ECE 742 Final Project

Michelle King Suraj Suri

February 1, 2019

1 Theory

1.1 PML

Perfectly Matched Layer (PML) boundary conditions are absorbing boundary conditions. PML BCs decay the wave within a boundary layer at the edge of the simulation. The edge of the simulation BC can be implemented as PEC. Well-implemented PML BCs completely decay the wave from the time it enters the boundary layer to the time after it reflects and attempts to leave.

1.2 Finite-Difference Derivation

Let's start with equation:

$$\nabla \times \vec{H} = j\omega\epsilon\bar{\bar{s}}\vec{E} \tag{1}$$

Evaluate the cross product and write in matrix form:

$$\begin{bmatrix} \frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z} \\ \frac{\partial H_x}{\partial z} - \frac{\partial H_z}{\partial x} \\ \frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y} \end{bmatrix} = \mathrm{j}\omega\epsilon \begin{bmatrix} \frac{s_y s_z}{s_x} & 0 & 0 \\ 0 & \frac{s_x s_z}{s_y} & 0 \\ 0 & 0 & \frac{s_x s_y}{s_z} \end{bmatrix} \begin{bmatrix} E_x \\ E_y \\ E_z \end{bmatrix} \tag{2}$$

Where the values in the second rank tensor can be described by:

$$s_x = \kappa_x + \frac{\sigma_x}{j\omega\epsilon_0} \tag{3}$$

$$s_y = \kappa_y + \frac{\sigma_y}{j\omega\epsilon_0} \tag{4}$$

$$s_z = \kappa_z + \frac{\sigma_z}{j\omega\epsilon_0} \tag{5}$$

To make the calculation computationally more managable, we can define the following relations:

$$D_x = \epsilon\frac{s_z}{s_x}E_x \tag{6}$$

$$D_y = \epsilon \frac{s_x}{s_y} E_y \tag{7}$$

$$D_z = \epsilon \frac{s_y}{s_z} E_z \tag{8}$$

such that now:

$$\begin{bmatrix} \frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z} \\ \frac{\partial H_x}{\partial z} - \frac{\partial H_z}{\partial x} \\ \frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y} \end{bmatrix} = \mathrm{j}\omega \begin{bmatrix} s_y & 0 & 0 \\ 0 & s_z & 0 \\ 0 & 0 & s_x \end{bmatrix} \begin{bmatrix} D_x \\ D_y \\ D_z \end{bmatrix} \tag{9}$$

Using the defined values of s and that $\frac{\partial}{\partial t} = j\omega$:

$$\begin{bmatrix} \frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z} \\ \frac{\partial H_x}{\partial z} - \frac{\partial H_z}{\partial x} \\ \frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y} \end{bmatrix} = \frac{\partial}{\partial t} \begin{bmatrix} \kappa_y & 0 & 0 \\ 0 & \kappa_z & 0 \\ 0 & 0 & \kappa_x \end{bmatrix} \begin{bmatrix} D_x \\ D_y \\ D_z \end{bmatrix} + \frac{1}{\epsilon_0} \begin{bmatrix} \sigma_y & 0 & 0 \\ 0 & \sigma_z & 0 \\ 0 & 0 & \sigma_x \end{bmatrix} \begin{bmatrix} D_x \\ D_y \\ D_z \end{bmatrix} \tag{10}$$

This has too much info. Since we are doing 2D, $\kappa_z = 1$, $\sigma_z = 0$ in this simulation. We are doing TM as well, so $D_x = D_y = 0$ and $H_z = 0$. Equation reduces to:

$$\begin{bmatrix} \frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z} \\ \frac{\partial H_x}{\partial z} - \frac{\partial H_z}{\partial x} \\ \frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y} \end{bmatrix} = \frac{\partial}{\partial t} \begin{bmatrix} \kappa_y & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \kappa_x \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ D_z \end{bmatrix} + \frac{1}{\epsilon_0} \begin{bmatrix} \sigma_y & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \sigma_x \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ D_z \end{bmatrix} \tag{11}$$

This leaves us with one equation to find an update quation for D_z:

$$\frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y} = \kappa_x \frac{\partial}{\partial t} D_z + \frac{\sigma_x}{\epsilon_0} D_z \tag{12}$$

Using semi-implicit definition

$$D^{n+1/2} = \frac{D^{n+1} + D^n}{2} \tag{13}$$

If we discretize around point i,j,k at timestep n
This equation discretizes to:

$$\begin{aligned} &\frac{1}{\Delta_x}(H_y^{n+1/2}(i+1/2,j) - H_y^{n+1/2}(i-1/2,j)) \\ &\quad - \frac{1}{\Delta_y}(H_x^{n+1/2}(i,j+1/2) - H_x^{n+1/2}(i,j-1/2)) = \\ &\frac{\kappa_x}{\Delta t}(D_z^{n+1}(i,j) - D_z^n(i,j)) + \frac{\sigma_x}{2\epsilon_0}(D_z^{n+1}(i,j) + D_z^n(i,j)) \end{aligned} \tag{14}$$

We can solve this to find update equation for D:

$$\frac{1}{\Delta_x}(H_y^{n+1/2}(i+1/2,j) - H_y^{n+1/2}(i-1/2,j)) - \frac{1}{\Delta_y}(H_x^{n+1/2}(i,j+1/2) - H_x^{n+1/2}(i,j-1/2)) = (\frac{\sigma_x}{2\epsilon_0} + \frac{\kappa_x}{\Delta t})D_z^{n+1}(i,j) + (\frac{\sigma_x}{2\epsilon_0} - \frac{\kappa_x}{\Delta t})D_z^n(i,j) \tag{15}$$

$$\frac{1}{\Delta_x}(H_y^{n+1/2}(i+1/2,j) - H_y^{n+1/2}(i-1/2,j)) - \frac{1}{\Delta_y}(H_x^{n+1/2}(i,j+1/2) - H_x^{n+1/2}(i,j-1/2)) = (\frac{\sigma_x\Delta t + 2\epsilon_0\kappa_x}{2\epsilon_0\Delta t})D_z^{n+1}(i,j) + (\frac{\sigma_x\Delta t - 2\epsilon_0\kappa_x}{2\epsilon_0\Delta t})D_z^n(i,j) \tag{16}$$

$$D_z^{n+1}(i,j) = -\frac{\sigma_x - 2\epsilon_0\kappa_x}{\sigma_x\Delta t + 2\epsilon_0\kappa_x}D_z^n(i,j) + \frac{2\epsilon_0\Delta t}{\Delta_x(\sigma_x\Delta t + 2\epsilon_0\kappa_x)}(H_y^{n+1/2}(i+1/2,j) - H_y^{n+1/2}(i-1/2,j)) - \frac{2\epsilon_0\Delta t}{\Delta_y(\sigma_x\Delta t + 2\epsilon_0\kappa_x)}(H_x^{n+1/2}(i,j+1/2) - H_x^{n+1/2}(i,j-1/2)) \tag{17}$$

Update Components for E

Start with rewriting 8:

$$s_z D_z = \epsilon s_y E_z \tag{18}$$

Using 4 and 5:

$$(\kappa_z + \frac{\sigma_z}{j\omega\epsilon_0})D_z = \epsilon(\kappa_y + \frac{\sigma_y}{j\omega\epsilon_0})E_z \tag{19}$$

Take the partial time derivative and use $\frac{\partial}{\partial t} = j\omega$

$$\frac{\partial}{\partial t}(\kappa_z D_z) + \frac{\sigma_z}{\epsilon_0}D_z = \epsilon(\frac{\partial}{\partial t}(\kappa_y E_z) + \frac{\sigma_y}{\epsilon_0}E_z) \tag{20}$$

Discretize this to find update equation for E:

$$E_z^{n+1}(i,j) = \frac{2\epsilon_0\kappa_y - \sigma_y\Delta t}{2\epsilon_0\kappa_y + \sigma_y\Delta t}E_z^n(i,j) + \frac{1}{2\epsilon_0\kappa_y + \sigma_y\Delta t}((2\epsilon_0\kappa_z + \sigma_z\Delta t)D_z^{n+1}(i,j) - (2\epsilon_0\kappa_z - \sigma_z\Delta t)D_y^n(i,j))/\epsilon \tag{21}$$

Similarly, we can find the B update equations to be:

$$B_x^{n+1/2}(i,j) = \frac{2\epsilon_0\kappa_y - \sigma_y\Delta t}{2\epsilon_0\kappa_y + \sigma_y\Delta t}B_x^{n-1/2}(i,j) + \frac{2\epsilon_0\Delta t}{2\epsilon_0\kappa_y + \sigma_y\Delta t}(E_z^n(i,j+1) - E_z^n(i,j))/dy \tag{22}$$

$$B_y^{n+1/2}(i,j) = \frac{2\epsilon_0\kappa_z - \sigma_z\Delta t}{2\epsilon_0\kappa_z + \sigma_z\Delta t}B_y^{n-1/2}(i,j) + \frac{2\epsilon_0\Delta t}{2\epsilon_0\kappa_z + \sigma_z\Delta t}(E_z^n(i+1,j) - E_z^n(i,j))/dx \tag{23}$$

And we can find the H update equations to be:

$$H_x^{n+1}(i,j) = \frac{2\epsilon_0\kappa_z - \sigma_z\Delta t}{2\epsilon_0\kappa_z + \sigma_z\Delta t}H_x^n(i,j) + \frac{1}{2\epsilon_0\kappa_z + \sigma_z\Delta t}((2\epsilon_0\kappa_x + \sigma_x\Delta t)B_x^{n+1}(i,j) - (2\epsilon_0\kappa_x - \sigma_x\Delta t)B_x^n(i,j))/\mu \tag{24}$$

$$H_y^{n+1}(i,j) = \frac{2\epsilon_0\kappa_x - \sigma_x\Delta t}{2\epsilon_0\kappa_x + \sigma_x\Delta t}H_y^n(i,j) + \frac{1}{2\epsilon_0\kappa_x + \sigma_x\Delta t}((= 2\epsilon_0\kappa_y + \sigma_y\Delta t)B_y^{n+1}(i,j) - (2\epsilon_0\kappa_y\sigma_y\Delta t)B_y^n(i,j))/\mu \tag{25}$$

1.3 Code Implementation

1.3.1 Update Equations

Using the derivation above, we can update field values in a loop. It is computationally more efficient to calculate the coefficients for these field values as an array outside of the loop. This gives us 18 coefficients and their corresponding simplified field update equations:

$$Ez \tag{26}$$

$$CBX1 = \frac{2\epsilon_0\kappa_y - \sigma_y\Delta t}{2\epsilon_0\kappa_y + \sigma_y\Delta t} \tag{27}$$

$$CBX2 = \frac{2\epsilon_0\Delta t}{2\epsilon_0\kappa_y + \sigma_y\Delta t} \tag{28}$$

$$CHX1 = \frac{2\epsilon_0\kappa_z - \sigma_z\Delta t}{2\epsilon_0\kappa_z + \sigma_z\Delta t} \tag{29}$$

$$CHX2 = \frac{1}{2\epsilon_0\kappa_z + \sigma_z\Delta t} \tag{30}$$

$$CHX3 = 2\epsilon_0\kappa_x + \sigma_x\Delta t \tag{31}$$

$$CHX4 = 2\epsilon_0\kappa_x - \sigma_x\Delta t \tag{32}$$

$$CBY1 = \frac{2\epsilon_0\kappa_z - \sigma_z\Delta t}{2\epsilon_0\kappa_z + \sigma_z\Delta t} \tag{33}$$

$$CBY2 = \frac{2\epsilon_0\Delta t}{2\epsilon_0\kappa_z + \sigma_z\Delta t} \tag{34}$$

$$CHY1 = \frac{2\epsilon_0\kappa_x - \sigma_x\Delta t}{2\epsilon_0\kappa_x + \sigma_x\Delta t} \tag{35}$$

$$CHY2 = \frac{1}{2\epsilon_0\kappa_x + \sigma_x\Delta t} \tag{36}$$

$$CHY3 = 2\epsilon_0\kappa_y + \sigma_y\Delta t \tag{37}$$

$$CHY4 = 2\epsilon_0\kappa_y - \sigma_y\Delta t \tag{38}$$

$$CDZ1 = \frac{2\epsilon_0\kappa_x - \sigma_x\Delta t}{2\epsilon_0\kappa_x + \sigma_x\Delta t} \tag{39}$$

$$CDZ2 = \frac{2\epsilon_0\Delta t}{2\epsilon_0\kappa_x + \sigma_x\Delta t} \tag{40}$$

$$CEZ1 = \frac{2\epsilon_0\kappa_y - \sigma_y\Delta t}{2\epsilon_0\kappa_y + \sigma_y\Delta t} \tag{41}$$

$$CEZ2 = \frac{1}{2\epsilon_0\kappa_y + \sigma_y\Delta t} \tag{42}$$

$$CEZ3 = 2\epsilon_0\kappa_z + \sigma_z\Delta t \tag{43}$$

$$CEZ4 = 2\epsilon_0\kappa_z - \sigma_z\Delta t \tag{44}$$

Some of these coefficients are identical, but we stored them separately for the purpose of organization and readable naming scheme.

1.3.2 Graded PML

The purpose of PML is to have the wave decay as the wave enters the PML, and a nonzero conductivity value will achieve this decay. However, a large discrepancy between the values of conductivity in the simulation region and the PML can result in unwanted reflections.

The Reflection Factor is given by the equation:

$$R(\theta) = \exp^{-2\eta cos(\theta)\int_0^d \sigma_x(x)dx} \tag{45}$$

Where σ_x is the graded conductivity of the PML material.
θ is the angle of incidence of the wave. So steeper angles of θ will result in higher values of reflection error.

We want to minimize reflection R but also make sure the wave decays completely within the PML.

1.3.3 Polynomial grading

One type of grading, and the one we have implemented, is the polynomial grading. We can describe the grading in our code by the factors σ and κ Where the graded conductivity for the PML in the x direction is:

$$\sigma_x = (\frac{x}{d})^m \sigma_{x,max}$$

And the graded value for κ for the PML in the x direction is:

$$\kappa_x = 1 + (\kappa_{x,max} - 1)(\frac{x}{d})^m$$

We will vary the values for κ_{max} and σ_{max} in our numerical experiments to test how different PML material conditions affect the effectiveness of the PML.

An optimal expression for the value of σ_{max} for a polynomial grading has been found numerically to be:

$$\sigma_{x,optimal} = \frac{0.8(m+1)}{\eta_0 \Delta x} \tag{46}$$

2 Experiment

We will test the effectiveness of our PML implementation by launching a gaussian pulse at the center of our free-space simulation. The peak of the gaussian will be at the 10th timestep, and the width of the gaussian is at the 5th. Our total simulation size is size 100 by 100 with a PML boundary layer of size 20. We ran all simulations over a maximum timestep size of 300.

Figures 1, 2, 3, and 4 show the initial pulse, the initial propegation of the wave, the wave hitting the PML, and the dissipation of the field after it hits the PML. These figures were generated for the conditions of m=3, σ_{max}=1, and κ_{max}=1.

We varied values of σ_{max}, maximum value for the conductivity, in experiment. Varying σ_{max} will change the properties of the PML and how it absorbs or reflects. Too high of sigma will result in reflections from the PML layer. Too small of sigma, and the wave fails to be absorbed by the PML layer.

We varied the value of m, a parameter in the polynomial grading. Varying values of m changes the steepness of the grading and will affect the amount of reflection from the PML layer.

3 Error Analysis

In order to test how well the PML is working to absorb the incoming reflection, we will calculate the relative error with respect to the case of no boundary conditions. Error values were calculated for the amount of the wave within the simulation region, not including the wave present in the PML.

Our figures show values of relative error with respect to timstep. The initial low to high increase in each plot at early timstep is when the generated wave is propegating in free space. The first peak is a result of the entire wave being inside the simualtion. After the wave hits the boundary layer, the total decreases, and the wave is absorbed by the boundary layer. If we look in figure 6, we can see that the relative error is decreasing for all valus of m, but appears to be lowest for m=1, 2, and 3 respectively. This is to be expected because our textbook claimed m=2,3 were optimal values.

In figure 5, the lowest amount of error we have are for σ_{max} are for the cases of10^0 and 10^1 (also 10^2). This is the middle range of conductivity, and we could see as higher levels of conducivity, we had a lot of reflection, and for lower levels of conductivity, not much of the field was absorbed.

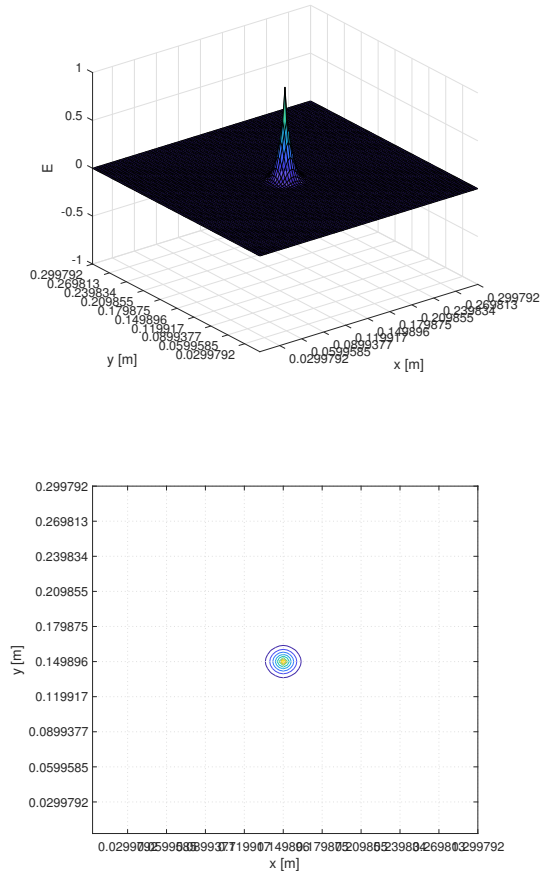

Figure 1: Formation of the Gaussian Pulse

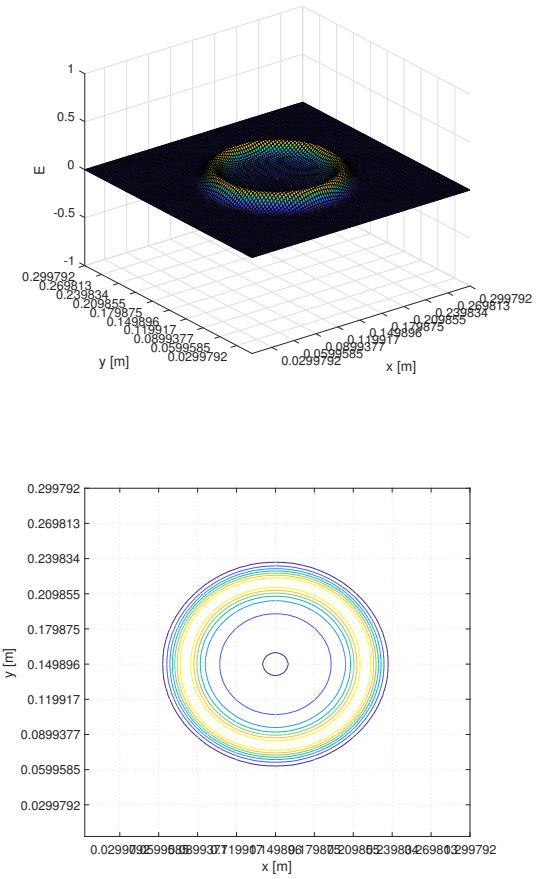

Figure 2: Propegation of the Gaussian Pulse

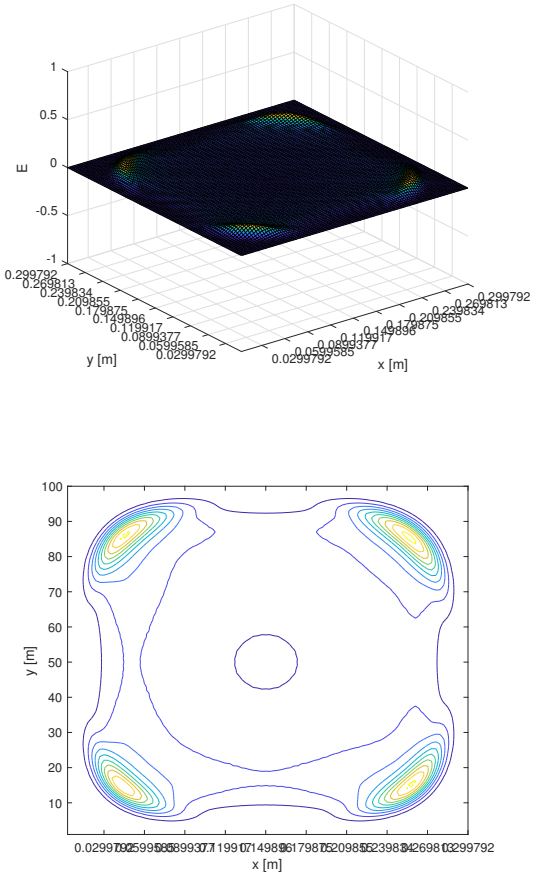

Figure 3: Nonzero field values interacting with PML layer

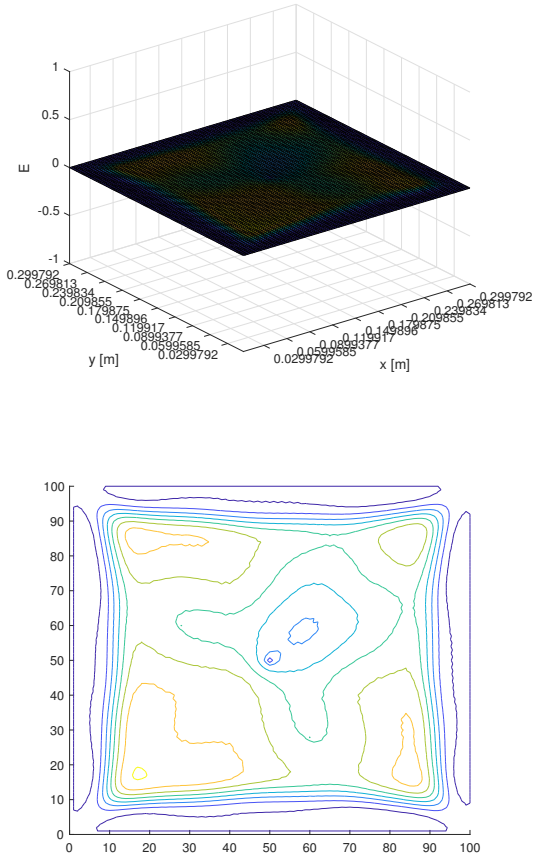

Figure 4: Absorption of Field by PML

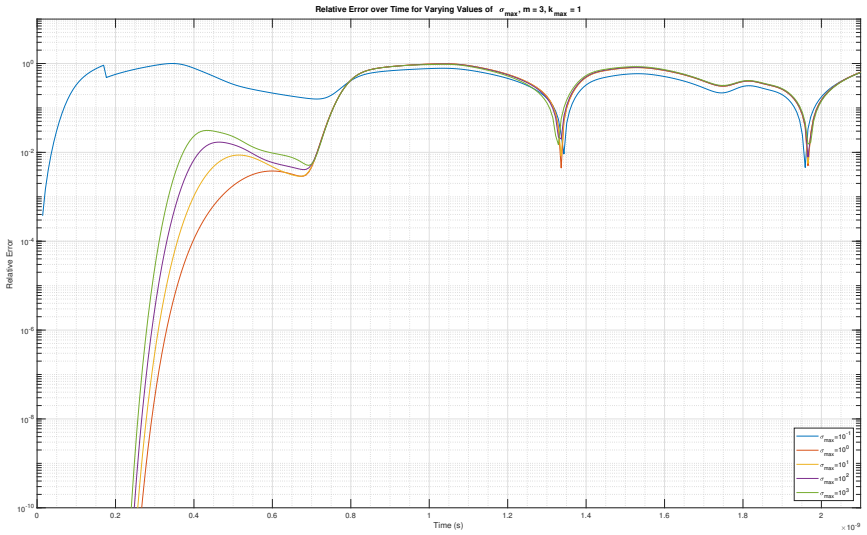

Figure 5: Plot of Relative Error with respect to time for values of σ_{max} of $10^{-1}, 10^0, 10^1, 10^2, 10^3$, m=3, $\kappa max = 3$

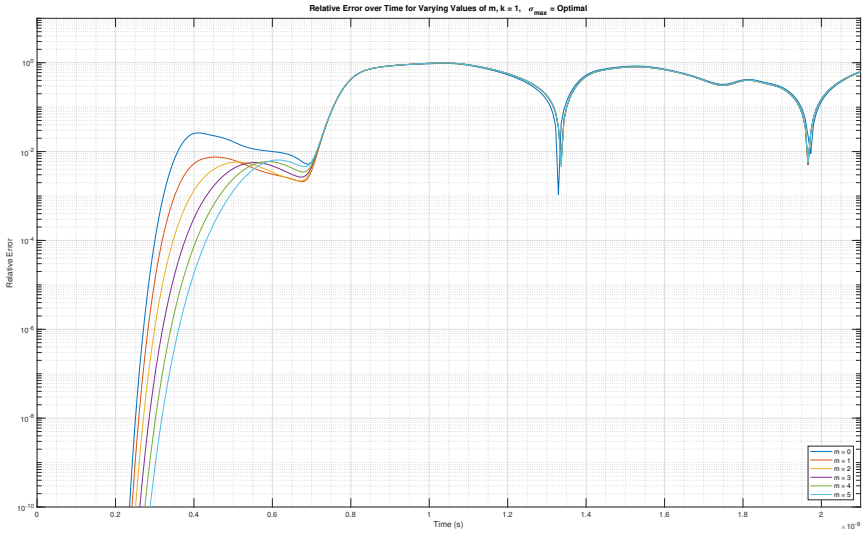

Figure 6: Plot of Relative Error with respect to time for values of m of 0, 1, 2, 3, 4, 5

4 Resources

Followed Beringer's Derivation in Susan's book - third edition