G-2 Internal
Version 1.0
November 25, 2019

Cornell fast rotation Monte Carlo method and user guide

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

Abstract

We derive a mathematical description of the fast rotation signal and its exponentially increasing additive noise in order to Monte Carlo fast rotation signals. We propose using the sum of skew Gaussian frequency distributions and longitudinal beam profiles with randomized parameters as inputs into the fast rotation signal. From the fast rotation signal we can use the Fourier method to recover the frequency distribution used to generate it. An ensemble testing of fast rotation signals is used to estimate the systematic uncertainty of the Fourier method. We find a conservative uncertainty of 35.96 ppb. We also show how the user can download and use the code to start generating their own Monte Carlo fast rotation signals including a description of all the possible parameters to configure the Monte Carlo code.

Contents

1 Introduction

The goal of the Fermilab g-2 experiment is to measure the magnetic moment of the muon which depends on the precession frequency of the muons as they spin about their momentum. We must account for the contribution of the electric field on the precession frequency by determining the radial distribution of the muons orbit. The radial distribution of the muons can be found directly from the frequency distribution because the muons are going around the ring at near the speed of light, which makes their velocities approximately constant. The frequency distribution of the muons can be found by taking the real part of the Fourier transformation of the fast rotation signal, where the fast rotation signal is the evolution of the intensity of the muon beam as it goes around the ring.

The E-field correction depends on the mean and standard deviation of the radial distribution. Our final goal is to recover the E-field correction within 20 ppb. We want to Monte Carlo fast rotation signals from known frequency distributions so that we can compare the known frequency distribution to our recovered frequency distribution in order to evaluate how well the Fourier method [1] performs. The E-field correction is found from both frequency distributions so that the simulated and recovered E-field correction can be compared.

The analytic form of the fast rotation signal is derived in [2]. We rederive the analytic form of the fast rotation signal given a frequency distribution and longitudinal beam profile and show how we can restrict the frequency distribution to be within the collimator aperture which allows us to numerically compute the fast rotation signal from its numerical form. We choose the form of the frequency distribution and longitudinal beam profile to be the sum of skew-Gaussians with randomized parameters to represent somewhat realistic frequency distributions and longitudinal beam profiles which are used to create the fast rotation signal. We also derive the form of the noise on the fast rotation signal and incorporate it into the Monte Carlo in order to better match real data.

With the Monte Carlo we are able to generate a large ensemble of fast rotation signals. We run the ensemble on the Fourier method and compare our recovered radial distribution to distributions used to generate the Monte Carlo for each fast rotation signal in the ensemble. We then use the compared statistics to estimate the systematic uncertainty in the Fourier method.

2 Mathematical Description

2.1 General Formulation of fast rotation signal

Here we will derive the general formulation of the fast rotation signal similar to [2]. The muons all share the same period of oscillation around the ring T=149.1 ns. The simplest description we can make of the intensity of the muons is to assume the initial muon distribution is a delta function with no momentum spread of beam profile. Then we get an intensity function which is a delta function at each period of the oscillation. This corresponds to a "Dirac comb" or "Bed of nails" function defined only for positive time like the following:

$$I(t) = \sum_{n=0}^{\infty} \delta\big(t - nT\big), \tag{1}$$

where n corresponds to the number of completed turns the muons travel around the ring and T is the period of the oscillation. We can give the muons a fractional energy offset of Δ so that each muon has a momentum of p(1+Δ). The muons are going at nearly the speed of light around the ring so they are in the highly relativistic realm. Therefore, the period of oscillation is directly proportional to the momentum of the muons in this relativistic limit since the muons speed is approximately independent from its momentum so dv/dp ≈ 0. The period of the particles with fractional energy offset is then $T(1+\Delta)$. If we give all the muons this fractional energy offset then we get a muon distribution as follows:

$$I(t,\Delta) = \sum_{n=0}^{\infty} \delta\Big(t - nT(1+\Delta)\Big) \tag{2}$$

If we also then give each particle a temporal offset of $t\prime$ in addition to the fractional energy offset then we get our distribution to become:

$$I(t,\Delta,t\prime) = \sum_{n=0}^{\infty} \delta\Big(t - t\prime - nT(1+\Delta)\Big). \tag{3}$$

We now want to consider giving the muons a momentum distribution instead of a constant fractional energy offset. We can consider a momentum distribution as a function of the fractional energy offset denoted $\rho(\Delta)$ and an initial temporal offset denoted $\xi(t)$. We get a fast rotation signal by summing over the distributions and the intensity:

$$
\begin{aligned}
S(t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} & I(t,\Delta,t\prime)\rho(\Delta)\xi(t\prime)dt\prime d\Delta \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\sum_{n=0}^{\infty}\delta\Big(t-t\prime-nT(1+\Delta)\Big)\rho(\Delta)\xi(t\prime)dt\prime d\Delta \\
&= \sum_{n=0}^{\infty}\int_{-\infty}^{\infty}\xi(t\prime)dt\prime\int_{-\infty}^{\infty}\rho(\Delta)\frac{\delta\Big(\frac{t-t\prime}{nT}-1\Big)}{nT}d\Delta \\
&= \sum_{n=0}^{\infty}\int_{-\infty}^{\infty}\xi(t\prime)\frac{\rho(\frac{t-t\prime}{nT}-1)}{nT}dt\prime,
\end{aligned}
\tag{4}
$$

where $\rho(\Delta)$ corresponds to the frequency distribution of the muons while $\xi(t)$ is the longitudinal beam profile. We need one final step where we add a delay time to the muons since the muons' beam does not enter the ring at exactly t = 0, but instead at some unknown later time which we denote as t_0. This delay time changes nT to $nT+t_0$ which gives us a form of the fast rotation signal of:

$$S(t) = \sum_{n=0}^{\infty}\int_{-\infty}^{\infty}\xi(t\prime)\frac{\rho(\frac{t-t\prime}{nT+t_0}-1)}{nT+t_0}dt\prime, \tag{5}$$

where t_0 is the time it takes for the center of mass of the muon beam to first cross the detector in the ring.

2.2 Confining the Frequency Distribution

We have derived the formula for the Fast Rotation Signal S(t) which is a function of frequency distribution $\rho(\Delta)$ and the longitudinal beam profile $\xi(t)$. We know that physically all the muons must be inside the ring which confines the frequency distribution to within the bounds of the collimator aperture. Therefore we can say that the frequency distribution is zero outside the collimator aperture which occurs at a fractional energy offset of Δ_c:

$$\rho(\Delta) = \begin{cases} \rho(\Delta) & |\Delta| \le \Delta_c \\ 0 & |\Delta| > \Delta_c \end{cases} \tag{6}$$

The collimator aperture has bounds of $\omega^- = 6.6628$ and $\omega^+ = 6.7477$ MHz. We solve for Δ_c simply by converting the bounds of the collimator aperture to a fractional energy offset:

$$\Delta_c = T\omega^+ - 1 = (.14912\mu\text{s})(6.7477\text{ MHz}) - 1 = 0.006217 = 0.6217\%. \tag{7}$$

We can redefine the fast rotation signal such that we only have to use points where the frequency distribution is non-zero inside the collimator aperture. This occurs when the argument of ρ is in the range of the collimator aperture, so the integral in S(t) is 0 except when:

$$|\frac{t-t\prime}{nT+t_0}-1| \leq \Delta_c. \tag{8}$$

This takes place only if:

$$t-(1+\Delta_c)(nT+t_0) \leq t\prime \leq t-(1-\Delta_c)(nT+t_0), \tag{9}$$

so we can replace the bounds of the integral to be within this range. Furthermore, in the infinite summation of n, only the n which occur inside the bounds Δ_c need to be included, since anything outside this bound will be zero. We can include only the sum from n=$\lfloor t\omega^- \rfloor$ to n=$\lceil t\omega^+ \rceil$ where we take the floor and the ceiling respectively in order to include only the closest integer with non-neglible contribution to the fast rotation signal. We can then replace our expression for the fast rotation signal with:

$$S(t)=\sum_{n=0}^{\infty}\int_{-\infty}^{\infty}\xi(t\prime)\frac{\rho(\frac{t-t\prime}{nT+t_0}-1)}{nT+t_0}dt\prime=\sum_{n=\lfloor t\omega^-\rfloor}^{\lceil t\omega^+\rceil}\int_{t-(1+\Delta_c)(nT+t_0)}^{t-(1-\Delta_c)(nT+t_0)}\xi(t\prime)\frac{\rho(\frac{t-t\prime}{nT+t_0}-1)}{nT+t_0}dt\prime. \tag{10}$$

We must do this since otherwise we would not be able to Monte Carlo the fast rotation signal because we cannot compute an infinite integral an infinite number of times. This new formulation of the fast rotation signal now only allows physical frequencies to be included in the calculation.

2.3 Modeling the Noise of the fast rotation signal

The muons decay while they go around the ring. This leads to an exponentially modified muon intensity signal. When actual data is used the muons intensity signal is fit for the exponential decay (among other parameters) and then divided by the decay to get the fast rotation signal. We may also assume that the intensity of the muons has additive Gaussian white noise. This can be explained by other particles passing through the detector at random intervals, and in the limit of a large number of particles (which we definitely meet with at least 10^8 muons), we can invoke the central limit theorem to assume that the noise is white and Gaussian. We may also assume that it is additive because any random particle that passed through the detector would only add to the intensity, so there could be no multiplicative noise. We can assume an exponentially decaying intensity signal with additive noise of the form:

$$I(t)=S(t)e^{-t/\tau}+n(t), \tag{11}$$

where τ is the decay time of the muons, I(t) is the intensity distribution of the muons, S(t) is the fast rotation signal, and n(t) is the additive Gaussian white noise. The muons have a momentum at their magic frequency, the center of their orbit at 6.705 MHz, 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 that by the gamma factor of:

$$\tau=\tau_0\sqrt{1+\frac{p_0^2}{m_\mu^2}}, \tag{12}$$

because of the time dilation of the highly relativistic muon beam. We get a muon decay time of $\tau = 64.46\ \mu$s. We solve for the fast rotation signal as a function of the intensity using the exponential decay rate as follows:

$$S(t) = I(t)e^{t/\tau} - e^{t/\tau}n(t). \tag{13}$$

The fast rotation signal without noise which we denote as $S\prime$ is:

$$S\prime(t) = I(t)e^{t/\tau}. \tag{14}$$

which is the intensity distribution corrected for the exponential decay of the muons. The overall sign of the noise does not matter so we can add the noise instead of subtracting it. We rewrite the fast rotation signal with noise in terms of the signal without noise like:

$$S(t) = S\prime(t) + e^{t/\tau}n(t), \tag{15}$$

so we have concluded that the fast rotation signal has additive exponentially growing Gaussian noise with a growth factor equivalent to the decay factor of the muons.

3 Monte Carlo

3.1 Frequency Distribution and Beam Profile

We have derived the equation for the fast rotation signal and its noise for a given frequency distribution, longitudinal beam profile, and phase t_0. We will now describe our method for creating randomized frequency distribution and beam profiles. We know that the frequency distribution should be approximately Gaussian centered at the magic frequency except for a small amount of asymmetry. Therefore we want a skewed Gaussian distribution. If we want to add even more complicated features to the frequency then we can sum over multiple skewed gaussians so long as the center of the distributions are close together so that there is only one peak. We can use the following equation for the frequency distribution which is the sum of modified Gaussian distribution:

$$\rho(\omega) = \sum_{i=1}^{3} A_i e^{-\frac{(\omega-\omega_i)^2}{2\sigma_i^2}} \operatorname{erfc}(\frac{\alpha_i(\omega-\omega_i)}{\sigma_i} - 1), \tag{16}$$

where the amplitude of each skewed Gaussian is A_i, mean ω_i, standard deviation σ_i and skew α_i.The normalization of the frequency distribution is arbitrary because we only care about the overall normalization of the final fast rotation signal, and the erfc function is defined by:

$$\operatorname{erfc}(x) = 1 - \operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} e^{-y^2} dy. \tag{17}$$

We must also specify a longitudinal beam profile. Here we use the sum of three of the skewed Gaussian distributions. Unlike with the frequency distribution the longitudinal beam profile can have multiple peaks since real data has a complex beam profile.

$$\xi(t) = \sum_{j=1}^{3} \tilde{A}_j e^{-\frac{(t-\tilde{t}_j)^2}{2\tilde{\sigma}_j^2}} \mathrm{erfc}(\frac{\tilde{\alpha}_j(t-\tilde{t}_j)}{\tilde{\sigma}_j} - 1) \tag{18}$$

We then add the equation for the frequency distribution and longitudinal beam profile into the equation for the fast rotation signal to get:

$$S(t) = \sum_{n=\lfloor t\omega^- \rfloor}^{\lceil t\omega^+ \rceil} \int_{t-(1+\Delta_c)(nT+t_0)}^{t-(1-\Delta_c)(nT+t_0)} \xi(t\prime) \frac{\rho(\frac{t-t\prime}{nT+t_0} - 1)}{nT + t_0} dt\prime$$

$$= \sum_{n=\lfloor t\omega^- \rfloor}^{\lceil t\omega^+ \rceil} \int_{t-(1+\Delta_c)(nT+t_0)}^{t-(1-\Delta_c)(nT+t_0)} \sum_{i=1,j=1}^{3} A_i \tilde{A}_j \big(e^{-\frac{(t\prime-\tilde{t}_j)^2}{2\tilde{\sigma}_j^2}} \mathrm{erfc}(\frac{\tilde{\alpha}_j(t\prime-\tilde{t}_j)}{\tilde{\sigma}_j} - 1)\big)\big(\frac{e^{-\frac{((\frac{t-t\prime}{nT+t_0}-1)-\Delta_i)^2}{2\sigma_i^2}} \mathrm{erfc}(\frac{\alpha_i((\frac{t-t\prime}{nT+t_0}-1)-\Delta_i)}{\sigma_i} - 1)}{nT + t_0}\big) dt\prime, \tag{19}$$

where we have to specify the parameters, $\tilde{A}_j, \tilde{t}_j, \tilde{\alpha}_j, \tilde{\sigma}_j$ for the longitudinal beam profile, $A_i, \Delta_i, \alpha_i, \sigma_i$ for the frequency distribution, and t_0 for the phase. Note that we write the frequency distribution as a function of the fractional energy offset so ω_0 corresponds to Δ_0.

3.2 Signal to Noise Ratio

We have seen in section 2.3 that the noise of the fast rotation signal will be exponentially growing by a growth factor of $\tau = 64.46$ μs. We are interested in generating the fast rotation signal from t = 0 μs to a time of t = t_m where we choose t_m to be how long we want to use for the Fourier analysis. For real data sets we do get data for times greater than 400 μs, but because of the exponentially increasing noise we only use the first 200-300 μs of data for the Fourier method. If we were to include data at greater time, the radial distribution of the muons is reconstructed less accurately because the huge signal to noise ratio at late times makes it so we would just be adding noise to the distribution.

A conservative range for the signal to noise ration is to choose uniformly between 900 and 1,000 which is slightly more than the noise amplitude we see in real data. The amplitude of the noise is found from the signal to noise ratio by taking the maximum value of the fast rotation signal minus the mean value, which is normalized to one, and then divide this amplitude by signal to noise ratio to get the noise amplitude. The maximum of the fast rotation signal is found after the first 4 μs since for real data the first 4 μs are effected by positron contamination, so we find the amplitude of the noise by:

$$A_{noise} = \frac{\max\{\mathrm{S(t)}\}}{S/N}, \tag{20}$$

where A_{noise} is the amplitude of the noise, S(t) is the fast rotation signal, and S/N is the signal to noise ration. For Gaussian white noise we use $\sigma = \frac{1}{3}A_{noise}$ as the standard deviation of the Gaussian noise with a mean of μ=0 so that the noise amplitude is reached when we get a value of 3σ away from the mean. The entire noise distribution is then multiplied by the exponential growth $e^{\frac{t}{\tau}}$, so for Gaussian noise we get a noise function of:

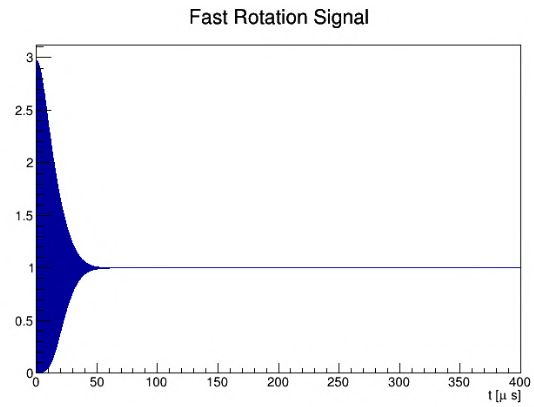
$$n(t) = e^{\frac{t}{\tau}} N(\mu = 0, \sigma = \frac{1}{3}A_{noise}), \tag{21}$$

where N is the normal distribution with mean μ and standard deviation σ. We assume because of the central limit theorem that the noise is Gaussian. We also consider uniform noise since we cannot say for certain what the exact form of the noise is, so it is useful to check that the frequency distribution can still be recovered with another model of the noise. The recovered frequency distribution should be independent of any noise we introduce. Assuming uniform noise, we get a distribution as follows where U is the uniform distribution between $-A_{noise}$ and A_{noise}:

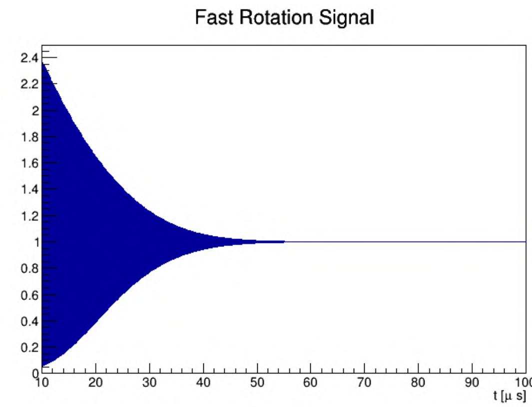
$$n(t) = e^{\frac{t}{\tau}} U(-A_{noise}, A_{noise}). \tag{22}$$

3.3 Example Gaussian frequency and beam profile fast rotation signal no noise

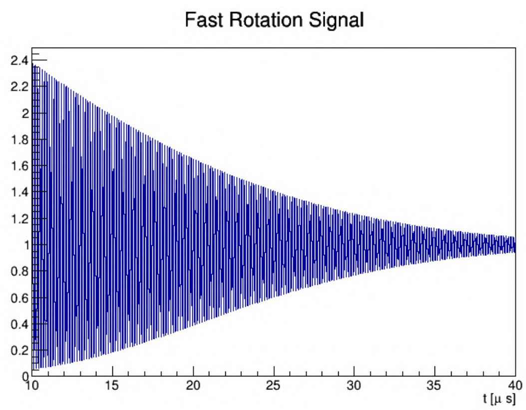
In figure 1 we show a fast rotation signal generated using a Monte Carlo simulation with the Gaussian frequency distribution and longitudinal beam profile shown in figure 2. The frequency distribution is centered at the magic frequency with a fractional energy offset of 0.15% and the longitudinal beam profile is centered at 0 with a standard deviation of 25 ns. There is no noise included in the signal, so the beam profile remains pristine even after 100 μs.



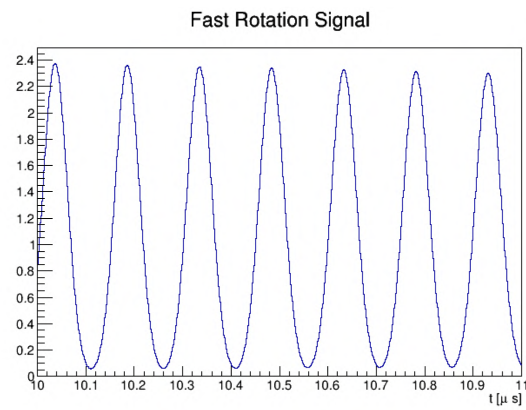
(a)



(b)

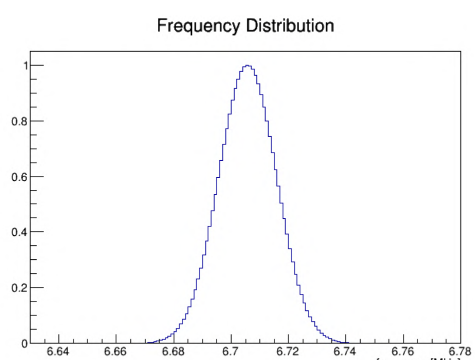


(c)

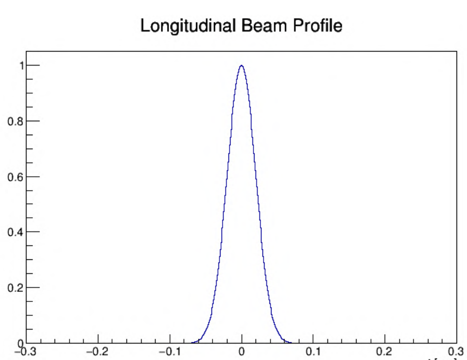


(d)

Figure 1: A fast rotation signal generated using a Monte Carlo simulation with Gaussian frequency distribution and longitudinal beam profile. The Frequency distribution is centered at the magic frequency with a fractional energy offset of 0.15% and the longitudinal beam profile is centered at 0 with a standard deviation of 25 ns shown in figure 2. Four time intervals are shown: (a) 0-400 μs, (b) 10-100 μs, (c)10-40 μs, (d) 10-11 μs.



(a)



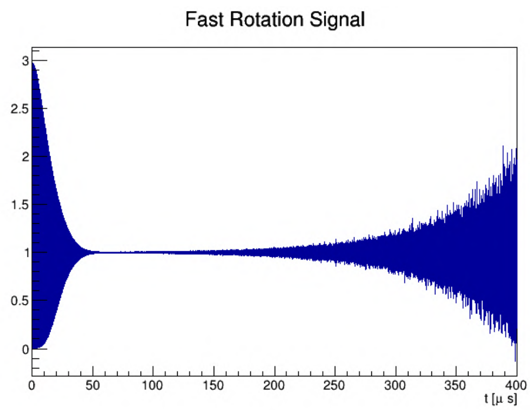
(b)

Figure 2: The frequency distribution and longitudinal beam profile used to generate 1: (a) frequency, (b) longitude.

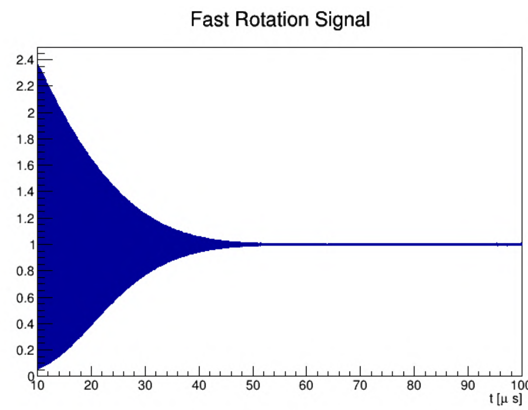
3.4 Example Gaussian frequency and beam profile fast rotation signal with noise

Shown in figure 3 is the same fast rotation signal from figure 1, but with exponentially increasing additive noise. For a comparison with real data, in figure 5 we show the fast rotation signal from 4 and 400 μs for the 60-hour data set, and we can see that the noise is also exponentially increasing with time.

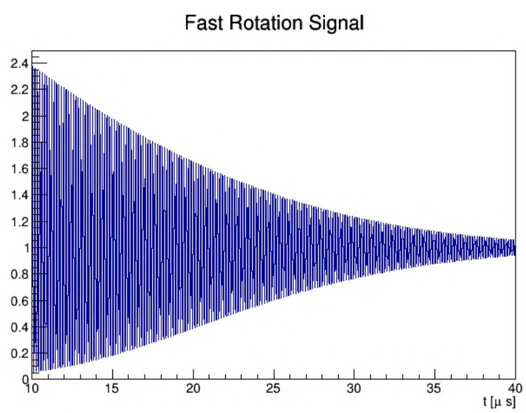
In figure 4 we zoom in on the Monte Carlo fast rotation signal at different time intervals to see the progression of the noise. At 100 μs the noise has already dominated the actual fast rotation signal which is what we want since the muons have already completely debunched around the ring at this point. For later times the amplitude of the noise increases but the form of the noise remains the same.



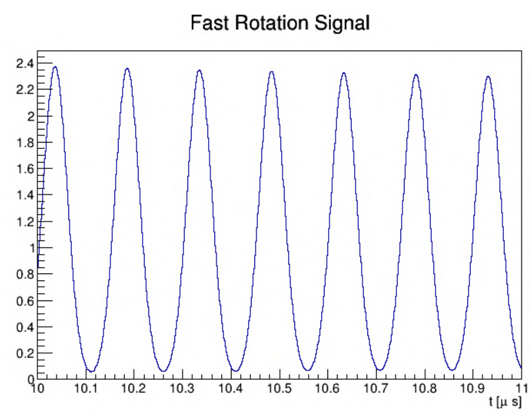
(a)



(b)

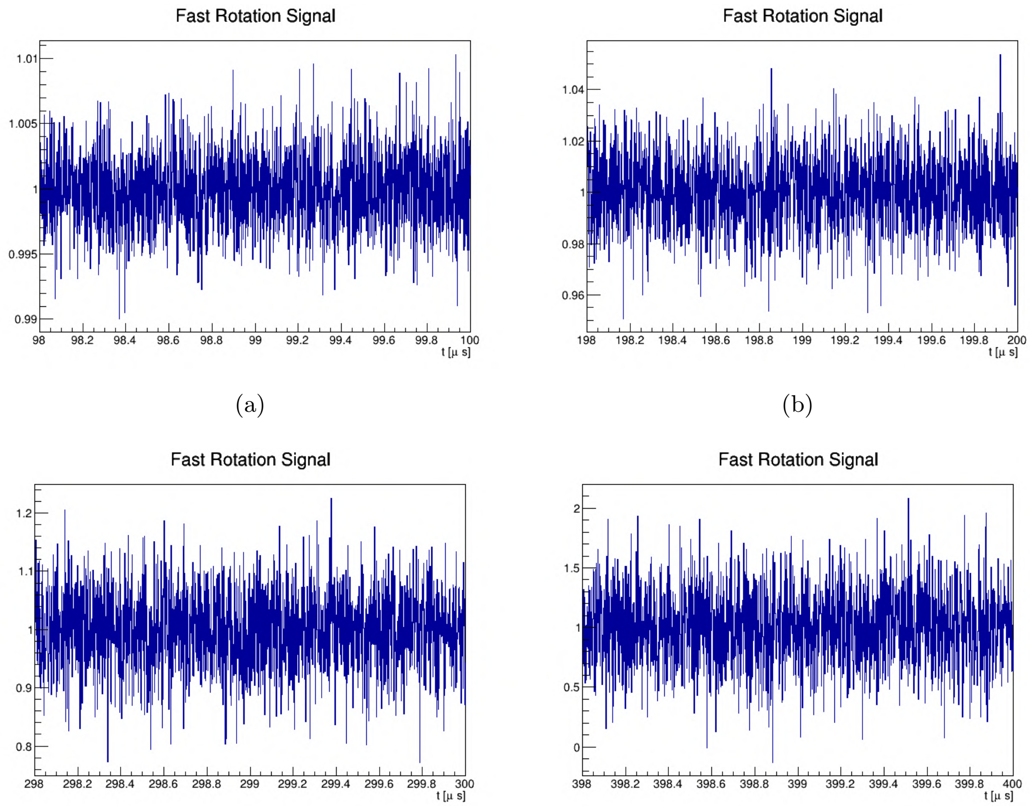


(c)



(d)

Figure 3: A fast rotation signal with exponentially increasing Gaussian additive noise generated using a Monte Carlo simulation with Gaussian frequency distribution and longitudinal beam profile. The frequency distribution is centered at the magic frequency with a fractional energy offset of 0.15% and the longitudinal beam profile is centered at 0 with a standard deviation of 25 ns shown in figure 2. Four time intervals are shown: (a) 0-400 μs, (b) 10-100 μs, (c) 10-40 μs, (d) 10-11 μs.



(a) (b)

(c) (d)

Figure 4: Zooming in on the fast rotation signal at different time intervals to see the progression of the noise. Notice the change in the y axis. By 100 μs the noise completely dominates the signal. Four time intervals are shown: (a) 98-100 μs, (b) 198-200 μs, (c) 298-300 μs, (d) 398-400 μs.

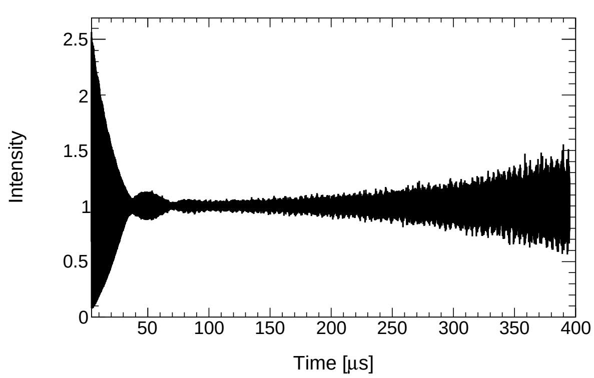


Figure 5: The fast rotation signal from the Run-1 60-hour data set between 4 and 400 μs.

3.5 Example complex frequency and beam profile

In equation (19) we showed the form of the fast rotation signal for a frequency distribution and longitudinal beam profile which are the sum of three skew-Gaussians. With three skew-Gaussians we are able to capture complex behavior without being totally unrealistic. We want to include frequency distributions which are highly asymmetric, yet do not have multiple peaks. The longitudinal beam profiles can have multiple peaks since we are able to reconstruct the frequency distributions, even for highly complex longitudinal beam profiles.

We want to test the Fourier method over a full ensemble of frequency distributions and longitudinal beam profiles so that we include all the possible realistic distributions. In figure 6 we show the reconstructed frequency distribution and the fast rotation signal between 4 and 5 μs for the Run-1 60-hour data set. This is real data so we must view the longitudinal beam profile by looking at the fast rotation signal directly. We can see that both the frequency distribution and the longitudinal beam profile are asymmetric. When we use the sum of three skew-Gaussians for the distributions we are able to capture this asymmetry.

We also want to include some distributions which are considerably worse than real data so that we can gauge the worst possible scenario of the Fourier method to get an upper bound on our ability to reconstruct the E-field correction. In figures 7 and 8 we show a Monte Carlo fast rotation signal with a frequency distribution and longitudinal beam profile generated using a sum of three skew-Gaussians. The frequency distribution has complicated asymmetries and the longitudinal beam profile has multiple peaks.

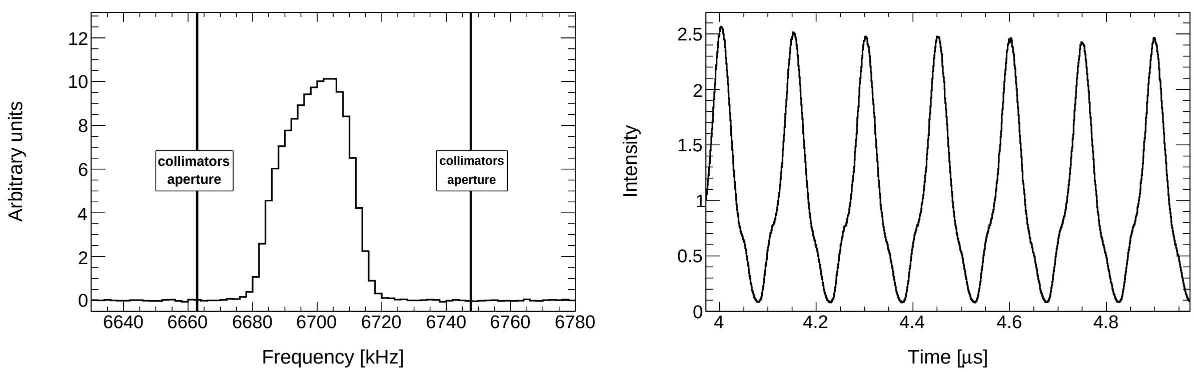


Figure 6: The frequency distribution and longitudinal beam profile for the Run-1 60-hour data set: (a) frequency, (b) longitude.

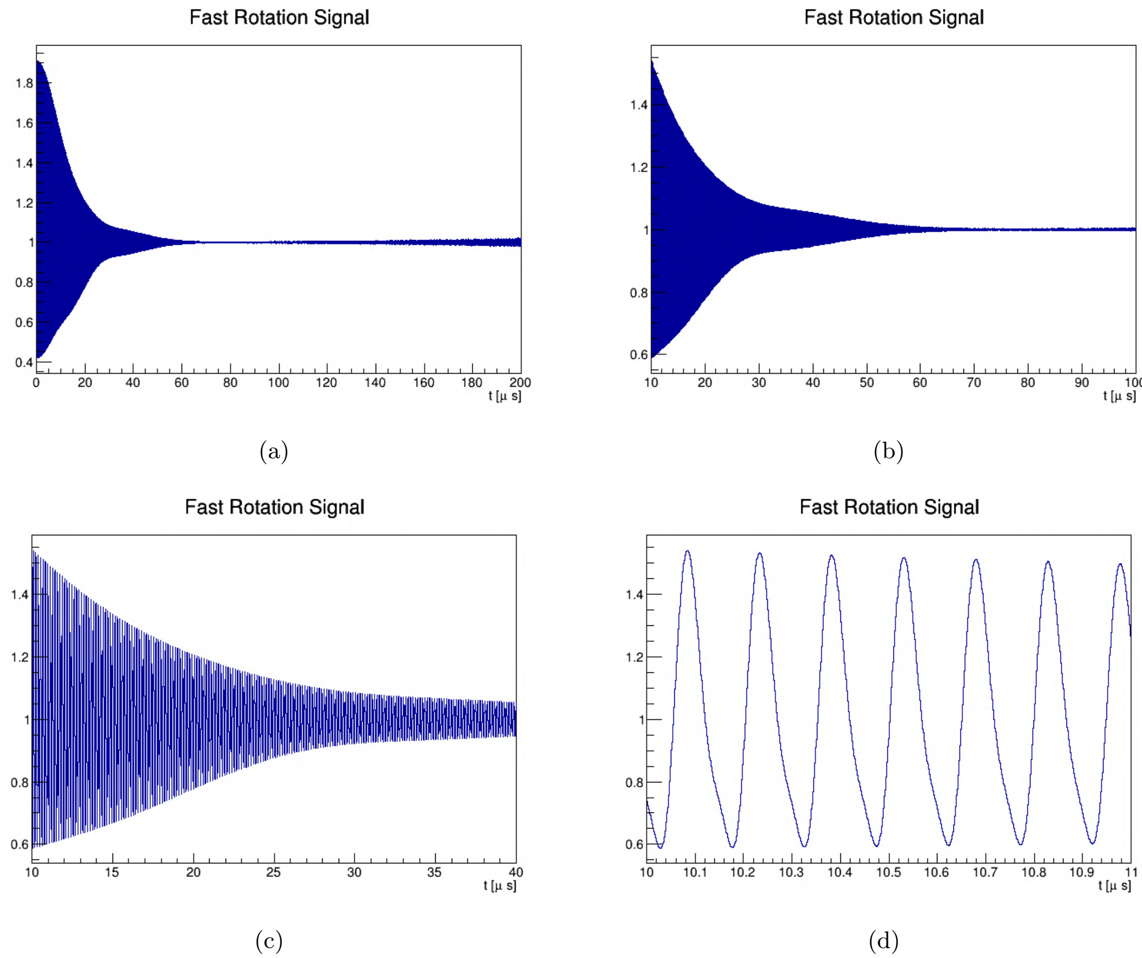


Figure 7: A Monte Carlo fast rotation signal with a frequency distribution and longitudinal beam profile generated using the sum of three skew-Gaussians. Four time intervals are shown: (a) 0-400 μs, (b) 10-100 μs, (c) 10-40 μs, (d) 10-11 μs.

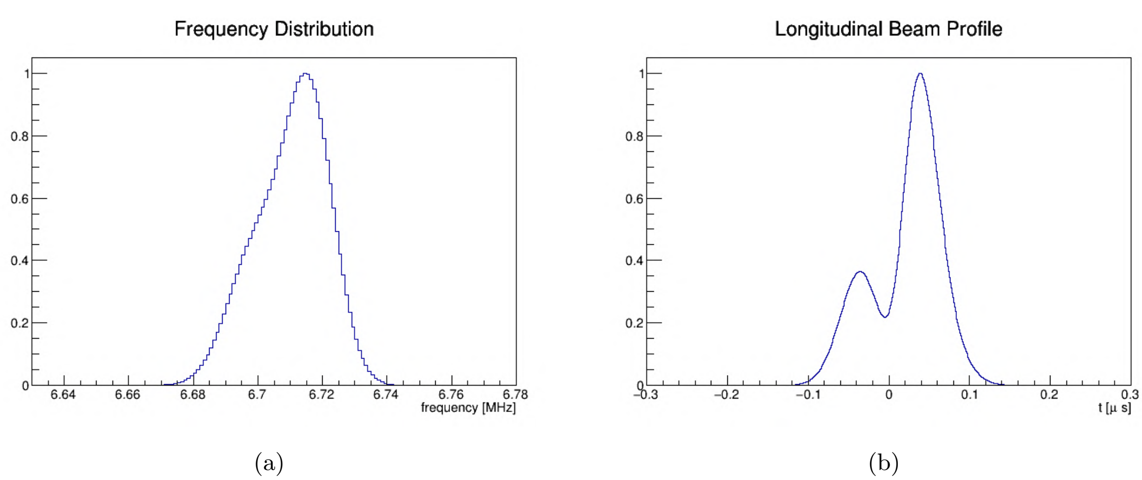
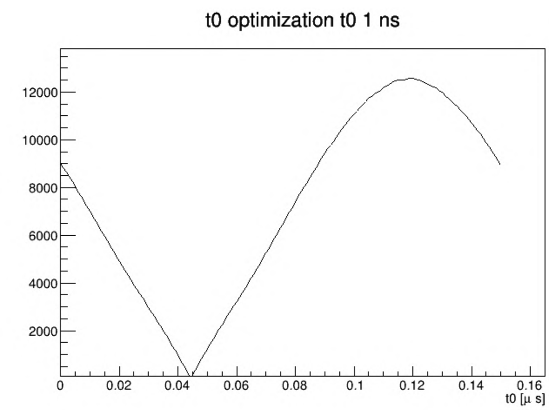


Figure 8: The frequency distribution and longitudinal beam profile used to generate 7: (a) frequency, (b) longitude.

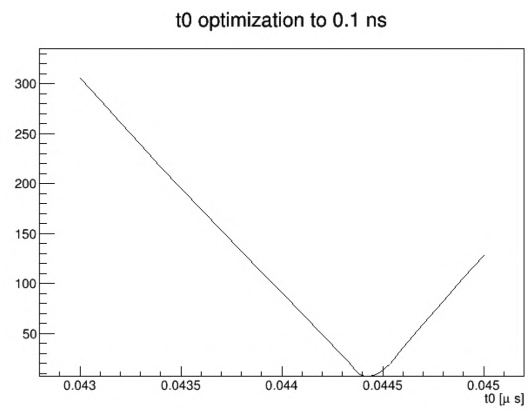
3.6 Real t_0 optimization

When we generate the fast rotation signal, we choose a value of time offset of the muon beam. We also choose a frequency distribution and longitudinal beam profile, however, the mean of these distributions affect when the mean of the muon beam first reaches the detector which is the real value of t_0. The fast rotation method extracts t_0 by fitting the background of the cosine Fourier transformation for different values of t_0 and minimizing the chi squared between the background and the fit function [1]. To do this chi squared minimization we first must know t_0 to within only a few nano seconds. We want to run Monte Carlo using the fast rotation Fourier method [3] so what we do is first find and approximate value of t_0 and then run the Fourier method within 2 ns of it to optimize the real value of t_0.

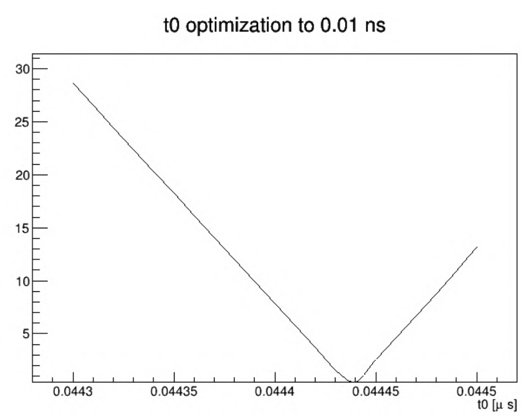
Our method for approximating t_0 is to find the value of t_0 which has the smallest difference in the minima of the cosine Fourier transformation. We found the minima since there will always be one minima on each side of the maxima (the main peak) of the cosine Fourier transformation. This works when t_s is small and the background to the cosine Fourier transformation has not merged with the signal. When t_s is large there is no way of distinguishing between the background and the signal [5].



(a)



(b)



(c)

Figure 9: Showing the approximate method for optimizing t_0 by matching the minima of the cosine Fourier transformation. Three optimizations are done to find t_0 to a higher accuracy. Optimize t_0 to: (a) 1 ns, (b) 0.1 ns, (c) 0.01 ns.

3.7 Momentum time correlation

In our model of the fast rotation signal we assume that the frequency distribution and the longitudinal beam profile are completely independent of one another. We cannot make this assumption for real data since we expect the tail of the muon beam will be at higher momentum than the head of the beam. We expect this phenomenon to yield a negative momentum time correlation and since momentum and frequency are inversely proportional, we expect a positive momentum time correlation.

The form of the momentum time correlation for real data is unknown, but we expect it to be approximately a third degree polynomial. When we add this correlation, the mean of the frequency distribution is shifted with the longitudinal beam profile. The true frequency distribution is then the sum of all the frequency distributions shifted by the position within the longitudinal beam profile with the amplitude determined by the longitudinal beam profile at that point. When saving the frequency distribution with linear momentum time correlation, we use the true frequency distribution instead of ρ.

The momentum time correlation also changes the symmetry of the fast rotation signal. When a momentum time correlation is introduced, we no longer expect the correct cosine transform to begin in the first turn, because the cosine transform should start when the muon beam is maximally debunched. We can fit the envelope of the fast rotation signal with the sum of three Gaussians and use the max value of the fit function as the approximate symmetry point of the fast rotation signal. We typically want to start this fit after 4 μs so that we mimic the beam-line positron contamination of the muon beam which we have in real data sets, forcing us to discard the first 4 μs of data. We also extend the fit into the negative time domain since a negative linear momentum time correlation, for example, will be maximally debunched in the negative time domain.

For more details on the momentum time correlation and its implementation in Monte Carlo see [6].

4 Source code

4.1 Prerequisites for running the code

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

- PyROOT
- NumPy
- SciPy
- Matplotlib
- Numba (Optional)

The package Numba is optional because the parameter "use_numba" can be set to "false" and the code will not use the package. Numba is not a commonly used python package which is why it can be turned off if the user does not have it installed and does not want to download it. The installation page can be found here.

It can be dowloaded easily however using pip by the following command:

```
pip install numba
```

Or using Anaconda by the following command:

```
conda install numba
```

Numba is important because it makes the Monte Carlo code run about ten times faster, but it is nonessential since it does not effect the output in any way.

4.2 Downloading the code

The Monte Carlo code is available for download on `GitHub` (here) and therefore the use of Git is highly recommended. `GitHub` must be installed prior to downloading the code. It can be directly dowloaded by the following terminal command:

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

4.3 Running the default Monte Carlo

Once the code has been downloaded, it can be run by the command:

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

The "-b" in the command is so that ROOT does not slow down the code by displaying graphs on the screen while the code is running. The code will open the config folder and use the parameters defined in the file `config.json`. The `config.json` file is initially set up to generate a Monte Carlo distribution with a Gaussian frequency centered at the magic frequency and a Gaussian longitudinal beam profile centered at 0 μs.

4.4 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 J. Fagin.

4.5 Output

If the parameter "put_in_directory" is set to "true" then the output of the Monte Carlo will all be put in a directory which is named by the parameter "name_directory". If "put_in_directory" is set to "false" then the output will go where the code was run from. It is important to note that if "name_directory" already exists, then it will be deleted and replaced with the new Monte Carlo.

A directory will always be created which contains the ROOT files `root_files`. Three TH1D histograms are stored within each ROOT file. The fast rotation signal stored under "frs", the frequency distribution stored under "freq" and the longitudinal beam profile stored under "longitude". In addition the radial distribution is stored as a TGraph under the name "radius".

When either the parameters "print_graphs" or "save_result_text" are set to "true" then a directory called `Plots_Results` will be created. This directory will contain subdirectories for each fast rotation signal called

Dist and the distribution number. The distribution number will begin after the parameter "start_num". If "print_graphs" is "true" then plots will be printed of the frequency distribution, longitudinal beam profile, radial distribution, and various different time ranges of the fast rotation signal. If the parameter "add_noise" is "true" then plots will be saved which zoom in close to the fast rotation signal at various points in order to show the progression of the noise which will be exponentially increasing if the parameter "exponential_increasing_noise" is "true". If the parameter "save_result_text" is set to "true", then a text file called `value.txt` will be created. This text file will store all the information used to generate the Monte Carlo. This includes the E-field correction, equilibrium radius, the standard deviation, the signal to noise ratio, and t0. Note that the equilibrium radius is measured in beam coordinates so distance from the magic radius. The height, center, standard deviation, and skew is also saved for each skewed Gaussian used to make the frequency distribution and longitudinal beam profile.

If the parameter "create_Fourier_config" is set to "true" then a `Fourier_config` directory will be created which will contain config files set up to run on the fast rotation Fourier analysis code [3]. The config file is controlled by using the parameters defined in 7.8. It is important that the parameter "real_t0_optimization" is set to "true" so that the value of t0 is approximated which reflects the mean of the frequency and longitudinal beam profile. Refer to section 3.6 for information on the t_0 optimization. The config file will then be created called `config_Fourier1.json` but with whatever number the distribution it is on starting after "start_num". Three plots will also be put in the `Dist` directory showing the optimization of t_0 first to the 1 ns, then 0.1 ns, and finally 0.001 ns. Recall that the t_0 optimization works by matching the minima of the cosine Fourier transformation but this is an approximation to find which range the actual Fourier analysis code should look over. Refer to section 3.6 for more details.

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. Many of the parameters of the config file are randomized within a range which the user defines. The values which are actually chosen are stored in the next file called `value.txt`.

5 Using Monte Carlo data

The goal of generating Monte Carlo data is to test the effectiveness and systematic uncertainty of the Fourier method. To do this we run the Fourier analysis code on the Monte Carlo fast rotation signals to try and recover the radial distribution used in the Monte Carlo and compare the recovered mean, standard deviation, and E-field correction to the actual values.

The parameter "create_Fourier_config" and "real_t0_optimization" should both be set to "true" when the Monte Carlo is run so that the config file for the Fourier method is created with the approximated t_0 value. If "real_t0_optimization" is set to "false", then the config file will be created using the fixed value of t_0 used to generate the fast rotation signal, but this will not be the right range for the Fourier analysis.

The parameter "save_result_text" must also be "true" so that the CE, equilibrium radius, and standard deviation of the radial distribution used in the Monte Carlo are all saved.

5.1 Running Monte Carlo using the fast rotation Fourier code

We have another `Python` script which takes the Monte Carlo fast rotation signals generated and runs them on the Fourier analysis code [3]. This is why there is the option to create a Fourier configuration json file.

If the outputs of the Fourier config files are within the `simulation_output/Fourier_config` directory, then the Fourier analysis is run over all of the Fourier config files created in the Monte Carlo by the following command:

```
python analyze_simulation.py simulation_output/Fourier_config
```

The Fourier analysis will then be run on all of the fast rotation signals to try and extract the frequency distribution. The process can be done faster by multiprocessing. To do this go into the file `analyze_simulation.py` and change "multi" to "True" and "numProcess" to the number of processors to be used.

5.2 Obtaining statistics of Monte Carlo

Once the Fourier analysis is run on all of the fast rotation signals, we want to compile all of the data into meaningful statistics. If the output of the Fourier analysis is stored in the directory called `results` and the text file containing the CE, equilibrium radius, and standard deviation of the radial distribution used in the Monte Carlo are stored in the directory `results simulation_output/Plots_Results`, then the following command will compare all the relevant information into meaningful figures:

```
python get_simulation_statistics.py results simulation_output/Plots_Results
```

The resulting distributions will be stored in another directory called `simulation_statistics` within the `results` directory.

5.3 Running the momentum time correlation scan and obtaining statistics

The momentum time correlation scan is run by setting both the parameters "add_momentum_time_correlation" and "momentum_time_correlation_scan" to "true". The number of fast rotation signals generated will no longer be determined by the parameter "num_frs" but instead by the bounds of the scan specified by the parameters "momentum_time_correlation_lower" and "momentum_time_correlation_upper" with spacing "momentum_time_correlation_scan_dr". The momentum time correlation is of a polynomial for of power "momentum_time_correlation_upper".

The momentum time correlation changes the symmetry of the fast rotation signal such that we no longer expect the proper t_0 to occur at the first period of the signal. Set the "fit_envelope" and "use_max_envelope_t0" parameters to "true" so that the envelope is fit with the sum of three Gaussians and the value of t_0 will be found around the period of maximal debunch, which should be approximately at the max of the upper envelope fit. The envelope will be fit after "start_envelope_fit" so that the beam-line contamination of the muon beam can be skipped to mimic real data.

The correlation scan can be run on the fast rotation Fourier code the same way the it is ran for normal Monte Carlo signals using the analyze_simulation.py script 5.1. Once the correlation scan is run the statistics can be compiled using the following command:

```
python get_correlation_scan_statistics.py results simulation_output/Plots_Results
```

The output will be put in another directory called `correlation_scan_statistics` within the `results` directory. There will be three figures of the difference in recovered and actual E-field correction, mean radius, and standard deviation. A linear best fit is given for each figure to measure the systematic uncertainty in the E-field correction, mean radius, and standard deviation associated with a momentum time correlation.

6 Fourier method uncertainty estimation using an ensemble of Monte Carlo

We use Monte Carlo data to measure the systematic uncertainty within the Fourier method [1] as an alternative to the systematic uncertainty estimation in [7]. Instead of trying to measure the systematic uncertainty in each step of the Fourier method, we run the method over an ensemble of Monte Carlo simulations. This is done by generating 1,000 Monte Carlo fast rotation signals and then comparing the recovered E-field correction to the actual E-field correction.

Each Monte Carlo is made from a frequency distribution and longitudinal beam profile which is the sum of three skew-Gaussians. The underlying equation for the Monte Carlo is shown in 19.The value of t_0 is randomly chosen between 45 and 60 ns. Each signal is generated for 200 μs and includes exponentially increasing, additive, Gaussian noise shown in equation (21). A realistic noise level is used with a signal to noise ratio chosen randomly between 900 and 1,000. We use the sum of three skew-Gaussians for the frequency distribution and longitudinal beam profile in order to include highly asymmetric frequency distributions and longitudinal beam profiles which in many cases are more asymmetric than what is found in data.

6.1 A couple of examples

Here are the first six of the Monte Carlo signals out of the 1,000 used. Most of the frequency distributions which are generated are approximately skew-Gauss, but there are cases like in 10 (d) where the frequency distribution has more complicated features. The frequency distribution extracted from real data is not nearly as complicated.

The longitudinal beam profile of the Monte Carlo distribution can be complicated since we know that the real longitudinal beam profile has multiple peaks. This is reflected in the Monte Carlo where we see that 11 (d) and (f) have multiple peaks.

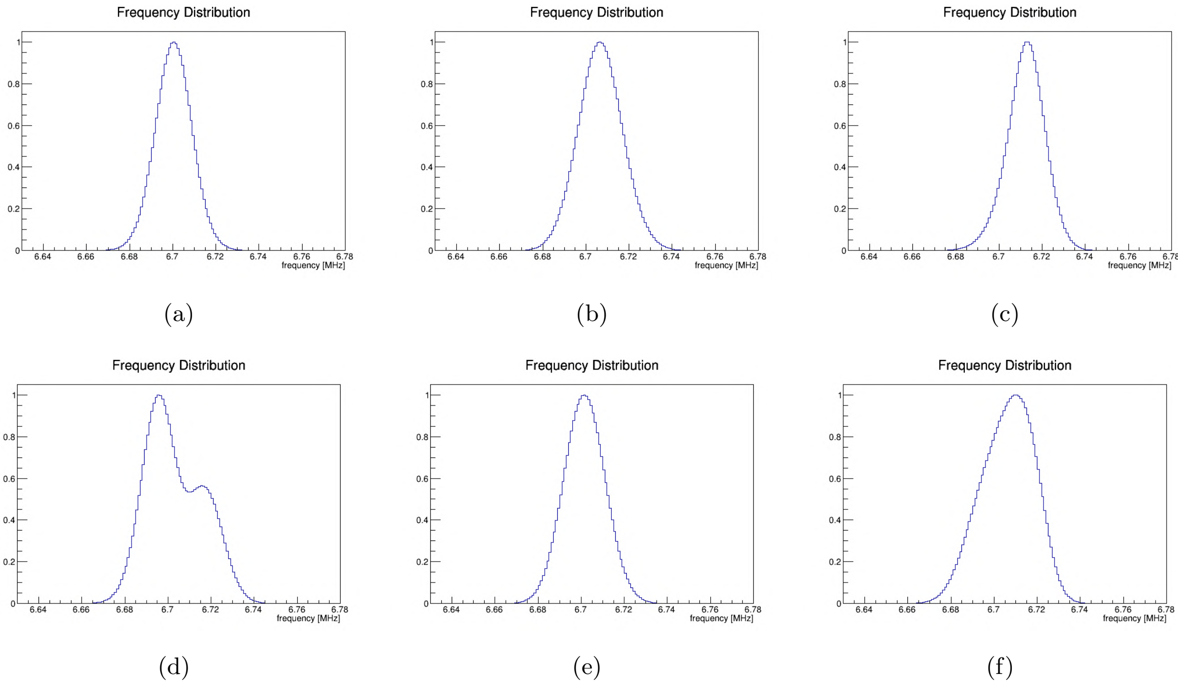


Figure 10: The frequency distributions of the first six Monte Carlo distributions.

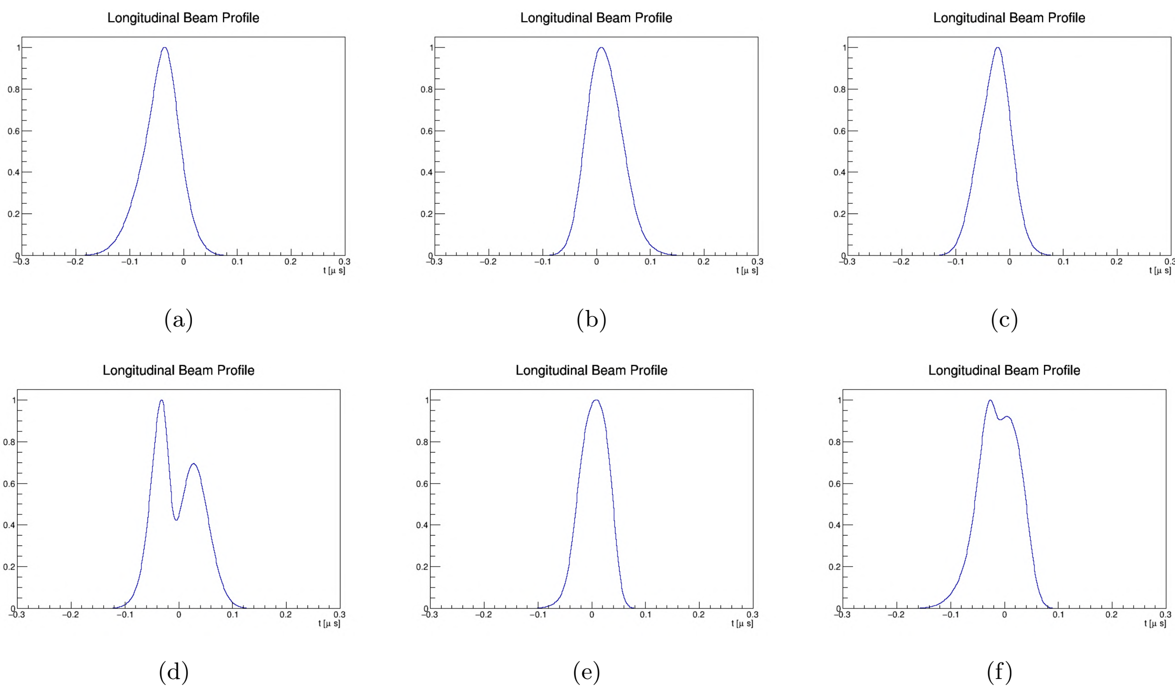


Figure 11: The longitudinal beam profiles of the first six Monte Carlo distributions.

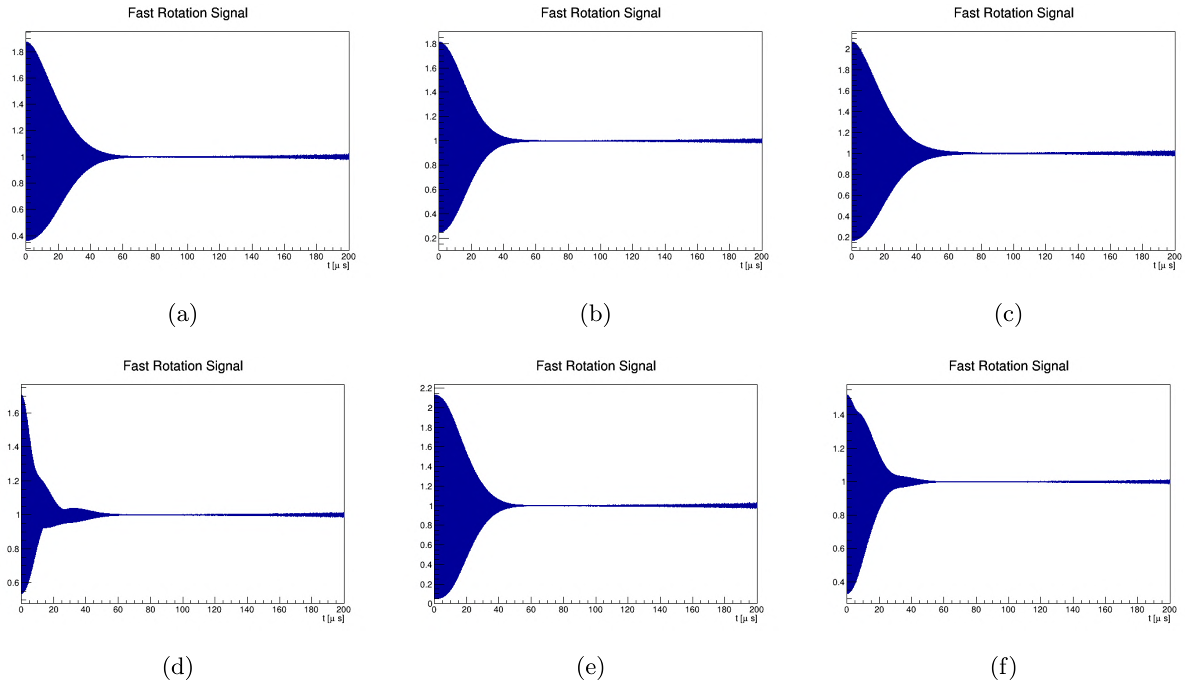


Figure 12: The full fast rotation signals of the five eliminated Monte Carlo distributions.

6.2 Eliminating Outliers

One problem with using highly asymmetric distribution is that there is a possibility for there to be outliers which severely skew the results. In order to make the results meaningful we eliminate five fast rotation signals which were not recovered within 100 ppb since they exhibited properties which are not seen in real data. This means that we eliminated 0.5% of the generated fast rotation signals from being included in the systematic uncertainty estimation.

The problem with these five distributions is that they have both complicated and wide longitudinal beam profiles. When the longitudinal beam profile is wide, the fast rotation signal gets skinnier and the Fourier method does not work as well.

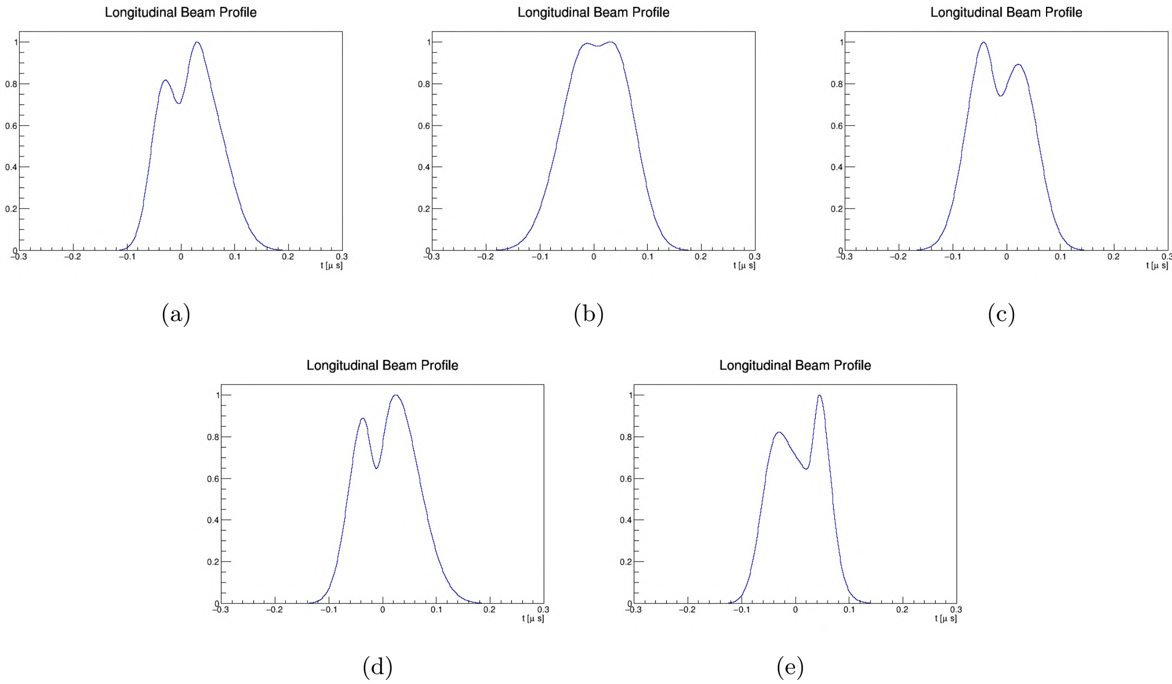


Figure 13: The longitudinal beam profiles of the five eliminated Monte Carlo distributions.

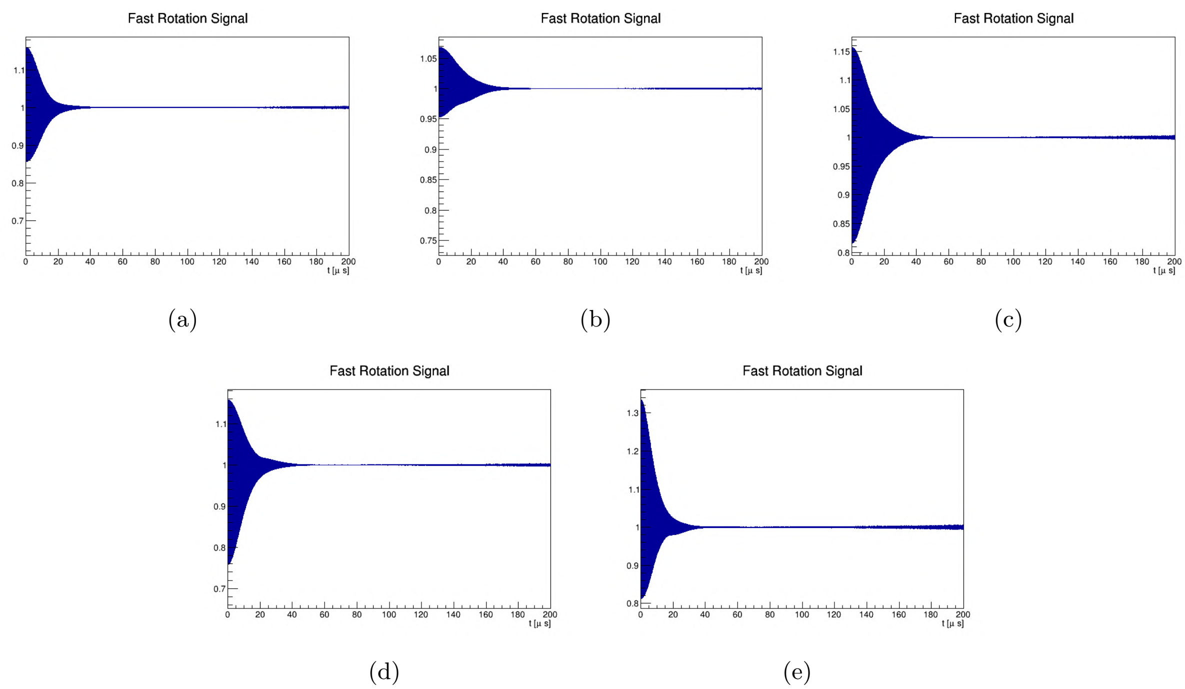
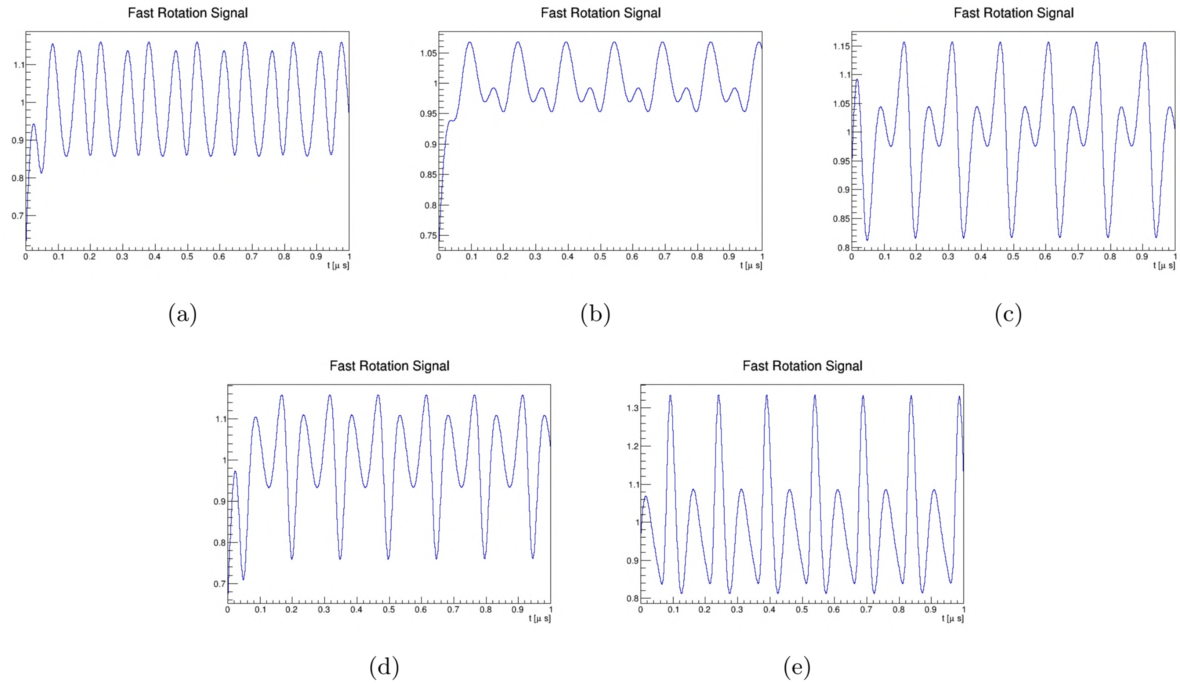
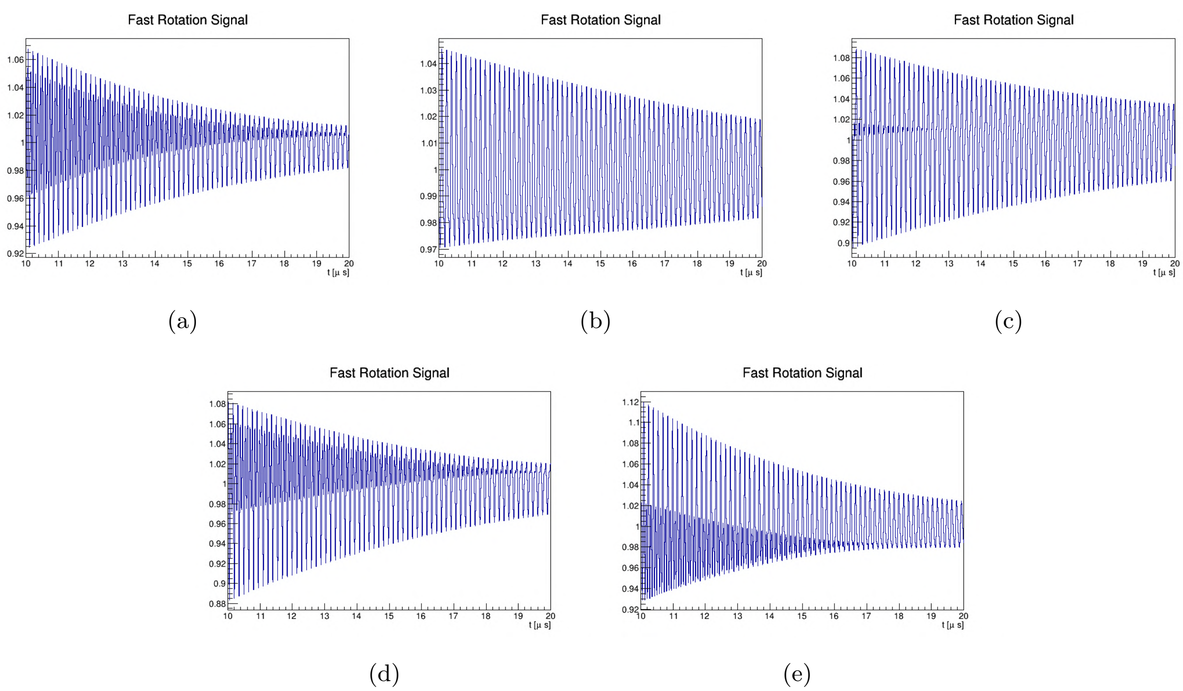


Figure 14: The full 200 μs of the fast rotation signal for the five eliminated Monte Carlo distributions.



(a) (b) (c) (d) (e)

Figure 15: The first 1 μs of the fast rotation signal for the five eliminated Monte Carlo distributions.



(a) (b) (c) (d) (e)

Figure 16: The fast rotation signals between 10 and 20 μs of the five eliminated Monte Carlo distributions.

6.3 Uncertainty estimation

Instead of trying to measure the systematic uncertainty in each step of the Fourier method, we run the method over an ensemble of Monte Carlo simulations. We use the Fourier method to recover the E-field correction and subtract the actual E-field correction from it to get a histogram of the error in the E-field correction which is shown in figure 17.

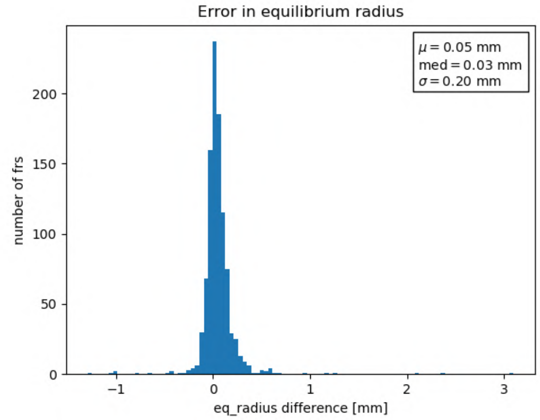
The difference in the recovered and known equilibrium radius (a) has a mean of 0.05 mm with a standard deviation of 0.20 mm yielding an uncertainty of 0.65 mm. The difference in the recovered and known standard deviation (a) has a mean of -0.09 mm with a standard deviation of 0.11 mm yielding an uncertainty of 0.24 mm.

The difference in the recovered and known E-field correction (c) has a mean of 6.35 ppb and a standard deviation of 9.87 ppb. We can take a conservative estimate of 3σ from the mean as our upper bound in systematic uncertainty, so we conclude that the maximum systematic uncertainty in the Fourier method is 35.96 ppb. A less conservative estimate would be to take just 1σ from the mean which would give a systematic uncertainty of 16.22 ppb.

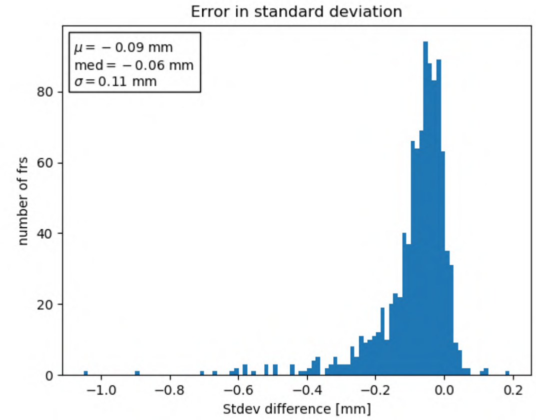
With human intervention when doing the Fourier analysis we would likely be able to further improve our ability to recover the E-field correction by running parameter scans to find the best configuration for each fast rotation signal. In this ensemble testing there was no human input except the elimination of five outliers.

It is still important to estimate the systematic uncertainty in each step of the Fourier method, since we do not know for sure that the ensemble testing directly corresponds to real data. The systematic uncertainty of the Fourier method is measured in several different datasets [8],[9],[10],[11]. Some effects like scraping can be present in real data and is not included in the Monte Carlo, but start time scans of the data sets have shown that the effects of scraping must be very small if they exist. Furthermore, no momentum-time correlation was used which may exist in real data causing its own uncertainty.

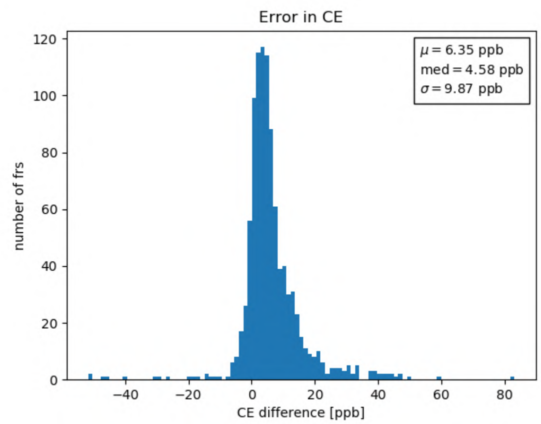
Note that the E-field correction is calculated using the linear approximation which depends on the field index. For Monte Carlo simulations the field index is arbitrary so we use a value of n = 0.1075 which to match the Run-1 60-hour data set.



(a)



(b)



(c)

Figure 17: The difference in the recovered vs known values for 1,000 Monte Carlo fast rotation signals: (a) equilibrium radius (mm), (b) standard deviation (mm), (c) E-field correction (ppb).

7 Parameters

Here we will define all the parameters which need to be defined in the configuration file to run the Monte Carlo. All parameters which use time are in units of μs and frequency is in unites of MHz.

7.1 Basic parameters

tm This is how long the fast rotation signal is generated for. When running the Fourier analysis we only use between 200 and 300 μs of data because by 100 μs the beam has fully debunched around the ring and by 300 μs the exponentially increasing noise dominates the signal. At the late time where noise dominates the signal, using more of the data yields worse results. Even though we get 600 μs or so of real data, we do not want to use it all. There is typically no need to generate a longer Monte Carlo fast rotation signal than what will be used for the Fourier analysis. Keep in mind that the computing time is not linear in t_m, so the Monte Carlo will take a long time to generate for large values of t_m.

num_frs This is the number of fast rotation signals which will be generated.

name This is the name of the ROOT files which are generated containing the fast rotation signal, frequency distribution, radial distribution, and longitudinal beam profile. Each distribution will be numbered after it.

put_in _directory The options are "true" or "false". If "true", then the output directories will be put in the directory named by "name_directory". Otherwise the output directories are just placed where the code is. If the directory already exists then it will be deleted.

name_directory This is the name of the directory which will store all of the output of the Monte Carlo. It is only relevant if "put_in _directory" is "true". If you want to store multiple directories of output, make sure to change the name so that the directories are not replaced by the new output.

start_num The distribution count starts after this number. This is so the ROOT files starting at a later distribution number can be copied and added to others which have already been generated.

binwidth This is the bin width of the fast rotation signal. Typically 1 ns bins are used which would be 0.001 μs for this parameter.

fmin/fmax This is the range which the frequency distribution is stored over.

use_numba The options here are "true" or "false". If "true" then the package `numba` will be used which will speed up the Monte Carlo simulation by about a factor of ten. There will be no change in the output of the Monte Carlo. If "use_numba" is set to "true" and `numba` is not installed then the Monte Carlo will continue without it.

multiprocess The options are "true" or "false". If "true" then the Monte Carlo will multiprocess so it can run multiple simulations on each core of the computer processor. The number of processors used is defined by the parameter "num_processors". With multiprocessing a four core computer then you can Monte Carlo 4 simulations at once, so it will take 4 times less time if the computer does not overheat. This can be combined with `numba` so the simulation can be sped by a factor of 40.

num_processors This is only relevant if "multiprocess" is "true". This parameter must be an integer and defines the number of simulations done at once. For the best outcome use the number of processors that your computer has.

print_time The options are "true" or "false". If "true" then the time it takes for the simulation to complete will be displayed at the end.

display_counter The options are "true" or "false". If "true" then a counter will be displayed which tell you the percentage of Monte Carlo simulations which have already been completed.

print_graphs The options are "true" or "false". If "true" then many plots will be saved which show the outcome of the simulation including of the fast rotation signal at different time intervals, the frequency distribution, the radial distribution, the longitudinal beam profile, and the t_0 optimization if applicable. All plots are saved within the `Plots_results` directory.

save_result_text The options are "true" or "false". If this is "true" then all the information used to generate the fast rotation signal will be saved in a text file called `values.txt` within the `Plots_results` directory.

7.2 t_0 parameters

random_t0 The options are "true" or "false". If "random_t0" is "true" then the t0 used to generate the fast rotation signal will be chosen randomly between the range of "t0_range_lower" and "t0_range_upper".

t0_range_lower/t0_range_upper This is the range which t_0 is chosen from. We can only reconstruct the frequency distribution if we know t_0 within one period so it is recommended that this range be less than a 149.1 ns. Note that this is not the t_0 that will be found when reconstructing the frequency distribution since that t_0 is also effected by the mean of the frequency distribution and longitudinal beam profile.

t0_val If "random_t0" is "false" then this is the value of t_0 which is used.

7.3 Frequency distribution parameters

frequency_random_num_gauss The options are "true" or "false". If true then the number of skew-Gaussians used for the frequency distribution will be chosen randomly between "frequency_num_gauss_lower" and "frequency_num_gauss_upper".

frequency_num_gauss_lower/frequency_num_gauss_upper The options here are only "1","2", or "3" and "frequency_num_gauss_lower" must be greater than or equal to "frequency_num_gauss_upper".

frequency_num_gauss The options here are only "1","2", or "3". If "frequency_random_num_gauss" is "false" then this is the number of skew-Gaussians chosen to make the frequency distribution.

frequency_random_center The options are "true" or "false". If "true" then the center of each skew-Gaussian will be chosen between "frequency_center_lower" and "frequency_center_upper". If "false" then the center will be "frequency_center". This is measured in fractional energy offset so a value of 0% corresponds to the frequency distribution being centered at the magic frequency. The frequency center should typically remain within the range of -0.0020 and -0.0020. It is important to make sure the frequency distribution remains within the collimator aperture which can be difficult since the skew and width can also make the frequency distribution go outside the physical frequency range.

frequency_center_lower/frequency_center_upper This is the range which can be used when "frequency_random_center" is "true".

frequency_center This is the center used when "frequency_random_center" is "false".

frequency_random_width The options are "true" or "false". If "true" then the width of each skew-Gaussian will be chosen between "frequency_width_lower" and "frequency_width_upper". If "false" then the width will be "frequency_width". The width of the frequency distribution should typically be less than 0.0020 to avoid going outside the collimator aperture.

frequency_width_lower/frequency_width_upper This is the range which can be used when "frequency_random_width" is "true".

frequency_width This is the width used when "frequency_random_width" is "false".

frequency_random_skew The options are "true" or "false". If "true" then the skew of each skew-Gaussian will be chosen between "frequency_skew_lower" and "frequency_skew_upper". If "false" then the skew will be "frequency_skew". It is suggested that values of the skew should typically remain within the range of -2 and 2.

frequency_skew_lower/frequency_skew_upper This is the range used when "frequency_random_skew" is "true".

frequency_skew This is the skew which is used when "frequency_random_skew" is "false".

frequency_random_height The options are "true" or "false". If "true" then the height of each skew-Gaussian will be chosen between "frequency_height_lower" and "frequency_height_upper". If "false" then the skew will be "frequency_skew". Since the normalization is arbitrary, only the relative height of each skew-Gaussian matters, and if only a signal skew-Gaussian is used then the height does not matter. The height must be positive.

frequency_height_lower/frequency_height_upper This is the range which can be used when "frequency_random_height" is "true".

frequency_height This is the height used when "frequency_random_height" is "false".

height_multigauss_factor_freq If two or three skew-Gaussians are used, then this value divides the height of the second and third. This can be used to try and obtain one larger central peak and smaller peaks on the sides.

7.4 Longitudinal beam profile parameters

longitude_random_num_gauss The options are "true" or "false". If true then the number of skew-Gaussians used for the longitude beam profile will be chosen randomly between "longitude_num_gauss_lower" and "longitude_num_gauss_upper". It is suggested that values of the skew should always remain within the range of -2 and 2.

longitude_num_gauss_lower/longitude_num_gauss_upper The options here are only "1","2", or "3" and "longitude_num_gauss_lower" must be greater than or equal to "longitude_num_gauss_upper".

longitude_num_gauss The options here are only "1","2", or "3". If "longitude_random_num_gauss" is "false" then this is the number of skew-Gaussians chosen to make the frequency distribution.

longitude_random_center The options are "true" or "false". If "true" then the center of each skew-Gaussian will be chosen between "longitude_center_lower" and "longitude_center_upper". If "false" then the center will be "longitude_center". A value for the longitude center should be within the range of -T/2 to T/2 to stay within one period of the ring. It is advised to use values substantially smaller when multiple peaks are used.

longitude_center_lower/longitude_center_upper This is the range which can be used when "longitude_random_center" is "true".

longitude_center This is the value center when "longitude_random_center" is "false".

longitude_random_width The options are "true" or "false". If "true" then the width of each skew-Gaussian will be chosen between "longitude_width_lower" and "longitude_width_upper". If "false" then the width will be "longitude_width". The width of the longitudinal beam profile should typically be less than T/2.

longitude_width_lower/longitude_width_upper This is the range which can be used when "longitude_random_width" is "true".

longitude_width This is the value width when "longitude_random_width" is "false".

longitude_random_skew The options are "true" or "false". If "true" then the skew of each skew-Gaussian will be chosen between "longitude_skew_lower" and "longitude_skew_upper". If "false" then the skew will be "frequency_skew".

longitude_skew_lower/longitude_skew_upper This is the range used when "longitude_random_skew" is "true".

longitude_skew This is the skew used when "longitude_random_skew" is "false".

longitude_random_height The options are "true" or "false". If "true" then the height of each skew-Gaussian will be chosen between "longitude_height_lower" and "longitude_height_upper". If "false" then the skew will be "frequency_skew". Since the normalization is arbitrary, only the relative height of each skew-Gaussian matters, and if only a signal skew-Gaussian is used then the height does not matter. The height must be positive.

longitude_height_lower/longitude_height_upper This is the range which can be used when "longitude_random_height" is "true".

longitude_height This is the value height when "longitude_random_height" is "false".

height_multigauss_factor_long If two or three skew-Gaussians are used, then this value divides the height of the second and third. This can be used to try and obtain one larger central peak and smaller peaks on the sides.

7.5 Noise parameters

add_noise The options are "true" or "false". If "add_noise" is "true" then additive noise will be included in the Monte Carlo fast rotation signal.

random_signal_to_noise_ratio The options are "true" or "false". If "true" then a value is chosen between "signal_to_noise_ratio_lower" and "signal_to_noise_ratio_upper". If "false" then "signal_to_noise_ratio" is used. A signal to noise ration between 900 and 1000 is a good conservative value since it is slightly more than the signal to noise ration for real data.

signal_to_noise_ratio_lower/signal_to_noise_ratio_upper This is the range of values which the signal to noise ration is chosen when "random_signal_to_noise_ratio" is "true."

signal_to_noise_ratio This is the value used when "random_signal_to_noise_ratio" is "false".

noise_type The options here are "gauss" or "uniform". If "noise_type" is "gauss" then we use a third of the noise amplitude as the standard deviation and the mean of the noise is centered at 0. If "noise_type" is "uniform", then the noise will be chosen uniformly between plus and minus the noise amplitude. This is only relevant when "add_noise" is "true".

exponential_increasing_noise The options are "true" or "false". If "exponential_increasing_noise" is "true" then the additive will be multiplied by an exponential function to model the exponentially increasing noise in real data. This is only relevant when "add_noise" is "true".

7.6 Momentum time correlation parameters

add_momentum_time_correlation The options are "true" or "false". If "momentum_time_correlation" is "true" then a correlation is added between the frequency distribution and the longitudinal beam profile. It is suggested to use values for the correlation between -0.0015 and 0.0015 to have less than a $\pm 0.15\%$ correlation which the expected upper bound of the correlation for real data. The correlation is model by a polynomial of degree "momentum_time_correlation_power".

random_momentum_time_correlation The options are "true" or "false". If "true" then the momentum time correlation will be chosen randomly between the range of "momentum_time_correlation_lower" and "momentum_time_correlation_upper". If "false" then the momentum time correlation will be "momentum_time_correlation".

momentum_time_correlation This is the value for the momentum time correlation used when "random_momentum_time_correlation" is "false".

momentum_time_correlation_power This is the power of the momentum time correlation. A value of 1 represents a linear momentum time correlation, 2 is a quadratic correlation, and so on.

momentum_time_correlation_lower/momentum_time_correlation_upper This is the range of values which the momentum time correlation is chosen between when "random_momentum_time_correlation" is "true".

momentum_time_correlation_scan The options are "true" or "false". If "true" then a momentum time correlation scan will be run which supersedes the "num_frs" value and instead generates fast rotation signals with correlations between "momentum_time_correlation_lower" and "momentum_time_correlation_upper" with a step size of "momentum_time_correlation_scan_dr". For the best results the scan should be run without randomizing the frequency distribution or the longitudinal beam profile so that the effects of the correlation can be isolated without other systematic issues at play.

momentum_time_correlation_scan_dr This is the step sized used for the momentum time correlation scan. It is only relivent when the parameter "momentum_time_correlation_scan" is set to "true".

fit_envelope The options are "true" or "false". This determines if a the envelope of the fast rotation signal will be fit with the sum of three Gaussian. Figures of the envelope and its fit will be saved including the max of the upper and lower envelopes. The max of the upper envelope is approximately which period the the cosine transformation should begin.

start_envelope_fit This is when the envelope fit will begin. This value only matters if "fit_envelope" is "true". In real data, the first approximately 4 μs suffer from positron contamination of the muon beam, so we can not include this part of the fast rotation signal in the envelope fit. The envelope fit can be started at 4 μs to mimic real data.

use_max_envelope_t0 The options are "true" or "false". If "true" then the value of t_0 which is optimized will be around the turn corresponding to the max of the envelope fit. This can be in the negative time domain since we extend the fit function 10 μs before the start of the fast rotation signal since negative correlation can yield a peak in the negative time domain. This only has an effect on the "real_t0_optimization" so that parameter must also be yet to "true". The config file created when "create_Fourier_config" will then be around the period found from the max of the upper envelope.

7.7 Real t_0 optimization parameters

real_t0_optimization The options are "true" or "false". If "real_t0_optimization" is "true" then we will approximate the value of t_0 by finding minimizing the distance between the minima of the cosine Fourier transformation using a value of ts defined by the parameter "ts_for_optimization". This optimized value of t_0 will be saved in the text file `value.txt` if the parameter "save_result_text" is "true". This t_0 is also what is used to make the config file for the Fourier analysis when "create_Fourier_config" is "true". The Fourier config file is made so that the Fourier analysis looks 2 ns away from the optimized value of t_0.

ts_for_optimization This is the t_s used to optimize the approximate value of t_0 when "real_t0_optimization" is "true". A value of $t_s = 4$ μs should work well in all cases.

7.8 Fourier method config parameters

We want to create the configuration file for the Fourier analysis code. All of the important information for the Fourier analysis configuration file can be changed here and all the optional parameters are turned off. For more information on the Fourier configuration parameters see the Fourier analysis user guide [3]. We will only briefly describe these parameters here.

create_Fourier_config The options are "true" or "false". If "true" then config files will be created for each Monte Carlo fast rotation signal so that the signal can be analyzed using the Fourier analysis code [3]. The config file will use the value of t_m used to generate the fast rotation signal, and search 2 ns away from the value of t_0 which was optimized when "real_t0_optimization" is "true".

Fourier_config_name This will be the name of the configuration file which is create followed by the number of the Monte Carlo.

n_t0_opt This is the number of iterations that t_0 is optimized with each iteration including more points in the fitting which are "t0_background_threshold" standard deviations away form the mean of the background. The first iteration only looks at the points which are outside of the collimator aperture.

background_fit The options here are "poly","sinc","erfi", or "triangle". This is the function that fits the background.

background_frequencies The options here are "all" or "physical". If "all", then the background is fit using points both outside and within the collimator aperture outside the collimator aperture. If "physical", then the points outside the collimator aperture will only be used for the first iteration in the t_0 optimization.

remove_background The options are "true" or "false". If "true" then the points which are determined to be noise are set to zero in the radial distribution and for the E-field calculation.

background_removal_threshold This number of standard deviations away form the mean are determined to be noise and are set to zero if "remove_background" is "true". The points determined to be noise are set by the t_0 optimization.

t0_background_threshold This is the number of standard deviations bellow which points are considered to be noise when for the background fit.

tS This is the start time which is used. A value of at least 4 μs should be used to avoid positron contamination.

freq_step_size This is the step size of the frequency distribution. A value of 2 kHz works well.

References

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

[2] 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

[3] A. Chapelain, J. Fagin, D. Rubin, D. Seleznev, *Cornell fast rotation Fourier analysis user guide*, GM2-doc-18460

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

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

[6] A. Chapelain, J. Fagin, D. Rubin, D. Seleznev, *Cornell momentum time correlation study*, GM2-doc-COMING-SOON

[7] A. Chapelain, J. Fagin, D. Rubin, D. Seleznev, *Cornell fast rotation Fourier analysis performance study with toy Monte Carlo simulation*, GM2-doc-19132-v2

[8] A. Chapelain, J. Fagin, D. Rubin, D. Seleznev, *Extraction of the radial distribution of the stored muon beam for the Run–1 60–hour data set via the Cornell fast rotation Fourier method: estimation of the electric field correction to the anomalous spin precession frequency*, GM2-doc-19150-v3

[9] A. Chapelain, J. Fagin, D. Rubin, D. Seleznev, *Extraction of the radial distribution of the stored muon beam for the Run–1 9–day data set via the Cornell fast rotation Fourier method: estimation of the electric field correction to the anomalous spin precession frequency*, GM2-doc-19252-v4

[10] A. Chapelain, J. Fagin, D. Rubin, D. Seleznev, *Extraction of the radial distribution of the stored muon beam for the Run–1 end game data set via the Cornell fast rotation Fourier method: estimation of the electric field correction to the anomalous spin precession frequency*, GM2-doc-19258-v2

[11] A. Chapelain, J. Fagin, D. Rubin, D. Seleznev, *Extraction of the radial distribution of the stored muon beam for the Run–1 high kick data set via the Cornell fast rotation Fourier method: estimation of the electric field correction to the anomalous spin precession frequency*, GM2-doc-COMING-SOON