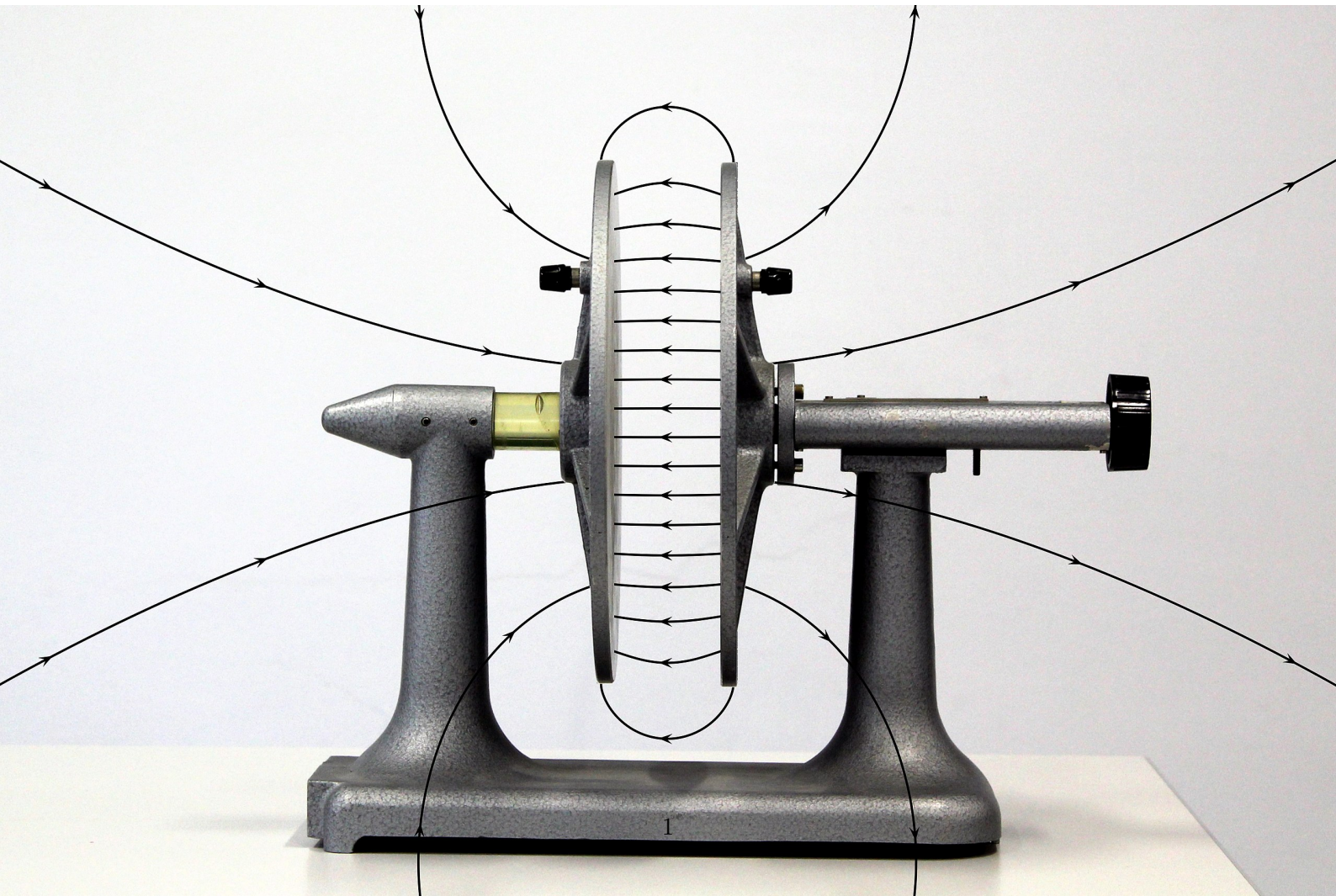
Electromagnetism Lab

Simulation of the Potential of a Finite Parallel Plate Capacitor

Bernat Frangi Mahiques (100454602)

Engineering Physics



Contents

1. Objectives

The aim of this lab is to simulate the potential generated around a charged, rectangular, finite parallel plate capacitor, situated inside a closed rectangular box, by using the numerical method of *relaxation of potential*.

2. Introduction

2.1 Finite and Infinite Parallel Plate Capacitor

A ***finite*** *parallel plate capacitor (ppc)* is formed of two finite parallel plates that are separated a distance d. On the other hand, an ***infinite*** *ppc* is formed of two *infinite* plates (meaning that they extend to infinity on all sides) separated a distance d.

In the case of the *infinite* ppc, the potential generated is uniform in all planes parallel to the plates of the capacitor. On the other hand, in the *finite* ppc, a set of *fringing fields appear in the sides of the capacitor*. The electric field is no longer uniform in the planes parallel to the plates of the ppc. These fringing fields are shown in **Figure 1**.

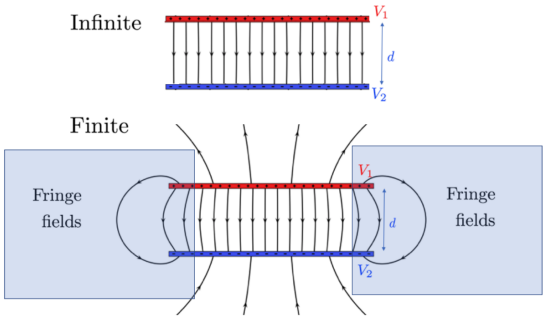


Figure 1: Difference Between a Finite and an Infinite Parallel Plate Capacitor

2.2 Description of the Problem

In *electrostatics*, the field $\boldsymbol{E}(\boldsymbol{r})$ that occupies an area of space, must satisfy *Gauss' Law*, which in differential form is:

$$\nabla \cdot \boldsymbol{E} = \frac{\partial E_x}{\partial x} + \frac{\partial E_y}{\partial y} + \frac{\partial E_z}{\partial z} = \frac{\rho(\boldsymbol{r})}{\varepsilon_0} \tag{2.1}$$

where $\rho(\boldsymbol{r})$ is the charge density present in the space. If $\rho(\boldsymbol{r}) = 0$, then we obtain:

$$\frac{\partial E_x}{\partial x} + \frac{\partial E_y}{\partial y} + \frac{\partial E_z}{\partial z} = 0 \tag{2.2}$$

Furthermore, as *the electrostatic field is conservative*, we have the relation:

$$\boldsymbol{E}(\boldsymbol{r}) = -\nabla V(\boldsymbol{r}) \tag{2.3}$$

If we substitute this into **Equation 2.2**, we obtain *Laplace's Equation*:

$$\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} + \frac{\partial^2 V}{\partial z^2} = 0 \tag{2.4}$$

In this case, we will solve *Laplace's Equation*, as we will consider that the only place where there is charge in our distribution is in the plates of the capacitor, so $\rho(\boldsymbol{r}) = 0$ everywhere except at the plates. At the plates, we will impose constant potential. The other boundary condition that we will impose is that the potential at the walls of the box where the capacitor is situated is zero.

2.3 Discretization of the Problem

As we are solving a differential equation numerically in a computer, we must consider all three dimensions (x, y and z) as discrete variables that form a spatial $3D$ **mesh**. At each point (j, k, l) of the mesh, the value of the potential is given for the x-position corresponding to the index j, the y-position corresponding to the index k and the z-position corresponding to the index l.

- **x-Position:** it is a discrete variable that will take the values

  $$x_{min},\ x_{min} + \Delta x,\ x_{min} + 2\Delta x,\ ...,\ x_{min} + (M_x - 1)\Delta x,$$

  where $x_{min} + (M_x - 1)\Delta x = x_{max}$. Therefore, there will be M_x spatial points x_j $(j = 0, 1, ..., M_x - 1)$ and the mesh will have M_x rows in the first dimension.

- **y-Position:** it is a discrete variable that will take the values

  $$y_{min},\ y_{min} + \Delta y,\ y_{min} + 2\Delta y,\ ...,\ y_{min} + (M_y - 1)\Delta y,$$

  where $y_{min} + (M_y - 1)\Delta y = y_{max}$. Therefore, there will be M_y spatial points y_k $(k = 0, 1, ..., M_y - 1)$ and the mesh will have M_y rows in the second dimension.

- **z-Position:** it is a discrete variable that will take the values

  $$z_{min},\ z_{min} + \Delta z,\ z_{min} + 2\Delta z,\ ...,\ z_{min} + (M_z - 1)\Delta z,$$

  where $z_{min} + (M_z - 1)\Delta z = z_{max}$. Therefore, there will be M_z spatial points z_l $(l = 0, 1, ..., M_z - 1)$ and the mesh will have M_z rows in the third dimension.

Once we have defined the geometry of the spatial mesh, we can easily store the values of the potential in that mesh, where we have a $j \times k \times l$ matrix with elements $V_{jkl} \equiv V(x_j, y_k, z_l)$ for $j = 0, 1, ..., M_x - 1$, $k = 0, 1, ..., M_y - 1$ and $l = 0, 1, ..., M_z - 1$.

2.4 Discretization of the Derivatives

We use the approach known as finite differences in order to calculate the three second order derivatives of the potential.

- **Second Derivative with Respect to x:** We can approximate the second derivative of the potential with respect to x at each point (j, k, l) of the spatial mesh using the expression:

$$\frac{\partial^2 V(x_j, y_k, z_l)}{\partial x^2} \equiv \left(\frac{\partial^2 V}{\partial x^2}\right)_{jkl} = \frac{V_{j+1,kl} - 2V_{jkl} + V_{j-1,kl}}{(\Delta x)^2} \tag{2.5}$$

- **Second Derivative with Respect to y:** Likewise:

$$\frac{\partial^2 V(x_j, y_k, z_l)}{\partial y^2} \equiv \left(\frac{\partial^2 V}{\partial y^2}\right)_{jkl} = \frac{V_{j,k+1,l} - 2V_{jkl} + V_{j,k-1,l}}{(\Delta y)^2} \tag{2.6}$$

- **Second Derivative with Respect to z:** Likewise:

$$\frac{\partial^2 V(x_j, y_k, z_l)}{\partial z^2} \equiv \left(\frac{\partial^2 V}{\partial z^2}\right)_{jkl} = \frac{V_{jk,l+1} - 2V_{jkl} + V_{jk,l-1}}{(\Delta z)^2} \tag{2.7}$$

2.5 Discretized Laplace Equation

If we now replace the derivatives in *Laplace's Equation* with the approximations, we obtain:

$$\frac{V_{j+1,kl} - 2V_{jkl} + V_{j-1,kl}}{(\Delta x)^2} + \frac{V_{j,k+1,l} - 2V_{jkl} + V_{j,k-1,l}}{(\Delta y)^2} + \frac{V_{jk,l+1} - 2V_{jkl} + V_{jk,l-1}}{(\Delta z)^2} = 0 \tag{2.8}$$

In this case, we are using $\Delta x = \Delta y = \Delta z = \Delta s$, so we can reduce this equation to:

$$V_{jkl} = \frac{V_{j+1,kl} + V_{j-1,kl} + V_{j,k+1,l} + V_{j,k-1,l} + V_{jk,l+1} + V_{jk,l-1}}{6} \tag{2.9}$$

This basically means that the value of the potential at each point must be equal to the average of the potential over its six closest neighbours (two on either side, one in front, one behind, one above and one below). These six points, as well as the point (j, k, l), located at the center of the other six, form what is called the **local stencil**, which is characterized by giving the location of the central node. The *local stencil* is shown in **Figure 2**.

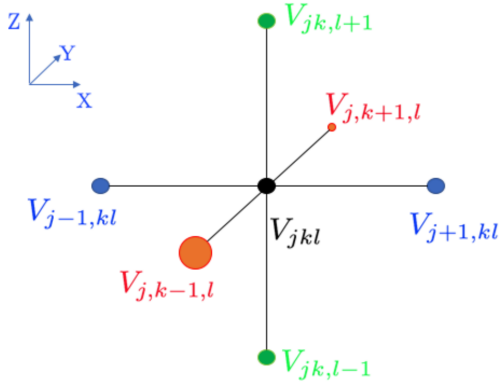


Figure 2: Local Stencil Formed by the Central Node and the 6 Closest Neighbours

2.6 Integration Method: Successive Relaxation

In order to find the numerical solution to this problem, we use the successive relaxation method. The process is as follows:

1. Define an initial value for the electrostatic potential in each node of the $3D$ mesh. Take into account the boundary conditions, and set the potential at the rest of the nodes to zero (or any value as close to the final value as possible, to ensure faster convergence).
2. Iterate over the mesh replacing the value of the potential at each node by the average over its *local stencil*. If the node corresponds to a point in the boundary of the box or in the plates of the capacitor, the value must not be modified (boundary conditions must be maintained).
3. This process is repeated over the whole mesh several times until either:
   (a) The maximum *local residual*, which is the difference between the potential at some node over two consecutive iterations, is smaller than some set tolerance.
   (b) The maximum *local residual* does not change in two consecutive runs of the process on the whole mesh.

2.6.1 Implementation of the Algorithm

There are different ways of implementing this algorithm programmatically. The choice of method is important in order to program efficient code. During the course of programming this simulation, I tried different methods including (from least efficient to most efficient):

- **Using `for` loops or list comprehension** (computation time $\sim$ 45 min): Here, the code iterates through each element of the matrix and replaces it with the arithmetic

average of its neighbours. Note that here, each time an element is updated, its new value is used to compute the average of the next adjecent element. This means faster convergence to the solution, but the `for` loops are very inefficient.

- **Changing python lists to `numpy` arrays and adding `numba` decorators** (computation time $\sim$ 6 min): `numba` is a library that translates python code to optimized machine code, which is more efficient and runs faster.

- **Removing `numba` decorators and changing the method from loops to matrix operations using `numpy` arrays** (computation time $\sim$ 30 sec): `numpy` arrays are much more efficient and take up less memory than python lists, so this speeds up the simulation considerably. However, the most important change here is the matrix method. Instead of iterating over the individual points of the potential mesh and replacing them with the local stencil average via loops, it is all done in one go by using matrix operations. The new potential mesh is obtained by adding up six different instances of the old mesh, each with the elements shifted one position in each direction, and dividing the resulting matrix by 6. The only problem here is that the average is calculated at each point using only values of the old mesh, so the convergence to the solution is slightly slower, meaning that we would need more iterations to obtain the same solution as with the loop method. This is compensated, however, by the much faster simulation times.

- **Changing `numpy` arrays to `pytorch` tensors** (computation time: 9 sec): `pythorch` tensors are even more efficient than `numpy` arrays in this case.

- **Moving `pytorch` tensors to the *GPU*** (computation time $\sim$ 5 sec): In this improvement, `pytorch` checks whether there is a *GPU* available and, if there is, it transfers the tensors to it. *GPUs* in modern computers can make calculations even faster than the *CPU*, improving further the speed of the simulation.

As mentioned before, the *matrix method* converges slower to the solution. This means that, using the same residual tolerance in both methods leads to a different actual error in the solution. This can be understood with the qualitative plot of **Figure 3**, which describes the convergence of each method to the real solution. The residual is the difference between two consecutive approximations of the solution, so it can be understood more or less as the absolute value of the slope of the graph of **Figure 3**. As can be seen, the slope (and therefore, the residual) is decreasing exponentially to zero. So, in the graph, we can interpret that the method stops when the absolute value of the slope of the curve has decreased to a certain value d. As the *matrix method* is slower-decreasing, it will reach the point $Slope = d$ at a point that is higher up in the graph than the point where the *loop method* has $Slope = d$. Thus, one same stop condition for both methods will lead to different closeness to the real solution in the end. Specifically, there will be more error in the *matrix method*.

I have been able to observe this difference in results when comparing my results to those provided in the guide. There are a few notable differences, especially in the plot of

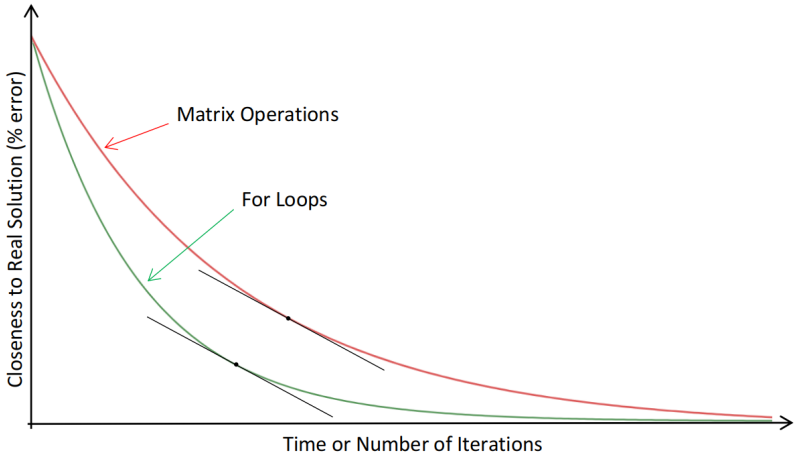


Figure 3: Time-Evolution of the Maximum Local Residual for the Matrix and Loop Methods

Figure 8. I also observed that using a fixed number of about 320 relaxations in the matrix method gives very similar results to those of the guide, taking only about 6 sec to compute the simulation. However, for the sake of following the guide, and since it was not really specified whether the loop method or the matrix method were to be used, I left the same residual condition as was indicated, which resulted in about 242 relaxations.

3. Results and Analysis

The code for this lab can be found in this is GitHub repository. All tasks have been solved in just one file called `simulation.py`. Another file named `initial_parameters.py` has been used to store all the initial parameters of the main simulation done in **Section 3.1**.

Note: in order to use the python programs, the following python libraries should be manually installed: `matplotlib`, `numpy` *and* `pytorch` *(instructions on how to do this are in the GitHub repo* `README.md`*). Also, this project has been made using* `python 3.8.5`.

3.1 Plot of the Potential Along Selected Axes

In this section, the a plot has been produced showing the variation of the electrostatic potential along the straight lines, parallel to the x direction, that cross through:

- the center of the plates O
- the mid-point of the left side of the plates A
- the mid-point of the top side of the plates B.

This has also been plotted along with the potential of an *infinite* ppc, which is given by:

$$V(x) = \begin{cases} V_1, & x < -d/2, \\ V_2 + (V_1 - V_2) \cdot \left(\frac{d-2x}{2d}\right), & -d/2 \leq x \leq d/2, \\ V_2, & x > d/2. \end{cases} \tag{3.1}$$

3.1.1 Results

The resulting plot is shown in **Figure 4**:

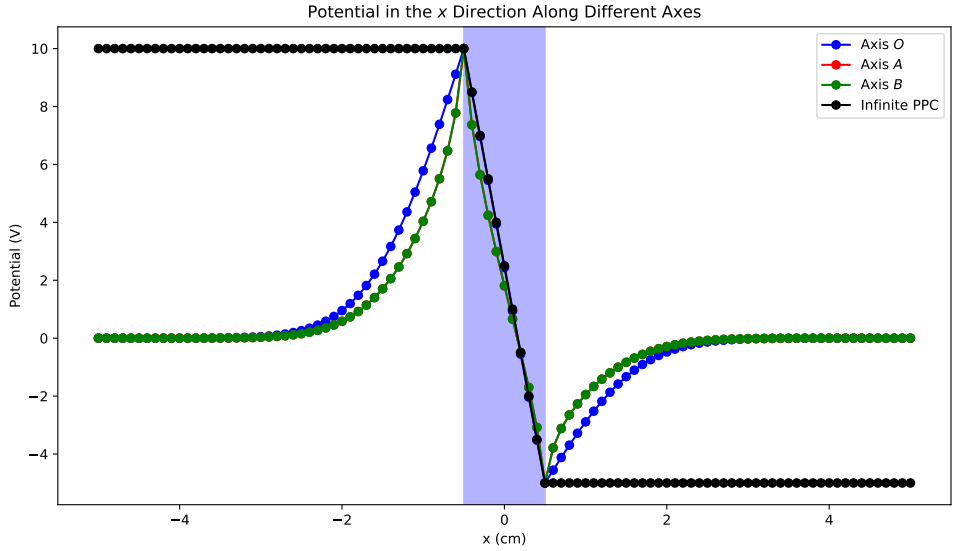


Figure 4: Potential in the x Direction Along the Selected Axes

In the plot, we can see the potential for the three selected axes O, A, and B, as well as the potential for the infinite capacitor. The potential for axis A is practically invisible, as it coincides almost perfectly with the potential along axis B.

In order to use the program `simulation.py` to generate the graph in **Figure 4**, the following steps should be followed:

1. Execute the program with `python3 simulation.py` while in the directory where the file `simulation.py` is stored.
2. You will be prompted to choose to either compute the potential mesh from scratch (takes about 9 to 12s) or import the mesh from a `.npy` file generated by `numpy`. if you have already computed and saved the mesh with this program, type `2` and press enter. If this is the first time you execute this program, type `1` and press enter.
3. Regardless of your previous choice, you will then be presented with a menu where you can choose between executing any of the 3 exercises of the lab guide. In this case, type `1` to choose *Exercise 1.*

4. You will then be given the option to save the figure that is going to be generated to your disk. You can choose to do so (type `Y` and press enter) or not (type `n` and press enter), but in the case you choose to export, you must make sure that you have a folder named `Plots` in the same directory where the `simulation.py` file is located, or you *will* get an error. This is the folder where the output files will be stored.

The initial parameters used for this simulation stored in `initial_parameters.py`.

3.1.2 Discussion

There are several similarities and differences between the potential of the *infinite* ppc and the *finite* ppc. We can distinguish two regions according to this distinction:

- **Outside Region:** The region where $|x| > d/2$ shows the biggest differences between the infinite and the finite ppcs.

  In the *infinite* ppc, the potential outside the capacitor is constant on each side and equal to the value of the potential at the capacitor plate of the corresponding side. This is consequence of the fact that in an infinite ppc the electric field outside the capacitor is zero, so there the potential does not change.

  On the other hand, as can be seen in **Figure 1**, the electric field outside a *finite* ppc is not zero, although the density of field lines *does* decrease as we get further away, which means that eventually the electric field is zero and the potential also approaches zero. We can also see this in our plot in **Figure 4**, where it is clear that the potential outside the capacitor is *not* constant, but rather it changes due to the action of the electric field until it reaches what we could call the "reference potential", which is the potential of the walls of the box (zero).

- **Inside Region:** The region where $|x| < d/2$ shows the biggest similarities between the infinite and the finite ppcs. Inside the *infinite* ppc, the field is uniform. However, inside the *finite* ppc, this is not exactly the case, especially at the extremes of the ppc plates, where fringing fields appear and distort the otherwise uniform-ish field present at the more central parts of the plates.

  However, in comparison with the outer region, the potential at the inner region is almost identical to that of the infinite ppc. Specifically, at the axis O, which is the one that traverses the central part of the capacitor, the potential of the inner region is identical to that of the infinite ppc. On the other axes (A and B), the potential is also very similar to that of the infinite ppc, but there are some clear distortions caused by the fringing fields (notice also, that between them, the potential curves for A and B are indistinguishable, as both are axes that pass through the edges of the capacitor).

3.2 Percentage Difference

In this section, the percentage difference between the potential for the infinite ppc and the finite ppc along the OY axis has been computed.

3.2.1 Results

The resulting graphs are the following:

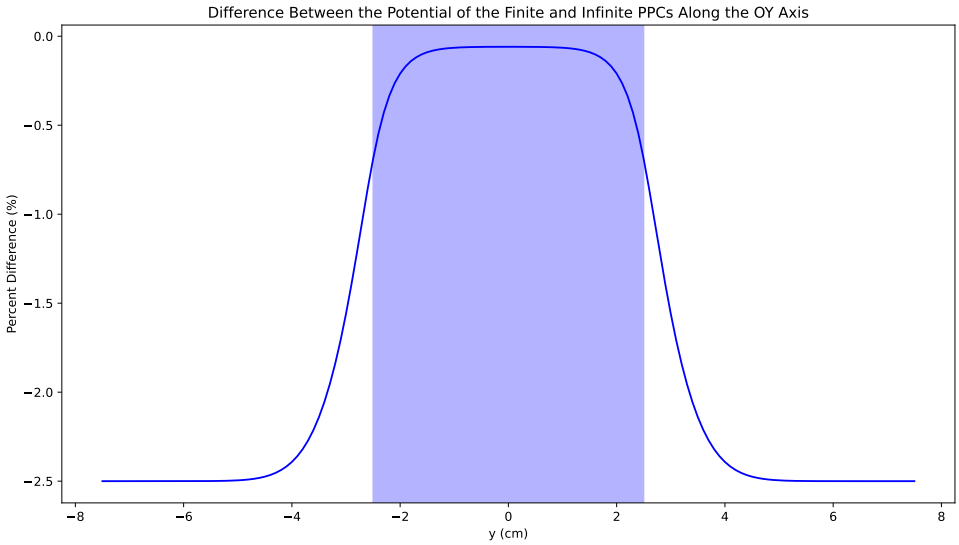


Figure 5: Difference Between the Potential of the Finite and Infinite PPCs Along the OY Axis

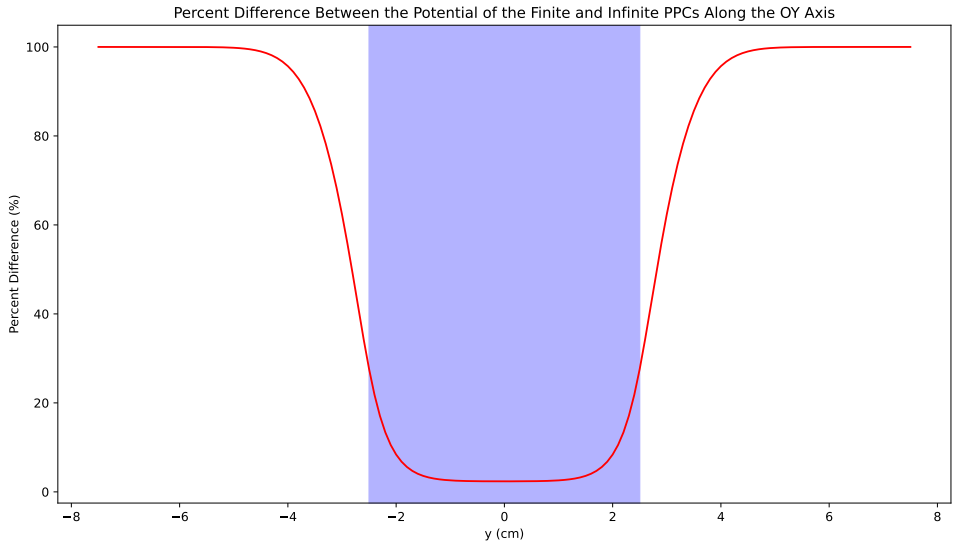


Figure 6: Percent Difference Between the Potential of the Finite and Infinite PPCs Along the OY Axis

In the first graph (**Figure 5**), we can see a plot of the difference between the potential of the finite and the infinite ppcs (finite minus infinite). In the second graph (**Figure 6**), we can see a plot of the percentage difference between the potential of the finite and the infinite ppcs (difference divided by infinite). Clearly, the biggest difference between both potentials is found outside the ppc, as seen also in **Section 3.1**.

It was also determined, using the program `simulation.py`, that *the distance from the origin along the Y axis beyond which the difference is larger than* 10% *is* 2.1 *cm* and *the percentage difference at the left edge of the finite capacitor is* 27.61%.

In order to use the program `simulation.py` to generate the graphs in **Figure 5** and **Figure 6**, the following steps should be followed:

1. Follow **Steps 1 to 3** of **Section 3.1**, but choose *Exercise 2* by typing `2` instead of `1` and pressing enter.

2. You will then be given the option to save the first figure that is going to be generated (**Figure 5**) to your disk. You can choose to do so (type `Y` and press enter) or not (type `n` and press enter), but in the case you choose to export, again you must make sure that you have a folder named `Plots` in the same directory where the `simulation.py` file is located, or you *will* get an error. This is the folder where the output files will be stored.

3. The first plot will appear. To carry on with the execution of the program, close it.

4. You will then be given the option to save the second figure that is going to be generated (**Figure 6**) to your disk. Do the same as before, and choose an option.

5. The second plot will appear. To carry on with the execution of the program, close it.

6. Once you have closed the second plot, two more lines will be printed on the console:

   - `The distance from the origin along the Y axis beyond which the difference is larger than 10 % is 2.1 cm`
   - `The percentage difference at the left edge of the finite capacitor is 27.61 %`

3.2.2 Discussion

In this section, we have further reinforced the point that the difference between the potentials of the infinite and finite ppcs is very low *inside* the ppc but increases very rapidly *outside* the ppc.

We can also see here in a clear way how the fringing fields distort the potential at the edges of the ppc, whereas in its center the potential is almost perfectly equal to that of an infinite ppc. If we want to consider this finite ppc as an infinite ppc, with a maximum error tolerance of 10% along its y-axis, we will need to work at a maximum distance of 2.1 cm from its center point in the y-axis direction.

3.3 Contour Plots

In this section, we will draw the level curves of the potential generated by the finite ppc on the XY plane, for potential levels from $-4\ V$ to $9\ V$, going up in steps of $1\ V$.

3.3.1 Results

The resulting graphs are the following:

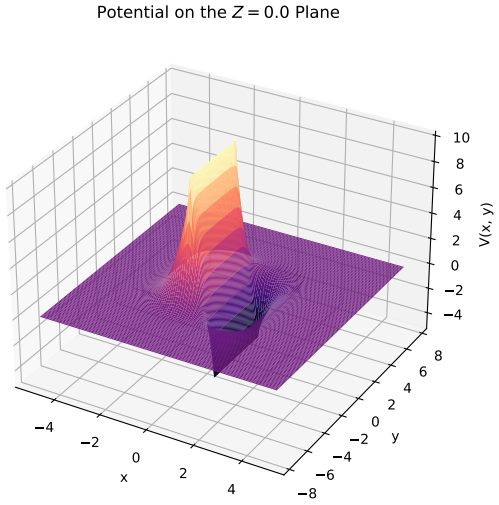


Figure 7: Potential on the $Z = 0$ Plane

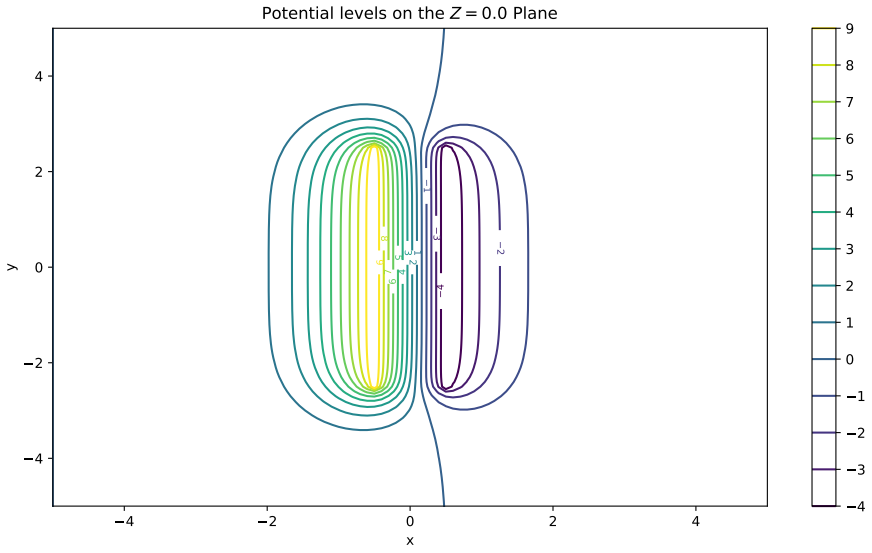


Figure 8: Potential Levels on the $Z = 0$ Plane

In **Figure 7**, we can see the $3D$ plot of the potential on the XY plane. The high peak corresponds to the capacitor plate with $V_1 = 10\ V$, and the low valley corresponds to the capacitor plate with $V_2 = -5\ V$.

3.3.2 Discussion

Looking at the plot from **Figure 8**, we can see that the place in the XY plane where the electric field is stronger is in between the two plates, as that is the place where the electric field lines, which are perpendicular to the potential level curves, would be closest together.

If we look at the XY plane but *outside* the capacitor, we see that the side where the electric field is greater is the left side, as it also has more density of potential level curves, and consequently more density of electric field lines. The reason why the electric field is stronger here is that, as the potential difference between the containing box and the plate is higher on the left plate, the potential has to drop faster to zero. Thus, its gradient, which gives the electric field, has a larger magnitude.

Figure 9 shows a few electric field lines sketched on top of the plot in **Figure 8**.

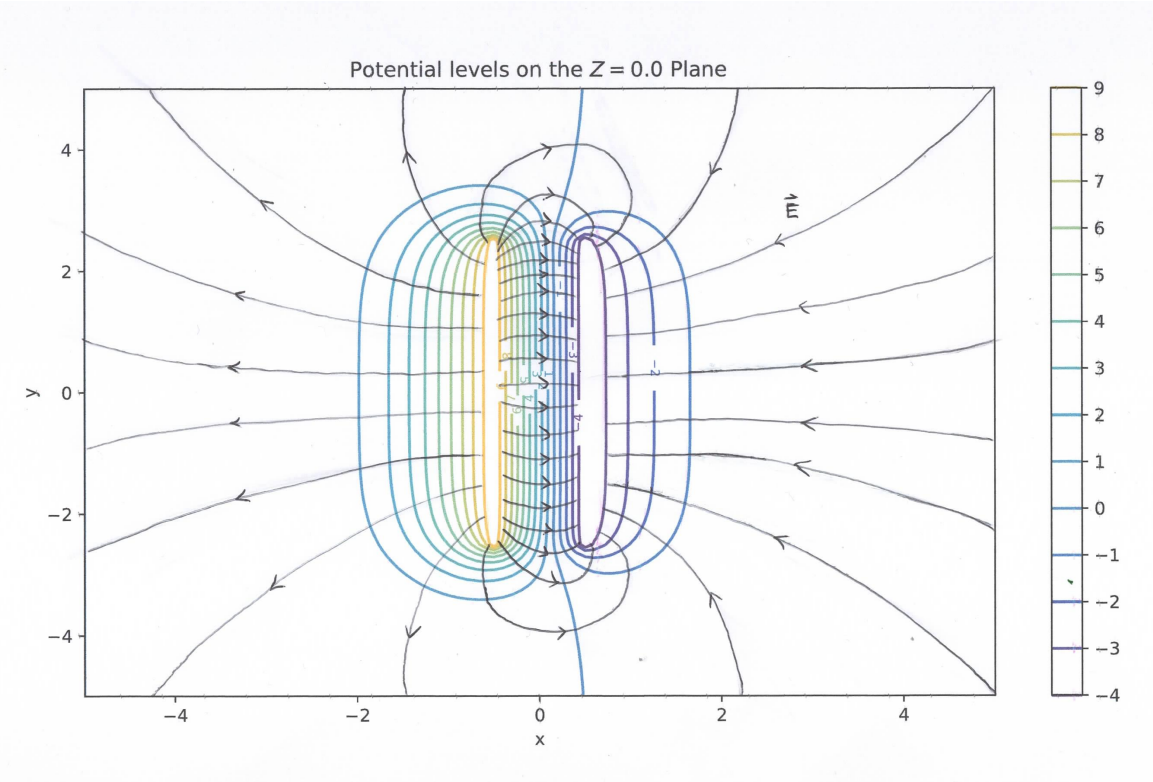


Figure 9: Potential Levels and Electric Field Lines for a Finite PPC

One interesting thing that we can see in the plot of **Figure 8** is that the contour line for $V = 0$ leaves the capacitor twisting towards positive x's instead of being parallel to the Y axis. The reason for this asymmetry is the fact that the potential on the left plate is much larger than the potential on the right plate (in absolute value). Therefore, we could say that the potential distribution is not symmetric. This causes the potential contour line of $V = 0$ to be asymmetric. For example, if the potential on the right plate was $-10\ V$ instead of $-5\ V$ (still keeping the left plate at $10\ V$), we would obtain the contour plot of **Figure 10**.

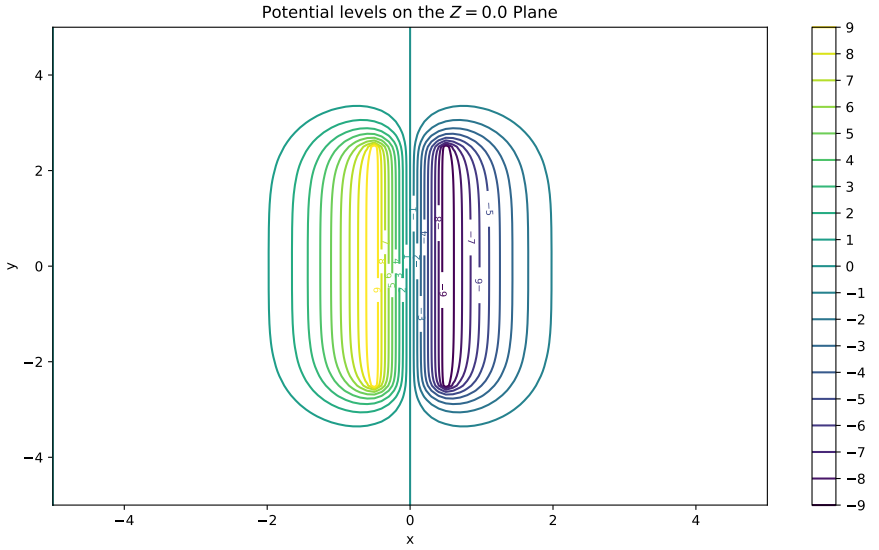


Figure 10: Potential Levels on the $Z = 0$ Plane for a PPC with $V_1 = 10\ V$ and $V_2 = -10\ V$

In fact, this symmetry is always relative to the reference potential that we choose, which is the potential of the box. Thus, we could also change the potential of the box to $V_{\text{box}} = \frac{V_1+V_2}{2} = 2.5\ V$, without changing those of the plates, and obtain the same effect (although now the vertical potential contour line will be at $2.5\ V$ instead of at $0\ V$). A plot of the contour lines for this setup is shown in **Figure 11**.

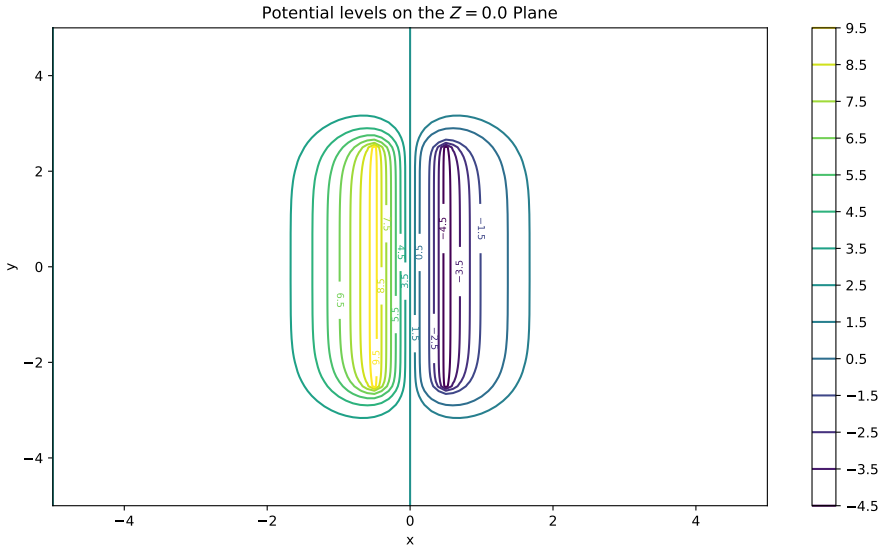


Figure 11: Potential Levels on the $Z = 0$ Plane for a PPC with $V_1 = 10\ V$ and $V_2 = -5\ V$ in a box with potential $2.5\ V$

4. Conclusions

After having done this lab, there are several conclusions that we can extract:

Firstly, it is possible to discretize the problem of Laplace's Equation and solve it for the finite parallel plate capacitor using the method of successive realxation. Furthermore, the error in this method can be measured by the maximum local residual of the mesh. Therefore, we can know the maximum error that we are introducing in our results and keep iterating the method until it falls below some chosen limit tolerance. In our case, this tolerance was chosen to be 0.01 V.

Secondly, we have been able to create several plots comparing the potential generated by a finite ppc and an infinite ppc along different axes. We have found that the outer region of both capacitors has a very different potential, with percent difference rapidly approaching 100%. On the other hand, the inner regions have very similar potential in both kinds of capacitors, especially when considering the central part of the finite ppc. This has served to show the importance of fringing fields in finite ppcs, and how they distort the potential at the edges of the plates.

In this lab, we have computed the maximum distance from the center of the capacitor along the y-axis at which, with a maximum error of 10%, we can consider the finite ppc as an infinite ppc.

We have also been able to plot the contour lines of the electostatic potential generated by the finite ppc in the OXY plane. This has enabled us to see more clearly what the electric field lines look like in that plane.

Finally, this lab has helped to consolidate knowledge and understanding of electromagnetism, particularly involving the workings of parallel plate capacitors, and the different implications there are when considering a finite ppc as being infinite.