PML Project* Summary

- This summary focuses on the implementation in the 2-D serial code.
- The Parallel 2-D and 3-D codes give similar results, so it's probably not just a "random" bug in this one.
- Here, we limit ourselves to the 1-D case of a uniform (along y) laser pulse moving in x.
- Launching a radially moving wave from the center gives similar results.
- I think that if we can get the 1-D pulse working, then I can get the radial wave to work too.
- The paper by Vay I reference here is his 2000 JCP paper.

*AKA the scourge of my life in June and July.

Ordering of the PML Routines in the Loop (1)

- PML is a treated as being separate from the box => separate arrays.
  - Allocate E and B arrays for the PML region.
  - Allocate and calculate the attenuation coefficients for each boundary.
- We have to do the $\Delta t/2\ \vec{B}$ and $\Delta t\ \vec{E}$ updates in separate routines.
  - Do $\Delta t/2\ \vec{B}$ push in Fourier space.
  - Shift $\vec{E}$ and $\vec{B}$ to Yee positions, copy $\vec{B}$ to buffer box array, FFT buffer $\vec{B}$ to real space.
- Call the $\vec{E}$ PML update routine.
  - Communicate $\vec{B}$ between the box buffer and the actual PML $\vec{B}$ arrays.
  - Communicate $\vec{B}$ between neighboring PML $\vec{B}$ arrays, if needed.
  - Propagate the $\vec{E}$ field in the PML arrays with the PML equations.
- Impose the $\vec{B}$ update on the actual box $\vec{B}$ array to use in the $\vec{E}$ update.
  - FFT the $\vec{B}$ box buffer to Fourier space, copy it to the simulation box $\vec{B}$ array, shift $\vec{E}$ and $\vec{B}$ back to integer grid points.

Ordering of the PML Routines in the Loop (2)

- Do the full $\vec{E}$ push on the actual box $\vec{E}$ array in Fourier space.
  - Shift $\vec{E}$ and $\vec{B}$ to Yee positions, copy $\vec{E}$ to buffer box array, FFT buffer $\vec{E}$ to real space.
- Call the $\vec{B}$ PML update routine.
  - Communicate $\vec{E}$ between the box buffer and the actual PML $\vec{E}$ arrays.
  - Propagate the $\vec{B}$ field in the PML arrays with the PML equations.
- Impose the $\vec{E}$ update on the actual box $\vec{E}$ array to use in the second $\vec{B}$ update.
  - FFT the $\vec{E}$ box buffer to Fourier space, copy it to the simulation box $\vec{E}$ array, shift $\vec{E}$ and $\vec{B}$ back to integer grid points.
- Call the second ½ $\vec{B}$ update, then repeat the loop.

The attenuation coefficients are calculated from Vay's paper (1)

- The t_js from Vay's Eq. 40 a distance x in the PML is $t(x) = e^{s_1 (s_2 x)^n}$.
  - Left/Lower: $t_j = t(j\Delta x - 1/2)$, $t_{j+1/2} = t(j\Delta x)$, $t_{j+1} = t(j\Delta x + 1/2)$.
  - Right/Upper: $t_j = t(j\Delta x)$, $t_{j+1/2} = t(j\Delta x + 1/2)$, $t_{j+1} = t(j\Delta x + 1)$.
  - $s_1 = -2/5(n+1)$, $s_2 = 1(T-3)$, I find $n = 4.6$ works better than $n = 4$.

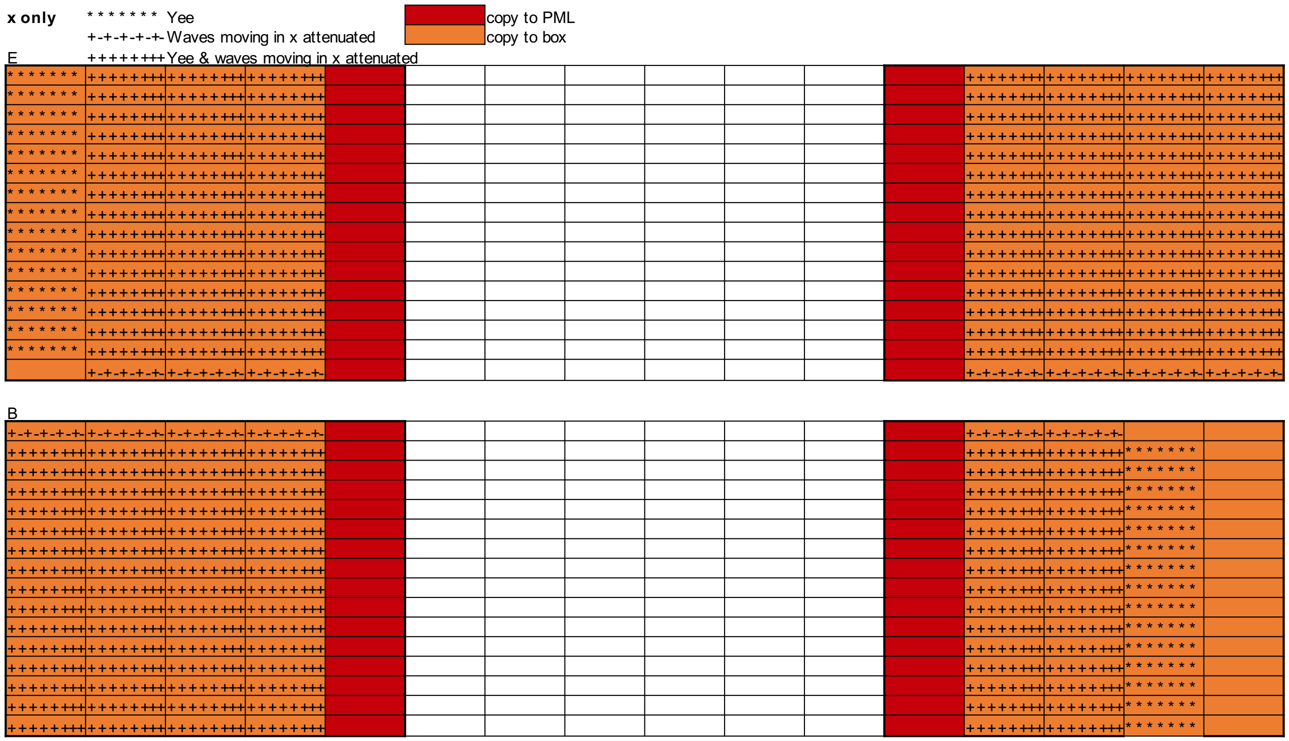
- The βs from Vay's Eq. 26 are
  - $\beta_{1E} = dt/dx\left(1 + \left(\frac{1-dt/dx}{1+dt/dx}\right)(1 - t_{j-1/2})\right)$.
  - $\beta_{1E} = dt/dx\left(1 + \left(\frac{1-dt/dx}{1+dt/dx}\right)(1 - t_{1/2})\right)$.
  - β_{2B} = $\beta_{2B} = dt/dx$.

The attenuation coefficients are calculated from Vay's paper (2)

- The coefficients for the fields in Vay's Eq. 27 are
  - Left/Lower:
  - $c_{1E}(T-j) = 1 - \beta_{1E} + t_{j+1/2}\beta_{2E}$ and $c_{B1}(T-j) = 1 - \beta_{1B} + t_{j1}\beta_{2B}$
  - $c_{2E}(T-j) = t_j\beta_{1E}$ and $c_{B2}(T-j) = t_{j+1/2}\beta_{1B}$
  - $c_{3E}(T-j) = c_{3E}(T-j) = \beta_{2E} = \beta_{2B} = dt/dx$.
  - Right/Upper:
  - $c_{E1}(j) = 1 - \beta_{1E} + t_{j+1/2}\beta_{2E}$ and $c_{B1}(j) = 1 - \beta_{1B} + t_{j1}\beta_{2B}$
  - $c_{E2}(j) = c_{B2}(j) = \beta_{2E} = \beta_{2B} = dt/dx$.
  - $c_{E3}(j) = t_j\beta_{1E}$ and $c_{B3}(j) = t_{j+1/2}\beta_{1B}$
- The code for these was copy/pasted from OSIRIS to UPIC.

The PML Overlaps with the Actual Simulation Box

- This is to comply with the periodic FFT.
- Whole PML is copied to box.
- Loop ranges to the standard Yee solver and attenuation regions are slightly different than in OSIRIS.
- Yes, that is a bug in the loop range on the right side in B.
  - Shouldn't't matter when thickness is large.



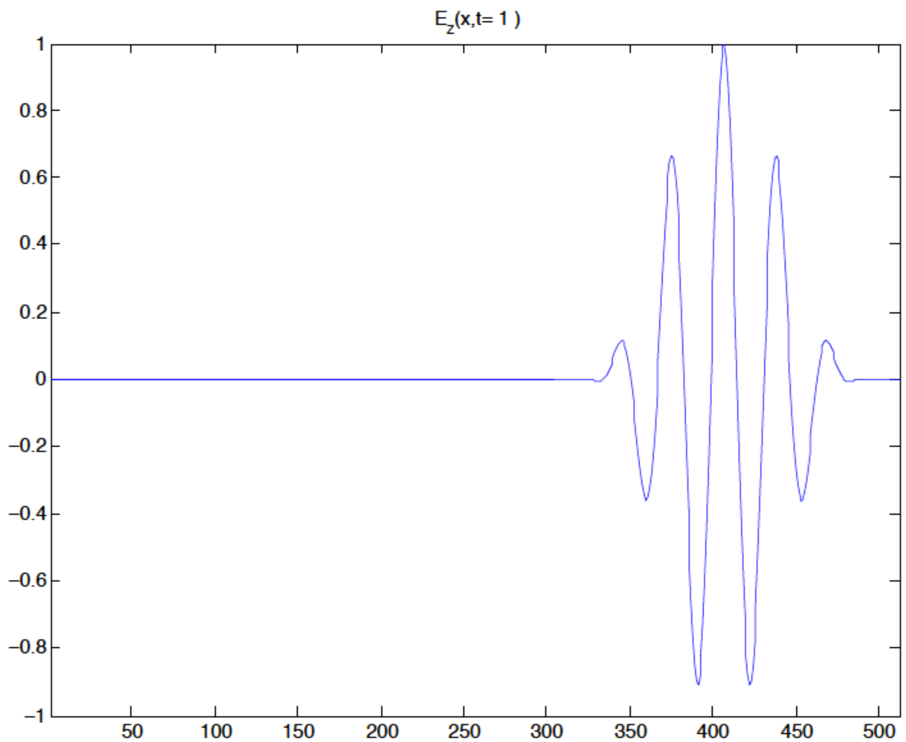
The Field Push Equations Come from the standard Yee solver and Vay's Paper

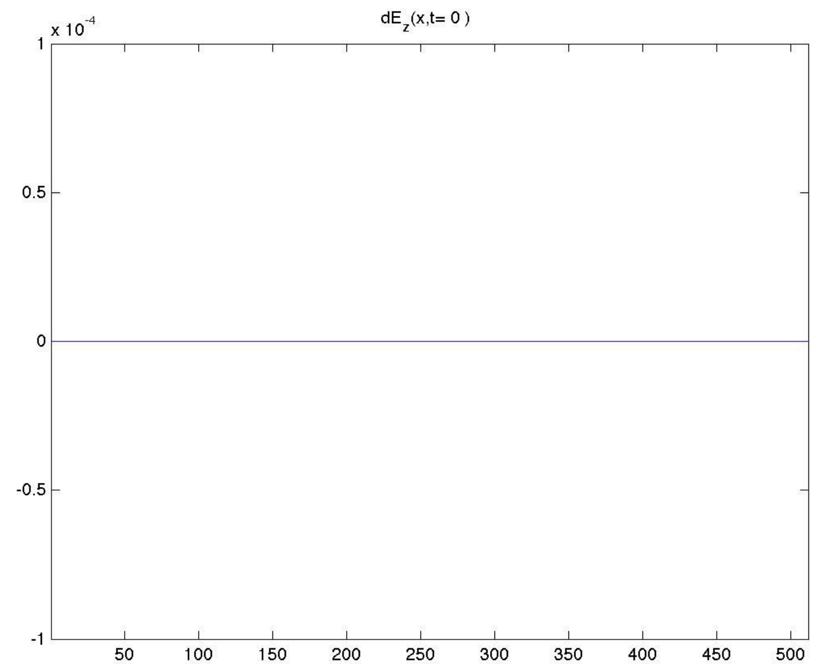
- The components of $\vec{E}$ moving in $\hat{y}$ are not attenuated in the x PML.
- $E_{xy}^{i,j} = E_{xy}^{i,j} + dt/dx\left(B_{zx}^{i,j} - B_{zx}^{i,j-1} + B_{zy}^{i,j} - B_{zy}^{i,j-1}\right).$
- $E_{zy}^{i,j} = E_{zy}^{i,j} - dt/dx\left(B_{xy}^{i,j} - B_{xy}^{i,j-1}\right).$
- The components of $\vec{E}$ moving in $\hat{x}$ are attenuated in the x PML with the 2-D version of Vay's Eq. 27.
- $E_{yx}^{i,j} = c_{1E}E_{yx}^{i,j} - c_{2E}\left(B_{zx}^{i,j} + B_{zy}^{i,j}\right) + c_{3E}\left(B_{zx}^{i-1,j} + B_{zy}^{i-1,j}\right).$
- $E_{zx}^{i,j} = c_{1E}E_{zx}^{i,j} + c_{2E}B_{yx}^{i,j} - c_{3E}B_{yx}^{i-1,j}.$
- Note: I give E and B both integer indices here because this is how I coded it up. They are staggered like the Yee grid, though.

The Field Push Equations Come from the standard Yee solver and Vay's Paper

- The components of $\vec{B}$ moving in $\hat{y}$ are not attenuated in the x PML.
- $B_{xy}^{i,j} = B_{xy}^{i,j} - dt/dx\left(E_{zx}^{i,j+1} - E_{zx}^{i,j} + E_{zy}^{i,j+1} - E_{zy}^{i,j}\right)$.
- $B_{zy}^{i,j} = B_{zy}^{i,j} + dt/dx\left(E_{xy}^{i,j+1} - E_{xy}^{i,j}\right)$.
- The components of $\vec{B}$ moving in $\hat{x}$ are attenuated in the x PML with the 2-D version of Vay's Eq. 27.
- $B_{yx}^{i,j} = c_{1B}B_{yx}^{i,j} + c_{2B}\left(E_{zx}^{i+1,j} + E_{zy}^{i+1,j}\right) - c_{3B}\left(E_{zx}^{i,j} + E_{zy}^{i,j}\right)$.
- $B_{zx}^{i,j} = c_{1B}B_{zx}^{i,j} - c_{2B}E_{yx}^{i+1,j} + c_{3B}E_{yx}^{i,j}$.
- Note: I give E and B both integer indices here because this is how I coded it up. They are staggered like the Yee grid, though.

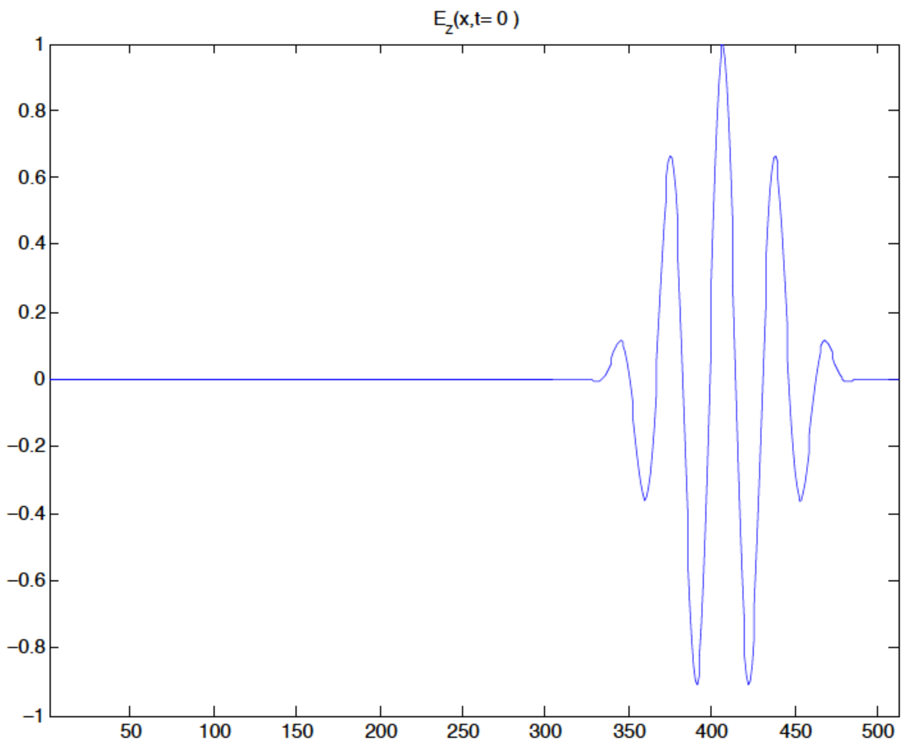
Using 20 PML Cells Results In Partial Attenuation

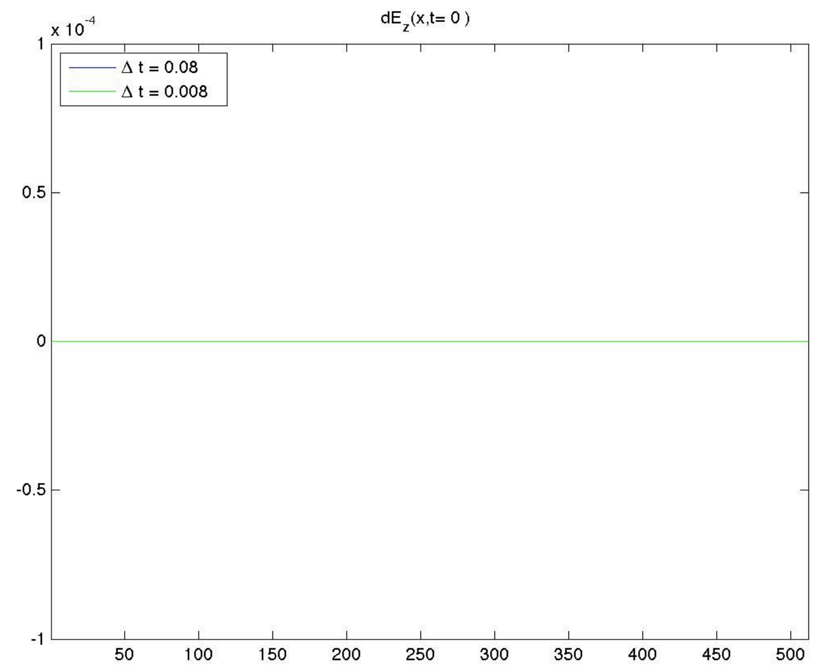




- dEz is calculated from the difference between a wave moving in a 512 cell box with 20 PML cells (493-512) and a 1024 box with no PML ("perfectly absorbing").

Reducing the Time Step Results in Roughly the Same Results





END

The PML Arrays Overlay the Simulation Box

- Everywhere except the red cells are copied to the box.
- It’s actually a bug that a whole column is being copied to the x PML, but it gets overwritten by the y PML right away so I guess it doesn’t matter..