Assignment 2

ECSE 543

Professor: Prof. Dennis  Giannacopoulos

Student Name: Chris Morin

Student Number: 260344722

Question 2

I selected top right quarter (quadrant 1) of the coax cross section as my mesh location. Instead of using the mirroring method to deal with the Neumann boundaries, I turned them into Dirichlet boundaries by computing their exact values via direct integration.

1. The file that generates the input file to SIMPLE2D is “meshGen.py”. In order to keep things simple, I ignored the bottom row of points so that my mesh was a rectangle with no “holes” (missing mesh points) when writing my program. Once the input file was created, I added the extra few mesh points ignored by hand. Although this isn’t a super clean way of doing it, it makes it very easy and fast to implement. See “fileForSimple2D.txt” for the results.
2. The output from SIMPLE2D can be found in “simple2Dresults.txt”. The potential at (x,y) = (0.06, 0.04) is 4.26483V.
3. The equation describing capacitance can be derived from the energy in a capacitor equation and the energy in a potential field equation. By equating and rearranging these two equations, we can find the following:

We compute the energies of the individual elements by using the equations developed in part a (expanding on them) and adding them all up. We were able to find the energy in a square when the potential at the corners is known. The equation describing (proportional to the energy) of said square is:

Note that this all assumes a depth of one unit. We found S in part a and is just a vector containing the four voltages in order. Assuming u1, u2, u3, u4 to the the potentials at the corners of the square in the assignment handout, is equal to u1^2 - u1\*u2 - u1\*u4 + u2^2 - u2\*u3 + u3^2 - u3\*u4 + u4^2. After finding, we plug it into our equation from above. Note that we need to multiply our final answer by four as our mesh only covers one quarter of the whole coax cable. We can see this as there bring 4 times the energy or that capacitors in parallel add.

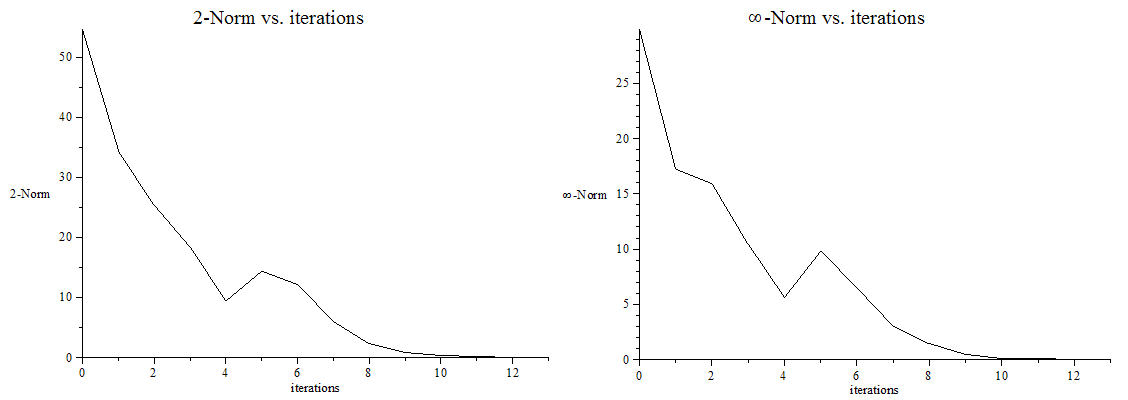
The capacitance of the coax cable per unit length turns out to be:

5.258e-11 F/m

Question 3

For this question, I used the “Matrices.py” class I created for assignment 1 as this question required much matrix algebra. I modified it to add some additional functionality like the ability to add and subtract matrices. I also implemented operator overloading so that instead of calling methods, I could just use the familiar “+, - and \*” operators to make my code more familiar. I wrote a new source file called “ConjugateGradient.py”. This file was based on “FiniteDiffSolver.py” from assignment 1. The functionality of this file is to create the matrix representing the mesh equations along with the “b” vector and then solve the Ax=b equation by either Choleski decomposition (functionality in Matrices file) or by the conjugate gradient method.

1. The Choleski decomposition failed as the algorithm attempted to take the square root of a negative number. This means that the original matrix wasn’t positive definite. To remedy this, we can multiply both “A” and “b” by “AT”. This retains the equality, ensuring the solution to the equation remains x, while making a new positive definite matrix that will allow our algorithms to function properly.
2. This is done in “ConjugateGradient.py”. The results can be seen in “conjugateGradientResults.txt”. It can be noted that the solutions are the same.
3. Here are the requested graphs:





|  |  |
| --- | --- |
| Method used | Potential at (x,y) = (0.06, 0.04) |
| Assignment 2: Choleski decomposition | 4.26481 |
| Assignment 2: Conjugate gradient | 4.26481 |
| Assignment 2: SIMPLE2D | 4.26483 |
| Assignment 1: SOR | 4.26483 |

The values match up very well as expected. The reason for this is that all methods approximate a solution to the Ax=b solution very closely and the true solution is the same regardless of the method used given identical meshes (node positions).

1. This can be done the exact same way that it was done in question 2. We must simple map each of the potential values to a position in the mesh and then run the program.