1. Introduction

This program has been tested in MATLAB R2016b, Windows 7 Ultimate. You should install Parallel Computing Toolbox.

2. Numerical examples

Please run 'RunMe1.m', 'RunMe2.m' or 'RunMe3.m' directly. You should input the number of CPU cores at the beginning. Three numerical examples are introduced as below:

The first example ('RunMe1.m') is to validate the connection boundary. The Schwarzschild radius is set to 1m. There is no scatterer in FDTD domain (Fig. 1). The FDTD mesh size is set to 0.05m. The FDTD domain is: 0.6m to 1.9m in x direction, -0.65m to 0.65m in y direction, and -0.65m to 0.65m in z direction. The number of PML layers is set to 10. The connection boundaries are: x=1.25$\pm$0.25m, y=$\pm$0.25m, and z=$\pm$0.25m. A z-directed electric dipole is placed at (1m, -1m, 0m) (the midpoint of the two charges). The waveform of charge is a Gaussian pulse

$$q(t) = A\exp\left[-\frac{1}{2}\left(\frac{t}{\sigma}-4\right)^2\right] \tag{1}$$

where $A = 1\times10^{-6}\,\mathrm{C}$ and σ=3.22ns. The z components of the electric field at five positions are calculated by FDTD method. These positions are Pt1 (1.25m, 0m, 0.025m), Pt2 (1.4m, 0m, 0.025m), Pt3 (1.25m, 0.15m, 0.025m), Ps1 (1.7m, 0m, 0.025m) and Ps2 (1.25m, 0m, 0.475m). Pt1~Pt3 are located at total field zone and Ps1~Ps2 are located at scatter field zone. The electric fields at Pt1~Pt3 are also calculated by Green function method (GFM). Fig. 2(a) ~ (c) show the results at Pt1~Pt3 by the two methods. It demonstrates that the numerical results obtained through both methods match perfectly. This example validates both the FDTD code and Green function code. The results at Ps1~Ps2 are shown in Fig. 2(d), in which the values of electric field are in dB: $20\lg\left(\left|E_z\right|/\max\left|E_{z1}\right|\right)$, where E_{z1} is the electric field at Pt1. It demonstrates that the numerical scatter fields are less than -65dB.

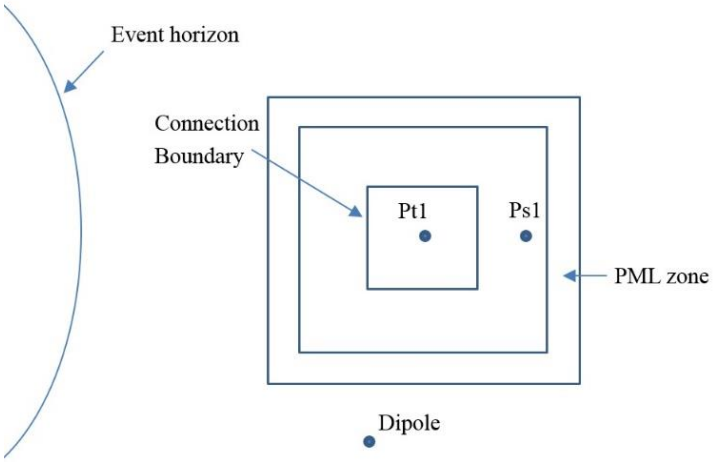
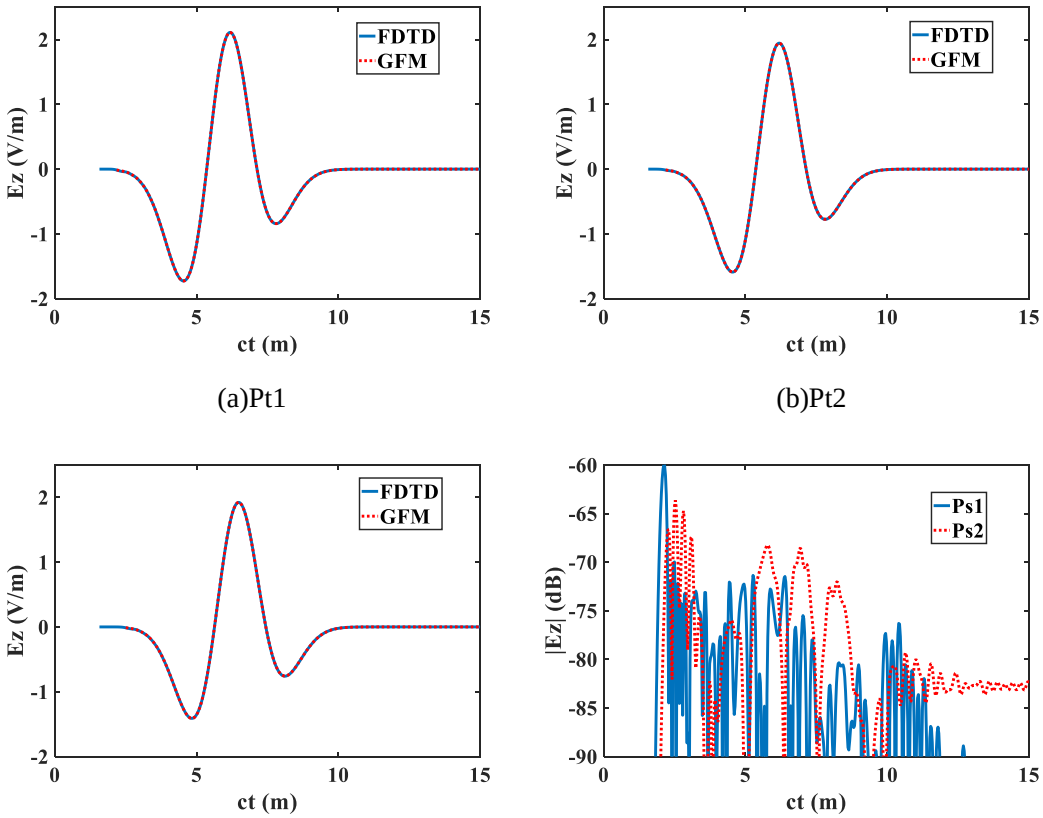


Fig. 1 Validating the connection boundary



(a)Pt1 (b)Pt2

(c)Pt3 (d)Ps1 and Ps2

Fig. 2. Ez at Pt1, Pt2, Pt3, Ps1 and Ps2

'vt1.txt', 'vt2.txt', 'vt3.txt', 'vs1.txt', 'vs2.txt' are results of FDTD method, 'dipole_GFM.mat' is result of GFM.

The second example ('RunMe2.m') is to validate the output boundary. The Schwarzschild radius is set to 1m. A z-directed electric dipole is placed at (1.5m, 0m, 0m). The waveform of charge is a Gaussian pulse with $A = 1 \times 10^{-10}$ C and σ=0.966 ns

(Eq. (1)). The FDTD mesh size is set to 0.01m and the number of PML layers is set to 10. The FDTD domain is: 1m to 2m in x direction, -0.5m to 0.5m in y direction, and -0.5m to 0.5m in z direction. The output boundaries are: x=1.5±0.05m, y=±0.05m and z=±0.05m (Fig. 3). The z components of the electric field at four positions are calculated by FDTD method. These positions are P1 (1.98m, 0m, 0.005m), P2 (1.02m, 0m, 0.005m), P3 (1.5m, 0.48m, 0.005m) and P4 (1.5m, 0m, 0.485m). The electric fields at these four positions are also calculated by integrating on output boundaries using Green function method. The results are shown in Fig. 4.

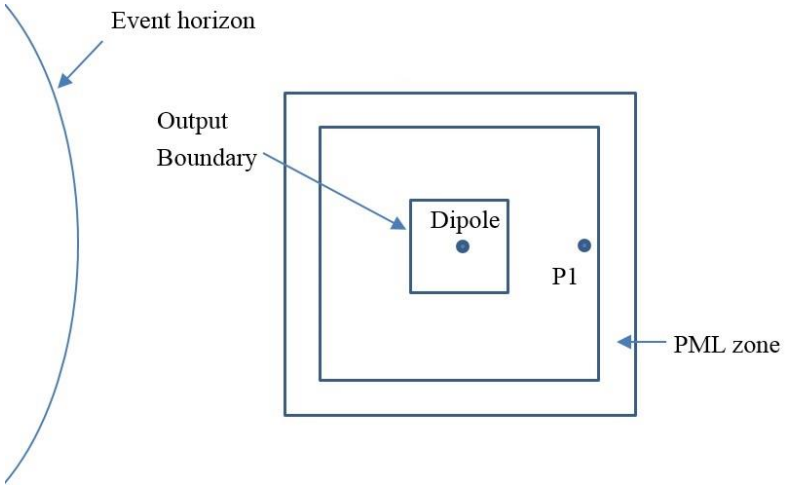
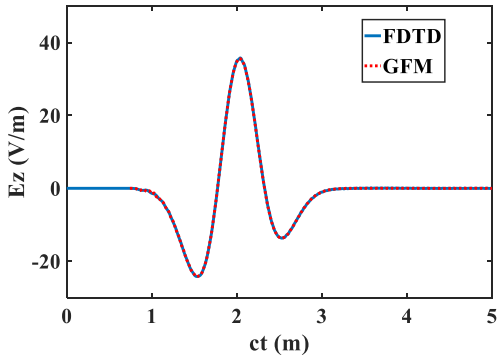
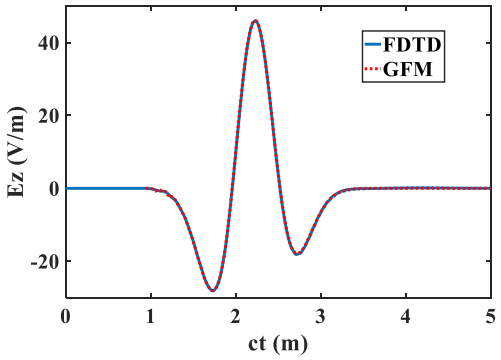


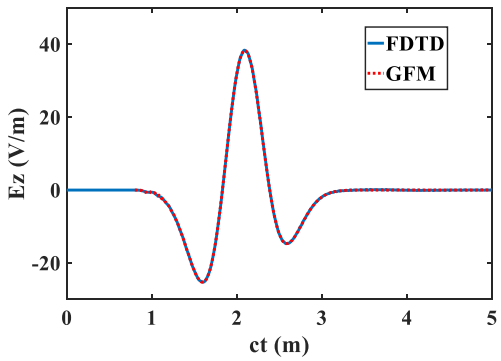
Fig. 3. Validate the output boundary



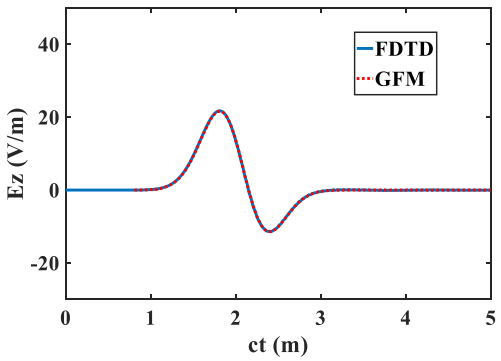
(a) P1



(b) P2



(c) P3



(d) P4

Fig. 4. Ez at P1, P2, P3 and P4

'v1.txt', 'v2.txt', 'v3.txt', 'v4.txt' are results of FDTD method, 'far.txt' is result of far fields by GFM.

The third example ('RunMe3.m') is scattering by a thin plate. The size of the thin PEC (Perfectly electric conductor) plate is 1m×1m. It spread out in the plane x=3m (Fig. 5), and the center locates at (3m, 0m, 0m). The Schwarzschild radius is set to 1m. A z-directed electric dipole is placed at P(7m, 0m, 0m). The waveform of charge is a Gaussian pulse with $A=1\times10^{-10}$ C and σ=2.415ns (Eq. (1)). The FDTD mesh size is set to 0.05m, and the number of PML layers is set to 10. The FDTD domain is: 2.4m to 3.6m in x direction, -1.1m to 1.1m in y direction, and -1.1m to 1.1m in z direction. The connection boundaries are: x=3±0.2m, y=±0.7m, and z=±0.7m. The output boundaries are: x=3±0.4m, y=±0.9m, and z=±0.9m. The z component of the scattered electric field (both in time domain and frequency domain) at P is shown in Fig. 6. The scattered electric field in flat space-time is also shown in Fig. 6. The effective light speed is smaller than that in flat space-time. This leads to time delay which is shown in Fig. 6(a). The inhomogeneity leads to pulse broading in time domain and red shift in frequency domain (Fig. 6(b)).

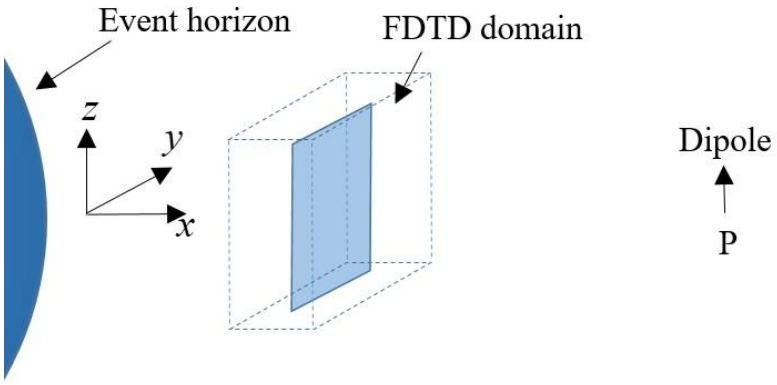
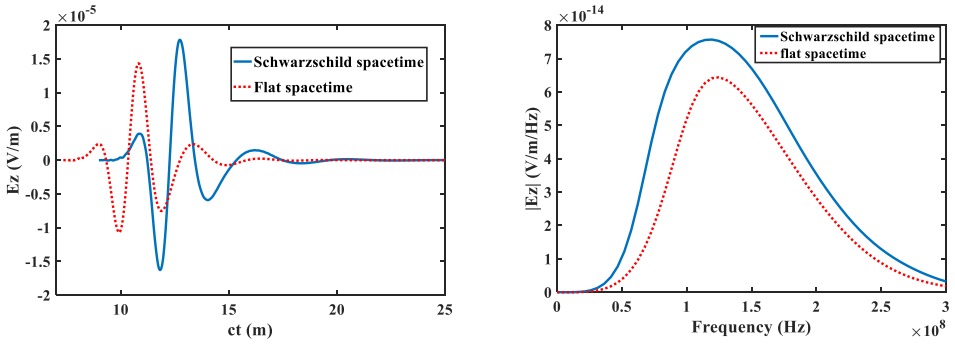


Fig. 5. Scattering by a thin plate



(a)time domain (b)frequency domain

Fig. 6. E_z at P

'far.txt' is result of far fields. 'far_flat.txt' is scattered electric field in flat space-time.

3. All files/folders

All files/folders:

fdtd_sw
output_files
conf1.mat
conf2.mat
conf3.mat
dipole_sw.m
far_flat.txt
fft_realsig.m
RunMe1.m
RunMe2.m
RunMe3.m
ReadMe.pdf
User Manual.pdf

The files 'conf1.mat', 'conf2.mat' and 'conf3.mat' includes the input data for 'RunMe1.m', 'RunMe2.m' and 'RunMe3.m' respectively. The file 'dipole_sw.m' is the program to calculate the radiation of an electric dipole by Green function method, and this program will be called by 'RunMe1.m' and 'RunMe2.m'. The file 'fft_realsig.m' is the program to calculate Fourier transform which will be called by 'RunMe3.m'. The file 'far_flat.txt' is scattered electric field in flat space-time, it will be read by 'RunMe3.m'.

The program for FDTD method is placed in the folder 'fdtd_sw'. The program for Green function method is placed in the folder ' fdtd_sw\GF_sw'.

The files in the folder 'fdtd_sw':

bound_init.m	: Initialize the FDTD boundary
dipole_add.m	: Add dipole source into the FDTD simulation
dipole_init.m	: Initialize the dipole parameters
epsrmur_sw.m	: The relative permittivity and permeability in Schwarzschild space-time
exc_init.m	: Initialize the excitation
far_accum_n.m	: Accumulating ZJ0,ZJx0,ZJy0,ZJz0,Jmx,Jmy,Jmz,Jm00
far_accum_nd.m	: Accumulating Jm0,Jmx0,Jmy0,Jmz0

far_accum_np.m : Accumulating ZJx,ZJy,ZJz,ZJ00
far_alloc.m : Allocate variables for far field
far_init.m : Initialize far field
far_Jm_x.m : Calculate $\tilde{J}_x$
far_Jm_y.m : Calculate $\tilde{J}_y$
far_Jm_z.m : Calculate $\tilde{J}_z$
far_out.m : Output far field
far_read.m : Read output files recording far field
far_ZJ_x.m : Calculate $Z_0 J_x$
far_ZJ_y.m : Calculate $Z_0 J_y$
far_ZJ_z.m : Calculate $Z_0 J_z$
farfield.m : Calculate far field
fdtd_sw.m : Main program
grid_init.m : Initialize the FDTD grid
ind_init.m : Convert type numbers to array indices
init.m : Initialize the FDTD parameters
iw_add.m : Add incident wave
iw_init.m : Initialize incident wave
iw_subtract.m : Subtract incident wave
iw_Zq.m : Incident waveform
iw_Zq1.m : Derivative of incident waveform
iw_Zq2.m : Two order derivative of incident waveform
med_bound.m : Find the medium parameter on the PML interface
out_init.m : Initialize the output parameters
out_post.m : Post-processing for output
output.m : Output
plane_init.m : Initialize the output plane
plane_out.m : Output electric field in the specified plane
plane_read.m : Read electric field in the specified plane from the output file
PML_conf.m : PML configuration
pml_init_sw.m : Initialize the PML parameters
PML_param.m : PML parameter
pml_update_e.m : Update electric field in PML domain
pml_update_h.m : Update magnetic field in PML domain
source_add.m : Add sources into the FDTD simulation
update_e_sw.m : Update electric field in non-PML domain
update_h_sw.m : Update magnetic field in non-PML domain
volt_add.m : Add voltage source into the FDTD simulation

volt_co_init.m : Initialize the coordinate of the voltage source
volt_out.m : Output voltage between the two specified points
wave_fun.m : Waveform function
wave_init.m : Initialize waveform parameters

The files in the folder ' fdtd_sw\GF_sw':

cart_equator.m : Rotate Cartesian coordinate so that the specified two points fall in the equatorial plane
cart2sph_sw.m : Convert Cartesian coordinate to spherical coordinate
Christoffel_sw.m : Christoffel symbols in spherical coordinate
Christoffel_sw_cart.m : Christoffel symbols in Cartesian coordinate
Contraction.m : Tensor contraction
ContracU2.m : Get $\square U_{\alpha\beta'}$ from $U_{\alpha\beta';\gamma\delta}$
dE.m : Differential element of electric filed
dot_TT.m : Contraction between two tensors
dot_TV.m : Contraction between a tensor and a vector
dxs_dxc.m : Transform matrix - from spherical coordinate to Cartesian coordinate
EulerTrans.m : Euler transform matrix
geodesic_sw_bvp.m : Solve geodesic equation
GF_sw_cart.m : Green function in Cartesian coordinate
GF_sw_sph.m : Green function in spherical coordinate
Int_null.m : Integral alone the null geodesic
Int_S_up.m : Solve $\sigma_{\alpha\beta'}$
Int_Sp_p.m : Solve $\sigma^{\alpha'}{}_{\beta'}$
Int_Sp_pp.m : Solve $\sigma^{\alpha'}{}_{\beta'\gamma'}$
Int_Su_u.m : Solve $\sigma^{\alpha}{}_{\beta}$
Int_Su_uu.m : Solve $\sigma^{\alpha}{}_{\beta\gamma}$
Int_Su_uuu.m : Solve $\sigma^{\alpha}{}_{\beta\gamma\delta}$
Int_U.m : Solve $U_{\alpha\beta'}$
Int_U_.m : Solve $U_{\alpha\beta'}$ alone the inverse direction
Int_U_p.m : Solve $U_{\alpha\beta';\gamma'}$
Int_U_u.m : Solve $U_{\alpha\beta';\gamma}$

Int_U_uu.m : Solve $U_{\alpha\beta';\gamma\delta}$

Int_V0.m : Solve $V^0_{\alpha\beta'}$

inv_metric_sw.m : Inverse metric

L0_s3.m : $\left(\frac{\sigma^{\alpha}{}_{\beta\gamma}}{t}\right)_{t\to 0}$

L0_U1.m : $\left(\frac{U_{\alpha\beta';\gamma}}{t}\right)_{t\to 0}$

matrix_dE : Matrix for calculating dE

metric_sw.m : Metric

prod_tensor.m : Tensor product

Riemann_sw.m : Riemann curvature

RM1_sw.m : The first order covariant derivative of Riemann curvature

RM2_sw.m : The second order covariant derivative of Riemann curvature