PHY64 Lab 2

Toki Nishikawa, Hannah Magoon

May 13th, 2023

Contents

1 Introduction

In Millikan's oil drop experiment, the goal was to measure the charge on an oil droplet by balancing the gravitational force with the electrical force acting on the droplet. The electrical force was applied through a capacitor field, and the velocity of the oil droplet was measured as it fell under the influence of gravity and the field.

2 Analysis method 1

2.1 Description of Analysis

First, the video recording taken during the lab was analyzed. During this analysis, the velocities of oil droplets were measured before and after an electric field is turned on. Special care was taken to begin this velocity calculation once the drops reached terminal velocity (and not during periods of acceleration). No data was taken for at least 4 frames immediately before and after the switching of field polarity. Data was only recorded for drops that had positions that were clearly in-focus and trackable across the screen. Generally, this means that we tended to select the slowest-moving drops during each transition. Faster moving drops yielded worse velocity fits for two reasons: first, they spent less time on the screen, meaning fewer datapoints could be collected. Second, faster drops tended to appear blurred on the camera, meaning their position was spread across scales of over 0.1mm. This means that our dataset is skewed heavily towards low values of charge accumulation on the drops.

An image of our acquisition setup is included below. Here, one can clearly see that some drops are large and out-of-focus with the camera. Other drops are in-focus, and it was these smaller and clearer drops that were used in our data acquisitions.

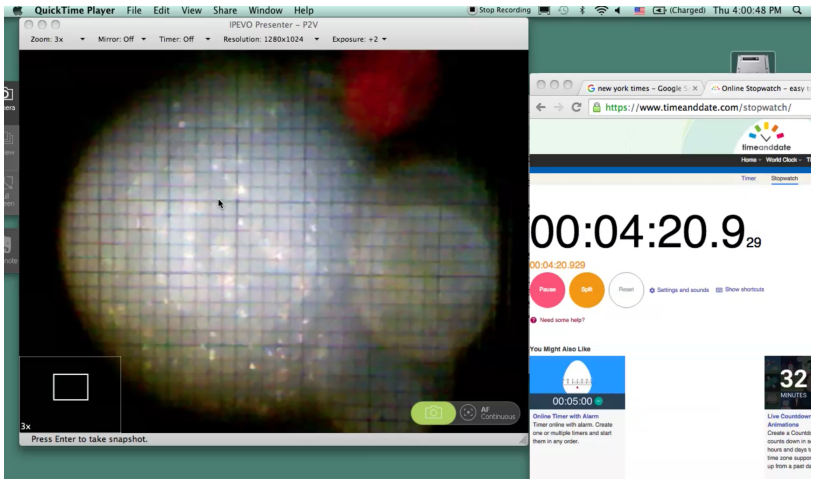


Figure 1: Data Acquisition Setup

For each velocity point, we recorded as many position/time values as possible. We required a minimum of 5 points for each velocity value. Typically we recorded somewhere between 7 and 15 points for each dot. A full list of our data is included in this spreadsheet. A total of 96 datapoints were measured for the purposes of this analysis. Collection of this data was quite difficult. Initially, the video was split in half and the work was split between two lab partners. However, during this

first analysis attempt, we did not collect enough position/time points per oil droplet. This led to excessively high uncertainty on the data, and a messy distribution of charge values on the droplets. To remedy this, the entire first dataset was scrapped. The video was re-analyzed painstakingly over the course of the following month. As the data was collected, preliminary versions of the position/time plots were generated to ensure that no period of acceleration was included, as this seemed to be the source of our issues the first time around. These velocity plots are included in the data spreadsheet.

Next, we imported these datavalues to python. Here, we compiled a list of the least squares fit value for velocity (and associated uncertainty) for each set of points. Note that with this fit method, each datapoint is assigned a unique uncertainty value. A more in-depth discussion of uncertainty is included later in the report. An randomly selected velocity dataset from our spreadsheet is included here for reference:

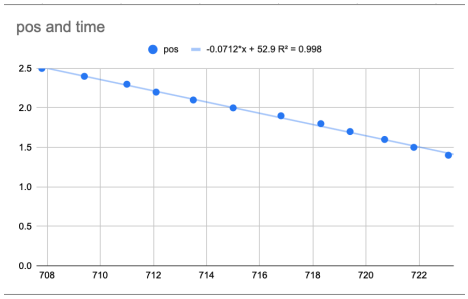


Figure 2: Measuring falling velocity of the drop

Following the equations outlined in the lab handout, we can use this experimental setup to calculate the charge on each oil drop. First, we draw force diagrams for the drops.

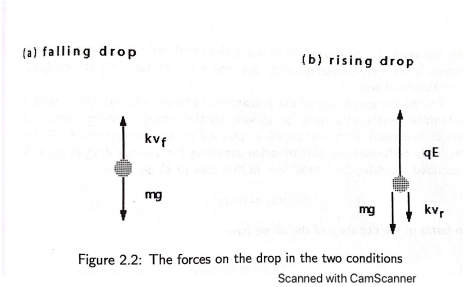


Figure 2.2: The forces on the drop in the two conditions

When terminal velocity is reached, the drops are in equilibrium and the following equations can be found. For the falling drop:

$$F_g = F_d \tag{1}$$

$$mg = kv_f \tag{2}$$

Where here, k is an unknown constant of proportionality for the drag force. The equation for the rising drop is given by:

$$qE = mg + kv_r \tag{3}$$

Next, solve each equation for k and set them equal. The equation for the falling drop yields $k = \frac{mg}{v_f}$ and the equation for the rising drop yields $\frac{qE-mg}{v_r}$. Setting these equal, we find

$$\frac{mg}{v_f} = \frac{qE - mg}{v_r} \tag{4}$$

$$\frac{mgv_r}{v_f} = qE - mg \tag{5}$$

$$1 + \frac{v_r}{v_f} = \frac{qE}{mg} \tag{6}$$

From this equation, we better understand how the ratio of droplet velocities relates to charge. If the value q is quantized, then the left hand side of this equation $1 + \frac{v_r}{v_f}$ will exhibit quantization as well. To get a preliminary sense of whether our data will be usable, we can make a histogram of this dimensionless quantity. As shown below, evidence of quantization may be present, but it is difficult to discern. However, this may be due to the way that the data was binned. During this preliminary phase of analysis, data handling was not optimized. The plot below sorts the data into 100 bins across the entire range of possible x-values. No time was spent optimizing this for improved delineation between clusters.

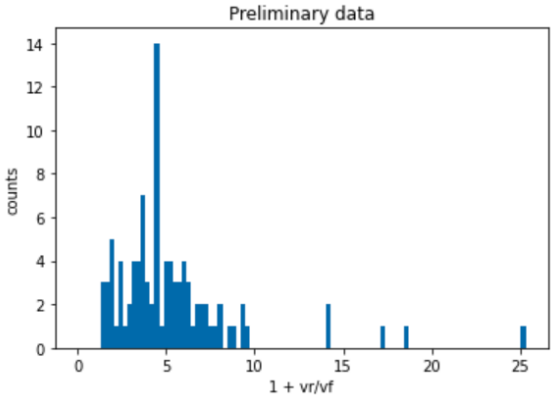


Figure 3: Preliminary dataplot

We can find a second relation by solving the two equations for the expression mg and setting them equal to one another:

$$kv_f = qE - kv_r \tag{7}$$

$$k(v_f + v_r) = qE \tag{8}$$

Next, we need to find a value to substitute for the mass m. To do this, we find their volume, and use the known density of oil to calculate their mass. Stokes' law is used to describe the speed at which a spherical particle passes through a medium as a function of the medium's viscosity and the size of the particle. It gives us a value for the constant k in terms of known system properties. We can use this relation to find drop radius:

$$mg = 6\pi a\eta v_f \tag{9}$$

Substituting mass for its volume times its density, we have:

$$\frac{4}{3}\pi a^3 \rho g = 6\pi a\eta v_f \tag{10}$$

Which yields an equation for radius:

$$a = \sqrt{\frac{9\eta v_f}{2\rho g}} \tag{11}$$

Although Stoke's law provides a clean way to substitute unknown/undefined quantities in our equation for known system paramters, it isn't entirely accurate. Stokes law assumes that the oil drops are

perfectly spherical, and are falling through a homogeneous medium. In reality, things like air pressure, anomalies found in the shape of the drops, and inhomogeneous medium break this assumption. Millikan proposed a corrected version of Stokes's law to account for these discrepancies:

$$q = \frac{6\pi}{E}(v_f + v_r)\sqrt{v_f}\left(\frac{1}{1 + b/pa}\right)^{3/2}\sqrt{\frac{9\eta^3}{2\rho g}} \tag{12}$$

Millikan verified this proposed correction by plotting $q^{2/3}$ vs $1/pa$ (where p is the air pressure, and a is the drop radius) to find confirm the relationship predicted by the equation above.

In this analysis, we will use the equation above to find the charge q of the falling droplets. We implemented this equation in python and evaluated its value for each measured droplet. By binning the data, we were able to make a histogram of all measured q values. This histogram is shown below for various bin sizes:

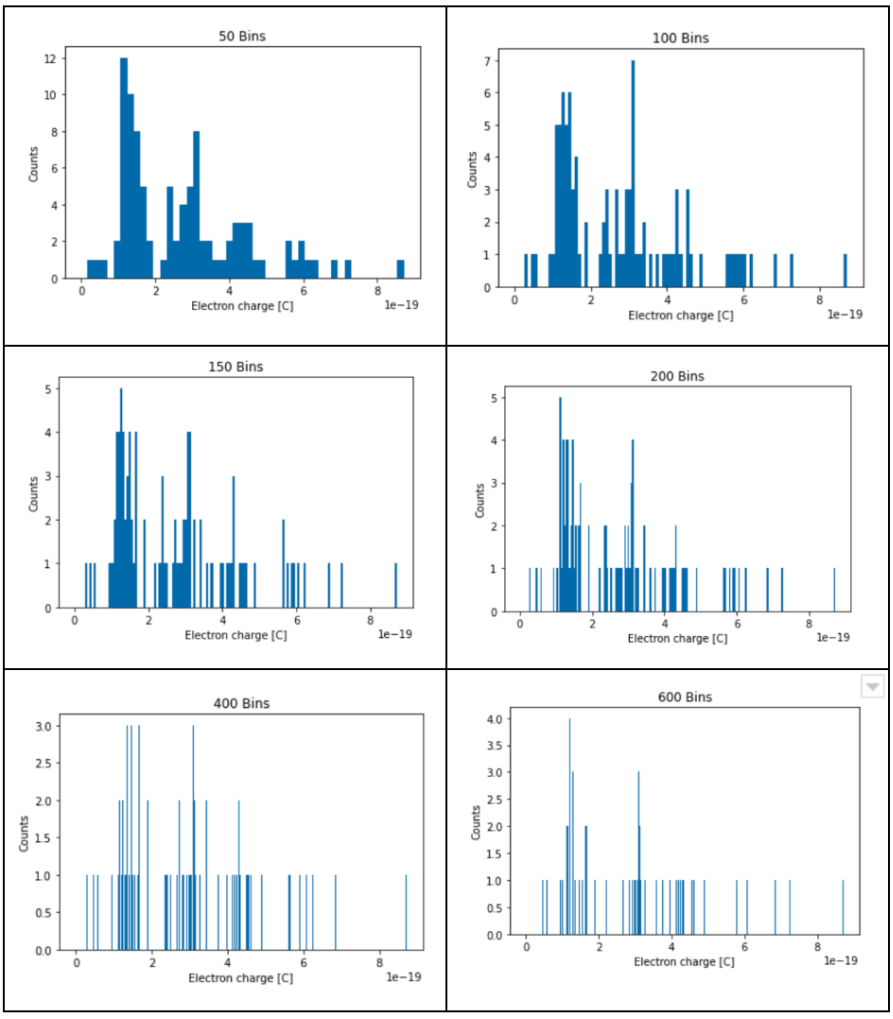


Figure 4: Histogram of calculated droplet charges, plotted with increasing number of histogram bins

We note that each datapoint in the histogram has an individual uncertainty value attached to

it, found from the linear fit to initial and final velocity. This information is lost in the histogram plotting method. Discussion of handing this uncertainty is addressed later in the report.

To fit the data, we constructed a periodic gaussian distribution. We designed the distribution using the following python function:

```
def periodic_gauss_function(x, x0, amp, sigma):
    x1 = (x%x0)
    gauss1 = amp*np.exp(-(x1-x0)**2/(2*sigma**2))
    gauss1 = 0
    x2 = x0-(x%x0)
    gauss2 = amp*np.exp(-(x2-x0)**2/(2*sigma**2))
    return gauss1 + gauss2
```

To understand how this function works, we can plot arbitary values $x0 = 20$, $\sigma = 1.6$, $amp = 10$. For this combination of inputs, the first term in the sum ($gauss1$) looks like:

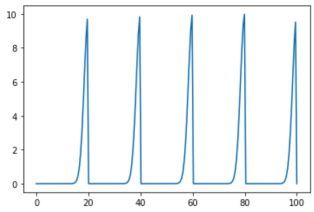


Figure 5: Left side of periodic gaussian distribution

And the second term ($gauss2$) plots as:

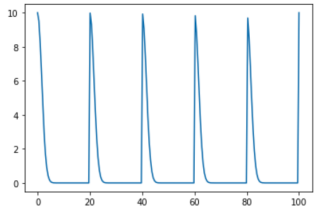


Figure 6: Right side of periodic gaussian distribution

By summing the terms, we can piece together a gaussian potential that repeats periodically around multiples of its central peak value $x0$. Note that we maintain the presence of a left-sided peak centered at $q = 0$.

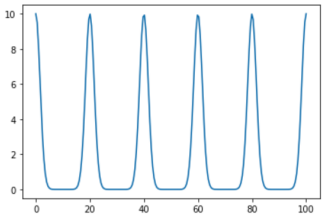


Figure 7: Periodic gaussian distribution

When fitting to this potential, we found it was necessary to scale our data. We multiplied the charge dataset by a factor of 10^{20} so that our datavalues would be larger. Without this scaling, the `scipy.optimize` fit mechanism we used was found to encounter "divide by zero" errors. This is likely due to rounding / int to float conversions in the code: python deufaults to 64-bit floats, and though it's possible to overwrite this, it can sometimes be difficult to find and fix every relevant instance. We found it much more straightforward to artificially scale the charge value as we fit the data. When we fit the data to this potential, we find plots of the following structure:

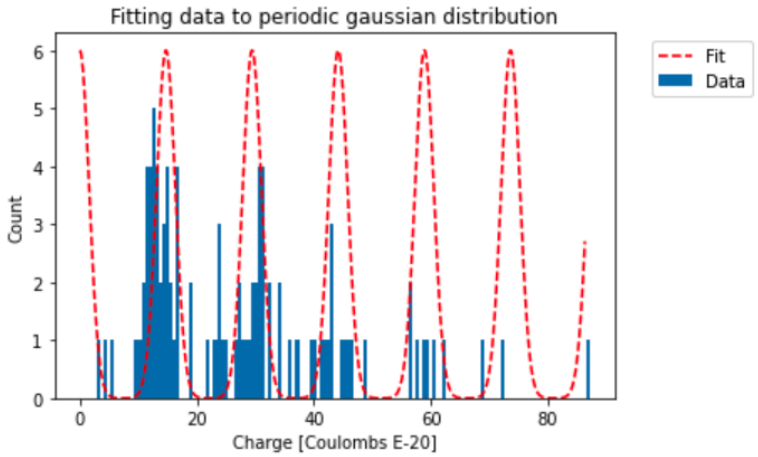


Figure 8: Histogram of data and fit to periodic gaussian, amplitude constrained to ±25% of the maxium occupancy

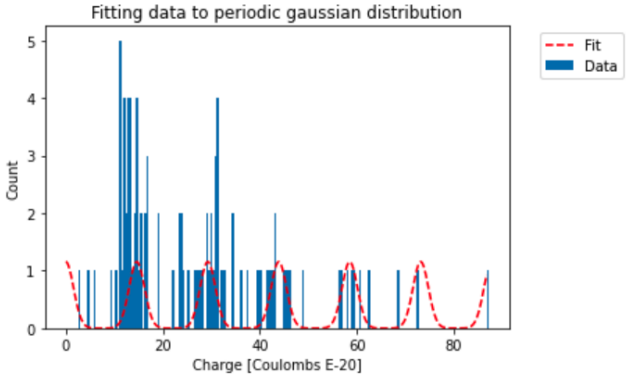


Figure 9: Histogram of calculated droplet charges, plotted with increasing number of histogram bins, unconstrained fit amplitude (not used for analysis). This results from the fact that we are fitting to our entire dataset, but out data predominantly lies in the q=1 to q=2 charge range. Reducing amplitude reduces overall uncertainty in the fit, but at the expense of the uncertainty to central charge values. Since amplitude is irrelevant to our analysis, we bounded its fit so that we could reduce uncertainty in the charge periodicity fits.

In comparison to the fit function, our data is shown to fall off rapidly towards higher charge quantities. This is largely due to our datataking method- we only collected data for dots that we could clearly track across the screen. This drove a selection for drops with the lowest charge values.

It's also due to the fact that it's less likely for the droplets to accumulate larger charge values. Although the amplitude of our data groupings clearly decays with charge, this decay cannot be easily modeled in a fit function. This is due to the fact that the decay is (somewhat) artificially generated by our selection choice. In reality, we expect some arbitrary distribution of charges. For the sake of simplicity, we kept amplitude a constant throughout the periods of the Gaussian function.

The uncertainty in each fit fluctuates quite a bit depending on the number of bins that we sort our data into. To optimize fits to our data-shape, we will first scan across a range of histogram bin-sizes. Second, we constrain plot amplitude to take on a value between $75\% - 125\%$ times the maximum "count" value seen on the histogram. This change was observed to improve fit fidelity unilaterally across all other parameters tested (as measured by covariance parameters). A comparison of the data-fits with and without this constraint is shown above.

The effects of bin-size on histogram shape can be seen clearly. As we plot fit charge and uncertainty in this fit charge value as a function of binning, we find the following trends:

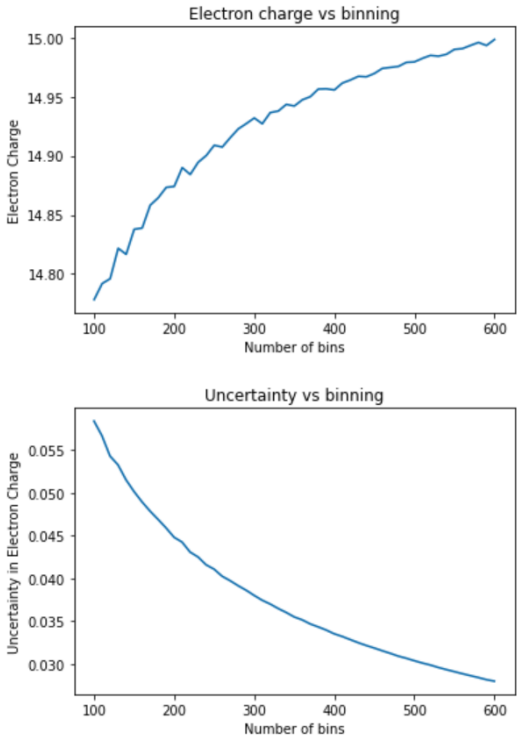


Figure 10: Uncertainty plotted as a function of measured charge

Initially, increasing binning greatly increases our fit quality. However, this increase in fidelity begins to level out as the occupancy of each bin approaches 1. This is due to the fact that smaller bins means higher resolution in data; charge values are plotted in locations closer to their actual value. From this plot, we pull the optimal bin method for our data, as determined by identifying the fit with a minimum uncertainty value. This was found to be with 580 bins. Here, electron charge (or periodicity of the data) was found to be centered around 14.998474428664206 with fit uncertainty of ± 0.027975179865209605

We compare this result to the actual mass of the electron by plotting both values as vertical lines

overlaid across the histogram:

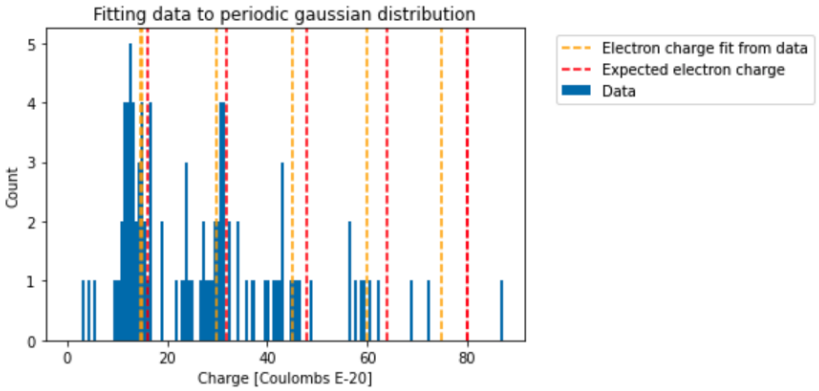


Figure 11: Histogram with fit and expected values overlaid, 150 bins

For the sake of visual clarity, the binning on the dataset displayed above is set to 150. It is important to note that these fit values were found with much more spacing between bin values. The actual binning used during the fit protocol is displayed below:

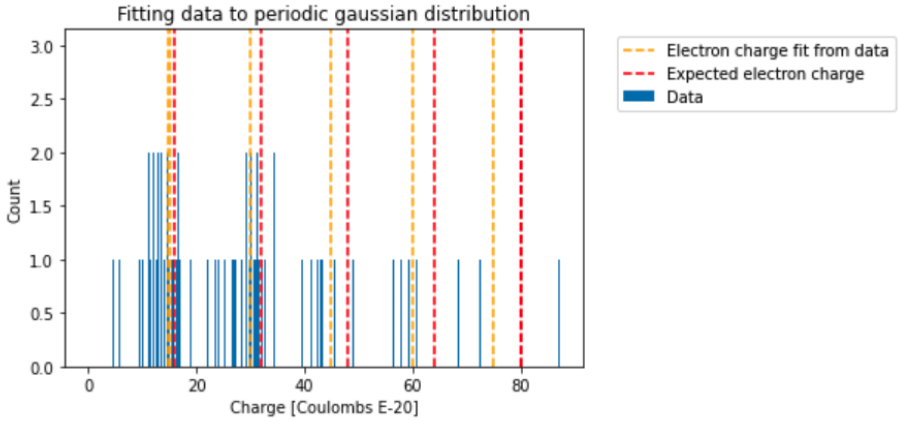


Figure 12: Histogram with fit and expected values overlaid, 580 bins

2.2 Reflection of Analysis method 1

Following a discussion with Professor Sliwa, we realized that although this analysis tactic provided a nice visualization of our data and yielded a fit quite close to the expected value of the electron's charge, it had a major shortcoming. In this model, uncertainty of individual datapoints is discarded. The only method of calculating uncertainty to this fit is averaging the uncertainty values of each datapoint and taking it to represent global error in measurement setup. The following sections describe a calculation of error in individual datapoints, and a python implementation of an alternate histogram binning method that accounts for these individual uncertainties.

3 Uncertainty

There are many sources of uncertainty beyond that of fitting that play a role in our data quality. Using the propagation of error formula, we will systematically find the error contribution from each of our measured values. Then, we will use python to add these contributions in quadrature.

$$\sigma^2(y_i) = \sum_{j=1}^{n} (\frac{\partial y_i}{\partial x_j})^2 \sigma^2(x_j) \tag{13}$$

In this section, we will find individual error contributions to the charge formula 2.10 from the lab handout:

$$q = (\frac{6\pi}{E}) \left(v_f + v_r\right) \sqrt{v_f} \left(\frac{1}{1 + b/pa}\right)^{3/2} \sqrt{\frac{9\eta^3}{2\rho g}} \tag{14}$$

3.1 Uncertainty due to electric field

First, we look at uncertainty due to the electric field. From the experimental setup, we noted that there is approximately 10V uncertainty in the applied capacitor voltage. This is due to the fact that the electronics used had analog control, and could potentially fluctuate over time. Capacitor voltage is converted to electric field using the following relation:

$$E = V/d \tag{15}$$

This substitutes into the equation

$$q = (\frac{6\pi d}{V}) \left(v_f + v_r\right) \sqrt{v_f} \left(\frac{1}{1 + b/pa}\right)^{3/2} \sqrt{\frac{9\eta^3}{2\rho g}} \tag{16}$$

The derivative of this equation with respect to V is given by:

$$q = (\frac{-6\pi d}{V^2}) \left(v_f + v_r\right) \sqrt{v_f} \left(\frac{1}{1 + b/pa}\right)^{3/2} \sqrt{\frac{9\eta^3}{2\rho g}} \tag{17}$$

We assign an uncertainty of 2% to this voltage value, meaning $\sigma(V) = 10V$. This means that The contribution to total error is therefore given by:

$$(\frac{\partial q}{\partial V})^2 \sigma^2(V) \tag{18}$$

$$\left[(\frac{-6\pi d}{V^2}) \left(v_f + v_r\right) \sqrt{v_f} \left(\frac{1}{1 + b/pa}\right)^{3/2} \sqrt{\frac{9\eta^3}{2\rho g}}\right] \left[10V\right] \tag{19}$$

3.2 Uncertainty due to velocities

Next, we note that there is an uncertainty term introduced by the velocity measurements. This uncertainty is easily found from the linear fits to velocity values. To see how these uncertainty values propagate to the equation for q, we follow the same procedure.

3.2.1 Uncertainty due to rising velocity

Differentiating q with respect to the v_r, we find:

$$\frac{dq}{dv_r} = \left(\frac{6\pi}{E}\right)\left(v_f\right)\sqrt{v_f}\left(\frac{1}{1+b/pa}\right)^{3/2}\sqrt{\frac{9\eta^3}{2\rho g}} \tag{20}$$

This means that the contribution to total error is given by:

$$\left(\frac{6\pi}{E}\right)\left(v_f\right)\sqrt{v_f}\left(\frac{1}{1+b/pa}\right)^{3/2}\sqrt{\frac{9\eta^3}{2\rho g}}\cdot\sigma(v_f) \tag{21}$$

3.2.2 Uncertainty due to falling velocity

Differentiating q with respect to the v_f, we find:

$$\frac{dq}{dv_f} = \left(\frac{6\pi}{E}\right)\left(v_r\right)\sqrt{v_f}\left(\frac{1}{1+b/pa}\right)^{3/2}\sqrt{\frac{9\eta^3}{2\rho g}} + \left(\frac{6\pi}{E}\right)\left(v_f+v_r\right)\frac{1}{2\sqrt{v_f}}\left(\frac{1}{1+b/pa}\right)^{3/2}\sqrt{\frac{9\eta^3}{2\rho g}} \tag{22}$$

And the contribution to total error is given by $\frac{dq}{dv_f}\sigma(v_f)$ We note that a has a dependency on v_f. To find the contribution of the uncertainty in a, we first propagate errors in v_f through to a, then propagate them to q. We note that a is given by:

$$a = \sqrt{\frac{9\eta v_f}{2\rho g}} \tag{23}$$

$$\frac{da}{dv_f} = \sqrt{\frac{9\eta}{4\rho g v_f}} \tag{24}$$

Assuming the uncertainty in a comes only from v_f, we find $\sigma(a)$:

$$\sigma(a) = \sqrt{\frac{9\eta}{4\rho g v_f}}\cdot\sigma(v_f) \tag{25}$$

Next, we find the dependence of q on a.

$$\frac{dq}{da} = \left(\frac{6\pi}{E}\right)\left(v_f+v_r\right)\sqrt{v_f}\left(\frac{3bp^{3/2}a^{1/2}}{2(1+b/pa)^{5/2}}\right)\sqrt{\frac{9\eta^3}{2\rho g}} \tag{26}$$

Combining the previous two equation, we find the error contribution of a:

$$\frac{dq}{da} = \left(\frac{6\pi}{E}\right)\left(v_f+v_r\right)\sqrt{v_f}\left(\frac{3bp^{3/2}a^{1/2}}{2(1+b/pa)^{5/2}}\right)\sqrt{\frac{9\eta^3}{2\rho g}}\cdot\sqrt{\frac{9\eta}{4\rho g v_f}}\cdot\sigma(v_f) \tag{27}$$

3.3 Total uncertainty

To find the total uncertainty, we combine these terms in quadrature, following the equation:

$$\sigma^2(y_i) = \sum_{j=1}^{n}\left(\frac{\partial y_i}{\partial x_j}\right)^2\sigma^2(x_j) \tag{28}$$

Because the equations are quite tedious and I am prone to calculation error, we did this entirely in python. The script for calculating these errors is shown below.

```
V = 500
sig_q = np.zeros(len(vi))

for i in range(0,len(vf)):
    v_i = np.abs(vi[i]) * 0.0001
    v_f = np.abs(vf[i]) * 0.0001

    sig_vi = calc_uncert(vi[i], ri[i], ni[i])
    sig_vf = calc_uncert(vf[i], rf[i], nf[i])
    sig__V = 10
    sig__a = np.sqrt((9*eta)/(4*rho*g*v_f)) * sig_vf

    dq_dvi = ((6*np.pi)/E)
    dq_dvi = dq_dvi * v_f * np.sqrt(v_i)
    dq_dvi = dq_dvi * ((1/1+b/(p*a[i]))**(2/3))
    dq_dvi = dq_dvi * np.sqrt((9*eta**3)/(2*rho*g))
    dq_dvi = dq_dvi + (  ((6*np.pi)/E)*(v_f + v_i)*(1/(2*np.sqrt(v_i)))*((1/1+b/(p*

    dq_dvf = ((6*np.pi)/E)
    dq_dvf = dq_dvf * v_i * np.sqrt(v_i)
    dq_dvf = dq_dvf * ((1/1+b/(p*a[i]))**(2/3))
    dq_dvf = dq_dvf * np.sqrt((9*eta**3)/(2*rho*g))

    dq__da = ((6*np.pi)/E)
    dq__da = dq__da * (v_f + v_i)
    dq__da = dq__da * np.sqrt(v_i)
    dq__da = dq__da * 3*b*(p**(3/2))*(a[i]**(1/2))
    dq__da = dq__da / (2)
    dq__da = dq__da / (1+(b/(p*a[i])))**(5/2)
    dq__da = dq__da * np.sqrt((9*eta**3)/(2*rho*g))

    dq__dV = ((-6*np.pi*d)/(V**2))

    dq__dV = dq__dV * (v_f + v_i)
    dq__dV = dq__dV * np.sqrt(v_i)

    dq__dV = dq__dV * ((1/1+b/(p*a[i]))**(2/3))
    dq__dV = dq__dV * np.sqrt((9*eta**3)/(2*rho*g))


    vi_term = (dq_dvi*sig_vi)**2
    vf_term = (dq_dvf*sig_vf)**2
    a__term = (dq__da*sig__a)**2
    V__term = (dq__dV*sig__V)**2

    sig_q[i] = np.sqrt(vi_term + vf_term + a__term + V__term)
```

Figure 13: Python code to calculate uncertainty in datapoints

We stored these values in an array and made use of them in the following section.

4 Analysis method 2

4.1 Analysis

We used a python script to distribute the histogram bin values across the range of their uncertainty. Included below is our source code:

With this setup, we make a histogram with 2000 bins within the range of q values calculated. Next, we consider the uncertainty range of each q value. Each q value should be displayed with area = 1 on the histogram. We divide this area by the number of histogram bins covered, and add the value to every bin within the range.

```python
q_scaled = q*1.0E20
sigma_scaled = sig_q*1.0E20
new_bins = np.linspace(min(q_scaled), max(q_scaled), num=2000)
new_binned_data = np.zeros(len(new_bins))

for i in range(0, len(q_scaled)):
    min_q = q_scaled[i] - sigma_scaled[i]
    max_q = q_scaled[i] + sigma_scaled[i]

    ## find total bins covered by this x-range,
    ## populate each with some fraction of the datapoint
    bins_covered = 0
    for j in range(0, len(new_bins)):
        if ((new_bins[j]>= min_q) and (new_bins[j]<= max_q)):
            bins_covered += 1

    ## For tiny uncertainty values, just put it into one bin
    if (bins_covered==0):
        min_dist = 10000
        min_bin_num = 0
        for j in range(0, len(new_bins)):
            dist = np.abs(q_scaled[i]-new_bins[j])
            if dist < min_dist:
                min_dist = dist
                min_bin_num = j
        new_binned_data[min_bin_num] = 1

    ## populate histogram array
    else:
        q_density = 1/bins_covered
        for j in range(0, len(new_bins)-1):
            if (new_bins[j]>=min_q) and (new_bins[j+1]<=max_q):
                new_binned_data[j] += q_density
```

Figure 14: Python code to make histogram

This yields the following plot:

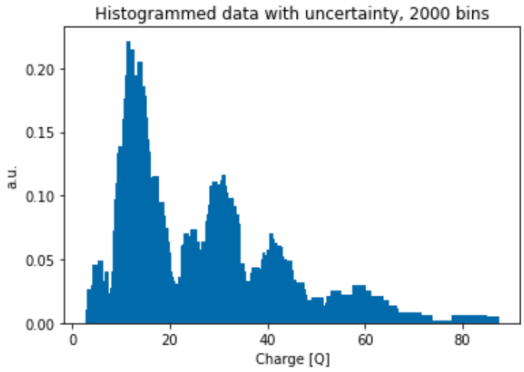
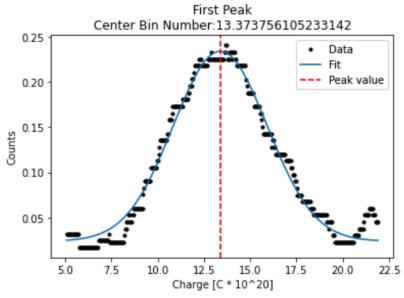
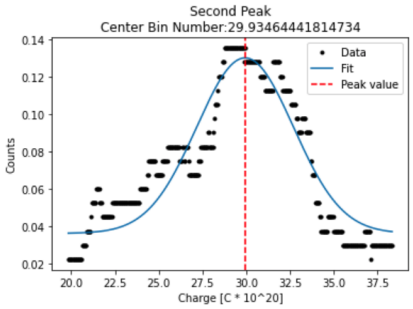
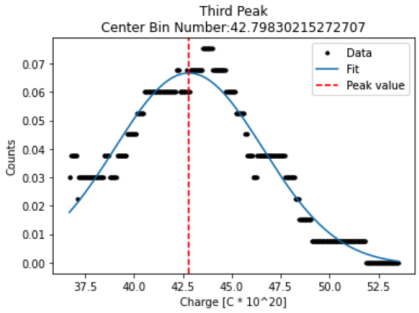


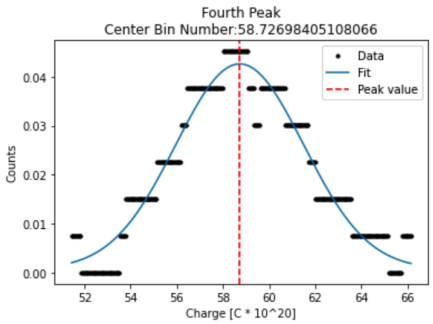
Figure 15: Python code to make histogram

To fit this data, we found it easiest to fit each peak to a Gaussian. It should be noted that this step is unnecessary, as all the individual charge values and their uncertainties can equivalently be fed into a linear fit function. We didn't realize this until after running the following analysis, however, and due to time restrictions we cannot compare the two procedures. Regardless, we believe that this is a valid and useful way to visualize the data. The peak fits are displayed below:









We compiled the fit results and the corresponding uncertainties in central charge value, and plotted them. We fit the results using a linear regression to find the quantization of the charge values:

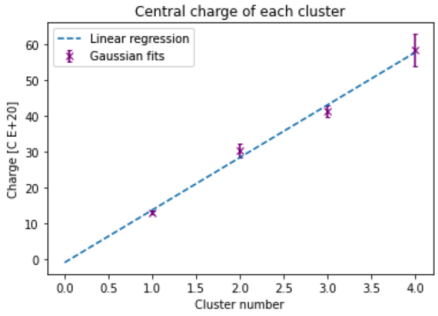


Figure 16: Linear fit of central charge value and index of the charge cluster

4.2 Discussion

From this fit, we find an electron charge of:

$$(14.6564 \pm 0.85795) \times 10^{-20} \tag{29}$$

We note that this is less than the expected charge of $1.6 \times 10^{-19}C$. There are many possible explanations for this discrepancy. Since our data is fairly nicely distributed, we infer that this error is likely due to a systematic error in our setup. It's possible that we mis-recorded the voltage on the power supply. It's also possible that we mis-measured the distance between capacitor plates. Human error on these measurements could propagate with this sort of effect. Alternately, we took our data late in the day. Thermal fluctuations could play a role, causing a systemic measurement of the voltage. In fitting data, we made a conscious effort to identify and measure slowly-moving drops. This was to reduce the uncertainty in our velocity measurements, since it allowed us to collect more datapoints per drop. However, the drops were likely evaporating over the course of our acquisition. Mass is directly proportional to the quantity a in the equations above. Reducing this quantity reduces the value of q, which could account for some of the discrepancy measured. Ahere is additional uncertainty introduced by the shutter speed of the screen-recording video we used to generate our data. We found that while parsing through, there was an electronics lag between the on-screen timer and frame rate of the video. This misalignment could have caused mis-measurements in velocity: if the videofeed of the experiment is updating at a different rate than the timer we were using, we introduce an additional level of measurement error. A final potential source of error is in the capacitor field. If some of the drops we picked were towards the edge of the capacitor, fringe effects could cause non-homogeneous field through the duration of the sample. Since we wanted to collect as many points as possible from our acquisition video, we measured every clearly visible droplet, regardless of its position in the frame. Avoiding drops on the outskirt of our viewing window would likely have improved data quality.

5 Alternate fitting attempts (unsuccessful)

While fitting data, we attempted to account for individual uncertainties in velocity points. In doing so, we plotted charge of the droplets as a function of uncertainty in their measurement. Our goal was to identify areas of correlated low-uncertainty charge values. A scatter plot with our data is included below. Unfortunately, this visualization method was not particularly illuminating.

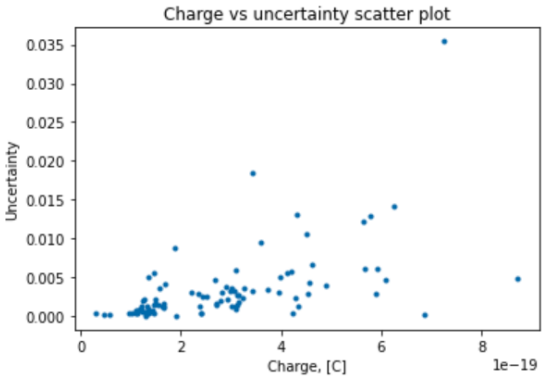


Figure 17: Velocity fit error plotted as a function of measured charge

Next, we attempted to make use of uncertainty values assigned to individual datapoints. We wrote a function to parse through the data and discarded the "worst" 1/3 of it, as measured by

the uncertainty in the velocity fits. The goal was to eliminate human error that may result from mis-identifying the period of time in which the droplets move at constant velocity. Below, the datapoints that survived this cut procedure are displayed. Vertical lines placed at the location of expected charge of the electron to serve as a visual aide.

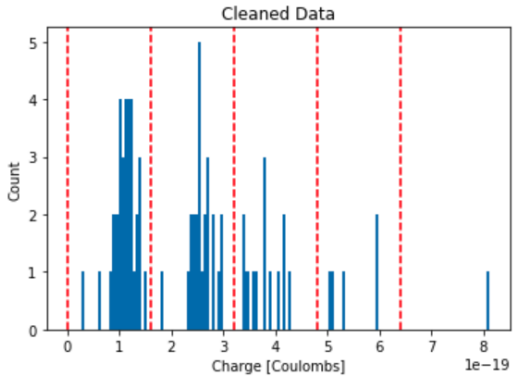


Figure 18: "Cleaned" histogram with expected values overlaid

The data that survived this cleaning process did not appear to have a significantly cleaner distribution than the entirety of the dataset. As a result, we abandoned this fit method. We decided that ultimately, the advantages of discarding potentially faulty data were outweighed by the advantages of an increased number of histogram points. We also began to grow worried that this form of data-cutting may introduce bias. Although we spent a great deal of time considering potential data cuts, we ultimately decided that there was no unbiased method of applying such eliminations; we did not have enough of our data (or any of the calibration data required) to perform a blinded cut procedure. As a result, we reverted to using the entire dataset for the purposes of our analysis.

In all, we think that the analysis method provided in Section 4 is the most accurate and logical way to fit the dataset.