CHAPTER 14

Analysis of periodic structures

Many electromagnetic applications require the use of periodic structures such as frequency selective surfaces (FSS), electromagnetic band gap structures, corrugated surfaces, phased antenna arrays, periodic absorbers or negative index materials. These structures extend to several wavelengths in size therefore their analyses are time-consuming and memory-extensive using the conventional FDTD method. To overcome the limitation of the conventional FDTD method, first, a periodic structure is considered as infinitely periodic. For instance, Figure 14.1 illustrates a unit cell of a two-dimensional periodic structure on a substrate that extends to infinity in the x and y directions. Then the assumption of infinite periodicity is utilized to develop FDTD algorithms that analyze only one unit cell of the periodicity instead of the entire structure and obtain results for the entire infinite size structure or structures composed of large number of unit cells with approximations. These algorithms mainly deal with the boundaries of a unit cell, therefore they are referred to as periodic boundary conditions (PBC).

14.1 Periodic boundary conditions

The PBC algorithms are divided into two main categories: field transformation methods and direct field methods [64]. The field transformation methods introduce auxiliary fields to eliminate the need for time-advanced data. The transformed field equations are then discretized and solved using FDTD techniques. The split-field method [65] and multispatial grid method [66] are two approaches in this category. The direct field category methods work directly with Maxwell's equations, and hence there is no need for any field transformation. The sine-cosine method [67] is an example of this category. In this method the structure is excited simultaneously with sine and cosine waveforms, therefore it is a single frequency method that does not maintain the wide-band capability of FDTD.

A simple and efficient PBC algorithm that belongs to the direct field category, referred to as constant horizontal wavenumber approach, is introduced in [68–70] and yet has a wideband capability. In this chapter, we will demonstrate the constant horizontal wavenumber approach and its MATLAB® implementation to calculate reflection and transmission coefficients of an infinite periodic structure.

It should be noted that in most applications that use periodic structures the reflection and/or transmission properties are of interest due to incident plane waves with a number of incident angles and a number of frequencies. One would seek to obtain the reflection and/or

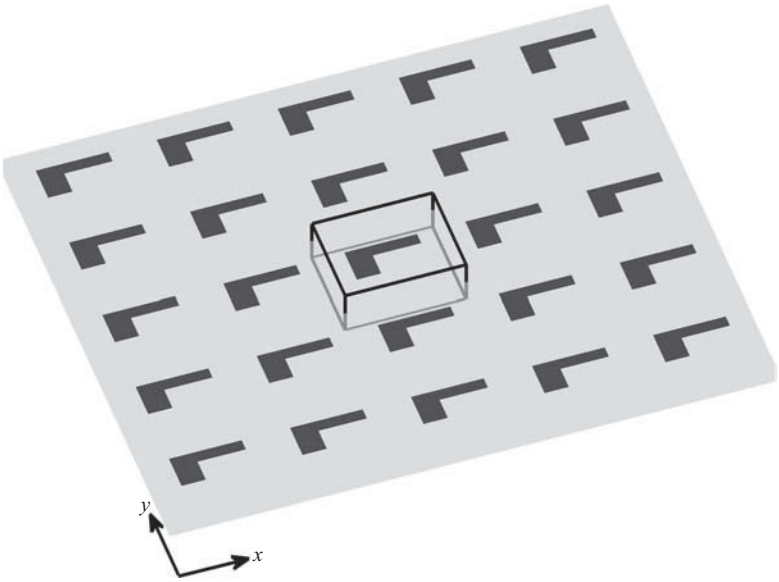

Figure 14.1 A periodic structure and a unit cell of two-dimension periodicity.

transmission coefficient distribution as a two-dimensional figure on which the axes are the frequency and angle of incidence. For instance, Figure 14.2 shows the magnitude of reflection coefficient of a dielectric slab with a thickness of 1 cm and relative permittivity of 4 due to obliquely incident TE plane waves. In this example the figure displays data up to 20 GHz for incident angles range from 0 to 90°. Here the figure is constructed using $N_f \times N_a$ data points where $N_f = 200$ is the number of frequencies and $N_a = 90$ is the number of incident angles for which the reflection coefficients are calculated.

The sine-cosine method can be used to perform calculations at a single frequency and a single angle, therefore, to obtain an angle-frequency plane figure such as the one in Figure 14.2 it is needed to run the PBC FDTD program $N_f \times N_a$ times, which makes it inefficient. For instance, Figure 14.2 shows a data point calculated by the sine-cosine method for 12 GHz and 60° using a single run.

On the other hand, for instance, the split-field method preserves the wideband capability and one can obtain wideband results for a wide frequency band in a single program run for one incident angle. Therefore, to obtain an angle-frequency plane figure one needs to run the split-field PBC FDTD program N_a times. For instance, Figure 14.2 shows the line on which data points, calculated by a single run of split-field method when the angle of incidence is 30°.

The split-field method and multispatial grid method have the wideband capability; however, there are two main limitations with these methods. First, the transformed equations have additional terms that require special handling such as splitting the field or using a multigrid algorithm to implement the FDTD, which increases the complexity of the problem. Second, as the angle of incidence increases from normal incidence (0°) to grazing incidence (90°), the stability factor needs to be reduced, so the FDTD time step decreases significantly [64]. As a result, smaller time steps are needed for oblique incidence to generate stable results, which increases the computational time for such cases [71].

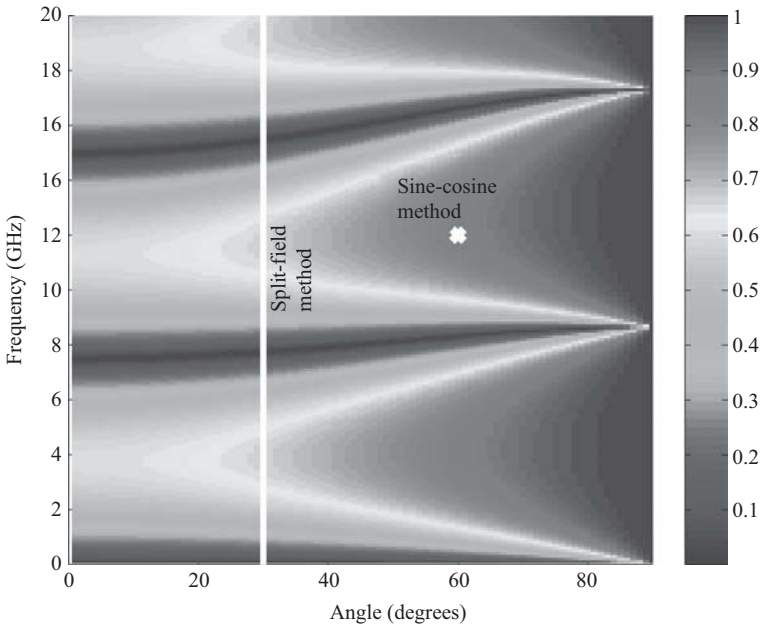

Figure 14.2 Magnitude of reflection coefficient in the angle-frequency plane.

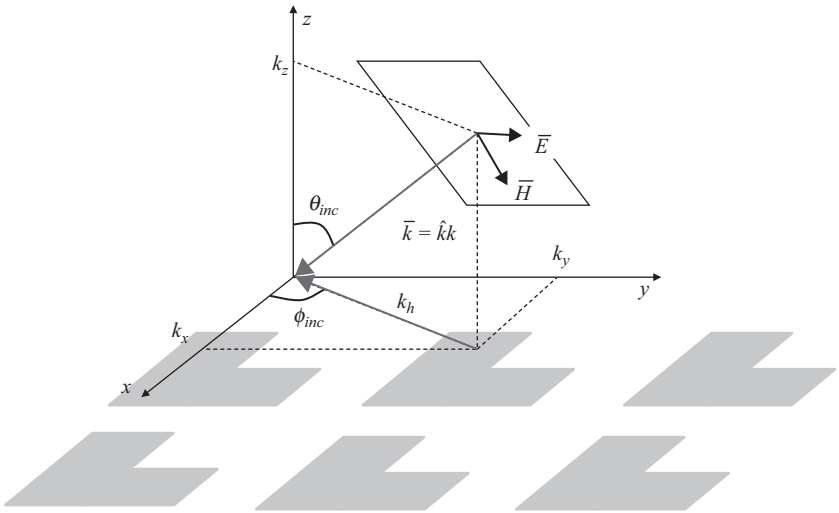

Figure 14.3 Propagation vector of an incident plane wave and its horizontal component.

While one way to display reflection and transmission coefficient distributions is to plot them on angle-frequency planes, an alternative way, which essentially shows the same data, is to plot them on horizontal wavenumber-frequency planes. Figure 14.3 illustrates the propagation vector, $\hat{k}$, of an incident plane wave. One can imagine an infinitely periodic structure in the x and y directions with z direction as the normal to the interface between the

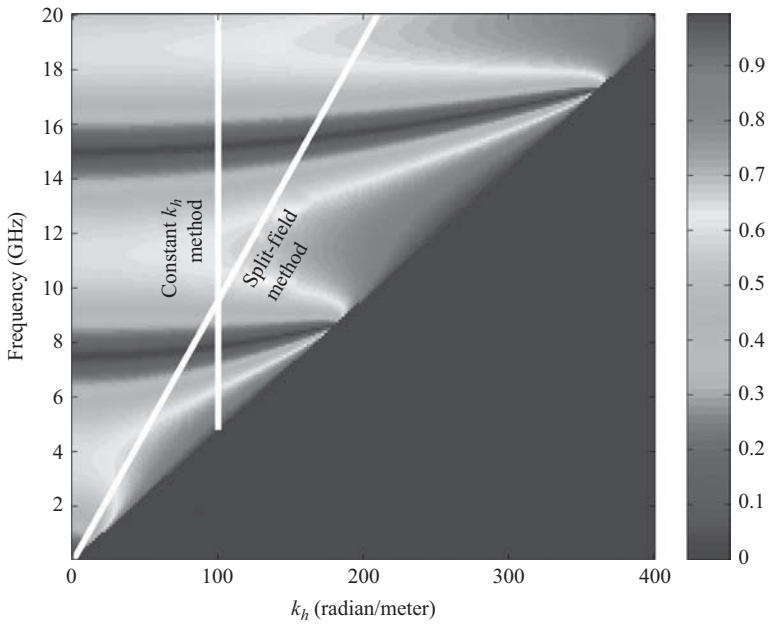

Figure 14.4 Magnitude of reflection coefficient in the horizontal wavenumber-frequency plane.

periodic structure and free space. Also shown in the figure is the horizontal component of the propagation vector, denoted with length k_h, such that $k_h = \sqrt{k_x^2 + k_y^2}$. The relation between the wavenumber, k, and its horizontal component, k_h, is fixed through the angle θ_{inc} such that

$$k_h = k \sin\theta_{inc} = \omega\sqrt{\mu_0\varepsilon_0}\sin\theta_{inc} = \frac{\omega}{c}\sin\theta_{inc}. \tag{14.1}$$

where ω is the angular frequency of the wave propagating in free space. Therefore, one can use this fixed relationship between θ_{inc} and k_h to display reflection and transmission coefficient distributions on horizontal wavenumber-frequency planes instead of the angle-frequency planes. For instance, the data in Figure 14.2 is displayed on a horizontal wavenumber-frequency plane in Figure 14.4. The set of data points calculated using the split-field method in Figure 14.2 is also projected to the data display in Figure 14.4.

It is possible to imagine many incident plane waves all with the same horizontal wavenumber. Figure 14.5 illustrates a number of propagation vectors all with the same horizontal wavenumber. Although the horizontal wavenumbers of these vectors are the same, but their wavenumbers, angles of incidence, as well as frequencies are different. In the constant horizontal wavenumber PBC algorithm an FDTD simulation is performed by setting a constant horizontal wavenumber instead of a specific angle of incidence. Therefore, it yields results supporting a number of angles each with a different frequency in a single PBC FDTD run. For instance, Figure 14.4 shows the data points on a vertical line that can be obtained with a single run of this method for the constant horizontal wavenumber of 100 radian/meter. To obtain the data distribution on the entire horizontal wavenumber-frequency plane, the simulation needs to be repeated for a number of constant horizontal wavenumbers, N_{chw}.

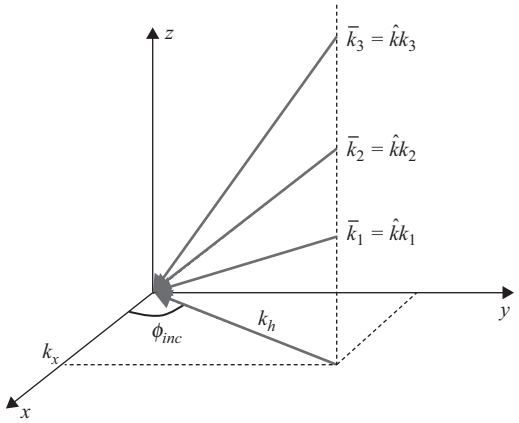

Figure 14.5 Propagation vectors with the same horizontal wavenumber.

The idea of using a constant wavenumber in FDTD was originated from guided wave analysis and eigenvalue problems in [72], and it was extended to the plane wave scattering problems in [73–75]. The approach offers many advantages, such as implementation simplicity, stability condition, and numerical errors similar to the conventional FDTD, computational efficiency near the grazing incident angles, and the wide-band capability [71]. The formulation of the constant horizontal wavenumber method and its MATLAB implementation are presented in the following sections.

14.2 Constant horizontal wavenumber method

Consider the example shown in Figure 14.6, where a unit cell of a periodic structure is shown. The periodic structure is a dielectric slab and a rectangular PEC patch is placed on top of the slab in this case. A time harmonic plane wave with angular frequency ω is incident on the periodic structure with oblique incidence. The direction of incoming wave can be denoted by θ_{inc} and ϕ_{inc}. The unit cell is terminated by CPML boundaries at the top and bottom sides. The other sides will be treated as periodic boundaries. Also shown in the figure below, the upper CPML region is a plane, referred to as the source plane, where incident field is injected into the computational space.

Figure 14.7 shows the field components on an xy plane-cut of the grid of this problem space. The size of the problem space is $P_x = N_x\Delta x$ in the x direction and $P_y = N_y\Delta y$ in the y direction. At steady state, a field component on the right boundary of the grid is a time-delayed equivalent of a field on the left boundary. For instance, one can write

$$E_y(x = P_x, y, z, t) = E_y\left(x = 0, y, z, t - \frac{P_x}{c}\sin\theta_{inc}\right), \tag{14.2}$$

and

$$H_z\left(x = P_x - \frac{\Delta x}{2}, y, z, t\right) = H_z\left(x = -\frac{\Delta x}{2}, y, z, t - \frac{P_x}{c}\sin\theta_{inc}\right). \tag{14.3}$$

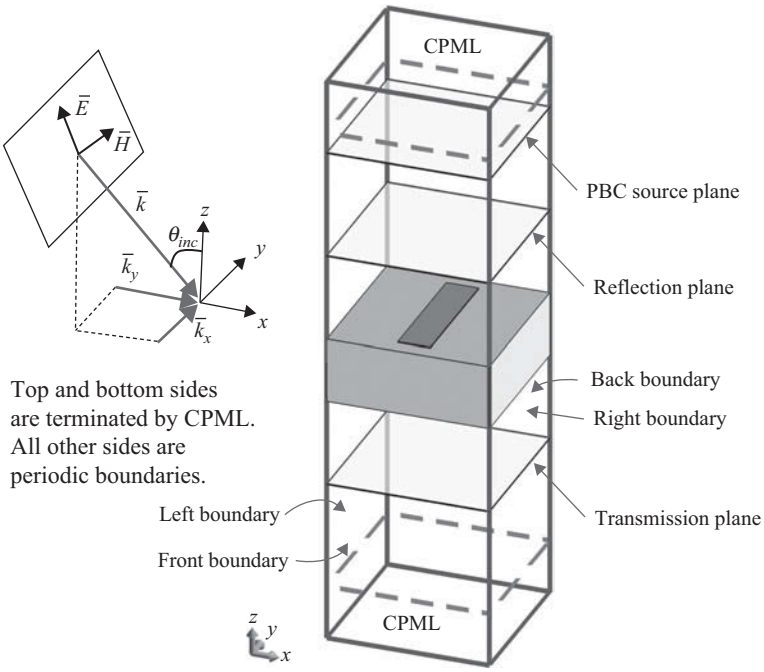

Figure 14.6 An incident plane wave and a unit cell.

Similarly, a field component on the back boundary of the grid is a time-delayed equivalent of a field on the front boundary such that

$$E_x(x, y = P_y, z, t) = E_x\left(x, y = 0, z, t - \frac{P_y}{c}\sin\theta_{inc}\right), \tag{14.4}$$

and

$$H_z\left(x, y = P_y - \frac{\Delta y}{2}, z, t\right) = H_z\left(x, y = -\frac{\Delta y}{2}, z, t - \frac{P_y}{c}\sin\theta_{inc}\right). \tag{14.5}$$

When transformed from time-domain to frequency-domain (14.2) can be written as

$$E_y(x = P_x, y, z, t) = E_y(x = 0, y, z, t)e^{-j\frac{\omega}{c}\sin\theta_{inc}P_x}, \tag{14.6}$$

which can be expressed using (14.1) as

$$E_y(x = P_x, y, z) = E_y(x = 0, y, z)e^{-jk\sin\theta_{inc}P_x} = E_y(x = 0, y, z)e^{-jk_xP_x}. \tag{14.7}$$

This is how the fields on a periodic structure can be expressed according to the Floquet theory, and it implies that any field component in the problem space of a periodic structure is a phase shifted equivalent of another field component that is distant by P_x in the x direction by a phase shift of k_xP_x or $-k_xP_x$. Similarly, any field component is a phase shifted equivalent of another field component that is distant by P_y in the y direction by a phase shift

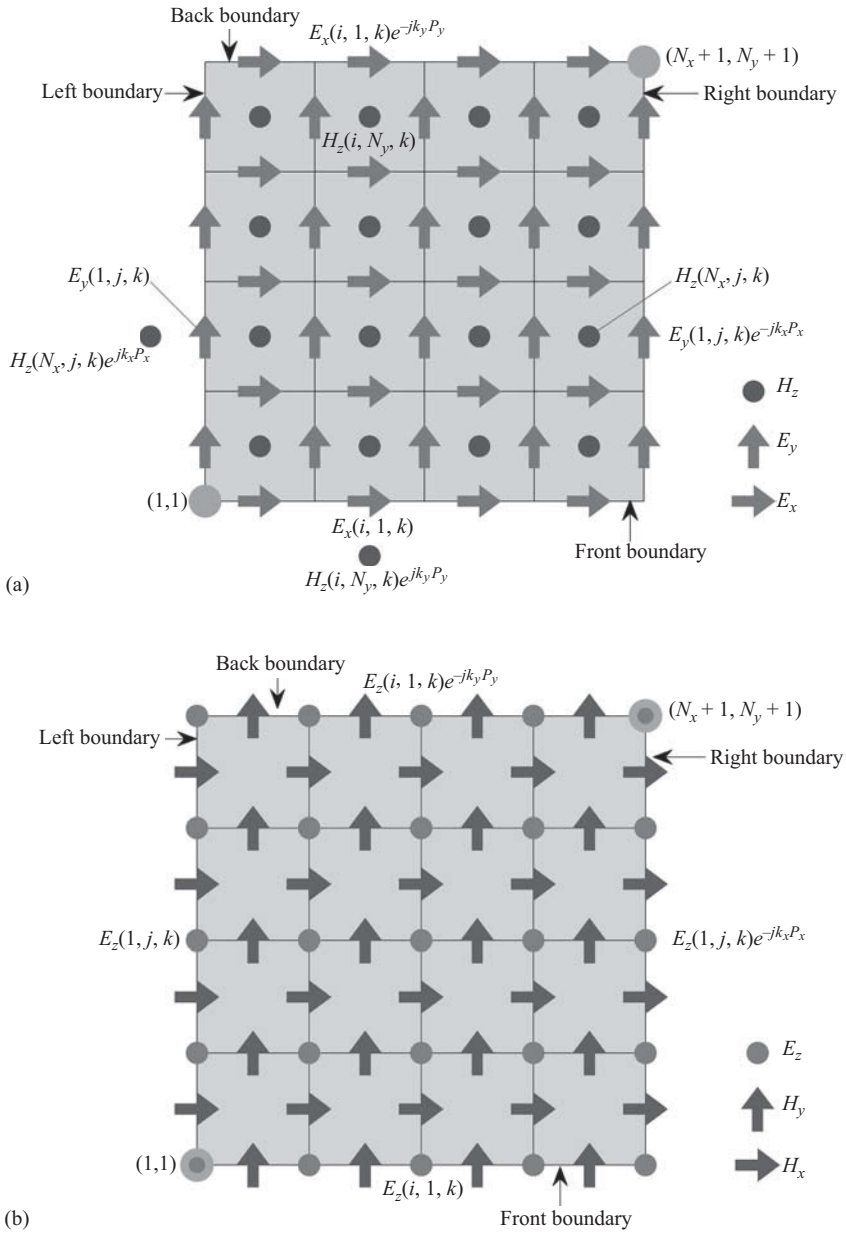

Figure 14.7 Field components on an *xy* plane-cut of a unit cell in a three-dimensional domain: (a) E_x, E_y, and H_z components and (b) H_x, H_y, and E_z components.

of k_yP_y or $-k_yP_y$. We can make use of this relationship between the field components and develop a PBC algorithm: while updating a field component on the boundary of a unit cell during an iteration of the FDTD time-marching loop, if we need to use the value of a neighboring field component for which the value is not readily available we can use a phase

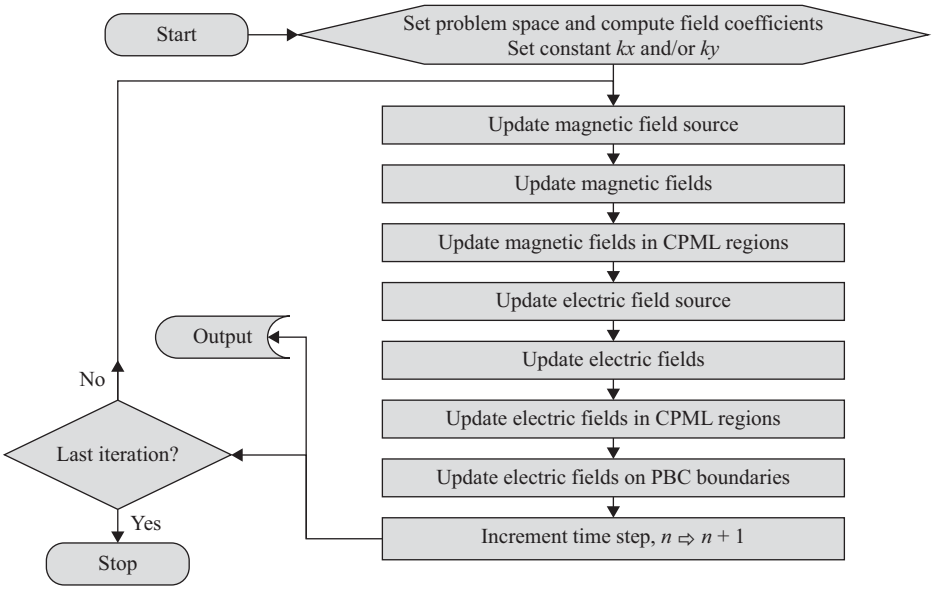

Figure 14.8 Flow chart of PBC FDTD algorithm.

shifted equivalent of another field component for which the value is known. This procedure will be elaborated in details in this section.

The general flow chart of PBC FDTD algorithm is illustrated in Figure 14.8. During an iteration of the FDTD time-marching loop, first, the magnetic field components on the source plane are updated to excite an incident wave, details of which are discussed in the next section. Then, all magnetic field components in the computational domain are updated for the new time step. Also, the components in the CPML regions are updated using the CPML equations. Afterwards, the electric field components on the source plane are updated, and all electric field components, except the ones lying on the PBC boundaries, are updated for the new time step followed by CPML updates. Then, the electric field components on the PBC boundaries are updated, with reference to Figure 14.7, as follows:

1. Update E_x components on the front and back boundaries.
2. Update E_y components on the left and right boundaries.
3. Update E_z components on all boundaries except for the corners.
4. Update E_z components at the corners.

Figure 14.7 shows the field components of a unit cell, the shaded regions, as well as the neighboring magnetic field components outside the unit cell. In order to calculate an electric field component on the front boundary, that is indexed as $E_x^{n+1}(i, 1, k)$, one needs the value of the magnetic field component below it, that could be indexed as $H_z^{n+\frac{1}{2}}(i, 0, k)$. While $H_z^{n+\frac{1}{2}}(i, 0, k)$ is not in the computational space of the unit cell in consideration and its value is not known, a phase shifted equivalent of it, $H_z^{n+\frac{1}{2}}\big(i, N_y, k\big)$, can be used instead following the Floquet theory as

$$H_z^{n+\frac{1}{2}}(i, 0, k) = H_z^{n+\frac{1}{2}}\big(i, N_y, k\big)e^{jk_yP_y}. \tag{14.8}$$

Then the electric field updating equation can be written for $E_x^{n+1}(i,\,1,\,k)$ as

$$\begin{aligned} E_x^{n+1}(i,\,1,\,k) &= C_{exe}(i,\,1,\,k) \times E_x^n(i,\,1,\,k) + C_{exhz}(i,\,1,\,k) \\ &\times \left[H_z^{n+\frac{1}{2}}(i,\,1,\,k) - H_z^{n+\frac{1}{2}}(i,\,N_y,\,k)e^{jk_yP_y}\right] + C_{exhy}(i,\,1,\,k) \\ &\times \left[H_y^{n+\frac{1}{2}}(i,\,1,\,k) - H_y^{n+\frac{1}{2}}(i,\,1,\,k-1)\right]. \end{aligned} \tag{14.9}$$

Similarly, to update an electric field component on the back boundary, $E_x^{n+1}\left(i,\,N_y+1,\,k\right)$, the value of $H_z^{n+\frac{1}{2}}\left(i,\,N_y+1,\,k\right)$ is needed, which is not available. One can notice that $E_x^{n+1}\left(i,\,N_y+1,\,k\right)$ is a phase shifted equivalent of $E_x^{n+1}(i,\,1,\,k)$, which has already been calculated using (14.9). Therefore, as a shortcut, $E_x^{n+1}\left(i,\,N_y+1,\,k\right)$ can be updated as

$$E_x^{n+1}\left(i,N_y+1,k\right) = E_x^{n+1}(i,1,k)e^{-jk_yP_y}. \tag{14.10}$$

The E_y components on the left boundary can be updated using the periodicity in the x direction as

$$\begin{aligned} E_y^{n+1}(1,j,k) &= C_{eye}(1,j,k) \times E_y^n(1,j,k) \\ &+ C_{eyhx}(1,j,k) \times \left[H_x^{n+\frac{1}{2}}(1,j,k) - H_x^{n+\frac{1}{2}}(1,j,k-1)\right] \\ &+ C_{eyhz}(1,j,k) \times \left[H_z^{n+\frac{1}{2}}(1,j,k) - H_z^{n+\frac{1}{2}}(N_x,j,k) \times e^{jk_xP_x}\right]. \end{aligned} \tag{14.11}$$

Then, $E_y^{n+1}(1,j,k)$ can be used to calculate the electric field component on the right boundary as

$$E_y^{n+1}(N_x+1,j,k) = E_y^{n+1}(1,j,k)e^{-jk_yP_y}. \tag{14.12}$$

The same procedure can be repeated for the E_z components as well. The field components on the left boundary, except the corners, can be updated as

$$\begin{aligned} E_z^{n+1}(1,j,k) &= C_{eze}(1,j,k) \times E_z^n(1,j,k) \\ &+ C_{ezhy}(1,j,k) \times \left[H_y^{n+\frac{1}{2}}(1,j,k) - H_y^{n+\frac{1}{2}}(N_x,j,k) \times e^{jk_xP_x}\right] \\ &+ C_{ezhx}(1,j,k) \times \left[H_x^{n+\frac{1}{2}}(1,j,k) - H_x^{n+\frac{1}{2}}(1,j-1,k)\right], \end{aligned} \tag{14.13}$$

while the components on the right boundary, except the corners, can be updated as

$$E_z^{n+1}(N_x+1,j,k) = E_z^{n+1}(1,j,k) \times e^{-jk_xP_x}. \tag{14.14}$$

For the components on the front boundary, except the corners, one can write

$$\begin{aligned} E_z^{n+1}(i,\,1,\,k) &= C_{eze}(i,\,1,\,k) \times E_z^n(i,\,1,\,k) \\ &+ C_{ezhy}(i,\,1,\,k) \times \left[H_y^{n+\frac{1}{2}}(i,\,1,\,k) - H_y^{n+\frac{1}{2}}(i-1,\,1,\,k)\right] \\ &+ C_{ezhx}(i,\,1,\,k) \times \left[H_x^{n+\frac{1}{2}}(i,\,1,\,k) - H_x^{n+\frac{1}{2}}(i,\,N_y,\,k) \times e^{jk_yP_y}\right] \end{aligned} \tag{14.15}$$

while for the components on the back boundary, except the corners,

$$E_z^{n+1}(i,\, N_y+1,\, k) = E_z^{n+1}(i,\, 1,\, k) \times e^{-jk_yP_y}. \tag{14.16}$$

The component of E_z at the front left corner, $E_z^{n+1}(1,1,k)$, needs the values of $H_x^{n+\frac{1}{2}}(1,0,k)$ and $H_y^{n+\frac{1}{2}}(0,1,k)$ that are unknown. Therefore, instead, following the Floquet theory, one can write

$$\begin{aligned} E_z^{n+1}(1,1,\,k) &= C_{eze}(1,1,\,k) \times E_z^{n}(1,1,\,k) \\ &+ C_{ezhy}(1,1,\,k) \times \left[H_y^{n+\frac{1}{2}}(1,1,\,k) - H_y^{n+\frac{1}{2}}(N_x,1,\,k) \times e^{jk_xP_x}\right] \\ &+ C_{ezhx}(1,1,\,k) \times \left[H_x^{n+\frac{1}{2}}(1,1,\,k) - H_x^{n+\frac{1}{2}}(1,N_y,\,k) \times e^{jk_yP_y}\right]. \end{aligned} \tag{14.17}$$

Once the component of E_z at the front left corner is known, the components at the other corners can be calculated as

$$E_z^{n+1}(N_x+1,1,\,k) = E_z^{n+1}(1,1,\,k) \times e^{-jk_xP_x}, \tag{14.18}$$

$$E_z^{n+1}(1,\, N_y+1,\, k) = E_z^{n+1}(1,1,\,k) \times e^{-jk_yP_y}, \tag{14.19}$$

$$E_z^{n+1}(N_x+1,\, N_y+1,\, k) = E_z^{n+1}(1,1,\,k) \times e^{-jk_yP_y} \times e^{-jk_xP_x}. \tag{14.20}$$

The above PBC update equations are derived for obliquely incident plane waves. Thus, in the case of normal incidence, the horizontal wavenumbers are $k_x = k_y = 0$ radian/meter, therefore all exponential terms in the field component update equations become equal to one.

14.3 Source excitation

In Section 14.1 we discussed that many incident waves with different frequencies can have the same horizontal wavenumber, while each of these waves will have a different angle of incidence. Therefore, one can imagine that if a wave with a time waveform which has a wide frequency band is incident on a unit cell, all these frequency components of the wave can have the same horizontal wavenumber, i.e., k_x and k_y, while each component will have a different angle of incidence that satisfies (14.1). If we excite a computational domain with such a waveform and apply the presented PBC conditions, we can obtain the results for the frequencies in the spectrum of the incident wave at their associated angles of incidence in one FDTD simulation. The results of such a simulation will lie on a vertical line on the horizontal wavenumber-frequency plane as shown in Figure 14.4. In Figure 14.4, a vertical data line starts from a minimum frequency: At grazing angle, where $\theta = 90°$, (14.1) becomes

$$k_h = k = \frac{\omega}{c}, \tag{14.21}$$

which implies that ω is at its minimum value, $\omega_{\min} = ck_h$, therefore the minimum frequency that can be calculated is

$$f_{\min} = \frac{ck_h}{2\pi}. \tag{14.22}$$

Consequently, a wideband waveform that is constructed to excite the FDTD computational space should have a minimum frequency of $f_{\min}$ in its spectrum. This also avoids the problem of horizontal resonance in which the fields do not decay to zero over time. Therefore, for instance, a cosine-modulated Gaussian waveform with a proper bandwidth, BW, can be considered as a source waveform, which has a modulation frequency of

$$f_c = \frac{k_h c}{2\pi} + \frac{BW}{2}. \tag{14.23}$$

Some considerations that need to be taken into account while determining the bandwidth and the modulation frequency are discussed in Section 14.5.2.

The incident waveform can be injected into a computational domain at the location of a source plane, shown in Figure 14.6. An easy way for this implementation is to calculate the tangential field components of the incident wave on the source plane and add them to the field components on the source plane at every time step. As a result the source plane will radiate the incident field in the lower direction toward the periodic structure. The source plane will radiate a mirror image of the incident field in the upper direction as well, however, since the upper boundary is terminated by CPML, the mirror image field will be absorbed by CPML and it will not interfere with the fields below the source plane.

This incident waveform can be constructed to simulate TE, TM, or TEM mode. It is straightforward to excite the TEM mode: the tangential electric field components on the source plane, i.e., $E_{inc,x}$ and $E_{inc,y}$, are uniform over the source plane, therefore all components of incident $E_{inc,x}$ will be the same as well as all components of incident $E_{inc,y}$ will have the same value, which will be added to the respective field components on the source plane.

For the TE mode with the oblique incidence, and the incident field satisfies the periodicity condition such that

$$E_{inc,x}\big(i,\, N_y + 1,\, ks\big) = E_{inc,x}(i,\, 1,\, ks) \times e^{-jk_y P_y}, \tag{14.24a}$$

$$E_{inc,y}(N_x + 1,\, j,\, ks) = E_{inc,x}(1,\, j,\, ks) \times e^{-jk_x P_x}, \tag{14.24b}$$

where ks is the k index of the source plane. To excite the incident field, we can choose a point, for instance, node (1,1) at the lower left corner shown in Figure 14.7, as a reference and apply the phase shift to all $E_{inc,x}$ and $E_{inc,y}$ components on the source plane with reference to this point. For instance, if the value of x component of the incident field is E_{0x}^n at time step n then,

$$E_{inc,x}^n(i, j,\, ks) = E_{0x}^n \times e^{-jk_x(i-0.5)\Delta x} \times e^{-jk_y(j-1)\Delta y}. \tag{14.25}$$

Similarly, if the value of y component of the incident field is E_{0y}^n at time step n then,

$$E_{inc,y}^n(i, j,\, ks) = E_{0y}^n \times e^{-jk_x(i-1)\Delta x} \times e^{-jk_y(j-0.5)\Delta y}. \tag{14.26}$$

For the TM mode, we can add incident magnetic field component to the respective tangential magnetic field components on the source plane. In this case, the source plane can be chosen to be on the magnetic field grid. To excite the incident field, again choose the point at node (1,1) as a reference and apply the phase shift to all $H_{inc,x}$ and $H_{inc,y}$ components on the

source plane with reference to this point. For instance, if the value of x component of the incident field is $H_{0x}^{n+\frac{1}{2}}$ at time step $n+\frac{1}{2}$ then,

$$H_{inc,x}^{n+\frac{1}{2}}(i,j,ks) = H_{0x}^{n+\frac{1}{2}} \times e^{-jk_x(i-1)\Delta x} \times e^{-jk_y(j-0.5)\Delta y}. \tag{14.27}$$

Similarly, if the value of y component of the incident field is $H_{0y}^{n+\frac{1}{2}}$ at time step $n+\frac{1}{2}$ then,

$$H_{inc,y}^{n+\frac{1}{2}}(i,j,ks) = H_{0y}^{n+\frac{1}{2}} \times e^{-jk_x(i-0.5)\Delta x} \times e^{-jk_y(j-1)\Delta y}. \tag{14.28}$$

It should be noted that the presented constant horizontal wavenumber method integrates complex exponential phase terms into a time domain simulation, therefore, unlike a conventional FDTD algorithm, the fields values are all complex rather than real. Hence, while the simulation results are not meaningful in time domain, the results are meaningful on a wide frequency-band in frequency domain.

14.4 Reflection and transmission coefficients

The reflection and/or transmission properties of periodic structures due to obliquely incident plane waves are sought in many applications. Figure 14.6 shows two planes, referred to as reflection and transmission planes, on which the fields can be captured and they can be used to calculate reflection and transmission coefficients. In this section we will discuss the equations to be used for the calculation of these coefficients.

It is possible to calculate co-polarized reflection, cross-polarized reflection, co-polarized transmission, as well as the cross-polarized transmission coefficients. Equations for these coefficients depend on the mode of excitation, i.e., TE, TM, or TEM modes.

Reflection coefficient of a periodic structure is usually presented in the frequency domain. The value of captured field components at a single point is sufficient to calculate the reflection coefficient, while the field components can be captured anywhere on the reflection plane illustrated in Figure 14.6. In the MATLAB implementation of the PBC FDTD, that will be discussed in the subsequent sections, an average of field components on the reflection plane is captured and used for reflection coefficient calculations. It should be noted that the field components on the reflection plane as well are subject to the phase shift discussed above. Before averaging, this phase shifts need to be adjusted so that all field components are at the same phase. Similar to the source excitation, we can choose the point at node (1,1) as a reference and update the phase shifts with reference to this point. The field components used for averaging after phase adjustments are represented as

$$E_{rp,x}^{n}(i,j,kr) = E_{x}^{n}(i,j,kr) \times e^{jk_x(i-0.5)\Delta x} \times e^{jk_y(j-1)\Delta y}, \tag{14.29a}$$

$$E_{rp,y}^{n}(i,j,kr) = E_{y}^{n}(i,j,kr) \times e^{jk_x(i-1)\Delta x} \times e^{jk_y(j-0.5)\Delta y}, \tag{14.29b}$$

$$E_{rp,z}^{n}(i,j,kr) = 0.5\left(E_{z}^{n}(i,j,kr-1) + E_{z}^{n}(i,j,kr)\right) \times e^{jk_x(i-1)\Delta x} \times e^{jk_y(j-1)\Delta y}, \tag{14.29c}$$

$$H_{rp,x}^{n+\frac{1}{2}}(i,j,kr) = 0.5\left(H_{x}^{n+\frac{1}{2}}(i,j,kr) + H_{x}^{n+\frac{1}{2}}(i,j,kr-1)\right) \times e^{jk_x(i-1)\Delta x} \times e^{jk_y(j-0.5)\Delta y}, \tag{14.29d}$$

$$H_{rp,y}^{n+\frac{1}{2}}(i,j,kr) = 0.5\left(H_y^{n+\frac{1}{2}}(i,j,kr) + H_y^{n+\frac{1}{2}}(i,j,kr-1)\right) \times e^{jk_x(i-0.5)\Delta x} \times e^{jk_y(j-1)\Delta y}, \quad (14.29e)$$

$$H_{rp,z}^{n+\frac{1}{2}}(i,j,kr) = H_z^{n+\frac{1}{2}}(i,j,kr) \times e^{jk_x(i-0.5)\Delta x} \times e^{jk_y(j-0.5)\Delta y}, \quad (14.29f)$$

where the subscript "*rp*" denotes the captured fields on the reflection plane. It should be noted that the reflection plane is considered to be an *xy* plane-cut of a unit cell that includes E_x, E_y, and H_z components as illustrated in Figure 14.7(a) for the above equations, where the plane-cut is indicated with the index *kr*. Therefore, E_x, E_y, and H_z components with *kr* index are used in (14.29a), (14.29b), and (14.29f), respectively, to calculate the respective phase adjusted fields. The components of E_z, H_x, and H_y do not lie on the same plane cut as the reflection plane. To obtain equivalent values for these field components on the reflection plane, the components above (with index *kr*) and below (with index $kr - 1$) the reflection plane are averaged and used in (14.29c), (14.29d), and (14.29e), respectively.

Similarly, the fields on the transmission plane can be captured and phase adjusted as well. After phase reversals, the field components are averaged and stored versus time in terms of the time step. After the FDTD time-marching loop is completed, these transient fields are transformed to frequency domain by discrete Fourier transform (DFT) and then the reflection and transmission coefficient calculations are performed.

One should notice that the fields that are captured on the reflection plane are total fields, such that

$$E_{rp} = E_{inc} + E_{ref}, \quad (14.30)$$

while the incident and reflected fields are needed for reflection coefficient calculations. Moreover, in particular, the co- and cross-polarized components of incident and reflected fields are needed for co- and cross-polarized reflection coefficient calculations.

14.4.1 TE mode reflection and transmission coefficients

Figure 14.9 illustrates the incident field in the TE mode. For the TE mode, the co-polarized total fields can be determined using the *x* and *y* components of the captured fields as

$$E_{rp,co} = E_{rp,y}\frac{k_x}{k_h} - E_{rp,x}\frac{k_y}{k_h}, \quad (14.31a)$$

$$H_{rp,co} = H_{rp,x}\frac{k_x}{k_h} + H_{rp,y}\frac{k_y}{k_h}, \quad (14.31b)$$

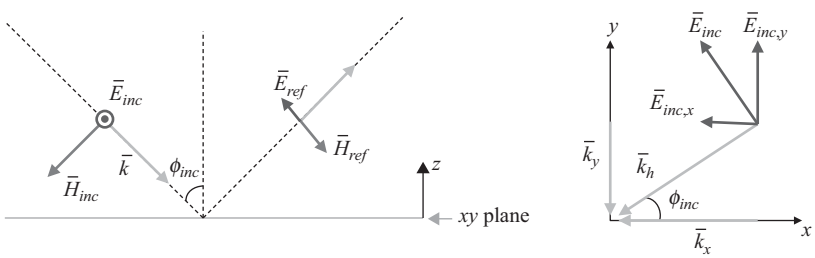

Figure 14.9 Illustration of incident fields in the TE case.

while the cross-polarized total electric field is the same as the cross-polarized reflected electric field, therefore

$$E_{ref,cr} = E_{rp,cr} = E_{rp,x}\frac{k_x}{k_h} + E_{rp,y}\frac{k_y}{k_h}. \tag{14.32}$$

The next step is to construct incident electric field such that

$$E_{inc} = \frac{1}{2}\left(E_{rp,co} + \eta_0 H_{rp,co}\frac{k}{k_z}\right), \tag{14.33}$$

where $k_z = \sqrt{k^2 - k_h^2}$. Once the incident electric field is computed, it can be subtracted from the co-polarized total electric to find the co-polarized reflected electric field as

$$E_{ref,co} = E_{rp,co} - E_{inc}, \tag{14.34}$$

which can be used to calculate the co-polarized reflection coefficient as

$$\Gamma_{co} = \frac{E_{ref,co}}{E_{inc}}. \tag{14.35}$$

Then, the cross-polarized reflected electric field can be used to determine the cross-polarized reflection coefficient as

$$\Gamma_{cr} = \frac{E_{ref,cr}}{E_{inc}}. \tag{14.36}$$

The fields captured on the transmission plane are the co- and cross-polarized components of the transmitted electric field which are similar to (14.31) and (14.32) such that

$$E_{tra,co} = E_{tra,y}\frac{k_x}{k_h} - E_{tra,x}\frac{k_y}{k_h}, \tag{14.37a}$$

$$E_{tra,cr} = E_{tra,x}\frac{k_x}{k_h} + E_{tra,y}\frac{k_y}{k_h}. \tag{14.37b}$$

Then, $E_{tra,co}$ and $E_{tra,cr}$ can be used together with E_{inc} to calculate the co- and cross-polarized transmission coefficients, respectively. However, one should notice that E_{inc} is calculated on the reflection plane while $E_{tra,co}$ and $E_{tra,cr}$ are obtained on the transmission plane. To calculate the transmission coefficients with the correct phases, E_{inc} should be evaluated on the transmission plane. If the distance between the refection and transmission planes is d_{rt}, the incident field on the transmission plane can be obtained by shifting the phase of previously defined E_{inc} by $k_z d_{rt}$, and hence co- and cross-polarized transmission coefficients can be calculated as

$$T_{co} = \frac{E_{tra,co}}{E_{inc}e^{jk_z d_{rt}}}, \tag{14.38}$$

$$T_{cr} = \frac{E_{tra,cr}}{E_{inc}e^{jk_z d_{rt}}}. \tag{14.39}$$

14.4.2 TM mode reflection and transmission coefficients

Figure 14.10 shows the incident field components for the TM mode. For the TM mode, the co-polarized total fields can be determined using the x and y components of the captured fields as

$$E_{rp,co} = -E_{rp,x}\frac{k_x}{k_h} - E_{rp,y}\frac{k_y}{k_h}, \tag{14.40a}$$

$$H_{rp,co} = H_{rp,y}\frac{k_x}{k_h} - H_{rp,x}\frac{k_y}{k_h}, \tag{14.40b}$$

while the cross-polarized total magnetic field is the same as the cross-polarized reflected magnetic field, therefore

$$H_{ref,cr} = H_{rp,cr} = H_{rp,x}\frac{k_x}{k_h} + H_{rp,y}\frac{k_y}{k_h}. \tag{14.41}$$

The next step is to construct the incident magnetic field such that

$$H_{inc} = \frac{1}{2}\left(H_{rp,co} + E_{rp,co}\frac{k}{\eta_0 k_z}\right), \tag{14.42}$$

Once the incident magnetic field is known, it can be subtracted from the co-polarized total magnetic field to find the co-polarized reflected magnetic field as

$$H_{ref,co} = H_{rp,co} - H_{inc}, \tag{14.43}$$

which can be used to calculate the co-polarized reflection coefficient as

$$\Gamma_{co} = \frac{H_{ref,co}}{H_{inc}}. \tag{14.44}$$

Then, the cross-polarized reflected magnetic field can be used to determine the cross-polarized reflection coefficient as

$$\Gamma_{cr} = \frac{H_{ref,cr}}{H_{inc}}. \tag{14.45}$$

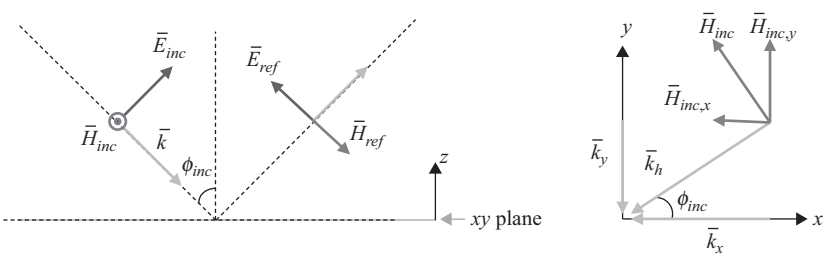

Figure 14.10 Illustration of incident fields in the TM case.

The fields captured on the transmission plane are transmitted fields, and the co- and cross-polarized components of the transmitted magnetic field can be found as

$$H_{tra,co} = H_{tra,y}\frac{k_x}{k_h} - H_{tra,x}\frac{k_y}{k_h}, \tag{14.46a}$$

$$H_{tra,cr} = H_{tra,x}\frac{k_x}{k_h} + H_{tra,y}\frac{k_y}{k_h}. \tag{14.46b}$$

Then, the co- and cross-polarized transmission coefficients can be written as

$$T_{co} = \frac{H_{tra,co}}{H_{inc}e^{jk_z d_{rt}}}, \tag{14.47}$$

$$T_{cr} = \frac{H_{tra,cr}}{H_{inc}e^{jk_z d_{rt}}}. \tag{14.48}$$

14.4.3 TEM mode reflection and transmission coefficients

Figure 14.11 illustrates the incident field in the TEM mode. For the TEM mode, the co-polarized total fields can be determined using the x and y components of the captured fields as

$$E_{rp,co} = E_{rp,x}\cos\phi + E_{rp,y}\sin\phi, \tag{14.49a}$$

$$H_{rp,co} = H_{rp,x}\sin\phi - H_{rp,y}\cos\phi, \tag{14.49b}$$

while the cross-polarized total electric field is the same as the cross-polarized reflected electric field, therefore

$$E_{ref,cr} = E_{rp,x}\sin\phi - E_{rp,y}\cos\phi. \tag{14.50}$$

Here, ϕ is the angle between the incident electric field vector and the x axis, which can be expressed as

$$\phi = \tan^{-1}\left(\frac{E_{inc,y}}{E_{inc,x}}\right). \tag{14.51}$$

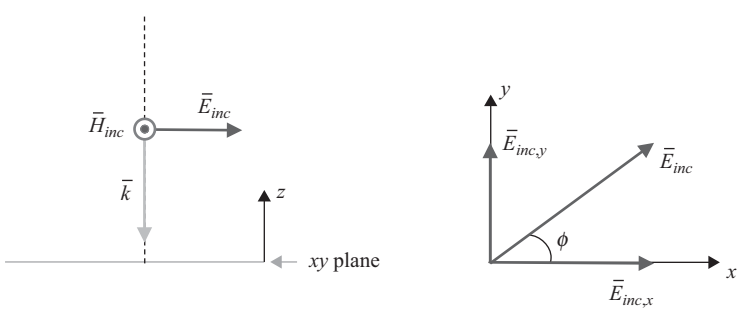

Figure 14.11 Illustration of incident fields in the TEM case.

The next step is the extraction of incident field as

$$E_{inc} = \frac{1}{2}\left(E_{rp,co} + \eta_0 H_{rp,co}\right). \tag{14.52}$$

Once the incident electric field is known, it can be subtracted from the co-polarized total electric field to find the co-polarized reflected electric field as

$$E_{ref,co} = E_{rp,co} - E_{inc}, \tag{14.53}$$

which can be used to calculate the co-polarized reflection coefficient as

$$\Gamma_{co} = \frac{E_{ref,co}}{E_{inc}}. \tag{14.54}$$

Then, the cross-polarized reflected electric field can be used to find the cross-polarized reflection coefficient as

$$\Gamma_{cr} = \frac{E_{ref,cr}}{E_{inc}}. \tag{14.55}$$

The fields captured on the transmission plane are transmitted fields, and the co- and cross-polarized components of the transmitted electric field can be found as

$$E_{tra,co} = E_{tra,x}\cos\phi + E_{tra,y}\sin\phi, \tag{14.56a}$$

$$E_{tra,cr} = E_{tra,x}\sin\phi - E_{tra,y}\cos\phi. \tag{14.56b}$$

Then, the co- and cross-polarized transmission coefficients can be written as

$$T_{co} = \frac{E_{tra,co}}{E_{inc}e^{jk_z d_{rt}}}, \tag{14.57}$$

$$T_{cr} = \frac{E_{tra,cr}}{E_{inc}e^{jk_z d_{rt}}}. \tag{14.58}$$

14.5 MATLAB® implementation of PBC FDTD algorithm

The presented constant horizontal wavenumber PBC algorithm is implemented into the base MATLAB FDTD code. The details of implementation are presented in the following subsections.

14.5.1 Definition of a PBC simulation

The parameters of a PBC simulation are specified in the subroutine ***define_problem_space_parameters***, as shown in Listing 14.1. Here, a parameter named as **periodic_boundary** is defined to store the several fields to specify various simulation parameters. First, a field mode is used to specify whether the mode of excitation will be TE, TM, or TEM. Depending on the mode, the relevant fields of **periodic_boundary** are further used to

Listing 14.1 define_problem_space_parameters.m

```
% ==<periodic boundary simulation parameters>========
disp('Defining periodic boundary simulation');

periodic_boundary.mode = 'TE'; % TE, TM, or TEM
periodic_boundary.E_phi = 1;
periodic_boundary.H_phi = 0;
periodic_boundary.E_x = 0;
periodic_boundary.E_y = 0;
periodic_boundary.kx = 100;
periodic_boundary.ky = 0;
periodic_boundary.source_z = 0.025;
periodic_boundary.reflection_z = 0.02;
periodic_boundary.transmission_z = -0.005;

% ==<boundary conditions>========
% Here we define the boundary conditions parameters
% 'pec' : perfect electric conductor
% 'cpml' : convolutional PML
% 'pbc' : Periodic Boundary Condition
% if cpml_number_of_cells is less than zero
% CPML extends inside of the domain rather than outwards

boundary.type_xn = 'pbc';
boundary.air_buffer_number_of_cells_xn = 0;
boundary.cpml_number_of_cells_xn = 8;

boundary.type_xp = 'pbc';
boundary.air_buffer_number_of_cells_xp = 0;
boundary.cpml_number_of_cells_xp = 8;

boundary.type_yn = 'pbc';
boundary.air_buffer_number_of_cells_yn = 0;
boundary.cpml_number_of_cells_yn = 10;

boundary.type_yp = 'pbc';
boundary.air_buffer_number_of_cells_yp = 0;
boundary.cpml_number_of_cells_yp = 8;

boundary.type_zn = 'cpml';
boundary.air_buffer_number_of_cells_zn = 20;
boundary.cpml_number_of_cells_zn = 8;

boundary.type_zp = 'cpml';
boundary.air_buffer_number_of_cells_zp = 60;
boundary.cpml_number_of_cells_zp = 8;
```

set simulation settings. For instance, if the mode is TEM, then the fields **E_x** and **E_y** are set with the values of the x and y components of the incident plane wave. If the mode is TE, then the field **E_phi** is the magnitude of the incident electric field, while parameters **k_x** and **k_y** are x and y components, k_x and k_y, of the constant horizontal wavenumber respectively. Similarly, if the mode is TM, then the field **H_phi** is the magnitude of the incident magnetic field, while parameters **k_x** and **k_y** are x and y components k_x and k_y. The parameters

source_z, **reflection_z**, and **transmission_z** are the *z* coordinates of the source, reflection, and transmission planes, respectively.

The presented PBC formulation assumes that four sides of the problems space – namely xn, xp, yn, and yp sides – are periodic boundaries while the other two sides, zn and zp, are CPML. These boundaries as well are specified in the ***define_problem_space_parameters***, as shown in Listing 14.1.

14.5.2 Initialization of PBC

A subroutine called as ***initialize_periodic_boundary_conditions*** is implemented to perform initialization for the PBC algorithm before the FDTD time-matching loop starts. Listing 14.2 shows the details of this subroutine.

First, the *k* indices of the source, reflection, and transmission planes are calculated and stored in respective parameters. Also, one-dimensional arrays are created for all six field components that will be used to store the transient fields captured on the reflection and transmission planes.

As discussed in the previous section, the field components need to be phase adjusted before they are averaged on the reflection and transmission planes. Since the phase correction is performed at every time step, it may be more practical to calculate and store the phase

Listing 14.2 initialize_periodic_boundary_conditions.m

```
periodic_boundary.source_ks = ...
    1+round((periodic_boundary.source_z - ...
    fdtd_domain.min_z)/fdtd_domain.dz);

if isfield(periodic_boundary,'reflection_z')
    periodic_boundary.calculate_reflection = true;
    periodic_boundary.reflection_ks = ...
    1+round((periodic_boundary.reflection_z -
    fdtd_domain.min_z)/fdtd_domain.dz);
    periodic_boundary.reflection_ex = zeros(1,number_of_time_steps);
    periodic_boundary.reflection_ey = zeros(1,number_of_time_steps);
    periodic_boundary.reflection_ez = zeros(1,number_of_time_steps);
    periodic_boundary.reflection_hx = zeros(1,number_of_time_steps);
    periodic_boundary.reflection_hy = zeros(1,number_of_time_steps);
    periodic_boundary.reflection_hz = zeros(1,number_of_time_steps);
else
        periodic_boundary.calculate_reflection = false;
end

if isfield(periodic_boundary,'transmission_z')
    periodic_boundary.calculate_transmission = true;
    periodic_boundary.transmission_ks = ...
    1+round((periodic_boundary.transmission_z -
    fdtd_domain.min_z)/fdtd_domain.dz);
    periodic_boundary.transmission_ex = zeros(1,number_of_time_steps);
    periodic_boundary.transmission_ey = zeros(1,number_of_time_steps);
    periodic_boundary.transmission_ez = zeros(1,number_of_time_steps);
    periodic_boundary.transmission_hx = zeros(1,number_of_time_steps);
    periodic_boundary.transmission_hy = zeros(1,number_of_time_steps);
    periodic_boundary.transmission_hz = zeros(1,number_of_time_steps);
    periodic_boundary.reflection_transmission_distance = ...
        dz*(periodic_boundary.reflection_ks - ...
        periodic_boundary.transmission_ks);
else
    periodic_boundary.calculate_transmission = false;
end
```

```
if strcmp(periodic_boundary.mode,'TEM')
   periodic_boundary.kx = 0;
   periodic_boundary.ky = 0;
end

kx = periodic_boundary.kx;
ky = periodic_boundary.ky;

pex = zeros(nx,ny);
pey = zeros(nx,ny);
pez = zeros(nx,ny);
phz = zeros(nx,ny);

for mi=1:nx
    for mj=1:ny
        pex(mi,mj) = exp(j*kx*(mi-0.5)*dx)*exp(j*ky*(mj-1)*dy);
        pey(mi,mj) = exp(j*kx*(mi-1)*dx)*exp(j*ky*(mj-0.5)*dy);
        pez(mi,mj) = exp(j*kx*(mi-1)*dx)*exp(j*ky*(mj-1)*dy);
        phz(mi,mj) = exp(j*kx*(mi-0.5)*dx)*exp(j*ky*(mj-0.5)*dy);
    end
end

periodic_boundary.phase_correction_ex = pex;
periodic_boundary.phase_correction_ey = pey;
periodic_boundary.phase_correction_ez = pez;

periodic_boundary.phase_correction_hx = pey;
periodic_boundary.phase_correction_hy = pex;
periodic_boundary.phase_correction_hz = phz;

periodic_boundary.Px = nx*dx;
periodic_boundary.Py = ny*dy;

initialize_plane_wave_source_for_pbc;
```

terms of the field components on a plane in two-dimensional arrays before the time-marching loop, and use these arrays at every time step. Two-dimensional phase arrays are defined and set following (14.29) as shown in Listing 14.2. One should notice that the phase array of H_x is the same as the phase array of E_y, since components of these fields are at the same coordinates in the xy plane, thus they experience the same amount of phase delay. Similarly, the phase array of H_y is the same as the phase array of E_x.

Finally, in Listing 14.2, another subroutine, ***initialize_plane_wave_source_for_pbc***, shown in Listing 14.3, is called to initialize the plane wave source. First, E_x and E_y, or H_x and H_y, components of the incident field are determined based on the mode of excitation. Then the excitation waveform, which is cosine-modulated Gaussian, is constructed. As discussed before in Section 5.1.4, one needs to determine mainly the center frequency and the bandwidth to construct a cosine-modulated Gaussian waveform. The minimum frequency in the spectrum of the waveform can be calculated using (22). The maximum frequency can be calculated using a predefined number of cells per wavelength at the highest frequency, as discussed in Section 5.1.2. In Listing 14.3, 40 cells per wavelength is used to determine the maximum frequency.

In Sections 5.1.2–5.1.4, the time constant τ was determined such that at the minimum and maximum frequencies the value of the waveform would be 10% of the maximum

Listing 14.3 initialize_plane_wave_source_for_pbc.m

```
% Here the incident plane wave is assumed to propagate in the -z direction
kx = periodic_boundary.kx;
ky = periodic_boundary.ky;
periodic_boundary.kxy = sqrt(kx^2+ky^2);
phi_incident = atan2(ky,kx);

if strcmp(periodic_boundary.mode,'TE')
    phi_incident = atan2(ky,kx);
    E_phi = periodic_boundary.E_phi;
    periodic_boundary.Exi0 = -E_phi * sin(phi_incident);
    periodic_boundary.Eyi0 = E_phi * cos(phi_incident);
    Exi = zeros(nx, nyp1);
    Eyi = zeros(nxp1, ny);
end
if strcmp(periodic_boundary.mode,'TM')
    phi_incident = atan2(ky,kx);
    H_phi = periodic_boundary.H_phi;
    periodic_boundary.Hxi0 = -H_phi * sin(phi_incident);
    periodic_boundary.Hyi0 =  H_phi * cos(phi_incident);
    Hxi = zeros(nxp1, ny);
    Hyi = zeros(nx, nyp1);
end
if strcmp(periodic_boundary.mode,'TEM')
    phi_incident = 0;
    periodic_boundary.Exi0 = periodic_boundary.E_x;
    periodic_boundary.Eyi0 = periodic_boundary.E_y;
    Exi = zeros(nx, nyp1);
    Eyi = zeros(nxp1, ny);
end

periodic_boundary.phi_incident = phi_incident;
f_min = c*periodic_boundary.kxy/(2*pi);
f_max = c/(40*max([dx,dy,dz]));%40 cells per wavelength
bandwidth = f_max-f_min;
frequency = (f_max+f_min)/2;
tau = (2*4.29/pi)./bandwidth;  %for edge of Gaussian 40 dB below max
t_0 = 4.5 * tau;
periodic_boundary.frequency_start = f_min;
periodic_boundary.frequency_end = f_max;
periodic_boundary.modulation_frequency = frequency;
periodic_boundary.bandwidth = bandwidth;
periodic_boundary.tau = tau;
periodic_boundary.t_0 = t_0;
periodic_boundary.waveform = ...
    cos(2*pi*frequency*(time - t_0)).*exp(-((time - t_0)/tau).^2);
```

value, in other words 20 dB less than the maximum value, in the frequency spectrum of the waveform. The problem of horizontal resonance, in which the fields do not decay to zero over time, needs to be avoided in a PBC simulation. The horizontal resonance would occur due to the frequencies lower than $f_{\min}$ in the spectrum. To filter out these lower frequencies better a more conservative figure than 20 dB is preferred and τ is calculated using 40 dB as a reference.

14.5.3 PBC updates in time-marching loop

Once the initializations are completed the marching loop can be modified for the PBC implementation.

14.5.3.1 Updating magnetic field PBC source on the source plane

As illustrated in Figure 14.8, first, magnetic fields are updated on the source plane to excite the source waveform if the excitation mode is TM. Listing 14.4 shows the subroutine ***update_magnetic_field_PBC_source***. Here, $H_{inc,x}$ and $H_{inc,y}$ are calculated and phase shifted following (14.27) and (14.28). Then, these field components are added to the corresponding field components on the source plane.

It should be noted that, the same shift is applied to field components at every time step, therefore, the phase shift terms could be stored as two-dimensional arrays and they could be used at every time step without recalculation, as done for the phase correction terms discussed in Section 14.5.2. In Listing 14.4 the phase shifts are recalculated to explicitly demonstrate the connection to (14.27) and (14.28).

14.5.3.2 Updating electric field PBC source on the source plane

As the next step in the time-marching loop, magnetic fields are updated in the entire computational domain followed by CPML updates. Afterwards, the electric field components on the source plane are updated in a subroutine ***update_electric_field_PBC_source*** as shown in

Listing 14.4 update_magnetic_field_PBC_source.m

```
% update magnetic field source for TM mode PBC
if strcmp(periodic_boundary.mode,'TM')

    ks = periodic_boundary.source_ks;

    Hxi(:,:) = periodic_boundary.Hxi0 ...
        * periodic_boundary.waveform(time_step);
    Hyi(:,:) = periodic_boundary.Hyi0 ...
        * periodic_boundary.waveform(time_step);

    for m = 1:nx
        Hyi(m,:) = Hyi(m,:) * exp(-j*kx*(m-0.5)*dx);
        Hxi(m,:) = Hxi(m,:) * exp(-j*kx*(m-1)*dx);
    end
    Hxi(nxp1,:) = Hxi(nxp1,:)*exp(-j*kx*nx*dx);

    for m = 1:ny
        Hyi(:,m) = Hyi(:,m) * exp(-j*ky*(m-1)*dy);
        Hxi(:,m) = Hxi(:,m) * exp(-j*ky*(m-0.5)*dy);
    end
    Hyi(:,nyp1) = Hyi(:,nyp1) * exp(-j*ky*ny*dy);

    Hx(:,:,ks) = Hx(:,:,ks) +  Hxi;
    Hy(:,:,ks) = Hy(:,:,ks) +  Hyi;

end
```

Listing 14.5 update_electric_field_PBC_source.m

```
% update electric field source for TE and TEM mode PBC

if strcmp(periodic_boundary.mode,'TE')
    ks = periodic_boundary.source_ks;

    Exi(:,:) = periodic_boundary.Exi0 ...
        * periodic_boundary.waveform(time_step);
    Eyi(:,:) = periodic_boundary.Eyi0 ...
        * periodic_boundary.waveform(time_step);

    for m = 1:nx
        Eyi(m,:) = Eyi(m,:) * exp(-j*kx*(m-1)*dx);
        Exi(m,:) = Exi(m,:) * exp(-j*kx*(m-0.5)*dx);
    end
    Eyi(nxp1,:) = Eyi(nxp1,:) * exp(-j*kx*nx*dx);

    for m = 1:ny
        Eyi(:,m) = Eyi(:,m) * exp(-j*ky*(m-0.5)*dy);
        Exi(:,m) = Exi(:,m) * exp(-j*ky*(m-1)*dy);
    end
    Exi(:,nyp1) = Exi(:,nyp1) * exp(-j*ky*ny*dy);

    Ex(:,:,ks) = Ex(:,:,ks) +  Exi;
    Ey(:,:,ks) = Ey(:,:,ks) +  Eyi;

end
if strcmp(periodic_boundary.mode,'TEM')
    ks = periodic_boundary.source_ks;

    Exi(:,:) = periodic_boundary.Exi0 ...
        * periodic_boundary.waveform(time_step);
    Eyi(:,:) = periodic_boundary.Eyi0 ...
        * periodic_boundary.waveform(time_step);

    Ex(:,:,ks) = Ex(:,:,ks) +  Exi;
    Ey(:,:,ks) = Ey(:,:,ks) +  Eyi;

end
```

Listing 14.5. Here, $E_{inc,x}$ and $E_{inc,y}$ are calculated and phase shifted following (14.25) and (14.26). Then, these field components are added to the corresponding field components on the source plane.

14.5.3.3 Updating electric field on PBC boundaries

The electric fields are then updated in the entire computational domain, except for the components lying on the PBC boundaries, followed by CPML updates. Afterwards, the electric field components are updated on the PBC boundaries in a subroutine ***update_electric_field_PBC_ABC*** as shown in Listing 14.6. Here, the components of E_x, E_y, and E_z are updated based on (14.9)–(14.20).

Listing 14.6 update_electric_field_PBC_ABC.m

```
if strcmp(periodic_boundary.mode,'TEM')
    Ex(1:nx,1,2:nz) = Cexe(1:nx,1,2:nz).*Ex(1:nx,1,2:nz) ...
        + Cexhz(1:nx,1,2:nz).*...
            (Hz(1:nx,1,2:nz)-Hz(1:nx,ny,2:nz)) ...
        + Cexhy(1:nx,1,2:nz).*...
            (Hy(1:nx,1,2:nz)-Hy(1:nx,1,1:nz-1));

    Ex(1:nx,nyp1,2:nz) = Ex(1:nx,1,2:nz);

    Ey(1,1:ny,2:nz) = Ceye(1,1:ny,2:nz).*Ey(1,1:ny,2:nz) ...
        + Ceyhx(1,1:ny,2:nz).* ...
            (Hx(1,1:ny,2:nz)-Hx(1,1:ny,1:nz-1)) ...
        + Ceyhz(1,1:ny,2:nz).* ...
            (Hz(1,1:ny,2:nz)-Hz(nx,1:ny,2:nz));

    Ey(nxp1,1:ny,2:nz) = Ey(1,1:ny,2:nz);

    Ez(1,2:ny,1:nz) = Ceze(1,2:ny,1:nz).*Ez(1,2:ny,1:nz) ...
        + Cezhy(1,2:ny,1:nz).* ...
            (Hy(1,2:ny,1:nz)-Hy(nx,2:ny,1:nz)) ...
        + Cezhx(1,2:ny,1:nz).*...
            (Hx(1,2:ny,1:nz)-Hx(1,1:ny-1,1:nz));

    Ez(nxp1,2:ny,1:nz) = Ez(1,2:ny,1:nz); ...

    Ez(2:nx,1,1:nz) = Ceze(2:nx,1,1:nz).*Ez(2:nx,1,1:nz) ...
        + Cezhy(2:nx,1,1:nz).* ...
            (Hy(2:nx,1,1:nz)-Hy(1:nx-1,1,1:nz)) ...
        + Cezhx(2:nx,1,1:nz).*...
            (Hx(2:nx,1,1:nz)-Hx(2:nx,ny,1:nz));

    Ez(2:nx,nyp1,1:nz) = Ez(2:nx,1,1:nz);

    Ez(1,1,1:nz) = Ceze(1,1,1:nz).*Ez(1,1,1:nz) ...
        + Cezhy(1,1,1:nz).* ...
            (Hy(1,1,1:nz)-Hy(nx,1,1:nz)) ...
        + Cezhx(1,1,1:nz).*...
            (Hx(1,1,1:nz)-Hx(1,ny,1:nz));

    Ez(nxp1,1,1:nz) = Ez(1,1,1:nz);

    Ez(1,nyp1,1:nz) = Ez(1,1,1:nz);

    Ez(nxp1,nyp1,1:nz) = Ez(1,1,1:nz);
end

if strcmp(periodic_boundary.mode,'TE') ...
        || strcmp(periodic_boundary.mode,'TM')

    kx = periodic_boundary.kx;
    ky = periodic_boundary.ky;
    Px = periodic_boundary.Px;
    Py = periodic_boundary.Py;

    Ex(1:nx,1,2:nz) = Cexe(1:nx,1,2:nz).*Ex(1:nx,1,2:nz) ...
        + Cexhz(1:nx,1,2:nz).*...
            (Hz(1:nx,1,2:nz)-Hz(1:nx,ny,2:nz)*exp(j*ky*Py)) ...
        + Cexhy(1:nx,1,2:nz).*...
            (Hy(1:nx,1,2:nz)-Hy(1:nx,1,1:nz-1));

    Ex(1:nx,nyp1,2:nz) = Ex(1:nx,1,2:nz)*exp(-j*ky*Py);
```

```
    Ey(1,1:ny,2:nz) = Ceye(1,1:ny,2:nz).*Ey(1,1:ny,2:nz) ...
        + Ceyhx(1,1:ny,2:nz).* ...
            (Hx(1,1:ny,2:nz)-Hx(1,1:ny,1:nz-1)) ...
        + Ceyhz(1,1:ny,2:nz).* ...
            (Hz(1,1:ny,2:nz)-Hz(nx,1:ny,2:nz)*exp(j*kx*Px));

    Ey(nxp1,1:ny,2:nz) = Ey(1,1:ny,2:nz)*exp(-j*kx*Px);

    Ez(1,2:ny,1:nz) = Ceze(1,2:ny,1:nz).*Ez(1,2:ny,1:nz) ...
        + Cezhy(1,2:ny,1:nz).* ...
            (Hy(1,2:ny,1:nz)-Hy(nx,2:ny,1:nz)*exp(j*kx*Px)) ...
        + Cezhx(1,2:ny,1:nz).*...
            (Hx(1,2:ny,1:nz)-Hx(1,1:ny-1,1:nz));

    Ez(nxp1,2:ny,1:nz) = Ez(1,2:ny,1:nz)*exp(-j*kx*Px); ...

    Ez(2:nx,1,1:nz) = Ceze(2:nx,1,1:nz).*Ez(2:nx,1,1:nz) ...
        + Cezhy(2:nx,1,1:nz).* ...
            (Hy(2:nx,1,1:nz)-Hy(1:nx-1,1,1:nz)) ...
        + Cezhx(2:nx,1,1:nz).*...
            (Hx(2:nx,1,1:nz)-Hx(2:nx,ny,1:nz)*exp(j*ky*Py));

    Ez(2:nx,nyp1,1:nz) = Ez(2:nx,1,1:nz)*exp(-j*ky*Py);

    Ez(1,1,1:nz) = Ceze(1,1,1:nz).*Ez(1,1,1:nz) ...
        + Cezhy(1,1,1:nz).* ...
            (Hy(1,1,1:nz)-Hy(nx,1,1:nz)*exp(j*kx*Px)) ...
        + Cezhx(1,1,1:nz).*...
            (Hx(1,1,1:nz)-Hx(1,ny,1:nz)*exp(j*ky*Py));

    Ez(nxp1,1,1:nz) = Ez(1,1,1:nz)*exp(-j*kx*Px);

    Ez(1,nyp1,1:nz) = Ez(1,1,1:nz)*exp(-j*ky*Py);

    Ez(nxp1,nyp1,1:nz) = Ez(1,1,1:nz)*exp(-j*kx*Px)*exp(-j*ky*Py);
end
```

14.5.3.4 Capturing fields on the reflection and transmission planes

Listing 14.7 shows the implementation of a subroutine, ***capture_fields_for_PBC***, used to capture fields on the reflection and transmission planes after the field updates at every time step. First, all field components on the reflection plane are captured and multiplied by two-dimensional phase correction arrays based on (14.29) as discussed in Section 14.4. Then the results are averaged and stored in transient reflection arrays. The same calculations are repeated to capture transmitted fields on the transmission plane.

14.5.3.5 Calculation of reflection and transmission coefficients

The transient fields on reflection and transmission planes are available in one-dimensional arrays after the FDTD time-marching loop is completed. These transients are used to calculate the reflection and transmission coefficients during the post-processing phase of the FDTD simulation in a subroutine ***calculate_reflection_and_transmission_for_PBC***, shown in Listing 14.8. The transient fields are transformed to frequency domain using DFT. Then, co- and cross-polarized reflection and transmission coefficients are calculated based on the

Listing 14.7 capture_fields_for_PBC.m

```
nxy = nx*ny;

if periodic_boundary.calculate_reflection

    ks = periodic_boundary.reflection_ks;

    ex = Ex(1:nx,1:ny,ks).*periodic_boundary.phase_correction_ex;
    ey = Ey(1:nx,1:ny,ks).*periodic_boundary.phase_correction_ey;
    ez = 0.5*(Ez(1:nx,1:ny,ks)+Ez(1:nx,1:ny,ks-1)) ...
          .*periodic_boundary.phase_correction_ez;

    hx = 0.5*(Hx(1:nx,1:ny,ks)+Hx(1:nx,1:ny,ks-1)) ...
       .*periodic_boundary.phase_correction_hx;
    hy = 0.5*(Hy(1:nx,1:ny,ks)+Hy(1:nx,1:ny,ks-1)) ...
       .*periodic_boundary.phase_correction_hy;
    hz = Hz(1:nx,1:ny,ks).*periodic_boundary.phase_correction_hz;

    periodic_boundary.reflection_ex(time_step) = sum(sum(ex))/nxy;
    periodic_boundary.reflection_ey(time_step) = sum(sum(ey))/nxy;
    periodic_boundary.reflection_ez(time_step) = sum(sum(ez))/nxy;

    periodic_boundary.reflection_hx(time_step) = sum(sum(hx))/nxy;
    periodic_boundary.reflection_hy(time_step) = sum(sum(hy))/nxy;
    periodic_boundary.reflection_hz(time_step) = sum(sum(hz))/nxy;
end

if periodic_boundary.calculate_transmission
    ks = periodic_boundary.transmission_ks;

    ex = Ex(1:nx,1:ny,ks).*periodic_boundary.phase_correction_ex;
    ey = Ey(1:nx,1:ny,ks).*periodic_boundary.phase_correction_ey;
    ez = 0.5*(Ez(1:nx,1:ny,ks)+Ez(1:nx,1:ny,ks-1)) ...
          .*periodic_boundary.phase_correction_ez;

    hx = 0.5*(Hx(1:nx,1:ny,ks)+Hx(1:nx,1:ny,ks-1)) ...
       .*periodic_boundary.phase_correction_hx;
    hy = 0.5*(Hy(1:nx,1:ny,ks)+Hy(1:nx,1:ny,ks-1)) ...
       .*periodic_boundary.phase_correction_hy;
    hz = Hz(1:nx,1:ny,ks).*periodic_boundary.phase_correction_hz;

    periodic_boundary.transmission_ex(time_step) = sum(sum(ex))/nxy;
    periodic_boundary.transmission_ey(time_step) = sum(sum(ey))/nxy;
    periodic_boundary.transmission_ez(time_step) = sum(sum(ez))/nxy;

    periodic_boundary.transmission_hx(time_step) = sum(sum(hx))/nxy;
    periodic_boundary.transmission_hy(time_step) = sum(sum(hy))/nxy;
    periodic_boundary.transmission_hz(time_step) = sum(sum(hz))/nxy;
end
```

Listing 14.8 calculate_reflection_and_transmission_for_PBC.m

```
frequencies = linspace(periodic_boundary.frequency_start, ...
     periodic_boundary.frequency_end, 200);

time_shift = 0;
reflection_ex_dft = time_to_frequency_domain( ...
    periodic_boundary.reflection_ex, dt, frequencies, time_shift);
reflection_ey_dft = time_to_frequency_domain( ...
    periodic_boundary.reflection_ey, dt, frequencies, time_shift);
reflection_ez_dft = time_to_frequency_domain( ...
    periodic_boundary.reflection_ez, dt, frequencies, time_shift);
time_shift = -dt/2;
reflection_hx_dft = time_to_frequency_domain( ...
    periodic_boundary.reflection_hx, dt, frequencies, time_shift);
reflection_hy_dft = time_to_frequency_domain( ...
    periodic_boundary.reflection_hy, dt, frequencies, time_shift);
reflection_hz_dft = time_to_frequency_domain( ...
   periodic_boundary.reflection_hz, dt, frequencies, time_shift);

if periodic_boundary.calculate_transmission
    time_shift = 0;
    transmission_ex_dft = time_to_frequency_domain( ...
        periodic_boundary.transmission_ex, dt, frequencies, time_shift);
    transmission_ey_dft = time_to_frequency_domain( ...
        periodic_boundary.transmission_ey, dt, frequencies, time_shift);
    transmission_ez_dft = time_to_frequency_domain( ...
        periodic_boundary.transmission_ez, dt, frequencies, time_shift);
    time_shift = -dt/2;
    transmission_hx_dft = time_to_frequency_domain( ...
        periodic_boundary.transmission_hx, dt, frequencies, time_shift);
    transmission_hy_dft = time_to_frequency_domain( ...
        periodic_boundary.transmission_hy, dt, frequencies, time_shift);
    transmission_hz_dft = time_to_frequency_domain( ...
        periodic_boundary.transmission_hz, dt, frequencies, time_shift);
end

% wavenumber components
k = frequencies*2*pi/c;
kx = periodic_boundary.kx;
ky = periodic_boundary.ky;
kh = sqrt(kx^2+ky^2);
kz = sqrt(k.^2-kh^2);

phi_incident = atan2(ky,kx);

eta_0 = sqrt(mu_0/eps_0);

if strcmp(periodic_boundary.mode,'TE')
   Erp_co = (reflection_ey_dft*kx/kh-reflection_ex_dft*ky/kh);
   Hrp_co = (reflection_hx_dft*kx/kh+reflection_hy_dft*ky/kh);
   Eref_cr = (reflection_ex_dft*kx/kh+reflection_ey_dft*ky/kh);

   Einc = (Erp_co+eta_0*Hrp_co.*k./kz)/2;
   Eref_co = Erp_co-Einc;

   Gamma_co = Eref_co./Einc;
   aGamma_co = abs(Gamma_co);                % Magnitude
   pGamma_co = angle(Gamma_co)/pi*180;       % Phase in degree

   Gamma_cr  = Eref_cr./Einc;
   aGamma_cr = abs(Gamma_cr);                % Magnitude
   pGamma_cr = angle(Gamma_cr)/pi*180;       % Phase in degree
```

```
    if periodic_boundary.calculate_transmission
63      Etra_co = (transmission_ey_dft*kx/kh-transmission_ex_dft*ky/kh);
        Etra_cr = (transmission_ex_dft*kx/kh+transmission_ey_dft*ky/kh);
65
        Einc = Einc .* ...
67         exp(j*kz*periodic_boundary.reflection_transmission_distance);

69      T_co = Etra_co./Einc;
        aT_co = abs(T_co);                      % Magnitude
71      pT_co = angle(T_co)/pi*180;             % Phase in degree

73      T_cr = Etra_cr./Einc;
        aT_cr = abs(T_cr);                      % Magnitude
75      pT_cr = angle(T_cr)/pi*180;             % Phase in degree
    end
77 end

79 if strcmp(periodic_boundary.mode,'TM')
    Erp_co  = (-reflection_ex_dft*kx/kh-reflection_ey_dft*ky/kh);
81  Hrp_co  = (reflection_hy_dft*kx/kh-reflection_hx_dft*ky/kh);
    Href_cr = (reflection_hx_dft*kx/kh+reflection_hy_dft*ky/kh);
83
    Hinc = (Hrp_co+Erp_co.*k./(eta_0*kz))/2;
85  Href_co = Hrp_co-Hinc;

87  Gamma_co = Href_co./Hinc;
    aGamma_co = abs(Gamma_co);                  % Magnitude
89  pGamma_co = angle(Gamma_co)/pi*180;         % Phase in degree

91  Gamma_cr = Href_cr./Hinc;
    aGamma_cr = abs(Gamma_cr);                  % Magnitude
93  pGamma_cr = angle(Gamma_cr)/pi*180;         % Phase in degree

95  if periodic_boundary.calculate_transmission
        Htra_co = (transmission_hy_dft*kx/kh-transmission_hx_dft*ky/kh);
97      Htra_cr = (transmission_hx_dft*kx/kh+transmission_hy_dft*ky/kh);

99      Hinc = Hinc .* ...
           exp(j*kz*periodic_boundary.reflection_transmission_distance);
101
        T_co = Htra_co./Hinc;
103     aT_co = abs(T_co);                      % Magnitude
        pT_co = angle(T_co)/pi*180;             % Phase in degree
105
        T_cr = Htra_cr./Hinc;
107     aT_cr = abs(T_cr);                      % Magnitude
        pT_cr = angle(T_cr)/pi*180;             % Phase in degree
109 end
111
    if strcmp(periodic_boundary.mode,'TEM')
113
        Exi0 = periodic_boundary.Exi0;
115     Eyi0 = periodic_boundary.Eyi0;
        Ei0 = sqrt(Exi0^2+Eyi0^2);
117     phi = atan2(Eyi0,Exi0);

119     Erp_co  = reflection_ex_dft*cos(phi) + reflection_ey_dft*sin(phi);
        Hrp_co  = reflection_hx_dft*sin(phi) - reflection_hy_dft*cos(phi);
121     Eref_cr = reflection_ex_dft*sin(phi) - reflection_ey_dft*cos(phi);

123     Einc = (Erp_co+eta_0*Hrp_co)/2;
        Eref_co = Erp_co-Einc;
```

```
    Gamma_co = Eref_co./Einc;
    aGamma_co = abs(Gamma_co);                  % Magnitude
    pGamma_co = angle(Gamma_co)/pi*180;         % Phase in degree

    Gamma_cr  = Eref_cr./Einc;
    aGamma_cr = abs(Gamma_cr);                  % Magnitude
    pGamma_cr = angle(Gamma_cr)/pi*180;         % Phase in degree

    if periodic_boundary.calculate_transmission
        Etra_co = transmission_ex_dft*cos(phi) + transmission_ey_dft*sin(phi);
        Etra_cr = transmission_ex_dft*sin(phi) - transmission_ey_dft*cos(phi);

        Einc = Einc .* ...
            exp(j*kz*periodic_boundary.reflection_transmission_distance);

        T_co = Etra_co./Einc;
        aT_co = abs(T_co);                      % Magnitude
        pT_co = angle(T_co)/pi*180;             % Phase in degree

        T_cr = Etra_cr./Einc;
        aT_cr = abs(T_cr);                      % Magnitude
        pT_cr = angle(T_cr)/pi*180;             % Phase in degree
    end
end

fGHz = frequencies*1e-9;

figure;
plot(fGHz,aGamma_co,'b-',fGHz,aGamma_cr,'r--','linewidth',1.5);
    legend('|\Gamma_{co}|','|\Gamma_{cr}|');
if periodic_boundary.calculate_transmission
    hold on;
    plot(fGHz,aT_co,'g-.',fGHz,aT_cr,'m:','linewidth',1.5);
    legend('|\Gamma_{co}|','|\Gamma_{cr}|','|T_{co}|', '|T_{cr}|');
end
grid on;
xlabel('frequency [GHz]','fontsize',12);
ylabel('magnitude','fontsize',12);

figure;
plot(fGHz,pGamma_co,'b-',fGHz,pGamma_cr,'r--','linewidth',1.5);
legend('|\Gamma_{co}|','|\Gamma_{cr}|');
if periodic_boundary.calculate_transmission
    hold on;
    plot(fGHz,pT_co,'g-.',fGHz,pT_cr,'m:','linewidth',1.5);
    legend('|\Gamma_{co}|','|\Gamma_{cr}|','|T_{co}|', '|T_{cr}|');
end
grid on;
xlabel('frequency [GHz]','fontsize',12);
ylabel('phase [degrees]','fontsize',12);
set(gca,'fontsize',12);

figure;
timens = time*1e9;
plot(timens, abs(periodic_boundary.reflection_ex),'b-', ...
     timens, abs(periodic_boundary.reflection_ey),'r--', ...
     timens, abs(periodic_boundary.reflection_ez),'k-.', ...
    'linewidth',1.5);
xlabel('time [ns]','fontsize',12);
ylabel('magnitude','fontsize',12);
grid on;
legend('reflection E_{x}','reflection E_{y}','reflection E_{z}');
set(gca,'fontsize',12);
```

```
if periodic_boundary.calculate_transmission
    figure;
    plot(timens, abs(periodic_boundary.transmission_ex),'b-', ...
         timens, abs(periodic_boundary.transmission_ey),'r--', ...
         timens, abs(periodic_boundary.transmission_ez),'k-.', ...
        'linewidth',1.5);
    xlabel('time [ns]','fontsize',12);
    ylabel('magnitude','fontsize',12);
    grid on;
    legend('transmission E_{x}','transmission E_{y}','transmission E_{z}');
    set(gca,'fontsize',12);
end
```

equations discussed in Sections 14.4.1–14.4.3 for TE, TM, and TEM modes, respectively. Once the reflection and transmission coefficients are calculated, their magnitudes and phases are displayed in separate figures.

14.6 Simulation examples

14.6.1 Reflection and transmission coefficients of a dielectric slab

A simple example that can be verified against analytical solutions is simulation of a dielectric slab. Here, we consider a dielectric slab case with thickness of 1 cm and relative permittivity of 4, for which reflection coefficient distribution is shown in Figures 14.2 and 14.4. The slab is illuminated by a TE waveform with horizontal wavenumber of $k_x = 100$ radian/meter and $k_y = 0$ radian/meter, as shown in Listing 14.1. Figure 14.12 shows magnitudes of the

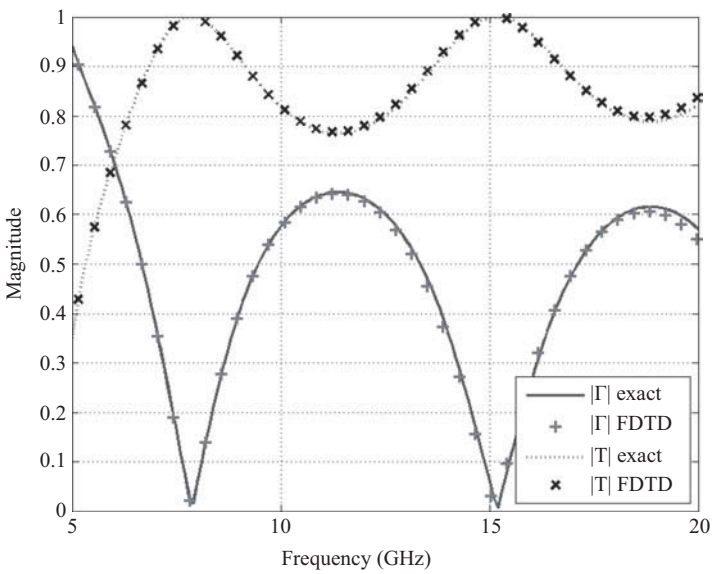

Figure 14.12 Reflection and transmission coefficients of a dielectric slab.

reflection and transmission coefficients calculated by FDTD and compared with the analytical solution. It should be noted that the minimum frequency that can be calculated using the presented PBC FDTD code is 4.8 GHz based on (14.22), hence the results are shown starting from 5 GHz in Figure 14.12.

14.6.2 Reflection and transmission coefficients of a dipole FSS

Figure 14.6 illustrates a unit cell in which a PEC rectangular patch placed on a dielectric slab. A periodic structure built on this kind of a unit cell is referred to as a Dipole Frequency Selective Surface (DFSS). In this example we consider a DFSS with a PEC patch of size 3 mm by 12 mm, on a 6 mm thick slab with 2.2 dielectric constant. The unit cell is 15 mm wide in x and y directions. The source plane is located 18 mm above the slab and it excites TE mode waves with horizontal wavenumbers $k_x = 20$ radian/meter and $k_y = 7.8$ radian/meter. The fields are captured 16 mm above and 3 mm below the slab as reflected and transmitted fields, respectively. FDTD simulation is run for 5000 time steps using a cell size of 0.5 mm on a side. Figure 14.13 shows the co- and cross-polarized reflection coefficients whereas Figure 14.14 shows the transmission coefficients calculated over the frequency range 2–15 GHz compared with solutions obtained from Ansoft Designer (AD) [76].

Next, the DFSS is simulated using TM mode waves with horizontal wavenumbers $k_x = 20$ radian/meter and $k_y = 7.8$ radian/meter as the excitation. To simulate the TM mode one can simply change the value of **periodic_boundary.mode** as 'TM' and set **periodic_boundary.H_phi** as 1 in Listing 14.1. Figure 14.15 shows the reflection coefficients whereas Figure 14.16 shows the transmission coefficients calculated by FDTD and compared with solutions obtained from Ansoft Designer.

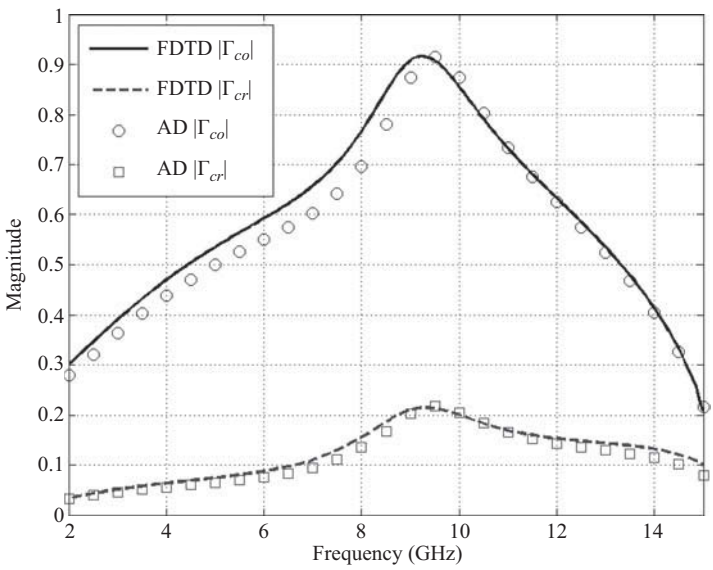

Figure 14.13 TE mode reflection coefficients of a DFSS: FDTD vs Ansoft Designer.

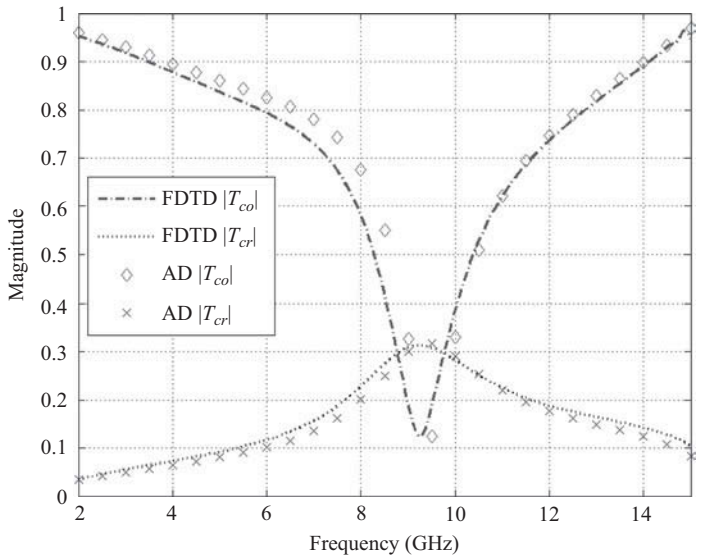

Figure 14.14 TE mode transmission coefficients of a DFSS: FDTD vs Ansoft Designer.

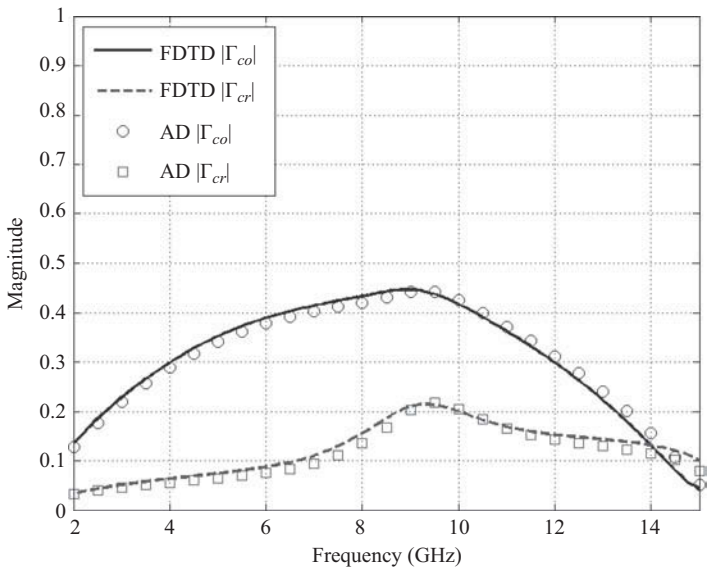

Figure 14.15 TM mode reflection coefficients of a DFSS: FDTD vs Ansoft Designer.

14.6.3 Reflection and transmission coefficients of a Jarusalem-cross FSS

In this example we consider a Jarusalem-Cross Frequency Selective Surface (JC-FSS) dimensions of which are illustrated in Figure 14.17. The PEC cross patch is in free-space

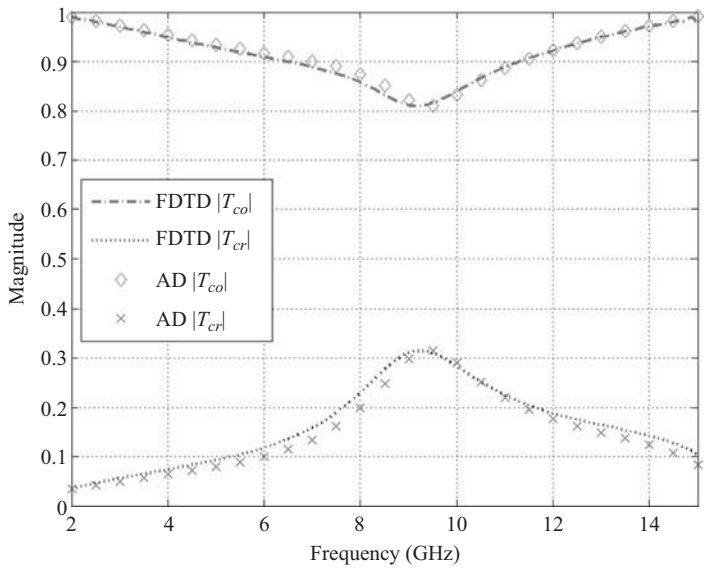

Figure 14.16 TM mode transmission coefficients of a DFSS: FDTD vs Ansoft Designer.

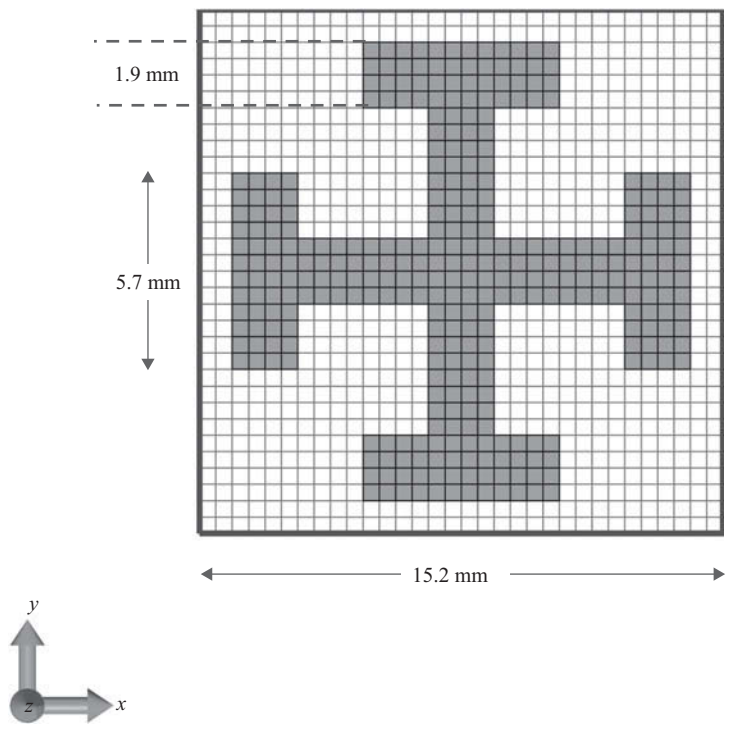

Figure 14.17 A Jarusalem-cross frequency selective circuit.

Listing 14.9 define_problem_space_parameters.m

```
% ==<periodic boundary simulation parameters>========
disp('Defining periodic boundary simulation');

periodic_boundary.mode = 'TEM';
periodic_boundary.E_x = 0;
periodic_boundary.E_y = 1;
periodic_boundary.source_z = 8e-3;
periodic_boundary.reflection_z = 6e-3;
periodic_boundary.transmission_z = -3e-3;
```

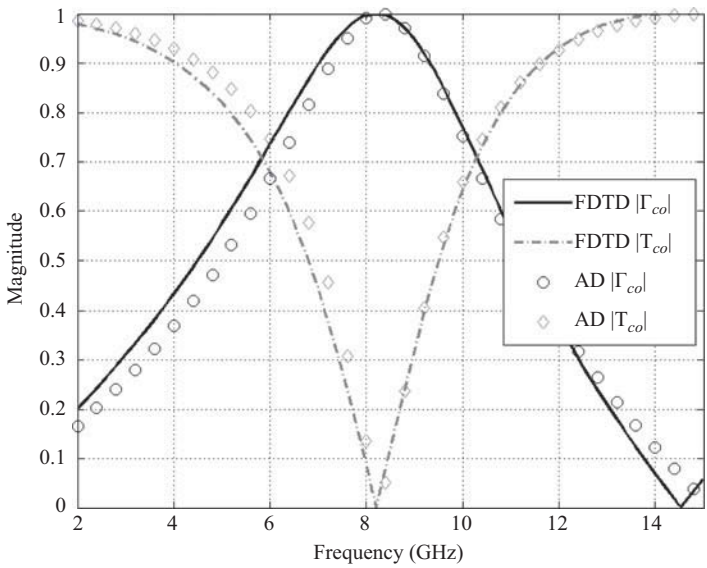

Figure 14.18 Reflection and transmission coefficients of a Jarusalem-cross FSS: FDTD vs Ansoft Designer.

without a dielectric slab support. The unit cell is 15.2 mm wide in x and y directions. The source plane is located 8 mm above the slab and it excites TEM mode waves with y polarization, as shown in Listing 14.9. The fields are captured 6 mm above and 3 mm below the patch as reflected and transmitted fields, respectively. FDTD simulation is run for 3,000 time steps using a cell size of 0.475 mm in x and y directions while 0.5 mm in z direction. Figure 14.18 shows the co- and cross-polarized reflection and transmission coefficients calculated over the frequency range 2–15 GHz compared with solutions obtained from Ansoft Designer.