Cgmx Reference Manual: An Overture Solver for Maxwell's Equations on Composite Grids

William D. Henshaw
Department of Mathematical Sciences,
Rensselaer Polytechnic Institute,
Troy, NY, USA, 12180. June 26, 2021

Abstract:
Cgmx is a program that can be used to solve the time-dependent Maxwell's equations of electromagnetics on composite overlapping grids in two and three space dimensions. This document is a companion to the Cgmx User Guide. This reference guide provides additional details about the equations and approximations used along with additional examples and numerical results.

Contents

1 Introduction

Cgmx is a program that can be used to solve Maxwell's equations of electromagnetics on composite overlapping grids [**?**]. It is built upon the **Overture** framework [**?**],[**?**],[**?**].

For an introduction to Cgmx and its usage see the Cgmx User Guide [**?**]

More information about **Overture** can be found on the **Overture** home page, `http://www.llnl.gov/casc/-Overture`.

2 Method NFDTD

3 Method Yee

4 Accuracy and convergence examples

In this section we provide some examples that demonstrate the accuracy of the methods.

4.1 A comparison of far field boundary conditions for scattering from a PEC cylinder

In this example we compare the accuracy of different far-field boundary conditions for the scattering of a plane wave from a PEC cylinder.
(1) Generate the grid using the command file `cicArg.cmd` in `Overture/sampleGrids`:

```
ogen -noplot cicArg -order=4 -interp=e -factor=4
```

(2) Run cgmx with the Engquist-Majda BC `abcEM2` and the PML BC `abcPML`:

```
cgmx -noplot cic.planeWaveBC -g=cice4.order4.hdf -diss=.5 -ic=zero -rbc=abcEM2 ...
     -tp=1. -tf=100. -show="cylEM2.show"
cgmx -noplot cic.planeWaveBC -g=cice4.order4.hdf -diss=.5 -ic=zero -rbc=abcPML ...
     -pmlWidth=21 -pmlStrength=50. -tp=1. -tf=100. -show="cylPML.show"
```

(3) To plot errors use plotStuff to display the show file. Plot the "sequence" with the errors and then save to a matlab file.

```
plot sequence:errors
  add Ey_error
  add Hz_error
  save results to a matlab file
    scatCyl.m
```

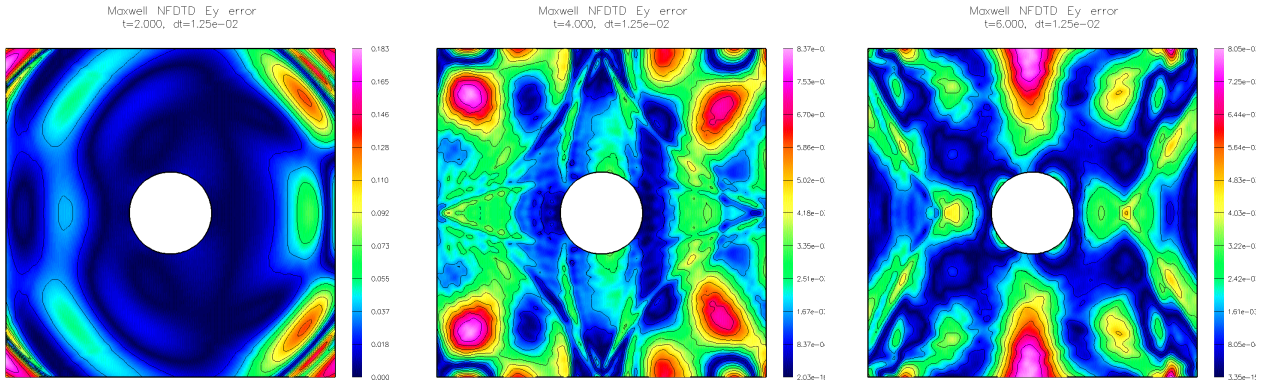

Figure 1: `abcEM2` : scattering of a plane wave by a PEC cylinder. Errors in E_y at $t = 2.$, 4. and 6.0.

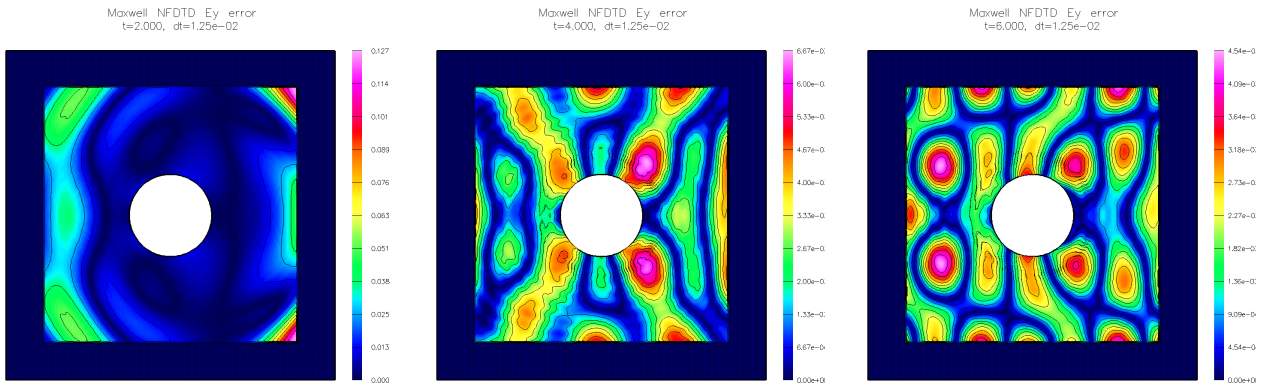

Figure 2: `abcPML` : scattering of a plane wave by a PEC cylinder. Errors in E_y $t = 2$, 4 and 6.

Figures 1-2 shows the errors in E_y for the EM2 and PML farfield conditions. Figure 3 shows the maximum errors over time for the two conditions.
Notes:

1. The initial conditions are taken as zero with inhomogeneous boundary conditions on the cylinder boundary.

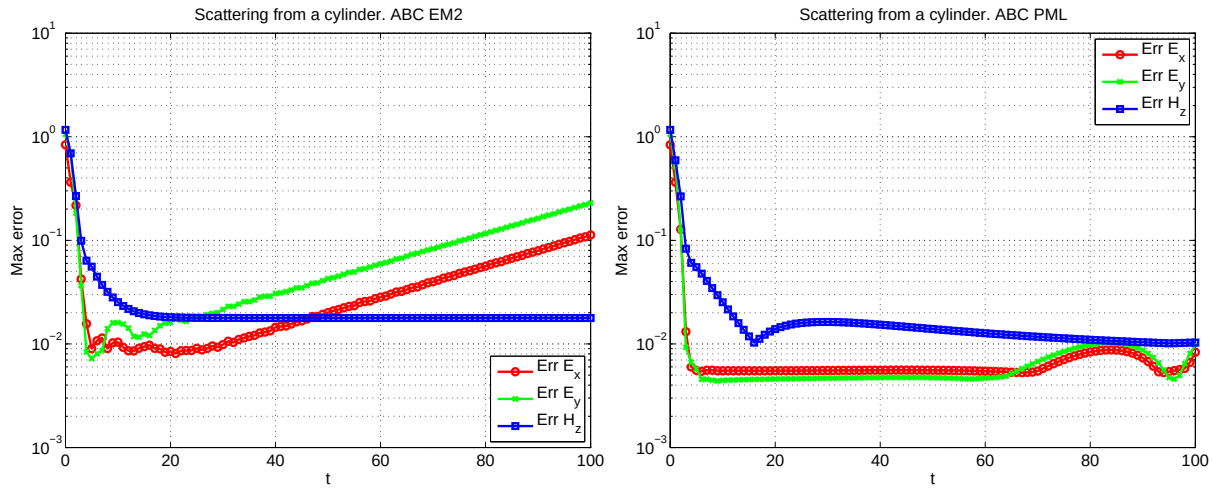

Figure 3: Scattering of a plane wave by a PEC cylinder. Errors over time for `abcEM2` (left) and `abcPML` (right).

2. The errors are initially large as the exact solution develops and propagates outward from the cylinder. These initial large errors can be seen in the plots at $t = 2$.

3. The EM2 errors show a fast growth for long times, except for the error in H_z. The H_z field seems to propagate more strongly in the x- and y- directions (rather than along diagonals) and thus will be treated more accurately by the far field conditions. The PML errors do not grow as fast but eventually seem to start to grow.

5 Boundary Conditions

5.1 Perfect electrical conductor boundary condition

The perfect electrical conductor boundary condition sets the tangential components of the electric field to zero

$$\boldsymbol{\tau}_m \cdot \mathbf{E} = 0, \qquad \text{for } \mathbf{x} \in \partial\Omega_{\text{pec}}. \tag{1}$$

Here $\boldsymbol{\tau}_m$, $m = 1, 2$, denote the tangent vectors to the boundary surface, $\partial\Omega_{\text{pec}}$.

5.2 Engquist-Majda absorbing boundary conditions

The boundary condition `abcEM2` uses the Engquist-Majda absorbing boundary condition (defined here of a boundary $x =$ constant),

$$\partial_t \partial_x u = \alpha \partial_x^2 u + \beta(\partial_y^2 + \partial_z^2)u \tag{2}$$

With $\alpha = c$ and $\beta = \frac{1}{2}c$, this gives a *second-order accurate* approximation to a pseudo-differential operator that absorbs outgoing traveling waves. Here u is any field which satisfies the second-order wave equation.

Notes on the Engquist-Majda absorbing boundary conditions can be found in Appendix A.

5.3 Perfectly matched layer boundary condition

The boundary condition `abcPML` imposes a perfectly matched layer boundary condition. With this boundary condition, auxillary equations are solved over a layer (of some number of specified grid points) next to the boundary. The PML equations we solve are those suggested in Hagstrom [**?**] and given by (defined here of a boundary $x =$ constant), (*check me*)

$$u_{tt} = c^2\Big(\Delta u - \partial_x v - w\Big), \tag{3}$$

$$v_t = \sigma(x)(-v + \partial_x u), \tag{4}$$

$$w_t = \sigma(x)(-w - \partial_x v + \partial_x^2 u). \tag{5}$$

Here u is any field which satisfies the second-order wave equation and v and w are auxillary variables that only live in the layer domain. The PML damping function $\sigma_1(\xi)$ is given by (Note: scale by c to make dimensionally correct, 2020/03/31)

$$\sigma(\xi) = c\, a\, \xi^p \tag{6}$$

where a is the strength, p is the power and where ξ varies from 0 to 1 through the layer.

A second-order accurate approximation to equations (3)-(5) is

$$\frac{u_{\mathbf{i}}^{n+1} - 2u_{\mathbf{i}}^n + u_{\mathbf{i}}^{n-1}}{\Delta t} = c^2\Big(\Delta_{2h} u_{\mathbf{i}} - D_{0x} v_{\mathbf{i}} - w_{\mathbf{i}}\Big), \tag{7}$$

$$\frac{v_{\mathbf{i}}^{n+1} - v_{\mathbf{i}}}{\Delta t} = \frac{3}{2} f_{\mathbf{i}}^n - \frac{1}{2} f_{\mathbf{i}}^{n-1}, \tag{8}$$

$$f_{\mathbf{i}}^n = \sigma(-v_{\mathbf{i}}^n + D_{0x} u_{\mathbf{i}}^n), \tag{9}$$

$$\frac{w_{\mathbf{i}}^{n+1} - w_{\mathbf{i}}}{\Delta t} = \frac{3}{2} g_{\mathbf{i}}^n - \frac{1}{2} g_{\mathbf{i}}^{n-1}, \tag{10}$$

$$g_{\mathbf{i}}^n = \sigma(-w_{\mathbf{i}}^n - D_{0x} v_{\mathbf{i}}^n + D_{+x} D_{-x} u_{\mathbf{i}}^n). \tag{11}$$

Here we have used a centered in time approximation for the first equation and Adams-Bashforth for the v and w equations.

Aside: Since $\sigma(\mathbf{x})$ is large we could use an integrating factor for the v and w equations and instead solve

$$\partial_t(e^{\sigma t} v) = \sigma e^{\sigma t} \partial_x u, \tag{12}$$

$$\partial_t(e^{\sigma t} v) = \sigma e^{\sigma t}(\partial_x v + \partial_x^2 u). \tag{13}$$

A fourth-order accurate approximation to (3)-(5) can be determined using the relations

$$\frac{u_{\mathbf{i}}^{n+1}-2u_{\mathbf{i}}^{n}+u_{\mathbf{i}}^{n-1}}{\Delta t^2} = \partial_t^2 u + \frac{\Delta t^2}{12}\partial_t^4 u + \mathcal{O}(\Delta t^4) \tag{14}$$

$$= c^2\Big(\Delta u - \partial_x v - w\Big) + \frac{\Delta t^2}{12}\Big(\Delta\partial_t^2 u - \partial_t^2\partial_x v - \partial_t^2 w\Big), \tag{15}$$

$$= c^2\Big(\Delta u - \partial_x v - w\Big) + \frac{\Delta t^2}{12}\Big(c^2\big(\Delta^2 u - \partial_x\Delta v - \Delta w\big) - \partial_t^2\partial_x v - \partial_t^2 w\Big), \tag{16}$$

$$v_{\mathbf{i}}^{n+1} = v_{\mathbf{i}}^{n} + \Delta t\partial_t v + \frac{\Delta t^2}{2}\partial_t^2 v + \frac{\Delta t^3}{3!}\partial_t^3 v + \frac{\Delta t^4}{4!}\partial_t^4 v + \mathcal{O}(\Delta t^5) \tag{17}$$

$$w_{\mathbf{i}}^{n+1} = w_{\mathbf{i}}^{n} + \Delta t\partial_t w + \frac{\Delta t^2}{2}\partial_t^2 w + \frac{\Delta t^3}{3!}\partial_t^3 w + \frac{\Delta t^4}{4!}\partial_t^4 w + \mathcal{O}(\Delta t^5) \tag{18}$$

By taking derivatives of the governing equations (3)-(5) we can approximate the various terms in the previous expansions. For example,

$$\partial_t^2 v = \sigma(x)(-\partial_t v + \partial_t\partial_x u), \tag{19}$$

$$\partial_t^3 v = \sigma(x)(-\partial_t^2 v + \partial_t^2\partial_x u), \tag{20}$$

$$= \sigma(x)\Big(-\partial_t^2 v + c^2\partial_x(\Delta u - \partial_x v - w)\Big) \tag{21}$$

$$\partial_t\partial_x v = \sigma(x)(-\partial_x v + \partial_x^2 u) + \partial_x\sigma(-v + \partial_x u), \tag{22}$$

$$\partial_t^2\partial_x v = \sigma(x)(-\partial_t\partial_x v + \partial_t\partial_x^2 u) + \partial_x\sigma(-\partial_t v + \partial_t\partial_x u), \tag{23}$$

We obtain second-order approximations for $\partial_t u$, $\partial_t\partial_x u$ and $\partial_t\partial_x^2 u$ from the follwing approximation to $\partial_t u$,

$$\partial_t u = \frac{u_{\mathbf{i}}^{n} - u_{\mathbf{i}}^{n-1}}{\Delta t} - \frac{\Delta t}{2}\partial_t^2 u + \mathcal{O}(\Delta t^2) \tag{24}$$

$$\tag{25}$$

5.3.1 PML edge regions

In the edge region outside an x and y boundary, we solve

$$u_{tt} = c^2\Big(\Delta u - \partial_x v^x - w^x - \partial_y v^y - w^y\Big), \tag{26}$$

$$v_t^x = \sigma(x)(-v^x + \partial_x u), \tag{27}$$

$$w_t^x = \sigma(x)(-w^x - \partial_x v^x + \partial_x^2 u), \tag{28}$$

$$v_t^y = \sigma(y)(-v^y + \partial_y u), \tag{29}$$

$$w_t^y = \sigma(y)(-w^y - \partial_y v^y + \partial_y^2 u). \tag{30}$$

Thus we require 4 auxillary variables in the edge region. A second-order accurate approximation can be otained in a similar manner to (7)-(11).

A fourth-order accurate approximation follows the procedure used to develop (14)-(18).

$$\frac{u_{\mathbf{i}}^{n+1}-2u_{\mathbf{i}}^{n}+u_{\mathbf{i}}^{n-1}}{\Delta t^2} = \partial_t^2 u + \frac{\Delta t^2}{12}\partial_t^4 u + \mathcal{O}(\Delta t^4) \tag{31}$$

$$= c^2\Big(\Delta u - \partial_x v^x - w^x - \partial_y v^y - w^y\Big) + \frac{\Delta t^2}{12}\Big(\Delta\partial_t^2 u - \partial_t^2\partial_x v^x - \partial_t^2 w^x - \partial_t^2\partial_y v^y - \partial_t^2 w^y\Big), \tag{32}$$

$$= c^2\Big(\Delta u - \partial_x v^x - w^x - \partial_y v^y - w^y\Big) \tag{33}$$

$$+ \frac{\Delta t^2}{12}\Big(c^2\big(\Delta^2 u - \partial_x\Delta v^x - \Delta w^x - \partial_y\Delta v^y - w^y\big) - \partial_t^2\partial_x v^x - \partial_t^2 w^x - \partial_t^2\partial_y v^y - \partial_t^2 w^y\Big), \tag{34}$$

NOTE: Currently the PML edge and corner regions are only approximated to 2nd order for the 4th-order version.

5.3.2 PML corner regions

In the corner region outside an x, y and z boundary we use 6 auxillary variables and solve

$$u_{tt} = c^2\Big(\Delta u - \partial_x v^x - w^x - \partial_y v^y - w^y - \partial_z v^z - w^z\Big), \tag{35}$$
$$v^x_t = \sigma(x)(-v^x + \partial_x u), \tag{36}$$
$$w^x_t = \sigma(x)(-w^x - \partial_x v^x + \partial_x^2 u), \tag{37}$$
$$v^y_t = \sigma(y)(-v^y + \partial_y u), \tag{38}$$
$$w^y_t = \sigma(y)(-w^y - \partial_y v^y + \partial_y^2 u), \tag{39}$$
$$v^z_t = \sigma(z)(-v^z + \partial_z u), \tag{40}$$
$$w^z_t = \sigma(z)(-w^z - \partial_z v^z + \partial_z^2 u). \tag{41}$$

6 Implementation of the interface conditions

In this section we discuss the serial and parallel implementation of the interface conditions. Suppose we have some discrete interface conditions for the non-dispersive or dispersive Maxwell equations. These equations take the generic form of N_i jump conditions

$$[\mathcal{D}_m \mathbf{E}^n_{\mathbf{i}}]_I = f^{(m)}, \qquad \mathbf{i} \in \Gamma_h, \quad m = 1, 2, \ldots, N_i \tag{42}$$

where $\mathcal{D}_m$ are difference operator such as $\mathcal{D}_m = \mathbf{n}\cdot\Delta_{2h}$, and $f^{(m)}$ are forcing terms that may arise from manufactured solutions or from polarization terms.

6.1 Fourth-order accurate interface conditions – tall cell instablity

The fourth-order accurate interface conditions are

$$\left[\mathbf{n} \times \left(c^2 \Delta_{4h} \mathbf{E}^n_{\mathbf{j}}\right)\right]_I = 0, \qquad \mathbf{j} \in \Gamma_h, \tag{43a}$$
$$\left[\mu_0^{-1} \mathbf{n} \cdot \Delta_{4h} \mathbf{E}^n_{\mathbf{j}}\right]_I = 0, \qquad \mathbf{j} \in \Gamma_h, \tag{43b}$$
$$\left[\mu_0^{-1} \mathbf{n} \times \left(\nabla_{4h} \times \mathbf{E}^n_{\mathbf{j}}\right)\right]_I = 0, \qquad \mathbf{j} \in \Gamma_h, \tag{43c}$$
$$\left[\nabla_{4h} \cdot \mathbf{E}^n_{\mathbf{j}}\right]_I = 0, \qquad \mathbf{j} \in \Gamma_h, \tag{43d}$$
$$\left[\mathbf{n} \times \left(c^4 \Delta^2_{2h} \mathbf{E}^n_{\mathbf{j}}\right)\right]_I = 0, \qquad \mathbf{j} \in \Gamma_h, \tag{43e}$$
$$\left[\mu_0^{-1} \mathbf{n} \cdot \left(c^2 \Delta^2_{2h} \mathbf{E}^n_{\mathbf{j}}\right)\right]_I = 0, \qquad \mathbf{j} \in \Gamma_h, \tag{43f}$$
$$\left[\mu_0^{-1} \mathbf{n} \times \left(c^2 \nabla_{2h} \times \Delta_{2h} \mathbf{E}^n_{\mathbf{j}}\right)\right]_I = 0, \qquad \mathbf{j} \in \Gamma_h, \tag{43g}$$
$$\left[c^2 \nabla_{2h} \cdot \left(\Delta_{2h} \mathbf{E}^n_{\mathbf{j}}\right)\right]_I = 0, \qquad \mathbf{j} \in \Gamma_h, \tag{43h}$$

where ∇_{4h} and Δ_{4h} denote fourth-order accurate approximations (with a five-point stencil in each coordinate direction) to the gradient and Laplace operators, respectively.

Direct solution of these equations can be unstable when the grid spacing normal to the interface is large compared to the tangential grid spacing – we call this the tall-cell instability. The problem is related the the cross-derivative terms such as those that appear in Δ^2,

$$\Delta^2 u = \partial_x^4 u + 2\partial_x^2 \partial_y^2 u + \partial_y^4 u \tag{44}$$

To over-come the instability we start by filling the first-ghost line values using the second-order accurate interface conditions to obtain predicted values $u^p_{\mathbf{i}}$. These predicted values are used to evaluate the cross-derivative terms. For example,

$$\Delta^2_{2h} u_{\mathbf{i}} = (D_{+x} D_{-x})^2 u_{\mathbf{i}} + 2(D_{+x} D_{-x})(D_{+y} D_{-y}) u^p_{\mathbf{i}} + (D_{+y} D_{-y}) u_{\mathbf{i}} \tag{45}$$

The cross-terms evaluated with $u^p_{\mathbf{i}}$ are then moved to the right-hand-side of the discrete jump conditions (43).

Note. It is important that the cross-derivative terms are entirely evaluated with the predicted values, i.e., the operators should not be split up so that the terms involving the ghost values are kept as unknowns. Otherwise smoothness is lost and the accuracy of the result is degraded.

6.2 Stencil-based update

This section provides notes to forming the stencil update for the ghost points. These notes are a work in progress. See the test routine `research/interface/wave.m`.

After solving the interface conditions, the interface values will be projected, while a given ghost point value will be given in terms of a stencil of values from either side of the interface. We can write the general form as a stencil evaluation for interface and ghost values $u^g_{\mathbf{i}}$ in the form

$$\bar{u}^{(1)}_{\mathbf{i}} = \bar{u}^{(2)}_{\mathbf{i}} = \alpha u^{(1)}_{\mathbf{i}} + \beta u^{(2)}_{\mathbf{i}} \qquad \text{(interface projection)} \tag{46}$$

$$u_{\mathbf{i}-\mathbf{e}_g} = \sum_{\mathbf{j}} a^{(1)}_{\mathbf{i},\mathbf{j}} u^{(1)}_{\mathbf{j}} \;+\; \sum_{\mathbf{j}} a^{(2)}_{\mathbf{i},\mathbf{j}} u^{(2)}_{\mathbf{j}} + \sum_{\mathbf{j}\in\Gamma_h} \sum_{m=1}^{N_i} b^{(m)}_{\mathbf{i},\mathbf{j}} f^{(m)}_{\mathbf{j}} \qquad \text{(ghost value updates)} \tag{47}$$

where $u_{\mathbf{j}}^{(s)}$ denotes values of a component of $\mathbf{E}$ on side $s = 1, 2$ and where the sum is over boundary and interior points. The forcing terms on the interface also contribute to the formulae.

The coefficients $a_{\mathbf{i},\mathbf{j}}^{(s)}$ can be computed in a pre-processing step, by the discrete-delta approach, for example.

The parallel implementation is straight-forward. On each side of the interface first evaluate the sum

$$g_{\mathbf{i}}^{(s)} = \sum_{\mathbf{j}} a_{\mathbf{i},\mathbf{j}}^{(s)} u_{\mathbf{j}}^{(s)} \tag{48}$$

These values are then communicated to the other side.

Notes on evaluating the stencil coefficients: Consider the second-order accurate (SOA) interface conditions. The basic steps to evaluate the residuals in the jump conditions are are

1. Project the interface values (requires values from both sides).
2. Extrapolate ghost values (used in any cross-terms to avoid tangential coupling).
3. Evaluate jump conditions.

The stencil coefficients $a_{\mathbf{i},\mathbf{j}}^{(s)}$ should reflect all these steps (?) .

Here is the residual function $\mathbf{R}(\mathbf{u}) = A\mathbf{u} - \mathbf{b}$,

$$\mathbf{R} = \text{function evalJump}(u_{\mathbf{i}}^{(1)}, u_{\mathbf{i}}^{(2)}), \tag{49}$$

$$u_{\mathbf{i}}^{(s)} \leftarrow \text{Project}(u_{\mathbf{i}}^{(1)}, u_{\mathbf{i}}^{(2)}), \qquad \mathbf{i} \in \Gamma_h, \tag{50}$$

$$u_{\mathbf{i}-1}^{(s)} = 3u_{\mathbf{i}}^{(I,s)} - 3u_{\mathbf{i}+1}^{(s)} + u_{\mathbf{i}+2}^{(s)}, \tag{51}$$

$$R_1 = \left[\Delta_{2h} u_{\mathbf{i}}^{(s)}\right]_I \tag{52}$$

$$\ldots \tag{53}$$

$$\text{end} \tag{54}$$

6.3 Residual based update

This section outlines an approach that is based on evaluating the residual in the jump conditions.

In general the interface conditions such as (55) take the form of jump conditions involving difference approximations of the components E_α of $\mathbf{E} \in \mathbb{R}^{n_d}$,

$$\left[\sum_{\mathbf{j}\in\text{ghost}} \sum_{\alpha=1}^{n_d} C_{\alpha,\mathbf{i},\mathbf{j}}^{(m)} E_{\alpha,\mathbf{i}+\mathbf{j}} - F_{\mathbf{i}}^{(m)}\right]_I = 0, \qquad m = 1, 2, \ldots, M, \qquad \mathbf{i} \in \Gamma_h, \tag{55}$$

where $F_{\mathbf{i}}^{(m)}$ consists of known values (boundary and interior values, as well as any forcing), and where the total number of conditions M is equal to the total number of ghost point values

$$M = 2\, N_g\, n_d, \tag{56}$$

where N_g is the number of ghost points on one side the the interface (e.g. $N_g = 1$ for second-order and 2 for fourth-order) and n_d is the number of space dimensions.

Suppose we have a function that can compute the residuals in the discrete interface conditions

$$R_{\mathbf{i}}^{(m)} = \left[\sum_{\alpha,\mathbf{j}} C_{\alpha,\mathbf{i},\mathbf{j}}^{(m)} E_{\alpha,\mathbf{i}} - F_{\mathbf{i}}^{(m)}\right]_I = \left\{\sum_{\alpha,\mathbf{j}} C_{\alpha,\mathbf{i},\mathbf{j}}^{(m)} E_{\alpha,\mathbf{i}} - F_{\mathbf{i}}^{(m)}\right\}_{\text{Side2}} - \left\{\sum_{\alpha,\mathbf{j}} C_{\alpha,\mathbf{i},\mathbf{j}}^{(m)} E_{\alpha,\mathbf{i}} - F_{\mathbf{i}}^{(m)}\right\}_{\text{Side1}} \tag{57}$$

Let's write our condition (57) as (drop the $\mathbf{i}$ to simplify the notation) a difference between values from the two sides of the interface

$$R^{(m)} = G_2^{(m)} - G_1^{(m)} \tag{58}$$

Let $\mathbf{u}_s^g$ denote the vector of ghost point values of $\mathbf{E}$ on side $s = 1, 2$ of the interface at point $\mathbf{i}$. Let $\mathbf{u}^g = [\mathbf{u}_1^g\,;\,\mathbf{u}_2^g]$ denote the combined vector of ghost points.

$G_s^{(m)}$ depends linearly on $\mathbf{u}_s^g$ and we can expose this dependence as

$$G_s^{(m)} = \mathbf{a}_{m,s}^T \mathbf{u}_s^g + g_s^{(m)} \tag{59}$$

where $g_s^{(m)}$ depends on interior and interface values but not ghost values. The vector $\mathbf{a}_{m,s}^T$ thus holds the coefficients in the stencil that multiply the ghost points. We write the total residual in equation m as

$$R^{(m)} = \mathbf{a}_m^T \mathbf{u}^g + g^{(m)} \tag{60}$$

where $\mathbf{a}_m = [-\mathbf{a}_{m,1}\,;\,\mathbf{a}_{m,2}]$ and $g^{(m)} = g_2^{(m)} - g_1^{(m)}$ (*check me*).

If we know the stencil coefficients $\mathbf{a}_m$ then to solve the interface conditions for the ghost point values we can solve the linear system

$$A\mathbf{u}^g + \begin{bmatrix} g^{(1)} \\ \vdots \\ g^{(M)} \end{bmatrix} = \mathbf{0} \tag{61}$$

where $A \in \mathbb{R}^{M \times M}$ is

$$A \stackrel{\text{def}}{=} \begin{bmatrix} \mathbf{a}_1^T \\ \vdots \\ \mathbf{a}_M^T \end{bmatrix} \tag{62}$$

Using (60) to solve for g_m, we can rewrite (61) as

$$A\mathbf{u}^g + \begin{bmatrix} R^{(1)}[p] \\ \vdots \\ R^{(M)}[p] \end{bmatrix} - A\mathbf{u}^g[p] = \mathbf{0} \tag{63}$$

where $R^{(m)}[p]$ denotes the residual computed using some predicted values $\mathbf{u}^g[p]$ for the ghost points. This leads to the formula for the ghost point values,

$$\mathbf{u}^g = A^{-1} \begin{bmatrix} R^{(1)}[p] \\ \vdots \\ R^{(M)}[p] \end{bmatrix} + \mathbf{u}^g[p] \tag{64}$$

Formula (64) thus only requires knowledge of the coefficients in A and a function to compute the residuals in the jump conditions.

6.4 Parallel implementation.

In parallel, the points on different sides of the interface may lie on different processors. A naive implementation would communicate all data for values of $\mathbf{E}_{\mathbf{i}}^n$, $\mathbf{P}_{m,\mathbf{i}}^n$, $\mathbf{P}_{m,\mathbf{i}}^{n-1}$, etc., near the interface from one side to the other and then apply the serial algorithm. This could be a lot of data if there are many polarization vectors.

We want to devise an approach to minimize the communication between the two sides. To solve (64) in parallel we can first compute the portion of the residual $R^{(M)}[p]$ that lives on either side of the interface. Since

$$R^{(m)}[p] = G_2^{(m)}[p] - G_1^{(m)}[p] \tag{65}$$

we first compute $\mathbf{w}_s \in \mathbb{R}^M$ on side s defined by

$$\mathbf{w}_s = A^{-1} \begin{bmatrix} G_s^{(1)}[p] \\ \vdots \\ G_s^{(M)}[p] \end{bmatrix}, \qquad \text{for sides } s = 1, 2, \tag{66}$$

$$\stackrel{\text{def}}{=} \begin{bmatrix} \mathbf{w}_{s1} \\ \mathbf{w}_{s2} \end{bmatrix} \tag{67}$$

Whence the required ghost values satisfy

$$\mathbf{u}^g = \mathbf{w}_2 - \mathbf{w}_1 + \mathbf{u}^g[p], \tag{68}$$

or

$$\begin{bmatrix} \mathbf{u}_1^g \\ \mathbf{u}_2^g \end{bmatrix} = \begin{bmatrix} \mathbf{w}_{21} \\ \mathbf{w}_{22} \end{bmatrix} - \begin{bmatrix} \mathbf{w}_{11} \\ \mathbf{w}_{12} \end{bmatrix} + \begin{bmatrix} \mathbf{u}_1^g[p] \\ \mathbf{u}_2^g[p] \end{bmatrix} \tag{69}$$

Thus to compute the $N = n_d N_g$ ghost values $\mathbf{u}_1^g$ on side 1 we just need the N values $\mathbf{w}_{21}$ from side 2. Similiarly to compute the N ghost values $\mathbf{u}_s^g$ on side 2 we just need the N values $\mathbf{w}_{12}$ from side 1. This approach appears to result in the minimal amount of communication between the two sides. Note that one key step is to multiply the data by A^{-1} before communication, which reduces the amount data needed by one-half.

6.5 Parallel implementation - Fourth-order accurate interface conditions with second-order accurate predictor

The fourth-order accurate interface conditions consist of two main stages:

Stage I Apply the second-order accurate interface conditions to get predicted values for the first ghost line.

Stage II Apply the fourth-order accurate interface conditions to fill in two ghost lines using the predicted values in any cross-derivative terms.

In parallel, we can also proceed in two stages. This requires communication during stage 1 and also communication during stage 2.

Question: Can we organize the computations so that there is a single commuication stage?

In Stage I we have a residual equation

$$R_{2h}^{(m)} = \mathbf{a}_{2h,m}^T \mathbf{u}_{2h}^g + g_{2h}^{(m)} \tag{70}$$

with solution

$$\mathbf{u}_{2h}^g = A_{2h}^{-1} \begin{bmatrix} R_{2h}^{(1)}[p] \\ \vdots \\ R_{2h}^{(M)}[p] \end{bmatrix} + \mathbf{u}^g[p] \tag{71}$$

In Stage II we have a residual equation involving the fourth-order accurate ghost values $\mathbf{u}_{4h}^g$ and the second-order accurate ones $\mathbf{u}_{2h}^g$

$$R_{4h}^{(m)} = \mathbf{a}_{4h,m}^T \mathbf{u}_{4h}^g + \mathbf{b}_{4h,m}^T \mathbf{u}_{2h}^g + g_{4h}^{(m)}, \tag{72}$$

which leads to the system of equations

$$A_{4h}\mathbf{u}_{4h}^g + B_{2h}\mathbf{u}_{2h}^g + \mathbf{g}_{4h} = 0 \tag{73}$$

If we know $\mathbf{u}_{2h}^g$ then we can solve this for $\mathbf{u}_{4h}^g$ following the usual parallel algorithm as follows. Given predicted values $\mathbf{u}^g[p]$ we evaluate $\mathbf{g}_{4h}$ as

$$\mathbf{g}_{4h} = \mathbf{R}_{4h}[p] - A_{4h}\mathbf{u}_{4h}^g[p] - B_{4h}\mathbf{u}_{2h}^g \tag{74}$$

and then solve

$$\begin{aligned} A_{4h}\mathbf{u}_{4h}^g &= -B_{2h}\mathbf{u}_{2h}^g - \mathbf{g}_{4h} && (75)\\ &= -B_{2h}\mathbf{u}_{2h}^g - \mathbf{R}_{4h}[p] + A_{4h}\mathbf{u}_{4h}^g[p] + B_{4h}\mathbf{u}_{2h}^g && (76)\\ &= -\mathbf{R}_{4h}[p] + A_{4h}\mathbf{u}_{4h}^g[p] && (77) \end{aligned}$$

Whence

$$\mathbf{u}_{4h}^g = -A_{4h}^{-1}\mathbf{R}_{4h}[p] + \mathbf{u}_{4h}^g[p] \tag{78}$$

$\mathbf{R}_{4h}[p]$ can be written as a difference of quantities on the two sides,

$$\mathbf{R}_{4h}[p] = \mathbf{G}_2 - \mathbf{G}_1 \tag{79}$$

To evaluate $\mathbf{G}_s$ on each side we need the second-order accurate (SOA) values $B_{2h}\mathbf{u}_{2h}^g$,

$$B_{2h}\mathbf{u}_{2h}^g = B_{2h}\mathbf{w}_2 - B_{2h}\mathbf{w}_1 + B_{2h}\mathbf{h}^g[p] \tag{80}$$

old
Otherwise, to solve for $\mathbf{u}_{4h}^g$ we actually just need $B_{2h}\mathbf{u}_{2h}^g$:

$$B_{2h}\mathbf{u}_{2h}^g = B_{2h}\mathbf{w}_2 - B_{2h}\mathbf{w}_1 + B_{2h}\mathbf{u}^g[p] \tag{81}$$

old

Given predicted values at two ghost lines, $\mathbf{u}^g[p1]$, we evaluate

$$\mathbf{g}_{4h}[p1] = \mathbf{R}_{4h}[p1] - A_{4h}\mathbf{u}^g[p1] - B_{2h}\mathbf{u}_{2h}^g[p1] \tag{82}$$

which leads to

$$A_{4h}\mathbf{u}_{4h}^g + B_{2h}\mathbf{u}_{2h}^g + \mathbf{R}_{4h}[p1] - A_{4h}\mathbf{u}^g[p1] - B_{2h}\mathbf{u}^g[p1] = 0 \tag{83}$$

$$\tag{84}$$

where **check me**

$$\mathbf{u}_{2h}^g[p1] = A_{2h}^{-1}\mathbf{R}_{2h}[p1] + \mathbf{u}^g[p1] \tag{85}$$

old
with solution (using new predicted values $\mathbf{u}^g[S1]$)

$$\mathbf{u}_{4h}^g = A_{4h}^{-1}\begin{bmatrix} R_{4h}^{(1)}[p] \\ \vdots \\ R_{4h}^{(M)}[p] \end{bmatrix} + \mathbf{u}^g[S1], \tag{86}$$

$$\tag{87}$$

Now

$$\mathbf{b}_{4h,m}^T\mathbf{u}_{2h}^g = \mathbf{b}_{4h,m}^T A_{2h}^{-1}\begin{bmatrix} R_{2h}^{(1)}[p] \\ \vdots \\ R_{2h}^{(M)}[p] \end{bmatrix} + \mathbf{b}_{4h,m}^T\mathbf{u}^g[p], \tag{88}$$

$$R_{2h}^{(m)}[p] = G_{2h,2}^{(m)}[p] - G_{2h,1}^{(m)}[p] \tag{89}$$

Suppose we compute on each side $s = 1, 2$

$$\mathbf{w}_s = \mathbf{b}_{4h,m}^T A_{2h}^{-1}\mathbf{G}_{2h,s}[p] \tag{90}$$

7 Computing the magnetic field and intensity from a time harmonic electric field

For time harmonic solutions we can easily compute the magnetic field given the electric field and its time derivative (or the electric field at two times). Thus we can compute the intensity given the electric field without having to explicitly solve Maxwell's equations for the magnetic field.

Consider a (complex valued) time harmonic solution to Maxwell's equations,

$$\widetilde{\mathbf{E}}(\mathbf{x},t) = e^{-i\omega t}\widehat{\mathbf{E}}(\mathbf{x}),$$
$$\widetilde{\mathbf{H}}(\mathbf{x},t) = e^{-i\omega t}\widehat{\mathbf{H}}(\mathbf{x}),$$

where the real and imaginary parts of $\widehat{\mathbf{E}}$ and $\widehat{\mathbf{H}}$ are denoted as

$$\widehat{\mathbf{E}}(\mathbf{x}) = \widehat{\mathbf{E}}_r(\mathbf{x}) + i\widehat{\mathbf{E}}_i(\mathbf{x}),$$
$$\widehat{\mathbf{H}}(\mathbf{x}) = \widehat{\mathbf{H}}_r(\mathbf{x}) + i\widehat{\mathbf{H}}_i(\mathbf{x}).$$

We actually compute the real part of $\widetilde{\mathbf{E}}$ and $\widetilde{\mathbf{H}}$,

$$\begin{aligned}\mathbf{E}(\mathbf{x},t) &= \Re\{(\cos(\omega t) - i\sin(\omega t))(\widehat{\mathbf{E}}_r + i\widehat{\mathbf{E}}_i)\}\\ &= \cos(\omega t)\widehat{\mathbf{E}}_r + \sin(\omega t)\widehat{\mathbf{E}}_i\\ \mathbf{H}(\mathbf{x},t) &= \cos(\omega t)\widehat{\mathbf{H}}_r + \sin(\omega t)\widehat{\mathbf{H}}_i\end{aligned}$$

From the time harmonic form of Maxwell's equations

$$-i\omega\epsilon\widehat{\mathbf{E}} = \nabla\times\widehat{\mathbf{H}},$$
$$i\omega\mu\widehat{\mathbf{H}} = \nabla\times\widehat{\mathbf{E}},$$

we obtain the relation for $(\widehat{\mathbf{H}}_r, \widehat{\mathbf{H}}_i)$ in terms of $(\widehat{\mathbf{E}}_r, \widehat{\mathbf{E}}_i)$

$$\omega\mu\widehat{\mathbf{H}}_r = \nabla\times\widehat{\mathbf{E}}_i, \tag{91}$$
$$-\omega\mu\widehat{\mathbf{H}}_i = \nabla\times\widehat{\mathbf{E}}_r. \tag{92}$$

Thus if we know $(\widehat{\mathbf{E}}_r, \widehat{\mathbf{E}}_i)$, we can determine $\mathbf{H}$. Given the computed solution for the electric field at two times,

$$\mathbf{E}(\mathbf{x},t_1) = \cos(\omega t_1)\widehat{\mathbf{E}}_r + \sin(\omega t_1)\widehat{\mathbf{E}}_i,$$
$$\mathbf{E}(\mathbf{x},t_2) = \cos(\omega t_2)\widehat{\mathbf{E}}_r + \sin(\omega t_2)\widehat{\mathbf{E}}_i,$$

we can solve for $(\widehat{\mathbf{E}}_r, \widehat{\mathbf{E}}_i)$,

$$\widehat{\mathbf{E}}_r(\mathbf{x}) = \Big(\quad \sin(\omega t_2)\mathbf{E}(\mathbf{x},t_1) - \sin(\omega t_1)\mathbf{E}(\mathbf{x},t_2)\Big)/\sin(\omega(t_2-t_1)), \tag{93}$$
$$\widehat{\mathbf{E}}_i(\mathbf{x}) = \Big(-\cos(\omega t_2)\mathbf{E}(\mathbf{x},t_1) + \cos(\omega t_1)\mathbf{E}(\mathbf{x},t_2)\Big)/\sin(\omega(t_2-t_1)). \tag{94}$$

Given $(\widehat{\mathbf{E}}_r, \widehat{\mathbf{E}}_i)$ we can compute $(\widehat{\mathbf{H}}_r, \widehat{\mathbf{H}}_i)$ from (91)-(92). Thus we can compute $\mathbf{H}(\mathbf{x},t)$ given $\mathbf{E}$ at two times.

The intensity is given by

$$\mathcal{I} = \frac{1}{P}\int_0^P \frac{1}{2}c\epsilon|\mathbf{E}|^2 + \frac{1}{2}c\mu|\mathbf{H}|^2\ dt,$$

where P is the period. Now we could compute the intensity from the electric field and magnetic field values determined above,

$$\begin{aligned}\frac{1}{P}\int_0^P |\mathbf{E}|^2\ dt &= \frac{1}{P}\int_0^P |\cos(\omega t)\widehat{\mathbf{E}}_r + \sin(\omega t)\widehat{\mathbf{E}}_i|^2\ dt\\ &= \frac{1}{2}\Big(|\widehat{\mathbf{E}}_r|^2 + |\widehat{\mathbf{E}}_i|^2\Big)\end{aligned}$$

and thus the intensity can be computed using

$$\mathcal{I} = \frac{1}{4}c\epsilon\Big(|\widehat{\mathbf{E}}_r|^2 + |\widehat{\mathbf{E}}_i|^2\Big) + \frac{1}{4}c\mu\Big(|\widehat{\mathbf{H}}_r|^2 + |\widehat{\mathbf{H}}_i|^2\Big), \tag{95}$$

$$= \frac{1}{4}c\epsilon\Big(|\widehat{\mathbf{E}}_r|^2 + |\widehat{\mathbf{E}}_i|^2\Big) + \frac{1}{4}\frac{c}{\mu\omega^2}\Big(|\nabla\times\widehat{\mathbf{E}}_i|^2 + |\nabla\times\widehat{\mathbf{E}}_r|^2\Big), \tag{96}$$

Alternatively, knowing $\mathbf{E}$ and $\nabla\times\mathbf{E}$, we could compute the intensity by the time average

$$\mathcal{I} = \frac{1}{P}\int_0^P \frac{1}{2}c\epsilon|\mathbf{E}|^2 + \frac{1}{2}c\mu|\mathbf{H}|^2\ dt,$$

$$= \frac{1}{P}\int_0^P \frac{1}{2}c\epsilon|\mathbf{E}|^2 + \frac{1}{2}\frac{c}{\mu\omega^2}|\nabla\times\mathbf{E}|^2\ dt,$$

which follows from (96).

Here is the suggested approach for computing the intensity whenever it is needed for output. Given $\mathbf{E}(\mathbf{x},t)$ at times $t_2 = t$, and the previous time $t_1 = t - \Delta t$,

1. Compute $\widehat{\mathbf{E}}_r(\mathbf{x})$ and $\widehat{\mathbf{E}}_i(\mathbf{x})$ from (93)-(94).
2. Compute $\nabla\times\widehat{\mathbf{E}}_r(\mathbf{x})$ and $\nabla\times\widehat{\mathbf{E}}_i(\mathbf{x})$ by difference approximation.
3. Compute $\mathcal{I}$ from (96).

Approach 2: If we know $\mathbf{H}$ as well as $\mathbf{E}$ then we can can compute $\widehat{\mathbf{H}}_r(\mathbf{x})$ and $\widehat{\mathbf{H}}_i(\mathbf{x})$ (following (93)-(94)) and then use (95) without the need to compute $\nabla\times\widehat{\mathbf{E}}_r(\mathbf{x})$ etc.

8 Numerial results

8.1 Chirped plane wave results

Here are some convergence results for verifying the chirped plane wave boundary forcing. To test the chirped plane wave boundary forcing we consider a simple geometry of a flat PEC boundary at $x = 0$. The scattered field due to an incident chirped plane-wave in the x-direction will just be another chirped wave.

Tables 5-6 show convergence results for scattering of a chirped plane wave off a planar PEC boundary at $x = 0$.

Chirped wave, flat PEC boundary, 2D, FD22					
grid	N	$E_j^{E_y}$	r	$E_j^{H_z}$	r
nonSquarenp1	1	9.0e-2		3.0e-1	
nonSquarenp2	2	4.1e-2	2.2	8.1e-2	3.7
nonSquarenp4	4	1.4e-2	3.0	2.0e-2	4.1
nonSquarenp8	8	3.3e-3	4.2	4.8e-3	4.1
nonSquarenp16	16	7.8e-4	4.2	1.2e-3	4.0
rate		1.73		2.00	

Table 1: Cgmx, chirped, method=CGFD2, max norm, order=2, $t = 1.5$, m=2, n=2, cfl=0.9, diss=0, dissOrder=−1, Sat Aug 13 19:03:01 2016

Chirped wave, flat PEC boundary, 2D, FD44					
grid	N	$E_j^{E_y}$	r	$E_j^{H_z}$	r
nonSquarenp1	1	3.1e-2		4.9e-2	
nonSquarenp2	2	8.9e-3	3.4	9.5e-3	5.2
nonSquarenp4	4	1.1e-3	8.4	1.1e-3	8.4
nonSquarenp8	8	6.8e-5	15.7	7.0e-5	16.1
nonSquarenp16	16	4.1e-6	16.6	4.2e-6	16.8
rate		3.28		3.41	

Table 2: Cgmx, chirped, method=CGFD4, max norm, order=4, $t = 1.5$, m=2, n=2, cfl=0.9, diss=0, dissOrder=−1, Sat Aug 13 19:04:55 2016

Chirped wave, flat PEC boundary, 2D, SOSUP22									
grid	N	$E_j^{E_y}$	r	$E_j^{H_z}$	r	$E_j^{\dot{E}_y}$	r	$E_j^{\dot{H}_z}$	r
nonSquarenp1	1	1.2e-1		3.6e-1		2.0e+		1.8e+	
nonSquarenp2	2	7.1e-2	1.7	6.2e-2	5.8	1.1e+	1.8	2.1e+	0.9
nonSquarenp4	4	2.9e-2	2.5	2.6e-2	2.4	4.8e-1	2.4	6.3e-1	3.3
nonSquarenp8	8	8.7e-3	3.3	8.1e-3	3.2	1.7e-1	2.9	1.6e-1	4.0
nonSquarenp16	16	2.3e-3	3.8	2.2e-3	3.7	4.6e-2	3.7	4.2e-2	3.8
nonSquarenp32	32	5.8e-4	4.0	5.6e-4	3.9	1.2e-2	3.9	1.1e-2	3.8
rate		1.58		1.80		1.50		1.60	

Table 3: Cgmx, chirped, method=sosup, max norm, order=2, $t = 1.5$, cfl=0.9, diss=0, dissOrder=−1, Sun Aug 14 07:13:47 2016

Chirped wave, flat PEC boundary, 2D, SOSUP44									
grid	N	$E_j^{E_y}$	r	$E_j^{H_z}$	r	$E_j^{\dot{E}_y}$	r	$E_j^{\dot{H}_z}$	r
nonSquarenp1	1	4.4e-2		4.7e-2		1.3e+		1.4e+	
nonSquarenp2	2	1.8e-2	2.5	1.9e-2	2.4	4.7e-1	2.8	5.0e-1	2.7
nonSquarenp4	4	3.3e-3	5.4	3.5e-3	5.4	9.9e-2	4.8	1.0e-1	4.8
nonSquarenp8	8	3.1e-4	10.6	3.2e-4	10.9	1.0e-2	9.7	1.1e-2	9.7
nonSquarenp16	16	2.2e-5	13.8	2.3e-5	14.1	7.7e-4	13.3	7.9e-4	13.6
nonSquarenp32	32	1.4e-6	15.4	1.5e-6	15.6	5.0e-5	15.3	5.1e-5	15.5
rate		3.05		3.07		2.99		2.99	

Table 4: Cgmx, chirped, method=sosup, max norm, order=4, $t = 1.5$, m=2, n=2, cfl=0.9, diss=0, dissOrder=-1, Sat Aug 13 19:11:08 2016

Chirped wave, flat PEC boundary, 3D, FD22			
grid	N	$E_j^{E_y}$	r
nonBoxnpp1	1	3.1e-2	
nonBoxnpp2	2	9.5e-3	3.3
nonBoxnpp4	4	2.4e-3	3.9
nonBoxnpp8	8	5.8e-4	4.1
rate		1.92	

Table 5: Cgmx, chirped, method=CGFD2, max norm, order=2, $t = 1.5$, cfl=0.9, diss=0, dissOrder=-1, Sun Aug 14 07:01:43 2016

Chirped wave, flat PEC boundary, 3D, FD44			
grid	N	$E_j^{E_y}$	r
nonBoxnpp1	1	5.1e-3	
nonBoxnpp2	2	4.9e-4	10.5
nonBoxnpp4	4	3.2e-5	15.0
nonBoxnpp8	8	2.0e-6	16.0
rate		3.78	

Table 6: Cgmx, chirped, method=CGFD4, max norm, order=4, $t = 1.5$, cfl=0.9, diss=0, dissOrder=-1, Sun Aug 14 07:17:09 2016

Chirped wave, flat PEC boundary, 3D, SOSUP22					
grid	N	$E_j^{E_y}$	r	$E_j^{\dot{E}_y}$	r
nonBoxnpp1	1	5.1e-2		8.6e-1	
nonBoxnpp2	2	1.8e-2	2.8	3.4e-1	2.5
nonBoxnpp4	4	5.1e-3	3.6	1.0e-1	3.4
rate		1.65		1.54	

Table 7: Cgmx, chirped, method=sosup, max norm, order=2, $t = 1.5$, cfl=0.9, diss=0, dissOrder=-1, Sun Aug 14 08:20:38 2016

Chirped wave, flat PEC boundary, 3D, SOSUP44					
grid	N	$E_j^{E_y}$	r	$E_j^{\dot{E}_y}$	r
nonBoxnpp1	1	1.1e-2		2.8e-1	
nonBoxnpp2	2	1.4e-3	7.2	4.6e-2	6.1
nonBoxnpp4	4	1.2e-4	12.2	4.2e-3	11.1
rate		3.23		3.04	

Table 8: Cgmx, chirped, method=sosup, max norm, order=4, $t = 1.5$, cfl=0.9, diss=0, dissOrder=-1, Sun Aug 14 07:46:45 2016

8.2 Scattering from some solid objects

Figure 5 shows results from the scattering of a Gaussian plane wave from a collection of solid dispersive spheres.
See `cg/mx/runs/solidObjects`.

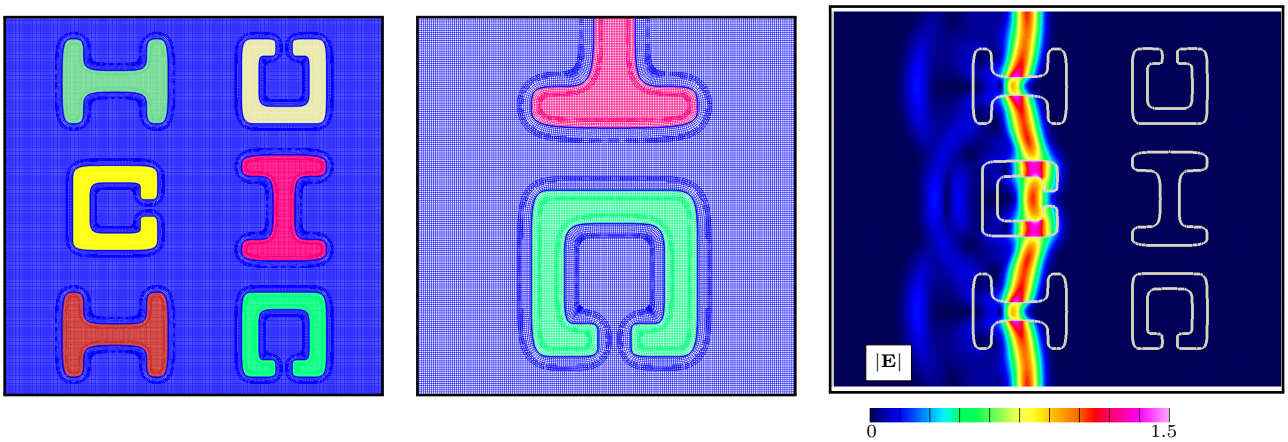

Figure 4: Left and middle: composite grid for some solid objects. Right: Scattering of a Gaussian plane wave from a collection of solid dispersive bodies.

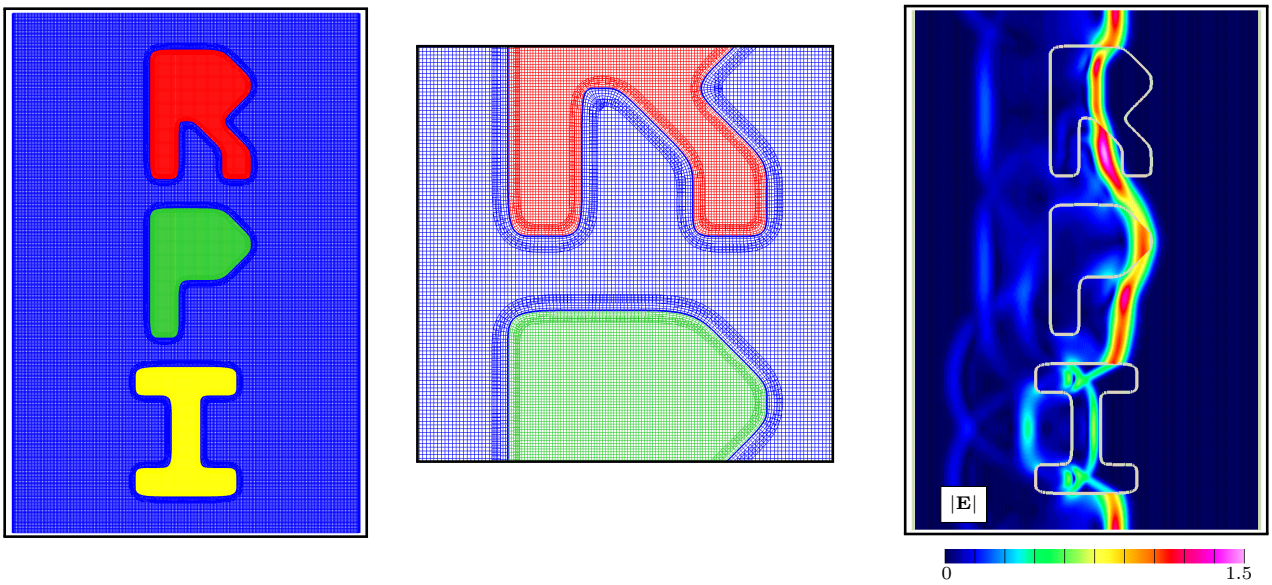

Figure 5: Left and middle: composite grid for some solid objects. Right: Scattering of a Gaussian plane wave from a collection of solid dispersive bodies.

8.3 Scattering from an array of dispersive solid spheres

Figure 6 shows results from the scattering of a Gaussian plane wave from a collection of 27 solid dispersive spheres. See `cg/mx/runs/solidSphereArray`.

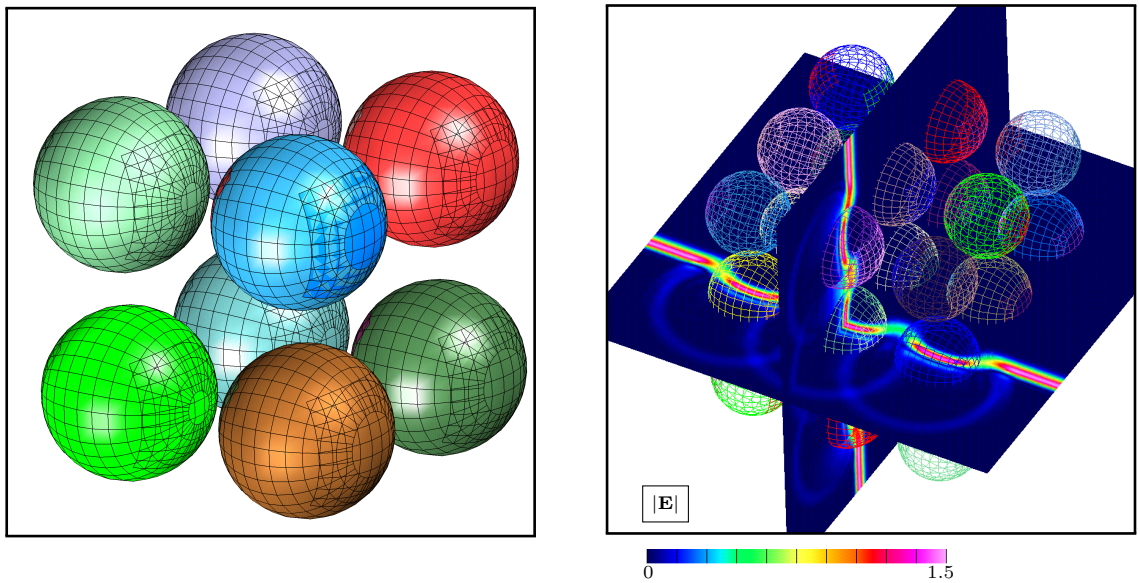

Figure 6: Left: composite grid for an array of eight solid spheres. Right: Scattering of a Gaussian plane wave from a collection of 27 solid dispersive spheres.

9 Maxwell's Equations

The time dependent Maxwell's equations for linear, isotropic and non-dispersive materials are

$$\partial_t \mathbf{E} = \frac{1}{\epsilon}\nabla \times \mathbf{H} - \frac{1}{\epsilon}\mathbf{J}, \tag{97}$$

$$\partial_t \mathbf{H} = -\frac{1}{\mu}\nabla \times \mathbf{E}, \tag{98}$$

$$\nabla \cdot (\epsilon \mathbf{E}) = \rho, \quad \nabla \cdot (\mu \mathbf{H}) = 0, \tag{99}$$

Here $\mathbf{E} = \mathbf{E}(\mathbf{x}, t)$ is the electric field, $\mathbf{H} = \mathbf{H}(\mathbf{x}, t)$ is the magnetic field, $\rho = \rho(\mathbf{x}, t)$ is the electric charge density, $\mathbf{J} = \mathbf{J}(\mathbf{x}, t)$ is the electric current density, $\epsilon = \epsilon(\mathbf{x})$ is the electric permittivity, and $\mu = \mu(\mathbf{x})$ is the magnetic permeability. This first-order system for Maxwell's equations can also be written in a second-order form. By taking the time derivatives of (98) and (97) and using (99) it follows that

$$\epsilon\mu\ \partial_t^2 \mathbf{E} = \Delta \mathbf{E} + \nabla\Big(\nabla \ln \epsilon\ \cdot \mathbf{E}\Big) + \nabla \ln \mu \times \Big(\nabla \times \mathbf{E}\Big) - \nabla(\frac{1}{\epsilon}\rho) - \mu \partial_t \mathbf{J}, \tag{100}$$

$$\epsilon\mu\ \partial_t^2 \mathbf{H} = \Delta \mathbf{H} + \nabla\Big(\nabla \ln \mu\ \cdot \mathbf{H}\Big) + \nabla \ln \epsilon \times \Big(\nabla \times \mathbf{H}\Big) + \epsilon \nabla \times (\frac{1}{\epsilon}\mathbf{J}). \tag{101}$$

It is evident that the equations for the electric and magnetic field are decoupled with each satisfying a vector wave equation with lower order terms. In the case of constant μ and ϵ and no charges, $\rho = \mathbf{J} = 0$, the equations simplify to the classical second-order wave equations,

$$\partial_t^2 \mathbf{E} = c^2\ \Delta \mathbf{E}, \qquad \partial_t^2 \mathbf{H} = c^2\ \Delta \mathbf{H} \tag{102}$$

where $c^2 = 1/(\epsilon\mu)$. There are some advantages to solving the second-order form of the equations rather than the first-order system. One advantage is that in some cases it is only necessary to solve for one of the variables, say $\mathbf{E}$. If the other variable, $\mathbf{H}$ is required, it can be determined by integrating equation (98) as an ordinary differential equation with known $\mathbf{E}$. Alternatively, as a post-processing step $\mathbf{H}$ can be computed from an elliptic boundary value problem formed by taking the curl of equation (97). Another advantage of the second-order form, which simplifies the implementation on an overlapping grid, is that there is no need to use a staggered grid formulation. Many schemes approximating the first order system (98-99) rely on a staggered arrangement of the components of $\mathbf{E}$ and $\mathbf{H}$ such as the popular Yee scheme [**?**] for Cartesian grids.

10 Dispersive Maxwell's Equations

FINISH ME...

10.1 Interface projection

The primary interface jump conditions (**??**) and (**??**) are not explicitly imposed when solving for the ghost point values at the interface, only the second time-derivative of these conditions are imposed. To strictly enforce (**??**) and (**??**) we apply an interface projection at each time-step before assigning the ghost values.

To define the projection we consider the solution to a Riemann problem at the interface. The initial condition for the Riemann problem, taken at a pseudo-time $t = 0$, consists of a constant left state for $x < 0$ and a constant right state for $x > 0$. The values for these left and right states are given by the predicted interface values coming from the left and right interior updates on the interface. The central state in the solution to the Riemann problem at $t = 0^+$ will define the projected interface state. To derive the form of the projected state, it is sufficient to consider the Maxwell's equations in two dimensions for a TE-z polarized wave, (taking $u = E_x$, $v = E_y$, $w = H_z$),

$$\epsilon_0 \partial_t u = \partial_y w - \epsilon_0^{-1} p_t, \tag{103a}$$
$$\epsilon_0 \partial_t v = -\partial_x w - \epsilon_0^{-1} q_t, \tag{103b}$$
$$\mu_0 \partial_t w = \partial_y u - \partial_x v, \tag{103c}$$

where p and q are the components of the polarization vector. Equations (103) define a hyperbolic system with lower order terms coming from p_t and q_t which can be thought of as being proprotional to u and v. Hence we can drop the polarization terms from the characteristic analysis. For an interface at $x = 0$ we consider waves in the x-direction and the equations then reduce to

$$\begin{aligned}
\epsilon \partial_t u &= 0, \\
\epsilon \partial_t v &= -\partial_x w, \\
\mu \partial_t w &= -\partial_x v
\end{aligned}$$

The variable u (normal component of E) is thus a characteristic variable with characteristic speed $\lambda = 0$. The variables v and w form a coupled wave equation with characteristics variables $\psi_\pm$, and characteristic speeds $\lambda_\pm$, given by

$$\begin{aligned}
\psi_+ &= v - \eta w, \qquad & \lambda_+ &= c, \\
\psi_- &= v + \eta w, \qquad & \lambda_- &= -c,
\end{aligned}$$

where η is the electric impedance,

$$\eta \stackrel{\text{def}}{=} \sqrt{\frac{\mu}{\epsilon}}.$$

Solving the Riemman problem with the interface jump conditions

$$\begin{aligned}
[\epsilon u + p] &= 0, \\
[v] &= 0, \\
[w] &= 0,
\end{aligned}$$

leads to the central state, denoted by (v^I, w^I), given

$$v_I = \frac{\eta_L^{-1} v_L + \eta_R^{-1} v_R}{\eta_R^{-1} + \eta_L^{-1}} + \frac{1}{\eta_R^{-1} + \eta_L^{-1}} \Big[w_R - w_L \Big], \tag{104a}$$
$$w_I = \frac{\eta_R w_R + \eta_L w_L}{\eta_R + \eta_L} + \frac{1}{\eta_R + \eta_L} \Big[v_R - v_L \Big]. \tag{104b}$$

It is seen that to leading order v^I is an inverse impedance average, while w^I is a impedance average. The second terms in (104) can usually be neglected as they are should proportional to the truncation error.

The treatment of u and (p, q) at the interface is not as straightforward and we propose two possible approaches. In each case we do not alter the polarization vectors and so these remain fixed.

Option 1: Given $D_L \stackrel{\text{def}}{=} \epsilon_L u_L + p_L$ and $D_R \stackrel{\text{def}}{=} \epsilon_R u_R + p_R$ choose the interface value D^I to be the left or right state based which side has a smaller ϵ_m,

$$D^I = \begin{cases} D_L, & \text{if } \epsilon_L \le \epsilon_R, \\ D_R, & \text{if } \epsilon_R < \epsilon_L. \end{cases} \tag{105}$$

Thus if $\epsilon_R < \epsilon_L$ we take $D^I = D_R$ and

$$u_L^I = \frac{\epsilon_R}{\epsilon_L} u_R + \frac{p_R - p_L}{\epsilon_L}, \quad u_R^I = u_R, \tag{106a}$$

otherwise we take $D^I = D_L$ and

$$u_L^I = u_L, \quad u_R^I = \frac{D^I - p_R}{\epsilon_R} = \frac{\epsilon_L}{\epsilon_R} u_L + \frac{p_L - p_R}{\epsilon_R}, \tag{106b}$$

The intuition for this choice is based on a numerical stability argument. In assigning u_L^I in (106a) any perturbations to $u_R \to u_R + \delta$ (e.g. round-off errors) will not be amplified when multiplying by $\epsilon_R/\epsilon_L < 1$.

Option 2: The variable u (or $d \stackrel{\text{def}}{=} \epsilon u + p$) has a characteristic speed $\lambda = 0$ and so the characteristics do not immediately indicate how to project two values of u on either side of the interface. In option 2 we consider what happens if we add an upwind type dissipation to the d equation, which then turns the equation into the heat equation,

$$\partial_t d = \partial_x (\mathcal{K}\, \partial_x d_m), \tag{107}$$

where the diffusion coefficient is taken to be of the form

$$\mathcal{K} = \begin{cases} \alpha_L c_L h_L & \text{for } x < 0, \\ \alpha_R c_R h_R & \text{for } x > 0, \end{cases} \tag{108}$$

where α_m is some non-dimensional parameter, $c_m = 1/\sqrt{\epsilon_m \mu_m}$ is chosen as a velocity scale, and h_m is a representative grid spacing. First note that there is a similarity solution to the heat-equation given by **CHECK ME***

$$d(x,t) = A + B \text{ erf}\left(\frac{x}{2\sqrt{\mathcal{K}\, t}}\right),$$

Now consider a Riemann problem for the heat equation for some left and right states,

$$d(x,0) = \begin{cases} D_L, & \text{if } x < 0, \\ D_R, & \text{if } x > 0. \end{cases}$$

The boundary conditions at infinity are

$$\begin{aligned} d &= D_R, \qquad x \to \infty, \\ d &= D_L, \qquad x \to -\infty, \end{aligned}$$

which implies the solution takes the form (since $\text{erf}(x)$ asymptotes to ± 1 as $x \to \pm\infty$),

$$\begin{aligned} d(x,t) &= A_L + (A_L - D_L) \text{ erf}\left(\frac{x}{2\sqrt{\mathcal{K}_L\, t}}\right), \qquad \text{for } x < 0, \\ d(x,t) &= A_R + (D_R - A_R) \text{ erf}\left(\frac{x}{2\sqrt{\mathcal{K}_R\, t}}\right), \qquad \text{for } x > 0. \end{aligned}$$

The jump conditions at $x = 0$ are

$$\begin{aligned} [d]_I &= 0, \\ [\mathcal{K}\, \partial_x d] &= 0, \end{aligned}$$

which gives (for $t > 0$)

$$A_L = A_R,$$
$$\mathcal{K}_L(A_L - u_L)\frac{1}{\sqrt{\mathcal{K}_L}} = \mathcal{K}_R(u_R - A_R)\frac{1}{\sqrt{\mathcal{K}_R}},$$

which implies that the value at the interface, $x = 0$, for $t > 0$ is

$$D^I = A_L = A_R = \frac{\sqrt{\mathcal{K}_R}\, D_R + \sqrt{\mathcal{K}_L}\, D_L}{\sqrt{\mathcal{K}_L} + \sqrt{\mathcal{K}_R}},$$

Taking $\alpha_L h_L = \alpha_R h_R$ as a typical case implies D^I is an weighted average D_L and D_R,

$$D^I = A_L = A_R = \frac{\sqrt{c_R}\, D_R + \sqrt{c_L}\, D_L}{\sqrt{c_L} + \sqrt{c_R}}, \tag{109}$$

with weights given by $\sqrt{c_m}$. Equation (109) shows that the solution from the side with a larger value for $c_m = 1/\sqrt{\epsilon_m \mu_m}$ (i.e. a smaller value for $\epsilon_m \mu_m$) is weighted more in the average. This roughly corresponds to option 1 where the side with smaller ϵ_m was taken to define D^I.

Summary. The tangential components of the electric field are projected using an inverse impedance weighted average,

$$\mathbf{n} \times \mathbf{E}^I_m = \frac{\eta_L^{-1}\mathbf{n} \times \mathbf{E}^L + \eta_R^{-1}\mathbf{n} \times \mathbf{E}^R}{\eta_L^{-1} + \eta_R^{-1}}, \qquad m = L, R.$$

The normal component of the field is projected using

$$\mathbf{n} \cdot \mathbf{E}^I_m = \frac{D^I - \mathbf{n} \cdot \mathbf{P}_m}{\epsilon_m}, \qquad m = L, R.$$

where D^I is defined by (105) or (109).

11 Acknowledgments

Thanks to Kyle Chand who contributed to parts of Cgmx. Thanks to Tom Hagstrom for contributing some far field boundary condition routines.

This work was performed under the auspices of the U.S. Department of Energy (DOE) by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 and by DOE contracts from the ASCR Applied Math Program.

A Absorbing Boundary Conditions

In this section we discuss absorbing boundary conditions (ABC's).

A.1 Engquist-Majda one-way wave equations

Consider the wave equation

$$u_{tt} = u_{xx} + u_{yy} + u_{zz}$$

We consider a boundary at $x = 0$. The wave equation can be formally factored using pseudo-differential operators

$$(D_x - D_t\sqrt{1 - S^2})(D_x + D_t\sqrt{1 - S^2}) = 0$$

where

$$S^2 = (D_y^2 + D_z^2)/D_t^2$$

The operator $G^- = D_x - D_t\sqrt{1-S^2}$ only supports waves moving to the left. Applying G^- to a wave function U will absorb waves moving to left (at any angle). To see this consider a wave that moves in the negative x-direction

$$U(\mathbf{x},t) = e^{i(\mathbf{k}\cdot\mathbf{x}+\omega t)}$$

with $k_x > 0$ and $\omega = \sqrt{\mathbf{k}\cdot\mathbf{k}} > 0$.

In Fourier space

$$\begin{aligned}\mathcal{F}\{G^-U\} &= ik_x - i\omega\sqrt{1-(k_y^2+k_z^2)/\omega^2} \\ &= ik_x - i\sqrt{\omega^2 - k_y^2 + k_z^2} \\ &= ik_x - i|k_x| \\ &= 0\end{aligned}$$

thus giving $D_t G^- U = 0$.

We will apply $G^- = D_x - D_t\sqrt{1-S^2}$ as a non-reflecting boundary condition. If applied exactly it will absorb (treat exactly without reflection) all outgoing plane waves. It will not handle evanescent modes.

Aside: Evanescent waves When a plane wave hits a material interface there will be a reflected wave and a refracted wave (transmitted wave). If the refracted ray bends toward the normal we have what is called external reflection. If it bends away from the normal it is called internal reflection. At a critical angle θ_c the refracted wave is parallel to the interface. For angles greater than θ_c there is no refracted wave and we have total internal reflection. There is however an evanescent wave that travels parallel to the boundary and decays exponentially into the second medium. The evanescent wave ensures that the tangential component of the electric field is continuous across the interface.

If the wave approaching the boundary is nearly normal incidence then $k_x^2 \gg k_y^2 + k_z^2$ and thus

$$\widehat{S^2} = (k_y^2+k_z^2)/\omega^2 = (k_y^2+k_z^2)/(k_x^2+k_y^2+k_z^2) \ll 1$$

Thus S^2 is thought of as being small and we approximate

$$\sqrt{1-S^2} \approx p_0 + p_2 S^2$$

Whence

$$\begin{aligned}D_t G^- &\approx D_t D_x - D_t^2(p_0 + p_2 S^2) \\ &\approx D_t D_x - p_0 D_t^2 - p_2(D_y^2 + D_z^2) \\ &= D_t D_x - p_0 D_x^2 - (p_0+p_2)(D_y^2+D_z^2)\end{aligned}$$

For $p_0 = 1$, $p_2 = -1/2$ this gives the approximate (second-order) Engquist-Majda ABC,

$$L_2^{em} u = \partial_t\partial_x u - \partial_t^2 u + \frac{1}{2}(\partial_y^2+\partial_z^2)u = 0 \tag{110}$$

$$= \partial_t\partial_x u - \partial_x^2 u - \frac{1}{2}(\partial_y^2+\partial_z^2)u = 0 \tag{111}$$

which is exact for waves impinging on the boundary in the normal direction (angle of incidence of zero). Mur's scheme is a discretization of the above BC – Mur centered his scheme at $t+\Delta/2$ and $\Delta x/2$ to give a second order approximation using only two time levels.

We can use a better approximation

$$\sqrt{1-S^2} \approx \frac{p_0 + p_2 S^2}{q_0 + q_2 S^2}$$

This give a third-order boundary condition

$$L_3^{em} u = \partial_x(q_0\partial_t^2 + q_2(\partial_y^2+\partial_z^2)) - \partial_t(p_0\partial_t^2 + p_2(\partial_y^2+\partial_z^2)) \tag{112}$$

$$= q_0\partial_x\partial_t^2 + q_2\partial_x(\partial_y^2+\partial_z^2) - p_0\partial_t^3 - p_2\partial_t(\partial_y^2+\partial_z^2) \tag{113}$$

Engquist and Majda suggested $p_0 = q_0 = 1$ and $p_2 = -3/4$, $q_2 = -1/4$ which is the Padé approximation (minimizing the error near $S = 0$.

Trefethen and Halpern considered other possibilities such as a Chebyshev or least squares. One could choose the coefficients to make the approximation exact at other angles.

A.2 Second-order accurate discretization

Consider Engquist-Majda ABC,

$$\partial_t\partial_x u = \alpha\partial_x^2 u + \beta(\partial_y^2+\partial_z^2)u \tag{114}$$

where, for example $\alpha = c$ and $\beta = \frac{1}{2}c$, will give a second-order acuurate approximation. We can discretize this equation with the centered second-order accurate approximation

$$D_0^t D_0^x U_{ij}^n = \alpha D_+^x D_-^x U_{ij}^n + \beta(D_+^y D_-^y + D_+^z D_-^z)U_{ij}^n \tag{115}$$

This equation will give the ghost point value U_{-1j}^{n+1} given interior and boundary values at time t^{n+1} and old values at time t^n.

Here is another second-order approximation, centred at $t^{n+\frac{1}{2}}$ that uses only two time levels,

$$D_+^t D_0^x U_{ij}^n = \mathcal{A}_+^t\Big(\alpha D_+^x D_-^x U_{ij}^n + \beta(D_+^y D_-^y + D_+^z D_-^z)U_{ij}^n\Big) \tag{116}$$

$$\mathcal{A}_+^t f^n = \frac{1}{2}(f^{n+1}+f^n) \tag{117}$$

The value at the ghost point U_{-1j}^{n+1} in equation (116) can be explicitly solved for given interior and boundary values at time t^{n+1} and old values at time t^n.

This gives the approximation

$$\Big(\frac{1}{2\Delta t\Delta x}+\frac{\alpha}{2\Delta x^2}\Big)U_{-1j}^{n+1} = D_+^t D_0^x U_{ij}^n + \frac{1}{2\Delta t\Delta x}U_{-1j}^{n+1} \tag{118}$$

$$- \mathcal{A}_+^t\Big(\alpha D_+^x D_-^x U_{ij}^n - \beta(D_+^y D_-^y + D_+^z D_-^z)U_{ij}^n\Big) + \frac{\alpha}{2\Delta x^2}U_{-1j}^{n+1} \tag{119}$$

or

$$\Big(1+\alpha\frac{\Delta t}{\Delta x}\Big)U_{-1j}^{n+1} = 2\Delta t\Delta x\Big[D_+^t D_0^x U_{ij}^n + \frac{1}{2\Delta t\Delta x}U_{-1j}^{n+1} \tag{120}$$

$$- \mathcal{A}_+^t\Big(\alpha D_+^x D_-^x U_{ij}^n - \beta(D_+^y D_-^y + D_+^z D_-^z)U_{ij}^n\Big) + \frac{\alpha}{2\Delta x^2}U_{-1j}^{n+1}\Big] \tag{121}$$

Note: The right-hand-side of the above expression does not depend on U_{-1j}^{n+1}.

A.3 Fourth-order accuracy

For fourth-order accuracy we have two ghost points and need an additional numerical boundary condition. We could use the normal derivative of one of the above approximations,

$$\partial_x L_2^{em} u = D_t D_x^2 - p_0 D_x^3 - (p_0+p_2)D_x(D_y^2+D_z^2)$$

It might be appropriate to use the first order approximation $L_1^{em} = \partial_t - \partial_x$ times the second-order

$$L_1^{em}L_2^{em}u = (\partial_t-\partial_x)\Big[D_tD_x - p_0D_x^2 - (p_0+p_2)(D_y^2+D_z^2)\Big]$$

A.4 Absorbing boundary conditions on a curvilinear grid

Consider a rotated rectangular grid. The wave equation in transformed coordinates is

$$\begin{aligned} u_{tt} &= (r_x^2+r_y^2+r_z^2)u_{rr} + (s_x^2+s_y^2+s_z^2)u_{ss} + ... \\ &= D_n^2 u + \Delta_\tau u \\ D_n &= \|\nabla_{\mathbf{x}} r\|\partial_r \end{aligned}$$

where D_n is the normal derivative to a boundary $r = \text{const}$. Following the same argument as before we can derive the (second-order) Engquist-Majda ABC as

$$\partial_t D_n u = \alpha D_n^2 u + \beta\Delta_\tau u$$

Now consider a general curvilinear grid. The wave equation in transformed coordinates is

$$\begin{aligned} u_{tt} &= Lu \\ &= (r_x^2 + r_y^2 + r_z^2)u_{rr} + (s_x^2 + s_y^2 + s_z^2)u_{ss} + 2(r_x s_x + r_y s_y + r_z s_z)u_{rs} \\ &\quad + (r_{xx} + r_{yy} + r_{zz})u_r + (s_{xx} + s_{yy} + s_{zz})u_s + ... \\ &= D_n^2 u + \Delta_\tau u \\ D_n &= \|\nabla_{\mathbf{x}} r\| \partial_r \\ \Delta_\tau &= (L - D_n^2)u \end{aligned}$$

where we have arbitrarily chosen D_n to be an approximation to the normal derivative. This gives the ABC

$$\partial_t D_n u = \alpha D_n^2 u + \beta \Delta_\tau u$$

which may reasonably accurate for a nearly orthogonal grid.

A more accurate approximation for non-orthogonal grids could be to set D_n to the actual normal derivative:

$$\begin{aligned} D_n = \mathbf{n} \cdot \nabla &= \frac{\nabla_{\mathbf{x}} r}{|\nabla_{\mathbf{x}} r|} \cdot \nabla_{\mathbf{x}} \\ &= |\nabla_{\mathbf{x}} r| \partial_r + (\mathbf{n} \cdot \nabla s) \partial_s \end{aligned}$$

A.5 Non-reflecting Boundary Conditions and Incident Fields

The non-reflecting boundary conditions were derived assuming that there are no incoming waves, just waves leaving the domain. We can treat the case of an incident field arriving from outside the computational domain in a couple of ways.

Approach I: In this approach we write the total electric field in the neighbourhood of the boundary as the sum of a given incident field $\mathbf{E}^i(\mathbf{x}, t)$ and a scattered field $\mathbf{E}^s(\mathbf{x}, t)$,

$$\mathbf{E}(\mathbf{x}, t) = \mathbf{E}^i(\mathbf{x}, t) + \mathbf{E}^s(\mathbf{x}, t) \ .$$

We assume that the incident field is an exact solution of Maxwell's equations near the boundary. We can apply the non-reflecting boundary condition to $\mathbf{E}^s(\mathbf{x}, t)$ by subtracting $\mathbf{E}^i(\mathbf{x}, t)$ from the total field. In the discrete case, at each time step we can subtract $\mathbf{E}^i(\mathbf{x}, t)$ from $\mathbf{E}(\mathbf{x}, t)$ on a few points near the boundary, apply the NRBC, and then add back $\mathbf{E}^i(\mathbf{x}, t)$.

Approach II: In this approach we change the NRBC or ABC to account for the incident field. Given the NRBC,

$$\mathcal{L}\mathbf{E}^s = 0, \tag{122}$$

by substituting $\mathbf{E}^s(\mathbf{x}, t) = \mathbf{E}(\mathbf{x}, t) - \mathbf{E}^i(\mathbf{x}, t)$ we get the new condition for the total field

$$\mathcal{L}\mathbf{E} = \mathcal{L}\mathbf{E}^i \tag{123}$$

For example,

$$\partial_t \partial_x \mathbf{E} - \left(\alpha \partial_x^2 \mathbf{E} + \beta(\partial_y^2 + \partial_z^2)\mathbf{E} \right) = \partial_t \partial_x \mathbf{E}^i - \left(\alpha \partial_x^2 \mathbf{E}^i + \beta(\partial_y^2 + \partial_z^2)\mathbf{E}^i \right) \tag{124}$$

Note: We could potentially set the RHS to $G^- \mathbf{E}^i$ if $\mathcal{L} \approx G^-$.