Searching for Solar Neutrinos and Building an Open-Source Future for Neutrino Astronomy

By

Jeffrey P. Lazar

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy
(Physics)

at the
University of Wisconsin – Madison
2023

Date of final oral examination: October 30, 2023

The dissertation is approved by the following members of the Final Oral Committee:
Francis Halzen, Professor, Physics
Carlos A. Argüelles, Assistant Professor, Physics
A. Baha Balantekin, Professor, Physics
Janet Conrad, Professor, Physics
Albrecht Karle, Professor, Physics
Amy Barger, Professor, Astronomy

Jean remembered the forewoman at the jade factory in China who was asked how you could tell good jade from bad jade. Through an interpreter, and through a megaphone that didn't work, came the reply, 'You look at it and by looking you tell its qualities.'

Julian Barnes

Abstract

Despite its immense success, the Standard Model of particle physics is unable to provide an explanation of the nature of dark matter and the origin of neutrino masses. A desire to jointly explain these two mysteries motivates careful studies of neutrino properties. The goal of this thesis is to explore possible connections of neutrinos to beyond Standard Model physics. In particular, we will focus on searches for high-energy astrophysical neutrinos and develop new tools that undergird these searches.

First, we perform an analysis looking for an excess of neutrinos from the direction of the Sun using data from the IceCube Neutrino Observatory. Such an excess would be a signature of dark matter captured by scattering on solar nuclei and annihilating to Standard Model particles. In addition, we will describe using this same analysis framework to search for the predicted, but yet-unobserved flux of solar atmospheric neutrinos created when cosmic rays interact and produce meson in the thin solar atmosphere. To perform these analysis, we develop the χ`aro`ν package, which simulates the neutrino yields from dark matter annihilation and decay.

Next, we will turn our attention to flavor physics, and discuss how new physics may manifest in the ratio of neutrino flavors at Earth. In particular, we will discuss the importance of tau neutrino identification in understanding the flavor triangle. Then we will introduce the `TauRunner` package which simulates the passage of the highest energy neutrinos through arbitrary media, including previously neglected effects. This new simulation framework will then be applied to simulating ultra-high-mass dark matter in the solar core, in an attempt to evade the solar opacity limit. Finally, we will describe the simulation framework that has been developed for the Tau Air-Shower Mountain-Based Observatory. This proposed, next-generation detector in the Colca Valley of Peru could provide a tau-pure flux of neutrinos in the 1 PeV–100 PeV energy range.

Finally, we will describe the `Prometheus` simulation package, an open-source framework for simulating neutrino telescopes with arbitrary geometries in water and ice. For the first time, this allows for a consistent simulation framework between the global network of neutrino telescopes that is currently being constructed. Furthermore, this allows for the rapid prototyping of new reconstruction and data storage techniques with easy, cross-detector application. We provide two examples of such techniques: a machine-learning-based reconstruction capable of running faster than the trigger rate of neutrino telescopes; and a demonstration of efficiently storing event-level data from neutrino telescopes in quantum memory.

Acknowledgements

I never could have foreseen the route graduate school would take me down. Neither in the places I would wind up, nor the projects I would work on, nor the people I would meet. I am immensely grateful to all who shepherded me down this long, winding path.

I would like to first and foremost thank Francis Halzen for his support throughout graduate school and for providing me with the right combination of liberty and guidance to allow me to find my own voice as a physicist. You always had my back and were able to bail me out of any trouble I may have found myself in, and for that, I owe you an immeasurable amount of gratitude.

I would also like to thank Carlos A. Argüelles, who guided me through the day-to-day challenges of physics research. You taught me that to do good physics, one must be fearless and unrelenting and that before doing anything else, one must wash the floor a bit. Thank you for instilling in me the value of writing one's own tools and questioning the accepted orthodoxy. It has been a great honor to be your mentee, collaborator, and, above all, friend.

I would like to thank the many other physicists who have supported me throughout my research career. Thank you to Szabi Márka, Zsuzsa Márka, and Stefan Countryman who gave me my first taste of high-energy physics research and guided me through graduate school selection. Thank you to Janet Conrad for helping me through the final throes of graduate school. Thank you to Gwenhaël de Wasseige for welcoming me into her group as a postdoctoral researcher.

I would also like to thank all of my friends who have made graduate school. Primary among them is my partner in crime and lifelong friend, Ibrahim Safa. It has been great growing together as physicists. Thank you to all of my physics friends—especially Zach, Dylan, Marjon, Neil, and Praful—whose help and commiseration made graduate tolerable. And thank you to all of my non-physics friends—especially Joe, Sam, Jenna, Alex and Lindsey—for giving me reprieve and understanding when I would disappear for months at a time.

I would also like to thank my family. Thank you to my mom, Kathy, and my dad, Jerry, for supporting me at each step of my education. Thank you to my brothers, Joe and John, for leading by example and teaching me the power of compound interest. Thank you to my nieces and nephews—Dagny, Wilde, Della, and Desmond—for being a constant source of joy.

Finally, thank you, Lauren Chicco, for your unwavering belief, undying love, and unconditional support. I do not deserve these gifts, but it is my greatest joy to have received them.

Contents

Chapter 1

Introduction

How does the Sun shine? This question has surely plagued humans since there were humans to plague, and it is this question that drove a debate two scientific greats of the the Nineteenth Century, Lord Kelvin and Charles Darwin. For Darwin estimated his theory of Natural Selection would need 300,000,000 years to produce the variety of species that had been observed at the time. Lord Kelvin, a believer that the species had come to be through an Almighty, refuted Darwin's theory on the grounds that the Sun could not sustain itself for that long before running out of energy. He argued that the Sun could only sustain itself for 30,000,000 million years if powered by absorbing the gravitational energy of primordial meteorites.

At the time of this debate, quantum mechanics, special relativity, and general relativity had not yet revolutionized our understanding of the natural world, and so nuclear fusion could not have been imagined. It would take contributions from some of the great physicists of the Nineteenth and Twentieth Centuries to realize the potential of this process. Without Marie Skłodowska-Curie's pioneering studies of radium, Albert Einstein's revolutionary realization that mass and energy can be interchanged, and Wolfgang Pauli's daring prediction of a potentially undetectable particle, it would have been impossible to realize that if hydrogen fuses into helium deep in the solar core, the Sun could shine for 30,000,000,000 years. It would take these contributions and more to understand that, if this is the Sun's power source, 10^{38} elusive—but not undetectable—particles would stream out of the Sun every second. And that a tiny fraction of these elusive *neutrinos* may be observed at Earth.

It is this possibility that drove Ray Davis and his team to bury 100,000 gallons of perchloroethylene, a common dry-cleaning ingredient, in the Homestake Mine. The simply-named Homestake Experiment would detect the solar flux of neutrinos by looking for the footprint they leave behind when passing through. A neutrino could capture on a chlorine atom, convert a neutron into a proton, thus creating an argon atom in place of the chlorine. After some time, usually around a month, the experimenters would flush the tank, and count all the argon inside. When measured, the presence of argon proved that nuclear fusion powers the Sun and showed that invoking primordial meteorites is not necessary to explain the Sun's light.

While this was a monumental step forward in understanding the most-studied celestial object, it did open a new question. For, the Homestake Experiment had observed only one third of the argon that John Bahcall, who performed calculations of the solar neutrino flux, had predicted. Over the course of the coming decades the deficit persisted in the face of refined calculations and new experiments, eventually coming to be known as the solar neutrino problem. All these experiments that sought to measure the solar neutrino flux were sensitive only to one particular type, or *flavor*, of neutrino. This was not expected to be an issue as only electron-flavor neutrinos are produced in the nuclear reactions that fuel the Sun, and so these electron-neutrino-detecting experiments should be sensitive to the entire solar neutrino flux.

Given the deficit's persistence, the Sudbury Neutrino Observatory (SNO) was constructed to test the theory that perhaps these previous experiments had not been seeing the entire flux. In addition to the same *charged-current* detection channel that had been used to detect electron neutrinos in the previous experiments, SNO used heavy water to open a *neutral-current* channel that could detect all neutrinos of all flavors. When SNO measured both the electron-only and all-flavor fluxes, they found the previously observed, theory-measurement disagreement in the former channel, but good theory-measurement agreement in the latter channel. These results showed that neutrinos were changing flavors as they travelled from the Sun to Earth. At about the same time, the Super-Kamiokande experiment observed a similar phenomenon in neutrinos produced in cosmic-ray showers. Taken together, these results proved that neutrinos were changing flavor in-flight, a phenomenon commonly called neutrino oscillation. This phenomenon requires neutrinos have non-zero mass, a fact that the Standard Model (SM) of particle physics cannot accommodate.

Armed with an *ad hoc* insertion of neutrinos masses into the SM, physicists set to work characterizing this newly discovered behavior, measuring the amplitude and frequency of oscillations as a function of the energy and propagation distance. Soon enough this yielded further surprises when first the Liquid Scintillator Neutrino Detector and later the MiniBooNE experiment observed an excess of low-energy events in each detector. Yet again, neutrinos produced an excess—this one termed the short-baseline anomaly—that could not be accommodated by the SM; however, this anomaly could be explained by introducing a fourth neutrino that does not interact directly with the SM, a so-called sterile neutrino. The evidence for such a neutrino has grown in recent years, with the data preferring a four-neutrino model over a three-neutrino model at greater than 6σ significance [1]. Even as the three-neutrino model becomes increasingly untenable, the path forward is far from clear. Data from oscillation experiments that probe different neutrino energies and propagation baselines are in 4.9σ tension with each other [1]. While more precise quantum mechanical descriptions of the neutrino production region [2, 3] can relax the tension to the 3.5σ level, there is no clear way to fully resolve the issue.

While we do not know what form the final theory of physics will take, we will need footholds along the way that we can use to anchor searches and guide extensions. The origin of neutrino masses and resolution of the short-baseline anomaly continue to elude us, guiding our efforts even as they beguile us. It seems reasonable to expect that further studies of neutrinos from new sources and at previously unexplored energies and baselines will continue to chip away at mysteries of beyond SM physics. Furthermore, it is always tempting to search for solutions that tie two open questions together, for instance a link between neutrinos and dark matter (DM).

Neutrino astronomy offers the opportunity to pursue this proven program by giving access to neutrinos with supraänthropogenic energies travelling from cosmic distances. The largest, currently operating neutrino telescopes is the IceCube Neutrino Observatory. Located at the geographic South Pole and deployed between 1.45 km and 2.45 km beneath the ice's surface, IceCube comprises a cubic kilometer of ultra-clear ice instrumented with 5,160 light-detecting digital optical modules (DOMs). These DOMs are arranged on 86 vertical strings which each support 60 DOMs. 78 of these strings have 17-m vertical spacing between DOMs and are arranged in a regular, hexagonal lattice with 125-m interstring. The remaining eight strings form a denser, more irregular infill region known as DeepCore. The DOMs detect Cherenkov light emitted by the charged byproducts of neutrino interactions, and the two distinct regions give access to neutrinos with initial energies spanning six order of magnitude from a few GeV to a few PeV. By looking at the timing and spatial patterns of the light in the detector, we are able to infer the neutrinos energy, direction, and flavor.

In the decade since its completion, IceCube has made great strides in the field of neutrino astronomy. Beginning with the discovery of the diffuse astrophysical neutrino flux in 2013 and culminating with strong evidence for emission from the Galactic Plane and from three known, extragalactic gamma-ray emitters in 2023, IceCube has delivered on its promise to open a new window to the Universe. These continued study and characterization of these new sources will enable the previously mentioned oscillation studies.

If the work of the past decade constitutes opening a window, the prospects for the next decade may be more akin to throwing open the flood gates. In addition to planned improvements to the IceCube array to lower its energy threshold and to increase the instrumented volume by a factor of eight, four water-based, northern-hemisphere neutrino telescopes are projected to be completed by 2035. When construction is finished on these next-generation detectors, the instrumented volume of Cherenkov neutrino telescopes will have increased from the 1.5 km^3 it sits at today to 25 km^3. In addition to this rapid expansion in the Cherenkov sector, the next-generation of neutrino telescopes will include new detection principles that push the energy frontier even higher and enable unambiguous measurements of the astrophysical flavor composition.

As we stand at the precipice of a new era in neutrino astronomy, we are undergoing a revolution in data analysis techniques. Machine learning (ML) has already proved an invaluable tool in neutrino astonomy, having played an integral role in IceCube's recent observation of the Galactic Plane and extragalactic point sources; however, it is clear that current techniques can be improved upon. For instance, IceCube recently hosted an open-science challenge through the Kaggle website. In this challenge, participants were tasked with reconstructing the incident direction of the neutrino using simulated events in the IceCube detector. Within two and half weeks of the context opening, a submission had beaten the state-of-the-art IceCube reconstruction. In addition to demonstrating the potential for improving our current methods, this case study shows the power of open science to collectively address current shortcomings.

In this thesis, I will present the work I have carried out as a graduate student under the joint supervision of Francis Halzen and Carlos A. Argüelles. This has included a search for connections between neutrinos and dark matter, simulations for next-generation neutrino telescopes, and open-science reconstructions for the multi-telescope era. In each chapter, I will begin by describing software that I helped develop and proceed to describe the analyses, calculations, and reconstructions these packages enabled.

In Chap. 2, I will describe χ`aro`ν, a package for computing neutrino yields from DM annihilation and decay. I will then report results of a recent search for high-energy neutrinos from the direction of the Sun, and frame these results in the context of DM annihilation and solar atmospheric emission. Next, In Chap. 3, I will describe the `taurunner` package which simulates the passage of extremely high-energy neutrinos through matter. I will then use this package to calculate neutrino yields from DM annihilation in the Sun beyond the solar opacity limit. I will describe the simulation framework for the next-generation TAMBO telescope, in which `TauRunner` plays an important role. Then, in Chap. 4, I will describe the `Prometheus` packages that provides end-to-end simulation for neutrino detectors with photon level output. Finally, in Chap. 5, I will conclude.

But before we set out, a kind warning to the reader. This document is not self-contained, and some prior knowledge is assumed. In particular, I will assume the reader has an understanding of particle physics at the level of the Halzen and Martin textbook [4]. Furthermore, I will assume that the reader is familiar with the statistical techniques of the Particle Data Group [5]. Finally, since I had no role in designing or deploying the IceCube Neutrino Observatory, I will leave detailed discussions of these topics to the IceCube detector paper [6]. With that out of our way, let's go!

Chapter 2

Searches for High-Energy Solar Neutrinos

The existence of (DM) has been verified by measurements probing length scales ranging from the galactic to the cosmological [7, 8, 9]. In the standard ΛCDM model of cosmology, DM comprises about 85% of the Universe's present matter content [9]; however, the SM cannot provide a DM candidate with appropriate interaction strength, abundance, and stability. This shortcoming is perhaps the most significant flaw of the SM.

This shortcoming has prompted a massive effort—both experimental and theoretical—to uncover the nature of DM. Numerous DM candidates have been put forward, but definitive detection has yet eluded us. One generic, frequently considered DM candidate class is weakly-interacting, massive particles (WIMPs). These particles have significant mass and interact with the SM with strengths at or below the weak interaction strength. One may use this small but non-zero cross section to search for WIMPs in three different ways. In direct detection experiments, one looks for excess energy deposited in a calorimeter when WIMPs scatter on SM nuclei. Next, in production searches, one looks for missing transverse momentum in colliders when WIMPs are produced and escape the detector without interacting. Finally, in indirect detection searches, one looks for the stable SM byproducts created when WIMPs annihilate with each other.

Neutrino telescopes play an important role in this last class of searches. Since WIMPs should accumulate in regions with large gravitational potential, one looks for the SM by-products coming from these regions. Since neutrinos interact weakly, they can escape unimpeded from these dense astrophysical environments. This chapter will focus on potential connections between neutrinos and

WIMPs. First, we will describe the χaroν software, which allows the user to simulate the yield of neutrinos from WIMP annihilation and decay. We will then proceed to describe a search for neutrinos from WIMP annihilation in the solar core using data from the IceCube Neutrino Observatory, a search that this software was designed to facilitate.

2.1 Indirect Searches for Dark Matter and the χaroν Package

The χaroν package calculates the neutrino flux from WIMPs annihilating or decaying in the Sun, the Earth, and the Galactic halo. In the name of conciseness, we will describe only the case of annihilation, but this case differs only from that of decay by rescaling the initial energies. Since the interaction of WIMPS with the SM is unknown, we take a pragmatic approach of computing the neutrino spectra for WIMP annihilation to $q\bar{q}$, gg, W^+W^-, Z^0Z^0, HH, l^+l^-, and direct neutrino channels $\nu_\alpha\bar{\nu}_\alpha$; the latter channel is relevant in the case of Kaluza-Klein DM [10]. Here, $q\bar{q}$ are the six quarks, l^+l^- are the three charged leptons, and $\nu_\alpha\bar{\nu}_\alpha$ are the three neutrino flavors. In the case that a model predicts a specific SM branching fraction, one may compute the flux via a weighted average of the χaroν base cases.

In χaroν, we calculate hadronization and final-state radiation of the initial annihilation products either by running Pythia [11, 12] or by using a calculation for high-mass particles [13]. We use the latter case when the energy of the initial state is above the electroweak scale, *i.e.* for WIMPs with masses greater than 250 GeV. After accounting for hadronization and final state radiation, we simulate the interaction of the secondary products with their environment using Monte Carlo methods. The general idea is to recursively simulate both the interaction and decay of the secondary products and weight the neutrino flux from each branch by the associated probability of that branch. In general, this will depend on the particle in question and the environment in which one wishes to propagate that particle. A high-level diagram of this process is shown in Fig. 2.1 and the result of the algorithm is shown in Fig. 2.2 for DM masses below and above the EW scale.

In order to simulate the effect of hadrons, we need to consider the competing effects of the hadron both interacting with the environment and decaying. Quantitatively, we need to consider the relative probabilities of interaction and decay given by:

$$p_{\text{int}} = \frac{\gamma\tau_{\text{dec}}}{\tau_{\text{int}} + \gamma\tau_{\text{dec}}} \text{ and } p_{\text{dec}} = \frac{\tau_{\text{int}}}{\tau_{\text{int}} + \gamma\tau_{\text{dec}}}, \tag{2.1}$$

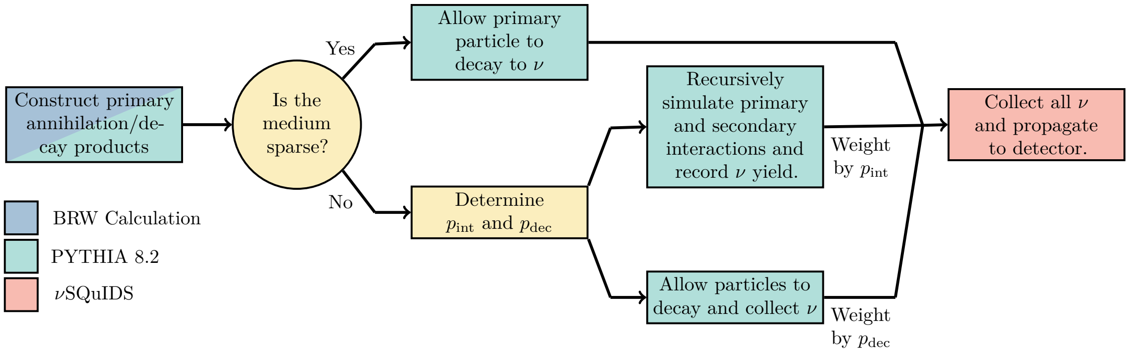

Construct primary annihilation/decay products
Is the medium sparse?
Yes
No
Allow primary particle to decay to ν
Determine $p_{\rm int}$ and $p_{\rm dec}$
Recursively simulate primary and secondary interactions and record ν yield.
Allow particles to decay and collect ν
Weight by $p_{\rm int}$
Weight by $p_{\rm dec}$
Collect all ν and propagate to detector.
BRW Calculation
PYTHIA 8.2
νSQuIDS

Figure 2.1: ***Diagram of code structure.*** Flow chart depicting the major steps in the calculating flux from DM annihilation or decay. The light yellow boxes indicate direct calculation or decision making; other colors indicate the main program used in each step.

where $\tau_{\rm dec}$ is the mean lifetime of the hadron, γ is the relativistic boost factor, and $\tau_{\rm int}$ is the mean interaction time given by [4]:

$$\tau_{\rm int} = (n\sigma v)^{-1} \approx 5.5 \times 10^{-8}\ s\ \times \left(\frac{\rm g/cm^3}{\rho}\right)\left(\frac{\rm mb}{\sigma}\right)\beta^{-1}, \tag{2.2}$$

where ρ is the density of the medium, σ is the hadron-nucleon cross section, and β is the velocity of the hadron normalized to the speed of light.

For most hadrons, one of these terms clearly dominates. For instance, hadrons composed only of up, down, or strange quarks yield a ratio of $p_{\rm int}$ to $p_{\rm dec}$ on the order of 10^4 and 10^3 in the solar and terrestiral cores respectively at $m_\chi = 1$ GeV. These ratios grow to 10^8 and 10^7 at $m_\chi = 1$ TeV. Thus, one can safely assume that such hadrons, namely $\pi^\pm$, $K^\pm$, K_L^0, and neutrons are fully stopped on account of rapid interactions before decaying. At the other extreme, since the top quark lifetime is a small fraction of the interaction time of the strong force, top quarks are unable to form hadrons. Thus, any top quarks undergo on-spot decay in our simulation.

When a hadron containing bottom or charm valence quarks is created in the simulation, interaction and decay must be balanced. In this case, we simulate both interaction and decay. In the case of decay, all products are weighted by $p_{\rm dec}$. In the case of interaction, the hadron is given a new energy $E' = \langle Z_x \rangle E$—where E' is the energy after interacting $\langle Z_x \rangle$ is the average fractional energy transfer, and E is the initial hadron energy—and weighted by $p_{\rm int}$. This may now decay or interact again, and this process is carried out recursively until the hadron has lost all kinetic energy. In this procedure, low-energy neutrinos produced in hadronic showers initiated by the aforementioned hadron are not tracked, and thus we underestimate the low-energy flux; see [14] for a detailed low-energy calculation. We follow the estimates of [15] and set $\langle Z_b \rangle = 0.7$ and $\langle Z_c \rangle = 0.6\frac{m_c}{m_{\rm hadron}}$. χ`aro`ν and `WimpSim` use

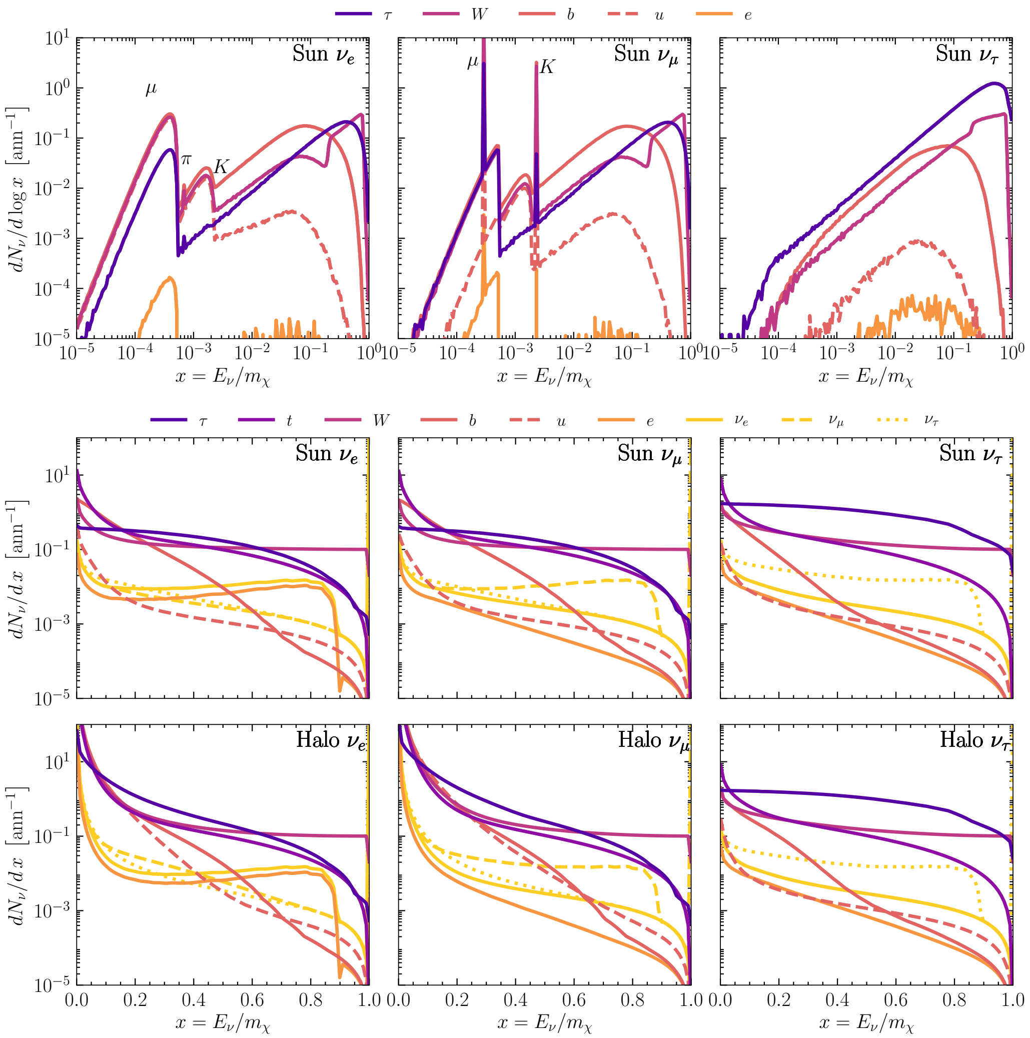


Figure 2.2: ***Differential neutrino flux at production.*** We show spectra for several representative annihilation channels, indicated by different line colors. The top panel shows the spectra for $m_\chi = 100$ GeV in the center of the Sun, generated without EW correction and with decays from stopped particles. Features related to μ, π, and K decays are indicated by their respective symbols. The two-row linear scale panel at the bottom shows spectra for a DM mass of $m_\chi = 1000$ GeV with EW correction and without decays from stopped particles. Different polarization states are averaged over. Due to EW corrections, direct neutrino channels no longer show a two-body decay distribution, but have a low-energy tail. The top row corresponds to the center of the Sun, while the bottom row is for the Galactic Halo. The different columns indicate the neutrino flavor, from left to right: electron, muon, and tau flavored neutrinos. For DM decay, m_χ should be multiplied by a factor of 2 and x be replaced by $x = 2E_\nu/m_\chi$. Quarks not shown in this plot lie between the dashed and solid u and b lines, and differ only by a normalization.

this energy loss estimation method, while PPPC samples from an energy loss distribution; we have checked that these two approaches yield comparable results.

We now turn our attention away from our treatment of hadrons and towards the case of simulating secondary leptons. The rate at which charged leptons lose energy in a plasma, such as the center of the Sun, is given in [16]. We find that the rate of energy loss, $\frac{dE}{dt}$, is larger than 10^9 GeV/s. This gives that the stopping time is bounded by $t_{\text{stop}} = m_\chi / \left(dE/dt\right)$. In the Sun, computing the ratio of t_{stop} to $\gamma\tau_{\text{dec}}$ for charged tau leptons and muons leads to the conclusion that in the relevant energies, the tau leptons will promptly decay, but most muons will stop before decaying.

For annihilations that happen in Earth's core, charged leptons lose energy through ionization, bremsstrahlung, photo-nuclear interaction, and electron pair production, which we estimate using the Bethe-Bloch formula [17]. The interaction length of charged tau leptons in Earth is significantly larger than the decay length up to τ energies of 10^9 GeV [18]. Thus for the WIMP mass range under discussion in this work, one can assume the charged tau leptons decay at production without interacting. Muons are longer lived, and at energies above a few GeV the interaction length is much shorter than the decay length. As in the case of the Sun, they can be assumed to come to rest before decaying.

Long-lived hadrons—$\pi^\pm$, $K^\pm$, K^0_L, and neutrons—are either fully absorbed by matter or decay after being stopped. Among them, π^- and K^- are captured by matter and would form atom-like systems, which prohibits them from further decays [19]. Due to different interactions of K^0 and $\bar{K}^0$ with nucleons, which causes quantum coherence loss, K^0_S are regenerated from K^0_L continuously and followed by hadronic decays [20]. Stopped $\mu^\pm$, π^+, and K^+ subsequently decay at rest. Neutrinos from decay of stopped particles are produced at energies below 1 GeV, thus are not of great importance for current indirect searches. χaroν provides an option to compute these low-energy neutrinos using Pythia. These neutrinos have also been considered in previous calculations, *e.g.* they are included in PPPC using Geant4 [21], a technique first developed in [14]; see also [22, 23, 24].

For DM masses above the EW scale, the neutrino flux generation in χaroν is coupled to a new computation of the EW correction which also accounts for polarization of annihilation and decay states and the evolution of the polarized particles to the EW scale [13]. As EW interactions are not fully implemented in Pythia, this calculation incorporates a more complete consideration such as the missing triple gauge couplings in the EW sector. Polarization and EW correction effects are also implemented in PPPC, which takes a different approach by augmenting leading order EW correction term [25]. Other works have studied the effects of polarization of annihilation or decay products without EW correction in [26, 27]. We find that the inclusion of this new estimation of EW corrections and polarization gives rise to different spectra, which may be softer or harder depending on the channel and DM mass. These differences will be discussed in more detail later when we compare the χaroν spectra to those from other softwares.

The neutrinos that remain after this initial step are must then be propagated from the production point to the detector, we consider two cases of interest: neutrino-opaque and neutrino-transparent media. The neutrino interaction length is an energy dependent quantity given by

$$\lambda^{\text{int}}(E_\nu) = \left[n_N \sigma^{\text{tot}}(E_\nu)\right]^{-1} = \left[n_N \sigma_{\text{CC}}(E_\nu) + n_N \sigma_{\text{NC}}(E_\nu)\right]^{-1}, \tag{2.3}$$

where n_N is the number density of nucleons, σ_{CC} is the neutrino-nucleon charged-current cross section, and σ_{NC} the neutral-current cross section; the neutrino-electron cross section is a subdominant contribution to the interaction rate and can be neglected in most applications [28].

For electron and muon neutrinos with energies above 100 GeV [29], the interaction length is essentially identical, and thus the interaction rate will be the same for both flavors of neutrinos. On the other hand, the tau-neutrino charged-current cross section differs significantly from the electron- and muon-neutrino charged-current cross sections due to phase-space suppression, which results in a 30% difference at 100 GeV [30, 31, 32]. This means that below this energy, environments will be less opaque to tau neutrinos than to neutrinos of other flavors. Furthermore, the neutrino yield is flavor-dependent because the competition between interaction and decay differs for each flavor of charged lepton. We do not consider neutrinos from electrons since they do not decay, and, in this work, we do not include neutrinos produced by showers induced by electrons. Muons can decay, but since they are relatively long-lived, they will interact with a dense medium, losing energy and reducing the yield of high-energy neutrinos. Charged tau leptons, on the other hand, are short-lived, and will decay quickly enough to give a significant fraction of their energy to the produced neutrino in a process known as tau-neutrino regeneration [33]. To be concrete, for energies less than 10 PeV and densities of $\mathcal{O}\left(\text{g}/\text{cm}^3\right)$, the charged tau lepton will decay promptly to a hard neutrino.

In addition to effects arising from the neutrinos' interactions with their environment, one must also consider the effect of neutrino flavor change, commonly called neutrino oscillations [34]. This phenomena is due to the fact that the mass and flavor bases are misaligned. These two bases are related by a unitary transformation known as the Pontecorvo-Maki-Nakagawa-Sakata (PMNS) matrix. This three-by-three, complex-valued matrix can be parameterized as a product of two real and one complex rotations, namely $U = R_{23}\bar{R}_{13}R_{12}$ [17] where R_{ij} is a real rotation and $\bar{R}_{13}$ a complex rotation. The real angles involved in this parameterization have been measured to percentage levels [35], and there is hint of a non-zero complex phase [36]. The elements of this matrix, which relate the flavor and mass states, dictate the flavor transition amplitude, while the characteristic length over which transitions occur depends on the mass-squared differences between the mass eigenvalues. This oscillation length is given by

$$\lambda^{\mathrm{osc,i}} = 4\pi \frac{E_\nu}{\Delta m^2_{i1}}, \tag{2.4}$$

where Δm^2_{i1}, with $i = 2$ or 3, are the two neutrino mass-squared-differences which have been measured from terrestrial and solar neutrino experiments and are known to the few-percent level [35].

In this work, we use the ν`SQuIDS` [37, 38] software package to consistently account for the above effects in neutrino transport. This package represents the initial flavor in the density matrix formalism, $\rho_\alpha = |\nu_\alpha\rangle \langle\nu_\alpha|$, the evolution of which is governed by

$$\frac{\partial\rho}{\partial r} = -i\left[H, \rho\right] - \left\{\Gamma, \rho_i\right\} + \int_E^\infty F(\rho, \bar{\rho}, r, E', E)dE', \tag{2.5}$$

where H is the full neutrino Hamiltonian, including both the neutrino kinetic terms and matter potentials; Γ is the interaction rate; and F is a functional which encodes the cascading down of neutrinos due to charged and neutral current processes, see [38] for details. The terms in this equation depend on the environment traversed by the neutrino, as well as the PMNS matrix and neutrino cross sections. For our nominal results and plots shown in this work we set the oscillation parameters to the best-fit values for normal ordering from NuFit-4.1 [35, 39] and model the neutrino nucleon cross section using the `nuSIGMA` software package [40].

After the neutrinos exit their production region—in this work the Sun center, the Earth center or the Galactic Halo —they must be propagated to the detector location at Earth. In this voyage, we need to account for neutrino oscillations in vacuum and propagation inside Earth if the flux traverses significant matter, which imply that the spectrum for ν_α at the detector can be written as

$$\frac{dN^{\oplus}_{\nu_\alpha}}{d\Omega dE_{\nu_\alpha} dt} = f(\Omega)\sum_{\beta}^{3} \frac{dN^{\mathrm{prod}}_{\nu_\beta}}{dE_{\nu_\beta}} \bar{P}_{\nu_\beta\to\nu_\alpha}(d, E_\nu), \tag{2.6}$$

where $dN^{\mathrm{prod}}_{\nu_\beta}/dE_{\nu_\beta}$ is the neutrino yield per annihilation or decay; $\bar{P}_{\nu_\beta\to\nu_\alpha}(d, E_\nu)$ is the production-region-averaged probability of ν_β oscillating to ν_α at distance d with energy E_ν; and $f(\Omega)$ is a function which encodes information about the rate of neutrino production and the geometry of the source.

In what follows, we give additional details relevant for the propagation in the three production environments considered in this work. The result of these processes is summarized in Fig. 2.3.

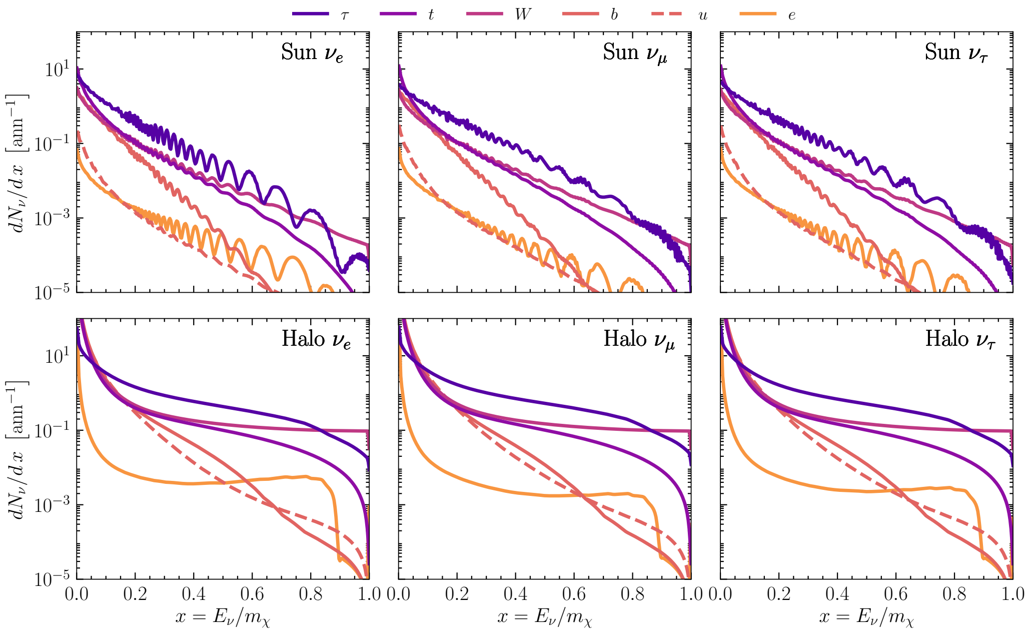


Figure 2.3: ***Neutrino flux from DM annihilation at Earth's surface.*** Colors and line styles have the same meaning as in Fig. 2.2. Results in the top panel are computed with a zenith angle of 60° while those in the bottom panel are computed with a zenith angle of 180°. The DM mass have been set to 1 TeV for annihilation; for decay it should be read as 2 TeV and x should be replaced by $x = 2E_\nu/m_\chi$.

The distance between the Galactic Center and the Earth is sufficiently large that all current- and next-generation detectors do not have sufficient energy resolution to resolve individual oscillations. Thus, the energy and distance dependence in Eq. (2.6) is given by its average value. In this regime, the flavor transition probabilities are give by

$$P(\nu_\alpha \to \nu_\beta) = \sum_i^3 |U_{\alpha i}|^2 \, |U_{\beta i}|^2 \approx \begin{bmatrix} 0.55 & 0.18 & 0.27 \\ 0.18 & 0.44 & 0.38 \\ 0.27 & 0.38 & 0.35 \end{bmatrix}, \tag{2.7}$$

where parameters are obtained by using the best-fit values from NuFit-4.1 with normal ordering.

Neutrinos produced in the solar center must travel through solar matter, vacuum, Earth's atmosphere, and the Earth itself to get the detector. In this article, we use the standard solar model given in [41] to propagate neutrinos from center of the Sun to the surface of the Sun. ν`SQuIDS` accounts for matter effects in solar matter in this process. Assuming the DM is at rest relative to the Sun, the neutrino flux will be emitted isotropically from the center of the Sun.

To compute the expected flux at the detector, one must consider both the detector position and the Earth's position relative to the Sun. We can compute the time-dependent flux which changes with the solar zenith angle and the time-integrated flux which depends on the time window.

For DM clustered in the center of the Earth, neutrinos propagate from the Earth center to the detector. The Earth density and composition is parameterized by the Preliminary Reference Earth Model (PREM) [42]. Since the detector is embedded in Earth, there is no time-dependent information of the flux; namely we assume that there is no relative motion between the DM distribution and the detector. As discussed above, we can use νSQuIDS to propagate neutrinos from the center of the Earth to the location we are interested in.

In this section, we compare our calculation to other results in the literature and estimate the impact of different sources of uncertainties on the final neutrino flux. For a discussion of QCD uncertainties on particle spectra from showering and hadronization, see [43]

Finally, we conclude the discussion of the χaroν by comparing the neutrino fluxes obtained from χaroν with those obtained from other packages. In some instances, our results differ significantly from previous works, so understanding this discrepancy's origin is important. Two broadly used calculations of the flux of neutrinos from DM annihilation or decay have been used in the literature: PPPC and WimpSim. Here, we briefly comment on the differences and similarities between our result and these calculations in the three sources of neutrinos from DM discussed in this work.

Previous works by IceCube [44, 45], ANTARES [46, 45], and Super-Kamiokande [47] searching WIMP annihilation in the Galactic Plane have used DarkSUSY [48, 49], PPPC, or direct PYTHIA calculation, has been used in results predominantly from DM annihilation from the Galactic Halo, see *e.g.* results by I. Our calculation using PYTHIA8.2 matches well with previous calculations with PYTHIA6. Since PYTHIA only partially includes EW interactions, both predicted fluxes are smaller than those given by PPPC, *e.g.* the PPPC calculation gives spectra which are ~ 20 times larger for DM annihilation to $e^- e^+$ with $m_\chi = 1$ TeV. When we incorporate EW corrections, by coupling our calculation with BRW [13], our results yield a flux greater than the PYTHIA versions, but less than the PPPC calculation by an average factor of 1.8, for the same parameters mentioned above, due to different implementation of the EW correction between these works.

Though PPPC provides a calculation of DM annihilation from the Sun, most experimental results use WimpSim. These calculations differ from each other as well as from χaroν in several meaningful ways, both in their treatment of fluxes at production, and in neutrino transport.

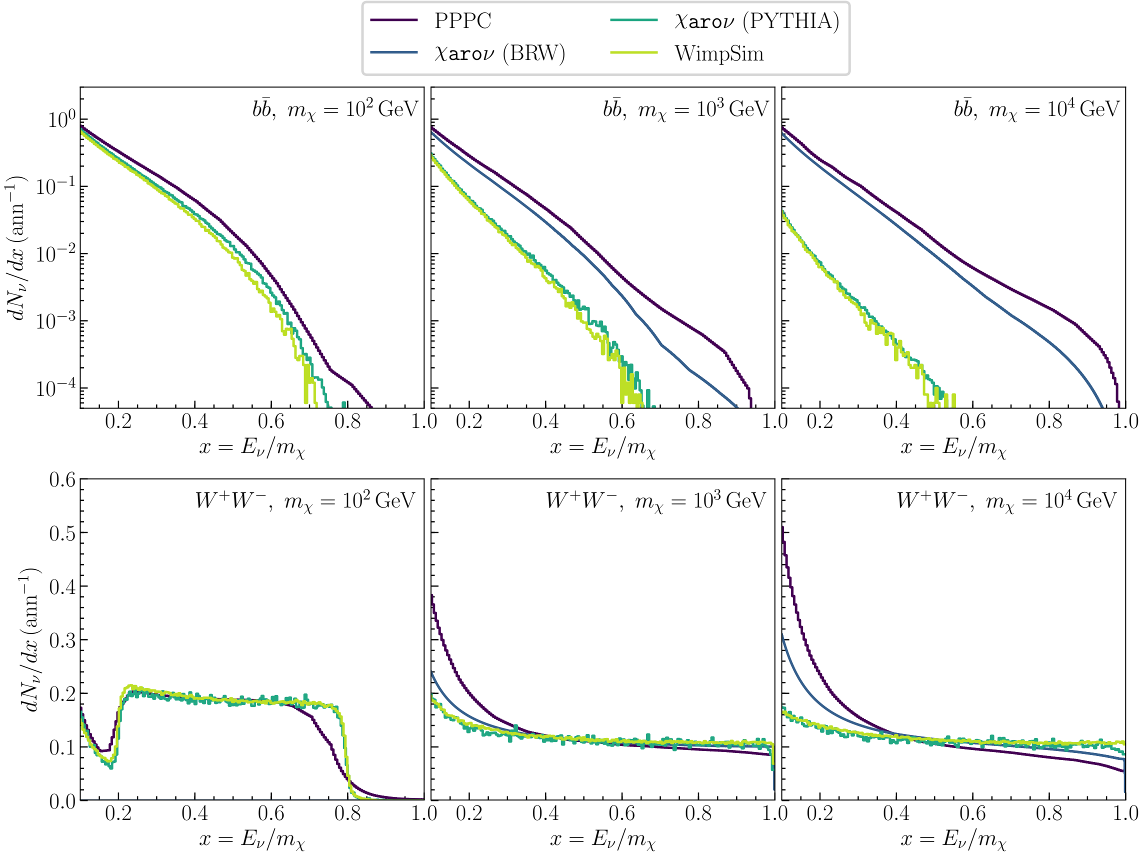


Figure 2.4: ***Comparison of ν_μ-yield using four different signal generators for DM at the Sun center.*** The major contribution to differences between the lines is that a more complete treatment of the EW correction has been implemented in PPPC and χ`aro`ν (BRW). As expected, the magnitude of this difference grows as the mass of the DM increases. When comparing the PYTHIA-based calculations, the $b\bar{b}$ channel in χ`aro`ν is slightly harder than `WimpSim` which is consistent with the result from [50]. The BRW calculation does not extend to masses below 500 GeV and so it is absent from the first column.

With respect to production, both PPPC and χ`aro`ν include a treatment of the EW correction, which is not implemented in `WimpSim`. Furthermore, PPPC handles interactions of stable and metastable particles using GEANT4, which tracks low-energy neutrinos produced in interactions of primary hadrons with the environment. These neutrinos are ignored by `WimpSim`, and as mentioned above, χ`aro`ν allows the inclusion of these low-energy neutrinos as an option. Lastly, PPPC samples energy losses of b and c hadrons from a distribution, whereas χ`aro`ν and `WimpSim` use the average energy loss for a given interaction. Though small, this effect is most notable notable in hadronic channels, *e.g.* $b\bar{b}$ [21].

With respect to propagation, both PPPC and νSQuIDS use an integro-differential equation approach, while WimpSim uses a Monte Carlo-based transport. The Monte Carlo approach allows for event-by-event simulation, which offers several advantages. Specifically, it is easier to couple such simulations to detector simulations and allows tracking the position of the Sun on an event-by-event basis. These advantages, however, come at the cost of significantly increasing propagation time. Additionally, the WimpSim calculation uses a neutrino cross section which uses the CTEQ6 parton distribution functions which which yields a more accurate cross section than other calculations at small-Q^2 values, *e.g.* [51] that use HERAPDF. Our calculation is more similar to PPPC than to WimpSim, and has the following notable differences compared to the latter: we treat DM emission to SM particles as a point-source; we propagate the neutrinos using a differential equation instead of a Monte Carlo method, we take into account the polarization of the τ when considering tau-neutrino regeneration, and, when coupled with BRW calculation [13], have a more complete treatment of EW corrections. As can be seen in Fig. 2.4 for a WIMP of 100 GeV our calculation and WimpSim's are in agreement. For DM masses above the EW scale, for definiteness at 10^3 GeV and 10^4 GeV in the figure, our calculation that includes EW corrections is significantly larger than the WimpSim and Pythia-only predictions. In this range the effects can be as large as a factor of four for the $b\bar{b}$ channel with $m_\chi = 1$ TeV. Lastly, we find the effect of PPPC's partial implementation of EW corrections is channel dependent. For example, in the $b\bar{b}$ channel, PPPC tends to overpredict the flux relative to our calculation over the whole energy range, whereas in the W^+W^- channel PPPC overpredicts the flux at low energies, but underpredicts it at high energies. WimpSim has been used in recent searches for DM annihilation from the Earth, see *e.g.* results by ANTARES [52], IceCube [53] and Super-Kamiokande [54]. In this case, we find that our results agree well with WimpSim when we use Pythia to generate the fluxes and are larger when seeding with the BRW [13] calculation as discussed in the Sun case, as expected.

2.2 Indirect Searches with the IceCube Neutrino Observatory

Equipped with this software, we now focus on searching for neutrinos from WIMP annihilation in the Sun with the IceCube Neutrino Observatory (IceCube). IceCube is a neutrino telescope comprised of 5,160 light-detecting digital optical modules (DOMs) buried between 1,450 m and 2,450 m beneath the surface of the Antarctic ice sheet. The DOMs detect the Cherenkov light emitted when the charged byproducts of neutrino interactions travel at velocities higher than the phase velocity of light in ice. These DOMs are arranged in vertical strings with 17 m vertically between, most of which are arranged in a hexagonal grid with an interstring spacing of 125 m. This allows IceCube to detect neutrinos with energies $\gtrsim$100 GeV. In the center of the detector is a more densely instrumented subarray, the DeepCore detector. The 7 m–10 m vertical spacing and 42 m–72 m interstring spacing in this region allow IceCube to detect neutrinos with energies as low as to $\sim$5 GeV.

IceCube can reconstruct the properties of the incident neutrino, such as its direction and energy, using the quantity of light and its timing information. In particular, this ability to reconstruct the direction of the incident neutrino allows IceCube to function as a telescope. Thus, IceCube may search for an excess of neutrino events coming from the direction of the Sun as a signature of WIMP annihilation.

The flux of neutrinos that arrives at IceCube is given by:

$$\Phi_\nu = \frac{\Gamma_{\text{ann}}}{4\pi r_\oplus^2}\frac{\mathrm{d}N_\nu}{\mathrm{d}E_\nu},$$

where Γ_{ann} is the WIMP annihilation rate, $r_\oplus$ is the astronomical unit, and $\mathrm{d}N_\nu/\mathrm{d}E_\nu$ is the differential distribution of neutrinos as a function of energy. In indirect searches that look towards the Galactic Plane, the annihilation rate is proportional to the thermally averaged WIMP annihilation cross section, $\langle\sigma_{\text{ann}}v\rangle$; however, in the searches for neutrinos coming from the Sun, the rate is proportional to the WIMP-nucleon scattering cross section. To understand this, it is helpful to look at the differential equation that governs the number of WIMPs in the Sun. This is given by:

$$\dot{N} = C - AN^2 - EN.$$

Here, the C is the rate of WIMP accretion due to capture, A is the thermally-averaged WIMP annihilation cross section per unit volume, and E is evaporation rate per WIMP [55, 56]. The evaporation rate for WIMPs with masses above a few GeV is negligible [56]; thus, we may safely set the last term on the right-hand side of the equation to zero. We are then left with a differential equation with a solution given by:

$$N(t) = \sqrt{\frac{C}{A}}\tanh\left(\frac{t}{\tau} - t_0\right),$$

where $\tau = 1/\sqrt{AC}$ and t_0 is an integration constant. Assuming the boundary condition that there are no WIMPs in the Sun when the Sun forms, we can find that this integration constant should be zero. For WIMP annihilation cross sections that reproduce the dark matter relic abundance and for weak-scale scattering cross sections, it has been shown that $t_\odot \gg \tau$, where $t_\odot$ is the age of the

Sun [56]. Thus, we find that:

$$\dot{N}(t_{\odot}) = C\mathrm{sech}\left(\frac{t_{\odot}}{\tau}\right) \simeq 0.$$

We can safely set the left-hand side of our differential equation to zero. At last, we can conclude that the rate of annihilation and capture are equal. Since the capture rate is proportional to the WIMP-nucleon scattering cross section, we can directly probe this cross section by looking for WIMPs captured in the Sun. This is the parameter that direct detection experiments are sensitive to. Thus, searches for neutrinos from WIMPs annihilating in the Sun offer a complementary probe of the scattering cross section.

The capture rate has been computed analytically in [56], and it depends on the dark matter mass, scattering cross section, and the composition of the Sun. We use the solar composition model from [41] for this analysis.

2.3 Solar Atmospheric Neutrinos

Since the neutrino cross section grows with energy, the Sun is only transparent up to certain energies. At 1 TeV, a neutrino produced in the solar core only has a 10% chance of escaping without interacting, and above 3 TeV, this probability has fallen below 5%. Thus, we need only consider sources of backgrounds with energies below ~ 3 TeV. In this energy range, the main source of neutrinos in IceCube is atmospheric neutrinos produced when cosmic rays interact in Earth atmosphere, creating charged mesons that decay to neutrinos. Assuming that the astrophysical neutrino flux follows a single, unbroken power law, it is buried many orders of magnitude beneath the astrophysical flux at these energies. Additionally, these cosmic-ray interactions also generate a flux of muons, which are able to travel from their production points in the atmosphere to IceCube. This gives rise to a large number of background events in the southern sky.

Both of these backgrounds are well-understood and have been measured. Furthermore, each of these backgrounds can be estimated from data, as will be described in Sec 2.5. In addition to these two terrestrial backgrounds, there is another production mechanism for neutrinos in the Sun. Just as in Earth's atmosphere, cosmic rays may interact with nuclei in the Sun and create charged mesons that can decay into neutrinos. This flux has been studied in the literature [59, 60, 61], but despite precise predictions, it remains yet undetected. A schematic representation of this process is shown in Fig. 2.5 This forms an intrinsic background for searches for neutrinos from WIMP annihilation in

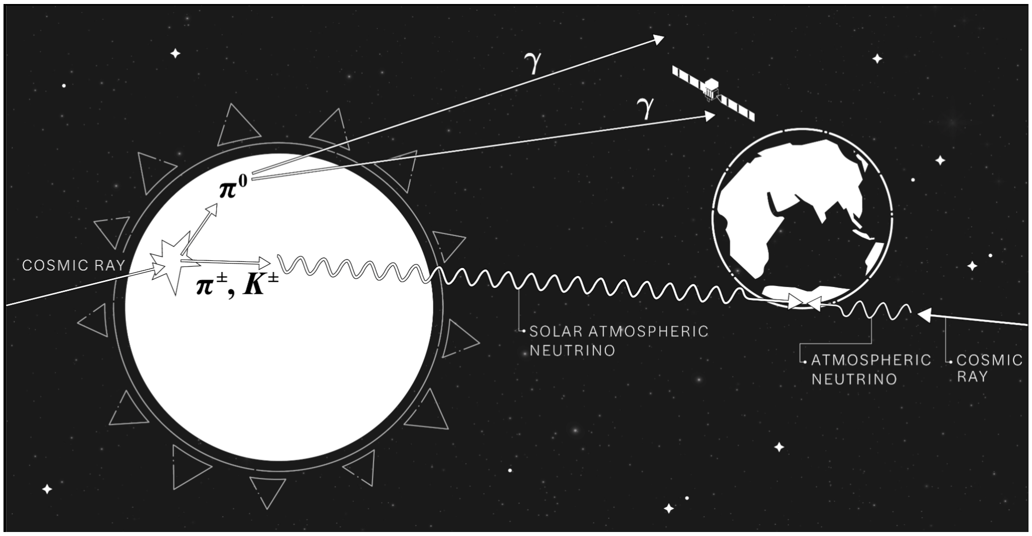


Figure 2.5: ***Schematic representation of solar atmospheric neutrinos.*** Cosmic rays arrive from left and interact in the solar atmosphere. This gives rise to charged and neutral mesons, which can decay to neutrinos and high-energy gamma rays. While the gamma-ray component has been observed [57, 58]

the Sun, and since the predicted value of the flux is ~ 5 on the neutrino flux that would be produced from by WIMPs annihilation at current limits on the cross sections, this cannot be neglected.

In this work, we model the background of solar atmospheric neutrinos using the calculation from [59]. This calculation uses the `MCEq` package to simulate the interactions of cosmic rays in the Sun and propagates the resulting flux with ν`SQuIDS`. Specifically, we will use the prediction that comes from using the primary cosmic-ray flux from [62], the `SIBYLL2.3` interaction model [63, 64], and the so-called MRS model of the prompt component of the flux [65] as our nominal model. When performing fits, we allow the normalization of this flux to take on any value within the 99% limits found by the previous IceCube analysis [66]. Additionally, we studied the effect of mismodelling by comparing the sensitivity of the analysis to mismodelling by injecting the flux from one calculation and trying to fit the flux from another. This showed no impact on the sensitivity.

In addition to treating this flux as a background, I performed a dedicated search for solar atmospheric neutrinos, the sensitivities and results of this analysis will be discussed later, alongside the same discussions for WIMP searches.

2.4 Event Selections

For this analysis, we wanted to target events spanning the entire energy range to which IceCube is sensitive. In order to meet this goal, we combined a selection of events dominated by events arising from muon-neutrino, charged-current interactions that produce track-like features in IceCube with a selection of lower-energy events created in the interaction of all flavors of neutrinos.

The former selection targets neutrinos with energies above a few hundred GeV. In this regime, muons can travel more than hundreds of meters in ice, creating long, track-like patterns of light in their wake that allow for reliable directional reconstructions. Furthermore, restricting the selection to the northern sky leverages the natural overburden of the Earth to filter out atmospheric muon. This enables a neutrino-pure selection that can point to the Sun with $\sim 1^{\circ}$ resolution. In this analysis, we use data taken between 2011 and 2021, for a total livetime of 3,804 days or approximately 10.42 years.

The latter selection targets neutrinos with energies below 100 GeV. At these energies, muons typically travel only a few hundred meters, and the long lever arm that makes a muon-pure selection advantageous at higher energies is diminished. In addition, muons are minimum ionizing particles in this regime and as such have a lower light yield per unit length. These two effects—the shorter lever arm and lower light yield—combine to make muon directional reconstruction more challenging at these energies. Considering the fact that the flux from solar neutrinos is split equally between the flavor while the the terrestrial flux is muon-dominated means that it is advantageous to consider all neutrino flavors in this selection. In order to ensure the statistical independence of these two selections, we remove any events that could enter the high-energy selection at early in the analysis chain. This reduces the total data rate by approximately a factor of two. For this selection, we use data taken between 2012 and 2021, with a total livetime 3,565.5 days, or approximately 9.76 years. The difference in livetimes between these two selections is due to the fact that the DeepCore subarray was deployed one year after the main IceCube array was finished, resulting in roughly one less season of data.

2.5 Analysis Methods

We are doing a binned analysis, and in this section, we will show the binned analysis-level distributions for each selection in signal and background. We use the azimuth scrambling—a data-driven approach—

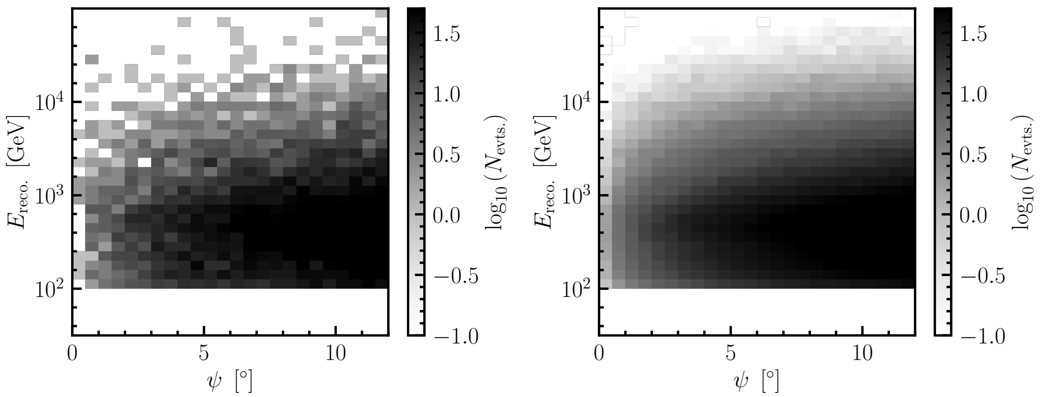


Figure 2.6: ***Effect of oversampling data for background distribution in point source selection.*** The left-hand plot shows the distribution of background events as a function of the reconstructed energy and angle from the center of the Sun after choosing one set of random azimuth angles. The right-hand plot shows the same distribution after oversampling the data 75 times. In addition to smoothing out the distribution, the oversampling procedure ensures that we do not have empty bins in our background distribution in the region where we expect the signal to come from. This is important to prevent an infinite likelihood, an issue that will be discussed in more detail later.

to model our background. This is possible because the signal is subdominant relative to the background. In this method, we replace the azimuth angle determined by the angular reconstruction with a random value drawn uniformly between 0 and 2π. This preserves the expected zenith dependence of the background atmospheric neutrinos and atmospheric muons while spreading the signal. When de-localized, the signal is so small that it has almost no impact on the background distribution. It is worth pointing out that this method assumes that the detector acceptance is uniform in azimuth. While this is not strictly true, the approximation works well enough for this purpose.

We use 360 linearly-spaced bins from 0° to 180° for the angle between the Sun and the reconstructed neutrino direction, and 60 logarithmically-spaced bins from 1 GeV and 10^6 GeV for the reconstructed energy. In addition to these common bins, we bin in the morphology identification boosted decision tree from the DeepCore selection. We adopt the binning used in other analyses that use three bins with edges at [0.0, 0.43, 0.8, 1.0].

To generate the isotropic, data-driven background distributions, we take an event from the experimental data and replace the reconstructed azimuth with a random number drawn uniformly from $[0.0, 2\pi)$. We then compute the zenith and azimuth of the Sun when the event arrived. We then find the reconstructed angular distance between the Sun's position and the new reconstructed neutrino direction. This quantity, the reconstructed neutrino energy, and, if applicable, the morphology

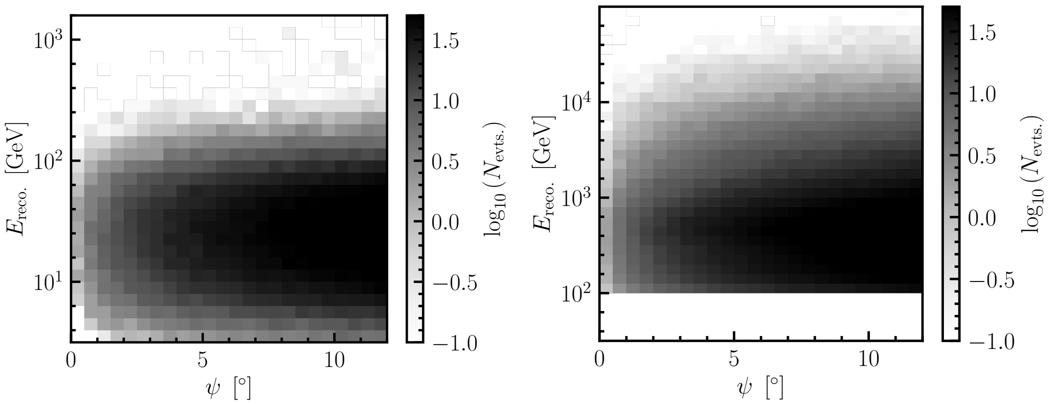


Figure 2.7: ***Background distribution drawn from data.*** Analysis-level background distributions for DeepCore and point-source selections made from oversampling the data. These were made by oversampling the data 65 and 75 times, respectively.

identification score are then binned in the previously described bins. This procedure is then repeated for each event. To have a well-populated background distribution, we repeat this process $N \sim 50$ times and divide the resulting distributions by N to get the distributions used in our analysis. See Fig. 2.6 for an example of this effect and see Fig. 2.7 for final background distributions for the point source and DeepCore selections.

In addition to the data-driven background distributions, two additional distributions must be computed from MC. These are the signal distributions and the background from solar atmospheric neutrinos. In both cases, we first sample a time, t, drawn uniformly over the sidereal year running from 00:00:00, 1 Jan., 2013 until 06:00:00 1 Jan., 2014. The start point is arbitrary and should not affect the expected distribution. We then compute the position and angular extension of the Sun at the time t. We select the events that originated inside the solar disk using the Sun's position and the true direction of each simulated event. We then histogram those events in the reconstructed quantities, weighting by the rate, *i.e.* by the oneweight times the flux. This process is repeated many times, typically on the order of 10^4, adding the histograms together and dividing by the total number of times. This is essentially computing the average rate via a Monte Carlo integration method. See Fig. 2.8 and Fig. 2.9 for the distribution of events for three different dark matter hypotheses in the DeepCore and point-source selections, respectively.

We use a binned, Poisson likelihood for this analysis, *i.e.* the :

$$\mathcal{L}(\vec{\theta}|n) = \prod_i \frac{e^{-\mu_i}\mu_i^{n_i}}{n_i!}$$

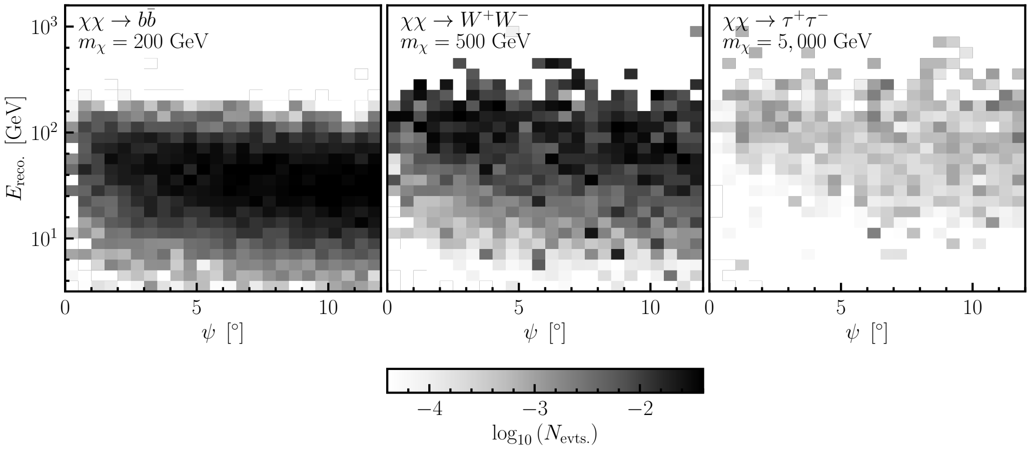


Figure 2.8: ***Distribution of signal events in DeepCore selection for example dark matter hypotheses.*** The number of events expected in the eleven-year livetime of the DeepCore selection. All three hypotheses have been computed at a reference cross section, $\sigma_{\chi N} = 10^{-40}$ cm^2.

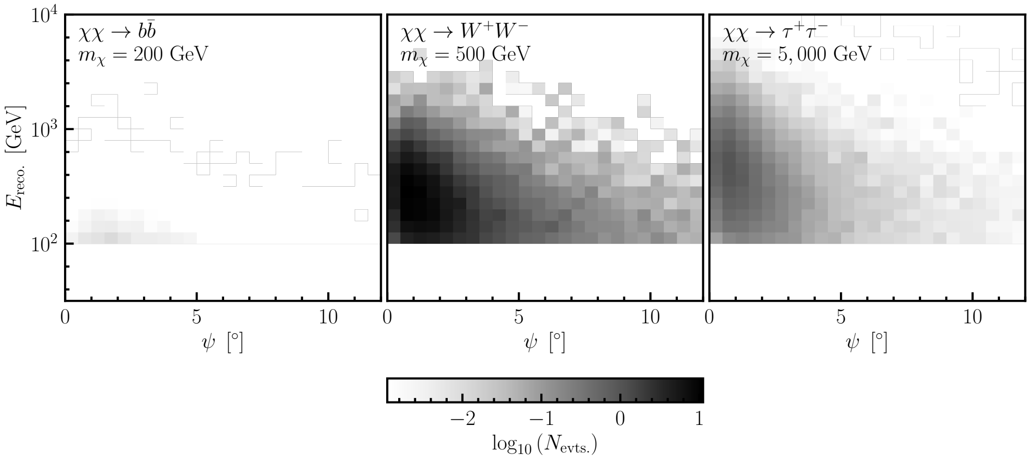


Figure 2.9: ***Distribution of signal events in point-source selection for example dark matter hypotheses.*** The number of events expected in the eleven-year livetime of the point-source selection. All three hypotheses have been computed at a reference cross section, $\sigma_{\chi N} = 10^{-40}$ cm^2. These are the same example cases and reference cross section as in Fig. 2.8.

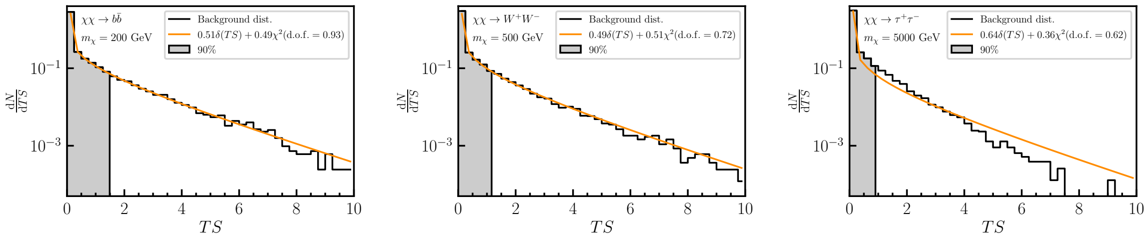


Figure 2.10: ***Trials test-statistic distributions for background-only models for example dark matter hypotheses.*** The orange line shows the best fit of the sum of a delta function, $\delta(TS)$, and a chi-squared distribution with a fit number of degrees of freedom. This is the result of 10^4 background only trials. We do not expect these to follow a chi-squared distribution because an assumption of Wilks' theorem is that we are not near any boundaries; whoever, we enforce that the fitter cannot fit negative signal.

Where $\vec{\theta}$ are the model parameters, n is the observed data, i indexes the bin, μ_i is the expected number of events from a given model in the i^{th} bin, and n_i is the observed number of events in the i^{th} bin. We then define the log-likelihood, LLH as:

$$\ln \mathcal{L} = \ln \left[\prod_i \frac{e^{-\mu_i} \mu_i^{n_i}}{n_i!} \right] = \sum_i -\mu_i + n_i \ln \mu_i - \ln n_i!$$

We may optimize over the model parameters, $\vec{\theta}$, to find the model that best describes the observed data. In this analysis, these model parameters are the normalizations of the nominal signal and background distributions, *i.e.* $\vec{\theta} = \langle n_s, n_b \rangle$, and the model is given by

$$\mu = n_s \, \mu_s + n_b \, \mu_b,$$

where μ_s and μ_b are the signal and background distributions computed as described previously.

Since we have ensured that all of the sub-selections that we are using are statistically independent, the likelihoods for each selection factor and the total likelihood is the product of the likelihood for each sub-selection, *i.e.*:

$$\mathcal{L}_{\text{tot.}} = \prod_{\text{sel.}} \mathcal{L}_{\text{sel.}}$$

and equivalently, the total log-likelihood is just the sum over the individual log-likelihoods, *i.e.*:

$$\text{LLH}_{\text{tot.}} = \sum_{\text{sel.}} \text{LLH}_{\text{sel.}}$$

We construct our sensitivity from trials for this analysis. To do this, we consider a model given by a set of model parameters, $\vec{\theta}_{\text{true}} = \langle \alpha_{\text{dm}}^{\text{true}}, \, 1, \, 1 \rangle$, where we have set both background normalizations to 1. We then draw data realizations, *i.e.* for each bin, we draw a number, n_i, from a Poisson distribution with a mean $\lambda = \alpha_{\text{dm}} \mu_{\text{dm},i} + \mu_{\text{iso},i} + \mu_{\text{sol},i}$. We then maximize the likelihood under two different model assumptions:

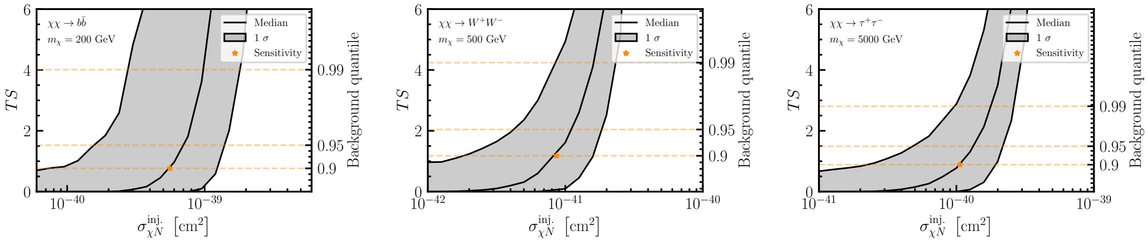


Figure 2.11: ***Median and 1σ inject-recover tests for example signal hypotheses.*** The red line gives the median test statistic for 10,000 pseudoëxperiments at each normalization. The gray band shows the 1σ range for the same set of pseudoëxperiments. Horizontal, dashed orange lines indicate the values of certain background quantiles test-statistic. We mark the point where we achieve our median 90% sensitivity with an orange star.

1. $\hat{\theta}_s = \langle \hat{\alpha}_{\rm dm},\, \hat{\alpha}_{\rm iso},\, \hat{\alpha}_{\rm sol} \rangle$

2. $\hat{\theta}_b = \langle 0,\, \hat{\alpha}_{\rm iso},\, \hat{\alpha}_{\rm sol} \rangle$

called the signal-plus-background model, and the background-only model, respectively. We then define our test statistic as:

$$\text{TS} = 2\left[\ln \mathcal{L}(\hat{\theta}_s|n) - \ln \mathcal{L}(\hat{\theta}_b|n)\right]$$

Simply put, this quantifies the degree to which the signal-plus-background model fits better than the background-only model. As long as the fitting procedure performs correctly, and the value of $\hat{n}_s$ is allowed to go to 0, this quantity should be non-negative.[1]

By repeating this procedure a number of times, we build a *test-statistic* distribution for a particular model. If we choose $n_s = 0$, we call this the background model, and the resulting test-statistic distribution the *background test-statistic distribution*; see Fig. 2.10 for the background test-statistic distribution from the different event selections. If the parameters being fit in the background-only case are far from any boundaries, the background test-statistic distribution should follow a known distribution, called the chi-squared distribution. We plot the chi-squared distribution for one degree of freedom in crimson in Fig. 2.10. It is clear that our test-statistic distribution does not quite follow this distribution, but this is expected since.

We then define our median sensitivity at q confidence, as the model whose test-statistic distribution has a median value that is greater than q% of the background test-statistic distribution. In words, this means that if this model were the true model, we would be able to reject the background-only model

[1] We note that it is also a requirement of this construction that all bins with a signal contribution must have a non-zero background contribution. This is because if an event occurs in that bin, the second term in this difference will be go to $-\infty$ since $n_i\mu_i \to 0$ for this supposed bin.

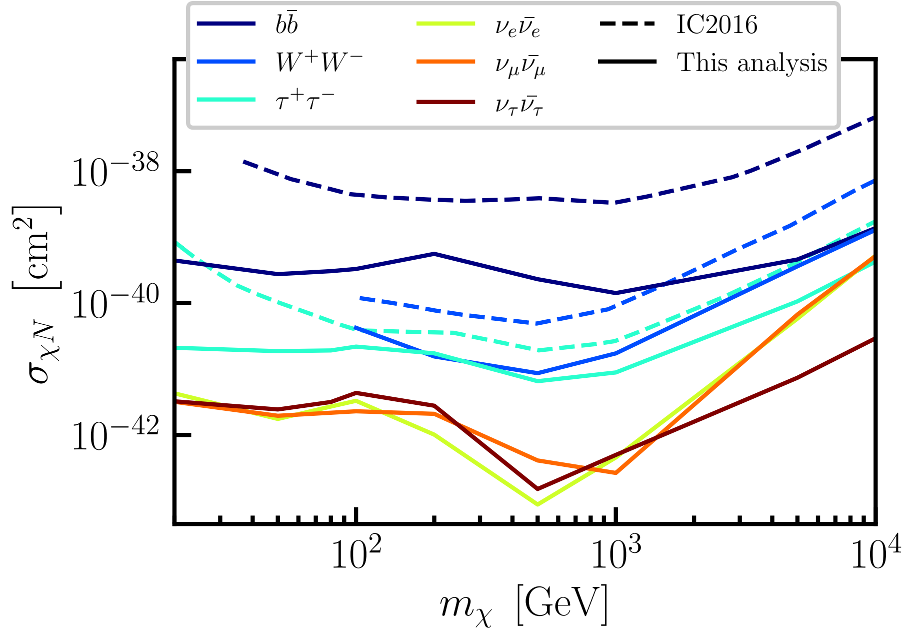


Figure 2.12: ***Sensitivities of this analysis to the spin-dependent WIMP-nucleon scattering cross section.*** This analysis's sensitivities improve upon limits from IceCube's previous solar WIMP analysis—shown as dashed lines—for every channel over the whole mass range. We see particularly dramatic improvements for high-mass WIMP annihilation to $b\bar{b}$ due to the new calculation of the initial neutrino flux used in χ`aro`ν. Additionally, we see large improvements in sensitivity to low-mass WIMPs, which are driven by improved reconstruction techniques for low-energy neutrinos.

at with a p-value of $p = 1 - q/100$ half the time. We should note that this departs from the sensitivity convention that is used in the IceCube point source group, in which the sensitivity is defined as the model at which 90% of test-statistics exceed the median of the background test-statistic distribution. We believe that the definition we have chosen lends itself to a cleaner statistical interpretation and is more in line with the colloquial understanding of the word "sensitivity." We find that using only the northern tracks selection, we have sensitivity to 2.55 times our nominal model. We also find that if the true model is the nominal model, we should expect to be able to see it at 95% confidence 14% of the time. See Fig. 2.11 for a visual representation of this definition, and Fig. 2.12 for the sensitivity of this analysis to all WIMP hypotheses we consider. The current sensitivities improve over limits from the previous analysis in the $b\bar{b}$, W^+W^-, and $\tau^+\tau^-$ channels. In some regimes—such as the high-mass $b\bar{b}$ and low-masses—this analysis improvement is greater than an order of magnitude. In the former case, this improvement is driven by the updated calculation of the initial neutrino spectrum introduced in χ`aro`ν. Furthermore, this is the first analysis that will test direct annihilation to neutrinos.

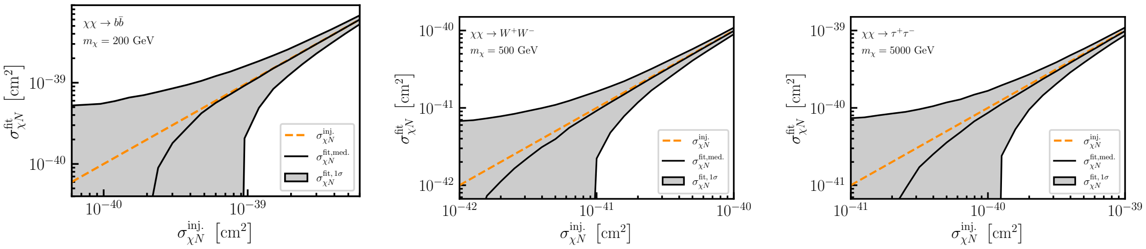


Figure 2.13: ***Inject-recover tests for the three example WIMP hypotheses.*** The injected cross section is shown as a dashed orange line, while the median and 1σ bands for the fit cross section are shown as a black line and grey envelope respectively. At cross sections below the sensitivity of this analysis, the median fit value does not match the injected value; however, above the sensitivity threshold, the median value and the injected value are in good agreement.

In addition to computing the sensitivity, we can also look at the fitted value of the model parameters as a function of the true injected parameters. The expectation is that, on average, the fit should recover the injected value. In the left panel of Fig. 2.13, we show the fitted signal normalization as a function of the injected value. The median, denoted by the crimson line, is in good agreement with the injected value, shown as a dashed grey line. Furthermore, the spread is symmetric about this line in regions where the fit is not limited by the physical boundary at $n^{\mathrm{fit}}_{\mathrm{sig.}} = 0$. In the middle and right panels of the same figure, we should the fitted background normalization as a function of the injected signal normalization for the signal-plus-background and background-only fits, respectively. Since we are using a data-driven background approach, we always inject at $n^{\mathrm{inj.}}_{\mathrm{bg}} = 1$. As expected, we are able to recover the background with a typical precision on the order of 0.1%.

In addition to the previously described search, we can use the same analysis framework to search for solar atmospheric neutrinos. In this search, we assume there is no dark matter in the Sun and compare the hypotheses of no emission localized at the Sun and our nominal model of solar atmospheric emission. To do this, we carry out the same procedure described above under the substitution:

$$\hat{\theta}_s \to \hat{\theta}'_s = \langle 0,\, \hat{\alpha}_{\mathrm{iso}},\, \hat{\alpha}_{\mathrm{sol}} \rangle$$
$$\hat{\theta}_b \to \hat{\theta}'_b = \langle 0,\, \hat{\alpha}_{\mathrm{iso}},\, 0 \rangle\,.$$

In Fig. 2.14, we show the background-only test-statistic distribution, signal test-statistic distribution, and inject-recover tests for the solar atmospheric analysis. The background test-statistic distribution matches the We find that we have sensitivity to a flux that is 2.75 times larger than our

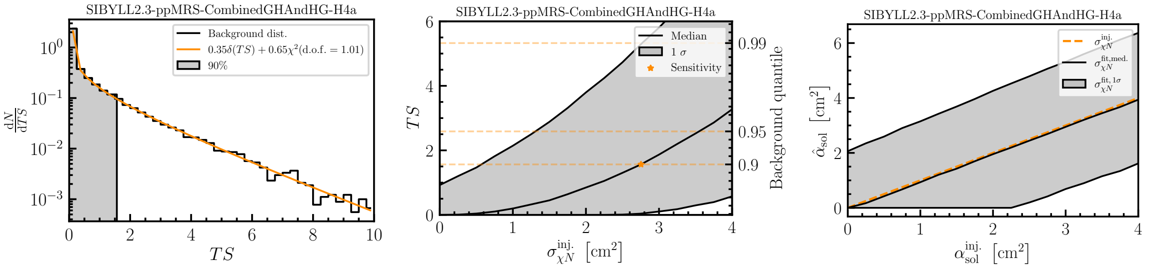


Figure 2.14: ***Test statistic, sensitivity, and inject-recover for solar atmospheric analysis.*** The left panel shows the test-statistic distribution for background fits of the solar atmospheric model. As expected, it follows a linear combination of chi-squared and delta distributions. The middle panel shows the median test statistic and 1σ containment as a function of the injected signal normalization. The sensitivity is shown as a star. We show the fitted signal normalization in the right panel versus the injected signal normalization. The median fitted value is in good agreement with the injected value.

nominal prediction. Furthermore, we find that we have a chance of observing the nominal model at 2σ 10% of the time due to statistical fluctuations.

2.6 Results and Interpretations

We show the unblinded data for this analysis in Fig 2.15. For each WIMP hypothesis, we fit the background-only and signal-plus-background hypotheses to the unblinded data and compute the test statistic between the two fits as previously described. We observed the largest preference for the

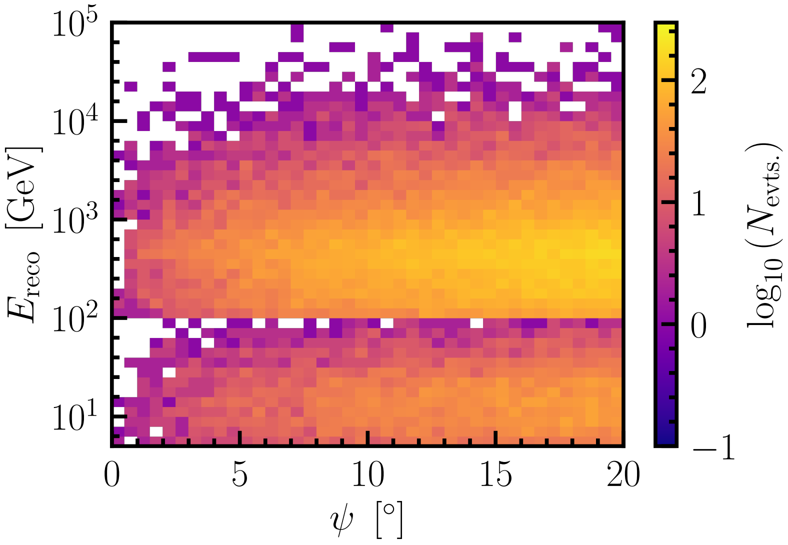


Figure 2.15: ***Unblinded event distribution for low- and high-energy selections.*** The event distribution as a function of the reconstructed energy and reconstructed angular distance from the center of the Sun. No significant excess of events from the Sun was found in these data.

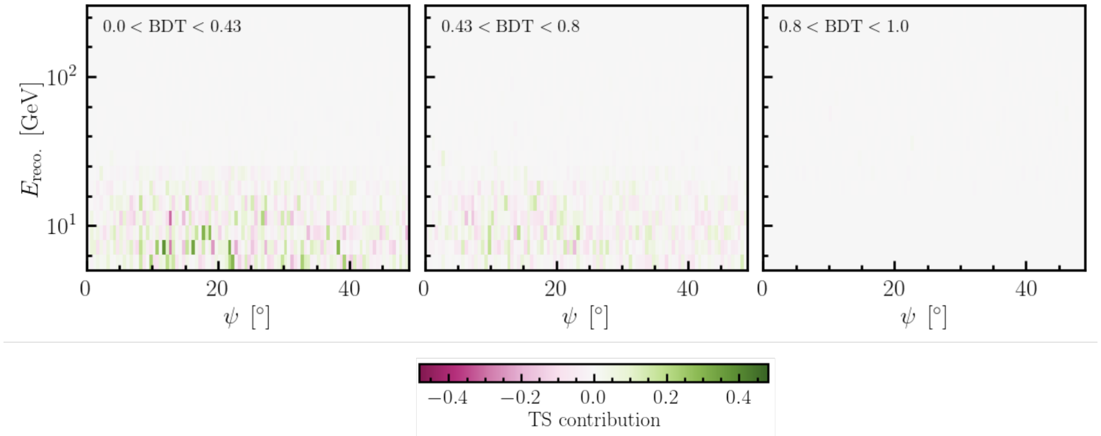


Figure 2.16: ***Per-bin test statistic for the channel with the largest preference for the signal-plus-background-fit.*** Green (pink) bins (dis)favor the signal-plus-background. The preference for the signal-plus-background hypothesis is driven by the leftmost panel, which is dominated by cascade-like events. Since the 20 GeV WIMP mass is below the simulation threshold for the high-energy selection, we only show these distributions for the low-energy DeepCore selection.

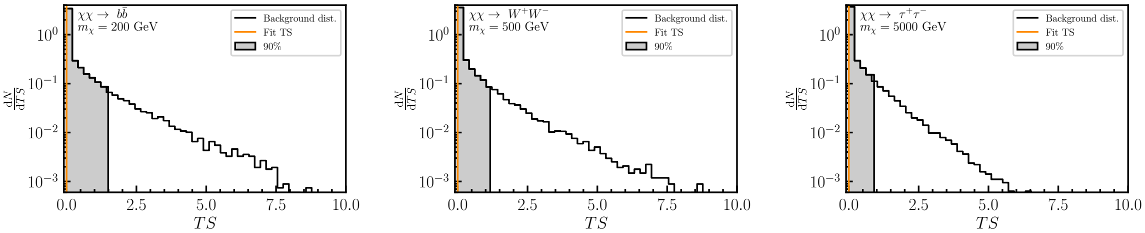


Figure 2.17: ***Background-only test-statistic distributions with fitted test statistic.*** In each case, the best-fit point of the background-only and signal-plus-background models was identical, resulting in a test statistic of zero.

signal-plus-background fit in the We observed the most significant preference for the signal-plus-background fit in the $\chi\chi \to b\bar{b}$ channel with a WIMP mass of 20 GeV. This WIMP hypothesis had a test statistic of 2.00, corresponding to a p-value of 0.076. We show the per-bin test statistic for this WIMP hypothesis in Fig 2.16. This figure shows that the preference for the signal-plus-background fit is driven by cascade-line events with reconstructed energies between 5 and 15 GeV.

This was the only WIMP hypothesis with a p-value less than 0.1, and this is consistent with statistical fluctuations for the number of WIMP hypotheses considered. We compare the fit test statistic value to the background-only trials for the example WIMP hypotheses in Fig. 2.17. This is consistent with no excess from the Sun due to WIMP annihilation so that we can set limits on the WIMP-nucleon scattering cross section.

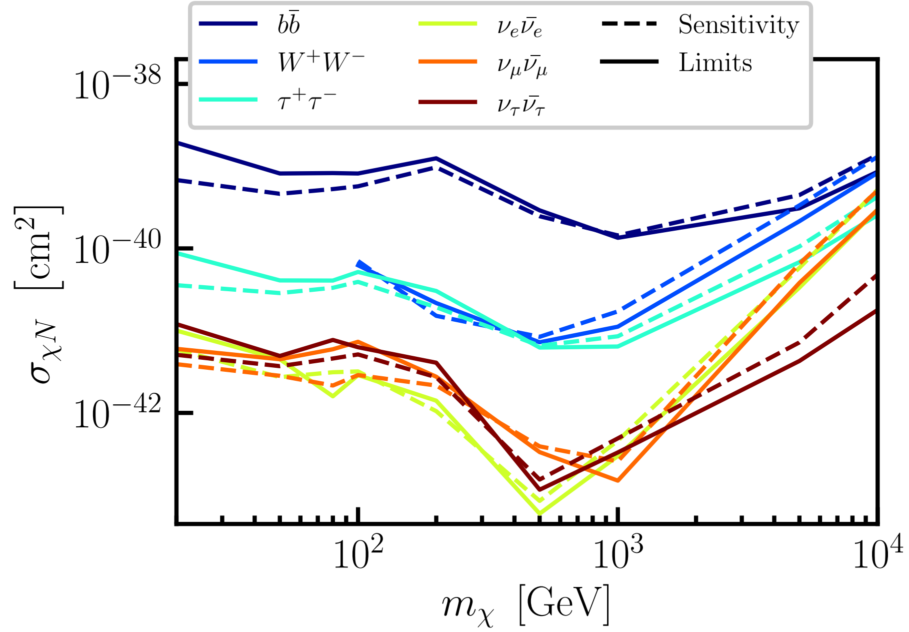


Figure 2.18: ***Limits on spin-dependent WIMP-nucleon cross section obtained in this analysis compared to analysis sensitivity.*** We set limits that are slightly stronger than the sensitivity in the high-mass regime and slightly weaker in the low-mass regime. All limits are within the expected statistical fluctuations of the analysis.

We increase the dark matter flux normalization to find the limit—holding the normalization on our data-driven and solar atmospheric backgrounds at the best-fit values—until the likelihood difference between this model and the background-only model exceeds a certain threshold. In our case, we choose this threshold to be the 90% containment of the background-only test statistics, and thus, we call these 90% upper limits. In Fig. 2.18, we compare the limits obtained in the analysis to the sensitivities. In the high-mass regime, we obtain limits that are slightly more constraining than our sensitivities. In contrast, in the low-mass regime, we obtain limits that are slightly less constraining than the sensitivities. These slight differences are not unexpected, and all limits are well within the expected statistical fluctuations of the analysis. In Fig. 2.19, we show the limits from this analysis compared to other experiments.

Additionally, we did not find evidence of solar atmospheric neutrino emission. The best-fit point was at a normalization relative to the nominal model at 0.0. This allowed us to set a limit on the flux of neutrinos from cosmic rays interacting in the solar atmosphere at 2.41 times the nominal model.

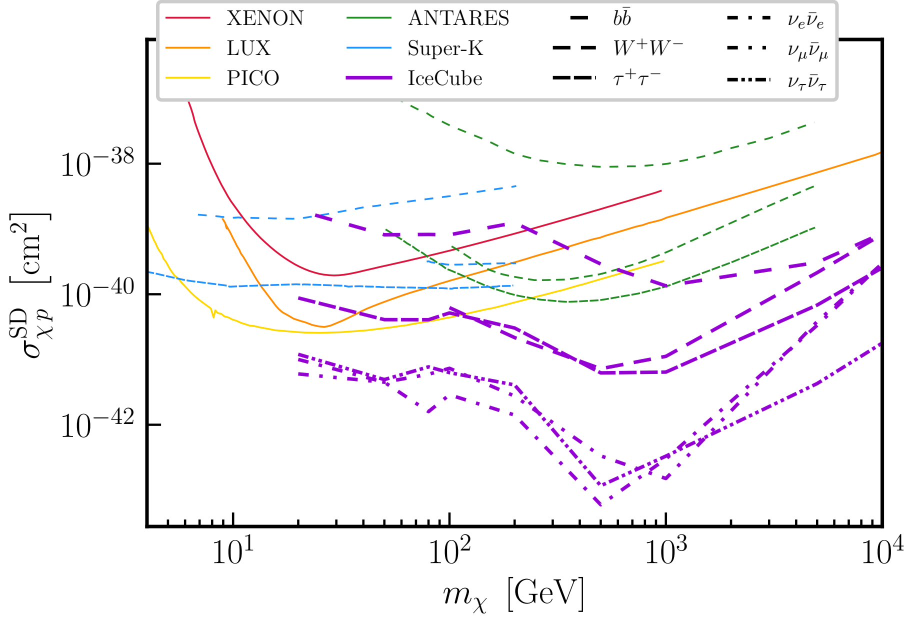


Figure 2.19: ***Limits on spin-dependent WIMP-nucleon cross section obtained in this analysis compared to limits from other experiments.***

Chapter 3

Exploring the Energy Frontier with Tau Neutrinos and the `TauRunner` Software

Since its completion a decade ago, IceCube has made significant progress in characterizing the diffuse spectrum of astrophysical neutrinos. The most recent fits [67] show that the energy dependence of this spectrum is consistent with a single power law up to a few PeV. A significant fraction of the next-generation neutrino telescopes will focus on measuring neutrinos with even higher energies to test whether the power-law fit continues to hold and to detect new fluxes of neutrinos [68, 69]. Furthermore, these observatories will attempt to make unambiguous measurements of the flavor composition of the astrophysical neutrino spectrum. Such measurements are currently limited by differentiating tau-neutrino-induced charged-current events from electron-neutrino-induced charged-current events and all-flavor neutral-current events.

ν_τ and the unique properties of $\tau^\pm$ will play a crucial role in both of these pursuits. In particular, the phenomenon of tau regeneration [70] needs to be treated properly. In this process, a ν_τ undergoes a charged-current interaction and creates a $\tau^\pm$, which quickly decays into another ν_τ and an electron or muon neutrino. This effect means that fluxes of high-energy ν_τ arriving at Earth are not attenuated, but are merely shifted to lower energies, allowing the high-energy fluxes to be seen in the low-energy secondaries.

Additionally, the short lifetime of the $\tau^\pm$ enables a class of searches that provide a pure sample of tau-neutrino events, thus breaking the previously mentioned flavor degeneracy. In these searches, one looks for evidence of the air shower formed when a $\tau^{\pm}$ decays hadronically emerging from a region that would filter out non-neutrino-induced events. This filtering is typically accomplished either by looking at a small region of the Earth near the horizon or into a mountain.

If we are to realize these goals, we must accurately model the behavior of high-energy neutrinos and charged leptons. The previous generation of software primarily concerned itself with neutrinos with energies below 1 TeV. In this energy range, the negligible scattering rates imply that the problem of neutrino transport requires considering only in-flight flavor change. Thus, the currently available packages for simulating neutrino transport [71, 72] are not well-suited to addressing the evolving needs of the neutrino community.

In this chapter, we will begin by describing `TauRunner`, a Python-based package for simulating neutrino transport that considers several previously neglected effects. We will then use this package to simulate the neutrino yields from extremely heavy, WIMP-like particles in the Sun. Finally, we will show the simulation framework for the Tau Air-Shower Mountain-Based Observatory, a proposed, next-generation telescope in the Peruvian Andes.

3.1 `TauRunner` Monte Carlo Package

To address the problem of neutrino transport, the neutrino community has developed analytical and numerical methods to compute the neutrino oscillation probabilities efficiently [73], *e.g.* `nuSQuIDS` [38] among others [74, 75, 76, 77]. In these first-generation packages tau regeneration was implemented by using the so-called on-spot tau decay approximation, which neglects tau energy losses. Though this approximation satisfies the needs of most current scenarios and experimental settings, next-generation neutrino telescopes aim to reach EeV energies [78, 79]. At these extremely high energies, the taus produced in neutrino interactions are sufficiently long-lived that their energy losses cannot be neglected. Recently, dedicated software packages have been made available to solve this problem in this energy regime. However, the bulk of the available solutions neglects the stochasticity of tau losses considering only their mean effect. This limits their ability to function as event generators in neutrino telescopes and produces mismodeling of the yield of tau-induced events for a small number of scatterings, where the stochastic nature of the losses is more relevant. A notable exception is the `NuPropEarth` [80] package developed for the KM3NeT experiment [81], which is presently being built in the Mediterranean Sea. Though `NuPropEarth` offers a complete solution, this package requires a large number of dependencies to function, making its distribution and installation difficult.

Software	Language	Input	Output	Medium	Energy losses($l^{\pm}$)
`TauRunner`	`Python`	ν_α, τ, μ	ν_α, τ, μ	Earth/Sun/ Moon/Custom	`PROPOSAL`
`NuPropEarth`[80]	`C++`	ν_α	ν_α, τ	Earth/Custom	`TAUSIC`
`nuPyProp`[90]	`Python/FORTRAN`	ν_τ	τ	Earth	Internal
`NuTauSim`[91]	`C++`	ν_τ	τ	Earth	Continuous

Table 3.1: ***Software comparison table.*** Each row of this table represents a given package. Input and output particles include their not explicitly mentioned antiparticles. Custom medium refers to a user-defined `Body` in `TauRunner`. The Energy losses column compares the treatment of charged particle losses.

Here, we describe a new package, `TauRunner`, that aims to provide a complete and versatile solution to the neutrino transport problem at high energies. Our `Python`-based package is designed to have minimal dependencies, to allow the user to construct arbitrary neutrino trajectories and propagation media, and to provide interfaces to modify physics inputs such as neutrino cross sections easily. This package was first introduced in [82, 83], where it was used to study the ANITA anomalous events [84, 85], and is currently used in studies relating to extremely high-energy neutrinos in IceCube [86]. With respect to the preliminary version, the version presented in this paper contains significant improvements in terms of performance and available features to the user. In this article, we describe the software and provide examples, benchmarks and comparisons to other packages that have similar aims. We expect that our software will be useful for next-generation neutrino detectors operating in liquid water (P-ONE [87]), solid water (IceCube-Gen2 [78]), mountains (Ashra NTA [88], TAMBO [89]), and outer space (POEMMA [79]). Our hope is that the success of neutrino oscillation measurements enabled by the previous generation of software will be mirrored in the study of high-energy neutrino properties with efficient propagation software such as the one presented in this paper.

The aim of this software is to solve the transport equation for high-energy neutrino fluxes passing through matter. The transport equation can be written as follows [92],

$$\frac{\mathrm{d}\vec{\varphi}(E,X)}{\mathrm{d}X} = -\sigma(E)\vec{\varphi}(E,X) + \int_E^\infty \mathrm{d}\tilde{E}\ f(\tilde{E},E)\vec{\varphi}(\tilde{E},X), \tag{3.1}$$

where E is the neutrino energy, X is the target column density, $\sigma(E) = \mathrm{diag}(\sigma_\nu, \sigma_{\bar{\nu}})$ holds the total ν and $\bar{\nu}$ cross section per target nucleon, $f(\tilde{E}, E)$ is a function that encodes the migration from higher to lower neutrino energies and between ν and $\bar{\nu}$, and $\vec{\varphi}(E,x) = \{\phi_\nu, \phi_{\bar{\nu}}\}$ contains the neutrino and anti-neutrino spectrum. At energies supported by this package, 10 GeV–10^{12} GeV, neutrino-nucleon deep inelastic scattering (DIS) is the dominant neutrino interaction process. The first term on the right hand side accounts for the loss of flux at energy E due to charged-current (CC) and neutral-current (NC) interactions, whereas the second term is the added contribution from neutrinos at higher energy, $\tilde{E}$, to E through NC interactions of $\nu_{e,\mu,\tau}$ and CC interactions in the ν_τ channel.

This latter channel is unique in that the short τ lifetime causes the decay of the charged lepton before losing a large fraction of the parent energy. The τ then decays into a daughter ν_τ, meaning that the primary ν_τ flux is not lost, but only cascades down in energy. Moreover, if the τ decays leptonically, $\bar{\nu}_\mu$ and $\bar{\nu}_e$ are created, contributing significantly to the outgoing flux, as was first pointed out in [93]. By default, `TauRunner` takes all those contributions into account. The story is simpler for the electron channel. There, CC interactions result in electrons which lose their energy quickly and are subsequently absorbed in the medium. As a result, electron losses are not modeled in `TauRunner` by default, though the capability exists if needed. For the muon flavor, muons resulting from CC interactions can travel $\mathcal{O}(1)$ kmwe. Therefore, it is important to model the propagation and losses of muons near the point of exit, and that is accounted for in `TauRunner` as well.

In `TauRunner`, Eq. (3.1) is solved using a Monte-Carlo approach. A flowchart of the `TauRunner` Monte-Carlo algorithm is shown in Fig. 3.1. Given an initial neutrino type, energy, and incident angle, it begins by calculating the mean interaction column depth, λ_{int}, which depends on the medium properties and neutrino cross section. A column depth is then randomly sampled from an exponential distribution with parameter λ_{int}, and the neutrino advances the corresponding free-streaming distance. If the neutrino does not escape the medium, either an NC or CC interaction is chosen via the accept/reject method. In the case of an NC interaction, the neutrino energy loss is sampled from the differential cross section, and the process repeats. In the case of a CC interaction, a charged lepton is created with energy sampled from the neutrino differential cross section.

The treatment of the charged lepton then varies according to the initial neutrino flavor. Electrons are assumed to be absorbed and the propagation stops there. μ and τ, however, are recorded and passed to `PROPOSAL` [18] to be propagated through the same medium. μ that do not escape will either decay at rest resulting in neutrinos that are below the energies supported by `TauRunner`, or get absorbed. Therefore a μ that does not escape is not tracked further. Finally, τs can either escape or decay. In the latter case, a secondary ν_τ is created whose energy is sampled from tau decay

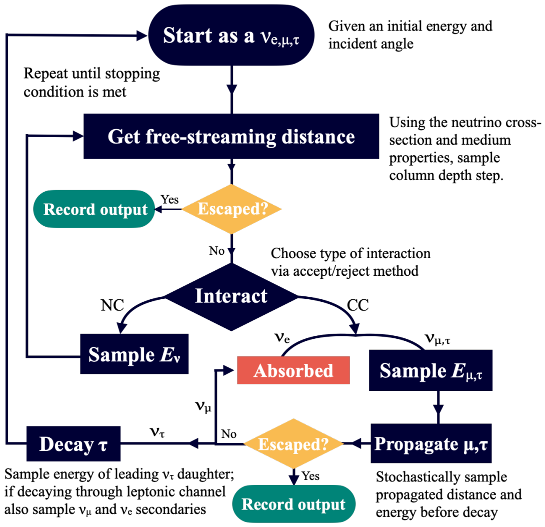

Start as a $\nu_{e,\mu,\tau}$
Given an initial energy and incident angle
Repeat until stopping condition is met
Get free-streaming distance
Using the neutrino cross-section and medium properties, sample column depth step.
Record output
Yes
Escaped?
No
Choose type of interaction via accept/reject method
Interact
NC
CC
Sample E_ν
ν_e
$\nu_{\mu,\tau}$
Absorbed
Sample $E_{\mu,\tau}$
ν_μ
Decay τ
ν_τ
No
Escaped?
Propagate μ,τ
Yes
Sample energy of leading ν_τ daughter; if decaying through leptonic channel also sample ν_μ and ν_e secondaries
Record output
Stochastically sample propagated distance and energy before decay

Figure 3.1: ***Flowchart of the* `TauRunner` *propagation algorithm.*** Square boxes indicate actions performed by the software. Diamond boxes indicate decision-making stopping points. Rounded-corner squared boxes indicate beginning and end of the algorithm. Note that users can select also charged leptons as the initial state, in which case the algorithm skips straight to the charged particle propagation step.

distributions provided in [94]. Additionally, if the τ decays leptonically, ν_e or ν_μ will be created. When this happens, the properties of the resulting secondaries are recorded and added to a basket which stores all secondary particles to be propagated together after the primary particle propagation is complete.

Measurements of neutrino cross sections with matter have been performed up to a few PeV in energy [95]. This includes a multitude of accelerator [96, 97] and reactor [98, 99] experiments as well as solar [100], atmospheric [101], and astrophysical neutrinos [102, 103]. However, the energy range supported by `TauRunner` goes far beyond the measurements, where the fractional momenta, x_{Bjorken}, of the quarks probed by the neutrino can reach $x_{\text{Bjorken}} \ll 10^{-8}$. The nucleon structure function is not measured at such low x_{Bjorken} and is extrapolated in cross section calculations [51, 80]. Such extrapolations neglect gluon color screening making perturbative QCD calculations of the neutrino cross section grow faster than allowed by unitarity at extremely high energies [104]. Phenomenological approaches to include gluon screening parameterize the extremely small x_{Bjorken} behavior using a dipole model [105] of the nucleon so as to result in a $\ln^2(s)$ dependence of the cross section at extremely high energies [106]. This ultimately results in a difference of a factor ~ 2 at 10^{12} GeV. `TauRunner` provides, by default, neutrino and anti-neutrino DIS cross section tables for two PDF models: a perturbative QCD calculation [51], and a dipole model [105]. Additionally, the user can provide their own cross sections.

In the Standard Model, when neutrinos undergo CC interactions, they convert to their charged partners through the exchange of a W boson. Charged particles lose energy in dense media through many processes, and the relative importance of each process depends on the lepton's mass and its energy [107]. At lower energies, a charged lepton can ionize atoms as it traverses the medium. This process is described by the Bethe-Bloche equation, and at higher energies scales logarithmically and becomes sub-dominant for all flavors. A charged lepton can also interact with the electric field of a nucleus, losing energy in the process through the emission of a photon. This process, called bremsstraahlung, scales like the inverse-sqaured mass of the lepton, and is therefore the dominant energy loss mechanism for electrons. Another possible interaction with the field of a nucleus leads to the production of electron-positron pairs. This process scales like the inverse of the lepton mass, and is one of the leading energy-loss mechanisms for charged mu and tau leptons. Finally, charged leptons can lose energy by exchanging a photon with a nucleon, in what is referred to as a photonuclear interaction. This process dominates tau energy losses at the highest energies ($\geq 10^9$ GeV). The aforementioned processes are implemented in `PROPOSAL`, which we use to model them in `TauRunner`. Apart from interacting, charged mu and tau leptons can also undergo weak decays. This process scales like the mass of the lepton to the fifth power, and is therefore the most likely outcome for taus propagating in Earth up to 10^9 GeV. Above this energy, the total interaction length for other processes becomes shorter than the decay length. Charged mu leptons, on the other hand, are much more likely to lose all of their energy before decaying at rest, or getting absorbed by a nucleus. Therefore, we only model decays of $\tau^{\pm}$ using parametrizations in [94].

A `Particle` instance contains the structure of a `TauRunner` event. This includes, among other quantities, the particle's initial and current energies, particle type, and position. Additionally, it has a number of methods for particle decay and interaction as well as charged lepton propagation. Finally, the τ decay parametrization is contained in `particle/utils.py`.

The user may propagate ν_e, ν_μ, ν_τ, μ^-, τ^-, or any of the corresponding anti-particles in `TauRunner`. To do this, the user should initialize the the `Particle` object with the corresponding Particle Data Group Monte Carlo number [107]. It should be noted that the user may create an $e^{\pm}$, but the internal logic of `TauRunner` assumes all $e^{\pm}$ are immediately absorbed and thus no propagation occurs; see Fig. 3.1.

The Track class contains the geometrical information about the particle's trajectory. A track is parametrized by an affine parameter which defines the position along the trajectory: 0 is the beginning of the trajectory, and 1 is the end. Almost all of the methods of the Track class are mappings between the affine parameter and physically relevant quantities, *e.g.* radius, distance traveled, and column depth. The only argument which is generic to the Track class is depth which specifies the distance below the surface of the body at which to stop propagation. This may intuitively be thought of as the depth of the detector to which the particles are propagated. An illustration of the TauRunner geometry and a diagram of the functional relation of physical quantities to the affine parameter is shown in Fig. 3.2

The Track class allows the user to make custom trajectories. The user need only specify mappings between the affine parameter and these variables. Different trajectories may require additional arguments from the user, depending on the nature of the trajectory. To illustrate this point, we can look at the two tracks which are implemented by default, the Chord and Radial trajectories. The former is used for paths which originate outside the Body and cross a section of Body. The latter is used for paths which originate at the center of the Body. The former Track describes neutrinos coming from space and passing through Earth on the way to a detector, as in the case of Earth-skimming τ searches, while the latter gives the trajectory of a neutrino originating in the center of the planet, relevant for searches for neutrinos from gravitationally trapped dark matter. Clearly, an incoming angle needs to be specified for the Chord trajectory. Thus, we can see that the necessary arguments for specifying a Track may vary from one geometry to another.

The Body class specifies the medium in which the Particle is to be propagated. In TauRunner, we require that all bodies be spherically symmetric, and so a Body may be minimally specified by a physical radius, and a density profile. The density profile may be a positive scalar, a unary function which returns a positive scalar, or a potentially-mixed list of positive scalars and such functions. The sole argument of the functions used to specify the density should be the radius at which the density is to be given, in units of the radius of the body, *i.e.* the domains should be $[0, 1]$. In this system $r = 0$ is the center of the body and $r = 1$ the surface. If the user wishes to make a layered body, *i.e.* one where a list specifies the density profile, they must pass a list of tuple with the length of this list equal to the number of layers. The first element of each tuple should be the scalar or function which gives the density, and the second element should be the right hand boundary of the layer in units of

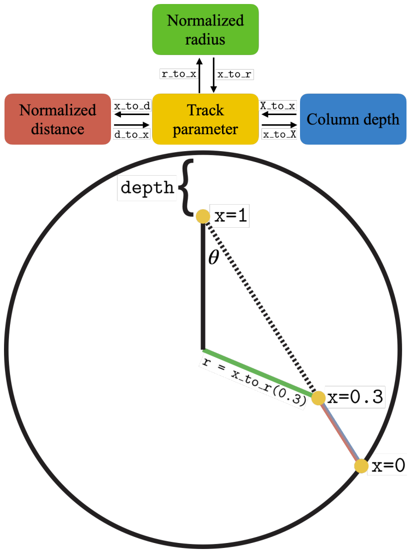


Figure 3.2: ***Schematic of* `TauRunner` *geometry as contained within the* `Track` *class.*** (a) shows the relation between the physical quantities relevant to propagation and the affine parameter that parametrizes the `Track`. The arrows connecting these quantities are labeled with the functions used to convert between them in `TauRunner`. Specifically, these are the functions a user must define in order to specify a custom `Track` geometry. All distances are normalized with respect to the radius of the body in which the track sits. (b) shows a diagram of these parameters within a spherical `TauRunner` body. Colors correspond to the boxes in (a). Additionally, it illustrates the `depth` parameter which intuitively gives the depth of the detector.

the radius. The last right hand boundary should always be 1 since $r = 1$ is the outer edge of the body. Lastly, all densities should be specified in g/cm^3.

In addition to a radius and a density profile, the user may also provide the `proton_fraction` argument to specify the fraction of protons to total nucleons in the body. By default, we assume that the propagation medium is isoscalar, *i.e.* we set the proton fraction to 0.5 throughout the entire body. As in the case of the density profile, this argument may be a scalar, a function, or a list of function-boundary tuples. The domains of any functions provided must be [0, 1], and the ranges must be in this same interval.

While the user can construct bodies themselves, there are five bodies implemented by default in `TauRunner`: the Earth, a high-metallicity Sun, and low-metallicity Sun, the moon, a constant density slab. We use the PREM parametrization to model the densities of Earth [42]. For the Sun, we use fits provided by [108]. To instantiate the `Earth` object, one calls the `construct_earth` function, which returns an `Earth` object. Additionally, this function allows one to pass in a list of additional layers which will be placed radially outward from the edge of the PREM Earth. This functionality may be useful for *e.g.* adding a layer of water or ice or adding the atmosphere for simulating atmospheric air showers. To initialize the Sun, one can use the `construct_sun` function. With this function, the user may specify 'HZ_Sun' or 'LZ_Sun' to use the high- and low-metallicity `TauRunner` suns respectively, or a path to a user defined solar model.

The `TauRunner` cross sections module defines the neutrino interactions. Internally, `TauRunner` assumes that cross sections are equal for all neutrino flavors. Additionally, `TauRunner` uses the isoscalar approximation by default, *i.e.* it assumes a medium is made of equal parts p^+ and n; however, this assumption may be changed by altering the `proton_fraction` of the `Body` object. The software includes both CSMS [51] and dipole [109] cross sections implemented by default; however, it is straightforward for the user to implement other cross section models by providing `scipy` splines in the appropriate format. For the total neutrino cross section these splines are `scipy.interpolate.UnivariateSpline` objects whose x-axis is the $\log_{10}$ of the neutrino energy in eV and whose y-axis is the $\log_{10}$ of cross section in cm^2. The differential cross section splines are `scipy.interpolate.RectBivariateSpline` objects whose x-axis is the $\log_{10}$ of the neutrino energy in eV, whose y-axis is a convenience variable which combines the incoming and outgoing neutrino energies, E_{in} and E_{out}, given by

$$\eta = \frac{E_{\text{out}} - 10^9 \text{ eV}}{E_{\text{in}} - 10^9 \text{ eV}},$$

and whose z-axis is the $\log_{10}$ of incoming energy times the differential cross section in cm^2.

To propagate charged leptons, `TauRunner` relies on `PROPOSAL`, an open source C++ program with `Python` bindings. A utility module to interface with `PROPOSAL`, `utils/make_propagator.py`, is provided with `TauRunner`. This function instantiates `PROPOSAL` particle and geometry objects, which are then used to create a propagator instance. Since `PROPOSAL` does not support variable density geometries, the `segment_body` function is used to segment the `TauRunner` body into a number of constant density layers. The number of layers is determined by solving for points in the body where fractional change in the density is equal to a constant factor, called `granularity`. This argument may be specified by the user, and by default is set to 0.5. A single propagator object is created for all $\tau^{\pm}$ and, if needed, for all $\mu^{\pm}$. Since `TauRunner` assumes $e^{\pm}$ are always absorbed, a propagator will never be made for these. Whenever a new geometry is used, `PROPOSAL` creates energy loss tables which are saved in `resources/proposal_tables`. The tables require a few minutes to generate, resulting in an overhead for new configurations, but subsequent simulations with the same geometry will not suffer any slow down.

TauRunner uses a natural unit system in which $\hbar = c = \text{eV} = 1$. As a consequence of this system, any energy passed to TauRunner must be in eV. TauRunner includes a units package to easily convert common units to the units TauRunner expects. This may be imported from the utils module, and its usage is demonstrated in several examples. Additionally, since TauRunner assumes that propagation occurs in a spherical body, the radius of this body establishes a natural length scale. Thus all distances are expressed as a fraction of this radius.

The run_MC function, which carries out the logic of TauRunner, returns a numpy.recarray. This array may be set to a variable if running TauRunner from a script of notebook, or printed or saved if running TauRunner from the command line.

In this paragraph, we will describe the fields of this output. The "Eini" field reports the initial energy of the lepton in eV. The "Eout" field reports the energy of the particle when propagation has stopped in eV. In the case that the particle was absorbed, this field will always read 0.0. The "theta" field reports the incident angle of the lepton in degrees. The "nCC" and "nNC" fields report the number of charged and neutral current interactions the particle underwent in its propagation. The "PDG_Encoding" field reports the particle type, using the Particle Data Group MC numbering scheme. The "event_ID" is a number carried byfield reports which initial lepton the particle comes from. The "final_position" field reports the track parameter when the propagation was ended. This may be used to physical quantities of a particle when it was absorbed, or when a user-defined stopping condition was met.

The results of several ν_τ simulation sets are illustrated in this section. Fig. 3.3 shows column-normalized distributions of outgoing neutrino energy fraction as a function of initial neutrino energy. Interestingly, the dashed line showing the median outgoing ν_τ energy fraction varies with a constant slope, corresponding to the energy at which Earth becomes transparent. That energy is roughly 10 PeV at the horizon (top left), $\mathcal{O}(1)$ PeV in the mantle (top right and bottom left), and $\mathcal{O}(10)$ TeV through the core (bottom right). This means that for a large fraction of the Northern Sky, ν_τ pile-up and escape at energies where the atmospheric neutrino background is relatively low. This idea is also made clear when illustrated for a monochromatic flux. In Fig. 3.4, EeV ν_τ are propagated and the outgoing energies are plotted as a function of nadir angle. A similar feature can be seen, where a majority of neutrinos in this simulation escape with energy above 100 TeV.

TauRunner has also been compared to several publicly available packages that perform similar tasks. A summary of the various tested packages and their features is shown in Tab. 3.1. Besides

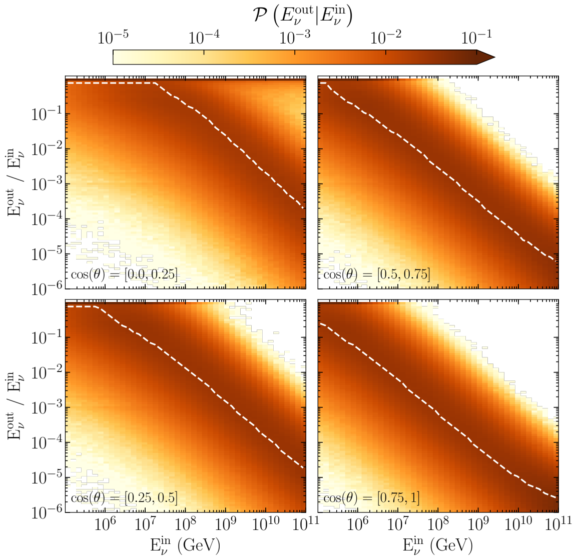


Figure 3.3: ***Outgoing ν_τ distributions for an E^{-1} power-law flux.*** Shown are the outgoing ν_τ energy fraction as a function of the primary tau-neutrino neutrino flux injected as an $\rm E^{-1}$ power-law from 100 TeV to 10 EeV, shown in slices of equal solid angle in the Northern Sky. The dashed line indicates the median outgoing energy

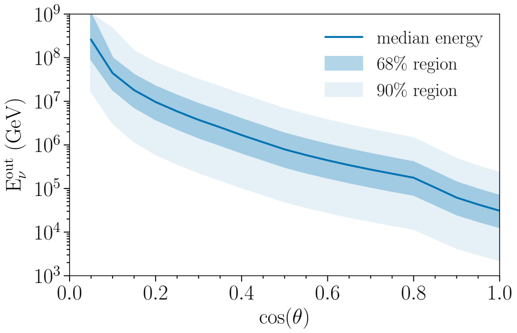


Figure 3.4: ***EeV ν_τ in Earth*** Median outgoing energies of secondary ν_τ shown as a function of nadir angle. Also, 68% and 90% probability contours for outgoing energies are included. The feature at approximately $\cos\theta$ of 0.8 is caused by the core.

`TauRunner`, only `NuPropEarth` offers a full solution in the case of ν_τ. To illustrate this, we show in Fig. 3.5 the output of both packages for an injected monochromatic flux of ν_τ at 10^{10} GeV and one degree below the horizon. For secondary taus and ν_τ, the two packages show excellent agreement. We note that comparisons with `NuPropEarth` use the trunk version of the code, which has a new treatment for charged particle propagation using `PROPOSAL` instead of `TAUSIC`. Secondary anti-muon and -electron neutrino distributions show slight disagreement in the tails, likely due to different tau polarization treatments. These differences are still being investigated, and will be addressed in an upcoming work.

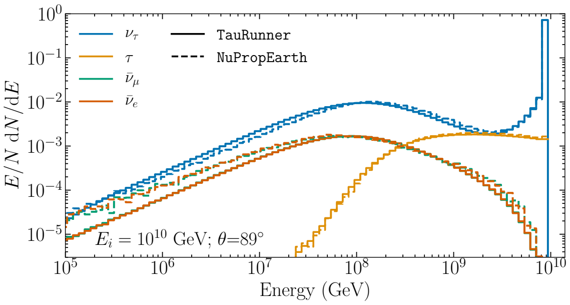


Figure 3.5: ***A monochromatic flux of*** ν_τ Outgoing particle energy distributions for a fixed angle and energy. We include secondary anti-electron and -muon neutrinos, as well as charged taus. `TauRunner` shows good agreement with `NuPropEarth`. This set assumes Earth as a body with a 4km layer of water.

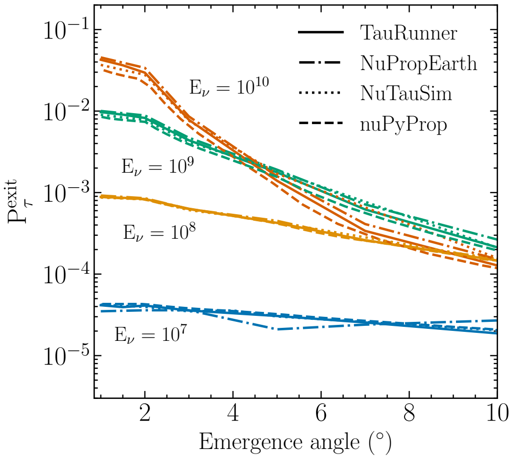


Figure 3.6: $\tau^{\pm}$ ***exit probability.*** Different colors correspond to four different monochromatic neutrino energies. The emergence angle is measured with respect to horizon. The `TauRunner` prediction (solid line) is compared to NuTauSim, NuPropEarth, and nuPyProp, which are shown in different linestyles.

Fig. 3.6 shows a comparison of the charged tau exit probability in Earth as a function of nadir angle. P^{τ}_{exit} is the probability that an incoming neutrino will exit Earth as a charged tau. This quantity is especially relevant for future neutrino observatories hoping to detect Earth-skimming ν_τ. In that scenario, exiting taus make up the bulk of the expected signal. `TauRunner` again shows great agreement overall with other packages.

3.2 Extremely Heavy Solar Dark Matter

When search for neutrinos from WIMP annihilation in the Sun, one runs into an issue when looking moving to higher WIMP masses. Since the Sun's core is both more than an order of magnitude denser and larger than that of the Earth, the Sun becomes neutrino-opaque at lower energies. Absorption begins to play a significant role for neutrinos with energies around 1 TeV and is over 99% opaque to neutrinos with energies NUMBER. This means indirect searches for WIMPs captured in the Sun are limited to WIMPs with masses below a few TeV. While this constraint is not an issue for WIMPs in the context of the "WIMP Miracle" [56], which typically have masses on the order of a few hundred GeV, heavier WIMPs may be invoked to explain other anomalies.

For instance, ultra-high mass WIMP-like particles have been proposed to explain the so-called ANITA anomalous events. The Antarctic Impulsive Transient Antenna (ANITA) is a balloon-based observatory that detects radio signals from Askaryan emission of extremely-high energy (EHE) charged particles. The primary goal of this observatory is to indirectly detect EHE ν_τ via the Askaryan radiation emitted by the $\tau^\pm$ created when the ν_τ undergoes a charged-current interaction. For this to happen, the ν_τ must come from with a few degrees below the horizon, undergo a single charged-current interaction to create a $\tau^\pm$ which exist the Earth and decays in the air. The signal can be differentiated from background Askaryan radiation created by EHE cosmic rays reflecting off the ice by observing the polarization of the signal. Reflected cosmic-ray signals will pick up an additional phase compared the the unreflected counterparts.

The tau-neutrino signal can only come from shallow angles because otherwise, there will be multiple charged-current interactions and the $\tau^\pm$ will be beneath the Askaryan threshold. However, in 2016 and 2018, the ANITA collaboration reported two events whose polarizations were consistent with unreflected signal and whose reconstructed direction was approximately 30° beneath the horizon [110, 111]. In Ref. [82], the authors show that any transient or steady SM flux that could produce such events is ruled out by IceCube's non-observation of lower-energy, regenerated counterparts. Having ruled out SM explanations of the flux, the authors of Ref. [112] postulated that a WIMP-like particle with a mass of approximately 480 PeV decaying to a ν_τ-$\bar{\nu}_\tau$ pair inside the Earth could create this signal.

If this were true, one would expect the same WIMP-like particle to accumulate in the Sun as well as the Earth, and may use the approach of the last chapter to probe this hypothesis; the extremely high mass of this particle presents a challenge. Even though tau regeneration is accounted for in packages like ν`SQuIDS`, which is used for neutrino transport in χ`aro`ν, these packages use the previously discussed, on-spot decay approximation. This assumes that the $\tau^\pm$ decays instantaneously without losing energy. This should hold in the energy range below approximately 50 TeV where the

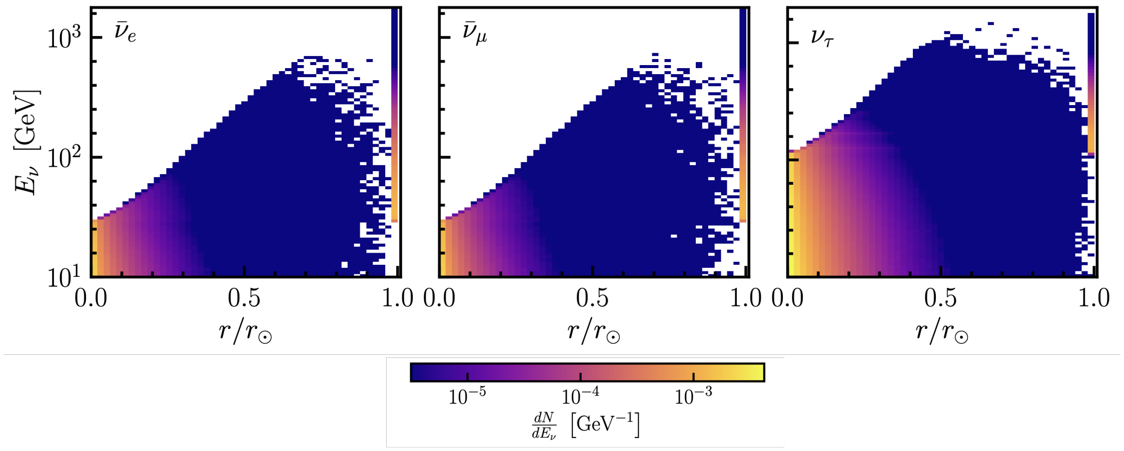


Figure 3.7: ***Neutrino flux output from `TauRunner` from initial ν_τ.*** The curve in each figure is determined both by the changing solar density and the remaining propagation distance. The yellow regions suggest most neutrinos lose sufficient energy to allow oscillations to dominate within the solar core; however, a non-negligible fraction reaches this criterion in the Sun's lower-density outer region. Furthermore, some neutrinos escape the Sun entirely, as can be seen by the bin on the right edge of each plot. The plots for the initial $\bar{\nu}_\tau$ are identical up to particle-anti-particle conjection.

tau travels only a few meters before decaying, but at higher energies, one must properly account for the energy losses of the $\tau^\pm$. Furthermore, these packages do not include secondary neutrinos produced from the $\tau^\pm$ decay.

The flexible implementation of propagation media and trajectories in `TauRunner` makes it capable of propagating the ν_τ from the WIMP-like particle's decay in the Sun. To do this, we inject a neutrino-anti-neutrino pair at the core of the Sun and propagate it radially. `TauRunner` follows its normal algorithm, with one additional stopping condition. Since `TauRunner` does not simulate neutrino oscillation, we need to stop the simulation if we reach neutrino energies where oscillations dominate, *i.e.* where the oscillation length becomes shorter than the interaction length. This consideration can be safely ignored in the Earth because the Earth becomes transparent to neutrinos below energies of approximately 100 TeV, where the oscillation length is on the order of 10^8 km, approximately 4,000 times larger than the diameter of the Earth.

After carrying out this procedure 10^8 times, we get the average distribution of neutrinos as a function of energy and the radius at which the propagation ended. The resulting distribution of ν_τ and electron- and muon-anti-neutrinos resulting from the initial ν_τ is shown in Fig 3.7. Along the top of each plot, there are some events that escape the Sun with sufficiently high energies that the oscillation length is never shorter than the interaction length. The curve that runs through the plot reflects the radial density profile of the Sun, which sets the interaction length.

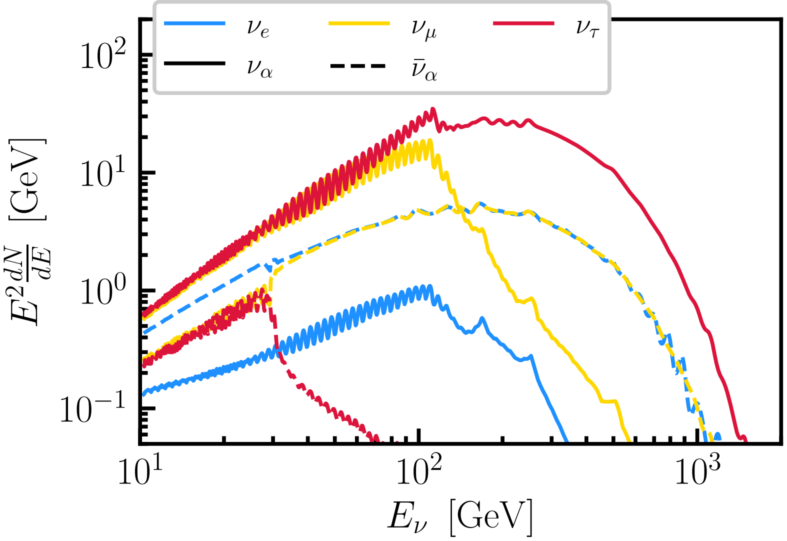


Figure 3.8: ***Neutrino flux at the Sun's surface after propagation with νSQuIDS.*** One can see two distinct regimes above and below the critical energy for each channel.

Since the resulting flux is safely in the regime where the on-spot approximation holds, we then pass these neutrinos to ν`SQuIDS` to propagate the neutrinos to the surface of the Sun. This is accomplished by using the "emitting body" functionality of ν`SQuIDS`, which continuously adds neutrinos as the propagation moves radially outward. Some care must be taken here as the sharp cutoffs at the critical energies of 30 GeV and 110 GeV can cause numerical instabilities in ν`SQuIDS`. To circumvent this, we perform two propagations, one above and below the critical energies for each channel. Once reaching the surface of the Sun, the results from both are combined and smoothed to get the total spectrum. In Fig. 3.8, we show the flux for ν_τ and electron- and muon-anti-neutrinos at the surface of the Sun.

The combined fluxes are then propagated to Earth, where we can convert them to a flux via:

$$\Phi_{\nu+\bar{\nu}} = \frac{N\Gamma}{4\pi r_\oplus^2}\left[\Phi_{\nu_\tau\to\nu_\alpha} + \Phi_{\bar{\nu}_\tau\to\nu_\alpha}\right],$$

where N is the total number of WIMP-like particles in the Sun, Γ is the decay rate, $r_\oplus$ is the distance between the Earth and the Sun, $\Phi_{\nu_\tau\to\nu_\alpha}$ is the flux of neutrinos from the initial ν_τ, and $\Phi_{\bar{\nu}_\tau\to\nu_\alpha}$ is the flux of neutrinos from the initial $\bar{\nu}_\tau$. We show this flux at the Earth's surface in Fig. 3.9.

We can now recast the previously published IceCube limits on solar WIMPs [113] as limits on the rate from this flux of neutrinos. We use the limits obtained for annihilation to an initial $\tau^+\tau^-$ pair since the shape of this spectrum is closest to the one we computed. This analysis considered three WIMP masses of 3 TeV or greater. Specifically, they considered 3 TeV, 5 TeV, 10 TeV WIMPs, and set limits on the combined ν_μ-$\bar{\nu}_\mu$ flux at $\Phi_{\nu_\mu+\bar{\nu}_\mu} = 29.0\ \text{km}^2\,\text{yr}^{-1}$, $\Phi_{\nu_\mu+\bar{\nu}_\mu} = 29.3\ \text{km}^2\,\text{yr}^{-1}$,

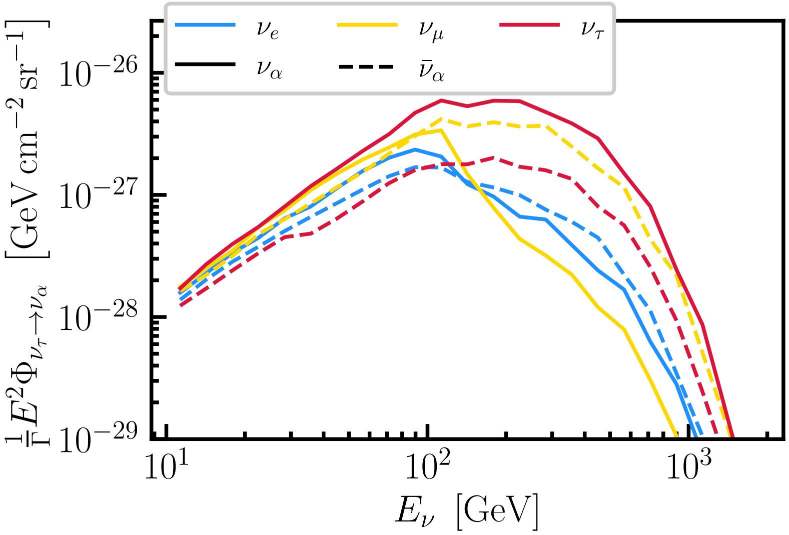


Figure 3.9: ***Flux of neutrinos after propagation to the surface of Earth.*** We have averaged this flux over energy bins that are 0.1 decades wide to filter out the effects of rapid neutrino oscillations at energy scales smaller than IceCube's energy resolution.

and $\Phi_{\nu_\mu+\bar{\nu}_\mu} = 32.1\ \text{km}^2\,\text{yr}^{-1}$. By integrating our flux in energy, we are able to set a limit at $N\Gamma = 5.26 \times 10^{15}$.

Interpreting this result in the context of the WIMP-like particle proposed in Ref. [112] is difficult as it is heavily dependent on the particle distribution in Earth, which the authors say does not follow a standard thermal distribution. Possible configurations that would reproduce the ANITA events is currently being investigated and should appear in future work. However, this calculation technique can be applied in general to study high-mass, WIMP-like particles in dense media.

3.3 Simulation for the Tau Air-Shower Mountain-Based Observatory

The Tau Air-Shower Mountain-Based Observatory (TAMBO) is among the proposed next-generation neutrino observatories. Deployed in the Colca Valley in the Peruvian Andes, this detector will comprise $\sim 20,000$ plastic scintillator panels set up on one side of the valley. A ν_τ traveling through the mountain may undergo a charged-current interaction. The $\tau^\pm$ produced in this interaction can travel a mean distance of approximately 50 m per PeV of energy before decaying. If the tau lepton exits the mountain in this time and decays hadronically, the resulting byproducts will induce an air shower of muons, electrons, and photons that the scintillator array can detect.

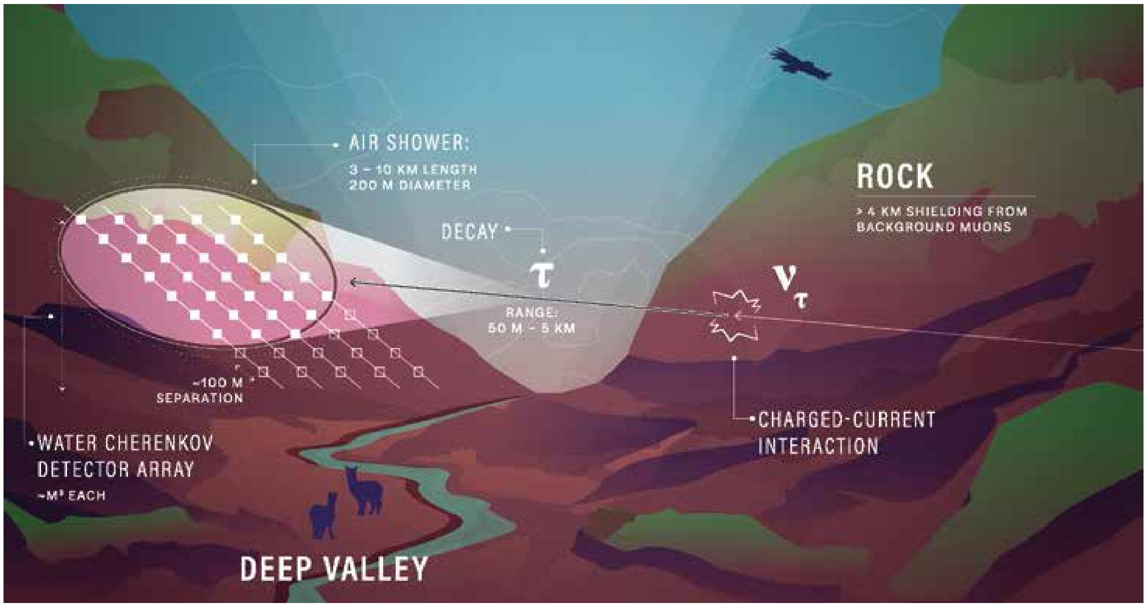


Figure 3.10: ***Schematic of the TAMBO design and detection principle.*** in this diagram, the incident ν_τ enters from the figure's right and interacts within the mountain. The resulting τ^- exits the mountain and decays in the valley. Particle detectors can detect the resulting air shower on the opposite face of the valley.

This air shower will not form for a ν_e or ν_μ in teracting in the valley. A ν_e will produce an electron that would lose all its energy in ~ 10 m, likely leading to becoming trapped in the mountain. A ν_μ, on the other hand, will produce a muon, which is expected to travel ~ 10 km at energies above a PeV. Thus, it will travel through the array before decaying, creating a signal in at most one scintillator panel. Thus, the valley's geometry ensures that the TAMBO will observe high-purity tau-neutrino events, allowing an unambiguous measurement of the tau-neutrino fraction of the astrophysical neutrino flux.

TAMBO will have peak sensitivity to neutrinos with energies between 500 TeV and 100 PeV. The short lifetime of the $\tau^\pm$ sets the lower limit, while the width of the valley sets the upper limit. This lower end of this range overlaps with the highest-energy neutrinos that IceCube has observed, guaranteeing a flux of neutrinos for TAMBO to observe. Furthermore, TAMBO will be sensitive to neutrinos that are over an order of magnitude more energetic than those seen by IceCube. This will allow TAMBO to differentiate between different astrophysical models and determine if there is an exponential cutoff of the astrophysical flux, as predicted by some models [114, 115].

Initial predictions of TAMBO's detector acceptance and event rate were performed in [89]. By extrapolating the flux reported by IceCube in [116], this work found that TAMBO would expect to see approximately 18 events in three years of observing. While this work shows that TAMBO's design is promising, it used a simplified canyon geometry, did not simulate $\tau^{\pm}$ energy losses in the rock, and relied on a parametrization of the air-shower physics. Furthermore, it did not have particle timing information, making detailed triggering and reconstruction studies impossible. To understand TAMBO's sensitivity more precisely, we developed a full simulation chain that properly treats these effects effects and includes per-particle timing information.

The simulation proceeds in several stages. First, we must select the properties—*e.g.* the energy, incident direction, and interaction vertex—of the incident neutrino. Given these neutrino properties, we must select the properties of the outgoing $\tau^{\pm}$. These two steps are typically called event *injection*. We then propagate the injected $\tau^{\pm}$ through the simulation region, accounting for the different media the lepton moves through along the way and any energy losses that may occur in flight. Next, we allow the $\tau^{\pm}$ to decay, and, if relevant, simulate the air shower that the decay products induce, find the distribution of air-shower products on the valley's face, and simulate the detector response.

At each step, we must strike a fine balance between covering the full phase space relevant to our detector while maintaining computational efficiency. To give an extreme example, we want to simulate interactions that happen in the mountain opposite our detector, but we certainly should not simulate a neutrino interacting in the Moon since such an interaction has a negligible chance of triggering our detector. Deciding where to draw this line between these two extremes requires some care and will be outlined later in this section. Furthermore, it may sometimes be more convenient to sample quantities in a way that does not follow the physical distribution. For instance, it might be easier to sample the incident neutrino energies from a power-law distribution or to ignore an angular dependence and sample uniformly in the angular phase space. In order to remove these convenient but non-physical choices, one must weight the events in order to convert them to a physical rate. One may then take the total rate of events as the weighted average over all simulated events.

To perform these steps, we use newly built software for sampling, weighting, and geometric handling supplemented with external packages to handle more involved simulation of the physics processes. Specifically, we use the previously described `TauRunner` package to simulate the passage of neutrinos from Earth's surface to the simulation region; `PROPOSAL` to simulate both the passage of the $\tau^{\pm}$ through matter and its subsequent decay; and `CORSIKA` [117, 118] to simulate the air shower induced by the decay products. Both the newly built code and interfaces to external modules are written in the `Julia` programming language [119]. All code used for this work is publicly available on GitHub [120].

Before describing the workings of the code, we will summarize our conventions and give an overview of the internal `struct`s that undergird the simulation. The `SimulationConfig` contains all the configuration information for setting up the simulation sequence. This includes metadata about the simulation run, the injection parameters, the charged lepton propagation energy cut settings, and the geometrical configuration. All inputs to this configuration should follow the unit conventions of TAMBO, in which eV=c=$\hbar$=1. Common units are available through the `units` utility, *e.g.* an energy of 1 PeV would be expressed as `1units.PeV`. Additionally, we adopt the convention that a particles direction is aligned with its momentum. This is opposite the convention of many neutrino observatories which choose to have direction anti-aligned following the convention of telescopes.

the interna geometry is handles by the `Geometry struct`. The `valley` field contains a spline of the valley and may be called as a function. Since the simulation coordinate system is centered on the TAMBO detector, this `struct`also contains the `tambo_offset` which converts to a coordinate system whose z-coordinate is at sea level. Additionally, this contains the `tambo_bounds` field, which is a list of xy-coordinates that roughly give the outer limits of the TAMBO detector, with the z-coordinates being given implicitly by the height of the valley. These are used when deciding whether to propagate an event in `CORSIKA` and will be discussed further later. The last aspect of `struct`is the ρ`air` and ρrock which give the mass density of the air and rock. By default, these are 1.2×10^{-3} g/cm^3 and 2.6 g/cm^3 respectively, expressed in TAMBO units. It should be noted that all geometry in the TAMBO simulation assumes that one is localized to a region where the Earth can be approximated as a plane. The maximum error that this induces is given by:

$$\Delta z_{\mathrm{max}} = \frac{r_{\mathrm{sim}}^2}{2R_{\oplus}}$$

where r_{sim} is the radius of the simulation region, and $R_{\oplus}$ is the radius of the Earth. This means that for the 30 km-by-40 km simulation region, the maximum error is $\sim$ 200 m. This is not insignificant *prima facia*, but, as we will describe later, almost all of the of the neutrinos are simulated within 10 km of the detector, where the maximum error is less than 10 m.

This `struct`can be be loaded by specifying a `JLD2`-formatted file that has two keys—`spline` and `mincoord`—and a `Coord struct`that gives the latitude and longitude of the center of TAMBO in radians. The former should give a two-dimensional spline that takes in an xy-position and outputs the height of the valley. These positions should be in unit of meters and the first knot should be at

x=y=0 m. The latter key should be a `Coord struct`that gives the latitude-longitude of the origin in radians.

Next, we turn our attention to the `Injector` object, which as the name may suggest, handles initial state injection. This `struct`has six fields: `config`, `powerlaw`, `xs`, `anglesampler`, `injectionvolume`, and `geo`. The first field contains an `InjectionConfig struct`that has configuration information used to construct the `Injector`. The last field is an instance of previously discussed `Geometry struct`that handles all geometrical considerations. The middle four fields are used for sampling the initial neutrino energy, the outgoing tau lepton energy, the initial neutrino direction, and the interaction vertex. These should be instances of `PowerLaw`, `CrossSection`, `UniformAngularSampler`, and `InjectionCylinder struct`s respectively. These are all general examples of `Sampler`s, other features of which will be disussed later. While there are other ways to sample initial energies and angles beyond what is currently implemented, these options should be sufficient as non-powerlaw energy distributions and non-uniform angular distributions can be attained by event weighting.

The event injection, carried out by running the calling the `Injector` as a function, first selects an incident direction uniformly in phase space—*i.e.* uniform in the azimuthal angle and uniform in the cosine of the zenith angle—and an initial energy. We then sample a point of closest approach on a two-dimensional surface oriented perpendicular to the incident direction. This shape is a cross section of the `injectionvolume`, and as `InjectionCylinder` suggests, this surface is a circle in the default implementation.The radius of this disc, $r_{\mathrm{inj.}}$ should contain the detector; in this work, we set it to 2 km, but this may be changed in the future. We note here that more studies are required to know if this is the most efficient choice since TAMBO is more approximately a rectangle than a circle. Having picked a direction and point of closest approach, we can then backtrace the direction of the neutrino and determine whether it had to cross the Earth in order to arrive at TAMBO.

Next, we must determine the energy of the neutrino when it enters the simulation region. If the neutrino did not have to traverse the Earth, we may draw an energy from a power-law distriubtion and set the equal to the energy of the neutrino at the edge of the simulation region. If the neutrino had to cross a portion of the Earth, we sample an energy from the same power-law distribution and set this equal to the energy of the neutrino at the surface of the Earth and propagate the neutrino to the detector. At the energies that TAMBO can probe, tau regeneration plays a crucial role in the tau-neutrino propagation. To simulate this, we interface to the `TauRunner` package. Although there are a variety of packages that specialize in treating tau regeneration, `TauRunner` is one of two—the

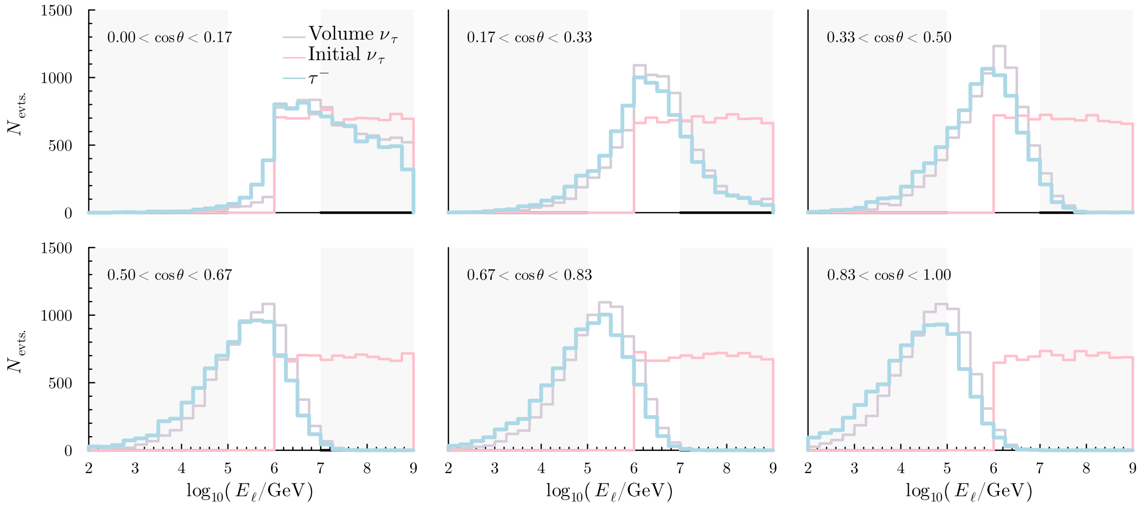


Figure 3.11: ***Lepton energies at different injection stages as a function or angle.*** All events are drawn from a power law with a spectral index of 1, as evidenced by the pink line being flat in each panel. As we move from Earth-skimming neutrinos in the top, left panel to core-traversing neutrinos in the bottom, right panel, the distribution of energies at the TAMBO simulation volume moves to lower energies due to tau regeneration; however, the total number of events is conserved. The distribution is further shifted to lower energies when the ν_τ creates a $\tau^\pm$. This effect is less pronounced for higher-energy neutrinos since the charged-current interaction becomes more elastic. The shaded regions roughly correspond to the energy range to which TAMBO is sensitive. Only angles shown in the top row of panels were considered in previous sensitivity estimates. Clearly, steeper angles contribute to the total number of neutrino interactions near the detector; however, further study is required to know if these events will trigger the detector.

other being `NuPropEarth` [80]—that gives the ν_τ flux. As such, only these two software are suitable for event injection; however, we opt for `TauRunner` since its `Python` implementation is easier to interface with from `Julia`. Once `TauRunner` has propagated the neutrino to the surface of the simulation volume, we set the final neutrino energy from `TauRunner` to the energy in the simulation volume. We now know the energy of the neutrino that we will force to undergo a charged-current interaction, and as such can sample the outgoing $\tau^\pm$ energy. To do this, we sample from the differential cross section as computed by Cooper-Sarkar, Mertsch, and Sarkar [121] and as implemented in [122]. We assume that the momentum of the outgoing tau lepton is parallel to that of the neutrino, and as such we only need to sample from the Bjorken-y distribution. The energy distributions for neutrino energies at Earth's surface, neutrino energies in the simulation region, and for $\tau^\pm$ are shown for different injected zenith angles in Fig. 3.11.

At last, we can sample an interaction vertex and thus conclude the event injection. We must make sure that we are sampling all vertices that could lead to the air-shower from the $\tau^\pm$ to trigger the detector. Since higher-energy tau leptons can travel for larger distances before decaying, this

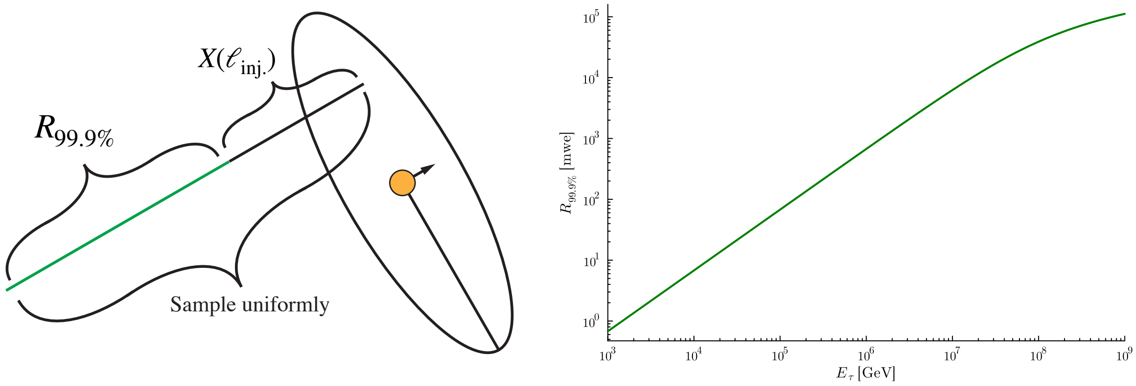


Figure 3.12: ***Diagram of ranged vertex sampling procedure.*** The so-called injection disc—the large black ring—is centered on the center of the detector, represented here as an orange circle, and is oriented perpendicular to the direction of the incident neutrino direction. An interaction vertex is the sampled by computing $R_{99.9\%}$—shown in the plot on the right—for the exiting $\tau^{\pm}$ and sampling a distance in the range $[0,\, \ell_{\text{inj.}} + \ell\,(R_{99.9\%}|\theta, \phi)]$ uniformly in column depth.

process should depend on energy. We solve this by computing $R_{99.9\%}(E_\tau)$, the 99.9% quantile of distances τ leptons can travel before decaying. This is computed for a number of energies, and is then fit to the parametrization:

$$R_{99.9\%}(E_\tau) = \frac{1}{\beta} \log \left[1 + E_\tau \frac{\beta}{\alpha} \right].$$

After propagating 10^5 tau leptons at energies ranging from 10^3 GeV to 10^8 GeV with the PROPOSAL software [123], we found best-fit values of $\alpha = 1.473 \times 10^3$ GeV mwe^{-1} and $\beta = 2.63 \times 10^{-5}$ mwe^{-1}. We then sample a distance from the point of closest approach in the range $d \in [0,\, \ell_{\text{inj.}} + \ell\,(R_{99.9\%}|\theta, \phi)]$, where $\ell_{\text{inj.}}$ is an injection parameter and $\ell(X|\theta, \phi)$ a function that converts a column depth to a distance, is sampled uniformly in column depth. We then set the interaction vertex a distance d from the point of closest approach, in the direction opposite the neutrino momentum.

This whole injection process creates an `InjectionEvent struct`, which contains three `Particle structs`— `initial_state`, `entry_state`, and `final_state`—that tell about the state of the particle at the different stages of the injection process. It has one additional field, `genX`, which gives the total column depth that was sampled from when determining the interaction vertex since this will be necessary when weighting the particle.

Next, we pass the final-state $\tau^{\pm}$ to `PROPOSAL` for propagation. This requires creating a `ProposalPropagator struct`. This `struct`contains `particledef_def` and `crosssections_dict` fields. These contain definitions that are useful to precompute as they do not change when propagating different charged particles; however, it is expected that the user should not need to interact with these. One may call

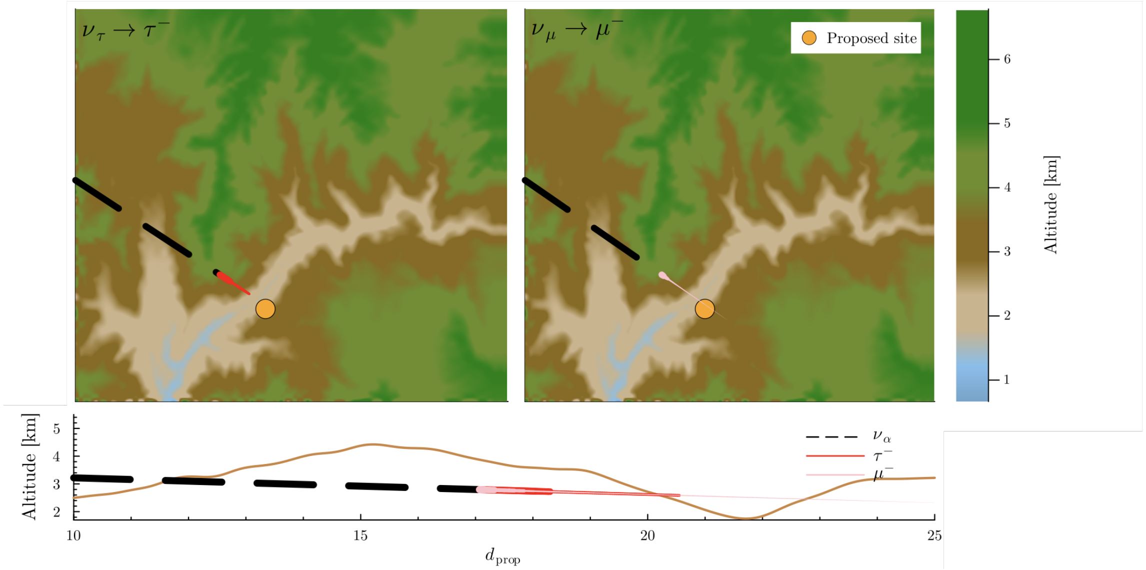


Figure 3.13: ***Propagation process for tau and mu neutrinos.*** The top panels show a bird-eye view of the propagation process, while the bottom panel shows the process from a vantage point perpendicular to the neutrino momentum. The thickness of the line in each plot is proportional to the nergy of the lepton. Since the simulation does not differentiate between neutrino flavors within the simulation region, we represent the neutrino as a dashed black line in both cases. Once a charged-current interaction takes place, we simulate the energy losses of the particle in matter. As mentioned in the main text, the charged mu lepton loses energy much more quickly than the $\tau^\pm$, and does not decay as promptly.

the `ProposalPropagator` as a function with a `InjectionEvent` and `Geometry` as arguments, and the `final_state` of the event will be propagated through the environment specified by the geometry. This can also be used for propagating any charged lepton by passing an appropriate `Particle` into the function call with a `Geometry`.

Fig. 3.13 shows the result of propagating a muon and $\tau^\pm$ resulting from identical, up to flavor, incident neutrinos. The top panels show a top-down view of the neutrino, represented as the dashed, black line, coming in from the north west. The transition to solid red and pink lines represent the conversion to a $\tau^\pm$ and muon respectively, and the thickness of this line is proportional to the energy. As one can see the muon loses energy much more rapidly than the $\tau^\pm$, and the $\tau^\pm$ decays in the air due to its muc shorter lifetime. This underscores the earlier discussion about TAMBOs ability to see a tau-pure flux. We neglect to show a panel for an incoming electron neutrino since the electron will stop within 20 m of the interaction vertex, and thus will create only a point at the resolution of this plot.

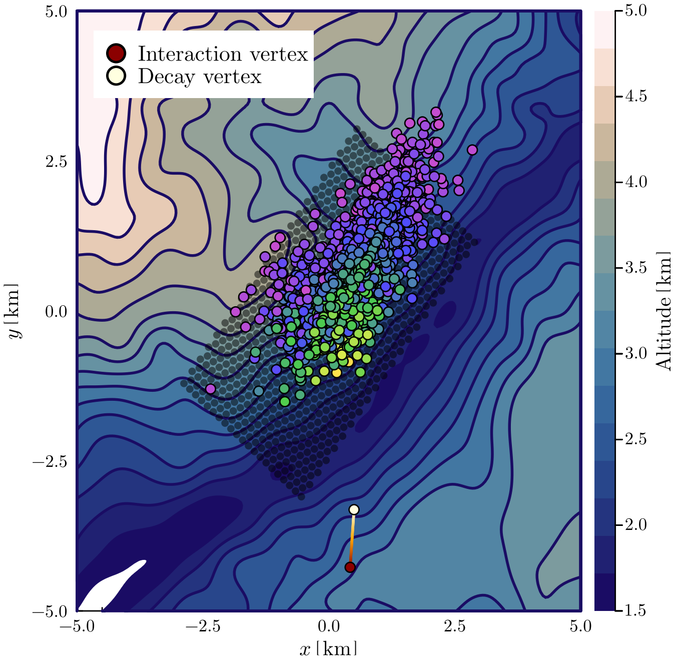


Figure 3.14: ***Top-down view of CORSIKA output overlain on the mountainside.*** Each dot colored represents a particle that reached the plane in the CORSIKA simulation. The color corresponds to time, with warm colors corresponding to earlier arrival times. The transparent, black points represent the position of TAMBO's detection units. Additionally, the interaction and decay vertices for the incident ν_τ and $\tau^\pm$ are also shown as a maroon and white scatter point respectively. The incident neutrino had an energy of 200 PeV.

If the $\tau^\pm$ decays hadronically, we take the decay products and use CORSIKA to simulate the extensive air shower. The typical usage for CORSIKA is simulating down-going air showers on a horizontal plane. We need to depart from this typical case in two ways. First, we need to simulate up-going air showers since we focus on this region of phase space to reject background. Second, we need to determine the output of the air shower on the inclined plane we use to approximate the valley.

At the time of development, these two requirements could not be simultaneously satisfied in CORSIKA7, the latest official release. We were able to obtain a beta version of CORSIKA8 from the CORSIKA developers and began developing within that framework. This is the first project to use this new CORSIKA in production for any experiment, and we have been working closely with the CORSIKA developers and occasionally making improvements to the CORSIKA8 code. One such improvement is a modification of the CORSIKA geometry to make it more general and suit TAMBO's unique needs. We expect them to make it into the production version of CORSIKA when it is released. Furthermore, this is the first use of CORSIKA8 to simulate up-going events, and have received interest from other experiments in using the code we developed in their simulation chains.

Running CORSIKA is the most time-consuming step of the simulation chain and as such, we perform some preliminary checks to see whether an event has any chance of producing hits within the detector. The simplest of these checks makes sure the decay happens in the air, and in fact, this is the cut that cuts most of our events, removing approximately 90% of injected events. We then perform checks on the direction of the particle relative to the plane's position to ensure it is not heading away from the detector approaching at such an oblique angle that the air-shower simulation will not terminate in a reasonable amount of time. Lastly, we check that the air shower has a path to the detector that is not blocked by the mountain. This could happen when the decay occurs in a gap between the mountains, but not in the instrumented portion of the valley. To do this, we check whether any path to one of the previously mentioned tambo_bounds does not pass through a mountain. If any such path exists, we propagate the particle.

Approximately 2.5% of events pass all checks. While this is somewhat low and perhaps points to an inefficiency in the injection step, it is still sufficiently fast that it is not a bottleneck in our simulation. This is because the injection steps run at about 18 Hz, meaning that we get one event that we will be run by CORSIKA per second; however, the CORSIKA simulation takes $\mathcal{O}(10\text{ s})$ and as such, even a perfectly efficient injection would only result in a 5-10% increase in the simulation speed.

Having simulated the shower, we then check if the shower products trigger the detector. To do this, we define a number of detector units arranged in a hexagonal grid. In the initial simulations, the detectors are circular and have a radius 1 m. If any of the particles from the CORSIKA simulation are within this radius of any detection unit, we add the event number to the total number of particles seen by that detection unit.

Defining the "number of particles" involves some subtlety since the electromagnetic component of the CORSIKA shower is thinned to boost computational efficiency. Particles belonging to this class include a so-called weight, which, roughly speaking, corresponds to the number of similar particles that would have arrived had thinning not taken place. To convert these events into an integer number of particles, we sample an integer from a Poisson distribution with a mean equal to the event weight. An individual detection unit is triggered if it detects three or more particles, and an event triggers detector readout if the number of particles in all activated detection units is at least 30.

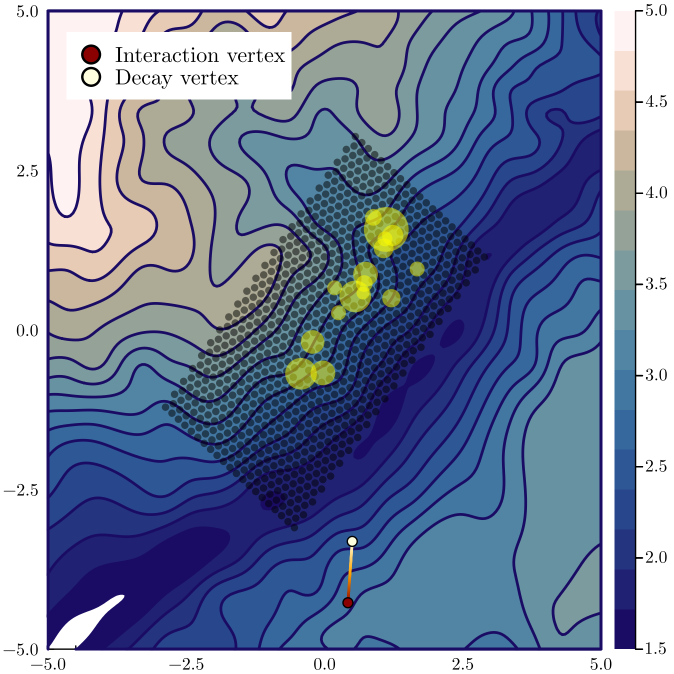


Figure 3.15: ***Triggered modules from a cosmic-ray air shower in previous figure.*** As in Fig. 3.14, each dot is a detection module. Black indicates an untriggered module, while yellow indicates a triggered module. The area of the yellow dot is proportional to the number of particles collected by that detector.

We then convert the events that pass this trigger criterion to a rate of events at the detector. To accomplish this, we must undo all unphysical choices that were made in the simulation chain, a process known as event weighting. Since `TauRunner`, `PROPOSAL`, and `CORSIKA` simulate according to physical processes, we need only be concerned with removing unphysical choices from event injection. Again drawing inspiration from [124], we define the probability of generating an event according to our injection procedure as:

$$p_{\mathrm{gen.}} = \frac{1}{A_{\mathrm{inj.}}\Omega_{\mathrm{inj.}}}\frac{\rho\left(\vec{x}_{\mathrm{int.}}\right)}{X_{\mathrm{gen.}}}\frac{1}{\sigma(E_\nu^{\mathrm{vol.}})}\frac{\partial\sigma}{\partial y}(E_\ell|E_\nu)\frac{\Phi(E_\nu^{\mathrm{gen.}})}{\int_{E^{\mathrm{inj.}}_{\mathrm{min.}}}^{E^{\mathrm{inj.}}_{\mathrm{max.}}}\mathrm{d}E_\nu\Phi(E_\nu)}.$$

Where $A_{\mathrm{inj.}}$ is the area of the injection disc, $\Omega_{\mathrm{inj.}}$ is the solid angle of the directional sampling, $\rho(\vec{x}_{\mathrm{int.}})$ is the mass density evaluated at the interaction vertex, $X_{\mathrm{gen.}}$ is the total column depth in the range $[0,\, \ell_{\mathrm{inj.}} + \ell\left(R_{99.9\%}|\theta,\phi\right)]$, $\sigma(E_\nu^{\mathrm{vol.}})$ is the total neutrino cross section evaluated at the neutrino energy when it enters the simulation volume, $\frac{\partial\sigma}{\partial y}(E_\ell|E_\nu)$ is the differential cross neutrino cross section, Φ is the power-law spectrum according to which the initial, *i.e.* at the surface, neutrino energies were drawn from, and $E_{\mathrm{min.}}$ and $E_{\mathrm{max.}}$ are the minimum and maximum energies that can be sampled. It

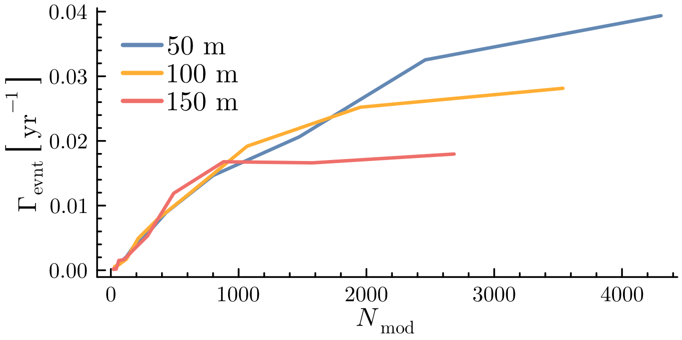


Figure 3.16: ***Event rate as a function of number of deployed detection modules.*** The three lines show the expected event rate for three different inter-module spacing. Under the current simulation settings, the event rate is sensitive only to the total number of detectors and not the inter-module spacing. We suspect this is related to the energy cut settings of the `CORSIKA` simulation discussed in the main text.

should be noted that all quantities with inj. are metaparameters of a given injection and do not vary from event to event.

The true rate of a physical event may be computed with:

$$\Gamma_{\mathrm{evt.}} = \frac{1}{p_{\mathrm{gen.}}}\frac{X_{\mathrm{phys.}}}{M_{\mathrm{iso.}}}\frac{\rho_{\mathrm{phys.}}\left(\vec{x}_{\mathrm{int.}}\right)}{X_{\mathrm{phys.}}}\frac{\partial\sigma}{\partial y}(E_\ell|E_\nu)\Phi_{\mathrm{phys.}}(E_\nu^{\mathrm{gen.}}),$$

where $X_{\mathrm{phys.}}$ is the column depth traversed by the neutrino en route to the interaction vertex, $M_{\mathrm{iso.}}$ is the isoscalar nucleon mass, $\rho_{\mathrm{phys.}}(\vec{x}_{\mathrm{int.}}$ is the mass density of the physical medium evaluated at the interaction vertex, and $\Phi_{\mathrm{phys.}}$ is the physical neutrino spectrum for which one wishes to compute the rate. One may then compute the average rate for an ensemble of events by taking the average, *i.e.*:

$$\Gamma_{\mathrm{tot.}} = \frac{1}{N_{\mathrm{inj.}}}\sum_{\mathrm{evts.}} w_{evt.}$$

Additionally, we typically define the *oneweight* for an event as:

$$w_{\mathrm{evt.}} = \frac{\Gamma_{\mathrm{evt.}}}{\Phi_{\mathrm{phys.}}(E_\nu^{\mathrm{gen.}})}.$$

This is a convenient quantity since one may then find the rate for any flux by taking the product of the flux and the oneweight. Furthermore, this quantity may be used to find the so-called effective area of TAMBO by computing the average value in a given zenith and energy range.

We used the software to simulate injection on a flat portion of the valley to evaluate TAMBO's performance. We simulated with an injection radius of 2 km, resulting in 4 km of coverage on the mountain's surface. Of the 10^5 events we injected, only 2,759 were passed to `CORSIKA`. Of the 97,241 events we did not pass to `CORSIKA`, 88,039 were cut because the $\tau^{\pm}$ decay occurred within the mountain. We then ran the triggering protocol on the resulting `CORSIKA` output and computed the total event rate for different detector configurations for the flux from [67].

In Fig. 3.16, we show the event rate as a function of the total number of detection modules for arrays with 50 m, 100 m, and 150 m inter-module spacing. The flattening at large number of modules is caused by the array extending beyond the injection region. These results suggest that the total event rate does not depend on the inter-module spacing. This does not agree with the previous estimation of the event rate from Ref. [89]. Further disagreements between these calculations emerge if we linearly extrapolate the event rate from the regime we simulate to the 22,000 modules used previously. Our calculation predicts a rate of approximately 0.4 events per year, whereas the previous analysis predicts approximately six events per year.

We are investigating the possibility of this discrepancy being caused by the `CORSIKA` energy cut setting. This setting controls the energy threshold below which particles are ignored. To allow the simulation to finish in a reasonable time frame, the threshold is currently set at 1 GeV; however, we estimate that our detection module should be sensitive to particles with energies as low as 5 MeV. Initial estimates indicate that including all particles down to this energy should raise the total particle count by a factor of 10-25 and increase the density of particles. The increase in the total number should increase the fraction of events that trigger the detector, and the higher density of particles should induce a dependence on the inter-module spacing. However, since the spatial distribution of particles is a function of their energies, it is non-trivial to understand how these cut particles will be distributed on the mountain. Unfortunately, running simulations with an energy cut of 5 MeV is computationally tractable since a single shower could take a week to simulate.

We are currently considering a variety of approaches to working around this constraint to get an estimate that accurately reflects the detector's capability. A straightforward approach would be to oversize the detection modules to approximate the effect of the increased particle density; however, the lower-energy particles are not distributed uniformly in space, an effect that this approach does not account for. We are also investigating developing phenomenologically driven models of shower development that would allow us to extrapolate the profile below the energy cut.

While the work presented here represents a significant step for TAMBO's simulations, improvements can still be made here. In the last few paragraphs of this section, I will comment on some of the improvements that I think are important as a note for posterity. The most critical change that needs to be implemented is moving `TauRunner` from the injection step to the weighting step. As it is currently implemented, we couple the TauRunner simulation, which implicitly contains cross-section information, to the expensive CORSIKA simulation. This is an issue as there are significant uncertainties on the high-energy neutrino cross section, and studying this is very much of interest to TAMBO. Including TauRunner at the injection stage has the additional negative effect that we occasionally get neutrinos at the simulation region below the energy threshold of TAMBO, wasting computational resources. This step requires a change to the weighting procedure, as a distribution of surface energies can give rise to the neutrino event at the simulation region. Thus, one must integrate this distribution.

Another improvement that comes to mind is implementing a mesh geometrical handling in CORSIKA. This does not impact the physics—and thus is of lower priority—but would allow for PROPOSAL and CORSIKA to both be run within the CORSIKA framework, decreasing the chance of introducing bugs through interfacing to multiple packages. Currently, since it cannot handle the complex valley geometry, PROPOSAL cannot be run within CORSIKA. This would facilitate moving our injection to LeptonInjector [124], which currently supports this mesh geometrical framework. This would allow all geometry to be handled within the same framework, removing the need for coordinate transforms when interfacing multiple packages. In turn, this reduces the chance of introducing subtle bugs. Both LeptonInjector and CORSIKA are well-tested and have more extensive user and maintainer bases. As such, switching to these should improve the long-term reliability of the TAMBO simulation framework.

Chapter 4

Open-Source Simulations, Reconstructions, and Technologies for the Future of Neutrino Telescopes

The role of ML in IceCube's recent observations of the neutrino emission from the Galactic Plane and from extragalactic point sources makes it clear that this technique will drive immense progress in neutrino astronomy. However, the results of the previously mentioned Kaggle challenge make it abundantly clear that neutrino astronomy is behind the cutting-edge ML techniques. If we are to catch up and realize the potential offered by ML, we need all the personpower we can muster.

Fortunately, the neutrino telescope community is growing rapidly. Currently, five new telescopes are being planned and deployed, increasing the instrumented volume from 1.5 km^3 to 24 km^3 in the coming decade. These telescopes operate under nearly identical detection principles. Thus, if experiments operate within a common language, they can share techniques and more rapidly close the gap between ML capabilities in industry and neutrino astronomy. However, each experiment currently uses a proprietary simulation framework. This makes developing combined solutions untenable, resulting in redundant efforts and reduplicated work.

In this section, we will describe `Prometheus` an open-source simulation framework for neutrino telescopes. It allows one to simulate ice- and water-based detectors with arbitrary geometries and thus, provides a common language for all current and planned Cherenkov neutrino telescopes. We will then apply this open-source simulation to new technologies that may find application in neutrino telescopes. In the first example, we use `Prometheus` simulation to design an ML-based reconstruction that is capable of running at speeds faster than the trigger rate in neutrino telescopes. In the second example, we will investigate the possibility of compressing data from neutrino telescopes using quantum processors.

4.1 `Prometheus`: an Open-Source Neutrino Telescope Simulation

Neutrino telescopes [125] are gigaton-scale neutrino detectors that use naturally occurring media such as glaciers [110, 111, 78], the Pacific Ocean [87], lakes [126, 127], seas [128, 129], insterstellar dust [79], or mountains [88, 89, 130] as a neutrino target. The subset of these that are deployed in liquid or solid water–hereafter referred to as water and ice respectively–have a long history [131], and most of the features of current and future detectors can be traced back to the DUMAND project [132]. The largest currently operating of these detectors is the IceCube Neutrino Observatory [6] located near the geographic South Pole. Two other collaborations are currently constructing detectors: the KM3NeT collaboration is constructing ORCA and ARCA [129] in the Mediterranean Sea and the BDUNT collaboration is currently building Baikal-GVD in Lake Baikal in Russia [127]. Additionally, two new experiments–P-ONE [87] off the coast of Vancouver in the Pacific Ocean and TRIDENT [128] in the South China Sea–and expansions of the IceCube Observatory [133, 78] are under development.

Since all these experiments operate on the same detection principle and are deployed in water or ice, they share many technological features. Each detector is comprised of individual optical modules (OMs) capable of detecting Cherenkov photons emitted by the charged byproducts of neutrino interactions. The arrangement and details of each OM vary from one detector to another depending on the optical properties of the medium, physics goals, and historical context of construction.

Unsurprisingly, these commonalities result in similar simulation chains. Most simulation chains follow some variation of the steps outlined in Fig. 4.1, before eventually convolving the detector response with the distribution of photons that arrive at the OMs. Since only this last step is not generic, there is an opportunity to develop a common software framework. `Prometheus` aspires to meet this opportunity by providing an integrated framework to simulate these common steps for arbitrary detectors in water and ice. The flexibility allows one to optimize detector configurations

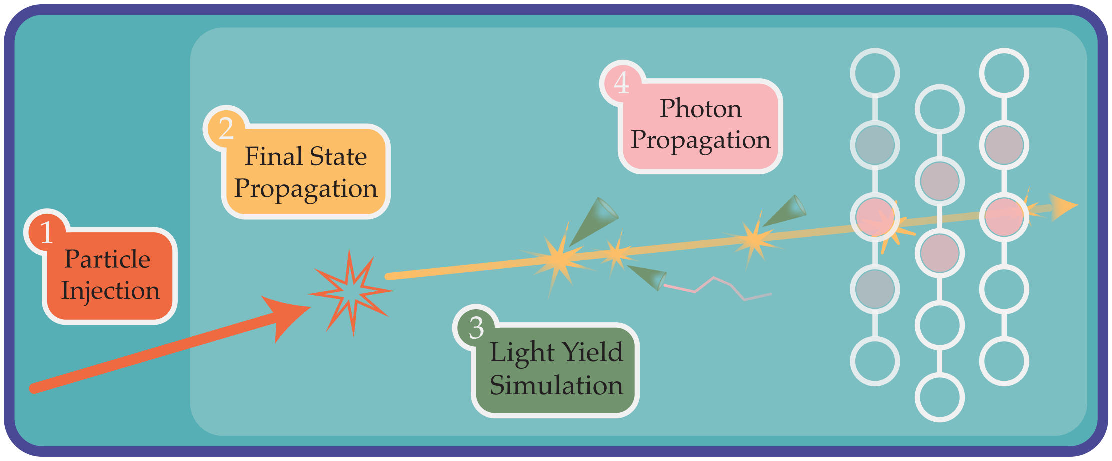


Figure 4.1: ***Schematic showing the physical processes* `Prometheus` *models.*** (1), `Prometheus` selects an interaction vertex within *simulation volume*, depicted here by the lighter-colored region. (2), the final states of this interaction are then propagated, accounting for energy losses and any daughter particles which may be produced. (3), these losses are then converted to a number of photons. (4), finally, these photons are then propagated until they either are absorbed or reach an optical module.

for specific physics goals, while the common format allows one to develop reconstruction techniques that may be applied across different experiments.

`Prometheus` builds upon several decades of experience in the design of neutrino telescope simulations by using publicly available, well-maintained software whenever possible. Neutrino event generation in these types of detectors can be traced to the first neutrino telescope event generator [134, 135, 136, 137]. The first simulation of a neutrino telescopes in ice dates back to AMANDA and was called `NuSim` [135], originally written in `Java`, while in water the earliest reference can be traced to ANTARES [138]. `NuSim` was latter ported to `C++` and released as `ANIS` [136]. This was then adapted into an internal IceCube event generator called `NuGen` [139]. Recently, the IceCube collaboration has released a new neutrino event generation that builds on these efforts called `LeptonInjector` [124]; see Ref. [140] for a similar effort in KM3NeT. `LeptonInjector` [141] performs only neutrino injection around the detector and leaves neutrino transport through Earth as an *a posteriori* weight [142] since it can be readily performed by packages such as those given in Refs. [137, 38, 143, 144, 80, 145, 146].

The propagation of high-energy muons is described in detail in Ref. [147]; see [148] for a recent revision. Muon propagation in detailed Monte Carlo simulation was implemented in `MUSIC` [149], primarily used in water-based neutrino experiments, and `MMC` [150] in ice-based experiments. The latest and most up-to-date muon propagator optimized for neutrino telescopes is called `PROPOSAL` [18]

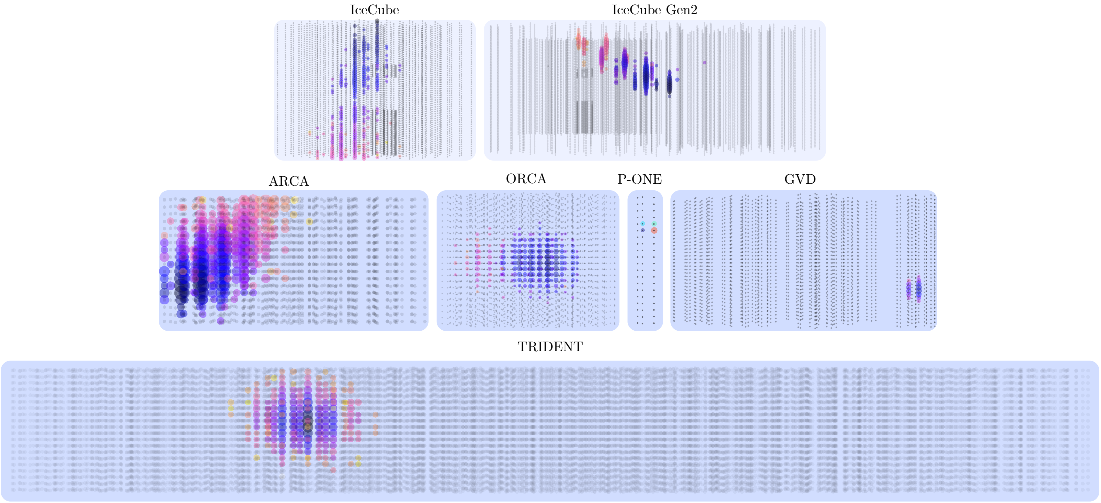


Figure 4.2: ***Event views for various detector geometries.*** This shows the events created by either ν_μ charged-current or ν_e charged-current interactions in a variety of geometries of current and proposed neutrino telescopes. Each black dot is an OM, while each colored dot indicates the average time at which photons arrived at the OM; black indicates an earlier arrival, orange indicates a later arrival, and purple an arrival in between. Furthermore, the size of the colored spheres is proportional to the number of photons that arrived at the OM. Detectors which appear against lighter blue backgrounds—the top row—are ice-based, while those against the darker blue backgrounds are water-based.

and builds on `MMC`. `Prometheus` uses `PROPOSAL` to simulate the propagation of muons. Tau propagation is also handled by `PROPOSAL`, though in most of the energy range of the experiments considered here, the tau losses are negligible; see [151, 32, 152, 153, 154] for discussions on tau energy losses.

The emission of Cherenkov light from hadronic and electromagnetic showers in water is discussed extensively in Refs. [155, 156]. In these references, the emission of light was parameterized from dedicated GEANT4 simulations, which have been recently refined in [157, 158]. The emission of light from hadronic or electromagnetic showers has been implemented in the `Cascade Monte Carlo` (`CMC`) package by the IceCube collaboration [124] following the physics outlined in [156]. Unfortunately, `CMC` is not publicly available and is only usable in ice. For this reason, we have reïmplemented the light yields produced by showers in `Prometheus` following the parameterizations given in [156, 157, 158]. These are implemented in a module called `Fennel` in `Prometheus`.

Finally, `Prometheus` solves the light transport problem using two different modules. In the case of light propagation in ice, we use the standalone, open-source version of `PPC`, which is the ray tracer used by the IceCube collaboration and can be found in [159, 160], while in the case of water, we implement our own ray tracing routines in a module called `Hyperion`,.

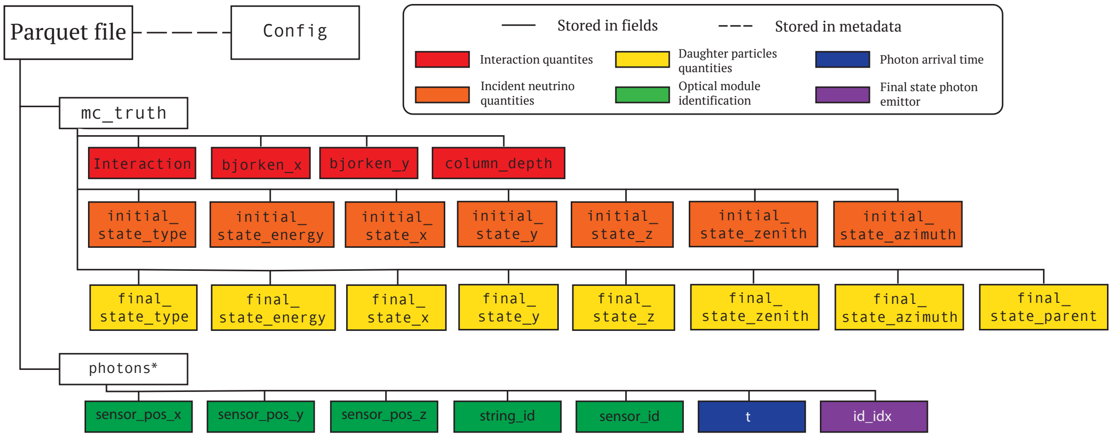


Figure 4.3: ***Output format for default*** **`Prometheus`** ***parquet files.*** The solid lines indicate that information is stored in `fields`, while the dashed line indicates that information is stored in the metadata. We delay detailed discussion until later, when we explain each field and compute basic quantities of interest. Fields with an asterisk can be renamed by the user to be compatible with legacy conventions.

These four steps—event injection, final state propagation, light yield simulation, and photon propagation— each come with their own set of unique challenges. As noted above, we will use publicly available and well-maintained packages to address these challenges whenever possible. However, in `Prometheus` we take these challenges as opportunities to provide new solutions to them when publicly available software is lacking. In what follows we will discuss the output of `Prometheus` and then outline the code structure and summarize the corresponding physics of each piece of code.

`Prometheus` events are serialized as `Parquet` files. `Parquet` is a columnar format developed by Apache [161] that supports nested data. Columnar storage has been shown to yield improved performance when processing data and increased data storage compression [162]. `Parquet` stores nested data structures using the technique introduced by Google in [163].

When choosing the output format for `Prometheus` we have surveyed multiple options used in the community [162] such as `HDF5` [164] and `ROOT` “n-tuples” [165]. Ref. [162], studied different format disk usage and access speed in the context of collider experiment events. They found that `Parquet` file size is comparable to `HDF5` and `ROOT` for uncompressed files and improved over the former when compressed by `zlib`. They found that `Parquet` files read access per event is a factor of five times faster when compared to `HDF5` when using uncompressed `Parquet` files, while a factor of three when compared to the compressed version. On the other hand, `Parquet` files have been shown to be a factor of three times slower than `ROOT` “n-tuples” when reading them. We have opted not to use `ROOT` for interoperability reasons and to reduce dependence on additional libraries needed to work with Prometheus output. Of the interoperable formats, we have decided to use Parquet over HDF5 due to its improved performance as discussed above.

Here, we will broadly describe the information contained in the output files and the general structure, delaying a detailed example until later. The output files contain two fields, mc_truth and photons. For compatibility with naming conventions used in beta versions of the software, the first of these fields may be changed by the user. This can be done by setting the "photon propagator/photon field name" configuration field to the desired name.

The mc_truth field contains information about the injection quantities, such as the interaction vertex; interaction Bjorken variables; column depth traversed by the initial neutrino; the initial neutrino type, energy, and direction; and the final state types, energies, directions, and parent particles. Since, in general, there can be any number of final-state particles, all final-state data are stored as one-dimensional arrays. The order of the arrays is determined by traversing the MC tree of children, depth first.

The photons field contains information about photons that reach OMs. This includes the OM identification numbers, OM position, photon arrival time, and an identification index that connects the photon to the final-state particle that created it. If available, the photon arrival direction and position of the photon on the OM will also be saved. This availability depends on which photon propagator is being used. These last two data can be useful for, *e.g.*, simulating the OM acceptance in cases where it is heavily directionally dependent.

The configuration information is also stored in the Parquet file as metadata. Once extracted, this may be dumped to a json file, and fed back into Prometheus to resimulate with the same parameters. This may be useful if you want to simulate the same event in different detectors in order to compare performance. We should remark that while most of the code can be seeded to ensure reproducibility, PPC does not allow for seeding. Previously there was a compile-time option to set the random state of PPC, but currently the random state is set using the time of day [166]. Thus, while simulations of water-based detectors are exactly reproducible, simulations for ice-based detectors will produce results that vary within Poisson fluctuations.

In addition to the main Parquet output, some steps in the Prometheus chain produce intermediate files. We delete these files unless they contain information not available in the final output. Currently, only the LeptonInjector .lic files, which contain the configuration information used in injection, meet this criterion. These files are needed in order to weight events to obtain an event rate. While it is possible to regenerate these after the fact, please do not remove these files if you intend to weight events.

At its core, Prometheus is a framework for shepherding particles through the steps outlined in Fig. 4.4 in a consistent manner. In the following sections, we will outline the Prometheus dataclasses that allow for this consistent treatment and explain the interfaces between Prometheus and the external packages. While this is not comprehensive, it should give a sufficient understanding to work with the package. We will point the reader to references that describe the external packages in more detail when appropriate. Along the way, we will point out ways to adapt Prometheus to different simulations needs, including extending Prometheus to work with additional external packages and discussing user-configurable parameters. We will refrain from discussing the “how" of using these parameters, deferring that discussion until later. Instead, we opt to describe the impact of the parameters on the simulation. Unless it is noted otherwise, all parameters can be found in the config file.

Before embarking, we should first discuss Prometheus conventions. Prometheus uses a unit system where the units of length, energy, time, and angle are the meter, GeV, nanosecond, and radian respectively. This means that all user input should be provided in these units and that all output will be provided in these units. When interfacing new, external packages, one should account for unit conversions from Prometheus units to the units of the new package. Furthermore, we follow a convention where the direction vector is aligned with the momentum of the particle. Thus, up-going events result from neutrinos with incident direction between 0° and 90° and down-going events have directions between 90° and 180°. This is the opposite convention that many observatories use, where the direction is anti-aligned with the particle momentum, thus describing where the neutrino originated. To better interface with the larger high-energy physics community, we have chosen the convention that is broadly used in accelerator neutrino experiments.

The fundamental dataclass of Prometheus is the Particle, which minimally contains the particle type, an integer following the Particle Data Group (PDG) convention given in [95]; the energy of the particle at creation; the direction in which the particle is travelling; and the position of the particle that is relevant to the simulations. This position may be either the interaction vertex, as is the case for incident neutrinos, or the point of creation, as is the case for secondary particles. While it is the case that for many situations of interest, the creation point of the secondary particles will overlap with the interaction vertex, this is not always the case. For instance, in the case of

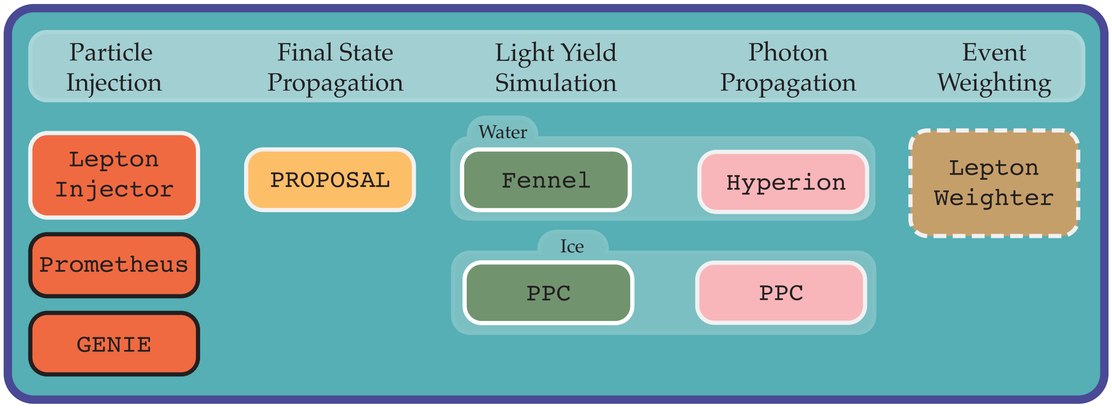


Figure 4.4: ***Summary of packages used for different stages in the code.*** The boxes outlined in white are the default packages used, while boxes outlined in black have optional interfaces. Event weighting has a dashed outline to denote that this step is optional. The default behavior of the light yield calculation and the photon propagation depends on the medium, as is shown by the light shaded regions.

tau-neutrino, charged-current interactions, the final state charged tau lepton may decay at a point significantly offset from the interaction vertex, producing charged particles and leading to unique event signatures [167]. Furthermore, some injection packages give detailed particle output from the initial interaction, and some of these particles may be created offset from the interaction vertex. Thus, we need to make this distinction in the definition of the position in order to accommodate these situations.

We resolve this ambiguity by defining a subclass of the `Particle`, the `PropagatableParticle`, which tracks all particles that can generate energy losses and ultimately light. It is precisely these particles that may be created at an offset from the interaction vertex. Thus, the position of any instance of this class is the point at which it was created, while the position of a `Particle` that is not an instance of this class, is the interaction vertex. In addition to this distinction, this subclass has four new attributes, `losses`, which tracks energy losses of the particle; `parent` which is a `Particle` or `PropagatableParticle` object; `children`, a potentially empty list of `PropagatableParticle` objects; and `hits`, a list of all photons produced by the `PropagatableParticle` that hit an optical module before being absorbed.

The information about the position of the OMs and the propagation medium is contained in the `Detector` object. The coordinate system in which the OM positions are specified has its origin at the water-air interface. Only the relative x- and y-coordinates will affect the simulation, but the z-coordinate plays a crucial role in particle injection and charged lepton propagation. The choice of medium specifies which light yield calculation and photon propagation calculations will be used. In the case of ice, PPC will be used to calculate both quantities, while for water, fennel and Hyperion will be used to calculate the light yield and photon propagation respectively.

While the user may manually specify the position and other properties of each OM, we provide several potentially more expedient ways to specify detectors. The first is through Prometheus geo files. These are text files with a specified format that, at minimum, give the locations of all optical modules and the detector medium. Although this is the only required information, they may include any additional detector metadata as needed.

Prometheus provides geo files with approximate OM locations for the IceCube, IceCube Upgrade, IceCube Gen-2, ORCA, ARCA, Baikal-GVD, P-ONE, and TRIDENT detectors, and each has an associated Earth model that will be used for particle injection and charged lepton propagation. For the water-based detectors, this is the PREM density profile, with an outer layer of water added with a thickness equal to the depth of the water at the detector site. Following LeptonInjector, ice-based detectors at the South Pole, use the PREM model with two layers of ice with two densities. This accounts for the glacial ice and the uncompacted layers of snow and ice above the detector.

In addition to specifying the detector via provided geo files, Prometheus provides utilities for generating new detector geometries. These include utilities to make a line of vertically aligned OMs, or a triangularly, hexagonally, orthogonally, or rhombically arranged set of such lines. These lines may then be combined using Python's built-in + function. This may be convenient for, *e.g.*, designing detectors in the vein of Baikal-GVD or P-ONE which are made up of a number of identical clusters. Detectors constructed in this way can then be exported to a geo for later use. While custom Earth models may be made for such detectors, we include two generic Earth models that may potentially be of use. For ice-based detectors, we use a generic South Pole Earth model from [141], and for water-based detectors, we use the PREM model with 2 km of water appended.

As discussed above, injection is the process of forcing a neutrino to interact, creating particles that may produce light, and thus trigger the detector. This requires balancing the need to simulate all interactions that may cause the detector to trigger while not wasting computational resources simulating events which have a negligible chance of doing so. While this problem has been addressed by a number packages, such as [168, 124, 80]. We will spend limited words describing the approaches these packages take to solve this problem focusing instead on injection options available in Prometheus and possibilities for extending these options.

By default, Prometheus uses LeptonInjector [124] to select the interaction vertex, initial neutrino energy and direction, and final-state energies. The energy sampling is done according to a power-law with an index of γ in the range $E \in [E_{\mathrm{min}}, E_{\mathrm{max}}]$. The incident direction is sampled uniformly in phase space, *i.e.*, uniformly in the azimuthal angle and uniformly in the cosine of the zenith angle. The azimuthal angle will lie in $\phi \in [\phi_{\mathrm{min}}, \phi_{\mathrm{max}}]$ and the zenith lies with $\theta \in [\theta_{\mathrm{min}}, \theta_{\mathrm{max}}]$. The parameters γ, E_{min}, E_{max}, ϕ_{min}, ϕ_{max}, θ_{min}, and θ_{max} can be set by the user.

The interaction vertex can be sampled in one of two ways: RangedInjection or VolumeInjection. These terms are described in detail in [124], but we will briefly summarize them and introduce relevant variables here. At energies above 1 TeV, $\mu^{\pm}$, $\tau^{\pm}$, and some of their daughter leptons, can travel distances $\gtrsim$ 1 km before stopping. This distance grows as the energy of the charged lepton increases, and as such the effective volume of the detector grows with increasing energy. RangedInjection accounts for this phenomenon and samples the distance between the interaction vertex and the detector in a manner appropriate to the particle energy. The maximum radius of closest approach, r_{inj}, and padding beyond the particle range, ℓ_{ec} additionally affect the injection region. VolumeInjection on the other hand, selects the interaction vertex within a cylinder with a symmetry axis aligned with the detector center of gravity in the xy-plane, and with radius and height r_{cyl} and h_{cyl}. This may be useful for simulating ν_e charged-current events, ν_α neutral-current events, or ν_α starting events. The parameters r_{inj}, ℓ_{ec}, r_{cyl}, and h_{cyl} may be set by the user, but we consider these advanced injection options, and by default we will select values that will sample the full injection space, accounting for the scattering and absorption of the medium.

Currently, LeptonInjector cross-section tables only support energies down to 100 GeV. If users wish to simulate neutrino events below this energy, please refer to the next section which describes providing injection from another software and using Prometheus to propagate the final-state particles.

While new injection can be generated only by LeptonInjector, Prometheus can accept injection files from other sources. The current iteration of the code provides interfaces for using both Prometheus and GENIE output files, and any files which take the form of LeptonInjector output. The first may be useful for simulating the same events in different detectors in order to compare detector performance, while the second is useful for energies lower than LeptonInjector's 100 GeV threshold. The last may be used to simulate exotic physics scenarios not supported by standard neutrino injectors. In order to access this feature, one needs only change the injector name, and then supply the name of the injection file in the appropriate field of the configuration file.

We expect that these three options should satisfy almost all needs; however, if a new use case arises that cannot be accommodated, adding a new injection interface is fairly straightforward if tedious to describe in words and varies significantly depending on the injection file format. We will refrain from doing so here, but welcome any user that should find themselves in this situation to contact the authors to avail themselves of this feature.

Once the final states have been generated in the injection step, the resulting particles must be propagated, accounting for energy losses and particle decay. We assume that K^0, $K^{\pm}$, π^0, and $\pi^{\pm}$ begin depositing energy immediately, and as such do not propagate them beyond the point of creation. Furthermore, we assume that all final-state neutrinos do not interact again near the detector, and may safely be ignored. This is a safe assumption since the neutrino interaction length is $\sim 10^7$ mwe at 10^6 GeV.

In order to propagate final-state charged leptons, we rely on the `PROPOSAL` package [123]. This Monte-Carlo-based propagation package includes up-to-date cross sections for ionization; bremsstrahlung; photonuclear interactions; electron pair production; the Landau–Pomeranchuk–Migdal and Ter-Mikaelian effects; muon and tau decay; and Molière scattering. While developing `Prometheus`, the most recent versions of `PROPOSAL` occasionally had trouble being installed via `pip` on certain operating systems. To accommodate these issues, `Prometheus` has interfaces to run with `PROPOSAL v6.1.6` or `PROPOSAL v7.x.x`; however, at the time of writing, these installations issues have been resolved, and so we strongly recommend running with using the latest version.

We have decided not to expose all of `PROPOSAL`'s options to the `Prometheus` user, preferring to restrict our options to those which have the largest impact in order to simplify the configuration experience. If a use case requires an option that is not available by default, they may define a function to create a `PROPOSAL` propagator according to their needs. This may then be interfaced with the appropriate `LeptonPropagator` class in `Prometheus` to supply the desired results. Furthermore, if a user desires to use a different package to propagate leptons, this may be accomplished by creating a new `LeptonPropagator` class and implementing the appropriate abstract methods.

The medium in which the propagation takes place is governed by the same Earth model from the injection step. For the final state propagation, however, we convert the layers given by the PREM, which are fit to a degree-three polynomial, to a layer of constant denisty. The value of the density is the mean of the density at either end of the layer. While this approximation breaks down near the center of the Earth, it holds in the region within 100 km of the detector. This is roughly the maximum range of a charged particle, and so it is only in this region where the approximation holds that final state propagation should be taking place.

After propagating the final states, `Prometheus` must convert the energy deposited in and around the detector to photons. `Prometheus` uses two different packages, depending on whether the detector being simulated is in water or ice. In the former case, we use `fennel`, a new package developed for this work, while in the latter case, we use a standalone version of `PPC` [160]. These both employ parameterizations of dedicated `GEANT4` simulations across a variety of energies.

When modeling neutrino detectors in water, `fennel` [169] is used to calculate the light yields and emission angles for the different losses occurring along a track, and from hadronic showers. It utilizes the parametrizations described in [158] to quickly model the Cherenkov light produced by particles and their secondaries. The parametrizations were produced by fitting `GEANT4` [170] shower distributions. The relevant distribution for light yields is the total track length of charged particles in the triggered shower above the Cherenkov threshold. A comparison between `fennel` and `GEANT4` is shown in Fig. 4.5. There we are comparing simulated electron, π^+, and K_L showers with 1, 5, and 10 GeV energies, respectively. Shown is the differential track length, l, depending on the shower depth z: $\frac{\mathrm{d}l}{\mathrm{d}z}$. For the most significant regions for light emission, the difference between the parametrization and MC simulation is less than 20%. The photons are then handed over to `hyperion` for propagation.

When modeling neutrino detectors in ice, `PPC` [171, 160] is used to calculate the number and angular distribution of photons from electromagnetic (EM) and hadronic cascades. Internally, `PPC` bases the EM photon yield on the parameterization from [158], the same as `fennel`. In this work, the simulated the photon yield and angular distribution from e^-, e^+, and γ—with energies ranging from

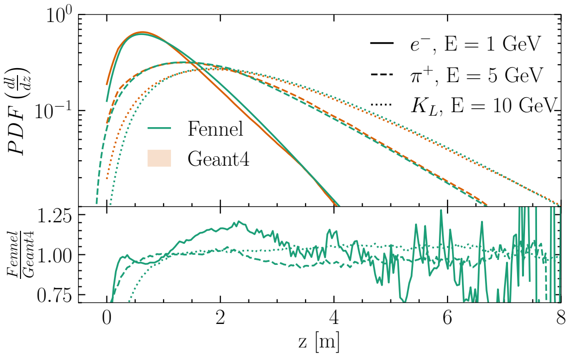


Figure 4.5: ***Comparison between `GEANT4` and `fennel` longitudinal profiles.*** Top: Shown are the differential track lengths for three particle showers each with a different energies. Note, the shift of the maximum depending on energy. Bottom: The ratio of the differential track lengths. For most of the shower's development, the differences between `GEANT4` and `fennel` are below 20%.

1 GeV to 10 TeV—in both ice and water using GEANT4. The resulting longitudinal distributions were then fit to a known functional form. The parameters of this fit for each EM particle agree within a factor of 10^{-3}, *i.e.*, the light yield for all EM particles is the same up to one part in one-thousand. The light yield for hadronic showers is calculated by rescaling the EM photon yield per unit length by a constant which varies for each type hadron.

Within PPC, the distance from the start of the cascade to the point of photon emission is sampled from a Γ distribution in order to properly account for the longitudinal development of the cascade. Details on the parameterization and fit values used in PPC can be found in [158]. The photons resulting from this are then propagated internally by PPC.

The photons generated in the light yield calculation must finally be propagated. This is usually solved via ray tracing of the photon until it is either absorbed or reaches an OM; however, if the Green's function of a photon reaching an OM is known, this may also be used to compute the transmission probability. One may then use the accept-reject method to determine if the OM "sees" the photon.

As is the case for the light-yield calculation, Prometheus uses a different package depending on whether the detector is in water or ice. In the former case, Prometheus uses hyperion, developed for this package, and can take advantage of the Green's function approach. In the latter case, Prometheus uses the same open-source version of PPC which is used to compute the photon yield and which only uses the ray-tracing approach.

Hyperion is used to propagate photons in water and without additional input, uses a Monte Carlo approach to do so. Photons are represented as photon states, which include information about the photon's current position, its direction, time (or distance) since emission, and wavelength. Photons are initialized by drawing the initial state from distributions that represent the photon emission spectrum for the source class to be simulated. The propagation loop involves three main steps: 1) Sample the distance to the next scattering step from an exponential distribution. 2) Calculate whether the photon path (given by the current photon position, distance to the next scattering step, and photon direction) intersects with a detector module. 3a) In case of intersection - the photon is stopped and its intersection position is recorded in the photon state. 3b) In case of no intersection - the photon is propagated to the next scattering site and a new direction is sampled using the scattering angle distribution.

In addition to handling the photon yield in ice, PPC also carries out the photon propagation. PPC uses Monte Carlo methods to propagate the photons until either they reach an optical module or they are absorbed.

The settings by which PPC propagates the photons may be set by a number of tables contained in specially-named text files. These tables set the angular acceptance, size, and efficiency of the OMs; the mean deflection angle of a scattered photon; the depth and wavelength dependence of the scattering and absorption; and the so-called "tilt" of the ice. In the /resources/PPC_tables/south_pole/, we provide tables that parametrize the ice beneath the south pole. This uses scattering and absorption taken from [172], a uniform angular acceptance, and no tilt paramterization, *i.e.* it assumes flat ice. More details than these are known about the South Pole ice, *e.g.* birefringence [173] and a non-zero-tilt [172]; however, these details are very difficult to reconstruct without access to internal information. Furthermore, the parametrization should provide and sufficiently accurate representation of the ice.

While the simulated events can be generated according to arbitrary user input, these can be reweighted to a physical flux. Prometheus does this via the LeptonWeighter [124] package by computing the oneweight. This quantity removes all the generation choices that were made when producing events, and, when multiplied by a desired flux, gives the rate for that neutrino event. Thus, this may be used to reweight to any flux. If the use case does not require physical rates, as is the case in many machine learning applications, weighting is not necessary. As such, we do not perform weighting by default. This may be done after the rest of the simulation has run with either the LeptonInjector HDF5 files or the Prometheus parquet files through the H5Weighter and ParquetWeighter objects respectively.

To weight events, we use the LeptonWeighter package. This is the companion to the LeptonInjector package, and requires the lic configuration files that are output at the injection step. In addition to the lic file, LeptonWeighter needs to be provided differential cross-section files.

These files may be specified by the user; however, if the cross-section files are not provided, Prometheus will attempt to find suitable files. The procedure for this depends on whether one is weighting from a parquet file or a h5 file. In the former case, it will search for appropriate cross section files in the directory provided in at configuration time, since this information is stored in the parquet file. If this fails, then it will default to using the cross sections provided in the

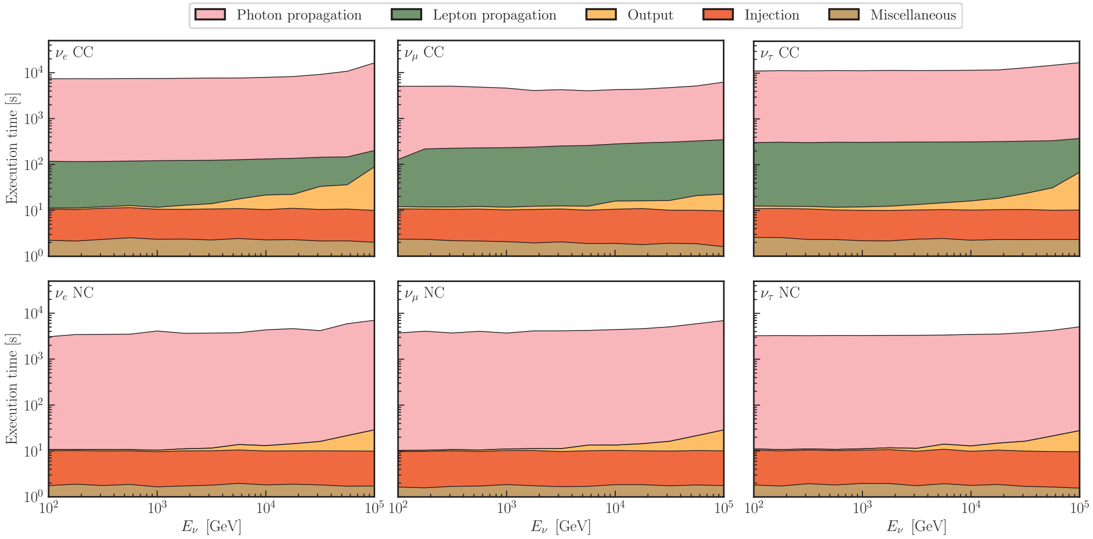


Figure 4.6: ***Wall time per 1000 events as a function of neutrino energy.*** The runtime for thousand events propagated in ice using `PPC`. As expected, higher incident neutrino energies require longer run time. The colors, each corresponding to a different stage of production, are stacked. Each panel shows the runtime for different a different interaction type.

`Prometheus` resources directory. In the latter case, `Prometheus` will use the cross sections provided in the resources directory directly.

In order to quantify the timing performance of `Prometheus`, we ran the full simulation chain on example ice-based and water-based detectors. For each detector, we simulated 10^3 events for each flavor and interaction type combination at 13 energies equally log-spaced between 10^2 GeV and 10^5 GeV. For the ice-based detectors, this test was run on a partition of an NVIDIA A100 GPU. This partition has 10 GB of CPU memory and 10 GB of GPU memory. The results of these tests for the ice-based detector can be seen in Fig. 4.6, while the results for the water-based detector can be found in Fig. 4.7. The water test was run on a 12th Gen Intel(R) Core(TM) i7-1255U, with 16 GB CPU memory.

Some interesting trends that shed light both on the code and the underlying physics can be observed in these plots. As one can see, photon propagation is the most time-consuming part of the simulation chain, taking up $\gtrsim$95% of the time. The next leading contribution is the charged lepton propagation. As one would expect, this only contributes to the runtime when charged-current interactions are simulated since charged leptons are not produced in neutral-current interactions. This is because we assume the neutrino emerging from a neutral-current interaction will not interact within the instrumented region, and thus, they do not require additional computational resources.

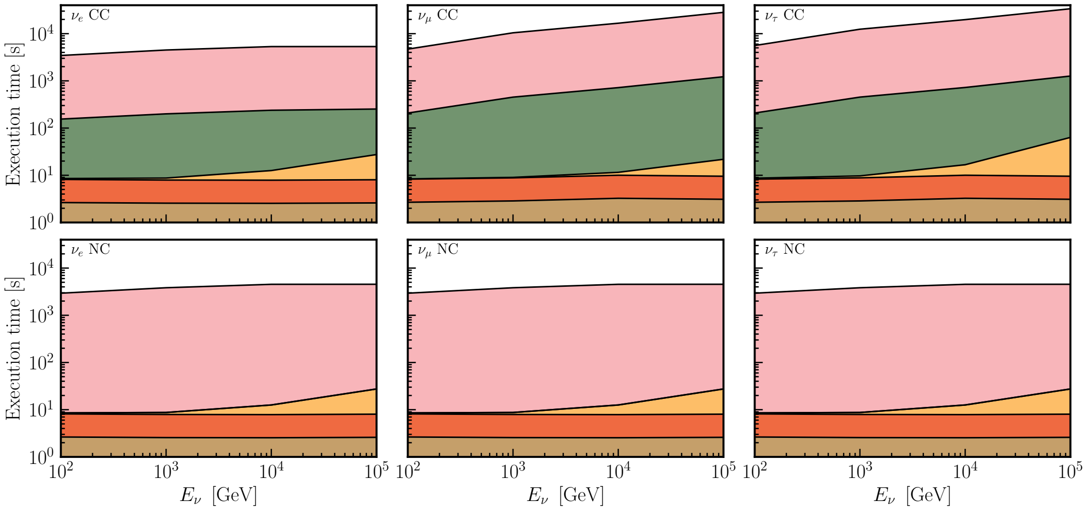


Figure 4.7: ***Wall time per 1000 events as a function of neutrino energy.*** The runtime for thousand events propagated in water using `fennel` and `Hyperion`. As expected, higher incident neutrino energies require longer run time. The colors, each corresponding to a different stage of production, are stacked. Each panel shows the runtime for different a different interaction type.

Careful observation reveals that the time required for this changes depending on whether ν_e, ν_μ, or ν_τ are simulated. This is because in the first case a propagator only needs to be made for e^-, while for the other cases, a propagator must be made for at least two charged leptons since the primary product can decay to a lighter charged lepton. Furthermore, it is worth noting that the initial particle injection and miscellaneous overhead do not scale with energy. This is sensible since injection primarily relies on random number generation, and there is a limited amount of computational difference across energies.

In the case of a water-based detector, we see a similar pattern where the photon propagator is the main driver of the runtime. Furthermore, we once again see the subdominant contribution of charged lepton propagation only in charged-current interactions. Once again, the overhead time is much larger for ν_μ and ν_τ due to the need to create multiple propagators.

In summary, the runtime is dominated by the photon propagation, implying the importance of researching methods for accelerating this process, such as those that have been proposed in [174].

As discussed above, `Prometheus` events are stored in the `Parquet` file format. Typically, the disk space required ranges between 1 kB and 10 kB per event for incident neutrino energies between 10^2 GeV and 10^5 GeV, see Fig. 4.8. This means that datasets that have millions of events, as is required for many machine-learning applications, can be stored in $\mathcal{O}(1\text{ GB})$ of memory. The exact value will depend on the energy range being simulated, as well as the initial energy spectrum.

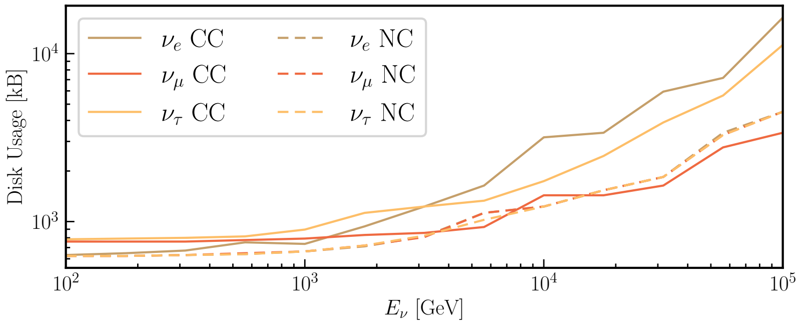


Figure 4.8: ***Disk space per thousand events as a function of energy.*** The disk space required to store `Prometheus` output as a function of the incident neutrino energy. Note that the relative disk requirements of each interaction type follow relative fraction of the initial neutrino energy that is deposited in the detector. Solid lines correspond to charged-current interactions while dashed lines correspond to neutral current interactions. Different colors indicate different neutrino flavors.

In general, the functional dependence on energy and relative ordering of different interactions in Fig. 4.8 align with expectations. For example, higher energy neutrinos lead to higher light yields, and thus a larger number of photons that must be stored. Furthermore, the relative ordering of the lines makes sense since the number of photons produced should be proportional to the energy deposited in the detector. This is the trend that is seen in Fig. 4.8 at energies above 3 TeV, where storing the photon arrival information drives disk space needs.

It should be noted that not every event produces enough light to trigger the detector. Our studies have found that the efficiency between injected events and those that trigger a detector is $\sim 60\%$ for ν_e charged-current interactions and $\sim 20\%$ for ν_μ charged-current interactions. This will depend on the trigger criteria, initial energy spectrum, interaction type, and generation specifications and we postpone detailed study of this for a future work.

Almost all the packages we use to model physical processes are published, and as such, have been well-verified. The only exceptions are the `fennel` and `hyperion` packages, which provide a new implementation of the parametrization that has already been shown to work in IceCube simulation [155, 158] and of ray-tracing-based photon propagation.

We must then show that all these packages are working in concert to produce physically meaningful simulation. Since the effective area is primarily governed by the physics implemented in `Prometheus`—the neutrino-nucleon cross section, lepton range, and photon propagation—and requires that all steps in the simulation are functioning together, it offers a good check of our code. In Fig. 4.9, we reproduce published effective areas of several water- and ice-based neutrino telescopes It is important to note that this effective area is always defined after some level of cuts, and while we have done

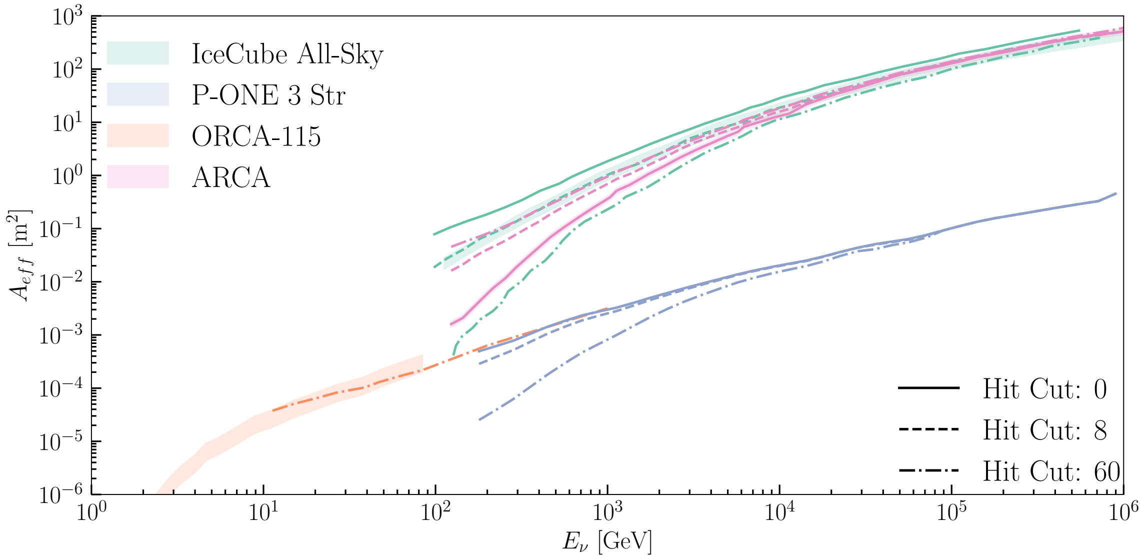


Figure 4.9: ***Effective area computed using `Prometheus` with comparisons to published results.*** We compare the ν_μ effective areas computed with `Prometheus` for IceCube, P-ONE3, ORCA, and ARCA for three different hit requirements, denoted by different line styles, to published effective areas. The IceCube effective area, taken from [175], is for $\nu_\mu + \nu_\tau$ events which pass the SMT-8 trigger and agrees with our calculation to within uncertainties. The ARCA [129] and ORCA [176] cases effective areas are constructed with more complicated hit requirements. Still, the scale and shape of the ORCA and ARCA effective areas and the `Prometheus` effective areas agree within uncertainties despite the simplified selection criterion. As of the publication of this paper, there is no published effective area for P-ONE3.

our best to reproduce the cuts from the references, there is not always sufficient information to do so perfectly. Furthermore, the OM response cannot be incorporated without access to proprietary information, and thus we expect differences of $\mathcal{O}(10\%)$. Users can extend, if needed, our simulation to incorporate these effects for a more detailed comparison.

4.2 Sparse Submanifold Convolutional Neural Networks for Trigger-Level Reconstruction

Gigaton-scale neutrino telescopes have opened a new window to the Universe, allowing us to study the highest energy neutrinos. While there are a variety of proposed designs, many follow the detection principle outlined by the DUMAND project [132] and consist of an array of optical modules (OMs) deployed in liquid or solid water. This detector paradigm shows great promise, and analyses by these experiments have already provided the first evidence of astrophysical neutrino sources [177, 178]. Before they can be analyzed, however, high-energy neutrinos must be isolated from the immense

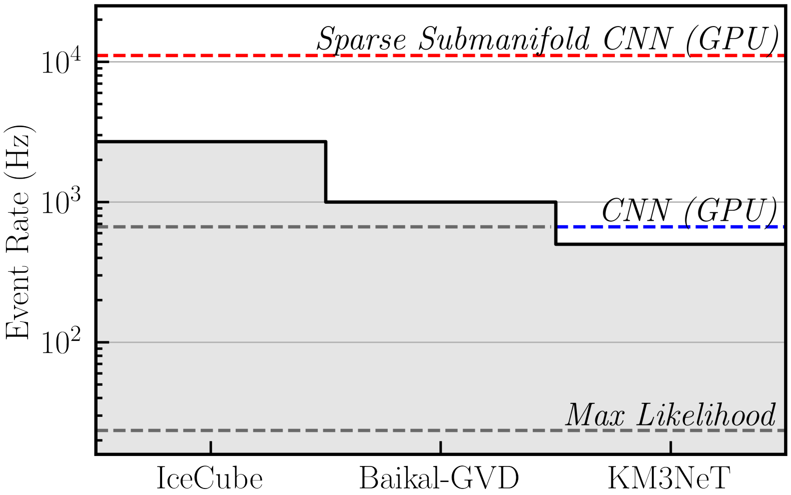


Figure 4.10: ***Event rates of triggers in different neutrino telescopes [179, 180, 181] compared to the run-times of various reconstruction methods.*** Sparse submanifold CNNs and their performance are detailed in this article. The CNN and maximum likelihood method run-times are taken from [182]. Notably, sparse submanifold CNNs can process events well above standard trigger rates in both ice- and water-based experiments.

cosmic-ray-muon-induced background. While a high-energy neutrino may trigger a detector once every few minutes, cosmic-ray muons typically induce a trigger rate on the order of kHz.

Since they are unable to traverse a substantial portion of the Earth without coming to rest, cosmic-ray muons have a distinct zenith dependence. This allows them to be removed by cutting on the reconstructed direction of an event. Thus, a reliable reconstruction that is capable of keeping up with the ~kHz background rate is the first step in isolating neutrinos. Moreover, a rapid reconstruction method could serve as part of an alert system that notifies researchers of events that are highly likely to be astrophysical neutrinos. For example, the real-time follow up of such an IceCube event led to the observation of the first astrophysical neutrino source candidate, TXS 0506+056, by detecting a neutrino in coincidence with a gamma-ray flare [177]. Along this line, similar efforts are underway in water-based detectors such as ANTARES, see [183] for a recent review.

At the trigger-level, a simple but fast reconstruction is typically done by solving a least squares problem via matrix inversion, as is the case for `LineFit` [184] in IceCube or `QFit` in ANTARES [185]. Machine learning has shown promise by delivering a comparable quality reconstruction with less runtime requirements [182, 186]; however, the fastest convolutional neural network (CNN) developed for high-energy neutrinos is not able to keep pace with a kHz-scale trigger-level rate. In this article, we introduce a reconstruction method using a sparse submanifold CNN (SSCNN), which overcomes this runtime issue. We will illustrate our method by focusing on solid-water detectors, but our results and conclusions readily generalize to water-based detectors. In this context, our SSCNN achieves better angular resolutions than methods such as `Linefit` while requiring a comparable run-time, enabling improved trigger-level cuts and serving as a better seed for the likelihood-based reconstruction. Fig. 4.10 summarizes typical event rates found in neutrino telescopes and compares these to the execution rate of various reconstructions. At the same time, SSCNN is also able to reconstruct the neutrino energy, a task which has not been done at trigger-level.

Convolutional neural networks (CNNs) have become the staple architecture for image-like data, and have achieved great success in a wide range of applications, including neutrino physics [187, 188, 189, 190]. However, data from neutrino telescopes presents inherent challenges to CNNs. In particular:

- *Non-regular geometry*: CNNs are designed to operate on images, which are arranged on Cartesian grids. Neutrino telescope sensors are typically spaced irregularly [191, 192, 180, 87], with varying distances and arrangements in between each sensor.

- *Sparsity*: Traditional CNNs use convolutions which operate on all points in the given input data. This leads to computational inefficiencies when the data is sparse.

- *High dimensionality*: Events occur in large spatial and temporal scales. This makes using traditional CNNs computationally unfeasible on raw 4D data (three spatial and one time) without information loss or significant pre-processing.

In this article, we propose a solution to these challenges using sparse submanifold convolutions [193]. This strategy has already shown success in liquid argon time projection chamber neutrino experiments [194, 195]. The usage of sparse submanifold convolutions in our network naturally solves the challenges laid out above. Sparsity and high dimensionality are no longer a concern, as the number of computations performed will depend only on the number of OM hits. With this improved computational efficiency, we can also handle non-regular geometries more smoothly by using the spatial coordinates of each OM hit (in meters from the center of the detector). This allows us to consider data of any shape or arrangement, without restricting ourselves to a Cartesian grid; thus our alogrithm can be easily adapted from our IceCube-like test case to, *e.g.* IceCube, KM3NeT, P-ONE, *etc.*

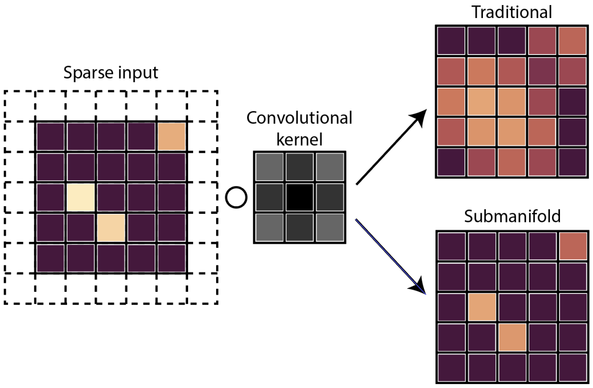


Figure 4.11: ***Comparison of conventional and submanifold convolution with a Gaussian kernel.*** The submanifold convolution maintains the sparsity of the input, while the traditional convolution blurs the input, making it less sparse. In this example, a traditional convolution would require 18 or 25 matrix multiplications for sparse and non-sparse convolution respectively, whereas the bottom image only requires three matrix multiplications.

Our SSCNN replaces traditional convolutions with sparse submanifold convolutions. While a traditional convolution extracts features by mapping a learned kernel over all input data, a sparse submanifold convolution operates only on the non-zero elements. This circumvents the inefficiency of using CNNs on sparse data, wherein the vast majority of operations are wasted multiplying zeros together. Furthermore, to preserve the sparsity of the data after applying multiple layers in succession, sparse submanifold convolutions enforces that the coordinates and number of output activations matches those of the input. In other words, the features do not spread layer after layer, as shown in Fig. 4.11. This is crucial for the efficiency of SSCNNs that are very deep, as the data would otherwise become progressively less sparse throughout the network. The lack of feature spreading will have a minimal impact on performance as long as the network can rely on local information. It should be noted that SSCNNs still compute over a grid-like structure, but this structure can be arbitrarily large because the network only operates on a submanifold of it.

As input, the SSCNN takes in two tensors: a coordinate tensor C, and a feature tensor F. In symbols:

$$C = \begin{bmatrix} x_1 & y_1 & z_1 & t_1 \\ \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & z_n & t_n \end{bmatrix}, F = \begin{bmatrix} h_1 \\ \vdots \\ h_n \end{bmatrix}, \tag{4.1}$$

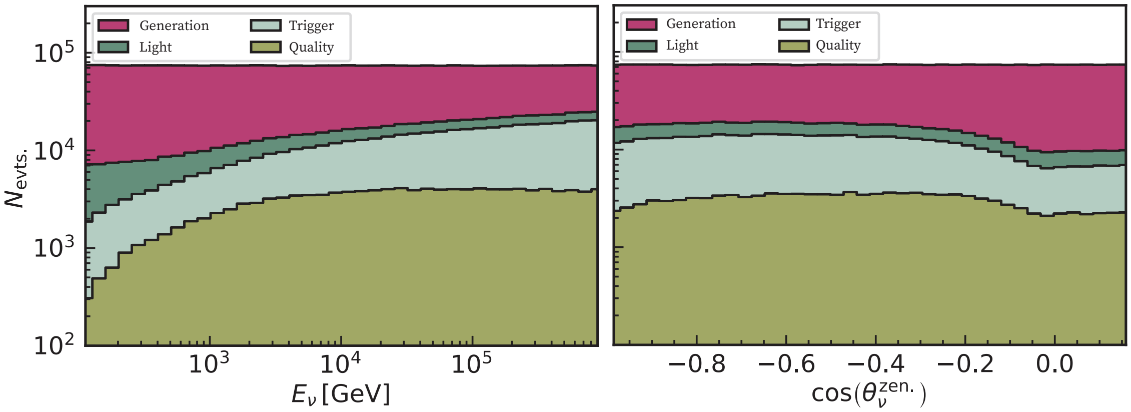


Figure 4.12: ***Distributions of events in true energy and zenith.*** *Left:* The distribution of events used as a function of true neutrino energy. As expected, the generated distribution is flat when binned logarithmically since the generation was sampled according to a E_ν^{-1} distribution. Furthermore, the fraction of generated events which produce light, and the fraction of light-producing events which pass the trigger threshold increase with energy. *Right:* The same distribution as a function of true neutrino zenith angle. Once again the generated distribution is flat in the cosine of this angle, which is proportional to the differential solid angle. nearly flat, with slightly lower efficiency near the horizon.

where the coordinate tensor is a $n \times 4$ tensor representing the space-time coordinates of the OMs in which there were a nonzero number of photon hits. The symbol n represents the total number of photon hits in the event, which is variable and can typically range from one to hundreds of thousands. The feature tensor contains the number of photon hits which occurred within a 1 ns time window on that OM, starting from the time indicated in the coordinate tensor. Nanosecond units were chosen for very fine timing resolution, as we aim to deploy the network on both low- and high-energy events. However, depending on the application, the 1 ns time window can be expanded to trade off timing resolution for even better run-time and memory efficiency.

It is worth noting that each coordinate can be associated with any feature vector, offering flexibility for various configurations to encode neutrino telescope data. For instance, we can approach this as a 3-dimensional problem, where we only consider the spatial positions as coordinates and use the timing information as features. However, we decided to treat the timing as coordinates to take advantage of the time structure inherent in neutrino telescope data.

Our benchmark case follows an ice-embedded IceCube-like geometry, where the OMs are spaced our approximately 125 meters horizontally and 17 meters vertically. The events used in this work are μ^- from ν_μ charged-current interactions. The initial neutrino sampling, charged lepton propagation, and photon propagation were simulated using the `Prometheus` package [196]. The incident neutrinos have energies between 10^2 GeV and 10^6 GeV sampled from a power-law with a spectral index of -1. Since most of the events that trigger neutrino telescopes are downward-going cosmic-ray muons, we generated a down-going dataset. Specifically, the initial momenta have zenith angles between 80° and 180°. It is worth noting that this definition of zenith angle is different from the convention which is typically used by neutrino telescopes, which take 0° to be downgoing. Internally, `LeptonInjector` [124] samples the energy, direction, and interaction vertex. `PROPOSAL` [123] then propagates the outgoing muon, recording all energy losses that happen within 1 km of the instrumented volume. `PPC` [197] then generates the photon yield from the hadronic shower and each muon energy loss, and propagates these photons until they either reach an OM or are absorbed. If a photon reaches an OM, the module ID, module position and time of arrival are recorded.

We then add noise in the style of [198] to the resulting photon distributions. This model accounts for nuclear decays in the OMs' glass pressure housings and thermal emission of an electron from a PMT's photocathode. The latter process strongly depends on the ambient temperature near the OM and varies between PMTs. Since this information is not publically available, we simplify the model and vary the thermal noise rate linearly from 40 Hz at the top of the detector to 20 Hz near the bottom, which approximately agrees with the findings from [198]. We then take the nuclear decay rate to be 250 Hz and generate a number of photons drawn from a Poisson distribution with a mean of 8 for each decay.

Before moving on, it is important to note that the photons generated in the previous steps are only tracked to the surface of the OM. In a full simulation of the detection process, one would need to simulate the electronics inside the OM, which could introduce timing uncertainties. Furthermore, the digitized signal reported by the *e.g.* the IceCube OMs must be unfolded to get the number of photons per unit time. These steps require access to proprietary information that is not available externally. Thus, we cannot include the effects from these detail detector performance in our simulation.

For example, the process by which IceCube unfolds the photon arrival times is described in [199]. They find that this process introduces a timing uncertainty typically on the order of 1 ns but that may grow up 10 ns under certain conditions. While this may affect our results, we expect the impact to be small since by group the photon arrival times into ns-wide bins, we are introducing a timing uncertainty with a similar scale.

Once all photons have been added, we then implement a trigger criteria similar to the one described in [6]. This requires that a pair of neighboring or next-to-neighboring OMs see light in a 1 μs time window. If 8 such pairs are achieved in a 5 μs time window, we consider the trigger to be satisfied. As before, the exact details of the triggering process require access to proprietary information; however, the events which pass our trigger should be qualitatively similar to those which would trigger IceCube. After this cut, we are left with 462,892 events from 3 million simulated events, which we split between the training and test data sets of 412892 and 50000 sizes respectively. One can see distributions of the events which pass this trigger as a function of true energy, zenith, and azimuth in Fig. 4.12.

In addition to the trigger-level dataset, we also evaluate the network performance on a dataset with further quality cuts, so that we can understand performance on events which are more likely to make it into a final analysis. In order to do this, we consider three quantities: N_{OM}, r_{COG}, and R_{ell}. The first two quantities—the number of distinct OMs that saw light and the distance between the charge-weighted center of gravity and the center of the detector—are fairly straightforward, but the last requires more explanation. To compute R_{ell}, we fit a two-parameter ellipsoid to all OMs which saw light, and then take the ratio of the long axis to the short axis. A ratio close to one indicates a spherical events, whereas larger ratios indicate longer, track-like events.

We perform straight cuts on these variables, requiring events have $N_{\mathrm{OM}} > 11$, $r_{\mathrm{COG}} < 400$ m, and $2 < R_{\mathrm{ell}} < 8$. The first cut removes low-charge events which are difficult to reconstruct, while the second removes "corner clipper" events caused by μ^- passing near the edge of the detector. The final cut on R_{ell} helps ensure that the events have a long lever-arm for reconstruction.

These cuts reduce the split training and testing dataset sizes to 108585 and 13183 events, respectively. The spatial sparsity of these improved quality events is about $\sim 3\%$, as there are 154 OMs that were hit on average, out of the 5,160 total OMs in our example detector. For the trigger level events, the spatial sparsity is about $\sim 2\%$. The time dimension adds another level of sparsity, as typical events can last tens of thousands of nanoseconds compared to the microsecond time window.

We utilize a ResNet-based architecture, taking advantage of residual connections between layers to promote robust learning for deeper networks. More details on the network architecture can be found in Fig. 4.13. A typical block of the network consists of a sparse submanifold convolution, followed by batch normalization and the parametric rectified linear unit (PReLU) activation function. The selection of the activation function was determined after examining prevalent alternatives, such as the conventional ReLU or a smooth approximation, like the SELU. Downsampling is performed

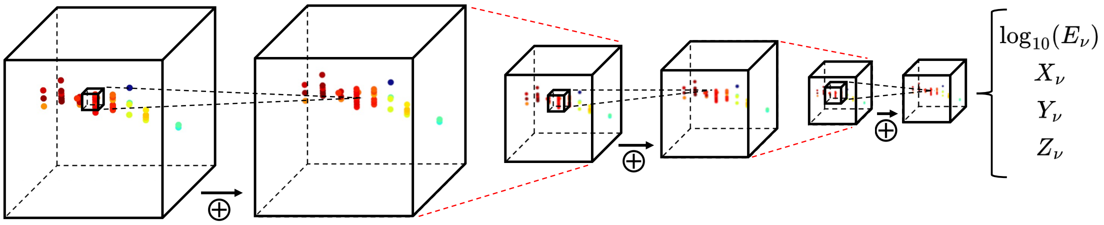


Figure 4.13: ***Network architecture overview.*** The network accepts as input a 4D point cloud of photon OM hits, as shown by the colored points in the figure. The color indicates the timing (red is earlier, blue is later). Residual connections, denoted with $\oplus$, are used in between convolutions. Downsampling (dashed red lines) is performed after a series of convolutions. The final layer of the network is a fully connected layer, which outputs the logarithmic ν_μ energy, and the three components of the normalized ν_μ direction vector.

using a stride 2 sparse submanifold convolution. We use the `PyTorch` deep learning framework and the `MinkowskiEngine` [200] library to implement the network. The network was trained on each dataset (trigger and quality) for 25 epochs using a batch size of 128 and the `AdamW` optimizer. The initial learning rate was set at 0.001 and was dropped periodically during training.

For the purpose of this article, we train the network to infer the primary neutrino energy E_ν, and the three components of its directional pointing vector, (X_ν, Y_ν, Z_ν). The directional vector is learned rather than the zenith and azimuth angles because of complications with azimuthal periodicity and undesirable boundary condition behavior at large or small angles. The network is trained to predict the logarithmic energy, $\log_{10}(E_\nu)$, and the normalized directional vectors, as they can vary over a wide range of magnitudes.

To train the energy reconstruction task, the `LogCosh` loss function is used, since it is more robust to outliers than the standard `MSE` loss. The loss function is defined as follows,

$$\mathcal{L}_E = \frac{1}{N}\sum_i^N \log\left(\cosh\left(x_i - y_i\right)\right), \tag{4.2}$$

where N is the number of events in the batch, x_i are the predictions, and y_i are the labels. For the angular reconstruction, an angular distance loss function is used, namely,

$$\mathcal{L}_A = \frac{1}{N}\sum_i^N \arccos\left(\frac{\vec{X}_i \cdot \vec{Y}_i}{||X_i||\,||Y_i||}\right), \tag{4.3}$$

where $\vec{X}_i$ and $\vec{Y}_i$ are the predicted and true directional vectors, respectively. Then, the total loss is given by

$$\mathcal{L}_{tot} = (1-\alpha)\mathcal{L}_E + \alpha\mathcal{L}_A, \tag{4.4}$$

where a weighting factor α is applied to each of the separate loss terms. An ensemble of networks was trained from scratch while α was varied. Performance on both energy and angular reconstruction was tested for each of these networks to determine the optimal value of α. We found that setting $\alpha = 0.7$ results in superior angular reconstruction performance, without significantly affecting the energy reconstruction.

SSCNN (GPU)	**0.090 ± 0.007 ms**
SSCNN (CPU)	**65.22 ± 117.04 ms**
Max Likelihood (CPU)	42.6 ± 175 ms

Table 4.1: ***Per-event average run-time performance.*** The forward pass run-times (mean ± STD) for SSCNN was evaluated on trigger-level events. A likelihood-based method for energy and angular reconstruction was included for reference [182, 201].

We evaluate the run-time performance of the SSCNN in terms of the forward pass duration on both CPU and GPU hardware. The CPU benchmark is performed on a single core of an Intel Xeon Platinum 8358 CPU, while the GPU benchmark uses a 40 GB NVIDIA A100. As is generally the case for neural networks, running on GPU is preferred due to its superior parallel computation capabilities. Additionally, the use of sparse submanifold convolutions has greatly enhanced our GPU memory efficiency, enabling us to run larger batch sizes during inference. The SSCNN can process events at a rate of 11,098 Hz on a 40 GB NVIDIA A100 GPU, while handling a batch size of 12,288 events simultaneously. This is fast enough to handle the expected ∼kHz current and planned large neutrino telescopes.

The run-time on a single-core CPU is slower and largely dependent on the number of photons hits in the event due to the limited parallel computation capabilities. However, the SSCNN run-time on a CPU core is comparable to that of the likelihood-based method and is more consistent, as indicated by the lower standard deviation on the run-time distribution. The run-time results on both GPU and CPU are summarized in Table 4.1.

We first test the network on reconstructing the direction of the primary neutrino. We measure performance using the angular resolution metric, which is calculated by taking the angular difference between the predicted and true directional vectors. Fig. 4.14 shows the angular resolutions as a function of the true neutrino energy. Lower-energy events generally produce less photon hits, leading to a shorter lever-arm and, consequently, worse resolution. As expected, the trigger-level events are generally harder to reconstruct due to the lower light yield and the presence of corner-clipper events.

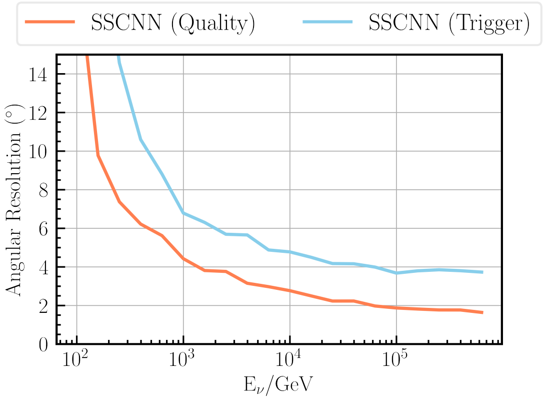


Figure 4.14: ***Angular reconstruction performance as a function of the true neutrino energy.*** The angular resolution results are binned by the true neutrino energy, with the median taken from each bin to form the lines shown.

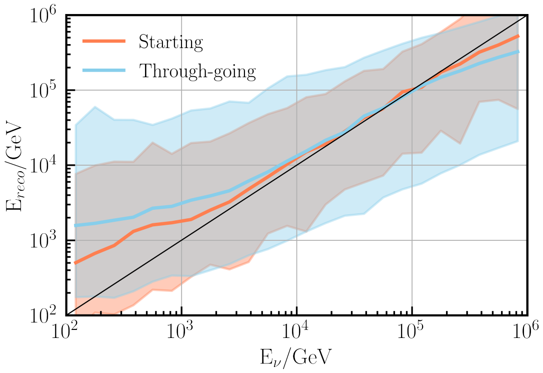


Figure 4.15: ***Energy reconstruction performance at the trigger-level.*** The solid lines show the median of the predicted $\log_{10}(E_\nu)$, while the shaded regions are the 5% to 95% confidence level bands. The events in the test dataset are separated into starting and through-going events. The solid black line serves as a reference for a perfect reconstruction.

This is especially true for angular reconstruction, where the SSCNN is able to reach under 4° median angular resolution on the highest-energy events. Enforcing the previously described quality cuts improves the results of the SSCNN by roughly 2° across the entire energy range. This performance is comparable to or better than current trigger-level reconstruction methods used in neutrino telescopes. For example, the current trigger-level direction reconstruction at IceCube is done using the traditional `Linefit` algorithm [184], which has a median angular resolution of approximately 10° on raw data.

We also test the networks by reconstructing the energy of the primary neutrino. Fig. 4.15 summarizes the energy reconstruction results. Events where the interaction point of the neutrino occurs outside the detector, known as through-going events, make up the majority of our dataset. As a result, predicting the neutrino energy has an inherent, irreducible uncertainty produced by the unknown interaction vertex and the muon losses outside of the detector. This missing-information problem leads to an intrinsic uncertainty in the logarithmic neutrino energy of approximately 0.4 for a through-going event. Additionally, the network performs noticeably worse at the lowest and highest energies, with a tendency to overpredict at low energies and underpredict at high energies. This behavior can be attributed to the artificial energy bounds on the simulated training dataset.

4.3 Quantum Embedding of Neutrino Telescope Data

Historically, our understanding of Nature has moved concurrently with our ability to centrally store and quickly access information. More recently, this relationship has grown more intimate. The invention of magnetic storage on tapes, disks, and finally solid state drives enabled storage and analysis on the immense amount of data generated by the increasingly complex detectors demanded by particle physics. This *Big Data* paradigm has allowed us to establish the incredibly successful Standard Model (SM) of particle physics, culminating in the 2012 discovery of the Higgs Boson by the Large Hadron Collider (LHC).

While the successes of our current paradigm are undeniable, major questions remain unanswered on the largest and smallest scales. For example, on galactic and cosmological scales, the SM cannot provide a viable dark matter candidate. Furthermore, at the sub-nuclear scale, the origin and smallness of neutrino masses continues to elude us. In attempts to answer this question, high-energy physics (HEP) experiments must reduce the data to a manageable level by relying on triggers templated on familiar physics; however, this approach leaves us vulnerable to the so-called streetlight effect, in which one only searches where it is easiest. As we continue to pursue new physics, we may need to relax these filters in order to allow in previously unconsidered types of events.

This need to expand our searches to new regions is happening against the backdrop of a crisis of data management and acquisition in HEP. For example, every day, the IceCube Neutrino Observatory produces one terabyte (TB) of data, most of which must be transferred via satellite, while the LHC produces nearly 300 TB in the same period. Additionally, both experiments have planned upgrades that may increase their data production by an order of magnitude [78, 202]. Beyond these two experiments, many other large-volume next-generation experiments such as DUNE in the US and Hyper-Kamiokande in Japan are expected to face similar challenges. While it may be possible to meet these data needs using current technologies, selecting and storing large fractions of the data will become increasingly untenable.

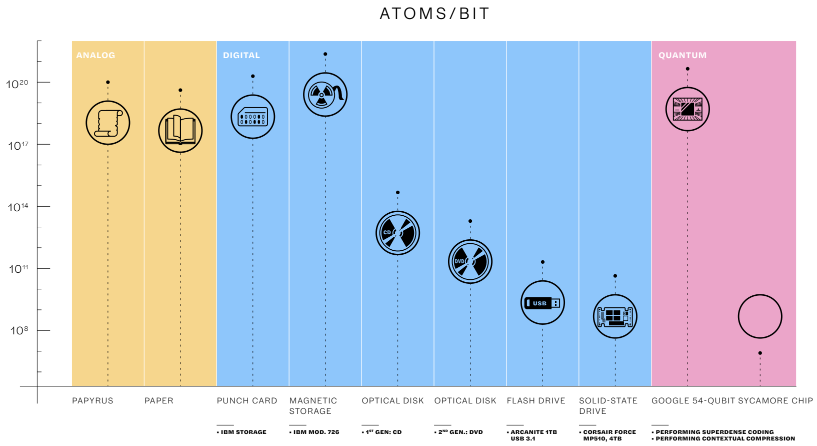


Figure 4.16: ***Data compression through history.*** The vertical axis shows the number of atoms needed to encode a single bit of information, while the horizontal axis shows the technology or medium used to store it. We calculated the number of atoms per bit in analog formats (shown in yellow) by counting the average number of characters per document in Greek and English, respectively. For digital medium (shown in blue), we estimated the number of atoms from typical ISO fabrication standards. For quantum devices (shown in pink), the number of bits accessible from 54-qubit quantum states in Google's Sycamore chip is first calculated for a straightforward *basis encoding* (left) and afterwards calculated in a *random access code* (right), extrapolating the method used in this work into that scale.

Quantum technologies use additional degrees of freedom provided by quantum physics to move beyond classical limitations in a broad range of tasks. For instance, quantum metrology improves the accuracy of sensors, pushing towards the Heisenberg limit, while quantum communication has established provable secure communication protocols to protect messages against possible eavesdroppers. Additionally, entanglement and superposition have been shown to speed up information processing tasks in quantum processors. Finally, one appealing application—which we investigate here—is the capacity of quantum systems to compress information.

Before describing the technical details of our protocol, we will attempt to describe the procedure in broad strokes. Our goal is to efficiently encode a classical bitstring, a sequence of 0s and 1s, in a system of qubits. Bitstrings are a natural choice for quantum storage since qubits are two-level systems, meaning that any measurement of a qubit's spin will yield only $\lambda_{\pm} = \pm 1$. These outcomes may be straightforwardly mapped to a bit value via $\text{bit}(\lambda) = (1 - \lambda)/2$; however, encoding one classical bit per qubit is very inefficient. We may extend this thinking and define the *parity* of a many-qubit system as the product of spin measurements on each qubit. Since the measurement of each qubit will yield ±1, the product over all measurements will also yield ±1. Furthermore, since we may measure the spin along either the x-, y-, or z-direction, we can construct 3^n *parity operators* (POs) for an n-qubit system.

With these POs, we can construct a classical bitstring of length 3^n by measuring our parity observables on an ensemble of n-qubit systems. Formally, we choose a mapping, $\mathcal{O}$, between the non-negative integers less than 3^n, $\{x \in \mathbb{Z} | 0 \leq x < 3^n\}$, and the set of all POs, $\{X,\, Y\, Z\}^{\otimes\, n}$. This mapping does not play a crucial role in our algorithm and may be chosen arbitrarily [1]. Thus, we have a mapping between a set of n-qubit systems and a so-called *parity bitstring*, b of length 3^n whose values are indexed by i via:

$$b_i = \text{round}\left[\text{bit}\left(\frac{1}{|\{|\psi\rangle\}|}\sum_j \langle\psi_j|\, \mathcal{O}(i)\, |\psi_j\rangle\right)\right], \tag{4.5}$$

where $\{|\psi\rangle\}$ is the set of n-qubit states, and $|\psi_j\rangle$ is a specific state within that set. Simply put, in order to compute the i^{th} bit, we take a classical average of the expectation values for $\mathcal{O}(i)$ over all states, transform it into a value between 0 and 1 via the bit$(\cdot)$ function, and then round this value. Thus, in order to store a given bitstring in a set of n-qubit systems we need only find a set of quantum states which faithfully reproduce this bitstring under the prescription given in Eq. (4.5).

The procedure of directly encoding classical bits to rounded PO measurements in a one-to-one mapping presents a challenge. Since quantum mechanics restricts the relationships between eigenvalues for certain spin measurements, it may not be possible to represent an arbitrary set of bits with a compact set of quantum states in this manner. The encoding procedure laid out in Ref. [203], circumvents this challenge by encoding the bits into pairs of PO measurements, rather than mapping each bit to a particular PO measurement. Under this procedure, if the rounded value of two quantum measurements are the same, it assigns the corresponding classical bit a value of 0; if they are unequal, it assigns a value of 1, *i.e.* the XOR gate is applied to these measurement outcomes. Formally, the i^{th} bit, b'_i, in this representation is obtained by two parity bits of the prior one-to-one-representation $b'_i = b_{2i} \oplus b_{2i+1}$, where $\oplus$ denotes the `XOR` operator and i ranges between 0 and $(3^n - 1)/2 - 1$.

While this may seem counter-intuitive, since this choice halves the number of classical bits one can encode in n-qubits, this achieves immense freedom since now a classical bitstring of length N can be equivalently represented by 2^N equivalent parity bitstrings, some of which may be more convenient than others.

[1] In the rest of the text, we assume the so-called alphabetical ordering, in which the operators are placed in order based on the position of the characters in the conventional ordering of the Latin alphabet.

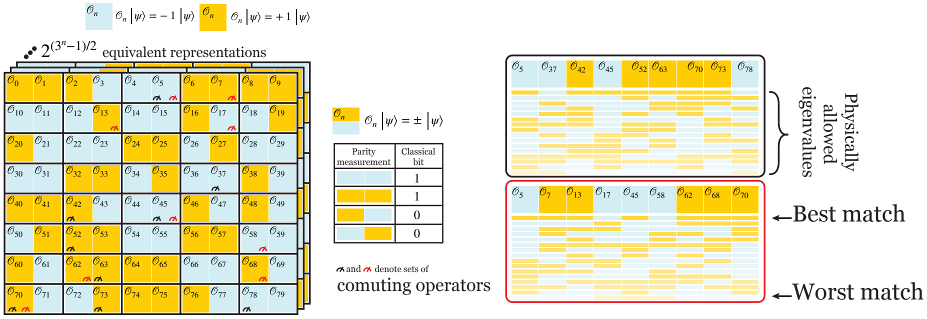

$\mathcal{O}_n$ $\mathcal{O}_n |\psi\rangle = -1 |\psi\rangle$ $\mathcal{O}_n$ $\mathcal{O}_n |\psi\rangle = +1 |\psi\rangle$
$2^{(3^n-1)/2}$ equivalent representations
$\mathcal{O}_0$ $\mathcal{O}_1$ $\mathcal{O}_2$ $\mathcal{O}_3$ $\mathcal{O}_4$ $\mathcal{O}_5$ $\mathcal{O}_6$ $\mathcal{O}_7$ $\mathcal{O}_8$ $\mathcal{O}_9$
$\mathcal{O}_{10}$ $\mathcal{O}_{11}$ $\mathcal{O}_{12}$ $\mathcal{O}_{13}$ $\mathcal{O}_{14}$ $\mathcal{O}_{15}$ $\mathcal{O}_{16}$ $\mathcal{O}_{17}$ $\mathcal{O}_{18}$ $\mathcal{O}_{19}$
$\mathcal{O}_{20}$ $\mathcal{O}_{21}$ $\mathcal{O}_{22}$ $\mathcal{O}_{23}$ $\mathcal{O}_{24}$ $\mathcal{O}_{25}$ $\mathcal{O}_{26}$ $\mathcal{O}_{27}$ $\mathcal{O}_{28}$ $\mathcal{O}_{29}$
$\mathcal{O}_{30}$ $\mathcal{O}_{31}$ $\mathcal{O}_{32}$ $\mathcal{O}_{33}$ $\mathcal{O}_{34}$ $\mathcal{O}_{35}$ $\mathcal{O}_{36}$ $\mathcal{O}_{37}$ $\mathcal{O}_{38}$ $\mathcal{O}_{39}$
$\mathcal{O}_{40}$ $\mathcal{O}_{41}$ $\mathcal{O}_{42}$ $\mathcal{O}_{43}$ $\mathcal{O}_{44}$ $\mathcal{O}_{45}$ $\mathcal{O}_{46}$ $\mathcal{O}_{47}$ $\mathcal{O}_{48}$ $\mathcal{O}_{49}$
$\mathcal{O}_{50}$ $\mathcal{O}_{51}$ $\mathcal{O}_{52}$ $\mathcal{O}_{53}$ $\mathcal{O}_{54}$ $\mathcal{O}_{55}$ $\mathcal{O}_{56}$ $\mathcal{O}_{57}$ $\mathcal{O}_{58}$ $\mathcal{O}_{59}$
$\mathcal{O}_{60}$ $\mathcal{O}_{61}$ $\mathcal{O}_{62}$ $\mathcal{O}_{63}$ $\mathcal{O}_{64}$ $\mathcal{O}_{65}$ $\mathcal{O}_{66}$ $\mathcal{O}_{67}$ $\mathcal{O}_{68}$ $\mathcal{O}_{69}$
$\mathcal{O}_{70}$ $\mathcal{O}_{71}$ $\mathcal{O}_{72}$ $\mathcal{O}_{73}$ $\mathcal{O}_{74}$ $\mathcal{O}_{75}$ $\mathcal{O}_{76}$ $\mathcal{O}_{77}$ $\mathcal{O}_{78}$ $\mathcal{O}_{79}$
$\mathcal{O}_n$ $\mathcal{O}_n |\psi\rangle = \pm |\psi\rangle$
Parity measurement
Classical bit
1
1
0
0
and denote sets of
comuting operators
$\mathcal{O}_5$ $\mathcal{O}_{37}$ $\mathcal{O}_{42}$ $\mathcal{O}_{45}$ $\mathcal{O}_{52}$ $\mathcal{O}_{63}$ $\mathcal{O}_{70}$ $\mathcal{O}_{73}$ $\mathcal{O}_{78}$
Physically allowed eigenvalues
$\mathcal{O}_5$ $\mathcal{O}_7$ $\mathcal{O}_{13}$ $\mathcal{O}_{17}$ $\mathcal{O}_{45}$ $\mathcal{O}_{58}$ $\mathcal{O}_{62}$ $\mathcal{O}_{68}$ $\mathcal{O}_{70}$
←Best match
←Worst match

Figure 4.17: ***Schematic of the representation optimization for a four-qubit system.*** In the left portion of the figure, each square represents a certain parity observable and the color of the square represents the value one wishes to obtain under the prescription given in Eq. 4.5. If two boxes are grouped by a black border, the XOR operation will be applied to those values to get the value that is to be encoded. Equivalent representations can be generated by flipping the color of both squares in any black box since the XOR operation is invariant under this transformation. The red and black measurement symbols denote sets of operators that commute and thus may have simultaneous eigenvalues. On the right, we look at all physically allowed configurations of the commuting states, ranked according to the degree of overlap they have with the proposed representations. The sum over all sets of commuting states of the hamming distance between the closest physically allowed state and the proposed representation forms the objective function for the representation optimization.

If we wish to compare the multiple equivalent representations, we must establish an objective function by which to rank them. In order to do this, it is helpful to introduce the concept of a *context*. This is a maximal set of commuting parity observables that with. Such a set represents the largest portion of the bitstring that can be stored in a single n-qubit system. We will refer to quantum states that have a definite eigenvalue as *context eigenstates* and the corresponding set of eigenvalues as *context eigenvalues*. We then iterate over all contexts, select the corresponding portion of the representation, and compare that portion to the appropriate context eigenvalues. We then find the allowed values that are closest to the representation and compute the hamming distance. The sum of all hamming distances then establishes our objective function, which we can use to compare different representations.

This process yields a representation that should be easy to represent with few quantum states but does not give insight into which states should be used. We must then perform another optimization to find a representation that balances using a limited number of quantum resources while faithfully expressing the desired representation. This comes down to selecting some subset of the states that are closest to the representation—having stored them in the representation optimization—computing the average expectation value for each parity observable, and comparing this to the desired value.

A harsh penalty is applied for parity observables that do not have the correct sign, and a milder penalty is applied for using more states.

In order to demonstrate the encoding protocol on high-energy physics data, we simulated ν_μ and ν_e CC events with the `Prometheus` package [196]. We then condensed each event into per-DOM summary data, using and approach that is similar to the one used in some IceCube machine-learning reconstructions. If a DOM saw light, we calculate the x-, y-, and z- position, the logarithm of the number of photons, and the average arrival time of the photons. These five summary variables are encoded into 32-bits and then concatenated to form a 160-bit bitstring. We then concatenate the bitstrings for each DOM that saw light into a bitstring that represents the full event. Currently, we are able to efficiently run our encoding procedure on eight-qubit systems, limiting us to events in which up to 20 DOMs saw light. 999 of our simulated events met this criterion, of which 485 originated from a ν_e CC interaction and 514 originated from ν_μ CC interaction.

After running our encoding protocol on all events, we found that the encoded bitstring could be represented with 680^{+18}_{-25} quantum states. The fidelity between the original bitstring and the encoded bitstring was found to be $84.09\%^{+0.78\%}_{-1.95\%}$. After encoding and reading out each event on the IBM Cairo quantum processor, we found that we recovered the encoded bitstring with a fidelity of $100.00\%^{+0.00\%}_{-1.04\%}$. While this method clearly has room for improvement, it is encouraging that it is limited by classical preprocessing and not by the state of quantum technology. Thus, if we can find an optimization procedure that can more faithfully express the target bitstring, this technique may be readily applied.

An immense amount of technical work was required to achieve these results. These can be roughly grouped into two categories: increasing the computational efficiency of the encoding steps and optimizing the quantum circuits to ensure minimal loss between encoding and readout.

In order to understand the computational optimizations, we must explore some of the underlying symmetries of the parity observables and the contexts in which we group them. To begin, we define two base contexts as:

$$\mathcal{C}_0^{\text{even}} = \{Z^{\otimes n}\} \cup \{X, Y\}^{\otimes n}_{\text{even number of } X\text{s}} \tag{4.6}$$

and

$$\mathcal{C}_0^{\text{odd}} = \{Z^{\otimes n}\} \cup \{X, Y\}^{\otimes n}_{\text{odd number of } X\text{s}} \; . \tag{4.7}$$

where X, Y, and Z are the dimensionless Pauli matrices. From these definitions, we may make two observations. The first is that the anti-commutation relations of the Pauli spin operators imply that

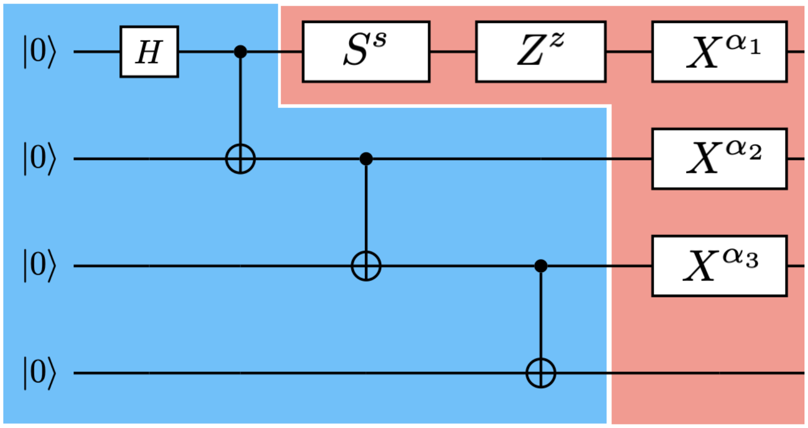


Figure 4.18: ***Circuit for generating context eigenstates*** The portion of the circuit with a blue background prepares a canonical GHZ state, while the portion on the red background prepares one of the 2^{n+1} context. Choosing $s = 0$ sets the state of the system to an even context, and choosing $s = 1$ sets it to an context

all elements of a context commute. The second is that $Z^{\otimes n}$ plays a special role as it is the only element that commuted between the two sets. We say that $Z^{\otimes n}$ *generates* these two base contexts.

Each of the contexts considered has a unique basis of 2^n eigenstates. One of the eigenstates of $\mathcal{C}_0^{\text{even}}$ is the n-qubit GHZ state $|\phi_0, \mathcal{C}_0^{\text{even}}\rangle = |\text{GHZ}\rangle \equiv \frac{1}{\sqrt{2}}(|00...0\rangle + |11...1\rangle)$, and one of the eigenstates of $\mathcal{C}_0^{\text{odd}}$ is the n-qubit i-phase GHZ state $|\phi_0, \mathcal{C}_0^{\text{odd}}\rangle = |\text{GHZ-}i\rangle \equiv \frac{1}{\sqrt{2}}(|00...0\rangle + i\,|11...1\rangle)$. The rest of the eigenstates can be attained from these base cases by applying an operator of the form:

$$|\phi_{z,\alpha}, \mathcal{C}_0^{\text{even}}\rangle = (X^{\alpha_1} \otimes X^{\alpha_2} \otimes ... \otimes X^{\alpha_{n-1}} \otimes \nVdash) \left(Z^z \otimes \nVdash^{\otimes(n-1)}\right) |\text{GHZ}\rangle$$

and

$$|\phi_{z,\alpha}, \mathcal{C}_0^{\text{odd}}\rangle = (X^{\alpha_1} \otimes X^{\alpha_2} \otimes ... \otimes X^{\alpha_{n-1}} \otimes \nVdash) \left(Z^z \otimes \nVdash^{\otimes(n-1)}\right) |\text{GHZ-}i\rangle$$

where $\nVdash$ is the identity operator, and both α_i and z are either 0 or 1. We may express this even more compactly in the form:

$$|\phi_{z,\alpha}, \mathcal{C}_0^{s}\rangle = (X^{\alpha_1} \otimes X^{\alpha_2} \otimes ... \otimes X^{\alpha_{n-1}} \otimes \nVdash) \left(Z^z \otimes \nVdash^{\otimes(n-1)}\right) \left(S^s \otimes \nVdash^{\otimes(n-1)}\right) |\text{GHZ}\rangle$$

where S is the operator $\left(\begin{smallmatrix} 1 & 0 \\ 0 & i \end{smallmatrix}\right)$ and s is 0 or 1. Under this construction, the even contexts are given by choosing $s = 0$ and odd by $s = 1$. A circuit that may be used to construct these context eigenstates is shown in Fig 4.18.

We then prepare each eigenstate and measure the eigenvalue of all operators for each eigenstate. Assuming that we have a canonical ordering for the operators and the eigenstates, we can construct a matrix—which we call the *fingerprint*—that tells us the eigenvalue for any operator in any eigenstate.

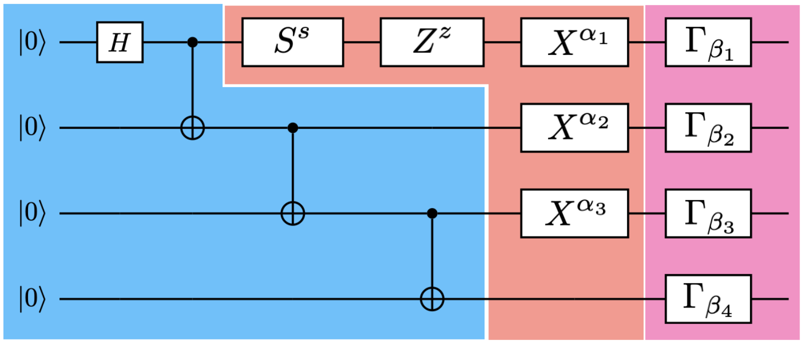


Figure 4.19: ***Context eigenstate preparation.*** Quantum circuit to prepare the 4-qubit context eigenstate $|\phi_{z,\alpha}, \mathcal{C}^s_\beta\rangle$. The blue area of the circuit trivially prepares a 4-qubit GHZ state $|GHZ\rangle = \frac{1}{\sqrt{2}}(|0000\rangle + |1111\rangle)$, the orange area performs local rotations based on binary parameters s, z, α_1, α_2 and α_3, and the pink area performs local rotations based on ternary parameters β_1, β_2, β_3 and β_4. Note that this didactic circuit prepares the GHZ state with depth $O(n)$ ($n-1$ layers of two-qubit-gates), but GHZ states can be prepared with lower depth if the qubits have more connectivity than what is used here, as will be seen in the next section.

Rephrasing this quantum mechanical question of parity operators' eigenvalues in terms of matrices is incredibly important for computational efficiency since computers are incredibly efficient at matrix operations.

We can now use our base contexts to construct all other contexts. To do this, we must introduce the Γ rotation operators, given by:

$$\Gamma_0 = \not{\mathbb{1}} \ , \ \Gamma_1 = H\,Z\,S \ , \ \Gamma_2 = S\,H \ ,$$

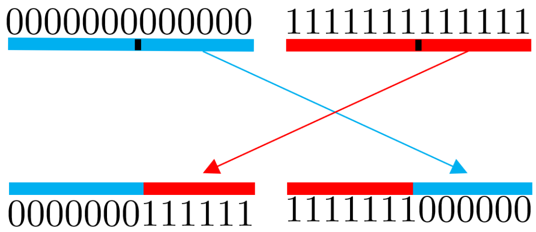
which "rotate" one Pauli operator to another . In this expression H is the Hadamard gate $H = \frac{1}{\sqrt{2}}\left(\begin{smallmatrix} 1 & 1 \\ 1 & -1 \end{smallmatrix}\right)$. Up to a global phase, Γ_1 one applies a positive Levi-Civita permutation to the X, Y, and Z operators while Γ_2 applies a negative Levi-Civita permutation to them. Applying a Γ rotation consistently across all parity observables in a context yields a new context. Since we are able to apply a one of three Γ rotations to any of the n qubits, we find that we have 3×3^n contexts. These may be constructed by modifying the previous circuit to the circuit shown in Fig. 4.19 Importantly, the eigenvalues of the new context are invariant under Γ rotations. Thus, as long as we are careful about consistently ordering the operators in a context, we may use the fingerprint that we calculated for the base contexts for every other context.

These observations allow us to compute the previously mentioned objective function by comparing the appropriate values of a representation to the relevant rows of our fingerprint, but we are still left with the problem of performing the optimization. Recalling that we can represent a bitstring of length $(3^n - 1)/2$ and that there are two ways to represent each bit, we find that there are $\sqrt{2}^{3^n-1}$ equivalent representations.

We implemented an optimized a genetic algorithm (GA) to carry out the optimization required to find the states that most closely match the selected representation of the input logical bit string. The motivation behind the GA, which gives rise to its name, is in biology itself, or specifically, evolution by natural selection, which is based on three pillars, namely, (1) *inheritance*: traits can be inherited in one generation from the previous one; (2) *variation*: random variations occur in the genetic code when creating new individuals, and (3) *competition*: individuals compete with each other and the one who is most fit is able to reproduce and pass on their traits to the next generation. Note the importance of the genetic code here: it is the information that can be inherited, can vary, and determines the fitness of the individual. This is precisely the reason why working with a GA is especially suitable in our case, since we can think of the individuals in the "population" as the bit strings, and by introducing variations and competition among the bit strings, we can select for the ones that are "most fit." More precisely, we define a cost function f that takes as input a bit string and returns a nonnegative real number as output that represents the distance between the input and target string. The fitness of an individual, *i.e.* a bit string, is determined by this function.

The first step in the GA is to create the population of bit strings. This is done by picking bit strings at random of the length desired. Their fitnesses are subsequently evaluated, and they are ranked according to their fitness. An iterative process then begins, whereby a new "generation" of bit strings is created every time, based on the fitness of every bit string. Every new generation is created in two basic steps, which are repeated until we reach the total population number, which remains constant throughout the optimization. These two steps are *crossover* and *mutation.*

In the crossover step two bit strings are selected at random from the current population, where, in selecting them, bit strings are weighted by their fitness. This weighting ensures that better bit strings are more likely to pass to the next generation. Once these two bit strings are selected, with probability p_{cross}, a crossover happens. This crossover probability p_{cross} is a meta-parameter of the algorithm, which we set to 0.95,. The crossover is based precisely on the same process that occurs in chromosomes, whereby one segment of one chromosome (respectively, bit string) is "taken" from this chromosome (bit string) and placed in the same location on the second chromosome (bit string), and vice versa. Note that the point at which the crossover occurs, which determines the length of the exchanged segment, is selected at random. This process is represented pictorially below.



In the mutation step one flipping one of the bits in the bit string at random, with a probability p_{mut}, another meta-parameter of the algorithm. That is, we iterate over the bit strings resulting from the crossover step, and for every bit in the bit string, a random number is drawn between 0 and 1, and if this number is less than p_{mut}, the bit is flipped: changed from 0 to 1 or 1 to 0.

Having finally developed machinery for encoding information into a quantum system, we must now understand how to read that information out. It is trivial to measure a state with observable $\mathcal{O}_0 = Z^{\otimes n}$, by performing canonical z-basis measurements on each qubit and counting the number of 1s. Obtaining an even (odd) number of 1s is equivalent to the outcome $+1$ (-1) of observable $\mathcal{O}_0$.

Similarly, it is also trivial to measure a state with observable $\mathcal{O}_{\beta'} = \mathbf{\Gamma}_{\beta'} Z^{\otimes n} \mathbf{\Gamma}_{\beta'}^{\dagger}$, rotating it via $\mathbf{\Gamma}_{\beta'}^{\dagger}$ and then performing regular (canonical) measurements on each qubit, counting the number of 1s. Obtaining an even (odd) number of 1s is equivalent to the outcome $+1$ (-1) of observable $\mathcal{O}_{\beta'}$.

However, both of these measurements fully collapse an n-qubit quantum state and yield a single bit ($+1$ or -1), whereas we would expect to obtain one in 2^n possible outcomes (n bits) when measuring in an n-qubit basis. Specifically, it should be possible to measure in a complete (sub)set of commuting observables $\mathcal{C}_{\beta'}^{s'}$ and obtain the parity for each observable in the set, by collapsing into one of the 2^n context eigenstates $\{|\phi_{z,\alpha}, \mathcal{C}_{\beta'}^{s'}\rangle\}_{z,\alpha}$. Note that we are adding primes to s and β in the measured context $\mathcal{C}_{\beta'}^{s'}$, so as to distinguish from the parameters used in the state-generation.

A circuit similar to the inverse of Fig. 4.19 could be used to map context eigenstates $\{|\phi_{z,\alpha}, \mathcal{C}_{\beta'}^{s'}\rangle\}_{z,\alpha}$ to the canonical basis. To this end, consider the alternative state-generation circuit depicted in Fig. 4.20, which produces the previously defined set of eigenstates $\{|\phi_{z,\alpha}, \mathcal{C}_{\beta}^{s}\rangle\}_{z,\alpha}$, but with slightly different parameters $\{|\phi_{\alpha'}, \mathcal{C}_{\beta}^{s}\rangle\}_{\alpha'}$. We employ the inverse of this circuit to construct a context-measurement circuit, and showcase it for $n = 4$ qubits in Fig. 4.21.

In its general form, this circuit produces the following evolution on an arbitrary n-qubit input state $|\psi\rangle$

$$\mathcal{M}(s', \beta') |\psi\rangle = \left(H \otimes \mathbb{1}^{\otimes(n-1)}\right) \left(S^{s'^{\dagger}} \otimes \mathbb{1}^{\otimes(n-1)}\right) \left(\prod_{i=2}^{n} \mathrm{CX}(1, i)\right) \left(\mathbf{\Gamma}_{\beta'}^{\dagger}\right) |\psi\rangle \,, \tag{4.8}$$

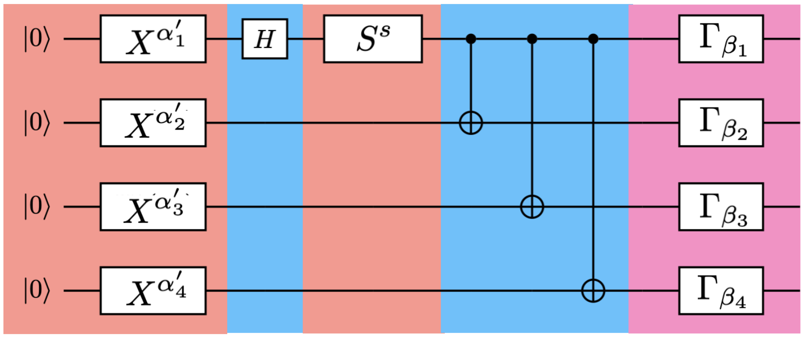


Figure 4.20: ***Alternative context eigenstate preparation.*** Alternative quantum circuit to prepare a context eigenstate. As before, orange sections perform local rotations based on binary parameters, the pink section performs local rotations based on ternary parameters, and the blue sections include the Hadamard and two-qubit gates (but no longer straightforwardly prepare a GHZ state). Furthermore, the circuit is showcased for $n = 4$ qubits, but is generalizable to any n. The original circuit has been modified so that the first gates applied are Pauli X gates which decide which of 2^n eigenstates to prepare. This allows to use the inverse of this circuit to perform a full-context measurement, by mapping context eigenstates to canonical-basis elements. The two-qubit gates employed in this circuit are harder to implement with low depth than those in Fig. 4.19, which is why this circuit is not used for state generation in our implementation.

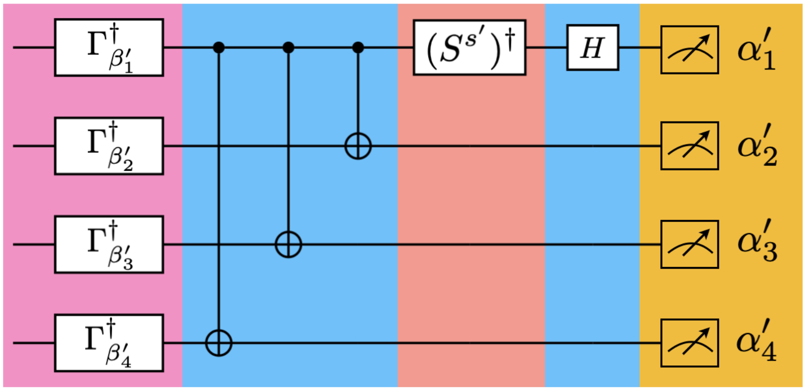

$\Gamma^\dagger_{\beta'_1}$
$\Gamma^\dagger_{\beta'_2}$
$\Gamma^\dagger_{\beta'_3}$
$\Gamma^\dagger_{\beta'_4}$
$(S^{s'})^\dagger$
H
α'_1
α'_2
α'_3
α'_4

Figure 4.21: ***Context measurement.*** Quantum circuit to measure an arbitrary input state in context $\mathcal{C}^{s'}_{\beta'}$. The orange section performs a local rotation based on binary parameter s', the pink section performs local rotations based on ternary parameters β'_1, β'_2, β'_3 and β'_4, the blue sections include the Hadamard and two-qubit gates (which transform GHZ-like states into the canonical basis) and the yellow section highlights the final measurements and their outcomes. The circuit is showcased for $n = 4$ qubits, but is generalizable to an arbitrary number of qubits. To achieve the desired measurement, this circuit maps eigenstates from context $\mathcal{C}^{s'}_{\beta'}$ into canonical-basis elements, and then measures in that basis, obtaining outcomes $\{\alpha'_1, \alpha'_2, \alpha'_3, \alpha'_4\}$.

after which it is measured in the canonical basis. Here, we have defined $\mathrm{CX}(i, j)$ as the two-qubit Control-NOT gate from qubit i (control) to qubit j (target) and the identity $\mathbb{1}$ on all other qubits, so that each $\mathrm{CX}(i, j)$ acts on the n-qubit phase space. The canonical measurement outputs binary values $\{\alpha'_1, \alpha'_2, ..., \alpha'_n\}$, which we can map back to z, α parameters in the context eigenstate notation

$\{|\phi_{z,\alpha}, \mathcal{C}^{s'}_{\beta'}\rangle\}_{z,\alpha}$ using the following rules:

$$\begin{aligned} \alpha_1 &= \alpha'_n \\ \alpha_j = \alpha'_j \oplus \alpha'_n \quad , \quad 2 &\leq j \leq n-1 \\ z &= \alpha'_1 \oplus (s'\alpha'_n). \end{aligned}$$

This way, an n-qubit quantum state $|\psi\rangle$ can be measured in context $\mathcal{C}^{s'}_{\beta'}$ by evolving it with $\mathcal{M}(s', \beta')$ and measuring in the canonical basis. The canonical-measurement outcomes $\{\alpha'_1, \alpha'_2, ..., \alpha'_n\}$ are mapped to z, α with the rules above, and the result of the overall measurement is that the system $|\psi\rangle$ has collapsed to state $|\phi_{z,\alpha}, \mathcal{C}^{s'}_{\beta'}\rangle$. Since the actual state of the system is now available to us from the measurement outcome, we can analytically determine the parity for each observable $\mathcal{O}_{\beta''}$ in context $\mathcal{C}^{s'}_{\beta'}$, effectively measuring $|\psi\rangle$ in those observables.

These encoding and readout protocols were run on the IBM Cairo processor, and since this does not have full connectivity between qubits, we must optimize these idealized circuits to the hardware. In this work, we have employed the state generation and measurement circuits in Figs. 4.19 and 4.21, generalizing them to systems of $n = 8$ qubits. It is desirable in their actual implementation to minimize the number of *layers* with two-qubit gates, also known as *circuit depth*, so as to minimize decoherence. In our state generation circuit, this means finding a low-depth implementation to prepare a GHZ state, and in our measurement circuit this means finding a low-depth implementation of $\prod_{i=2}^{n} \mathrm{CX}(1, i)$, that is, a sequence of Control-NOTs from qubit 1 (control) to all other qubits (target). The implementations need to take into account the architecture used as well. We have employed the IBM-Qiskit Cairo backend, with two-qubit-gate connectivity shown in Fig. 4.22.

Regarding the state generation circuit, note that a k-qubit GHZ state can be turned into a $2k$-qubit GHZ state in a single two-qubit layer if the architecture has sufficient connectivity. This is done by applying CNOTs from each of the k qubits (control), conforming the GHZ state into k qubits (target) initialized in $|0\rangle$. Thus, a GHZ state can be constructed exponentially fast and an 8-qubit GHZ state can be constructed with as few as 3 two-qubit-gate layers. With Cairo's connectivity, we are able to construct it with 4 layers, in the following manner:

$$|\mathrm{GHZ}\rangle = \Big(\mathrm{CX}(7,6)\Big) \Big(\mathrm{CX}(3,5)\,\mathrm{CX}(4,7)\,\mathrm{CX}(1,0)\Big) \Big(\mathrm{CX}(2,3)\,\mathrm{CX}(1,4)\Big) \Big(\mathrm{CX}(1,2)\Big)\, H_1 \,|00000000\rangle\,, \tag{4.9}$$

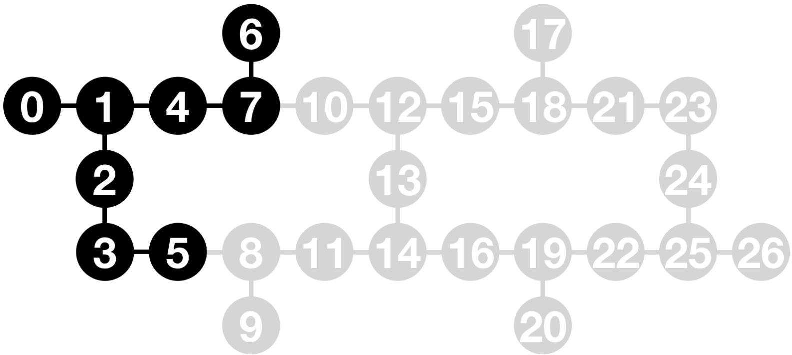


Figure 4.22: ***IBM-Qiskit Cairo connectivity.*** Map of the two-qubit gate connectivity in the Cairo real backend of IBM-Qiskit. Each circle represents a qubit with its corresponding index, and each line represents the possibility of directly performing two-qubit gates between the connected qubits. In our work, we employed the qubits (and connectivity lines) in black, selecting them due to appropriate connectivity for our scheme and lower error rates in a GHZ preparation with respect to other qubits at the time.

where we explicitly indicate each two-qubit-gate layer with big parentheses, and where H_1 denotes Hadamard H on qubit index 1 and identity $\mathbb{1}$ on all other qubits. The actual order of the qubits in $|00000000\rangle$ is irrelevant, as the gates have been specified in terms of their indexes.

Regarding the measurement circuit, an operation that is equivalent to Control-NOTs from one qubit to all others cannot grow exponentially fast in the number of two-qubit-gate layers. Instead k qubits with linear connectivity can do it in $2k-3$ layers such that the common control qubit is operated on just once (the operation order is read from right to left):

$$\begin{aligned}\prod_{i=2}^{k} \mathrm{CX}(1,i) = \Big(\mathrm{CX}(k-1,k)\Big)\Big(\mathrm{CX}(k-2,k-1)\Big)\ldots \\ \Big(\mathrm{CX}(2,3)\Big)\Big(\mathrm{CX}(1,2)\Big)\Big(\mathrm{CX}(2,3)\Big)\ldots\Big(\mathrm{CX}(k-2,k-1)\Big)\Big(\mathrm{CX}(k-1,k)\Big), \quad (4.10)\end{aligned}$$

where we explicitly indicate each two-qubit-gate layer with big parentheses.

This allows to construct an operation in Cairo equivalent to CNOTs from qubit index 1 (control) to all others (target) by considering three groups of linearly-connected qubits in the architecture: $\{1,2,3,5\}$,$\{1,4,7,6\}$ and $\{1,0\}$. The operations in Eq. (4.10) allow to use 5, 5 and 1 layers respectively for each group, requiring to operate on qubit index 1 in the 3rd, 3rd and 1st (and only) layer respectively. The layer in which qubit 1 is operated on is important because that is the only overlap between the groups, and all other two-qubit gates can be done in parallel. This way, the operations for group $\{1,2,3,5\}$ and group $\{1,0\}$ can start at the same time, while group $\{1,4,7,6\}$ is delayed one layer, such that its operation over qubit 1 happens at the 4th layer and does not conflict with group $\{1, 2, 3, 5\}$. This allows to perform $\prod_{i\neq 1} \mathrm{CX}(1, i)$ over $n = 8$ qubits in Cairo using only 6 two-qubit-gate layers, as shown in the following (the operation order is read from right to left):

$$\begin{aligned}\prod_{i\neq 1} \mathrm{CX}(1, i) = &\Big(\mathrm{CX}(7, 6)\Big)\ \Big(\mathrm{CX}(3, 5)\,\mathrm{CX}(4, 7)\Big)\ \Big(\mathrm{CX}(2, 3)\,\mathrm{CX}(1, 4)\Big)\\ &\Big(\mathrm{CX}(1, 2)\,\mathrm{CX}(4, 7)\Big)\ \Big(\mathrm{CX}(2, 3)\,\mathrm{CX}(7, 6)\Big)\ \Big(\mathrm{CX}(1, 0)\,\mathrm{CX}(3, 5)\Big), \end{aligned} \tag{4.11}$$

where we explicitly indicate each two-qubit-gate layer with big parentheses.

This way, our 8-qubit state generation circuit is constructed in a real backend with a depth of 4 two-qubit-gate layers, and the corresponding measurement circuit is constructed with a depth of 6 two-qubit-gate layers, taking advantage of Cairo's connectivity surrounding qubit 1. This relatively low depth of 10 layers allows us to obtain the very high experimental fidelity that we report in our conclusions.

Chapter 5

Conclusion

The field of neutrino astronomy has made great strides in the last decade, driven mostly by the efforts of the IceCube Neutrino Observatory. Since discovering the diffuse astrophysical flux in 2013, IceCube has observed galactic and extragalactic neutrino emission. In this thesis, I have presented IceCube's most recent efforts to study the Sun as a source of high-energy neutrinos. These efforts have improved IceCube's sensitivity to the spin-dependent WIMP-nucleon cross section across three order of magnitude in WIMP mass, with order-of-magnitude improvements seen from updated calculations of the initial neutrino spectrum and improved low-energy reconstruction techniques. Even as we celebrate our recent successes, we must also look toward the future.

The field of neutrino astronomy is at an inflection point. The first sources of astrophysical neutrinos are just now coming into focus, and soon, the instrumented volume of Cherenkov neutrino telescopes will grow by more than an order of magnitude, bringing these sources into sharper relief. Additionally, new experiments using novel detection methods will shed light on currently unexplored aspects of the neutrino sky. This will move us away from a unipolar world in which IceCube alone drives our understanding of astrophysical neutrinos to a world in which many experiments each make up a piece of the puzzle. Furthermore, the promise of machine learning is just now being realized, and there is evidence to suggest that this is only the tip of the iceberg.

In light of these forthcoming monumental changes, it may be prudent to reflect on what we want the future of this field to look like. I am twice biased toward a future driven by open science and collaboration. On one hand, I believe this paradigm leads to less reduplicated work and redundant effort and thus enables us to address the challenges of new detectors and technologies more readily. On the other hand, I believe physics serves a social good by uniting people towards a common goal of advancing our collective understanding. I do not think we can fully achieve that good without open collaboration.

I hope that in addition to contributing to advancing the field of neutrino astronomy scientifically, the work described in this thesis will also push us toward this more collaborative, open future.

List of Figures

List of Tables

Bibliography

[1] J.M. Hardin, I. Martinez-Soler, A. Diaz, M. Jin, N.W. Kamp, C.A. Argüelles et al., *New Clues about light sterile neutrinos: preference for models with damping effects in global fits*, *JHEP* **09** (2023) 058 [2211.02610].

[2] C. Fuji, Y. Matsuura, T. Shibuya and S.Y. Tsai, *A Wave-Packet View of Neutrino Oscillation and Pion Decay*, hep-ph/0612300.

[3] C.A. Argüelles, T. Bertólez-Martínez and J. Salvado, *Impact of wave packet separation in low-energy sterile neutrino searches*, *Phys. Rev. D* **107** (2023) 036004 [2201.05108].

[4] F. Halzen and A.D. Martin, *QUARKS AND LEPTONS: AN INTRODUCTORY COURSE IN MODERN PARTICLE PHYSICS*, Wiley (1984).

[5] Particle Data Group collaboration, *Review of Particle Physics*, *PTEP* **2020** (2020) 083C01.

[6] IceCube collaboration, *The IceCube Neutrino Observatory: Instrumentation and Online Systems*, *JINST* **12** (2017) P03012 [1612.05093].

[7] K.G. Begeman, A.H. Broeils and R.H. Sanders, *Extended rotation curves of spiral galaxies: Dark haloes and modified dynamics*, *Mon. Not. Roy. Astron. Soc.* **249** (1991) 523.

[8] E. Corbelli and P. Salucci, *The Extended Rotation Curve and the Dark Matter Halo of M33*, *Mon. Not. Roy. Astron. Soc.* **311** (2000) 441 [astro-ph/9909252].

[9] Planck collaboration, *Planck 2018 results. VI. Cosmological parameters*, *Astron. Astrophys.* **641** (2020) A6 [1807.06209].

[10] D. Hooper and G.D. Kribs, *Probing Kaluza-Klein dark matter with neutrino telescopes*, *Phys. Rev.* **D67** (2003) 055003 [hep-ph/0208261].

[11] T. Sjöstrand, S. Ask, J.R. Christiansen, R. Corke, N. Desai, P. Ilten et al., *An Introduction to PYTHIA 8.2*, *Comput. Phys. Commun.* **191** (2015) 159 [1410.3012].

[12] T. Sjöstrand, *The PYTHIA Event Generator: Past, Present and Future*, *Comput. Phys. Commun.* **246** (2020) 106910 [1907.09874].

[13] C.W. Bauer, N.L. Rodd and B.R. Webber, *Dark Matter Spectra from the Electroweak to the Planck Scale*, 2007.15001.

[14] N. Bernal, J. Martín-Albo and S. Palomares-Ruiz, *A novel way of constraining WIMPs annihilations in the Sun: MeV neutrinos*, *JCAP* **08** (2013) 011 [1208.0834].

[15] S. Ritz and D. Seckel, *Detailed Neutrino Spectra From Cold Dark Matter Annihilations in the Sun*, *Nucl. Phys.* **B304** (1988) 877.

[16] J.D. Jackson, *Classical electrodynamics; 3rd ed.*, Wiley, New York, NY (1999. See chapter 13 problem 13.6(b).).

[17] Particle Data Group collaboration, *Review of particle physics*, Phys. Rev. D **98** (2018) 030001.

[18] J.H. Koehne, K. Frantzen, M. Schmitz, T. Fuchs, W. Rhode, D. Chirkin et al., *PROPOSAL: A tool for propagation of charged leptons*, Comput. Phys. Commun. **184** (2013) 2070.

[19] L. Ponomarev, *Molecular structure effects on atomic and nuclear capture of mesons*, Ann. Rev. Nucl. Part. Sci. **23** (1973) 395.

[20] M.L. Good, *Relation between Scattering and Absorption in the Pais-Piccioni Phenomenon*, Phys. Rev. **106** (1957) 591.

[21] P. Baratella, M. Cirelli, A. Hektor, J. Pata, M. Piibeleht and A. Strumia, *PPPC 4 DMν: a Poor Particle Physicist Cookbook for Neutrinos from Dark Matter annihilations in the Sun*, JCAP **1403** (2014) 053 [1312.6408].

[22] C. Rott, J. Siegal-Gaskins and J.F. Beacom, *New Sensitivity to Solar WIMP Annihilation using Low-Energy Neutrinos*, Phys. Rev. D **88** (2013) 055005 [1208.0827].

[23] C. Rott, S. In, J. Kumar and D. Yaylali, *Dark Matter Searches for Monoenergetic Neutrinos Arising from Stopped Meson Decay in the Sun*, JCAP **11** (2015) 039 [1510.00170].

[24] C. Rott, S. In, J. Kumar and D. Yaylali, *Directional Searches at DUNE for Sub-GeV Monoenergetic Neutrinos Arising from Dark Matter Annihilation in the Sun*, JCAP **01** (2017) 016 [1609.04876].

[25] P. Ciafaloni, D. Comelli, A. Riotto, F. Sala, A. Strumia and A. Urbano, *Weak Corrections are Relevant for Dark Matter Indirect Detection*, JCAP **1103** (2011) 019 [1009.0224].

[26] C. Garcia-Cely and J. Heeck, *Indirect searches of dark matter via polynomial spectral features*, JCAP **08** (2016) 023 [1605.08049].

[27] C. Niblaeus, J.M. Cornell and J. Edsjö, *Effect of polarisation and choice of event generator on spectra from dark matter annihilations*, JCAP **1910** (2019) 079 [1907.02488].

[28] R. Gandhi, C. Quigg, M.H. Reno and I. Sarcevic, *Neutrino interactions at ultrahigh-energies*, Phys. Rev. D **58** (1998) 093009 [hep-ph/9807264].

[29] J. Formaggio and G. Zeller, *From eV to EeV: Neutrino Cross Sections Across Energy Scales*, Rev. Mod. Phys. **84** (2012) 1307 [1305.7513].

[30] S. Kretzer and M. Reno, *Tau neutrino deep inelastic charged current interactions*, Phys. Rev. D **66** (2002) 113007 [hep-ph/0208187].

[31] E. Paschos and J. Yu, *Neutrino interactions in oscillation experiments*, Phys. Rev. D **65** (2002) 033002 [hep-ph/0107261].

[32] K. Hagiwara, K. Mawatari and H. Yokoya, *Tau polarization in tau neutrino nucleon scattering*, Nucl. Phys. B **668** (2003) 364 [hep-ph/0305324].

[33] F. Halzen and D. Saltzberg, *Tau-neutrino appearance with a 1000 megaparsec baseline*, Phys. Rev. Lett. **81** (1998) 4305 [hep-ph/9804354].

[34] C. Giunti and C.W. Kim, *Fundamentals of Neutrino Physics and Astrophysics*, Oxford University Press (4, 2007).

[35] I. Esteban, M.C. Gonzalez-Garcia, A. Hernandez-Cabezudo, M. Maltoni and T. Schwetz, *Global analysis of three-flavour neutrino oscillations: synergies and tensions in the determination of θ_{23}, δ_{CP}, and the mass ordering*, *JHEP* **01** (2019) 106 [1811.05487].

[36] T2K collaboration, *Constraint on the matter–antimatter symmetry-violating phase in neutrino oscillations*, *Nature* **580** (2020) 339 [1910.03887].

[37] C.A. Argüelles, J. Salvado and C.N. Weaver, "ν-SQuIDS." https://github.com/Arguelles/nuSQuIDS, 2015.

[38] C.A. Argüelles, J. Salvado and C.N. Weaver, *A Simple Quantum Integro-Differential Solver (SQuIDS)*, *Comput. Phys. Commun.* **255** (2020) 107405.

[39] I. Esteban, M.C. Gonzalez-Garcia, A. Hernandez-Cabezudo, M. Maltoni and T. Schwetz, "NuFIT." NuFIT4.1(2019),www.nu-fit.org, 2019.

[40] J. Edsjö, "Calculation of neutrino cross sections and the nusigma neutrino-nucleon scattering monte carlo."

[41] N. Vinyoles, A.M. Serenelli, F.L. Villante, S. Basu, J. Bergström, M.C. Gonzalez-Garcia et al., *A new Generation of Standard Solar Models*, *Astrophys. J.* **835** (2017) 202 [1611.09867].

[42] A.M. Dziewonski and D.L. Anderson, *Preliminary reference earth model*, *Phys. Earth Planet. Interiors* **25** (1981) 297.

[43] S. Amoroso, S. Caron, A. Jueid, R. Ruiz de Austri and P. Skands, *Estimating QCD uncertainties in Monte Carlo event generators for gamma-ray dark matter searches*, *JCAP* **05** (2019) 007 [1812.07424].

[44] IceCube collaboration, *Search for Neutrinos from Dark Matter Self-Annihilations in the center of the Milky Way with 3 years of IceCube/DeepCore*, *Eur. Phys. J. C* **77** (2017) 627 [1705.08103].

[45] ANTARES, IceCube collaboration, *Combined search for neutrinos from dark matter self-annihilation in the Galactic Centre with ANTARES and IceCube*, 2003.06614.

[46] A. Albert et al., *Results from the search for dark matter in the Milky Way with 9 years of data of the ANTARES neutrino telescope*, *Phys. Lett. B* **769** (2017) 249 [1612.04595].

[47] Super-Kamiokande collaboration, *Indirect Search for Dark Matter from the Galactic Center and Halo with the Super-Kamiokande Detector*, 2005.05109.

[48] P. Gondolo, J. Edsjo, P. Ullio, L. Bergstrom, M. Schelke and E. Baltz, *DarkSUSY: Computing supersymmetric dark matter properties numerically*, *JCAP* **07** (2004) 008 [astro-ph/0406204].

[49] T. Bringmann, J. Edsjö, P. Gondolo, P. Ullio and L. Bergström, *DarkSUSY 6 : An Advanced Tool to Compute Dark Matter Properties Numerically*, *JCAP* **07** (2018) 033 [1802.03399].

[50] M. Cirelli, N. Fornengo, T. Montaruli, I.A. Sokalski, A. Strumia and F. Vissani, *Spectra of neutrinos from dark matter annihilations*, *Nucl. Phys.* **B727** (2005) 99 [hep-ph/0506298].

[51] A. Cooper-Sarkar, P. Mertsch and S. Sarkar, *The high energy neutrino cross-section in the Standard Model and its uncertainty*, *JHEP* **08** (2011) 042 [1106.3723].

[52] ANTARES collaboration, *Search for Dark Matter Annihilation in the Earth using the ANTARES Neutrino Telescope*, *Phys. Dark Univ.* **16** (2017) 41 [1612.06792].

[53] IceCube collaboration, *First search for dark matter annihilations in the Earth with the IceCube Detector*, *Eur. Phys. J. C* **77** (2017) 82 [1609.01492].

[54] Super-Kamiokande collaboration, *Dark matter searches with the Super-Kamiokande detector*, *J. Phys. Conf. Ser.* **888** (2017) 012210.

[55] K. Griest and D. Seckel, *Cosmic Asymmetry, Neutrinos and the Sun*, *Nucl. Phys. B* **283** (1987) 681.

[56] G. Jungman, M. Kamionkowski and K. Griest, *Supersymmetric dark matter*, *Phys. Rept.* **267** (1996) 195 [hep-ph/9506380].

[57] Fermi-LAT collaboration, *Fermi-LAT Observations of Two Gamma-Ray Emission Components from the Quiescent Sun*, *Astrophys. J.* **734** (2011) 116 [1104.2093].

[58] HAWC collaboration, *Discovery of Gamma Rays from the Quiescent Sun with HAWC*, *Phys. Rev. Lett.* **131** (2023) 051201 [2212.00815].

[59] C.A. Argüelles, G. de Wasseige, A. Fedynitch and B.J.P. Jones, *Solar Atmospheric Neutrinos and the Sensitivity Floor for Solar Dark Matter Annihilation Searches*, *JCAP* **07** (2017) 024 [1703.07798].

[60] K.C.Y. Ng, J.F. Beacom, A.H.G. Peter and C. Rott, *Solar Atmospheric Neutrinos: A New Neutrino Floor for Dark Matter Searches*, *Phys. Rev. D* **96** (2017) 103006 [1703.10280].

[61] J. Edsjo, J. Elevant, R. Enberg and C. Niblaeus, *Neutrinos from cosmic ray interactions in the Sun*, *JCAP* **06** (2017) 033 [1704.02892].

[62] A. Fedynitch, J.B. Tjus and P. Desiati, *Influence of hadronic interaction models and the cosmic ray spectrum on the high energy atmospheric muon and neutrino flux*, *Physical Review D* **86** (2012) .

[63] F. Riehn, R. Engel, A. Fedynitch, T.K. Gaisser and T. Stanev, *Charm production in SIBYLL*, *EPJ Web Conf.* **99** (2015) 12001 [1502.06353].

[64] F. Riehn, R. Engel, A. Fedynitch, T.K. Gaisser and T. Stanev, *A new version of the event generator Sibyll*, *PoS* **ICRC2015** (2016) 558 [1510.00568].

[65] A.D. Martin, M.G. Ryskin and A.M. Stasto, *Prompt neutrinos from atmospheric $c\bar{c}$ and $b\bar{b}$ production and the gluon at very small x*, *Acta Phys. Polon. B* **34** (2003) 3273 [hep-ph/0302140].

[66] G. Roellinghoff, *Search for Solar Atmospheric Neutrinos using 9 years of IceCube Neutrino Detector data*, Master thesis, Sungkyunkwan University, 2022.

[67] IceCube collaboration, *The IceCube high-energy starting event sample: Description and flux characterization with 7.5 years of data*, *Phys. Rev. D* **104** (2021) 022002 [2011.03545].

[68] K. Greisen, *End to the cosmic ray spectrum?*, *Phys. Rev. Lett.* **16** (1966) 748.

[69] G.T. Zatsepin and V.A. Kuzmin, *Upper limit of the spectrum of cosmic rays*, *JETP Lett.* **4** (1966) 78.

[70] F. Halzen, T. Stelzer and M. Kamionkowski, *Signatures of dark matter in underground detectors*, *Phys. Rev.* **D45** (1992) 4439.

[71] C.A. Argüelles, J. Salvado and C.N. Weaver, "nuSQuIDS." https://github.com/arguelles/nuSQuIDS, 2015.

[72] "Prob3++ software for computing three flavor neutrino oscillation probabilities.." http://www.phy.duke.edu/~raw22/public/Prob3++/.

[73] G. Barenboim, P.B. Denton, S.J. Parke and C.A. Ternes, *Neutrino Oscillation Probabilities through the Looking Glass*, Phys. Lett. B **791** (2019) 351 [1902.00517].

[74] P. Huber, J. Kopp, M. Lindner, M. Rolinec and W. Winter, *New features in the simulation of neutrino oscillation experiments with GLoBES 3.0: General Long Baseline Experiment Simulator*, Comput. Phys. Commun. **177** (2007) 432 [hep-ph/0701187].

[75] R.G. Calland, A.C. Kaboth and D. Payne, *Accelerated Event-by-Event Neutrino Oscillation Reweighting with Matter Effects on a GPU*, JINST **9** (2014) P04016 [1311.7579].

[76] M. Wallraff and C. Wiebusch, *Calculation of oscillation probabilities of atmospheric neutrinos using nucraft*, 1409.1387.

[77] C.A. Argüelles and B.J.P. Jones, *Neutrino Oscillations in a Quantum Processor*, Phys. Rev. Research. **1** (2019) 033176 [1904.10559].

[78] IceCube-Gen2 collaboration, *IceCube-Gen2: the window to the extreme Universe*, J. Phys. G **48** (2021) 060501 [2008.04323].

[79] POEMMA collaboration, *The POEMMA (Probe of Extreme Multi-Messenger Astrophysics) observatory*, JCAP **06** (2021) 007 [2012.07945].

[80] A. Garcia, R. Gauld, A. Heijboer and J. Rojo, *Complete predictions for high-energy neutrino propagation in matter*, JCAP **09** (2020) 025 [2004.04756].

[81] KM3Net collaboration, *Letter of intent for KM3NeT 2.0*, J. Phys. G **43** (2016) 084001 [1601.07459].

[82] I. Safa, A. Pizzuto, C.A. Argüelles, F. Halzen, R. Hussain, A. Kheirandish et al., *Observing EeV neutrinos through Earth: GZK and the anomalous ANITA events*, JCAP **01** (2020) 012 [1909.10487].

[83] O. Vazquez, I. Safa, J. Lazar, A. Pizzuto, C. Arguelles, A. Kheirandish et al., *TauRunner: A Monte Carlo for Very-High-Energy Tau Neutrino Propagation*, PoS **ICRC2021** (2021) 1030.

[84] ANITA collaboration, *Characteristics of Four Upward-pointing Cosmic-ray-like Events Observed with ANITA*, Phys. Rev. Lett. **117** (2016) 071101 [1603.05218].

[85] ANITA collaboration, *Observation of an Unusual Upward-going Cosmic-ray-like Event in the Third Flight of ANITA*, Phys. Rev. Lett. **121** (2018) 161102 [1803.05088].

[86] IceCube collaboration, *IceCube Search for Earth-traversing ultra-high energy Neutrinos*, PoS **ICRC2021** (2021) 1170.

[87] P-ONE collaboration, *The Pacific Ocean Neutrino Experiment*, Nature Astron. **4** (2020) 913 [2005.09493].

[88] M. Sasaki and T. Kifune, *Ashra Neutrino Telescope Array (NTA): Combined Imaging Observation of Astroparticles — For Clear Identification of Cosmic Accelerators and Fundamental Physics Using Cosmic Beams —*, JPS Conf. Proc. **15** (2017) 011013.

[89] A. Romero-Wolf et al., *An Andean Deep-Valley Detector for High-Energy Tau Neutrinos*, in *Latin American Strategy Forum for Research Infrastructure*, 2, 2020 [2002.06475].

[90] NuSpaceSim collaboration, *Monte Carlo simulations of neutrino and charged lepton propagation in the Earth with nuPyProp*, *PoS* **ICRC2021** (2021) 1203 [2109.08198].

[91] J. Alvarez-Muñiz, W.R. Carvalho, A.L. Cummings, K. Payet, A. Romero-Wolf, H. Schoorlemmer et al., *Comprehensive approach to tau-lepton production by high-energy tau neutrinos propagating through the Earth*, *Phys. Rev. D* **97** (2018) 023021 [1707.00334].

[92] M.C. Gonzalez-Garcia, F. Halzen and M. Maltoni, *Physics reach of high-energy and high-statistics icecube atmospheric neutrino data*, *Phys. Rev.* **D71** (2005) 093010 [hep-ph/0502223].

[93] J.F. Beacom, P. Crotty and E.W. Kolb, *Enhanced Signal of Astrophysical Tau Neutrinos Propagating through Earth*, *Phys. Rev. D* **66** (2002) 021302 [astro-ph/0111482].

[94] S.I. Dutta, M.H. Reno and I. Sarcevic, *Secondary neutrinos from tau neutrino interactions in earth*, *Phys. Rev. D* **66** (2002) 077302 [hep-ph/0207344].

[95] Particle Data Group collaboration, *Review of Particle Physics*, *PTEP* **2020** (2020) 083C01.

[96] MiniBooNE collaboration, *First Measurement of the Muon Neutrino Charged Current Quasielastic Double Differential Cross Section*, *Phys. Rev.* **D81** (2010) 092005 [1002.2680].

[97] NuTeV collaboration, *Precise measurement of neutrino and anti-neutrino differential cross sections*, *Phys. Rev.* **D74** (2006) 012008 [hep-ex/0509010].

[98] P. Vogel and J.F. Beacom, *Angular distribution of neutron inverse beta decay, anti-neutrino(e) + p —> e+ + n*, *Phys. Rev.* **D60** (1999) 053003 [hep-ph/9903554].

[99] A. Kurylov, M.J. Ramsey-Musolf and P. Vogel, *Radiative corrections to low-energy neutrino reactions*, *Phys. Rev.* **C67** (2003) 035502 [hep-ph/0211306].

[100] BOREXINO collaboration, *Comprehensive measurement of pp-chain solar neutrinos*, *Nature* **562** (2018) 505.

[101] Super-Kamiokande collaboration, *Measurement of the tau neutrino cross section in atmospheric neutrino oscillations with Super-Kamiokande*, *Phys. Rev.* **D98** (2018) 052006 [1711.09436].

[102] IceCube collaboration, *Measurement of the multi-TeV neutrino cross section with IceCube using Earth absorption*, *Nature* **551** (2017) 596 [1711.08119].

[103] IceCube collaboration, *Measurement of the high-energy all-flavor neutrino-nucleon cross section with IceCube*, 2011.03560.

[104] M. Froissart, *Asymptotic behavior and subtractions in the Mandelstam representation*, *Phys. Rev.* **123** (1961) 1053.

[105] C.A. Argüelles, F. Halzen, L. Wille, M. Kroll and M.H. Reno, *High-energy behavior of photon, neutrino, and proton cross sections*, *Phys. Rev. D* **92** (2015) 074040 [1504.06639].

[106] M.M. Block and F. Halzen, *Experimental Confirmation that the Proton is Asymptotically a Black Disk*, *Phys. Rev. Lett.* **107** (2011) 212002 [1109.2041].

[107] Particle Data Group collaboration, *Review of particle properties. Particle Data Group*, *Phys. Rev. D* **50** (1994) 1173.

[108] G. de Wasseige. personal communication, 8, 2020.

[109] M.M. Block, L. Durand and P. Ha, *Connection of the virtual γ^*p cross section of ep deep inelastic scattering to real γp scattering, and the implications for νn and ep total cross sections*, *Physical Review D* **89** (2014) 094027.

[110] ANITA collaboration, *Characteristics of Four Upward-pointing Cosmic-ray-like Events Observed with ANITA*, *Phys. Rev. Lett.* **117** (2016) 071101 [1603.05218].

[111] ANITA collaboration, *Observation of an Unusual Upward-going Cosmic-ray-like Event in the Third Flight of ANITA*, *Phys. Rev. Lett.* **121** (2018) 161102 [1803.05088].

[112] L.A. Anchordoqui, V. Barger, J.G. Learned, D. Marfatia and T.J. Weiler, *Upgoing ANITA events as evidence of the CPT symmetric universe*, *LHEP* **1** (2018) 13 [1803.11554].

[113] IceCube collaboration, *Search for annihilating dark matter in the Sun with 3 years of IceCube data*, *Eur. Phys. J. C* **77** (2017) 146 [1612.05949].

[114] M. Ackermann et al., *High-energy and ultra-high-energy neutrinos: A Snowmass white paper*, *JHEAp* **36** (2022) 55 [2203.08096].

[115] IceCube collaboration, *The IceCube Neutrino Observatory - Contributions to ICRC 2017 Part II: Properties of the Atmospheric and Astrophysical Neutrino Flux*, 1710.01191.

[116] IceCube collaboration, *A combined maximum-likelihood analysis of the high-energy astrophysical neutrino flux measured with IceCube*, *Astrophys. J.* **809** (2015) 98 [1507.03991].

[117] D. Heck, J. Knapp, J.N. Capdevielle, G. Schatz and T. Thouw, *CORSIKA: A Monte Carlo code to simulate extensive air showers*, .

[118] CORSIKA collaboration, *The particle-shower simulation code CORSIKA 8*, *PoS* **ICRC2023** (2023) 310 [2308.05475].

[119] J. Eschle et al., *Potential of the Julia programming language for high energy physics computing*, 2306.03675.

[120] J. Lazar et al., *Tambo-mc*, GitHub, https://github.com/Harvard-Neutrino/TAMBO-MC/tree/main.

[121] A. Cooper-Sarkar, P. Mertsch and S. Sarkar, *The high energy neutrino cross-section in the Standard Model and its uncertainty*, *JHEP* **08** (2011) 042 [1106.3723].

[122] KM3NeT collaboration, *High-energy neutrino event simulation at NLO in Genie for KM3NeT and other observatories*, *PoS* **ICRC2019** (2020) 895 [1908.10077].

[123] J.-H. Koehne, K. Frantzen, M. Schmitz, T. Fuchs, W. Rhode, D. Chirkin et al., *Proposal: A tool for propagation of charged leptons*, *Computer Physics Communications* **184** (2013) 2070.

[124] IceCube collaboration, *LeptonInjector and LeptonWeighter: A neutrino event generator and weighter for neutrino observatories*, *Comput. Phys. Commun.* **266** (2021) 108018 [2012.10449].

[125] J.R. Klein et al., *SNOWMASS Neutrino Frontier NF10 Topical Group Report: Neutrino Detectors*, 2211.09669.

[126] BAIKAL collaboration, *A search for neutrino signal from dark matter annihilation in the center of the Milky Way with Baikal NT200*, *Astropart. Phys.* **81** (2016) 12 [1512.01198].

[127] A.D. Avrorin et al., *Status and recent results of the BAIKAL-GVD project*, *Phys. Part. Nucl.* **46** (2015) 211.

[128] Z.P. Ye et al., *Proposal for a neutrino telescope in South China Sea*, 2207.04519.

[129] KM3NeT collaboration, *Letter of intent for KM3NeT 2.0*, J. Phys. G **43** (2016) 084001 [1601.07459].

[130] A.M. Brown, M. Bagheri, M. Doro, E. Gazda, D. Kieda, C. Lin et al., *Trinity: An Imaging Air Cherenkov Telescope to Search for Ultra-High-Energy Neutrinos*, in *37th International Cosmic Ray Conference*, 9, 2021 [2109.03125].

[131] C. Spiering, *Towards High-Energy Neutrino Astronomy. A Historical Review*, Eur. Phys. J. H **37** (2012) 515 [1207.4952].

[132] A. Roberts, *The birth of high-energy neutrino astronomy: A personal history of the dumand project*, Rev. Mod. Phys. **64** (1992) 259.

[133] IceCube collaboration, *The IceCube Upgrade - Design and Science Goals*, *PoS* **ICRC2019** (2021) 1031 [1908.09441].

[134] A. Roberts, *Monte Carlo Simulation of Inelastic Neutrino Scattering in DUMAND*, in *DUMAND - Deep Underwater Muon and Neutrino Detection 1978 Summer Workshop, Session 2: Ultra High Energy Interactions and Astrophysical Neutrino Sources*, 12, 1978, DOI.

[135] G.C. Hill, *Experimental and theoretical aspects of high energy neutrino astrophysics*, Ph.D. thesis, Adelaide U., 9, 1996.

[136] A. Gazizov and M.P. Kowalski, *ANIS: High energy neutrino generator for neutrino telescopes*, Comput. Phys. Commun. **172** (2005) 203 [astro-ph/0406439].

[137] S. Yoshida, R. Ishibashi and H. Miyamoto, *Propagation of extremely - high energy leptons in the earth: Implications to their detection by the IceCube Neutrino Telescope*, Phys. Rev. D **69** (2004) 103004 [astro-ph/0312078].

[138] D.J. Bailey, *Monte Carlo tools and analysis methods for understanding the ANTARES experiment and predicting its sensitivity to Dark Matter*, Ph.D. thesis, Wolfson College, 2002.

[139] T.R. De Young, *IceTray: a Software Framework for IceCube*, .

[140] KM3NeT collaboration, *gSeaGen: The KM3NeT GENIE-based code for neutrino telescopes*, Comput. Phys. Commun. **256** (2020) 107477 [2003.14040].

[141] IceCube, "`LeptonInjector` code." https://github.com/icecube/LeptonInjector, 2020.

[142] IceCube, "`LeptonWeighter` code." https://github.com/icecube/LeptonWeighter, 2020.

[143] A.C. Vincent, C.A. Argüelles and A. Kheirandish, *High-energy neutrino attenuation in the Earth and its associated uncertainties*, JCAP **11** (2017) 012 [1706.09895].

[144] I. Safa, J. Lazar, A. Pizzuto, O. Vasquez, C.A. Argüelles and J. Vandenbroucke, *TauRunner: A public Python program to propagate neutral and charged leptons*, Comput. Phys. Commun. **278** (2022) 108422 [2110.14662].

[145] C.A. Argüelles, J. Salvado and C.N. Weaver, *nuSQuIDS: A toolbox for neutrino propagation*, Comput. Phys. Commun. **277** (2022) 108346 [2112.13804].

[146] D. Garg et al., *Neutrino propagation in the Earth and emerging charged leptons with* ***nuPyProp***, 2209.15581.

[147] P. Lipari and T. Stanev, *Propagation of multi - TeV muons*, Phys. Rev. D **44** (1991) 3543.

[148] A. Fedynitch, W. Woodley and M.-C. Piro, *On the Accuracy of Underground Muon Intensity Calculations*, *Astrophys. J.* **928** (2022) 27 [2109.11559].

[149] P. Antonioli, C. Ghetti, E.V. Korolkova, V.A. Kudryavtsev and G. Sartorelli, *A Three-dimensional code for muon propagation through the rock: Music*, *Astropart. Phys.* **7** (1997) 357 [hep-ph/9705408].

[150] D. Chirkin and W. Rhode, *Muon Monte Carlo: A High-precision tool for muon propagation through matter*, hep-ph/0407075.

[151] E.V. Bugaev and Y.V. Shlepin, *Photonuclear interaction of high-energy muons and tau leptons*, *Phys. Rev. D* **67** (2003) 034027 [hep-ph/0203096].

[152] S.I. Dutta, Y. Huang and M.H. Reno, *Tau neutrino propagation and tau energy loss*, *Phys. Rev. D* **72** (2005) 013005 [hep-ph/0504208].

[153] O.B. Bigas, O. Deligny, K. Payet and V. Van Elewyck, *Tau energy losses at ultra-high energy: Continuous versus stochastic treatment*, *Phys. Rev. D* **77** (2008) 103004 [0802.1119].

[154] C.A. Argüelles, D. Garg, S. Patel, M.H. Reno and I. Safa, *Tau depolarization at very high energies for neutrino telescopes*, *Phys. Rev. D* **106** (2022) 043008 [2205.05629].

[155] C.H.V. Wiebusch, *The Detection of Faint Light in Deep Underwater Neutrino Telescopes, Chapter 7*, Ph.D. thesis, Physikalishe Institute RWTH Aachen, 12, 1995.

[156] V. Niess and V. Bertin, *Underwater acoustic detection of ultra high energy neutrinos*, *Astroparticle Physics* **26** (2006) 243–256.

[157] L. Rädel and C. Wiebusch, *Calculation of the Cherenkov light yield from low energetic secondary particles accompanying high-energy muons in ice and water with Geant4 simulations*, *Astroparticle Physics* **38** (2012) 53 [1206.5530].

[158] L. Rädel and C. Wiebusch, *Calculation of the Cherenkov light yield from electromagnetic cascades in ice with Geant4*, *Astropart. Phys.* **44** (2013) 102 [1210.5140].

[159] IceCube collaboration, *Photon Propagation with GPUs in IceCube*, in *GPU Computing in High-Energy Physics*, pp. 217–220, 2015, DOI.

[160] D. Chirkin, "ppc." https://github.com/icecube/ppc, 2022.

[161] "Documentation | Apache Parquet." https://parquet.apache.org/docs/.

[162] J. Blomer, *A quantitative review of data formats for hep analyses*, *Journal of Physics: Conference Series* **1085** (2018) 032020.

[163] S. Melnik, A. Gubarev, J.J. Long, G. Romer, S. Shivakumar, M. Tolton et al., *Dremel: Interactive analysis of web-scale datasets*, in *Proc. of the 36th Int'l Conf on Very Large Data Bases*, pp. 330–339, 2010, http://www.vldb2010.org/accept.htm.

[164] J. Blomer, *A quantitative review of data formats for hep analyses*, in *Journal of Physics: Conference Series*, vol. 1085, p. 032020, IOP Publishing, 2018.

[165] A. Buckley, T. Eifert, M. Elsing, D. Gillberg, K. Koeneke, A. Krasznahorkay et al., *Implementation of the atlas run 2 event data model*, in *Journal of Physics: Conference Series*, vol. 664, p. 072045, IOP Publishing, 2015.

[166] D. Chrikin. personal communication.

[167] IceCube collaboration, *Detection of astrophysical tau neutrino candidates in IceCube*, *Eur. Phys. J. C* **82** (2022) 1031 [2011.03561].

[168] C. Andreopoulos, C. Barry, S. Dytman, H. Gallagher, T. Golan, R. Hatcher et al., *The GENIE Neutrino Monte Carlo Generator: Physics and User Manual*, 1510.05494.

[169] S. Meighen-Berger, *Fennel: Light from tracks and cascades*, 2022.

[170] GEANT4 collaboration, *GEANT4–a simulation toolkit*, *Nucl. Instrum. Meth. A* **506** (2003) 250.

[171] D. Chirkin, "Photon tracking with gpus in icecube." 2011.

[172] M. Rongen, *Calibration of the IceCube Neutrino Observatory*, Ph.D. thesis, RWTH Aachen U., 2019. 1911.02016. 10.18154/RWTH-2019-09941.

[173] IceCube collaboration, *Light diffusion in birefringent polycrystals and the IceCube ice anisotropy*, *PoS* **ICRC2019** (2020) 854 [1908.07608].

[174] G.H. Collin, *Using path integrals for the propagation of light in a scattering dominated medium*, 1811.04156.

[175] IceCube collaboration, *IceCube*, *PoS* **ICRC2009** (2010) [1003.5715].

[176] KM3NeT collaboration, *Sensitivity for astrophysical neutrino searches with KM3NeT-ORCA*, *PoS* **ICRC2019** (2020) 934.

[177] IceCube collaboration, *Neutrino emission from the direction of the blazar TXS 0506+056 prior to the IceCube-170922A alert*, *Science* **361** (2018) 147 [1807.08794].

[178] IceCube collaboration, *Evidence for neutrino emission from the nearby active galaxy NGC 1068*, *Science* **378** (2022) 538 [2211.09972].

[179] IceCube, MAGIC, VERITAS collaboration, *Very High-Energy Gamma-Ray Follow-Up Program Using Neutrino Triggers from IceCube*, *JINST* **11** (2016) P11009 [1610.01814].

[180] Baikal-GVD collaboration, *BAIKAL-GVD: Gigaton Volume Detector in Lake Baikal*, .

[181] B. Bakker, *Trigger studies for the Antares and KM3NeT neutrino telescopes*, July, 2011.

[182] M. Hünnefeld, *Online Reconstruction of Muon-Neutrino Events in IceCube using Deep Learning Techniques*, Master thesis, Technische Universität Dortmund, 2017.

[183] A. Albert et al., *Review of the online analyses of multi-messenger alerts and electromagnetic transient events with the ANTARES neutrino telescope*, 2211.07551.

[184] M. Aartsen et al., *Improvement in fast particle track reconstruction with robust statistics*, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **736** (2014) 143.

[185] ANTARES collaboration, *A fast algorithm for muon track reconstruction and its application to the ANTARES neutrino telescope*, *Astropart. Phys.* **34** (2011) 652 [1105.4116].

[186] ANTARES collaboration, *Deep learning reconstruction in ANTARES*, *JINST* **16** (2021) C09018 [2107.13654].

[187] A. Radovic, M. Williams, D. Rousseau, M. Kagan, D. Bonacorsi, A. Himmel et al., *Machine learning at the energy and intensity frontiers of particle physics*, *Nature* **560** (2018) 41.

[188] A. Aurisano, A. Radovic, D. Rocco, A. Himmel, M.D. Messier, E. Niner et al., *A Convolutional Neural Network Neutrino Event Classifier*, *JINST* **11** (2016) P09001 [1604.01444].

[189] MicroBooNE collaboration, *Convolutional Neural Networks Applied to Neutrino Events in a Liquid Argon Time Projection Chamber*, *JINST* **12** (2017) P03011 [1611.05531].

[190] R. Abbasi et al., *A Convolutional Neural Network based Cascade Reconstruction for the IceCube Neutrino Observatory*, *JINST* **16** (2021) P07041 [2101.11589].

[191] KM3NeT collaboration, *KM3NeT: Technical Design Report for a Deep-Sea Research Infrastructure in the Mediterranean Sea Incorporating a Very Large Volume Neutrino Telescope*, .

[192] IceCube collaboration, *IceCube Preliminary Design Document*, .

[193] B. Graham and L. van der Maaten, *Submanifold sparse convolutional networks*, 2017. 10.48550/ARXIV.1706.01307.

[194] P. Abratenko et al., *Semantic segmentation with a sparse convolutional neural network for event reconstruction in MicroBooNE*, *Physical Review D* **103** (2021) .

[195] DeepLearnPhysics Collaboration collaboration, *Scalable deep convolutional neural networks for sparse, locally dense liquid argon time projection chamber data*, *Phys. Rev. D* **102** (2020) 012005.

[196] J. Lazar, S. Meighen-Berger, C. Haack, D. Kim, S. Giner and C.A. Argüelles, *Prometheus: An Open-Source Neutrino Telescope Simulation*, 2304.14526.

[197] D. Chirkin, "PPC standalone code." https://icecube.wisc.edu/~dima/work/WISC/ppc/, 2020.

[198] M.J. Larson, *Simulation and identification of non-Poissonian noise triggers in the IceCube neutrino detector*, Ph.D. thesis, The University of Alabama, 2013.

[199] IceCube collaboration, *Energy Reconstruction Methods in the IceCube Neutrino Telescope*, *JINST* **9** (2014) P03009 [1311.4767].

[200] C. Choy, J. Gwak and S. Savarese, *4d spatio-temporal convnets: Minkowski convolutional neural networks*, in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE Computer Society, jun, 2019, DOI.

[201] IceCube collaboration, *An improved muon track reconstruction for IceCube*, *PoS* **ICRC2019** (2021) 846 [1908.07961].

[202] G. Apollinari, I. Béjar Alonso, O. Brüning, M. Lamont and L. Rossi, eds., *High-Luminosity Large Hadron Collider (HL-LHC) : Preliminary Design Report*, .

[203] G. Gatti, D. Huerga, E. Solano and M. Sanz, *Random access codes via quantum contextual redundancy*, *Quantum* **7** (2023) 895 [2103.01204].