Dissertation

Francesco Tamburi

October 2023

Introduction

Results

To begin with, spectra with two positron lifetimes, τ_1 and τ_2 were simulated with PALSSIM, using an instrument resolution function (IRF) made up of three gaussians (see Table 1). With τ_1 kept fixed at 180ps, spectra were generated for τ_2 = 220-280ps at 10ps intervals using PALSSIM. For each value of τ_2, three spectra were generated, with intensities of τ_1 and τ_2 (respectively) set to 20%-80%, 50%-50%, 80%-20%. The goal was to evaluate how good PALSFIT is at extracting the lifetimes τ_1, τ_2 and their respective intensities, for each simulated spectrum.

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

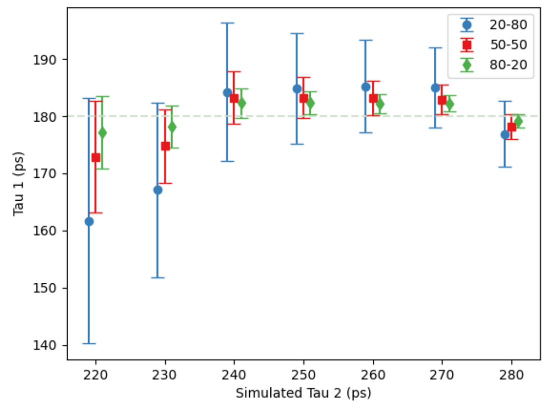
Table 1: Instrument resolution function

Starting with τ_1, we can see in Figure 1a that for some values of τ_2 the software performed well at fitting the shorter lifetime, while it struggled for others. Zooming in specifically to the τ_2 = 250-270ps range, we can see in Figure 1b that, for these values in particular, it struggled to determine τ_1 in the 50-50 and 80-20 case.

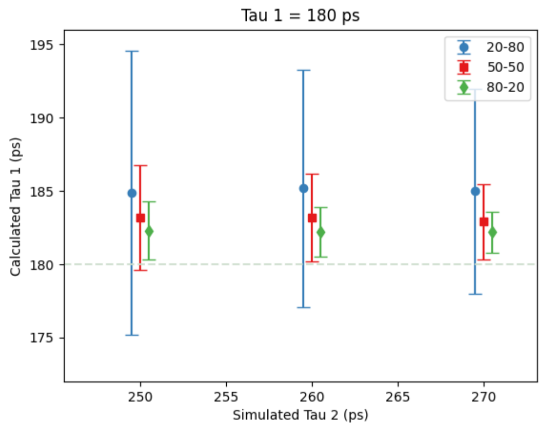
To evaluate how the software performs in detecting τ_2, we do the same as before, but instead of plotting τ_1 on the y axis, we can plot the difference between the result and the original simulated values of τ_2 to essentially tells us how far off PALSFIT is from the simulated value. Doing so we can see in Figure 1c that, aside from the 20-80 spectrum for τ_2 = 220, here PALSFIT performs reasonably well.

Observing the error bars in the figures lets us know how confident our fit for the two lifetimes is. From here we can notice two important factors that affect the precision of the fitting. By looking at Figure 1a and Figure 1c, we can deduce that the relative intensities of the two lifetimes is one of them. In Figure 1a, where we are analysing τ_1, the error bars are the smallest for the 80-20 data, where the shorter lifetime is more intense, and in Figure 3, the opposite is the case and we have the 20-80 data being the most precise.

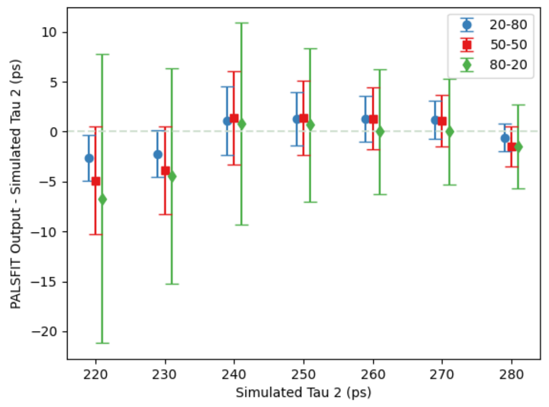
Figure 1



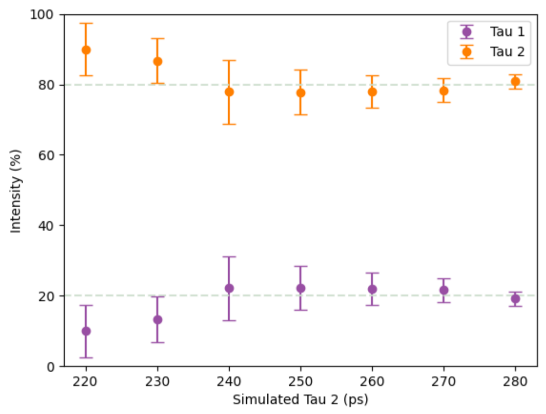
(a) fixed $\tau_1 = 180ps$



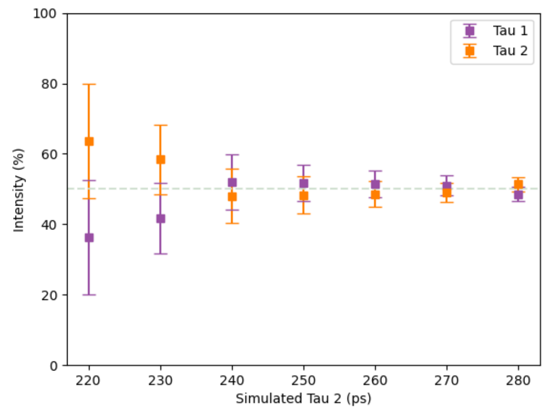
(b) close-up $\tau_1 = 180ps$



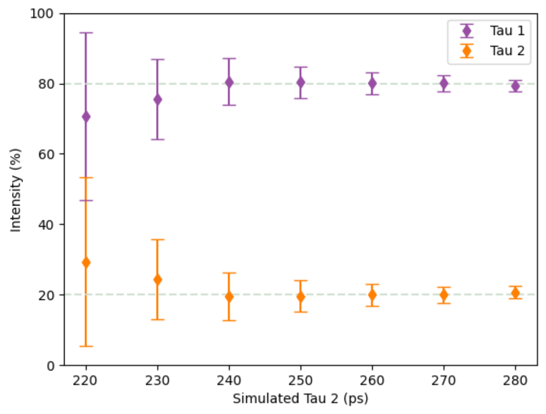
(c) τ_2 difference



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



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

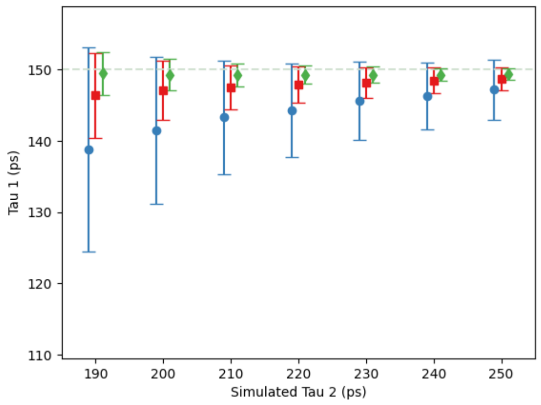


(f) $\tau_1 = 80\%, \tau_1 = 20\%$

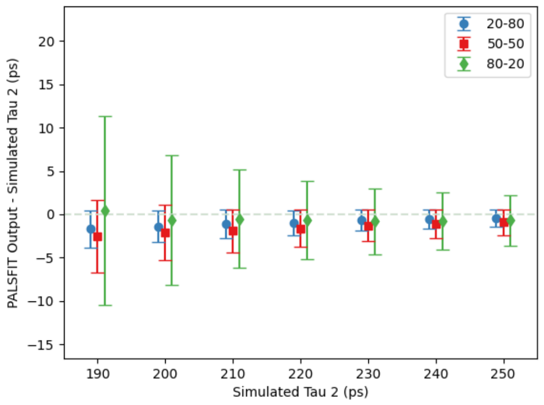
Looking at all the figures, we can also notice that the error bars progressively shrink as the time interval between the lifetimes, which is the second factor, increases. Both these facts make intuitive sense.

In Figure 1d - 1f, we can observe that when the relative intensity of τ_2 was greater or equal to τ_1, then PALSFIT was able to calculate the appropriate lifetime intensities. However, when the intensity of τ_2 was at 80%, as we can see in Figure 1d, for the first two simulated values of τ_2, the software was unable to output the correct intensities. Looking at our error bars, we can see the same relationship between the size of the error bars and the time separation of the two lifetimes.

On the whole then, PALSFIT seemed to perform reasonably well, but not perfectly. As the first lifetime, τ_1, was kept constant at 180ps, the next step would be to change τ_1 and see how that might alter our results. We can bring down τ_1 to 150 and perform the same analysis. To keep things consistent, we'll modify our τ_2 range to maintain the spacing between the lifetimes consistent between runs. As such, we'll set τ_2 = 190-250ps. In Figure **??** we can see that all the fitted values fall within error of their original, simulated, values.

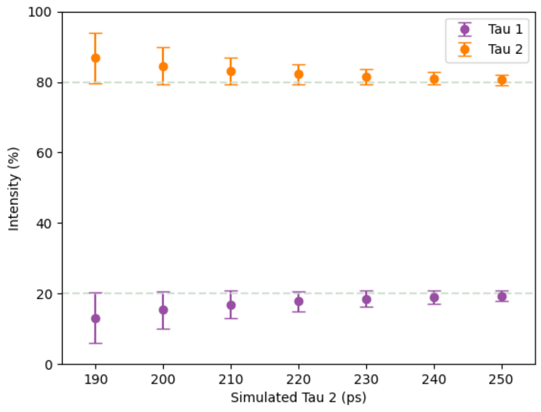


(a) fixed $\tau_1 = 150ps$

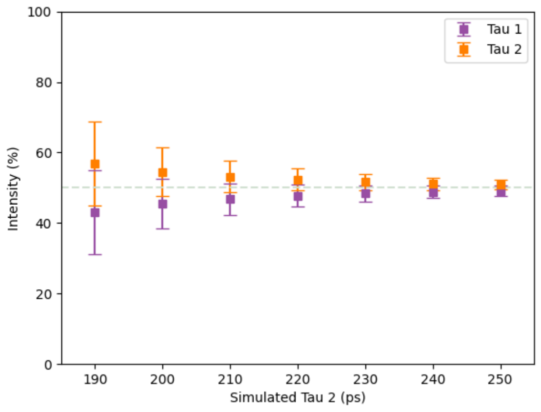


(b) τ_2 difference

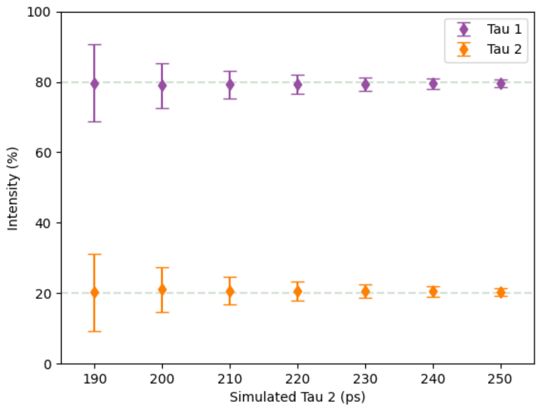
Figure 2



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



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



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

Figure 3

0.1 Instrument resolution function

Using Python, we can plot the three gaussian functions that compose the instrument resolution function. The equation for a Gaussian function with parameters a, b and c (corresponding to the height of the peak, the position of the peak and the width of the graph) is given by the equation:

$$g(x) = a \exp\left(-\frac{(x-b)^2}{2c^2}\right)$$

While a and b map easily to the intensity and shift columns in Table 1 , c is expressed as a standard deviation, and not the full-half-width-maximum

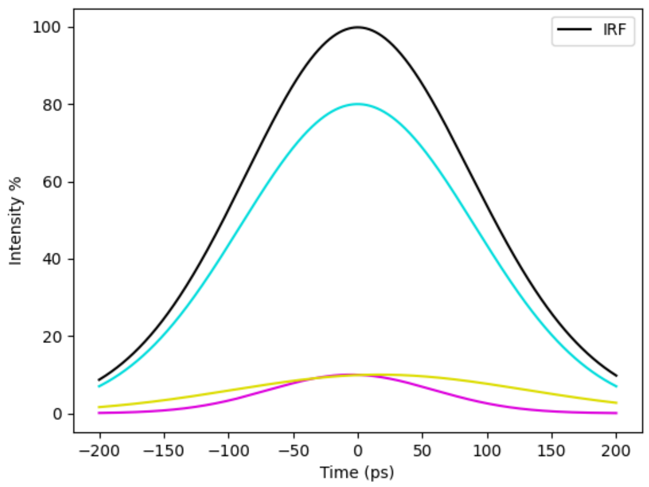


Figure 4: instrument resolution function

given in the table. To convert from one to the other, we can use the following expression:

$$FWHM = 2\sqrt{2\ln(2)} \approx 2.355c$$

The resultant resolution function is, then, just the sum of the three gaussians $g_s(x)$, where

$$g_s(x) = \sum_{i=1}^{3} a_i \exp\left(-\frac{(x-b)^2}{2c^2}\right)$$

Plotted out, this resolution function looks quite similar to a gaussian function itself, and thus it seems sensible to investigate the feasability of approximating a more complex 3 gaussian resolution function, with a single gaussian.

The intensity of the single gaussian approximation is the easiest parameter to determine, as it's simply 100%. We could try to determine b and c analytically by solving the resulting $g_s(x)$ analytically, but, as we already have already generated a list of values for $g_s(x)$ when plotting the function, it's much easier to just extract the needed values numerically using Python.

Finding b is as easy as finding the largest element of the list, and its corresponding x value, which works out to be approximately 0ps (0.0610ps, to be precise). As PALSSIM requires the FWHM rather than the standard deviation c, we just need to find the difference between the two values of x for which $g_s(x)$ is closest to half of the IRF. This gives us a value for the single-gaussian FWHM of 210ps.

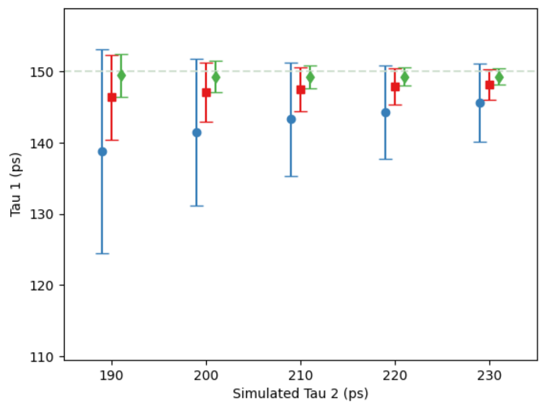
Thus these are the values we'll be using for our single-gaussian resolution function, summarised in Table 2.

FWHM (ps)	Shift (ps)	Intensity (%)
210	0	100

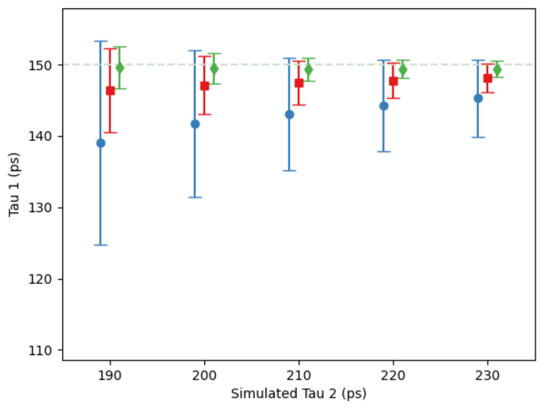
Table 2: Single gaussian Instrument resolution function

Plugging in these values into PALSSIM gives us very similar results to the three gaussian IRF, as we can see in Figures 5-9. This means that we can comfortably simplify the three gaussian resolution function down to a single gaussian.

Figure 5: fixed $\tau_1 = 150ps$

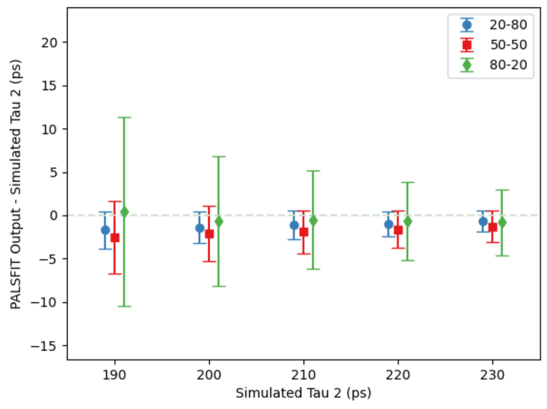


(a) regular irf

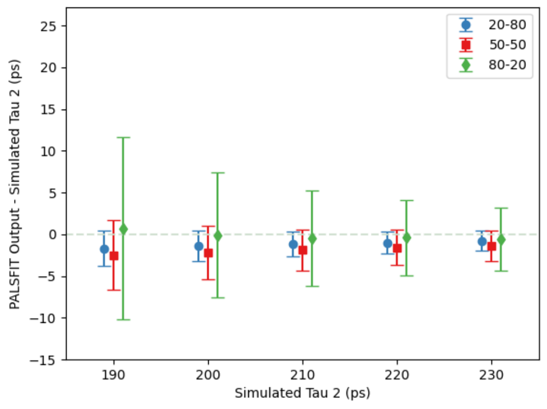


(b) single gaussian

Figure 6: τ_2 difference

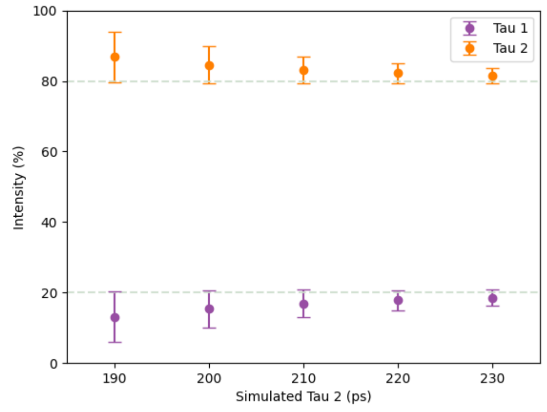


(a) regular irf

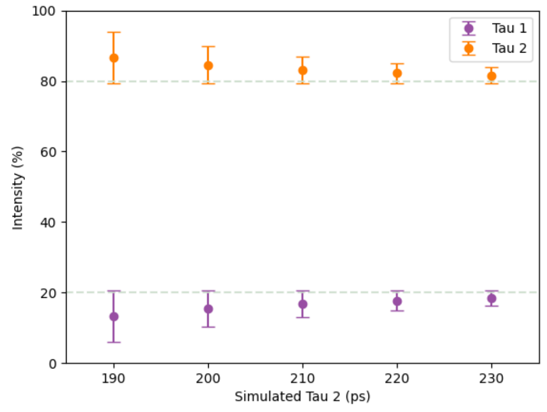


(b) single gaussian

Figure 7: $\tau_1 = 20\%$, $\tau_2 = 80\%$

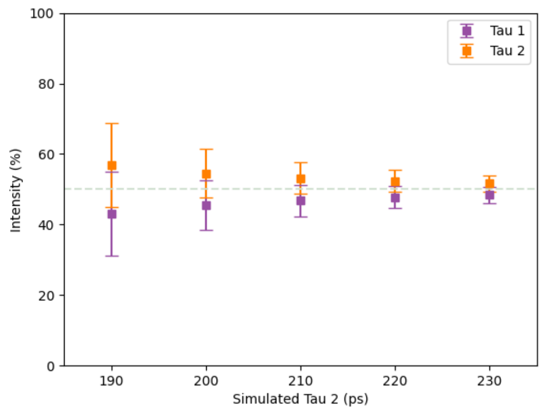


(a) regular irf

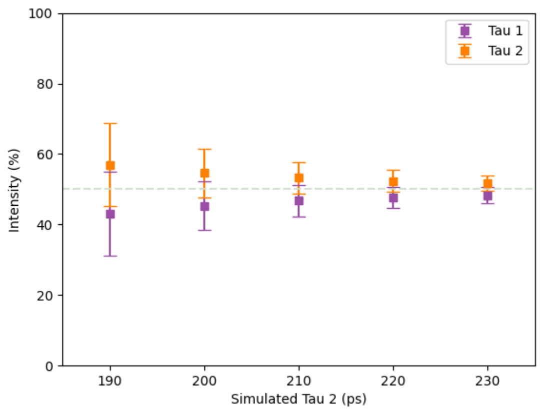


(b) single gaussian

Figure 8: $\tau_1 = 50\%$, $\tau_2 = 50\%$

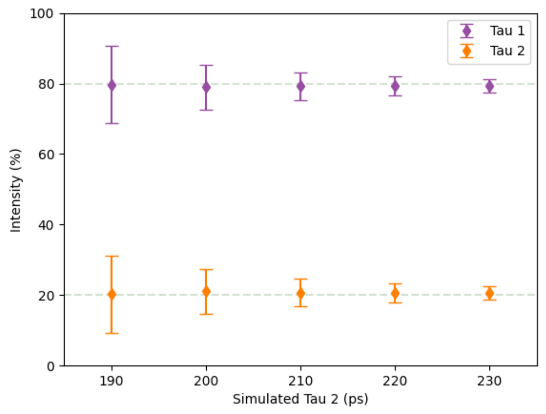


(a) regular irf

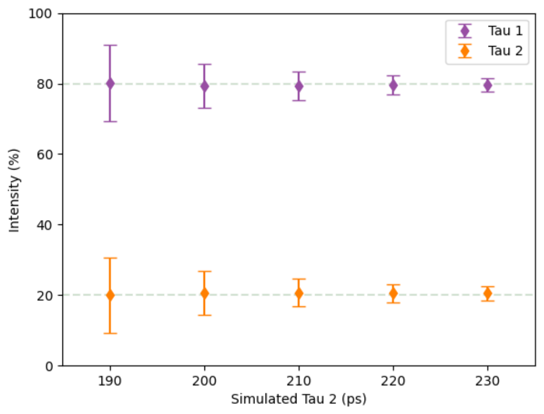


(b) single gaussian

Figure 9: $\tau_1 = 80\%$, $\tau_2 = 20\%$



(a) regular irf

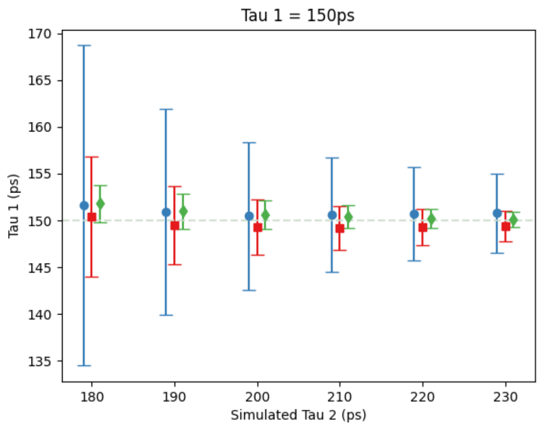


(b) single gaussian

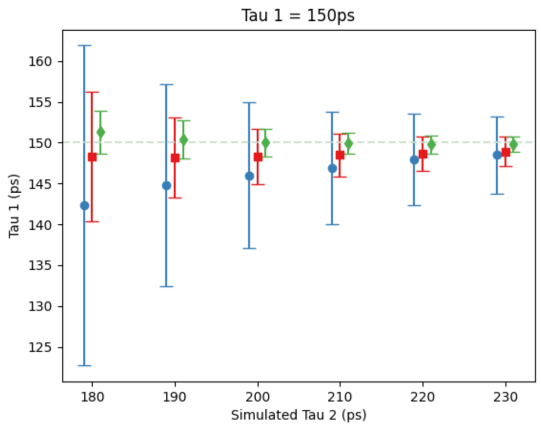
The next step would be to examine how the width of the instrument resolution function (IRF) affects the results. The values for τ_1 and τ_2 can be set to be kept the same between runs, with τ_1 fixed to 150ps, as we had the best result with this value, and τ_2 ranging from 180-230ps. The full-width half maximum of our single gaussian IRF can be set to the following values: 100ps, 150ps, 180ps, 210ps, with 100% intensity and

The results are as follows, looking at specifically at τ_1 and τ_2:

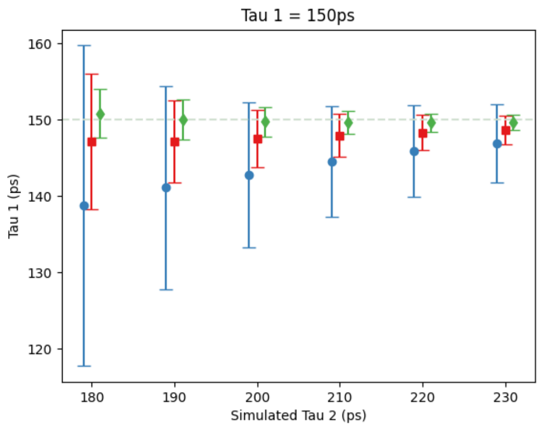
Figure 10: τ_1 comparison



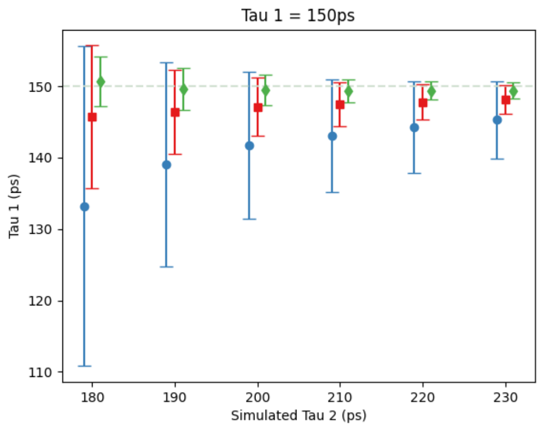
(a) 100ps IRF



(b) 150ps IRF

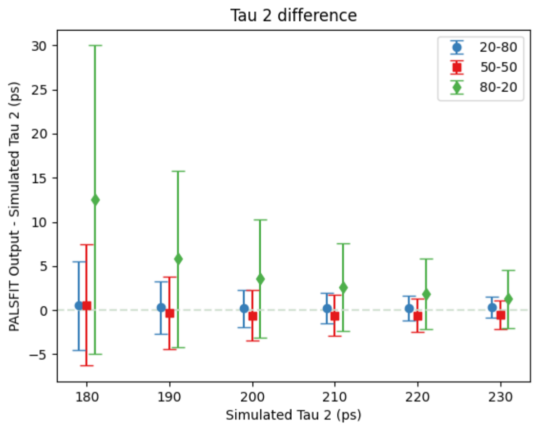


(c) 180ps IRF

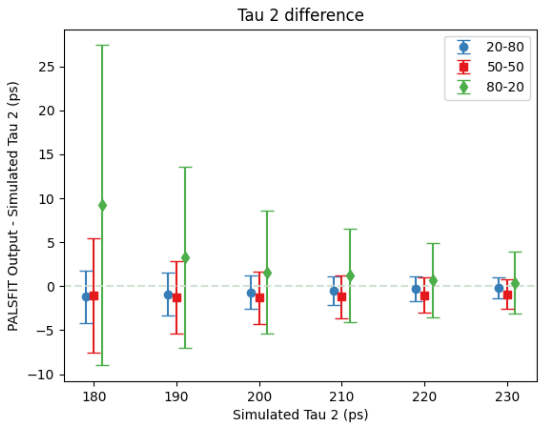


(d) 210ps IRF

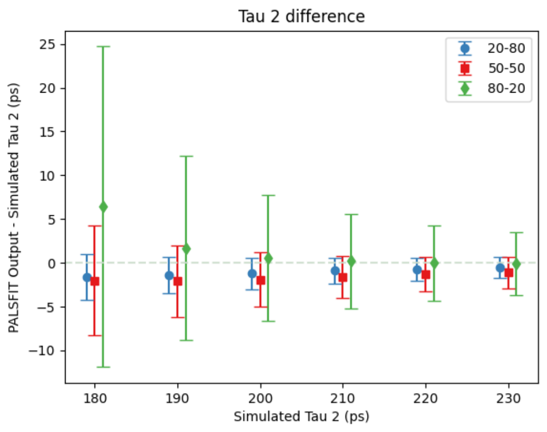
Figure 11: τ_2 (difference) comparison



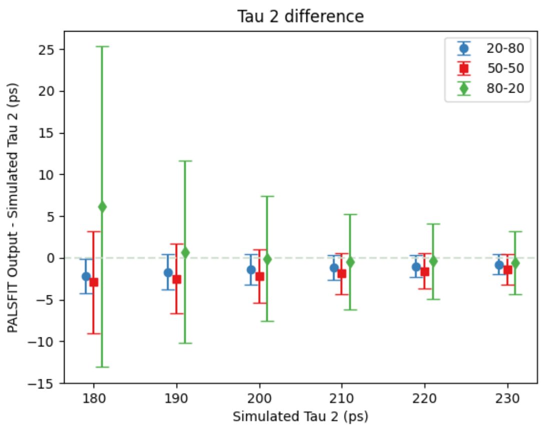
(a) 100ps IRF



(b) 150ps IRF



(c) 180ps IRF



(d) 210ps IRF

What we'd expect is that as the resolution function narrows, we'd have a generally improving fit. While this is true looking at some of the data – looking at the accuracy of the 20-80 fits for tau_1 (see Figure 10), for example – this is not universally true. In fact, bizarrely, for some datasets we have the opposite happening, with the fit getting worse as IRF width decreases. A good example of this is if we look at Figure 10 again, concentrating, this time, on the accuracy of the 80-20 fits.