MMM - Group 1 - Project 1

Bjarne De Bruyn and Sander De Meyer

April 16, 2023

1 Introduction

1.1 Definition of the problem

The equations governing the EM TM case have to be solved. They will be expressed in the fields e_z, h_y, and b_x.

$$\frac{\partial e_z}{\partial x} = \frac{\partial}{\partial t}(\mu h_y) \tag{1}$$

$$\frac{\partial e_z}{\partial y} = -\frac{\partial b_x}{\partial t} \tag{2}$$

$$\frac{\partial h_y}{\partial x} - \frac{\partial}{\partial y}\left(\frac{b_x}{\mu}\right) = \frac{\partial}{\partial t}(\epsilon e_z) + \sigma e_z + j_{e,z} \tag{3}$$

These will be solved with the unidirectionally-collocated hybrid implicit-explicit (UCHIE) FDTD method, for which the location of the various fields are shown in the middle of figure 1.

1.2 Approach used in this project

Many of the interesting structures consist of blocks of (stacked) materials. If the materials are stacked in the x-direction, the material properties are step-wise constant in the y-direction. Assuming that the full UCHIE region is homogeneous in the y-direction would greatly reduce the computational cost, as will be explained in section 3. This is the approach that will be taken in this project. To still be able to describe y-dependent systems, 2 approaches are implemented: UCHIE-Yee hybridisation and UCHIE-double. These have the flexibility to describe y-dependent (piecewise constant) systems, while retaining their advantage of being much faster.

2 The code files and their uses

The relevant files are [1]

1. **UCHIE_homogeneous:** UCHIE code where no y-dependence is present.
2. **UCHIE_double:** 2 UCHIE regions, each y-homogeneous, are stacked on top of each other in the y-direction.
3. **hybrid_separate:** UCHIE-Yee hybridisation code where the UCHIE-region is y-homogeneous. There is only hybridisation in the y-direction.
4. **Yee_zelf:** Yee code without UCHIE
5. **functions:** Auxiliary functions needed for e.g. update matrices of UCHIE
6. **inversions:** File that implements different ways of matrix inversion
7. **material_properties**: File that contains classes to implement materials into the grid.

All schemes use periodic boundary conditions (PBC). In the hybrid codes, there is overlap of one cell in the y-direction whenever two regions overlap. The field values in these overlapping regions which are not correct (due to PBC or the overlap structure) are either overwritten by the correct values of the other regions, set to zero, or retained when no correct field values depend on them. These can appear in the visualisation as nonsensical and should thus be ignored.

[1] The downside to this is that on the interface between UCHIE and Yee or 2 UCHIE regions, some unphysical reflections may be present.

3 UCHIE update equations

For the discretization, one needs basis sets, the same basis sets as the syllabus are used. With these definitions, the finite elements can be introduced for the fields. Only b_x, e_z, and h_y are defined, since these are the only nonzero fields in the 2D TM case.

$$b_x(x,y,t) = \sum_{i,j,n} \hat{\hat{b}}_x|_{i,j}^n \Lambda_i(x)\hat{\Pi}_j(y)\Lambda_n(t) \tag{4}$$

$$e_z(x,y,t) = \sum_{i,j,n} \hat{\hat{e}}_z|_{i,j}^n \Lambda_i(x)\Lambda_j(y)\hat{\Pi}_n(t) \tag{5}$$

$$h_y(x,y,t) = \sum_{i,j,n} \hat{\hat{h}}_y|_{i,j}^n \Lambda_i(x)\hat{\Pi}_j^*(y)\hat{\Pi}_n(t) \tag{6}$$

One needs to be careful with these definitions, as they are normalized. With the hybridisation with Yee, this has to be taken into account. Introducing these expressions into equation (2) gives an explicit update equation

$$\hat{\hat{b}}_x|_{i,j}^{n+1} = \hat{\hat{b}}_x|_{i,j}^n - \left(\hat{\hat{e}}_z|_{i,j+1}^n - \hat{\hat{e}}_z|_{i,j}^n\right) \tag{7}$$

Equation (2) can thus be satisfied in the strong sense. The other 2 equations can only be satisfied in the weak sense, and will thus have to be tested with testing functions.

The used testing functions were

$$f(x,y,t) = \hat{\Pi}_{i'}(x)\Lambda_{j'}(y)\Lambda_{n'}(t) \tag{8}$$

Equations (1) and (3) will be multiplied with $f(x,y,t)$ and integrated over xn y, and t. To evaluate the integrals over a product of basis functions, the trapezoidal rule will be used, together with mass-lumping, to make the final matrix equations sparser.

The following are examples of integrals

$$\int \hat{\Pi}_n(t)\Lambda_{n'}(t)dt = \frac{1}{2}\left(\delta_{n',n+1} + \delta_{n',n}\right) \tag{9}$$

Example of mass-lumping:

$$\int \Lambda_n(t)\Lambda_{n'}(t)dt = \Delta t^* \delta_{n,n'} \tag{10}$$

Example of an integral for which integration by parts had to be used:

$$\int \frac{\partial}{\partial t}\left(\hat{\Pi}_n(t)\right)\Lambda_{n'}(t)dt = \frac{1}{\Delta t}\left(\delta_{n,n'} - \delta_{n,n'-1}\right) \tag{11}$$

3.1 Implicit update equations

For each value of i and j, the following equations were obtained (respectively the result of equations (1) and (3))

$$\begin{aligned} &\frac{\Delta y_{j'}^*}{\Delta x_{i'}}\hat{\hat{e}}_z|_{i'+1,j}^{n'} - \frac{\Delta y_{j'}^*}{\Delta x_{i'}}\hat{\hat{e}}_z|_{i',j}^{n'} - \frac{\mu(i'+1)}{\Delta t}\hat{\hat{h}}_y|_{i'+1,j'}^{n'} - \frac{\mu(i')}{\Delta t}\hat{\hat{h}}_y|_{i',j'}^{n'} \\ &= -\frac{\Delta y_{j'}^*}{\Delta x_{i'}}\hat{\hat{e}}_z|_{i'+1,j}^{n'-1} + \frac{\Delta y_{j'}^*}{\Delta x_{i'}}\hat{\hat{e}}_z|_{i',j}^{n'-1} - \frac{\mu(i'+1)}{\Delta t}\hat{\hat{h}}_y|_{i'+1,j'}^{n'} - \frac{\mu(i')}{\Delta t}\hat{\hat{h}}_y|_{i',j'}^{n'} \end{aligned} \tag{12}$$

$$
\begin{aligned}
&\left(\left(\frac{\epsilon}{\Delta t}+\frac{\sigma}{2}\right)\Delta y^{*}\hat{\hat{e}}_z\right)\Big|_{i'+1,j'}^{n'}+\left(\left(\frac{\epsilon}{\Delta t}+\frac{\sigma}{2}\right)\Delta y^{*}\hat{\hat{e}}_z\right)\Big|_{i',j'}^{n'}-\left(\frac{\hat{\hat{h}}_y}{\Delta x_{i'}}\right)\Big|_{i',j}^{n'}+\left(\frac{\hat{\hat{h}}_y}{\Delta x_{i'}}\right)\Big|_{i'+1,j}^{n'}\\
&=\left(\left(\frac{\epsilon}{\Delta t}-\frac{\sigma}{2}\right)\Delta y^{*}\hat{\hat{e}}_z\right)\Big|_{i'+1,j'}^{n'-1}+\left(\left(\frac{\epsilon}{\Delta t}-\frac{\sigma}{2}\right)\Delta y^{*}\hat{\hat{e}}_z\right)\Big|_{i',j'}^{n'-1}+\left(\frac{\hat{\hat{h}}_y}{\Delta x_{i'}}\right)\Big|_{i'+1,j}^{n'-1}-\left(\frac{\hat{\hat{h}}_y}{\Delta x_{i'}}\right)\Big|_{i',j}^{n'-1}\\
&-\Delta t\left(\left(\frac{\hat{\hat{b}}_x}{\mu\Delta y}\right)\Big|_{i'+1,j'}^{n'}+\left(\frac{\hat{\hat{b}}_x}{\mu\Delta y}\right)\Big|_{i',j'}^{n'}-\left(\frac{\hat{\hat{b}}_x}{\mu\Delta y}\right)\Big|_{i'+1,j'-1}^{n'}-\left(\frac{\hat{\hat{b}}_x}{\mu\Delta y}\right)\Big|_{i',j'-1}^{n'}\right)\\
&-\left(\frac{\Delta y^{*}j_{e,z}}{2}\right)\Big|_{i'+1,j'}^{n'}-\left(\frac{\Delta y^{*}j_{e,z}}{2}\right)\Big|_{i',j'}^{n'}-\left(\frac{\Delta y^{*}j_{e,z}}{2}\right)\Big|_{i'+1,j'}^{n'-1}-\left(\frac{\Delta y^{*}j_{e,z}}{2}\right)\Big|_{i',j'}^{n'-1}
\end{aligned}
\tag{13}
$$

These result in the (sparse) matrix equations

$$AX^{new}=BX^{old}+C+D \tag{14}$$

$$x=\begin{bmatrix} e_z \\ h_y \end{bmatrix} \tag{15}$$

$$A=\begin{bmatrix} \Delta y^{*}D_1 & -\frac{1}{\Delta t}A_1 \\ \Delta y^{*}A_2 & -D_1 \end{bmatrix} \tag{16}$$

$$B=\begin{bmatrix} -\Delta y^{*}D_1 & -\frac{1}{\Delta t}A_1 \\ \Delta y^{*}A_3 & D_1 \end{bmatrix} \tag{17}$$

$$D_1=\begin{bmatrix} \Delta x_1^{-1} & & \\ & \ddots & \\ & & \Delta x_M^{-1} \end{bmatrix}\times\begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ \vdots & & \ddots & & \vdots \\ & & & -1 & 1 \\ 1 & & & & -1 \end{bmatrix} \tag{18}$$

$$A_1=\begin{bmatrix} \mu(1) & & \\ & \ddots & \\ & & \mu(M) \end{bmatrix}\times\begin{bmatrix} 1 & 1 & & & \\ & 1 & 1 & & \\ \vdots & & \ddots & & \vdots \\ & & & 1 & 1 \\ 1 & & & & 1 \end{bmatrix} \tag{19}$$

$$A_2=\begin{bmatrix} \frac{\epsilon(1)}{\Delta t}+\frac{\sigma(1)}{2} & & \\ & \ddots & \\ & & \frac{\epsilon(M)}{\Delta t}+\frac{\sigma(M)}{2} \end{bmatrix}\times\begin{bmatrix} 1 & 1 & & & \\ & 1 & 1 & & \\ \vdots & & \ddots & & \vdots \\ & & & 1 & 1 \\ 1 & & & & 1 \end{bmatrix} \tag{20}$$

$$A_3=\begin{bmatrix} \frac{\epsilon(1)}{\Delta t}-\frac{\sigma(1)}{2} & & \\ & \ddots & \\ & & \frac{\epsilon(M)}{\Delta t}-\frac{\sigma(M)}{2} \end{bmatrix}\times\begin{bmatrix} 1 & 1 & & & \\ & 1 & 1 & & \\ \vdots & & \ddots & & \vdots \\ & & & 1 & 1 \\ 1 & & & & 1 \end{bmatrix} \tag{21}$$

Matrices C and D correspond to the 3rd and 4th line of equation (13) respectively. They can be obtained by simple cyclic permutations of the matrix $[b_x]$ and $[j_{e,z}]$ after element-wise multiplication with the relevant prefactors.

Matrices C and D have to be computed each iteration step, but matrices A and B can be computed beforehand. Based on those matrices, the matrices A^{-1} and $A^{-1}B$ can be precomputed, which together with C and D determine the iteration step. Here, the true power of the y-independent approach arises. For a large number of cells, the biggest computational cost is the inversion of matrix A. The fact that there is no y-dependence makes sure A has a shape of $2M \times 2M$, with M the number of cells in the x-direction, independent of the number of cells in the y-direction. The matrices X, C, and D have shape $2M \times N$. The dependence on N is thus only found in the multiplication of (sparse) matrices, not in taking the inverse. The total computation time then consists of taking the inverse of a $2M \times 2M$ matrix, plus a term that scales linearly in M, N, and the number of iterations. To introduce y-dependent behavior, a number of UCHIE cells have to be stacked on top of each other (the code works only for 2 UCHIE regions, but can easily be extended to more. Alternatively, the Yee-UCHIE hybridisation can be used), which determines the number of matrices that need to be inverted. Apart from that, the iteration time still scales linearly in the total number of cells in the y-direction.

4 Yee update equations

$$b_x(x,y,t) = \sum_{i,j,n} \hat{\hat{b}}_x|^n_{i,j} \Pi^*_i(x)\Lambda^*_j(y)\Lambda_n(t) \tag{22}$$

$$e_z(x,y,t) = \sum_{i,j,n} \hat{e}_z|^n_{i,j} \Lambda_i(x)\Lambda_j(y)\Lambda^*_n(t) \tag{23}$$

$$h_y(x,y,t) = \sum_{i,j,n} \hat{\hat{h}}_y|^n_{i,j} \Lambda^*_i(x)\Pi^*_j(y)\Lambda_n(t) \tag{24}$$

$$h_y\Big|^n_{i,j} = h_y\Big|^{n-1}_{i,j} + \frac{\Delta t}{\mu \Delta x}\left(e_z\Big|^{n-1}_{i+1,j} - e_z\Big|^{n-1}_{i,j}\right) \tag{25}$$

$$b_x\Big|^n_{i,j} = b_x\Big|^{n-1}_{i,j} - \frac{\Delta t}{\Delta y}\left(e_z\Big|^{n-1}_{i,j+1} - e_z\Big|^{n-1}_{i,j}\right) \tag{26}$$

$$e_z\Big|^n_{i,j} = \frac{1}{\frac{\epsilon}{\Delta t} + \frac{\sigma}{2}}\left[\left(\frac{\epsilon}{\Delta t} - \frac{\sigma}{2}\right) e_z\Big|^{n-1}_{i,j} - \frac{1}{2}\left(j_{ez}\Big|^n_{i,j} + j_{ez}\Big|^{n-1}_{i,j}\right) + \frac{1}{\Delta x}\left(h_y\Big|^n_{i,j} - h_y\Big|^n_{i-1,j}\right) - \frac{1}{\mu \Delta y}\left(b_x\Big|^n_{i,j} - b_x\Big|^n_{i,j-1}\right)\right] \tag{27}$$

5 Yee-UCHIE hybridisation

The definition of the fields is visualized in figure 1. The upper and lower region are the Yee regions, the middle region is the UCHIE region. blue and red arrows/dots means staggering in time.

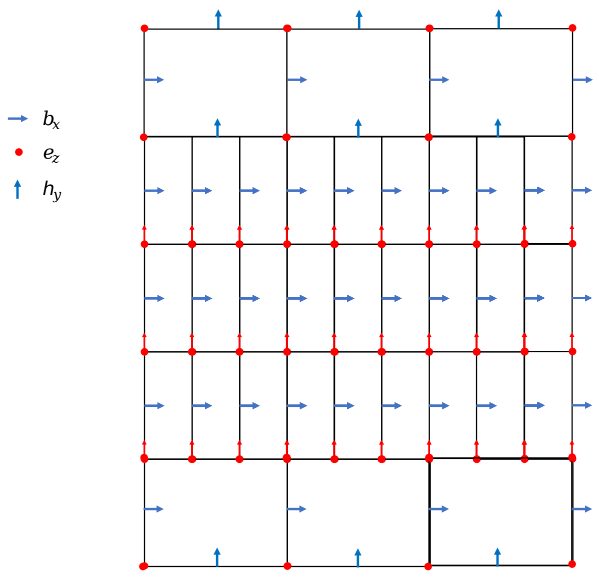


Figure 1: Definition of the fields in the UCHIE and Yee region. The middle part is also used for the pure UCHIE codes.

For the update of b_x at the top and the bottom of the UCHIE region, e_z of the Yee region has to be interpolated. This can be done by multiplying an interpolation matrix with the array of e_z values just above or just below the UCHIE region. In this interpolation, one needs to be careful with the exact normalisations used above.

To make the connection between the UCHIE and Yee region, the values of the fields are sometimes transferred to the other region. To make this connection, one cell at the top and one cell at the bottom of UCHIE region is also included in the Yee-region. Only the b_x fields in these Yee cells are physically relevant, however (the e_z and h_y fields take into account PBC for the UCHIE region, which is not physical here), and these fields are overwritten by the b_x fields at the same location of the UCHIE region.

5.1 Left and right interface conditions

To have a full hybridisation with Yee, the left and right interfaces also need special care. The equations are, at the right interface

$$\begin{aligned}\frac{\mu}{\Delta t} H_y|^n_{i,j} &= \frac{\mu}{\Delta t} H_y|^{n-1}_{i,j} - \frac{1}{\Delta x} E_z|^{n-1}_{i,j} + \frac{1}{\Delta x} E_z^{Yee}|^{n-1}_{i+\Delta x,j} \left(\frac{\epsilon}{\Delta t} + \frac{\sigma}{2}\right) E_z|^n_{i,j} + \frac{1}{\Delta x} H_y \\ |^n_{i,j} &= \left(\frac{\epsilon}{\Delta t} - \frac{\sigma}{2}\right) E_z|^n_{i,j} - \frac{1}{\Delta x} H_y|^n_{i,j} - j_z|^n_{i,j} + \frac{2}{\Delta x} H_y^{Yee}|^n_{i-1,j} - \frac{1}{\mu \Delta y}\left(B_x|^n_{i,j} - B_x|^n_{i,j-1}\right)\end{aligned} \tag{28}$$

At the left interface

$$\begin{aligned}\frac{\mu}{\Delta t} H_y|^n_{i,j} &= \frac{\mu}{\Delta t} H_y|^{n-1}_{i,j} + \frac{1}{\Delta x} E_z|^{n-1}_{i,j} - \frac{1}{\Delta x} E_z^{Yee}|^{n-1}_{i\Delta x,j} \\ \left(\frac{\epsilon}{\Delta t} + \frac{\sigma}{2}\right) E_z|^n_{i,j} - \frac{1}{\Delta x} H_y|^n_{i,j} &= \left(\frac{\epsilon}{\Delta t} - \frac{\sigma}{2}\right) E_z|^n_{i,j} + \frac{1}{\Delta x} H_y|^n_{i,j} - j_z|^n_{i,j} - \frac{2}{\Delta x} H_y^{Yee}|^n_{i-1,j} - \frac{1}{\mu \Delta y}\left(B_x|^n_{i,j} - B_x|^n_{i,j-1}\right)\end{aligned} \tag{29}$$

Implementing these, however, unfortunately resulted in an unstable code. This is why it was opted to only implement the hybridisation in the y-direction, and to make the UCHIE cell the same size in the x-direction as the complete simulation domain. Still, periodic boundary conditions were implemented in the y-direction (Yee region) and the x-direction (both Yee and UCHIE region)

6 Inversion of matrices

As updating the field values in the UCHIE region requires the inversion of the matrix A, multiple options were explored for the computation of this inverse. The 4 methods that were tested can be divided into 2 groups: direct inversions and inversions using the Schur complement. For the inversion using the Schur complement, the $2M \times 2M$ matrix A was subdivided into 4 submatrices of dimensions $M + 1 \times M + 1$, $M - 1 \times M - 1$, $M + 1 \times M - 1$ and $M - 1 \times M + 1$. This subdivision was chosen because some of the $M \times M$ matrices of the equal division were singular. The quantity of interest is the time it takes to invert the matrix A using the different methods. For both methods the time required for inversion of a dense matrix using the numpy package was compared to the required time for inversion of sparse matrices using the scipy package. The results are given in figure 2. This figure shows that generally both direct and Schur complement based inversion are faster when using the numpy package. It should be mentioned that when going to even larger M, order 10^4, the usage of sparse matrices for direct inversion proved to be more efficient in terms of computation times, which is hinted at when considering the inversion times for the largest value of M. For this value of M the inversion times for direct inversion using numpy and scipy and Schur complement based inversion using numpy are about equal. For such large matrices, it is also more efficient to use sparse algorithms in the matrix multiplication. The dimensions that will be considered in the actual implementation are however in a range below 1000 such that methods using numpy are preferred to their scipy counterpart.

7 Stability analysis

The UCHIE update equations are of the form

$$X^{new} = U X^{old} + V \tag{30}$$

With U the update matrix, based on the matrices A and B defined in 3

$$U = A^{-1} B \tag{31}$$

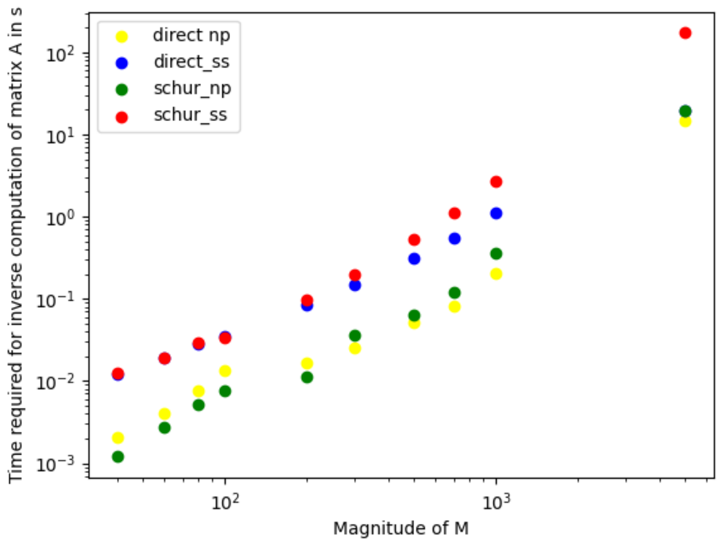


Figure 2: Comparison of the efficiency of the 4 different methods that can be used to invert the $2M \times 2M$ matrix A. The time it takes to invert the matrices in seconds is given as a function of the magnitude of M in a log-log plot. Direct_np and direct_ss denote direct inversion of the matrix A using numpy and scipy, respectively. Schur_np and schur_ss denote inversion of the matrix A by usage of the Schur complement using numpy and scipy, respectively.

The UCHIE code will be stable only if all the eigenvalues of U are located within the unit circle (this is a necessary, but not sufficient, condition). The resulting eigenvalues are shown in figure 3 for a courant number equal to 1. Indeed, for both the vacuum and the MIS structure, the eigenvalues are located within the unit cell, up to numerical errors of the order 10^{-15}. As expected, eigenvalues with a modulus smaller than 1 are present in the case of the lossy system: the fields, denoted by X in equation 30, can be expanded as a function of the eigenvectors of U such that action of U on X, i.e. updating the fields, can be seen as the multiplication of every eigenvector with its corresponding eigenvalue. As the system shows losses, the fields will lose energy corresponding to the diminishing contribution of the eigenvectors with modulus of the eigenvalue smaller than 1.

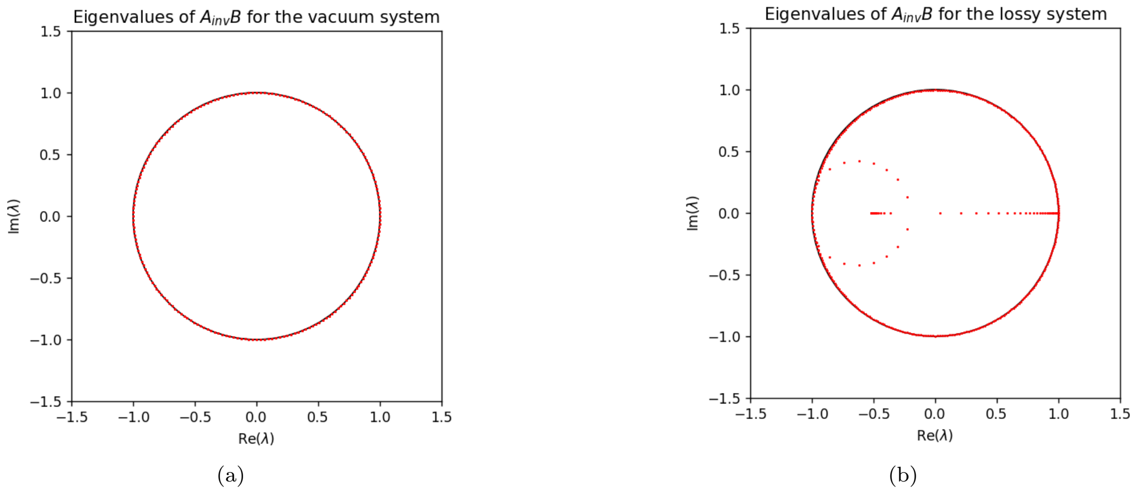


Figure 3: Eigenvalues for (a) vacuum and (b) the lossy structure with a block of a material with $e_r = 3$ and $\sigma = 0.1S/m$

8 Comparison with analytical solution

A validation of the UCHIE scheme can be done by looking at the frequency domain. If the simulation is done for a dirac pulse (which is an elementary line source in the frequency domain), $e_z(x, y, t)$ for a given (x, y) should be equal to

$$e_z(x,y,\omega) = -J_0 \frac{\omega\mu_0}{4} H_0^{(2)}(\frac{\omega}{c}\sqrt{(x-x_s)^2+(y-y_s)^2} \tag{32}$$

With $H_0^{(2)}$ the 0^{th}-order Hankel function of the 2^{nd} kind. This can also be done (as in our case) for a modulated Gaussian source. The DFT of e_z then has to be divided by the DFT of the source itself before comparison with equation (32). The result is given in figure 4. It is indeed seen that the DFT follow the Hankel function for values in the bandwidth, which is around 6 GHz. For higher frequencies, deviations of the computational results to the analytical solution are seen, as expected. Comparison with our own Yee code gave the same result.

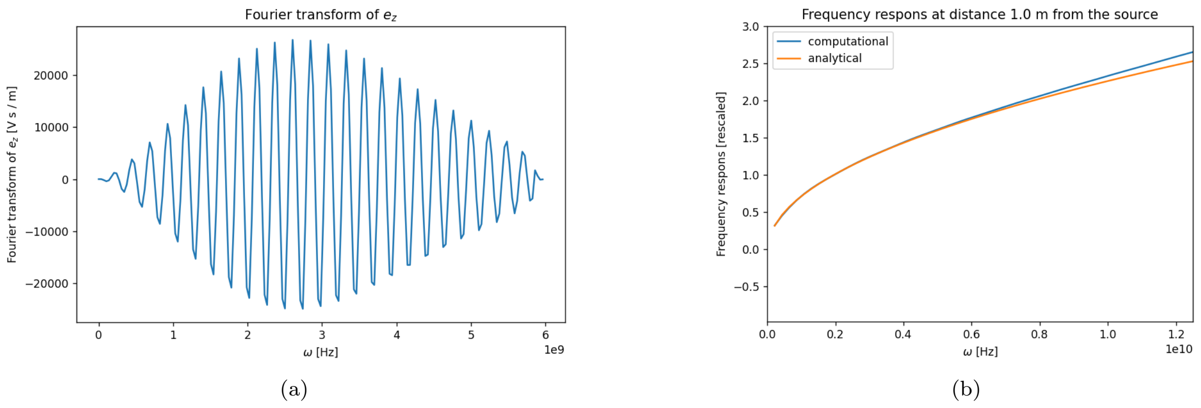


Figure 4: (a) real part of the DFT transform of e_z and (b) comparison of the analytical and computation result.

9 Results / Comparison Yee with UCHIE

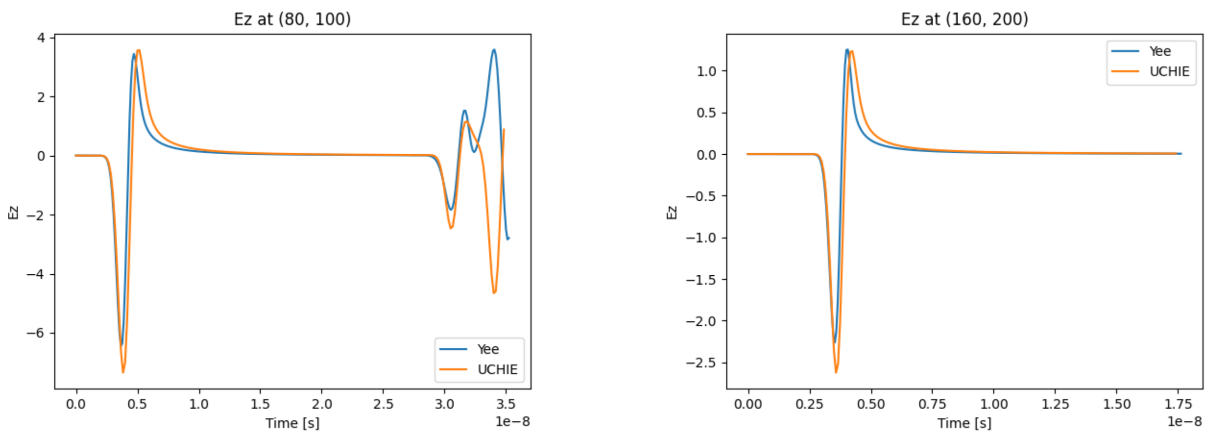
9.1 General comparison

Finally, the UCHIE and Yee implementations are compared in terms of computation times and the results for the fields in case of the presence of a vacuum and a MIS structure. In all cases considered the source is a modulated Gaussian. It should be mentioned that apart from the results given here, the implementations provide animations showing the progression of the wave front throughout the simulation and can simulate other structures, e.g. dielectrics and the microstrip given in figure 2.28 of the syllabus. The results for the vacuum situation are given in figure 5, where both implementations are compared in the case of 200 cells (figure 5a) and 400 cells (figure 5b) in the x- and y-direction. It should be noted the number of iterations of the UCHIE was chosen such that the total simulation time of both UCHIE and Yee were equal. UCHIE and Yee yield similar results for the electric field, though the amplitude of the field calculated by UCHIE is slightly larger than the field calculated by Yee. The differences at the end of the simulation in figure 5a are of no concern as they are the result of the PBC that were implemented. An overview of the time it takes for the whole simulation, all the iterations and the the inversion of the matrix in the UCHIE are given in table 1. This table indicates that both methods are relatively fast, with Yee being the faster out of the two. It can be seen that for both methods most of the total time is spent on the iterations required to update the fields, wheres the time required for the matrix inversion in case of UCHIE is negligible comparable to the total time. This would change for higher values of M.

System	Yee M = 200	UCHIE M = 200	Yee M = 400	UCHIE M = 400	Yee M = 800	UCHIE M = 800
Iterations[s]	0.99	1.88	6.49	9.42	23.06	80.60
Matrix inversion [s]	/	0.01	/	0.06	/	0.12
Total time [s]	1.39	2.17	6.89	9.77	23.52	80.60

Table 1: An overview of the time it takes Yee and UCHIE to do all iterations as well as the matrix inversion and the total time for both methods for several grids of $M \times M$ cells in the case of a vacuum. All simulation were performed using 300 iterations for Yee and a number of iterations for UCHIE such that the total simulation times were equal

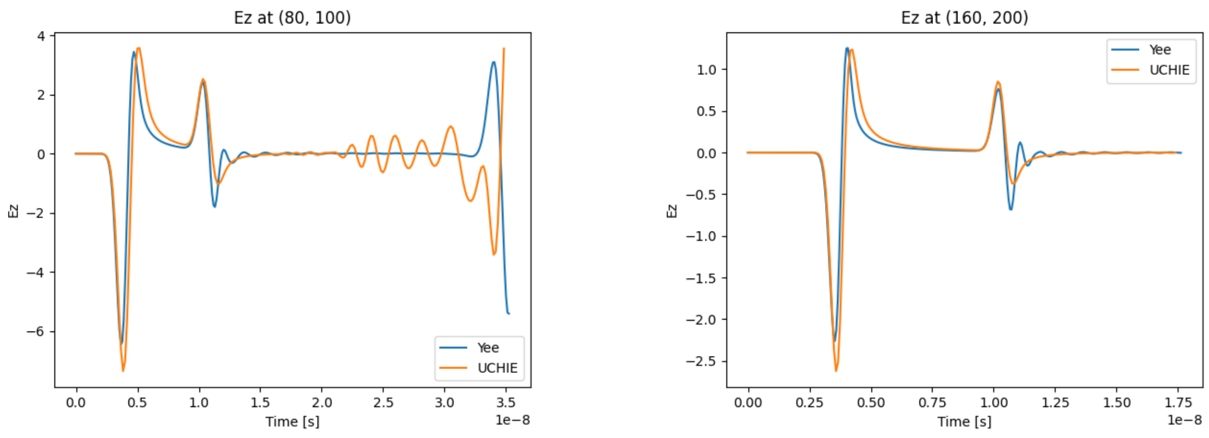
The same analysis can now be performed in the case of a MIS structure instead of the vacuum. The analysis of the time it takes for the iterations and matrix inversion as well as the total simulation time is very similar to the vacuum case and will not be included. The observation point lies between the source and the first material boundary. This implies that as opposed to the vacuum case, there will be reflections presents for the field due to the presence of these



(a) (b)

Figure 5: Comparison of the results of the Yee and UCHIE implementations for the z-component of the electric field from a modulated gaussian source in a vacuum. (a) shows the results in case 200 cells are used in both the x- and y- direction, (b) the results in the case 400 cells are used in both directions. Note that the coordinates of the observation point are scaled with a factor 2 as to compensate for the finer discretization.

different material boundaries on the grid. A comparison of the results for the Yee and UCHIE implementation are given in figure 6 with figure 6a giving the results for a grid of dimensions 200×200 and figure 6b giving the results for a grid of dimensions 400×400. It can be seen that for both grids the first 2 peaks are identical to the peaks



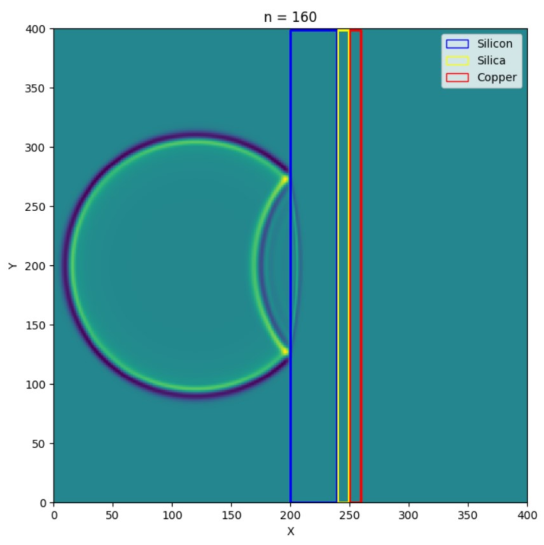
(a) (b)

Figure 6: Comparison of the results of the Yee and UCHIE implementations for the z-component of the electric field from a modulated gaussian source for the MIS structure. (a) shows the results in case 200 cells are used in both the x- and y- direction, (b) the results in the case 400 cells are used in both directions. Note that the coordinates of the observation point are scaled with a factor 2 as to compensate for the finer discretization.

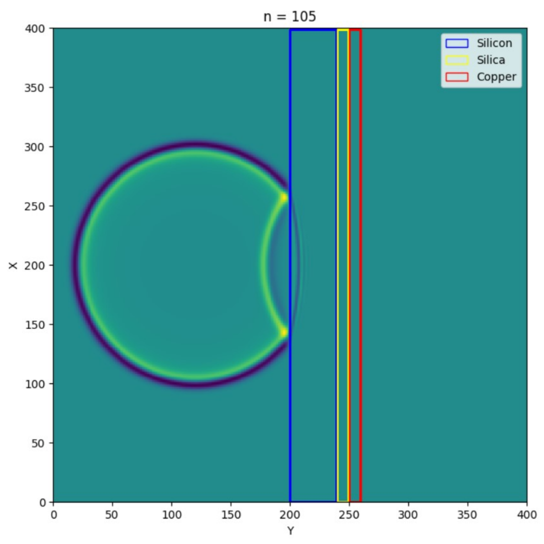
observed in the case of a vacuum. As was the case in figure 5a, figure 6a shows oscillatory behaviour at the end of the simulation. Once again these fields are a result of the PBC such that they are of no concern. The main difference between figures 6 and 5 is the presence of 2 new peaks around 10 ns into the simulation of the MIS structure. There is a slight difference between the electric field computed by UCHIE and Yee as Yee contains some oscillations whereas UCHIE does not. As expected the z-component of the electric field experiences a phase change of 180°, as can be seen in figure 6. As mentioned before, these wavefronts can be visualized in animations. Two snapshot of these animations are given figure 7, figure 7a visualizing the results for Yee and figure 7b visualizing the results for UCHIE. These figures show the reflection of the wavefront at the silicon boundary resulting in the 2 extra peaks observed in figure 6. It can also be seen that the wave travels more slowly in Silicon, as expected by the formula

$$c' = \frac{c}{\sqrt{\epsilon_r \mu_r}} \tag{33}$$

In the MIS structure copper was included, which can be regarded as a (nearly) PEC. This can be validated by



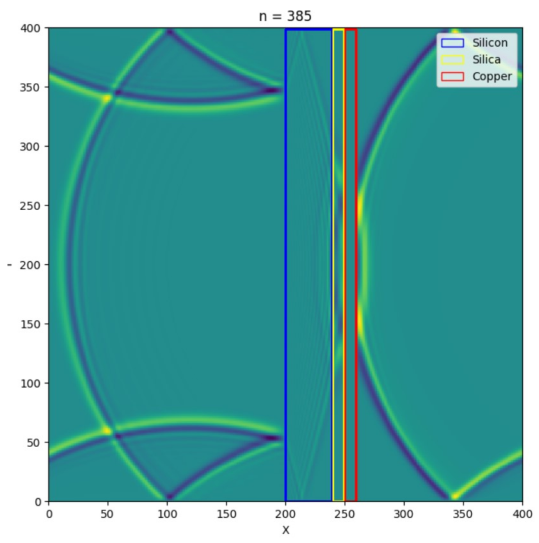
(a)



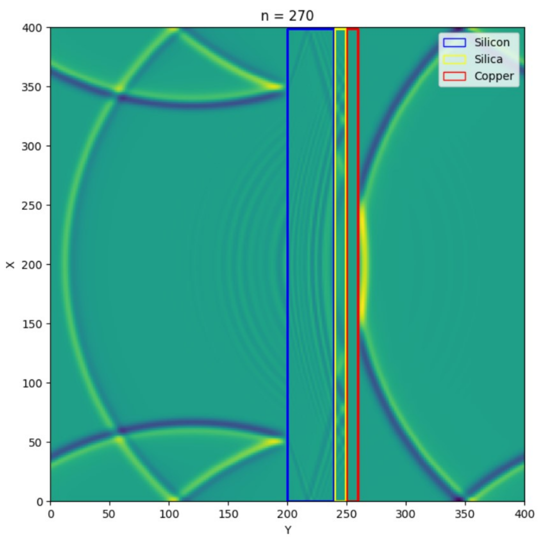
(b)

Figure 7: Snapshot of a visualization of the results of the Yee and UCHIE implementations for the z-component of the electric field from a modulated gaussian source for the MIS structure on a 400 × 400 grid. (a) shows the results for Yee (b) shows the results for UCHIE

taking a snapshot of the visualization after the wavefront reaches the copper boundary. Such snapshots are given in figure 8, which shows that for both Yee (figure 8a) and UCHIE (figure 8b) the z-component of the electric field does not propagate into the copper region, as desired.



(a)

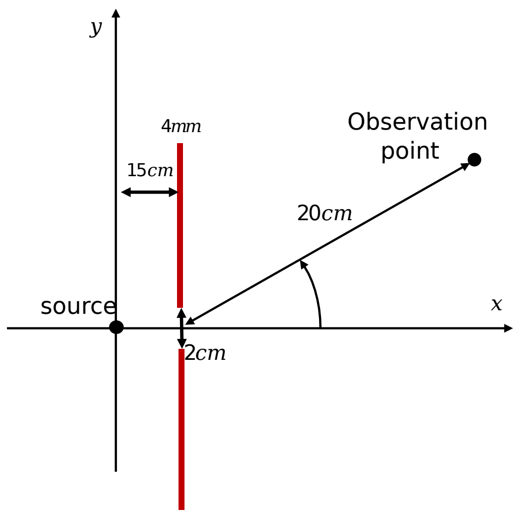


(b)

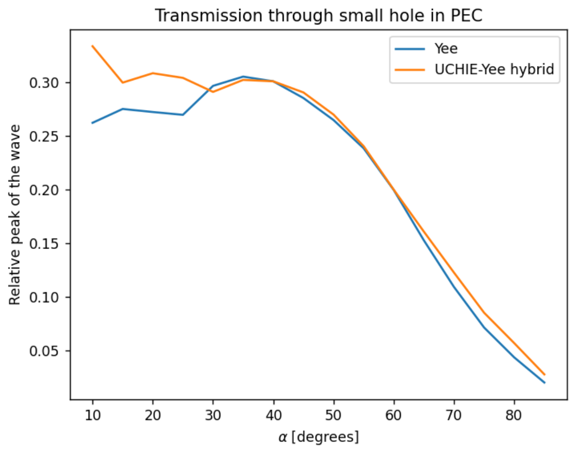
Figure 8: Snapshot of a visualization of the results of the Yee and UCHIE implementations for the z-component of the electric field from a modulated gaussian source for the MIS structure on a 400 × 400 grid at a time after reflection of the wavefront at the copper boundary. (a) shows the results for Yee (b) shows the results for UCHIE

9.2 Transmission through hole in PEC

Finally, the result of a wave impinging on a PEC with a small hole is investigated. The peak of the wave travelling through a point at a certain distance and angle from the hole is compared with the peak of the wave from the source. The result is given in figure 9. As expected, the transmission decreases for more shear incidence. The same calculation for Yee and UCHIE-Yee hybrid code are compared. They give approximately the same results, apart from low angles. This may be because at low angles, thus close to the Yee-UCHIE interface, interface effects trouble the results.



(a)



(b)

Figure 9: Transmission through a hole in a PEC in function of the angle α. (a) shows the setup, (b) the result

10 Division of the work

Sander: Derivation of the UCHIE update equations, derivation of the Yee update equations, derivation of the UCHIE-Yee hybridisation scheme equations. Writing of code files UCHIE_homogeneous, UCHIE_double, hybrid_separate, Yee_zelf, functions, and material_properties. Visualization with material properties. FFT and comparison with hankel function. Comparison different methods (Yee, UCHIE, UCHIE-Yee and UCHIE-double). Overleaf: section 1,2,3,4,5,7,8,9.2

Bjarne: Implementation of matrix inversion, writing of file inversions. FFT and comparison with hankel function. Comparison of Yee and UCHIE. Overleaf: section 6, 9.1. Explanation at the end of section 7