HONORS CONTRACT WRITEUP

AIDEN LEWIS

PHY 151 — DR. ZANIEWSKI

Abstract

The project I created to complete this contract was a program, written in the coding language Python, generating a graphical representation of the equipotential surfaces of a user-provided charge distribution — given the magnitudes, positions, and signs of each charge, my code calculates the electric potential throughout the surrounding region and then displays several surfaces containing points with the same potential values in both two and three dimensions. Essentially, it produces an easy-to-understand visualization of one of the core principles underpinning electrostatic physics and electromagnetism as a whole.

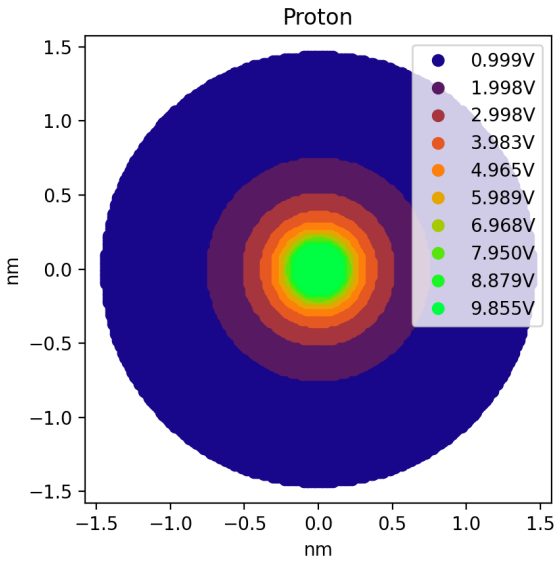
Background

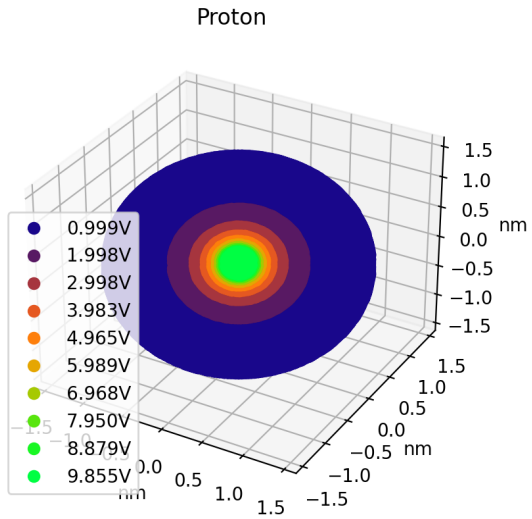
The interactions between charged particles and electric fields form the foundation of electrostatics — objects with some charge inside an electric field experience a force dependent on the charge's value and the strength of that field. Coulomb's Law, or $F_E = \frac{kq_1q_2}{r^2}$, describes this force, and it can be manipulated to find the equation for the electric potential of a particular point in space, $V = \int \frac{k^* dq}{r}$. Electric potential refers to the amount of energy it would take to move a hypothetical positive test charge from a point with zero electric potential energy (i.e. at an infinite distance away) to a different point closer to the charges under consideration. The lines (in two dimensions) and surfaces (in three dimensions) connecting points of equal electric potential are known as equipotentials, and they can provide a visual, more intuitive representation of concepts like the inverse square law and the gradient of the electric field over some distance in space.

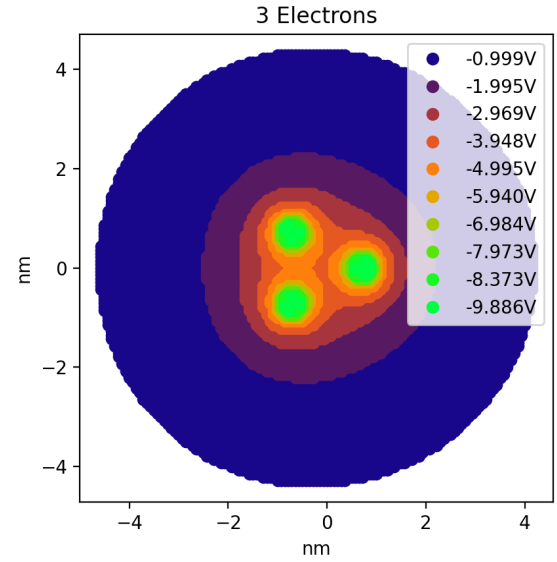
Approach

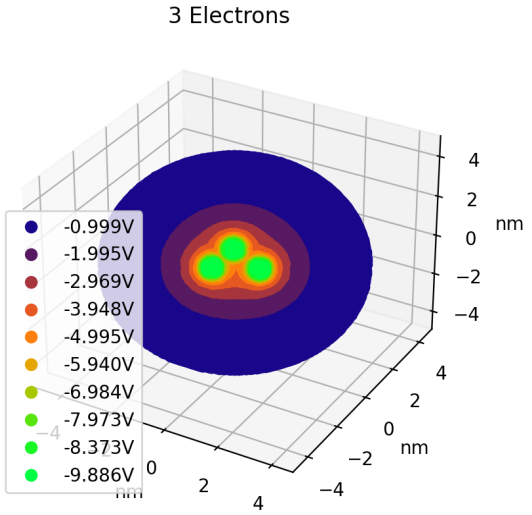
I elected to use Python (specifically, an IPython/Jupyter notebook) as the coding language for this project because it has a plethora of built-in packages (such as NumPy, Matplotlib, and many others) for making calculations, plotting data, and generally performing the tasks inherent to any science-oriented programs; overall, it was the option best suited to my particular purposes. The code itself is quite involved, at over three hundred lines in length, but the majority of it simply takes the information given by the user and processes it into an easily-graphed form. Parsing the user's input and defining parameters composes the bulk of the code. When running, it prompts the user to provide the number of charges to model, their values

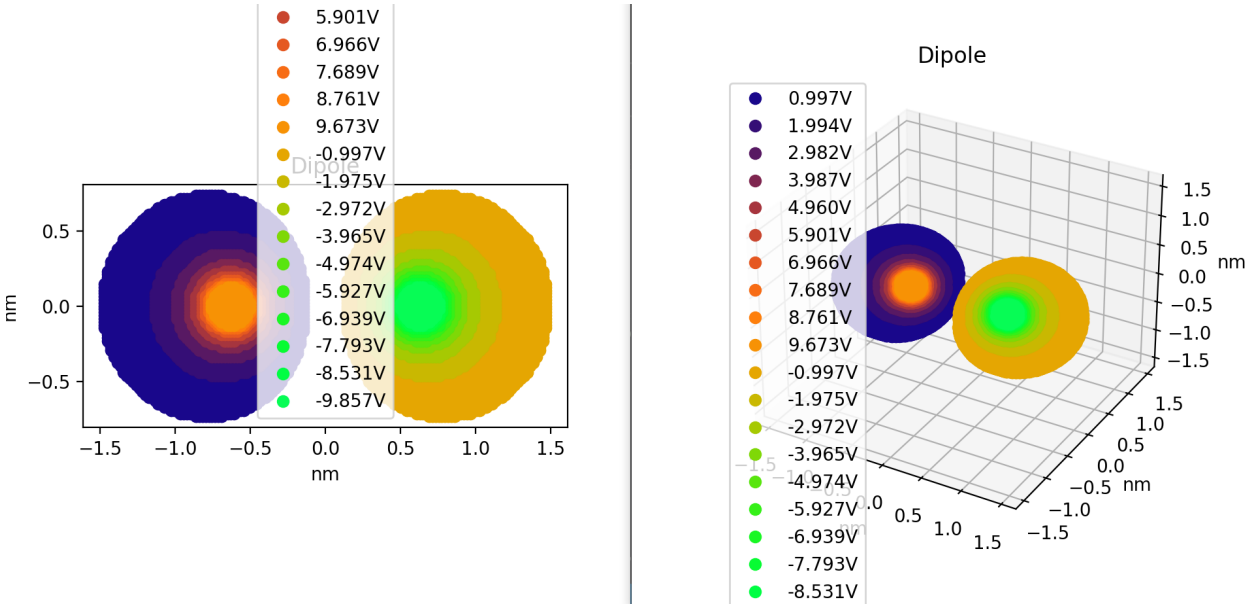
and positions, and information pertaining to saving the generated distribution or loading a previously-saved distribution. Depending on the specifics of the input, it can take around a minute to perform all the necessary calculations, and once finished it displays both an interactive three-dimensional representation of the surfaces and a static two-dimensional cross-section more akin to standard equipotential line diagrams. Pictured below are the outputs of some of the default distributions:











The user can load one of the provided .dat save files to avoid the time taken to complete the initial calculations, which consist largely of a list of x-y-z point coordinates and the corresponding voltage values for each, in addition to the number of surfaces used, binary values representing whether positive and negative charges are present, and the overall number of charges modelled. This allows the user to effectively 'skip' all of the computationally-intensive code, though the 3D plot windows themselves run fairly slowly due to the volume of data being plotted. (For this reason, it is not recommended that either the 'accuracy' nor 'number of surfaces' variables be changed in the code — nonetheless, it can still be done).

What I Learned

Throughout the course of this project, I gained a much deeper and more thorough conceptual understanding of the physical principles behind electric potential and equipotential surfaces/lines; deciding how to approach it, writing the calculation code, and evaluating the results the program produced required me to actively think about what it needed to do and how it needed to do it, particularly with regards to fixing the various problems that appeared throughout the coding process. Furthermore, until this semester, I had done very little with scientific applications of coding, and so the time I spent on this project gave me invaluable experience working with both physics and computer science alongside each other — as a physics and computational mathematics dual major, this will certainly benefit me greatly throughout the courses of my academic and, eventually, professional careers alike.