Thesis Report

Ecole Polytechnique | Bachelor of Science

Laboratoire Leprince-Ringuet

Direction Reconstruction with the Low Energy Analysis Framework for $\mathcal{O}(10)$ MeV-scale Events in the Hyper-Kamiokande Experiment

Author & Intern:
Alper Tezcan
alper.tezcan@polytechnique.edu

Supervisor:
Prof. Thomas Mueller
thomas.mueller@llr.in2p3.fr

January 15th, 2023 — March 15th, 2023

Direction Reconstruction with the Low Energy Analysis Framework for $\mathcal{O}(10)$ MeV-scale Events in the Hyper-Kamiokande Experiment

Alper TEZCAN[*a]

[a] *Ecole Polytechnique, Bachelor of Science*, 91128, Palaiseau, FRANCE

March 2024

Abstract

This analysis describes the position and direction reconstructions in the Hyper-Kamiokande experiment for low energy ($\mathcal{O}(10)$ MeV) neutrinos. Position reconstruction based on the negative-log-likelihood (NLL) minimization spanned by the probability density function (PDF) generated by the hittime residuals is reviewed. A similar NLL minimization is done for the direction reconstruction with a PDF generated by the half-angle of the Cherenkov cone. The effect of darkhits is investigated for direction resolution. An introductory level of detector-physics knowledge and familiarity with ROOT is recommended for the reader.

Contents

[*] E-mail for correspondence: alper.tezcan@polytechnique.edu

I Introduction

Low-energy neutrinos are of great interest to physicists as they provide valuable insights into fundamental particle interactions and astrophysical phenomena. For example, neutrinos emitted by the Sun or supernovae lie in the $\mathcal{O}\,(10)$ MeV-scale and are crucial for several analyses such as solar neutrino oscillations [1], transient supernova [2], and Diffuse Supernova Neutrino Background (DSNB)[1] [3]. The Cherenkov effect[2] is an effective and common way to probe neutrinos, exemplified by the Super-Kamiokande (SK) experiment. The SK experiment —located in Japan— is a water Cherenkov (WC) detector consisting of a cylindrical tank filled with 50 kilotons of ultra-pure water and surrounded by photomultiplier tubes (PMTs). As a neutrino interacts with a water molecule in the tank, particles such as electrons, muons, and hadrons are produced, which can travel faster than the speed of light in water and emit Cherenkov radiation. PMTs on the tank's surface detect the Cherenkov light emitted from these secondary particles.

In these events, we record information such as the positions of the hit PMTs on the tank, the timing of hits, and intensity of received light. Therefore, the signal detection and the measurement of neutrino properties, such as their energy and flavor, rely on the reconstruction of the produced particles.

The future Hyper-Kamiokande (HK) experiment includes a low-energy part similar to the SK. BONSAI, the current algorithm in use for SK, has been superseded by a new algorithm called the Low Energy Analysis Framework (LEAF) written in C++ for the development of the HK. This algorithm is based on the maximization of a likelihood (i.e., minimization of a negative-log-likelihood (NLL)) generated for the arrival time of the light to the PMTs.

Even if the current version of LEAF achieves a better resolution for the reconstruction of the vertex (i.e., the interaction point of the neutrino) position, it does not reconstruct the direction of the secondary particle or its energy. The direction and energy fits are done sequentially after the position fit, which introduces degeneracy between reconstructed position, direction, and energy values.

In this study, we will look for improvements in the LEAF by implementing the direction reconstruction in the algorithm. We will check performances with respect to different energies and distances from the tank's walls. We will use simulations generated by Monte Carlo (MC) methods as data.

[1] The DSNB is a theoretical population of neutrinos (and anti-neutrinos) originating from all core-collapse supernovae events throughout the history of the visible Universe. It is theorized to be isotropic and consists of $\mathcal{O}\,(10)$ MeV neutrinos.

[2] The Cherenkov radiation is a shock phenomenon when a charged particle passes through a medium with refractive index n faster than the speed of light in the medium, (i.e., $\beta > 1/n$). The wavelength of the emitted light is in the $[300, 600]$ nm range.

The report starts with a brief description of the future HK detector (section II). Then, we delve deeper into the physics of the detection process with section III. Later in this section, we introduce our simulation tool WCSim, and analyze the simulation samples generated by MC methods. The core of this study is discussed in section IV. After a brief introduction to the LEAF, we focus on position (section IV.1) and direction (section IV.2) reconstructions. The results are discussed in section V after a breakdown of the future work (section IV.3). Some details of likelihood theory (App. A.1), tested NLL candidates (App. A.2), simulations (App. B.1), and likelihood optimization tools (App. B.2) can be found in the appendices.

II The Hyper-Kamiokande Experiment

The HK detector, proposed as the successor to the SK experiment, is a next-generation, large-scale water Cherenkov neutrino detector. The optimal tank design is determined as two cylindrical detectors that are 60 m in height and 74 m in diameter with 40% photocoverage (see figs. 1 and 2), where a staging between the first and second tank is considered [4]. As shown in Table 1, the HK detector will become an invaluable tool for studying neutrinos and probing fundamental physics phenomena, with its larger size, improved sensitivity, and advanced technology compared to other water Cherenkov detectors.

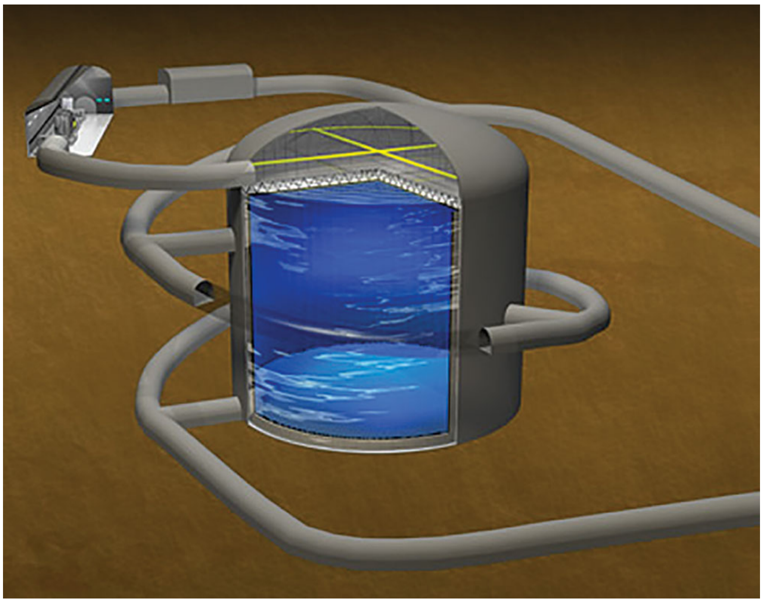

FIGURE 1: Illustration of the HK first tank (Adapted from [4]).

HK will be a multipurpose neutrino detector with a rich physics program concerning several analyses. Studies on neutrino oscillations from accelerator, atmospheric, and solar neutrinos will improve the measurements mixing angles and mass squared differences, aiming to detect asymmetries in neutrino and antineutrino oscillations, potentially caused by a CP-violating phase. These studies, therefore, could provide promising explanations for the matter-antimatter asymmetry in the Universe. The search for nucleon decays will

[3] Fiducial volume is the effective analysis volume, defined as the region located at least 2 m away from any inner detector wall.

[4] Currently, the Hyper-Kamiokande has funding for only 20000 PMTs. Therefore, the analysis baseline is set to 20000 PMTs (see section III).

	KAM	SK	HK first tank
Depth	1000 m	1000 m	650 m
Dimensions of water tank			
diameter	15.6 m ϕ	39 m ϕ	74 m ϕ
height	16 m	42 m	60 m
Total volume	4.5 kton	50 kton	258 kton
Fiducial volume[3]	0.68 kton	22.5 kton	187 kton
Outer detector thickness	$\sim$ 1.5 m	$\sim$ 2 m	1 $\sim$ 2 m
Number of PMTs			
inner detector (ID)	948 (50 cm ϕ)	11129 (50 cm ϕ)	40000[4] (50 cm ϕ)
outer detector (OD)	123 (50 cm ϕ)	1885 (20 cm ϕ)	6700 (20 cm ϕ)
Photo-sensitive coverage	20%	40%	40%[4]
Single-photon detection efficiency of ID PMT	unknown	12%	24%
Single-photon timing resolution of ID PMT	$\sim$ 4 nsec	2-3 nsec	1 nsec

TABLE 1: Parameters of past (KAM [5, 6]), running (SK [7, 8]), and future HK first tank [4] water Cherenkov detectors. The KAM and SK have undergone several configuration changes, and the parameters shown in the table are for KAM-II and SK-IV. The single-photon detection efficiencies are products of the quantum efficiency at peak ($\sim$ 400 nm), photo-electron collection efficiency, and threshold efficiency.

test results from Grand Unified Theories. Similar to SK, the Hyper-Kamiokande detector will observe more statistics in supernovae, providing crucial data for understanding the explosion mechanism. Furthermore, detecting astrophysical neutrinos from sources like dark matter annihilation, gamma-ray burst jets, and pulsar winds could advance our understanding of the Universe.

The photosensors are mounted on a stainless steel supporting framework as in the configuration shown in figure 3. The grid size of the photosensor array in the inner detector is about 70 cm, while the size for the outer detector is roughly 2 m. Spaces between the inner PMTs are lined with opaque black sheets to prevent light leaks. Meanwhile, gaps between outer PMTs are filled with reflective sheets to enhance light collection. The stainless steel framework and the PMT cables are placed in the 60 cm thick insensitive region between the inner and outer detectors.

The design of the Hyper-K tank, such as the waterproof tank liner and the PMT-supporting framework, is cost-effective and satisfies the demands of future HK physics programs.

III Detection Processes and Simulation Samples

III.1 Physical considerations

Let us delve deeper into the physics of events for neutrino detection in SK/HK. We start by considering two characteristic phenomena observed in the low energy regime, solar neutrinos and DSNB (see fig. 4 for energy spectra of these two sources).

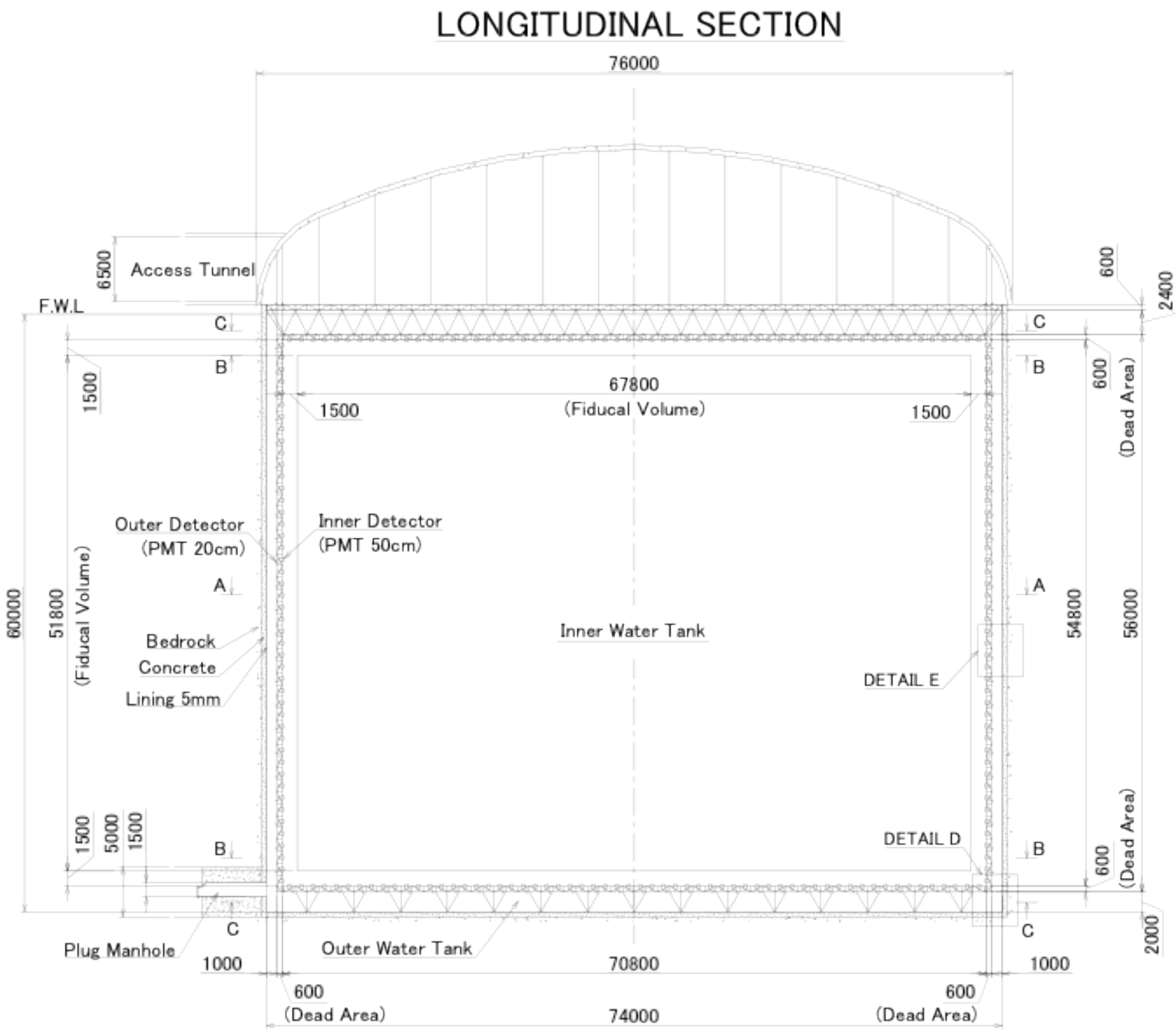

FIGURE 2: Water tank dimensions for HK (Adapted from [4]).

The main detection channel for solar neutrinos is the elastic scattering of neutrinos with electrons of H or O in the water molecules inside the tank. In general, the elastic scattering is given as,

$$\begin{aligned} \nu_l + e^- &\longrightarrow \nu_l + e^- \\ \overline{\nu}_l + e^- &\longrightarrow \overline{\nu}_l + e^- \end{aligned} , \tag{III.1}$$

where $l \in \{e, \mu, \tau\}$ is the flavor. In the context of solar neutrinos, there is no antineutrino production by solar fusion. Even if we receive some antineutrinos due to β^- decays of radioactive elements or photo-fission of heavy isotopes in the Sun, the effect of these is negligible in our analysis since *(i)* their energies are $\mathcal{O}(1)$ MeV, *(ii)* their flux is negligible compared to solar neutrinos [9]. Furthermore, it should be noted that electron neutrinos are detected more dominantly compared to other flavors since the total flux of elastic scattering is given as,

$$\Phi_{tot} = \Phi_{\nu_e} + 0.1553 \times \Phi_{\nu_\mu,\nu_\tau} . \qquad [10] \tag{III.2}$$

An example elastic scattering of ν_e is shown in Tab. 2. For a more detailed theoretical treatment of neutrino-electron elastic scattering with different energy scales, the reader can consult [11].

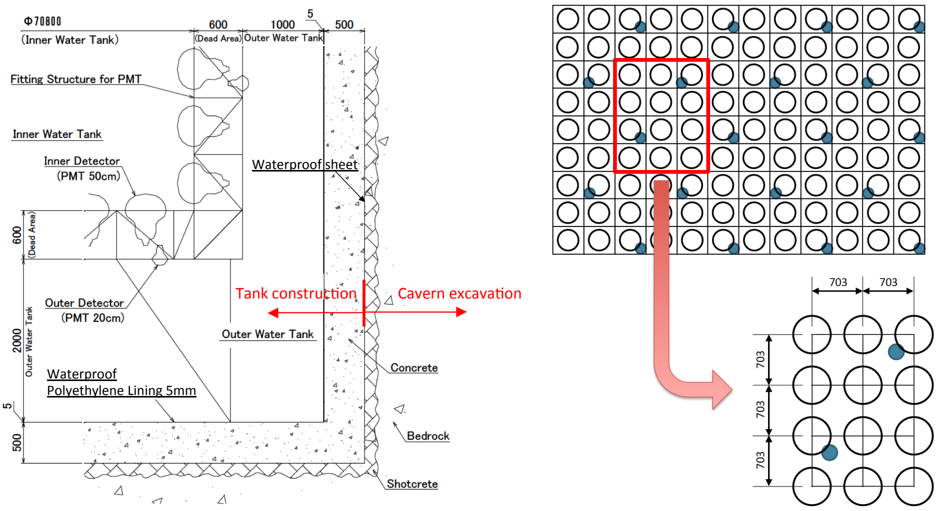

FIGURE 3: Schematics for HK tank surface. Left: Boundary of the water tank, PMTs for inner/outer detector regions are visible. Right: Photosensor configuration for inner (open circles) and outer (blue circles) detector. (Adapted from [4].)

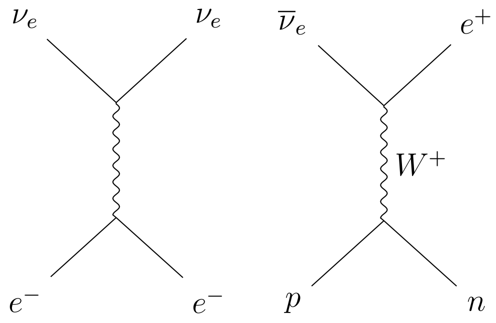

TABLE 2: Two examples for events. Left: Elastic scattering. Right: Inverse beta decay.

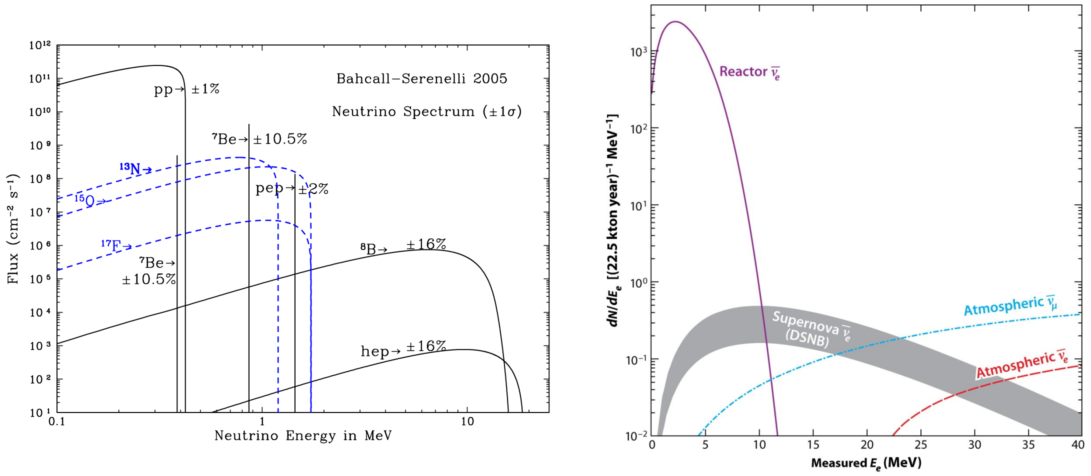

FIGURE 4: Energy spectra for two cases. Left: Solar neutrino energy spectrum for the solar model BS05 [12]. Right: Detection rates in SK assuming 100% neutron capture efficiency. [13].

Now, let us focus on the DSNB neutrinos. Neutrinos and antineutrinos from the supernova background are emitted across all the flavors. Therefore, it is crucial to consider

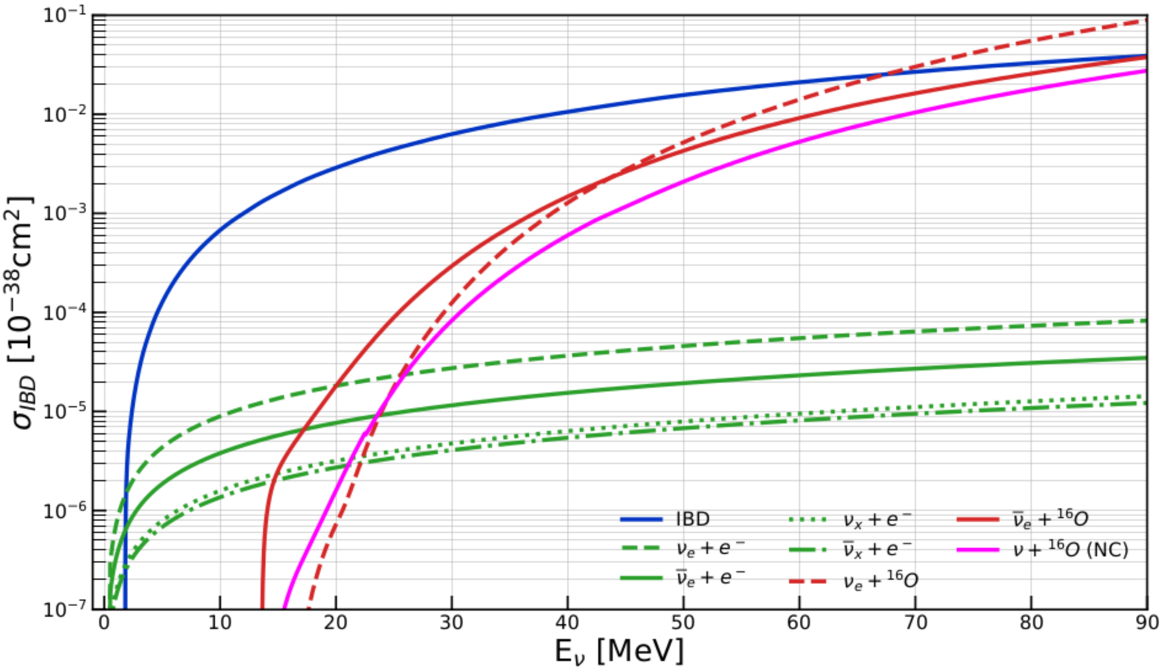

FIGURE 5: Cross-sections for the main processes to detect neutrinos in WC detectors. Blue: protons in the H atoms via inverse beta decay. Green: the electrons in the H atoms via charged current interactions. Red: ^{16}O atoms via charged current interactions. Pink: ^{16}O atoms via neutral current interactions. Here, $x \in \{\mu, \tau\}$ and the NC cross-section can refer to any ν or $\overline{\nu}$ species. (Adapted from [14].)

several interactions in the water. In fact, one will see that the interaction cross-section of the inverse beta decay (IBD) is several orders of magnitude greater than other possible interactions, so is the dominantly-observed scenario (see fig. 5) [14].

In a more general setting, charged current (CC) interactions can be neutrino or antineutrino-driven, that is,

$$\begin{aligned} \nu_l + n &\longrightarrow l^- + p \\ \overline{\nu}_l + p &\longrightarrow l^+ + n \end{aligned} . \tag{III.3}$$

Oxygen atoms are the only neutron sources in the water and neutrino-driven CC interaction is the red-dashed curve in fig. 5. It is comparable to antineutrino-driven CC interactions on ${}^{16}O$ (red-plain curve) and negligibly small compared to the IBD on H atoms. Furthermore, IBD involving other antineutrino flavors is sufficiently small for low energies. Therefore, the most dominantly-observed interaction is shown in table 2. For DSNB, as shown in fig. 4, there is an irreducible background due to reactor $\overline{\nu}_e$. Due to such backgrounds, improvements in low-energy reconstruction play a crucial role in increasing the number of statistics for DSNB studies.

Throughout this study, we will narrow our analysis to Cherenkov light coming from electrons inside the water tank since the dominant events have electrons and positrons as secondary particles. It should be noted that there is no difference in the trajectory of electrons and positrons due to the absence of a magnetic field in the SK/HK tanks. Naturally, the Cherenkov cones (or light detected by PMTs, better to say) are the same.

Now, one might ask about the light emerging from the interaction of positrons with the electrons in water molecules. In the center-of-mass frame of the e^-e^+, the cross-section for electron-positron annihilation is proportional to $1/v$,[5] and therefore, becomes significant when their energies are close. Given that the electron (of the water molecule) is at rest, the positron has to lose its energy to interact with an electron. In that case, the emitted photons will have 0.511 MeV energy, which is below our experimental energy threshold.

As we will focus on electron reconstruction for this study, we can improve our computation times by specifying our MC simulations in certain ways. For example, as (anti-)neutrino-water interactions have infinitesimal cross-sections, one can observe only a couple of electrons for 10^{42} neutrinos generated by the simulation. Instead of generating infinitely many neutrinos, one can directly simulate electrons with desired properties (energy, position, direction, etc.). One further physical consideration is critical for direction reconstruction (see section IV.2). In direction-related computations, we will use the position of the vertex instead of the electron. However, the electron position is changing as opposed to the vertex position. Studying the kinematics, one will see that the track of an electron at $\mathcal{O}(10)$ MeV is at the order of $\mathcal{O}(1)$ cm, which is negligibly small compared to the vertex resolution of the position reconstruction (see section IV.1) and the size of the tank.

III.2 Simulation tools

The Water Cherenkov Simulation (WCSim) package is a flexible, Geant4-based code to simulate the geometry and physics of user-defined WC configurations. WCSim is open-source and can be accessed from https://github.com/WCSim/WCSim.

WCSim takes several inputs such as the detector geometry, PMT types, and their coverage specified by the user through `.mac` card files. The simulation of the detector response uses Monte Carlo methods. A macro card is provided in appendix B.1 as an example.

In the MC generation, we used WCSim version 1.12.2 and ROOT6. In our analysis, we generated events using the Hybrid Multi-PMT geometry. However, we set PMT coverage to 0% for all the PMTs except for PMT-type0 (box-and-line PMTs). We simulated the detector response for the monoPMT framework because this preliminary version of LEAF is implemented only for the monoPMT framework. As mentioned earlier, Hyper-Kamiokande was planned to have 40% PMT coverage. However, the collaboration has funding for only 20% PMT coverage, and the baseline configuration for many analyses, including this study, is set to 20.215% PMT (type0) coverage. Photodetector parameters in the simulation include the timing resolution, dark noise rate[6], and the overall efficiency for a photon to register a charge (including the quantum efficiency, collection efficiency, and hit efficiency). Some parameters in this study are copied from SKDETSIM values (SK detector simulation). For example, the simulation samples are either generated without darkhits (equivalently darkrate = 0.0 kHz) or with darkhits at 4.2 kHz darkrate.

The output for the WCSim code includes both the raw hit and the digitized information. The raw hit information consists of all the PMTs that received a hit and the relevant details of those hits. The digitized information has the number of hits only in a trigger

[5] Here, v is the velocity of e^- and e^+. The explicit expression for cross-section is $\sigma = \frac{4\pi}{cv}\left(\frac{\hbar\alpha}{m}\right)^2$.

[6] Darkhit, Dark Noise: The relatively small electric current that flows through photosensitive devices such as PMTs even when no photons enter the device; it is the current due to the random generation of electrons and holes within the depletion region of the device.

window, the charge, and the timing of the hit tubes.

III.2.1 Bug report: `TriggerType==-1`

In the reconstruction step of MC-generated samples, we observed several crashes in the algorithm for some events. After further investigation, it is observed that all these events have `TriggerType=-1`, which represents the absence of digitized hits. It is expected for energies < 5 MeV since the electrons are less energetic and radiate less Cherenkov photons. It can be physically justified for vertices close to the borders of the tank as the photons emitted in these regions might not be detected by PMTs. However, in the $5-10$ MeV range, some events located at the center of the tank did not pass the trigger conditions, which is unexpected. One possible explanation is that the Cherenkov light emitted far from PMTs also cannot pass the trigger conditions (or, "gets lost", in a daily perspective). For this study, such events are passed by a simple `if` condition. However, this problem requires further checks. The table 3 shows the percentage of such events for different energies.

Energy (MeV)	`TriggerType == -1` (%)
2	94.1
3	65.0
5	9.41
10	0.480
20	0.300
50	0.100

Table 3: Presentage of events that cannot pass the triggers

III.3 Testing the simulation samples

We have discussed the physics of events and introduced our simulation tool WCSim. Now, we will plot several properties of our MC samples and justify the distributions. In this way, let us verify that our WCSim events are well-prepared. In reality, some of these plots are used for the probability density functions used in the likelihood (see fig. 12). To consider all possible types of events, let us look at the plots for two extreme cases.

1) Random position, random direction, mono-energetic: In the fig. 6, the first thing we see is the similarity between x and y axes, compared to z. This similarity is simply due to the cylindrical symmetry of the tank. We see a circle-like distribution for binning in x and y. This is due to fact that, for given δx slice (i.e., the slice between x and $x+\delta x$), number of events are given by,

$$\delta n_x = \rho \times 2H \times 2\sqrt{R^2 - x^2} \times \delta x \tag{III.4}$$

here, ρ is the number-density of events (which is constant since the vertices are randomly generated in the tank), R is the radius, and H is the half-height of the cylinder. The same trivially follows for y due to symmetry arguments. On the other hand, for the z axis, $\delta n_z = \rho\pi R^2 \delta z$ which predicts a uniform profile as seen in figure 6.

On the other hand, vertex direction is a unitary vector and is uniformly distributed over $[-1, 1]$ in each direction for sufficiently large events. The uniform distribution is visible

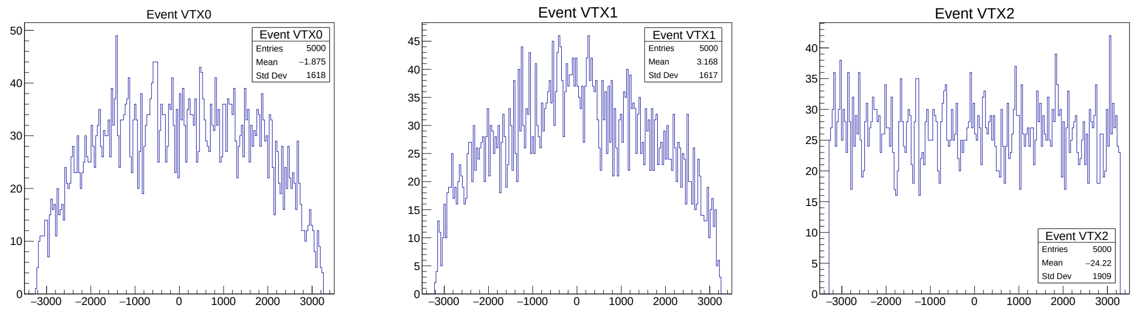

FIGURE 6: Position histograms for a given sample. Left: True vertex position for binning in x-direction. Middle: True vertex position for binning in y-direction. Right: True vertex position for binning in z-direction. (x-axes are in cm, y-axes are counts.)

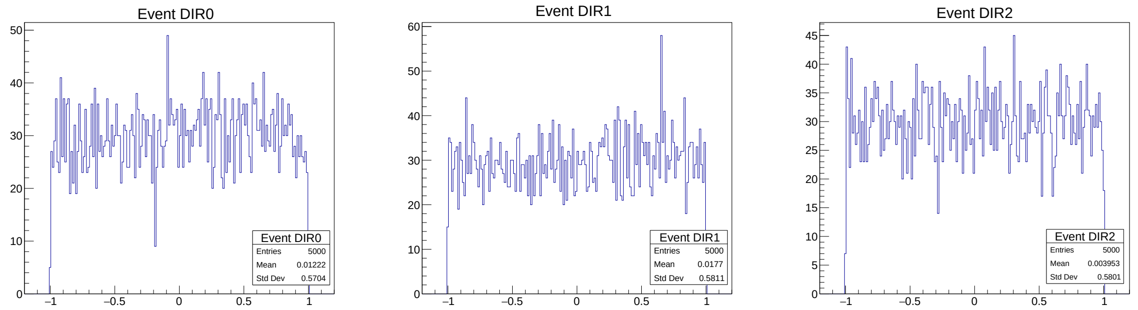

FIGURE 7: Direction histograms for a given sample. Left: True direction for binning in x-direction. Middle: True direction for binning in y-direction. Right: True direction for binning in z-direction. (x-axes are magnitudes of components of the direction vector, y-axes are counts.)

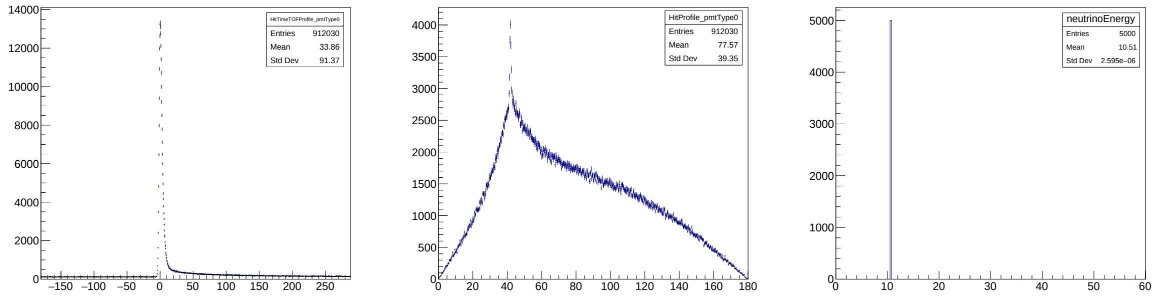

FIGURE 8: Other histograms for a given sample. Left: ToF-corrected hittime profile. (x-axis is in ns, y-axis is counts.) Middle: Hit profile. (x-axis is in degrees, y-axis is counts.) Right: Energy spectrum. (x-axis is in MeV, y-axis is counts.)

in fig. 7. Lastly, we have some additional plots shown in fig. 8, which is fed into spline generating code (see fig. 12). In the time-of-flight-corrected (ToF-corrected) hittime profile, we see that the majority of hits outside the $[-50, 50]$ range are dark hits and observe different counts in this region. In the hit profile (i.e., number of hits recorded at given angular bins), as our electrons are ultrarelativistic, we see a significant peak at

$\sim 42°$. This indeed corresponds to

$$\cos\theta = \frac{1}{\beta n} \implies \theta = \arccos\frac{1}{1.33\times 1} \approx 41° \tag{III.5}$$

Therefore, we mathematically verify that this peak corresponds to the Cherenkov light emitted. The last plot in fig. 8 shows the energy spectrum. As we generated fixed-energy events, a single peak at

$$E_{tot} = E_T + m_e c^2 = 10\text{ MeV } + 0.51\text{ MeV } = 10.51\text{ MeV} \tag{III.6}$$

is observed.

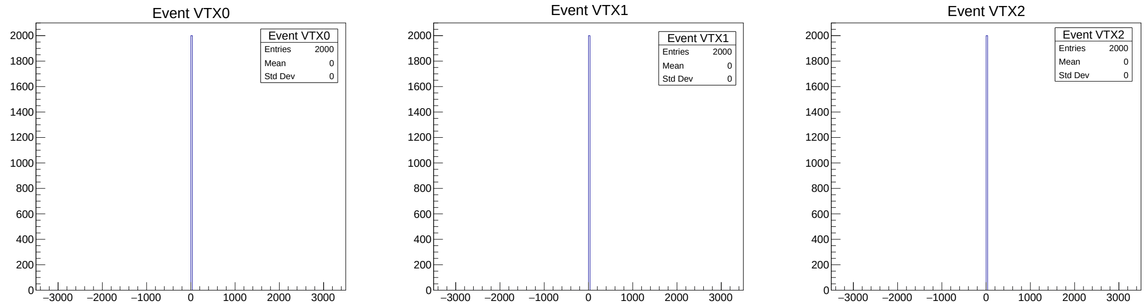

FIGURE 9: Position histograms for a given sample. Left: True vertex position for binning in x-direction. Middle: True vertex position for binning in y-direction. Right: True vertex position for binning in z-direction. (x-axes are in cm, y-axes are counts.)

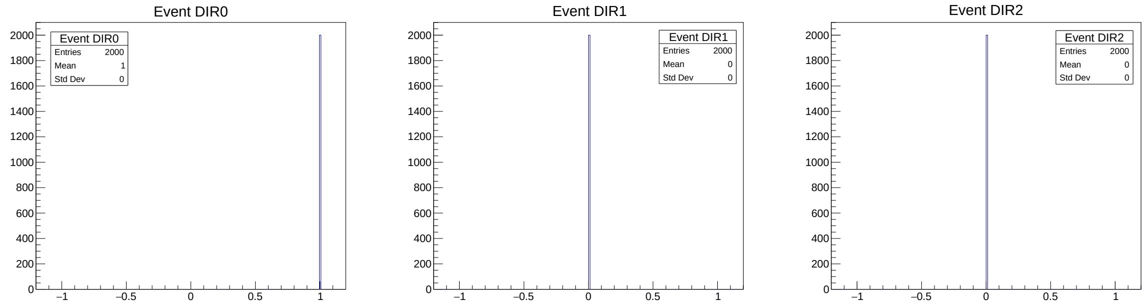

FIGURE 10: Direction histograms for a given sample. Left: True direction for binning in x-direction. Middle: True direction for binning in y-direction. Right: True direction for binning in z-direction. (x-axes are magnitudes of components of the direction vector, y-axes are counts.)

2) Fixed position, fixed direction, Gaussian energy profile: In this case, all the electrons are generated at the center of the tank. Therefore, the position distributions in fig. 9 are all single-peaked at 0. Similarly, directions at y and z are peaked at 0, and the x-components of directions are always equal to 1 (fig. 10). This is expected since the electrons are directed purely to the x axis. When we look at the fig. 11, we see that the shape of the ToF-corrected hittime profile is the same as in the previous case. Therefore, we deduce that ToF-corrected hittime does not depend on positional distributions or energy. This is indeed true since ToF-corrected hittime is calculated as,

$$\text{hittime}_{\text{ToF-corr}} = \underbrace{\text{real hittime}}_{\text{Time by the hit-PMT}} - \underbrace{\text{triggertime}}_{\text{global trigger time}} - \underbrace{\text{ToF}}_{=\frac{\text{vertex pos}-\text{hit PMT pos}}{c/n}} \tag{III.7}$$

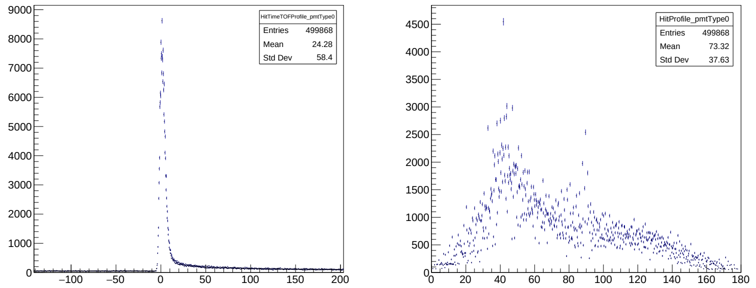

Figure 11: Other histograms for a given sample. Left: ToF-corrected hittime profile. (x-axis is in ns, y-axis is counts.) Middle: Hit profile. (x-axis is in degrees, y-axis is counts.) Right: Energy spectrum. (x-axis is in MeV, y-axis is counts.)

Therefore, ToF-correction on the hittime corrects the delay due to vertex position. Secondly, we check for the hit profile. Even if the peak at 42 degrees is visible, the resolution of the hitprofile is drastically reduced in the non-mono-energetic case. Therefore, we will restrict our analysis to mono-energetic simulation samples. This reduction can be justified by considering that our hitprofile in the fig. 11 is the superposition of smooth hitprofiles for different energies (as in fig. 8). Furthermore, we will later study the energy dependence of darkhits and its effect on the hitprofile in section IV.2. The raw energy plots show a Gaussian energy profile in fig. 11 centered at 20.5 MeV with 2 MeV standart deviation.

IV Reconstruction

Before going into the details of LEAF, let us introduce the method for using LEAF. As mentioned in section III, we generate MC samples with WCSim and use an additional code to generate plots of the true event information. Later, we use a pdf-generating code to produce normalized splines based on the plots of true events. Then, we reconstruct the events by LEAF and compare the fit results with the true information. To visualize the resolutions, we use a plot-generating analysis code.

The LEAF algorithm works as follows:

1. In the reconstruction-running code (analysis), we call the `MakeFit` function of LEAF. This is the main function of the entire fit process.

2. `MakeFit` creates a position list of candidates for coarse-grid. The position list has 300^4 candidates in the (x, y, z, t) format such that,

$$t \in [-50, 50]\,\text{ns}, \ (x, y) \in [0, R]^2, \ z \in [-H, H] \tag{IV.1}$$

3. Then, it creates a direction list of 64800 candidates in (θ, φ) format. To avoid the degeneracy between three components of the direction (since $\boldsymbol{d}$ is a unit vector), we generate two angle-grids, one for zenith angle $\theta \in [0, \pi]$ and the other for the azimuthal angle $\varphi \in [0, 2\pi]$. Then, for the fitting and output computations, we map

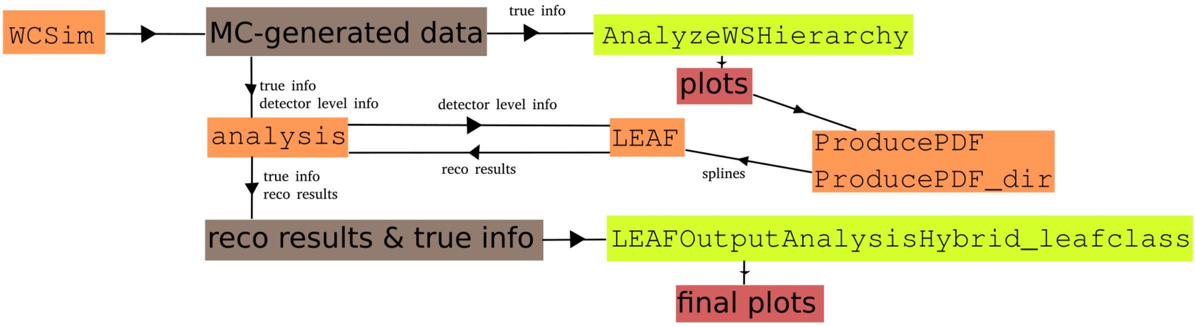

FIGURE 12: General map to demonstrate the usage of several files in reconstruction studies. MC events are generated and plotted. With plots, splines are prepared and probability density functions are prepared. With these splines, we perform LEAF reconstruction and re-plot the reconstructed events compared with the corresponding true event information. Orange: Core codes. Yellow: Analysis codes. Brown: Root files with only numeric info. Red: Root files with visual info.

each (θ, φ) couple to $\boldsymbol{d} = (dx, dy, dz)$ by,

$$\begin{aligned}(\theta, \varphi) &\longmapsto (\sin\theta\cos\varphi, \sin\theta\sin\varphi, \cos\theta)\\ (dx, dy, dz) &\longmapsto \left(\text{arctan2}\frac{dy}{dx}, \arccos dz\right)\end{aligned} \tag{IV.2}$$

where, the arctan2 is the quadrant-sensitive arctan. It computes the angle and finds the correct quadrant based on the values of dy and dx.

4. It declares the output variables and initializes them to 0.

5. Coarse-grid search for position is started. It is implemented in a multi-threaded way. To do so, it uses three functions: `SearchVertex_Main`, `SearchVertex_Callthread`, and `SearchVertex_thread`. However, the candidate score is calculated in the `_thread` function. Then, `_Main` function returns the best 60 candidate positions.

6. Best 60 candidates are passed to the minimization step. It is also implemented in a multi-threaded way by three functions: `MinimizeVertex_Main`, `MinimizeVertex_Callthread`, and `MinimizeVertex_thread`. The likelihood optimization is performed in the `_thread` function. Then, `_Main` function returns the best candidate position.

7. Coarse-grid search for direction is started. It is implemented in a multi-threaded way. To do so, it uses three functions: `SearchDIR_Main`, `SearchDIR_Callthread`, and `SearchDIR_thread`. These functions require a vertex position as input for computation. The candidate score is computed in the `_thread` function, and `_Main` function returns the best 60 candidate directions.

8. Best 60 candidates are passed to the minimization step. It is also implemented in a multi-threaded way by three functions: `MinimizeDIR_Main`, `MinimizeDIR_Callthread`, and `MinimizeDIR_thread`. These functions require a vertex position as input for computation. The likelihood optimization is performed in the `_thread` function. Then, `_Main` function takes the best candidate direction for the given vertex position.

9. The position-direction couples with the minimum negative-log-likelihood is filled to the output variable of the `MakeFit`.

10. The output variable is returned in the reconstruction-running algorithm.

11. The same process repeats for all the primary events (i.e., detected electrons).

The details of `_thread` functions, i.e. the entire log-likelihood optimization, and final plots will be discussed in corresponding sections.

IV.1 Position reconstruction

As the baseline, we will first study the position reconstruction. We will develop an understanding of likelihood-related sections of the LEAF algorithm and comment on the resolutions for different samples.

In the coarse-grid search, due to the high number of candidates and the fact that the majority of these candidates are unlikely to happen, we use a simple check instead of a thorough optimization. For this task, we will compute the following:

1. `SearchVertex_thread` takes the list of all candidates. The position and hit information is passed to `FindNLL_NoLikelihood` for each candidate.

2. In this function, for a given candidate, we loop over all the hits. For each hit, we first find the pmtType and hittime. Then, we compute the distance between the positions of the PMT and the candidate vertex. Knowing this distance, we evaluate time-of-flight. Thus, we compute the residual between ToF-corrected hittime and the time of the candidate vertex as,

$$\text{residual} = \text{hittime}_{\text{ToF-corr}} - t = \text{hittime} - \text{ToF} - t \tag{IV.3}$$

   For the ideal case (i.e., a perfectly reconstructed vertex), the residual should be 0 since the $\text{hittime}_{\text{ToF-corr}}$ corresponds to vertex time t.

3. We check if residual lies between the $[-3, 4]$ ns range. If the condition is satisfied, the score is increased by 1. Then, NLL is computed as $-\log(\text{score})$. If none of the hits passes the range condition, we set NLL to 10^{15} (to infinity, almost unachievable for the other cases).

4. The positions and corresponding NLL values for the candidates are stored in a list and are sorted with respect to NLLs. The first 60 elements of this array are returned, i.e., the 60 candidates with the smallest NLLs. By this function, we choose the candidates with the most meaningful hits.

For the fine-grid search, we define the function `FindNLL_Likelihood` to be fed into Minuit-MIGRAD (see appendix B.2). For each candidate (among the best 60 found by coarse-grid search), we run Minuit to fit four parameters: (x, y, z, t). For each parameter in the fit, we create a parameter grid of $[\text{Candidate position} - 2 \times \text{stepsize}, \text{Candidate position} + 2 \times \text{stepsize}]$, initialized at the Candidate position. The minimum value of `FindNLL_Likelihood` and the optimal parameters found by Minuit are returned. The fine-tuned candidates are sorted with respect to their NLLs, and the first element is returned. Therefore, we get the best-reconstructed position.

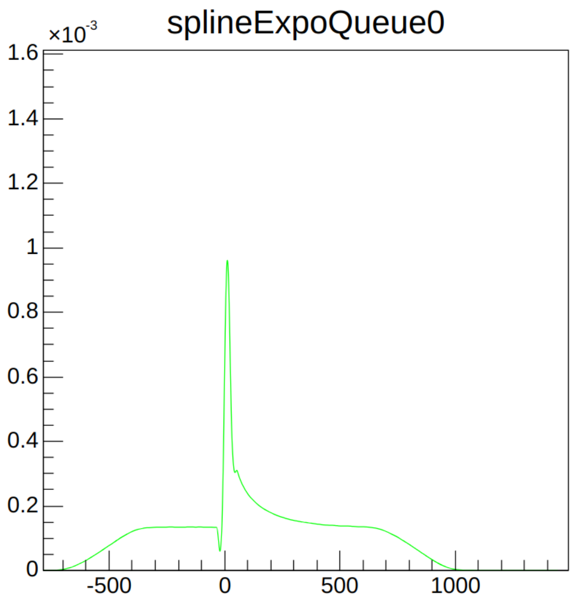

FIGURE 13: Normalized Spline of the TOF-corrected hittime to be used as the probability density function in `FindNLL_Likelihood`. (x-axis is time in ns, y-axis is the probability.)

The function `FindNLL_Likelihood` is written as follows:

1. NLL is initialized to 0.
2. For given position parameters by Minuit, we loop over all the hits. For each hit, we first find the pmtType and hittime. Then, we compute the distance between the PMT position and the candidate vertex proposed by the Minuit. Knowing this distance, we evaluate time-of-flight. Thus, we compute the residual between ToF-corrected hittime and the time of the candidate vertex as,

$$\text{residual} = \text{hittime}_{\text{ToF-corr}} - t = \text{hittime} - \text{ToF} - t \tag{IV.4}$$

3. We initialize the probability to 0 and evaluate the probability for the given residual by the probability density function generated by the normalized spline of the Hittime (fig. 13). If the probability is zero or below (small negative numbers can appear as probability due to interpolation of the spline), we set the probability to an infinitesimal positive value, 10^{-20}.
4. The value of negative-log-likelihood is increased by $-\log(\text{proba})$.
5. After the loop is finished, NLL is returned.

The implementation is as follows:

```
double LEAF::FindNLL_Likelihood(std::vector<double> vertexPosition, int nhits, double lowerLimit, double upperLimit){

  double NLL = 0;

  for(int ihit = 0; ihit < nhits; ihit++){
    Hit lHit = fHitCollection->At(ihit);

    int iPMT = lHit.PMT;
    double hitTime = (fTimeCorrection + lHit.T) / TimeDelta::ns;
    PMTInfo lPMTInfo = (*fPMTList)[iPMT];

    int pmtType = Astro_GetPMTType(iPMT);
    double distance = Astro_GetDistance(lPMTInfo.Position,vertexPosition);

    double tof = distance / fLightSpeed;
    double residual = hitTime - tof - vertexPosition[3];
    double proba = 0;

    bool condition;

    condition = residual > lowerLimit && residual < upperLimit;

    if(condition){
      proba = fSplineTimePDFQueue[pmtType]->Eval(residual);
    }else{
      proba=0;
    }

    if(proba<0){
      std::cout<<"Error␣in␣PDF"<<std::endl;
      if(proba>-1e-1) proba=0; //Since spline sometimes slightly goes below 0 due to interpolation
      else{
        std::cout <<"Error␣in␣"<< residual << "ns␣where␣proba␣=␣" << proba << std::endl;
        return 0;
      }
    }

    if(proba==0) proba=1e-20;
    NLL += -TMath::Log(proba);
  }

  return NLL;
}
```

Let us begin to see the results of the position reconstruction with our baseline energy, 10 MeV. Fig. 14 shows that there is no bias for any of the components of the vertex position. Thus, for other energies, it is sufficient to compare the full vertex resolution instead of component-by-component comparison.

The resolution plots for several energies are given in fig. 15. Vertex resolution increases as the energy gets higher. This variation is expected because the number of hits increases for higher energies.

One last check might be the change in resolution with respect to `dwall` and `towall`.

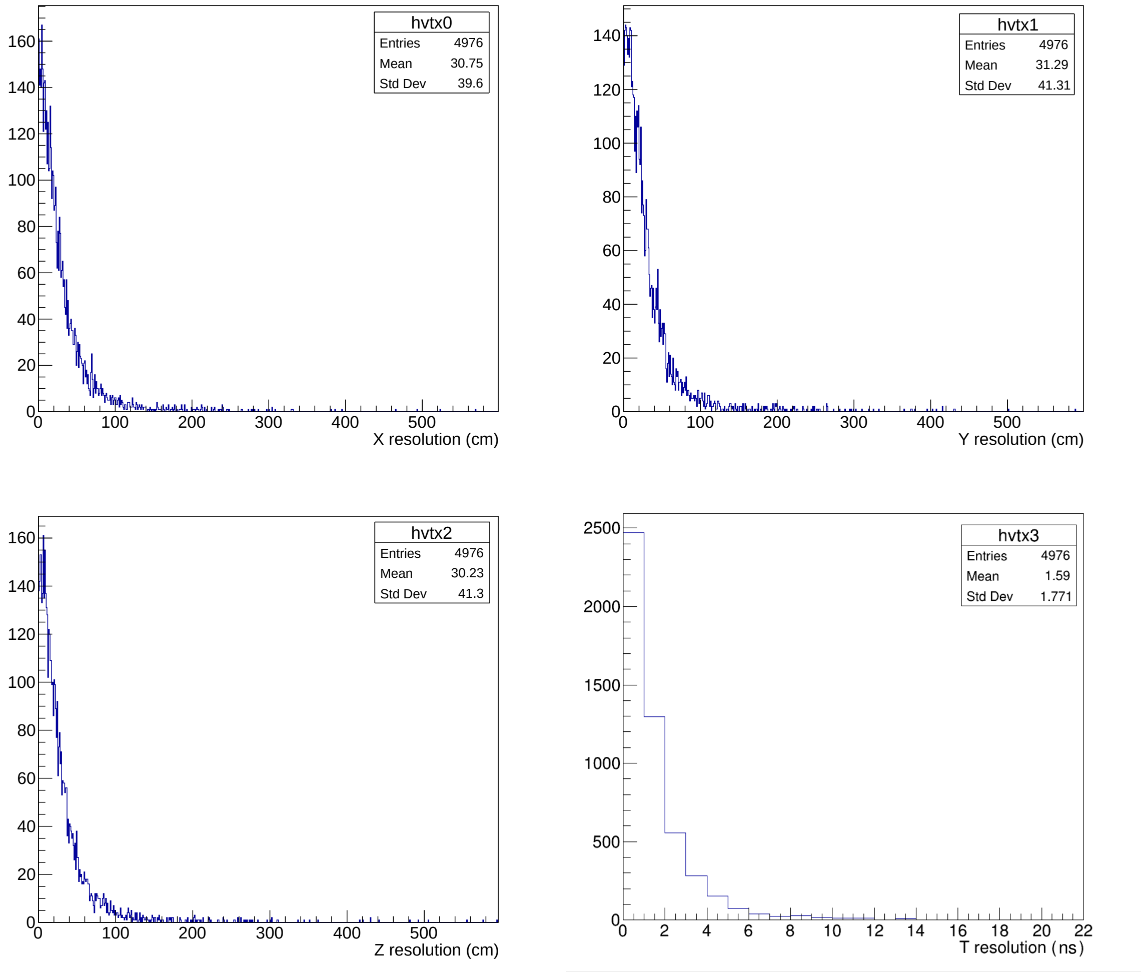

FIGURE 14: Resolution plots for vertex position fits: $|\text{leaf}_i - \text{true}_i|$. Left-top: x-axis. Right-top: y-axis. Left-bottom: z-axis. Right-bottom: t-axis.

`dwall` is the shortest distance from the vertex to the wall, and `towall` is the distance to the wall from the vertex along the direction. The plots are shown in fig. 16. `towall` binning shows an important observation about reconstruction. Vertex resolution drops down as the events are located further from the wall. However, as the electrons positioned closer to wall and directed to the wall at the same time, vertex resolution is also dropped significantly down due to insufficient number of hits recorded.

IV.2 Direction reconstruction

For the direction reconstruction, we have to consider a few details. First, as mentioned earlier, we need position information to reconstruct direction. Therefore, we have to break down this task into three parts:

1. Only-direction fit: Feed the reconstruction algorithm with the true positions and perform only the direction fit. It is important for the validation of the reconstruction algorithm.

2. Sequential fit: Fit the position using LEAF. Then, use the reconstructed position to fit the direction. This would be the first complete version of LEAF-direction fit.

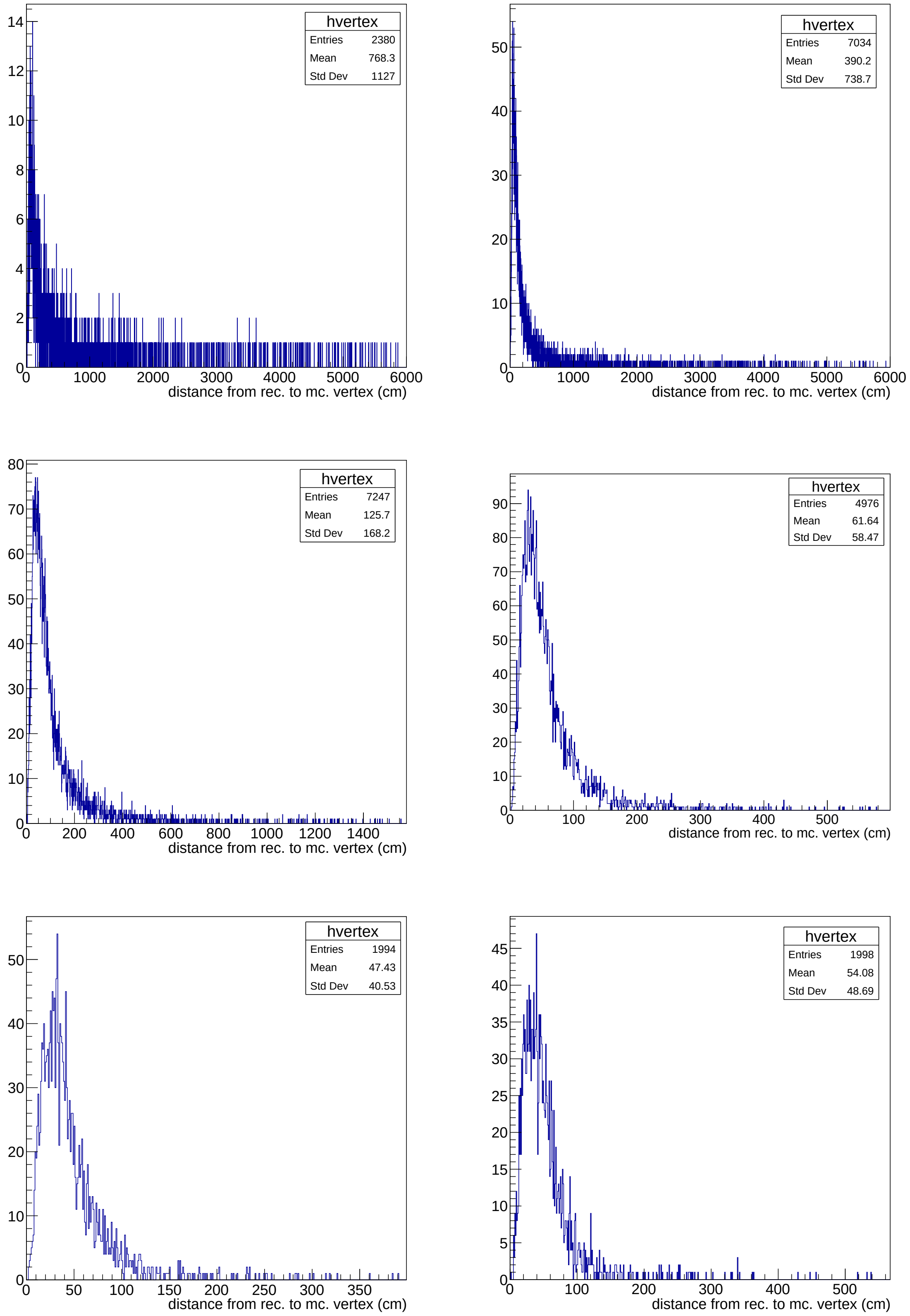

FIGURE 15: Resolution plots for different energies: $\sqrt{\sum_{i\in\{x,y,z,t\}} |\text{leaf}_i - \text{true}_i|^2}$. Top-left: 2 MeV. Top-right: 3 MeV Middle-left: 5 MeV. Middle-right: 10 MeV. Bottom-left: 20 MeV. Bottom-right: 50 MeV.

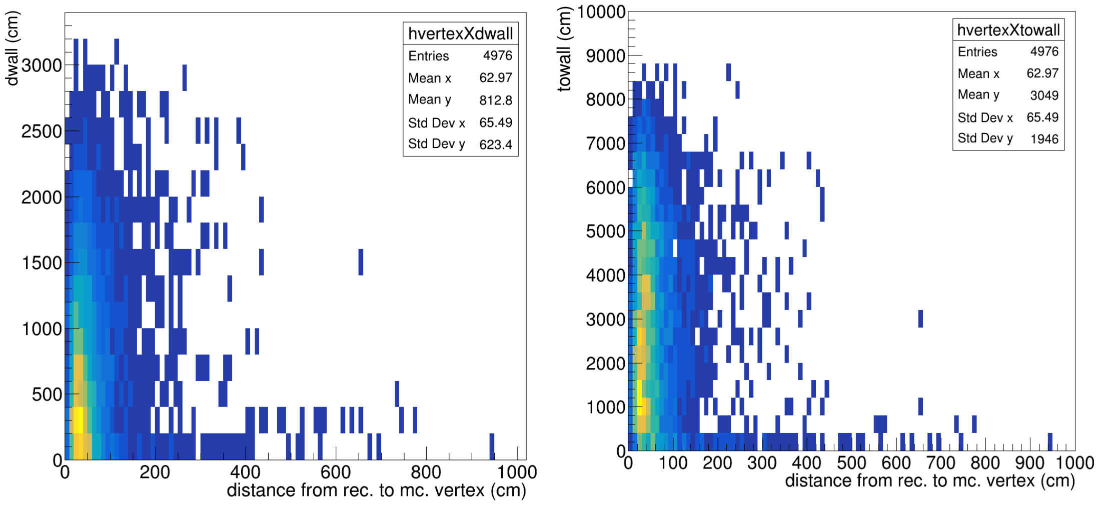

FIGURE 16: Resolution plots with binnings in `dwall` (left) and `towall` (right).

3. Simultaneous fit: Fit both the position and the direction at the same time. Use 8-parameter grids for coarse-grid and Minuit steps. This would be the final version of the LEAF-direction method.

During this thesis, the first two methods are implemented. However, the results did not reach the expectations for the only-direction step. For the only-direction fit, several NLL computation methods and splines are tested. After introducing the algorithm and fit results for some cases, future work and possible ideas for improving the direction resolution will be shared.

Another important detail for the direction reconstruction is choosing the proper probability density function. Finding the center of the Cherenkov ring is one of the best ways to determine direction. However, a direct reconstruction of the Cherenkov ring is challenging due to complex geometry, long computation times of mathematical transforms (such as the Hough transform), and inhomogeneity of hits. On the other hand, computing the opening angle of the Cherenkov light, i.e., the cone half-angle can be done without any long mathematical computations and is independent of the azimuthal distribution of hits. We have two strong candidates for the opening angle fit: *(i)* the charge profile and *(ii)* the hit profile. Fig 17 shows that these profiles heavily depend on the energy of electrons and the presence of darkhits. Furthermore, fig. 11 shows that the hit profile for non-mono-energetic events is not a clear profile. Figs. 18 and 19 indicate that using hitprofile is better due to less relative change for different energies.

The algorithm will be explained for the only-direction fit framework. In the coarse-grid search, due to the high number of candidates and the fact that the majority of these candidates are unlikely to happen, we use a simple check instead of a thorough optimization. For this task, we will compute the following:

1. `SearchDIR_thread` takes the list of all candidates. For each candidate, spherical angles and hit information are passed to `FindNLL_dir_NoLikelihood`. The true vertex is provided for computations. In the function, the spherical angles (referred to as candidate angles) are converted to Cartesian direction vector $\boldsymbol{d}$ (referred to as candidate direction) with the transformation given in eq. (IV.2).

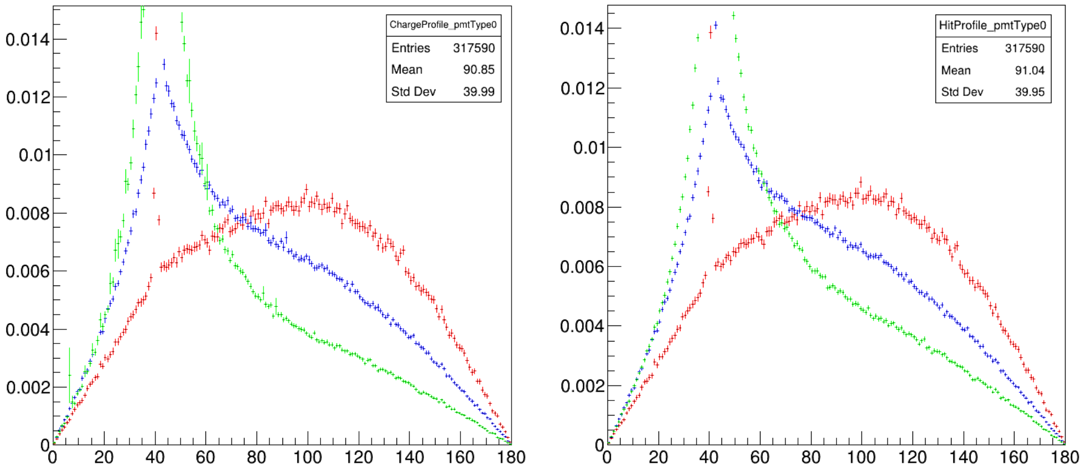

Figure 17: Normalized charge (left) and hit (right) profiles with respect to different energies. Red: 2 MeV, Blue: 10 MeV, Green: 50 MeV. (x-axis is in degrees, y-axis is the probability.)

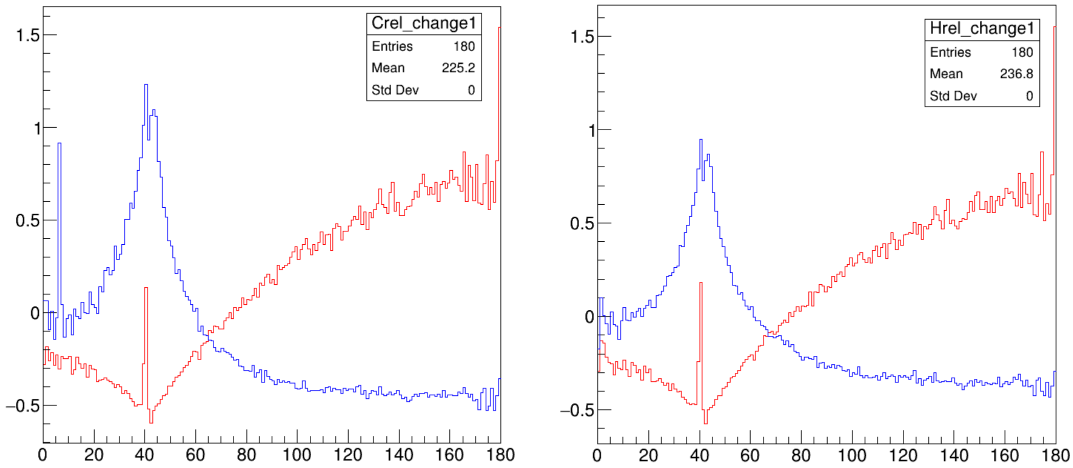

Figure 18: Relative change plots for charge (left) and hit (right) profiles with respect to different energies. Red: $\frac{2MeV-10MeV}{10MeV}$, Blue: $\frac{50MeV-10MeV}{10MeV}$. ($x$-axis is in degrees.)

2. For the given candidate, we loop over all the hits. For each hit, we first find the pmtType and hittime. Then, we compute the unitary vector that points the hit PMT position from the vertex position $\boldsymbol{d}_{to\,pmt}$ (referred to as Vertex-to-PMT vector). Then, we compute the angle between the candidate direction and the Vertex-to-PMT vector,
$$\theta_{opening} = \arccos(\boldsymbol{d} \cdot \boldsymbol{d}_{to\,pmt}) \tag{IV.5}$$
For the ideal case (i.e., a perfectly reconstructed vertex), the opening angle must be close to 42° since the opening angle corresponds to the half-angle of the Cherenkov cone. However, dark hits create a significant background for the hitprofile.

3. We check if the angle lies between the $[38, 44]\,^\circ$ range. If the condition is satisfied, the score is increased by 1. After the loop, the NLL value is calculated as $-\log(\text{score})$. If none of the hits passes the range condition, we set NLL to 10^{15} (to infinity, almost unachievable for the other cases).

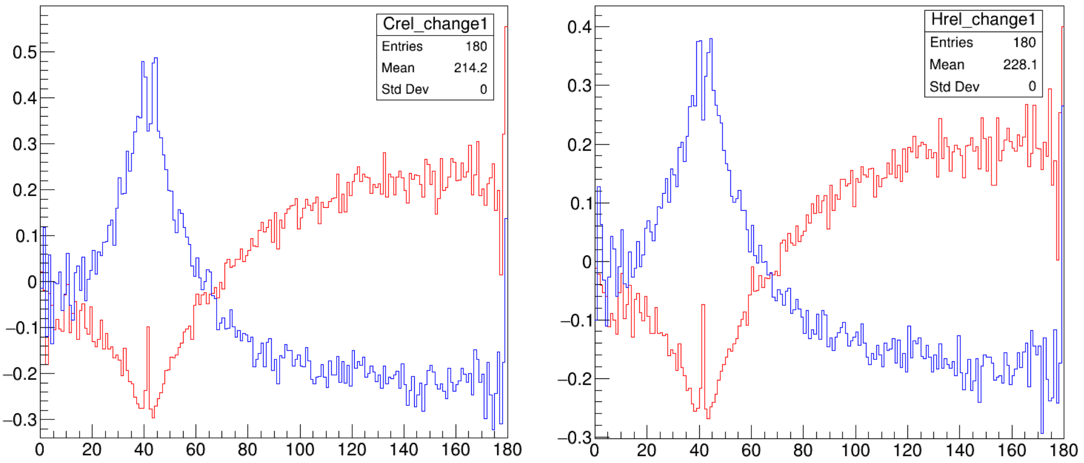

FIGURE 19: Relative change plots for charge (left) and hit (right) profiles with respect to different energies. Red: $\frac{5MeV-10MeV}{10MeV}$, Blue: $\frac{20MeV-10MeV}{10MeV}$. ($x$-axis is in degrees.)

4. The spherical angles and corresponding NLL values for the candidates are stored in a list and are sorted with respect to NLLs. The first 60 elements of this array are returned, i.e., the 60 candidates with the smallest NLLs. By this function, we choose the candidates with the most meaningful hits.

For the fine-grid search, we define the function `FindNLL_dir_Likelihood` to be fed into Minuit-MIGRAD (see appendix B.2). For each candidate (among the best 60 found by coarse-grid search), we run Minuit to fit two parameters: (θ, φ). For each parameter in the fit, we create a parameter grid of $[\text{Candidate angle} - 1°, \text{Candidate angle} + 1°]$, initialized at the Candidate angle. The minimum value of `FindNLL_dir_Likelihood` and the optimal parameters found by Minuit are returned. The fine-tuned candidates are sorted with respect to their NLLs, and the first element is returned. Therefore, we get the best reconstructed spherical angles. In the output of the LEAF code, we convert these spherical angles to direction vectors as given in equation (IV.2).

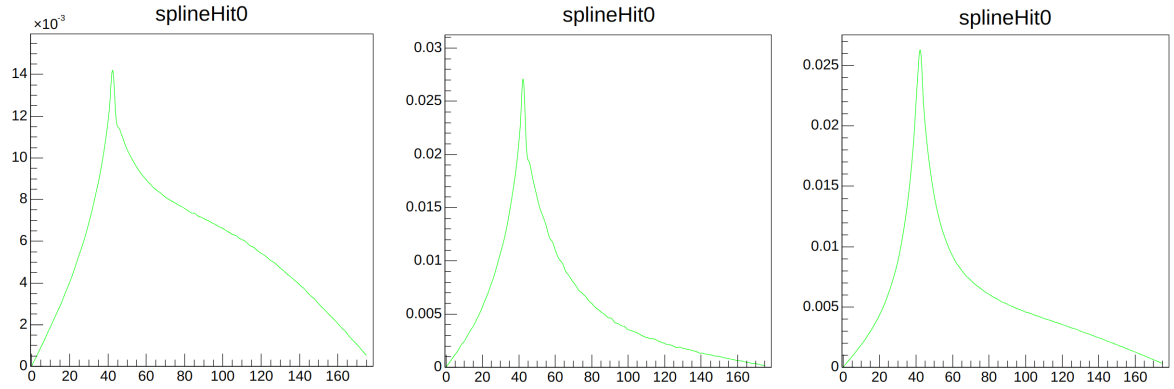

FIGURE 20: Normalized Spline of the HitProfile to be used as the probability density function in `FindNLL_dir_Likelihood`. Left: Baseline PDF generated by 10 MeV events with darkhits. Middle: PDF generated by 10 MeV events without darkhits. Right: PDF generated by 50 MeV events with darkhits. (x-axes are in degrees, y-axes are the probabilities)

The function `FindNLL_dir_Likelihood` is written as follows:

1. NLL is initialized to 0.
2. For given position parameters by Minuit, we loop over all the hits. For each hit, we first find the pmtType and hittime. Then, we compute the vector that points PMT position from the true vertex position $\boldsymbol{d}_{to\,pmt}$. Knowing this vector, we compute the opening angle of the candidate direction (computed from the candidate spherical angles proposed by the Minuit) $\boldsymbol{d}$ as,

$$\theta_{opening} = \arccos(\boldsymbol{d} \cdot \boldsymbol{d}_{to\,pmt}) \tag{IV.6}$$

3. We initialize the probability to 0 and evaluate the probability for given $\theta_{opening}$ by the probability density function generated by the normalized spline of the Hit-profile (fig. 20). If the probability is zero or below (small negative numbers can appear as probability due to interpolation of the spline), we set the probability to an infinitesimal positive value, 10^{-20}.
4. The value of negative-log-likelihood is increased by $-\log(\text{proba})$.
5. After the loop is finished, NLL is returned.

The implementation is as follows:

```
double LEAF::FindNLL_dir_Likelihood(std::vector<double> vertexPosition, std::vector<double> vertexAngles, int nhits){
  double NLL_dir = 0;

  std::vector<double> vertexDirection(3, 0.);
  vertexDirection[0] = TMath::Sin(vertexAngles[0]) * TMath::Cos(vertexAngles[1]);
  vertexDirection[1] = TMath::Sin(vertexAngles[0]) * TMath::Sin(vertexAngles[1]);
  vertexDirection[2] = TMath::Cos(vertexAngles[0]);

  for(int ihit = 0; ihit < nhits; ihit++){
    Hit lHit = fHitCollection->At(ihit);

    int iPMT = lHit.PMT;
    int pmtType = Astro_GetPMTType(iPMT);

    std::vector<double> dpmt = VectorVertexPMT(vertexPosition, iPMT);

    double d_scalar = 0.;
    for(int i = 0; i<3; i++) d_scalar += dpmt[i]*vertexDirection[i];

    double theta = TMath::ACos(d_scalar) * 180. / TMath::Pi();

    double proba = 0.;

    proba = fSplineDIR_50[pmtType]->Eval(theta);

    if(proba<=0) proba=1e-20;
    NLL_dir += -1.*TMath::Log(proba);
  }

  return NLL_dir;
}
```

The direction resolution for the first reconstruction attempt is given in fig. 21. It is clear to see that the reconstruction algorithm did not work well. There might be several reasons for this failure:

- PDF-related problems
- NLL-related problems
- Minuit-related problems

Checking the Minuit performance is the most trivial one. For this purpose, one can skip the coarse-grid search and directly feed the Minuit with the true direction information as starting parameters and widen the parameter grid (to $[0, 180]^\circ$, in our case). The result for such only-Minuit search gave the resolution given in fig. 22. This plot shows that fine-grid search indeed works as desired, and Minuit can reconstruct events well if good candidates are provided. This also means that the Minuit cannot really swipe over the entire parameter grid and find results close to the starting parameters, even if the parameter grid covers the entire parameter space. This also highlights the necessity to have a good-performing coarse-grid search.

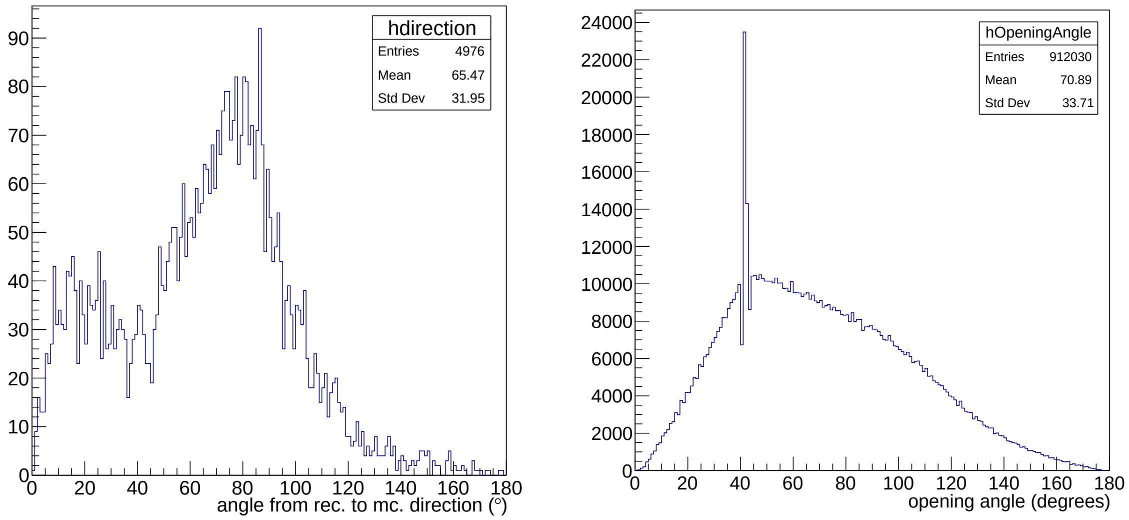

Figure 21: Left: Direction resolution for the first reconstruction. It is computed as the angle between the reconstructed and the true directions. Right: Opening angle profile for the best-reconstructed vertex. It should match (at least resemble) the spline given in fig. 20.

To check if any problems with loading the spline and reading the information exist, one can plot the opening angles between the best-reconstructed direction and the vectors pointing to the hits (see fig. 21). After this check, we also ensure that our splines are loaded and processed correctly.

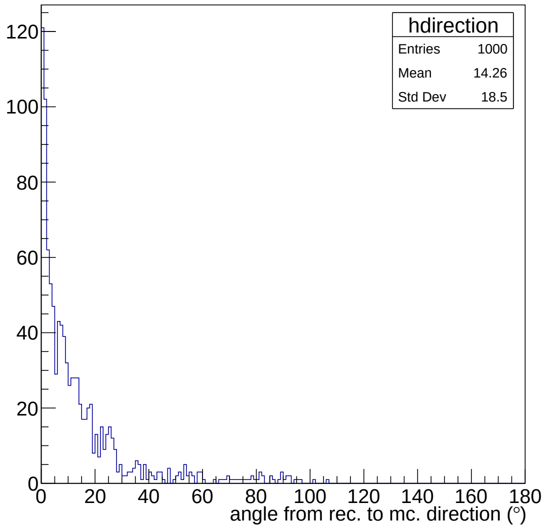

FIGURE 22: Direction resolution for fit results obtained by only-Minuit. The entire parameter space is fed and the true vertex position is set as starting point.

Therefore, the problem is reduced to an issue related to NLL computation in the coarse-grid search. We increased the algorithm's verbosity and implemented more implemented print lines. Furthermore, numerous reconstruction attempts with different likelihood (and no-likelihood) computation methods, splines, and datasets are made. The NLL (likelihood and no-likelihood) computation methods described earlier in this report are the ones that performed the best with minimal need for extra tuning. Other tested NLL candidates will be mentioned in appendix A.2. In the rest of this section, we will use the earlier-introduced NLL method. Further investigation of our NLL methods showed that the coarse-grid search fails in the presence of dark hits. To see this effect, we considered three cases,

1. 10 MeV events without darkhits. Achieving good resolution in this case is the first mission.
2. 50 MeV events with darkhits. This is a middle step since the effect of darkhits is small (but present) due to the higher number of hits from the Cherenkov cone in higher energies.
3. 10 MeV events with darkhits. Fully-functional Reconstruction.

and three different PDFs shown in fig. 20.

Figures 21, 24, 25, 27 show that darkhits in the detector-level-information significantly reduces the reconstruction performance. As the darkhits are less dominant, the secondary peak at 80° disappears. Furthermore, the opening angle distributions indicate that better resolution is achieved for PDFs, which are more peaky at 42°. Intuitively, this leads us to two main requirements of NLL minimization:

1. The PDF needs to prioritize hits close to the cone angle and penalize the darkhits. Such a PDF fits opening angles close to 42°. However, it also increases the tendency to consider darkhits in the opening angle optimization.

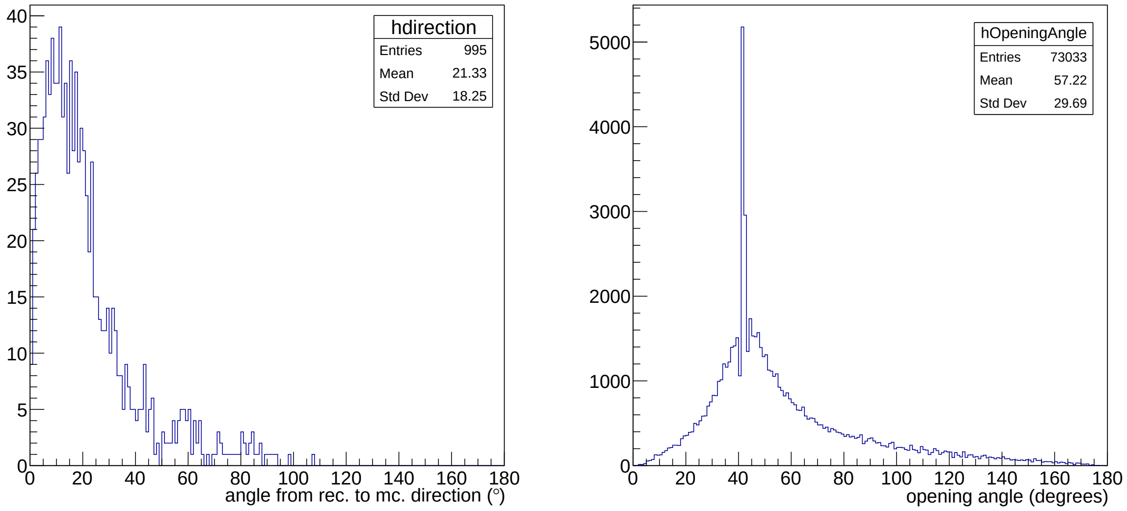

Figure 23: Fit results for 10 MeV events without darkhits. The PDF in the fit is also generated by the 10 MeV events without darkhits. Left: Direction resolution for the first reconstruction. Right: Opening angle profile for the best-reconstructed vertex. (y-axes are counts.)

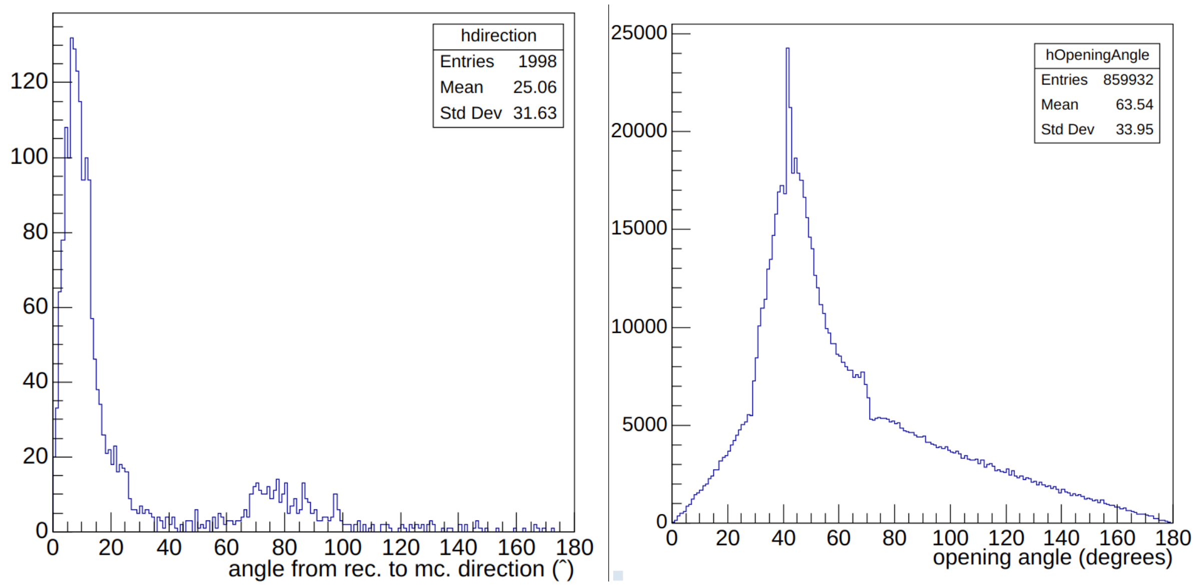

Figure 24: Fit results for 50 MeV events with darkhits. The PDF in the fit is generated by the 10 MeV events without darkhits. Left: Direction resolution for the first reconstruction. Right: Opening angle profile for the best-reconstructed vertex. (y-axes are counts.)

2. The fitting algorithm has to separate darkhits and pass only the meaningful hits. It can be done with a cut-based approach or several signal/background identification tasks. Timing cuts on hits or angular cuts are some strong candidates for this task.

One last remark is to check if the direction reconstruction has any bias in particular directions. To do so, we will check the relative differences of direction components. As shown in figs. 28 and 29, all the reconstructed results are centered at 0 and form a Gaussian. We deduce that no bias is present.

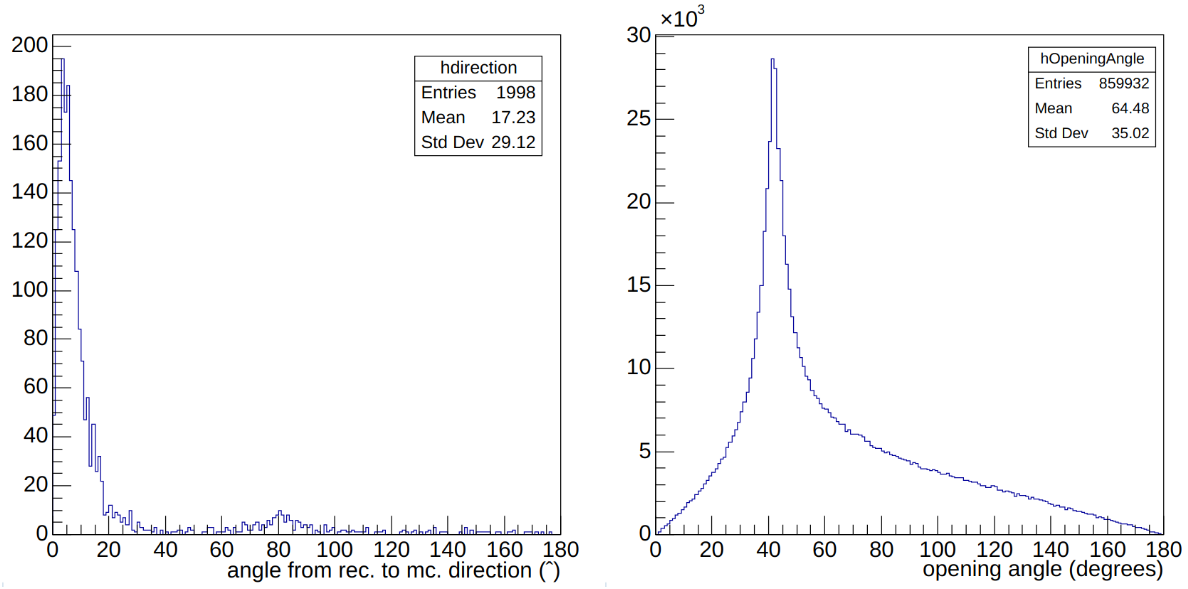

FIGURE 25: Fit results for 50 MeV events with darkhits. The PDF in the fit is also generated by the 50 MeV events with darkhits. Left: Direction resolution for the first reconstruction. Right: Opening angle profile for the best-reconstructed vertex. (y-axes are counts.)

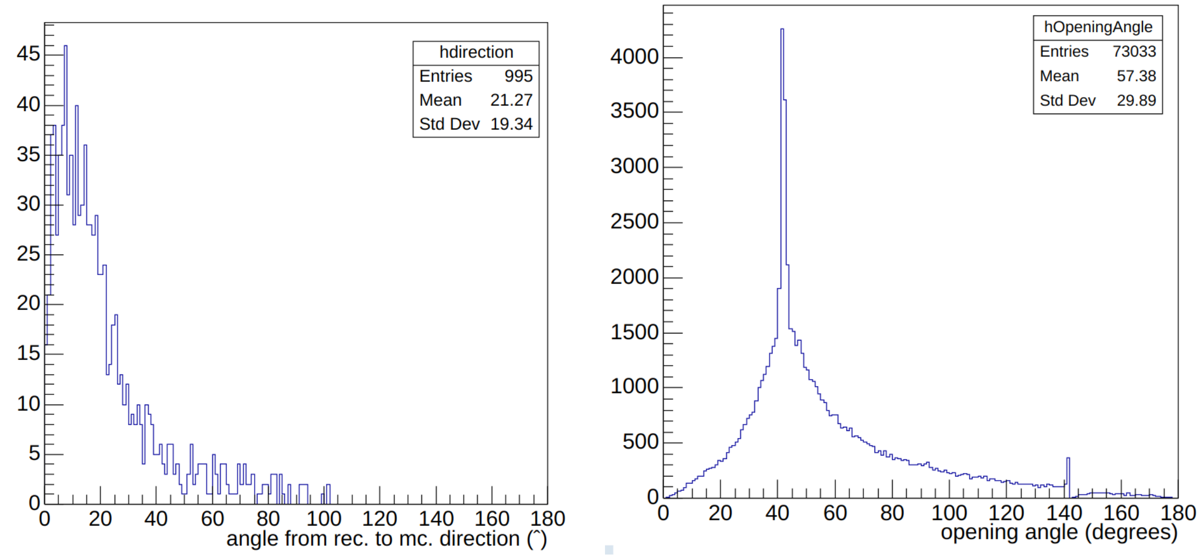

FIGURE 26: Fit results for 10 MeV events without darkhits. The PDF in the fit is generated by the 10 MeV events with darkhits. Left: Direction resolution for the first reconstruction. Right: Opening angle profile for the best-reconstructed vertex. (y-axes are counts.)

IV.3 Future work on LEAF

Even if some background reduction is done and direction resolution is improved from $\sim 80°$ (fig. 21) to $\sim 20°$ (tab. 5), the secondary peak due to dark hits is still a problem. Furthermore, there are several optimizations to be done for direction reconstruction. Even after this, to have a fully functional LEAF, the LEAF-position and LEAF-direction fits have to be performed simultaneously. The LEAF has the potential to handle several

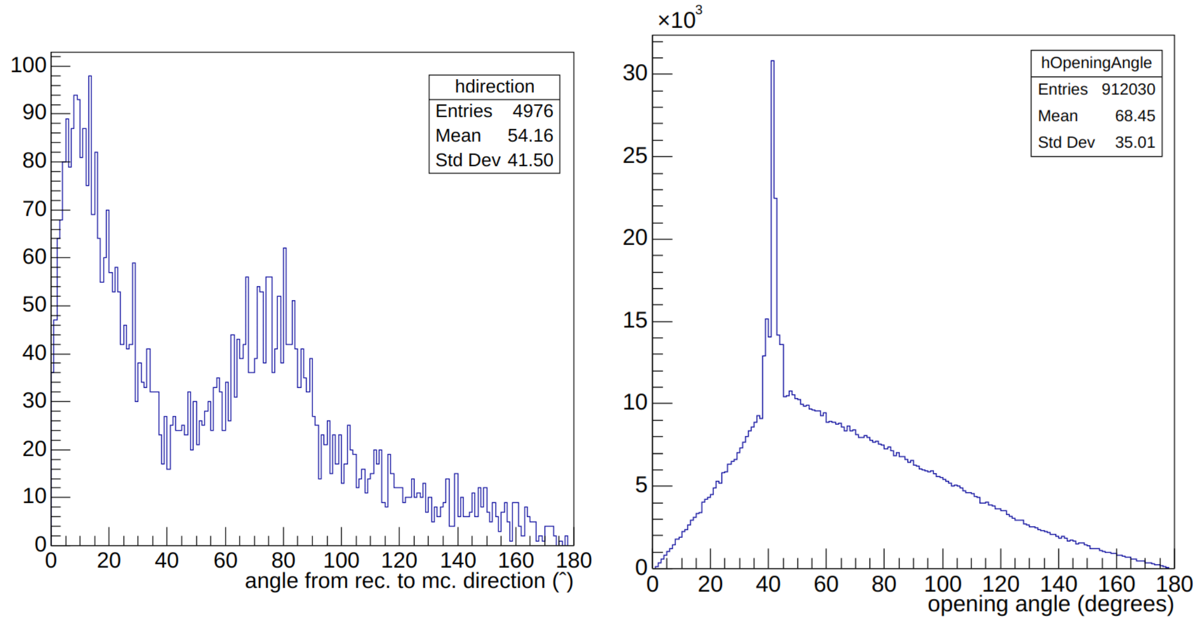

Figure 27: Fit results for 10 MeV events with darkhits. The PDF in the fit is also generated by the 10 MeV events with darkhits. Left: Direction resolution for the first reconstruction. Right: Opening angle profile for the best-reconstructed vertex. (y-axes are counts.)

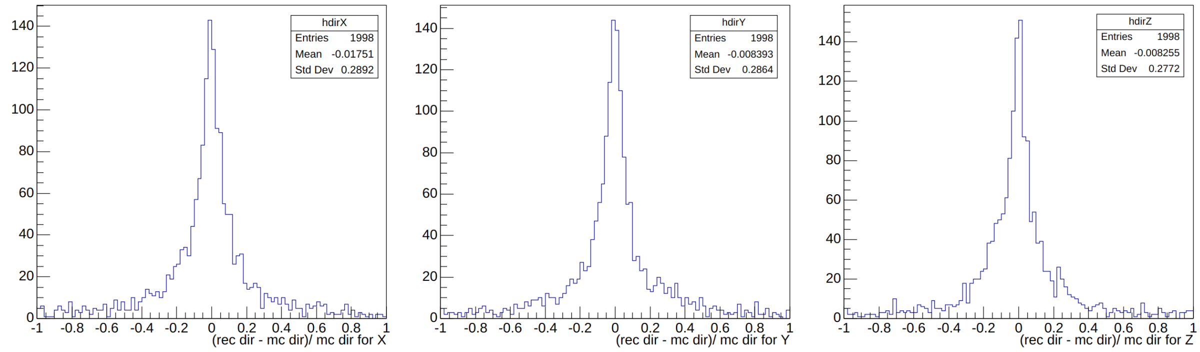

Figure 28: Relative differences of direction components for the fit shown in 25. Left: x-axis. Middle: y-axis. Right: z-axis.

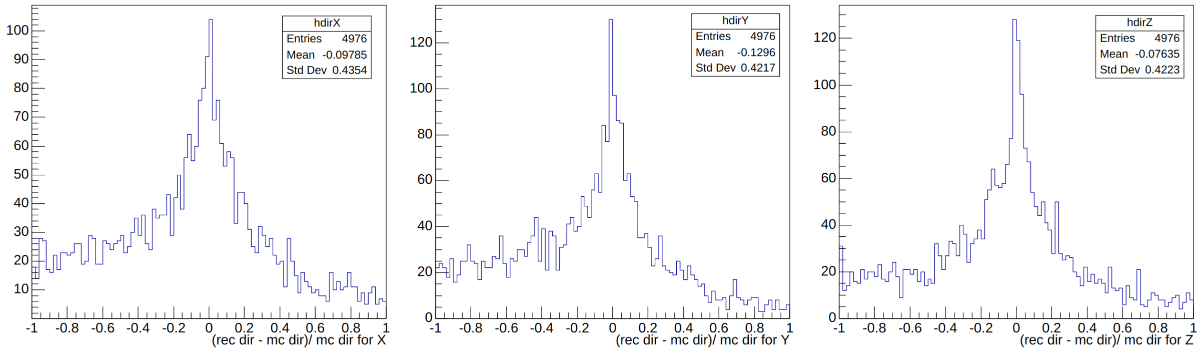

Figure 29: Relative differences of direction components for the fit shown in 27. Left: x-axis. Middle: y-axis. Right: z-axis.

reconstruction tasks but requires much more testing and optimization. Here, we provide a roadmap for future development of LEAF:

- Darkhit reduction in direction reconstruction.
- Modification of NLLs for higher performance.
  - Timing cuts on hits.
  - Cuts on opening angle.
- Testing of sequential fits. It is implemented but not tested.
- Implementation of simultaneous fits. High runtimes are expected. Thus, an optimization of the algorithm would be needed.
- Implementation of energy fit. Testing only-energy, sequential, and simultaneous fits.
- Generation of 2D splines for hitprofiles. Usage of 2D PDFs in the likelihood. Therefore, a more precise energy-sensitive direction reconstruction is expected.

V Conclusion

Performances for the position fit by the old version of LEAF (https://github.com/hyperk/LEAF/) are retrieved. Compared with BONSAI resolutions [15], it is found that LEAF-position resolutions are comparable with BONSAI (see tab. 4). Better resolutions for positions are expected for the sequential and simultaneous fits with the direction information.

Energy Algorithm	> 10 MeV	< 10 MeV
BONSAI	~ 50 cm	~ 160 cm
LEAF	~ 50 cm	~ 125 cm

Table 4: Vertex position resolutions for BONSAI [15] and LEAF 15.

For direction fit analysis, 10 MeV is selected as the baseline. The direction-fitting algorithm is implemented and tested. The energy dependency of the PDFs is tested. The resolution did not reach the expected values (see tab. 5). Several tests are made with different NLL computation methods, and the 6° angular cut on the likelihood performed the best. The algorithm is also tested on events without darkhits and 50 MeV events (where the darkhits become less significant). It is observed that the algorithm fails to interpret darkhits. Several suggestions are made in section IV.3.

Fit data PDF sample	10 MeV w/ dark	50 MeV w/ dark	10 MeV w/o dark
10 MeV w/ dark	$\sim 40°$	–	$\sim 20°$
50 MeV w/ dark	–	$\sim 15°$	$\sim 25°$
10 MeV w/o dark	–	–	$\sim 20°$

Table 5: Vertex direction resolutions for LEAF.

A Physical and Mathematical Appendices

A.1 Negative-log-likelihoods

Let us make a brief review of probability theory and likelihoods. Here, the concepts are introduced in a more general manner than our specific NLL.

The likelihood function is the joint probability density of data as a function of the parameters of a statistical model. In maximum likelihood estimation, the $\arg\max_\theta$ of the likelihood function serves as a point estimate for θ.

For example, consider a random variable X. Given a probability density $x \mapsto f(x|\theta)$, the likelihood function is defined as $\mathcal{L}(\theta|x)\colon \theta \mapsto f(x|\theta)$. Following this definition, one can prove, for an absolutely continuous PDF,

$$\arg\max_\theta \mathcal{L}(\theta|x_j) = \arg\max_\theta f(x_j|\theta) \tag{A.1}$$

and thus, the maximization of the PDF at x_j corresponds to maximizing the likelihood of the specific observation x_j.

However, from the computational perspective, it is easier to implement a minimization than a maximization. Also, dealing with big numbers is inefficient for memory usage and runtime. Therefore, we do the following:

$$\begin{aligned} &\mathcal{L}(\theta|x_1,\ldots,x_n) = \mathbb{P}_\theta(X=x_1)\cdots\mathbb{P}_\theta(X=x_n) \\ &\implies \ln\mathcal{L}(\theta|x_1,\ldots,x_n) = \sum_{i=1}^{n}\ln\mathbb{P}_\theta(X=x_i) \\ &\implies NLL = -\ln\mathcal{L}(\theta|x_1,\ldots,x_n) = -\sum_{i=1}^{n}\ln\mathbb{P}_\theta(X=x_i) \end{aligned} \tag{A.2}$$

Then, instead of maximization, we perform

$$\arg\min_\theta NLL = \arg\max_\theta \mathcal{L}(\theta|x_1,\ldots,x_n) = \arg\max_\theta \prod_{i=1}^{n}\mathbb{P}_\theta(X=x_i) \tag{A.3}$$

which is known as the negative-log-likelihood minimization.

A.2 Other NLL computation candidates

Some of the tested candidates. NLLs are modified by angular cuts and angle-sensitive coefficients (e.g., $\frac{1}{|\theta-42°|}$). Both the resolution plots and the modified NLLs are shown in table 6.

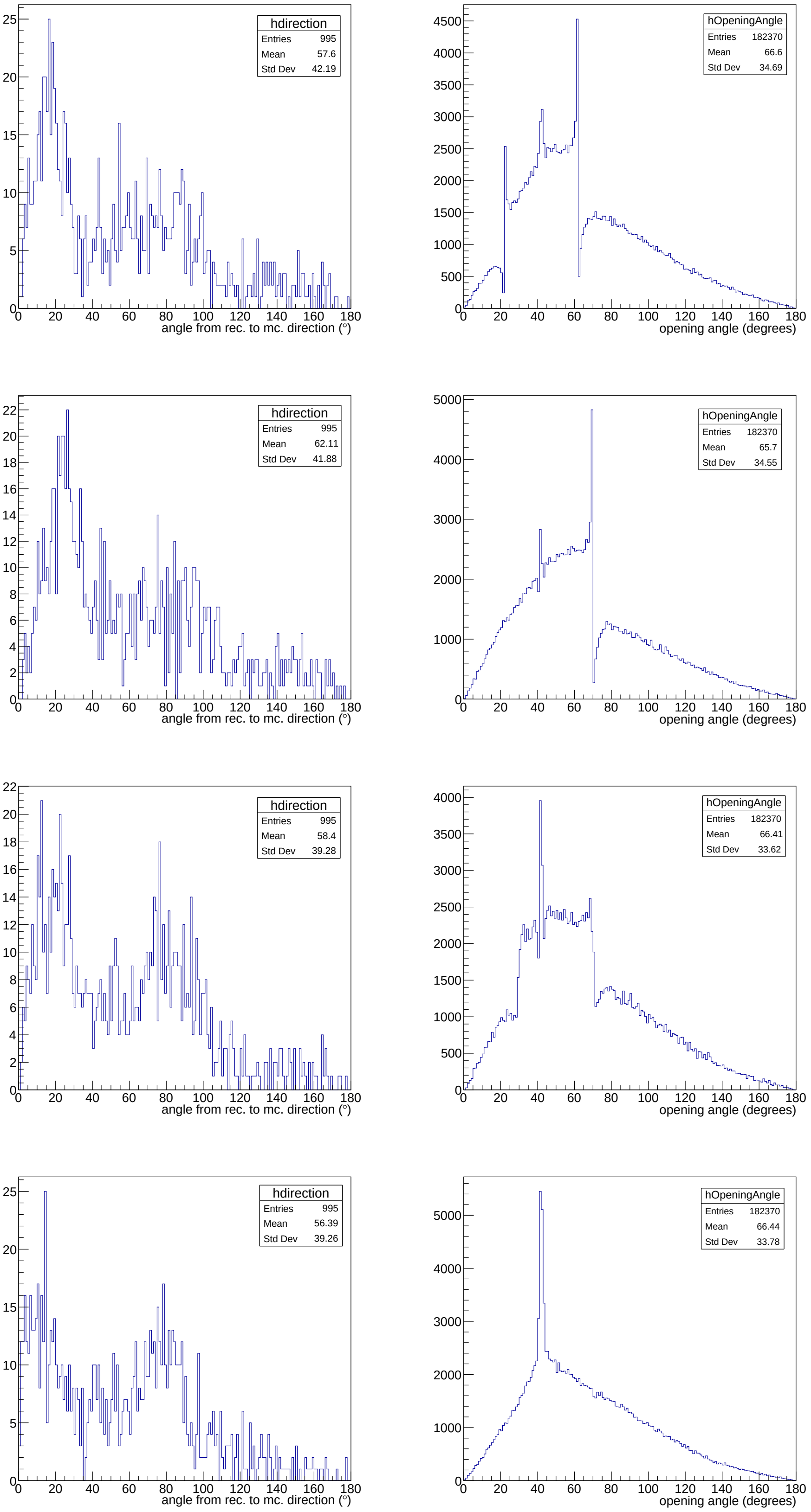

Table 6: Direction resolution plots (left) for different NLLs computation methods (right).

B Programming Appendices

B.1 WCSim macro card example

Sample setup macro with no visualization. It generates 5000 electrons with 10 MeV energy, randomly positioned inside the tank and random in direction. Darkrate is 4.2 kHz, and quantum efficiency is applied.

```
/run/verbose 1
/tracking/verbose 0
/hits/verbose 0
/grdm/verbose 0

# GEOMETRY
/WCSim/SetPMTPercentCoverage  20.2150576375662
/WCSim/SetPMTPercentCoverage2 0.
/WCSim/WCgeom HyperK_HybridmPMT
/WCSim/Construct

# MISCELLANEOUS
/WCSim/PMTQEMethod     SensitiveDetector_Only
/WCSim/PMTCollEff on
/WCSim/SavePi0 false

# DIGITISER & TRIGGER
/DAQ/Digitizer SKI
/DAQ/Trigger NDigits
/control/execute macros/daq.mac

# DARK NOISE
/DarkRate/SetDetectorElement tank
/DarkRate/SetDarkRate 4.2 kHz
/DarkRate/SetDetectorElement tank
/DarkRate/SetDarkMode 1
/DarkRate/SetDarkWindow 4000

# VISUALISATION
/Tracking/fractionOpticalPhotonsToDraw 0.0

# PHYSICS
/mygen/generator gps
/gps/particle e-

# Position: a cylinder
/gps/pos/type Volume
/gps/pos/shape Cylinder
/gps/pos/halfz 32.8755 m
/gps/pos/radius 32.4 m

# Direction: a sphere
/gps/ang/type iso

```

```
# Energy: monoNRG
/gps/ene/type Mono
/gps/ene/mono 10 MeV

## Needed to prevent other decay than Bi-214
/grdm/nucleusLimits 214 214 83 83

# OUTPUT
/WCSimIO/SaveRooTracker 0
/WCSimIO/RootFile RND/10MeV/wcsim_output_e10_rnd.root

# NUMBER OF EVENTS TO RUN
/run/beamOn 5000
```

B.2 Fine-grid fit tools: Minuit and `MIGRAD`

Minuit is a tool to optimize multi-parameter functions and analyze the shape of the function around the extrema. The Minuit package acts on a multi-parameter Fortran function, defined and supplied by the user. The value of this function usually depends on several parameters, and the Minuit determines optimal values for those parameters according to the requested values of the function. One of the most common user-requested values is minimization. In our scenario, we define a function that finds the NLL, and we aim to determine the parameters for candidate positions/directions. Using Minuit commands, we request Minuit to minimize our NLL with respect to the parameters.

The principal application is foreseen for statistical analysis, working on chi-square or log-likelihood functions, to compute the best-fit parameter values and uncertainties, including correlations between the parameters. On the other hand, Minuit is not intended for the repeated solution of identically parametrized problems (such as track fitting in a detector), where a specialized program will be much more efficient. Thus, we do not use the Minuit for coarse-grid fits.

Minimizing and analyzing parameter errors for all possible user functions with any number of variable parameters is challenging and requires some assumptions to avoid evaluating the function for all possible points in the parameter grid. In our specific case, we will provide the minimization strategy to the Minuit as `MIGRAD`.

`MIGRAD` is a variable-metric method with inexact line search, a stable metric updating scheme, and positive-definiteness checks. It is a suitable (and maybe the best) minimizer for almost all user-defined functions. Its main weakness is that it depends heavily on knowledge of the first derivatives and fails if they are inaccurate. If the first derivatives have issues, they can be computed by the user and fed to Minuit. The user can also try to improve the accuracy of Minuits numerical approximation by adjusting values using the `SET EPS` and `SET STRATEGY` commands. For example, the `MIGRAD` will run faster if you `SET STRATEGY 0` and will be more reliable if you `SET STRATEGY 2`.

One should visit [16] for further details. Now let us show an example usage of Minuit:

```
TFitter * minimizer = new TFitter(7); //nb de params
TMinuit * minuit = minimizer->GetMinuit();
```

```

double arglist[20];
int err=0;
double p1 = verbose-1;
inimizer->ExecuteCommand("SET␣PRINTOUT",&p1,1); //quiet mode
minuit->SetErrorDef(1);
if(VERBOSE<2){
    minuit->mnexcm("SET␣NOWarnings",0,0,err);
}
arglist[0]=2;
minuit->mnexcm("SET␣STR",arglist,1,err); //Set strategy = 2!!
minimizer->SetFCN(MinuitLikelihood_dir); //here is function to minimize

double Mig[2]={1e6,1e0}; //maxcalls and tolerance

minimizer->SetParameter(0,"vertex0", vertexPos[0], 0.001, vertexPos[0]-1, vertexPos[0]+1);

minimizer->SetParameter(1,"vertex1", vertexPos[1], 0.001, vertexPos[1]-1, vertexPos[1]+1);
minimizer->SetParameter(2,"vertex2", vertexPos[2], 0.001, vertexPos[2]-1, vertexPos[2]+1);
minimizer->SetParameter(3,"vertex3", vertexPos[3], 0.001, vertexPos[3]-1, vertexPos[3]+1);

double pTheta = TMath::ACos(coarseDirCandidates[icand][2]);
double pPhi = TMath::ATan2(coarseDirCandidates[icand][1], coarseDirCandidates[icand][0]);

minimizer->SetParameter(4, "pTheta", pTheta, stepSize, pTheta-limPerStep*stepSize, pTheta+limPerStep*stepSize);
minimizer->SetParameter(5, "pPhi", pPhi, stepSize, pPhi-limPerStep*stepSize, pPhi+limPerStep*stepSize);

minimizer->SetParameter(6,"nhits",nhits,nhits,nhits-1,nhits+1);

minimizer->FixParameter(0);
minimizer->FixParameter(1);
minimizer->FixParameter(2);
minimizer->FixParameter(3);
minimizer->FixParameter(6);

minimizer->ExecuteCommand("MIGRAD",Mig,2);

std::vector<double> tCandAngles(2, 0.);
std::vector<double> tCandRecoVtxDir(3,0.);

tCandAngles[0] = minimizer->GetParameter(4);
tCandAngles[1] = minimizer->GetParameter(5);

```

```
tCandRecoVtxDir[0] = TMath::Sin(tCandAngles[0]) * TMath::Cos(tCandAngles[1]);
tCandRecoVtxDir[1] = TMath::Sin(tCandAngles[0]) * TMath::Sin(tCandAngles[1]);
tCandRecoVtxDir[2] = TMath::Cos(tCandAngles[0]);

struct FitDirection hDir;
hDir.Dir = tCandRecoVtxDir;
hDir.NLL_dir = 0;

// Get minuit
minuit=minimizer->GetMinuit();
double fmin, fedm, errdef;
int npar,nparx,istat;
minuit->mnstat(fmin,fedm,errdef,npar,nparx,istat);

hDir.NLL_dir = fmin;

tDirContainer.push_back(hDir);
```

References

[1] K. Abe et al. (Super-Kamiokande Collaboration). “Solar neutrino measurements using the full data period of Super-Kamiokande-IV”. In: (Feb. 2024). arXiv: 2312.12907 [hep-ex].

[2] M. Mori et al. (Super-Kamiokande Collaboration). “Searching for Supernova Bursts in Super-Kamiokande IV”. In: *The Astrophysical Journal* 938.1 (Oct. 2022), p. 35. DOI: 10.3847/1538-4357/ac8f41. URL: https://dx.doi.org/10.3847/1538-4357/ac8f41.

[3] K. Abe et al. (Super-Kamiokande Collaboration). “Diffuse supernova neutrino background search at Super-Kamiokande”. In: *Phys. Rev. D* 104 (12 Dec. 2021), p. 122002. DOI: 10.1103/PhysRevD.104.122002. URL: https://link.aps.org/doi/10.1103/PhysRevD.104.122002.

[4] K. Abe et al. (Hyper-Kamiokande Proto-Collaboration). *Hyper-Kamiokande Design Report*. 2018. arXiv: 1805.04163 [physics.ins-det].

[5] A. Suzuki et al. “Improvement of 20 in. diameter photomultiplier tubes”. In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 329.1 (1993), pp. 299–313. ISSN: 0168-9002. DOI: https://doi.org/10.1016/0168-9002(93)90949-I. URL: https://www.sciencedirect.com/science/article/pii/016890029390949I.

[6] Masataka Fukugita et al. *Physics and Astrophysics of Neutrinos*. English. 1st ed. Japan: Springer Tokyo, Dec. 2013. ISBN: 978-4-431-67029-2. URL: https://doi.org/10.1007/978-4-431-67029-2.

[7] S. Fukuda et al. "The Super-Kamiokande detector". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 501.2 (2003), pp. 418–462. ISSN: 0168-9002. DOI: https://doi.org/10.1016/S0168-9002(03)00425-X. URL: https://www.sciencedirect.com/science/article/pii/S016890020300425X.

[8] K. Abe et al. "Calibration of the Super-Kamiokande detector". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 737 (Feb. 2014), pp. 253–272. ISSN: 0168-9002. DOI: 10.1016/j.nima.2013.11.081. URL: http://dx.doi.org/10.1016/j.nima.2013.11.081.

[9] K. Abe et al. "Search for solar electron anti-neutrinos due to spin-flavor precession in the Sun with Super-Kamiokande-IV". In: *Astroparticle Physics* 139 (2022), p. 102702. ISSN: 0927-6505. DOI: https://doi.org/10.1016/j.astropartphys.2022.102702. URL: https://www.sciencedirect.com/science/article/pii/S0927650522000159.

[10] Thomas Mueller. *Lecture 3: Neutrino oscillations in matter - Two-flavour approximation - Solar neutrinos*. PHY651P: Neutrino Physics. In: Institut Polytechnique de Paris. Sept. 2023. URL: https://llr.in2p3.fr/~mueller/Enseignement/PHY651P-2023/lecture3.pdf.

[11] Oleksandr Tomalak et al. "Theory of elastic neutrino-electron scattering". In: *Physical Review D* 101.3 (Feb. 2020). ISSN: 2470-0029. DOI: 10.1103/physrevd.101.033006. URL: http://dx.doi.org/10.1103/PhysRevD.101.033006.

[12] John N. Bahcall et al. "New Solar Opacities, Abundances, Helioseismology, and Neutrino Fluxes". In: *The Astrophysical Journal* 621.1 (Jan. 2005), p. L85. DOI: 10.1086/428929. URL: https://dx.doi.org/10.1086/428929.

[13] John F. Beacom. "The Diffuse Supernova Neutrino Background". In: *Annual Review of Nuclear and Particle Science* 60.1 (Nov. 2010), pp. 439–462. ISSN: 1545-4134. DOI: 10.1146/annurev.nucl.010909.083331. URL: http://dx.doi.org/10.1146/annurev.nucl.010909.083331.

[14] Alberto Giampaolo. "À la recherche du fond diffus de neutrinos de supernovas avec Super-Kamiokande : dernières analyses et perspectives pour l'ère du Gdlatest searches and prospects for the Gd era". Theses. Institut Polytechnique de Paris, Jan. 2023. URL: https://theses.hal.science/tel-04268620.

[15] Mahdi Taani. *Neutron Vertex Reconstruction For The Surface Detector*. University of Edinburgh. In: Bonsai Hyper-K PPE Group Meeting. May 2017. URL: https://indico.ph.ed.ac.uk/event/27/contributions/366/attachments/328/364/Bonsai24May17PPE.pdf.

[16] F. James. "MINUIT Function Minimization and Error Analysis: Reference Manual Version 94.1". In: *CERN Program Library Long Writeup D506* (1994). URL: https://inspirehep.net/literature/1258343.

l'X ÉCOLE POLYTECHNIQUE | INSTITUT POLYTECHNIQUE DE PARIS

Statement of Academic Integrity Regarding Plagiarism

I, the undersigned ....Alper TEZCAN.......... [family name, given name(s)], hereby certify on my honor that:

1. The results presented in this report are the product of my own work.

2. I am the original creator of this report.

3. I have not used sources or results from third parties without clearly stating thus and referencing them according to the recommended rules for providing bibliographic information.

Declaration to be copied below:

I hereby declare that this work contains no plagiarized material.

Date 18/03/2024..........

Signature