G-2 Internal
Version 1.0
August 14, 2019

Cornell fast rotation Fourier analysis user guide

J. Fagin[1], A. Chapelain[1], D. Rubin[1], and D. Seleznev[1]
[1]Cornell University

Abstract

The Fourier method for extracting the Radial distribution of the muons about the ring is briefly described, and the results of the analysis is shown for the example 60–hour data set. We show how to download the Fourier analysis code and how to run the example. Then we describe how to switch to any other dataset the user may wish to run the Fourier analysis on including both real and Monte Carlo data. The ability to run parameter scans and pseudo data experiments are also shown to measure systematic and statistical uncertainties with the method. We also provide a comprehensive list of all possible options for adjusting parameters for the Fourier analysis code with detailed explanations for what to choose for each parameter depending on the data set and what the user is trying to accomplish.

Contents

1 Introduction

The experimental measurement of the anomalous magnetic moment of the muon depends on the precession frequency of the muon's spin about its momentum. In order to account for the contribution that the electric field makes to the precession frequency we must determine the momentum distribution of the muons in the muon storage ring. The momentum distribution is calculated from the fast rotation signal, which is the evolution of the intensity of the muon beam as it goes around the ring. Figure 1 shows an example of a simulated fast rotation signal using a toy Monte Carlo.

In actual data, we are given the intensity of the muons as they hit the detector over time and the fast rotation signal can be calculated by fitting the intensity signal with a function, which accounts for the decay of the muons over time and then dividing the intensity by this function. Once we have the fast rotation signal, we can calculate the frequency distribution of the signal by doing a modified Fourier transformation. From the frequency distribution the E-field correction (C_E) can be calculated. The final goal is to recover a C_E with a margin of error less than 20 ppb.

Here we describe the complete Fourier method for extracting the E-field correction and radial distribution of the fast rotation signal. We then show how the user can download and run the method and how the analysis is done on the example 60–hour data set. Then we describe what needs to be done for the user to run the method on any real or simulated data set. A list of all the possible parameters which may be changed is also given and all the options for changing each parameter.

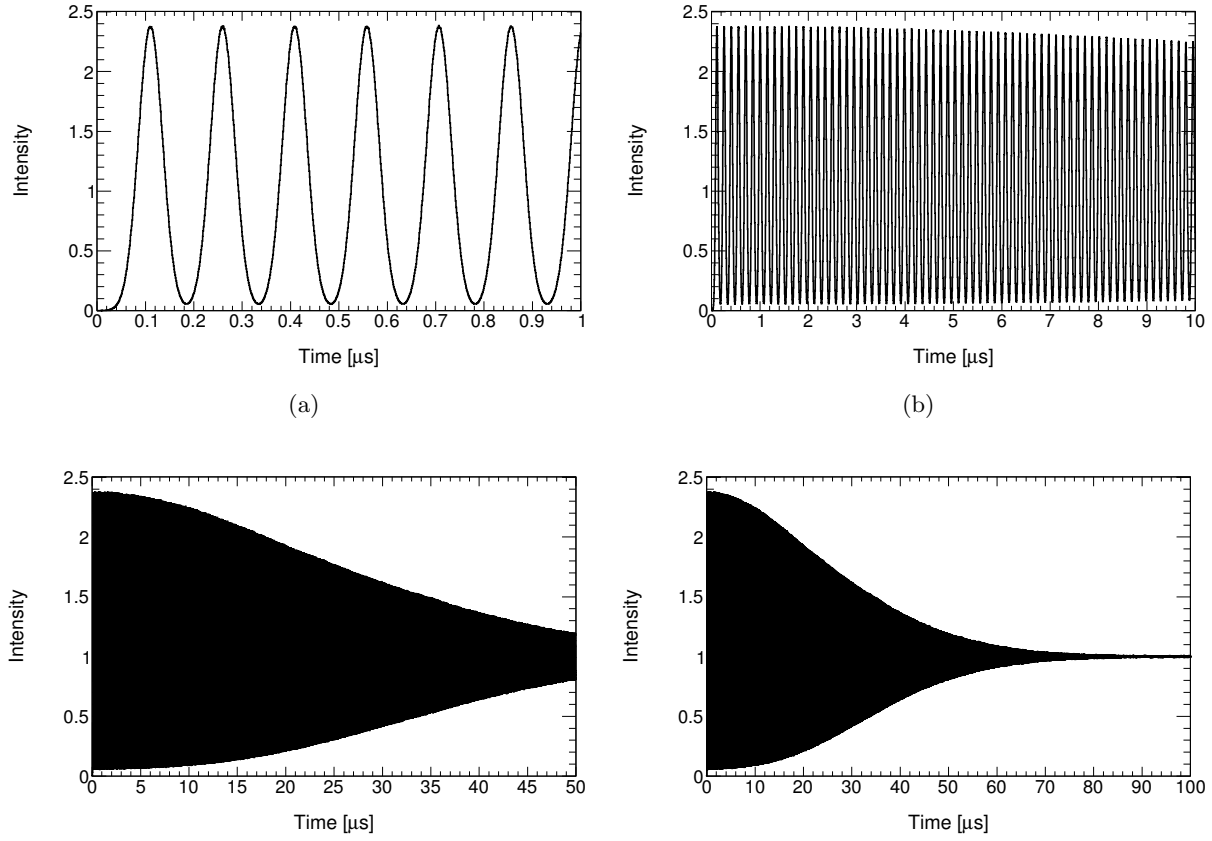


Figure 1: Fast rotation signal as a function of time [1]. This simulated signal has a Gaussian frequency/-momentum distribution with a width (one standard deviation) of 0.09% (6 kHz/0.003 GeV/c), a Gaussian initial longitudinal beam profile with a width (one standard deviation) of 25 ns and zero emittance (no transverse beam size). Four time intervals are shown: (a) 0-1 μs, (b) 0-10 μs, (c) 0-50 μs, (d) 0-100 μs.

2 Fourier analysis

For a more detailed explanation of the Fourier method refer to [2].

2.1 Fitting intensity data

For real data we do not start out with the fast rotation signal. We only have the intensity of the muons as they hit the detector. Figure 2 shows the intensity distributions of the muons for the 60–hour data set. We can get from the intensity distribution of the muons to the fast rotation signal by fitting the intensity for various parameters and then dividing by the fit function. The most distinct feature which is fit for is the exponential decay of the muons. The muons have an average momentum at the center of their orbit, of $p_0 = 3.094$ MeV and mass of $m_\mu = 0.1057$ MeV. The muon has a resting decay time of $\tau_0 = 2.197$ μs. We then multiple the muons decay time by the Lorentz factor to get a relativistic decay time of $\tau = \tau_0\sqrt{1 + \frac{p_0^2}{m_\mu^2}}$. This gives us a decay time of $\tau = 64.46$ μs. This decay of the intensity distribution over time needs to be accounted for by fitting the intensity with this decay and then dividing the intensity distribution by it to cancel its effects. The same thing must be done for other parameters. We can do a 5 parameter fit or a 9 parameter fit which includes accounting for the CBO frequency. The 5 parameter fit is given by the following [3].

$$F(t) = \frac{S(t)}{Ne^{-t/\tau}[1 + A\cos(\omega_a t + \phi)]} \tag{1}$$

Where we fit for the total amplitude N, the decay time τ, the precession amplitude A, the spin precession frequency ω_a, and a precession phase ϕ.

Once we divide by this fit function, F(t), we get the fast rotation signal which is normalized to oscillate around 1. In figure 4 the generated fast rotation signal for the 60–hour data set is shown. If instead of real data, we use Monte Carlo simulations, then we do not need to do this step because the fast rotation signal should be directly simulated and normalized to 1.

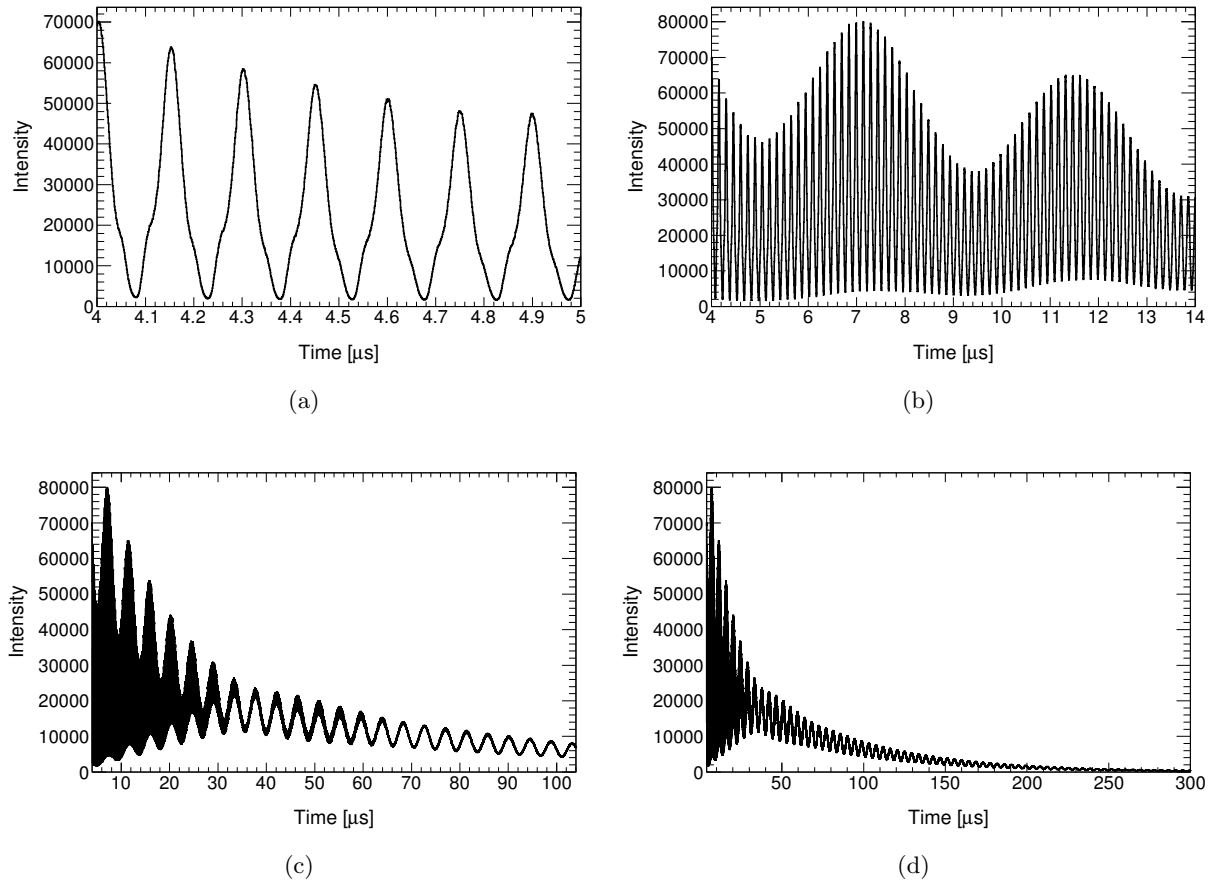


Figure 2: The intensity distribution for the 60–hour data set. Four time intervals are shown: (a) 4-5 μs, (b) 4-14 μs, (c) 4-104 μs, (d) 4-300 μs.

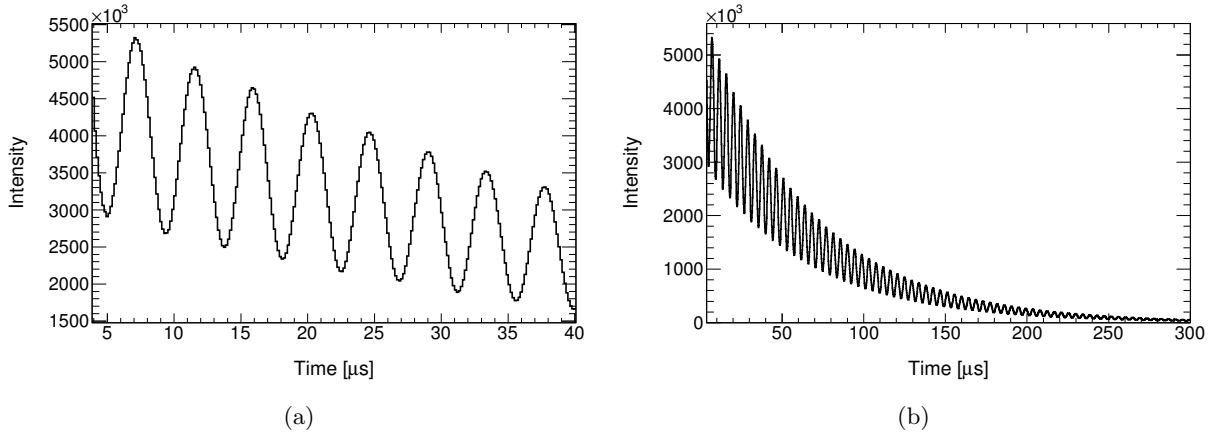


Figure 3: The wiggle plot fitted to the intensity distribution of the 60–hour data set using a 9 parameter fit. The intensity distribution is divided by this to create the fast rotation signal. Two time intervals are shown: (a) 4-40 μs, (b) 4-300 μs.

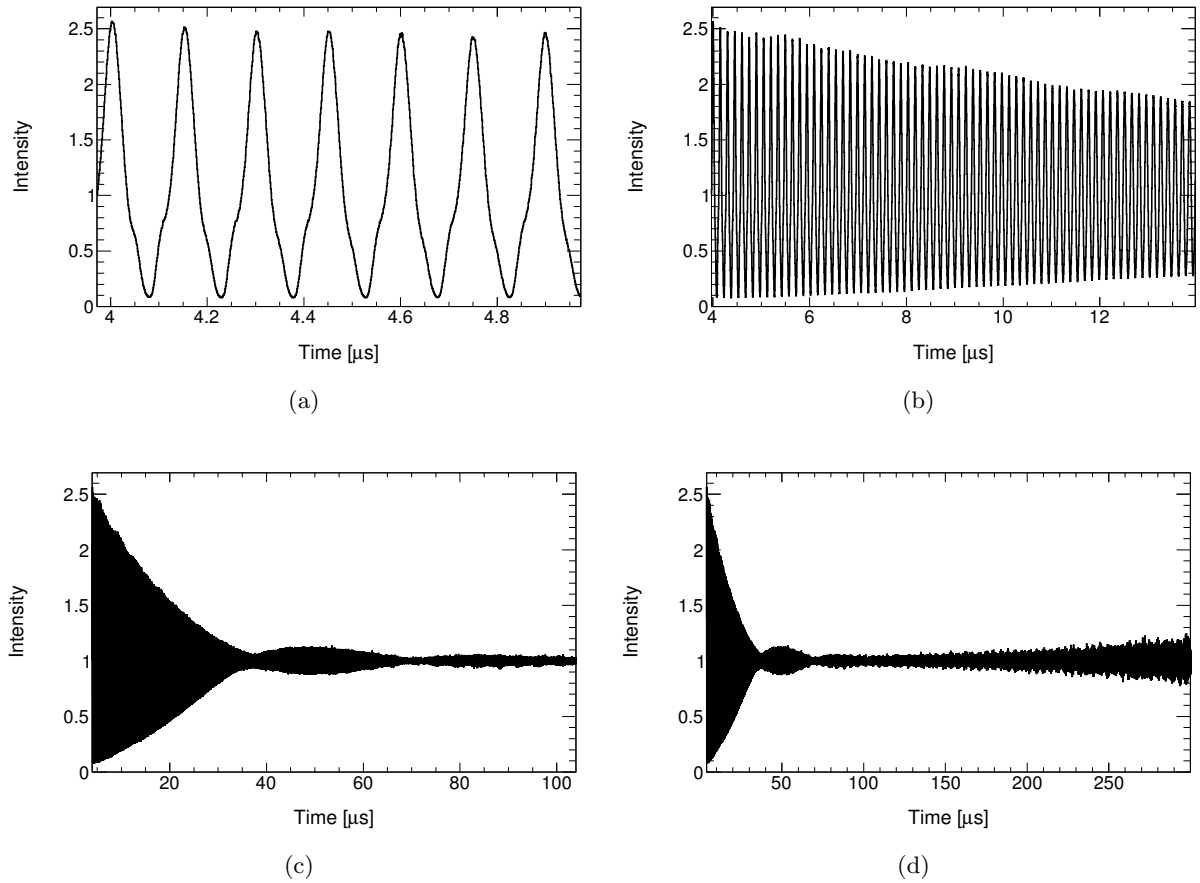


Figure 4: The fast rotation signal generated from the 60–hour data set. Four time intervals are shown: (a) 4-5 μs, (b) 4-14 μs, (c) 4-104 μs, (d) 4-300 μs.

2.2 Modified Fourier transformation

We want to recover the frequency distribution of the fast rotation signal. To do this we must do a Fourier transformation of the fast rotation signal. If the correct phase of the Fourier transformation is used, then all of the information in it is stored in the real part of the Fourier transformation [4]. The phase of the Fourier transformation for the case of the fast rotation signal corresponds to the starting time which we denote as t_0. Since all the information is in the real part of the Fourier transformation, we are only concerned with its real part. The real part of the Fourier transformation is the cosine Fourier transformation, so we can just take the cosine Fourier transformation of the fast rotation signal starting at t_0. This start time physically corresponds to the time when the center of mass of the muon beam first reaches the detector. The cosine Fourier transformation should then be done from t_0 to infinity, but since we are only have a finite amount of data, the signal cannot go on forever so we take the cosine Fourier transformation from t_0 until a later time which we denote as t_m. In figure 5 we show the recovered frequency distribution for the example 60–hour data set using the complete method.

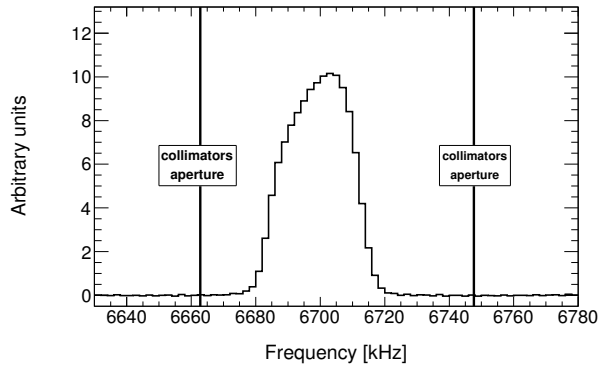


Figure 5: The recovered frequency distribution for the 60–hour data set using the complete Fourier method.

2.3 Fitting the frequency background

In the actual data, the first several microseconds since there is beam line positron interference of the muon beam. Figure 6 shows the beam line positron interference of the muon beam on the example 60–hour data set. We want to skip this section so it does not bias our recovered frequency distribution. You can see that the first approximately 4 μs are sporadic and have greater amplitude. Instead of starting the cosine Fourier transformation at t_0 we start at a later time t_s so that we skip the initial positron interference. We have to account for this missing time in the cosine transform.

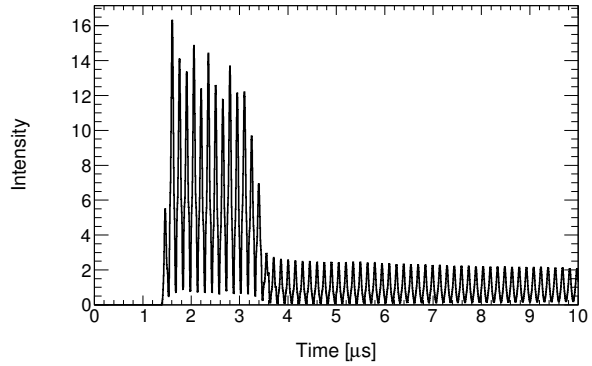


Figure 6: This is the first 10 μs of the fast rotation signal for the 60 hour data set. Notice the effect of positron interference on the first 4 μs of the fast rotation signal which makes this section unusable.

For a fast rotation signal S(t), we can recover the frequency distribution by doing the cosine transform.

$$\tilde{S}(\omega) = \sqrt{\frac{2}{\pi}} \int_{t_0}^{\infty} S(t) \cos\omega(t - t_0) dt = \sqrt{\frac{2}{\pi}} \int_{t_s}^{\infty} S(t) \cos\omega(t - t_0) dt + \sqrt{\frac{2}{\pi}} \int_{t_0}^{t_s} S(t) \cos\omega(t - t_0) dt \quad (2)$$

We let the integral between t_0 and t_s be the correction to the fast rotation signal denoted $\Delta(\omega)$, then we get a correction equal to the following equation where ω^+ and ω^- are the bounds of the collimator aperture [4].

$$\Delta(\omega) = \sqrt{\frac{2}{\pi}} \int_{t_0}^{t_s} S(t) \cos\omega(t - t_0) dt \approx \frac{1}{\pi} \int_{\omega^-}^{\omega^+} \tilde{S}(\omega\prime) \frac{\sin[(\omega - \omega\prime)(t_s - t_0)]}{\omega - \omega\prime} d\omega\prime \tag{3}$$

The correction $\Delta(\omega)$ can be approximated to be a sinc function so we can fit the background of the cosine Fourier transformation for a sinc function and then add $-\Delta(\omega)$ to cancel out the background. We could also Taylor expand the sinc function to be represented as a polynomial and fit the background with a degree N polynomial [5]. Figure 7 shows the fitting of the background of the frequency distribution with a sinc function for the 60–hour data set with t_s= 4 μs and the correct value of t_0 = 121.47 ns.

We can also use more complicated functions to fit the background by using the analytical form of the background for a Gaussian or triangular frequency distribution which can allow us to use larger values of t_s as high as 25 μs which allows us to skip scraping which continues for the first 30 μs and may have an effect on the radial distribution of the muons [5].

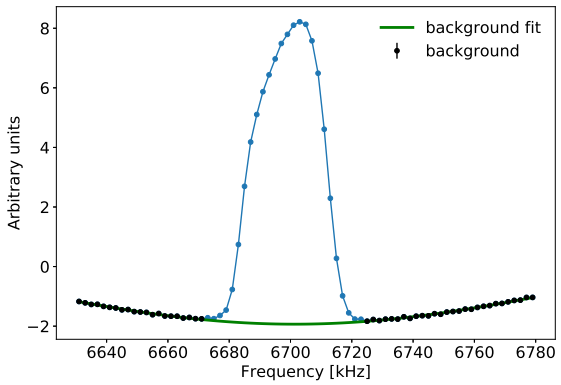
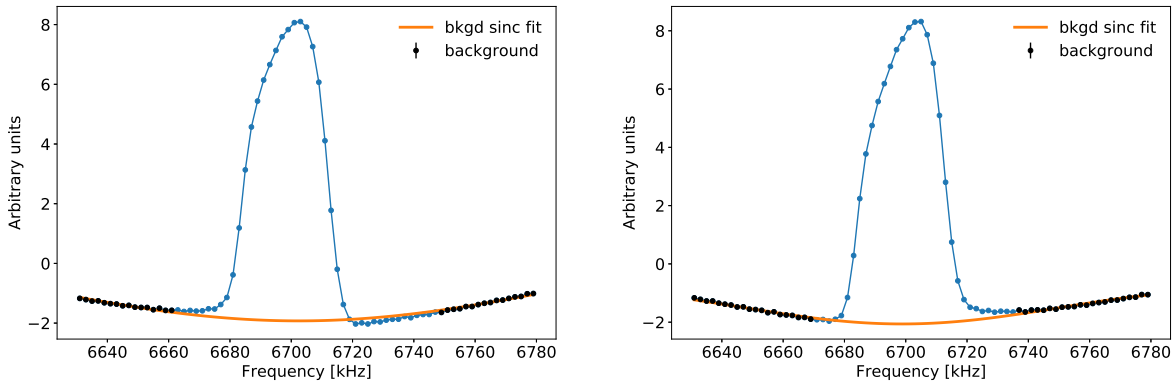


Figure 7: Fitting the background of the frequency distribution for the 60–hour data set using a sinc function with t_s = 4 μs and t_0 = 121.47 ns.

2.4 Chi square optimization of t_0

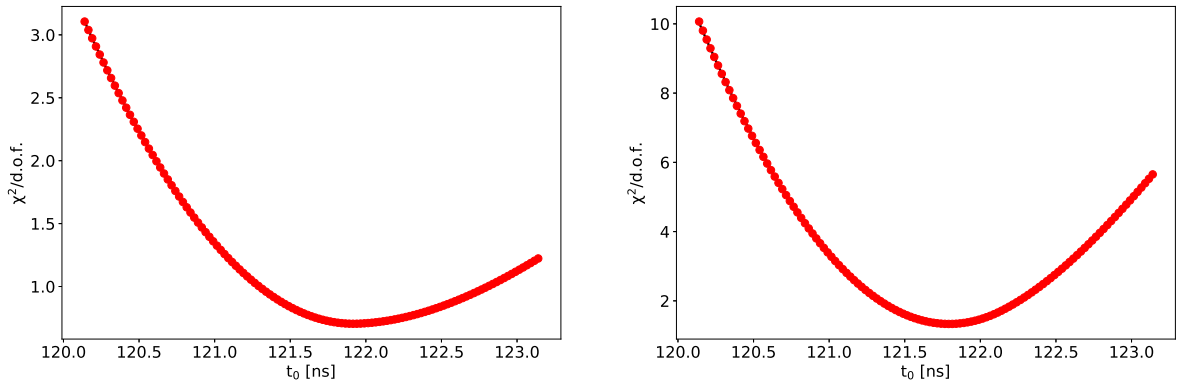
We do not know the exact value of t_0 because there is no way of knowing when the average of the beam first hits the detector. Since we do not know the value of t_0 we have to figure it out from the frequency distribution. A range of values are considered for t_0 and then the chi squared between the background fit and the frequency distribution is taken. The chi square of the background fit can be minimized where the minima of the chi squared corresponds to the correct value of t_0. We can see in figure 8 that when the wrong value of t_0 is used, the background cannot be fit correctly because the phase of the frequency distribution will be off. When a value of t_0 used is less than the actual value, like in (a), the frequency distribution will slant to the left and if the value of t_0 used is greater than the actual value, like in (b), it will slant to the right. With the wrong value of t_0 the background is no longer a continuous function because it is off phase so the background cannot be fit with a sinc function or a polynomial yielding a higher chi squared than when the real t_0 is used.



(a) (b)

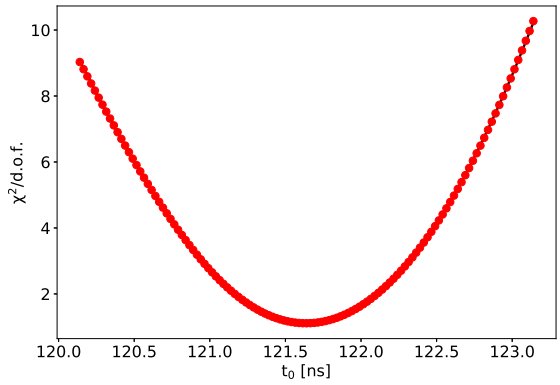
Figure 8: Phase changing with the wrong value of t_0. $t_s = 4$ μs and the true value of $t_0 = 121.47$ ns. Two values of t_0 are looked at: (a) $t_0 = 120.140$ ns, (b) $t_0 = 123.140$ ns.

When the correction is being made, the background is fit iteratively to the frequency distribution. First only the points outside the collimator aperture are used for the fit. This way we know for sure that those points correspond to non physical frequencies, since no muon can exist outside the ring. These points are considered to be frequency noise because they are unphysical values. The mean and standard deviation is calculated from the noise. In the second iteration all points which are within a standard deviation threshold from the mean are then also considered to be noise, and the t_0 optimization is done again to get a second estimate at the value of t_0. This second iteration is including points within the collimator aperture. Figure 9 shows the optimization of t_0 for 4 iterations. Notice that the minima changes between the first 3 iterations but between iteration 3 and 4 the value of t_0 converges.

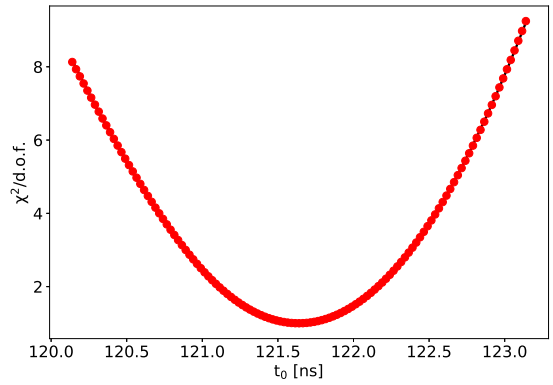


(a)

(b)



(c)



(d)

Figure 9: The χ^2 optimization of t_0. Four iterations are shown: (a) iteration 1, (b) iteration 2 (c) iteration 3, (d) iteration 4.

The t_0 minimization is done by fitting the chi squared distribution with a degree 2 polynomial. Then we find its minima by finding where the derivative of the polynomial is 0. Note that t_0 corresponds to the phase of the fast rotation signal with a period corresponding to the period of the ring T = 149.1 ns. Therefore there will be minima corresponding to the value of t_0 every period and every half a period there will be a maximum when the frequency distribution is exactly half a period out of phase. Figure 10 shows what happens when the cosine Fourier transformation is done with a value of t_0+T/2.

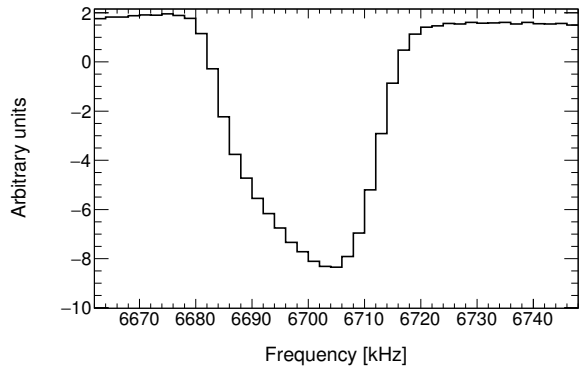


Figure 10: This is the cosine Fourier transformation for the 60–hour data set off by half a period $t_s = 4$ μs and $t_0 = 197.98$ ns.

If the value of t_0 used is off by a full period, we can still recover the frequency distribution in almost exactly the same way. In figure 11 we see that we still recover the frequency distribution correctly. We get a E-field correction which differs only by 4 ppb. If we were to use a t_0 value which is several periods away from the actual value we will get an E-field correction which significantly different from when the correct t_0 is used.

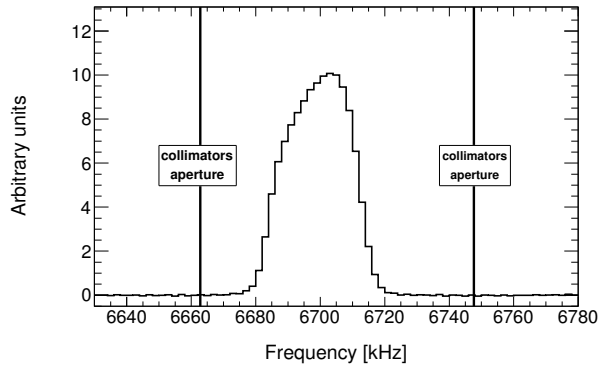


Figure 11: This is the corrected frequency distribution for the 60–hour data set off by a full period $t_s = 4$ μs and $t_0 = 270.91$ ns.

2.5 Radial distribution

We are interested in reconstructing the frequency distribution of the muons in order to calculate the electric field correction. The E-field correction can be calculated from the radial distributions of the muons. We can approximate the muons going around the ring in a circle by dividing the muons speed by the frequency distribution. Since the muons are going around the ring at nearly the speed of light, the speed of the muons are approximately constant and given by the following equation.

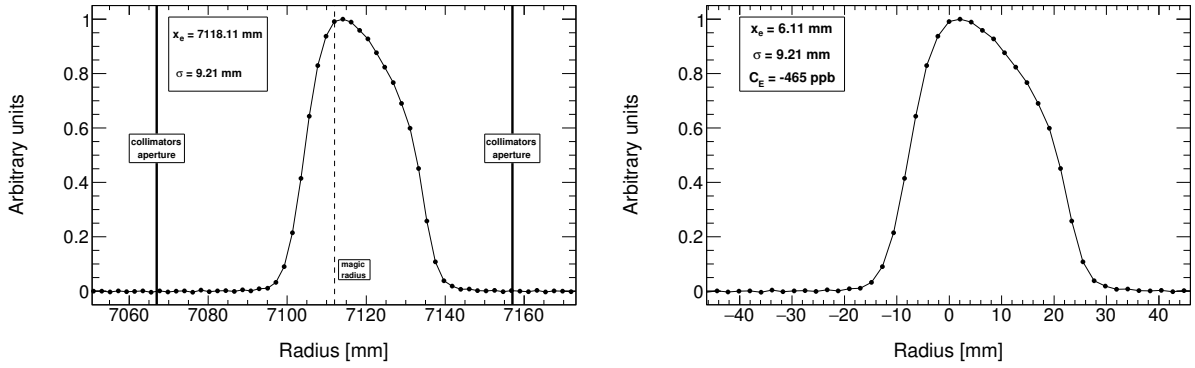
$$v_\mu = \frac{\gamma p_0}{m_\mu} = \frac{p_0/m_\mu}{\sqrt{1 + \frac{p_0^2}{m_\mu^2}}} \tag{4}$$

The velocity dependence of the muons as a function of the frequency has a negligible effect on the

radial distribution so we do not include it. The radial distribution is obtained as follows since the muons are moving approximately in a circular orbit about the ring.

$$r(x) = \frac{v_\mu}{2\pi f(\omega)} \tag{5}$$

In figure 12 we show the recovered radial distribution for the 60–hour data set. This is calculated using the corrected frequency distribution shown in figure 5.



(a) (b)

Figure 12: Radial distribution for the 60 hour data (a) Recovered radial distribution, (b) Recovered radial distribution in beam coordinates.

2.6 E-Field correction

From the radial distribution we use the linear approximation of the E-field.

$$C_E = -2\beta^2 n(1-n)(\frac{\Delta r}{r_0})^2 \tag{6}$$

Where r_0 is the magic radius, $\Delta r = r - r_0$, $\beta = v_\mu/c$, and n is the field index. This can be calculated directly from finding Δr from the radial distribution. For the 60–hour data set we get a value of C_E = -464.77 ppb.

3 Source code

3.1 Prerequisites for running the code

The Fourier analysis is implemented in `Python` and works only with `Python 3`. Its required that the following packages be installed:

- PyROOT
- NumPy
- Matplotlib
- SciPy

3.2 Downloading the code and example

The Fourier method code is available for download on GitHub (here). You can directly clone the repository by executing the following command

```
git clone https://github.com/atc93/CornellFastRotationFourier.git/
```

3.3 Running the example

Once the code has been downloaded, run it using the command.

```
python run_fourier_analysis.py config/config.json -b
```

The -b tag in the command configures ROOT to not show graphs on the screen while the code is running to prevent unnecessary slow-downs. The code will open the config folder and use the parameters defined in the file `config.json`. By default the results of the analysis are stored in a directory called `results`. The `config.json` file is initially set up to do the basic analysis on the example 60–hour data set. If the example has been downloaded then the analysis can be run immediately without changing the `config.json` file.

3.4 The important files

The important files to do the analysis includes the file `config.json` within the /`config` directory and the master script `fourier_analysis.py`. All that is required to run the analysis is to edit the `config.json` file. This is the only file which needs to be changed to fit your specific dataset. The `config.json` is initially set up to run the 60–hour data set. The 60–hour data set it stored in a ROOT file called `example.root` within the /`data` directory. The analysis is run by giving a ROOT file in the folder /`data` and then setting up the `config.json` for the specific data file. In section 4 we go over how to change the `config.json` to use other data sets.

3.5 Example output

Once the analysis is complete, the results will be stored in the file called `example.root` within the /`results` directory. Within `example.root` there will be plots of the analysis which took place. There will be plots of the intensity distribution along with the wiggle plot which is fit to it, and the residuals of that fit. Also shown is the fast rotation signal created by dividing the intensity by the wiggle plot. All of these plots are shown for several different time domains. Plots of the cosine Fourier transformation of the fast rotation signal are also saved as well as the background which is fit to it. Furthermore, the resulting corrected frequency and radial distributions are plotted. The t_0 optimization which is done will be stored within another directory called `t0_optimization` which shows the background being fit for each value of t_0 used in the optimization.

The resulting distributions are also stored within ROOT files so that the resulting histograms are saved. A ROOT file called `results` will be created which contains the cosine Fourier transformation, the corrected frequency distribution, and the radial distribution so that they can be saved and compared. Another ROOT file called `fastrotation.root` will contain the fast rotation signal created from the intensity distribution.

A text file will also be made called `results.txt` containing the resulting values used for the analysis. There is a lot of important information stored within this text file. The E-field correction is under *c_e* measured in ppb. The mean radius is under *eq_radius* and is measured in beam coordinates so mm from the magic radius. The standard deviation of the radial distribution is also stored under *std* measured in mm.

The json file which is used to configure all the parameters for the analysis, called `config.json` by default, is also stored so that the user can go back and see which values they chose for each parameter.

3.6 Editing the code and bug repots

The user is not able to push any edits they may want to make to GitHub. If there is a feature you want to add or there is a bug in the code email A. Chapelain.

4 First analysis

Here we will describe how to configure the `config.json` file to any data set you want to analyze. A detailed list of all the possible parameters that can be changed can be found in section 6.

4.1 Saving data

All data used should be stored using a ROOT TH1D histogram time series. The data set can be placed in the directory called `data`. Then in your `config.json` file within the `config` directory, change the parameter "root_file" to the path of the data file. If it was called `new_data.root` and stored within the `data` directory then the parameter "root_file" should be `data/new_data.root`. The parameter "hist_name" should be changed to match the name of the TH1D histogram within the ROOT file. The parameter "tag" can also be changed to the name of the output of the analysis placed in the results directory. The parameter "field_index" should also match the field index of the data set being used. For Monte Carlo data the field index may be somewhat arbitrary and a value matching the 60–hour data set can be used for n = 0.108.

4.2 Creating the fast rotation signal

The data ROOT file should either contain the intensity of the muons for real data or the fast rotation signal for Monte Carlo data. If Monte Carlo data is being used so that the input data is already the fast rotation signal, within the `config.json` file the parameter "n_fit_param" should be set to "0" so that no wiggle function is fit. If instead an intensity distribution of muons is being used like in real data, then "n_fit_param" should be set to "9" for a 9 parameter fit which does the normal 5 parameter fit and also fits for the CBO amplitude. When using the 9 parameter fit, the CBO frequency is set by the parameter "cbo_freq". You may use the 5 parameter fit by setting "n_fit_param" to "5" and you can see that the analysis should yield nearly identical results. The parameter "rebin_wiggle_factor" should also be changed to match the binning of the intensity distribution. If your data uses 1 ns binning then it is recommended to use a "rebin_wiggle_factor" of around 149 to get optimal fitting. This factor can be scaled accordingly, however, if you are using 2 ns time bins you may wish to only use a "rebin_wiggle_factor" of 75. If the time binning of your data is too fine then the parameter "rebin_frs_factor" can be used to make reasonable

time bins. So if your data is in 0.1 ns time bins you can set a "rebin_frs_factor" of 10 to get 1 ns time bins and use a "rebin_wiggle_factor" of 149. For all purposes 1 ns time bins are sufficient.

Another thing that must be considered in creating the fast rotation signal is when the wiggle fit should start. The first approximately 4 μs are effected by positron interference which can be seen in figure 6 for the 60–hour data set. We also have to consider the effects of scraping which occurs for about the first 30 μs. The parameter "start_fit_time" can be set to a value of at least 30 μs so that these effects are skipped. There is a tradeoff though because the muons debunch over time so the early times hold some of the best data. A value of 30 μs should be ideal to skip scraping while using the most possible data.

4.3 Choosing t_s and t_m

When doing the cosine Fourier transformation only the fast rotation signal between t_s and t_m is used. The value of t_m must be less than the length of the data being used but greater than t_s. There is a tradeoff in the value of t_m used. The more signal used the higher the resolution of the recovered frequency distribution will be. The less signal used the more the frequency distribution will be effected by spectral leakage. On the other hand, the muons will debunch around the ring over time, and by at most 100 μs the muons will be fully debunched around the ring. This makes data at later times hold much less information about the muons orbits, so if later values are used greater than 100 μs then you will just be mostly adding noise. This noise in the fast rotation signal also exponentially increases over time which comes from the fitting of the exponentially decay intensity distribution. In figure 4 (a) you can start to see the exponentially increasing noise of the fast rotation signal for the 60–hour data set after 100 μs. A value of t_m should be chosen between 150 and 400 μs so that neither of these issues are too prominent.

The choice of t_s is a quintessential step. If Monte Carlo data is being used then a value of t_s can be used so long as $t_s \geq t_0$. The value $t_s = t_0$ corresponds to when there is no background which needs to be fit because the full cosine Fourier transformation is taken. For real data, however, we want to skip the initial positron interference which last for about the first 4 μs. This is seen in 6 for the 60–hour data set. We must therefore use a value of t_s which is greater than 4 μs to skip the interference. We may also be interested in skipping the effects of scraping which lasts up to the first 30 μs of the fast rotation signal. There is a tradeoff here when choosing t_s in that the larger the value of t_s, the harder it is to fit the background of the frequency distribution. The limitation on the current method makes using a value of t_s which is greater than 20 μs highly inaccurate because the background will highly interfere with the signal. This means that not all of the scraping can be skipped without huge issues in the background fitting. It is recommended to then use a value of $t_s = 4$ μs which skips the positron interference since scraping does not seems to have a significant effect on the reconstructed frequency distribution.

4.4 Optimizing t_0 and background fit

The parameter "background_correction" should be set to "fit" and "background_frequencies" set to "all" so that the background is fit using all of the relevant frequency distribution. The value of t_0 should first be optimized over a large range of values to find t_0 using a large range of "lower_t0" and "uppter_t0" with a large "t0_step _size". This is best done by first setting the parameter "background_fit" to "poly" with a "poly_order" of 2. This does the most basic fit which works best when looking over a large range of t_0

since the chi square will be the most parabolic. The interval you look over t_0 should be much less than half a period so less than 74 ns to ensure that a minimum can be found. For the large range a step size of 0.001 μs which is 1 ns is sufficient. Then the analysis should be run as described in section 3.3. The value of t_0 optimized will be outputted in the terminal as well as in the text file `results.txt`. As stated in section 2.4 the range of t0 looked at must also know within one period to correctly optimize t_0.

Once an initial value of t_0 is estimated using the "poly" fit, a final background fit can be done using the "sinc" fit. The `config.json` file should be edited again to run the final analysis by changing the "lower_t0" and "upper_t0" parameters to be $\pm$ 1 ns from the estimated t_0. The step size should then be made much finer to a value of "t0_step_size" like 0.000025 μs which is 25 ps. The "background_fit" should be changed to "sinc" to get more accurate results. Run the analysis once again and this will be the final result with the best possible optimization of t_0 and background fit.

4.5 Comparing results

If Monte Carlo data is being used, then the recovered frequency distribution can be compared to the simulated frequency distribution. The simulated frequency distribution must be stored as a TH1D histogram inside a ROOT file. The comparison is done by setting the parameter "compare_with_truth" to "true" and "truth_root_file" should be the path to the ROOT file. If you store the ROOT file within the `data` directory and called it `truth.root`, then this parameter should be set to "data/truth.root". The parameter "truth_histo_name" should be set to the name of the histogram within the ROOT file. If real data is being used like in the example 60–hour data set, then there is nothing to compare your results to since the real frequency and radial distributions are unknown.

5 Advanced analysis

We have so far gone over the basic analysis which can be done with the code, but there are more optional features to help assess the statistical and systematic uncertainties which exist with the Fourier method.

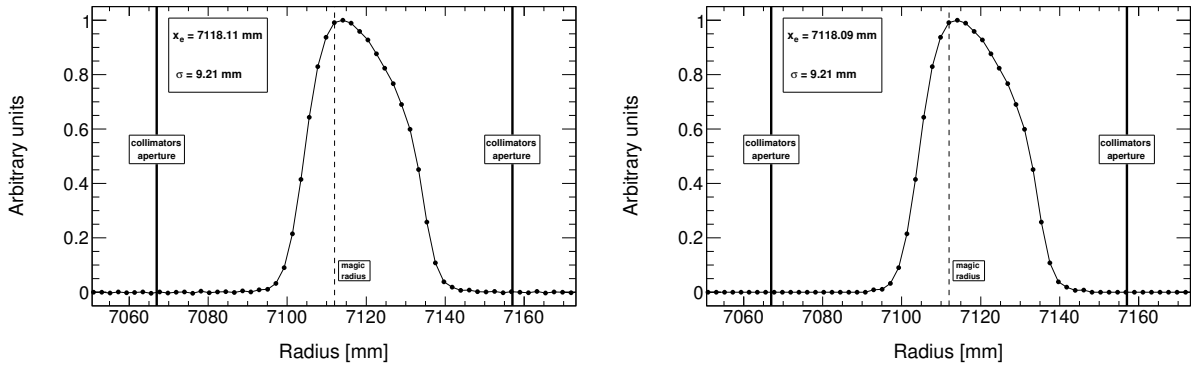
5.1 Background removal

The frequency background is fit iteratively by determining which points are noise based on the standard deviation of the noise. All points which are less than the parameter "t0_background_threshold" number of standard deviations away are considered to be noise.

Removing the background works the same way in that when the final radial distribution is found, the standard deviation of all points which are considered to be noise is calculated, and then all points which are less than the parameter "background_removal_threshold" number of standard deviations away are determined to be noise and are then set to zero so they do not have an effect on the E-field calculation. This is done because the E-field calculation described in 2.6 is proportionate to the distance from the magic radius squared. This means that radial bins far away from the magic radius which are close to zero can still have non-negligible effects on the E-field calculation.

The background is only removed when the parameter "remove_background" is set to "true." It is recommended that the parameter "background_removal_threshold" should not exceed 3 otherwise it is

possible that part of the signal can be removed along with the noise. Shown in figure 13 is the comparison of the recovered radial distribution for the 60–hour data set with and without removing the background for a "background_removal_threshold" of "3". The recovered E-field only differs by 1.18 ppb so removing the background does not make much of a difference in this case.



(a) (b)

Figure 13: A comparison of the recovered radial distribution of the muons for the 60–hour data set with and without removing the background. The background removal threshold is 3σ. (a) Background included, C_E = -464.77 ppb (b) Background removed C_E = -463.59 ppb.

5.2 Statistical fluctuation

The statistical fluctuation of the method can be measured by seeing how the value of the E-field correction changes when the method is run using a number of pseudo data experiments. The pseudo data experiments are done by varying each time bin in the intensity distribution of the muons. For a time bin with N muons in it, we vary this value by $\pm\sqrt{N}$. This follows from the poisson statistics for large values of N. We do this for each time bin to create a new statistically fluctuated intensity distribution of the muons. For each statistically fluctuated intensity distribution we run the entire analysis to obtain a final value for the E-field corrections. If enough statistical fluctuations are done then the distribution of E-field correction will be approximately Gaussian because of the central limit theorem. It is recommended that at least 100 pseudo distributions are used in order for the central limit theorem to apply. We can use some threshold like 3 standard deviations of the E-field correction distribution as a measure of the statistical uncertainty.

Figure 14 shows the results from statistical fluctuations on the Run-1 60-hour data set. We can see that the E-field correction is centered at -463.05 ppb with a spread of 0.91 ppb, t_0 is centered at -121.64 ns with a spread of 0.016 ns, the equilibrium radius is centered at 6.11 mm with a spread of 0.012 mm, and the width is centered at -9.13 mm with a spread of 0.010 mm. The statistical uncertainty in the E-field correction has a spread of less than one ppb so the statistical uncertainty is very minor compared to the systematic uncertainty.

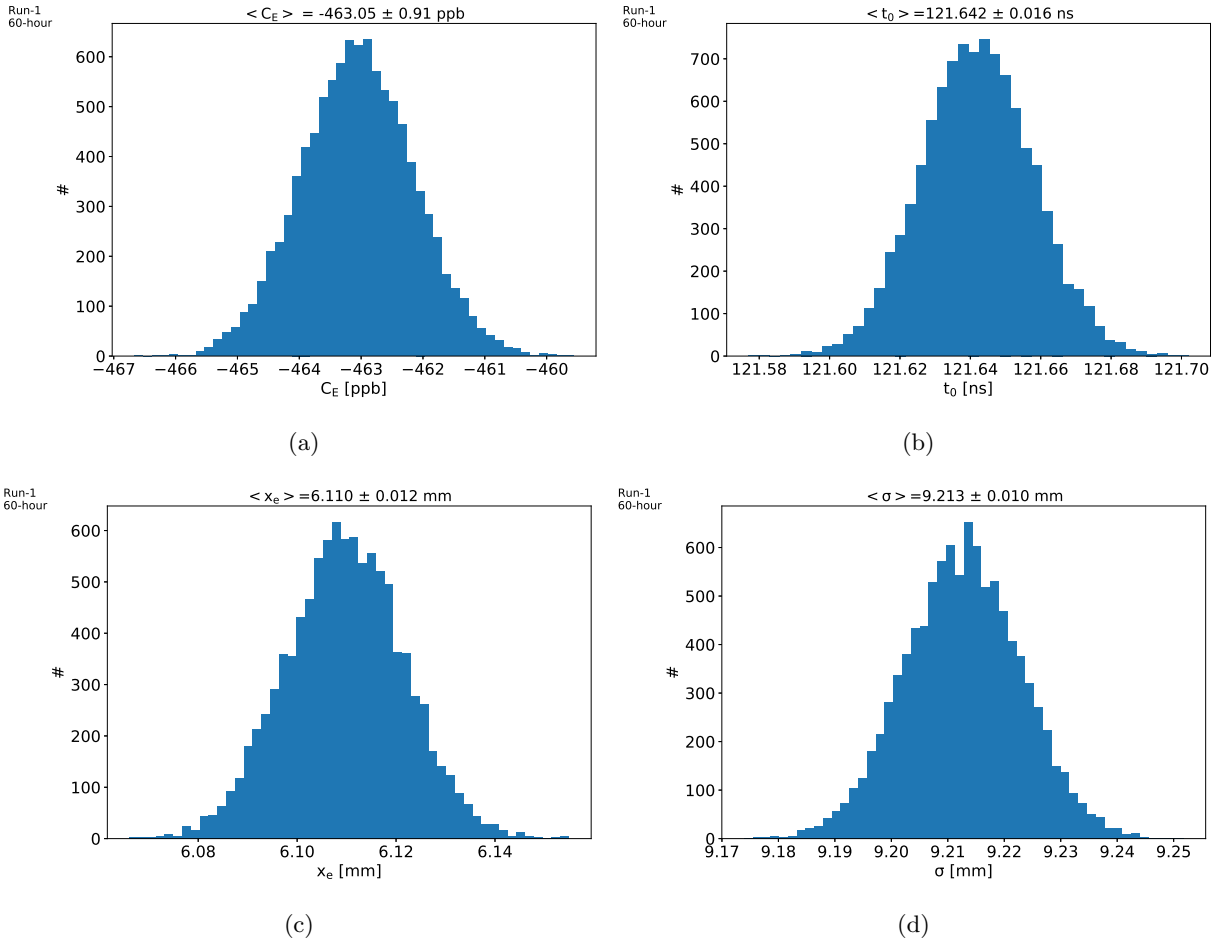


Figure 14: The results from statistical fluctuations of the example 60–hour data set. Four different start times are shown: (a) E-field, (b) t_0 (c) equilibrium radius, (d) standard deviation.

5.3 Scans

The following scans will print out the analysis for the range of values used in the scan parameters. This can help to assess the systematic uncertainty associated with changes in the scanned parameters. The resulting E-field correction of each step of the scans are stored within the text file `results.txt`. Note that only one scan can be run at a given time.

When running a scan, the range of the background used for fitting can be fixed in order to eliminate the boundary as a source of systematic uncertainty which can interfere with the results of a scan. To do this set "fix_fit_bound" to "true" and then set the boundary "fit_lower_bound" and "fit_upper_bound" to the desired boundary range. Values for "fit_lower_bound" and "fit_upper_bound" may be found by first running the analysis with "fix_fit_bound" as "false" and then use the lower and upper bounds found which were optimized.

5.3.1 t_0 scan

The t_0 scan can allow the user to asses the systematic uncertainty associated with using a wrong value of t_0 by saving the results for each value of t_0 used in the chi squared optimization. Refer to figure 8 to see how

using the wrong value of t_0 effects the cosine Fourier transformation and makes the background fit have a higher chi squared. The scan is run by changing the parameter "run_t0_scan" to "true". The t_0 then outputs each results between the parameters "lower_t0" and "upper_t0" with the step size of "t0_step_size".

5.3.2 t_s scan

The background becomes more nonlinear the larger the value of t_s and it is harder to fit with a sinc function. The background cannot be fit by a polynomial after around $t_s = 5$ μs. The background can no longer be reasonably be fit by a sinc function after about $t_s = 15$ μs. We can also use the analytic form of a Gaussian or Triangular frequency distribution which allows us to reliably use values of t_s larger than 25 μs [5].

In figure 15 we show the background being fit fo different values of t_s using the sinc fit. Notice that by $t_s = 16$ μs the background cannot be properly fit because the background significantly interferes with the signal. This is why we cannot accurately recover the frequency distribution when t_s is this large. Doing a t_s scan can assess the upper bounds with how large t_s can be for a given data set. The scan is run by changing the parameter "run_tS_scan" to "true". Then "lower_tS" and "upper_tS" should be set to the upper and lower bounds of the t_s values which you want to scan. The parameter "tS_step_size" is the step size within the range of "lower_tS" and "upper_tS".

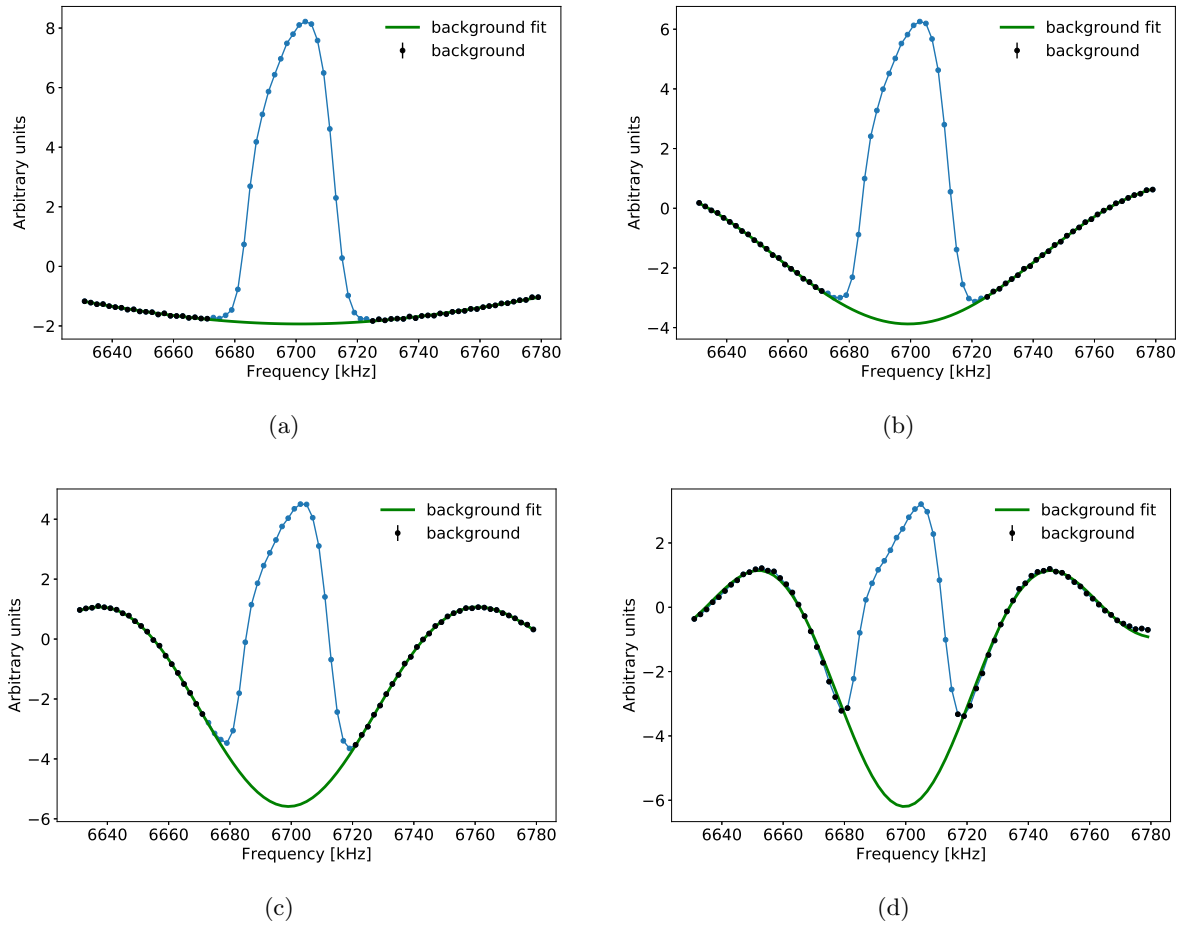


Figure 15: The background fit for different values of t_s in the example 60–hour data set. Four different start times are shown: (a) $t_s = 4$ μs, (b) $t_s = 8$ μs (c) $t_s = 12$ μs, (d) $t_s = 16$ μs.

5.3.3 t_m scan

There is a tradeoff between using more data with a large value of t_m, or including less noise with a small value of t_m. This is because the muons debunch around the ring over time and by 100 μs the they are fully debunched so using more data than that just adds noise to the signal. On the other hand, the less data we use the lower the frequency resolution will be leading to more spectral leakage. The t_m scan can help to find the sweet spot where we do not add too much, but we add enough of the signal to avoid problems with spectral leakage. The ideal value of t_m will be somewhere between 150 and 400 μs. In figure 16 we show the recovered frequency resolution for different values of t_m. Notice that when $t_m = 100$ μs there are small ripples outside the collimator aperture which comes from spectral leakage. When $t_m = 400$ μs we can see small amounts of noise outside the collimator aperture as well which comes from adding a large amount of data where the muons are fully debunched around the ring. We can see that the value of t_m does not make a huge difference but the t_m scan can help to find which values are reasonable. The scan is run by changing the parameter "run_tM_scan" to "true". Then "lower_tM" and "upper_tM" should be set to the upper and lower bounds of the t_m values which you want to scan. The parameter "tM_step_size" is the step size within the range of "lower_tM" and "upper_tM".

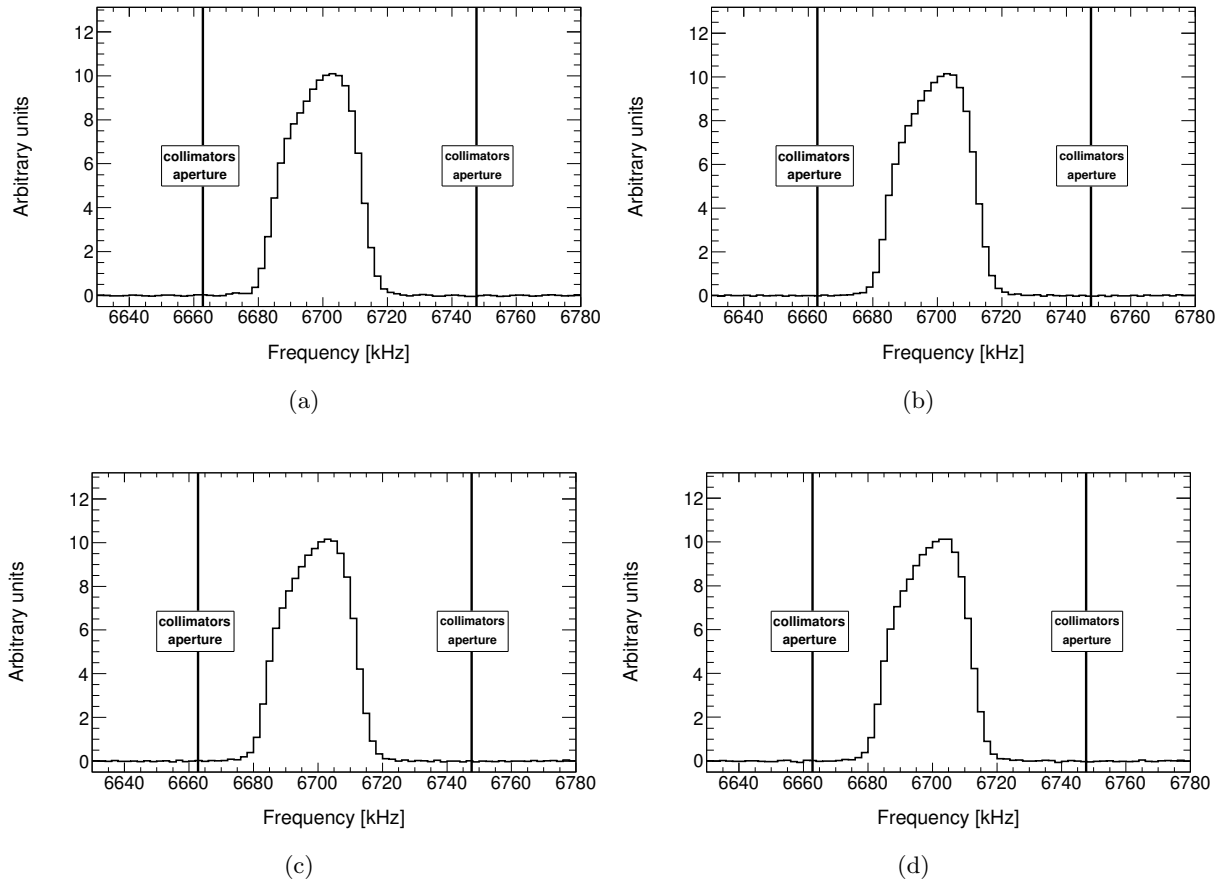


Figure 16: The corrected frequency distribution for $t_s = 4$ μs with different values of t_m. Four different stop times are shown: (a) $t_m = 100$ μs, (b) $t_m = 200$ μs (c) $t_m = 300$ μs, (d) $t_m = 400$ μs.

5.3.4 Frequency step scan

There is a similar tradeoff as with the value of t_m, since using a large frequency step size will limit the amount of noise but lose some information that falls in between frequency bins, while using a small frequency step size can give a crisper frequency distribution but has extra noise coming from spectral leakage. The sweet spot for the frequency step size is somewhere in between a 1 and 3 kHz bin width. In figure 17 we show the frequency background fitting for different frequency step sizes. For a step size of 1 kHz there are many points to fit, however if a small value of t_m is used then the frequency can suffer from spectral leakage since the frequency resolution is being oversampled. Here we use a $t_m = 300$ μs so there is not a lot of spectral leakage. For the frequency step size of 4 kHz there are not many points for the background to be fit with and information can be lost between frequency bins so we get a worse fit.

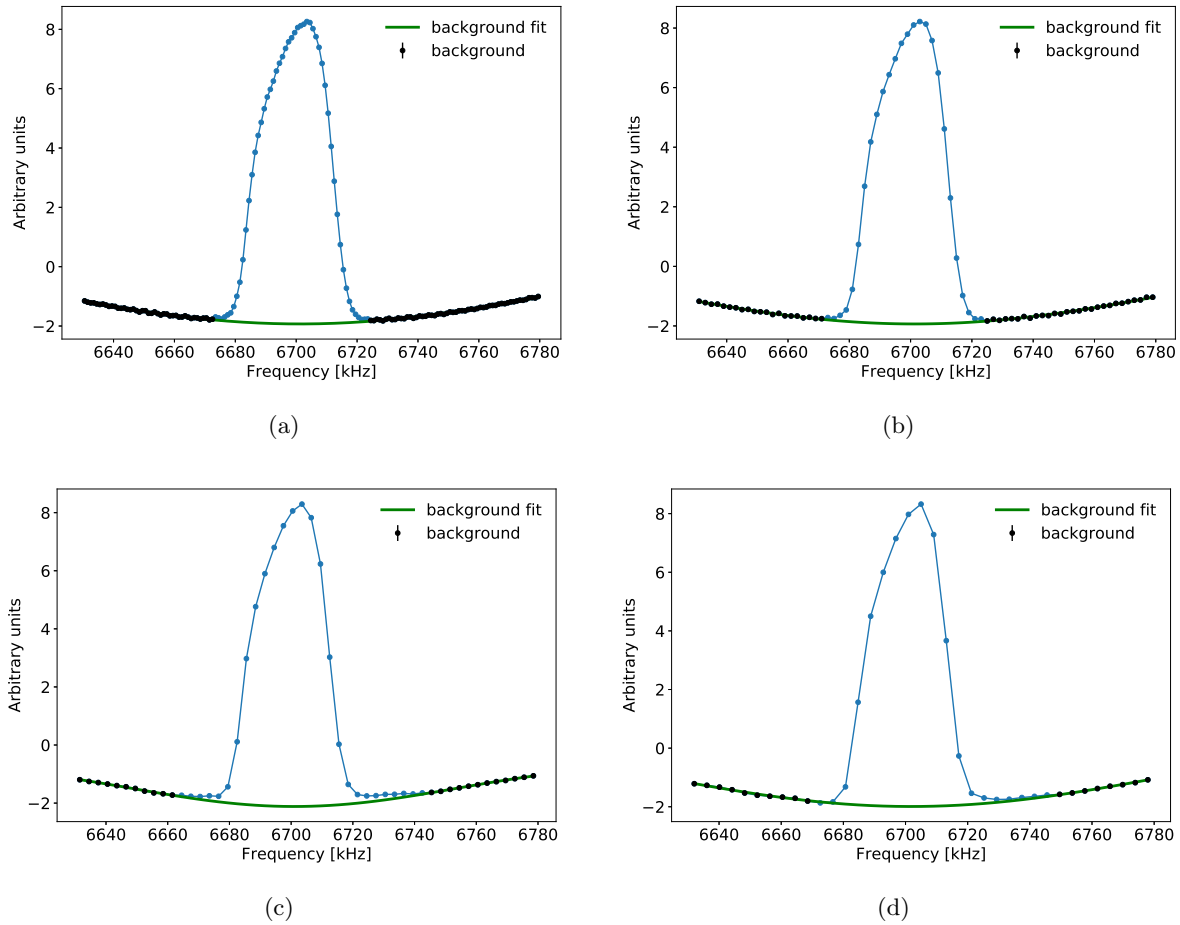


Figure 17: The fitted background for different frequency step sizes for $t_s = 4$ μs and $t_m = 300$ μs. Four different step sizes are shown: (a) 1 kHz, (b) 2 kHz, (c) 3 kHz, (d) 4 kHz.

5.3.5 Background threshold scan

The noise is iteratively found by first considering only the frequency bins outside collimator aperture to be noise, then the mean and standard deviation are calculated from the noise. The points which are within the background threshold standard deviations away from the mean are also considered to be noise. The mean and standard deviations are then recalculated and points are found to be noise again. This is done iteratively and converges after at most 5 iterations. Only the points considered to be noise are used in fitting the background. The scan is run by changing the parameter "run_background_threshold_scan" to "true". Then "lower_background_threshold" and "upper_background_threshold" should be set to the upper and lower bounds of the background threshold which you want to scan. The parameter "background_threshold_step_size" is the step size within the range of "lower_background_threshold" and "upper_background_threshold".

The background threshold scan can help the user make sure that the points used in fitting the background are actually part of the background and not the signal. If the threshold is too small then not enough points will be fit leading to a smaller chi squared in your fit and a less accurately recovered frequency distribution. If too many points are used though then part of the signal may be used in the background fitting

again giving a worse fit. A good middle ground for the background threshold is between 2 and 3 standard deviations. This scan can help find the optimal value. In figure 18 different background thresholds are shown. We can see that only a couple of extra points near the signal are included in the fit between a background threshold of 1σ and 4σ. It is still important because for 3σ and 4σ thresholds shown in (c) and (d), it appears that part of the signal is included in the fit which we want to avoid.

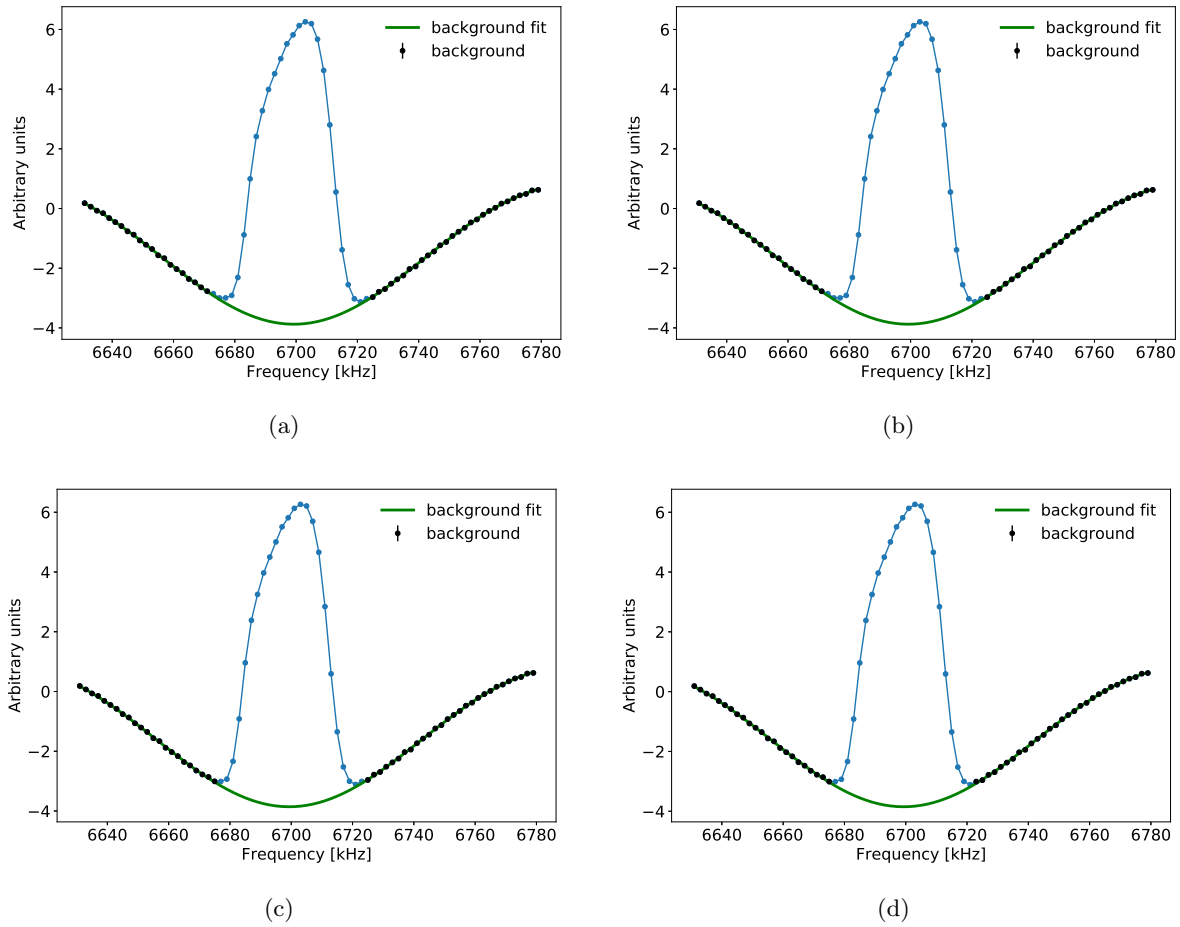


Figure 18: The fitted background for different background thresholds for $t_s = 8$ μs. Four different background thresholds are shown: (a) 1σ, (b) 2σ, (c) 3σ, (d) 4σ

5.3.6 Remove background threshold scan

This is a scan for the optional removal of the noise background on the radial distribution. This is only relevant if the user is removing the background. The standard deviation and mean of the noise are calculated and all bins which are greater than the remove background threshold are set to be zero. There is no clear threshold which says when the noise ends and the signal begins so this scan allows the user to eliminate the noise without eliminating part of the signal. The scan is run by changing the parameter "run_background_removal_threshold_scan" to "true". Then "lower_background_removal_threshold" and "upper_background_removal_threshold" should be set to the upper and lower bounds of the background threshold which you want to scan. The parameter "background_removal_threshold_step_size" is the step size within the range of "lower_background_removal_threshold" and "upper_background_removal_threshold".

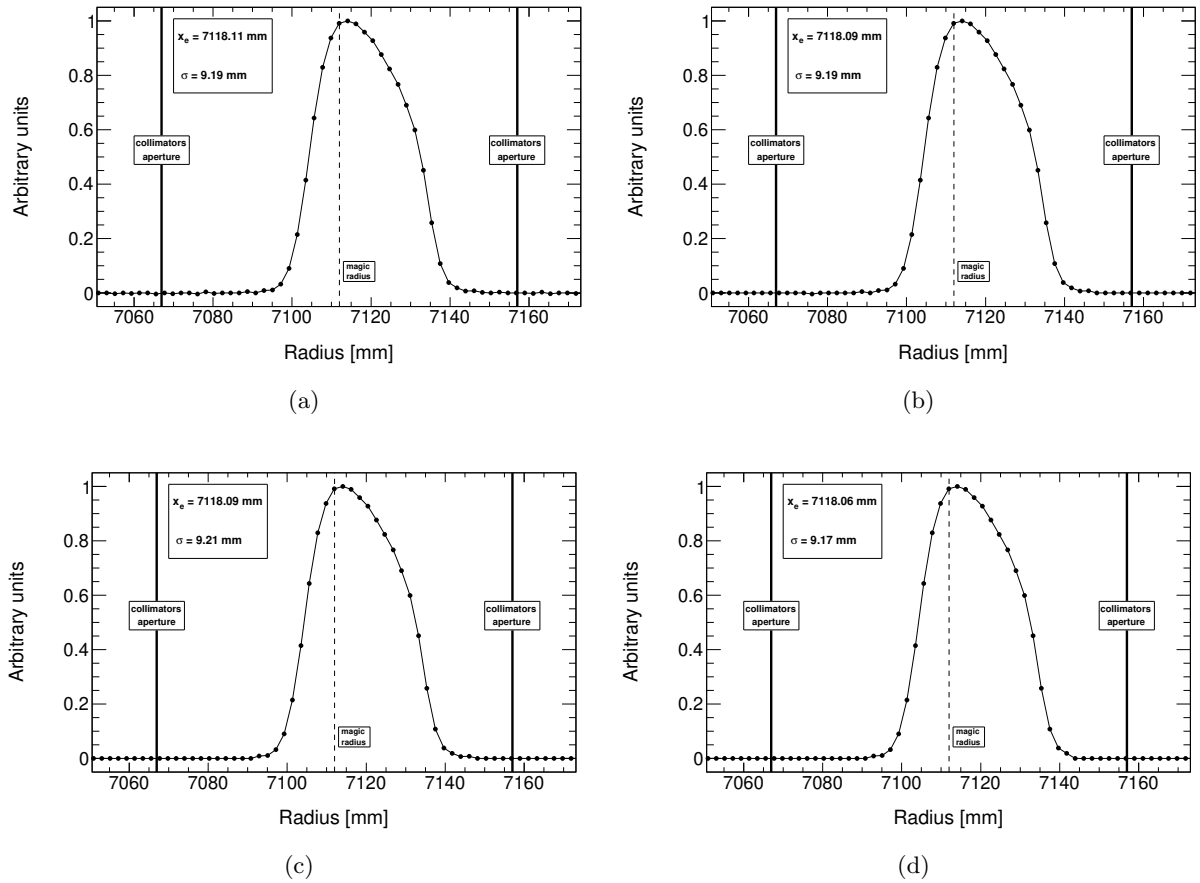


Figure 19: The radial distribution for different remove background thresholds. Four different remove background thresholds are shown: (a) 1σ, (b) 2σ, (c) 3σ, (d) 4σ

6 Parameters

Here we will define all the possible changes that can be made to the `config.json` file to tailor the analysis to the needs of the users with other data sets including both real and simulated data. Note that all time values should be entered in units of μs and all frequency values should be entered in kHz.

6.1 Basic Parameters

tag This will be the name of the directory which is output inside the `results` directory.

root_file This is the path which tells the program where to find the data one wishes to use. In the example, the data is stored in the ROOT file `example.root` inside the directory `data` so the path this parameter will be is `data/example.root`.

histo_name Within the root file the intensity distribution or the fast rotation signal must be stored by some name. This parameter must match that name so that the data can be extracted. For the example, the ROOT file `example.root` contains the intensity spectrum for each calorimeter with each bunch. The first histogram is named `allCalosallBunches\intensitySpectrum`. This contains

the merged data from each colorimeter and bunch. If you wants to looks at individual calorimeters and bunches then the name should be changed to the desired one.

background_correction The options here are "fit" or "integral." The "fit" option is to do the standard sinc/polynomial fit which we described above. If one chooses to use the "integral" option then the background will be calculated into the equation 3. We directly calculate the correction by plugging the frequency distribution into the equation for the correction $\Delta(\omega)$. The caveat is that we need to plug in the corrected frequency distribution into the equation, which is what we want to find. We can get around this problem by making a first approximation to estimate what the actual frequency distribution should look like and then plugging this in for the correction. This first approximation is made by taking the minimum of the frequency distribution and then lifting it up to the axis. Then this first approximation is plugged into the correction equation to get the frequency background. This background is then optimized by multiplying and adding constants which minimize the amount of noise outside the collimator aperture. Overall the fit feature is usually more accurate.

background_fit The options here are either "sinc" or "poly." The sinc optimization is usually more accurate. If one is not sure within a small couple ns ranger t_0 is, then one should use the "poly" option to scan over a large range for t_0. Then once the t_0 is found within a small range, one can switch to the "sinc" fit to get the most accurate results.

background_frequencies The options here are either "all" or "physical." If "all" is used then the background is fit using points both outside and within the collimator aperture by iteratively finding which points are considered to be noise and which are actually the signal. If one chooses "physical", then the points outside the collimator aperture will only be used for the first iteration.

remove_background The options here are "true" or "false." If "true" then the points which are determined to be noise are set to zero in the radial distribution. This means those points considered to be noise are not included in the E-field calculation. The points are determined to be noise through the iterative process used in the χ^2 optimization of t_0. This is useful to do because points which are far away from the magic radius can have large effects on the E-field calculation because it scales like $(r - r_0)^2$. Removing the background should eliminate the unphysical tail of the radial distribution.

background_removal_threshold If remove_background is set to "true" then this is the standard deviation threshold that which determines which points are noise. After the iterative process initially determines which points are noise, the standard deviation and mean of those noise points are calculated and any point in the radial distribution which is less than this threshold σ away from the mean are set to 0.

t0_background_threshold This is the number of standard deviations bellow which points are considered to be noise when for the background fit. If a value of 0 is used then only the points within the collimator aperture. It is recommended to only use a maximum of about 3σ because if a higher threshold is used then it is possible for parts of the signal to be considered noise.

lower_t0/upper_t0 This is the range of value which are considered when optimizing t_0. It is important that when using the "sinc" function for the background fit that this range is less than 5 ns or t_0 will

not be properly optimized. It is recommended to first find t_0 within a few nano seconds using the "poly" fit, and then switching to the "sinc" fit with a tight range for t_0. The range for t_0 should always be less than half a period or it is possible to get a t_0 which is wrong by half a period and get an upside down frequency distribution like in figure 10.

t0_step_size This is the step size that is the chi squared optimization looks over between lower_t0 and upper_t0. When going over a wide range, a t0_step_size of 1 ns can be used and then when fine tuning the value of t_0 around a small range a t0_step_size of 25 ps can be used.

tS This is the time at which the cosine Fourier transformation will start. No time less than t_s in the fast rotation signal will be looked at. It is recommended that the value of t_s used is at least 4 μs for real data so that the initial scraping is skipped. The analysis begins to significantly lose accuracy if a value for t_s is greater than 15 μs. One should never use a value for t_s which is less than t_0.

tM This is the time at which the cosine Fourier transformation ends. No time less greater than t_m in the fast rotation signal will be used. Make sure the value of t_m is not greater than the length of the fast rotation signal. The muon beam will spread out along the ring over time, so the first approximately 50μs contain most of the useful frequency information. After around this time the fast rotation signal begins to be dominated by noise, so because of this is not always better to use a longer signal. It is recommended to use a value for t_m which is between 150 and 400 μs. One should experiment to see which value yields the best results, because there is a tradeoff of having a better frequency resolution with a large value for t_m but having less noise for small values of t_m.

n_t0_opt This is the number of iteration used to find the value of t_0. Each iteration the noise mean and standard deviation is recalculated to fit for more points. We can see in figure 9 that the by the third and four iteration, the optimization converges on a value. It is recommended to use a value of at least 3. A value of 5 should work for all purposes and one can see that the fourth and fifth iterations should yield almost exactly the same value of t_0

n_fit_param This is the number of parameters used when fitting intensity distribution to create the fast rotation signal. The options are "0", "2", "5", or "9". The option 0 means that nothing is fit, meaning it is assumed that the data is already the fast rotation signal. The option 0 should be used for Monte Carlo simulations of the fast rotation signal. The option 2 fits only for the amplitude and the decay. The option 5 fits additionally fits for the ω_a, and the option 9 fits also for the CBO frequency. If the option "9" is used, then the CBO frequency is set by the parameter "cbo_freq".

cbo_freq This is the frequency for the CBO which is used when fitting the intensity distribution for the wiggle function which we then divide the intensity by to get the fast rotation signal. This parameter is only relevant when "n_fit_param" is "9" so that a 9 parameter fit is used which fits for the CBO frequency.

freq_step_size This is the resolution of the frequency distributed measured in kHz. There is an underlying resolution of the cosine Fourier transformation itself, however, we can choose the resolution by oversampling the frequency. The underlying resolution depends on how long the cosine Fourier transformation so $t_m - t_s$. It is recommended for optimal outcome to use a resolution of between 1

and 3 kHz. If the resolution is too fine then there will be a significant amount of spectral leakage caused by the windowing of the Fourier transformation. If the resolution is too high then information can be lost between bins in the frequency domain and accuracy decreases.

lower_freq/upper_freq This is the frequency range which is generated when doing the cosine Fourier transformation. It is important that lower_freq $< \omega^-$ and upper_freq $> \omega^+$ because we need information outside the bounds of the collimator aperture for the first iteration of the noise. A lower_freq = 6630 kHz and a upper_freq = 6780 kHz should be sufficient for all purposes.

rebin_wiggle_factor When real data is being used, and the Intensity distribution has to be fit with the wiggle function in order to create the fast rotation signal, the intensity distribution is rebinned by this factor since the normal binning is too fine. If a Monte Carlo fast rotation signal is being used then this parameter can be ignored.

field_index This is the electric field index used in the calculation of the E-field correction. The E-field correction is proportional to n(n-1). This should be matched to whatever experimental data one is interested or for Monte Carlo data just choose a reasonable value. For the 60–hour data set we use a field index of n = 0.108.

start_fit_time This is the time at which the intensity the fitting of the wiggle plot to get from the intensity distribution to the fast rotation signal begins. This is only relevant when real data is being used so the fast rotation signal has to be generated. The first approximately 4μs are effected by positron interference. This can be seen in figure 6 for the 60–hour data set. We also have to consider the effects of scraping which occurs for the first approximately 30 μs. We can avoid both these effects for the wiggle fit by starting the fit after this threshold. If a value of 30 μs is used then both the positron interference and scraping can be skipped.

poly_order This is the order of the polynomial used when fitting the background with a polynomial. This is only relevant when the option background_correction = "fit" and background_fit = "poly." The value of N should be at least 2. For all purposes a value of N = 5 is sufficient. The larger the value of $t_s - t_0$ the higher the polynomial order must be since for large values of $t_s - t_0$ there will be highly nonlinear terms. If the value of $t_s - t_0$ is small of the order of 4 μs then the polynomial order N = 2 is sufficient because the background will be approximately a parabola for mostly symmetric frequency distributions.

6.2 Positron hit parameters

check_positron_hits The options are "true" or "false". If "true" then the number of positrons between the time range of t_s and t_m are counted based on the input intensity histogram. This is used as a check to see how high the statistics are, and if the number of positrons counted are less than the parameter "positron_hits_threshold", then the analysis stops since the statistics are too low. We want to avoid running the analysis on data which have statistics which are too low since the results will not be meaningful. This is meant to be used only for when the input is the muon intensity distribution, not for a Monte Carlo fast rotation signal.

positron_hits_threshold This is the threshold used to stop the analysis if the statistics are too low when the parameter "check_positron_hits" is set to "true". This can be used to make sure we only use data which has statistics high enough to yield meaningful results.

6.3 Fixed bound parameters

fix_fit_bound The options are "true" or "false". If "true" then the background to the cosine Fourier transformation is fit to to the frequency range between "fit_lower_bound" and "fit_lower_bound" instead of finding the appropriate bounds by using the iterative process to find where "t0_background_threshold" sigma away from the mean of the background. It is useful to fix the bounds when running scans in order to not have the eliminate the boundary as a source of systematic uncertainty when scanning over a parameter.

fit_lower_bound/fit_lower_bound These are the lower and upper bounds of the fit used to fit the background of the cosine Fourier transformation when "fix_fit_bound" is set to "true".

6.4 Printing options parameters

print_plot The options are either "true" or "false." If "false" then no graphs are saved. Only ROOT files with the resulting distributions are saved. If "true" then all the graphs are saved as .eps pictures as well as the ROOT files.

calc_sine The options are either "true" or "false." If "true" is selected then the sine transformation is saved as well as the normal cosine Fourier transformation. This is the imaginary part of the Fourier transformation which we do not use in our analysis.

verbose The options are "0", "1", or "2." This controls how much information is outputted in the terminal. With option "0" only the steps and final values are displayed. With option "1" the t_0 optimization is displayed as well, and with option "2" the parameters used for fitting is also displayed.

append_results The options are "true" or "false." If Monte Carlo data is being used then it is useful to compare the recovered and simulated results in order to test the constraints of the method.

compare_with_truth The options are "true" or "false." This can be used if Monte Carlo data is being used with a known frequency distribution.

truth_root_file This is the path to the ROOT file containing the frequency distribution to be compared with the recovered results. The ROOT file should contain the simulated frequency distribution as a TH1D histogram. If you store the ROOT file within the data directory and called is truth.root, then this parameter should be set to "data/truth.root".

truth_hist0_name This is the name of the histogram stored within the ROOT file which should be set in the parameter "truth_root_file". It must be a type TH1D histogram.

6.5 Statistical fluctuation parameters

Refer to section 5.2 for plots of what the scans show.

stat_fluctuation The options are "true" or "false." If "stat_fluctuation" is "true" then "n_stat_fluctuation" number of pseudo data is created. The pseudo data is made by varying each time bin for the intensity distribution of the muons by plus or minus the square root of the number of muons in the time bin. This follows from poison statistics with a large number of muons. The pseudo data will yield a distribution of E-field corrections which will be approximately Gaussian for large amounts of pseudo data experiments. Then some number of standard deviations can be considered the statistical uncertainty of the method.

n_stat_fluctuation This is the number of pseudo data experiments which are done. It is recommended that the value of "n_stat_fluctuation" be at least 100 so that the central limit theorem can be invokes so that the standard deviation calculated from the E-field correction distributoin will be reasonable. This is a very computationally intensive process so large values of "n_stat_fluctuation" will take a long time to complete.

6.6 Scan parameters

Refer to section 5.3 for more information on the scans. It is important to know that multiple scans cannot be run at once. If two different scan parameters are true, then only one of the scans will run.

6.6.1 t_0 scan

fix_t0 The options are "true" or "false." If this is "true" then there will be no optimization for t_0. The background will be fit with the fixed value of t_0.

fixed_t0_value A fixed value of t_0 can be used if the value is already known. This means that no t_0 optimization is done. This value is only relevant if "fix_t0" is "true".

run_t0_scan The options are "true" or "false." A t_0 scan can be done to estimate the systematic uncertainty of using a value of t_0 which differs from the actual value. The t_0 scan will save the recovered results for the range of t_0 values used.

6.6.2 t_s scan

run_tS_scan The options are "true" or "false." A t_s scan can be done to estimate the systematic uncertainty of using a value of t_s which differs from the actual value.

lower_tS/upper_tS This is the lower and upper bounds for values of t_s used. We want to skip the initial positron interference in the fast rotation signal so it is useful to use a value of t_s greater than 4 μs. We can scan with values of t_s less than this though to see how the positron interference can bias our reconstructed frequency distribution. Furthermore, values for greater t_s can be used in order to see the effects of scraping. Note that the "background_fit" of "sinc" breaks down by less than 15 μs and the "poly" breaks down by 8 μs. This can be seen using this scan.

tS_step_size This is the step size for using at t_s scans.

6.6.3 t_m scan

run_tM_scan The options are "true" or "false." The t_m scan can be used to asses the systematic uncertainty in choosing a value of t_m. It can also be used to optimize the tradeoff between better frequency resolution and adding more noise to the signal for a large t_m.

lower_tM/upper_tM This is the bounds of the t_m scan. Using a value for t_m of less than 100 μs is bad because the frequency resolution is small enough where spectral leakage will have a large effect. A value of t_m larger than 400 μs is bad because the added noise begins to significantly harm our recovered frequency distribution. This can be seen using the t_m scan and a optimal value of t_m will be somewhere within the range for any given data set.

tM_step_size This is the step size for using at t_m scans.

6.6.4 Frequency step scan

run_freq_step_scan The options are "true" or "false." This can estimate the systematic uncertainty in choosing the resolution of the frequency distribution. There is an intrinsic resolution of the frequency distribution which comes from the resolution of the fast rotation signal. When we take the Fourier transformation of the fast rotation signal we can oversample or undersample it in order to choose the bin size of the frequency that we want. This can lead to spectral leakage if bin sizes are too small. If bin sizes are too large, however, then some of the information can be lost between frequency bins. There is a good middle ground between 1 and 3 kHz bin size where neither of these problems are very bad so long as $150\mu\text{s} \leq t_m \leq 400\mu\text{s}$. This scan can be used to find the optimal frequency bin sizes.

lower_freq_step_size/upper_freq_step_size This is the upper and lower bound in frequency bin sizes. The frequency distribution should be mostly stable between 1 and 3 kHz, but outside of this range the method will suffer.

freq_step_size_increment This is the step size in frequency binning.

6.6.5 Background threshold scan

run_background_threshold_scan The options are "true" or "false." This scans the threshold used in the fitting of the frequency background. This threshold is used in determining which points of the background are considered to be background noise and which are signal. Only the points considered to be background noise are used to in the fitting of the background and the points considered to be noise are found iteratively by finding a new standard deviation of the noise each iteration and then all points with values within this threshold of the mean are considered noise. This iterative process begins by only considering the points outside the collimator aperture to be noise and then working its way in with each iterations. This scan can measure the systematic uncertainty in choosing which points are part of the background for purposes of doing the background fit.

lower_background_threshold/upper_background_threshold These are the bounds of the uncertainty threshold. There is a tradeoff in if the threshold is too low then less points are used for the background fit which will lower the chi squared of the fit, but if the threshold is too high then some of the points used in the background fit will actually contain part of the signal which will also lower the chi squared of the fit. Bounds between 2 and 3 sigma should be acceptable, but points near the edge of the signal will always be questionable if they contain the signal or are part of the background. This scan can help to figure that out.

background_threshold_step_size This is the step size in the run_background_threshold_scan in sigma.

6.6.6 Remove background threshold scan

Refer to section 5.1 for more information on the background threshold.

run_background_removal_threshold_scan The options are "true" or "false." This allows the user to test the systematic effects associated with removing the noise background. It can be tested where the optimal removal threshold for a given data set. There is no clear cutoff where actual data begins and noise ends so this is a way to asses if actual data is being eliminated along with the noise. The background removal threshold scan is only relevant if the parameter "remove_background" is "true" since otherwise the background will not be removed in the scan.

lower_background_removal_threshold/upper_background_removal_threshold This is the start and end of the scan measured in standard deviations of the noise. The standard deviation of the noise is calculated using the "t0_background_threshold" to determine which points are noise and which are data, but then this threshold eliminates all data less than the "remove_background_threshold." A range of values up to 5 sigma can be used, however, it is likely that using a threshold greater than 3 sigma may eliminate some actual data.

background_removal_threshold_step_size This is the step size for the background_removal_threshold_scan in sigma.

References

[1] See 'TMC #1' in: GM2-doc-13759

[2] A. Chapelain, J. Fagin, D. Rubin, D. Seleznev, *Cornell fast rotation Fourier method*, GM2-doc-18901

[3] Y. Orlov et al., NIM A 482 (2002) 767-755.

[4] A. Chapelain, D. Rubin, D. Seleznev, *Extraction of the Muon Beam Frequency Distribution via the Fourier Analysis of the Fast Rotation Signal*, E989 note 130, GM2-doc-9701

[5] A. Chapelain, J. Fagin, D. Rubin, D. Seleznev, *On the background correction of the Cornell fast rotation Fourier method*, GM2-doc-19225-v1