Experiment 2:
Analysis of Multi-dimensional Experimental Datasets using Python

Neuron Calcium Activity Inside the C. elegans Brain

Maya Amit
Lab Partner: Hayley Calloway
Northeastern University Department of Physics

Abstract
Fluorescence intensity data of *C. elegans* neurons expressing the GCaMP protein were analyzed to quantify photobleaching effects. Time-series images were collected and preprocessed using a box averaging blurring filter in OpenCV to reduce noise. A binary mask, centered at (18, 22) with a radius of 5 pixels, was applied to isolate the brightest regions of the neuron. The 0.85 intensity quantile was used to define the roughly-average pixel intensity for each frame, and data was aggregated by averaging the this intensity per 30 frames. A clear fluorescence decay was observed, and a double exponential model was fitted to the data. The best-fit parameters revealed an initial rapid decay within the first 200 seconds, followed by a slower, more gradual decline. The final fit parameters were $A_{\text{OLS},1} = 3.840$, $\beta_{\text{OLS},1} = 0.02041$, $A_{\text{OLS},2} = 0.732$, $\beta_{\text{OLS},2} = 0.02041$, and $C_{\text{OLS}} = 2.283$. These results highlight the impact of photobleaching and demonstrate the effectiveness of image processing techniques in fluorescence quantification.

1 Introduction

Modern physics experiments often yield large amounts of multidimensional data. Processing this data is, therefore, necessary for interpreting results. Python is a particularly useful language for working with data of this nature because of it's intuitive handling on array of arrays. An array of arrays is just a way of organizing data in a structured form, like a table with rows and columns. Instead of having a single list of numbers, one can have a list where each item is another list (or array). This is useful when dealing with data that has multiple dimensions.

A common example of multidimensional data is a sequence of multi-color images recorded over time. This can be represented as a three-dimensional array, where each individual image is a two-dimensional grid of pixel intensity values, and the third dimension corresponds to time. Conceptually, this structure is similar to a stack of papers, where each sheet is an image, and the stack's thickness represents the number of frames in the sequence. A python array or arrays allows for efficient storage and processing of time-dependent image data. Furthermore, the Open Source Computer Vision Library, or OpenCV, is a library used for image processing and analysis that supports Python. It is indeed useful for handling large datasets consisting of stacks of images, specifically for object detection and filtering.

These datasets are both large and often contain noise and complex patterns that cannot be identified manually. This experiment explores Python's ability to handle and process such data efficiently using array-based structures[4].

2 Background

2.1 C. elegans and GCaMP

The multidimensional dataset used in this experiment is a series of images of a neuron in Caenorhabditis elegans, or C. elegans. C. elegans are small, transparent round worms with a simple nervous system that are ideal for studying in biological and genetic research. To image this neuron, it is genetically modified or treated to express the fluorescent protein GCaMP. When calcium ions bind to GCaMP, the protein fluoresces green. This makes it possible to monitor calcium levels, which are linked to neuronal activity. When imaged, the florescence (which manifests in brighter pixels in the image) is directly indicative or neuron activity.

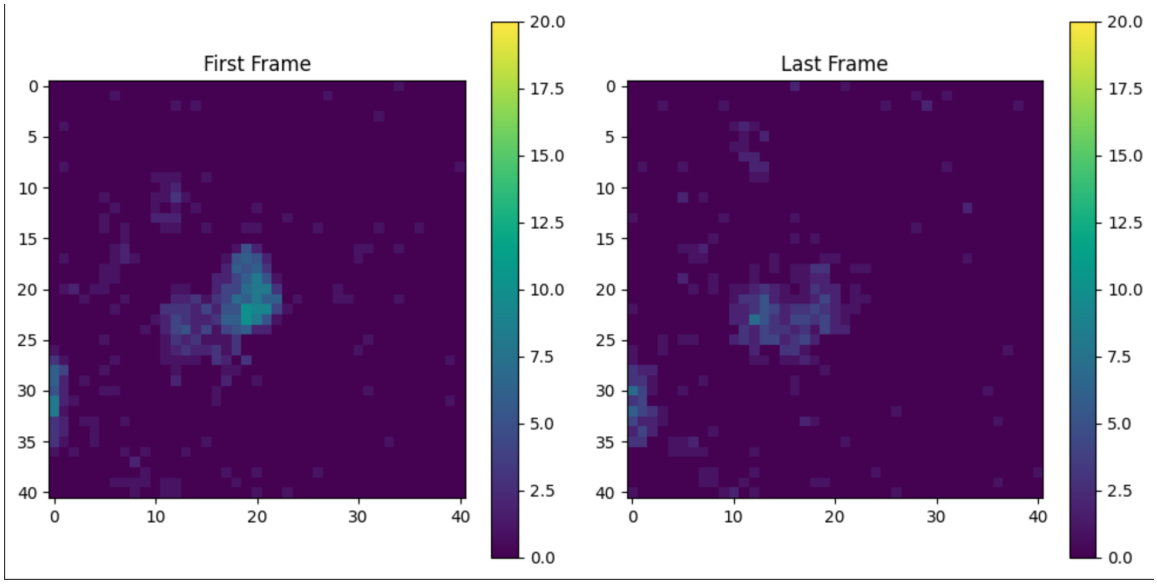


Figure 1: First and Last Frames in C. Elegans dataset

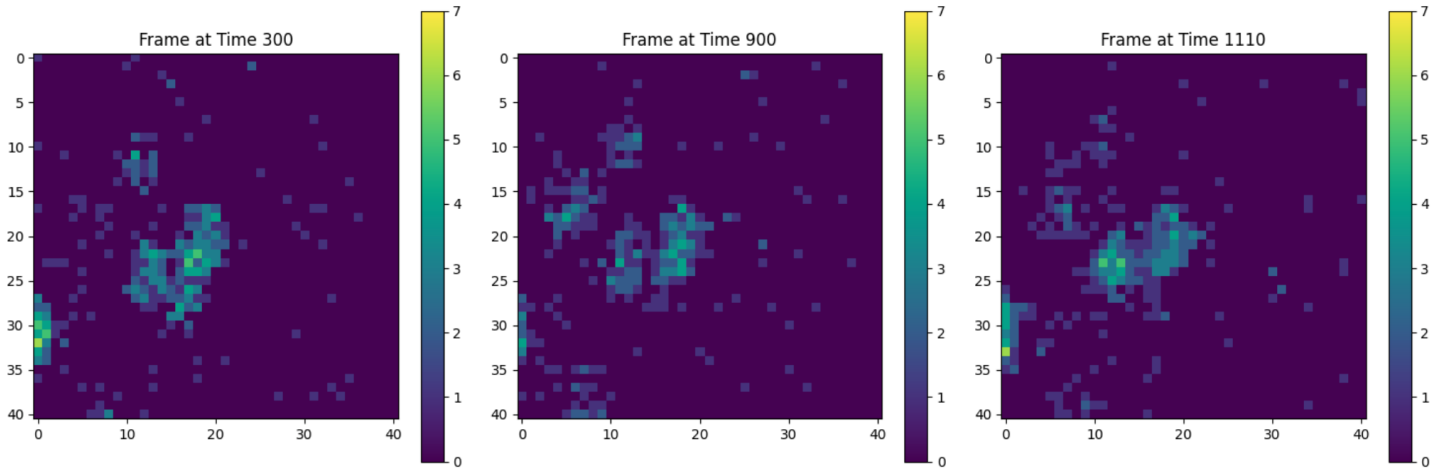


Figure 2: Three snapshots to show photobleaching over time

2.2 Photobleaching

However, the fluorescence of GCaMP-tagged neurons gradually decreases due to prolonged exposure to intense light[3]. GCaMP proteins fluoresce when excited by specific wavelengths, but repeated excitation over time chemically damages the fluorescent molecules. This process, known as photobleaching, results in a progressive loss of brightness. When analyzing the array of images, the fluorescence within the neuron will visibly diminish over time, and the rate of this decay will be determined.

3 Materials and Methods

3.1 The Data

To begin the analysis, the data is read into a Python notebook and stored as the variable `img_gcamp`. This allows for easy access and manipulation of the dataset. For example, calling `print(img_gcamp)` will

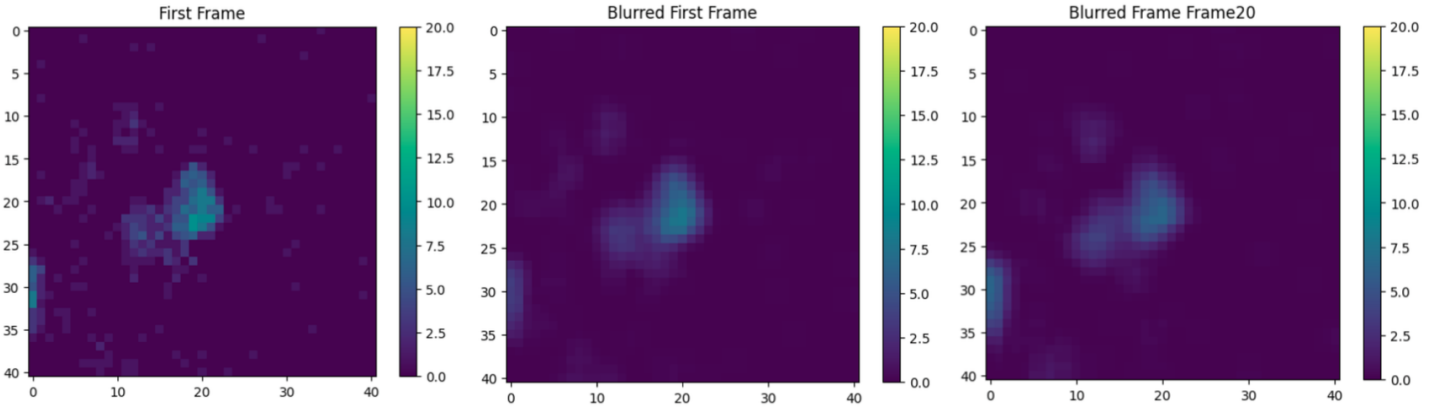


Figure 3: Original first frame, blurred first frame and 20th frame with Box Averaging

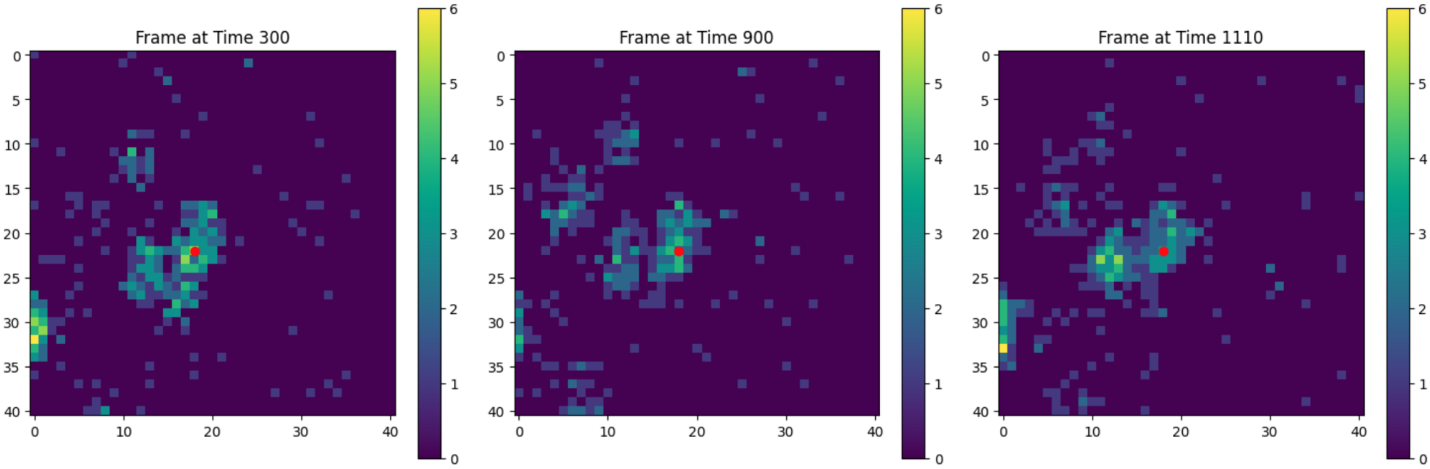


Figure 4: Frame at time 300, 900, and 1110 with a red dot at 18,22

display the data. The structure of the data is as follows: the first axis corresponds to time (frames), the second axis represents the image width, and the third axis represents the image height. The size of this data set is 1650 by 41 by 41. Specific values can be accessed using indexing, a feature of Python arrays. For instance, `img_gcamp[500, 10, 100]` retrieves the pixel located at the 10th row, 100th column, in the 500th frame of the dataset.

The first and last frames are shown in Figure 1, with the color bar serving as a reference for pixel intensity. It is evident that the neuron's activity remains relatively low.

3.2 Noise in the Data

A major component of image processing and analysis is noise mitigation. Noise often originates from camera sensors and appears as randomly bright or dark pixels that do not reflect the true physical conditions. A common approach to noise reduction is blurring, which involves averaging a pixel's intensity with that of its neighboring pixels. If an isolated bright pixel appears in a predominantly dark region, it is likely a noise fluctuation. By averaging it with surrounding pixels, its intensity is reduced. The same goes for suspiciously dark pixels in a sea of bright ones. This spatial averaging helps suppress outliers and enhances the accuracy of the recovered signal. There are several ways to perform averaging by weighting the surrounding pixels differently. For example, using a Gaussian distribution assigns less weight to pix-

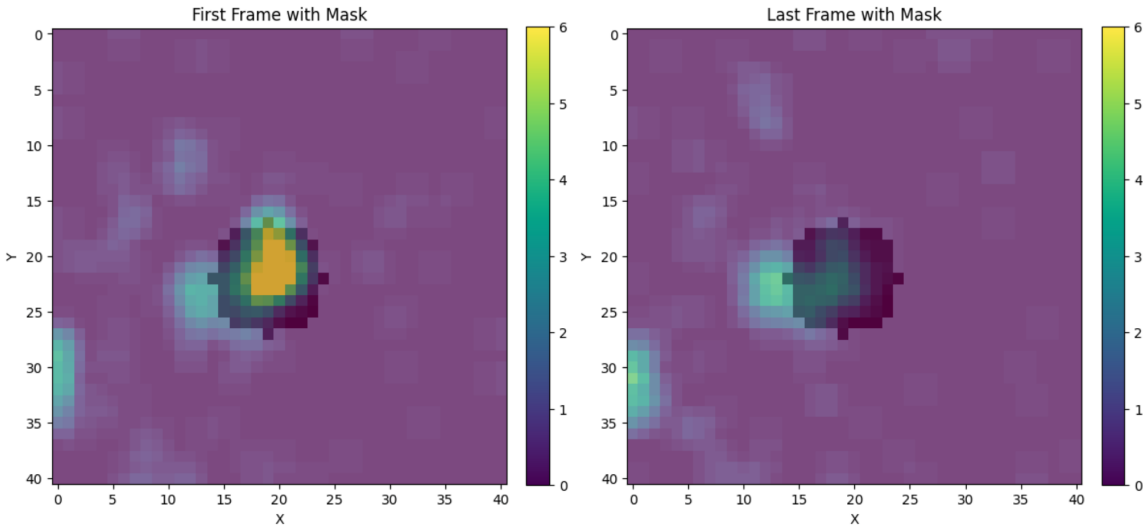


Figure 5: First and last frame with applied mask

els farther from the central pixel, as distant pixels should have less influence on the smoothing process. However, since the noise in this experiment is relatively sparse, a simpler weighting method is used. The box averaging method, in this case, forms a 3×3 grid around each pixel, where all neighboring pixels are weighted equally.

OpenCV is used to perform this blurring, as it provides a built-in blur function, `cv.blur`, that simplifies the implementation of box averaging[2]. A simple function `blur_images` is made that takes in an array or arrays (the many frames in this data set) and calls `cv.blur` iteratively through all the frames, returning an array of blurred frames. This function also takes in an integer parameter that represents a given frame. When called with the value 20, for example, this function also plots the 20th blurred frame. Figure 3 illustrate the blurred first frame, alongside the original first frame for reference. It also displays the 20th blurred frame, although any of the roughly 1600 frames can be displayed.

3.3 Localizing the Cell and Masking

To better analyze the effects of photobleaching, it is useful to focus on pixels corresponding to bright regions within the cell. A simple way to achieve this is by applying a mask. The mask is a binary array with the same dimensions as a single frame, where most pixels are assigned the boolean value False, except for a selected small circular region, which is set to True. Following Python's logical operations:

$$value \times True = Value \tag{1}$$

$$value \times False = 0 \tag{2}$$

Multiplying the mask by an image frame results in an image where the selected region retains its original intensity values, while all other pixels are set to zero. Repeating this process for each frame in the sequence produces an array of masked frames with dimensions (Number of frames, mask dimension).

The optimal size and location of the mask were determined to be at position [: , 18, 22] with a radius of 5 pixels. This selection involved some trial and error, aiming to ensure that the mask consistently captures signal across all frames while avoiding dark regions outside or at the cell's edges. Additionally, the chosen size allows for the inclusion of spatial variations within the cell, which may result from the image noise. Figure 4 shows that the mask's center coordinates remain within the cell throughout the duration of the

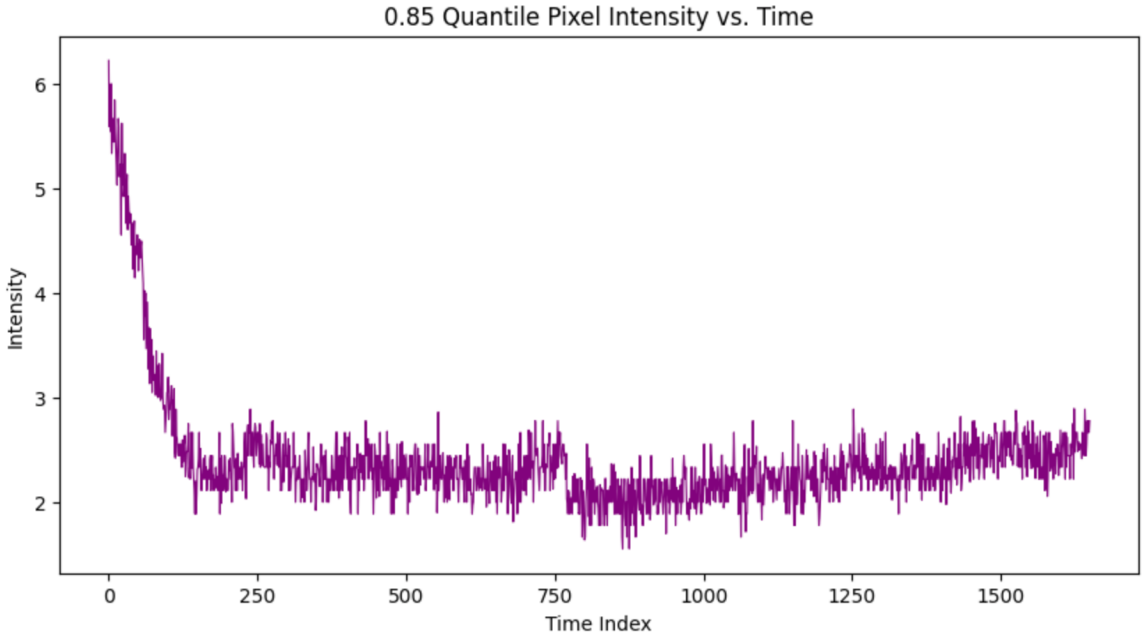


Figure 6: 85th quantile pixel intensity over time

experiment. Figure 5 illustrates the mask overlaid on both the first and last frames. However, since the mask remains stationary while the cell gradually shifts to the left over time, it does not perfectly track the signal, leading to a slight misalignment in later frames. Note that the color bar is manually decreased to artificially enhance the signal for the purposes of tracking the signal. This increased contrast does not represent the true variations in intensity, and is only created for visual purposes.

3.4 Intensity Over Time

Now that only relevant pixels remain in the data frame, their intensity can be analyzed. Specifically, examining the average intensity of cell pixels across frames helps assess the effects of photobleaching. Instead of using the average, a more robust statistical measure—the quantile approximation—is applied. Given a quantile value X, the quantile of a group of pixels represents the intensity below which X% of the pixels fall. In this experiment, the 85th percentile is calculated for each frame to better capture intensity variations while reducing the influence of outliers. This value will be referred to as the "representative intensity" of each frame. Figure 6 illustrates how the representative intensity changes over time, revealing a key trend: intensity decays exponentially as time progresses. This decay is a direct consequence of photobleaching. To enhance visual clarity, the data is aggregated. Figure 7 presents the same data, but each point represents the average of 30 data points, reducing noise and making the trend more apparent.

3.5 Exponential Fit

Fitting the aggregated data to an exponential function reveals the rate at which photobleaching causes a decline in cell signal intensity. The ordinary least squares (OLS) method, a type of linear regression, is used to fit a curve to the data. The OLS method minimizes the "sum of the squared residuals (deviations or errors) between the observed data points and the fitted curve" [1]. The fit, then, is by definition weighted by the standard error.

To begin this fitting process, a general exponential fit function, `func_exp`, is defined such that it takes in

t, A, and beta, and returns:

$$y(t) = Ae^{\frac{-\beta \times t}{100}} \tag{3}$$

Where A is a the initial intensity and and β is the decay rate. The objective now is to find the best-fit parameters (A_ols and β_ols) for the exponential function `func_exp`. To do so, an initial guess is provided where A is the maximum intensity and β is a guessed small decay. The `curve_fit` function, then, optimizes these parameters using the OLS method, given the aggregated data, the exponential function, the initial guess, and a time scale.

The initial attempt at curve fitting was unsuccessful for several reasons. First, the fitted curve trends toward zero, whereas the actual data plateaus at an intensity of around 2. More critically, the intensity drops sharply within the first 200 seconds, but the fitting process appears to treat these early data points as outliers. As a result, the estimated decay rate is too small, causing the fit to resemble a nearly linear trend rather than capturing the expected exponential decay.

The solution to this issue is to fit the data using a double exponential function, which accounts for both the rapid initial drop and the more gradual decay that follows. This model consists of two exponential terms—one capturing the sharp initial decline and the other representing the slower long-term decay. The double exponential function is defined as:

$$y(t) = A_1 e^{\frac{-\beta_1 \times t}{100}} + A_2 e^{\frac{-\beta_2 \times t}{100}} + C \tag{4}$$

Where A_1 and A_2 are the initial amplitudes of the two exponential components, β_1 and β_2 are the corresponding decay rates ($\beta_1 > \beta_2$) and C is an offset that accounts for the plateauing behavior of the data. Respectively, the initial guesses for this trial were $3.0, 0.05, 1.5, 0.001$ and 2.2. The resulting best fit parameters were: $A_{\text{OLS},1} = 3.840$, $\beta_{\text{OLS},1} = 0.02041$, $A_{\text{OLS},2} = 0.732$, $\beta_{\text{OLS},2} = 0.02041$, and $C_{\text{OLS}} = 2.283$. The single line and double line fit are displayed in Figures 8a and 8b respectively. Note that the mean intensity is still the 85th quantile intensity, but averaged over every 30 frames.

4 Discussion

It is surprising that such a large portion of the decay occurs within the first 200 seconds, as this suggests that a substantial fraction of molecules are particularly susceptible to photobleaching almost immediately after exposure. This rapid drop may indicate that a subset of GCaMP molecules are in a highly excitable or unstable state, making them more prone to damage than expected. Alternatively, external factors such as light intensity or imaging conditions may be contributing to this pronounced early decay.

The observed plateau suggests that a portion of the fluorescence remains stable over time, potentially due to the presence of the molecules that fluoresce in regions that are less exposed to direct excitation or due to a baseline level of background fluorescence. However, the overall trend confirms that photobleaching significantly affects the fluorescence signal, which must be accounted for when analyzing neuronal activity. These results highlight the necessity of photobleaching correction methods when working with time-series fluorescence imaging.

Regardless of the unexpected rapid intensity decay, this analysis provided valuable insights into key image processing techniques. Blurring with OpenCV was utilized to reduce noise and smooth intensity variations, enhancing the clarity of fluorescence trends. Additionally, masking effectively isolated the region of interest,

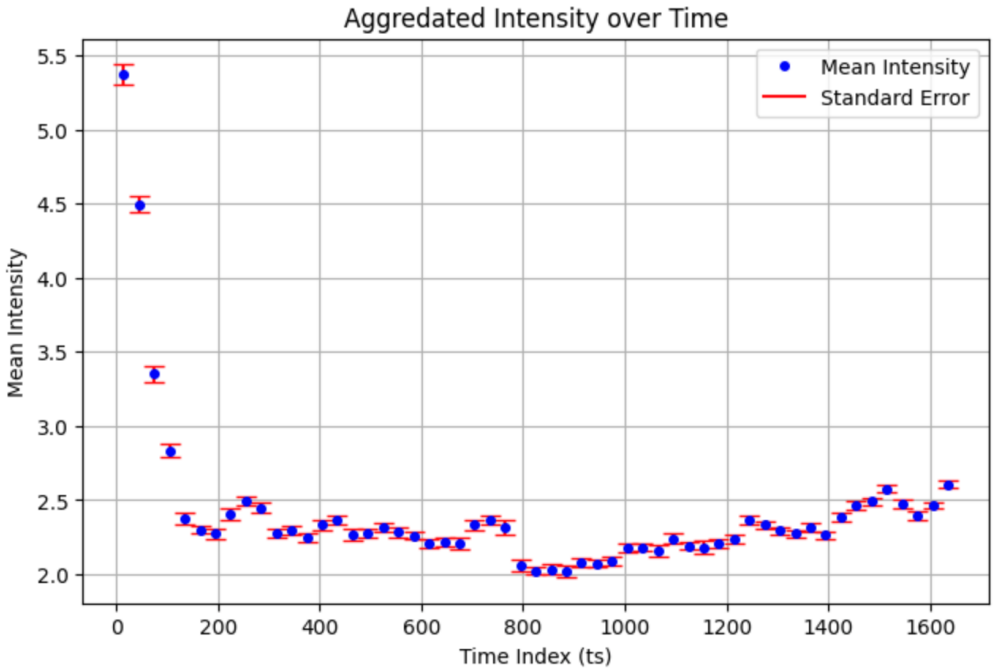


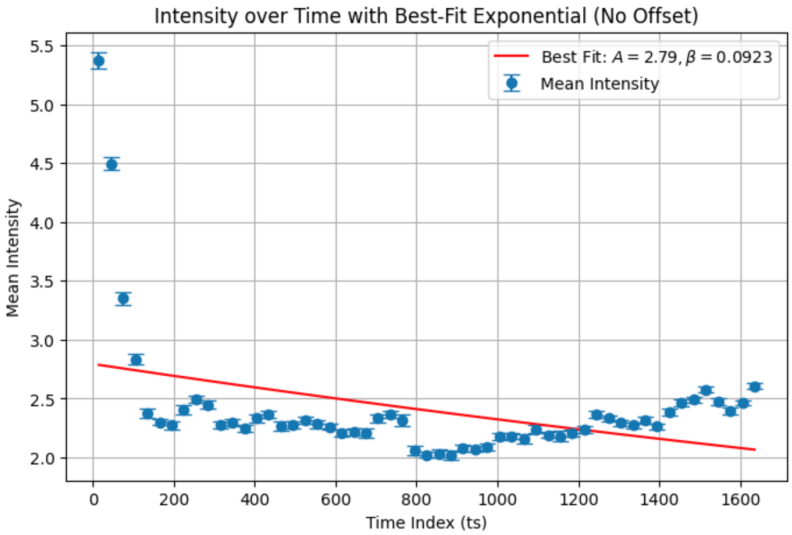
Figure 7: Aggregated 85th quantile pixel intensity over time

ensuring a focused analysis of photobleaching while minimizing background interference. These techniques are essential for improving the accuracy and reliability of fluorescence imaging in biological research.

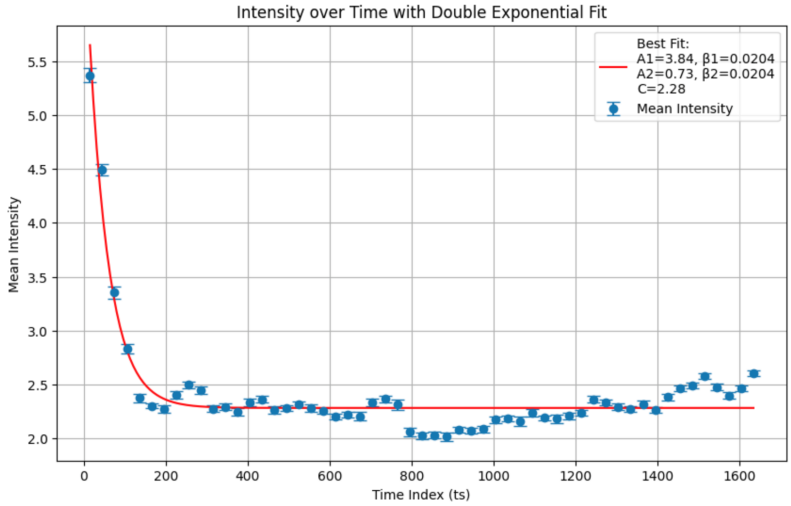
5 Conclusion

The fluorescence intensity data of *C. elegans* neurons expressing the GCaMP protein were analyzed to investigate photobleaching effects. Applying a box averaging blurring filter with OpenCV reduced image noise, while the masking process effectively isolated the bright regions of the cell. A binary mask, centered at (18, 22) with a radius of 5 pixels, was applied to each frame, preserving intensity values only within the selected region. This ensured consistent signal extraction throughout the experiment, though the stationary mask introduced slight misalignment in later frames as the cell gradually shifted left.

A clear decline in fluorescence was observed due to photobleaching. To quantify this decay, a double exponential model was fitted to the fluorescence intensity data. The results revealed two distinct phases: a rapid initial drop in fluorescence within the first 200 seconds, characterized by the larger amplitude component, followed by a slower, more gradual decay captured by the second exponential term. The best-fit parameters were found to be $A_{\text{OLS},1} = 3.840$, $\beta_{\text{OLS},1} = 0.02041$, $A_{\text{OLS},2} = 0.732$, $\beta_{\text{OLS},2} = 0.02041$, and $C_{\text{OLS}} = 2.283$.



(a) Aggregated pixel intensity over time with single exponential fit



(b) Aggregated pixel intensity over time with two exponential fits

Figure 8: Comparison of single and double exponential fits applied to aggregated pixel intensity over time.

References

[1] BYJU'S. Least square method - definition, graph, and formula, 2025. Accessed: 2025-02-04.

[2] OpenCV. Image processing filters - opencv documentation, 2025. Accessed: 2025-02-04.

[3] Thermo Fisher Scientific. Photobleaching principles - fluorescence fundamentals, 2025. Accessed: 2025-02-04.

[4] Northeastern University. Experiment 2: Analysis of multi-dimensional experimental datasets using python, 2025. Accessed: 2025-02-04.