Vrije Universiteit Amsterdam

Universiteit van Amsterdam

Master Thesis

Data processing pipeline for the KM3NeT neutrino telescope

Konrad Karaś

VU: 2624767 | UvA: 12017256

Supervisors

Universiteit van Amsterdam:	Adam Belloum
Netherlands eScience Center:	Ben van Werkhoven Stijn Heldens

A thesis submitted in fulfillment of the requirements for the joint UvA-VU Master of Science degree in Computer Science

August 13, 2019

Abstract

Placed deep in the Mediterranean Sea, the KM3NeT research infrastructure is considered to be the next generation of neutrino telescopes. Registering the flashes of Cherenkov radiation emitted as a side effect of neutrino particles interaction in the Earth or in the seawater, the KM3NeT aims to reconstruct the paths of these particles to search for their distant astrophysical sources. However, the sensitive infrastructure of photodetectors is exposed to various, undesirable factors (as ^{40}K potassium decay in seawater) which rise the background noise making the neutrino signal detection complex and challenging. The goal of this thesis is to propose fast and effective data processing pipeline that could be used as a filtration module for the KM3NeT software. Based on simulated detections we design and implement a solution that is able to successfully classify given time slices of data whether a neutrino event occurred. In this work we explain and evaluate each phase of the pipeline comparing its performance to previous implementations and state-of-the-art approaches. Because the pipeline is implemented on a GPU infrastructure we also provide an estimation for eventual live data processing.

Keywords— KM3NeT, neutrino detection, hit correlation, community detection, classification, GPU implementation

Contents

CONTENTS

Chapter 1

Introduction

High-energetic neutrino particles due to their low mass and electrical neutrality can serve as a great information medium as they weakly interact with matter. Emitted by different processes in the Universe, cosmic neutrinos can travel long distances without being absorbed or distracted by magnetic fields so their paths point back to their sources. Detection of such particles gives possibility to investigate the areas of the Universe that are not accessible to optical telescopes. At present it is technologically impossible to directly detect neutrino particles with decent sensitivity, thus scientists look for other, intermediate methods.

Placed deep in the Mediterranean Sea, the KM3NeT telescope [1] aims to detect high-energetic extraterrestrial neutrinos by registering flashes of Cherenkov light that are emitted as a side effect of the interaction of neutrinos with the Earth's surface or seawater. However, due to the sensitivity of the photomultiplier tubes of the detector, the acquired signal comes along with noise generated by other background factors (mostly potassium ^{40}K decay) not related to the neutrino event particles. Because it is computational expensive to analyze the raw data itself, it is necessary to develop an efficient data processing pipeline that would give an opportunity for event signal filtration and further neutrino path reconstruction.

1.1 The KM3NeT project

The KM3NeT (Cubic Kilometre Neutrino Telescope) is an European research infrastructure of the next generation neutrino telescope [1]. Still under construction, the telescope is expected to detect the first neutrinos in 2020. This innovatory project became a great

background for many research projects from all over the world in different domains of physics [2], electronic hardware [3] and software engineering [4].

Because neutrino particles are difficult to be detected directly, the goal of the KM3NeT telescope is to detect flashes of the Cherenkov light emitted by charged secondary particles resulting from neutrino interactions in the matter surrounding the telescope. In order to achieve that, the KM3NeT telescope will eventually consist of 1200 pressure-resistant spherical glass modules called Digital Optical Modules (DOM) attached to 600 strings anchored to the bottom of the sea. Each DOM consists of 31 photomultiplier tubes (PMT) aimed to different directions for better sensitivity and detection precision. The construction of the KM3NeT telescope will be distributed over three locations in the Mediterranean: KM3NeT-Fr (off Toulon, France), KM3NeT-It (off Portopalo di Capo Passero, Sicily, Italy) and KM3NeT-Gr (off Pylos, Peloponnese, Greece).

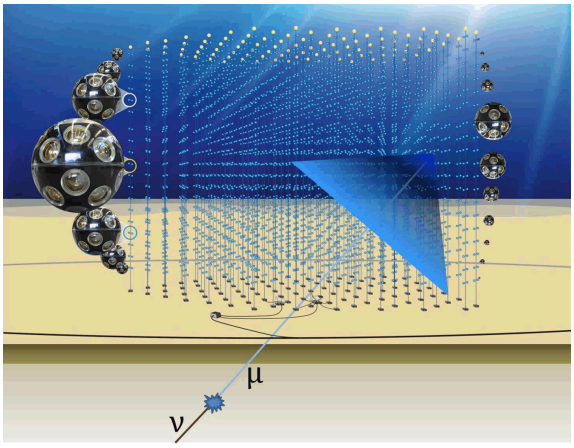


Figure 1.1: Artist's impression of the KM3NeT neutrino telescope placed in the depth of the sea. The figure explains how the DOM spheres are distributed in a cubic shaped area. Also, the conical beam of Cherenkov light was depicted as an effect of neutrino interaction with matter in the Earth. (source: *www.km3net.org*)

Emission and detection of Cherenkov radiation

For better understanding the background of the detection and further analysis of the acquired signal, it is worth to explain the basis of the emission of the Cherenkov radiation [5]. It is important to understand this phenomena before discussing the design and evaluation of the hit correlation criterion, which is part of the data processing pipeline.

It is assumed that speed of the light in vacuum is absolute. However, when the beam of light is traveling through transparent medium (e.g. seawater) its speed is decreased due to refractive index of that medium.

Charged particles, however, are not slowed down by the refractive index so it can occur that they will be travelling faster than light in water. As a result, the particle outruns its own electric field which can be compared to a plane traveling with supersonic speed. When a charged particle travels through a given medium, it is assumed that the medium is being electrically polarized by the particle's electric field and to return to its previous state, the energy contained in this disturbance radiates as a coherent shockwave - a visible light called Cherenkov radiation.

When a neutrino particle interacts with matter, a charged and accelerated lepton (muon or electron) is created, which causes the emission of Cherenkov radiation in a conical shape. This radiation can then be detected by the photomultiplier tubes (PMTs) of the detector.

KM3NeT event triggers

The readout of the KM3NeT telescope follows the "all-data-to-shore" concept in which analogue signals from the PMTs that pass a predefined threshold are being digitized and sent to the shore where the data processing occurs. To provide a decent filtration of the acquired signal, several filter levels were defined [1]. For the need of this thesis, it is important to discuss first two filtration levels, defined as level-zero and level-one, as the pipeline proposed in this thesis is to operate on level-zero hits only.

The level-zero (L0) filter refers to the threshold that is applied on the analogue signal in the PMTs. Signals that are stronger than predefined value (typically 0.3 photo-electrons) are being sent to shore for further processing.

The level-one (L1) filter is being applied on shore and it refers to a coincidence of at least two L0 hits from different PMTs in the same DOM. To consider that L1 hit took place, the coinciding hits must occur in predefined time window which typical value is $\Delta T = 10ns$ (due to the scattering of the light in seawater).

As the L1 filter provides acceptable level of filtration, it has to be applied for every set of L0 hits which is computationally demanding. The L1 filter also filters out all L0 event hits which were not classified as coinciding. The goal of this thesis is to propose a new method of filtration based on L0 hits, which aims to offer faster and preferably more accurate solution.

Related projects

The idea of constructing neutrino Cherenkov detectors had been already realized in the past. To justify the assumptions of the KM3NeT is it worth to mention two related

projects: the Super-Kamiokande and the IceCube.

In 1996 a neutrino observatory located under Mount Ikeno in Japan started operation [6]. So called Super-Kamiokande detector, placed 1000m underground in the Mozumi Mine, was designed to detect high-energy neutrinos to search for proton decay, study solar and atmospheric neutrinos, and keep watch for supernovae in the Milky Way Galaxy. The detector consisted of a cylindrical stainless steel tank filled with 50,000 tons of ultrapure water surrounded by 13,000 photomultiplier tubes. In 1998 the first evidence of neutrino oscillation was announced [7]. Further research resulted in Nobel Prize in Physics in 2015 for the work on neutrino oscillations.

Another neutrino observatory was founded in 2010 at the Amundsen-Scott South Pole Station in Antarctica [8]. The goal of the IceCube project was to detect emission of the Cherenkov light deep in ice. The detector consists of thousands of spherical optical sensors, each with a photomultiplier tube, buried beneath the surface and spread over a cubic kilometer from 1,450 meters to 2,450 meters depth. In 2013, it was announced that IceCube observatory had detected 28 neutrinos sourced outside the Solar System [9].

The KM3NeT project combines the ideas of both detectors. As it is classified as a water Cherenkov detector, it aims to detect the light flashes occurred in the water like the Super-Kamiokande, however the DOMs are distributed in a form of three-dimensional cubic kilometer grid which resembles the IceCube construction.

1.2 Research Questions

The goal of this thesis is to propose, implement and evaluate a new design of the data processing pipeline for the KM3NeT project. The purpose of such a pipeline is to filter out irrelevant data records and indicate time slices, in which a neutrino event could have occurred. What is important, is the fact that the input data for the pipeline consists of all detected hits, which come directly from the detector without any intermediate preprocessing. The pipeline is supposed to be fast and accurate so its output could be easily used for the neutrino path reconstruction.

The idea for the pipeline is to process the input data in the following flow: check whether there is a relation between detected signal hits and form a graph which would represent it treating the data records as graph nodes. Then, by applying graph community detection algorithms designate interesting groups of nodes (data records) and classify them based on given metrics. The pipeline itself will consist of three phases executed sequentially and each phase will be discussed and evaluated in oncoming chapters. The elements of the pipeline

are: correlation criterion, graph community detection and community classification. To structurize the paper properly it is necessary to define relevant research questions related to the pipeline and each of its phases.

RQ 1: *Is it possible to classify time slices of raw telescope data based on whether a neutrino event occurred?*

The KM3NeT detector is constantly recording every single detection for further processing. As most of the gathered data relates to the background noise only, it is computationally expensive and time consuming to extract subsets of records which may contain relevant data. The goal of this research is to propose and design a computationally feasible method to decide if a given time slice of data should be stored or not. To answer this research question, additional question were formulated:

RQ 2: *Based on the input data, how to designate pairs of correlated hits?*

The input data set consists of series of detections enriched with features like position coordinates and a time stamp. The goal of the correlation criterion is to define pairs of hits that might be causally related as it is assumed that event hits should occur close in space and time, in contrast to background noise hits. In this paper, we propose a new and accurate method for correlating hits and compare it to the current state-of-the-art. The output of the correlation criterion will serve as a graph in the next phase of the pipeline.

RQ 3: *What graph community detection algorithm would be feasible for revealing correlated groups of records?*

The adjacency matrix formed by the correlation criterion can be used as a basis for the graph creation. Assuming that correlation criterion defines pairs of correlated hits, it seems reasonable to consider the use of graph community detection algorithm to reveal groups of highly correlated records by dividing the graph into clusters.

RQ 4: *Is it possible to label the communities based on whether they contain a neutrino event?*

Having the graph communities (or clusters) defined, it is needed to designate which ones contain neutrino event hits. By taking into account different graph metrics, we will try to provide simple yet effective classifier that would finally be used to indicate if a given time slice is worth storing or not.

1.3 Thesis outline

As this thesis proposes a design of the data processing pipeline, the outline of this work will be formed similarly discussing step by step each of the considered phase. In next chapters, for each phase of the pipeline, we will explain in detail what was the approach and how the phase was implemented and evaluated. We will start from analyzing the given data set to have a better overview of the challenge that we have to face. Next we will discuss each step of the pipeline and finally we will provide a performance estimation for eventual real-time neutrino event detection.

Chapter 2

The pipeline design

The design of the pipeline was based on two data sets provided by the Nikhef[1] institute: a simulated data set of hits caused by ^{40}K (potassium-40) decay in the seawater which serves as background noise and simulated data of hits caused by neutrino events, which is the signal that we want to retrieve. The data set of event hits was divided in form of groups, each representing different neutrino interaction. Both data sets were generated and calibrated with the use of the KM3Pipe framework [10]. In this chapter both data sets will be explored and the design of the pipeline will be explained.

2.1 Data set exploration

Table 2.1 describes the data set and Table 2.2 describes some of available features of that data set.

Table 2.1: Data set description

number of background hits	45,325,246
number of event hits	334,686
number of simulated events	24,914
number of simulated events (group size $>$ 50)	1,475
detector dimensions	190m x 220m x 160m

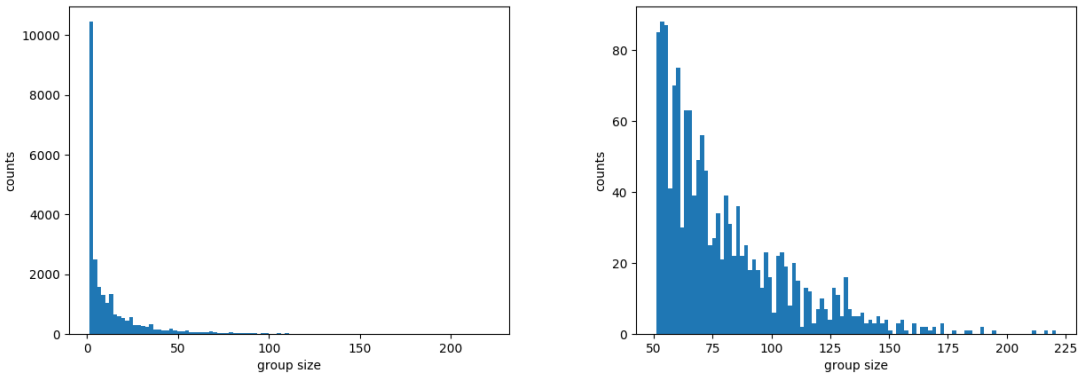
Distribution of event group sizes (see Figure 2.1), representing hits caused by different neutrino interactions with matter, follow exponential distribution where great majority is represented by very small groups. Such groups, caused by weak interactions or incomplete

[1] Dutch National Institute for Subatomic Physics

detection, might be hard to be found as the amount of L1 hits is much lower in comparison to bigger groups (Figure 2.2a).

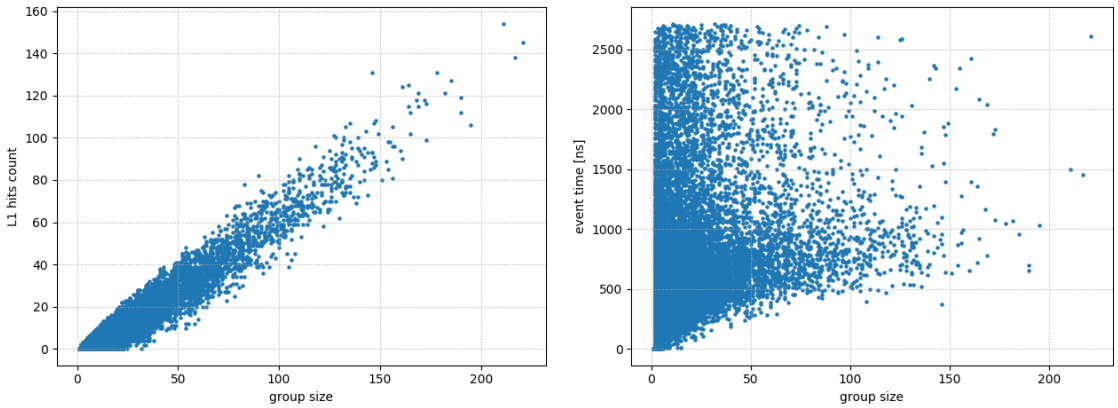
Table 2.2: Feature description of the simulated data set

Feature name	Description
channel_id	Unique ID of sting to which the DOM is attached to
pos_x/y/z	Three-dimensional position coordinate (in meters) of PMT in the detector volume
dir_x/y/z	Direction coordinate of PMT in DOM
dom_id	Unique ID of DOM
floor	Floor number where given DOM is placed in the detector volume
tot	Time over threshold (ToT) expresses charge accumulated in given time over predefined threshold (the stronger the signal was the higher the ToT value is)
time	Detection time stamp in nanoseconds
group_id (events only)	Unique ID of event group (events are supposed to be uncorrelated)



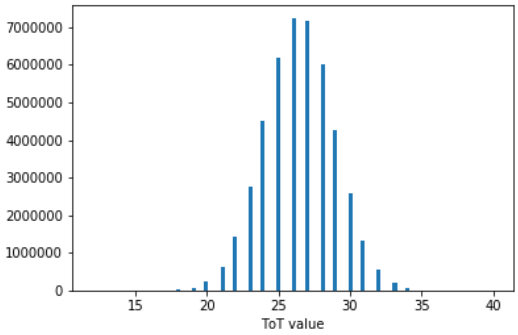
(a) **(b)**

Figure 2.1: Distribution of event groups sizes for the simulated data set. Left: distribution of all groups. Right: distribution of groups with sizes greater than 50. The sizes of given event groups follow exponential distribution, where small groups are the great majority.

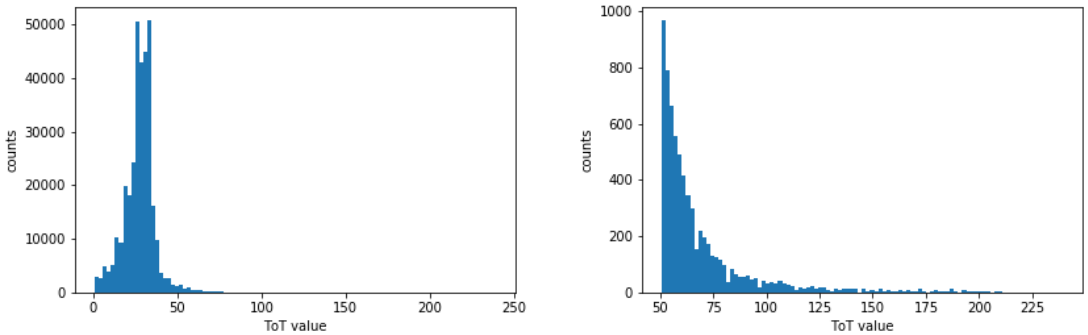


(a) **(b)**

Figure 2.2: Left: Amount of L1 hits for given group size. Right: Time of event duration for given group size.

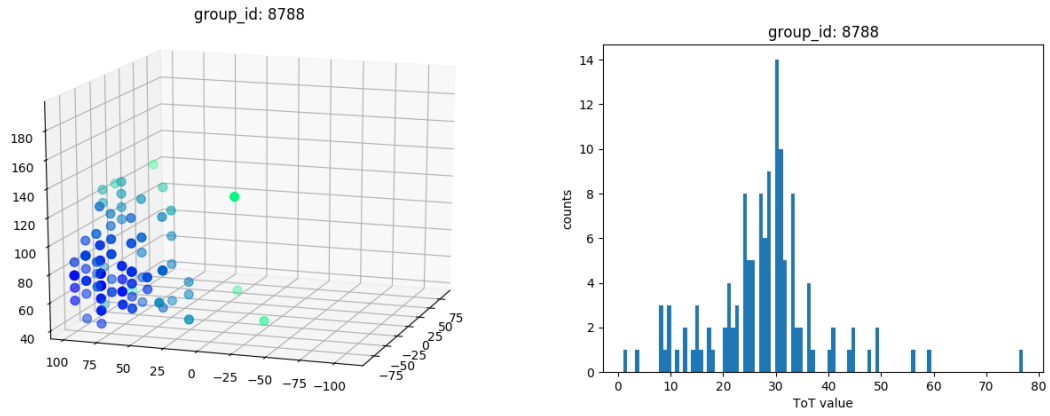


(a) ToT distribution of background noise



(b) ToT distribution of event hits **(c)** ToT distribution of event hits (the tail)

Figure 2.3: Time-over-threshold (ToT) distribution for the simulated data sets. Figure 2.3a shows distribution for simulated background noise (^{40}K decay in seawater) whereas Figure 2.3b shows distribution of event hits. Both data sets follow normal ToT distribution however event hits distinguish with a long tail for higher values (Figure 2.3c).



(a) (b)

Figure 2.4: Left: three-dimensional visualization of one event group (id: 8788) with 130 hits detected. Right: ToT distribution of given event group. The scatter plot shows that event hits are mostly gathered in limited area of the detector volume (hits marked in blue occurred earlier in time than hits in green). ToT distribution of that event mostly overlaps with ToT distribution of the background noise (see Figure 2.3a).

According to [1] the time-over-threshold (ToT) values could serve as a key factor for the signal detection. The more signal was accumulated in given time, the higher the probability of neutrino event is. However, as Figure 2.3 shows, for the simulated data sets, the ToT distributions of both event hits and background noise mostly overlap except the long tail of higher ToT values for event hits.

Figure 2.4 describes one of the simulated event groups. Regarding the ToT distribution of that group, we can see that values for most of the hits are in the range of ToT distribution of background noise, so detection of these hits solely based on ToT value is impossible. There are only few hits standing out with high ToT values but their amount is insufficient for accurate event detection.

This is the case for all other event groups, as such ToT values will not be taken into account for designing the data processing pipeline. However, it is possible that in real world scenario the pipeline accuracy could be improved by taking the ToT values into account.

2.2 Designing the pipeline

The goal of the pipeline is to process the input that consists of raw L0 hits and classify if given time slices might be related to neutrino events.

The pipeline consists of three sequential steps: correlation, community detection, and classification. The rest of this section gives a brief introduction of each step in the pipeline.

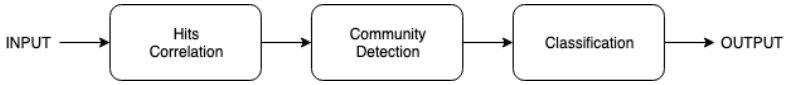


Figure 2.5: The scheme of the pipeline.

2.2.1 Hit correlation

The first part of the pipeline is the correlation step. The goal of the correlation step is to find pairs of hits that might have a causal relationship, meaning that they might be caused by the same muon traveling through the detector.

Based on the xyz-positions and time stamps of input records, a correlation criterion can be used to designate correlated pairs of hits. Neutrino event hits are likely to occur close in space and time, whereas hits that originate from background noise are evenly spread in the volume of the detector.

The correlation criterion plays important role in the pipeline as it serves as first filtration step and any inaccuracies will be propagated into next phases influencing the overall result. The output of the correlation step is an adjacency matrix of the hits, which is stored in the form of a CSR format.

2.2.2 Community detection

The next part of the data processing pipeline is based on a graph community detection algorithm. In this phase, the sparse matrix from the correlation step is used to form a graph network where hits are represented as graph nodes and their mutual correlations are expressed as undirected edges. Next, the graph is processed into clusters or communities. It is assumed that these clusters should represent a groups of data records which are highly correlated with each other and weakly correlated with the rest of the graph. Preferably, the communities should gather event hits records together, while keeping background hits apart or in other communities.

2.2.3 Classification

Having the communities defined, it is important to select proper metrics that could reveal which of the communities actually consist of neutrino event hits. In this research, we mostly focus on such metrics as community size (the number of nodes in given community) and

density (the number of interconnecting edges). By analyzing these two factors, we expect to find communities with neutrino event hits to stand out from the other communities that mostly gathered background noise hits. Next, the classifier should label suspected event hits in the data set so it would be possible to define the relevant time slice when the neutrino event occurred.

Chapter 3

Correlating hits

The first part of the data processing pipeline is the correlation step. Here all detected hits (denoted as L0 hits) are being processed and based on the features of the data set, their correlation is evaluated. The goal of correlating hits is to find pairs of hits that might be causally related. The output of the correlation step is an adjacency matrix, which can be used for further processing as a basis for creation of the graph. Checking the correlation is also the first filtration step, a good criterion should not correlate background noise with neutrino event hits.

In this chapter, an existing criterion [11] will be discussed and a new method will be proposed. Both solutions will be evaluated to check their recall and precision. Finally, the best solution will be selected for implementation in the pipeline.

3.1 Existing criteria

In this section, we will evaluate the existing correlation criterion called *Match 3B*, which was built on top of the *Quadratic Difference* criterion [11]. As Figure 2.4a shows, event hits are usually gathered in limited space of detector volume and the event duration is relatively short (see Figure 2.2b). Thus, it seems reasonable that main features of the correlation criterion are the position coordinates and time stamp of the detection.

The Quadratic Difference criterion calculates the distance between two detected hits and compares it to the distance that would be traveled with a speed of light in the same amount of time. If two given hits occurred closer, then they are assumed to be correlated.

The Match 3B criterion is more restrictive as it also takes into account a maximum distance that a photon can travel through seawater so it reduces the amount of unreasonably distant correlations that occurred in short time. The main factor that is taken into

account by the Match 3B criterion is so called *road width* parameter (expressed in meters). That parameter defines the distance where 90% of emitted light is being absorbed in the seawater. It is assumed that for the Cherenkov radiation in seawater the road with value is equal to 90m [11].

However, taking into account the dimensions of the detector (Table 2.1) and time duration of the events (Figure 2.2b) it is clear that criterion based on time difference between hits might be insufficient. In the time range of $500ns$ to $1000ns$, the light can travel a distance of $150m$ to $300m$ which actually covers the volume of the detector so a lot of noise will be correlated as well. Also, limiting the road width to $90m$ might still be insufficient. In Section 3.3 we reveal weaknesses of the Match 3B criterion which forced us to define a new criterion.

3.2 Defining a new criterion: Pattern Matrix

To define a new, more accurate criterion, it was necessary to analyze the simulated data set. The goal was to check how neutrino event hits are correlated with each other and with the background noise. To achieve that, for each pair of recorded hits in given time slice, we calculated the frequency of the spatial distance and time difference (see Figure 3.1).

It occurred that every event group follows similar pattern of correlation. Correlations between event hits mostly occur in short distance and low time difference (see Figure 3.1a and 3.1b). It is interesting to note that correlations between pairs of background noise hits mostly occur in similar distance ranges, irrespective of their time difference (Figure 3.1d). This observation suggests that the background noise is random and correlations in distance follow normal distribution. Figure 3.1c presents correlations between event hits and background noise which results are similar to background noise correlation only.

To effectively filter out the background noise, it would be crucial to mostly focus on correlations with short distance and low time difference, where the probability of correlation between neutrino event hits is the highest and correlations between background noise hits is the lowest.

It occurred that all the neutrino event groups follow similar patterns of correlations like ones depicted in the Figure 3.1. This observation lead to the idea of creating a correlation criterion based on the general pattern that would best fit the simulated data set. We decided to base the new criterion on the probability that a given spatial distance and time difference occurs between two neutrino event hits. We limited the area of investigation to $100m$ distance and $300ns$ time difference as mutual event hit correlations usually take place

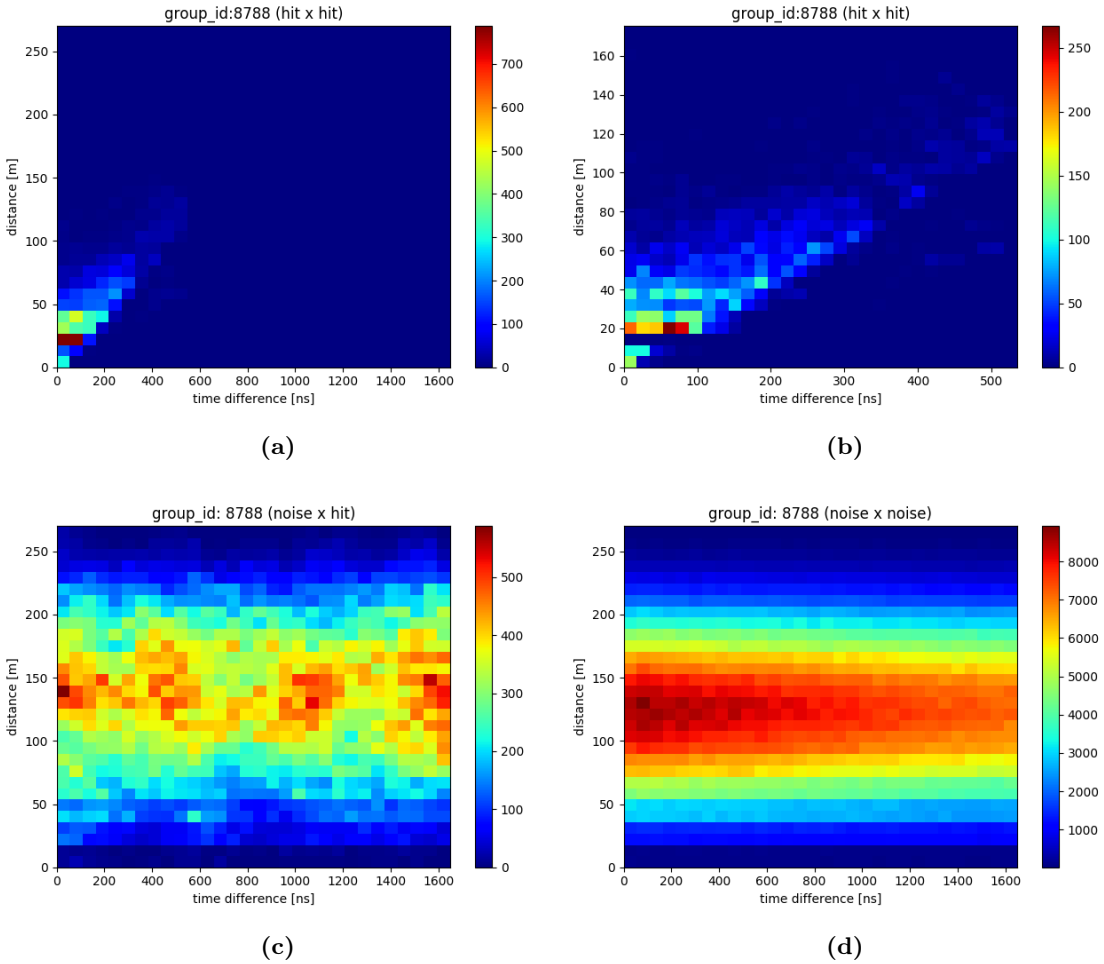


Figure 3.1: Figures show correlations between given pair of hits taking into account their distance and time difference. Colorbar represents number of occurrences. Figure 3.1a: correlations between detected event hits. Figure 3.1b: correlations between detected event hits with fitted scale. Figure 3.1c: correlations between event and background hits. Figure 3.1d: correlations between pairs of background noise hits.

within these boundaries. From these two ranges we create a grid of 50 rows and 50 columns, where each cell or bin represents a part of this space. Then we sorted each pair of hits within neutrino events based on their distance and time difference into these bins. Finally, for each bin we calculate the ratio of mutual event hit correlations to other correlations. As a result we obtained a matrix that represents two dimensional grid of probabilities of event hit correlations (see 3.2). By applying a minimal probability threshold value it is possible to manage the sensitivity of the correlation criterion.

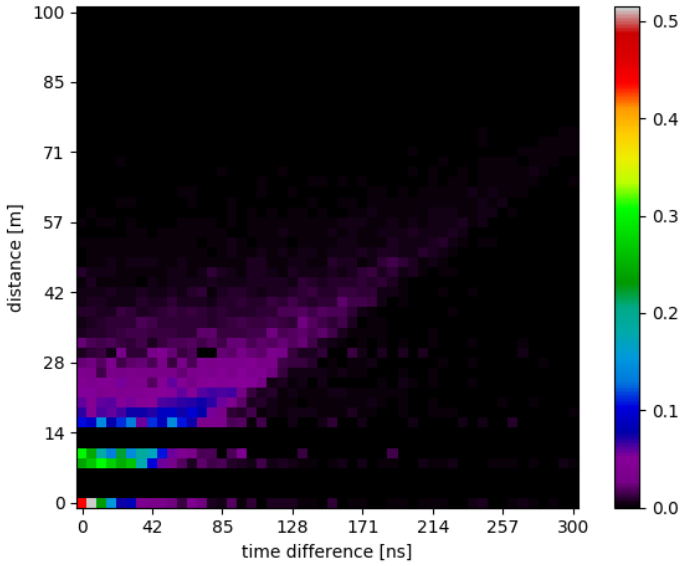


Figure 3.2: Correlation probability matrix based on a complete data set. The colorbar describes the probability that two event hits are correlated for given distance and time difference between them. Two visible horizontal stripes of missing data are due to the construction of the detector - there are no two DOMs placed such that these distances can occur.

3.3 Criteria evaluation

In order to evaluate the Pattern Matrix criterion and compare it to the Match 3B criterion, we prepared a data set which consists of 130 event hits (see Figure 2.4) and 5000 background hits. The data set was processed by the criteria and the output was validated with a confusion matrix. For evaluation we consider:

- *true positive (TP)* as correlation between two event hits
- *false positive (FP)* as correlation between even hit and background noise hit or two background noise hits
- *true negative (TN)* as no correlation between even hit and background noise hit or two background noise hits
- *false negative (FN)* as no correlation between two neutrino event hits

We use the following metrics to evaluate the correlation criteria:

- *recall* - a fraction of correctly guessed correlations among all possible correlations in data set
- *precision* - a fraction of correctly guessed correlations among all correct guesses
- F_1 *score* - a harmonic mean of precision and recall

These three metrics can be calculated as follows:

$$recall = \frac{TP}{TP + FN} \qquad precision = \frac{TP}{TP + FP} \qquad F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{3.1}$$

Match 3B evaluation

The results revealed weaknesses of the Match 3B criterion (see Figure 3.3). By adjusting the road width value it is possible to see whether and how this parameter influences the effectiveness of the correlation criterion. By increasing the road width value, there is a significant gain in recall as we correlate more hits by increasing the distance that a photon travel in the seawater. High recall means that we correlate most of the event hits in the data set, however we also correlate significant amount of the background noise. For the whole range of road width values we can see that precision is very low, which indicates that the Match 3B criterion is not able to provide a good distinction between event hits and background noise, as the majority of correlated hits refer to background noise. F_1 score, which is a harmonic average of the precision and recall, also remains at a low value, dominated by the score of the precision.

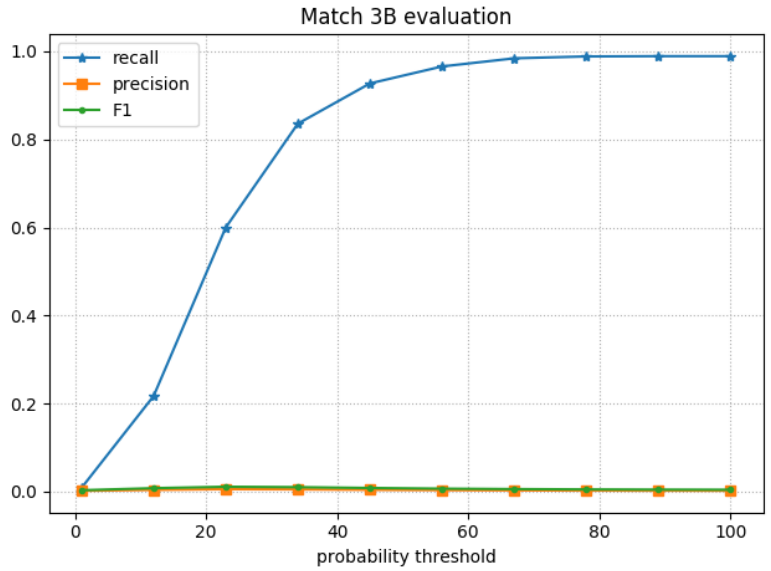


Figure 3.3: Match 3B criterion evaluation. Figure presents how adjustment of road width value influences correctness of the criterion.

Pattern Matrix evaluation

The same data set and similar evaluation approach are used as for the Pattern Matrix criterion. However, for the evaluation of the pattern matrix criterion we vary the probability

threshold instead of the road width parameter. Obtained results are presented in Figure 3.4.

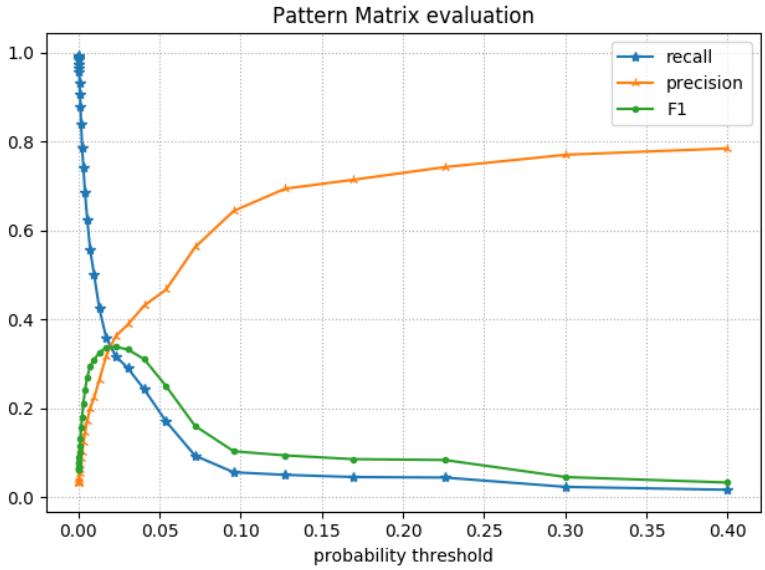


Figure 3.4: Pattern Matrix criterion evaluation. By adjusting the probability threshold we can balance between high recall and high precision.

For very low values of the probability threshold (close to 0) the criterion correlates almost every given pair of hits. This is the reason for high recall as all L0 hits are taken into account. It is clear that unlimited correlation results in low precision as the background noise is also being correlated.

With the increase of the probability threshold, we can see a quick increase of the precision with recall decreasing accordingly. It is caused by the fact that the criterion becomes more selective. Because the probability of correlating event hits with background noise is quite low (see Figure 3.2), the noise is being filtered out by the criterion. By further increasing the probability threshold, we limit the selectivity mostly to L1 hits which occur in very short distance with very low time difference ($\Delta T = 10ns$). High selectivity results in high precision, however a low recall value suggests that large amounts of L0 hits are also being filtered out by the criterion.

Because Pattern Matrix criterion performs significantly better than the Match 3B criterion, it will be used as part of the data processing pipeline in this thesis.

3.4 Defining the probability threshold value for the Pattern Matrix criterion

To define the reasonable range of the values of the probability threshold for the correlation criterion, we will again use the F-measures.

The Equation 3.1 for the F_1 score can be generalized by defining a β parameter. Then, by adjusting the β parameter, we can balance between higher recall or precision. The general formula for the F_β-score is the following:

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \tag{3.2}$$

We can define additional F measures like F_2 measure, which focuses more on recall than precision, and $F_{\frac{1}{2}}$ measure, which weights precision higher than recall. By evaluating this measures we can check how the criterion performs for different probability threshold values. By optimizing the F_2 measure the criterion should output more, however less precise correlations whereas for optimized $F_{\frac{1}{2}}$ measure the correlations should be more restricted but with significantly higher loss in event hit information.

Thus, for the pipeline evaluation we can assume that the range of probability threshold values for the correlation criterion should vary between peaks of F_2 and $F_{\frac{1}{2}}$ measures.

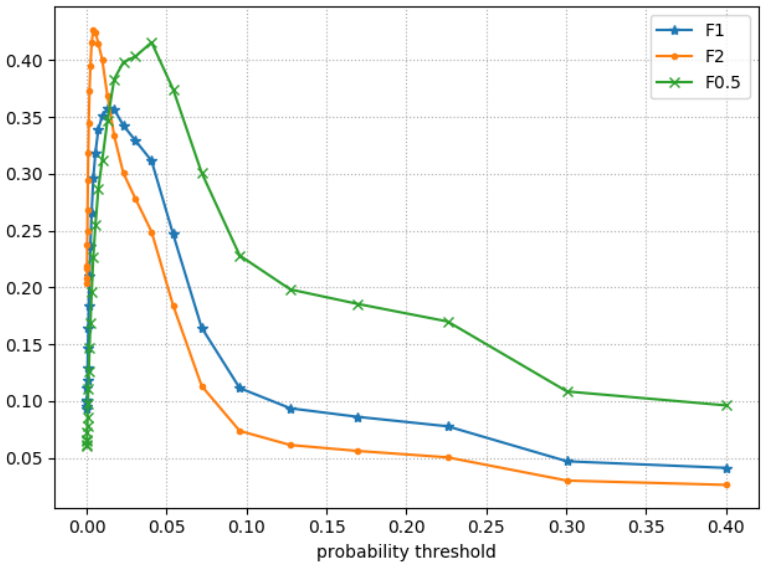


Figure 3.5: $F_{\frac{1}{2}}$, F_1 and F_2 scores of correlation criterion for given data set.

Figure 3.5 presents results of different F measures. For given data set F_2 reached its maximum value for probability threshold equal to 0.004, F_1 for threshold equal to 0.01,

and $F_{\frac{1}{2}}$ for 0.04. Finally we assume that for satisfying results, the probability threshold value of the pattern matrix criterion should vary between 0.004 and 0.04 depending on what is the intention of the end user.

Chapter 4

Graph community detection

In this chapter, we present the design and implementation of the community detection step in the pipeline. The community detection step groups the pairs of correlated hits from the correlation step into communities of correlated hits.

The output of the correlation step, which expresses correlations between pairs of L0 hits, can be interpreted as a graph adjacency matrix. In this graph, L0 hits are represented by vertices and correlations by undirected edges connecting these vertices. In this chapter, we include a brief introduction of the elements of graph theory and the principals of community detection methods. Next, we select two promising community detection algorithms and evaluate their performance on the simulated data set to select the algorithm that will be used in the data processing pipeline.

4.1 Elements of graph theory

We consider a graph G as a pair of sets (V,E), where V is a set of vertices (called also nodes or points) and E is a set of edges (or links). An edge is *adjacent* to each of its endpoints. We call a graph *directed* when edge is an ordered pair of vertices (u,v) beginning at u and ending at v and *undirected* otherwise. Graphs can be also *weighted*, i.e. there are real numbers associated to each edge of the graph called the *weight* of the edge. In this paper we denote the number of vertices as n and the number of edges as m. We call vertices *neighbors* when connected by an edge and the number of neighbors of a single vertex is called its *degree* indicated as k_v for a given vertex v.

4.1.1 Community detection

When analyzing the structures of graphs, it is possible to notice irregular concentrations of densely connected vertices with fewer external links (see Figure 4.1). This can lead to assumptions that there are vertices with common properties and roles in graph. These groups are called *communities* or *clusters* [12].

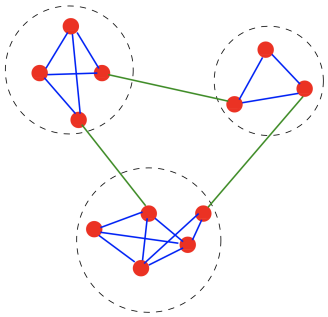


Figure 4.1: A graph with three communities denoted by the dashed circles [12]

Graph communities provide a great amount of information about the graph itself. When analyzing them, we can reveal additional properties of certain groups or even investigate relations between them. In fact, communities can be found in many networked systems from biology, chemistry, engineering, sociology etc. (e.g. members of families, species of animals or world wide web) [12].

4.1.2 Graph Communities metrics

The aim of community detection algorithms is to partition the graph into K mutually exclusive groups, where each vertex belongs to one community only. Considering C as a subgraph of graph G, where $|C| = n_c$ and $|G| = n$, for each vertex v we can define the *internal* degree k_{int}^v and *external* degree k_{ext}^v as the number of edges connecting v to other vertices of C or to the rest of the graph, respectively. If $k_{ext}^v = 0$, then the vertex is connected only to the neighbors within subgraph C (and should be a part of this cluster) and if $k_{int}^v = 0$, then the vertex is disjoint from C (and should be considered as a part of other cluster). Also, we can define internal degree k_{int}^C as the number of internal edges of subgraph C and k_{ext}^C as the number of edges connecting C to the rest of the graph. Then, we can define the *intra-cluster density* $\delta_{int}(C)$ as the ratio between the number of internal edges of C and the number of all possible internal edges:

$$\delta_{int}(C) = \frac{k_{int}^{C}}{n_c(n_c - 1)/2} \tag{4.1}$$

The *inter-cluster density* $\delta_{exp}(C)$ is the ratio between the number of edges connecting C to the rest of the graph and maximum number of inter-cluster edges possible:

$$\delta_{ext}(C) = \frac{k_{ext}^{C}}{n_c(n - n_c)} \tag{4.2}$$

Then, to consider C as a proper community, we expect $\delta_{int}(C)$ to be larger than $\delta_{ext}(C)$, which signifies that the density of edges connecting vertices in the community is significantly larger than the density of edges connecting C with the rest of the graph. A good balance between sufficiently large $\delta_{int}(C)$ and small $\delta_{ext}(C)$ is the goal of most community detection algorithms.

To the quality of the output of a community detection algorithm is, Newman [13] proposed a quality function called modularity. Firstly used as stopping criterion for the Girvan-Newman algorithm [14], modularity became a fundamental factor for modularity-based clustering methods. Modularity function Q can be expressed as follows:

$$Q = \frac{1}{2m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{2m}) \delta(C_i, C_j) \tag{4.3}$$

where A_{ij} is the weight of the edge between v_i and v_j, k_i is the degree of v_i, C_i is the community of v_i and $\delta(C_i, C_j)$ yields 1 if v_i and v_j are in the same community ($C_i = C_j$), zero otherwise.

4.2 Selection of the graph community detection methods

Because the task of partitioning a graph is considered as an NP-hard problem [15], there have been many approaches proposed to effectively divide the graph into communities [12]. Depending on assumptions, type of data and goals, different methods can be applied balancing between their complexity, accuracy and performance. The goal of this section is to investigate graph community detection algorithms and consider their feasibility for the community detection step of our pipeline. To choose relevant methods we decided to form three criteria that the algorithms must meet:

1. *unsupervised* - The algorithm should be free of parameters related to the size or the number of communities to detect, as the number of hits related to neutrino events

vary greatly and we do not know in advance how many events will be present in the data.

2. *scalable for large graphs* - As the KM3NeT telescope produces data at a high data rate, the algorithm must have with very good scalability. Algorithms with high computational complexity will be unable to process the data in a reasonable amount of time.
3. *applicable in parallel* - An important aspect of the data processing pipeline is the performance. It should be possible to implement the community detection algorithm on parallel computing platforms, such as GPUs.

During the research and literature study we have considered various methods of graph community detection considering their complexity and feasibility for implementation for the KM3NeT detector. In next paragraphs we briefly describe different types of considered methods and judge their usability for the desing of the pipeline.

Traditional methods Although traditional graph partitioning methods provide a good background for graph processing, they require a predefined the number of desired clusters, often focusing on graph bipartitioning [12]. Hierarchical algorithms based on vertex similarity may seem to be a good choice for graph partitioning as they do not require to input the size or number of clusters [16]. However, they are usually based on a vertex similarity measure not taking actual connections into account, which is not applicable to the case of the designed pipeline.

Divisive algorithms. Divisive algorithms aim to reveal communities by successively removing edges, calculating their centrality or betweenness [12], and estimating their importance in the network. Such removal of edges results in gradual partitioning of graphs giving similar output to the hierarchical clustering. Divisive algorithms gained popularity due to their implementation adjustability and straightforward approach for graph processing [12]. Although Moon et al. [17] managed to implement the Girvan-Newman algorithm in parallel, they did not achieve sufficiently good performance as the algorithm requires a large number of iterations and does not scale well to large graphs. According to their work, the proposed solution processed a graph of 5000 nodes and 30,000 edges in time of 30 minutes, which is too slow for our use case.

Function optimization methods. Community detection algorithms based on the optimization of the modularity function (see Equation 4.3) are commonly used as they offer a good balance between performance and accuracy. The Louvain algorithm offers decent performance, scalability, and accuracy. Naim et al. [18] managed to successfully implement this algorithm for parallel processing of large graphs, composed of several millions of vertices. This makes the algorithm worth considering in the case of the KM3NeT pipeline. It is also worth to consider the Constant Potts Model (CPM) [19], which is similar to the Louvain algorithm. CPM offers resolution-limit-free method, which may be beneficial as we are looking for relatively small communities in large graphs.

Conclusion For the design of the pipeline, the Louvain and CPM modularity-based methods are selected as promising candidates. In the next section, we explain these methods in more detail and evaluate their performance to select an algorithm for graph community detection of the data processing pipeline.

4.2.1 Function optimization methods

To evaluate how well the graph is partitioned, Newman [13] proposed a measure called *modularity* Q (described in Section 4.1.2). Initially considered as a stopping criterion of the Girvan and Newman algorithm, this measure become a popular basis that drives a large collection of algorithms that aim to optimize modularity, while manipulating the graph partitioning [12]. In this section, we explain two optimization methods: the modularity-based Louvain algorithm and Constant Potts Model.

Louvain algorithm

One of the most common methods based on the optimization of modularity was introduced by Blondel et al. [20]. This greedy optimization method is iteratively performing two steps: optimizing local modularity by assigning adjacent nodes into common communities and aggregating nodes that belong to the same community, building a new graph, where new nodes represent communities. The algorithm runs until reaching the maximum of global modularity factor. The complexity of this method is not known, but it is considered by the authors to run in time $O(n \log n)$ mostly consumed by the first iteration.

Constant Potts Model

Modularity-based community detection methods offer a good balance between accuracy and complexity. However, because of the fact that modularity function is sensitive to the

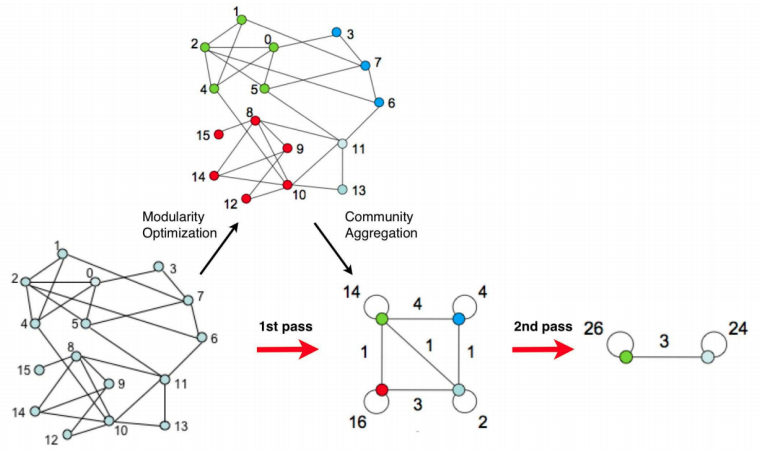


Figure 4.2: Visualization of the steps of the Louvain algorithm [20]

size of the graph (see Equation 4.3), it may occur that the results of the partition might be inaccurate or counter-intuitive [20] (see Figure 4.3).

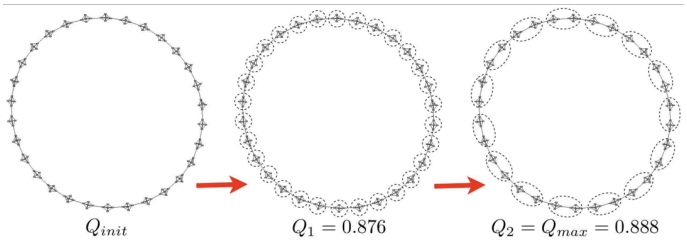


Figure 4.3: Result of modularity-based community detection algorithm application [20]. The first pass of the algorithm defines natural partition of the network of 30 cliques. The second pass, aiming for global modularity maximization, combines cliques into groups of two.

Traag et al. [19] proposed resolution-limit-free community detection approach, termed as Constant Potts Model (CPM), that also aims to optimize given function but on the basis of a constant factor γ. In general, the CPM cost function can be expressed by simple equation:

$$H = -\sum_{ij}(A_{ij} - \gamma)\delta(C_i, C_j) \tag{4.4}$$

where A_{ij} is the weight of the edge between vertices v_i and v_j, and γ is the constant value parameter. Traag et al. [19] assumes optimal constant value $min_{ij}A_{ij} \leq \gamma \leq max_{ij}A_{ij}$.

By denoting the number of edges inside community c by $e_c = \sum_{ij} A_{ij}\delta(C_i, c)\delta(C_j, c)$, and the number of nodes in community c by $n_c = \sum_i \delta(C_i, c)$, Equation 4.4 can be expressed

as:

$$H = -\sum_{c} e_c - \gamma n_c^2 \tag{4.5}$$

CPM, aiming into function minimization, tries to maximize the number of internal edges keeping the community sizes relatively small. It was also shown that CPM performs better than modularity-based methods [19].

4.3 Evaluation of selected methods

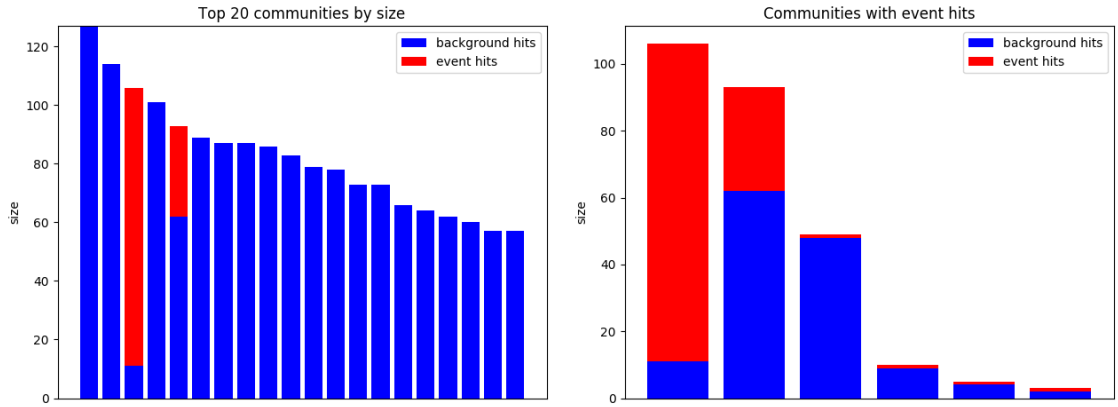
The goal of this section is to evaluate and compare two graph community detection algorithms: Louvain and CPM. For the purpose of the evaluation we used an open-source implementation that offers both approaches [21].

Both algorithms processed the same data set of a group of 130 event hits (visualized in Figure 2.4) and arbitrarily selected 4900 background noise hits. The data sample was chosen in such a way that the time slices of event group and background noise overlap with each other and the amount of background hits surpasses the amount of event hits to underline the need for resolution-limit-free approach. To create the graph, we used the adjacency matrix created using the correlation criterion proposed in Section 3.2 with a probability threshold of 0.01, as it gave the best balanced F1 score. The CPM resolution parameter γ was set to 0.1, however its proper adjustment will be explained in the next sections.

As a result of the evaluation, we expect that the algorithms divide the data set into numerous communities, preferably gathering event hits together. Because of the correlation criterion, neutrino event hits, treated now as graph vertices, should be much more densely connected to each other than to background noise hits.

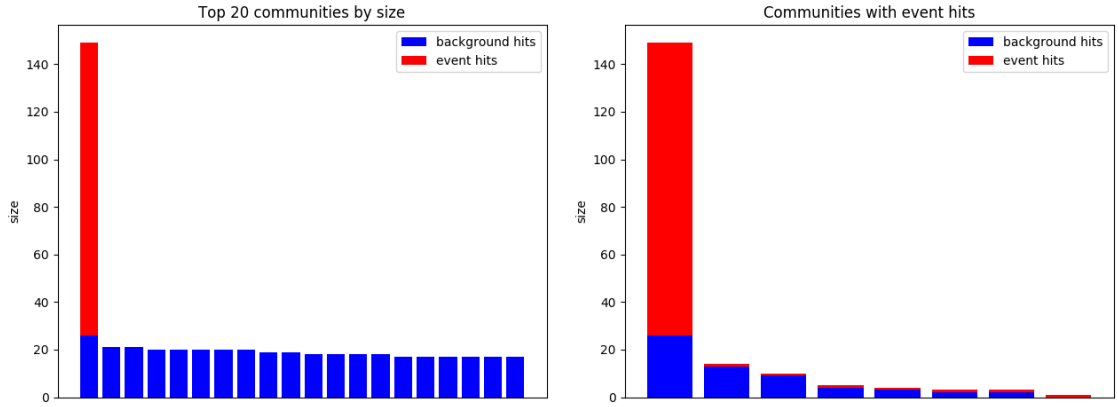
The Louvain community detection resulted in defining around 1300 communities. Figure 4.4 presents two figures with communities ordered by size. The first figure depicts top 20 largest communities, among which we can see two communities that gathered records from prepared neutrino event whereas the second figure presents all communities containing neutrino event records. We can see that the event hits were mostly gathered into two communities, however these communities do not stand out with their sizes among other communities containing background noise only. It is hard to distinguish valuable communities among the Louvain output.

In comparison, the CPM algorithm defined similarly around 1500 communities, however their amount highly depends on the value of resolution parameter. Like previously, Figure



(a) **(b)**

Figure 4.4: Evaluation of the Louvain algorithm. Left: top 20 communities by size. Right: communities with event hits only. We can see that event hits were mostly gathered in two communities which can be also found in top 20 biggest communities.



(a) **(b)**

Figure 4.5: Evaluation of the CPM algorithm. Left: top 20 communities by size. Right: communities with event hits only. Almost all event hits were gathered in one community which also stands out with its size. All other communities are significantly smaller.

4.5 presents the classification result of the CPM, which looks much more promising. Here almost all the event hits were gathered in one community that clearly stands out among other, much smaller communities.

The advantage of the CPM algorithm is the fact that it keeps backgorund noise communities significantly smaller than the community that contains event hits. In contrary, Louvain algorithm outputs a bunch of large communities among which it is hard to distin-

guish communities with event hits.

The results of the evaluation showed that Constant Potts Model (CPM) performs exceptionally well when it comes to detect communities with event hits correlated by the new criterion based on probability matrix. Because CPM can be parallelized and implemented in the same way as Louvain algorithm [18], we will use it for the community detection part of the data processing pipeline presented in this thesis.

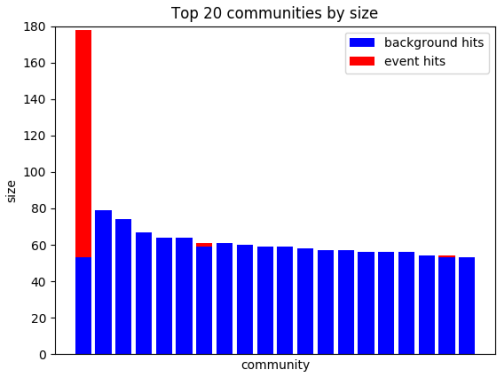
Adjusting the CPM resolution parameter

As [19] mentions, the value for the CPM resolution parameter for an unweighted graph should be in the range of values between 0 and 1.

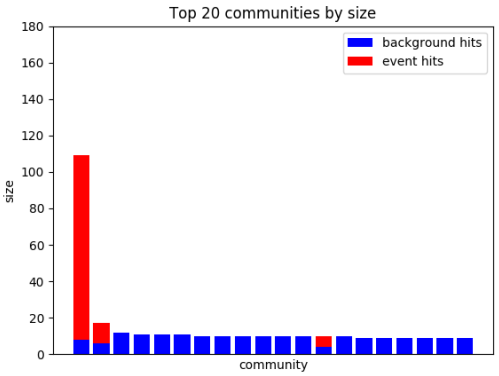
Figure 4.6 presents how the CPM resolution parameter γ influences the output of the community detection phase of the pipeline. There are two observations that should be taken into account: how the resolution parameter affects on the sizes of defined communities and also the number of defined communities.

For low values of γ (less than 0.1) the output consists of fewer and larger communities. Also, we can see that great majority of event hits are gathered in the largest community as it was discussed previously. When increasing γ, the number of communities increases and community size decreases. In the result the event hits of evaluated group were distributed among several communities, however it is worth to mention that communities that contain event hits contain less noise then communities formed with lower CPM γ value.

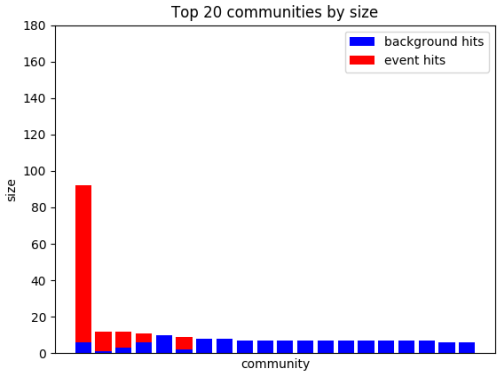
In the next chapter we will discuss how the probability threshold of Pattern Matrix criterion cooperates with the CPM resolution parameter γ and we will try to find the best configuration of these two parameters that gives the most optimal output.



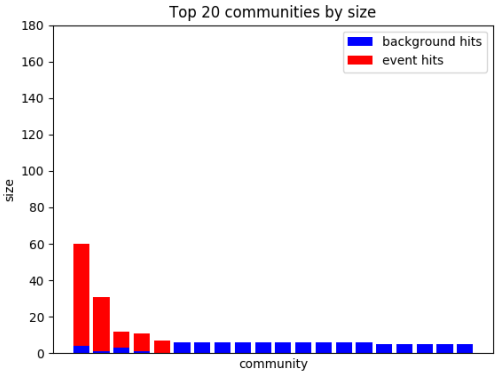
(a) $\gamma = 0.01$; communities: 256



(b) $\gamma = 0.3$; communities: 1429



(c) $\gamma = 0.5$; communities: 2102



(d) $\gamma = 0.7$; communities: 2306

Figure 4.6: Influence of the CPM resolution parameter γ on the output of the community detection phase.

Chapter 5

Classification

The previous chapter showed that the pipeline is able to reliably group records into communities, gathering neutrino event hits mostly into one community and background noise hits in other communities. In this chapter we will define the third phase of the pipeline: the classification.

The phases of correlation criterion and community detection offer two adjustable parameters: a probability threshold of the Pattern Matrix criterion and a resolution parameter γ of the CPM algorithm. In this chapter we will choose such a configuration that gives the most optimal output. Then, based on the output of that configuration we will explain how the classification phase works.

Because the simulated neutrino event group sizes vary from small ones of few hits to large ones up to hundreds of hits (see Figure 2.1a), it is necessary to prepare a data set that could be used for correct evaluation. Table 5.1 describes the prepared data set which will be used in this chapter.

Table 5.1: The data set prepared for the pipeline adjustments and evaluation

sample size	1,000,000
number of event hits	7,190
number of background hits	992,810
number of simulated events	510
number of simulated events (group size $>$ 50)	35

5.1 Adjusting the pipeline configuration

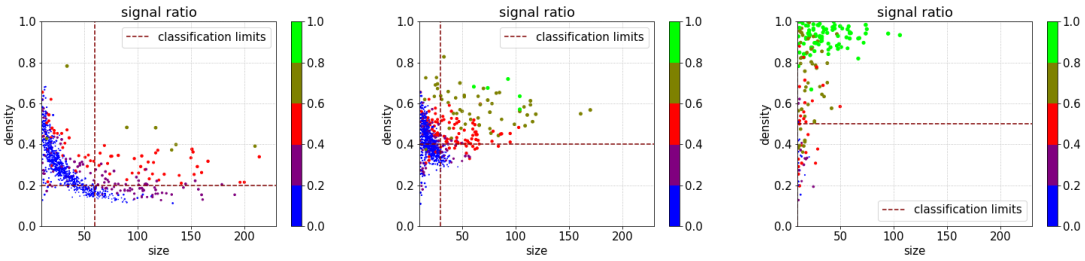
To check how the pipeline behaves under different conditions, we evaluated the data sample from Table 5.1 with different pipeline configurations. In Section 3.4, we discussed what should be the value range for the probability threshold of the correlation criterion. We also know what should be the value range of the CPM resolution parameter. In this section we will show and explain how these parameters together affect the output and how it can be used to make a classification.

By adjusting the probability threshold value of the Pattern Matrix criterion, we influence on the number of correlations which then affects the number of edges in the graph. The CPM resolution parameter expresses how likely the graph nodes are to change their community assignment dividing the graph into communities with different sizes. In the result, such management of numbers of graph edges and community sizes affects on the inter-cluster density discussed in Section 4.1.2. Thus, for the evaluation we will also take into account the density of the communities formed by the pipeline.

Figure 5.1 depicts how the density and size of the communities are distributed for different configurations of probability threshold (PT) and CPM resolution (γ). Each point represents single community and its color defines the signal ratio of event hits to the community size. Because the pipeline forms numerous small communities which density is skewed (for very small communities the density value is highly sensitive to small changes in the amount of interconnecting edges), we only consider communities of size greater than 10.

With the increase of the probability threshold value (PT), the pipeline becomes more selective as only hits that occurred very close to each other in time and distance are being considered as correlated (for very high PT values, the filtration is similar to L1). Although the PT value can serve as a good filtration parameter, there is a risk of loosing a lot of event hits as they remain uncorrelated or form unreasonably small communities. Please note that for high PT value (PT = 0.04) the number of defined communities almost reached the number of investigated data records.

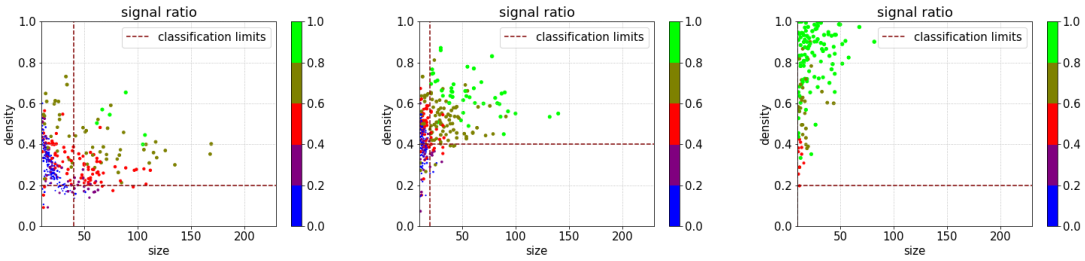
The Figure 5.1 shows that both PT and γ values affect on the sizes and densities of formed communities. We can see that the communities with low signal ratio (blue/purple points) have low size and density compared to communities with higher signal ratio (green points) which form bigger and denser groups. We use this relation to define a simple classifier based on community size and density.



(a) PT = 0.004; $\gamma = 0.02$
communities: 82,042
event hits: 6,882

(b) PT = 0.004; $\gamma = 0.1$
communities: 143,336
event hits: 6,502

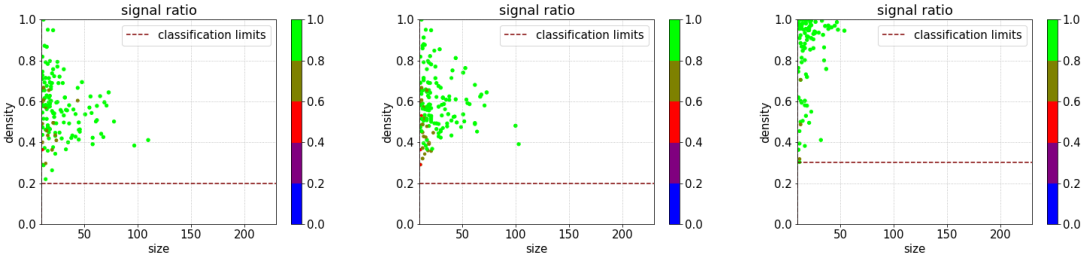
(c) PT = 0.004; $\gamma = 0.7$
communities: 396,253
event hits: 3,932



(d) PT = 0.01; $\gamma = 0.02$
communities: 388,754
event hits: 5,444

(e) PT = 0.01; $\gamma = 0.1$
communities: 405,579
event hits: 5,023

(f) PT = 0.01; $\gamma = 0.7$
communities: 529,695
event hits: 2,973



(g) PT = 0.04; $\gamma = 0.02$
communities: 827,913
event hits: 3,501

(h) PT = 0.04; $\gamma = 0.1$
communities: 826,475
event hits: 3,542

(i) PT = 0.04; $\gamma = 0.7$
communities: 831,681
event hits: 1,811

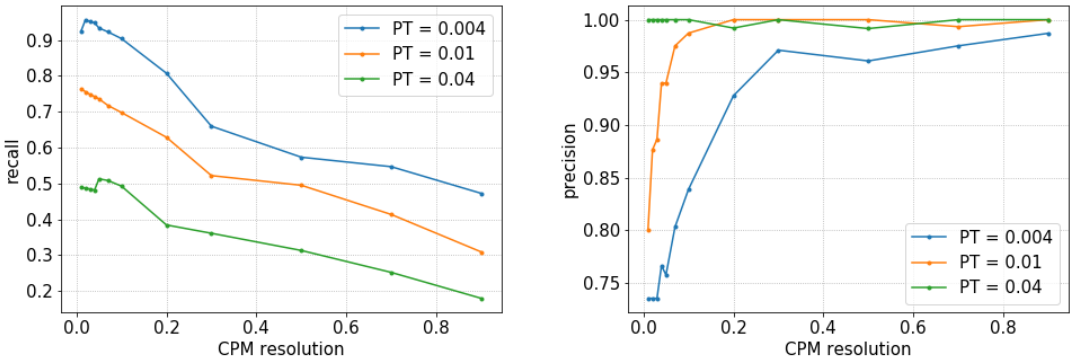
Figure 5.1: Pipeline configuration evaluation. Plots show the distribution of detected communities according to different pipeline configurations of pattern probability threshold (PT) and CPM resolution (R). Each dot represents single community and their color represents signal ratio i.e. the ratio of the number of event hits to given size of the community. Only communities of size greater than 10 are presented. Each configuration is described with number of formed communities and total number of possible neutrino event hits that can be retrieved from them. Dashed lines show the best classification setup for given configuration.

5.2 Classification of the communities

The goal of classifying communities is to reveal those that contain a sufficiently high signal to noise ratio, i.e. the number of gathered event hits to the community size.

For this thesis, we define the following classification method: all hits inside of a community having size and density exceeding predefined thresholds are marked as *event* hits, while all other hits are marked as *noise* hits. Then, because we operate on labeled data, we use a confusion matrix to check how well this classification performs. Figure 5.1 shows, for each considered configuration PT and γ, dashed lines indicating optimal size and density threshold pair that that gives the best result based on the F_1 score

In order to find the best configuration settings of the pipeline we compared the most optimal classification outputs for different configurations of PT and γ. For the evaluation we took into account the *recall* to check the ratio of classified event hits in the data set, the *precision* to check the ratio of actual event hits among the classified ones, and the *F-scores* to choose the best balance between recall and precision.



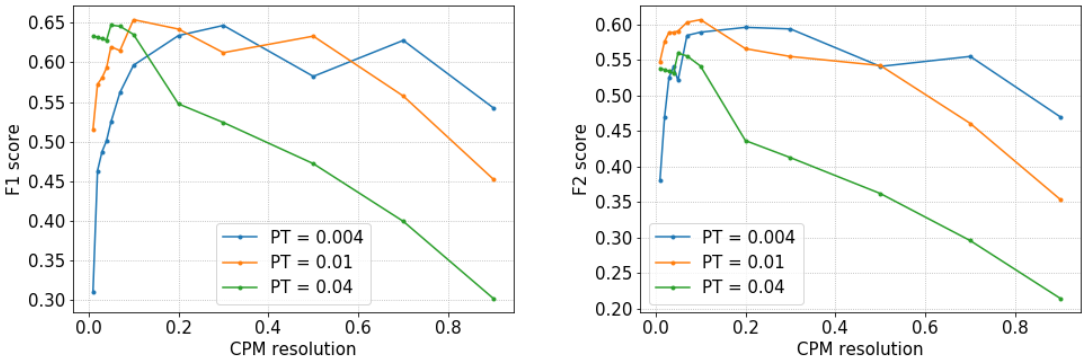
(a) Highest possible recall score for given CPM resolution.

(b) Highest possible precision score for given CPM resolution.

Figure 5.2: Plots show highest possible values of recall and precision out of all configurations of PT and γ.

Figure 5.2 presents maximal possible values of recall and precision that could be achieved for different pipeline configuration settings. Based on the plots, we can see that the probability threshold (PT) plays significant role in the quality of the output. For the complete range of the CPM resolution values, low PT values offer higher recall and lower precision which seems reasonable as low PT is more likely to correlate distant hits. In contrast, the pipeline based on higher PT values becomes more precise as we are closer to L1 filtration.

However the precise output might occur unacceptable as we are losing a lot of neutrino event hits, which is expressed by significantly lower recall.



(a) Highest possible F_1 score for given CPM resolution.

(b) Highest possible F_2 score for given CPM resolution.

Figure 5.3: Plots show highest possible values of the F score measures out of all configurations of PT and γ.

To obtain a good balance between recall and precision, it is worth to make use of the F-measures. Figure 5.3 depicts the F_1 and F_2 scores based on achieved recall and precision. Figure 5.3a suggests that there are three possible configurations that give optimal score as the peaks of the F_1 score function reach similar values. However, if we take a look at the Figure 5.3b, which depicts the F_2 score function, we can see that there is one dominating peak (at PT $= 0.01$, $\gamma = 0.1$). What is more interesting, the peak of the F_2 function is being reached for the same configuration as one of the peaks of F_1 function which makes that configuration worth to consider. Also, as F_2 score function weights recall more than precision, it makes this configuration an even better candidate as the risk of missing neutrino events is reduced.

Finally, the most optimal output was achieved for the following configuration (also depicted in Figure 5.4):

- correlation probability threshold (PT) $= 0.01$
- CPM resolution (γ) $= 0.1$
- community density threshold $= 0.4$
- community size threshold $= 20$

By changing the probability threshold of Pattern Matrix criterion and resolution parameter γ of the CPM, we can tune the pipeline output quality. For further processing we will

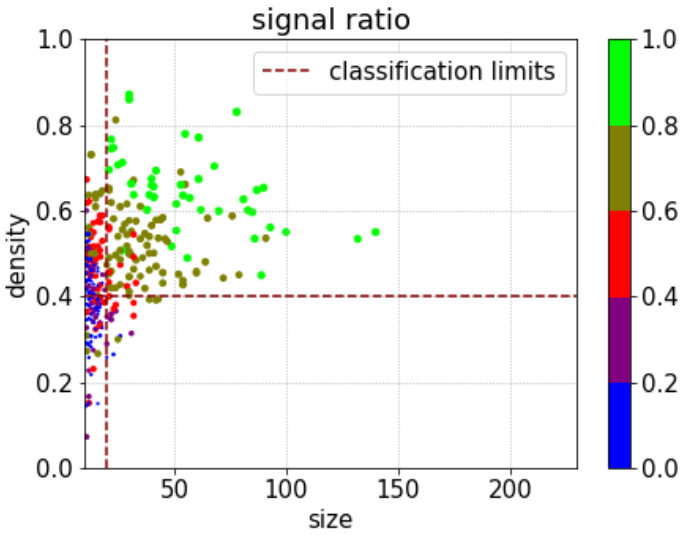


Figure 5.4: Final configuration of the pipeline. Dashed lines show the density and size limits above which the communities are classified as the positive ones.

use the most optimal configuration. In the next chapter, we will use this final configuration to check, how well the pipeline performs when classifying time slices of hits.

Chapter 6

Pipeline evaluation

Having the complete pipeline designed, implemented and optimized, it is worth to focus on its evaluation. In this chapter, we will discuss how the pipeline could be used to process time windows of telescope data, processing the streaming data while offering better filtration level than previous methods [11]. We also present the pipeline performance estimating its feasibility for real-time processing all the data from the KM3NeT telescope.

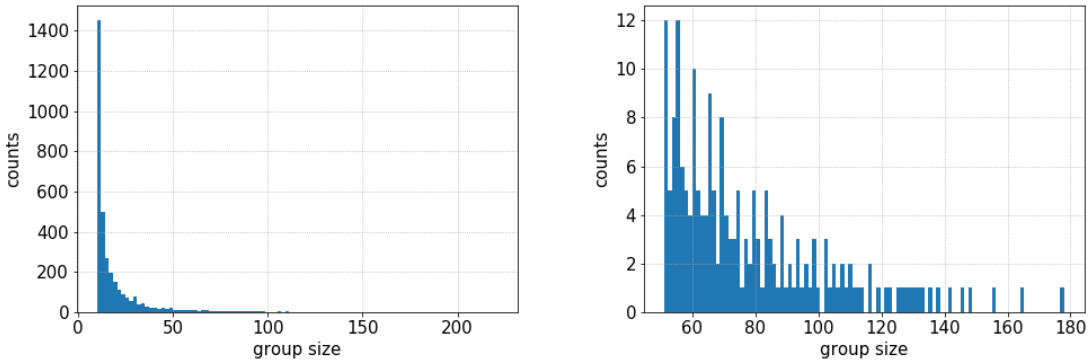
6.1 Classification of time slices

In this section we will show how the pipeline can be used for the time slices classification. The goal is to check what is the minimal neutrino event group size that is still detectable by the pipeline. By the time slice we assume a fixed size set of sequential hits among which the group of event hits is included.

Data set preparation

To evaluate the pipeline, a data set was prepared consisting of over 3,000 event groups combined with corresponding background noise hits forming together a time slice. The distribution of prepared group sizes (see Figure 6.1) is similar the distribution of complete simulated data set. The data set is organized in time slices, each containing 2,000 records (event and background hits).

The goal of this evaluation is to see if the pipeline is able to detect event hits in every given time slice and what is the minimal detectable event group size. Because the number of L1 hits is highly correlated with event group size (see Figure 2.2a), we will mostly focus on small groups (of size less than 50). As larger groups contain significantly more highly correlated L1 hits, it can be assumed that they are easier to detect.



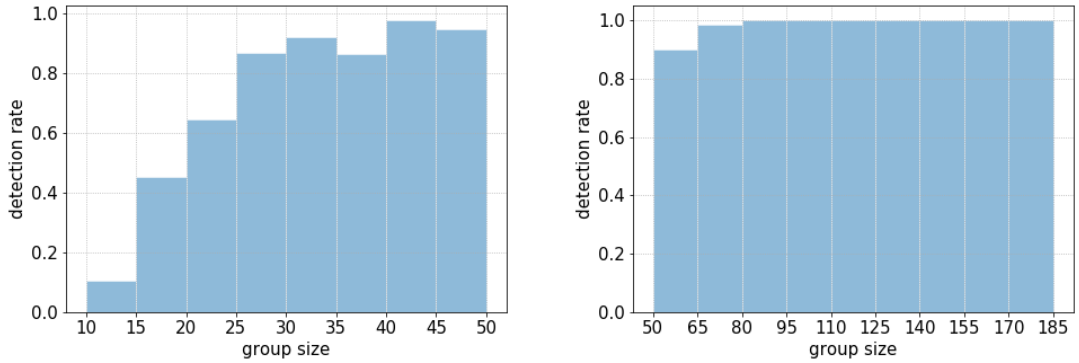
(a) Group size distribution

(b) The tail of larger groups

Figure 6.1: Event group size distribution of the prepared data set. Left: group size distribution of data set. Right: group size distribution of groups larger than 50 hits.

Time slice classification

The goal of time slice classification is to decide if a set of records contains groups of hits that might be related to neutrino event. If the pipeline is able to classify any defined community as related to a neutrino event, then the time slice is also classified as such. Each time slice contains 2,000 hits (including event hits) and is classified using the pipeline configuration defined in previous chapter.



(a) Detection rate for small event groups

(b) Detection rate for large event groups

Figure 6.2: The figure presents the average detection rate of correctly classified time slices for given event group size. Detection rate equal to 1 means that all neutrino event groups of given size were classified successfully.

Figure 6.2 presents how precise the classification is for a group size. We can see that for

groups of size 40 and higher we are able to correctly classify all groups. For smaller groups the precision of classification decreases but still the detection rate of positively classified slices remains at high level until the size of 20. For groups smaller than 20, the majority of events is not classified (the detection rate is below 0.5).

6.2 Window processing

In the real world scenario the pipeline would be used for processing the data streamed directly from the detector strings. The pipeline should then serve as a filtration module judging the usefulness of storing a given time slice for further processing.

To evaluate the pipeline we prepared several data sets each consisting of 100,000 records. Each data set contained hits of one neutrino event group and a set of corresponding background noise hits. The goal is to process each given data set in a form of sequential slices, called windows, to check if the pipeline is able to correctly indicate the windows that contain neutrino event hits. To not accidentally divide the event signal but two contiguous windows, we arbitrarily chose the window size of 1000 records with 500 records step for each sequence. The pipeline configuration remains unchanged since last optimization presented in Chapter 5. Then, we will compare the output of the pipeline with the current state-of-the-art filtration based on L1 hits [1].

Figures 6.3, 6.4 and 6.5 make a comparative presentation of signal processing between L1 and pipeline filtration. For each of the given time windows, the algorithm calculates the number of actual neutrino event hits, the L1 triggered hits, and the hits classified by the pipeline.

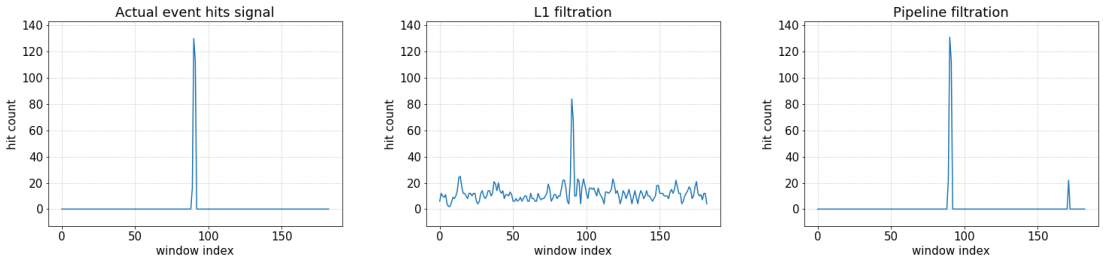


Figure 6.3: Window processing of the neutrino event group of **130** hits.

Since the L1 filtration is suitable for indicating very large event groups, its efficiency decreases when processing smaller ones. At some point it is impossible to distinguish the

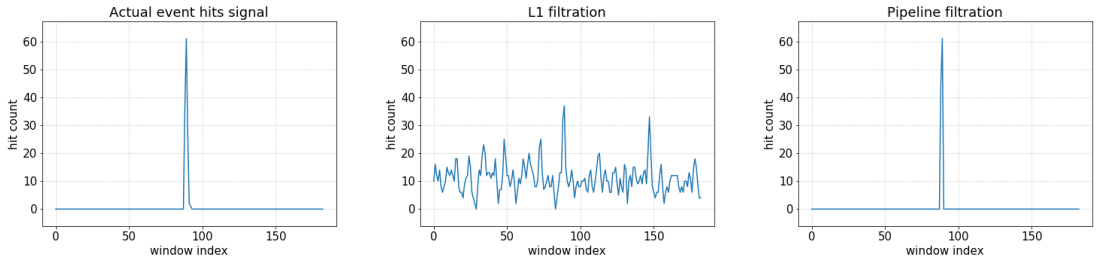


Figure 6.4: Window processing of the neutrino event group of **63** hits.

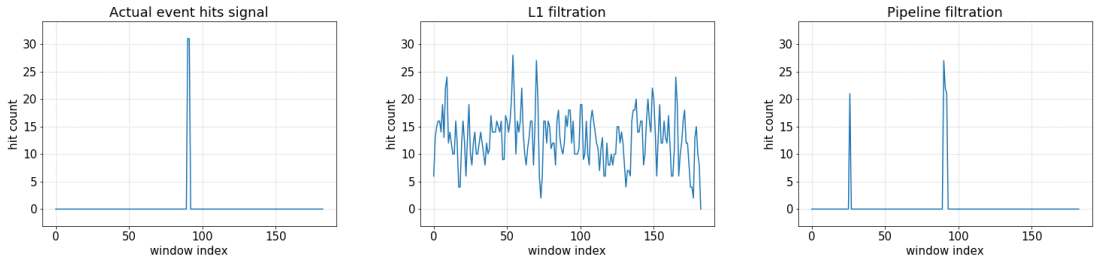


Figure 6.5: Window processing of the neutrino event group of **31** hits.

peaks of actual neutrino event signal out of the background noise that is also able to trigger the L1 hits.

In comparison, the pipeline classification seems to perform much better. The background noise is much less likely to trigger positive classification as it happens for the L1 filtration. Also, the neutrino event signal peaks are preserved with only a small loss of the amount of data records.

In the result, the designed pipeline gives significantly better output when it comes to neutrino event signal filtration. Potential candidates of the time slices indicated by the peaks are much more relevant than ones revealed by L1 filtration.

6.3 Runtime performance estimation

In order to provide estimations for real-time data processing, we had to analyzed the runtime of the pipeline. We prepared five data sets that represent five time windows of different sizes containing respectively 10^3, 10^4, 10^5, 10^6 and 10^7 hits.

We have implemented our pipeline on the GPU using NVIDIA CUDA to enable fast

processing. For this evaluation, we used *nvprof*[1] profiling tool to calculate the processing time of each phase of the pipeline. The performance tests were executed on Titan X GPU with 12 GB of memory (Pascal architecture), and with CUDA Compute capability 6.1.

The performance evaluation took into account different phases of the pipeline to check whether any part is more sensitive to changes in data volume than others. The pipeline consists of three stages: correlation, community detection and classification.

Figure 6.6 presents the results of performance analysis for different data volumes. We can see that for window size of 10^5 records and larger, the execution time appears to grow linearly with the size of the data set.

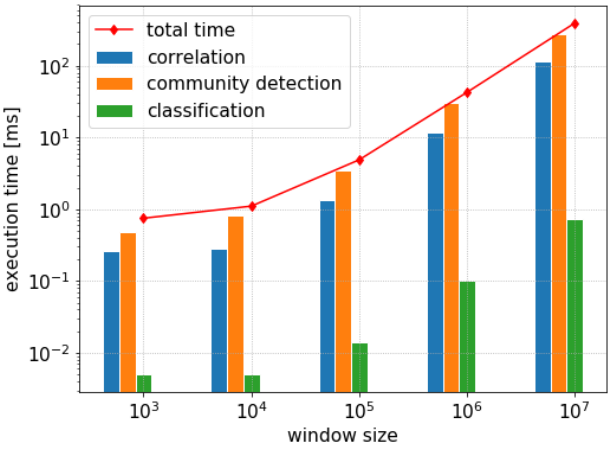


Figure 6.6: The pipeline runtime for different window sizes.

To provide performance estimation for real-time data processing, we defined a throughput parameter of the pipeline. This parameter, based on total processing time of the pipeline and data set volume, measures how many data records we are able to process in time of one millisecond.

Figure 6.7 presents the throughput estimations for different data sets. It appears that the throughput is not constant and depends on the size of the data set. For small data sets the throughput is significantly lower in comparison to larger data sets.. At size 10^5 and greater, the throughput seems to reach stabilization which also explains the linear increase for total time execution depicted in Figure 6.6.

Finally, having the throughput defined, we can estimate how many GPUs are necessary for real-time data processing. Taking into account that the simulated data set contains

[1] docs.nvidia.com/cuda/profiler-users-guide/

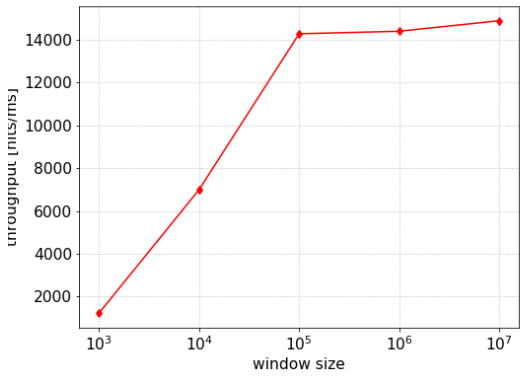


Figure 6.7: The pipeline throughput for different window sizes. The throughput expresses number of processed hits per millisecond.

450,000 records per millisecond and assuming that the throughput that can be achieved by the current pipeline implementation is around 14,000 records per millisecond, means that real-time processing requires at least 33 GPUs (Titan X or similar).

Chapter 7

Future work

The goal of this work was to check the feasibility of the pipeline for the event signal filtration which was successfully presented. Because the design consists of several phases, there are also different directions for the future work.

Correlation Criterion Regarding the correlation criterion, the approach for creating the pattern matrix could be replaced by more accurate solution. The current approach assumed arbitrarily chosen time and distance limits based on the analysis of the signal correlation for different groups of simulated neutrino event hits. The Pattern Matrix took into account the probability of correlation between two event hits for given distance and time difference only but there are more features that could be considered, e.g. by analyzing of the causally triggered DOMs or PMTs. Also, it is worth to consider another scoring approach. Calculation of the probability of the positive correlation resulted in criterion which thresholds were adjusted in a range of very low values making that approach highly sensitive to any configuration changes. One solution would be the use of the time-over-threshold (ToT) values for weighted correlations to highlight the high energetic correlations.

Community Detection Since the Constant Potts Model is similar in implementation to the Louvain algorithm, it is worth to consider the improvements of the community detection phase by analyzing different implementations of the latter one. It seems to be worth to try the application of the Louvain heuristics mentioned in [22], where authors propose several approaches improving both accuracy and performance of the algorithm. Current implementation of community detection phase is straightforward and not well optimized. That approach highly affects the processing time as it was depicted in the

Figure 6.6. However, as [18] proposed, it is possible to improve the calculation performance by preprocessing the neighboring communities in advance in order to reduce the runtime.

Classification The final phase of the pipeline: the classification of the communities, is currently based on predefined size and density limits which were chosen in such a way to maximize the balance between recall and precision of the output. However, we could probably gain better results by analyzing different classification models e.g. logistic regression or decision tree.

Chapter 8

Conclusion

This thesis proposes the design and implementation of the data processing pipeline for the KM3NeT neutrino detector. In this work, we showed how the raw signal from the detector can be processed and filtrated in three phases. To conclude this thesis, we will recall again the research questions from Chapter 1 and answer them firstly focusing on secondary questions to finally give an answer for the main research question.

RQ2: *Based on the input data, how to designate pairs of correlated hits?*

In Chapter 3 we showed that Match 3B correlation criterion proposed in [11] is too less restrictive to provide sufficient output for further processing. By analyzing the simulated data set we proposed a criterion that outperforms the previous work offering more optimal output. The investigation of the simulated neutrino event groups showed that hit correlations occur mostly in specific distance and time difference. Based on that knowledge we proposed a general pattern of mutual event hit correlations which could serve as a probability matrix offering an adjustment feature by managing the threshold value. That is how the first phase of the pipeline was formed. The output of the defined Pattern Matrix criterion is used to form a graph for the next phase of the pipeline.

RQ3: *What graph community detection algorithm would be feasible for revealing correlated groups of records?*

Chapter 4 focused on graph community detection algorithms. Among different types of methods we selected two promising candidates: the modularity-based Louvain algorithm and similar in implementation Constant Potts Model. The evaluation of the Louvain algorithm exposed its weaknesses in finding relatively small communities in large data set due to the resolution limit which depends on the graph size. It occurred that the Constant

Potts Model suited better to the needs of the pipeline. Comparative evaluation of the Louvain and CPM explained why the resolution-limit-free approach was required for the community detection phase. It was shown that for the optimal resolution parameter γ, the CPM is able to gather the event hits mostly in one community of size greater than the communities that consisted of background hits only.

RQ4: *Is it possible to label the communities based on whether they contain a neutrino event?*

For the last phase of the pipeline, in Chapter 5, we explained how the probability threshold of the correlation criterion and the CPM resolution parameter γ influence on the characteristics of the defined communities. We showed that communities with high signal ratio are tend to be larger and denser than the communities with low signal ratio. Then, we proposed simple yet effective approach for classifying the communities defining their minimal sizes and densities. Among different configurations and classification setups we chose the most optimal one that resulted in the highest F_1 and F_2 scores.

RQ1: *Is it possible to classify time slices of raw telescope data based on whether a neutrino event occurred?*

Finally, having the whole pipeline defined and optimized, we could focus on the main research question of the thesis. In Chapter 6 we evaluated the implementation of the pipeline under different conditions. Firstly, by processing the time slices, we checked what is the minimal neutrino event group size that is still detectable in the corresponding background noise. Then, we explained how the pipeline can be used in form of traveling window processing where each window in evaluated and classified. It occurred that the pipeline can serve as a great filtration phase that outperforms the state-of-the-art L1 filtering. In the end of the chapter we also checked the pipeline runtime considering different sizes of the data set to estimate the number of GPU units required for live processing.

Discussed eventual future work showed that each phase of the pipeline can be improved or replaced by alternative methods improving the runtime performance and increasing the output quality. All in all, this thesis showed that proposed data processing pipeline is able to support the detection of cosmic neutrino particles and gives a good background for further research and development.

References

[1] S. Adrian-Martinez *et al.*, "Letter of intent for KM3NeT 2.0," *J. Phys.*, vol. G43, no. 8, p. 084001, 2016. 1, 3, 10, 39

[2] B. Dasgupta and R. Laha, "Neutrinos in icecube/km3net as probes of dark matter substructures in galaxy clusters," *Phys. Rev. D*, vol. 86, p. 093001, Nov 2012. 2

[3] B. Herold, O. Kalekin, and J. Reubelt, "Pmt characterisation for the km3net project," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 639, no. 1, pp. 70 – 72, 2011. Proceedings of the Seventh International Workshop on Ring Imaging Cherenkov Detectors. 2

[4] C. Kopper, "A software framework for km3net," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 602, no. 1, pp. 107 – 110, 2009. Proceedings of the 3rd International Workshop on a Very Large Volume Neutrino Telescope for the Mediterranean Sea. 2

[5] W. Grimus and H. Neufeld, "Cherenkov radiation of neutrinos," *Physics Letters B*, vol. 315, no. 1, pp. 129 – 133, 1993. 2

[6] S. Fukuda *et al.*, "The super-kamiokande detector," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 501, no. 2, pp. 418 – 462, 2003. 4

[7] Y. Fukuda *et al.*, "Evidence for oscillation of atmospheric neutrinos," *Phys. Rev. Lett.*, vol. 81, pp. 1562–1567, Aug 1998. 4

[8] M. Aartsen *et al.*, "The IceCube neutrino observatory: instrumentation and online systems," *Journal of Instrumentation*, vol. 12, pp. P03012–P03012, mar 2017. 4

REFERENCES

[9] M. G Aartsen, "Evidence for high-energy extraterrestrial neutrinos at the icecube detector," *Science*, vol. 342, p. 1242856, 01 2013. 4

[10] G. Tamas and M. Lotze, "Km3pipe." `https://git.km3net.de/km3py/km3pipe`, 2019. 7

[11] B. Bakker, "Trigger studies for the Antares and KM3NeT neutrino telescopes," Master's thesis, University of Amsterdam, the Netherlands, 2011. 13, 14, 37, 45

[12] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3, pp. 75 – 174, 2010. 22, 23, 24, 25

[13] M. E. J. Newman, "Detecting community structure in networks," *The European Physical Journal B*, vol. 38, pp. 321–330, Mar 2004. 23, 25

[14] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002. 23

[15] K. Andreev and H. Räcke, "Balanced graph partitioning," in *Proceedings of the Sixteenth Annual ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '04, (New York, NY, USA), pp. 120–124, ACM, 2004. 23

[16] M. Dash, S. Petrutiu, and P. Scheuermann, "Efficient parallel hierarchical clustering," in *Euro-Par 2004 Parallel Processing* (M. Danelutto, M. Vanneschi, and D. Laforenza, eds.), (Berlin, Heidelberg), pp. 363–371, Springer Berlin Heidelberg, 2004. 24

[17] S. Moon, J.-G. Lee, M. Kang, M. Choy, and J. woo Lee, "Parallel community detection on large graphs with mapreduce and graphchi," *Data & Knowledge Engineering*, vol. 104, pp. 17 – 31, 2016. 24

[18] M. Naim, F. Manne, M. Halappanavar, and A. Tumeo, "Community detection on the gpu," *2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pp. 625–634, 2017. 25, 29, 44

[19] V. A. Traag, P. Van Dooren, and Y. Nesterov, "Narrow scope for resolution-limit-free community detection," *Phys. Rev. E*, vol. 84, p. 016114, Jul 2011. 25, 26, 27, 29

[20] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, p. P10008, oct 2008. 25, 26

[21] V. A. Traag, "leidenalg." `https://github.com/vtraag/leidenalg`, 2016. 27

[22] H. Lu, M. Halappanavar, and A. Kalyanaraman, "Parallel heuristics for scalable community detection," *Parallel Computing*, vol. 47, pp. 19 – 37, 2015. Graph analysis for scientific discovery. 43