Measuring the Half-Lives of ^{108}Ag and ^{110}Ag by Observing Decay

Andrew Parmet

May 22, 2016

Abstract

The half-lives of ^{108}Ag and ^{110}Ag are measured by video taping their simultaneous decay on an analog counter with a geiger counter. The tape is then parsed with Matlab to generate data which is fitted using two methods: fitting a line to the logarithm of decay rate, and a direct paramaterized fit. After fitting the data we calculate the ratio of the production cross sections of the silver isotopes.

1 Background and Theory

1.1 Production and Decay

Silver occurs naturally in two stable isotopes: ^{107}Ag and ^{109}Ag. These isotopes can be transformed into the radioactive isotopes ^{108}Ag and ^{110}Ag by bombarding them with low-energy neutrons [1]:

$$^{1}_{0}\mathrm{n} + {}^{107}\mathrm{Ag} \rightarrow {}^{108}\mathrm{Ag} \tag{1}$$

$$^{1}_{0}\mathrm{n} + {}^{109}\mathrm{Ag} \rightarrow {}^{110}\mathrm{Ag} \tag{2}$$

The radioisotopes then decay with the reactions:

$$^{108}\mathrm{Ag} \rightarrow {}^{108}\mathrm{Cd} + \mathrm{e}^{-} + \bar{\nu} \tag{3}$$

$$^{110}\mathrm{Ag} \rightarrow {}^{110}\mathrm{Cd} + \mathrm{e}^{-} + \bar{\nu} \tag{4}$$

The goal of this lab is to isolate the half-lives of each isotope. To produce the radioisotopes we place a cylinder of silver foil into a chamber containing radioactive americium, beryllium, and paraffin. The americium emits high-energy helium nuclei which then fuse with beryllium atoms, producing carbon and a high-energy neutron. The neutron then undergoes inelastic collisions with the hydrogen atoms in the paraffin to slow down to velocities at which it can react with the stable silver to produce the radioisotopes in reactions (1) and (2) above.

The sample is irradiated until the radioisotopes are *saturated* - when the rate of production of new radioactive atoms equals the rate of decay of radioactive atoms already produced. The sample is then taken from the americium-beryllium chamber and placed around a cylindrical geiger counter to detect entering elections produced in reactions (3) and (4). Assuming that electrons emerge isotropically from the silver foil, the number of electrons that will travel inwards towards the geiger counter will be proportional to the total number produced.

Using the geiger counter to record to number of events and their time signatures will give us count data of the form:

$$C(t) = C_1(t) + C_2(t), \tag{5}$$

where $C_1(t)$ represents the number of electrons detected from decay of ^{108}Ag and $C_2(t)$ represents the number of electrons detected from decay of ^{110}Ag.

The theory of radioactive decay predicts that the rate of decay of a radioisotope is proportional to the number of atoms of that isotope present in a sample [1]:

$$\frac{\mathrm{d}N_i(t)}{\mathrm{d}t} = -\lambda_i \cdot N_i(t). \tag{6}$$

Solving this differential equation yields:

$$N_i(t) = N_i(0)\ e^{-\lambda_i t} \tag{7}$$

which also reflects the initial condition that the number of radioactive atoms at time $t = 0$ is $N_i(0)$. We now have an equation for the number of atoms left after some time t:

$$N(t) = N_1(0)\ e^{-\lambda_1 t} + N_2(0)\ e^{-\lambda_2 t}. \tag{8}$$

We subtract this quantity from the number of initial atoms $N_1(0) + N_2(0)$ to get:

$$C(t) = N_1(0)\ (1 - e^{-\lambda_1 t}) + N_2(0)\ (1 - e^{-\lambda_2 t}). \tag{9}$$

Given a high enough time resolution in our data we can attempt to fit this equation directly to our data. We can also linearize the data and fit it piecewise with two lines to find the values of λ_1 and λ_2. We take the derivative of Equation (9):

$$\frac{\mathrm{d}C(t)}{\mathrm{d}t} = -\lambda_1 N_1(0)\ e^{-\lambda_1 t} - \lambda_2 N_2(0)\ e^{-\lambda_2 t}. \tag{10}$$

If we assert that the half-life of one isotope will be very large compared to the other, we can take the logarithm of Equation (10) at a large time $t_1 \gg 1/\lambda_1$:

$$\log\left[\frac{\mathrm{d}C(t > t_1)}{\mathrm{d}t}\right] \approx \log\big[-\lambda_2 N_2(0)\big] - \lambda_2 t \tag{11}$$

We then inspect logarithm of the derivative of the data and fit a line to the data when it appears to be linear at a late time. Once we have values for λ_2 and $N_2(0)$ we then subtract the $N_2(t)$ term from equation (9) and repeat the above process to solve for λ_1 and $N_1(0)$.

This process does not take into account the dead time of the geiger counter or the data counter, or the contribution due to room background. The dead time of the geiger counter is on the order of microseconds, and a simple run of the radioactivity yields an event rate no larger than 100Hz, so this effect likely does not contribute to any error. Room background was calculated by letting the counter run without the sample, and the resulting contribution was very small relative to the count registered due to the sample's radioactivity. The data counter is simply assumed to have negligible dead time.

1.2 Parsing the Video: The k-NN Method

This lab requires an accurate measurement of counts over short time intervals, which is difficult to do manually as the counts are registered since everything happens so quickly. At the suggestion of a previous student I took a video of the counter to analyze later. Instead of laboriously playing and pausing the video to discern times and counts I implemented a simplified version of the k-nearest neighbors (k-NN) machine learning algorithm to parse the video and obtain data at a very high time resolution.

k-NN is a simple machine learning algorithm that attempts to classify data points with a set of predefined labels. In this case the goal is to label images of digits with their values. All machine learning algorithms must make assumptions about their data; k-NN works with one basic assumption: Given data in a numerical vector representation, data points that are close together under a distance metric are closely related [3][1]. Images are easy to transform into suitable vectors: render the image in black and white. The result is a two dimensional matrix of values. Reshape this matrix by lining up every row sequentially. If each image has the same rectangular dimensions then this method produces vectors that are closely related under the l^2 norm when the images are of the same digit.

[1] See [2] for another excellent overview of nearest-neighbor classification methods.

Figure 1: A typical parsable digit.

The k-NN classifier takes a large set of 'training' points with associated labels at the start of the algorithm. For any input 'test' vector, the algorithm computes the distance from the test vector to every training vector. The algorithm then outputs the majority vote among the k nearest neighbors to the test point.

In general machine learning algorithms are sensitive to noise in their inputs. The k-NN classification algorithm accounts for noise by taking a vote among the k nearest neighbors. The application in this lab is highly specific - it is possible to set up the camera and the counter in very stable positions so that each image of a number is nearly identical to any other image of the same number. This simplification allows a dramatic reduction in parsing time and complexity by simply taking one image as a reference for each digit (a 1-nn algorithm). Due to the different angles between the camera lens and each place value on the counter, forty images were taken as references - one for each digit for each place value.

Matlab's image processing toolbox was used to analyze each frame of the video. For each frame the algorithm attempted to parse each digit on the counter and assemble their values into a count number. If the counter was in the process of changing its value during the frame capture, the algorithm saw an image blurred between two numbers and did not see one distance substantially smaller than the others. In these cases, the algorithm created a time 'bin' and simply expanded the bin until it found a parsable frame, at which point it recorded the moment in time and the count parsed from the image as a data point. Figure 1 shows a parsable digit in the one's place, Figure 2 shows a non-parsable frame, and Figure 3 shows a parsable frame.

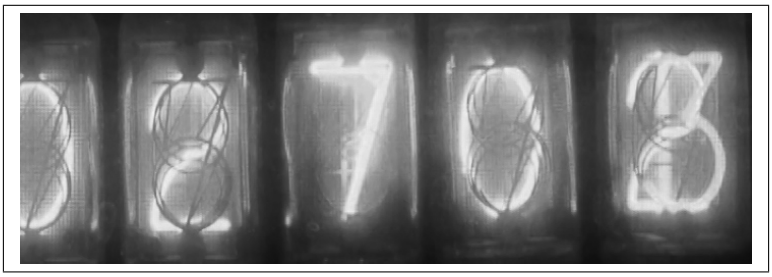

Figure 2: A typical non-parsable frame caught between transitions from 2701 to 2702 and 2702 to 2703.

2 Experimental Details

2.1 Experimenter vs. Decay Time: the Race

As one of the silver decay times is very short, the process of removing the sample from the radioactive chamber and moving it to the geiger counter to start counting is very important. I practiced moving the sample to the geiger counter very quickly a few times, then let the sample bake for about 20 minutes. Then I started the camera, moved the sample to the geiger counter, started the data counter, and sat back to watch the silver decay.

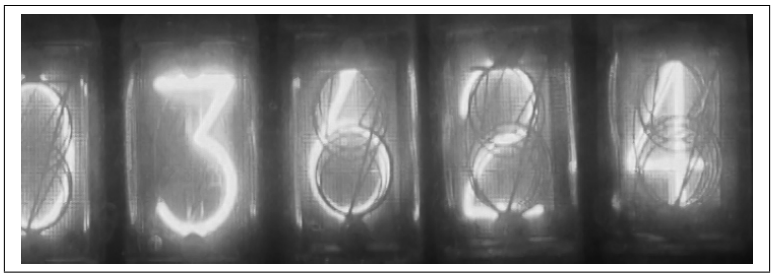

Figure 3: A typical parsable frame. Compare the appearance of the upper-left tail of the '2' in the thousands place of Figure 2 and the tens place here; this comparison is the reason why a different set of reference digits was used for each place value. It is possible (actually, probably quite likely) that the parsing algorithm would have worked without these separate sets, but it was not hard to simply repeat the reference extraction.

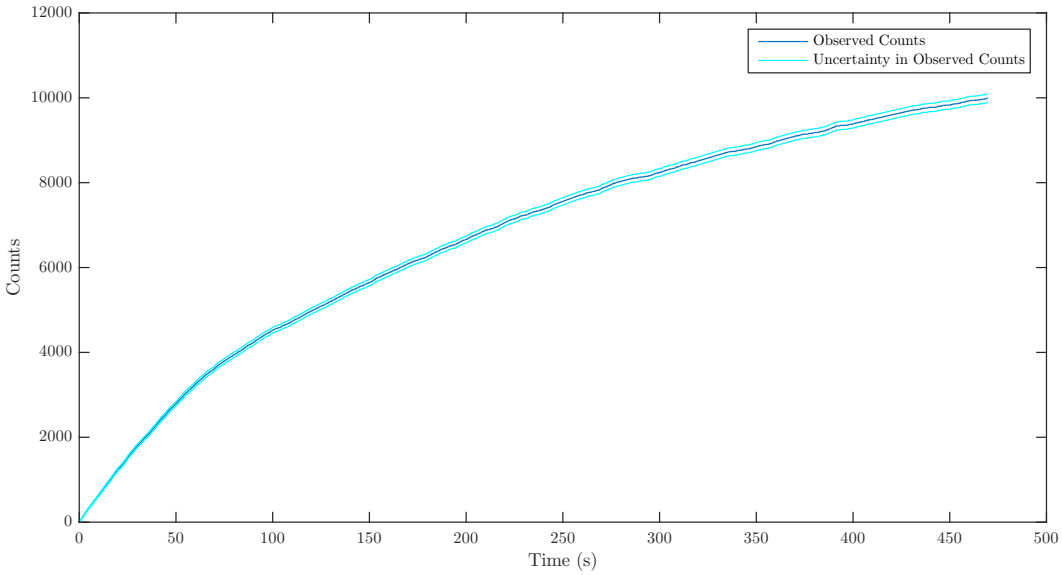

Figure 4: The data parsed by Matlab.

2.2 The Parsed Data

Running the Matlab script yields 7917 parsable frames plotted in Figure 4.

3 Results

3.1 Numerical Stability and the Linear Fit to the Log

Analyzing the raw numerical data using derivatives proved tricky - the data was so fine-grained that numerical derivatives often diverged. I developed two methods to achieve better statistics. First I tried grouping the data into larger bins by only considering counts at constantly separated times - effectively throwing away the majority of data in favor of large bins that may show a cleaner derivative. This method nearly worked, but it was highly unstable. Late derivatives tended to vary greatly because many bins had no change from their neighbors. Increasing the bin size further reduced the number of early bins to be too few for good statistics.

To fix this problem I tried grouping the data into bins of exponentially increasing size. Since radioactive decay is an exponential process, these differentially sized bins were well-suited to the data and yielded a much more stable numerical derivative. The bin locations are plotted in Figure 5, and the numerical derivatives

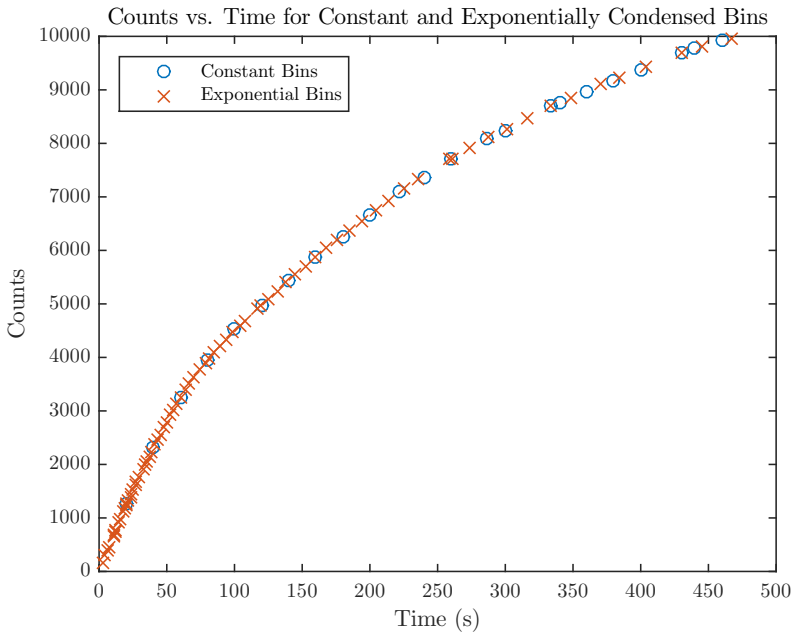

Figure 5: The data left after condensing to bins of constant 20 second size and exponentially increasing size. Note the low frequency of bin for small times when constant sized bins are used as opposed to exponential bins.

are plotted in Figure 6.

Ultimately two exponential bin sizes produced interesting results: bins that increased in size by a factor of 1.05 and by a factor of 1.03. I then attempted to reconstruct my original data using the fits. Results for bins of size 1.05 are shown in Figures 7 and 8. The reconstruction for bin size 1.05 is shown in Figure 9. Only the final reconstruction with its equation is shown for bins of size 1.03 in Figure 10.

3.2 A Direct Fit

I also fit the raw counting data using Matlab's curve fitting toolbox.[2] I parameterized the fit as $F(t) = N_1(1 - e^{\lambda_1 t}) + N_2(1 - e^{\lambda_2 t})$. This parametrization enforces that both decay counts were zero at $t = 0$. Since fitting curves with the sum of two expontentials is in general an ill-posed numerical computation problem, I gave Matlab modest restrictions on fitting parameters that converged well. I constrained λ_1 and λ_2 to be between -1 and 0, and N_1 and N_2 to be at least 0. These constraints seem very reasonable and unlikely to bias the fit in favor of *a priori* known results. The result of this fit is plotted in Figure 11. To make the quality of the fit more apparent, expanded subplots are shown in Figures 12 and 13.

3.3 Calculation of the Product Cross Sections

The production cross sections can be calculated by dividing the initial values of the exponential terms after scaling them to reflect the proportions of silver isotopes found naturally (51.839% ^{107}Ag to 48.161% ^{109}Ag). This calculation assumes that the electrons emitted by each isotope are detected with equal likelihood; this is in fact not true, but it's close enough to hopefully yield a reasonable result.

[2]We discussed in class how this fit is flawed because the error between data points is correlated. What I did not realize is that this should not hurt the fit; while neighboring points in a region of high activity will all sit above the fitted curve, all points that follow will regress to the fit because there are now fewer atoms to decay overall. Errors between points on the log-counting-rate curve are actually similarly correlated.

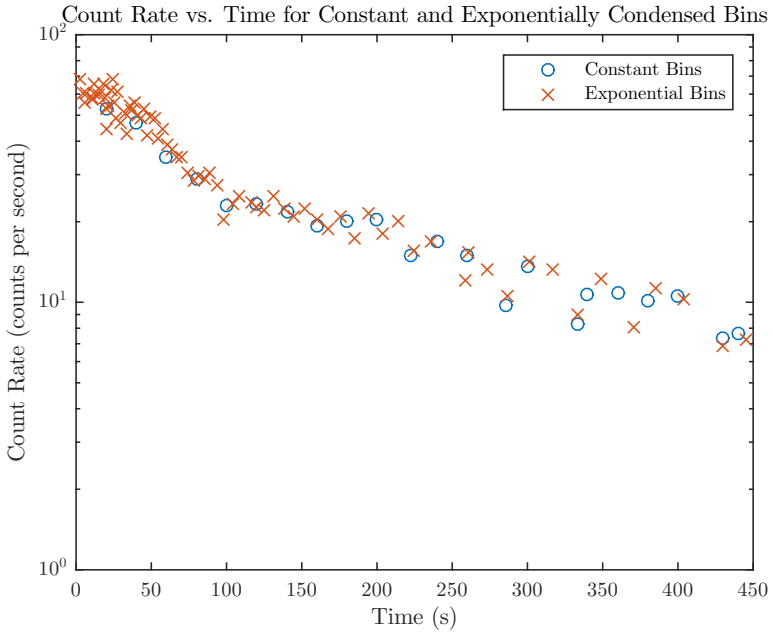

Figure 6: Count rates for constant sized bins and exponential bins. Note the low frequency of available derivatives at early times for constant sized bins paired with the instability of late-time derivatives. In contrast the exponential bins show good statistics throughout the time range.

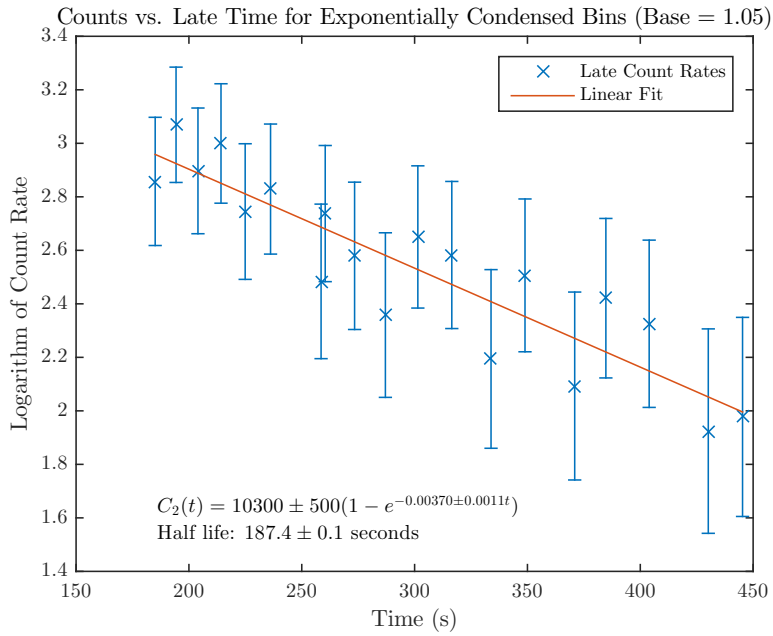

Figure 7: Linear fit to the logarithm of the count rate for late times for exponentially size bins with base 1.05. I chose late time to start at $t = 180$ by trying several values and seeing a clear linear pattern emerge.

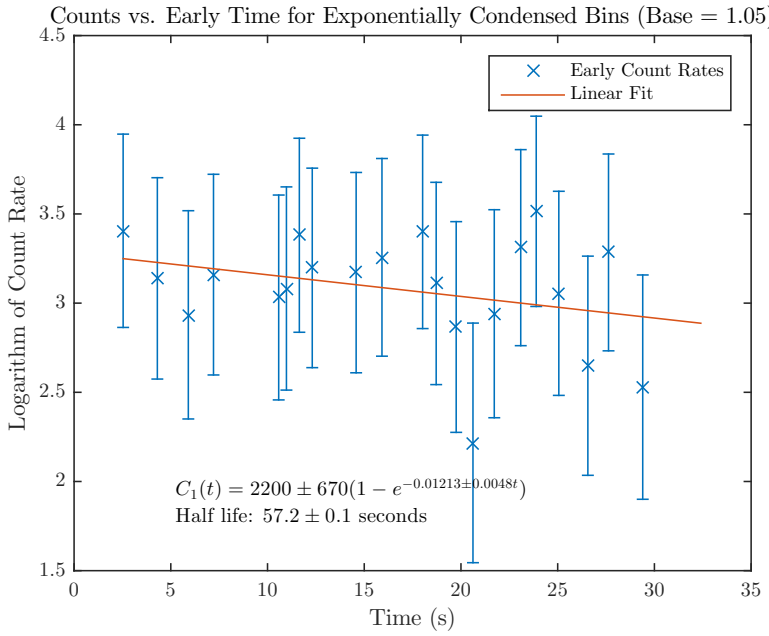

Figure 8: Linear fit to the logarithm of the count rate for early times (with contributions from the slow decay subtracted) for exponentially size bins with base 1.05. I chose early time to end at $t = 30$ by trying several values much smaller than the late time and seeing a clear linear pattern emerge.

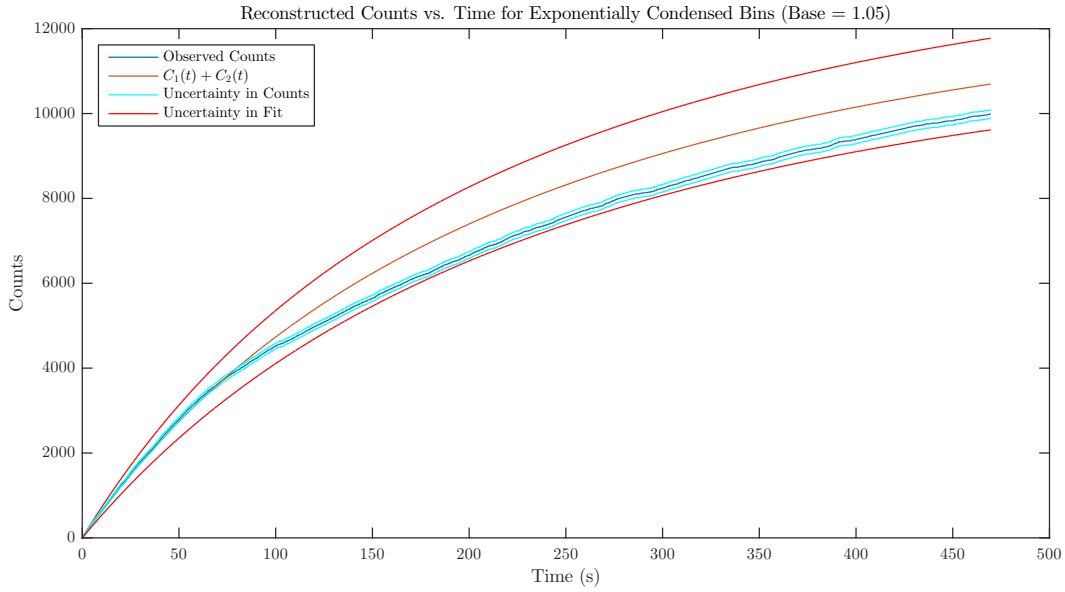

Figure 9: The reconstruction of the data from the exponential bins with base size 1.05. Note the systematic overcounting for late times. The plot of the uncertainty for the fit considers only the uncertainty in the coefficients of the exponents $N_1(0)$ and $N_2(0)$. Attempting to plot coefficient and exponential factor uncertainty together yielded strange results.

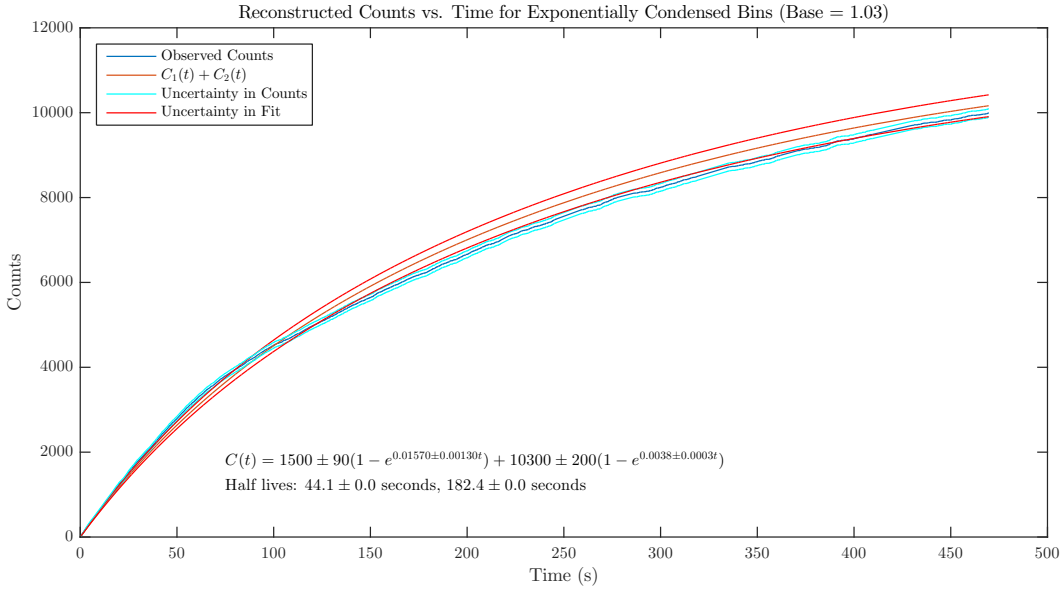

Figure 10: The reconstruction of the data from the exponential bins with base size 1.03. These smaller bins yielded a closer fit but still overcounted at late times. The quality of this fit is questionable - it undercounts at early times and overcounts at late times. Nevertheless I thought it was interesting. The plot of the uncertainty for the fit considers only the uncertainty in the coefficients of the exponents $N_1(0)$ and $N_2(0)$.

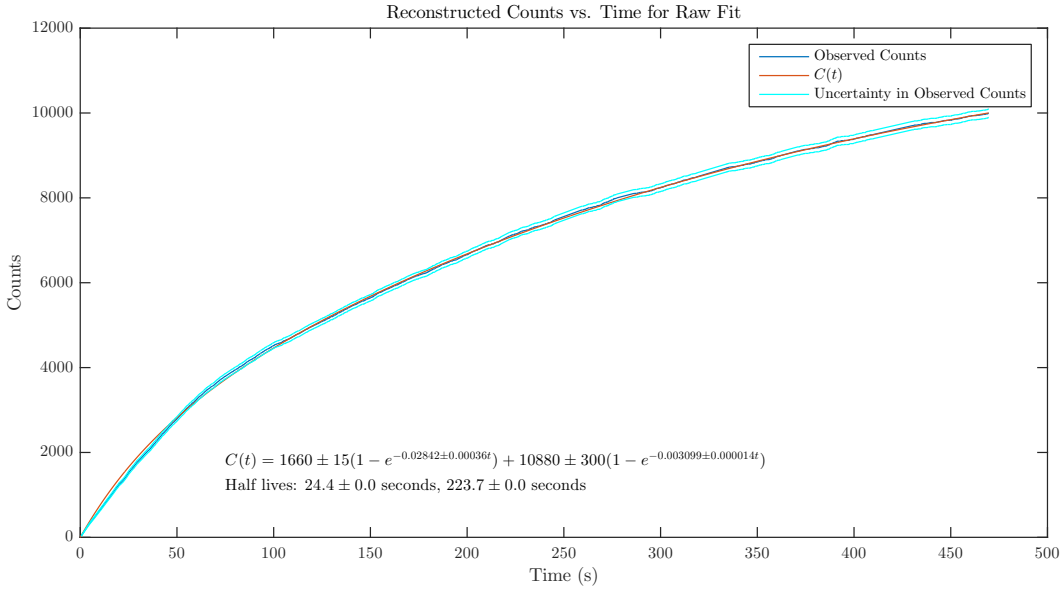

Figure 11: A parametrized direct fit using Matlab's curve fitting toolbox. The fit does very well for late times ($t > 50$ seconds) and overcounts slightly for early times ($t < 50$ seconds). The high time resolution of the data yielded incredibly small uncertainties in the fit.

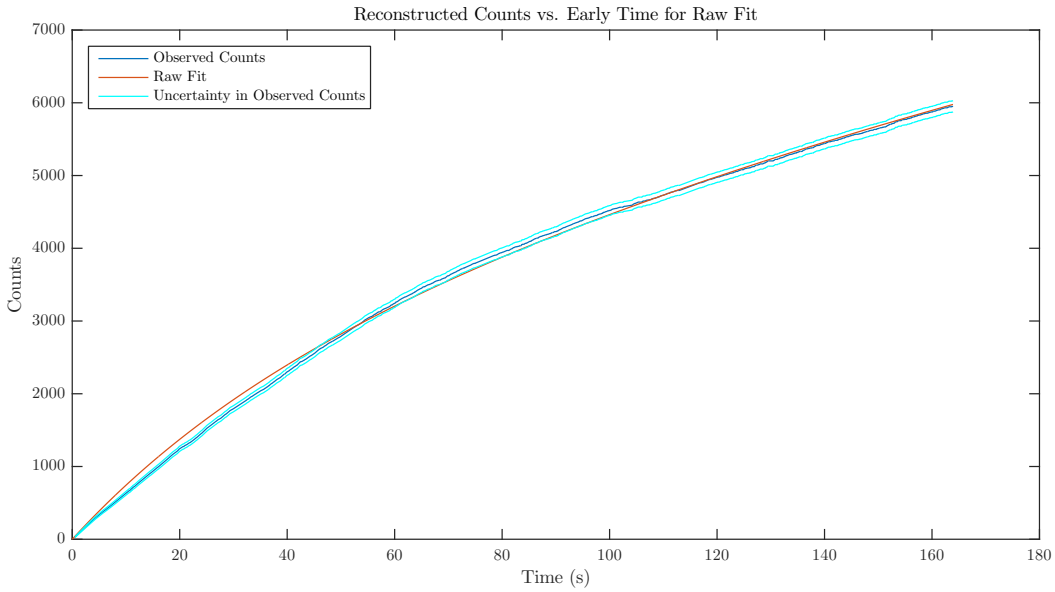

Figure 12: A closer look at the direct fit at early times. Note the early overcounting.

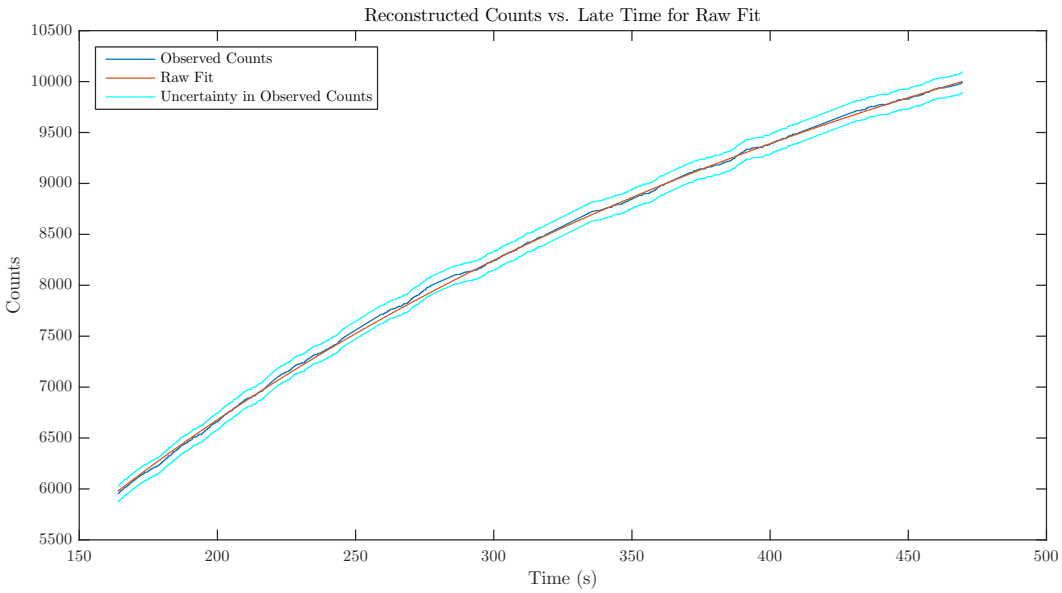

Figure 13: A closer look at the direct fit at late times.

For the log-fit with bins of base 1.05 the cross section is calculated to be:

$$\frac{10300 \pm 500}{2200 \pm 670} \cdot \frac{48.161}{51.839} = 4.68 \pm 1.65. \tag{12}$$

For the log-fit with bins of base 1.03 the cross section is calculated to be:

$$\frac{10300 \pm 200}{1500 \pm 90} \cdot \frac{48.161}{51.839} = 6.87 \pm 0.54. \tag{13}$$

For the direct fit the cross section is calculated to be:

$$\frac{10880 \pm 300}{1660 \pm 15} \cdot \frac{48.161}{51.839} = 6.09 \pm 0.22. \tag{14}$$

4 Discussion and Conclusions

4.1 Discussion of Results

The numerical fits to the condensed bins gave relatively high uncertainties. My best guess as to the origin of these uncertainties is from the high relative uncertainty between the value of the logarithm of the count rate and the small elapsed time provided to the linear least-square fitting algorithm. This high relative uncertainty could be reduced by parsing many data runs and taking averages across the runs.

In constrast the direct fit had much cleaner statistics. Once I figured out how to get the curve fitting toolbox to converge it was easy to get reasonable values for the parametrized fit. As I hoped, the uncertainties for the parameters were very small, reflecting well-parsed data that adhered to the model equation. This made the overhead of implementing the k-NN algorithm worthwhile - it took longer to write and sample the 40 images for reference than it would have to simply parse the video by hand with 10 second bin sizes, but I was able to parse several runs and pick a good long run in much less time overall.

4.2 Conclusions

The correct values of the half lives for ^{110}Ag, ^{108}Ag and the ratio of the cross sections are 24.4 seconds, 145.2 seconds, and 2.9, respectively. The log-linear fits missed the half lives considerably, but the cross section falls within the uncertainty for the first fit with bins of base 1.05. In contrast the direct fit nailed the short half life perfectly but missed the long half life and claimed to be incredibly certain about it. The direct fit also missed the cross section by a factor of two and was similarly confident.

Even accounting for the slightly higher detection rate of the higher-energy decay electrons which contributes to a slightly higher calculated ratio, this error is quite large. The spectacular miss of the longer decay time and production cross-section ratio is disappointing. Considering the accuracy with which former students were able to calculate these values using much coarser data collection methods, I had expected to do very well with my data analysis. I am surprised at how well the direct fit stays within the uncertainty of the observed counts at late times given how poorly it calculated the longer half-life, since that is the only contribution at those late times.

I attempted to explain this discrepancy by examining the contribution due to room background - in my initial data analysis I did not account for it explicitly. After directly subtracting its contribution I saw no difference in the analysis. To investigate further I simulated the effect of much higher frequency background and only found a substantial change when the frequency of background events reached 1 Hz, much higher than their actual incidence.

Given more time I would average decay counts over several runs and try a direct fit again. Perhaps a longer run would be justified as well, since it seems there were plenty of atoms left to decay at the end of the run - the count rate had not quite leveled out yet.

References

[1] E. Cassel. *Physics 310 Study of Nuclear Beta Decay.* Nov. 1988.

[2] Ben Taskar. *Nearest Neighbor Methods*. URL: https://web.archive.org/web/20100712115558/http://alliance.seas.upenn.edu/~cis520/wiki/index.php?n=Lectures.LocalLearning (visited on 05/21/2016).

[3] K. Weinberger. *Lecture 2: k-nearest neighbors*. URL: http://www.cs.cornell.edu/courses/cs4780/2015fa/web/lecturenotes/lecturenote02.html (visited on 05/21/2016).