. 3.1 Domain area of 2D experiment

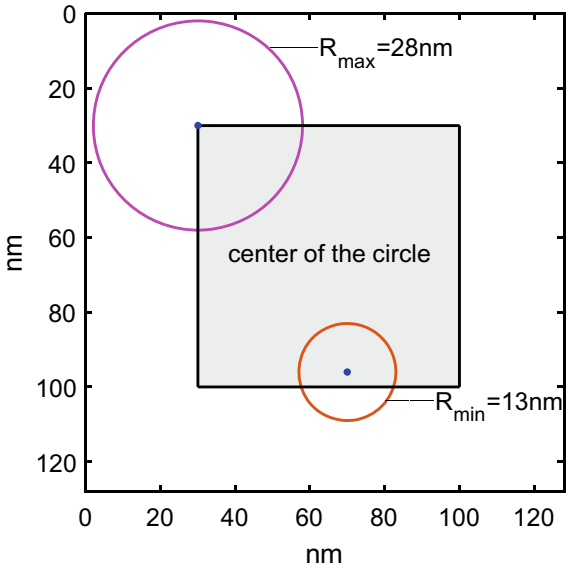


Fig. 3.2 Principle of circle generation

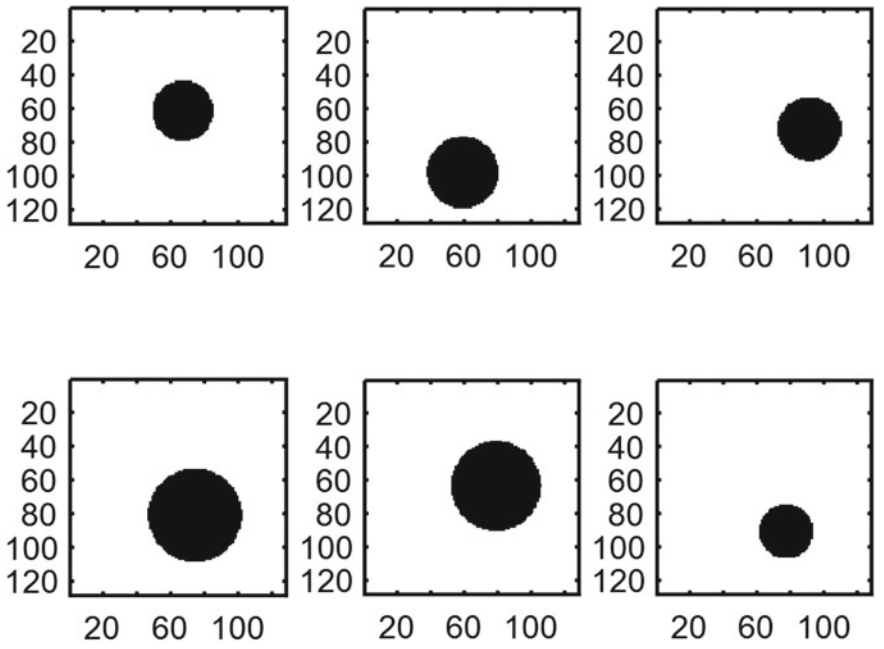


Fig. 3.3 Samples of the generated circles

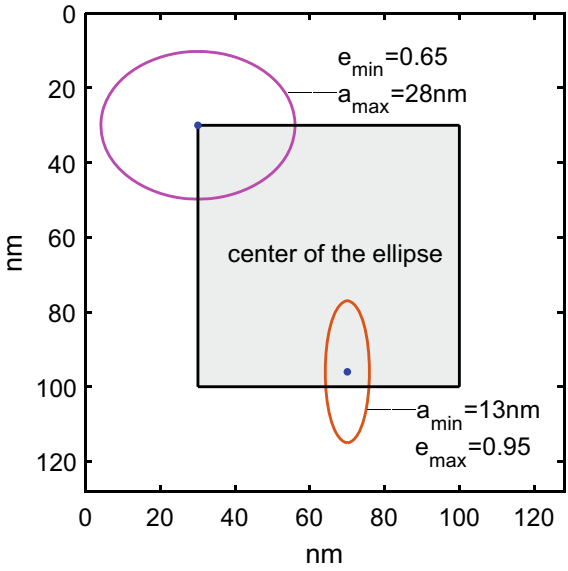


Fig. 3.4 Principle of ellipse generation

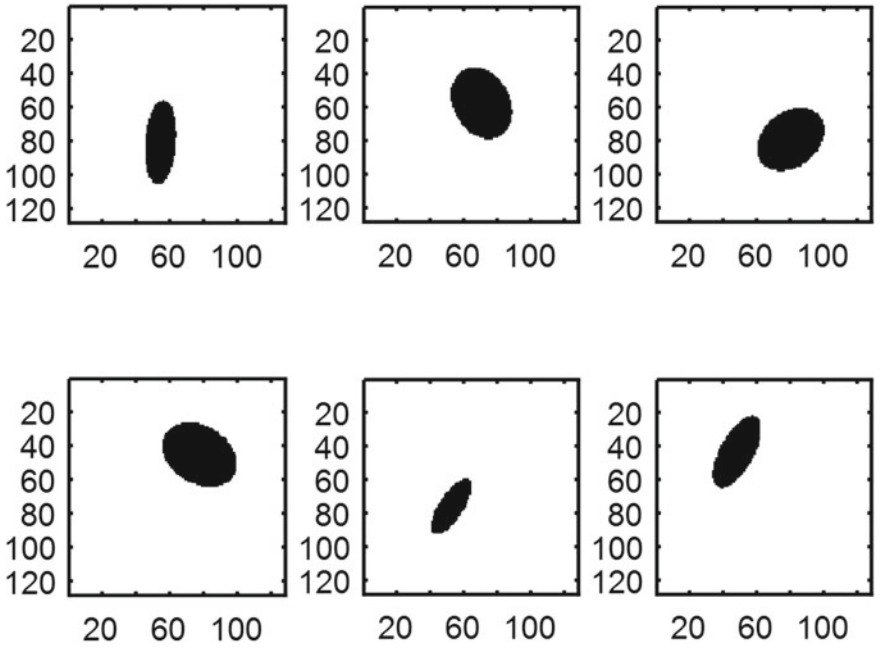


Fig. 3.5 Samples of ellipses

point in the generating domain. Compute the cross product of the vectors from the point and the vertex, e. g. for a random point $\boldsymbol{S}$ in the generating domain, compute the cross products of $\lceil SP_1, SP_2 \rceil$, $\lceil SP_2, SP_3 \rceil$, …, $\lceil SP_{N-1}, SP_N \rceil$, $\lceil SP_N, SP_1 \rceil$. If the all the results have the same sign, this point is inside this polygon, or vise versa.

Now that all the points have been traversed, the figure will be generated. This algorithm has the same complexity as the one used to generate circles or ellipses. Here, Algorithm 1 demonstrates the details of the algorithm.

If a regular polygon is needed, only the second step, determine the vertex of the polygon should be specified. When determining the vertex, every chord should be the same.

Algorithm 1 Polygons Generation

Algorithm 1: Polygons generation algorithm

Input: Number of vertex (N), length of each chord (L), angle of corresponding chord (θ) and length of figure (C)

Output: An figure contains one polygon (F)

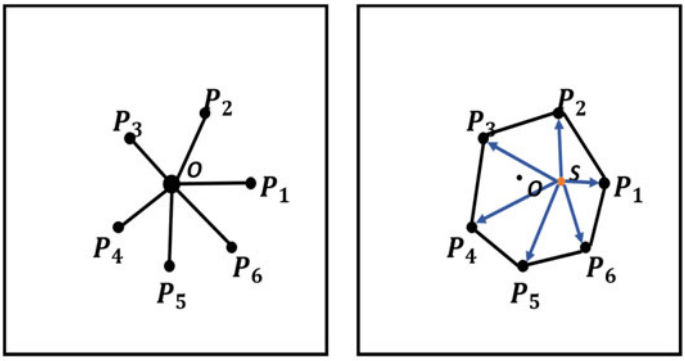
$F \leftarrow 0$;
for $k = 0$ to $N - 1$ do
  $V(k,0) = O(0) + L(k)cos(\theta(k,0))$ /*V is the vertex and O represents the center point*/
  $V(k,1) = O(1) + L(k)sin(\theta(k,1))$
end
for $i = 0$ to $C - 1$ do
  for $j = 0$ to $C - 1$ do
    $cnt = 0$
    for $k = 0$ to $N - 1$ do
      $temp0(0) = i - V(k,0)$
      $temp0(1) = i - V(k,1)$
      if $k == N - 1$ then
        $k = -1$
      end
      $temp1(0) = j - V(k+1,0)$
      $temp1(1) = j - V(k+1,1)$
      if $temp0 \times temp1 > 0$ then
        $cnt = cnt + 1$
      end
      if $cnt == 0$ or $cnt == N$ then
        $F(i,j) = 1$
      end
    end
  end
end

Figures 3.6 and 3.7 show how the polygons are generated.

The edge of polygons ranges from 32 to 64 nm with the eccentricity ranging from 0.65 to 0.95 while the center of the circle ranges from (30, 30) nm to (100, 100) nm.

Some examples of random polygons are shown in Fig. 3.8.

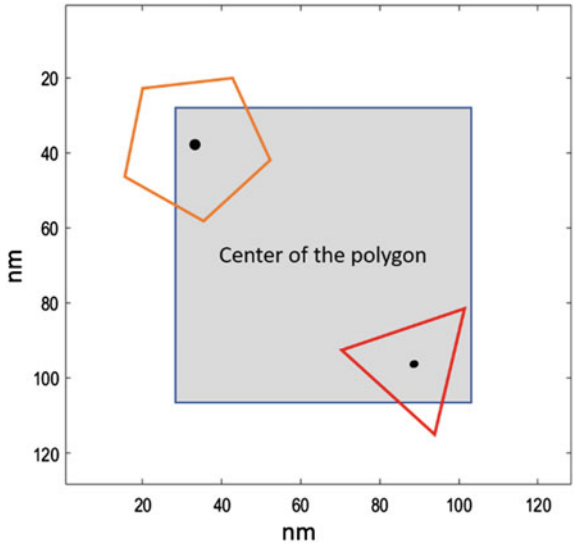
This method can only be used to generate convex polygons. For concave polygons, they can be realized by the combination of multiple convex polygons.



(a) Set vertexes of polygons (b) Traverse every point in the domain

Fig. 3.6 Schematic of polygons generation

Fig. 3.7 Principle of polygons generation



3.2.2 3-D Geometry Generator

In the 3-D experiment, to reduce the burden of calculation, the domain area is set to be a cube with the edge length of 32 nm as Fig. 3.9 shows.

The center of spheres is in the area of (7, 25) nm in all three directions and their radius range from 7 to 15 nm. Also, the center of spheroids is in the area of (7, 25) nm in all three directions and their three semiaxis range from 7 to 15 nm. Figure 3.10 shows some examples of the spheres and spheroids.

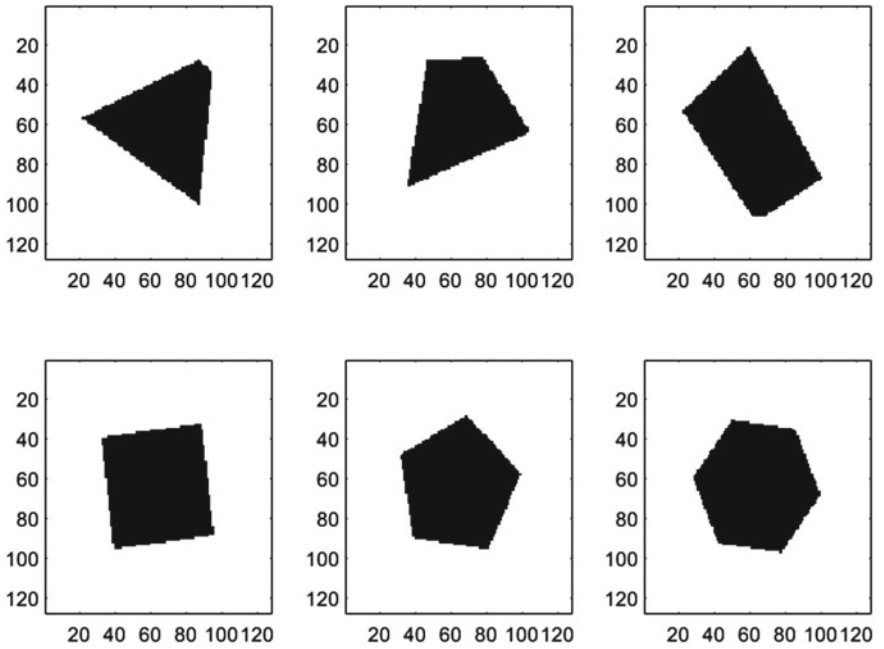


Fig. 3.8 Samples of polygons

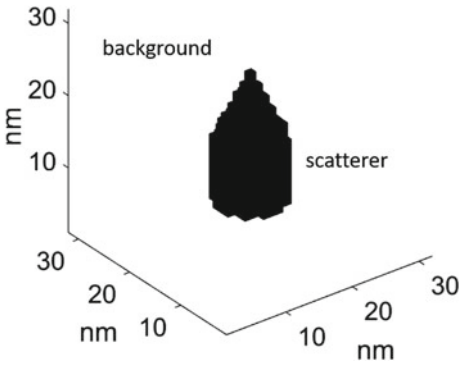


Fig. 3.9 Domain area of 2D experiment

3.3 Data Validation

Combining the random scatter generation and FDFD program, data of scatter field can be calculated automatically. However, before building dataset, we need to validate the generated EM-field.

In this work, the FDFD program is validated by using both the analytical method (series expansion) [9–13] and the finite element method (COMSOL Multiphysics). To be specific, we use FDFD program and other methods to calculate the same

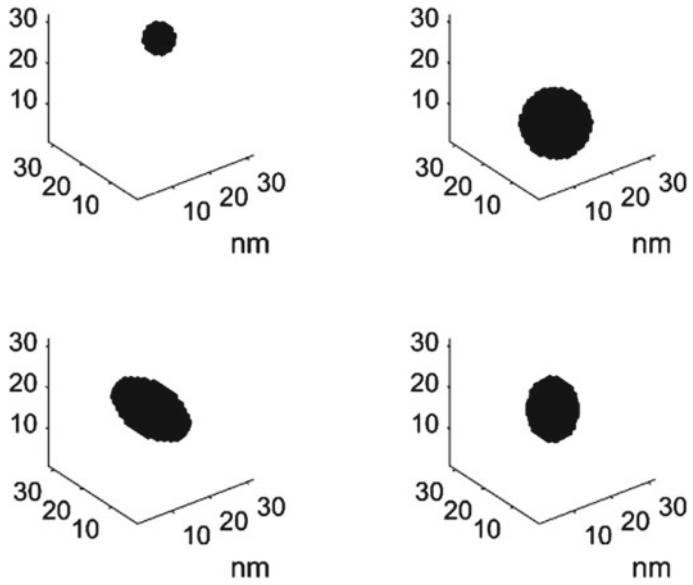


Fig. 3.10 Samples of 3D scatters

case and compare the difference in each pixel. Also, statistical discrepancies will be presented to verify the accuracy of the data from FDFD program.

3.3.1 Analytical Solution

Among all the calculation methods, only the analytical solution is absolutely accurate, so it is often used as the criterion of other methods. For most practical problems, analytic solutions do not exist. Therefore, we only consider the solutions of an infinitely long cylinder (2-D) and a sphere (3-D).

For the two-dimensional case, we compare the results of the FDFD program and the analytical method in calculating the scattering fields of plane waves by an infinitely long cylindrical medium [14, 15]. The calculation diagram is displayed in Fig. 3.11.

Our first example is a cylinder located at the origin with the radius of 15 nm, and its relative permeability is 2. The region to be calculated is a square with a side length of 128 nm. The plane wave is a TEz polarized wave, propagating along the $+x$ direction whose wavelength is 80 nm. In the first chapter, we have described the theory of plane waves scattering by dielectrics. For TEz waves, only the z component of the magnetic field is required to represent the other components of the electromagnetic fields. Here, in order to simplify the calculation, we only calculate the magnetic field in z direction. Figure 3.12a shows the schematic diagram of the scatterer and

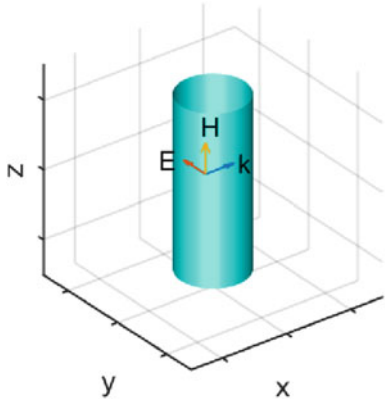


Fig. 3.11 Scattering of the plane wave by a perfect dielectric cylinder with infinite length

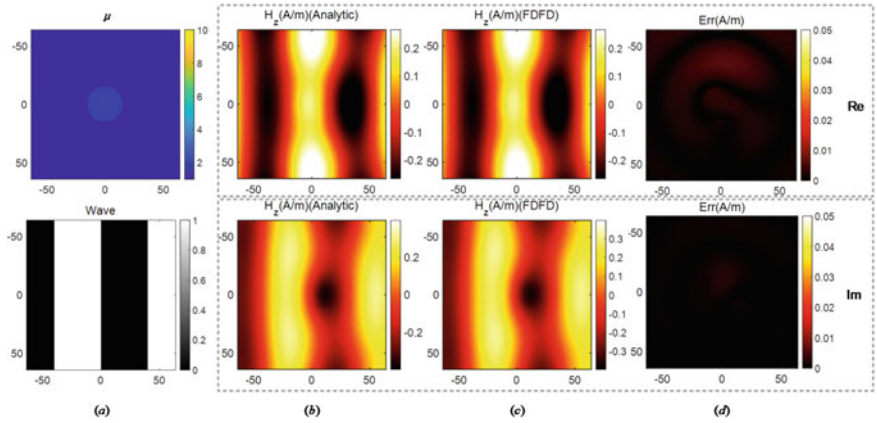


Fig. 3.12 The comparison between the analytical solution and the FDFD solution of a cylinder (r = 15 nm). **a** Permeability distribution and incident electromagnetic wave. Real part and imaginary part of **b** the analytical solution; **c** the FDFD result and **d** the error

the incident electromagnetic wave respectively. Let the amplitude of the incident electric field be 100 V/m, the magnetic field can be expressed as

$$H_z^i = H_0 e^{-jkx} = H_0 \sum_{n=-\infty}^{+\infty} j^{-n} J_n(k\rho) e^{jn\varphi}, \tag{3.34}$$

where $H_0 = 100/\eta_0$. The total field can be expressed as

$$H_z = \begin{cases} H_0 \sum\limits_{n=-\infty}^{+\infty} j^{-n} b_n J_n(k\rho) e^{jn\varphi}, & \rho \le a \\ H_0 \sum\limits_{n=-\infty}^{+\infty} j^{-n} \left[a_n H_n^{(2)}(k\rho) + J_n(k\rho) \right] e^{jn\varphi}, & \rho > a \end{cases}. \tag{3.35}$$

The coefficients can be obtained as

$$a_n = \frac{\sqrt{\mu_1} J_n(ka) J_n'(k_1 a) - \sqrt{\mu} J_n(k_1 a) J_n'(ka)}{\sqrt{\mu} J_n(k_1 a) H_n^{(2)\prime}(ka) - \sqrt{\mu_1} J_n'(k_1 a) H_n^{(2)}(ka)}, \tag{3.36}$$

$$b_n = \frac{J_n(ka) + a_n H_n^{(2)}(ka)}{J_n(k_1 a)}. \tag{3.37}$$

The physical quantities satisfy

$$\begin{cases} ka = \frac{3\pi}{8} \\ k_1 a = \frac{3\sqrt{2}\pi}{8} \end{cases}, \tag{3.38}$$

$$\begin{cases} \mu = \mu_0 \\ \mu_1 = \sqrt{2}\mu_0 \end{cases}. \tag{3.39}$$

Take the truncation term as $n = 30$ (i. e. $\sum_{n=-30}^{30}(\cdot)$), so we can obtain the corresponding analytical solution. The real part and imaginary part are shown in Fig. 3.12b. We also display the real part and imaginary parts calculated by the FDFD program in Fig. 3.12c. Figure 3.12d shows the error, which is defined as the absolute value of the difference between the two as

$$H_{err} = \left| H_{FDFD} - H_{Analytic} \right|. \tag{3.40}$$

It can be seen from the calculation results that the scattering field calculated by the FDFD algorithm is remarkably close to the analytic solution, and the error is negligible. In order to quantitatively compute the error value, we define the relative error and the average relative error as

$$Error(i, j) = \frac{\sqrt{\left(H_{r_{FDFD}}(i, j) - H_{r_{Analytic}}(i, j)\right)^2 + \left(H_{i_{FDFD}}(i, j) - H_{i_{Analytic}}(i, j)\right)^2}}{\sqrt{H_{r_{Analytic}}^2(i, j) + H_{i_{Analytic}}^2(i, j)}}, \tag{3.41}$$

$$Err_{ave} = \frac{\sum_{i=1}^{128} \sum_{j=1}^{128} Error(i, j)}{128^2}. \tag{3.42}$$

According to the above definition, the average relative error in this case is 2.82%. Considering the accuracy of the difference algorithm, the result is satisfactory. In order to further verify the rationality of the algorithm, we test another target.

The second case is a cylinder located at the origin with a radius of 25 nm whose relative permeability is 5. The plane wave is a TEz polarized wave, propagating along the $+x$ direction whose wavelength is 40 nm. The magnetic field in z direction is calculated by FDFD and analytical method respectively. Figure 3.13a shows the

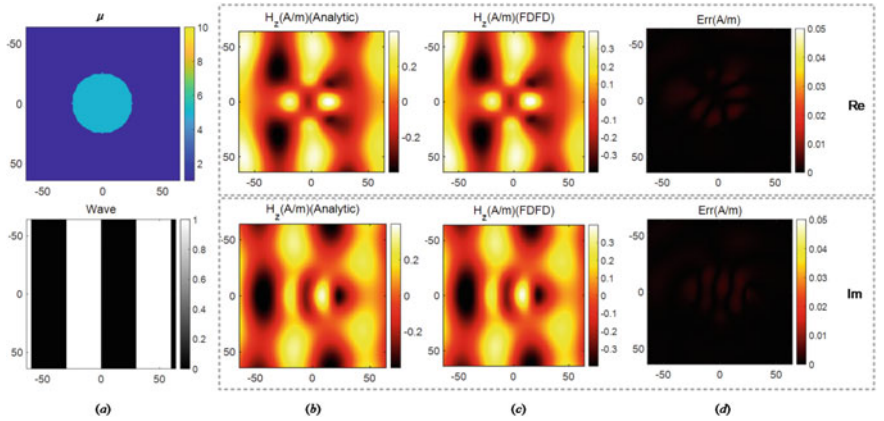


Fig. 3.13 The comparison between the analytical solution and the FDFD solution of a cylinder (r = 25 nm). **a** Permeability distribution and incident electromagnetic wave, **b** real part and imaginary part of the analytical solution, **c** real part and imaginary part of the FDFD result, **d** real part and imaginary part of the error

schematic diagram of the scatterer and the incident electromagnetic wave respectively, in which the amplitude of the incident electric field is 100 V/m, and the physical quantities satisfy

$$\begin{cases} ka = \frac{3\pi}{8} \\ k_1 a = \frac{3\sqrt{2}\pi}{8} \end{cases}, \tag{3.43}$$

$$\begin{cases} \mu = \mu_0 \\ \mu_1 = \sqrt{5}\mu_0 \end{cases}, \tag{3.44}$$

According to the above definition, the average relative error in this case is 2.31%. Therefore, our FDFD forward program can calculate the electromagnetic field of the scatterer with high precision. Since the target scattering field of deep learning is generated by FDFD code, the high-precision forward program ensures the accuracy of the training data set.

In addition to verify the correctness of the two-dimensional program, we also need to verify the three-dimensional program. For this reason, we need to use the analytic formula of the scattering of plane waves by a sphere. Assuming that the plane wave propagates along the z direction, the wave number in the vacuum is k, which can be expressed as

$$\mathbf{E}^i = \mathbf{i}_x E_0 e^{-jkz} = \mathbf{i}_x E_0 e^{-jkr\cos\theta}, \tag{3.45}$$

$$\mathbf{H}^i = \mathbf{i}_y \frac{E_0}{\eta_0} e^{-jkz} = \mathbf{i}_y \frac{E_0}{\eta_0} e^{-jkr\cos\theta}. \tag{3.46}$$

A perfect conducting sphere is placed at the origin with a radius of a. The relationship between the wave vector, polarization direction and the geometric position of the sphere is shown in Fig. 3.14.

In order to obtain the analytical solution of electromagnetic field in spherical coordinate systems, we usually compute the electromagnetic vector potential first, and then derive the electromagnetic field from the vector potential as

$$A_r = E_0 \frac{\cos\varphi}{\omega} \sum_{n=1}^{\infty} \left[a_n \widehat{J_n}(kr) + b_n \widehat{H_n^{(2)}}(kr) \right] P_n^1(\cos\theta), \tag{3.47}$$

$$F_r = \frac{E_0}{\eta} \frac{\sin\varphi}{\omega} \sum_{n=1}^{\infty} \left[a_n \widehat{J_n}(kr) + c_n \widehat{H_n^{(2)}}(kr) \right] P_n^1(\cos\theta), \tag{3.48}$$

where

$$\widehat{J_n}(kr) = \frac{J_n(kr)}{kr}, \tag{3.49}$$

$$\widehat{H_n^{(2)}}(kr) = \frac{H_n^{(2)}(kr)}{kr}, \tag{3.50}$$

The total field can be obtained through the vector potential as

$$E_r = \frac{1}{j\omega\mu\varepsilon} \left(\frac{\partial^2}{\partial r^2} + k^2 \right) A_r, \tag{3.51}$$

$$E_\theta = \frac{1}{j\omega\mu\varepsilon} \frac{1}{r} \frac{\partial^2 A_r}{\partial r \partial\theta} - \frac{1}{\varepsilon} \frac{1}{r\sin\theta} \frac{\partial F_r}{\partial\varphi}, \tag{3.52}$$

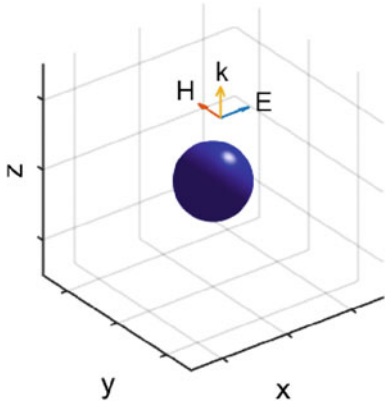


Fig. 3.14 The relationship between the wave vector, polarization direction and the geometric position of the sphere

$$E_\varphi = \frac{1}{j\omega\mu\varepsilon}\frac{1}{r\sin\theta}\frac{\partial^2 A_r}{\partial r\partial\varphi} + \frac{1}{\varepsilon}\frac{1}{r}\frac{\partial F_r}{\partial\theta}, \tag{3.53}$$

$$H_r = \frac{1}{j\omega\mu\varepsilon}\left(\frac{\partial^2}{\partial r^2} + k^2\right)F_r, \tag{3.54}$$

$$H_\theta = \frac{1}{\mu}\frac{1}{r\sin\theta}\frac{\partial A_r}{\partial\varphi} + \frac{1}{j\omega\mu\varepsilon}\frac{1}{r}\frac{\partial^2 F_r}{\partial r\partial\theta}, \tag{3.55}$$

$$H_\varphi = -\frac{1}{\mu}\frac{1}{r}\frac{\partial A_r}{\partial\theta} + \frac{1}{j\omega\mu\varepsilon}\frac{1}{r\sin\theta}\frac{\partial^2 F_r}{\partial r\partial\varphi}. \tag{3.56}$$

where

$$a_n = j^{-n}\frac{(2n+1)}{n(n+1)}, \tag{3.57}$$

In order to solve $|b_n|$ and $|c_n|$, the boundary conditions $\left|E_\theta^t\right|$ and $\left|E_\varphi^t\right|$ are introduced.

$$\begin{cases} b_n = -a_n \dfrac{\widehat{J_n'(ka)}}{\widehat{H_n^{(2)\prime}(ka)}} \\ c_n = -a_n \dfrac{\widehat{J_n(ka)}}{\widehat{H_n^{(2)}(ka)}} \end{cases} \tag{3.58}$$

The total field can be obtained by substituting the coefficients a_n, b_n and c_n into the formula.

Here we consider a perfect conducting sphere with a radius of 10 nm placing at the center of a cube region, whose side length is 84 nm. The uniform plane wave incidents along $+z$ direction while the electric field and the magnetic field polarize long the $+x$ and $+y$ direction, respectively. The wavelength of the wave in vacuum is 20 nm. We apply the above analytic formula and the FDFD program to calculate the three components of the electric field at the z = 0 plane (the three components of the magnetic field can be calculated by the electric field through Maxwell's equations). We carefully compare the real and imaginary parts of the three components of the electric field calculated by the analytical method and the FDFD method. The results are displayed in Fig. 3.15.

By comparing the FDFD results with the analytic solutions, we can conclude that they are in good agreement. A precise 3D-FDFD solver guarantees reliability in our data generation.

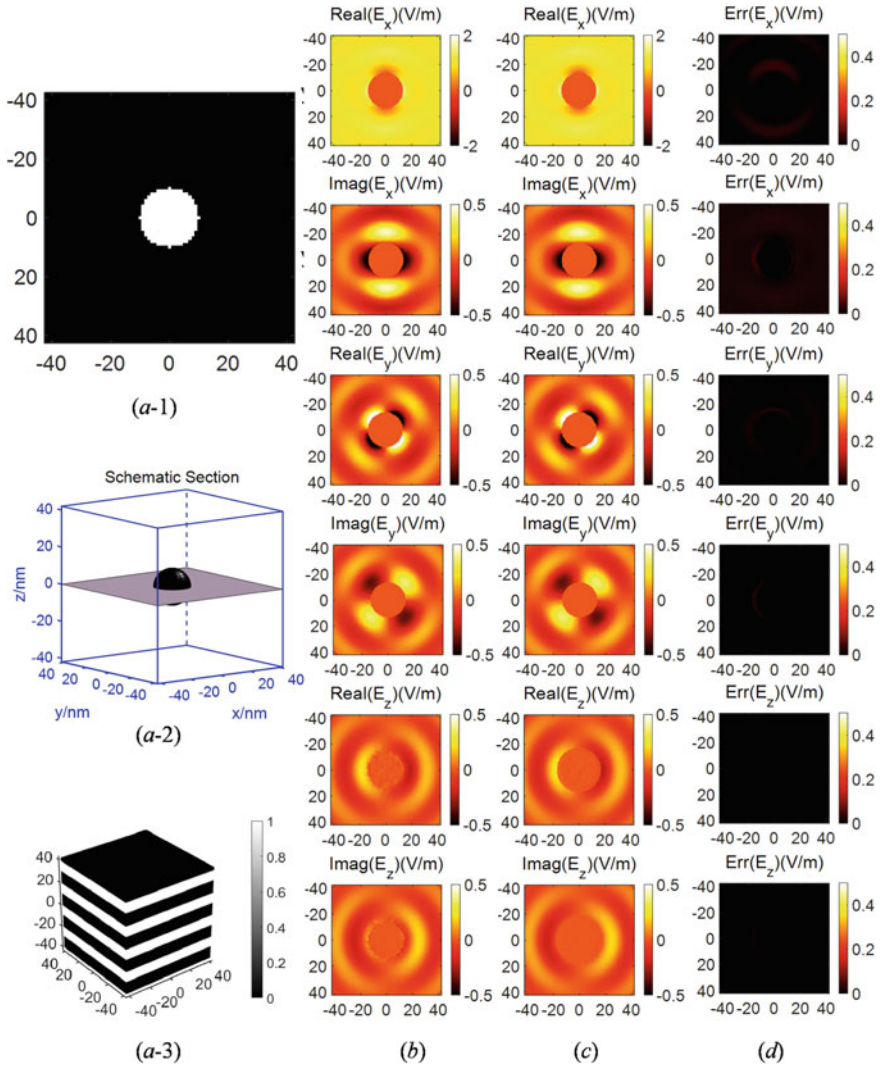


Fig. 3.15 Comparison of FDFD and analytical solution in 3-D case. **a-1** Cross section of the scatterer (z = 0). **a-2** Spatial position of the scatterer and the cross section. **a-3** Propagation direction and wavelength of the electromagnetic wave. **b** Electric field calculated by FDFD. **c** Analytic solution. **d** The error distribution between FDFD and analytical solution

3.3.2 Finite Element Methods

As most scatterers with complex geometries do not have analytical solutions, it is not enough to verify the performance of FDFD program only by comparing the results with analytic solutions. We also need to verify the solution with more generalized

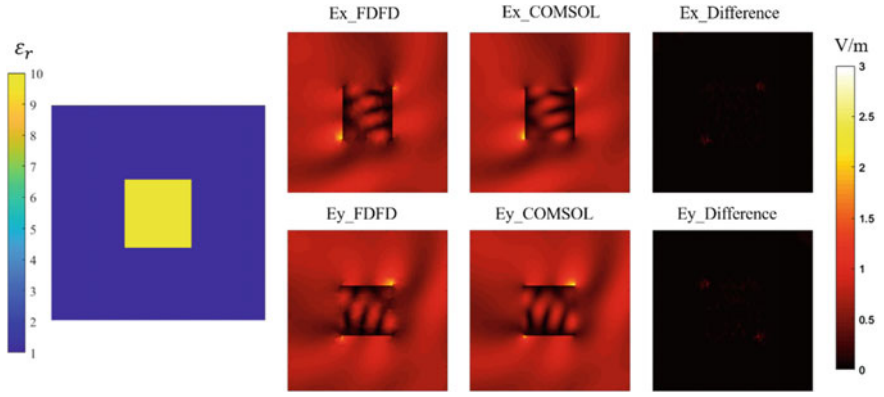


Fig. 3.16 2D validation of the FDFD code for a square scatter

scatterers. For this reason, we employ the commercial simulation software COMSOL as the reference of the FDFD program. The software uses finite element method (FEM) [16–18] to discretize the partial differential equations. Firstly, the space is discretized into non-overlap elements. The weak form of the Maxwell's equations are derived, and the system matrices are assembled to form a system of ordinary differential equations (ODE) with the variable of time. Then, advanced methods including implicit and/or explicit methods of time stepping are used to solve these ordinary differential equations and obtain the corresponding solutions. Here, we believe that the solution obtained by mature commercial software is accurate. In order to quantitatively compute the difference between FDFD and COMSOL, we define the relative difference and the average relative difference as

$$Diff(i,j) = \frac{\sqrt{\left(E_{r_{FDFD}}(i,j) - E_{r_{COMSOL}}(i,j)\right)^2 + \left(E_{i_{FDFD}}(i,j) - E_{i_{COMSOL}}(i,j)\right)^2}}{\sqrt{H_r^2(i,j) + H_i^2(i,j)}}, \tag{3.59}$$

$$Diff_{ave} = \frac{\sum_{i=1}^{128}\sum_{j=1}^{128} Diff(i,j)}{128^2}, \tag{3.60}$$

where E_{r_FDFD} and E_{r_COMSOL} represent the real part of the results from FDFD and COMSOL, while E_{i_FDFD} and E_{i_COMSOL} represent the imaginary part of the results from FDFD and COMSOL. (i, j) is the index of the cell. $Diff_{ave}$ presents the difference between the results from FDFD code and COMSOL Multiphysics.

In the 2-D experiment, the computational domain is a square with the edge length of 128 nm. Four cases are used to validate the FDFD code in various circumstances. In each case, the strength of the excitation source is set to be 1 V/m and the wavelength is 80 nm.

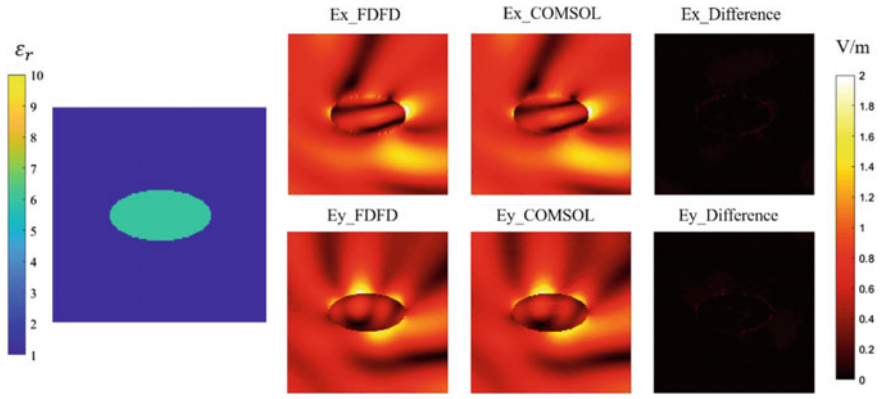


Fig. 3.17 2D validation of FDFD code for an ellipse scatter

In the first case as shown in Fig. 3.16, the scatter is a simple square in the center with the edge length of 40 nm and the relative permittivity is 10. The propagation direction of the excitation source can be expressed as [−1 1 0], while the polarization direction is [1 1 0].

After comparing these two results, the difference of the *x*-and *y*-components of the electric fields from FDFD and COMSOL Multiphysics are 0.0350 and 0.0354.

The second case as shown in Fig. 3.17 is an ellipse scatter laying in the center of the domain. The semi-major axis of the ellipse is 30 nm and the semi-minor axis of is 15 nm. The relative permittivity of the scatter is 6. The propagation direction of the excitation source can be expressed as [3 4 0], while the polarization direction is [−4 3 0]. The difference of the *x*-and *y*-components of the electric fields from FDFD and COMSOL Multiphysics are 0.0380 and 0.0388.

The third case as shown in Fig. 3.18 is a combination scatter consist of an ellipse and a square. The square lays in the center of the domain with the edge length of 40 nm. The ellipse is 20 nm above the center and its semi-major axis is 30 nm and semi-minor axis is 15 nm. The relative permittivity of the scatter is 4. The propagation direction and polarization direction of the excitation source are [−1 2 0] and [2 1 0], respectively. The difference of the *x*-and *y*-components of the electric fields from FDFD and COMSOL Multiphysics are 0.0364 and 0.0508.

In the last case shown in Fig. 3.19, the scatter consists of two identical ellipses, which is vertical to each other. The semi-major axis of the ellipse is 20 nm and the semi-minor axis of is 10 nm. The scatter is put in the center of the domain with the relative permittivity of 7. The propagation direction and polarization direction of the excitation source are [1 1 0] and [1 −1 0], respectively. The difference of the *x*-and *y*-components of the electric fields from FDFD and COMSOL Multiphysics are 0.0330 and 0.0326.

In order to verify the reliability of 3D-FDFD solver, the results are also compared with the commercial simulation software COMSOL. As we use TE wave, the direction of the electric field is perpendicular to that of the propagation vector. Therefore,

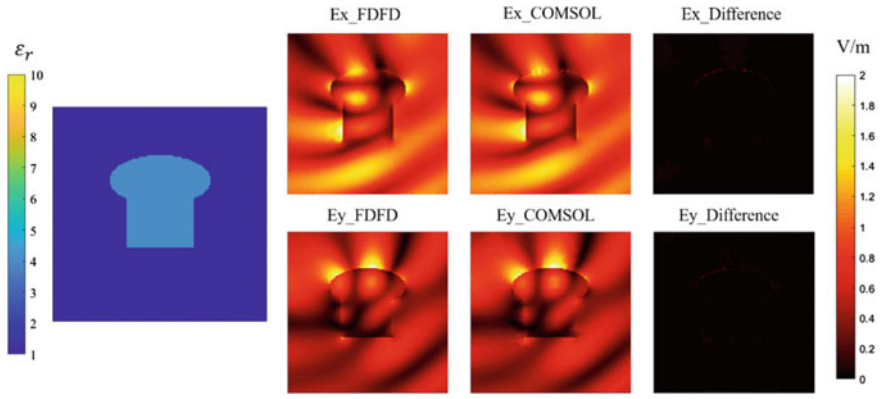


Fig. 3.18 2D validation of FDFD code for a complex scatter which is composed of an ellipse and a square

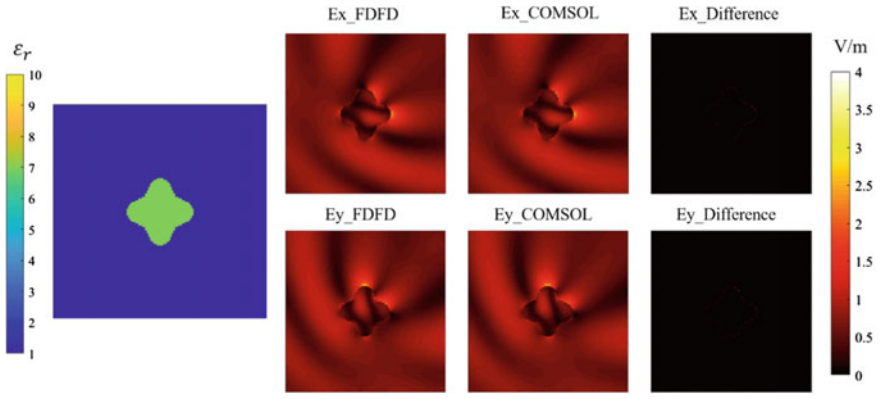


Fig. 3.19 2D validation of FDFD code for a complex scatter composed of two perpendicular ellipses

the magnetic field can be obtained by calculating the curl of the electric field, so we only show the distribution of the electric field here. In the 3-D experiment, 6 cases are chosen to validate the FDFD program. The calculation domain is a cube with the edge of 64 nm. In each case, the strength of the excitation source is set to be 1 V/m. In order to quantitatively compute the difference between 3D-FDFD and COMSOL, we define the average relative difference as

$$Err_{ave} = \frac{1}{N^3} \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} \left\| \frac{E_{FDFD}(i,j,k) - E_{COMSOL}(i,j,k)}{E_{COMSOL}(i,j,k)} \right\|, \tag{3.61}$$

Figure 3.20 show the first case of 3D validation, which is a sphere with radius of 10 nm laying in the center of the domain. The relative permittivity of the scatter is 4. The excitation source is a planewave with the wavelength of 40 nm, whose propagation direction can be expressed as [0.8 −0.60] while the polarization direction is $[3/5\sqrt{2}\ 4/5\sqrt{2}\ 1/\sqrt{2}]$. The difference of the *x*-, *y*- and *z*-components of the electric fields from FDFD and COMSOL Multiphysics are 0.676, 0.0519 and 0.0361, respectively.

The second 3D case, which is shown in Fig. 3.21, is a cube with the edge of 20 nm laying in the center of the domain. The relative permittivity of the scatter is 5. The excitation source is a planewave with the wavelength of 40 nm, whose propagation direction can be expressed as $[0\ -1/\sqrt{2}\ 1/\sqrt{2}]$ while the polarization direction is $[1/\sqrt{3}\ 1/\sqrt{3}\ 1/\sqrt{3}]$. The difference of the *x*-, *y*- and z-components of the electric fields from FDFD and COMSOL Multiphysics are 0.0593, 0.0596 and 0.0596, respectively.

In the third 3D case, which is shown in Fig. 3.22, the scatter is a spheroid in the center of the domain. The lengths of the three semi-axes of the ellipsoid are 10 nm, 6 nm, 6 nm respectively. The relative permittivity of the scatter is 3. The propagation direction of the excitation source can be expressed as [−1 0 1], while the polarization direction is [1 −1 1]. The difference of the *x*-, *y*- and *z*-components of the electric fields from FDFD and COMSOL Multiphysics are 0.0591, 0.0446 and 0.0606, respectively.

In the last 3D case, the scatter consists of a half-sphere and a cylinder, as is shown in Fig. 3.23. The cylinder lays in the center of the domain in the z direction with radius of 10 nm and height of 20 nm. The half sphere has a radius of 10 nm and it is 10 nm above the center. The relative permittivity of the scatter is 3. The propagation direction of the excitation source is [1 0 −1], while the polarization direction is [1 1

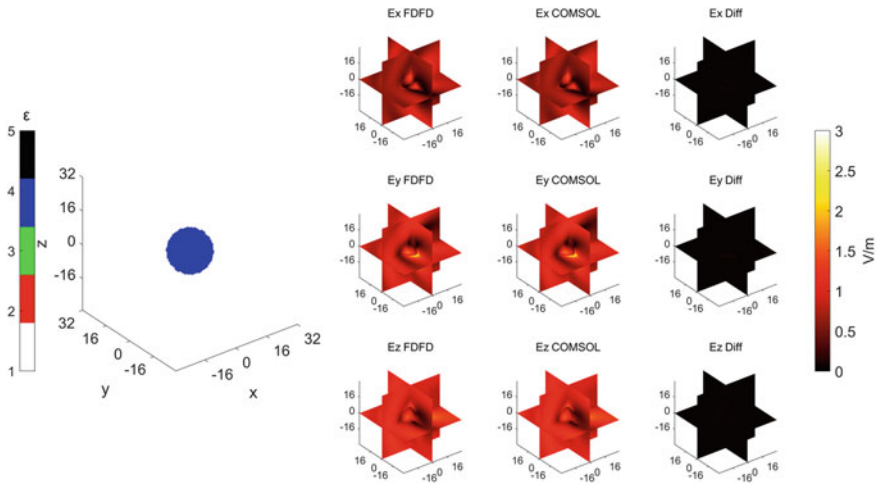


Fig. 3.20 3D validation of FDFD code for a sphere

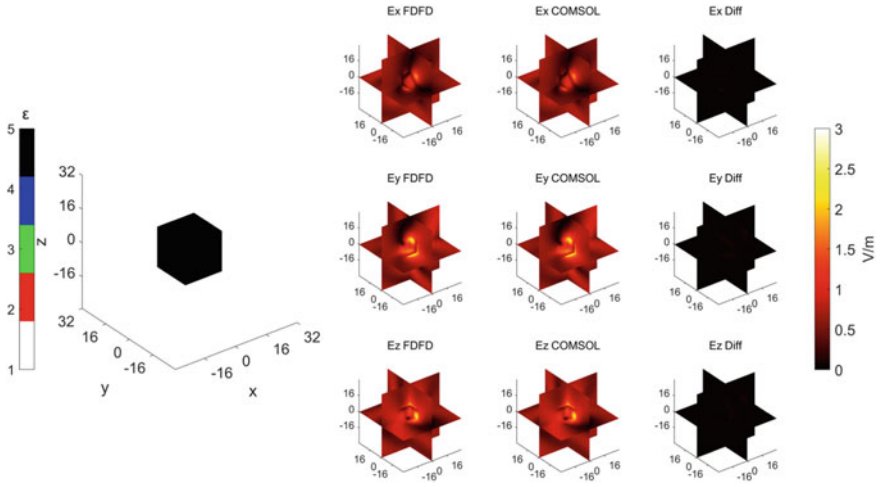


Fig. 3.21 3D validation of FDFD code for a cube

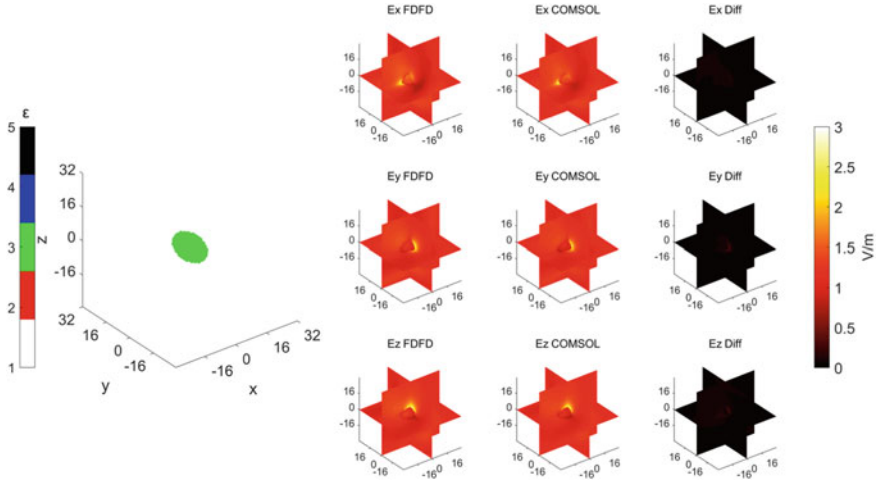


Fig. 3.22 3D validation of FDFD code for a spheroid

1]. The difference of the x-, y- and z-components of the electric fields from FDFD and COMSOL Multiphysics are 0.0461, 0.0303 and 0.0451, respectively.

Through the above comparison between the FDFD code and commercial software COMSOL, we verify the reliability of the program. This builds a solid foundation for us to use this code to generate a large quantity of training data.

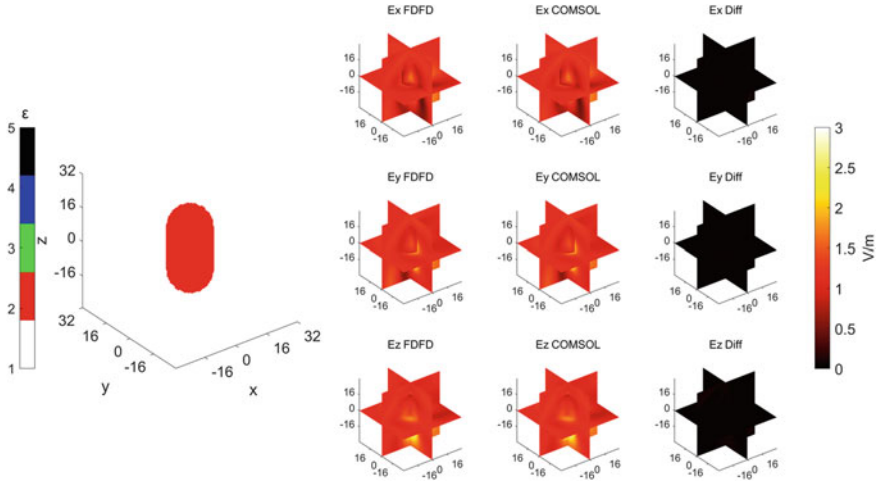


Fig. 3.23 3D validation of FDFD code for a complex scatter composed of a cylinder and two semi-spheres

3.4 The Format of the Dataset

In this work, one set of data includes three parts, the shape of scatter, the characteristic of the excitation source, and the ground truth from the FDFD solver. The material proprieties of the scatter and the wave characteristic will be used as the input to train the network. The output of the network is assumed to be the EM-field. So, the result from FDFD solver will be used to help the network converge in the training stage and validate the accuracy of the network in the testing stage.

3.4.1 Representation of Scatter

In the training stage, one matrix is used to store the information of the scatters. Each pixel in the matrix is the relative permittivity of the material at that location. If one pixel is inside the scatter, the value for that pixel in the matrix will be the relative permittivity of the pixel. While if the pixel is in the background, the value for the pixel will be 0. In this way, the shape and the material of the scatter can be represented by this matrix.

In the 2-D experiment, the scatter matrix only contains one layer in the z direction, so it can be presented as a figure. Figure 3.24 provides some examples of the scatter matrix used in the experiments.

In the 3-D experiment, the scatter matrix contains multiple layers in each dimension, therefore, it can only be perceived through a specific angle. Figure 3.25 gives some examples of the scatter matrix used in the 3-D experiments.

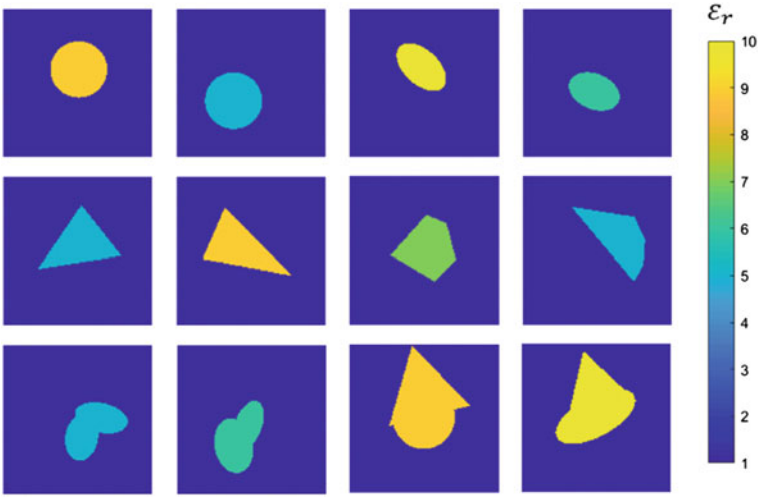


Fig. 3.24 Samples of transforming 2D scatters into images

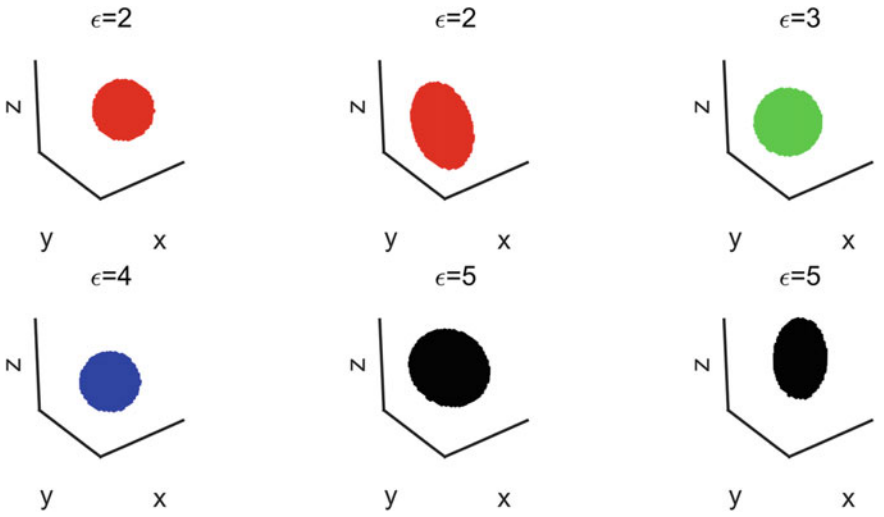


Fig. 3.25 Samples of transforming 3D scatters into images

3.4.2 Representation of Wave

In the training stage, another matrix is used to store the characteristic of the excitation wave. In the wave matrix, the values of 1 and 0 indicate that the phase of the excitation source at this location is in the range of $[0, \pi)$ and $[-\pi, 0)$, respectively. Since the phase of plane wave changes from positive to negative near the zero-phase point in the

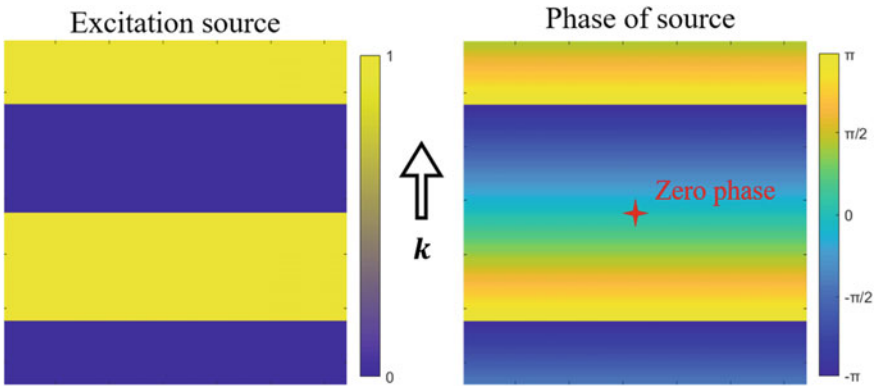


Fig. 3.26 Explanation of judging wave direction from the phase matrix

propagation direction and we fix this point at the center of the square, the propagation direction of the wave can be easily obtained from the image. Besides, TEz polarization is used in 2-D experiment, hence only Hz is fed to the network for training. For 3D cases, we utilize linear transformation to transform TEM wave propagating along any direction into wave propagating along $+z$ direction and polarized in $+x$ direction. Since the incident fields are the same after transforming, our 3-D network does not contain the incident field.

Also, in the 2-D experiment one wave matrix can be presented through a picture. In Fig. 3.26, it is easy to tell the propagation direction of the wave which is from yellow part to blue part (at the center).

Since the strength of the EM -field varies linearly, amplitude of the excitation wave is set to be 1 V/m in this work. Then, the wave matrix will not include wave's amplitude.

3.4.3 Ground Truth

In each dataset, the ground truth contains all components of the EM field. Two matrices will be used to represent one component of EM field, one for the real part and the other for the imaginary part, which are employed for the training and testing stage. In the training stage, the ground truth is used to determine the level of training and the difference between ground truth and the output of the network. In the testing stage, the ground truth is used to evaluate the accuracy of the well-trained network.

In the 2-D experiment, since we use TE wave, E_x and E_y can be calculated from H_z by Eqs. (3.15–3.17). The output of the network are the real and imaginary part of H_z. In the training process, they will be compared with the ground truth in the corresponding dataset and provide information for the network. In this way, the network will find the path of convergence and automatically change the network

parameters. In the testing process, the output will also be compared with the ground truth and the difference will be used to indicate the degree of network training and the final error rate. Figure 3.27 shows the real and imaginary part of H_z in one dataset.

Similarly, the output of the network is real and imaginary part of E_x, E_y and E_z. Then, components of magnetic field can be derived from Eqs. (3.12–3.14). Figure 3.28 shows the real and imaginary part of E_x, E_y and E_z in one 3-dimensional dataset.

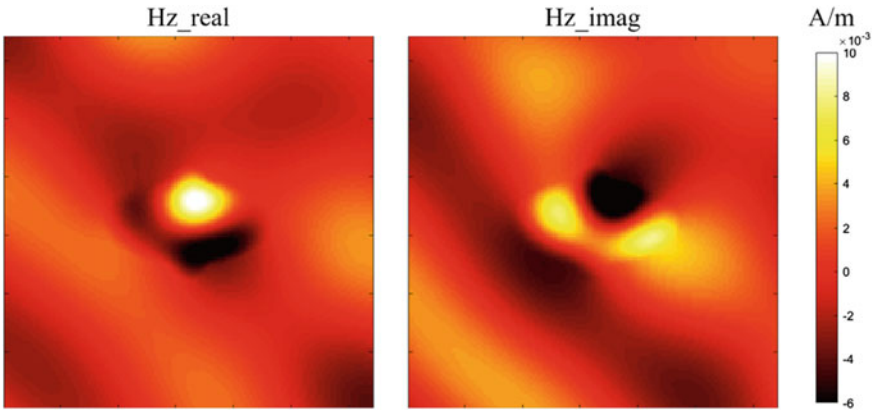


Fig. 3.27 An example of ground truth for 2D case

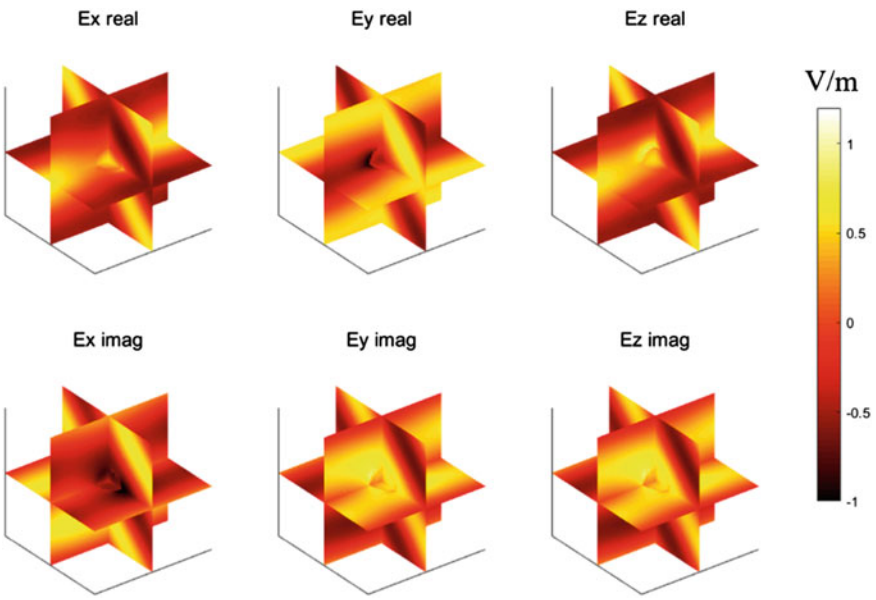


Fig. 3.28 An example of ground truth for 3D case

3.5 Summary

In this part, the process of building database is stated. A random geometry generator is used to feed Finite-Difference Frequency-Domain code to obtain rich and universal data for the network training and testing in both 2-D and 3-D experiments. Then, the format of the data is clarified. The matrices, which represent the illumination wave and the geometry and material of the scatters, are used to train the network. The ground truth from FDFD code can improve the convergence of the network and show the error rate, which will reflect the maturity of training. Based on the versatile datasets, the network can carry out different forms of training attempts and feedback the best use of EM-net.

References

1. The cifar 10 dataset. https://www.cs.toronto.edu/~kriz/cifar.html
2. Big Bad NLP Database. https://datasets.quantumstat.com
3. Kane Y (1966) Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media. IEEE Trans Antennas Propag 14(3):302–307. https://doi.org/10.1109/TAP.1966.1138693
4. Golub GH, Loan CFV (1996) Matrix computations, 3rd edn. Johns Hopkins University Press, USA
5. Demmel JW (1997) Applied numerical linear algebra. Society for Industrial and Applied Mathematics, Philadelphia
6. Saad Y (2003) Iterative methods for sparse linear systems, 2nd edn. SIAM, Philadelphia
7. Simoncini V, Szyld DB (2007) Recent computational developments in Krylov subspace methods for linear systems. Numer Linear Algebra Appl 14(1):1–59
8. Shin W, Fan SH (2012) Choice of the perfectly matched layer boundary condition for frequency-domain Maxwell's equations solvers. J Comput Phys 231(8):3406–3431. https://doi.org/10.1016/j.jcp.2012.01.013
9. Harrington RF (1993) Field computation by moment methods. Wiley-IEEE Press, Piscataway
10. Balanis CA (2012) Advanced engineering electromagnetics. Wiley, New Jersey
11. Kong JA (1986) Electromagnetic wave theory. Wiley, New Jersey
12. Jin JM (2015) Theory and computation of elctromagnetic fields. Wiley-IEEE Press, Piscataway
13. Chew WC (1999) Waves and fields in inhomogeneous media. Wiley-IEEE Press, Piscataway
14. Zhang S (1996) Computation of special functions. Wiley-Interscience, Piscataway
15. Sadiku MNO (2018) Computational electromagnetics with MATLAB. CRC Press, New York
16. Mittra R (2014) Computational electromagnetics: recent advances and engineering applications. Springer, Berlin
17. Rylander T, Ingelström P, Bondeson A (2013) Computational electromagnetics (Texts in applied mathematics (51)). Springer, Berlin
18. Ciarlet PG (2002) The finite element method for elliptic problems. Classics in applied mathematics, vol 40. Society for Industrial and Applied Mathematics, Philadelphia, PA

Chapter 4
Two-Dimensional Electromagnetic Scattering Solver

Within geometries generated in Chap. 3, it is feasible to predict the scattering field. In this chapter, we exhibit the detailed process in database preparing, training and validation. In database preparing, we first apply the method in Chap. 3 to obtain a large quantity of geometries. The geometries are then assigned with random permittivity. At the same time, the illuminated field is generated by another program. Both the geometries and the illuminated field will be fed into the traditional FDFD solver to calculate the electromagnetic field. In the training process, the illuminated field, the scatterer, and the electromagnetic field will be sent into the proposed framework. In validation, the acceleration and the generalization ability of the network are analyzed. The statistical distribution of the error rate on the test set is also displayed at the end of this part.

4.1 Database Preparing

4.1.1 Problem Description

At the beginning, it is essential to have a clear description of the problem. As illustrated in Chap. 1, we mainly aim at solving the scattering field with an incident plane wave. The computational region is a square with the side length of 128 nm. The region is meshed into 128×128 grids, the area of which is 1 nm^2. Inside each grid, we assume that the electromagnetic parameters and field quantities are uniform. As shown in Fig. 4.1, the point (0, 0) is set as the center of the region.

To obtain the database, we have to generate a large number of scatters and incident waves.

Q. Ren et al., *Sophisticated Electromagnetic Forward Scattering Solver via Deep Learning*, https://doi.org/10.1007/978-981-16-6261-4_4

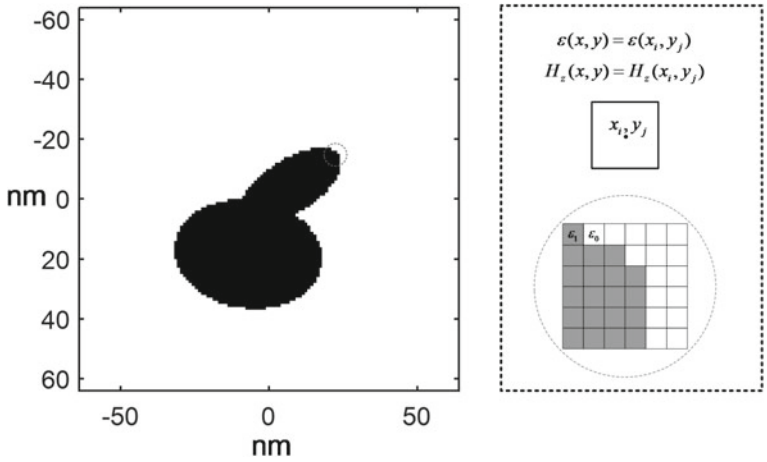


Fig. 4.1 The computational region and the mesh (The electromagnetic parameters and field quantities inside a grid are supposed to be uniform)

4.1.2 Geometries the Scatters

The first step in constructing the 2D scatterer database is to generate adequate geometries. The generating algorithm has been discussed in detail in Chap. 3 thus will not repeated here. The 2D geometries in our database include circles, ellipses, polygons and the combinations of them [1]. Figure 4.2 shows the combinations of the geometries.

The sophisticated geometries enhance the complexity of the training process, so it has a positive impact on the generalization ability of the network. To test the ability, we introduce an open-source dataset, which involves more than 1200 shapes of 70 categories. These complex geometries are resized into 128 $\times$ 128 pixels [2], some of which are shown in Fig. 4.3.

4.1.3 Permittivity the Scatters

After generating these geometries, we need to assign the permittivity to the scatterers. Categorized by the relative permittivity, there are two kinds of scatterers. For dielectric material, the relative permittivity is a real number. For lossy material, it is a complex number.

Table 4.1 shows the relative permittivity of several common inorganic solid materials such as nonmetallic elements, oxides, and salts [3–5]. It can be seen that the relative permittivity of most solid inorganic materials is between 2 and 10.

According to the listed permittivity, we assign the relative permittivity of scatterers to random numbers between 2 and 10. In fact, there are two types of permittivity in

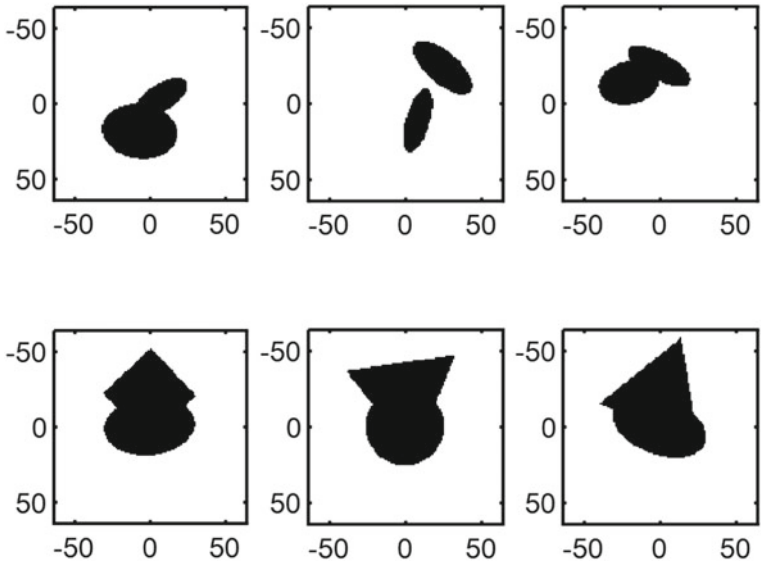


Fig. 4.2 The combination of geometries in the database

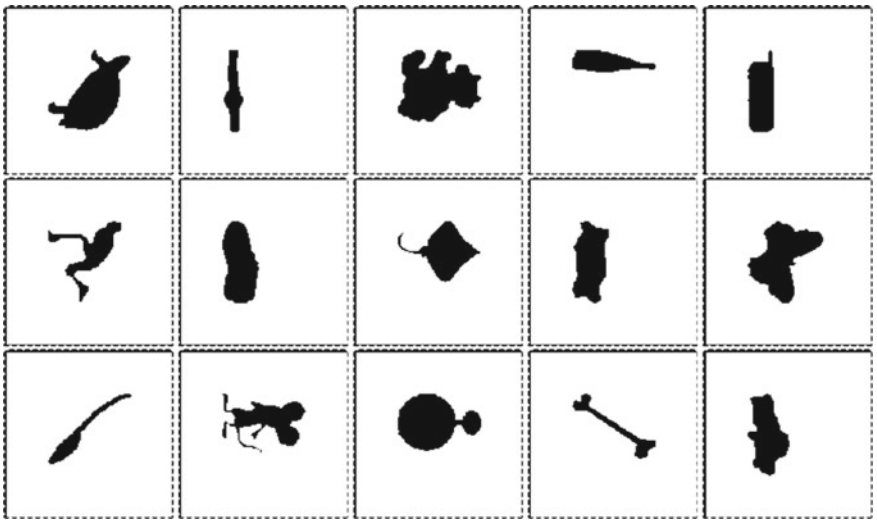


Fig. 4.3 Geometries in the open-source database

the dataset. One type is an integer, while the other is a real number. These two subsets are referred to as D1 and D2, respectively. Obviously, D1 is a subset of D2 which has lower complexity. Here, we plot several examples of D1 and D2 in Figs. 4.4 and 4.5, respectively.

Table 4.1 Permittivity of inorganic materials

Chemical formula	Name	ε_{ijk}
$AgNO_3$	Silver nitrate	9.0
BN	Boron nitride	7.1
C	Diamond	5.87 ± 0.19
$CuSO_4{\cdot}5H_2O$	Cupric sulfate pentahydrate	6.60
Fe_2O_3	Ferric sesquioxide	4.5
KCl	Potassium chloride	4.86 ± 0.02
KI	Potassium iodide	5.00
$MgCO_3$	Magnesium carbonate	8.1
P_4	White phosphorus	3.6
$Pb(CH_3COO)_2$	Lead acetate	2.6
SiC	Silicon carbide	9.72

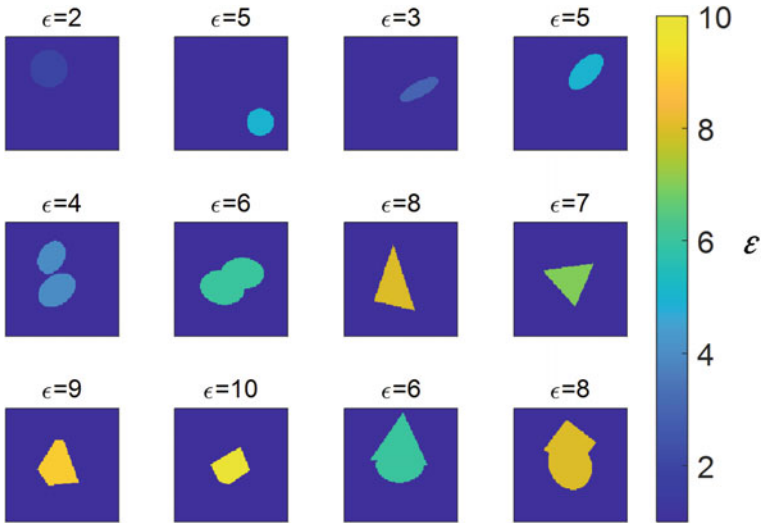


Fig. 4.4 Some scatters with the permittivities belonging to D1

For low frequency electromagnetic wave, many materials can be regarded as lossless. With a real relative permittivity, the electromagnetic wave can propagate without attenuation. When the frequency is very high (such as visible light and ultraviolet light), many materials become lossy. Therefore, it is necessary to extend the electromagnetic field solver to lossy medium. For lossy materials, the relative permittivity is a complex number [6, 7]. When the EM wave propagates in these materials, the amplitude decays and part of the energy convert to Joule heat. The complex permittivity can be written as

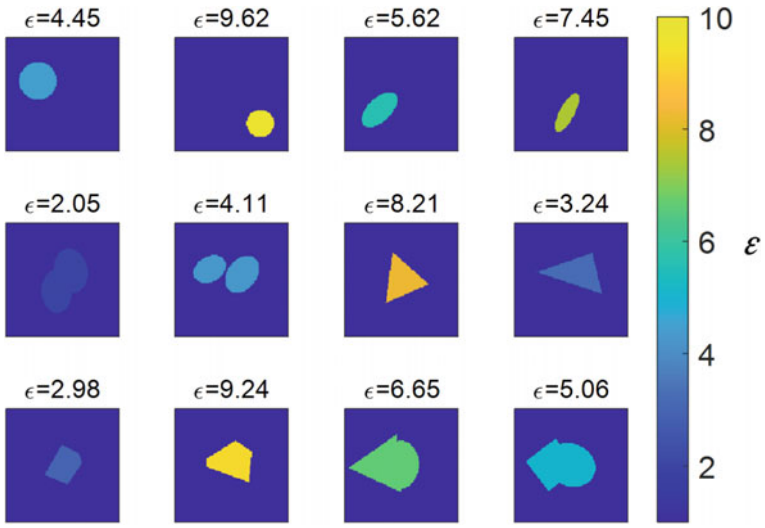


Fig. 4.5 Some scatters with the permittivities belonging to D2

$$\dot{\varepsilon} = \varepsilon_1 - j\varepsilon_2, \tag{4.1}$$

where ε_1 and ε_2 indicate the dispersion and loss [8–11]. According to definition of the complex refractive index ($\dot{N}$), we have

$$\dot{N} = n - jk, \tag{4.2}$$

where n is the real refraction index and k is the absorption index [12, 13]. As

$$\dot{\varepsilon} = \dot{N}^2, \tag{4.3}$$

we have

$$\varepsilon_1 = n^2 - k^2, \tag{4.4}$$

$$\varepsilon_2 = 2nk. \tag{4.5}$$

According to the Planck–Einstein relation [14], we have

$$E = h\nu = h\frac{c}{\lambda}, \tag{4.6}$$

where E (eV) is the energy of the photon. Considering that Plank constant h is 6.626×10^{-34} [14] and the speed of light c is 2.9979×10^8 m/s, the relationship between the wavelength $\lambda(\mu\text{m})$ and photon energy can be obtained as

Table 4.2 Optical properties of metals

Metal	n	k	ϵ_r
Cu	1.04	0.82	0.41−1.71j
Ag	1.46	0.56	1.82−1.64j
Au	1.37	0.80	1.24−2.19j
Rh	1.17	0.69	0.89−1.61j
Pt	1.46	1.15	0.81−3.36j
Pd	1.14	0.65	0.88−1.48j
Hg	1.06	0.57	0.81−1.20j
Ir	1.45	1.01	1.08−2.93j

$$\lambda = \frac{1.2398}{E}. \tag{4.7}$$

Suppose that λ is 123.98 nm and E equals to 10 eV. Under the given circumstance [3, 4], optical properties of several metallic media [15] are listed in Table 4.2.

Similar to D1 and D2, we establish the dataset D3 and D4. In these two data sets, the relative permittivity of the scatterers is complex number listed in Table 4.2. It is worth pointing out that the geometries in D3 and D4 are from self-generated and open-source data sets, respectively. Several examples in D3 and D4 are displayed in Figs. 4.6 and 4.7, respectively.

In the four sub-datasets, the relative permittivity is stored as a matrix. In D1 and D2, the matrix element is the value of the relative permittivity, while in D3 and D4, it is the label ranging from 1 to 8, representing the 8 metals listed in Table 4.2.

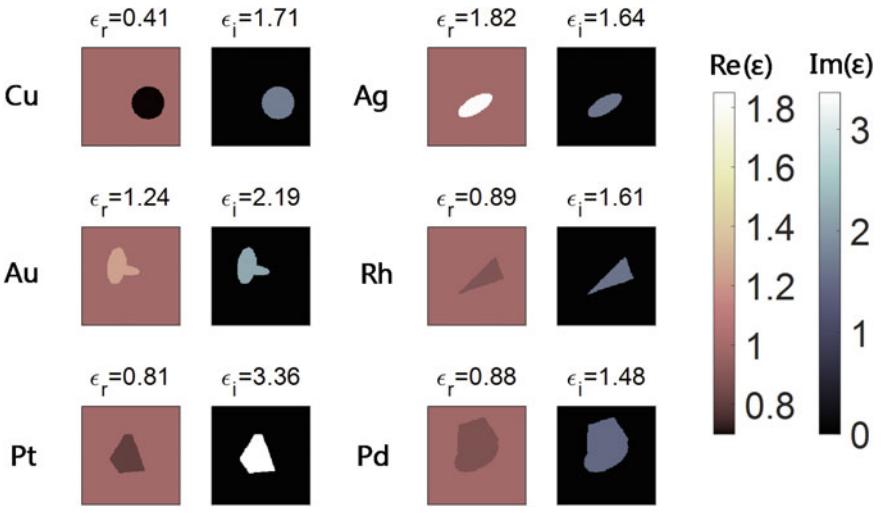


Fig. 4.6 Some scatters with the permittivities belonging to D3

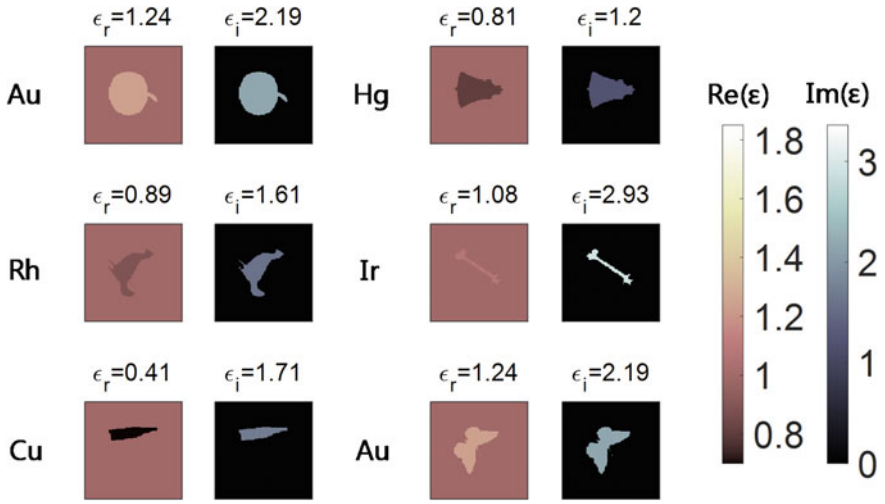


Fig. 4.7 Some scatters with the permittivities belonging to D4

4.1.4 Illumination Setting

To acquire the total field, both the scatterer and the excitation source are indispensable. Generally speaking, there are two types of sources widely used in electromagnetic scattering problems, which are the plane wave source and dipole source [16]. In our studies, we mainly focus on the plane wave source.

The basic concept of the plane wave has been illustrated in Chap. 1. Here the incident wave can be expressed as

$$\mathbf{A} = \mathbf{A_0} e^{-j\mathbf{k}\cdot\mathbf{r}}, \tag{4.8}$$

where $\mathbf{A_0}$ indicates the amplitude of the field intensity and $\mathbf{A}$ indicates the field intensity at $\mathbf{r}$ while $\mathbf{k}$ indicates the wave vector. Since $\mathbf{A_0}$ is fixed in our studies, we can use the phrase of $\mathbf{A}$ to represent the whole incident field [16–18]. As we only care about the propagation direction and the wavelength, we can simplify the phrase $\varphi(\mathbf{r})$ by utilizing its sign instead of the value, namely

$$W(\mathbf{r}) = \begin{cases} 0 & \varphi(\mathbf{r}) < 0 \\ 1 & \varphi(\mathbf{r}) \geq 0 \end{cases}. \tag{4.9}$$

Figure 4.8 illustrate the phrase $\varphi(\mathbf{r})$ and its sign $W(\mathbf{r})$. It is clear that the framework can easily obtain the propagating direction and wavelength through $W(\mathbf{r})$.

The dataset consists of 4 sub-datasets. In W1, the propagating direction of the wave is along the x-axis, while in W2, W3 and W4, the propagation direction can be arbitrary. The wavelength in W1 and W2 is 80 nm, and 123.98 nm in W3. While it is a

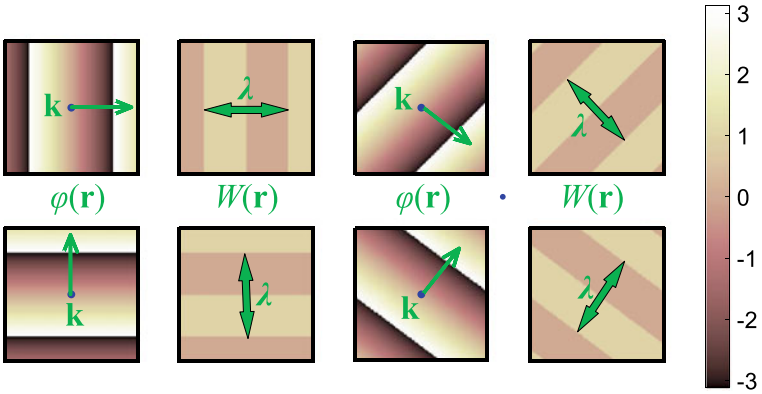


Fig. 4.8 The phrase $\varphi(\mathbf{r})$ and its sign $W(\mathbf{r})$. The propagating direction and wavelength can be acquired from $W(\mathbf{r})$

random integer ranging from 75 to 85 nm in W4. Similar to the relative permittivity, W1, W2, W3 and W4 are all stored as matrices whose elements are $W(\mathbf{r})$.

4.1.5 Calculation of the Model

With the FDFD program [19] represented in Chap. 3, it is feasible to obtain the scattering field through the scatterer and the incident field. Here, Fig. 4.9 exhibits an example of the scattering field computed by the program.

As we solve Maxwell's equations in the frequency domain, the field quantity is a complex number. We extract the real and imaginary parts of the result and feed them to the framework with the pre-acquired scatterer and incident field.

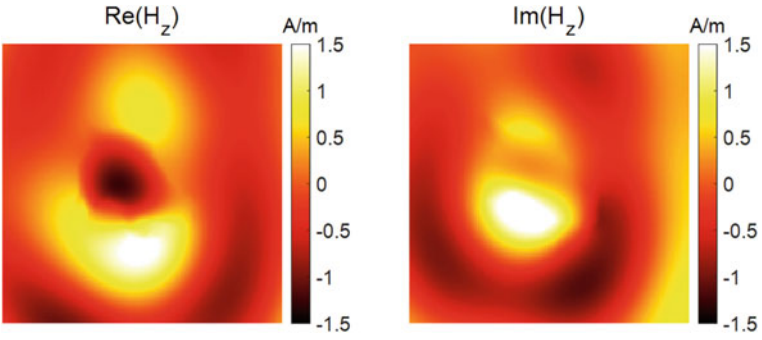


Fig. 4.9 The real and imaginary parts of the output magnetic field

4.2 Architecture of the DL Framework

After generating the training database, we will deal with the details of the architecture of the neural network, which is designed to solve a special physics problem [20–22]. The proposed one is mainly based on the U-net [23–26], a specific type of convolutional neural network which is originally used for biomedical image segmentation, and later proved to be useful at prediction tasks where input and output pairs are correlated in a spatial way.

Specifically, the U-net is able to extract high-level information from input image pairs through the downsampling path. In this way, strided convolution layers with filters defined by generic hyperparameters (Height $\times$ Width $\times$ Channel) increase the effective receptive field of the neural network, while the following pooling operations make it trainable and help to aggregate information over large areas of the input feature maps, which intuitively serves to reveal the physical interaction between the scatterer and the incident wave. However, in order to get paramount coarse features from the input image pairs, the downsampling path enlarges the number of channels to extract high-level features while inevitably shrinking the sizes of input images. Therefore, an upsampling path is introduced to recover the original sizes of input images for per-pixel magnetic field predictions. Unlike the downsampling, the upsampling path involves the so-called transposed convolution layers [27], they are used to recollect coarse features from different channels in order to generate fine field predictions.

Faced with a large number of variations existed in the chosen scattering problem, improvements compared with the original U-net are discussed. (1) Short Cut Connection: a distinct drawback of the downsampling path is the information loss. On top of that, we implement upsampling with added shortcut connections from the corresponding lower layers in the downsampling path. By doing this, the information loss happened in the downsampling can be recovered. This structure has been proved to be efficient in improving prediction accuracy. (2) Residual Connection: multivariant learning task requires a large network capacity. Simply adding more network layers can expand the capacity, but the accompanying problems such as exploding and vanishing gradients limit the range of its application. Hence, we employ the neural network model by using residual connection [28] with gated structure [29, 30], we further prove that this structure guarantees the accuracy of prediction and fast convergence even when a considerable number of layers are applied. Therefore, by merging residual structure with every convolution/transposed convolution operation, we construct over 80 layers for magnetic field prediction. (3) Tailored designs of input and output feature map: our proposed deep learning methods are required to perform field prediction in the forms of complex values, while generic tasks via U-net are mostly conducted in the field of real values, such as biomedical image segmentation. Thus, it is particularly important to make sure that the input and output feature map (usually refer to scattering variants and field pattern in our case) employed in the framework is able to generate complex field distributions.

The detailed structure of the neural network is illustrated in Fig. 4.10. There are six units, each of which consists of 2 residual blocks that implement downsampling

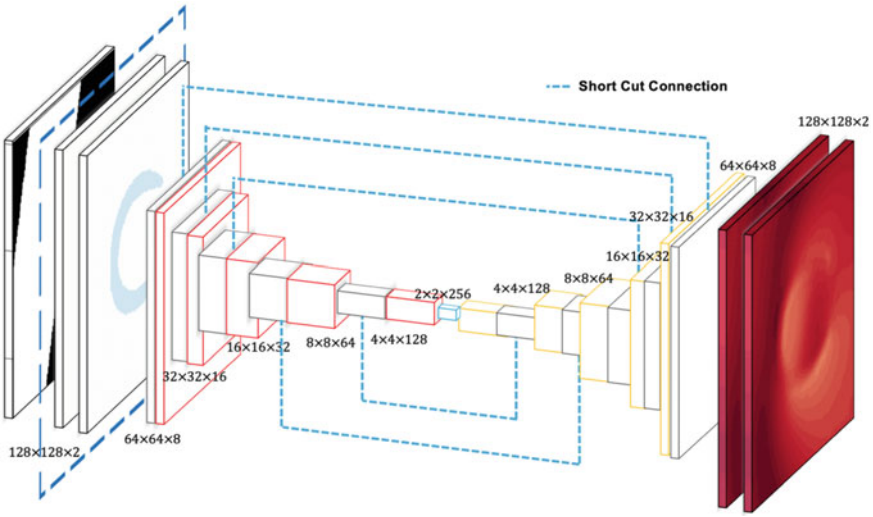


Fig. 4.10 The detailed neural network structure. *Source* Li et al. [2]

step by step. One block contains 3×3 convolutional layers with a stride of 2, and the other with a stride of 1. At every downsampling, the output channels double, while the input size reduces by a factor of 4. At the same time, 6 units with each consisting of 1 residual block and a 3×3 transport convolution layer implement upsampling step by step. At every upsampling, the output channels reduce by half, while input size scales up by a factor of 4. At the bottom of the U-shape architecture, one additional residual block containing a 3×3 convolutional layer with a stride of 1 is used to connect the downsampling and upsampling modules and essentially serves as an intermediate unit. Notably, each convolution/transposed convolution layer is followed by an active function to add nonlinearity. In the following sections, this network is referred to as EM-net for short.

Here, several experiments are conducted to prove that our structure can obtain higher prediction accuracy and faster convergence. The impact of residual blocks and skip connection on EM-net is first investigated. The EM-net with the other 3 networks are compared, which are U-Net with skip connection, U-Net with residual blocks, and simple U-Net without both, respectively. The four networks are trained under the same condition and the experimental results are shown in Fig. 4.11a. Obviously, the error rate of the EM-net is lower while the convergence speed is faster.

Besides, the influence of the number of sampling units on the EM-net is further studied. The number of sampling units are set to be 3, 4, 5, 6 and trained under the same condition. The experimental results are shown in Fig. 4.11b. It has been proved that more downsampling units could contribute to a larger network capacity, which is essential for a better approximation of the FDFD method. For our problem, 6 downsampling units are enough to generate satisfactory optimization results.

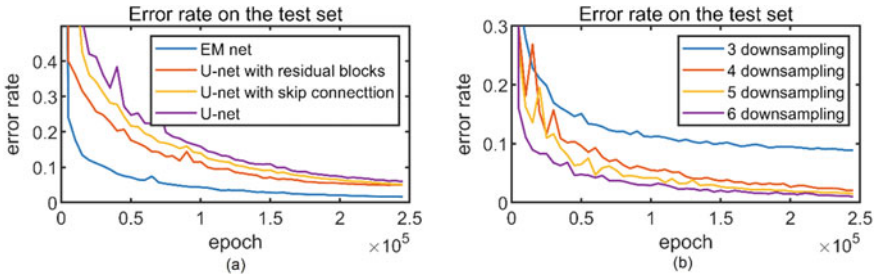


Fig. 4.11 Performance of different architectures. **a** Comparison between the EM-net and the U-Net with skip connection/U-Net with residual blocks/U-net. **b** Comparison between the EM-nets with 3/4/5/6 downsampling units

4.3 Training of the DL Framework

After generating the database and constructing the framework, we are able to embark on the training process. The deep-learning framework is performed on TensorFlow 1.0 [31, 32], which is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible and powerful ecosystem of tools, libraries and community resources that can facilitate the in-depth research and progress of deep learning.

The aim of the training is to optimize the loss function [33–35], which can be defined as

$$Loss = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}|F_{FDFD}(i,j) - F_{Framework}(i,j)|^2, \tag{4.10}$$

where F_{FDFD} and $F_{Framework}$ refer to the electromagnetic fields calculated by the FDFD program and predicted by the network, respectively. The Adam optimizer is utilized to reduce the loss function in the iterative process. The training was performed on two NVDIA GTX 1080 Ti graphic cards on Think station P920. The sufficient computing capability ensures the stability of training.

In reality, it is indispensable to conduct several pre-experiments before the ultimate training to determine the relevant parameters, which optimize the training process. The pre-experiment and the selection of related parameters will be discussed in detail in Chap. 5.

4.4 Results on the Test Set

After enough training, the network has an excellent performance in predicting the field with the corresponding input. To measure the performance of the network quantitatively, we introduce the average relative error. Suppose p is the serial number of the pixel, its relative error can be defined as

Table 4.3 Experimental conditions

Experiment group index	Database	Wavelength (nm)	Direction of the incident wave	Relative permittivity
#1	D1 + W1	80	$+x$	Integer
#2	D1 + W2	80	Arbitrary	Integer
#3	D3 + W3	123.98	Arbitrary	Complex

$$\mathrm{Err}(p) = \frac{\sqrt{\left|H_r(p) - H_r'(p)\right|^2 - \left|H_i(p) - H_i'(p)\right|^2}}{\sqrt{H_r^2(p) + H_i^2(p)}}, \tag{4.11}$$

where $H_i(p)/H_r(p)$, $H_i'(p)/H_r'(p)$ refer to the imagine/real part of the calculated and predicted field. Then the average relative error can be formulated as

$$\mathrm{Err}_{\mathrm{ave}} = \frac{\sum_{i=1}^{N} Err(i)}{N}, \tag{4.12}$$

where N is the total number of the pixels. To measure the performance of the proposed network in different situations, we have carried out a series of experiments. As demonstrated earlier, we mainly focus on the influence of the scatterer and the incident wave, and the specific experimental conditions are displayed in Table 4.3. In each experiment, some of the results randomly selected from the test set are shown.

4.4.1 Experiment Group #1

In Experiment Group #1, the incident wave is a TM wave propagating along the $+x$ direction with a wavelength of 80 nm (W1). The scatterer is self-generated geometries whose relative permittivity is a random integer ranging from 2 to 10 (D1). The dataset contains 60,000 specimens, 10% of which are test sets. Figure 4.12 exhibits several samples randomly chosen in the test sets. Here, (*a*) reflects the geometry and relative permittivity of the scatterer, and (*b*) indicates the propagating direction and wavelength of the incident field. (*c*) and (*d*) demonstrate the field calculated by the FDFD program and predicted by the network, respectively. (*e*) is the error between (*c*) and (*d*).

It is obvious that the prediction of the DL framework matches well with the FDFD calculation. In fact, the average relative error rate is only 0.81% on the test set, which demonstrates a strong predictive ability.

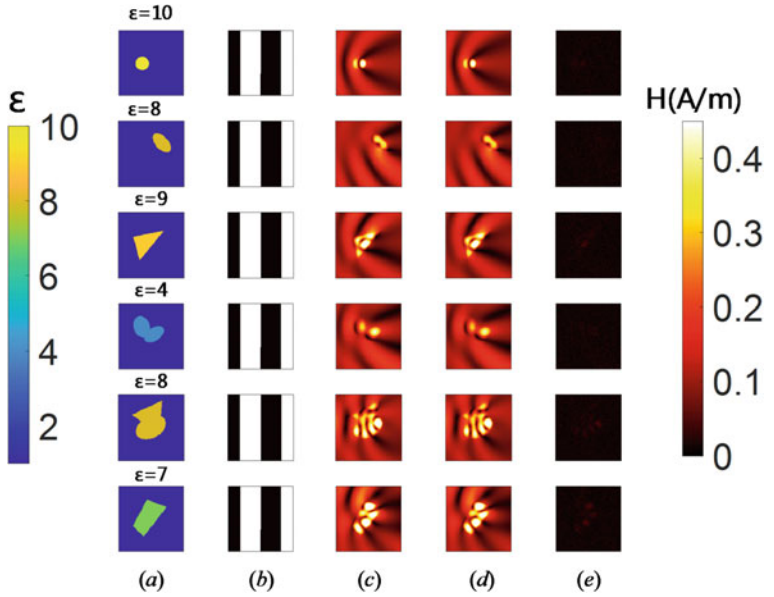


Fig. 4.12 Examples in Experiment Group #1. **a** The geometry and relative permittivity of the scatterer, **b** the propagating direction and wavelength of the incident field, **c** the field calculated by the FDFD program, **d** the field predicted by the network, **e** the error between (**c**) and (**d**)

4.4.2 Experiment Group #2

In practical computational scenarios, electromagnetic waves always propagate along different directions, hence it is necessary to verify the corresponding cases. In Experiment Group #2, the incident wave is a TM wave propagating along an arbitrary direction with a wavelength of 80 nm (W2). The scatterer is the same as Experiment Group #1 whose relative permittivity is also a random integer ranging from 2 to 10 (D1). The whole dataset contains 60,000 specimens, 6000 of which are test cases. Figure 4.13 shows several samples randomly selected in the test sets. Here, (*a*) represents the geometry and relative permittivity of the scatterer, and (*b*) reflects the propagating direction and wavelength of the incident field. (*c*) and (*d*) are the field calculated by the FDFD program and predicted by the network, respectively. (*e*) is the error between (*c*) and (*d*).

Figure 4.13 indicates that the DL framework can give effective predictions even if the incident field propagates along arbitrary directions. In fact, the average relative error rate is only 1.18% on the test set. Although a little higher than Experiment Group #1, it is still pretty accurate considering the more sophisticated data set.

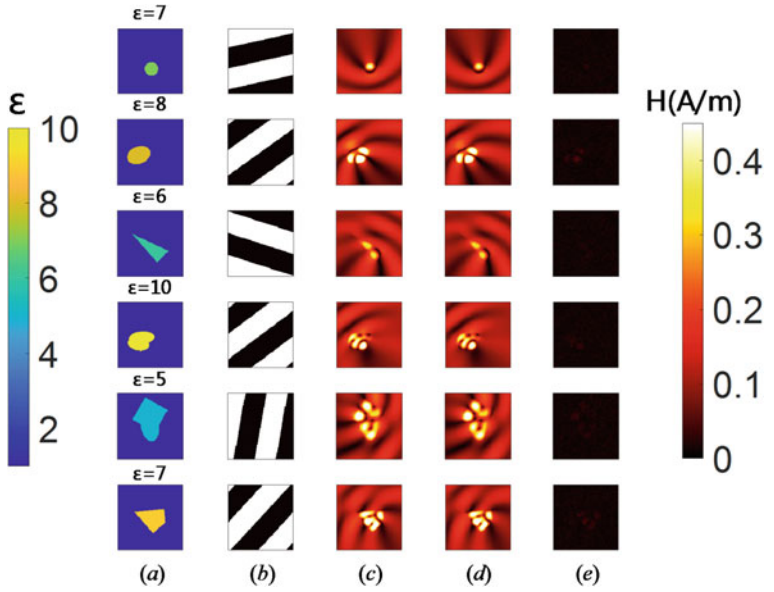


Fig. 4.13 Examples in Experiment Group #2. **a** The geometry and relative permittivity of the scatterer, **b** the propagating direction and wavelength of the incident field, **c** the field calculated by the FDFD program, **d** the field predicted by the network, **e** the error between (**c**) and (**d**)

4.4.3 Experiment Group #3

The permittivity discussed in the previous two groups of experiments are real numbers. In fact, for visible or ultraviolet light, many materials exhibit lossy characteristics. Therefore, the permittivity can be a complex number. To handle the complex permittivities, two matrices are required to reflect the real part and the imaginary part respectively, which is infeasible theoretically without altering the input structure. Nevertheless, if we sacrifice the generalization ability of the framework and only consider the eight materials listed in Table 4.2, the complex permittivity can be considered as eight categories numbered from 1 to 8. Implementing this cunning technique, it is practicable to predict the magnetic field without any modification to the network.

In Experiment Group #3, the incident wave is a TM wave propagating along an arbitrary direction with a wavelength of 123.98 nm (W3). The geometry is the same as the previous two experiments while the relative permittivity is a complex number in Table 4.2 (D3). The dataset also contains 60,000 specimens, and 10% are test sets. Figure 4.14 exhibits provide a view of some examples in the test sets. Here, (*a*)/(b) show the geometry and relative permittivity of the scatterer, while (*c*) reflects the propagating direction and wavelength of the incident field. (*d*) and (*e*) exhibit the field calculated by the FDFD program and predicted by the network, respectively. (*f*) is the error between (*d*) and (*e*).

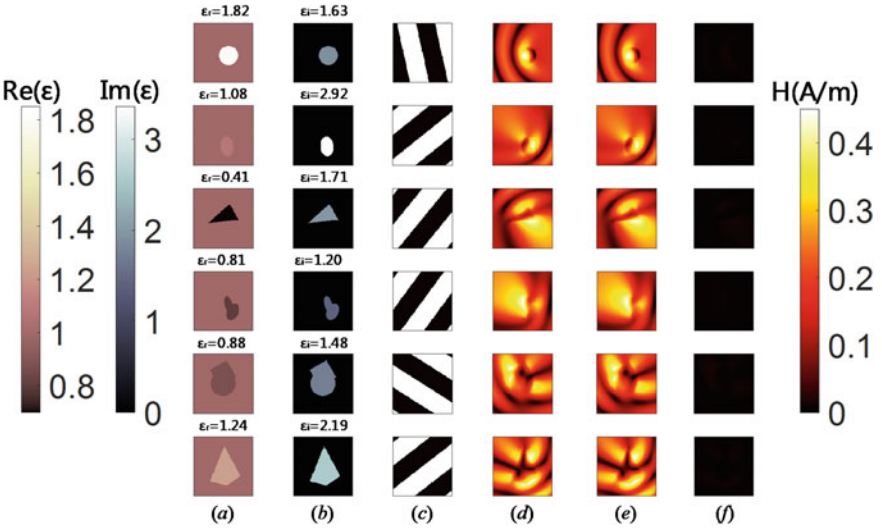


Fig. 4.14 Examples in Experiment Group #3. **a** The geometry and the real part of the relative permittivity, **b** the geometry and the imaginary part of the relative permittivity, **c** the propagating direction and wavelength of the incident field, **d** the field calculated by the FDFD program, **e** the field predicted by the network, **f** the error between (**d**) and (**e**)

Figure 4.14 illustrates that the framework is capable of predicting the scattering field of scatterers with complex permittivities. The average relative error rate is only 0.792% on the test set, demonstrating an excellent performance on predicting the field.

4.4.4 Acceleration

The purpose of the proposed EM-net is to forecast the magnetic field faster than the conventional numerical methods. After well-trained, the EM-net can obtain the magnetic field almost simultaneously once the matrices representing the incident wave and scatters are input into the network. However, it is often quite time-consuming for the conventional numerical algorithms. Here, to quantitatively measure the acceleration ability of the network, we record the calculation time spent by the traditional algorithm and neural network, respectively. In order to ensure the accuracy of the experiment, the two groups of measurements are carried out on the same computing platform with the same data set.

Figure 4.15 demonstrates the solving time by EM-net and the FDFD program. Both the two methods emerge a linear relationship between the time consumption and the number of samples. To solve 100 samples, the FDFD algorithm takes around 3626 s while the EM-net only takes 1.76 s. Therefore, it can be concluded that the

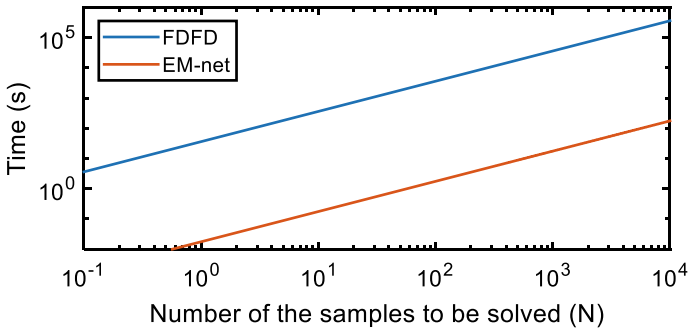


Fig. 4.15 Time consumption of FDFD method and EM-net to solve N samples. The EM-net is more than three magnitudes faster than FDFD

EM-net can accelerate the solving process by more than three orders of magnitudes without sacrifice accuracy.

4.5 Generalization Ability

Deep learning technology is famous for its high accuracy and computing speed. Although this technique has been widely used in computational physics, it also has some defects. A fatal problem is that overfitting often exists in the training process. That is to say, it can only have good outcomes when the input data is similar to the training set, while it will perform extremely poorly when the input is different. Therefore, it is significant to verify that the proposed EM-net exactly learns the underlying physics of the scattering problem, rather than simply memorize the trained samples. The criterion is often referred to as the generalization ability. A typical pattern to verify the generalization ability is to observe the performance of the pre-trained network on a test set with completely different characteristics.

To this scheme, we construct several data sets to validate the trained framework. Here, we mainly focus on the generalization ability of the permittivity, wavelength and geometry shape. The test experiment groups are labeled as #4, #5 and #6 respectively, whose specific conditions are illustrated in Table 4.4.

Table 4.4 Experimental conditions

Experiment group index	Database	Wavelength (nm)	Relative permittivity	Network to be tested
#4	D2 + W2	80	Real	#2
#5	D1 + W4	75–85	Integer	#2
#6	D4 + W3	123.98	Complex	#3

4.5.1 Experiment Group #4

In Experiment Group #4, the generalization ability of the EM-net for the permittivity is investigated. Here, the incident wave is a TM wave propagating along an arbitrary direction with a wavelength of 80 nm (W2). The scatterer is self-generated geometries whose relative permittivity is a random real number ranging from 2 to 10 (D2). The framework to be tested is trained in Experiment Group #2, whose permittivity is an integer. Figure 4.16 exhibits several samples randomly chosen in the test sets. Here, (*a*) displays the geometry and relative permittivity of the scatterer, and (*b*) shows the propagating direction and wavelength of the incident field. (*c*) and (*d*) reflect the field calculated by the FDFD program and predicted by the network, respectively. (*e*) is the error between (*c*) and (*d*).

It can be acquired from Fig. 4.16 that the DL framework trained in Experiment Group #2 can give a relatively accurate forecast even if the permittivity to be predicted is a real number. In fact, the average relative error rate is about 8.85%, exhibiting a robust generalization ability in predicting the field considering that the training data set only contains integer permittivity.

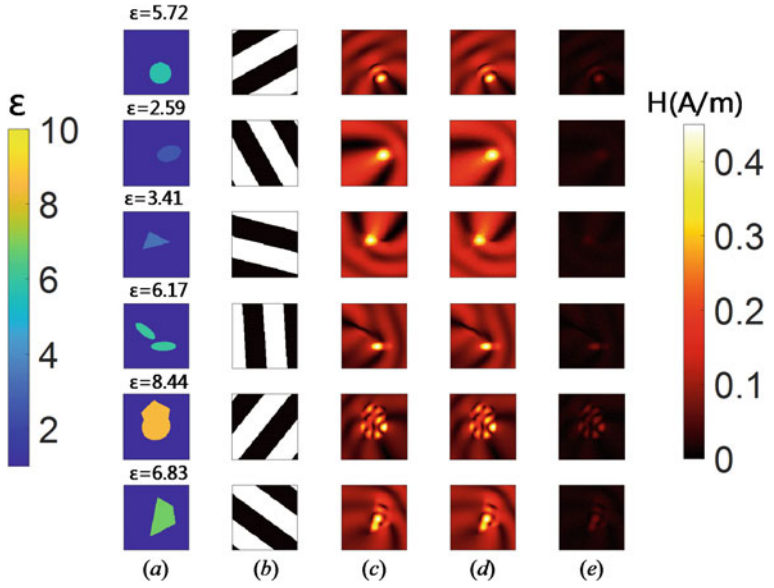


Fig. 4.16 Examples in Experiment Group #4. **a** The geometry and the relative permittivity of the scatterer, **b** the propagating direction and wavelength of the incident field, **c** the field calculated by the FDFD program, **d** the field predicted by the network, **e** the error between (**c**) and (**d**)

4.5.2 Experiment Group #5

In Experiment Group #5, the generalization ability of the EM-net for the wavelength is studied. Here, the incident wave is a TM wave propagating along an arbitrary direction with a wavelength ranging from 75 to 85 nm (W4). The scatterer is the same as Experiment Group #2, which consists of self-generated geometries with the integer relative permittivity ranging from 2 to 10 (D1). The framework to be tested is also trained in Experiment Group #2, whose wavelength is fixed at 80 nm. Figure 4.17 presents some of the examples randomly selected in the test sets. Here, (*a*) demonstrates the geometry and the relative permittivity of the scatterer, and (*b*) illustrates the propagating direction and wavelength of the incident field. (*c*) and (*d*) reflect the field calculated by the FDFD program code and predicted by the network, respectively. (*e*) is the error between (*c*) and (*d*).

Figure 4.17 demonstrates that the DL framework can give relatively reliable predictions even if the wavelength is a variable. In fact, the average relative error rate is around 11%. Although it cannot put on a par with the previous experiments, it still exhibits certain generalization ability in predicting the field considering that the training data set only contains a fixed wavelength.

In order to study the generalization ability more detailly, we conduct statistical analysis about the error rates distribution for different wavelengths. The results are illustrated in Fig. 4.18. It can be obtained that the error increases when the wavelength

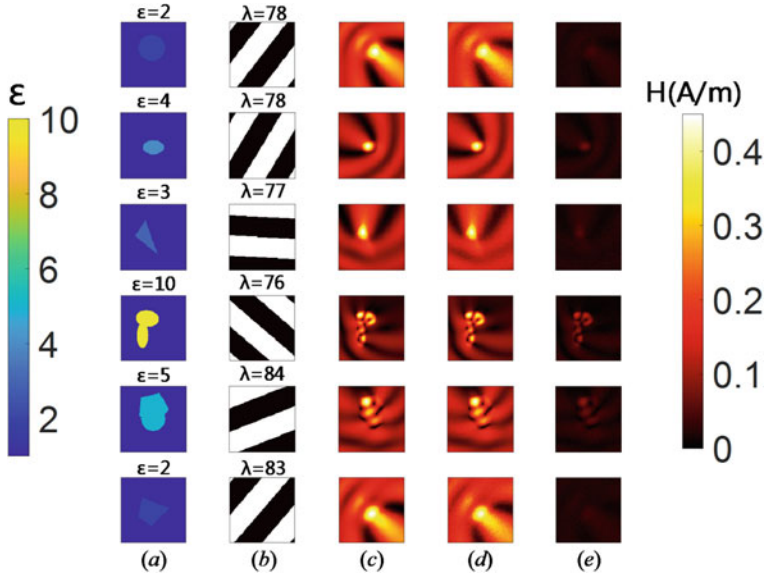


Fig. 4.17 Examples in Experiment Group #5. **a** The geometry and the relative permittivity of the scatterer, **b** the propagating direction and wavelength of the incident field, **c** the field calculated by the FDFD program, **d** the field predicted by the network, **e** the error between (**c**) and (**d**)

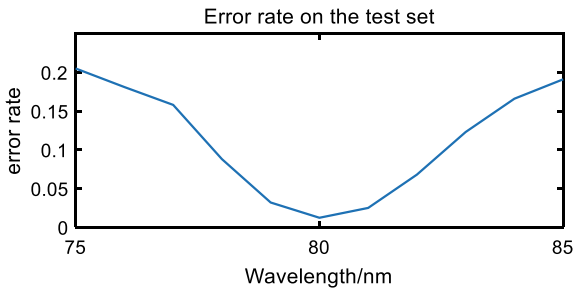


Fig. 4.18 Error rates for different wavelengths ranging from 75 to 85 nm

is far away from 80 nm. So that the generalization ability for the wavelength is not as good as those for other parameters.

4.5.3 Experiment Group #6

In Experiment Group #6, the generalization ability of the geometry shape is considered. Here, the incident wave is the same as Experiment Group #3, which is a TM wave propagating along an arbitrary direction with a wavelength of 123.98 nm (W3). The scatterer is obtained from an open-source data set whose relative permittivity is a complex number in Table 4.2 (D4). The framework to be tested is also trained in Experiment Group #3, whose geometry is self-generated. Figure 4.19 provides a clear view of several specimens randomly chosen in the test sets. Similarly, (*a*)/(b) reflect the geometry and relative permittivity, while (*c*) represents the propagating direction and wavelength of the incident field. (*d*) and (*e*) are the field calculated by the FDFD program and predicted by the network, respectively. (*f*) is the error between (*d*) and (*e*).

It can be accessed from Fig. 4.19 that the DL framework can give relatively accurate predictions even if the geometry for testing has wide variations. In fact, the average relative error rate is about 4.98%. For such complex scatterers, it is excellent to achieve such a low average relative error rate, which exhibits a quite robust generalization ability.

To sum up, the three experiments validate the generalization abilities of the EM-net. Among them, the DL framework has a relatively strong generalization ability for permittivity and geometry shape. When it comes to the wavelength, it has a lot to be improved in future work.

4.6 Statistical Analysis

In the previous sections, the predicting ability and generalization ability of the proposed framework have been analyzed. The average relative error rate is utilized

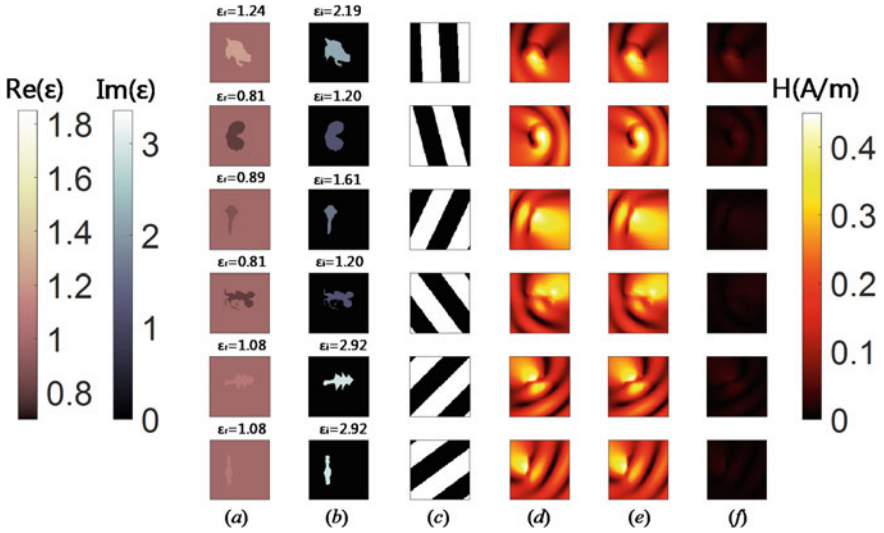


Fig. 4.19 Examples in Experiment Group #6. **a** The geometry and the relative permittivity of the scatterer, **b** the propagating direction and wavelength of the incident field, **c** the field calculated by the FDFD program, **d** the field predicted by the network, **e** the error between (**c**) and (**d**)

to measure the misfit between the ground truth and the prediction. This measurement of errors is from a holistic perspective and is not applicable for an uneven error distribution. For example, when a small part of the scattering field has a great error while the vast majority are exceedingly small, the average cannot reflect the error characteristics. To this scheme, statistical analysis is conducted on the results.

Here, the box plot [36–38] are utilized to analyze the result. In descriptive statistics, a box plot is a method for graphically depicting groups of numerical data through their quartiles. It also has lines extending from the boxes to indicate variability outside the upper and lower quartiles. Typically, the display scheme is based on a five-number summary, namely the minimum (Q_0), first quartile (Q_1), median (Q_2), third quartile (Q_3), and maximum (Q_4), respectively. The interquartile range (IQR) is defined as the distance between the third and first quartile.

$$IQR = Q_3 - Q_1 \tag{4.13}$$

The box is drawn from Q_1 to Q_3 with a horizontal line in the middle to denote the median. Besides, a whisker is drawn up from Q_3 to the maximum observed point within a distance of $1.5IQR$. Similarly, another whisker is drawn up from Q_1 to the minimum points with a distance of $1.5IQR$. The other points are drawn as outliers, demonstrating that the value is abnormal. In our studies, the error rates serve as the statistics to be considered, which are drawn as box plots to demonstrate the characteristics. In the following parts, the error rate distribution is analyzed on the representative Experient #3 and #6.

4.6.1 Experient Group #3

In Experient Group #3, 3600 samples are randomly selected from the test set. They are firstly partialized into five classes according to the geometric shapes, including circles, ellipses, triangles, quadrilaterals, and their combinations. The size and average error rate in each category is illustrated in Table 4.5. We use the professional statistical tools Origin® to draw the box plot, which is displayed in Fig. 4.20.

It can be concluded from Table 4.5 and Fig. 4.20 that the average error rate and the outlier ratio of the hybrid subsets are the highest, which is due to the highest complicacy. It is worth noting that the outlier ratio in all the subsets is lower than 6%, which proves the robustness of the model.

After investigating the error rate distribution of different geometric shapes, it will turn to different metal materials. The data set is classified into 8 categories in accordance with their permittivity. Table 4.6 exhibits the size and the average error

Table 4.5 The size and average error rate for different geometric shapes

Categories	Size	Average err (%)
Circle	600	0.482
Ellipse	600	0.537
Hybrid	1200	1.181
Quadrilateral	600	0.821
Triangle	600	0.660
Total	3600	0.792

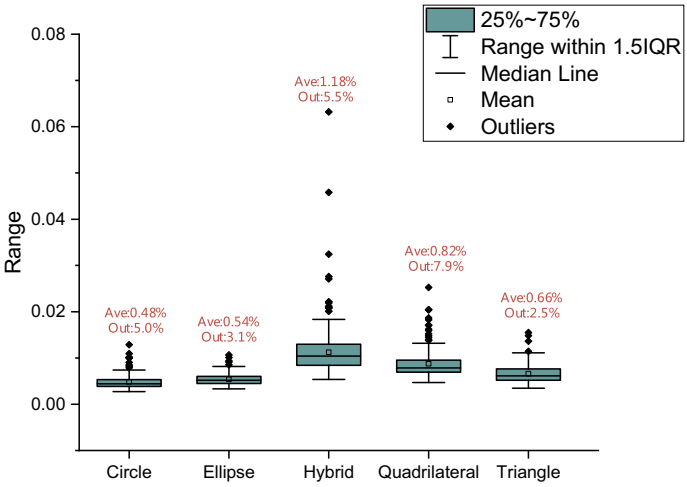


Fig. 4.20 Error rates distribution of different geometric shapes

Table 4.6 The size and the average error rate of different metal materials

Categories	Average err (%)	Categories	Average err (%)
Cu	0.907	Pt	1.127
Ag	0.691	Pd	0.604
Au	0.789	Hg	0.669
Rh	0.620	Ir	1.003

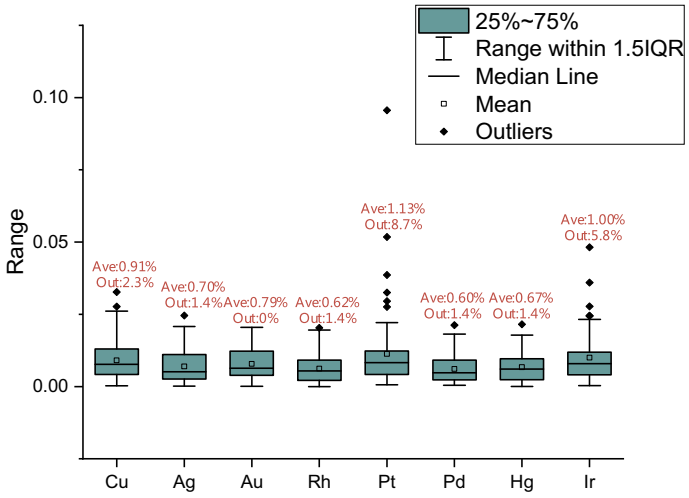


Fig. 4.21 Error rate distribution of different metal materials

rate in each category. Similarly, the result is drawn in the box plot as illustrated in Fig. 4.21.

It can be obtained from Table 4.6 and Fig. 4.21 that the difference of the average error rate in different subsets is not significant. Among all the metals, the platinum (Pt) has the highest outlier ratio due to the strongest scattering. Nevertheless, the ratio is still lower than 9%, which validates the reliability of the model.

4.6.2 Experient Group #6

In Experient Group #6, 500 samples are randomly selected from the open-source dataset. The overall error distribution is illustrated in Table 4.7 and Fig. 4.22.

Due to the complicated geometric shapes, both the error rate and the outlier ratio are higher than the previous experiment, but both are within the acceptable range. Similar to Experient Group #3, the error rate distribution of different materials is illustrated in Table 4.8 and Fig. 4.23.

Table 4.7 The size and the error rate on the open- source dataset

Categories	Size	Average err
Open-source dataset	500	4.909%

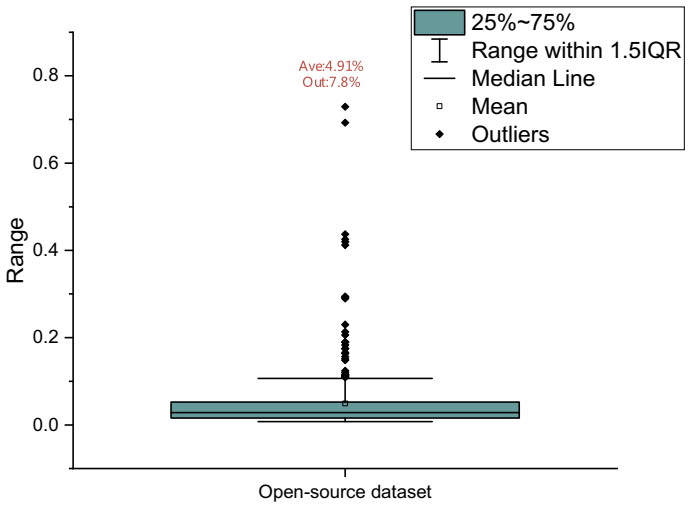


Fig. 4.22 Error rate distribution on open- source dataset

Table 4.8 Error rate of different metal materials on open-source dataset

Categories	Average err (%)	Categories	Average err (%)
Cu	5.173	Pt	9.163
Ag	3.002	Pd	3.749
Au	4.478	Hg	4.096
Rh	2.893	Ir	7.147

From Table 4.8 and Fig. 4.23, it is obvious that the difference of the average error rate in different subsets is not significant. However, the situation is vastly different for the outliers. Most materials share an error rate of less than 20%, which proves the resilience of the model. Nevertheless, it even jumps to around 80% for platinum. Although this metal has a strong scattering, such a high error rate is hard to stomach. Therefore, how to reduce the error rate of strong scattering materials is the effort direction of our future work.

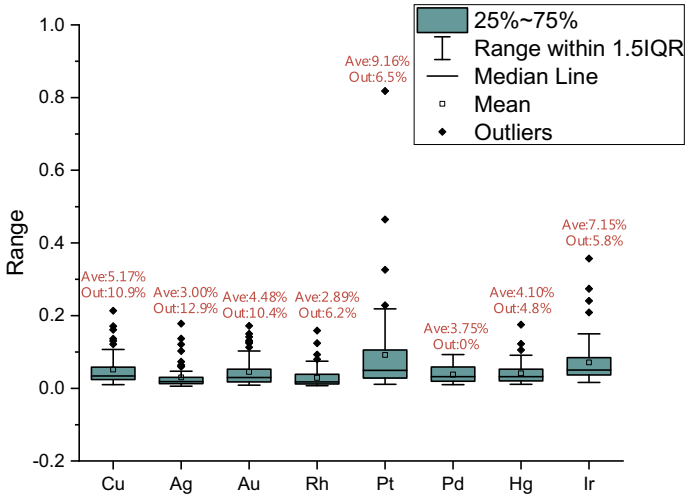


Fig. 4.23 Error rate distribution of different metal materials on open-source dataset

4.7 Summary

In this chapter, the process in constructing the 2D electromagnetic scattering solver based on the proposed EM net is illustrated in detail. The FDFD algorithm is utilized first to prepare the database being trained later. Then, the architecture of the EM-net is present. At the same time, several experiments are conducted to verify the effectiveness of modifications. After well trained, the test result exhibits that the proposed framework can accelerate the solving process by more than three orders of magnitude without sacrificing accuracy. Furthermore, an open-source data set is also introduced to validate the generalization ability. The outcome demonstrate that the DL network is capable to handle complex scatterers with completely different geometries. At the end of this chapter, the box plot is employed to analyze the statistical characteristics of the test set. Both the error rate and the outlier ratio present a good performance on most samples, which lays a solid foundation for the challenge of the 3D solver in Chap. 5.

References

1. Qi ST, Wang Y, Li YZ, Wu X, Ren Q, Ren Y (2020) Two-dimensional electromagnetic solver based on deep learning technique. IEEE J Multiscale Multiphys Comput Tech 5:83–88
2. Li YZ, Wang YP, Qi ST, Ren Q, Kang L, Campbell SD, Werner PL, Werner DH (2020) Predicting scattering from complex nano-structures via deep learning. IEEE Access 8:139983–139993
3. Rumble J (2020) CRC handbook of chemistry and physics. CRC Press, New York
4. Jonscher AK (1983) Dielectric relaxation in solids. Chelsea Dielectric Press, London

5. Young KF, Frederikse HPR (1973) Compilation of the static dielectric constant of inorganic solids. J Phys Chem Ref Data 2:313
6. American Institute of Physics, Gray DE (1972) American Institute of Physics handbook. Section editors: Billings BH [and others]. Coordinating editor: Gray DE, 3d edn. McGraw-Hill, New York
7. Cottancin E, Celep G, Lerme J, Pellarin M, Huntzinger JR, Vialle JL, Broyer M (2006) Optical properties of noble metal clusters as a function of the size: comparison between experiments and a semi-quantal theory. Theor Chem Acc 116(4–5):514–523. https://doi.org/10.1007/s00214-006-0089-1
8. Lipson SG, Lipson H (1981) Optical physics. Cambridge University Press, Cambridge
9. Motulevich GP, Malyshev VI, Skobel'tsyn DV (1973) Optical properties of metals. Proceedings (Trudy) of the P N Lebedev Physics Institute, vol 55. Consultants Bureau, New York
10. Palik ED, Ghosh G (1998) Handbook of optical constants of solids. Academic Press, San Diego
11. Rakić AD, Djurišić AB, Elazar JM, Majewski ML (1998) Optical properties of metallic films for vertical-cavity optoelectronic devices. Appl Opt 37(22):5271–5283. https://doi.org/10.1364/AO.37.005271
12. Herrera LJM, Arboleda DM, Schinca DC, Scaffardi LB (2014) Determination of plasma frequency, damping constant, and size distribution from the complex dielectric function of noble metal nanoparticles. J Appl Phys 116(23). Artn 233105. http://doi.org/10.1063/1.4904349
13. Johnson PB, Christy RW (1972) Optical constants of the noble metals. Phys Rev B 6(12):4370–4379. https://doi.org/10.1103/PhysRevB.6.4370
14. Dirac PAM (1958) The principles of quantum mechanics. International series of monographs on physics, 4th edn. Clarendon Press, Oxford
15. Speight JG (2005) Lange's handbook of chemistry. McGraw-Hill Education, New York
16. Balanis CA (2012) Advanced engineering electromagnetics, 2nd edn. Wiley, Hoboken, N.J.
17. Griffiths DJ (2017) Introduction to electrodynamics. Pearson, Boston
18. Jackson JD (1962) Classical electrodynamics. Wiley, New York
19. Shin W, Fan SH (2012) Choice of the perfectly matched layer boundary condition for frequency-domain Maxwell's equations solvers. J Comput Phys 231(8):3406–3431. https://doi.org/10.1016/j.jcp.2012.01.013
20. Salimans T, Karpathy A, Chen X, Kingma DP (2017) PixelCNN++: improving the PixelCNN with discretized logistic mixture likelihood and other modifications. ArXiv: 1701.05517
21. Gupta A, Shillingford B, Assael Y, Walters TC (2019) Speech bandwidth extension with wavenet. In: 2019 IEEE workshop on applications of signal processing to audio and acoustics (WASPAA), pp 205–208
22. Krizhevsky A, Sutskever I, Hinton G (2012) ImageNet classification with deep convolutional neural networks. Paper presented at the proceedings of the 25th international conference on neural information processing systems
23. Ronneberger O, Fischer P, Brox T (2015) U-net: convolutional networks for biomedical image segmentation. In: International conference on medical image computing and computer-assisted intervention. Springer, Berlin, pp 234–241
24. Lee B, Yamanakkanavar N, Choi JY (2021) Automatic segmentation of brain MRI using a novel patch-wise U-net deep architecture (vol 15, e0236493, 2020). Plos One 16(1). ARTN e0246105. http://doi.org/10.1371/journal.pone.0246105
25. Saood A, Hatem I (2021) COVID-19 lung CT image segmentation using deep learning methods: U-Net versus SegNet. Bmc Med Imaging 21(1). ARTN 19. http://doi.org/10.1186/s12880-020-00529-5
26. Zheng S, He ZZ, Liu HL (2021) Generating three-dimensional structural topologies via a U-Net convolutional neural network. Thin Wall Struct 159. ARTN 107263. http://doi.org/10.1016/j.tws.2020.107263
27. Dumoulin V, Visin F (2016) A guide to convolution arithmetic for deep learning. ArXiv: 1603.07285v2

28. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. Paper presented at the proceedings of the IEEE conference on computer vision and pattern recognition
29. Chung J, Gulcehre C, Cho K, Bengio Y (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. Computer Science
30. Wang HZ, Wang Y, Zhang Q, Xiang SM, Pan CH (2017) Gated convolutional neural network for semantic segmentation in high-resolution images. Remote Sens 9(5):446
31. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M (2016) Tensorflow: a system for large-scale machine learning. Paper presented at the 12th symposium on operating systems design and implementation, Savannah, GA
32. Atienza R (2020) Advanced deep learning with TensorFlow 2 and Keras: Apply DL, GANs, VAEs, deep RL, unsupervised learning, object detection and segmentation, and more. Packt Publishing Ltd., Birmingham
33. Chollet F (2017) Deep learning with python. Manning Publications, New York
34. Efron B, Hastie T (2016) Computer age statistical inference: algorithms, evidence, and data science. Institute of Mathematical Statistics monographs. Cambridge University Press, New York, NY
35. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press, Cambridge, Massachusetts
36. Bickel PJ, Doksum KA (2015) Mathematical statistics. Basic ideas and selected topics. Chapman and Hall/CRC, London
37. Hastie T, Tibshirani R, Friedman JH (2009) The elements of statistical learning: data mining, inference, and prediction. Springer series in statistics, 2nd edn. Springer, New York, NY
38. James G, Witten D, Hastie T, Tibshirani R (2013) An introduction to statistical learning: with applications in R. Springer texts in statistics, vol 103. Springer, New York

Chapter 5
Three-Dimensional Electromagnetic Scattering Solver

In Chap. 4, the process in predicting the 2D scattering field through deep learning technique is discussed. In this chapter, it will concentrate on the 3D case. Similar to Chap. 4, the database preparing, training and validation will be introduced, respectively. Firstly, the scatterers are generated with the algorithm illustrated in Chap. 3. They will be fed into the 3D-FDFD program with the incident field to generate sufficient training data. The proposed framework is trained to excavate the internal relations between the scatterer and the scattering field. In this chapter, how to choose applicable model parameters in pre-experiments are also investigated. The final results on the test set are illustrated at the end of this chapter.

5.1 Database Preparing

5.1.1 Problem Description

At the very beginning, it is indispensable to have a brief introduction to the 3D scattering model. As illustrated in the previous chapter, we mainly aim at solving the scattering field with an incident plane wave. The computational region is a cube with the side length of 32 nm. The region is meshed into $32 \times 32 \times 32$ grids, each of which is 1 nm^3. In each grid, it is assumed that the electromagnetic parameters and field quantities are uniform. The center of the region is set as (0, 0, 0) as shown in Fig. 5.1.

To obtain the database, a large number of scatters and the corresponding scattering fields are generated.

Q. Ren et al., *Sophisticated Electromagnetic Forward Scattering Solver via Deep Learning*, https://doi.org/10.1007/978-981-16-6261-4_5

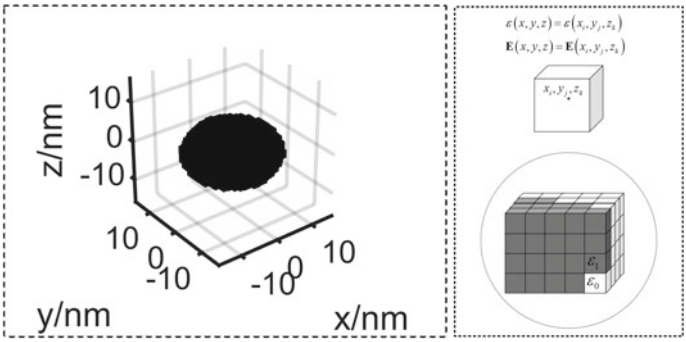


Fig. 5.1 The computational region. The electromagnetic parameters and field quantities in a grid are supposed to be uniform

5.1.2 Geometries of the Scatterers

In constructing the 3D scatterer database, it is necessary to generate adequate geometries. The generating algorithm has been discussed detailly in Chap. 3 thus will not be repeated here. The 3D geometries in the database include spheres and ellipsoids. Two sub-classes are considered in this chapter. In the first one, the scatterers are laid up at the center of the region. Figure 5.2 shows several examples in this case.

The scatterers in the second subclass are randomly placed in the cube, some of which are displayed in Fig. 5.3.

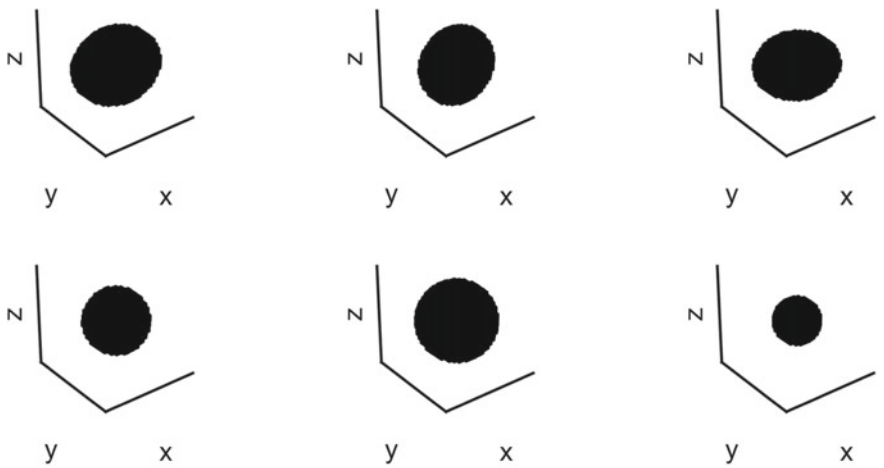


Fig. 5.2 Several examples in the first subclass of the data set, in which the scatterers are laid up at the center of the region

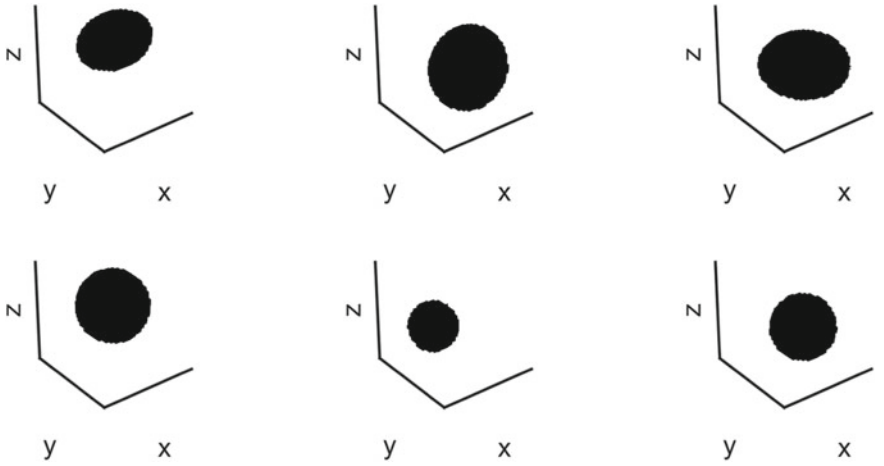


Fig. 5.3 Several examples in the second subclass of the data set, in which the scatterers are randomly placed in the region

5.1.3 Permittivity of the Scatterers

After generating these geometries, it is necessary to assign permittivity to the scatterers. According to Table 4.1, the relative permittivity can be divided into 2 categories, a constant of 4 and a random integer ranging from 2 to 5.

Combining the geometries and permittivities, three data sets are constructed, namely D5, D6 and D7, respectively. In D5, the scatterer is placed at the center, with a permittivity of 4. Figure 5.4 displays several examples in D5.

In D6, the scatterer is placed at the center, with a permittivity ranging from 2 to 5. Figure 5.5 displays several examples in D6.

In D7, the scatterer is randomly placed in the cube, with a permittivity ranging from 2 to 5. Figure 5.6 displays several examples in D7.

5.1.4 Illumination Setting

In the forward pass of the 3D scattering process, the incident wave is a TEM wave. As illustrated in Chap. 1, a TEM wave refers to the electromagnetic wave without electric field and magnetic field in the propagation direction [1, 2]. Particularly, Fig. 5.7a demonstrate a TEM wave propagating along $+z$ direction with the electric field polarizing along x axis. More generally, Fig. 5.7b demonstrate a TEM wave with random propagating and polarizing direction.

Obviously, it is resource-demanding to handle an arbitrary direction wave. However, through a simple transformation of the coordinate system [3, 4], it is possible to alter situation (b) into (a). Suppose O is the global coordinate system,

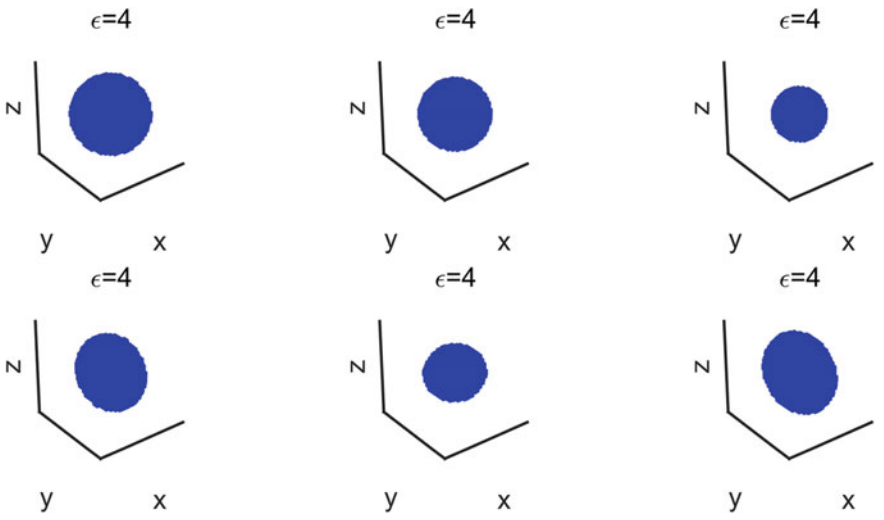


Fig. 5.4 Several examples in D5. The scatterer is placed at the center with a permittivity of 4

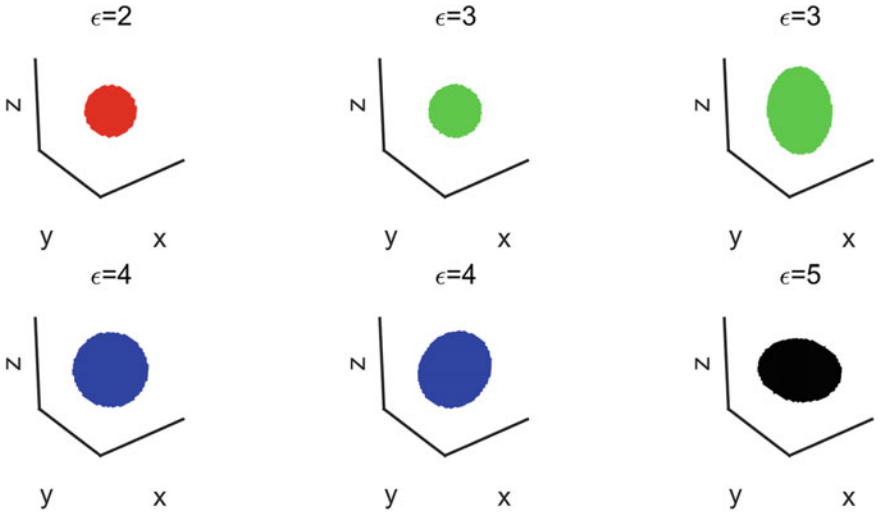


Fig. 5.5 Several examples in D6. The scatterer is placed at the center, with a permittivity ranging from 2 to 5

where the coordinates of each point reflect the true position. Corresponding, O' is the local coordinate system, whose x, y axes are defined as the polarization direction of the electric field and magnetic field while the z-axis is defined as the propagating direction. The coordinates of a point are denoted as (x_0, y_0, z_0) and (x_1, y_1, z_1) in the global and local coordinates, respectively. Besides, the base vectors in O' are

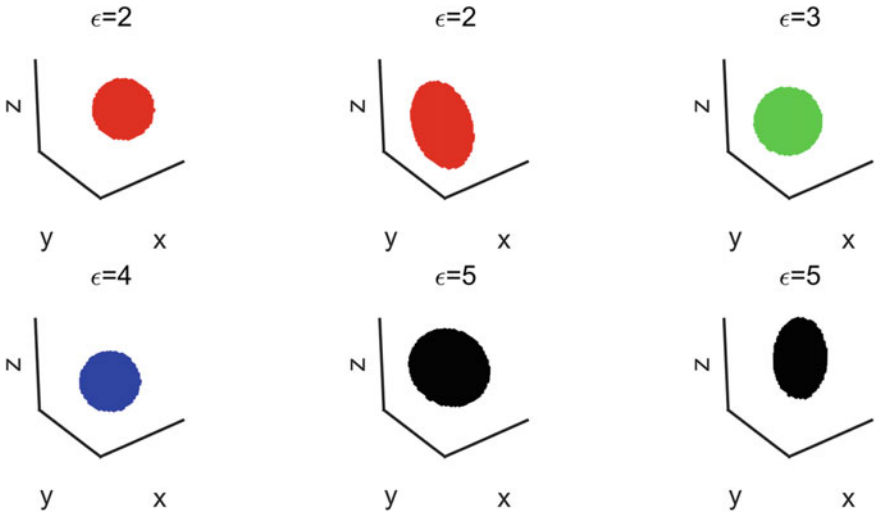


Fig. 5.6 Several examples in D6. The scatterer is randomly placed in the region, with a permittivity ranging from 2 to 5

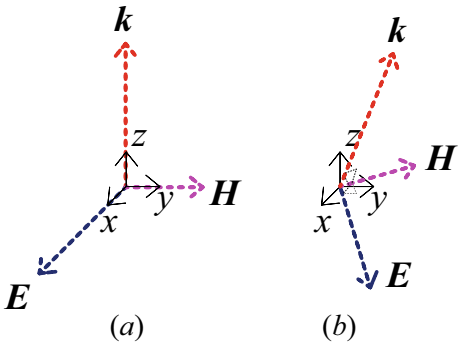


Fig. 5.7 A TEM wave. **a** propagating along $+z$ direction with the electric field polarizing along x axis and **b** propagating and polarizing along an arbitrary direction

defined as

$$\mathbf{i_1} = \frac{\mathbf{E}}{|\mathbf{E}|}, \tag{5.1}$$

$$\mathbf{i_2} = \frac{\mathbf{H}}{|\mathbf{H}|}, \tag{5.2}$$

$$\mathbf{i_3} = \frac{\mathbf{k}}{|\mathbf{k}|}. \tag{5.3}$$

The relationship between the base vectors in local and global coordinate system are

$$\begin{bmatrix} \mathbf{i_1} \\ \mathbf{i_2} \\ \mathbf{i_3} \end{bmatrix} = \begin{bmatrix} g_{11} & g_{12} & g_{13} \\ g_{21} & g_{22} & g_{23} \\ g_{31} & g_{32} & g_{33} \end{bmatrix} \begin{bmatrix} \mathbf{i_x} \\ \mathbf{i_y} \\ \mathbf{i_z} \end{bmatrix} = \mathbf{G} \begin{bmatrix} \mathbf{i_x} \\ \mathbf{i_y} \\ \mathbf{i_z} \end{bmatrix}, \tag{5.4}$$

where $(g_{i1}, g_{i2}, g_{i3})(i = 1, 2, 3)$ are the coordinates of (x_1, y_1, z_1) in the global coordinate system$O$. G is the transition matrix between the two sets of bases. For any point (x_0, y_0, z_0) in the global coordinate system, the electric field can be expressed as

$$\mathbf{E}(x_0, y_0, z_0) = E_x\mathbf{i_x} + E_y\mathbf{i_y} + E_z\mathbf{i_z} = \begin{bmatrix} E_x\mathbf{i_x} \\ E_y\mathbf{i_y} \\ E_z\mathbf{i_z} \end{bmatrix}, \tag{5.5}$$

where E_z, E_y, E_z are the field components to be calculated. As (x_1, y_1, z_1) can be obtained by

$$\begin{bmatrix} x\mathbf{i_x} \\ y\mathbf{i_y} \\ z\mathbf{i_z} \end{bmatrix} = \begin{bmatrix} x_0 & 0 & 0 \\ 0 & y_0 & 0 \\ 0 & 0 & z_0 \end{bmatrix} \begin{bmatrix} \mathbf{i_x} \\ \mathbf{i_y} \\ \mathbf{i_z} \end{bmatrix} = \begin{bmatrix} x_0 & 0 & 0 \\ 0 & y_0 & 0 \\ 0 & 0 & z_0 \end{bmatrix} G^{-1} \begin{bmatrix} \mathbf{i_1} \\ \mathbf{i_2} \\ \mathbf{i_3} \end{bmatrix} = \begin{bmatrix} x_1\mathbf{i_1} \\ x_2\mathbf{i_2} \\ x_3\mathbf{i_3} \end{bmatrix}, \tag{5.6}$$

it is feasible to obtain the field at (x_1, y_1, z_1) through the 3D-FDFD program. Utilizing (5.4), we have

$$\mathbf{E}(x_1, y_1, z_1) = \begin{bmatrix} E_1 & 0 & 0 \\ 0 & E_2 & 0 \\ 0 & 0 & E_3 \end{bmatrix} \begin{bmatrix} \mathbf{i_1} \\ \mathbf{i_2} \\ \mathbf{i_3} \end{bmatrix} = \begin{bmatrix} E_1g_{11} & E_1g_{12} & E_1g_{13} \\ E_2g_{21} & E_2g_{22} & E_2g_{23} \\ E_3g_{31} & E_3g_{32} & E_3g_{33} \end{bmatrix} \begin{bmatrix} \mathbf{i_x} \\ \mathbf{i_y} \\ \mathbf{i_z} \end{bmatrix}. \tag{5.7}$$

As $\mathbf{E}(x_0, y_0, z_0) = \mathbf{E}(x_1, y_1, z_1)$, it is practicable to obtain the electric field components at any point (x_0, y_0, z_0) by

$$\begin{cases} E_x = E_1g_{11} + E_2g_{21} + E_3g_{31} \\ E_y = E_1g_{12} + E_2g_{22} + E_3g_{32} \\ E_z = E_1g_{13} + E_2g_{23} + E_3g_{33} \end{cases}. \tag{5.8}$$

Therefore, it is reasonable to only consider the wave propagating along +z direction with its electric field polarizing along x axis. The wavelength is 20 nm, around 2/3 of the side length. As the direction of incident field is fixed, we merely match the scatterer and the field pattern through the training process.

5.1.5 Calculation of the Model

With the 3D FDFD program [5–7] introduced in Chap. 3, it is feasible to obtain the scattering field distribution. Slightly different from the 2D case, the field quantity cannot be acquired directly. In fact, the generating process contain 2 steps. The pre-processing program is employed first to generate the grids. Then the processed data is fed to the solver and calculate the field quantities. Both two steps are carried out on Ubuntu 20.02. Undoubtedly, the computing time of the 3D cases will be vastly increased compared to the 2D cases.

5.2 Pre-experiments

Pre-experiment is to train a small number of samples with changeable hyperparameters, split ratios and activation functions before the formal experiment, so as to find out the best experimental conditions and to lay the foundation for the formal experiment. Therefore, it can avoid the waste of computing resources due to the poorly designed experiments. For the sake of simplicity, it firstly concentrates on the learning rate, batch size, and decay rate and then turned to the split ratio and activation function [8–10].

5.2.1 Learning Rate

In deep learning, the learning rate is a parameter in the optimization algorithm that determines the step size at each iteration while moving toward a minimum of a loss function [11–13]. Generally, the learning rate satisfies

$$\theta = \theta - \alpha \frac{\partial}{\partial \theta} J(\theta), \tag{5.9}$$

where α, θ, $J(\theta)$ are the learning rate, the parameter to be learned and the loss function, respectively. It can be seen that α metaphorically decides the speed of how fast the model updates. In fact, there is a trade-off between convergence and overshooting in selecting the learning rate. Therefore, it is arduous to configure an appropriate value. Figure 5.8 illustrates the different situations we may encounter when determining the learning rate.

It can be obtained from Fig. 5.8 that a too-high learning rate induces the learning jump over the minima while a too-slow one either spends too long to converge or is stuck at the local minima [14, 15]. Therefore, an elaborate learning rate implies that we can gain a better convergence in a shorter time. Searching for such a proper value is a sophisticated and repetitive task. Nevertheless, it is achievable to acquire a relatively precise learning rate via a series of experiments. In our experiments, the

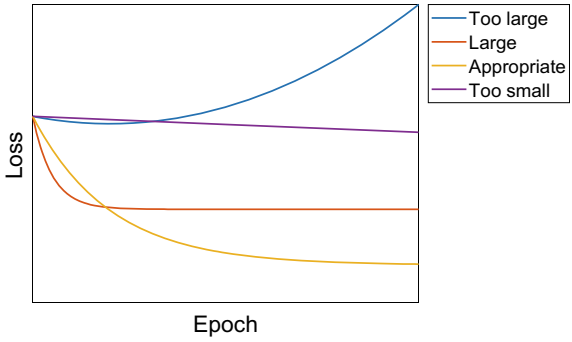


Fig. 5.8 The influence of different learning rates on the training process

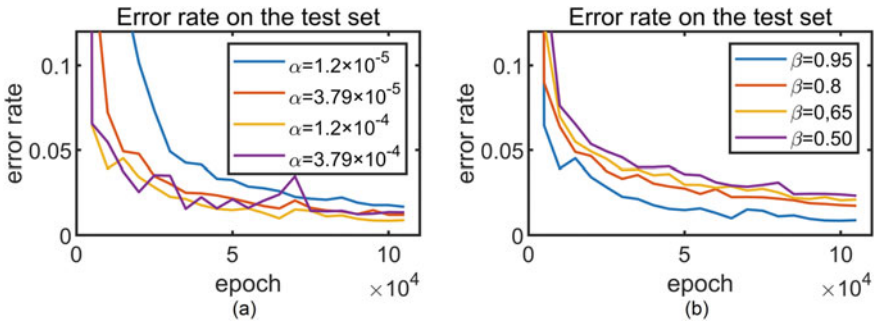


Fig. 5.9 The influence of the hyperparameters on the training process. **a** The learning rate α and **b** the decay rate β

learning rate ranges among 1.2×10^{-5}, 3.79×10^{-5}, 1.2×10^{-4} and 3.79×10^{-4}. In order to assure the rigorous experimental conditions, the batch size is fixed at 16 while the decay rate at 0.95.

As is illustrated in Fig. 5.9a, when the learning rate is 3.79×10^{-4}, the training process is exceedingly unstable. While when it is less than 3.79×10^{-5}, the convergence process is time-consuming. Therefore, we compromise to set the learning rate to be 1.2×10^{-4}.

5.2.2 Decay Rate

The learning rate decay is a de facto technique in modern deep learning which begins with a large learning rate and then decays multiple times [15, 16]. It is empirically observed to have positive impact on both optimization and generalization. However, seeking for a suitable decay rate is challengeable. Here, a reasonable value is used through experiments. In these studies, the decay rate begins at 0.95 and gradually

reduces to 0.50. To assure the rigorous experimental conditions, the learning rate is fixed at 1.2×10^{-4} while the batch size is set as 16.

The results are exhibited in Fig. 5.9a. It can be summarized that with the decreasing decay rate, the error rate can converge slower. Therefore, we assign the decay rate to be 0.95 in the formal experiments.

5.2.3 Batch Size

Batch size is another crucial hyperparameter to tune in modern deep learning techniques [17, 18], which defines the number of samples that will be propagated through the network at one time. Generally, this technique provides a less resource-demanding training process. It is essential to introduce this concept when it is not able to fit the whole dataset in the memory. In fact, selecting an appropriate batch size is exceedingly important. With mini-batches, typically networks train faster while fluctuating much more. Practitioners often want to use a larger batch size to train their model as it allows computational speedups from the parallelism of GPUs. However, it is well known that a too large batch size will induce poor generalization ability. This is intuitively explained by the fact that smaller batch sizes permit the framework to "begin learning before having to see all the data." Thus, for convex loss functions to be optimized, there is an inherent tug-of-war between the benefits of smaller and bigger batch sizes.

Here, under little computational constraints, it is often advised to start training at a small batch size, reaping the benefits of faster training dynamics, and steadily grows the batch size through training, also reaping the benefits of guaranteed convergence. In our studies, the batch size begins at 4 and gradually increases to 32. To assure the rigorous experimental conditions, the learning rate is fixed at 1.2×10^{-4} while the decay rate at 0.95.

Figure 5.10a, b demonstrate the error rate and training time with different batch sizes. It can be concluded that with the increment of the batch size, the error rate can

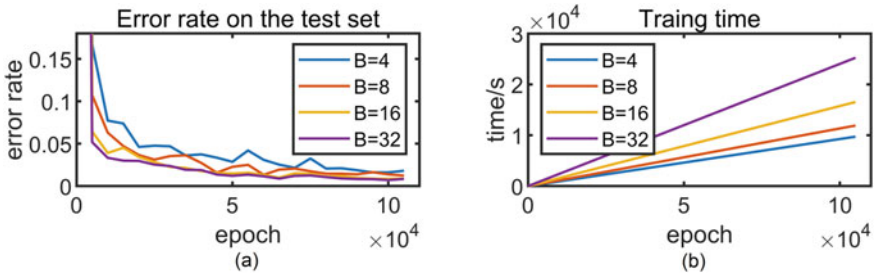


Fig. 5.10 The influence of the batch size B on the training process. **a** The error rate and **b** the training time

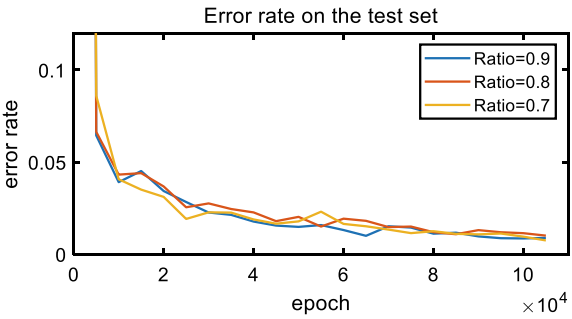


Fig. 5.11 Error rates of different split ratios

converge to a lower level. However, the training time raises rapidly at the same time. Therefore, we compromise to set the batch size to be 16 in the formal experiments.

5.2.4 Split Ratio

The validation process requires a separated dataset supposed to be independent to the training dataset [19]. A common pattern is to divide the data set into the training set and test set with a certain ratio. This split ratio can vary with the size of the dataset. Generally speaking, a data set with the size of less than 1000 samples has a larger validation set, the percentage of which can be as large as 1/4. When it comes to a set with millions of specimens, 1% is a suggested ratio of partition. To determine the impact of the split ratio on training, a series of experiments have been conducted to figure out an appropriate value.

The controlling variable method is employed here. The pre-experiment is carried out on Experiment Group #4, the scatterers of which are composed of eight kinds of metal materials. The initial learning rate is set to be 1.2×10^{-4} with a decay rate of 0.95 while the batch size is 16. The results are compared when the training ratio is 90%, 80% and 70%, respectively. The error rate curve of different split ratios with epochs is illustrated in Fig. 5.11.

It can be concluded that the proposed model is insensitive to the split ratio. Here, it is set to be 0.9 in the formal experiments.

5.2.5 Activation Function

In neural networks, the activation function of a node determines the output of the node for a given input [20–23]. In fact, only nonlinear activation functions allow neural networks to use a few nodes to solve nontrivial problems. Generally speaking, activation functions can be divided into three categories: ridge, radial and fold activation functions. The ridge activation function, a single variable function acting on the linear

combination of input variables, is the most extensively employed. It is biologically inspired and represents the firing rate of neurons. The activation function has the following mathematical characteristics.

Nonlinearity: it can be proved that a two-layer network containing a nonlinear activation function can be a universal function approximator. The network composed of multi-layer identical activation function is equivalent to a two-layer network.

Continuous differentiability: the activation functions are continuous differentiable except for finite points of discontinuity, ensuring the reliability of the gradient algorithm.

Monotone: when the activation function is monotone, the loss function of the system can be guaranteed to be convex, which is suitable for convex optimization.

Several commonly used activation functions are introduced and their performance will be investigated in the following experiments. Three basic activation functions will be compared first and then the three modified ones will be discussed.

5.2.5.1 Basic Activation Functions

The Sigmoid, Tanh and ReLU activation functions will be explained, respectively. The Sigmoid was first drawn into machine learning by Hinton et al. [24] and used in automatic speech recognition. It can be expressed as

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{5.10}$$

Figure 5.12 demonstrates the sigmoid function and its derivative. It can be seen from Fig. 5.12 that the Sigmoid function maps $[-\infty, +\infty]$ to $[0, 1]$. It is worth pointing out that the output of the sigmoid function is always positive, so that each parameter updates in the same direction during an iteration (increase or decrease at the same time), resulting in a tortuous optimization process.

The Tanh function was widely used in natural language processing. It has a similar expression to the Sigmoid, which is defined by

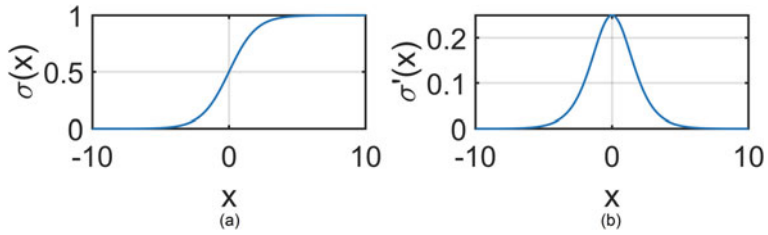


Fig. 5.12 The Sigmoid activation function. **a** The Sigmoid function and **b** the derivative of the Sigmoid function

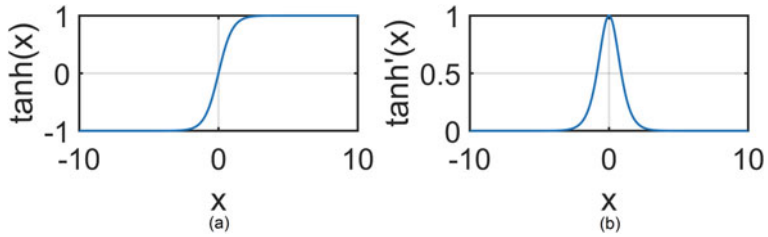


Fig. 5.13 The Tanh activation function. **a** The tanh function and **b** the derivative of the tanh function

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{5.11}$$

This function is also monotonically increasing, mapping $[-\infty, +\infty]$ to $[-1, 1]$. Figure 5.13 demonstrates the Tanh function and its derivative.

It can be observed that the gradient of Tanh function ranges from 0 to 1. In most optimization algorithms, the parameters updating process is completely dependent on the gradient value. In such methods, each of the weights in the framework receives an update proportional to the partial derivative of the loss function with respect to the current value in each iteration process. As the backpropagation computes gradients by the chain rules, the gradient decreases exponentially with n by multiplying these small numbers together. Under this circumstance, the gradient will be vanishingly small, effectively impeding the weight from changing its value. In the worst case, it may entirely stop the network from further training. This is called the vanishing gradient problem. In order to avoid this phenomenon, an activation function called ReLU is proposed. It was first introduced by Hahnloser et al. [25, 26] in 2000 with biological motivations and mathematical justifications. The famous applications include the Alex-net [27] in computer vision in 2012 and Resnet [28] in 2015. This structure and its variations have become the most widely used activation functions in deep learning [29, 30]. The ReLU function is defined as

$$\mathrm{ReLU}(x) = \begin{cases} 0 \ for \ x < 0 \\ x \ for \ x \geq 0 \end{cases} \tag{5.12}$$

Figure 5.14 illustrates ReLU and its derivative. It is noteworthy that the gradient of ReLU can only be taken as 0 or 1, which ensures that the gradient multiplication will never converge to 0, and the result can only be taken as 0 or 1. If the value takes 1, the gradient keeps unchanged and continues forward propagation; while if the value is 0, the gradient stops propagation there. In addition, the ReLU function is one-sided saturated (the value of the function equals to zero when the input is negative), so the neuron is more robust to noise. Besides, ReLU is computationally efficient in calculating the derivative compared to Sigmoid or tanh function, as it can only takes 0 or 1. By truncating the negative value, it accelerates the propagating process.

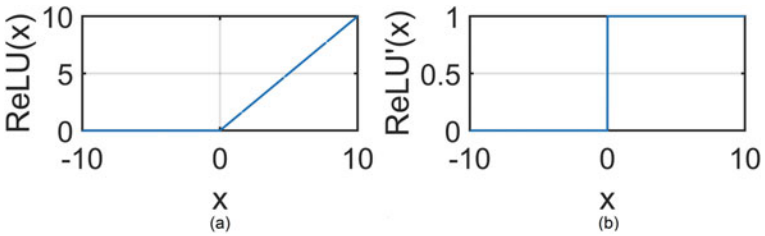


Fig. 5.14 The ReLU activation function. **a** The ReLU function and **b** the derivative of the ReLU function

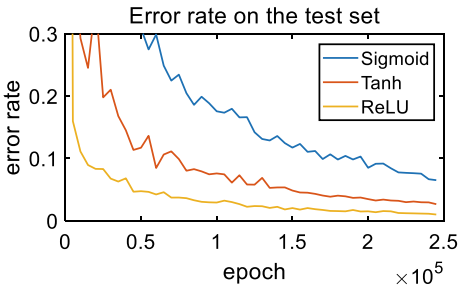


Fig. 5.15 Error rates of different activation functions

In the EM-net [31, 32], each convolution/transposed layer is followed by an activation function to add nonlinearity. Experiments are conducted to measure the performance of the Sigmoid, tanh and ReLU activation functions, respectively. In order to ensure the rationality of the experiment, the controlling variable method is adopted. The pre-experiment was carried out on Experiments Group #4, the scatterers of which are composed of eight kinds of metal materials. The initial learning rate is set to be 1.2×10^{-4} with a decay rate of 0.95 while the batch size is 16.

Figure 5.15 illustrates the error rates on the test set of three functions. It can be concluded that the ReLU performs best among those functions, which is consistent with the previous theoretical analysis.

5.2.5.2 Modified Activation Functions

Although the ReLU activation function often performs better than the Sigmoid and Tanh functions, it also has inherent defects, which is the so-called dying problem. That is, neurons are sometimes pushed into states in which they become inactive for essentially all inputs. In this state, no gradients flow backward through the neuron, and therefore the neuron becomes stuck in a perpetually inert state. This phenomenon can be alleviated by introducing the modified activation functions, which assign a small positive slope for $x < 0$.

The PReLU [33] is defined as

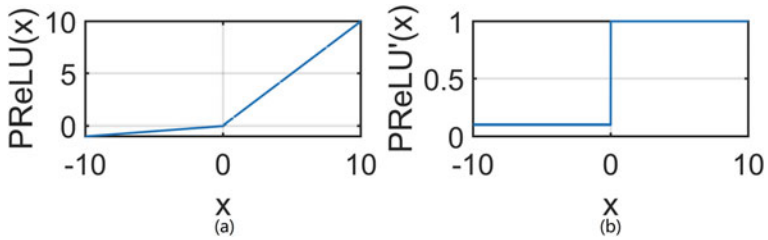


Fig. 5.16 The PReLU activation function. **a** The PReLU function and **b** the derivative of the PReLU function

$$\mathrm{PReLU}(x) = \begin{cases} \alpha x \ \ for \ x < 0 \\ x \ \ for \ x \geq 0 \end{cases}. \tag{5.13}$$

When $\alpha = 0.01$, it degrades to LeakyReLU [34]. Compared to the traditional ReLU, the PReLU introduces an extra parameter to learn, overcoming the dying problem at the same time. Figure 5.16 illustrates PReLU and its derivative when $\alpha = 0.1$.

Although the PReLU settles the dying problem, it is not a one-sided saturated activation function. Therefore, the ELU [35] comes on stage. This function is defined as

$$\mathrm{ELU}(x) = \begin{cases} \alpha(e^x - 1) \ \ for \ x < 0 \\ x \ \ for \ x \geq 0 \end{cases}. \tag{5.14}$$

Figure 5.17 illustrates ELU and its derivative when $\alpha = 1$. It can be obtained that the ELU function is fully continuous differentiable. Compared to the traditional ReLU, the ELU not only solves the dying problem, but also tends to converge faster and achieves a better generalization ability.

The performance of the ReLU, PReLU, and ELU activation functions are compared. The controlling variable method is adopted to ensure the rationality of the

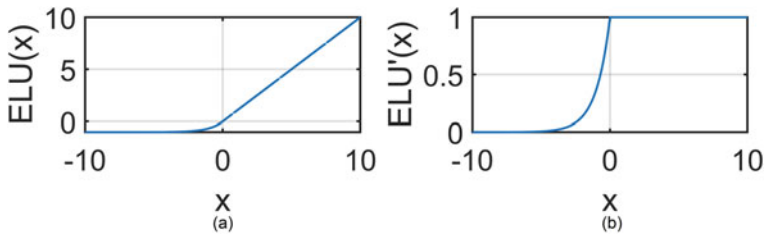


Fig. 5.17 The ELU activation function. **a** The ELU function and **b** the derivative of the ELU function

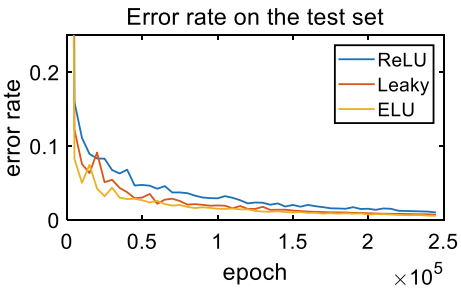


Fig. 5.18 Error rates of different activation functions

experiment. The pre-experiment was carried out on Experiment Group #4, the scatterers of which are composed of eight kinds of metal materials. The initial learning rate is set to be 1.2×10^{-4} with a decay rate of 0.95 while the batch size is 16.

The error rates of the test set on the three modified activation functions are presented in Fig. 5.18. The results show that there is little difference among these activation functions, while the ELU is slightly better. Therefore, the ELU is adopted as the nonlinear unit between the convolution/transposed layers. The experimental results are consistent with the previous theoretical analysis.

5.3 Training of the DL Framework

After generating the dataset, it is feasible to embark on the training process. The deep-learning framework is performed on TensorFlow 2.0 [36–38]. This newer version provides the researchers a more flexible environment to debug.

The aim of the training is to optimize the loss function, which can be defined as

$$Loss = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} |F_{FDFD}(i, j, k) - F_{Framework}(i, j, k)|^2, \tag{5.15}$$

where F_{FDFD} refers to the electromagnetic field calculated by the 3D-FDFD program, while the $F_{Framework}$ is predicted by the network. The Adam optimizer is utilized to reduce the loss function in the iterative process. With the sharp rise of the computation complexity, a mighty processor is indispensable. The training is performed on an NVIDIA RTX 2080Ti graphic card on Dell Precision 7920 Tower. Compared to GTX 1080Ti applied in the previous experiments, RTX 2080Ti has an average performance improvement of 41% on FP32 computing [39]. The sufficient computing capability ensures the stability of training.

5.4 Result on the Test Set

With the architecture proposed in the previous chapter and reasonable experimental conditions determined by the pre-experiment, it is essential to exam the prediction ability of the framework. A series of experiments are carried out to measure the performance of the proposed network in different situations quantitatively. Similar to the 2D case, we employ the relative average error rate defined in Eq. 4.11. As illustrated in the previous section, we mainly concentrate on the impact of the scatterer. The concrete experimental conditions are listed in Table 5.1. In each experiment, some of the results randomly chosen from the test set are displayed.

5.4.1 Experiment Group #7

In Experiment Group #7, the incident wave is a TEM wave propagating along the $+z$ direction and polarizing along $+x$ direction (electrical field) in the local coordinate system O'. The scatterers are spheres or ellipsoids located at the center with the relative permittivity of 4. The data set contains a total of 6000 samples, 10% of which are the test sets. The curves of the error rate and the loss with the training epochs are illustrated in Fig. 5.19.

After 210,000 rounds of training in 2 h, the error rate on the test set reduces to 0.89%, while the loss decreases to 11.98. Figures 5.20 and 5.21 exhibit several samples randomly chosen in the test sets. Here, (a) reflects the geometry and relative

Table 5.1 Experimental conditions

Experiment group index	Database	Position of the scatter	Relative permittivity
#7	D5	Center	4
#8	D6	Center	2/3/4/5
#9	D7	Arbitrary	2/3/4/5

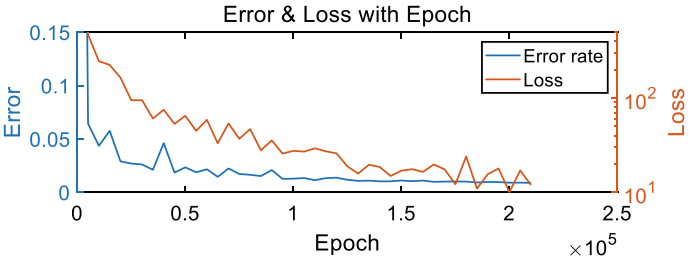


Fig. 5.19 The error rate and the loss with the training epochs on Experiment Group #7

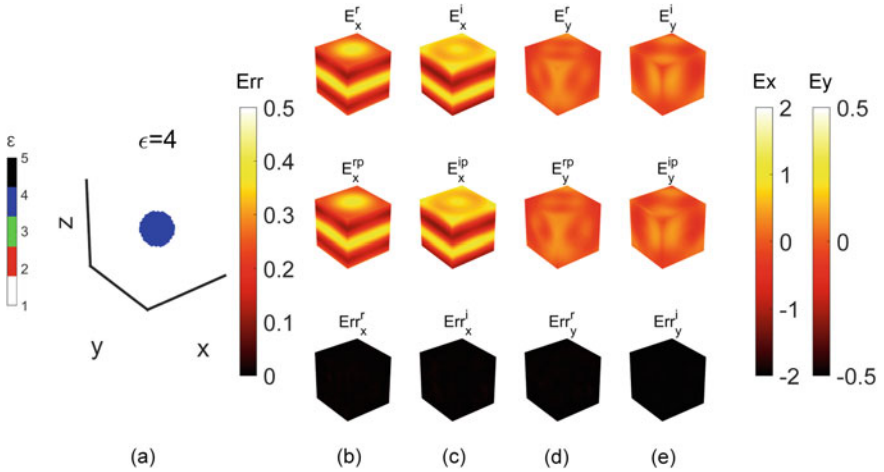


Fig. 5.20 An example in Experiment Group #7 (Sphere). **a** The geometry and the relative permittivity of the scatterer; **b** the ground truth, the prediction and the absolute error of $\mathfrak{Re}(E_x)$; **c** the ground truth, the prediction and the absolute error of $\mathcal{I}m(E_x)$; **d** the ground truth, the prediction and the absolute error of $\mathfrak{Re}\big(E_y\big)$ and **e** the ground truth, the prediction and the absolute error of $\mathcal{I}m\big(E_y\big)$

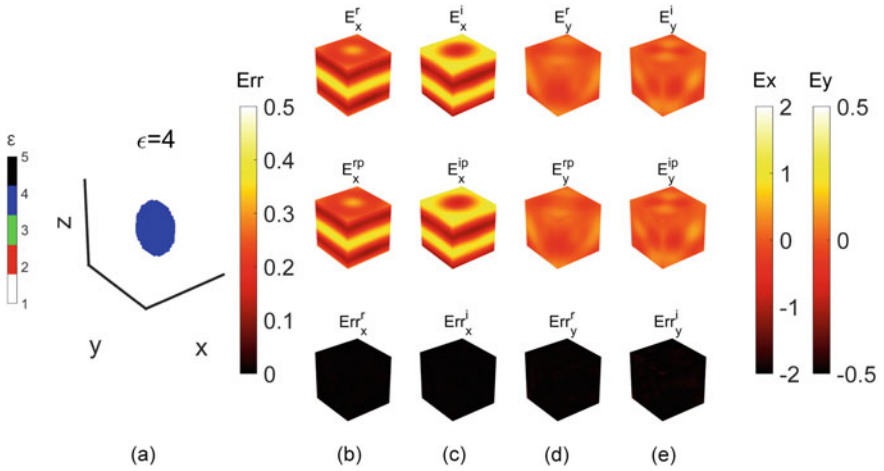


Fig. 5.21 Another example in Experiment Group #7 (Ellipsoid). **a** The geometry and the relative permittivity of the scatterer; **b** the ground truth, the prediction and the absolute error of $\mathfrak{Re}(E_x)$; **c** the ground truth, the prediction and the absolute error of $\mathcal{I}m(E_x)$; **d** the ground truth, the prediction and the absolute error of $\mathfrak{Re}\big(E_y\big)$; **e** the ground truth, the prediction and the absolute error of $\mathcal{I}m\big(E_y\big)$

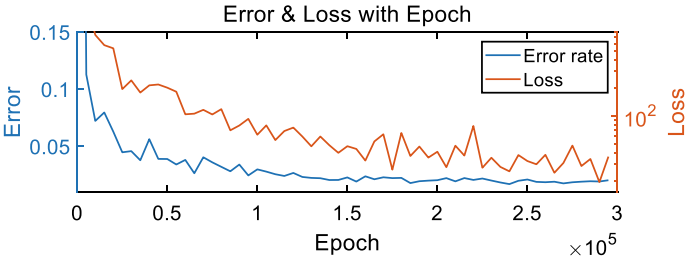


Fig. 5.22 The error rate and the loss with the training epochs on Experiment Group #8

permittivity of the scatterer, (*b*) and (*c*) represent the ground truth, prediction and absolute error of the real/imaginary part of E_x while (*c*) and (*d*) are those of E_y.

It can be concluded that the prediction of the framework is consistent with the numerical solution by the 3D-FDFD program.

5.4.2 *Experiment Group #8*

In Experiment Group #8, the incident wave is the same as Experiment Group #8. The scatterers have the same geometries and locations with a relative permittivity ranging from 2 to 5. The data set contains a total of 8000 specimens, and 800 are the test sets. Figure 5.22 exhibits the curves of the error rate and the loss with the training epochs.

Since the dataset is larger and more complex, it takes 29,500 epochs and about 3 h for the framework to reach convergence. The final error rate and loss on the test set are 2.0% and 30, respectively. Some of the specimens randomly selected in the test set are presented in Figs. 5.23 and 5.24. Here, (*a*) demonstrates the geometry and the relative permittivity of the scatterer; (*b*) and (*c*) reflect the ground truth, prediction and the absolute error of the real/imaginary part of E_x while (*d*) and (*e*) are those of E_y.

By comparing the outcome, the conclusion can be drawn that the prediction of the framework coincides with the numerical solution by the 3D-FDFD program.

5.4.3 *Experiment Group #9*

In Experiment Group #9, the incident wave is the same as the two previous experiments. It is worth noting that although the scatterers are still spheres and ellipsoids, their positions are randomly distributed in the cube. As this dataset contains more complex scatterers, more samples are required for training. In fact, the whole data set includes a total of 24,000 samples, while 2400 serve as test examples. Similarly,

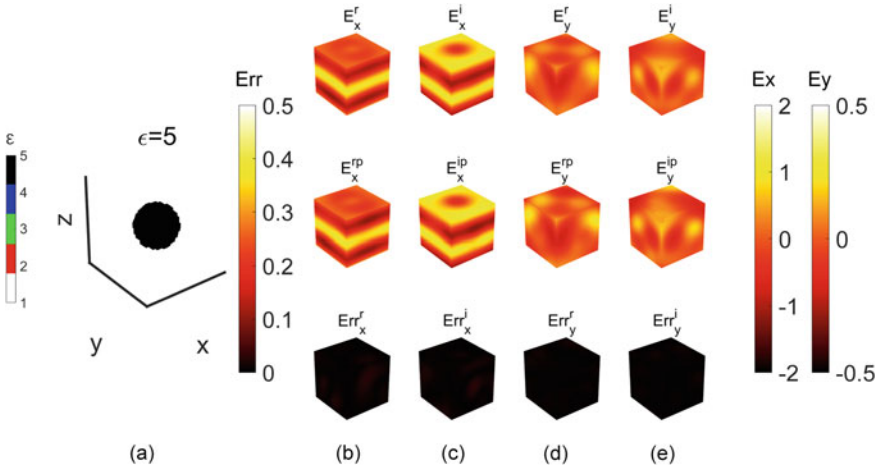


Fig. 5.23 An example in Experiment Group #8. **a** The geometry and the relative permittivity of the scatterer; **b** the ground truth, the prediction and the absolute error of $\Re e(E_x)$; **c** the ground truth, the prediction and the absolute error of $\mathcal{I}m(E_x)$; **d** the ground truth, the prediction and the absolute error of $\Re e\big(E_y\big)$ and **e** the ground truth, the prediction and the absolute error of $\mathcal{I}m\big(E_y\big)$

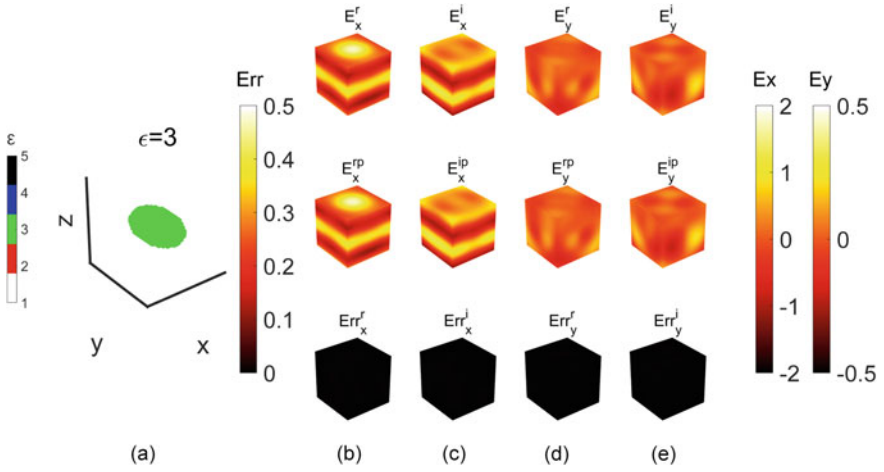


Fig. 5.24 Another example in Experiment Group #8. **a** The geometry and the relative permittivity of the scatterer; **b** the ground truth, the prediction and the absolute error of $\Re e(E_x)$; **c** the ground truth, the prediction and the absolute error of $\mathcal{I}m(E_x)$; **d** the ground truth, the prediction and the absolute error of $\Re e\big(E_y\big)$ and **e** the ground truth, the prediction and the absolute error of $\mathcal{I}m\big(E_y\big)$

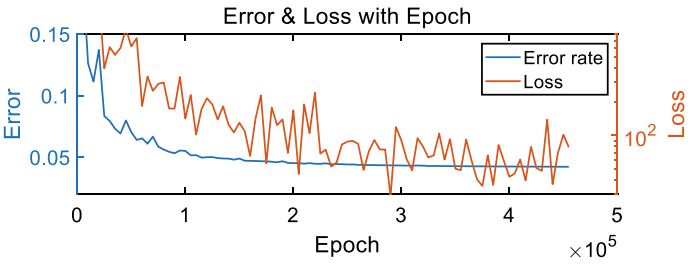


Fig. 5.25 The error rate and the loss with the training epochs on Experiment Group #9

the curves of the error rate and the loss during the training process are presented in Fig. 5.25.

To reach convergence, it takes around 45,500 epochs and 4.5 h for training. The final error rate reduces to 4.2% and the loss is lowered to 78. In order to intuitively exhibit the prediction capability of the network, two examples randomly selected in the test sets are shown in Figs. 5.26 and 5.27. Here, (a) demonstrates the geometry and the relative permittivity of the scatterer, (b) and (c) exhibit the ground truth, the prediction and the absolute error of the real/imaginary part of E_x while (c) and (d) reflect those of E_y.

Although the relative average rate in Experiment Group #9 is higher than the previous two experiment Groups, it is still acceptable considering the more complex

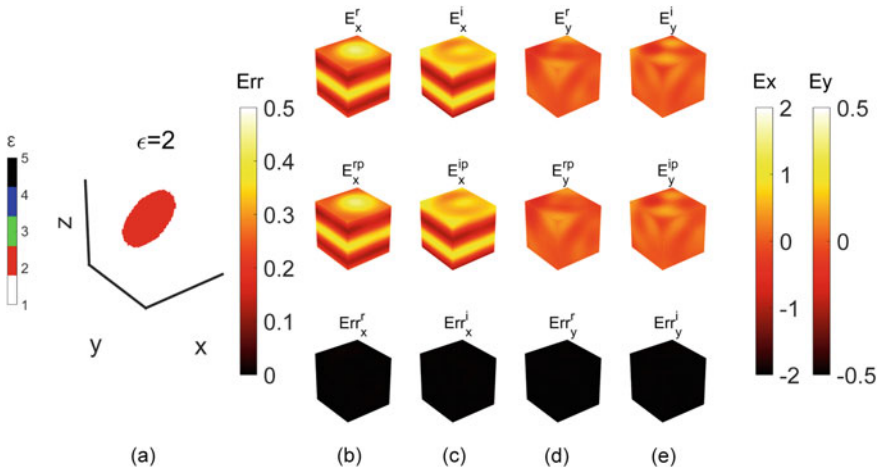


Fig. 5.26 An example in Experiment Group #9. **a** The geometry and the relative permittivity of the scatterer; **b** the ground truth, the prediction and the absolute error of $\mathfrak{Re}(E_x)$; **c** the ground truth, the prediction and the absolute error of $\mathcal{I}m(E_x)$; **d** the ground truth, the prediction and the absolute error of $\mathfrak{Re}\left(E_y\right)$ and **e** the ground truth, the prediction and the absolute error of $\mathcal{I}m\left(E_y\right)$

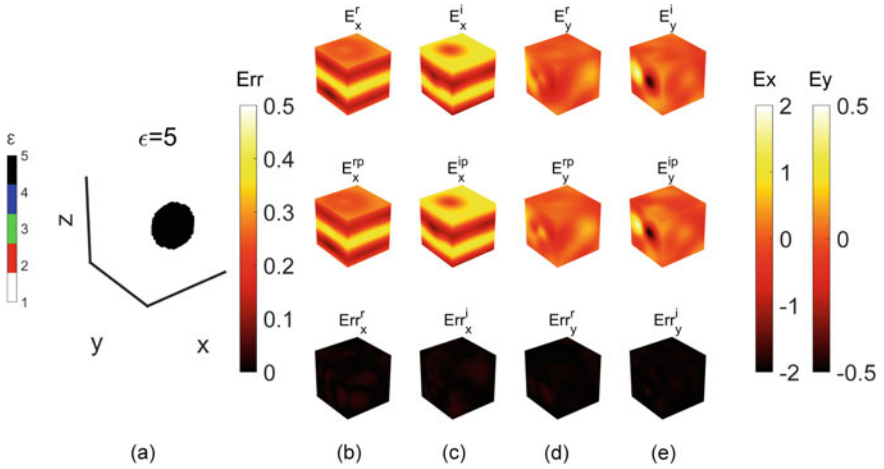


Fig. 5.27 Another example in Experiment Group #9. **a** The geometry and the relative permittivity of the scatterer; **b** the ground truth, the prediction and the absolute error of $\Re e(E_x)$; **c** the ground truth, the prediction and the absolute error of $\mathcal{I}m(E_x)$; **d** the ground truth, the prediction and the absolute error of $\Re e\big(E_y\big)$; **e** the ground truth, the prediction and the absolute error of $\mathcal{I}m\big(E_y\big)$

scatterers. In summary, the three experiment groups expound that the EM-net framework has excellent performance on predicting the field even in the 3D cases, further emerging its mighty computational ability.

5.4.4 Acceleration

Compared to the traditional numerical algorithms, a significant preponderance of the proposed EM-net is the predicting speed. In fact, given the input scatterer, the well-trained framework can give the output field quantities almost simultaneously without losing precision. Here, to quantitatively measure the acceleration ability, the calculation time spent by the traditional algorithm (FDFD) and by the trained neural network are both recorded. In order to ensure the accuracy of the experiment, the two groups of measurements are carried out on the same computing platform with the same data set.

Figure 5.28 demonstrates the time consumption by EM-net and the 3D FDFD program for calculating the scattering field. Both the two methods emerge a linear relationship between the time and the number of samples. To solve 100 samples, the 3D FDFD algorithm takes around 8977 s while the EM-net only takes 2.11 s. Therefore, it can be concluded that the EM-net can accelerate the solving process by more than 4000 times without sacrificing the accuracy.

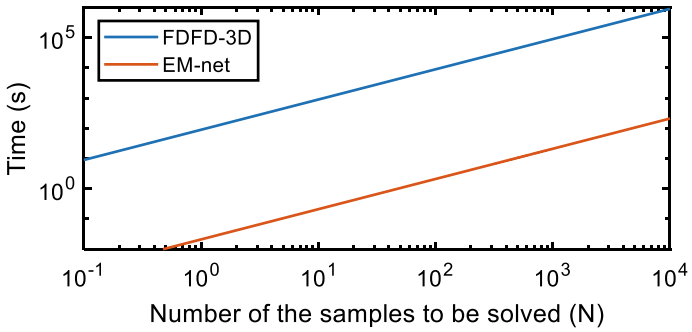


Fig. 5.28 Time consumption of 3D scattering problems with 3D FDFD and EM-net to calculate N samples

5.5 Summary

In this chapter, the detailed process to structure the 3D electromagnetic scattering solver is discussed. The 3D FDFD algorithm is first utilized to prepare the database to be trained later. Then, several pre-experiments are conducted to determine the most appropriate experimental conditions. The result exhibits that the proposed framework is capable to accelerate the solving process more than 4000 times without sacrificing the accuracy, which will vastly raise the solving efficiency. We foresee that the booming of the deep learning technique will bear many applications in computational electromagnetics.

References

1. Balanis CA (2012) Advanced engineering electromagnetics, 2nd edn. Wiley, Hoboken, N.J.
2. Jackson JD (1962) Classical electrodynamics. Wiley, New York
3. Lay DC (2006) Linear algebra and it's applications, 3rd edn. Pearson/Addison-Wesley, Boston
4. Cai G, Chen BM, Lee TH (2011) Coordinate systems and transformations. In: Cai G, Chen BM, Lee TH (eds) Unmanned rotorcraft systems. Springer London, London, pp 23–34. http://doi.org/10.1007/978-0-85729-635-1_2
5. Shin W, Fan SH (2012) Choice of the perfectly matched layer boundary condition for frequency-domain Maxwell's equations solvers. J Comput Phys 231(8):3406–3431. https://doi.org/10.1016/j.jcp.2012.01.013
6. Shin W, Fan SH (2012) Choice of the perfectly matched layer boundary condition for iterative solvers of the frequency-domain Maxwell's equations. Proc Spie 8255. Artn 82550n. http://doi.org/10.1117/12.906869
7. Shin W (2015) MaxwellFDFD Webpage. https://github.com/wsshin/maxwellfdfd
8. Murphy KP (2012) Machine learning: a probabilistic perspective. Adaptive computation and machine learning series. MIT Press, Cambridge, MA
9. Chollet F (2017) Deep learning with python. Manning Publications, New York
10. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press, Cambridge, Massachusetts

11. Smith LN (2017) Cyclical learning rates for training neural networks. Ieee Wint Conf Appl 464–472. http://doi.org/10.1109/Wacv.2017.58
12. Bengio Y (2012) Practical recommendations for gradient-based training of deep architectures. arXiv:1206.5533
13. Ruder S (2016) An overview of gradient descent optimization algorithms. arXiv:1609.04747
14. Plagianakos VP, Magoulas GD, Vrahatis MN (2001) Learning rate adaptation in stochastic gradient descent. In: Hadjisavvas N, Pardalos PM (eds) Advances in convex analysis and global optimization: honoring the memory of C. Caratheodory (1873–1950). Springer US, Boston, MA, pp 433–444. http://doi.org/10.1007/978-1-4613-0279-7_27
15. You K, Long M, Wang J, Jordan MI (2019) How does learning rate decay help modern neural networks? arXiv:1908.01878
16. Zhang T, Li W (2020) k-decay: a new method for learning rate schedule. arXiv:2004.05909
17. Smith SL, Kindermans P-J, Ying C, Le QV (2017) Don't decay the learning rate, increase the batch size. arXiv:1711.00489
18. Goceri E, Gooya A (2018) On the importance of batch size for deep learning. Paper presented at the an Istanbul meeting for world mathematicians, minisymposium on approximation theory and minisymposium on math education, Istanbul, Turkey
19. Pawluszek-Filipiak K, Borkowski A (2020) On the importance of train-test split ratio of datasets in automatic landslide detection by supervised classification. Remote Sens 12(18):3054
20. Ramachandran P, Zoph B, Le QV (2017) Searching for activation functions. arXiv:1710.05941
21. Mhasker HN, Micchelli CA (1993) How to choose an activation function. Paper presented at the Proceedings of the 6th international conference on neural information processing systems
22. Nwankpa C, Ijomah W, Gachagan A, Marshall S (2018) Activation functions: comparison of trends in practice and research for deep learning. arXiv:1811.03378
23. Pedamonti D (2018) Comparison of non-linear activation functions for deep neural networks on MNIST classification task. arXiv:1804.02763
24. Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. Neural Comput 18(7):1527–1554. https://doi.org/10.1162/neco.2006.18.7.1527
25. Hahnloser RLT (1998) On the piecewise analysis of networks of linear threshold neurons. Neural Netw 11(4):691–697. https://doi.org/10.1016/S0893-6080(98)00012-4
26. Hahnloser RHR, Sarpeshkar R, Mahowald MA, Douglas RJ, Seung HS (2000) Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. Nature 405(6789):947–951. https://doi.org/10.1038/35016072
27. Krizhevsky A, Sutskever I, Hinton G (2012) ImageNet classification with deep convolutional neural networks. Paper presented at the proceedings of the 25th international conference on neural information processing systems
28. He K, Zhang X, Ren S, Sun J (2015) Deep residual learning for image recognition. arXiv:1512.03385
29. Nair V, Hinton GE (2010) Rectified linear units improve restricted Boltzmann machines. In: ICML 2010
30. Jarrett K, Kavukcuoglu K, Ranzato M, LeCun Y (2009) What is the best multi-stage architecture for object recognition? In: 2009 IEEE 12th international conference on computer vision, 29 Sept–2 Oct 2009, pp 2146–2153. http://doi.org/10.1109/ICCV.2009.5459469
31. Qi ST, Wang Y, Li YZ, Wu X, Ren Q, Ren Y (2020) Two-dimensional electromagnetic solver based on deep learning technique. IEEE J Multiscale Multiphys Comput Tech 5:83–88
32. Li YZ, Wang YP, Qi ST, Ren Q, Kang L, Campbell SD, Werner PL, Werner DH (2020) Predicting scattering from complex nano-structures via deep learning. IEEE Access 8:139983–139993
33. He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: surpassing human-level performance on ImageNet classification. In: 2015 IEEE international conference on computer vision (ICCV), 7–13 Dec 2015, pp 1026–1034. http://doi.org/10.1109/ICCV.2015.123
34. Zhang X, Zou Y, Shi W (2017) Dilated convolution neural network with LeakyReLU for environmental sound classification. In: 2017 22nd international conference on digital signal processing (DSP), pp 1–5

35. Clevert D-A, Unterthiner T, Hochreiter S (2016) Fast and accurate deep network learning by exponential linear units (ELUs)
36. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M (2016) Tensorflow: a system for large-scale machine learning. Paper presented at the 12th symposium on operating systems design and implementation, Savannah, GA
37. Géron Al (2019) Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: concepts, tools, and techniques to build intelligent systems, 2nd edn. O'Reilly Media, Inc., Sebastopol, CA
38. Atienza R (2020) Advanced deep learning with TensorFlow 2 and Keras: Apply DL, GANs, VAEs, deep RL, unsupervised learning, object detection and segmentation, and more. Packt Publishing Ltd., Birmingham
39. Nvidia (2019) Performance. https://www.nvidia.com/en-us/geforce/20-series/

Index

Q. Ren et al., *Sophisticated Electromagnetic Forward Scattering Solver via Deep Learning*, https://doi.org/10.1007/978-981-16-6261-4