

# Dissertation

Francesco Tamburi

October 2023

# Todo list

|  |    |
|--|----|
| expand paragraph . . . . .                           | 4  |
| reference to justification, placed earlier . . . . . | 12 |
| possibly expand . . . . .                            | 12 |

# Introduction

# Results

## 0.1 First analysis

Using PALSSIM, a series of spectra containing two positron lifetimes,  $\tau_1$  and  $\tau_2$ , were generated with a three gaussian instrument resolution function (see Table 1 and Figure 1).  $\tau_1$  was kept fixed at 180ps and  $\tau_2$  varied from 220-280ps, in 10ps intervals. For each value of  $\tau_2$ , three spectra were generated with the relative intensities of  $\tau_1$  and  $\tau_2$  set to 20%-80%, 50%-50% and 80%-20%. All the resulting spectra were then analyzed using PALSFIT, in order to evaluate how well the program could extract the two lifetimes, and their respective intensities, from each simulated spectrum.

In Figure 2a, we can observe how the values of  $\tau_1$  outputted by the program change, as the simulated value of  $\tau_2$  (on the horizontal axis) increases and the relative intensities (indicated by color and shape of marker) vary. Zooming in to the  $\tau_2 = 250$ -270ps range, we can see in Figure 2b that, for these values in particular, PALSFIT struggles to determine  $\tau_1$  in the 50-50 and 80-20 case.

Unlike for  $\tau_1$ , where the simulated value of the lifetime is fixed, the simulated value of  $\tau_2$  (our point of comparison) changes in between spectra. To make the data easier to visualize, rather than the fitted value of  $\tau_2$ , the difference between the result and the original is plotted instead (see Figure 2c). In the figure we can see that, aside from the 20-80 spectrum for  $\tau_2 = 220$ , the software performs better than for  $\tau_1$ .

The error bars represent the standard deviation of – and thus the confidence of the program in – the lifetimes. From them, we see two factors that affect the size of the bars. The first is the relative intensities of the two intensities. In fact, in Figure 2a, which plots  $\tau_1$ , the error bars are the smallest for the 80-20 data, where the shorter lifetime is more intense, and in Figure 2c, tracking  $\tau_2$ , the opposite is the case and we have the 20-80 data is most precise. Observing all the figures, we see the second factor: as the time interval between the two lifetimes increases, the size of the error bars decrease.

In Figures 2d - 2f the fitted intensities of the two lifetimes are plotted against simulated  $\tau_2$ , for each combination of simulated intensities. In the figures, we see that when the relative intensity of  $\tau_2$  was greater or equal to  $\tau_1$ , then PALSFIT was able to calculate all the appropriate lifetime intensities. However, when the intensity of  $\tau_2$  was set to 80%, shown in Figure 2d, the software

Table 1: Instrument resolution function

| FWHM (ps) | Shift (ps) | Intensity (%) |
|-----------|------------|---------------|
| 213.3     | 0          | 80            |
| 150       | -5         | 10            |
| 267       | 17         | 10            |

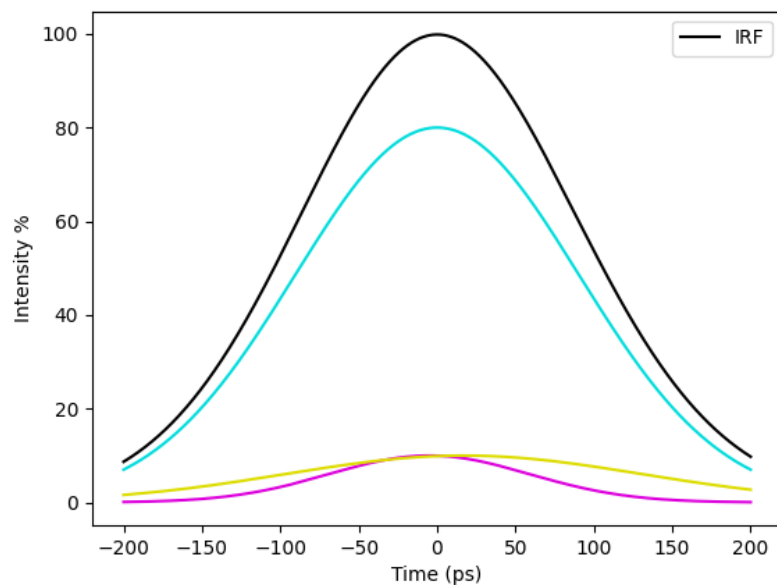


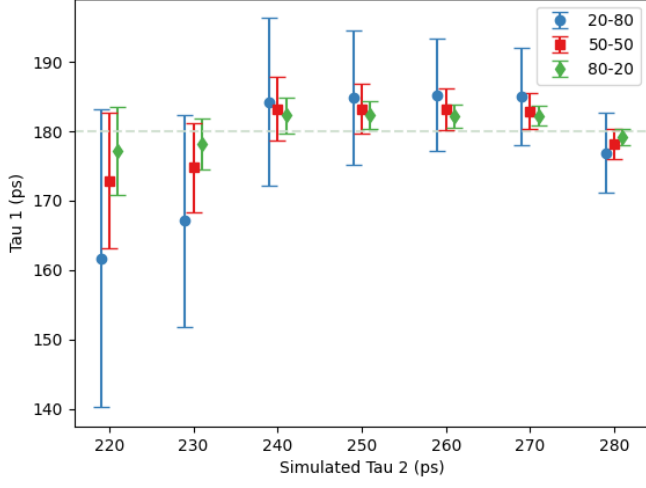
Figure 1: instrument resolution function

was unable to output the correct intensities for the first two simulated values of  $\tau_2$ . Looking at our error bars, we can see the same relationship between their size and the lifetime separation mentioned earlier.

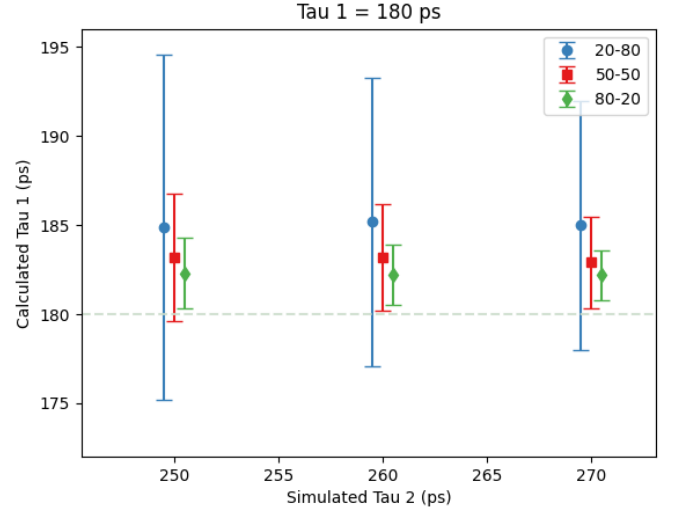
On the whole, PALSFIT seems to have performed well, but not perfectly. As the first lifetime,  $\tau_1$ , was kept constant at 180ps, the next step would be to change  $\tau_1$  and see how that affects our results.

expand  
paragraph

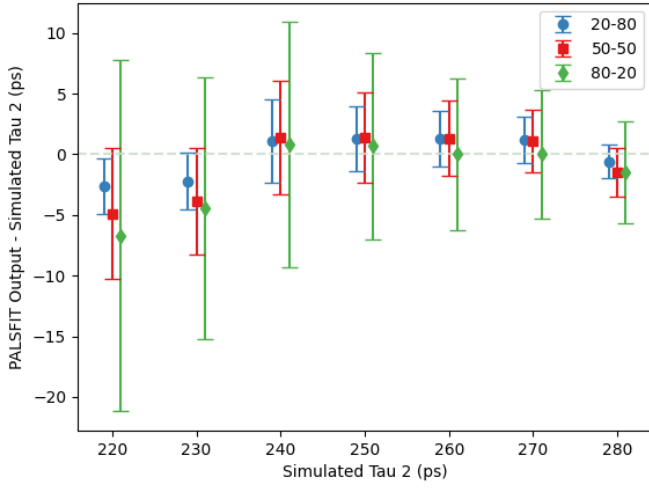
Figure 2



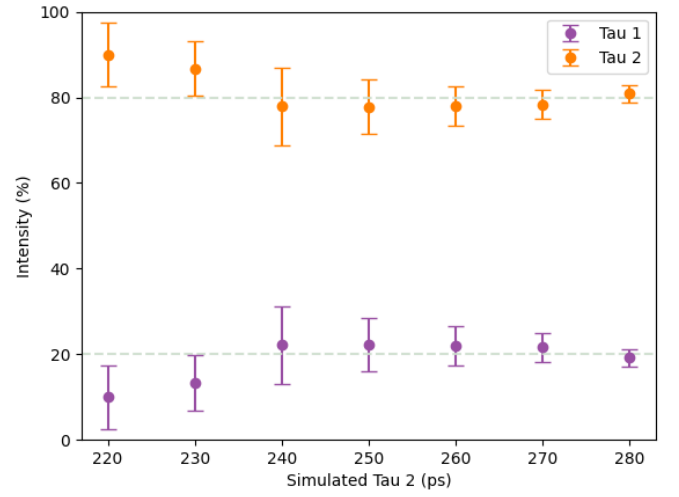
(a) fixed  $\tau_1 = 180ps$



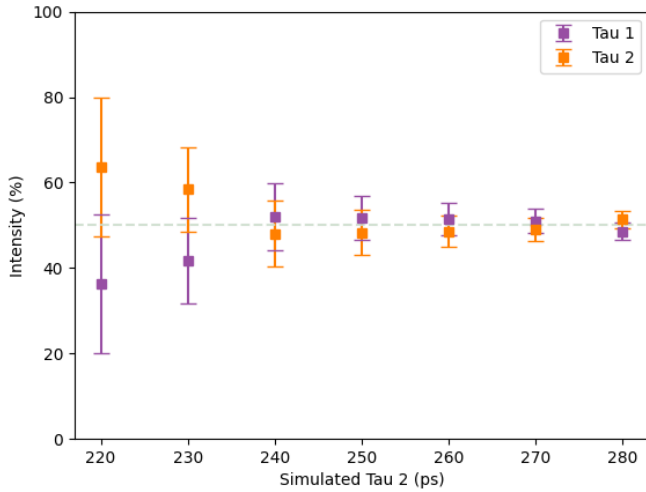
(b) close-up  $\tau_1 = 180ps$



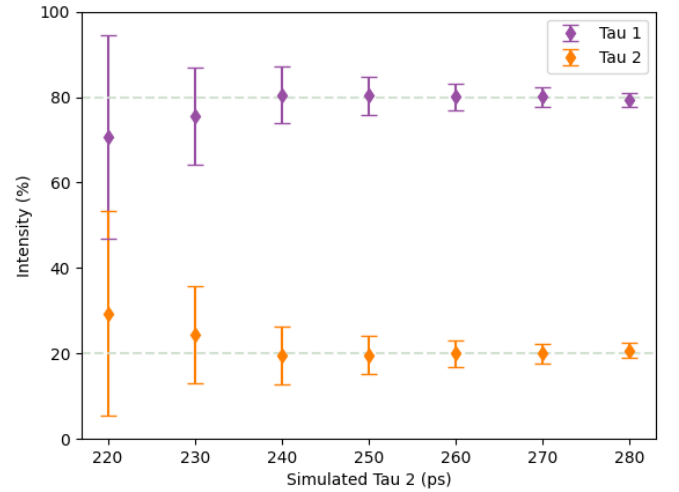
(c)  $\tau_2$  difference



(d)  $\tau_1 = 20\%, \tau_2 = 80\%$



(e)  $\tau_1 = 50\%, \tau_2 = 50\%$



(f)  $\tau_1 = 80\%, \tau_2 = 20\%$

## 0.2 Modifying $\tau_1$

A similar procedure was performed for  $\tau_1=150$  and  $\tau_1=220$ . To keep the relative time interval the same in between batches, the corresponding spacing between  $\tau_1$  and the  $\tau_2$  range was kept consistent. For  $\tau_1 = 150$ , this meant a  $\tau_2$  range of 190-250ps, and for  $\tau_1 = 220$ , this meant a corresponding  $\tau_2$  range of 260-320ps.

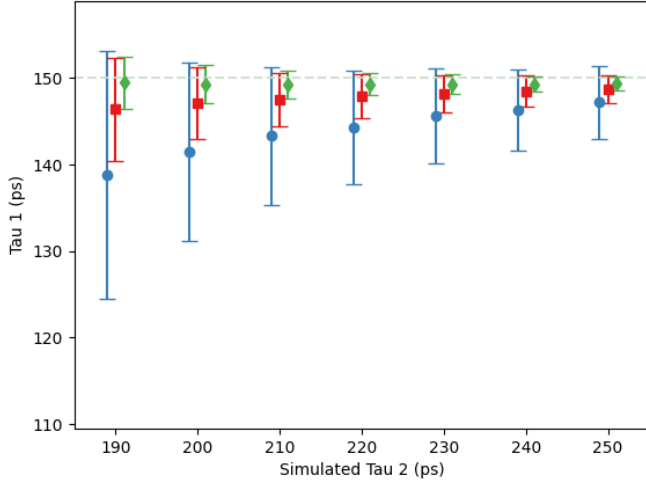
This was first done for  $\tau_1 = 150$ . As can be seen in Figure 3, the results for this batch are all within error, with both accuracy and precision getting better as the lifetime separation increases, in line with what would be expected.

The other batch, meanwhile, where  $\tau_1$  was set to 220ps, was not as successful. The results can be seen in Figure 4, but in general, PALSFIT struggled with fitting the lowest values for  $\tau_2$  and the error bars are noticeably larger. The program even gives nonsensical results when  $\tau_1 = 260$ ps and the relative  $\tau_1$ - $\tau_2$  intensity is set to 80%-20%, as can be seen in Figures 4a, 4b and 4e.

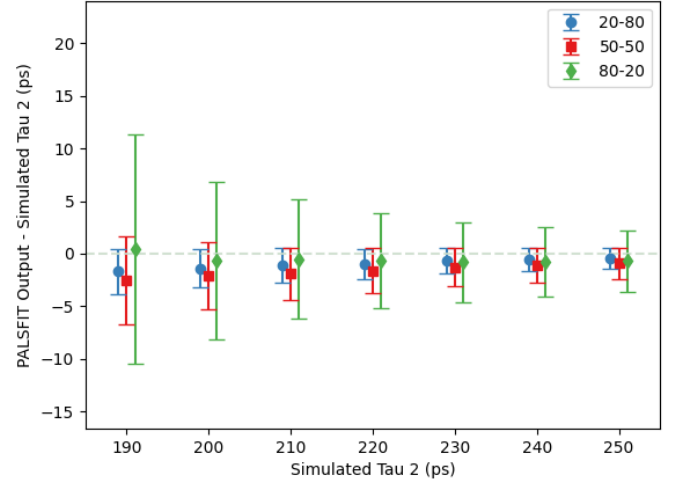
In order to compare all three datasets, the absolute difference between the fitted and simulated values of each relevant variable,  $\tau_1$ ,  $\tau_2$  and intensity was plotted. Additionally, plots for their respective standard deviations were generated. The resulting plots are represented in Figures 5-7. Two general observations are apparent from analysis of these figures:

The first is that the  $\tau_2 = 180$ ps data seems much more scattered than the other two datasets, which both follow a relatively clear increase in precision and accuracy as the lifetime separation increases. The second is that as  $\tau_1$  decreases, PALSFIT seems better able to resolve the two lifetimes, with both the difference from the true value and the size of the error bars being the smallest when  $\tau_1 = 150$ ps.

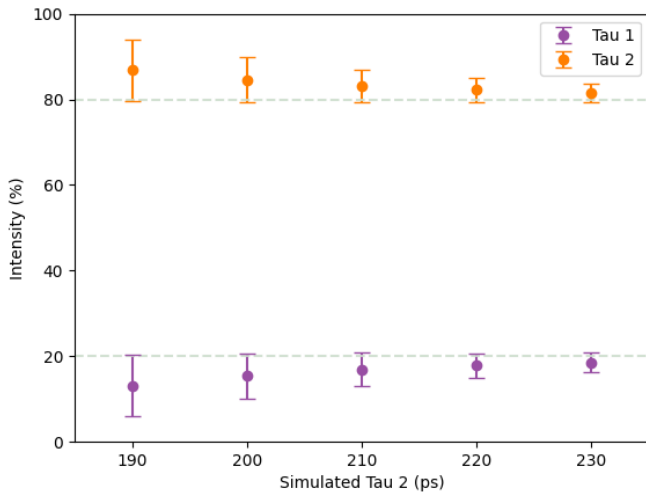
Figure 3



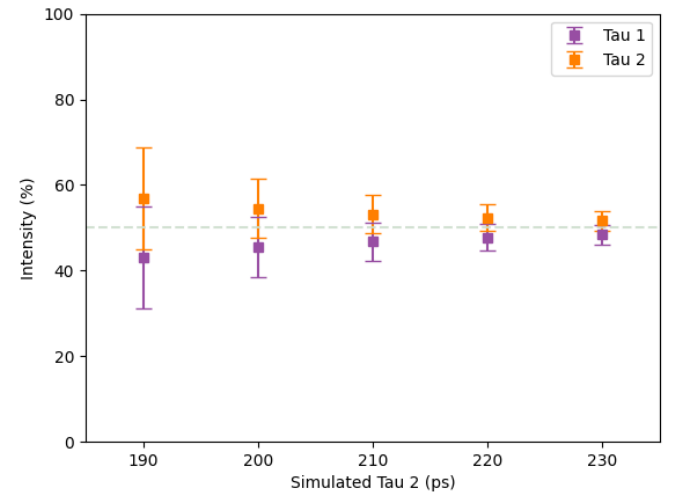
(a) fixed  $\tau_1 = 150$  ps



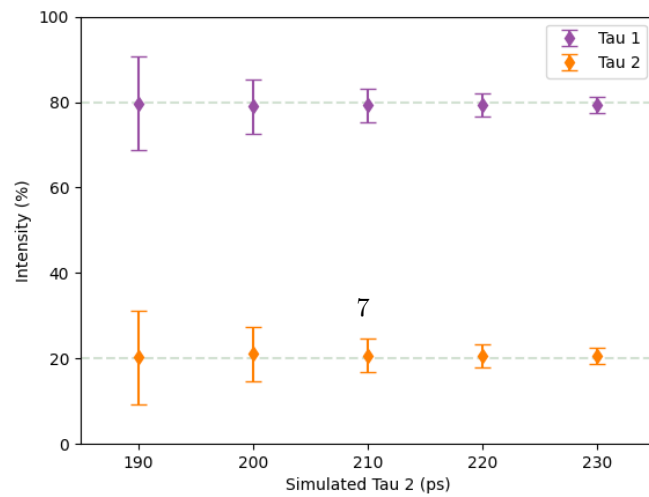
(b)  $\tau_2$  difference



(c)  $\tau_1 = 20\%$ ,  $\tau_2 = 80\%$



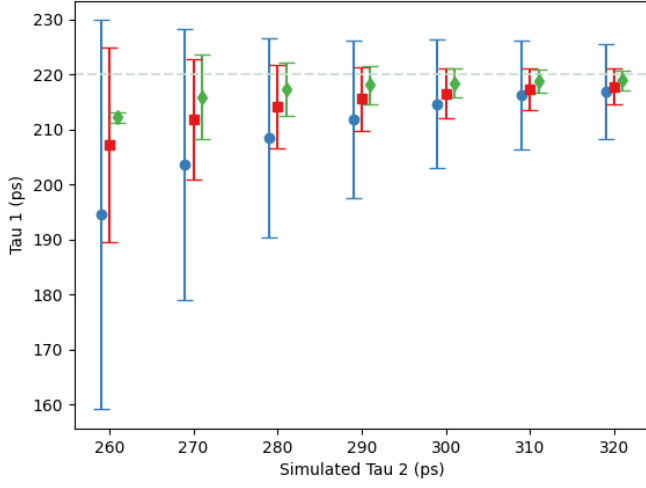
(d)  $\tau_1 = 50\%$ ,  $\tau_2 = 50\%$



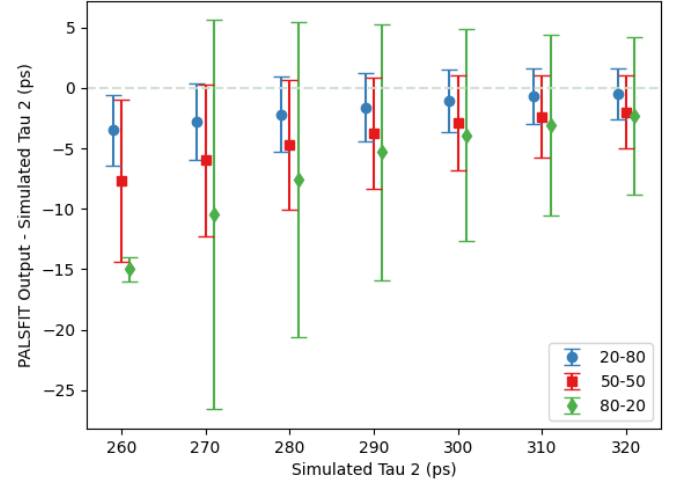
(e)  $\tau_1 = 80\%$ ,  $\tau_2 = 20\%$



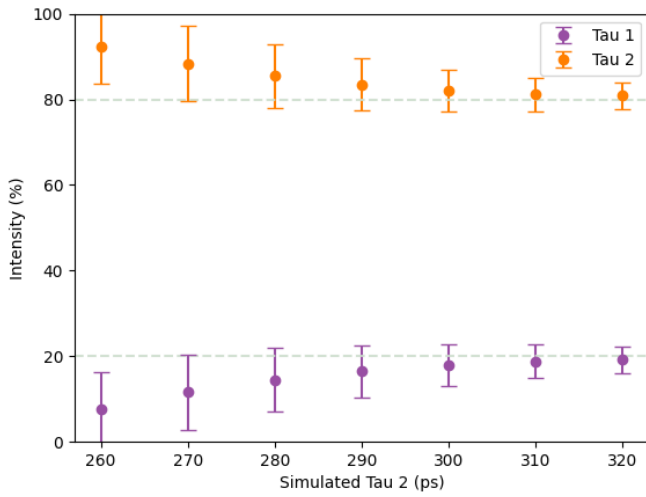
Figure 4



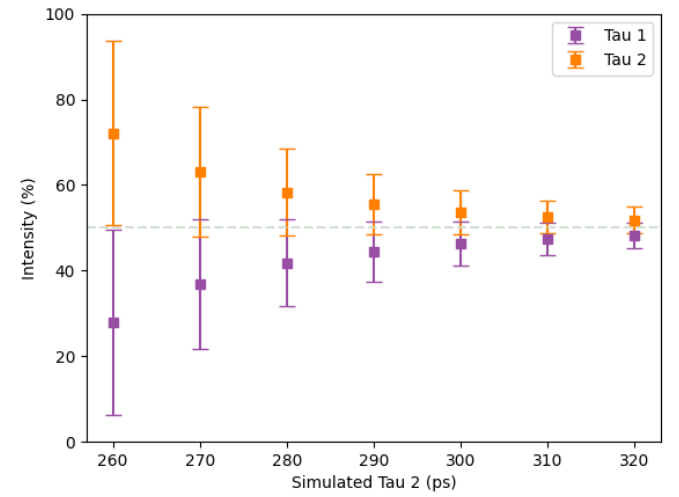
(a) fixed  $\tau_1 = 220ps$



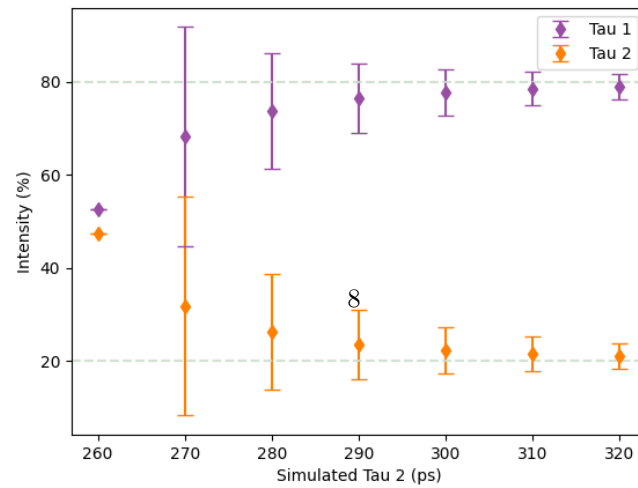
(b)  $\tau_2$  difference



(c)  $\tau_1 = 20\%, \tau_2 = 80\%$

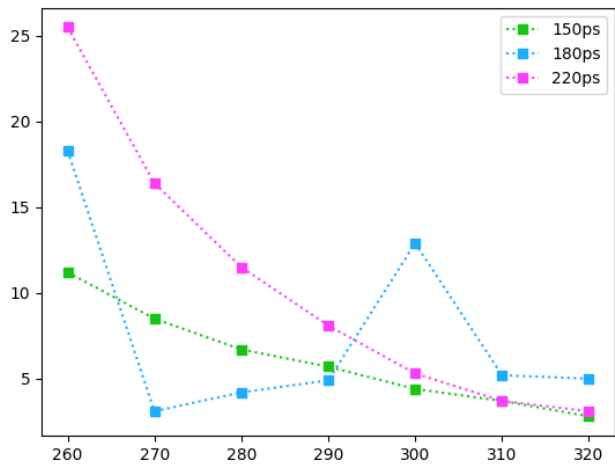


(d)  $\tau_1 = 50\%, \tau_2 = 50\%$

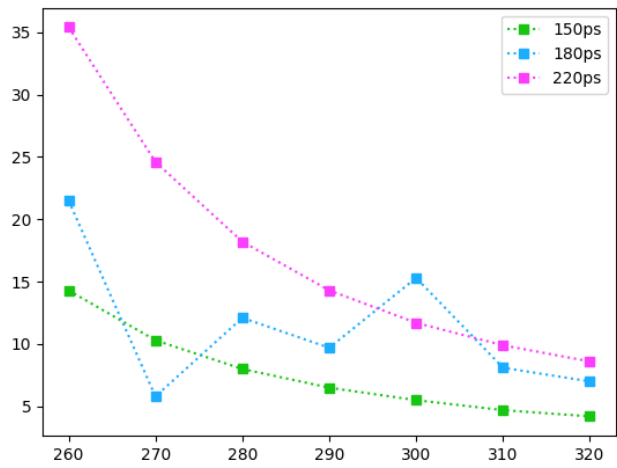


(e)  $\tau_1 = 80\%, \tau_2 = 20\%$

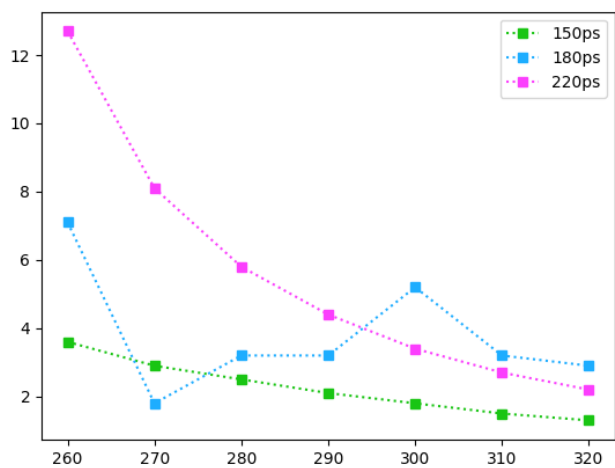
Figure 5:  $\tau_1$ , rows = 20-80, 50-50, 80-20, columns = diff, std dev



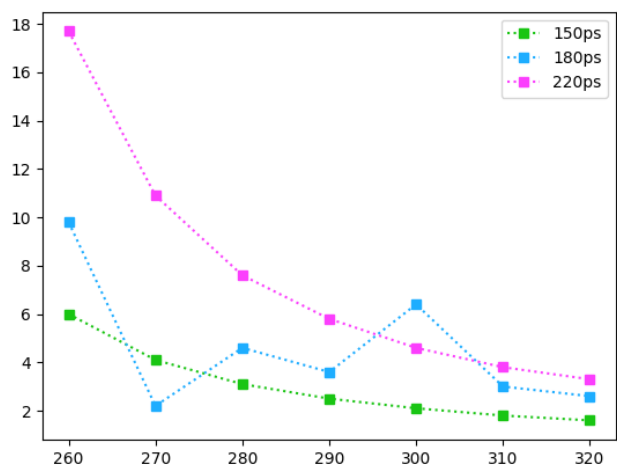
(a)



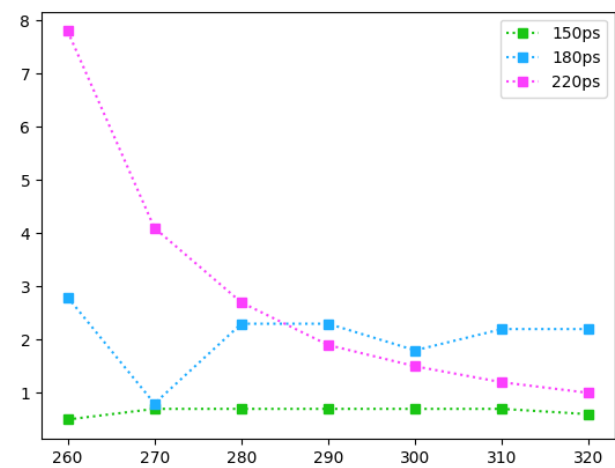
(b)



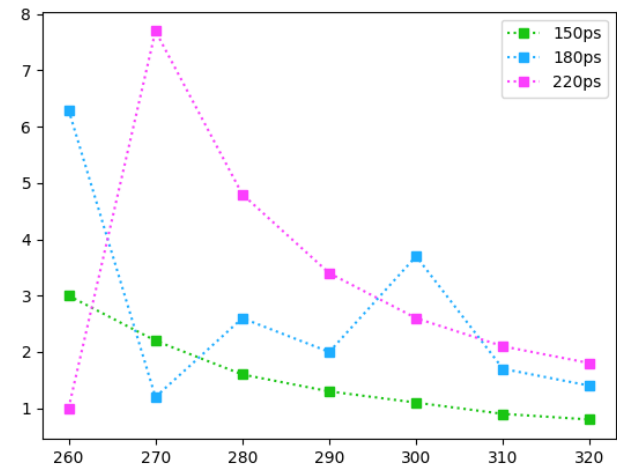
(c)



(d)

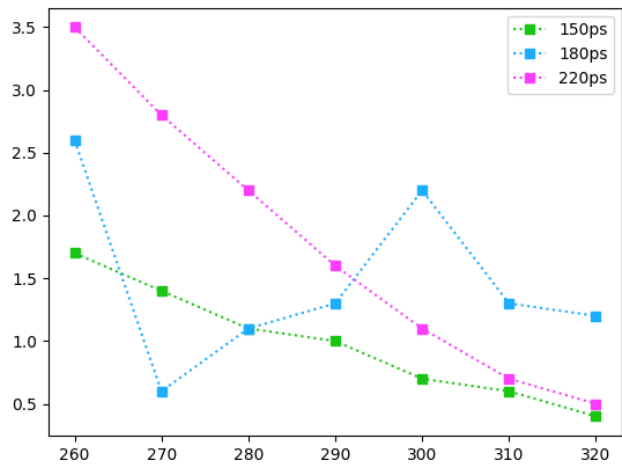


(e)

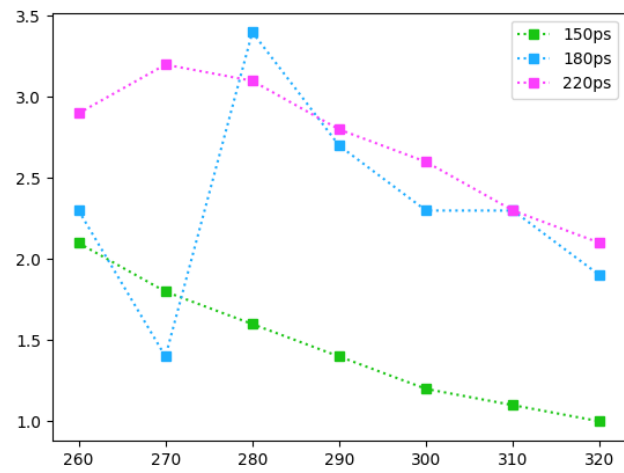


(f)

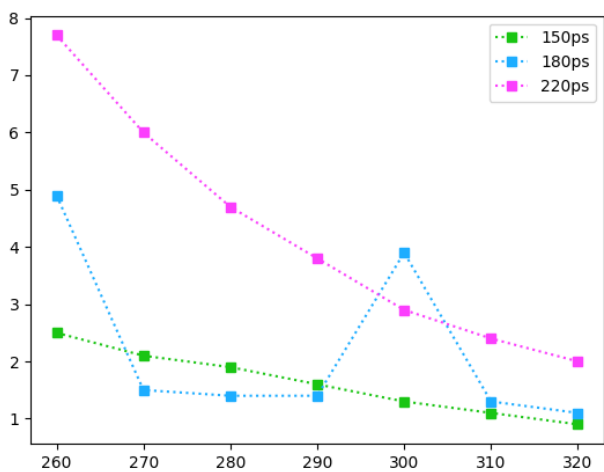
Figure 6:  $\tau_2$ , rows = 20-80, 50-50, 80-20, columns = diff, std dev



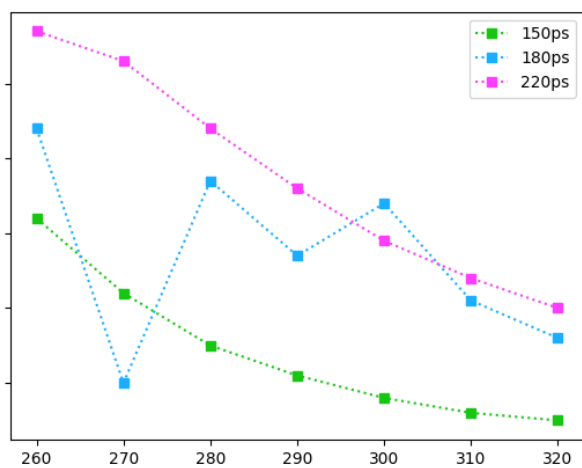
(a)



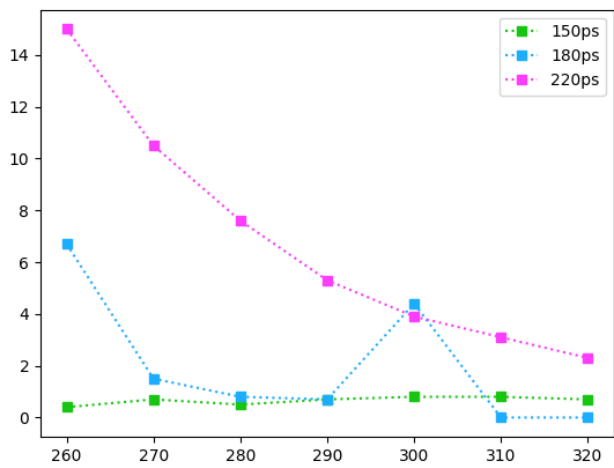
(b)



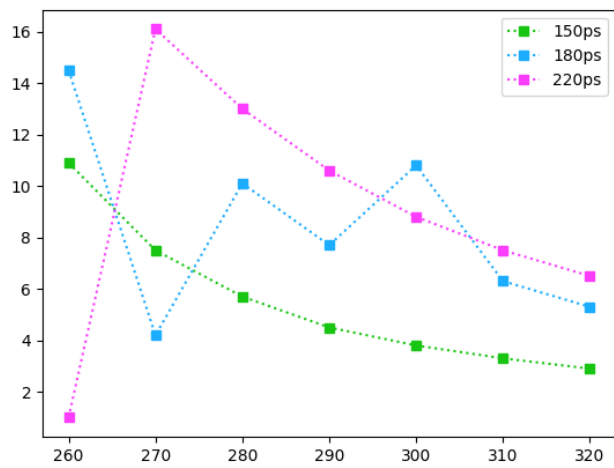
(c)



(d)

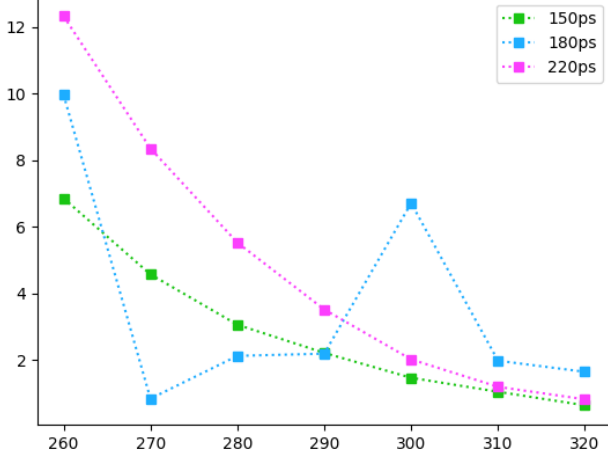


(e)

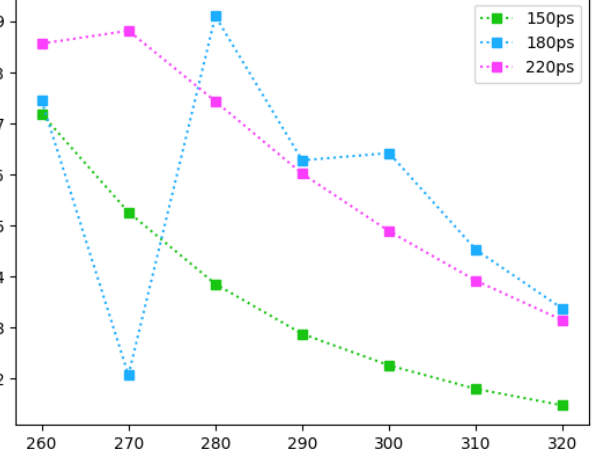


(f)

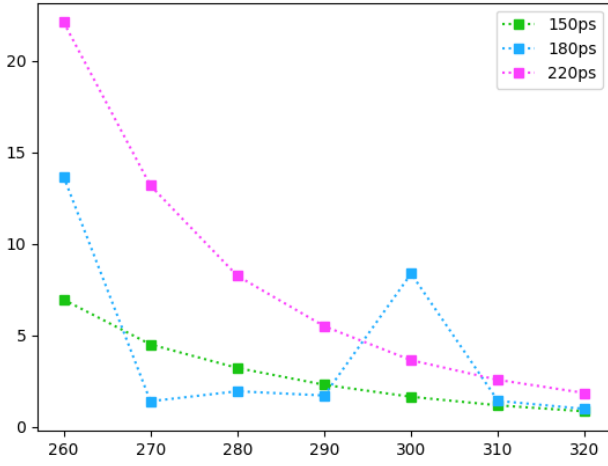
Figure 7: intensities, rows = 20-80, 50-50, 80-20, columns = diff, std dev



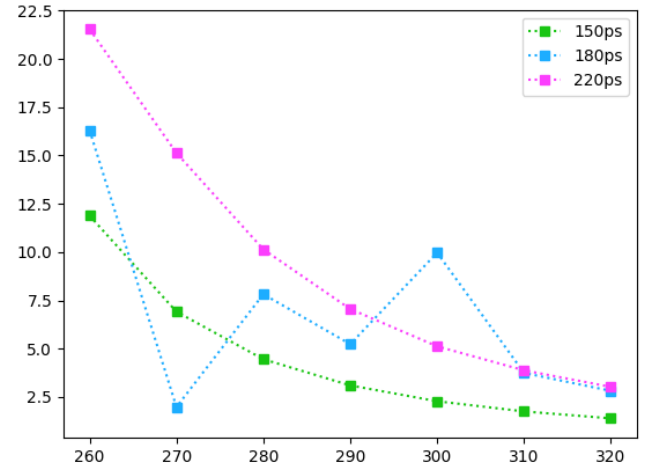
(a)



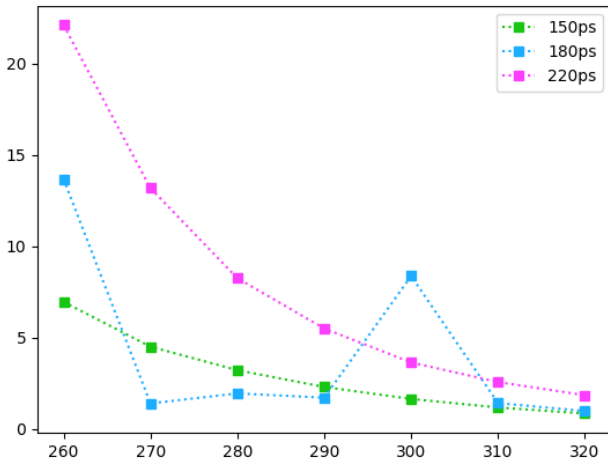
(b)



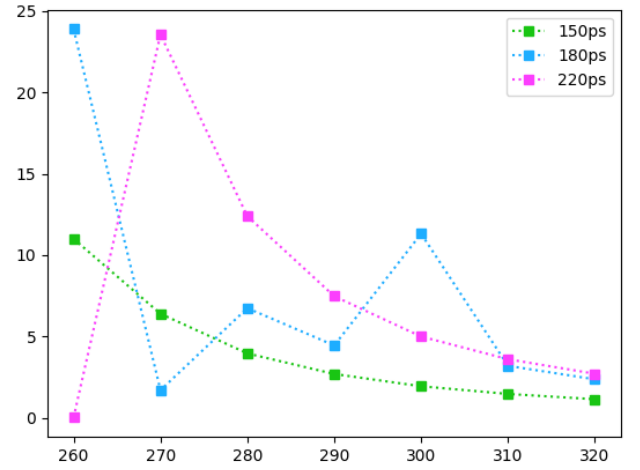
(c)



(d)



(e)



(f)

### 0.3 Single Gaussian IRF

To investigate how the instrument resolution function affects the software, the three Gaussian resolution function in use so far can be simplified to a single Gaussian (refer to ...) . As a sanity check, we can compare how the program performs with a single Gaussian resolution function versus the previous three Gaussian resolution function. Setting  $\tau_1 = 150\text{ps}$ , as this previously gave us the best results, we get the data in Figure 8. We can then compare this to Figure 3 and see that the single Gaussian resolution function produces remarkably similar results, demonstrating the single Gaussian approximation to be valid.

reference to justification, placed earlier

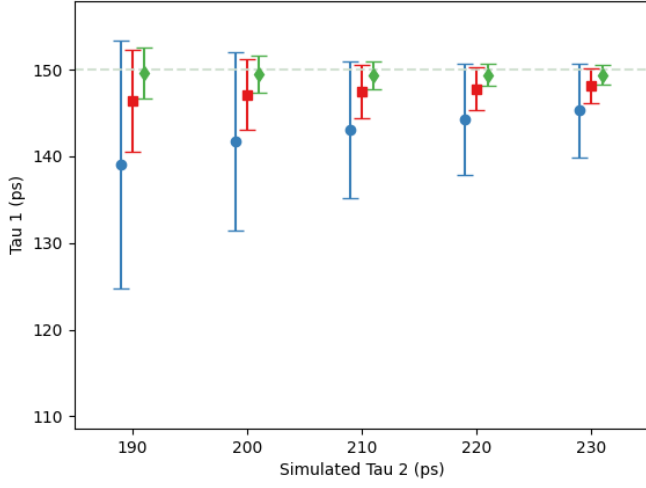
With that established, spectra were generated and fitted for FWHM values of 220ps, 180ps, 150ps and 100ps, with  $\tau_1$  set to 150ps and  $\tau_2$  ranging from 180-230ps. In general, the trend we expect is the same increase in precision and accuracy with increasing  $\tau_2$  that we see previously, but also an increase in both with narrower resolution functions. We would see this as multiple, separate, decreasing trendlines, ordered so that those corresponding to wider resolution functions were always higher. The results we get are shown in Figures 9-11, and we can analyse them in two parts.

If we just look at the subfigures on the left ((a),(c) and (e) for each figure), which represent the difference between the fit and original data, we see that for the 20%-80% and 50%-50% data the data follows the expected trend. When the  $\tau_1$  intensity is set to %80, however, we notice that accuracy doesn't seem to follow the expected relationship with the width of the resolution function. In many cases we have an outright reversal of the predicted relationship and in each figure we have significant crossing between the trendlines.

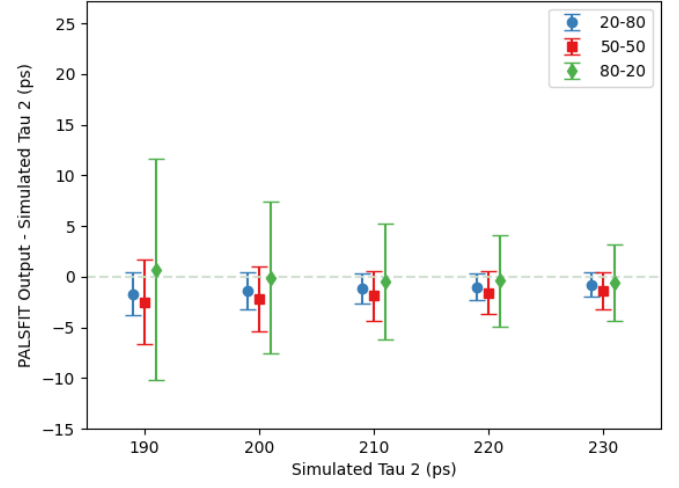
Looking at the subfigures on the right, representing the standard deviation of the fitting given by PALSFIT, we again see three outliers where the predicted relationship between precision and resolution function width is reversed, in Figures 10b, 10d and 11b, though in the case of 10d, this reversed relationship quickly reverts to the expected one as  $\tau_2$  increases.

possibly expand

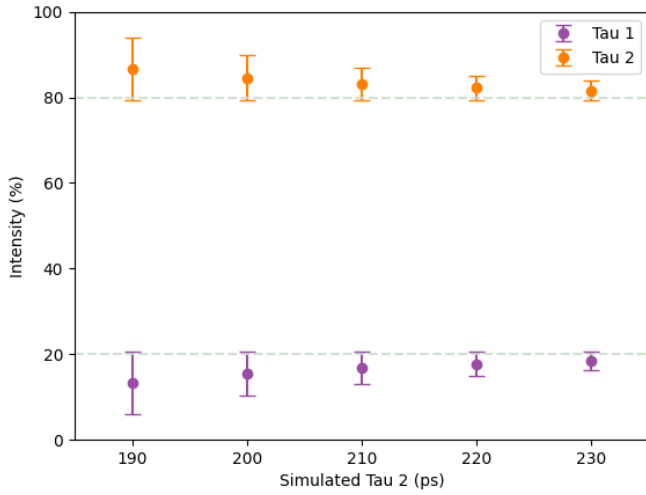
Figure 8



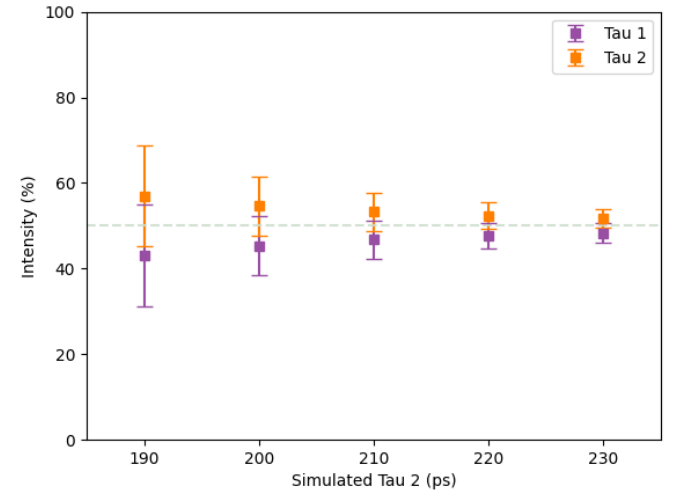
(a) fixed  $\tau_1 = 150$  ps



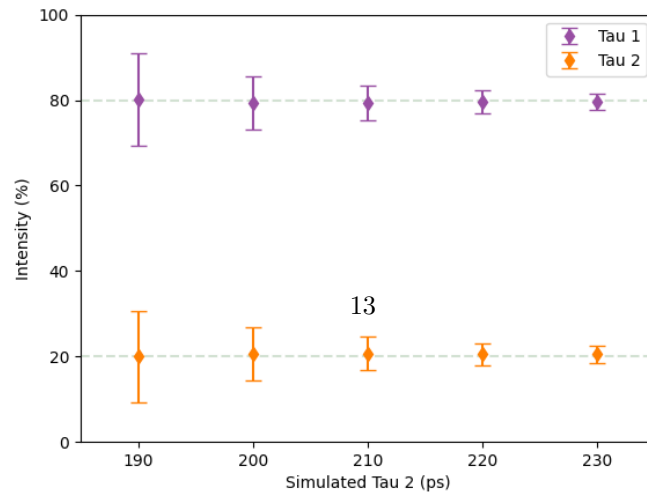
(b)  $\tau_2$  difference



(c)  $\tau_1 = 20\%$ ,  $\tau_2 = 80\%$

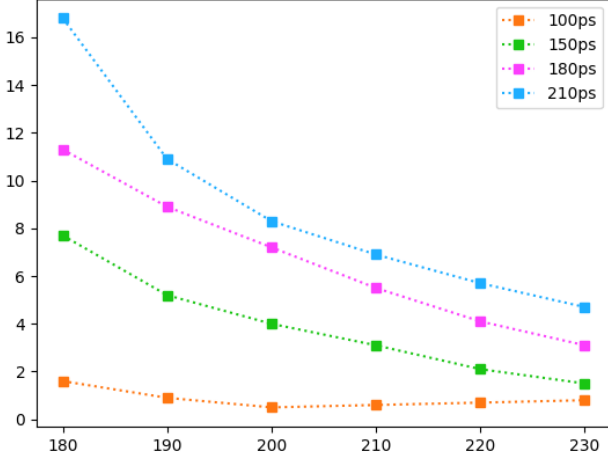


(d)  $\tau_1 = 50\%$ ,  $\tau_2 = 50\%$

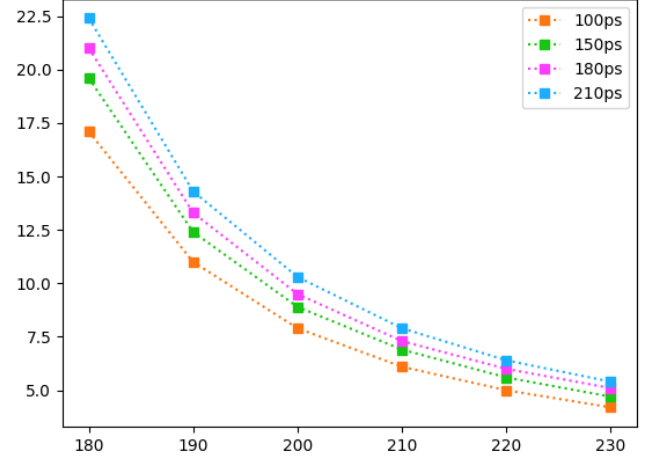


(e)  $\tau_1 = 80\%$ ,  $\tau_2 = 20\%$

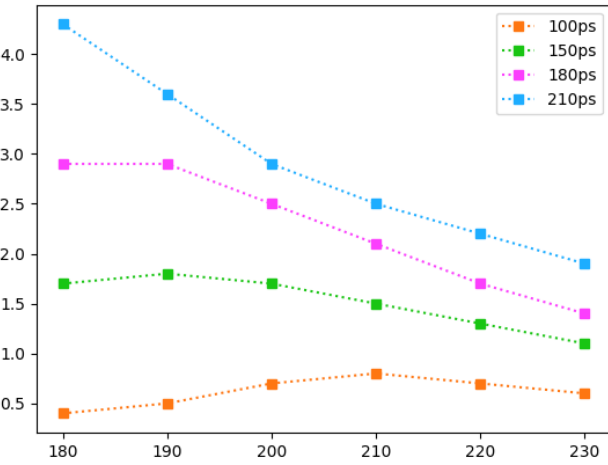
Figure 9:  $\tau_1$ , rows = 20-80, 50-50, 80-20, columns = diff, std dev



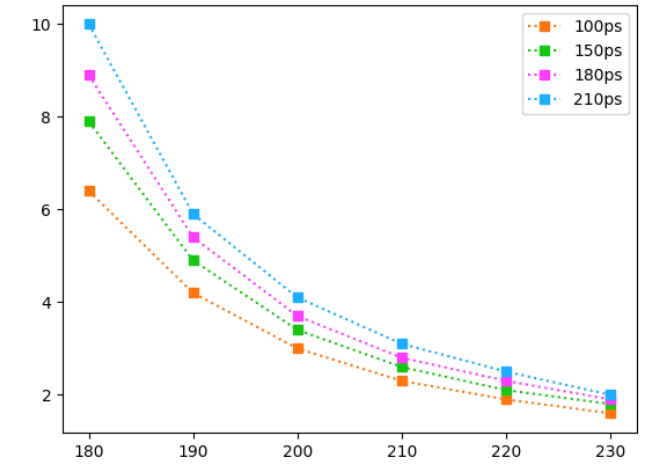
(a)



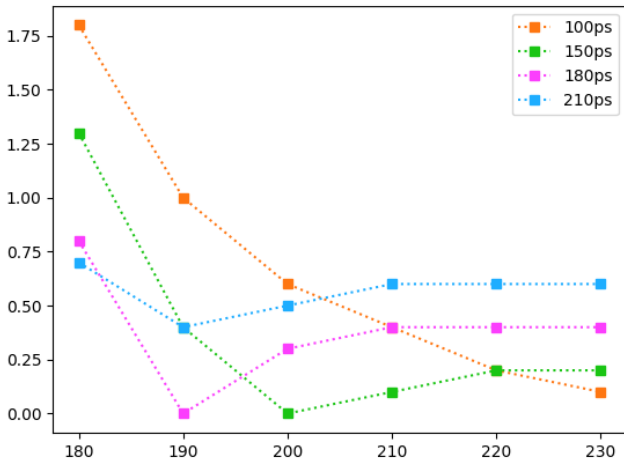
(b)



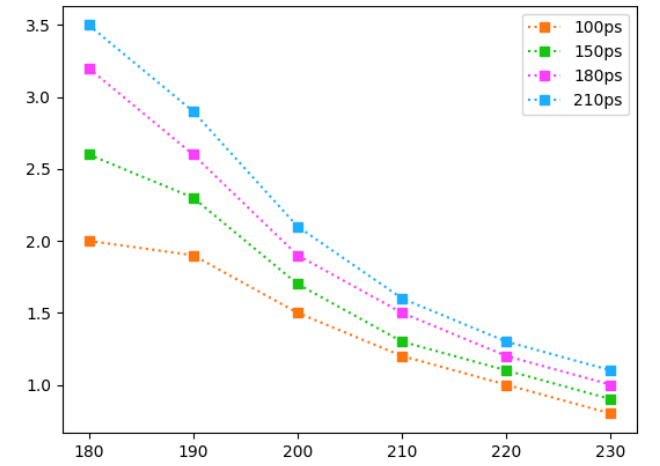
(c)



(d)

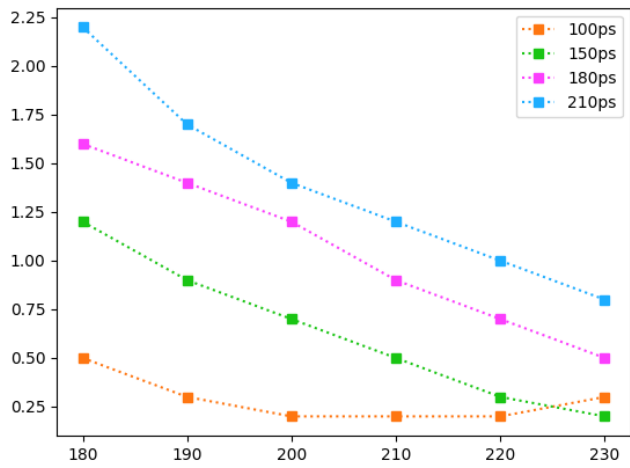


(e)

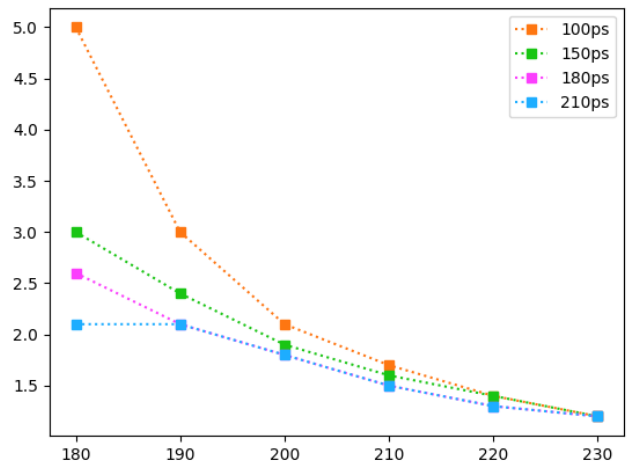


(f)

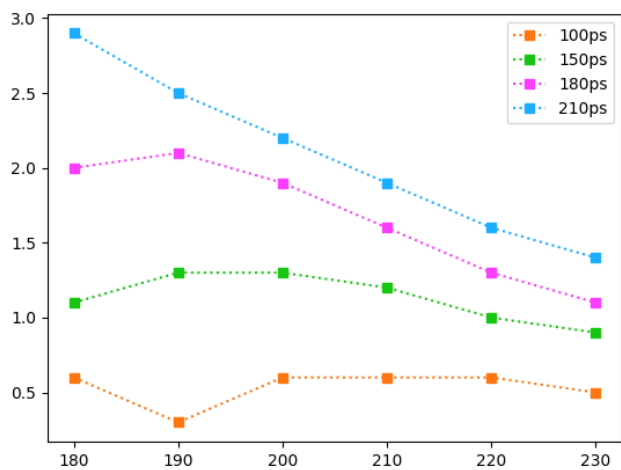
Figure 10:  $\tau_2$ , rows = 20-80, 50-50, 80-20, columns = diff, std dev



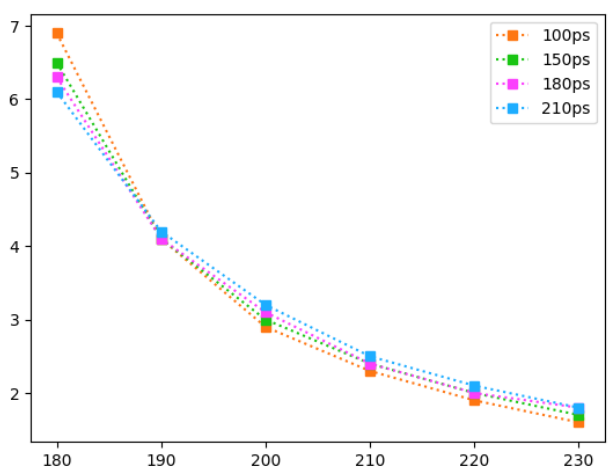
(a)



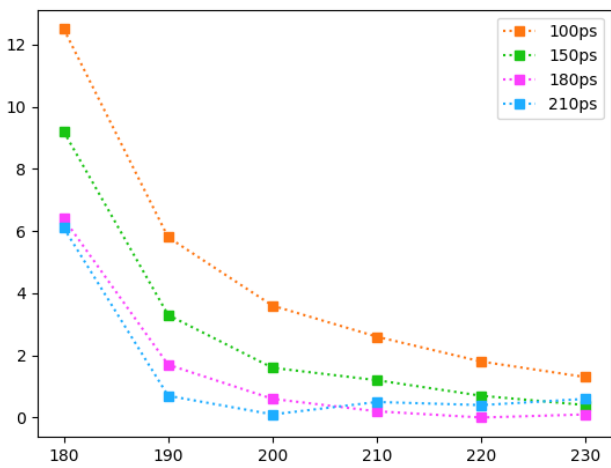
(b)



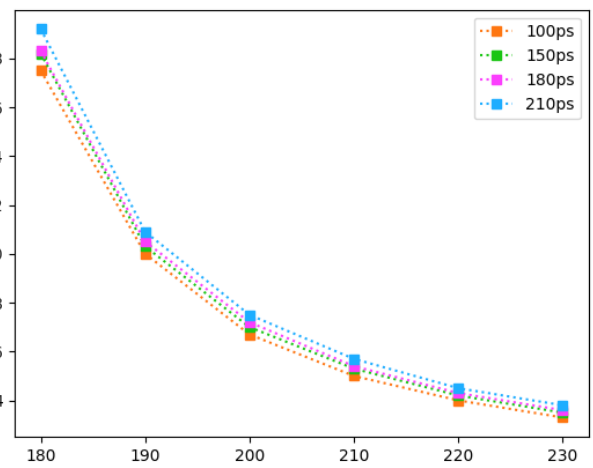
(c)



(d)



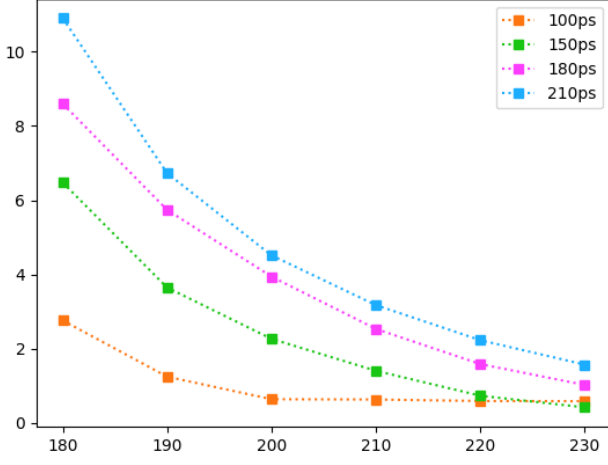
(e)



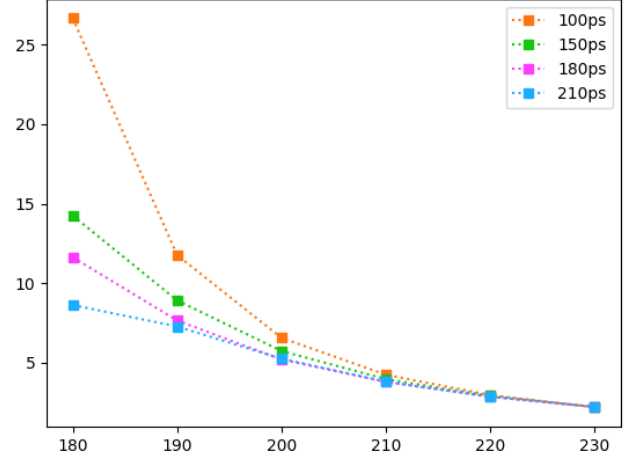
(f)



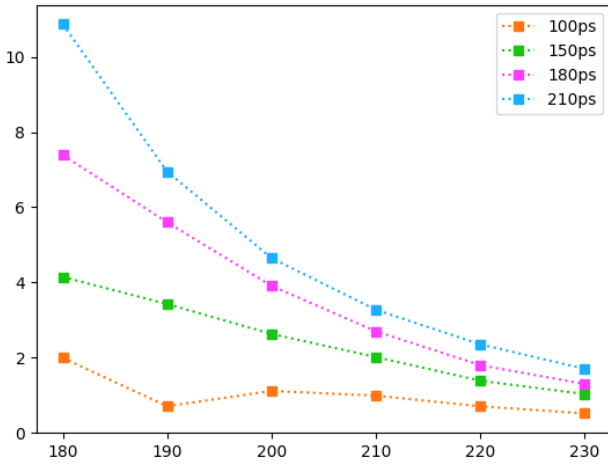
Figure 11: intensities, rows = 20-80, 50-50, 80-20, columns = diff, std dev



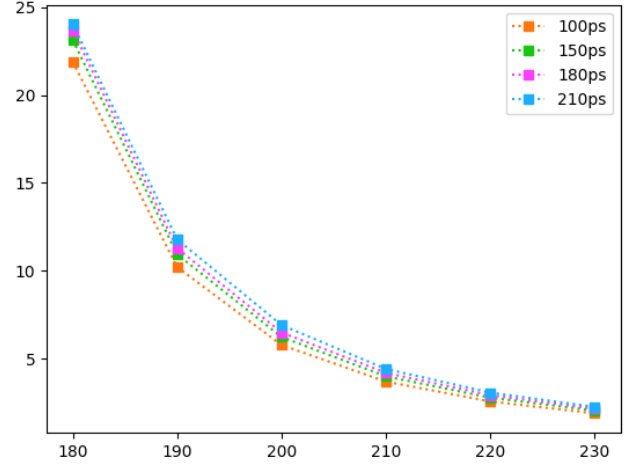
(a)



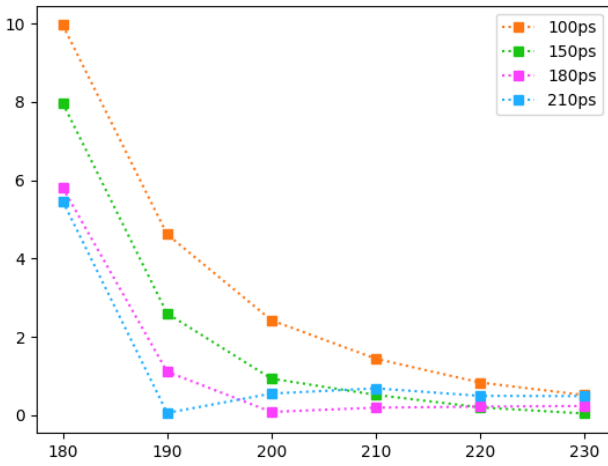
(b)



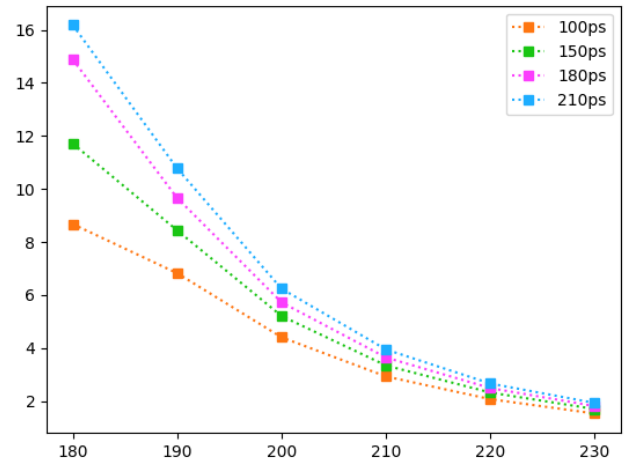
(c)



(d)



(e)



(f)

## 0.4 Number of counts

Another variable worth examining is the total number of counts in a given spectrum. When generating spectra using PALSSIM, this is given as the area of the spectrum without the background and a separate background value. Both were kept constant in all previous runs, with the area set to  $5.65 \times 10^6$  and the background value set to 8.5. Expectations are that a higher number of counts would make spectrum analysis easier. To test this hypothesis, spectra were generated for  $\tau_1 = 150\text{ps}$ ,  $\tau_2 = 180\text{-}230\text{ps}$  and a 210ps FWHM single gaussian resolution function, tripling the background area to  $1.7035 \times 10^7$ .

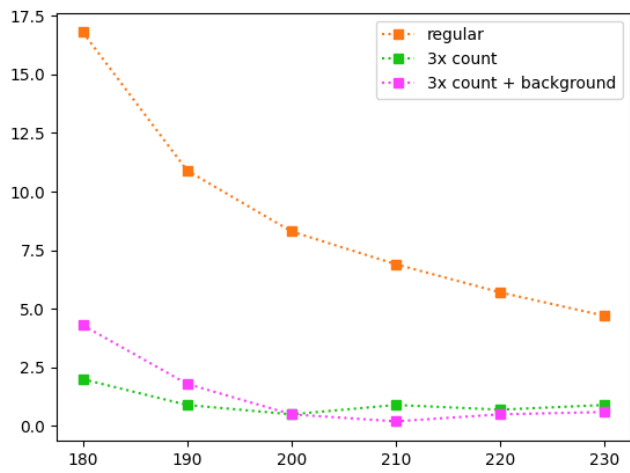
If we just increase the total area of the spectra and maintain the same background value as before, however, we inadvertently end up increasing the signal to noise ratio. While this would be desirable, in a practical setting an increase in total number of counts most likely corresponds to an increase in background noise. As such, spectra were also generated with both the tripled area and a proportional 3x increase in the background value (set to 25.5).

The generated spectra were then analyzed with PALSFIT, and the results are summarized in Figures 12-14, with a "regular" run for comparison. The regular run is simply the one represented in Figure 8, which used the original, unaltered area and background value.

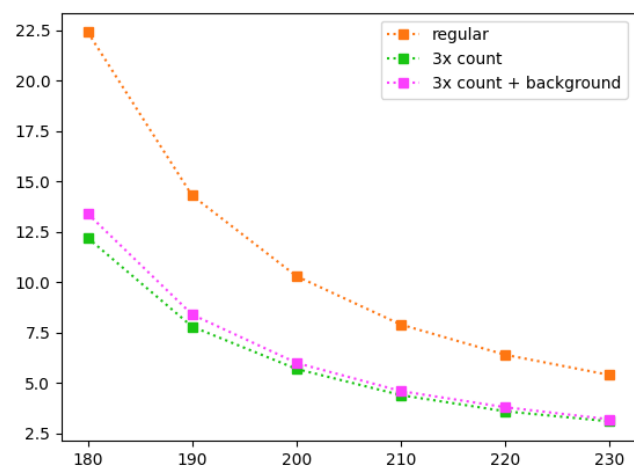
Looking first at the standard deviations (subfigures (b), (d) and (f), on the right side of the page), we see that, as expected, the data from the regular run is in most cases significantly less precise than the other two runs. Additionally, the data with the proportionally scaled background value seems to be slightly less precise than its unscaled counterpart, though in many cases just barely so, if at all. Exceptions tend to occur, however, for the shortest lifetime separation, when  $\tau_2$  is set to 180ps.

Looking at the left side of the figures ((a), (c) and (e)), we see that for the most part the first trend seem to hold true, with the regular run being significantly less accurate than the other two. The exception to this seems to be when the relative  $\tau_1$ - $\tau_2$  intensity is 80%-20%, in which we often have a reversal of expectations, at times with some significant crossover of the relevant trendlines. While, at a glance, the normal and scaled background triple area runs are still for the most part relatively close, with the proportional background looking less accurate overall, this isn't as universally true as it is when looking at the standard deviation. Specifically in Figures 12a, 13a and 14a, when  $\tau_1 = 20\%$  and  $\tau_2 = 80\%$ , PALSFIT seems to have performed worse with the non-scaled background, at least for significant portions of the graphs.

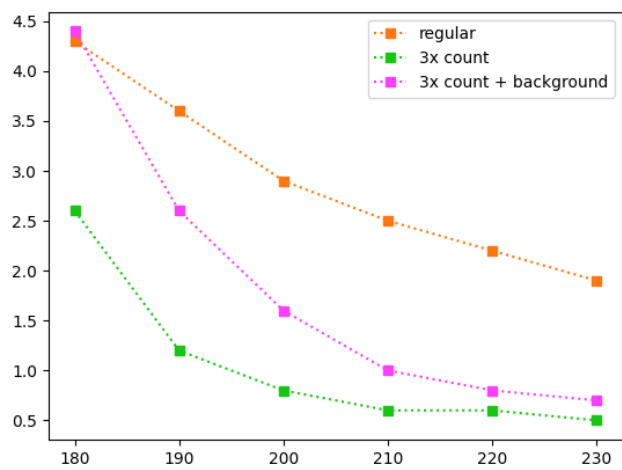
Figure 12:  $\tau_1$ , rows = 20-80, 50-50, 80-20, columns = diff, std dev



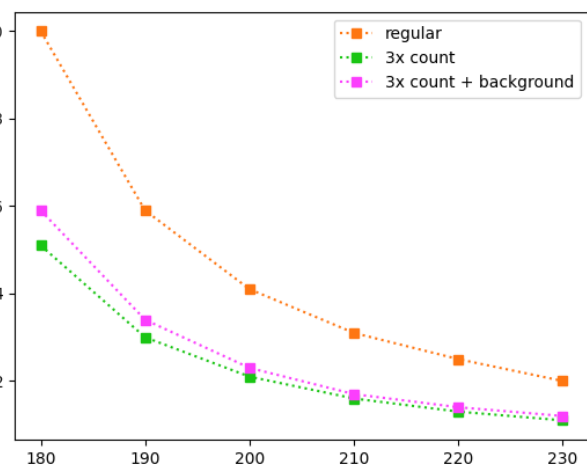
(a)



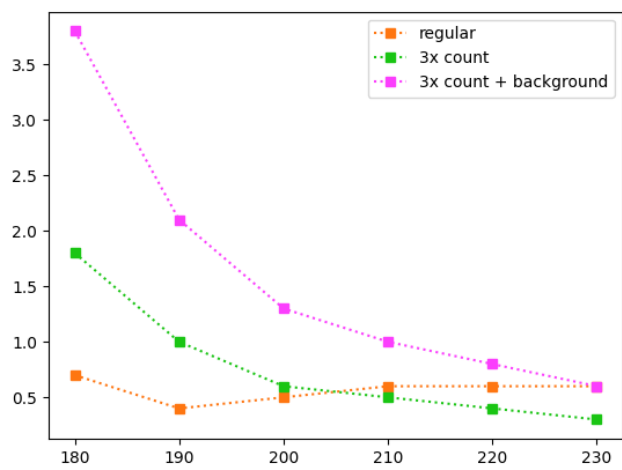
(b)



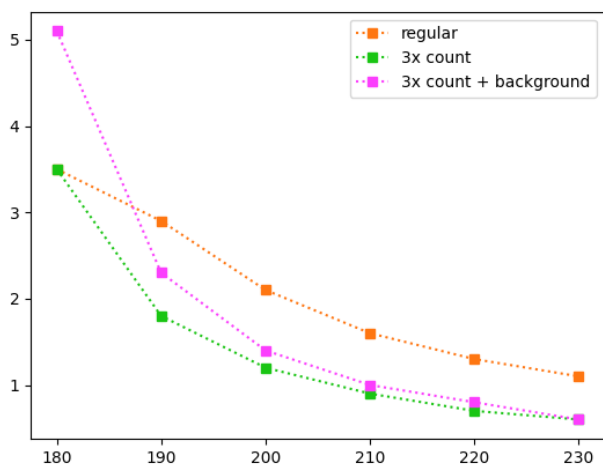
(c)



(d)

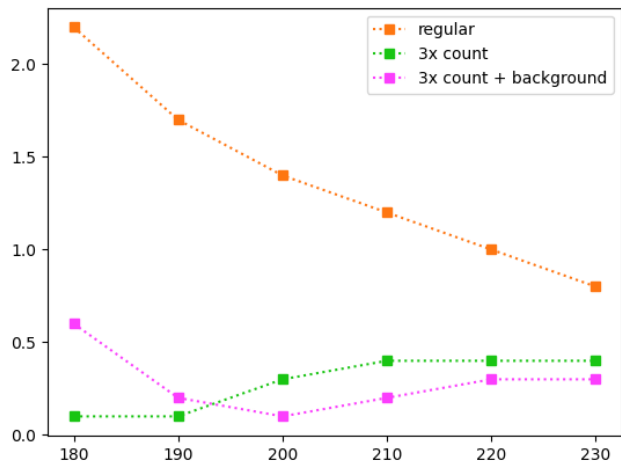


(e)

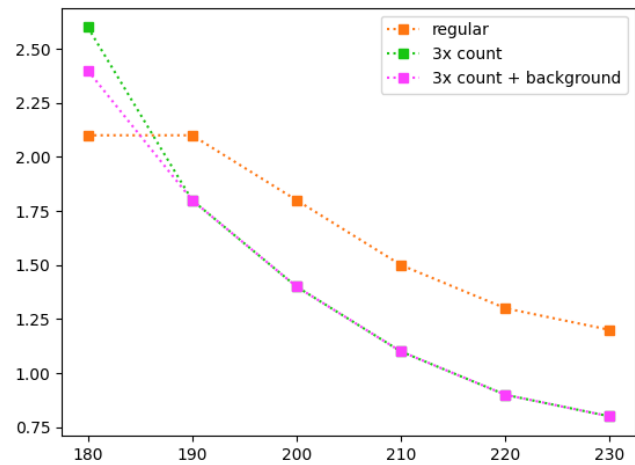


(f)

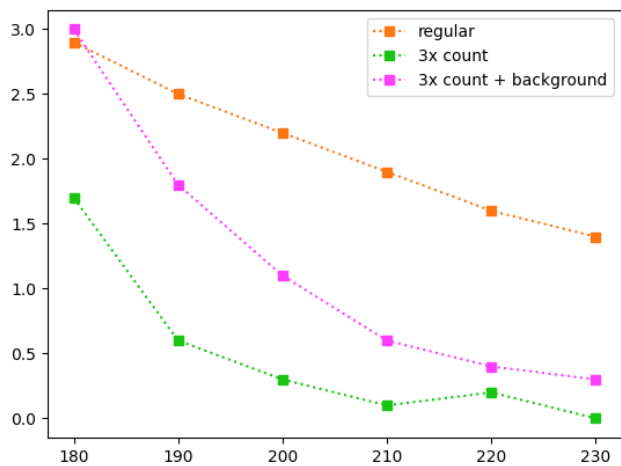
Figure 13:  $\tau_2$ , rows = 20-80, 50-50, 80-20, columns = diff, std dev



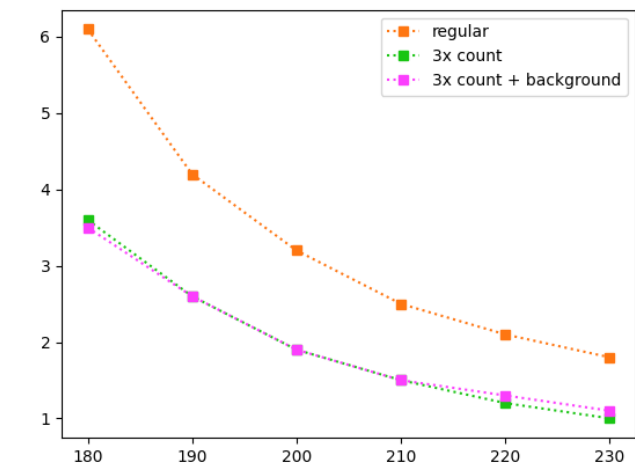
(a)



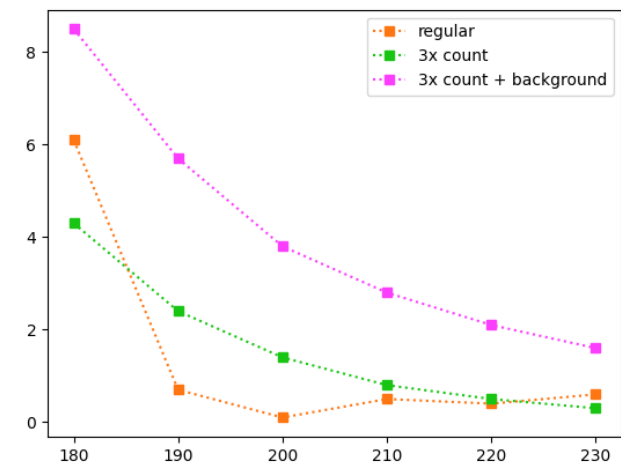
(b)



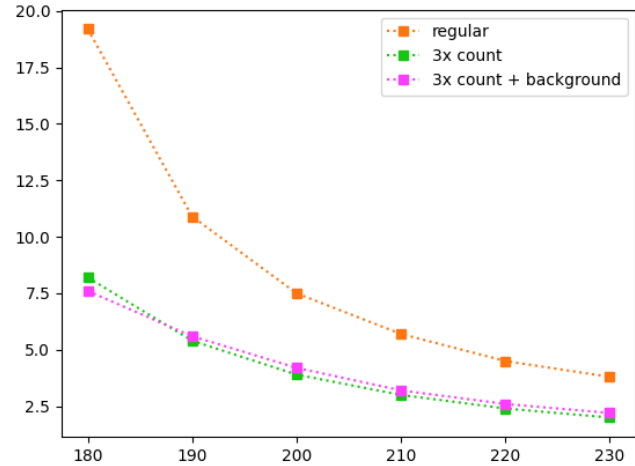
(c)



(d)

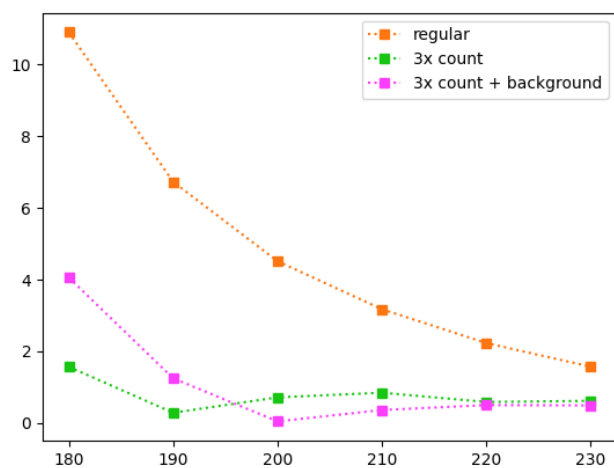


(e)

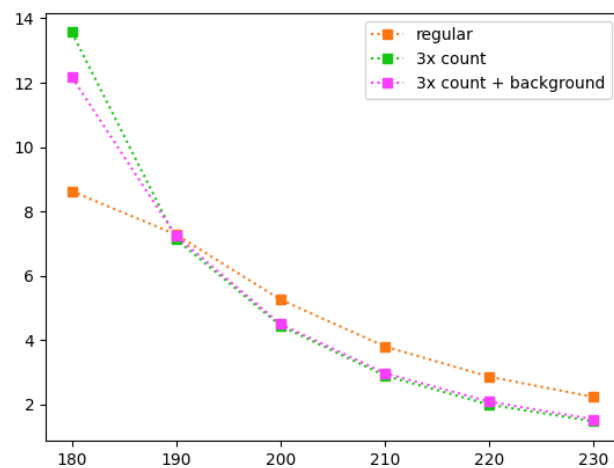


(f)

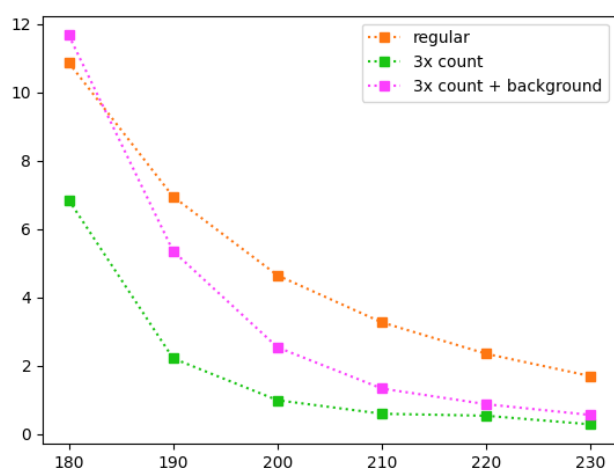
Figure 14: intensities, rows = 20-80, 50-50, 80-20, columns = diff, std dev



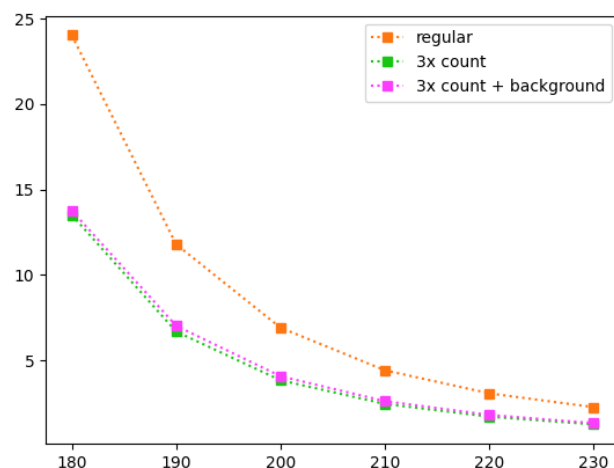
(a)



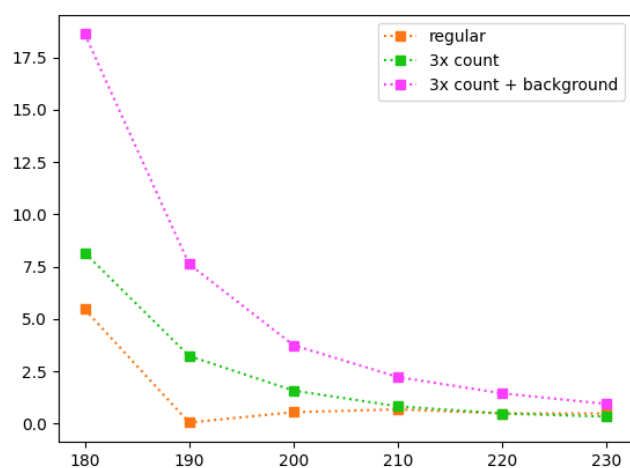
(b)



(c)

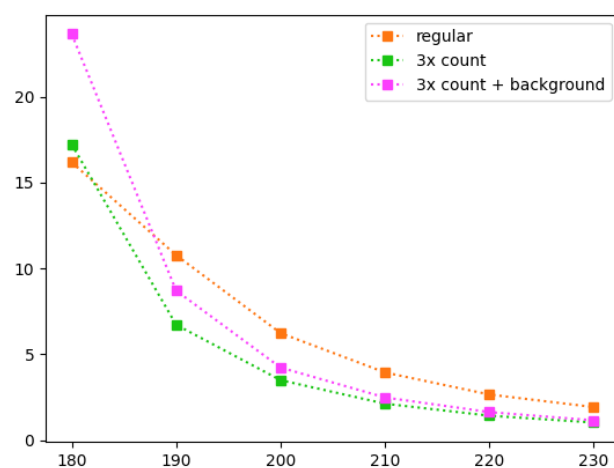


(d)



(e)

20



(f)

## 0.5 Realistic data

Analysis of real world data often differs from how we've conducted our analysis so far. One key factor is that usually the number of lifetimes present in a real life spectrum isn't known beforehand. As PALSFIT requires the user to input the number of fitted lifetimes, this might lead to situations in which the user predicts the wrong number of lifetimes, thus it might be useful to explore how the software behaves in such a situation.

To do so, three materials (?) were modeled, with three lifetime components. All three have two shorter components that differed between them, and a long lifetime component set to 2.6ns with a 0.15% intensity that was kept equal. The relative intensities of the two shorter lifetime components were adjusted to the following values:

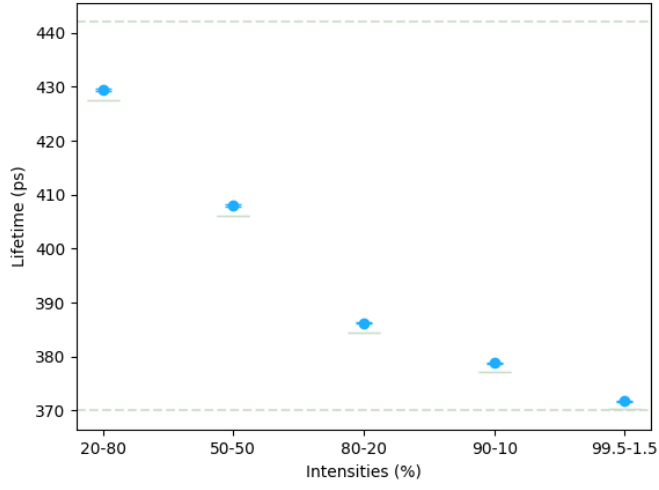
| relative |          | absolute  |          |
|----------|----------|-----------|----------|
| $\tau_1$ | $\tau_2$ | $\tau_1$  | $\tau_2$ |
| 99.5%    | 0.5%     | 99.3508%* | 0.4993%* |
| 90%      | 10%      | 89.865%   | 9.985%   |
| 80%      | 20%      | 89.88%    | 19.97%   |
| 50%      | 50%      | 49.925%   | 49.925%  |
| 20%      | 80%      | 19.97%    | 9.985%   |

(a) intensities

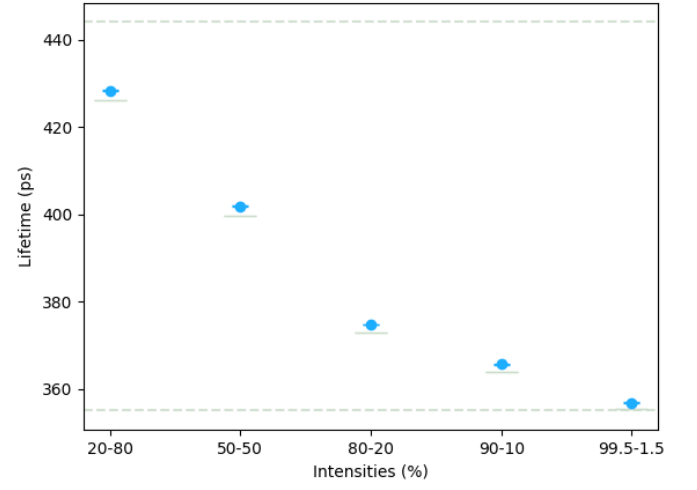
| $\tau_1$ (ps) | $\tau_2$ (ps) | $\tau_3$ (ns) |
|---------------|---------------|---------------|
| 370           | 442           | 2.6           |
| 355           | 444           | 2.6           |
| 348           | 440           | 2.6           |

(b) lifetimes

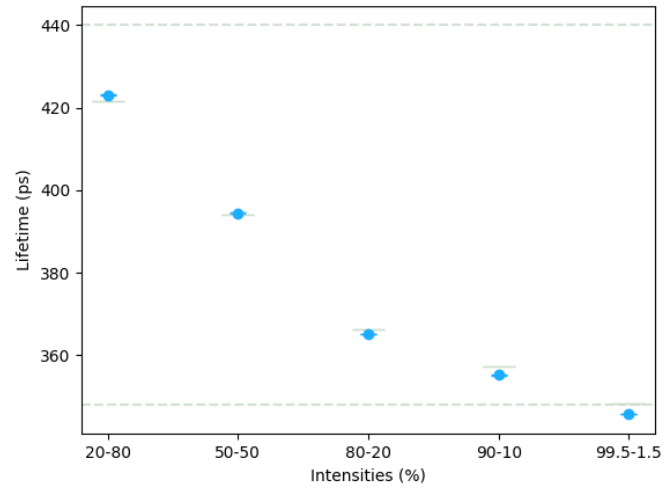
When a single lifetime is fitted, the resulting lifetime seems to tend towards a weighted average of the two shorter lifetimes, as can be seen in the following figures:



(a)  $\tau_1 = 370$ ,  $\tau_2 = 442$

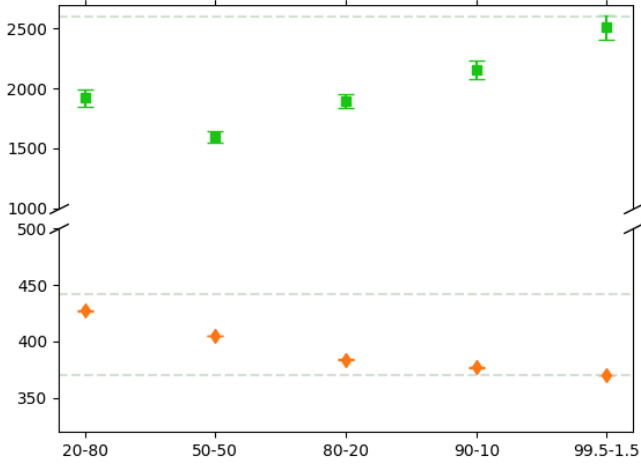


(b)  $\tau_1 = 355$ ,  $\tau_2 = 444$

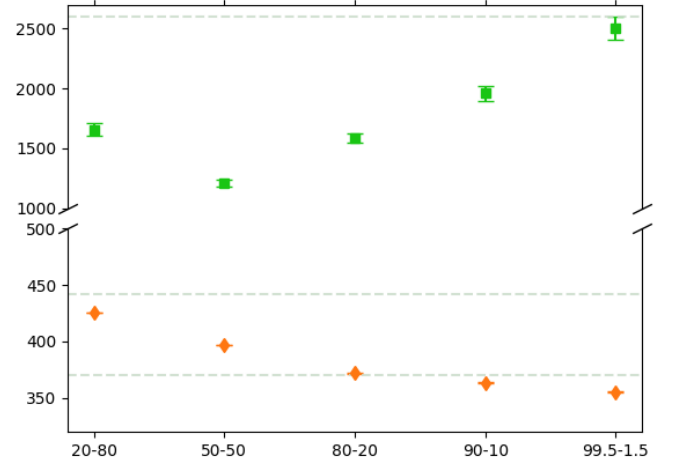


(c)  $\tau_1 = 370$ ,  $\tau_2 = 440$

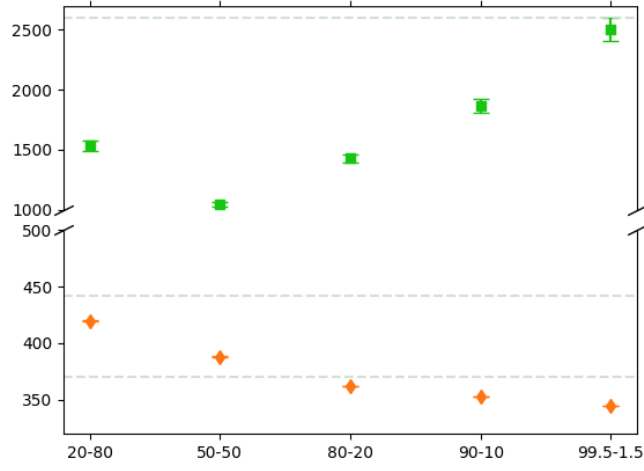
Figure 15: single lifetime fit



(a)  $\tau_1 = 370, \tau_2 = 442$



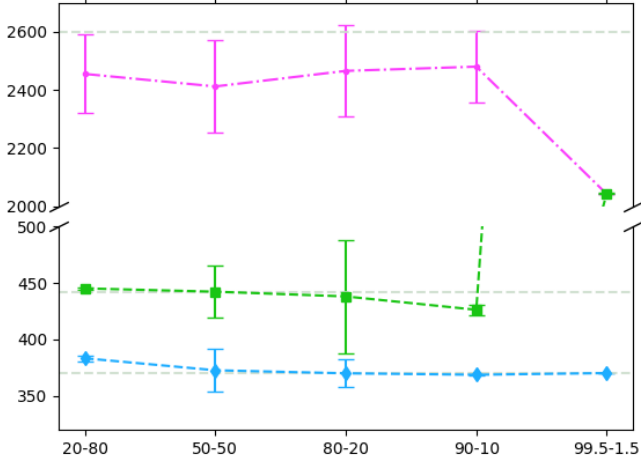
(b)  $\tau_1 = 355, \tau_2 = 444$



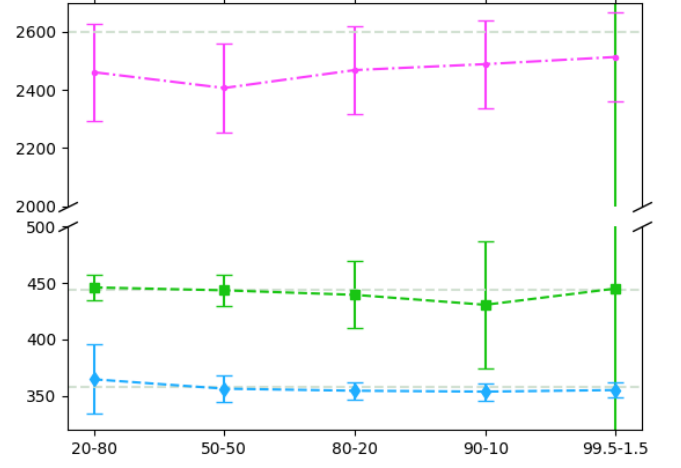
(c)  $\tau_1 = 370, \tau_2 = 440$

Figure 16: two lifetime fit

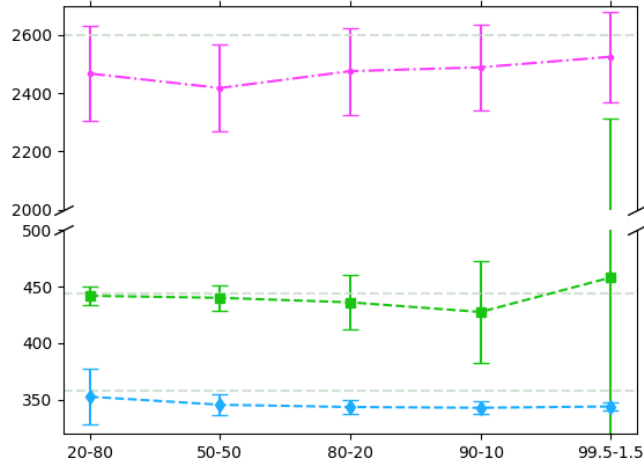




(a)  $\tau_1 = 370, \tau_2 = 442$



(b)  $\tau_1 = 355, \tau_2 = 444$



(c)  $\tau_1 = 370, \tau_2 = 440$

Figure 17: three lifetime fit