

## Table of Contents

I

### Theory and Experiment

II

### Simulation, Processing and Analysis

<b>1</b>	<b>Simulation: Event Generation and Propagation</b>	<b>7</b>
<b>1.1</b>	<b>Generation</b>	<b>9</b>
1.1.1	Background simulation	9
1.1.2	Signal simulation	11
<b>1.2</b>	<b>Propagation</b>	<b>13</b>
1.2.1	PROPOSAL	13
1.2.2	Photoelectron generators	14
<b>1.3</b>	<b>Detector simulation</b>	<b>15</b>
<b>1.4</b>	<b>Burn sample</b>	<b>16</b>
1.4.1	Standard ??	16
1.4.2	Event viewer	16
<b>2</b>	<b>Reconstruction, Cleaning and Analysis Techniques</b>	<b>17</b>
<b>2.1</b>	<b>Reconstruction</b>	<b>17</b>
2.1.1	Likelihood	17
2.1.2	Line-Fit	18
2.1.3	SPE and MPE	19
2.1.4	Millipede	20
<b>2.2</b>	<b>Pulse cleaning</b>	<b>20</b>
<b>2.3</b>	<b>IceHive</b>	<b>21</b>
2.3.1	HiveSplitter	22
2.3.2	HiveCleaning	23
<b>2.4</b>	<b>CoincSuite</b>	<b>23</b>
<b>2.5</b>	<b>Analysis techniques</b>	<b>24</b>

<b>2.6</b>	<b>Boosted Decision Tree Classifiers</b>	<b>24</b>
<b>3</b>	<b>The SPACE Analysis</b>	<b>25</b>
3.1	Motivation	25
3.2	General strategy	25
3.3	Event cleaning	25
3.4	Variables	25
3.5	BDT results	25
3.6	Pull validation	25
3.7	Systematic Uncertainties	25
3.8	Results	25
<b>4</b>	<b>Summary and Discussion</b>	<b>27</b>
	<b>Appendices</b>	<b>29</b>
<b>A</b>	<b>Gauge symmetries</b>	<b>31</b>
<b>B</b>	<b>Planck's law</b>	<b>33</b>
<b>B.1</b>	<b>Electromagnetic waves in a cubical cavity</b>	<b>33</b>
B.1.1	Classical approach	34
B.1.2	Quantum approach	34
<b>C</b>	<b>Statistics</b>	<b>35</b>
<b>D</b>	<b>Distributions</b>	<b>37</b>
<b>D.1</b>	<b>Spherical random numbers</b>	<b>37</b>
<b>D.2</b>	<b>Power law distributions</b>	<b>38</b>
<b>D.3</b>	<b>Angular distributions</b>	<b>39</b>
<b>D.4</b>	<b>Weighting</b>	<b>40</b>
<b>5</b>	<b>Some useful things for LaTeX</b>	<b>41</b>
<b>5.1</b>	<b>Definitions</b>	<b>41</b>
<b>5.2</b>	<b>Remarks</b>	<b>41</b>
<b>5.3</b>	<b>Corollaries</b>	<b>41</b>
<b>5.4</b>	<b>Propositions</b>	<b>41</b>
5.4.1	Several equations	42
5.4.2	Single Line	42
<b>5.5</b>	<b>Examples</b>	<b>42</b>
5.5.1	Equation and Text	42
5.5.2	Paragraph of Text	42
<b>5.6</b>	<b>Exercises</b>	<b>42</b>
<b>5.7</b>	<b>Problems</b>	<b>42</b>
<b>5.8</b>	<b>Vocabulary</b>	<b>42</b>
	<b>Bibliography</b>	<b>43</b>
	<b>Index</b>	<b>45</b>





# Simulation, Processing and Analysis

<b>1</b>	<b>Simulation: Event Generation and Propagation .....</b>	<b>7</b>
1.1	Generation	
1.2	Propagation	
1.3	Detector simulation	
1.4	Burn sample	
<b>2</b>	<b>Reconstruction, Cleaning and Analysis Techniques .....</b>	<b>17</b>
2.1	Reconstruction	
2.2	Pulse cleaning	
2.3	IceHive	
2.4	CoincSuite	
2.5	Analysis techniques	
2.6	Boosted Decision Tree Classifiers	
<b>3</b>	<b>The SPACE Analysis .....</b>	<b>25</b>
3.1	Motivation	
3.2	General strategy	
3.3	Event cleaning	
3.4	Variables	
3.5	BDT results	
3.6	Pull validation	
3.7	Systematic Uncertainties	
3.8	Results	
<b>4</b>	<b>Summary and Discussion .....</b>	<b>27</b>
	<b>Appendices .....</b>	<b>29</b>
<b>A</b>	<b>Gauge symmetries .....</b>	<b>31</b>
<b>B</b>	<b>Planck's law .....</b>	<b>33</b>
B.1	Electromagnetic waves in a cubical cavity	
<b>C</b>	<b>Statistics .....</b>	<b>35</b>
<b>D</b>	<b>Distributions .....</b>	<b>37</b>
D.1	Spherical random numbers	
D.2	Power law distributions	
D.3	Angular distributions	
D.4	Weighting	
<b>5</b>	<b>Some useful things for LaTeX .....</b>	<b>41</b>
5.1	Definitions	
5.2	Remarks	
5.3	Corollaries	
5.4	Propositions	
5.5	Examples	
5.6	Exercises	
5.7	Problems	
5.8	Vocabulary	
	<b>Bibliography .....</b>	<b>43</b>
	<b>Index .....</b>	<b>45</b>





## 1. Simulation: Event Generation and Propagation

*Soon there will be virtual reality, and augmented reality. If you assume any rate of improvement at all, then games will become indistinguishable from reality . . . , it would seem to follow that the odds we are in base reality are one in billions. ~ Elon Musk*

In order to be able to search for new physics, one has to have a good handle on the detector response on known physics events. Depending on the analysis, some processes are more interesting than others. In general, the particle interactions of interest are referred to as *signal events*. Other interactions, which mimic or obscure the signal events, are typically called *background events*. These events are simulated using Monte Carlo\* (MC) simulations, where one makes use of a model that describes the interactions and their probability to occur. A typical MC simulation consists of hundreds to millions of events that are constructed using these models from random number generators. To determine the detector response of a particle interaction, one first has to start with the particle generation, which defines the properties of the interaction. Afterwards, the propagation of the particle in the detector (medium) is simulated as best as possible. Below, we give an overview of the important background and signal simulations that are used in this analysis. A flowchart of the simulations steps is given in Fig. 1.1.

**Corollary 1.0.1 — The software framework.** *IceTray* is a modular framework written and used by the IceCube collaboration and mostly written in C++ for fast computation. A python interface for most modules is provided for fast and easy implementation of the code. The framework is used in both online and offline processing. The framework is stream-based with modules that act on events in the stream and is essentially follows the flowchart that is provided by the user.

To process the large amount of simulation that is required for the collaboration, a data processing and management framework called *IceProd* was developed. The setup is very light-weight, running as a python application. It uses (complex) workflow DAGs (see below) across distributed computing grids in order to optimize usage of resources. A *dataset* is set

\*While recovering from an illness in 1946, Stanislaw Ulam figured that the actual counting of successful attempts in playing a card game would yield him a much faster answer to the probability of success rather than doing the actual calculus. His work, shared with John von Neumann, needed to remain secret and adopted the code word “Monte Carlo”, referring the gambling games in the Monte Carlo Casino in Monaco.

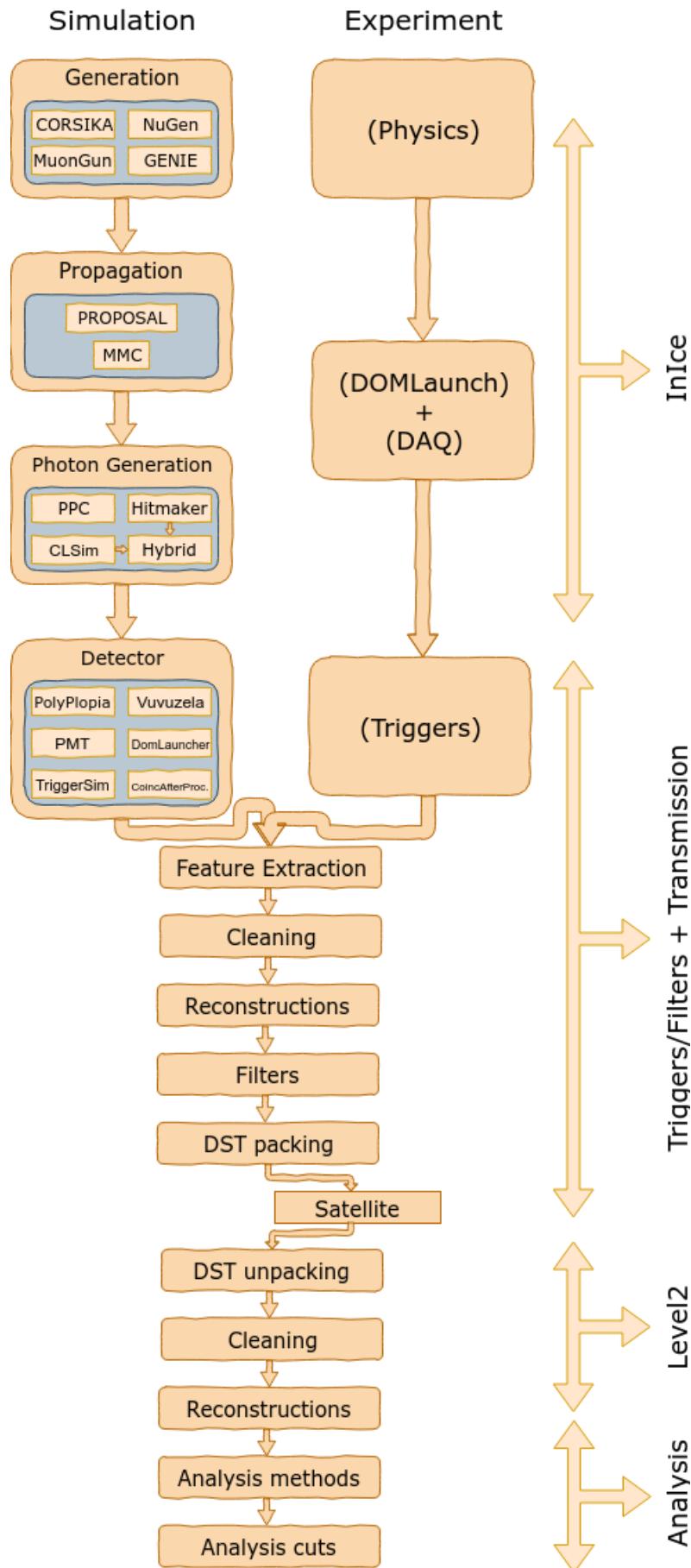


Figure 1.1: Flowchart of the simulation layout. On the left it is shown that particles are injected and their interactions are simulated to final digitized waveforms. The right shows real data processing. After triggering, data and simulation go through the same processing chain to prepare for analysis.

up by running hunders to thousands of jobs in parallel over multiple computing resources all over the world. Each dataset has specific input parameters that are fixed. Distributions in physical parameters such as the direction, energy, position, etc. of the particle(s) are privided by random number generators [1].

*HTCondor* is an open source computing software that provides a job queueing mechanism, scheduling policy, priority scheme, resource monitoring, and resource management. Users submit their serial or parallel jobs to HTCondor, HTCondor places them into a queue, chooses when and where to run the jobs based upon a policy, carefully monitors their progress, and ultimately informs the user upon completion.

*DAGMans* (Directed Acyclic Graph Managers) are meta-schudulers for the execution of computations. They submit the programs to HTCondor in an order that is represented by a DAG and processes the results. DAGMans are often used by analyzers for bulk computations on large amounts of data.

## 1.1 Generation

Simulations start with setting up the starting conditions of the physical processes one wants to simulate. For example, a shower event by itself is not well defined. The type of primary particle (H, He, Fe,...), the energy, the inclination and so on will all define the properties of the air shower that will be produced. Multiple different generators used in the IceCube collaboration serve other purposes; some are explained in more detail below.

### 1.1.1 Background simulation

#### 1.1.1.1 CORSIKA

A free, publicly available software framework that is widely used in the astrophysics community for the simulation of cosmic ray interactions is called CORSIKA (COsmic Ray SImulations for Kascade). It was originally developed for the KASCADE experiment and now used by most people and collaborations to simulate air shower events. In-ice analyses, such as this one, use CORSIKA simulations to simulate the muonic component that is able to reach the in-ice detector.

A particle of specific type, energy, direction and position is injected in the top of the atmosphere and propagated. The distribution of particles in the shower is saved and read out at a certain altitude. Because the flux of cosmic rays is exceedingly small at the highest energies, too many resources and too much time would be required to simulate an energy distribution as measured in experiments. Therefore, one often simulates a much harder spectrum and reweights the events accordingly later on (see Section D.4). Simulation datasets are often subdivided into a low-energy and high-energy dataset. The former ranges from primary energies between 600 GeV to 100 TeV and uses a spectral index that is close to what is measured. The spectral index of the latter is smaller, resulting in a harder spectrum, and the primary energy ranges from 100 TeV to 100 EeV. The lower limit of the energy range is due to the limited penetration depth of muons through the ice. An overview is given in Table 1.2.

The spectrum used for this analysis, after reweighting, follows the following energy distribution:

$$\Phi_i(E_{\text{prim}}) = \sum_{j=1}^3 a_{i,j} E^{-\gamma_{i,j}} \cdot \exp\left[-\frac{E}{Z_i R_{c,j}}\right]. \quad (1.1)$$

where we sum over the three populations that are mentioned in Section ??,  $\gamma$  is the spectral index,  $Z$  the particle atomic number and  $a_{i,j}$  the normalization constants for primary  $i$  in population  $j$ . The 5 groups that are assumed to contribute significantly to the flux are: p, He, CNO, Mg-Si and Fe. This is the convention that is used in Ref. [2]. Table 1.1 shows the best fits for the normalization constants to describe the data.

Table 1.1: Best fit for parameters in Eq. 1.1. Numbers taken from Ref. [2]

$j$	$R_c$ [V]	$\gamma$					$a_{i,j}$				
		p	He	CNO	Mg-Si	Fe	p	He	CNO	Mg-Si	Fe
1	$4 \cdot 10^{15}$	1.66	1.58	1.63	1.67	1.63	7860	3550	2200	1430	2120
2	$30 \cdot 10^{15}$			1.4			20			13.4	
3	$2 \cdot 10^{18}$			1.4			1.7			1.14	

## Interactions

The atmosphere composition is always set at 78.1% N<sub>2</sub>, 21% O<sub>2</sub>, and 0.9% Ar, which is a good description of reality. However, the density of the air above the detector changes significantly during the year because of temperature differences in the Arctic Summer and Winter. Most analyses that treat the muonic component as a background and are not interested in the details of the showers and how it's changes during the year use an average of the atmospheric density.

The shower propagation and composition depends on the models that are used to simulate these high-energy interactions. The lowest energies are simulated with FLUKA (FLUktuierende KAskade) [3]. This model covers the energy range that can be compared with accelerator experiments. Which model is the best for the highest energies is not known at the time of writing, if there even is one, since there are no controlled laboratory measurements that are capable of reaching these energies. Several studies seem to indicate that the composition changes drastically at the highest energies [**SAMCITEREN+andere**]. Fortunately, this is of no large importance for this analysis.

CORSIKA is written in FORTRAN 77, but a C++ version is currently in the making [4].

### 1.1.1.2 NuGen

The neutrino-generator is a neutrino event generator program that works with the IceTray framework (see Section ???). With this module, one can inject a primary neutrino on the surface of the Earth by specifying a few parameters in the steering file.

The physics implemented in this program is based on the ANIS-All Neutrino Interaction Generator [5]. However, the cross sections have been updated and the structure of code has been changed significantly from ANIS to incorporate it in the IceTray framework.

The generator requires the first interaction to be near the detector and

- prepares a primary neutrino and injects it to the Earth,
- propagates the neutrino and makes interactions inside the Earth (if it happens),
- makes a forced interaction inside the detection volume (only if any neutrino reaches the detector site),
- stores injected neutrinos and all generated secondaries
- stores interaction weight information

Since the generators forces the interaction to occur (to optimize computational resources), the interaction weight has to take this into account.

The generator also does not distinguish between neutrino and antineutrino and assumes a ratio of (1:1).

The spectrum used for this analysis, after reweighting, follows the Honda2006 spectrum [6] for atmospheric neutrinos, SarcevicStd for the prompt component [7], and astrophysical from [8]. The astrophysical flux measured by the IceCube collaboration follows the following energy spectrum

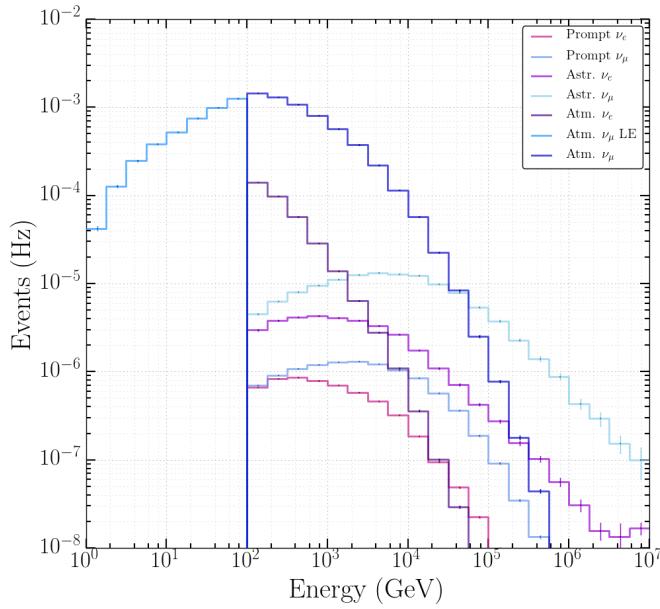


Figure 1.2: Distribution of weighted neutrino fluxes that were used for this analysis. The atmospheric  $\nu_\mu$  and  $\nu_e$  fluxes were derived from Ref. [6], prompt from Ref. [7], and astrophysical from Ref. [8].

$$E^2(\Phi) = 1.5 \cdot 10^{-8} \left( \frac{E}{100 \text{ TeV}} \right)^{-0.3} \text{ GeV cm}^{-2}\text{s}^{-1}\text{sr}^{-1} \quad (1.2)$$

The distribution for these different components can be seen in Fig. ??.

### 1.1.1.3 GENIE

To include the lowest energies, which are not accounted for by ANIS/NuGen, the GENIE (Generates Events for Neutrino Interaction Experiments) neutrino generator was implemented in IceTray. It is a well established generator, used by collaborations worldwide and written in C++ [9, 10].

The spectrum used for this analysis, after reweighting, follows the Honda2015 spectrum [Honda:2015fna] for low-energy atmospheric neutrinos.

## 1.1.2 Signal simulation

As mentioned in Section ??, the signal is assumed to be isotropic close to the detector. The SMP starting points are randomly placed on a disk with a direction perpendicular to it as shown in Fig. 1.3 below. The disk has a radius of 800 m and is located at a distance of 1000 m from the detector center. The disk itself is randomly rotated around the detector center to simulate an isotropic flux. The distribution of the azimuth,  $\phi$  and cosine of the zenith\*,  $\cos \theta$ , is shown in Fig. 1.4.

Because slow moving particles would require specific treatment, the minimal velocity of the particles is set as  $\beta > 0.95$  and simulated with a  $E^{-1}$  spectrum. The spectrum is later normalized a flux of  $10^{-14} \text{ GeV cm}^{-2}\text{s}^{-1}\text{sr}^{-1}$  with an  $E^{-2}$  spectrum. The absolute flux is only necessary for illustrative purposes, see Section ???.

Similar to the background, SpiceLea was used as the nominal ice model.

---

\*See Appendix D.3 why we show the cosine of the zenith.

Table 1.2: Overview of the datasets used in this analysis. GaisserH3a from Ref. [2], Honda2015 from Ref. [Honda:2015fna], Honda2006 from Ref. [6], Sarcevic from Ref. [7], and astrophysical from Ref. [8]

Generator	Type	Range [GeV]	Simulated $\gamma$	Weighted $\gamma$	Ice	Dataset
CORS.	HE 5-comp.	$10^5 - 10^{11}$	2	GaisserH3a	SpiceLea	11937
CORS.	LE 5-comp.	$600 - 10^5$	2.6	GaisserH3a	SpiceLea	11499
CORS.	LE 5-comp.	$600 - 10^5$	2.6	GaisserH3a	SpiceLea	11808
CORS.	LE 5-comp.	$600 - 10^5$	2.6	GaisserH3a	SpiceLea	11865
CORS.	LE 5-comp.	$600 - 10^5$	2.6	GaisserH3a	SpiceLea	11905
CORS.	LE 5-comp.	$600 - 10^5$	2.6	GaisserH3a	SpiceLea	11926
CORS.	LE 5-comp.	$600 - 10^5$	2.6	GaisserH3a	SpiceLea	11943
CORS.	LE 5-comp.	$600 - 10^5$	2.6	GaisserH3a	SpiceLea	12161
CORS.	LE 5-comp.	$600 - 10^5$	2.6	GaisserH3a	SpiceLea	12268
GENIE	$\nu_\mu$	$0.5 - 100$	1	Honda2015	SpiceMie	12475
NuGen	$\nu_\mu$	$100 - 10^8$	2	atmos.: Honda2006 prompt: Sarcevic astro.: Astro.	SpiceLea	11029
NuGen	$\nu_\mu$	$100 - 10^8$	2	atmos.: Honda2006 prompt: Sarcevic astro.: Astro.	SpiceLea	12346
NuGen	$\nu_\mu$	$100 - 10^8$	2	atmos.: Honda2006 prompt: Sarcevic astro.: Astro.	SpiceLea	11883
NuGen	$\nu_e$	$100 - 10^8$	2	atmos.: Honda2006 prompt: Sarcevic astro.: Astro.	SpiceLea	12034
NuGen	$\nu_e$	$100 - 10^8$	2	atmos.: Honda2006 prompt: Sarcevic astro.: Astro.	SpiceLea	12646

Table 1.3: Overview of the datasets used for systematic uncertainties. Polyg(onato) follows from Ref. ???, GaisserH4a from Ref. ??? and Bartol from Ref. ???.

Generator	Type	Range [GeV]	Sim. $\gamma$	Weighted $\gamma$	Ice	Dataset	Syst. Eff.
CORS.	Hoerandel	$600 - 10^5$	Polyg.	GaisserH3a	SpiceLea	11527	DOM eff. -10%
CORS.	Hoerandel	$600 - 10^{11}$	Polyg.	GaisserH3a	SpiceLea	11526	DOM eff. +10%
CORS.	LE 5-comp.	$600 - 10^{11}$	2.6	GaisserH3a	SpiceLea	12388	Abs. +10%
							Scat. +10%
							Abs./Scat. -7.1%
CORS.	All datasets from Table 1.2			GaisserH4a	SpiceLea	Table 1.2	GaisserH4a
NuGen	$\nu_\mu$	$100 - 10^8$	2	Bartol (syst.)	SpiceLea	11029	Bartol flux
NuGen	$\nu_\mu$	$100 - 10^7$	2	Honda2006 + Bartol (syst.)	SpiceLea	11883	DOM eff. +10% DOM eff. -10% Abs. +10% Scat. +10% Abs./Scat. -7.1% Bartol flux
NuGen	$\nu_\mu$	$100 - 10^8$	2	Honda2006 + Bartol (syst.)	SpiceLea	12346	DOM eff. +10% DOM eff. -10% Abs./Scat. -7.1% Bartol flux

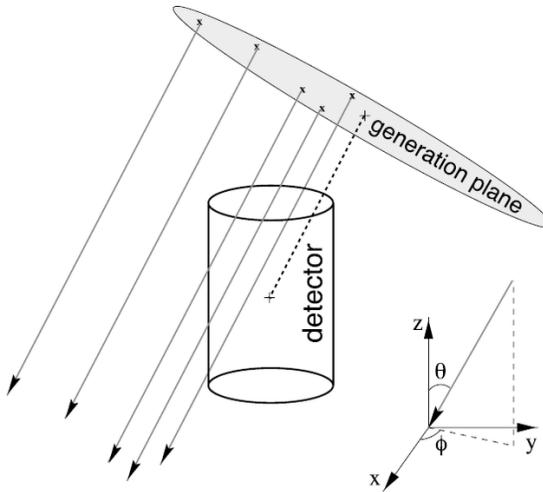


Figure 1.3: Illustration of how the particle injection works. The particle is first randomly positioned on a disk following a uniform distribution. The disk is then randomly rotated to simulate an isotropic flux.

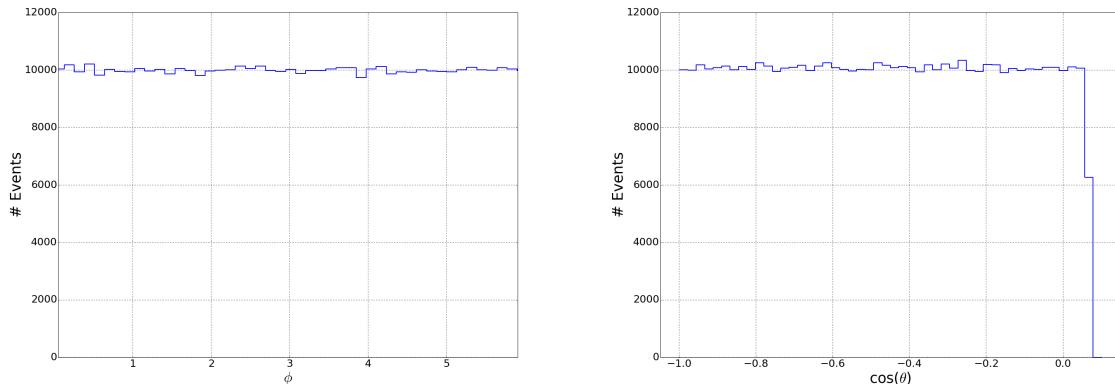


Figure 1.4: Illustration of uniform distributions of azimuth and cosine of the zenith for the particle injection in agreement with an isotropic flux (see Appendix ??).

## 1.2 Propagation

After generation, the particles need to be propagated through the medium. The particles will interact, lose energy, produce new particles, and generate light. The particle interactions and light production are done in two different modules as photon simulation is done with GPU. The former module is called *PROPOSAL*, the latter *ppc*.

### 1.2.1 PROPOSAL

Using the cross sections of the important interaction, together with the properties of the traversing medium and the particles (mass, charge, spin, decay time, etc.) it is possible to simulate the energy losses, secondary production and the consequent interaction of these daughter particles. This is done in the software package PROPOSAL (the Propagator with Optimal Precision and Optimized Speed for All Leptons), fully written in C++. It was based on the former program MMC (Muon Monte Carlo), which was written in Java. This year, a substantial improved version of PROPOSAL was finalized. An illustration of the workings of the code is given in Fig. 1.5 and an in depth documentation is given in Ref. [11].

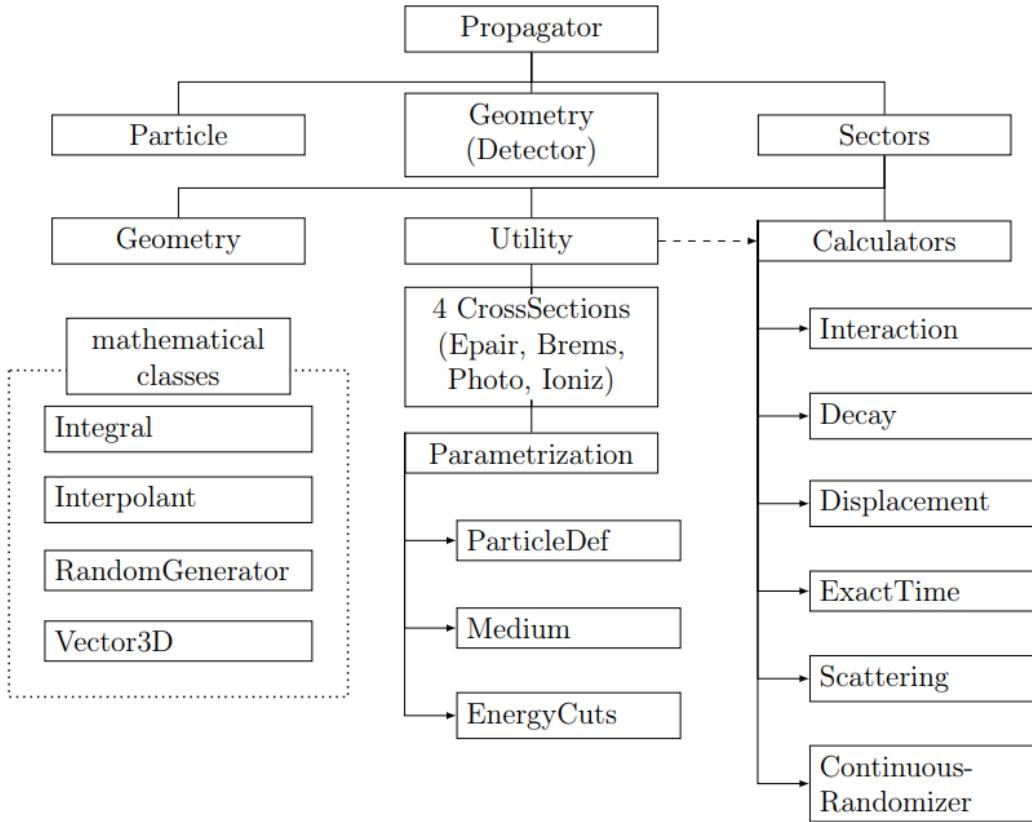


Figure 1.5: Overview of the class structure in PROPOSAL, from Ref. [11].

### PROPOSAL for SMPs

Since we assume the SMPs to behave leptonically, it was chosen to use PROPOSAL for the signal propagation as well. The mass and charge of the particle are set in the input parameters and the cross section dependence on these parameters can be seen in Section ???. In general, there is a small dependence on the mass and a squared dependency on the charge (Eq. ???), except for bremsstrahlung that has a quadratic charge dependency. These effects only become prominent and important for highly relativistic particles, which as will be seen in Section ???, do not have a dominating contribution to the total signal.

The PROPOSAL module keeps track of all the particles that are produced during propagation and the accompanying energy losses in a tree-like structure (called an *I3MCTree*). This collection of particles and their interactions are forwarded to a light production computation module.

#### 1.2.2 Photoelectron generators

In Section ?? we already explained how the ice is simulated in the IceCube detector. The parameters  $b_e(400)$  and  $a_{dust}(400)$  define the photon propagation through the ice and determine if they are absorbed or hit a DOM. To optimze computing time the DOMs were scaled (mostly with a factor of 5) to force more photon interactions. The number of photons emitted was then appropriately scaled down with the square of this scaling factor\*. The DOM acceptance curves, as shown in Fig. ??, together with the Frank-Tamm formula (Eq. ??) allow to calculate the expected number of photons produced per unit length:

$$\frac{dN}{dx} = \int_{\lambda_1}^{\lambda_2} \frac{2\pi\alpha}{\lambda^2} \sin^2(\theta_c) d\lambda = 2\pi\alpha \sin^2(\theta_c) \left( \frac{1}{\lambda_1} - \frac{1}{\lambda_2} \right). \quad (1.3)$$

\*The surface of a sphere scales with the square of the radius.

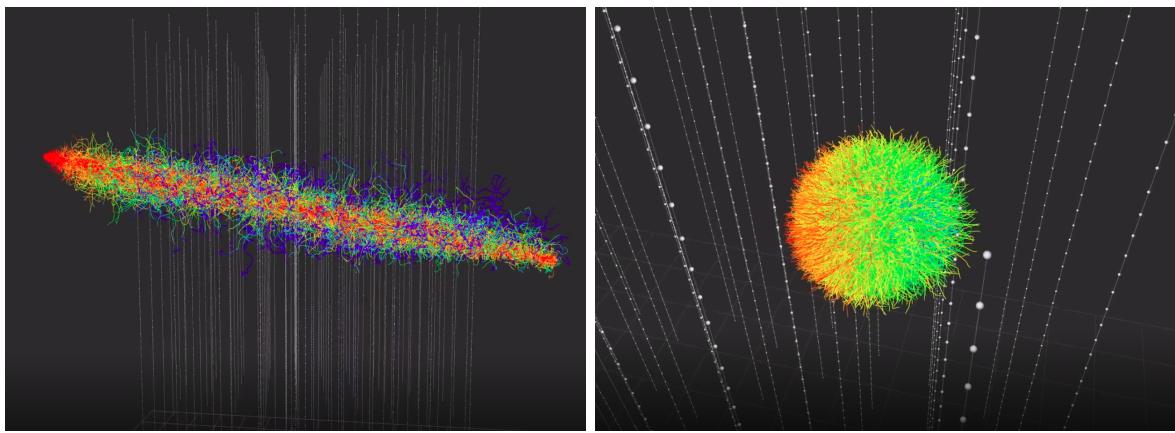


Figure 1.6: *Left:* simulation of a track event in IceCube. Each line represents a photon path and colors indicate how far they have traveled from their generation point. *Right:* simulation of a cascade event in IceCube.

From this formula, we find that the expected rate of a Cherenkov emission profile is equal to  $\approx 350$  photons/cm. Together with the DOM acceptance, which has an overall average of around 7%, the expected *seen* number of photons per meter is equal to  $2450 \text{ m}^{-1}$ .

PPC is a Photon Propagation Code, written in C++ and runs on graphic processing units (GPUs). This allows the code to run up to a hundred times faster than in a CPU-only environment. PPC employs both CUDA (NVIDIA GPU only) and OpenCL programming interface (both NVIDIA and AMD GPUs) together with multiple CPU environments. GPU environments allow the tracking of thousands of photons simultaneously, vastly improving the computational speed. For more information, see Ref. [12].

Previous photon propagation code, such as *Photonics* [13], produced 6-dimensional photon tables (3 spatial, 2 directional and 1 temporal). This meant that at least one set of tables had to be produced per particle type and per velocity and interpolation methods had to be used, with the accompanying inaccuracies. These tables also required significant disk space and the method was therefore replaced with the GPU-codes. Direct photon simulation also allows for other non-trivial implementations such as the tilting of ice layers.

Another photon propagation code is called *CLSim*, which uses GEANT4 to propagate particles. A hybrid version called *HybridCLsim* is sometimes where muons are propagated using PROPOSAL/MMC and their stochastic losses (which are showers) are simulated from tables whereas the “bare muons” (with their stochastics) are simulated using direct propagation. This avoids time loss for the rare but very computational high-energy cascade events.

An illustration of photon propagation in the IceCube detector for both a cascade and track simulation is shown in Fig. 1.6.

### 1.3 Detector simulation

Further processing of the simulations involve

- Polyplopia: a project dedicated to merge multiple events to account for coincident events (that are simulated independently). An estimated 15% of CORSIKA events result in coincident events and make up de bulk of bad reconstructions where down-going muons are simulated as up-going;
- Vuvuzela: the PMT noise is simulated as having an exponential component from thermal and radioactive decays, and a log-normal contribution for scintillation;
- PMT: the time from the first photon entering the PMT to the readout after passing along multiple dynodes has an uncertainty, referred to as “PMT jitter”. The amplification of

Figure 1.7

photoelectrons by the PMT is also not constant and is simulated in this module. The module also accounts for prepulses, late pulses, afterpulses and saturation of the PMT. More information can be found in Refs. [14, 15].

- DOMLauncher: the digitization of the PMT pulses and other behavior of the DOM mainboard (as explained in Section ??) is done in this module. The three main features of the DOM that are simulated to generate launches are the discriminator, LC, and digitization.
- trigger-sim: simulation of the trigger behavior as explained in Sec. ??.

## 1.4 Burn sample

Getting the intricate details of nature just right is non-trivial. In many steps of the way, simulations use fits and estimations. Some simulation datasets are reasonable to compare to the data, depending on the phase space one is looking at, while other datasets need other specifications. For example, analyses dedicated to measuring the cosmic ray interactions need much more fine-tuning in their models for the atmosphere, composition, interaction models, etc. than an analysis dedicated to search for muon tracks that first propagated through the Earth and have atmospheric muons as a background.

It is for this reason, most analyses select a certain subset of the data they want to analyze to compare to the Monte Carlo simulations. For this analysis, it was chosen to look at 10% of the total data, called the *burn sample*. As indicated in Section [subsec:datahandling], the data is saved in 8-hour runs and the burn sample consists of every run ending with a '0'. The burn sample also allows to estimate the robustness of certain reconstructions and variables regarding differences in data and simulation.

More info on the burnsample see Section???

### 1.4.1 Standard ??

Processing ergens: online L0, L1, L2, uw filters,... Beter

### 1.4.2 Event viewer

After a full simulation, it is possible to visualize the event in an event viewer called *Steamshovel*. Typical events in the IceCube detector are shown with this interface and are loaded from i3files that contain information about the detector geometry and the full event (DOM positions and calibrations, detector hits, timestamps, trigger hits, etc.). Simulated events also contain the true values of the particles and can be compared with reconstructed variables. Event viewers allow for first guesses in how background events are able to be separated from signal, although both can have wide varieties in possible outputs.

An example is given in Fig. 1.7.

## 2. Reconstruction, Cleaning and Analysis Techniques

*Shall I refuse my dinner because I do not fully understand the process of digestion? ~ Oliver Heaviside*

Because the in-ice IceCube detector is a sparsely distributed detector, it is not straightforward to unambiguously reconstruct the particle (interactions). The scattering and absorption of photons, tilt of ice sheets, bubble column, etc. lead to uncertainties and make reconstruction challenging. Over the years, multiple reconstruction methods have been developed in the collaboration. They range from very fast (and simple) reconstructions, necessary for online filtering, to slow (and more refined) ones. Multiple reconstruction algorithms have been used in this analysis and are explained in more detail in this chapter.

### 2.1 Reconstruction

#### 2.1.1 Likelihood

Reconstruction algorithms usually have no unique solutions to describe the set of measured values of an event. The likelihood  $\mathcal{L}(\vec{x}|\vec{a})$  describes the probability of a set of parameters  $\vec{a}$  and a set of experimentally measured values  $\vec{x}$ . The parameters,  $\vec{a}$ , typically define the particle's characteristics (energy, direction, position, type, etc.) while the measured values  $\vec{x}$  are determined from the detector response (number of PE, timing, position of hit DOMs, etc.). This likelihood is equal to the cumulative probability

$$\mathcal{L}(\vec{x}|\vec{a}) = \prod_i p(x_i|\vec{a}), \quad (2.1)$$

where  $p(x, \vec{a})$  is the probability that we measure a certain value  $x$  from a set of independent values  $\vec{x}$  given an initial set of parameters  $\vec{a}$ . The best possible guess for the unknown parameters  $\vec{a}$  is the most likely set that will result into the experimental values. This is done by maximizing the likelihood  $\mathcal{L}$ . The reconstruction algorithms below rely on following parameters that assume a single, long track

$$\vec{a} = (\vec{r}_0, t_0, \vec{p}, E_0), \quad (2.2)$$

where  $r_0$  is the position vector of the particle at a time  $t_0$  with a direction  $\vec{p}$  and initial energy  $E_0$ .

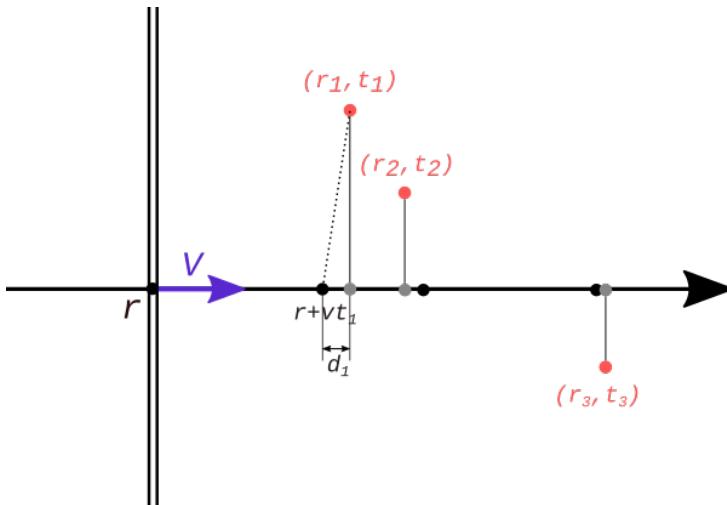


Figure 2.1: Figure illustrating how LF works. The position,  $\vec{r}$ , and velocity,  $\vec{v}_{\text{part}}$  minimizing the distance of the DOMs to the track is calculated. The dotted line is one distance that is minimized in Eq. 2.4.

### 2.1.2 Line-Fit

Dit niet voor IceHive? Reconstr. wordt daar ook geïntroduceerd... Toch in Coinsuite

One of the most simple approaches in constructing a parameter profile is by calculating the track that, overall, has the closest approach of all the hit doms and is called **Line-Fit** (LF) [Ahrens:2003fg]. If we assume that a particle starts at a position,  $\vec{r}$  at a time 0 and travels at a velocity of  $\vec{v}_{\text{part}}$ , then its position at any given time is

$$\vec{r}' = \vec{r} + \vec{v}_{\text{part}}t. \quad (2.3)$$

We want to calculate the best possible estimate of the velocity  $\vec{v}_{\text{part}}$  and the position  $\vec{r}$ . Each DOM has a known location,  $\vec{r}_i$ , and measured time of a pulse,  $t_i$ . In this algorithm one assumes that a wavefront perpendicular to the particles direction is traveling along with the particle. If the velocity  $\vec{v}_{\text{part}}$  is fixed, then the position of the particle is known (black points in Fig. 2.1). However, because the Cherenkov wavefront should be set at an angle and scattering, PMT jitter, noise, etc. are not taken into account, this will not agree with the DOM position projected along the particle path (grey dots). The unknown velocity  $\vec{v}_{\text{part}}$  and position  $\vec{r}$  are the analytical solutions after minimizing the distances  $d_i$  as shown in the figure\*

$$\begin{aligned} S(\vec{r}, \vec{v}_{\text{part}}) &= \sum_{i=1}^{N_{\text{hit}}} \rho(\vec{r}, \vec{v}_{\text{part}}, \vec{r}_i, t_i)^2 \\ &\equiv \sum_{i=1}^{N_{\text{hit}}} (\vec{r}_i - \vec{r} - \vec{v}_{\text{part}}t_i)^2, \end{aligned} \quad (2.4)$$

where  $N_{\text{hit}}$  are the number of hits. The analytical solution by minimizing this equation is equal to

$$\vec{r} = \langle \vec{r}_i \rangle - \vec{v}_{\text{part}} \quad \text{and} \quad \vec{v}_{\text{part}} = \frac{\langle \vec{r}_i t_i \rangle - \langle \vec{r}_i \rangle \langle t_i \rangle}{\langle t_i^2 \rangle - \langle t_i \rangle^2}, \quad (2.5)$$

where  $\langle x \rangle$  denotes the average of a parameter  $x$  over all hits  $i$ . Because this is an analytical equation, this algorithm is very fast and therefore often used in online processing.

---

\*Minimizing  $r_i - r'$  (dotted line in Fig. 2.1) is the same as minimizing  $d$ .

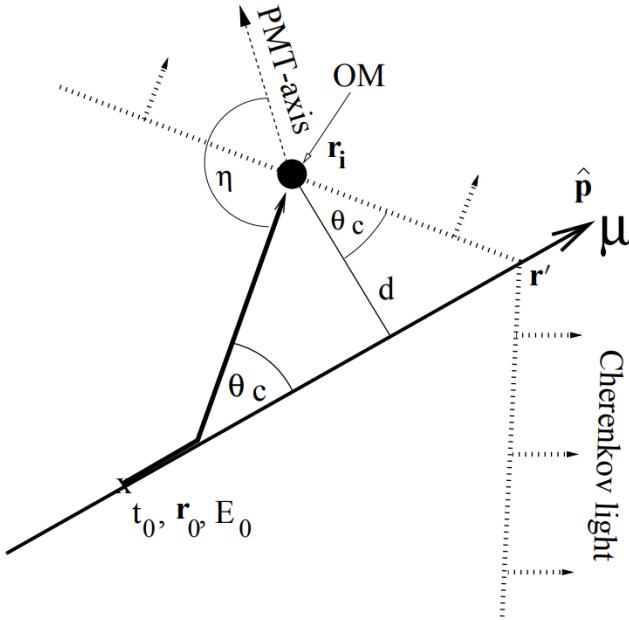


Figure 2.2: Figure illustrating a muon track passing close by an optical module and defining the parameters used in the reconstruction algorithms.

### 2.1.2.1 Improved Line-Fit

While disregarding the Cherenkov profile is inherent to the simplified LF model chosen for computational reasons, removing hits generated by photons that scattered for a significant length of time will mitigate the effect of ignoring the photon scattering in the ice. It was found that a basic filter could identify these scattered hits, and improve accuracy by almost a factor of two by removing them from the dataset. More formally, for each hit  $h_i$ , the algorithm looks at all neighboring hits within a neighborhood of  $\mu$ , and if there exists a neighboring hit  $h_j$  with a time stamp that is  $t$  earlier than  $h_i$ , then  $h_i$  is considered a scattered hit, and is not used in the basic reconstruction algorithm. Optimal values of  $\mu$  and  $t$  were found to be 156 m and 778 ns by tuning them on simulated muon data [Aartsen:2013bfa].

This “delay cleaning” is done by computing the Huberfit on the remaining data points and minimizing

$$\sum_{i=1}^{N_{\text{hit}}} \phi(\rho(\vec{r}, \vec{v}_{\text{part}}, \vec{r}_i, t_i)), \quad (2.6)$$

where  $\rho$  was defined in Eq. 2.4 and the Huber penalty function  $\phi$  is defined as

$$\phi(\rho) \equiv \begin{cases} \rho^2 & \text{if } \rho < \mu \\ \mu(2\rho - \mu) & \text{if } \rho \leq \mu \end{cases}. \quad (2.7)$$

Because of the overall performance increase of this method, all LF computations were done with the improved version (although still often referred to as “Line-Fit”).

### 2.1.3 SPE and MPE

A more intricate method of track reconstruction is done by taking the geometrical shape of the Cherenkov cone into account and relying on simulation fits where a seed track is implemented (usually from the fast Line-Fit algorithm).

Let us assume a particle is traveling close to a DOM with parameters defined in Eq. 2.2 as illustrated in Fig. 2.2. The minimal distance of the track to the DOM is equal to  $d$  and the PMT-axis (downwards relative to DOM) has an angle offset of  $\eta$  degrees of the Cherenkov wave

direction. In perfect conditions, the *time residual* (time between the observed hit time and the “expected” time) is a delta function, where

$$t_{\text{res}} \equiv t_{\text{hit}} - t_{\text{geo}}, \quad (2.8)$$

with

$$t_{\text{geo}} = t_0 + \frac{\vec{p} \cdot (\vec{r}_i - \vec{r}_0) + d \tan \theta_c}{c_{\text{vac}}}, \quad (2.9)$$

which is equal to the time of the particle to travel from the position  $\vec{r}_0$  to  $\vec{r}'$  as illustrated in the figure. The accompanying Cherenkov wavefront that sent out photons at a time  $t_0$  from  $\vec{r}_0$  will cross the DOM when the particle is at a position  $\vec{r}'$ . Due to noise effects, PMT jitter, light from secondary interactions, DOM orientation, etc. the time residual is smeared and shifted. The p.d.f. was estimated with photon simulations in ice and fitted to a Podel function ???. The time likelihood profile for single photons  $i$  at the locations of the hit DOMs is then

$$\mathcal{L}_{\text{time}} = \sum_{i=1}^{N_{\text{hit}}} p_1(t_{\text{res}} | \vec{a} = d_i, \eta_i, \dots). \quad (2.10)$$

An initial particle position and direction are found by maximizing the likelihood and iterated a couple of times to find the global maximum instead of a local. This fitting is called the Single PhotoElectron (SPE) fit.

The description of single photons arriving at the optical modules cannot be correct since electrical and optical signal channels can only resolve multiple photons separated by a few 100 ns and  $\approx$  10 ns, respectively. In the Multi-PhotoElectron (MPE) fit, one accounts for the fact that the early photons in a DOM hit scattered less in the ice. The p.d.f. for the first photon out of a total of  $N$  to arrive with a time residual of  $t_{\text{res}}$  is

$$p_N^1(t_{\text{res}}) = N \cdot p_1(t_{\text{res}}) \cdot \left( \int_{t_{\text{res}}}^{\infty} p_1(t) dt \right)^{(N-1)} = N \cdot p_1(t_{\text{res}}) \cdot (1 - P_1(t_{\text{res}}))^{(N-1)}, \quad (2.11)$$

where  $P_1$  is the cumulative distribution of the single photon p.d.f..

#### **2.1.4 Millipede**

To have a better handle on the particle energy and cascades along the track, the module **Millipede** was developed. The number of photons seen at each optical module depends on multiple factors (that were mentioned throughout this text, such as the ice characteristics, timing, etc. In this module, the expected number of photons is said to depend on the energy that was deposited and a *light yield factor* that depends on the DOM position and the location of emission along the track

$$N_{\text{exp},k} = \rho_k + \sum_{i=1}^n \Lambda(\vec{r}_k, \vec{r}_i) E_i, \quad (2.12)$$

where  $k$  refers to a certain DOM and  $i$  refers to a certain energy deposit such as illustrated in Fig. ??.

More information can be found in ... and Stef Verpoest his master thesis.

#### **2.1.5 FiniteReco**

#### **2.1.6 Paraboloid**

#### **2.2 Pulse cleaning**

As explained in Section ??, each DOM in IceCube has an intrinsic noise rate. This dark noise is observed in every triggered event and seen as random hits in the detector added to the hit pattern

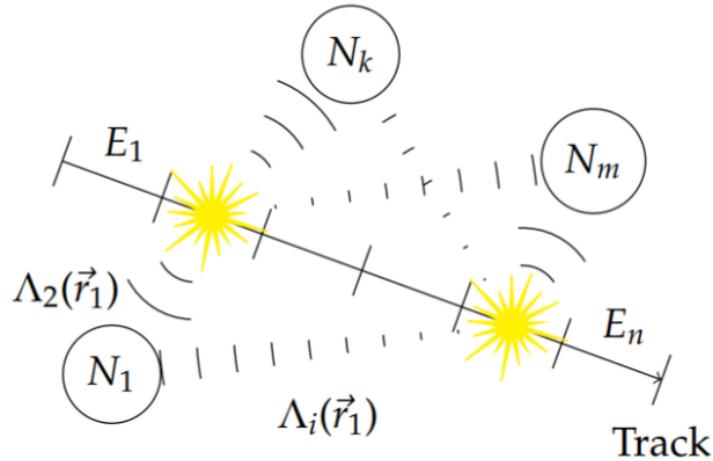


Figure 2.3: Blub

of tracks and cascades. These spurious hits are a large nuisance factor in event reconstructions, leading to misidentification and errors in the result. Noise cleaning should be done in early stages of event processing and analysis to reduce a large rate of bad reconstructed events that pass cut selections. One of the most conservative ways is to only look at HLC hits (*HLC cleaning*), but is too demanding for most low-energetic events that will have multiple hits from isolated DOMs.

Another, more conservative method, is the *seededRT* algorithm. This method relies on the “RT cut”, which was already implemented in AMANDA.  $R$  is a designed radius and  $T$  refers to the time between multiple possible pulse times (e.g. the pulse of one DOM starts during the time window of another DOM’s pulse, or stops during the time window). The full description can be found in Ref. [RTcutwiki], but can be summarized as: DOMs are required to be in a temporal and spacial coincidence that is physically possible (e.g. signal between DOMs cannot exceed the speed of light in vacuum). This method is however computationally expensive since all DOM pairs have to be looped over\*. The seededRT algorithm takes a subset of seeds that are considered to be mostly signal related hits. These seeds can be provided by, for example, using HLC information. By adding all further hits found within the seeds RT-range to the list of seed hits and iterating until a convergence, only those (SLC) hits are kept which cluster around the initial seed hits. Outlying noise hits are supposedly not added and thus removed in the cleaned output. This method does not scale as drastically as the original RT cut method.

### 2.3 IceHive

In Section ??, it was explained how multiple triggers were combined into one global trigger. In a first step, Q-frames are simply re-split into the individual events that belong to the different subtriggers. In about 10% of cases the data read out in one of these P-frames contains more than one primary interaction. This pile-up effect is referred to as *coincident events*. This is a direct result of the traversal time of a couple of microseconds in the detector\*, the large flux of low-energetic events and the trigger time windows of a couple of microseconds. This can be problematic for reconstructions, as can be seen in Fig. ?? where two downgoing muons can be reconstructed as an upgoing track.

There are two modules that try to clean events more thoroughly than pulse cleaning alone. The first is *TopolocalSplitter*(TS), which starts from the Q-frames and loops over pulses and splits the event into clusters of pulses that contain at least a number of causally† connected pulses within a certain time window. Some extra cleaning, similar to seededRT cleaning, is done in addition

\*The number of pairs for  $n$  DOMs is equal to  $\frac{1}{2}n(n - 1)$  and scales with  $n^2$ .

\*The speed of light in vacuum is equal to  $\approx 0.3$  m/ns, meaning the particle travels around 100 m in 0.3  $\mu$ s, without accounting for the delayed photon propagation necessary for detection.

†The time between two DOM hits cannot be less than the time that light may have taken.

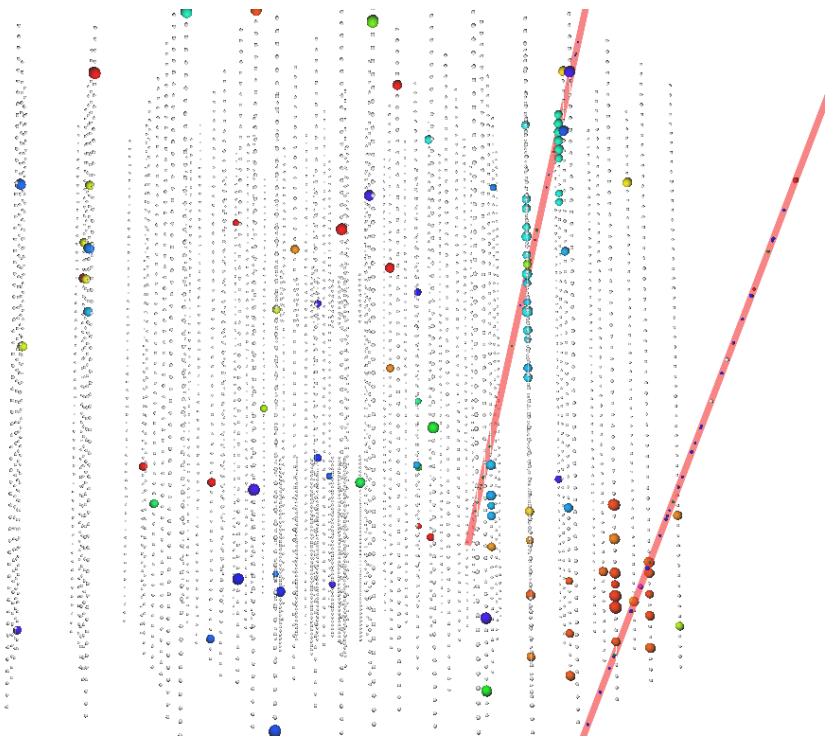


Figure 2.4: Event display of a simulated coincident event of two downgoing muons. The colors of the event range from red (early) to blue (late). The first muon hits the bottom of the detector, while the second traverses mainly the upper part. These events are often reconstructed as up-going and therefore result in a large background contribution. The scattered isolated hits are due to noise effects and mostly removed by pulse cleaning.

and can split coincident events that have overlapping readout windows, but are geometrically separated.

The second module, and also used in this analysis, is called `IceHive`. A full description can be found in the doctoral thesis of M. Zoll [[mzollthesis](#)]. The module consists of two main parts: one that splits events and handles coincident events, `HiveSplitter` and another that has a refined pulse cleaning, `HiveCleaning`.

### 2.3.1 `HiveSplitter`

The module assumes that individual particles will create *clusters* of hits in the detector. A cluster can grow within a certain time window, but is separated from another cluster if it's not spatially connected. An initial cluster is formed if the multiplicity of hits exceeds a certain threshold (usually 3 or 4). The main difference in this module versus TS is that it uses hexagons to describe the detector instead of assuming a spherical parameterization. It makes more sense to optimize the search volume, where hits are clustered together, with a shape that describes the detector well and uses a discrete spacing between larger volumes instead of a uniformly growing sphere. The hexagonal shape is set by defining three heights. The first height is defined along the string of the hit DOM and is equal to the vertical distance along the string where another. The second height is the vertical height along the neighboring strings. The third height is the vertical height along the next-to-neighboring strings. An example is shown in Fig. 2.4.

When the active region is set, additionally it is checked if DOMs can be “connected”. `IceHive` assumes certain emission profiles (for both cascades as tracks) where light is produced. Three possible connections are checked:

1. Hits occur at the same time, but at a spatial distance in agreement with the Cherenkov emission profile (hits C&1 and 2&3 in Fig. 2.4)
2. Hits occur at a different time and a different location, but in agreement with the Cherenkov emission profile (hits C&2 in Fig. 2.4).

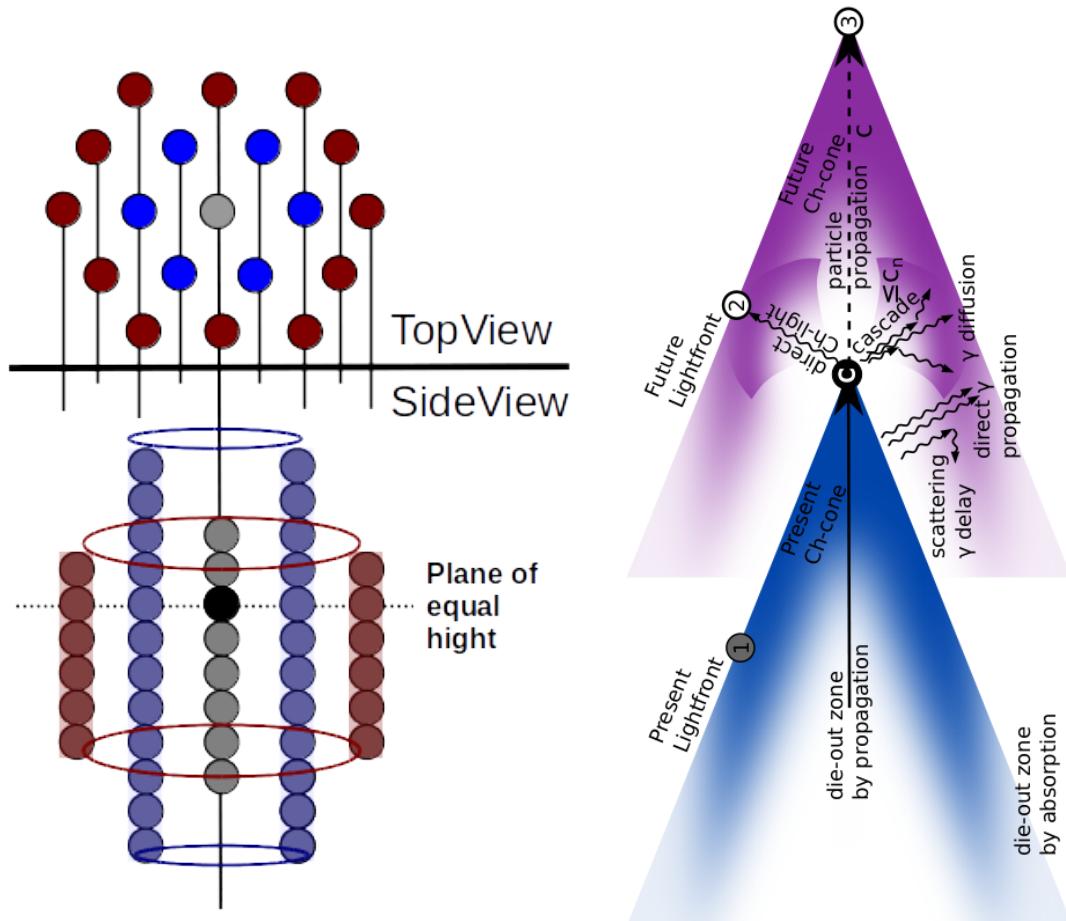


Figure 2.5: *Left:* The black circle illustrates a DOM that triggered a hit in the detector. The grey circles symbolize the DOMs along the string of the hit DOM. The number of DOMs that can be included in the active volume depends on the height defined by the module. The blue/purple DOMs belong to the neighboring strings and the red DOMs to the next-to-neighboring strings. The heights of both these sets of DOMs are also set by the module. This example shows  $h_2 > h_1 > h_3$ , the heights are also asymmetric in this example. *Right:* Illustration of Cherenkov emission profile of a traversing particle. Both figures from Ref. [mzollthesis].

3. Hits on topologically identical sites of an emission pattern that has moved along with the propagation of the particle (hits C&3 and hits 1&2 on Fig. 2.4).
- These clusters are finally separated into different events, P-frames.

### 2.3.2 HiveCleaning

Additionally, a similar cleaning as explained in Section 2.2 can be performed. Isolated hits that do not have neighbouring hits occurring within a certain distance within a certain time window, are removed. The main difference between this and seededRT cleaning is that the module again uses the hexagons as defined in the previous section.

the usage of `IceHive` has a great performance in separating coincident events, but often “overperforms” and splits clusters of hits that are originating from the same particle. This is predominantly the case for dim tracks that have large separations in between clusters (most of the triggered SMP events are of this type). It is because of this that the module `CoincSuite` was designed.

## 2.4 CoincSuite

Several testing algorithms allow to check if two or more split P-frames can originate from a single event. Five different scenarios were tested in this analysis, the first four are also chronologically

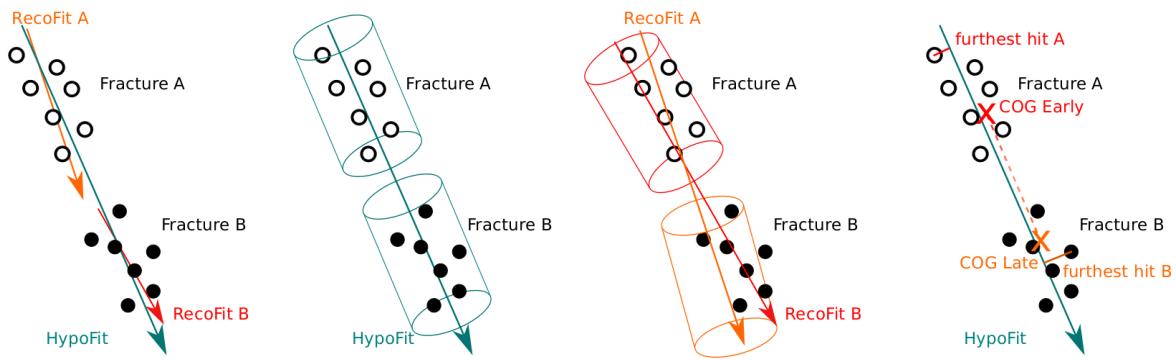


Figure 2.6: Schematic illustrations of possible recombination scenarios with the **CoincSuite** module.

shown in Fig. 2.5:

tw

1. Cluster alignment: the reconstructed direction of the individual clusters is compared to the direction of a reconstruction that uses the combined hits (HypoFit). The directions should be within a certain criticle angle.
2. Cylinder cluster containment: the DOMs of the individual clusters should be able to be grouped together in a cylinder that has its center and direction along the HypoFit.
3. Cylinder cluster alignment: a cylinder around the reconstruction of each cluster is draw. The cylinders should overlap within a certain fraction.
4. COG\* connection: the second quarter of the COG of the first cluster and the third quarter of the COG of the second cluster are computed. These COG should lie close enough and have to be in the vicinity of the HypoFit.
5. Velocity test: tests if the velocity of the HypoFit is close to the speed of light.

These tests lead to a successful recombination of signal events while the advantage **IceHive** was not lost.

## 2.5 Analysis techniques

### 2.6 Boosted Decision Tree Classifiers

---

\*Uitleggen wat COG is



### 3. The SPACE Analysis

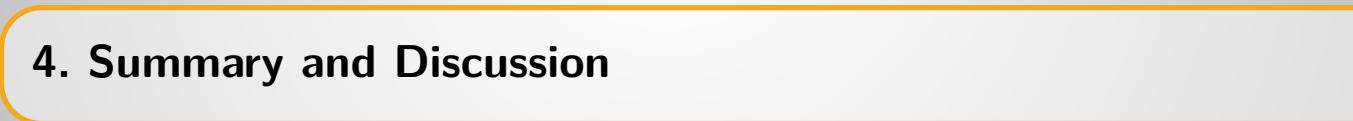
De uitleg van temperature modulations is goed in Sam zijn thesis <https://arxiv.org/pdf/physics/0312102v1.pdf>  
<https://arxiv.org/pdf/1310.1284.pdf>

Ook ergens een tabel maken met info over je data runs. Duidelijk maken wat de livetime is  
bv en ook zeggen van wanneer to wanneer een bepaalde run liep (2011: mei 2011- mei 2012)

Klaus zijn paper? <https://arxiv.org/abs/1806.05696>

- 3.1 Motivation**
- 3.2 General strategy**
- 3.3 Event cleaning**
- 3.4 Variables**
- 3.5 BDT results**
- 3.6 Pull validation**
- 3.7 Systematic Uncertainties**
- 3.8 Results**



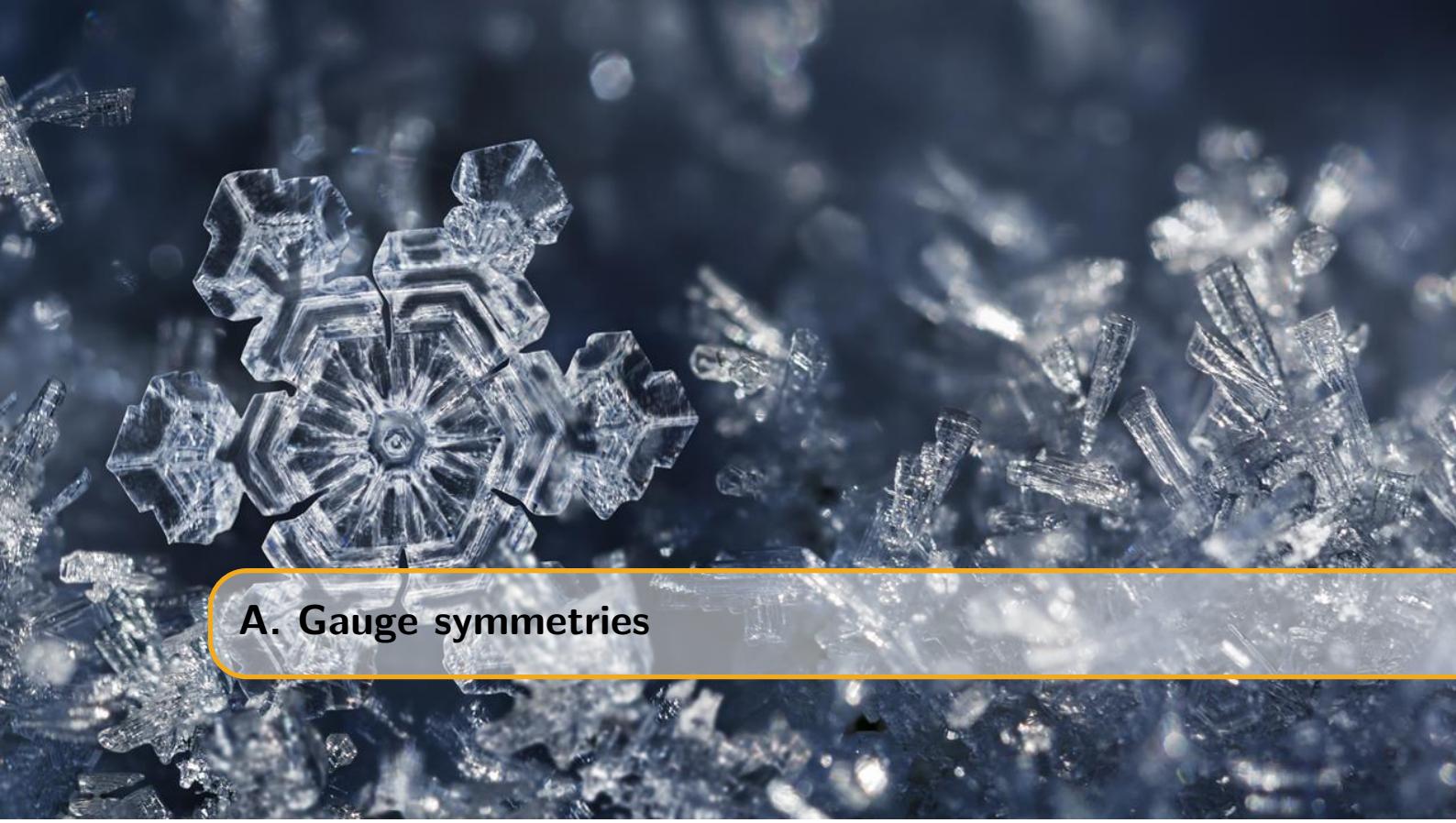


## **4. Summary and Discussion**



# Appendices





## A. Gauge symmetries

### NOG NIET GEDAAN

The difference between global and local symmetries are not straightforward for everybody. In this appendix I try to give a better view of the matter.

Imagine that at each point in space and time there is a circle attached to it. If one shifts all circles of all points with a fixed angle the underlying physics hasn't changed. If we look at the whole in a different angle, nothing seems to be changed as everything holds the same relative orientation. This is a global symmetry. For local symmetries we instead shift each circle through a different angle, but an angle that changes smoothly from point to point and in a way that we can say how that angle is varying between different nearby regions. Then it will turn out that we can describe that rotation angle by means of a so-called gauge field, which just lets us transport the charged scalar field from one point in space time to another, taking account of how the rotation angle of the circle is changing. A gauge is a kind of coordinate system that is varying depending on the location with respect to some underlying space. In physics we are almost always concerned with space-time as the underlying space, and we are typically interested in theories that are invariant with respect to the choice of gauge or coordinate system.

Dan wat uitleg vanuit je QFT boek en de dingen hieronder: Je wilt je derivative anders doen werken in je theory onder een transformatie, maar daarvoor heb je een veld nodig. M.a.w.: dankzij een veld heb je lokale ijktransformatie mogelijk!



## B. Planck's law

bron: <http://hyperphysics.phy-astr.gsu.edu/hbase/quantum/rayj.html>

### B.1 Electromagnetic waves in a cubical cavity

Suppose we have EM waves in a cavity at equilibrium with its surroundings. These waves must satisfy the wave equation in three dimensions:

$$\frac{\partial^2 \Psi}{\partial x^2} + \frac{\partial^2 \Psi}{\partial y^2} + \frac{\partial^2 \Psi}{\partial z^2} = \frac{1}{c^2} \frac{\partial^2 \Psi}{\partial t^2}. \quad (\text{B.1})$$

The solution must give zero amplitude at the walls. A non-zero value would mean energy is dissipated through the walls which is in contradiction to our equilibrium assumption. A general solution takes the form of

$$\Psi(x, y, z, t) = \Psi_0 \sin k_1 x \sin k_2 y \sin k_3 z \sin k_4 t, \quad (\text{B.2})$$

which, after requiring  $k_n L = n\pi$  with  $n = 0, 1, 2, \dots$  and  $k_4 \frac{\lambda}{2c} = \pi$ , leads to

$$\Psi(x, y, z, t) = \Psi_0 \sin \left( \frac{n_1 \pi x}{L} \right) \sin \left( \frac{n_2 \pi y}{L} \right) \sin \left( \frac{n_3 \pi z}{L} \right) \sin \left( \frac{2\pi c t}{\lambda} \right). \quad (\text{B.3})$$

From the wave equation it is easy to find that

$$n^2 = n_1^2 + n_2^2 + n_3^2 = \frac{4L^2}{\lambda^2}, \quad (\text{B.4})$$

which span up a sphere in “n-space” with a volume of  $\frac{1}{8} \frac{4}{3} \pi n^{3/2}$ , where the first term originates from the positive nature of  $n_{1,2,3}$ . Because there are two possible polarizations of the waves one has to multiply with an additional factor 2. The number of modes per unit wavelength is equal to

$$\frac{dN}{d\lambda} \times \frac{1}{L^3} = \frac{d}{d\lambda} \left[ \frac{8\pi L^3}{3\lambda^3} \right] \times \frac{1}{L^3} = - \left[ \frac{8\pi}{\lambda^4} \right]. \quad (\text{B.5})$$

### B.1.1 Classical approach

Following the principle of equipartition of energy, each standing wave mode will have an average energy  $kT$  with  $k$  the Boltzmann constant and  $T$  the temperature in Kelvin. The energy density is then:

$$\frac{du}{d\lambda} = -kT \frac{8\pi}{\lambda^4}. \quad (\text{B.6})$$

In function of frequency  $\nu = \frac{c}{\lambda}$ :

$$\frac{du}{d\nu} = -\frac{c}{\lambda^2} \frac{du}{d\lambda} = \frac{8\pi k T \nu^2}{c^3}, \quad (\text{B.7})$$

also known as the Rayleigh-Jeans law\*. Problem: divergence

### B.1.2 Quantum approach

The energy levels from a quantized harmonic oscillator are equal to

$$E_r = h\nu \left( r + \frac{1}{2} \right) = \frac{hc}{\lambda} \left( r + \frac{1}{2} \right) \quad \text{with } r = 0, 1, 2, \dots \quad (\text{B.8})$$

Implementing eq. B.4

$$E = \left( r + \frac{1}{2} \right) \frac{hc}{2L} \sqrt{n_1^2 + n_2^2 + n_3^2} \quad (\text{B.9})$$

According to statistical physics the average energy is now not equal to  $kT$  but follows a probability distribution

$$p(\nu, r) = \frac{e^{-r h \nu}}{\sum_{r=0}^{\infty} e^{-r h \nu}}, \quad (\text{B.10})$$

where we reference to the ground state of the oscillator:  $E'_r = E_r - E_0$ .

The average energy is now:

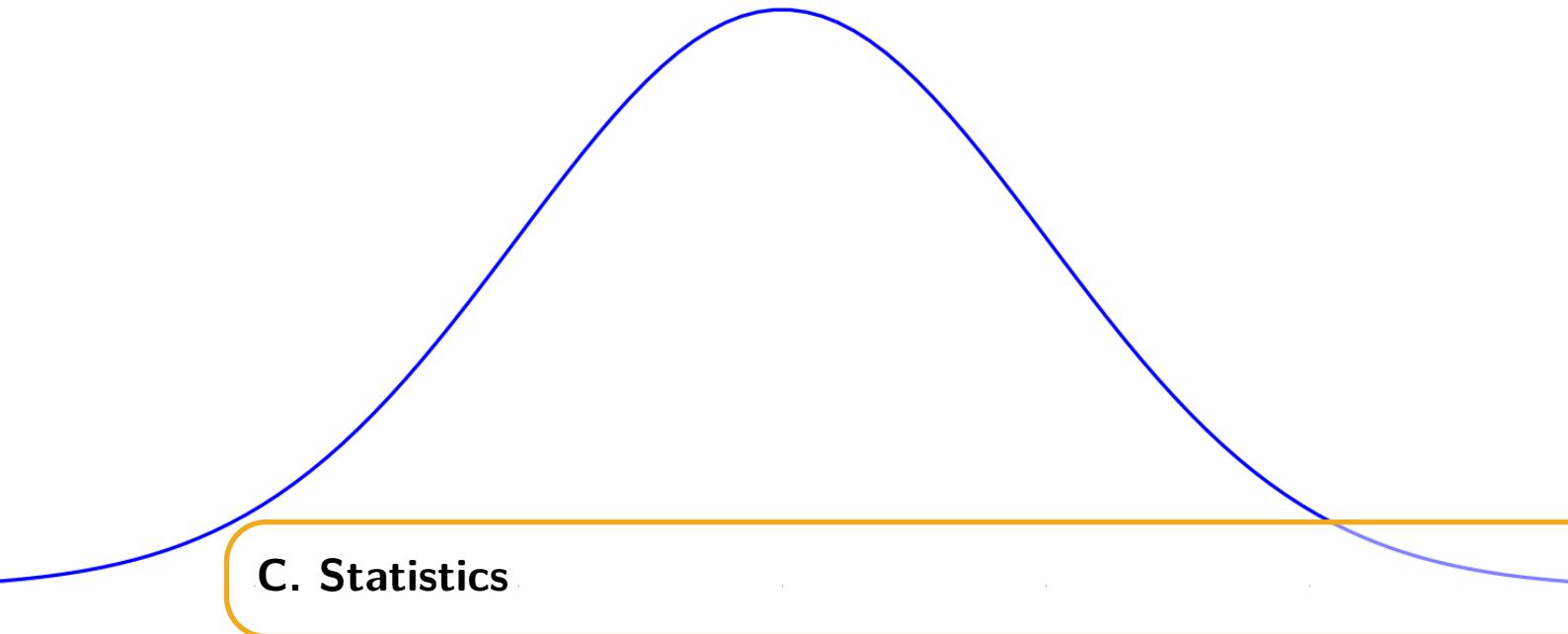
$$\begin{aligned} \langle E(\nu) \rangle &= \sum_{r=0}^{\infty} E(\nu, r) \cdot p(\nu, r) = \frac{\sum_{r=0}^{\infty} r h \nu e^{-r h \nu}}{\sum_{r=0}^{\infty} e^{-r h \nu}} \\ &= \frac{h \nu}{e^{h \nu / k T} - 1} \end{aligned} \quad (\text{B.11})$$

Substituting this for  $kT$  in eq. B.7 we find Planck's equation:

$$\frac{du}{d\nu} = \frac{8\pi h \nu^3}{c^3} \frac{h \nu}{e^{h \nu / k T} - 1} \quad (\text{B.12})$$

---

\*This is often quoted per unit of steradian, which results in  $\frac{2kT\nu^2}{c^3}$



## C. Statistics

A word that is often mentioned in this work is “statistics”. It refers to the statistical error of a counting experiment, i.e. the Poissonian error. The Poisson distribution is a discrete probability of a certain number of  $n_{\text{events}}$  occurring in a fixed time interval. The Poisson probability function is given by

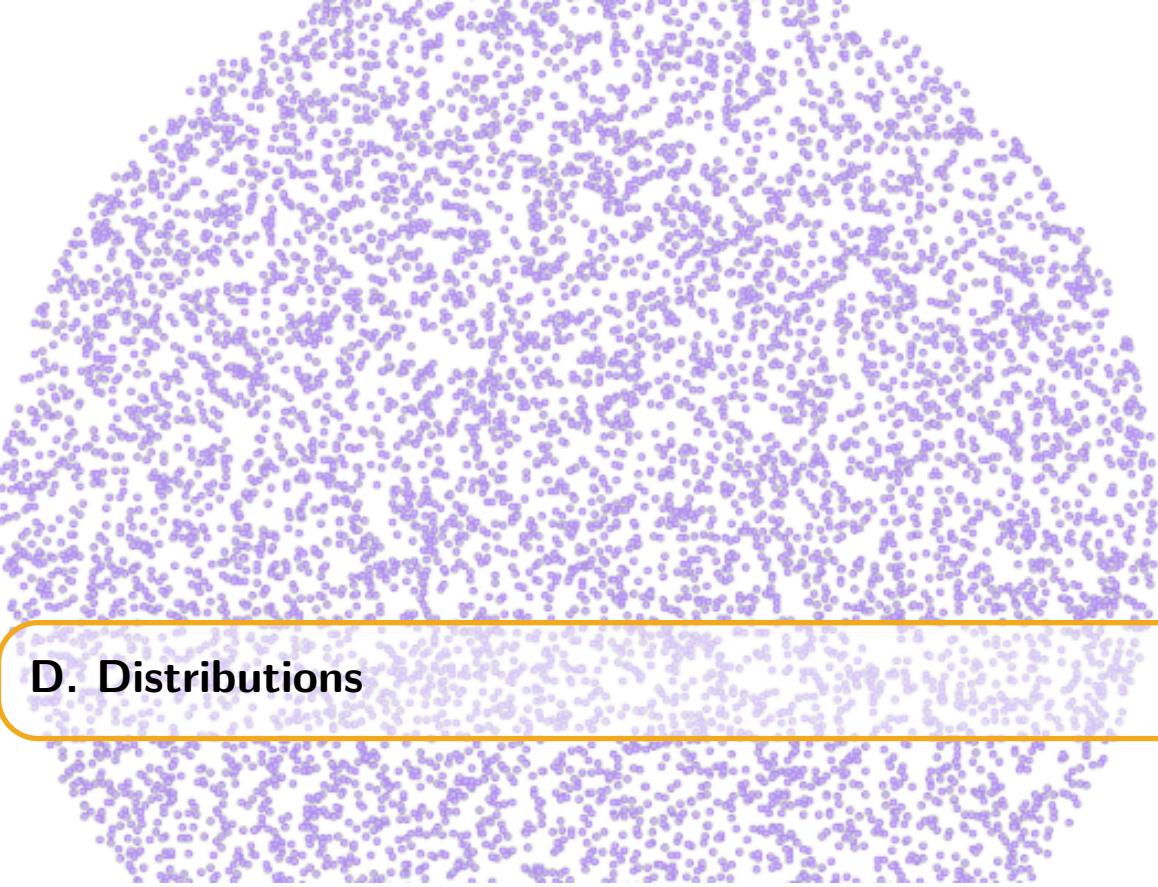
$$P(n) = \frac{\lambda^n e^{-\lambda}}{n!}, \quad (\text{C.1})$$

where  $\lambda$  is the expected number of events and also equal to the variance. An experiment that counted  $N$  events therefore has a statistical error of

$$\sigma = \sqrt{N} \quad (\text{C.2})$$

In other words: higher statistics denotes a lower statistical error.





## D. Distributions

### D.1 Spherical random numbers

Most random number generators provide uniform distributions from between the range [01,]. Assume we want to make a uniform distribution along a sphere with angles  $\phi$  and  $\theta$  and radius  $r$ , in spherical coordinates. Random numbers between  $[0, \pi]$ ,  $[0, 2\pi]$  and  $[0, R]$  (the ranges of the coordinates) would not give a uniform distribution as illustrated in Fig. D.1 (left).

The differential surface area,  $dA$ , is equal to  $dA(d\phi, d\theta) = r^2 \sin(\phi) d\phi d\theta$ . If we want the distribution  $f(v)$  to be constant for a uniform distribution, then  $f(v) = \frac{1}{4\pi}$  since  $\int \int_S f(v) dA = 1$  and  $\int \int_S dA = 4\pi$ . We want the distribution in function of the angles, so

$$f(v)dA = \frac{1}{4\pi}dA = f(\phi, \theta)d\phi d\theta. \quad (\text{D.1})$$

Since we know the expression for  $dA$ , we find that

$$f(\phi, \theta) = \frac{1}{4\pi} \sin(\phi), \quad (\text{D.2})$$

and separating the angles:

$$f(\theta) = \int_0^\pi f(\phi, \theta) d\phi = \frac{1}{2\pi}, \quad (\text{D.3})$$

$$f(\phi) = \int_0^{2\pi} f(\phi, \theta) d\theta = \frac{\sin(\phi)}{2}, \quad (\text{D.4})$$

where it is clear that  $f(\pi)$  scales with  $\sin(\phi)$ ; there are more points needed at the equator (this makes sense, as the total surface at the equator is much larger!).

The question is now how one can get a sample to follow the distribution of  $f(\phi)$ . For this, we use the *inverse transform sampling* method where one makes use of the cumulative distribution function,  $F(\phi)$ , which increases monotonically

$$F(\phi) = \int_0^\phi f(\phi') d\phi' = \frac{1}{2} (1 - \cos(\phi)). \quad (\text{D.5})$$

The method shows that if  $u$  is a random variable drawn from a uniform distribution, we have to find the inverse function of  $F$ ,

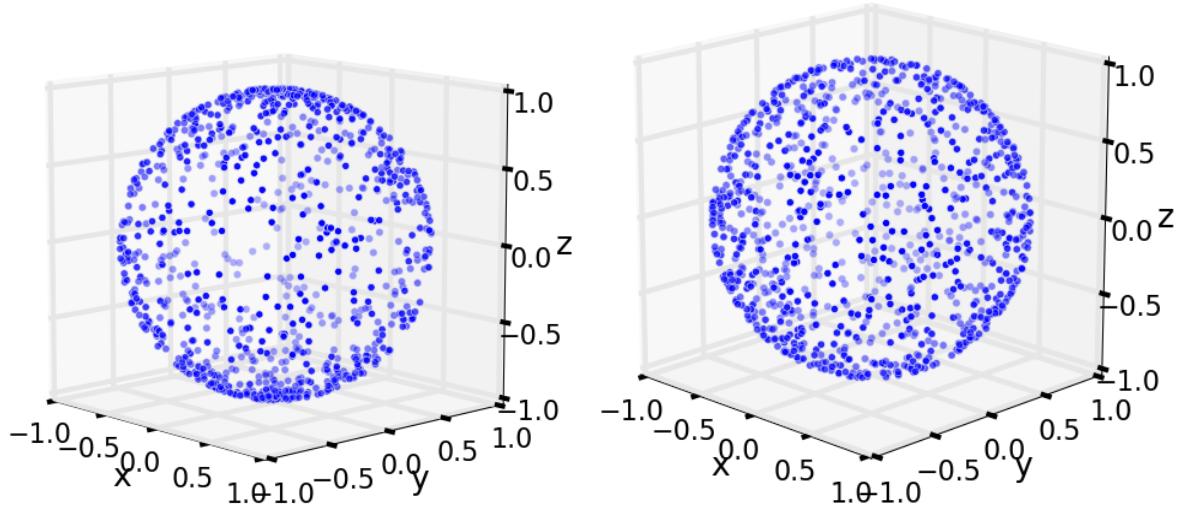


Figure D.1: Blub

$$F(F^{-1}(u)) = u \quad (\text{D.6})$$

$$\frac{1}{2} (1 - \cos(F^{-1}(u))) = u \quad (\text{D.7})$$

$$F^{-1}(u) = \arccos(1 - 2u). \quad (\text{D.8})$$

In other words: if  $u$  is a random variable drawn from a uniform distribution, then  $\phi = \arccos(1 - 2u)$  follows a distribution necessary for a uniform spherical distribution. Similarly,  $\theta = \frac{1}{2\pi}u$ .

## D.2 Power law distributions

Analogous to what was written in the previous section, one can produce a power law distribution from random numbers using the inverse transform sampling method:

$$\begin{aligned} f(E) &= A \cdot E^{-\gamma} \quad (\text{powerlaw}) \\ F(E) &= \int_{E_{min}}^E A \cdot E^{-\gamma} dE = u \quad (\text{inverse sampling, } u \text{ random number } [0,1]) \\ &= A \left[ \frac{E^{-\gamma+1}}{-\gamma+1} \right]_{E_{min}}^E \\ &= \frac{A}{-\gamma+1} (E^{-\gamma+1} - E_{min}^{-\gamma+1}) \end{aligned} \quad (\text{D.9})$$

Because we know that  $F(F^{-1}(u)) = u$ , we can find an expression for  $F^{-1}(u)$ :

$$\begin{aligned} u &= \frac{A}{-\gamma+1} \left( (F^{-1}(u))^{-\gamma+1} - E_{min}^{-\gamma+1} \right) \\ &\Rightarrow \end{aligned} \quad (\text{D.10})$$

$$F^{-1}(u) = \left( \left( \frac{-\gamma+1}{A} \cdot u \right) + E_{min}^{-\gamma+1} \right)^{1/(-\gamma+1)}$$

To find  $A$ , we use the property of a CDF:

$$F(E_{max}) = 1 \Rightarrow A = \frac{-\gamma+1}{E_{max}^{-\gamma+1} - E_{min}^{-\gamma+1}}, \quad (\text{D.11})$$

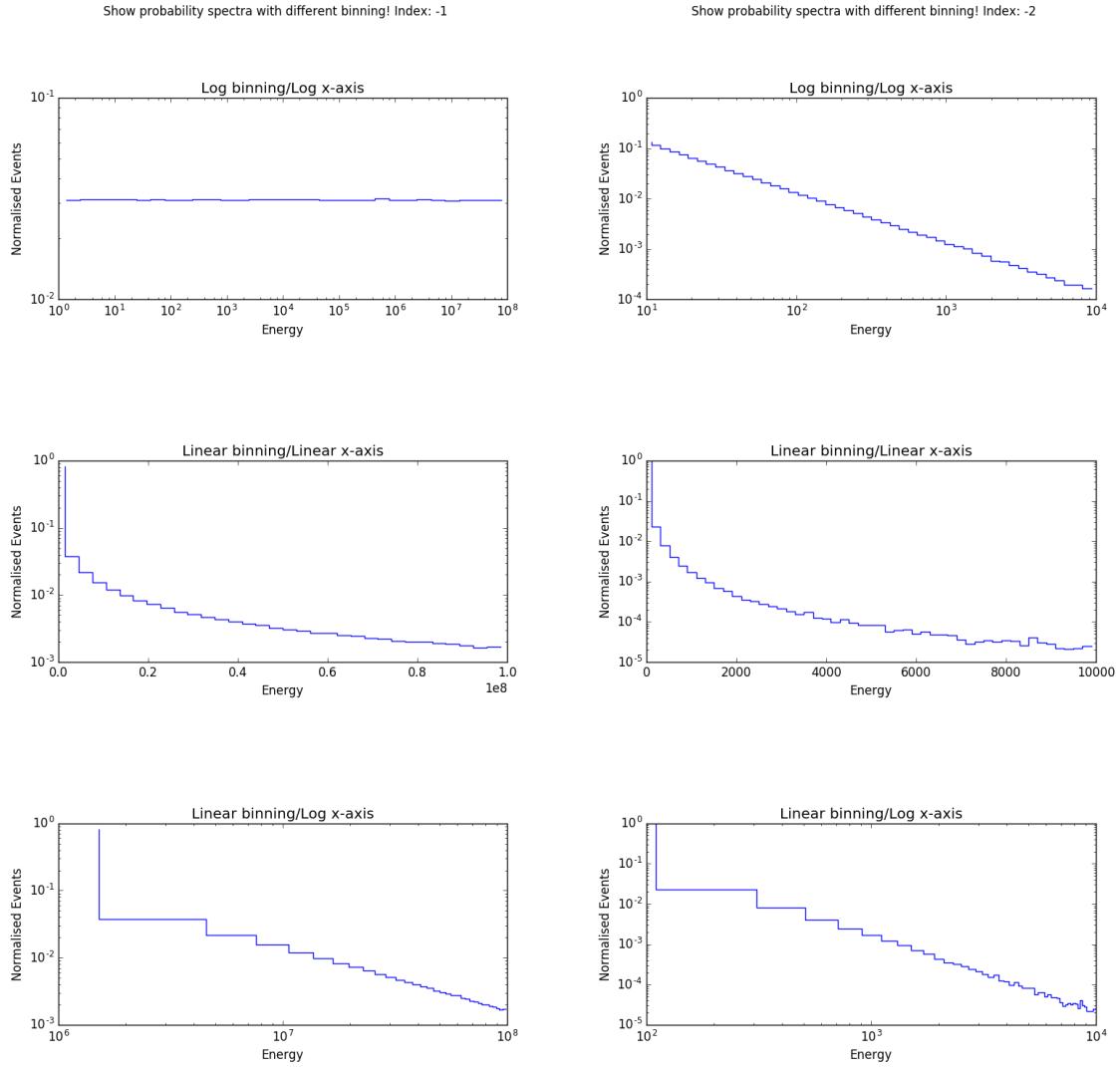


Figure D.2: Blub

leading to

$$F^{-1}(u) = \left( (1-u) \cdot E_{min}^{-\gamma+1} + u \cdot E_{max}^{-\gamma+1} \right)^{1/(-\gamma+1)}, \quad (\text{D.12})$$

which shows how one can draw a distribution in function of  $E$  following  $f(E)$  with a uniform random number  $u$ .

For  $\gamma = -1$ , the computations are analogous and one can see that this will produce a uniform distribution in log space. This is shown in Fig. D.2.

$$\begin{aligned} E &= E_{min} \cdot 10^{u \cdot \log \frac{E_{max}}{E_{min}}} \\ &= 10^{u[\log E_{min}, \log E_{max}]} \end{aligned} \quad (\text{D.13})$$

### D.3 Angular distributions

As seen in Section D.1, the differential space angle  $d\Omega$  is equal to

$$d\Omega = \sin \theta d\theta d\phi. \quad (\text{D.14})$$

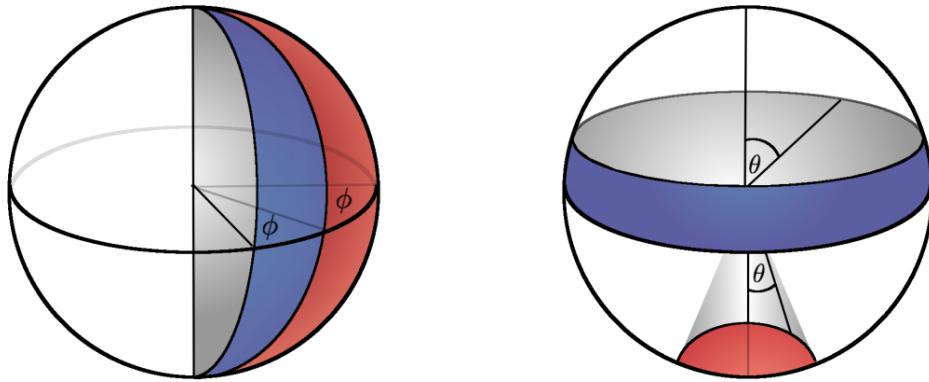


Figure D.3: Illustration of angle distributions in spherical coordinates. The blue and red surfaces are equal in size. The left figure clearly shows the surface to be proportional to the azimuth. The right figure shows how there is a non-trivial dependence on the zenith angle for equal partitions on the surface of a sphere.

If one shows the distribution of  $\phi$  and/or  $\theta$ , then this is the same as showing partial integrations per bin. We find that

$$\Omega \propto \cos \theta \phi, \quad (\text{D.15})$$

or in other words: the space angle is proportional to the azimuth and the cosine of the zenith. An example is shown in Fig. ??.

## D.4 Weighting

A method that is often used in simulations is *weighting*. The simulated and expected differential flux of particles is often not the same, mainly due to two reasons:

- The flux has no uniform power law behavior. As can be seen in Fig. ??, there are multiple “kinks” and changes in the spectrum. Instead of simulating the flux according to one model, a general uniform flux is used and later reweighted to be able to fit to other models more easily.
- A steep power law indicates very few events at the highest energy bins. This means large CPU time would be necessary to simulate these events. As an example, let us assume two different fluxes

$$f_1 = A \cdot x^{-1} \quad (\text{D.16})$$

$$f_2 = B \cdot x^{-2}, \quad (\text{D.17})$$

where  $A = 10^3$  and  $B = 10^4$ , so the fluxes cross at a value of  $x^{-1+2} = x = \frac{10^4}{10^3} = 10$ . In the interval  $x \in [10^3, 10^4]$ , the number of events for  $f_1$  is equal to  $10^3$ , whereas for  $f_2$  this is equal to 9.

Simulating with harder spectra\* leads to more statistics in high-energy bins.

The weights can be generally written down as

$$w = \frac{dN_{exp}}{dAd\Omega dEdt} \times \frac{dAd\Omega dE}{dN_{sim}}. \quad (\text{D.18})$$

A disadvantage of using weights is that certain events with a high weight are rare but can dominate or obscure the sample in the tails of certain distributions.

---

\*Harder spectra equals to a lower gamma, since there will be more high-energy events.

## 5. Some useful things for LaTeX

### 5.1 Definitions

This is an example of a definition. A definition could be mathematical or it could define a concept.

**Definition 5.1.1 — Definition name.** Given a vector space  $E$ , a norm on  $E$  is an application, denoted  $\|\cdot\|$ ,  $E$  in  $\mathbb{R}^+ = [0, +\infty[$  such that:

$$\|\mathbf{x}\| = 0 \Rightarrow \mathbf{x} = \mathbf{0} \quad (5.1)$$

$$\|\lambda\mathbf{x}\| = |\lambda| \cdot \|\mathbf{x}\| \quad (5.2)$$

$$\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\| \quad (5.3)$$

### 5.2 Remarks

This is an example of a remark.



The concepts presented here are now in conventional employment in mathematics. Vector spaces are taken over the field  $\mathbb{K} = \mathbb{R}$ , however, established properties are easily extended to  $\mathbb{K} = \mathbb{C}$ .

### 5.3 Corollaries

This is an example of a corollary.

**Corollary 5.3.1 — Corollary name.** The concepts presented here are now in conventional employment in mathematics. Vector spaces are taken over the field  $\mathbb{K} = \mathbb{R}$ , however, established properties are easily extended to  $\mathbb{K} = \mathbb{C}$ .

### 5.4 Propositions

This is an example of propositions.

### 5.4.1 Several equations

**Proposition 5.4.1 — Proposition name.** It has the properties:

$$|||\mathbf{x}|| - ||\mathbf{y}||| \leq ||\mathbf{x} - \mathbf{y}|| \quad (5.4)$$

$$|| \sum_{i=1}^n \mathbf{x}_i || \leq \sum_{i=1}^n ||\mathbf{x}_i|| \quad \text{where } n \text{ is a finite integer} \quad (5.5)$$

### 5.4.2 Single Line

**Proposition 5.4.2** Let  $f, g \in L^2(G)$ ; if  $\forall \varphi \in \mathcal{D}(G)$ ,  $(f, \varphi)_0 = (g, \varphi)_0$  then  $f = g$ .

## 5.5 Examples

This is an example of examples.

### 5.5.1 Equation and Text

■ **Example 5.1** Let  $G = \{x \in \mathbb{R}^2 : |x| < 3\}$  and denoted by:  $x^0 = (1, 1)$ ; consider the function:

$$f(x) = \begin{cases} e^{|x|} & \text{si } |x - x^0| \leq 1/2 \\ 0 & \text{si } |x - x^0| > 1/2 \end{cases} \quad (5.6)$$

The function  $f$  has bounded support, we can take  $A = \{x \in \mathbb{R}^2 : |x - x^0| \leq 1/2 + \epsilon\}$  for all  $\epsilon \in ]0; 5/2 - \sqrt{2}[$ . ■

### 5.5.2 Paragraph of Text

■ **Example 5.2 — Example name.** Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris. ■

## 5.6 Exercises

This is an example of an exercise.

**Exercise 5.1** This is a good place to ask a question to test learning progress or further cement ideas into students' minds. ■

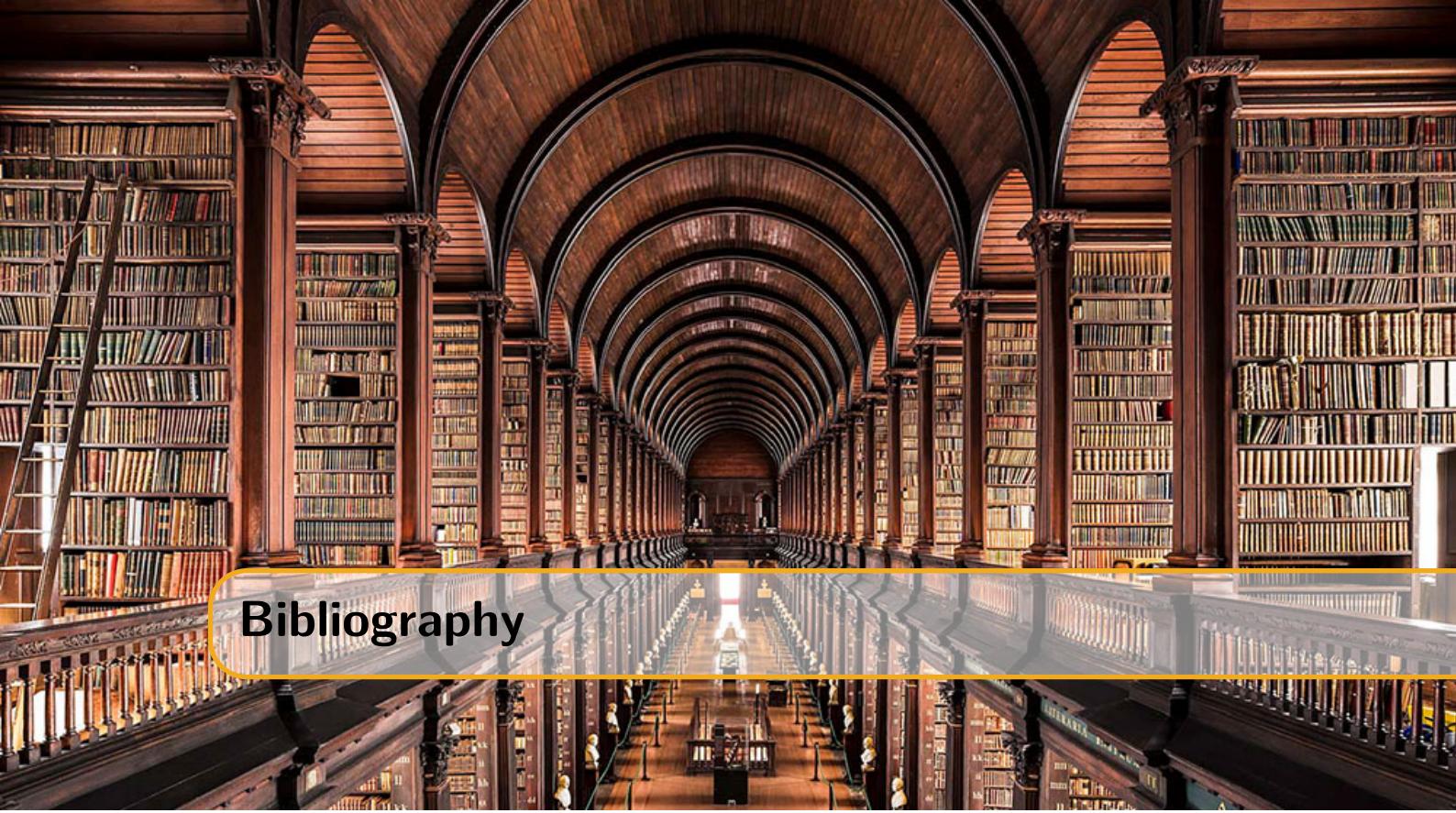
## 5.7 Problems

**Problem 5.1** What is the average airspeed velocity of an unladen swallow?

## 5.8 Vocabulary

Define a word to improve a students' vocabulary.

**Vocabulary 5.1 — Word.** Definition of word.



## Bibliography

- [1] D Schultz. “IceProd 2: A Next Generation Data Analysis Framework for the IceCube Neutrino Observatory”. In: *Journal of Physics: Conference Series* 664.6 (2015), page 062056. URL: <http://stacks.iop.org/1742-6596/664/i=6/a=062056> (cited on page 9).
- [2] Thomas K. Gaisser, Todor Stanev, and Serap Tilav. “Cosmic Ray Energy Spectrum from Measurements of Air Showers”. In: *Front. Phys. (Beijing)* 8 (2013), pages 748–758. DOI: [10.1007/s11467-013-0319-7](https://doi.org/10.1007/s11467-013-0319-7). arXiv: [1303.3565 \[astro-ph.HE\]](https://arxiv.org/abs/1303.3565) (cited on pages 9, 10, 12).
- [3] Giuseppe Battistoni et al. “Overview of the FLUKA code”. In: *Annals Nucl. Energy* 82 (2015), pages 10–18. DOI: [10.1016/j.anucene.2014.11.007](https://doi.org/10.1016/j.anucene.2014.11.007) (cited on page 10).
- [4] Ralph Engel et al. “Towards the next generation of CORSIKA: A framework for the simulation of particle cascades in astroparticle physics”. In: (2018). arXiv: [1808.08226 \[astro-ph.IM\]](https://arxiv.org/abs/1808.08226) (cited on page 10).
- [5] Askhat Gazizov and Marek P. Kowalski. “ANIS: High energy neutrino generator for neutrino telescopes”. In: *Comput. Phys. Commun.* 172 (2005), pages 203–213. DOI: [10.1016/j.cpc.2005.03.113](https://doi.org/10.1016/j.cpc.2005.03.113). arXiv: [astro-ph/0406439 \[astro-ph\]](https://arxiv.org/abs/astro-ph/0406439) (cited on page 10).
- [6] Morihiro Honda et al. “Calculation of atmospheric neutrino flux using the interaction model calibrated with atmospheric muon data”. In: *Phys. Rev. D* 75 (2007), page 043006. DOI: [10.1103/PhysRevD.75.043006](https://doi.org/10.1103/PhysRevD.75.043006). arXiv: [astro-ph/0611418 \[astro-ph\]](https://arxiv.org/abs/astro-ph/0611418) (cited on pages 10–12).
- [7] Rikard Enberg, Mary Hall Reno, and Ina Sarcevic. “Prompt neutrino fluxes from atmospheric charm”. In: *Phys. Rev. D* 78 (2008), page 043005. DOI: [10.1103/PhysRevD.78.043005](https://doi.org/10.1103/PhysRevD.78.043005). arXiv: [0806.0418 \[hep-ph\]](https://arxiv.org/abs/0806.0418) (cited on pages 10–12).
- [8] M. G. Aartsen et al. “Observation of High-Energy Astrophysical Neutrinos in Three Years of IceCube Data”. In: *Phys. Rev. Lett.* 113 (2014), page 101101. DOI: [10.1103/PhysRevLett.113.101101](https://doi.org/10.1103/PhysRevLett.113.101101). arXiv: [1405.5303 \[astro-ph.HE\]](https://arxiv.org/abs/1405.5303) (cited on pages 10–12).
- [9] C. Andreopoulos et al. “The GENIE Neutrino Monte Carlo Generator”. In: *Nucl. Instrum. Meth.* A614 (2010), pages 87–104. DOI: [10.1016/j.nima.2009.12.009](https://doi.org/10.1016/j.nima.2009.12.009). arXiv: [0905.2517 \[hep-ph\]](https://arxiv.org/abs/0905.2517) (cited on page 11).
- [10] Costas Andreopoulos et al. “The GENIE Neutrino Monte Carlo Generator: Physics and User Manual”. In: (2015). arXiv: [1510.05494 \[hep-ph\]](https://arxiv.org/abs/1510.05494) (cited on page 11).

- [11] Mario Dunsch et al. “Recent Improvements for the Lepton Propagator PROPOSAL”. In: (2018). arXiv: 1809.07740 [hep-ph] (cited on pages 13, 14).
- [12] Dmitry Chirkin. “Study of South Pole ice transparency with IceCube flashers”. In: *Proceedings, 32nd International Cosmic Ray Conference (ICRC 2011): Beijing, China, August 11-18, 2011*. Volume 4, page 161. DOI: 10.7529/ICRC2011/V04/0333 (cited on page 15).
- [13] Johan Lundberg et al. “Light tracking for glaciers and oceans: Scattering and absorption in heterogeneous media with Photonics”. In: *Nucl. Instrum. Meth.* A581 (2007), pages 619–631. DOI: 10.1016/j.nima.2007.07.143. arXiv: astro-ph/0702108 [ASTRO-PH] (cited on page 15).
- [14] R. Abbasi et al. “Calibration and Characterization of the IceCube Photomultiplier Tube”. In: *Nucl. Instrum. Meth.* A618 (2010), pages 139–152. DOI: 10.1016/j.nima.2010.03.102. arXiv: 1002.2442 [astro-ph.IM] (cited on page 16).
- [15] K. J. Ma et al. “Time and Amplitude of Afterpulse Measured with a Large Size Photomultiplier Tube”. In: *Nucl. Instrum. Meth.* A629 (2011), pages 93–100. DOI: 10.1016/j.nima.2010.11.095. arXiv: 0911.5336 [physics.ins-det] (cited on page 16).



## Index

### Symbols

<i>IceTray</i> .....	7
<i>Photonics</i> .....	14
“Monte Carlo” .....	7
“PMT jitter’ .....	14

### B

burn sample .....	15
-------------------	----

### C

called <i>CLSim</i> .....	14
CORSIKA .....	8, 9

### F

FLUKA .....	9
-------------	---

### I

IceTray .....	10
---------------	----

### M

MMC.....	14
----------	----

### N

neutrino-generator .....	9
--------------------------	---

### P

PPC.....	13
PROPOSAL .....	12, 14