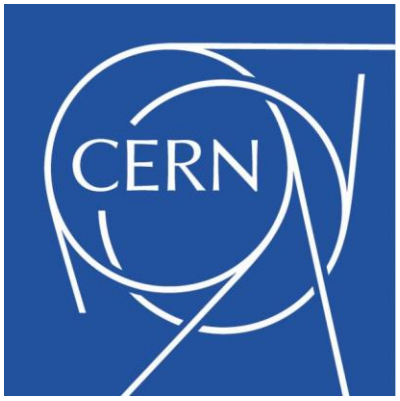

CERN Summer Student Report

Upgrading TDPAC Data Acquisition and Evaluation software Interlude

JOONATAN LAULAINEN
UNIVERSITY OF CAMBRIDGE

Supervised by Joao Guilherme Correia, Karl Johnston, and Juliana Schell

Contents

ABSTRACT: In Time-Differential Perturbed Angular Spectroscopy, local hyperfine fields are probed through the observation of perturbed decay time spectra of an intermediate excited state in a γ-ray cascade. At ISOLDE Materials and Life Science groups, the data acquisition and evaluation is done using the program Interlude. However, Interlude had numerous outstanding bugs and issues, which have now been removed. During the construction of the observable function with the raw data, a systematic bias was identified in the fitting algorithm, and significantly reduced through the addition of a more complicated fitting function. Steps were taken to generalising the program for a variety of end users and input file types, with aim to share the program among users and publish a paper on it in early 2018.

1. Introduction

Through nuclear magnetic dipole interactions and electric quadrupole interactions, Time-Differential Perturbed Angular Spectroscopy (TDPAC) is an excellent tool for probing the local hyperfine fields and nanoscopic atomic phenomenology in materials, compounds, and soft matter [1]. This is due to its sensitivity to the charge density symmetry and polarisation of the probe nucleus and its surrounding lattice, allowing extremely localised measurements. In TDPAC, an excited probe nucleus that decays via a γ-ray cascade (a successive decay of at least two γ-rays) is implanted or dissolved into a target material, and the subsequent decay of the excited state is observed by a scintillator detector. TDPAC has been used in the study the nature of vacancy and interstitial sites, for example in the multiferroic material $YMnO_3$ [2]. In TDPAC, the measured spectra are the product of a nuclear decay and a perturbation function $G(t)$, typically approximated as a sum of cosines as

$$A_{22}G(t) = A_{22}^{\theta}\sum_{k=0} a_k cos(\omega_k t), \qquad \text{where } A_{22}^{90^\circ} = -\tfrac{1}{2}A_{22}^{180^\circ}.$$

The experimentally obtained perturbation function $R(t) \sim A_{22}G(t)$ must be separated from the decay exponential to measure the local electric quadrupole and magnetic dipole moments.

In a TDPAC measurement, typically four or six detectors are set up at 90° from each other, such as shown below for four detectors in Figure 1. Each of these detectors has set energy windows under which to record a start and stop signal, corresponding to the energy of the first and second γ-rays of the decay respectively. The decay time spectrum is typically recorded on a computer reading a multichannel analyser connected to an analogue electronic detector setup or by reading a digital signal processing setup. Once an experiment is run for a sufficiently long time, Interlude communicates with the acquisition setup to extract the data to the analysing computer.

Subsequently, Interlude is used to analyse the data and extract the perturbation function $R(t)$. First, each spectrum is fitted to obtain the channel delay time t_0, which represents the time when the first γ-ray γ_1 is emitted. The fitted function is (with 7 parameters):

$$N_1 \operatorname{erf}\left[\frac{1}{\sqrt{2}}\left(\lambda\sigma - \frac{t-t_0}{\sigma}\right)\right] * e^{-\lambda(t-t_0)} + bkgd(1+Bx) + N_2 e^{\frac{-(t-t_0)^2}{2\sigma^2}}$$

where typically B and N_2 are zero.

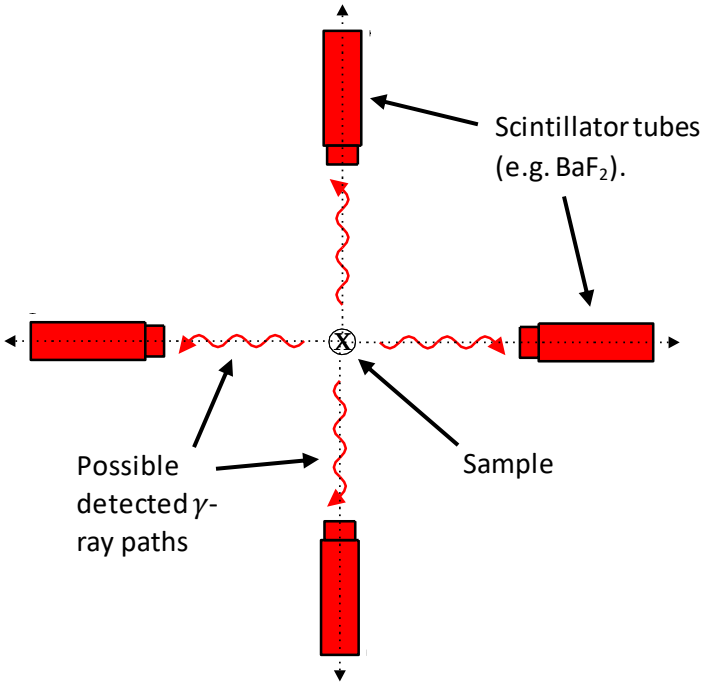

Figure 1: An example of a 4-detector TDPAC setup. Each detector has two set energy windows, one for the start of a measurement, and one for the stop. If one detector detects a start, it then waits for another to detect a stop signal soon after to record the measurement. The 5th and 6th detectors would be in and out of the page. Adapted from my supervisor Joao Guilherme Correia.

This fitting is done via a Levenberg-Marquardt algorithm, and so only reaches a local minimum (as typical for fitting algorithms), and so is susceptible to the initial values for the parameters. This fitting is necessary due to the disparities in t_0 between spectra, which occurs due to differences in triggering of analogue electronics and length of cabling between different detectors as the t_0 values are separated only by few tens of nanoseconds. The t_0 values are fitted to subsequently shift the spectra to the same t_0, to get the perturbation functions in phase.

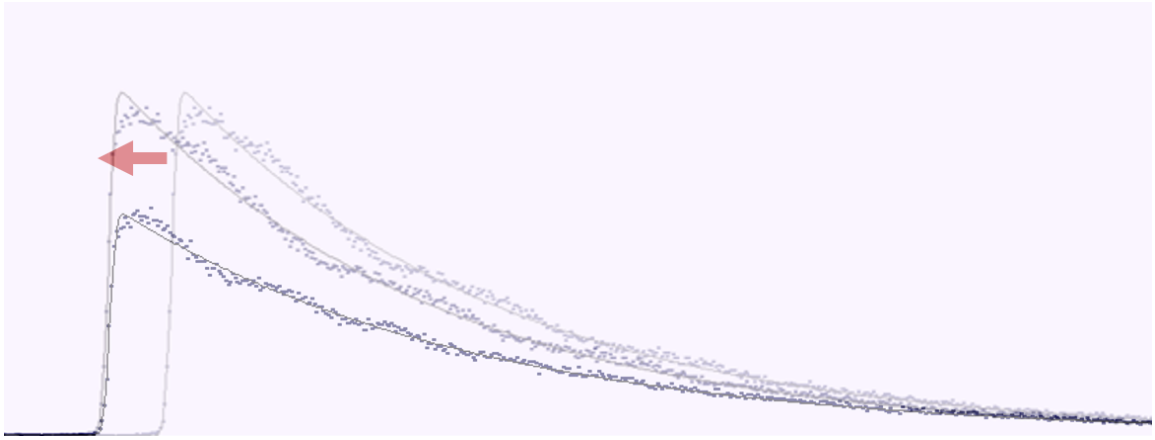

Figure 2: Two overlaid spectra at different t_0 values, and one of the them subsequently shifted to bring the perturbation functions in phase. Figure from Interlude.

Subsequently, an average intensity spectrum is generated for 180° and 90° by take the product of the spectra at that angle and then the respective root. For a six detector setup, these formulas are:

$$N(180°,t) = \sqrt[6]{\prod_{k=1}^{6} N_k(t-t_0)}, \qquad N(90°,t) = \sqrt[24]{\prod_{k=1}^{24} N_k(t-t_0)}$$

After which the perturbation function $R(t)$ is calculated by

$$R(t) = 2\frac{N(180°,t) - N(90°,t)}{N(180°,t) + 2N(90°,t)}.$$

After which Interlude has done its job, as the $R(t)$ spectrum is analysed in another program. This perturbation function holds within it information about the symmetry and magnitude of the electric quadrupole moment and magnetic dipole moment as well as the number of sites the radioactive atom sits in.

2. Interlude

Interlude, originally developed in the late 1990s at the University of Leipzig as Prelude , has since changed considerably to become the Data Acquisition and Evaluation program it is now. It has two primary functionalities, to handle the data acquisition from several TDPAC machines, as well as extracting the perturbation function $R(t)$. In addition, it has various secondary functionalities. It allows for the comparison of $R(t)$ spectra, modifications of the data, and analyses of the Fourier transforms of the perturbation functions. Interlude also allows the exclusion of specific combinations of detectors if there are electronic faults in the machines, which can cause, for example, reflection of the data. In short, it is a robust tool for acquiring TDPAC data, identifying problems in that data, and for extracting the perturbation function $R(t)$.

3. Motivation

On my arrival, Interlude had numerous bugs. This varied from the problem crashing on attempting to automatically MCA transfer, to the "Fit All Groups" being completely unreliable meaning that each spectrum had to be fitted individually. Both issues caused a lot of unnecessary headache and time spent to the users of the program. Hence a lot of time could be saved by fixing these bugs, as well as incorporating functionalities already present (such as Guilherme "curing" spectra procedure, detailed below).

In addition, Guilherme and I had a plan to publish the program as a general PAC data acquisition and evaluation program and write a small paper on it. To do this, several processes in the program, such as file input and output, as well as the method of communicating with the electronics to acquire data, would have to be generalised. These two formed the primary motivation for my project during my 12-week stay.

4. Results and changes

Many changes were made to the program. Some of them are detailed below, some I have forgotten (particularly bug fixes).

4.1. Bug fixes

Fixed an issue with the "Fit All Groups" function in fitting t_0 values.

Fixed the Auto-Repeat data transfer causing the program to crash.

Fixed an issue with several menus not loading up properly.

Fixed an issue with the program wanting to read a parameter file from the data folder each time.

Removed several non-functioning menus, including the print function.

Fixed issues with colour of overlaying R(t) spectra.

Fixed an issue with the carryover of failed σ-fitting.

Fixed an issue with empty spectra causing the fitting window to change.

4.2. Added functionalities

Added a method "cure" spectra. This is caused by an electronics malfunction, resulting in a specific channel having an abnormally high or low value. The data point is replaced by the average of the surrounding data points, and adding a Poisson error.

Determined the file format of BIN file output in the Konstanz machines, and made these files readable.

Made the detector combination and group visible on screen while moving the mouse.

Changed the colour scheme of the background and the data, making the program much easier to read.

Implemented a preliminary method to remove empty groups from the "All" view.

Simplified the order of the menus.

Added another format (.nnr) to save $R(t)$ as, to not require file conversion before reading in Nightmare.

Removed a needless "Loupe" functionality that had no effect.

Significantly streamlined the code to determine initial guesses for the different parameters, as well changing the initial guesses for t_0, N_1, and λ, significantly improving their accuracy.

Added an option to save calculated $R(t)$ in program memory, as well as loading or overlaying it, allowing for easier comparison.

Tested various new fitting procedures (details below).

Generalised file input and output (details below).

There may also be others, but they have been left out for now.

4.3. Improvements in the fitting

Any systematic error in the fitting algorithm was investigated. This was done by generating a set of realistic data using a python script. The data was generated as follows:

- An error function with a random known t_0 value was generated (range 0 to 1) and then multiplied by the amplitude N_1.
- Then, the function is multiplied by a product of an exponential decaying from t_0 and a perturbation function of the form $[1 + A_{22}(a_0 + a_1 \cos\omega_1(t - t_0) + a_2 \cos\omega_2(t - t_0) + a_3 \cos\omega_3(t - t_0))]$ with $\omega_2 = 2\omega_1$ and $\omega_3 = 3\omega_1$ which is a common relation between the frequencies.
- Then a background is added.
- Finally for each point in the spectrum, a random value from a Poisson distribution with the mean as the value of the point is generated. This replaces the value, and acts as a realistic way to generate the statistical noise that occurs in real data.

This data was then put into Interlude, and the t_0 values fitted. Since the exact t_0 values are known from the generation step, the two can then be compared. This showed that with a large perturbation, the t_0 values have a systematic error, which depends on the sign of the perturbation (the A_{22} term above). The error is shown in a graph below in Figure 3:

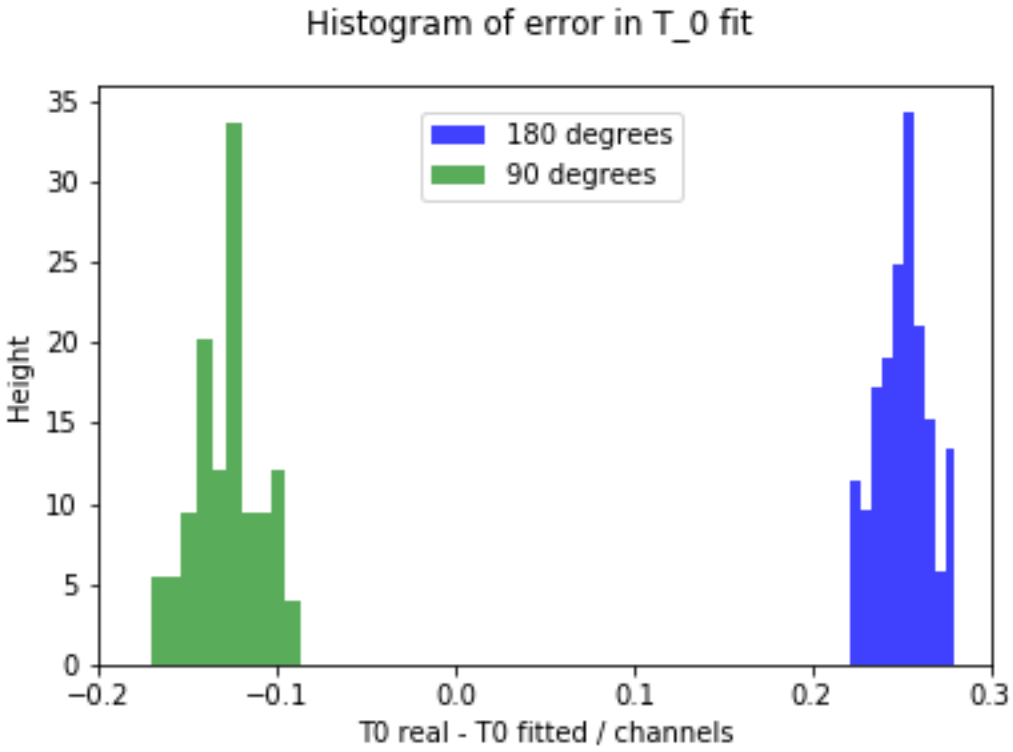

Figure 3: A histogram of the error in t_0 fit. Note the difference between 180 and 90 degree spectra, due to the relation of -0.5 in their respective A_{22} values. The data had N=8000, bkgd=100, and $A_{22}(180)$=0.3.

This error is interesting because it cannot be fixed through the fitting function present in the program, as it is a shift of the minimum of the fitting algorithm. Hence, to combat this, a second fitting procedure, involving a function with cosine terms, was added. In this case, the fitted function was:

$$N_1 \operatorname{erf}\left[\frac{1}{\sqrt{2}}\left(\lambda\sigma - \frac{t-t_0}{\sigma}\right)\right] * e^{-\lambda(t-t_0)}\left(1 + A_{22}\left(a_0 + a_1 \cos\omega_1(t-t_0) + a_2 \cos\omega_2(t-t_0)\right)\right) + bkgd(1+Bx) + N_2 e^{\frac{-(t-t_0)^2}{2\sigma^2}}$$

The data used above was also fitted with this function, to gauge if a reduction in error had been achieved. A roughly two-thirds reduction of the difference between t_0 fits was removed, as shown below in Figure 4. The effect was more pronounced in the 180° case.

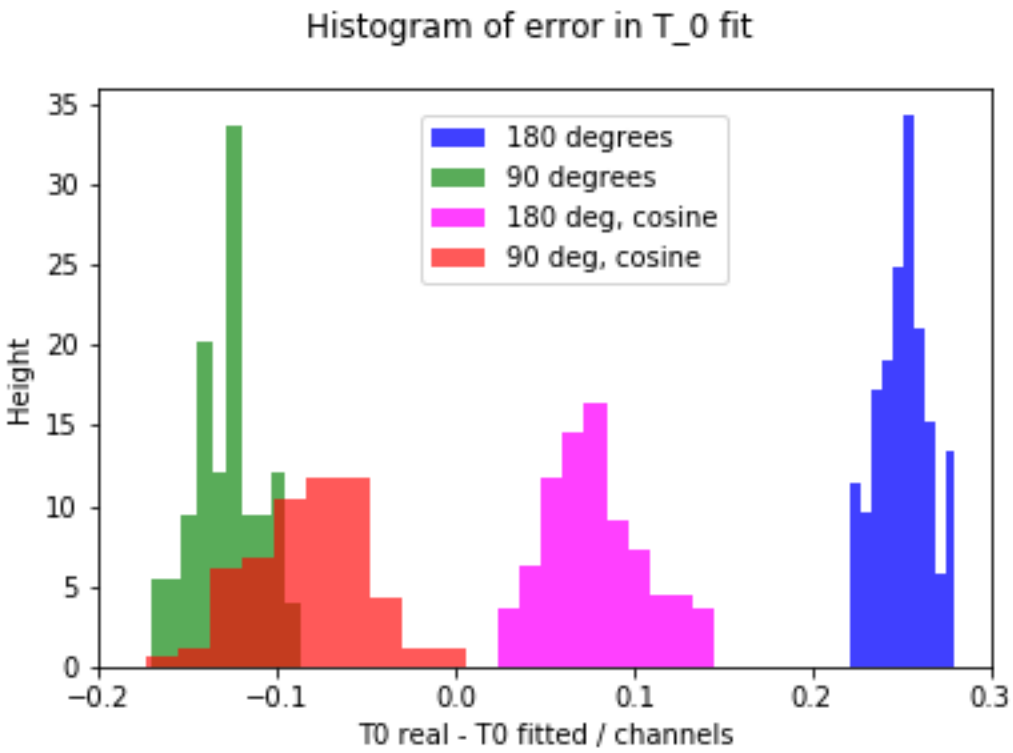

Figure 4: A histogram of the error in t_0 fit, with both the traditional fitting function and the cosine fitting function. Note the reduction in . The data had N=8000, bkgd=100, and $A_{22}(180)$=0.

The cosine fit significantly reduces the systematic error, leading to a more faithful representation in the first channels of the generated perturbation function $R(t)$, as shown below in Figure 5.

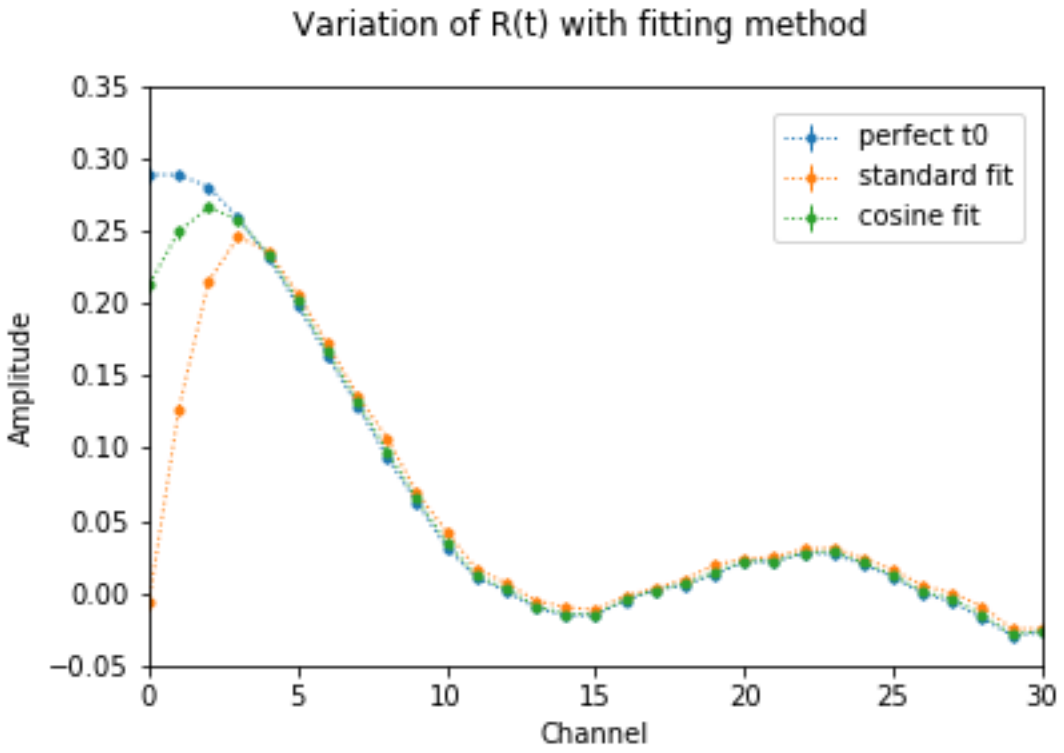

Figure 5:Calculated R(t) for the generated data, depending on whether exact, fitted, or cosine-fitted t_0 values were used. The cosine fit shows a considerable improvement over the standard fit.

Unfortunately, making reasonable first guesses for the added parameters is basically impossible, so the program currently requires the user to input sensible initial guesses. This is particularly important as cosine function have numerous local minima, meaning that it is easy for the fitted function to converge to a poor solution. **I recommend using the cosine function in cases where the perturbation is large.**

4.4. Generalisation of the program

For the publishing of the program, a key component is having a file input/output system that can deal with a large variety of filetypes. The PAC machines at CERN alone produce 4 or 5 different file types (of which at least three need to pass through Interlude), and all are very different. Hence the file formatting system I created allows for the reading of:

- Binary files, with any size of header, any number of bytes per data point, any number of groups with any number of channels (as long as the groups are specified), both big and little byte order, and any header size. This should cover any binary files except the most exotic.
- Text files, with any number of groups with any number of channels (as long as the groups are specified), any header size as long as the header size is confined to a number of lines in the beginning of the program, and the lines can be formatted in almost any way as long as the data value itself is not split up. This should cover most typical file types.

A picture of the method through which a file format is defined is shown below in Figure 6. The file formats defined through this method are stored in Machines.par, which is stored in the program directory, and can be accessed through a number of different menus.

A positive side effect of creating this file system was that the binary files from the Konstanz machines in the lab can now also be read by the program, removing the need to save two sets of files.

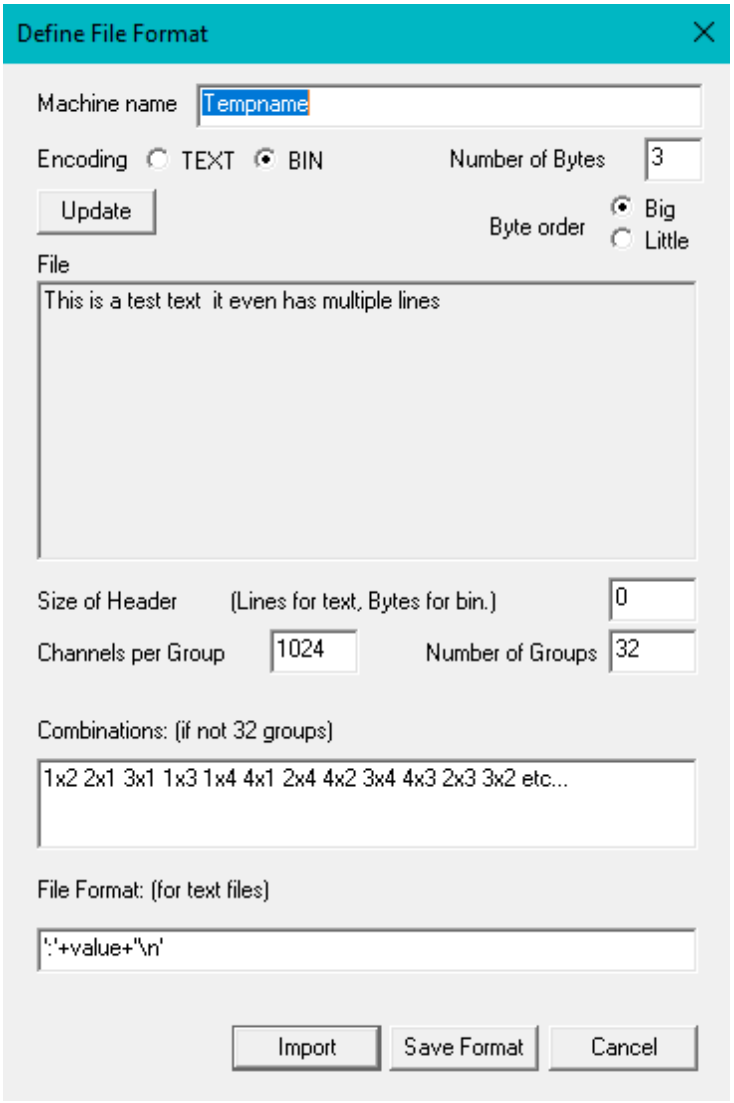

Figure 6: Generalised file system input/output, defining the file format. Formats defined this way can then subsequently be accessed from another menu, which can be done very quickly.

5. Conclusion

Interlude has received a number of much needed bug fixes and improvements, as well as being well on its way to a general PAC Data Acquisition and Evaluation software. A few additional changes still need to be done, which I will complete on my return to University. This will be followed by publishing the program and writing a paper on it. These include adding checks to the file reading system to prevent program crashes, adding a few functionalities to the fitting procedure, and changing the structure of the program to be able to accommodate plugins from other users. Tests of the current fitting procedure demonstrated systematic bias in the procedure, which has been significantly reduced by changing the fitting function.

In addition, I spent considerable time in the lab during the course of the August cadmium beam time, as well as some additional time supervising Emission Channeling experiments and another beam time in September. I took part in the Summer Student lecture course and workshops, and have thoroughly enjoyed my stay here at CERN.

6. Steps necessary before publication

To finalise the program before sharing it, a few tasks still need to be done, and I will do over the course of the next few months. They are (in no particular order):

- Fix bugs and functionalities to the general file reading system
- Create a new native data file type, mainly removing the !P32! from the header
- Create a "reduced view" for data with less than 30 groups, showing only non-empty spectra.
- Improve the cosine fit procedure to take into account 90° and 180° groups.
- Calculate the errors for having measured the A2G2 by a sum.
- Add a fitting procedure when there is an additional decay spectra to the left of t_0.
- Remove references to program version Prelude32 and Prelude 16.
- Change the format of the program to allow a data transfer plugin.
- Change $R(t)$ file type, such that the time per channel is included in the data.
- Make the program save all 32 data groups rather than just 30.

References:

[1] Karl Johnston et al. 2017 J. Phys. G: Nucl. Part. Phys. 44 104001

[2] T. M. Mendonça et al. 2010 Hyperfine Interact. 197:83–88