ECAP-2009-002

Reconstruction of neutrino induced hadronic showers with the ANTARES neutrino telescope

Diploma thesis by Florian Folger

March 2009

Physikalisches Institut der Friedrich-Alexander-Universität Erlangen-Nürnberg

One is a discovery,
two are a spectrum.

Francis Halzen
about the detection of cosmic
neutrinos (SAT 2006)

Abstract

Neutrino astronomy is a comparatively novel field of research in astronomy. As it has always been the aim of all kinds of telescopes to search the sky for various cosmic sources, also newbuilt neutrino telescopes like *IceCube* and *ANTARES* try to identify cosmic neutrino sources by detecting and reconstructing neutrino induced muons that are produced if such a cosmic neutrino interacts with the earth's matter. As these muons traverse the detector with a velocity exceeding the speed of light in the medium, it is possible to reconstruct the direction of the muon and thus the origin of the neutrino.

But reconstructing muon tracks is not the only possibility to register neutrinos. As neutrinos can for instance also interact in neutral current reactions (where no muon is produced), it might be worthwile to also keep the eyes open for hadronic showers that are induced in such neutral current events. By doing so one extends the sensitivity of the detector to all three neutrino flavors and not just to muon neutrinos. The software group for *ANTARES* in Erlangen developed a reconstruction algorithm for hadronic showers, called *ShAuerReco*.

The aim of this thesis is to test and adjust this algorithm for the reconstruction of real detector data, as it has by now only been tested on Monte Carlo (MC) events. Afterwards it was tried to analyze real data with the adaptions, which turned out to be more difficult than expected due to some problems with the indentification of the hadronic showers in the whole data sample. However, this thesis might be informative to anyone who intends to make data analysis based on hadronic shower events. The reader can see what already has been tried out and might be inspired to tie in with this work and hopefully can reconstruct a shower event one day.

In chapter 1 a short overview of the history of astronomy and its way to neutrino telescopes is given. As one representative of a neutrino detector the *ANTARES* detector is described in chapter 2 before in chapter 3 a detailed introduction in the functionality of the *ShAuerReco* algorithm is given. Thereby not only the physical properties of the detector and the handling of the reconstruction module is mentioned but also problems that occur with different background sources.

The extensive analysis of the *ShAuerReco* algorithm to prepare it for real data re-

construction is described in chapter 4. Suitable fit boundaries for the reconstruction have been derived and the effects from hardware properties and background noise on the reconstruction quality has been investigated. Within this context also attainable reconstruction resolutions for direction and shower energy are derived. In chapter 5 also MC files containing muon track events have been processed with the shower reconstruction algorithm in order to find cut parameters to filter out all muon events and identify the shower events in real data samples. Various cut parameters and strategies and their effects on the purity and quality are presented.

Although some problems appeared during the event classification, first real data has been reconstructed in chapter 6, of course not to present physics results, but to see whether the reality follows somehow the MC studies of this work. The thesis concludes with some possible explanations for discrepancies between real and MC data.

In the appendix an overview of the used software is given. Part A describes the basic functionality of the *KM3Tray* framework that has been used for the analysis within this thesis. In B a list of the steering parameters of the *ShAuerReco* is given. And finally C introduces the *Root-plots* project that has been implemented to create the huge amount of required plots during these studies and that is now available within the framework to be used by everyone. To make it easier for future users to run the *ShAuerReco* reconstruction module and to use the *Root-plots* project a detailed manual for these two modules including examples for steering python scripts has been added.

Contents

List of Figures

Chapter 1

Neutrino astronomy

1.1 From counting the stars to neutrino astronomy

This introductory section is a short overview of the history of astronomy focused on the instruments that have been used for the observations and the obtained knowledge. The historical facts are basically a very short summary of the paragraphs on the history of astronomy in [1] and [2].

All times man has been fascinated by the mysterious dark that has surrounded our planet since time began. Day after day the sun rises anew and gives us the warmth and the light we need to survive. But at the end of a day, when the sunlight vanishes beyond the horizon, it can easily and clearly be seen, that there is so much more out there than just our own small planet, the moon and the sun. A countless number of small lightpoints is spread all over the sky, and all of them are moving - most of them following each other in the same direction, but some move quite different...

The huge interest in what is above us made astronomy the oldest and for a long time the most important science of all. It is not quite certain, when exactly mankind began to explore the universe. It is bequeathed that the ancient Egyptians and Babylonians indeed observed the movement of the planets and the moon, but it was the ancient Greeks who tried for the first time to find explanations for these movements and came up with the idea of orbits.

During the following centuries up to the end of the middleages all astronomical observations had one thing in common: They were strongly connected to the religion and mythology of the observer's homeland. Groups of well visible stars were formed to constellations, mostly named after some kind of god. Church dictated the discoveries that *had* to be found and those that were forbidden to propagate. In Europe this led to a total stagnancy of the progress in astronomical knowledge, whereas for example in India and China the observations had been persued.

Exploring the universe differs from other sciences like geology, chemistry, etc., in the impossibility to study the objects of interest directly, as they are of course not on earth, and it is mostly not possible to go there for a visit. So the only information one can gather and from which the conclusions have to be drawn is what arrives from outer space on earth, which is in principle the light and radio waves. Before the invention of the telescope at the beginning of the 16th century, the only available astronomical instrument was the observer's eye, which restricted him of course to only the brightest stars and the planets in our solar system. Johannes Kepler (1571 - 1630) and Galileo Galilei (1564 - 1642) used the first optical telescopes to have a closer look into the vastness of the universe and discovered a huge number of new stars that are to weak to be visible with the bare eye. Extended star countings could be done now and several big stellar catalogues came out after the new built telescopes grew more and more in their size.

But from then it took another 300 years until K.G.Jansky by chance discovered radio radiation from the milky way in 1932. The development of the radar technology, pushed by military concerns during the second world war, made it possible to scan the sky in the radio wavelength range from a few to several hundred meter. This resulted in the discovery of the 21-cm hydrogen line, which offered a great possibility to explore the movements of the interstellar matter via the doppler effect.

To make life possible on earth, it was essential to prevent the surface from harmful radiation that comes from all over the universe. Our natural shield against this thread is the atmosphere, which filters out all electromagnetic radiation except the visible light (including some infrared and ultraviolet frequencies) and the radio radiation. In order to extend the wavelength range of the sky scanning to shorter wavelength beyond the optical window it was necessary to leave our planet and build telescopes on earth orbiting satellites or at least measure on balloons in higher layers of the atmosphere. The infrared spectrum, which lies between the optical and the radio window and is not so strongly absorbed by the atmosphere as frequencies in the UV and above, can thus be detected with balloons that have been sent into the stratosphere. The infrared satellite COBE was used to scan the cosmic microwave background radiation, which gave us an important insight to what might have happened during the Big Bang.

The extension of the survey to higher energies brought similar enhancements of our understanding of the universe, as the astronomers now have the chance to study objects that produce more energy than a simple star, for example supernovae, accreting neutron stars or black holes. Being able to measure up to the extreme ultraviolet frequencies means to cover the whole spectrum of the hydrogen atom up to the Lyman edge, which made it possible to study the chemical evolution of single stars and the whole universe by modelling stellar atmospheres, evaluate their absorbtion and emission lines, and

fitting them to the measured spectra. [3]

In 1970 the first satellite to measure X-ray signals, called UHURU, was launched. One of its most important discoveries was the identification of the X-ray source Cygnus X-1, which ought to be the first candidate for an astrophysical black hole. At the same time the development of gamma ray telescopes was promoted regarding space telescopes like COS-B. After the turn of the millennium also ground based gamma ray telescopes like H.E.S.S. have been developed that allow for reconstructing the direction of cosmic particles (γ's, ...) by detecting air showers induced in the atmosphere. Today the survey of interstellar objects in the whole electromagnetic spectrum is guaranteed by GLAST and INTEGRAL, two gamma ray space telescopes, the X-ray orbital telescopes Chandra and XMM Newton, the Hubble telescope that measures from infared to ultraviolet frequencies and the radio-telescopes on earth. A selection of these telescopes is shown in Figure 1.1.

But since Victor Hess' ascention in his balloon in 1912 it is well known that the comsic radiation doesn't consist only of electromagnetic radiation but has also a hadronic and leptonic component. This implies the necessity to also investigate these two components to get as much information as possible out of them. Every extension in the history of telescopes to shorter wavelengths provided new fundamental knowledge (see Figure 1.2). That raises hope to find answers to unsolved questions and gather new information. Hence it is worth to extend the studies further on this particles that Hess discovered and are called *cosmic rays*.

1.2 The cosmic rays

The cosmic rays consist of two components. The first is called the *primary component*, which is a denotation for all the particles hiting earth from the universe. As some of them are quite high energetic they will react with particles in the earth's atmosphere, where thus a whole shower of secondary particles is produced, which is called the *secondary component*. Whether the primary or the secondary component can be measured depends on where the detector is located. On the earth's surface approximately 80% of all charged particles from cosmic rays are muons, which are part of the secondary component. In higher levels of the atmosphere the muon abundance decreases, where protons become more and more present.

The hadronic part of the primary compoment is composed of protons ($\approx 85\%$), α particles ($\approx 12\%$), and nuclei of heavier elements [12]. The abundances of the elements occurring in the primary component of cosmic rays can be used to make statements on the chemical evolution of the universe and the processes that can transform matter of

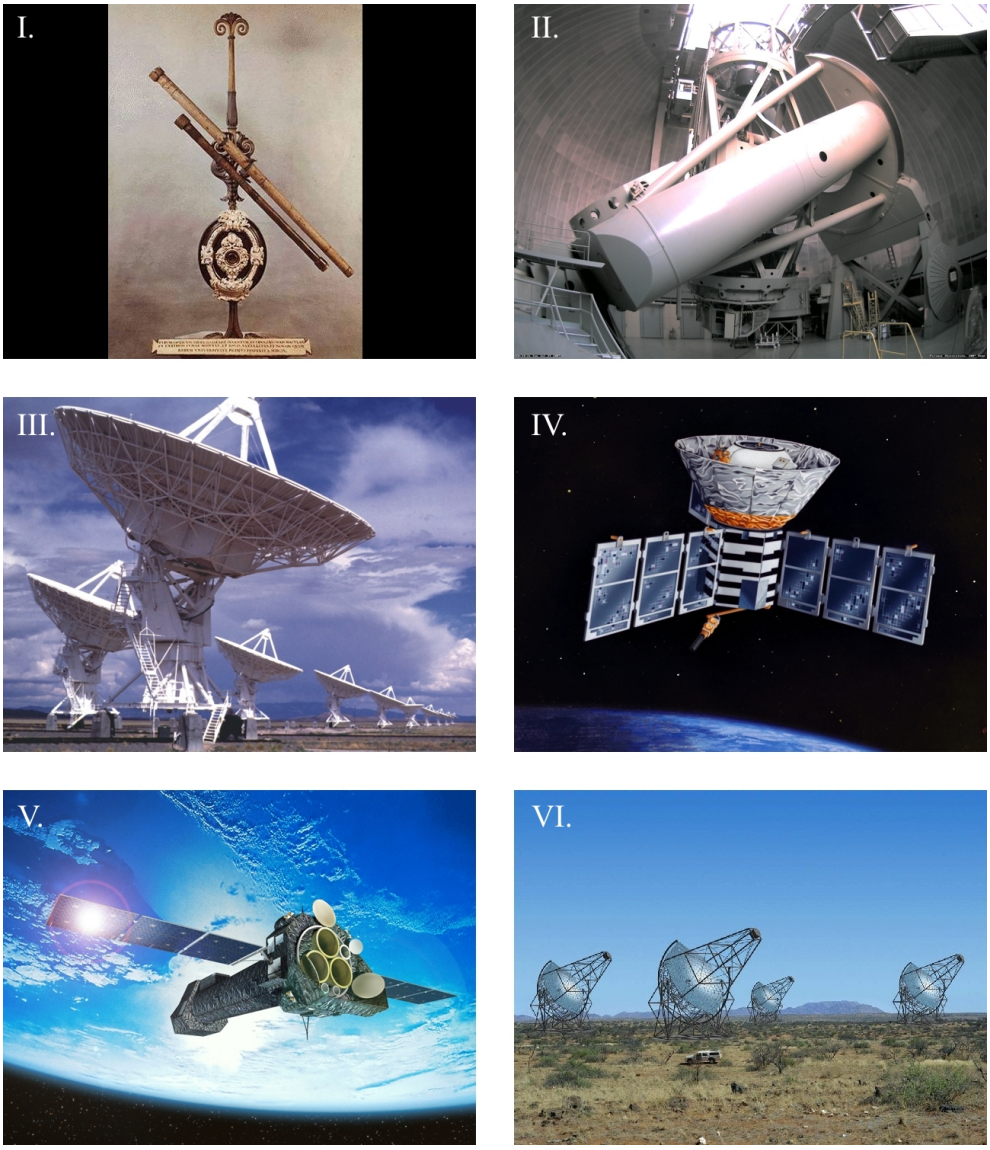

Figure 1.1: *Astronomical instruments throughout the history of space observations.* **I.** Galileo Galilei's telescope [4]. **II.** The Hale refraction telescope at the Palomar Observatory, California [5]. **III.** The Very Large Array radio telescopes, New Mexico [6]. **IV.** COBE infrared satellite (illustration) [7] **V.** XMM Newton x-ray satellite (illustration) [8] **VI.** H.E.S.S. gamma ray telescopes, Namibia [9]

one kind into another, like supernovae or the energy generation within a star, where nuclear reactions provide the necessary power. Its energy spectrum is shown in Figure 1.3. Up to $10^{15}eV$ the cosmic rays is believed to have its origin within our own galaxy, whereas for higher energies the spectrum gets steeper. The breakpoint in between is called the *knee*. The spectrum follows a potential law like

$$N(E) \sim E^{-\gamma} \tag{1.1}$$

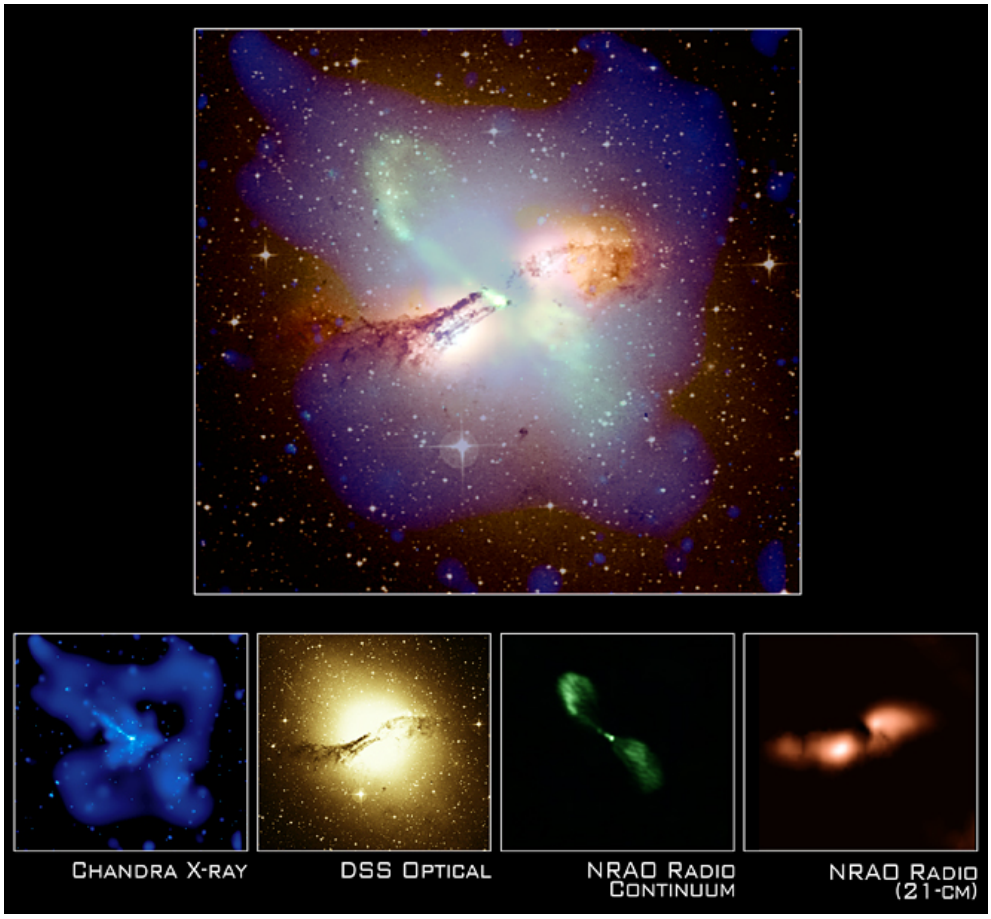

Figure 1.2: *A composite image of the galaxy Centaurus A.* In the optical band (yellow) a usual elliptical galaxy with a dust belt around its bulge can be seen. The radio continuum image shows a jet like structure (green) that isn't visible through optical telescopes and indicates the presence of an active galactic nucleus (AGN). A view in the 21-cm line shows high abundances of hydrogen in a belt perpendicular to the jet, which might be a sign for H II regions within the dust belt, where new stars are formed [10]. In x-rays (blue) one can see two great arks in the outskirts of the galaxy that must contain very hot gas. The image has been composed by [11].

where below the knee γ is represented by 2.7 and above by 3.0. A possible explanation for the knee may be the fact that 10^{15} eV is about the highest energy that can be produced by a supernova. Thus all particles with higher energies must have been produced in other, probably extragalactic, accelerators [12]. Above 10^{19} eV the spectrum flattens again, which is called the *ankle*.

At energies above the ankle statistics become quite bad, as the fluxes are very low. Therefore it is quite interesting to study the cosmic rays at those high energies in order to prove theories and find explanations to what happens here and how particles can be accelerated to such high energies. In 1966 Kenneth Greisen, Vadim Kuzmin and Georgiy Zatsepin computed a cutoff of the cosmic rays at 10^{20} eV [14], as protons with

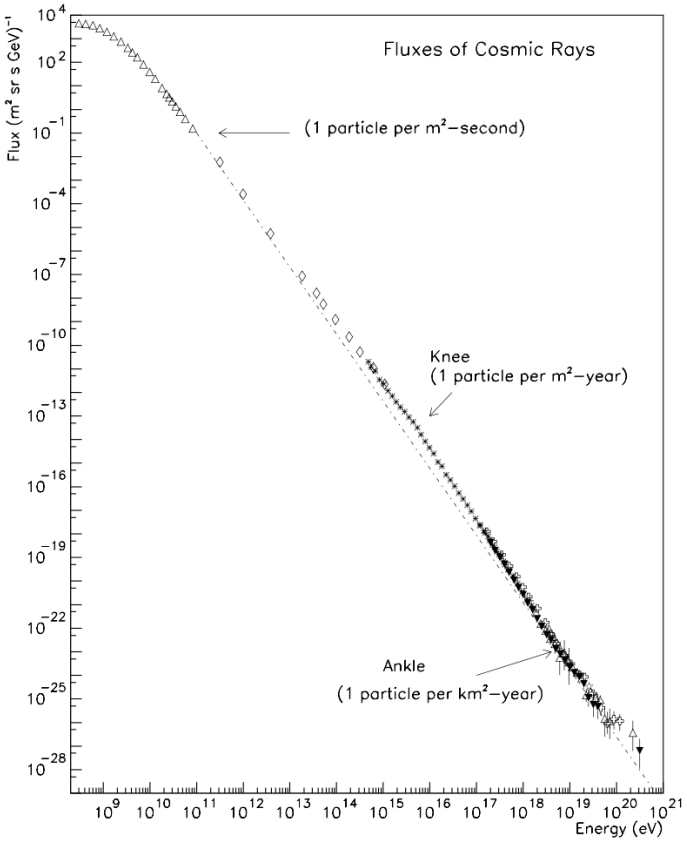

Figure 1.3: *All particle spectrum of primary cosmic rays.* [13]

an energy above a threshold of $6 * 10^{19}$ eV would interact with the cosmic 2.7 Kelvin blackbody radiation and lose $\approx 20\%$ of their initial energy in each of such reactions, where an excited state of the proton, called Δ-resonance, and further a pion is produced

$$\gamma + p \rightarrow p + \pi^0 \qquad \gamma + p \rightarrow n + \pi^+ \,. \tag{1.2}$$

A proton will produce pions as long as its energy decreased below the threshold of $6 * 10^{19}$ eV or a neutron is produced [15]. Thus a small accumulation of proton energies between the ankle and this threshold will occur, which might explain the flattening of the spectrum.

Another aspect to the study of cosmic rays is not only to know the energy spectrum but also its origin. At best one could make images of single sources, just as it is done with light. Unfortunately due to the ionized state of the these particles, it is impossible to determine their source, as charged particles are deflected by the magnetic field that is present in the whole universe. Thus one cannot use the hadronic component to scan the sky.

The leptonic part of the primary component consists mainly of electrons, positrons

and neutrinos. Of course, as electrons and positrons are also charged particles, one has the same problems as with protons. To be able to get information, where the cosmic rays is produced, a particle is needed that is not influenced by magnetic fields, is stable enough to survive its long way to earth and will not be absorbed by the interstellar matter or the 2.7 Kelvin background radiation. The neutrino is a particle that matches to all the requirements! If the energy and the direction of a cosmic neutrino could be determined, one can probably gather a lot of information about its source, check theories and models of this source and complete or disprove theories on the evolution of the universe.

1.3 Cosmic neutrino sources

The above mentioned ideal properties of the neutrino, which make it a good probe, are disadvantages in the detection process. A neutrino that was produced in a far galaxy propagates almost undisturbed on its way to earth, penetrating all kind of matter along the way. So it is quite likely that it will also penetrate the earth without anyone taking notice of it. Solar neutrinos, for example, have an energy of a few 100 keV, and thus their cross section for neutrino-nucleon scattering is

$$\sigma(\nu_e N) \approx 10^{-45}\,\mathrm{cm}^2, \tag{1.3}$$

wherefrom the interaction probability for such a neutrino with the whole earth can be calculated to

$$p_{\mathrm{interaction}} = 2 * \sigma * N_{\mathrm{Avogardo}} * R_{\mathrm{Earth}} * \rho_{\mathrm{Earth}} \approx 4 * 10^{-12}. \tag{1.4}$$

This means out of $7 * 10^{10}$ solar neutrinos arriving at earth per cm^2 and s only **one** can be detected, assumed the whole earth (!) being the detector [12]. So one can easily appreciate the need of big and massive detectors! The reasons why it is worth making such huge efforts to build gigantic detectors are quite obvious and will be described in the following.

Figure 1.4 shows the expected neutrino flux from sereval sources in different energy regimes. In the following section a short description of these sources is given, except for the cosmological neutrinos, which are a relict of the Big Bang. Their energies are in the $\mu\mathrm{eV} - \mathrm{meV}$ range, where at present it is inconceivable that such low energetic neutrinos can be measured at all [12].

Most of the measurable low energy neutrinos are solar neutrinos that are byproducts of the sun's nuclear fusion reactions. The sun generates energy by burning hydrogen to helium in the so-called proton-proton chain reaction, of which the most frequent is:

$$^1\mathrm{H}(\mathrm{p}, \mathrm{e}^+\boldsymbol{\nu_e})^2\mathrm{D}(\mathrm{p},\gamma)^3\mathrm{He} \qquad \rightarrow \qquad ^3\mathrm{He}(^3\mathrm{He}, 2\mathrm{p})^4\mathrm{He}\,. \tag{1.5}$$

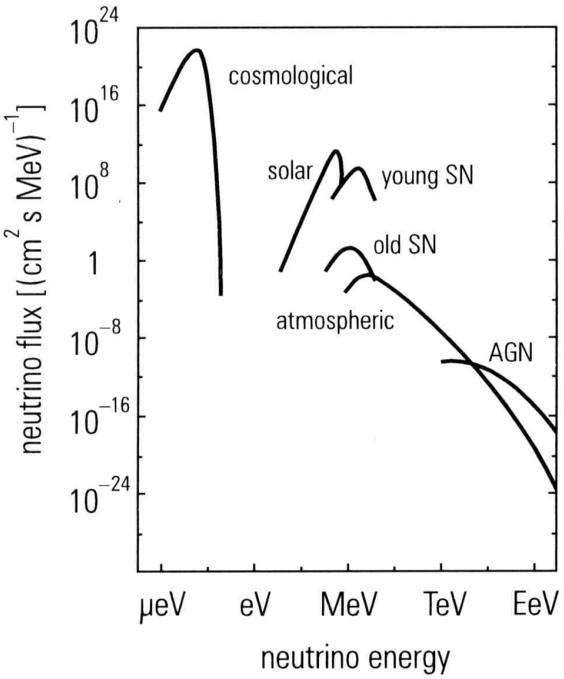

Figure 1.4: *Expected neutrino flux of different cosmic sources.* (SN: Supernovae, AGN: Active galactic nuclei) [12]

The processes running in the sun's core were thought to be quite well understood and thus the results of experiments like *Kamiokande* and *Super-Kamiokande* could be seen as a first indication for neutrino oscillations. The solar standard model predicts a well calculated flux of (electron-) neutrinos, whereas the *Super-Kamiokande* experiment, which is sensitive only to electron neutrinos, measured only 40% of the expected flux [12]. This caused many discussions about the validity of the solar standard model and it finally was not proved before 2001 by the *Sudbury Neutrino Observatory* that, assuming the validity of the solar standard model, the measured discrepancy is an evidence for the electron neutrinos having changed their flavor. And the only way to do this without interacting with another particle, which is quite unlikely due to our close distance to the sun, is neutrino oscillations [16]. But as the energy of solar neutrinos does not exceed $14,6\,\mathrm{MeV}$ [16] they are much too low energetic to be detected in a string detector like *ANTARES* or *IceCube*, which are built to detect high energetic neutrinos and thus have a density of photomultipliers that is too low for the detection of MeV neutrinos.

Further neutrinos are produced in supernovae that are the last phase in stellar evolution of massive stars. When the nuclear fuels within the star's core, which is mostly hydrogen, helium and depending on the stellar mass some heavier elements, run out, the star tries to gain energy out of its gravitation by contracting itself, which

creates enough high temperatures and densities to make the inverse beta decay possible.

$$p + e^- \rightarrow n + \boldsymbol{\nu_e} \tag{1.6}$$

What can be seen as bright light is only approximately 1% of the energy produced by the supernova. The remaining 99% are carried away by neutrinos, which effectively cools down the very hot star. Supernova neutrinos have an average energy of 10 MeV [16]. Although supernovae are quite rare events, the huge amount of produced neutrinos contributes a great part to the flux arriving on earth. However, the energies are also too low for being detected with *ANTARES*.

What *ANTARES* actually tries to find are high energetic neutrinos from cosmic accelerators. Detecting those neutrinos and measuring their energies could give us insights into the ways cosmic accelerators work and where they are. To give a brief example of the argumentation: If a source is powerful enough to accelerate protons to very high energies ($>$ TeV) this source might be a neutrino source! High energetic protons will interact with the matter, if existent, just like they do when producing secondary cosmic rays in the earth's atmosphere. In scattering processes with nucleons, a lot of pions are produced.

$$p + \text{nucleon} \rightarrow \pi^+ + \pi^- + \pi^0 + \text{other particles} \tag{1.7}$$

As pions have a short lifetime ($\tau_{\pi^\pm} = 2.6 * 10^{-8}\,\text{s}$ and $\tau_{\pi^0} = 8.4 * 10^{-14}\,\text{s}$, [17]) they will decay immediately, whereat the charged ones produce muons, which on their part decay further producing electrons. During each of these decays neutrinos are generated.

$$\begin{aligned} \pi^+ &\rightarrow \mu^+ + \boldsymbol{\nu_\mu} & \qquad \pi^- &\rightarrow \mu^- + \boldsymbol{\bar{\nu}_\mu} \\ &\hookrightarrow e^+ + \boldsymbol{\nu_e} + \boldsymbol{\bar{\nu}_\mu} & &\hookrightarrow e^- + \boldsymbol{\bar{\nu}_e} + \boldsymbol{\nu_\mu}\,. \end{aligned} \tag{1.8}$$

The neutral pion decay results in two photons

$$\pi^0 \rightarrow \gamma + \gamma\,. \tag{1.9}$$

If neutrinos *and* photons are detected from a source, one knows its acceleration mode, which has to be hadronic. If only neutrinos are detected, maybe the interacting matter is dense enough to absorb all photons. And if one measures only photons, hadronic acceleration can be excluded as power source.

The question is what cosmic objects are considered as proton accelerators. There are basically four types mentioned (after [16]): **Young supernova remants** with a rapidly rotating pulsar or black hole in its center have still enough energy to accelerate protons via the also rapidly rotating magnetic field even one year after the supernova

took place. The target matter for these protons to produce neutrinos is the atmospheric shell that has been rejected during the supernova.

Another possible object is a **binary system**, in which one part is a compact object, like a neutron star or a small black hole. If the big companion star exceeds its size above the Roche limit, it will emit its mass to the compact object. The so formed accretion disc rotates in the compact object's strong magnetic field and thus induces - like a huge dynamo - an electric field that accelerates protons to high energies. The scattering and neutrino production takes place in the atmosphere of the giant star.

The most important objects to study among proton accelerators are, however, **active galactic nuclei (AGN)**, which are huge galaxies with a supermassive black hole in their center. A characterstic feature of an AGN is the relativistic jet that is emitted perpendicular to its disk. Although it exists a quite sophisticated model of an AGN (see [2]), a lot of details are still unknown about it. Matter is expected to be accelerated in two different ways: by the accretion disk and in the relativistic jets. Neutrinos from the AGN could help us to study its morphology closer and strengthen our understanding of the physical processes taking place there.

Gamma ray bursts (GRB) are most luminous events occurring all over the universe. During a gamma ray burst an extremely bright flare that lasts only for a few seconds can be seen in gamma rays. Possible explanations for this phenomenon could be a violent supernova, called hypernova, where the star collapses into a rotating black hole, or the melting of two neutron stars [12]. However the physics of gamma ray bursts are not well understood by now, so that possible neutrinos from a GRB would carry important information about whether the acceleration is hadronic or not (in the latter case, no neutrinos will be detected). Due to its short duration, it is difficult to locate the exact origin of the GRB and identify it with a known object. If the acceleration is hadronic, neutrinos will be produced, and as neutrinos experience less interaction than photons, they will arrive at earth shortly before the flare, caused by the gamma photons. Hence detecting neutrinos from a GRB could not only be used for understanding the physics but also as a kind of trigger for gamma ray telescopes to adjust their focus on the object before the flare appears. Efforts to create such a trigger system are made at the moment by the *IceCube Neutrino Observatory* [18].

For the sake of completeness also some more exotic sources of possible neutrino emission shall be introduced now to conclude this chapter. Neutrino astronomy can be used to prove or disprove the existence of theoretical particles like the *neutralino* or the *magnetic monopole*. The neutralino is the supersymmetric partner of the neutrino and a candidate for a weakly interacting massive particle (WIMP), which dark matter in the universe is expected to consist of. As neutralinos are subject only to the weak interaction and gravitation, they can accumulate within a massive object, like a star or

a whole galaxy, where they can annihilate with another neutralino and hence produce neutrinos

$$\chi + \chi \rightarrow b + \bar{b} \qquad \text{if}(m_\chi < m_W) \tag{1.10}$$

$$\chi + \chi \rightarrow W^+ + W^- \qquad \text{if}(m_\chi > m_W)\ . \tag{1.11}$$

If the (by now unknown) mass of the neutralino is less than that of the W-boson it will annihilate to a b-quark/anti-b-quark pair, which hardonizes and finally ends up as pions and kaons decaying further to leptons and neutrinos. If the mass is bigger the annihilation ends up in two W-Bosons that decay directly into leptons and neutrinos. The energy of these neutrinos grows with the mass of the neutralino. To measure a higher neutrino rate from the center of the sun or some other massive objects could be interpreted as an evidence for the existence of such a particle [16].

Magnetic monopoles are theoretically predicted particles by the Grand Unification Theory (GUT) that tries to combine all four natural forces to one. If such a monopole exists, it would produce several protons along its track, wherefrom Čerenkov emission (see chapter 2.1) will be emitted and possibly be detected with one of the neutrino detectors.

1.4 Atmospheric neutrino background

Apart from the small cross section, which makes it necessary to build huge detectors for neutrino astronomy, another big problem is the atmospheric neutrino background. What *ANTARES* intents to measure is the neutrinos from the primary cosmic rays to find point sources in the sky. But as told in chapter 1.2 a secondary component including whole showers of particles is produced in the earth's atmosphere. In the same argumentation as used for cosmic proton accelerators, where the high energetic protons produce those neutrinos that one is looking for, also the protons that arrive at the earth's atmosphere produce neutrinos that have no further information about the direction or energy of the source, as the interacting protons arrive at earth isotropically. A proton hitting a nucleus in the atmosphere will produce pions that decay to muons and electrons in the same way as mentioned for cosmic accelerators in (1.7) and (1.8). Hence the expected ratio of overall detected numbers is expected to be

$$\frac{N(\nu_\mu, \bar{\nu}_\mu)}{N(\nu_e, \bar{\nu}_e)} = 2 \tag{1.12}$$

However measurements with *Super-Kamiokande* showed a quite smaller ratio, which was the discovery of neutrino oscillations.

What can be seen is that atmospheric neutrinos were quite useful for the validation of neutrino oscillations, whereas in contrast they are quite annoying for neutrino astronomy, because there is no possibility to separate them from cosmic neutrinos. The only way to prove a cosmic neutrino is to detect it as an overage above the atmospheric neutrino background. Figure 1.4 shows that atmospheric neutrinos are dominant in the energy range between GeV and some TeV. A cosmic point source emitting neutrinos with such energies has to be a very strong neutrino source to be detected. Below 100 GeV the flux of the atmospheric neutrinos follows the potential law of the primary component (see chapter 1.1) with $\gamma = 2.7$. Above this energy the spectrum steepens to $\gamma = 3.7$, which makes it much easier to detect point sources with high energetic neutrinos, as above 100 TeV neutrino energy the atmospheric background is negligible here.

The following chapters will show that atmospheric neutrinos are not the major problem when looking for cosmic neutrinos. Because Čerenkov detectors like *ANTARES* reconstruct the tracks of muons that have been induced by throughgoing neutrinos, they are sensitive for all kind of muons, no matter where they are produced! Although muons have a short livetime of $\tau_{\pi^\pm} = 2.6 * 10^{-8}$ s it is due to relativistic effects that not all atmospheric muons created from decaying pions after (1.8) decay before arriving at the earth's surface, or at the *ANTARES* detector below sea level. A quite big amount of them is still present at a depth of 2400 m, where *ANTARES* is located, and will be reconstructed as well as neutrino-induced muons. To filter them out is still a great challenge and the problem of separation of atmospheric from neutrino induced muons has not been solved by now.

Chapter 2

Hunting for neutrinos

2.1 Detection of neutrinos using the Čerenkov effect

As described in chapter 1.3, neutrinos are weakly interacting particles and thus their detection can be quite challenging. First experiments to detect neutrinos, like the *GALLEX*-experiment, used huge tanks filled with pure gallium and checked the deposit of neutrino induced impurities in regular intervals. The detector was located deep under the surface to be sure that excluding neutrinos all other particles from atmospheric showers are filtered out by the massive rock. A throughgoing neutrino has enough energy to, if interacting with one of the gallium atoms, produce a germanium atom by transforming a neutron within the nucleus into a proton [19].

$$\nu_e + {}^{71}\mathrm{Ga} \rightarrow {}^{71}\mathrm{Ge} + e^- \tag{2.1}$$

By counting the impuriting germanium atoms one can make statements about the neutrino flux or the interaction cross section. But as these experiments are only counting experiments, it is not possible to assess the energy or the source direction of the neutrino! To be able to get this information huge efforts have been made to build large-scale Čerenkov detectors.

In the following part the functionality of such a Čerenkov detector is explained. A neutrino passing through a detection medium, like water or ice, can interact with an atom and produce multiple secondary particles, which propagate further at a speed that is greater than light velocity in the detection medium. Analogously to the Mach cone and the resulting sonic boom that occurs if, for example, a plane moves faster than sound velocity here a so-called Čerenkov cone is produced from a traversing charged particle. This effect is named after the Russian physicist Pavel Alekseyevich Čerenkov, who discovered this effect [20].

In Figure 2.1 a relativistic particle moving from S' to S is shown. If the particle is charged, Čerenkov photons will be emitted along its track due to the polarization of its

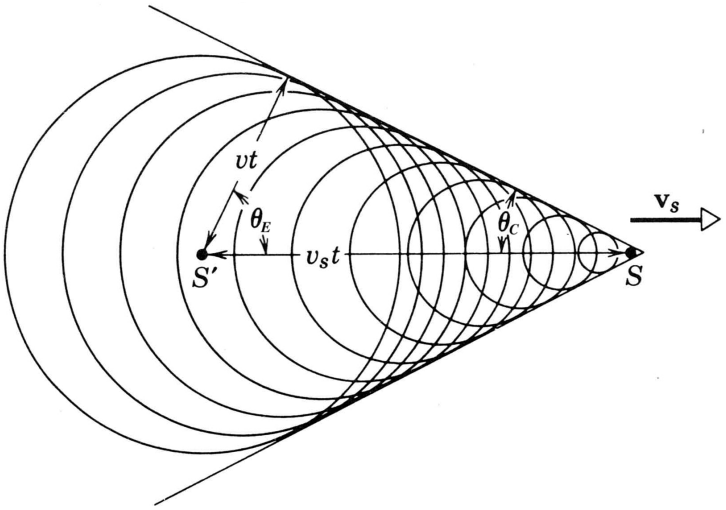

Figure 2.1: *Čerenkov cone of a particle exceeding light velocity in a medium*, modified from [21]

environment and the thus induced electromagnetic waves that interfere coherently with each other. As the particle propagates faster than the speed of light in the medium, it will cover a greater distance (i.e. $v_s t$) than the outcoming photons (vt) in the same time, assuming v as light velocity in the medium. The thus created wave front can be detected as weak lightflare in the visible spectral range from 400 nm to 700 nm [12]. From simple geometry one can derive from the figure that this Čerenkov radiation is emitted at an angle of

$$\Theta_E = \arccos \frac{vt}{v_s t} = \arccos \frac{1}{n\beta} \tag{2.2}$$

where $n = c/v$ is the refraction index of the medium, c the light velocity in the vacuum, and $\beta = v_s/c$. At the *ANTARES* site the water's refraction index is about 1.34, slightly varying with wavelength, water temperature and salinity, according to [22]. According to (2.2) all photons will be emitted at an angle of 42° respective to the particle track, as these neutrino induced particles move through the water at almost the speed of light ($\beta \approx 1$). Note: The so-called Čerenkov angle is defined as

$$\Theta_C = \arcsin \frac{1}{n\beta} \tag{2.3}$$

and describes half the opening angle of the cone, which is, however, less important for the reconstructions described in this work.

By detecting and counting the Čerenkov photons at multiple positions and measuring their arrival time, one can reconstruct the direction and energy of the emitting particle and thus, if the particle is a muon directly induced by a cosmic neutrino, determine the source direction of the neutrino. At detectable energies ($> 1\,\mathrm{GeV}$) the

difference between the direction of the produced muon and that of the primary neutrino is smaller than the angular resolution of the used reconstructions and can thus be neglegted [23]. This detection principle can also be applied to shower events. In this case it is not just one muon that emits the Čerenkov photons but a whole sample of secondary shower particles. Figure 2.2 shows an illustration of how neutrinos can be detected on earth.

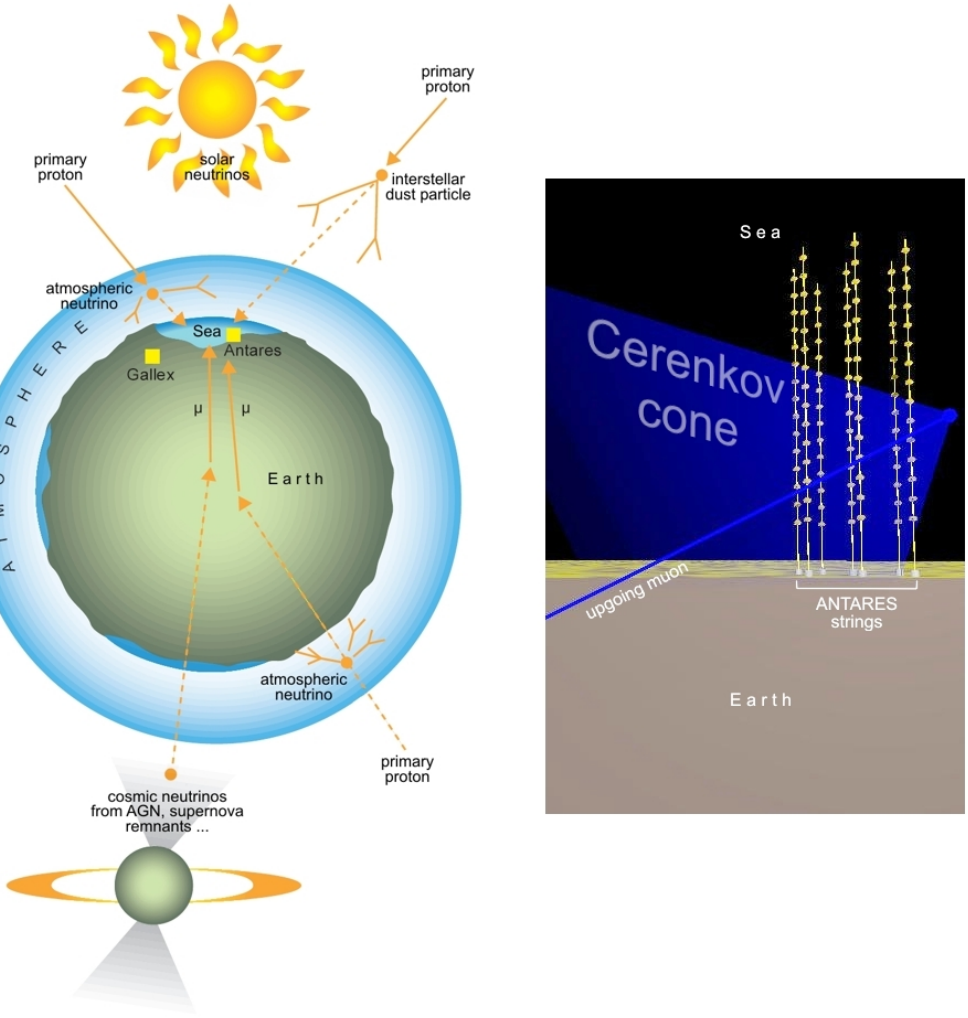

Figure 2.2: *Illustration of sources, wherefrom neutrinos can be detected* (left) [24]. Cosmic protons interacting with interstellar dust particles produce neutrinos that could be measured as a diffuse cosmic neutrino flux with *ANTARES*. When hitting the earth's atmosphere the protons produce atmospheric neutrinos that are the neutrino background described in chapter 1.4. Cosmic neutrinos can thus only be detected as an overage above the atmospheric neutrino spectrum.
Čerenkov cone of a throughgoing muon track in the ANTARES detector (right) [25]. The neutrinos can interact with nuclei in the earth crust or the sea inducing muons that can be detected via their Čerenkov light (right).

To introduce some currently operating Čerenkov telescopes, *Super-Kamiokande* in

Japan, *AMANDA/IceCube* at the south pole and of course *ANTARES* shall be mentioned here. *Super-Kamiokande* was build 1 km beneath the surface in the Japanese Kamioka mine and has a detector volume of 50000 m^3 filled with pure water that is surrounded by 13031 photomultipliers (PM) [26]. Its comparatively small volume constrains the experiment on the detection of low energy neutrinos, like solar neutrinos or those from supernovae (see chapter 1.3) but a high density of PMs offers a good reconstruction quality. *AMANDA/IceCube* has been built to detect high energetic neutrinos, for what a lower density of PMs is sufficient, where as due to low fluxes the detector volume has to be increased by far. 80 vertical strings with altogether 4800 PMs have been placed into the ice shell of Antarctica, where 1 km^3 of the ice is used as detection medium [27]. The design of *ANTARES* is similar to *IceCube* with a major difference concerning the detection medium that is sea water. As *ANTARES* is a prototype for underwater telescopes it has a comparatively small instrumented volume of 0.01 km^3, where a larger scaled detector, called *KM3Net*, is already in process of planning [28].

2.2 The *ANTARES* detector

This paragraph gives a short overview of the technical design of the *ANTARES* telescope, which is located about 30 kilometers south off the French Mediterranean coastline nearby Toulon. A detailed description of the detector is given in [29] and [30].

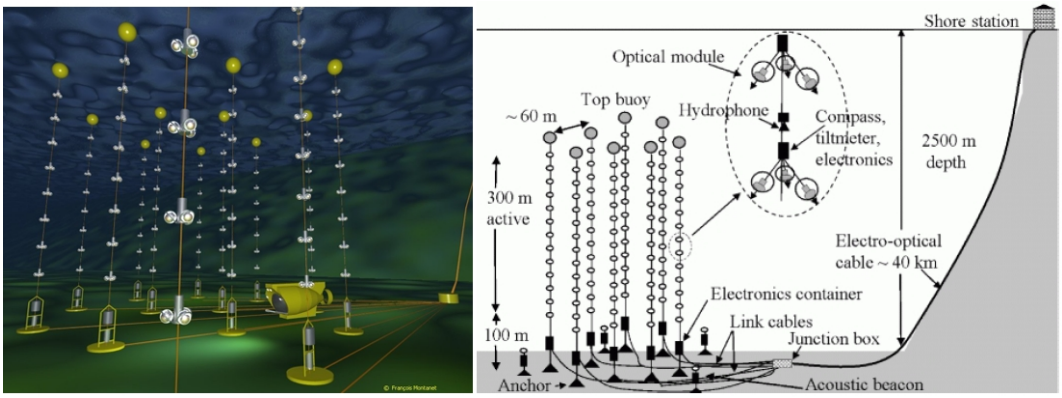

Figure 2.3: *Schematic views of the ANTARES telescope*, [25] (left), [31] (right)

The detector consists of 12 vertical strings (lines), of which each has a total length of 460 metres, and where each is hold in its position by a buoy on the top of the line and an anchor on the ground. A line is equipped with 25 storeys, of which each contains three optical modules (OM). A storey and an OM are shown in Figure 2.4. The storeys have a vertical distance of 14.5 metres to each other along the string, which provides an active instrumented length of 350 metres on the line. Each line is fixed at its bottom

string socket, wherefrom the acquired data from all OMs on the string is sent to the junction box that is the only interface to the shore station. The single strings are mounted on the ground in a horizontal distance of about 60 metres to each other and thus a total volume of about $0.01\,\mathrm{km}^3$ is instrumented.

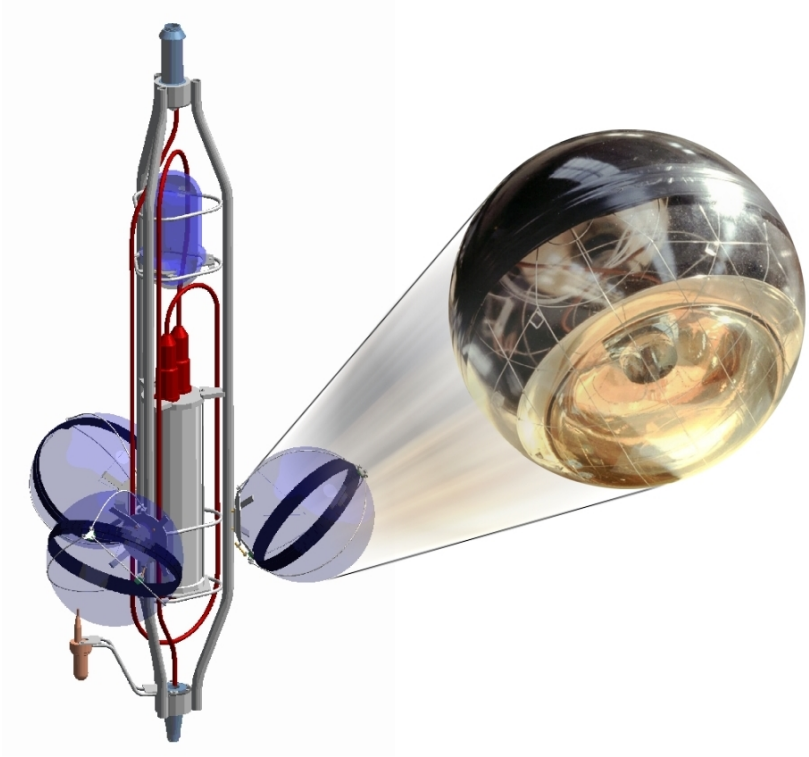

Figure 2.4: *Schematic view of an ANTARES storey and photograph of an optical module*, storey graphic from [29], OM from [31]

The three optical modules on a storey are arranged symmetrically with the local control module (LCM) in its center. Each OM consists of a photomultiplier (PM) that is built in a glass sphere that is robust enough to withstand a water pressure of 260 bars. The photo multipliers are orientated at an angle of 45 degrees below the horizon to be more sensitive on upgoing muons (and thus neutrinos) and suppress atmospheric muons that come only from above the equator as effectively a possible (see chapter 2.4 for more information of the atmospheric muon background). The assembly of the three OMs at an angle of 120 degrees between two of them provides an azimuthal coverage of 360 degrees around the string, whereas the vertical acceptance is higher for upgoing events.

As the upper end of a string is not fixed the string will move following the currents of the deep sea. To determine the exact position of the OMs, hydrophones have been installed on some storeys. Four autonomous acoustic transmitters have been placed at

the corners of a 300 m × 300 m square around the detector. From the arrival time of the acoustical signals at the hydrophones the positions of the storeys and thus the line shape can be derived. Compasses and tiltmeters on each storey provide the information about the orientation and tilt of the OMs.

2.3 Detectable events

Neutrinos that interact with the matter in or around the detector can produce different signatures depending on the kind of reaction. Figure 2.5 shows Feynman graphs of possible reactions induced by muon and electron neutrinos. The graph on the left side describes the reaction that was intended to be detected with *ANTARES*. A muon neutrino (the same signatures apply also for anti neutrinos, which are not mentioned furthermore) interacting with a nucleus in the ocean or the earth's crust exchanges a charged W-boson, where a muon and a hadronic shower is produced. Interactions exchanging a W-boson are called charged current reactions (CC). As the track of the induced muon can be several kilometers long (depending on the energy of the primary neutrino, see Figure 2.6) also those neutrinos can be detected that have interacted far outside the detector. Hence the detection volume for neutrino induced muon events exceeds the instrumented detector volume by far. On the other hand, a long track makes it difficult to reconstruct the muon energy, as only a small part of the Čerenkov photons that are emitted along the whole track can be seen within the detector.

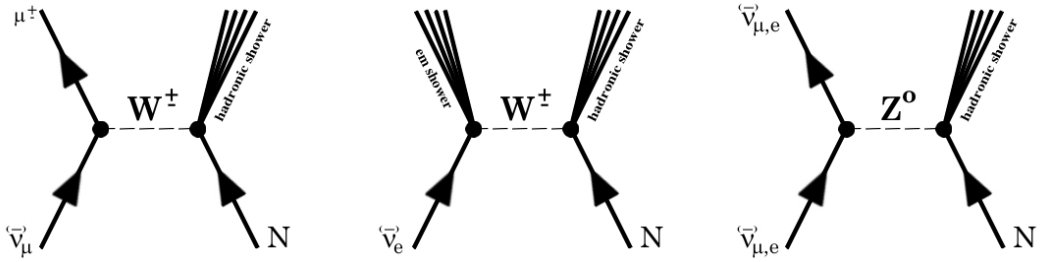

Figure 2.5: *Possible interactions of muon and electron neutrinos with a nucleus*

The two latter graphs show events that only produce showers. Hadronic and electromagnetic showers have an extension of only a few meters (see Figure 2.6), and thus only those of them that are produced in the closest distance to the detector can be detected. In the middle graph an electron neutrino interacts with a nucleus and thus produces a high energetic electron that induces an electromagnetic shower. Those showers only consist of three particles that are photons, electrons and positrons. The primary induced electron will suffer bremsstrahlung and thus produce photons that will again produce electron-positron pairs via pair production.

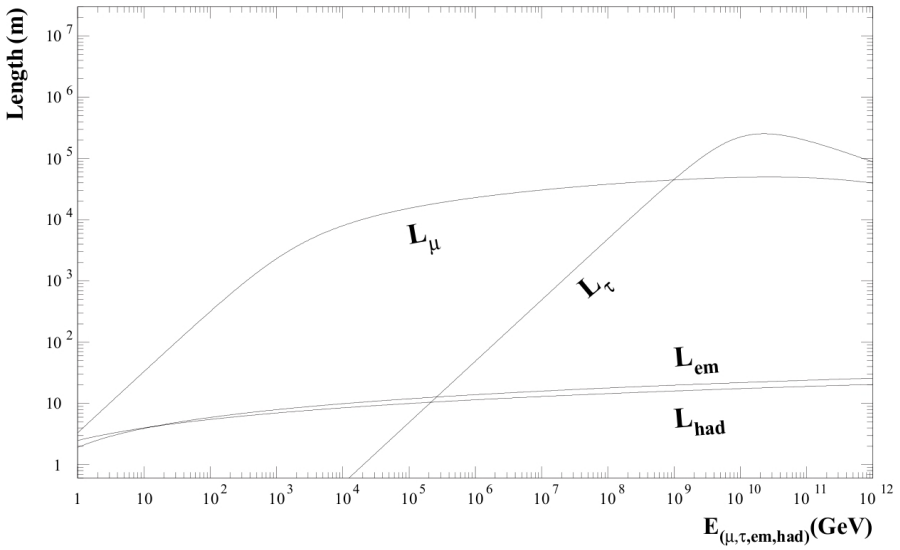

Figure 2.6: *Length of muon and tau tracks compared to the extension of electromagnetic and hadronic showers.* [32]

The graph on the right side, where a Z-Boson is exchanged, shows a neutral current event (NC), where only a hadronic shower is produced. The outgoing neutrino will leave the detector without being noticed further. The composition of an hadronic shower is more complex, as depending on the shower energy, different kinds of shower particles are produced. Figure 2.7 shows the relative abundance of the long-lived primary particles in an hadronic shower that are mostly pions. These pions decay analogously to the equations (1.8) and (1.9). Note that in the plot the short-lived π^0 does not appear but is represented by the resulting γ's. For increasing shower energy the production of π^0 and thus the fraction of γ's in the shower increases. These photons induce electromagnetic sub-showers that are regarded as part of the hadronic shower within this thesis, as due to the low PM density and the small extension of the showers, they can not be separated.

In this thesis those neutral current events (NC) have been studied with a reconstruction algorithm for hadronic showers, developed by Ralf Auer (see chapter 3). Concentrating on muons only means the loss of much information, as muons can of course only be a product of a reaction with a muon neutrino. Other neutrino flavours (electron and tau neutrinos) can not be reconstructed when using *ANTARES* just as a muon detector. Therefore it is worthwhile to take also neutral current events into account, as they can be induced by all three flavors! The probability whether a neutrino induces a charged or a neutral current reaction depends on the cross sections shown in Figure 2.8. Hence about every fourth reaction will be a neutral current reaction producing a hadronic shower. However, the smaller effective volume will decrease the

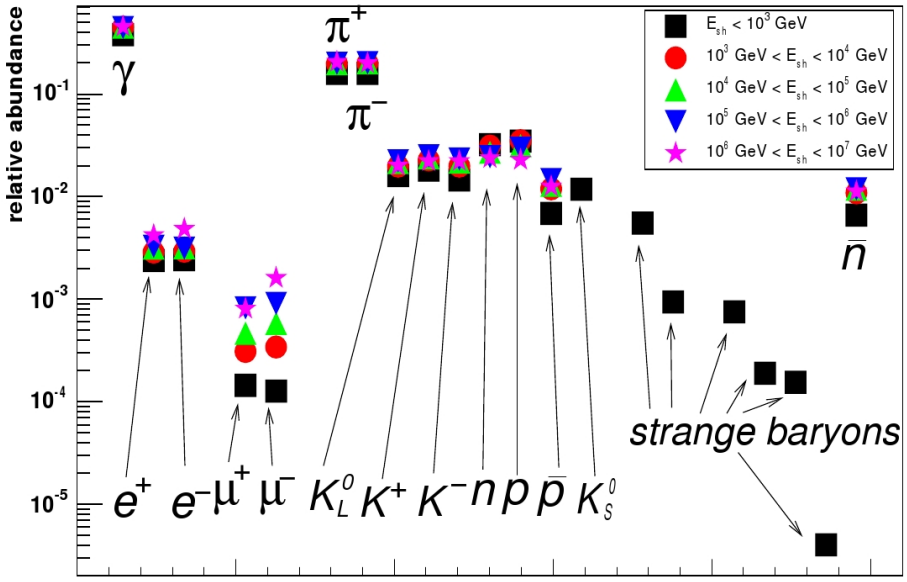

Figure 2.7: *Relative abundances of long-lived particles in hadronic showers for different primary energies.* Note that only primary particles are displayed. Secondary particles from further decays, like photons and electrons in electromagnetic sub-showers, do not contribute to the abundances [33].

ratio of detected neutral/charged current neutrinos.

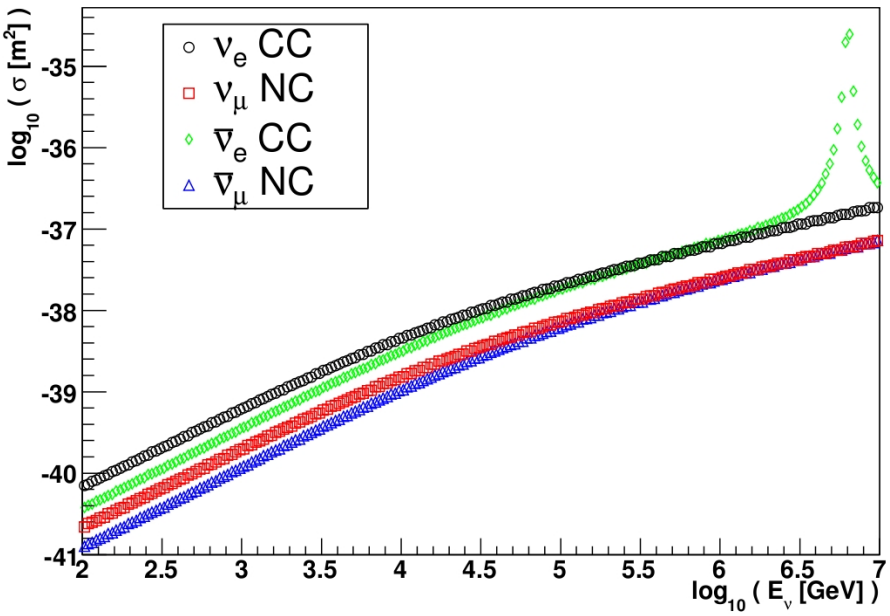

Figure 2.8: *Cross sections for neutral and charged current shower producing events* The peak at 6.4 PeV in the $\bar{\nu}_e$ cross section is due to the *Glashow* resonance, that is the production of a W-boson from the electron anti neutrino and an electron from the target atom [33].

2.4 Background sources at the *ANTARES* site

As the *ANTARES* detector is sensitive for throughgoing muons, also atmospheric muons that are part of the secondary cosmic rays (see chapter 1.2) will be reconstructed. Although the OMs are orientated below the horizon, still a huge amount of atmospheric muons is detected. Atmospheric muons actually represent the greatest part of registered events as shown in Figure 2.9. To filter out these events is probably the greatest challenge when trying to analyze *ANTARES* data.

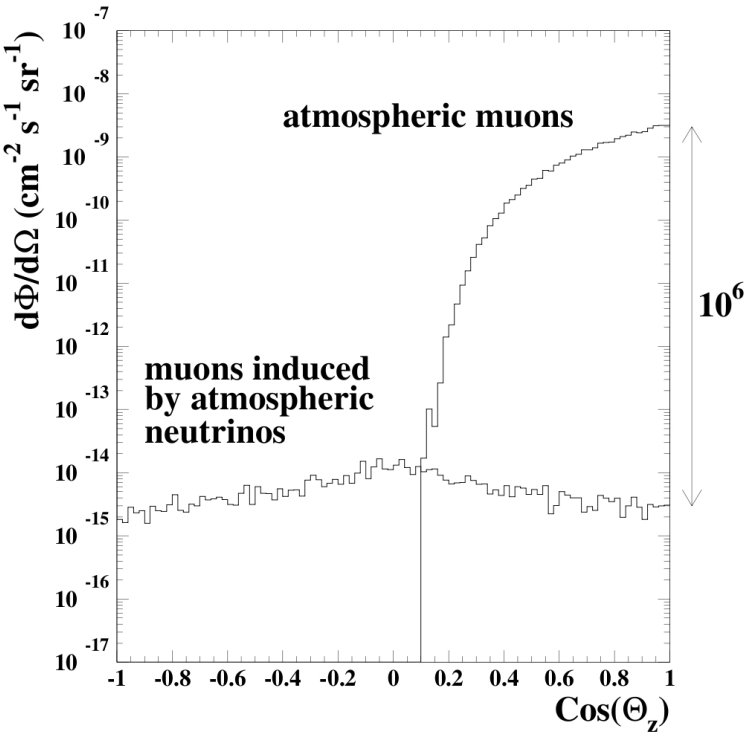

Figure 2.9: *Zenith angular distribution of the muon flux above 1 TeV from atmospheric muons and atmospheric neutrino induced muons at 2300 m water equivalent depth* [23].

While atmospheric muons cause pure background events, it also exists an optical background that produces noise hits that occur in each event, even in those, that are real neutrino events. Due to the β-decay of radioactive ^{40}K nuclides, which are part of the salt water in the sea, a stochastic background noise occurs on every OM. ^{40}K mostly decays to ^{40}Ca emitting one electron that has a maximum energy of 1.3 MeV, which is above the threshold for Čerenkov radiation. A thus produced electron will emit on average about 70 Čerenkov photons [34] along its whole track, which restricts the amplitude of a background hit to barely more than one photon. Therefore in MC simulations the ^{40}K optical noise is simulated by a white noise of single photon hits. The calculated average noise rate at the *ANTARES* site is expected to be 34 ± 7 kHz

for each OM, whereas measurements yield a rate of about 55 kHz [35], which is an indication for some biological contributions to the background counting rate.

At a water depth of several hundred meters, where the sunlight is not able to penetrate to, over 90% of the animals, which are mostly plankton, shrimps and fish, are luminescent, i.e. they are emitting some kind of light for communication, attracting prey or for whatever reason. The duration of such light emissions ranges from a few milliseconds to several days. Some species emit short lightpulses over a period of a few minutes, others glow constantly for hours, or even days. The intensities of these emissions reach from 10^3 up to 10^{11} photons per second. Hence low energy bioluminescent lightpulses are quite similar to the ^{40}K decay and increase thus the counting rates. Bioluminescence of high intensity of course can damage a whole sample of events, depending on its duration and intensity and has to be filtered out completely [34].

In order to filter out such bioluminescent bursts and identify real events in the amount of background hits, a trigger system has been developed, which is described in the following chapter.

2.5 The *ANTARES* software trigger algorithm

This section describes the *ANTARES* software trigger and follows the explanations in [36] and [37].

To provide as much flexibility as possible concerning the data acquisition, *ANTARES* follows the concept of "all-data-to-shore". That means that the LCMs at the detector do not have any hardware trigger systems to prefilter the data, but all data is sent ashore, where software triggers try to identify physics events. This filtering process is done in several steps, where at the end single physics events will be stored on disk. As this trigger works completely without any hardware, it can also be applied to MC events. Figure 2.10 gives a schematic overview of the single trigger stages.

The **Level 0 Trigger** is the first stage, at which a time slice of raw data arrives from the deep sea. Actually Level 0 is not quite a real trigger, as nothing is filtered out here. The data containing an AVC (amplitude vs. channel) and TVC (time vs. channel) value for each hit is calibrated and put into a data format that is required by the following stages. As the photomultipliers have a certain integration time, the TVC value defines the arrival time of the first photon, whereas all further arriving photons within the integration time window will be merged with the first one to one hit. The AVC value thus contains the information about how many photons have arrived within the integration time. One photomultiplier has two ADCs (analog to digital converter), of which each provides an integration time window that used to be 25 ns but has been

changed many times over the years followed by a deadtime interval of 250 ns. Hence after taking data of two times the integration time, the whole PM will be dead for (250 ns - integration time). The dynamic range of the readout was used to be 20 pe but also has been changed several times, and moreover can vary from PM to PM.

The **Level 1 Trigger** does a first preselection concerning two local aspects. This stage searches within the whole time slice for large hits and local coincidences. A large hit is a hit with an amplitude above a particular threshold that is one of many parameters, by which the trigger can be run. Local coinciding hits are two hits that occur within a certain time window on one storey. All hits that match to at least one of these conditions are put together into the list of *L1 hits*. In this thesis all MC studies, whenever a trigger was used, have been made with a large hit threshold of 2.5 photo electrons and a coincidence window of 20 ns. Thus the probability for a L1 hit being a ^{40}K background noise hit is quite low, as hits from ^{40}K are barely bigger than 2.5 photo electrons and it is quite unlikely to have two of them on the same storey in such a short time window.

The *L1 hits* are passed on to the **Level 2 Trigger** that is the actual trigger and searches for correlations between the preselected hits. Correlated hits are those that are causally connected according to space-time relations between the signals from a possible muon track or shower. The speed of light in water is $c/n \approx 22\,\mathrm{cm/ns}$ and, assuming an unscattered light propagation, the maximum time difference for two different hits to be causally connected is

$$\Delta t < \frac{n}{c}\Delta x + x_{\mathrm{corr}} \tag{2.4}$$

where Δx is the distance between the OMs and x_{corr} is a correction that accounts uncertainties of the OM positions and time calibration. The maximum possible time difference is about $t_{\mathrm{max}} = 2.2\,\mu\mathrm{s}$, which corresponds to a maximum Δx of the approximate diameter of the detector that is $\approx 500\,\mathrm{m}$. Multiple trigger algorithms are now available to be switched on independently or successively. Each of them has different requirements on the *L1 hits* to be selected as an event. Originally the aim of this thesis was to compare real 5 line *ANTARES* data with MC simulations, why the MC events in this thesis has been processed with the *3N* trigger, which was the chosen algorithm in the 5 line data taking period in 2007. The *3N* trigger defines an event, if a minimum of *L1 hits* that fulfill the condition (2.4) with a correction of $x_{\mathrm{corr}} = 20\,\mathrm{ns}$ within the maximum time window of $2.2\,\mu\mathrm{s}$ is found. All hits in one event that match to these requirements are referred as *triggered hits*.

In the **3rd level** overlapping events are merged. In order to include all signals that are possibly induced by a physics signal the time span of the *triggered hits* is extended by adding t_{max} after the *first* and before the *last* triggered hit. All hits within this new time window are referred as *snapshot hits* (see also figure 2.10). Two overlapping

events are defined as events, where at least one triggered hit lies within the snapshot of the other. In this case the triggered hits of both will be merged to one event.

Finally **Level 4** builds the event by collecting and combining all hits within the snapshot. For MC files the trigger chain ends with Level 2, since after the MC simulations the hits are already split in single events.

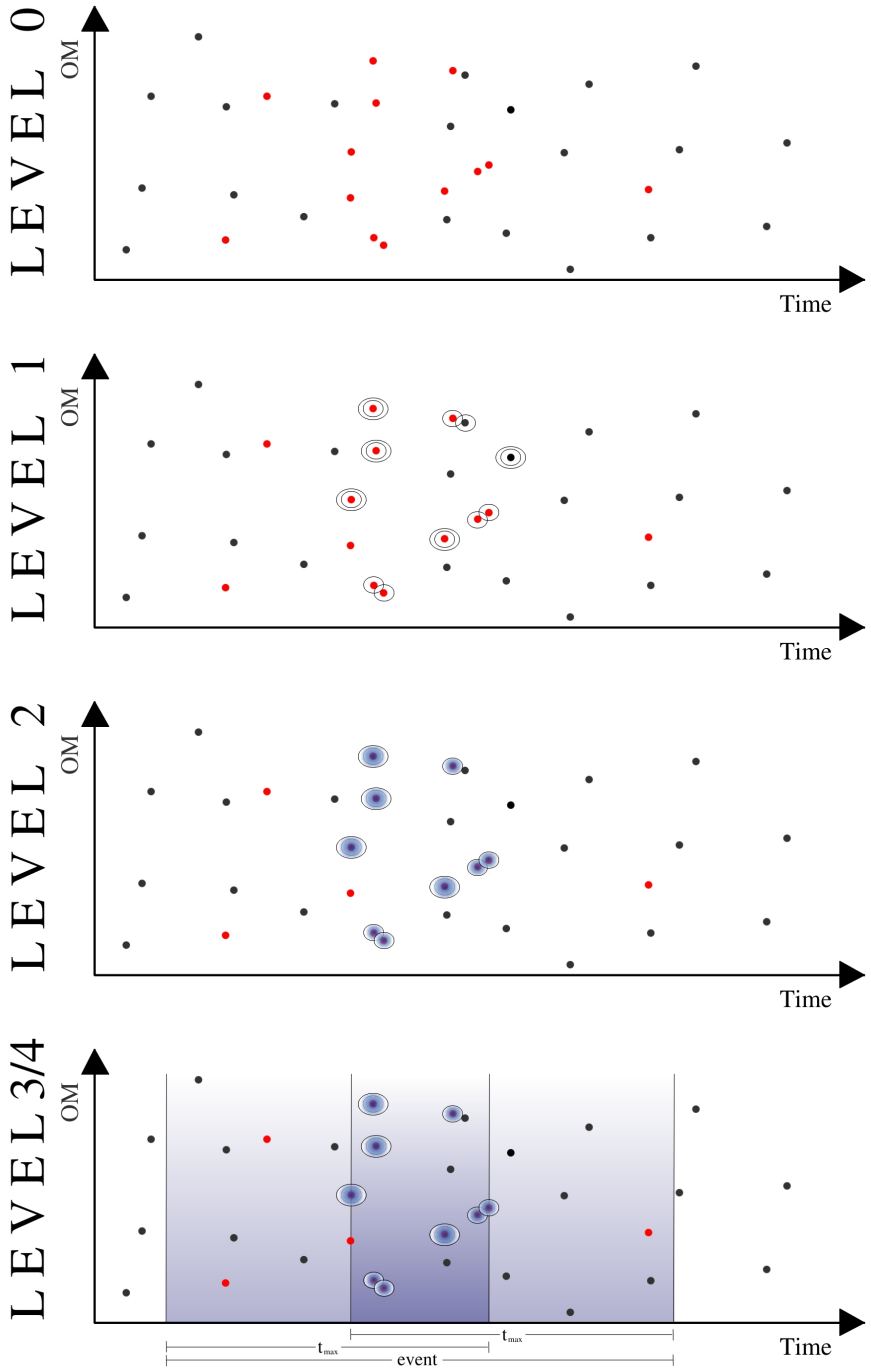

Figure 2.10: *Schematic description of the trigger stages.* **Level 0:** a sample of hits containing physics hits (red) and background (black). **Level 1:** Marked *L1 hits*, where large hits are encircled twice and coincident hits once. Two background hits are also marked as *L1 hits*, as the one seems to have a high amplitude and the other is in coincidence with a true hit. **Level 2:** The causality scan removes the noise hits from the selected hits. The blue colored circles contain the final *triggered hits*. **Level 3/4:** The event is built by adding twice a time window with a length of t_{max}.

Chapter 3

Reconstructing hadronic showers

3.1 The *ShAuerReco* algorithm

In order to make use of the maximum possible detection efficieny for neutrinos a reconstruction algorithm has been developed by Ralf Auer that is called *I3ShAuerReco*[1] [38] and is included in the *KM3Tray* framework (see appendix A for more details on this framework). The algorithm aims at reconstructing events, in which only a shower but no muon track is created, i.e. pure NC events and CC events from an electron neutrino. The electron neutrino gives rise to an electromagnetic shower that is however dominated by the hadronic one and can thus be seen as part of the overall shower. Due to the distance between two storeys in *ANTARES* that is 14.5 metres along the line and 60 meter between two strings and the typical shower extension of less than 10 meter (see Figure 2.6), a shower event is seen as a point-like light source. Therefore the emitted light will be emitted more isotropically into a sphere which makes it more difficult to reconstruct the direction of the primary neutrino.

While consequently the angular resolution will be worse than for muon events, one can expect a better energy reconstruction, as the shower event deposits all its induced Čerenkov light within the detector (unless it takes place at the edge of the detector). Hence the reconstruction of shower events is more appropriate to determine the spectrum of the diffuse neutrino flux instead of looking for point light sources. However, the algorithm reconstructs all shower observables, i.e. the shower energy E, the direction Θ, Φ, the interaction vertex position x, y, z and the absolute interaction time t. Note that the shower energy is not the energy of the primary neutrino, as only a part of the primary energy is transferred to the shower. The remaining energy in NC events will be carried away by the outgoing neutrino.

To determine these seven observables the parameters of a shower hypothesis are

[1] The characters *I3* in *I3ShAuerReco* indicate that the algorithm is part of the *KM3Tray* framework, and will be left out in further mentions.

varied until the probability (likelihood) of the set fitting to a measured signature is in its maximum, which is equivalent to the negative log-likelihood in its minimum. The following procedure is also described in [39]. From the timing information of the single hits one can derive the vertex position, the interaction time and the particle's energy. The energy will of course be reconstructed quite badly, when considering only hit times, as it bears more on the hits' amplitudes. However, the energy is needed in this fit, as the shape of a shower event depends on its energy. A low energetic neutrino will cause an almost spherical photon distribution, whereas a high energetic one makes the shower more expanded along the shower axis. The time log-likelihood that is to be minimized is expressed as the summation over all 900 OMs.

$$-\ln(P^T) = -\frac{1}{900}\sum_{i=1}^{900}\ln[p_i^T(x,y,z,E,t)] \tag{3.1}$$

The probabilities for one hit are obtained from MC simulations and stored in a three dimensional look-up table: relative arrival time of the photon at the OM according the interaction time t, distance of the OM from the vertex position x, y, z and the energy E. Figure 3.1 shows the probability versus the photon arrival time and OM distance.

The same scheme can be applied to the amplitude information, where now six observables, i.e. all except interaction time, are taken into account in the fit. The amplitude log-likelihood hence is expressed as

$$-\ln(P^A) = -\frac{1}{900}\sum_{i=1}^{900}\ln[p_i^A(x,y,z,\Theta,\Phi,E)] \tag{3.2}$$

The look-up table for the amplitude information is shown in Figure 3.2. Here the vertex position and the direction are merged to one parameter α that is the direction of the emitted photon with respect to the shower axis (which is assumed to be the neutrino direction). What is shown in the plot is the projection of the measured amplitude onto the unit sphere around the vertex.

The user can choose, where these two likelihood function are used in the *ShAuerReco* algorithm. In this thesis the time likelihood is used for a prefit that provides startvalues for the final fit. The final fit is then a combined fit over all observables using timing and amplitude information. The combined likelihood can be written as

$$-\ln(P) = -W_A * \ln(P^A) - W_T * \ln(P^T)\ , \tag{3.3}$$

where W_A and W_T are weighting values for the corresponding likelihoods. In this thesis both weights have been set to 1, i.e. both look-up tables are considered as equipollent. This procedure is named *FitType 2* (see appendix B.1 for detailed information about the parameters that control the algorithm and that can be set in a python script).

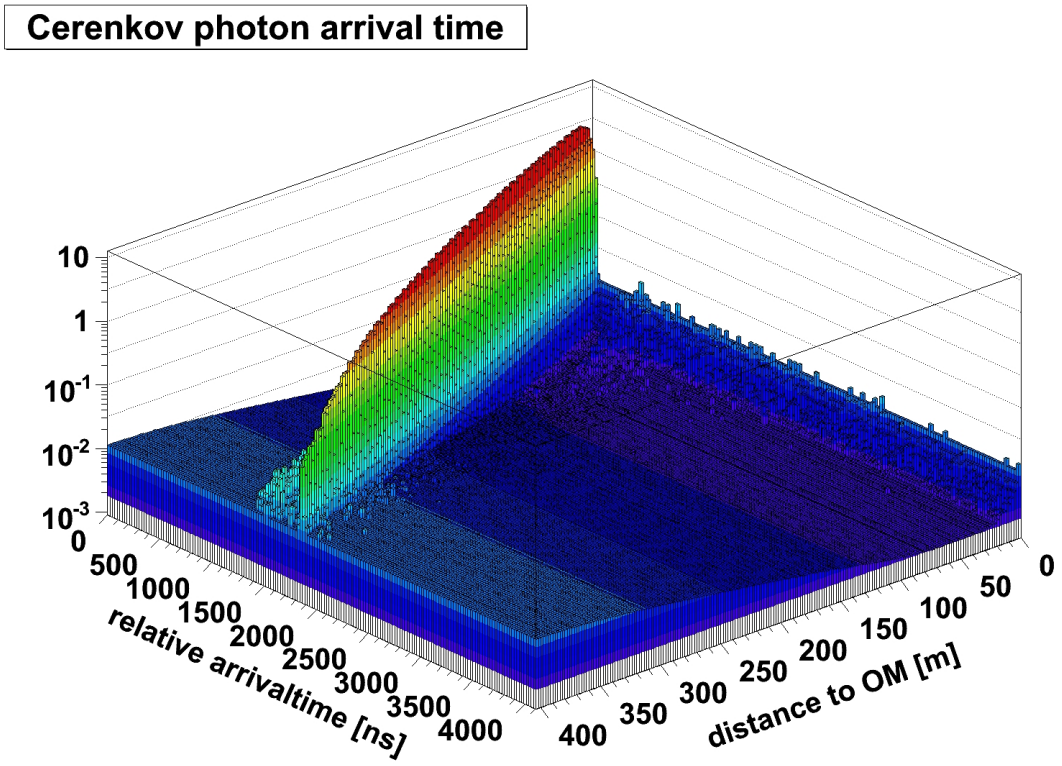

Figure 3.1: *Probability distribution of the Čerenkov photons arriving at an OM, depending on its distance to the vertex and the arrival time relative to the interaction time for shower energies between* $10^2 - 10^7$ GeV. One can see the expected linear dependence according to $v = s/t$ to the left. The broadening is due to effects caused by the data acquisition in *ANTARES*, like PM integration time (see Level 0 Trigger in chapter 2.5). The late arriving photons close to the vertex come from very low energetic shower muons with a low propagation velocity that decay almost where they have been produced. To avoid the division by zero a constant baseline has been added, that can be seen as a wave-like background, which is due to the normalization of all values for each distance bin.

3.2 Minimization by *Simulated Annealing*

The following section gives an outline of the *Simulated Annealing* algorithm that is used to minimize the log-likelihood functions (3.1) and (3.2). As seven observables have to be fitted when reconstruction a shower, it is essential to have a stable minimizer that will reliably find the global minimum in a seven-dimensional space, and not get stuck in a local minimum. Simulated Annealing was proven in many applications to be a good compromise between correct convergence and acceptable computing time. For more detailed descriptions see [40].

The algorithm is motivated by the cooling process of molten metals. After slow cooling (annealing) the metal will reach a lower energy state than a quickly cooled metal, as due to random energy fluctuations an annealed system has more time to

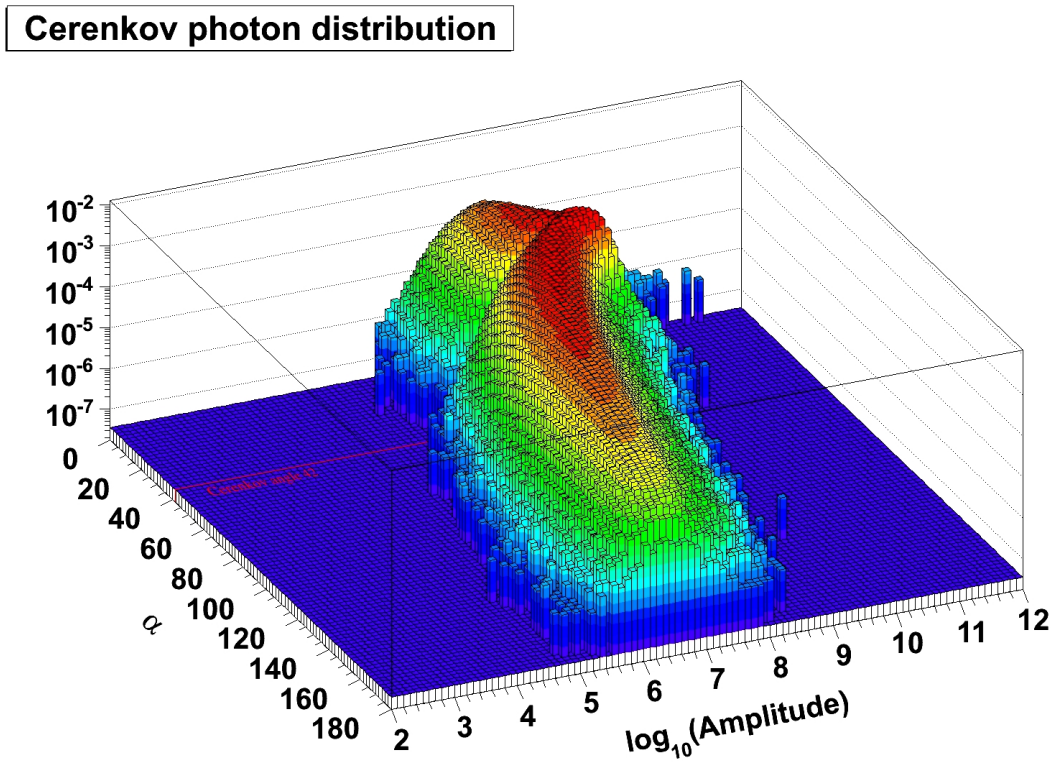

Figure 3.2: *Probability distribution of the Čerenkov photon flux on the unit sphere around the vertex, depending on the angle α between the shower axis and the photon direction and the measured amplitude for shower energies between* $15 - 80$ TeV. The 4π distribution of the photons can be seen clearly, where the probability for backward going Čerenkov photons is of course smaller than for forward emitted photons. At the Čerenkov angle of 42° higher amplitudes can be measured.

escape from local minima. To assign this idea to a minimization algorithm a value T has to be introduced that corresponds to the temperature and that is decreased (cooling) during the minimization process. What is needed as startvalues for the algorithm is an initial temperature T_0, an n-dimensional starting vector $\vec{X}$ containing the n parameters that will minimize the system and a n-dimensional steplength vector $\vec{V}$ that defines the search space for the algorithm. In this case $\vec{X}$ contains seven values, i.e. those for $t, x, y, z, E, \Theta, \Phi$, and $\vec{V}$ has to be adjusted in a way that in all directions the whole detector volume and the chosen energy range is covered by the search space. Figure 3.3 shows a graphical description of how the algorithm works.

The first step is to calculate the value of a function $f(\vec{X})$, which means here to get the log-likelihood value for the given start observables from the look-up tables. Afterwards, $\vec{X}$ will be modified to $\vec{X}'$ by varying the first component (which is t in this example) according to

$$x_i' = x_i + r \cdot v_i \tag{3.4}$$

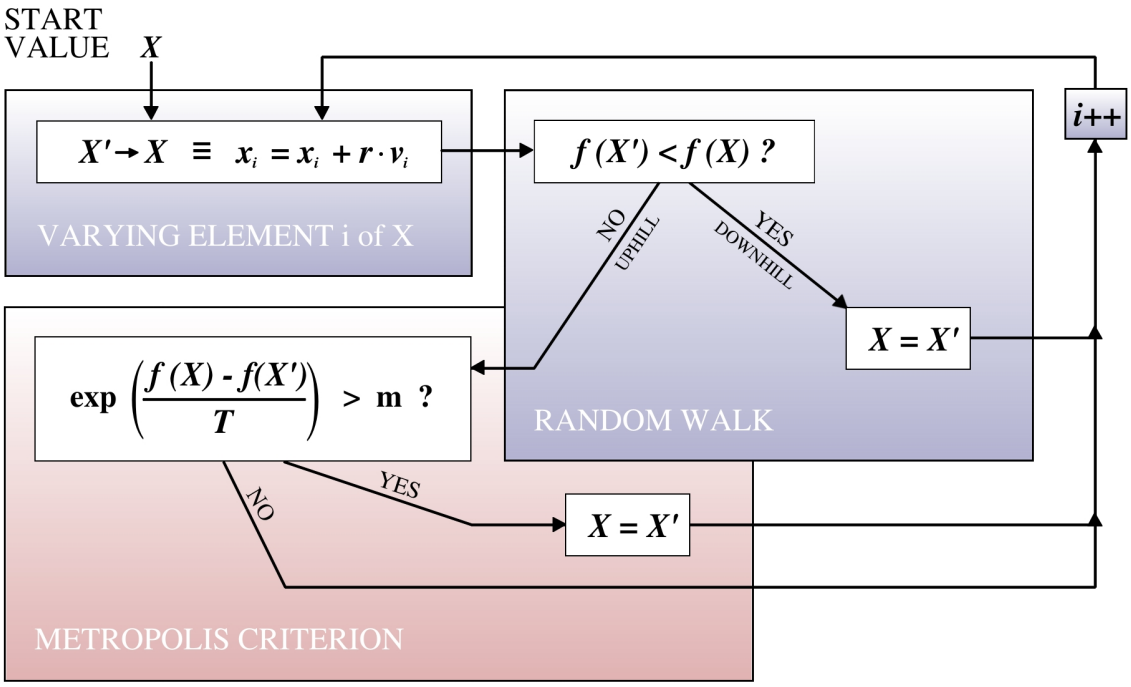

Figure 3.3: *Functional principle of the Simulated Annealing algorithm.* For more explanations see the text in chapter 3.2. The illustration follows a figure from [41].

where i is 1, as it is the first step, v_i is the steplength for this component (i.e. here for time) and r a random number in the interval $[-1, 1]$. Now the log-likelihood value of the new vector $\vec{X}'$ will be looked up from the tables. If it is smaller than that for $\vec{X}$ the *downhill* step to $\vec{X}'$ is accepted and $\vec{X}'$ becomes the new "startvector". Otherwise it would be an *uphill* step, i.e. one in the wrong direction. A standard *random walk* minimization will never accept an uphill step and will therefore risk to be trapped in a local minimum. In order to avoid this, *Simulated Annealing* gives rejected steps a second chance. A criterion motivated by thermodynamics, called the *Metropolis criterion*, decides, whether also an uphill step can be also accepted or not. If

$$\exp\left(\frac{f(\vec{X}) - f(\vec{X}')}{T}\right) > m \tag{3.5}$$

the step is accepted, even if the log-likelihood of $\vec{X}'$ is bigger than that of $\vec{X}$. The value m is a random number in the interval $[0, 1]$.

Then, this procedure will be repeated six times, i.e. for all remaining components of $\vec{X}$. After N_S such steps through all components, the steplength vector will be adjusted so that 50% of all moves are accepted. After N_T of these adjustments the temperature T will be decreased by multiplying it with a reduction factor r_T that can be set to a

value between [0,1[

$$T' = r_T \cdot T \tag{3.6}$$

The algorithm terminates by comparing the minimum log-likelihood values from the last N_ϵ temperature reduction steps. If the differences between these values are not bigger than ϵ the minimization has found its convergence. The other way to finalize the algorithm is to set a maximum of allowed number of steps of the type in (3.4). If this number is reached before the convergence criterion is fulfilled, the algorithm terminates without having converged.

The *Simulated Annealing* algorithm is able to find the global minimum in very large search spaces, which might be the most important advantage, as no prefit is needed. In the *ShAuerReco*, however, a prefit is done to make the algorithm converge more quickly. The probability for an accepted uphill move decreases with lower temperatures and larger jumps in the log-likelihood value. So if the start temperature is set high enough, the algorithm is allowed to jump out of deeper local minima and to do a first rough mapping of the search space topology, where towards the end, when temperature becomes very low, an uphill step becomes more and more unlikely. Another advantage of the algorithm is the independence of the steplength vector's single components. Whereas in many other minimizing algorithms the variation of the startvector $\vec{X}$ is done for all components in common, *Simulated Annealing* varies each component separately (equation (3.4)). This offers great flexibility, as different steplengths can be chosen for each component, which is quite useful, as for example the search interval for t may be a few milliseconds whereas that for E extends over several orders of magnitude.

3.3 Hit constraints for the reconstruction

As the likelihood look-up tables, described in chapter 3.1, are created for pure shower hits without any background noise, the reconstruction will be fragile when trying to reconstruct events with hits that do not originate from the shower. Therefore it is necessary to get rid of these hits. According to the explanations in chapter 2.4 the incidental background hits from ^{40}K decay have mostly an amplitude of only one photo electron. An accidental coincidence of two background hits can create a noise hit of two photo electrons, but higher amplitudes are quite unlikely. Hence cutting away all hits with an amplitude less than three photo electrons will suppress most of the background and make the reconstruction work better.

Due to the more isotropic extension of the shower, the Čerenkov light will not be emitted just at the Čerenkov angle of 42° with respect to the neutrino track, but in all directions (see also Figure 3.2), which gives more ambiguities in directional recon-

struction. Therefore it is required to set a minimum constraint on the hits in an event. Only events with a minimum of five hits, that appear on 5 different OMs and at least 3 different storeys will be reconstructed. This constraint is even stronger than the 3N-Trigger described in chapter 2.5. Hence the greatest part of shower events that pass these constraints will not be filtered out and thus decimated futher by the trigger. On the other hand it is due to the strongness of the constraints, that a lot of, especially low energetic events, will be filtered out in the first place, which decreases the efficiency of the algorithm. For 5 line MC shower events over 90% of all hadronic showers that have already passed the trigger will be filtered out by these constraints. For 12 line MC shower events over 85% don't pass this filter (will be described in detail in chapter 5).

As the fluxes are already very low, it is very unsatisfying to filter out this huge number of events even before any quality cut is done. Unfortunately any attempt to relax these constraints and to increase the efficiency failed (see chapter 4.7), so that the two constraints have to be regarded as a necessary requirement for the *ShAuerReco* algorithm.

Chapter 4

Preparing *ShAuerReco* for real data reconstruction

4.1 The hadronic shower Monte Carlo files

At the beginning of this thesis the *ShAuerReco* algorithm was already tested and worked reasonable well with MC events. Ralf Auer, the original author of the algorithm, published some preliminary results for the reconstruction of a MC shower event sample for a 12 line *ANTARES* detector. The neutrino events were created with an E^{-1} neutrino spectrum to improve statistics for high energy events. The energy range in this sample reaches from 10^2 GeV to 10^7 GeV with neutrino directions distributed isotropically over 4π and a background noise rate of 120 kHz. The paper reports a median of 3.04 m for the vertex position error, a median of 4.77° for the direction error and a RMS of 0.47 for the logarithmic error of reconstructed shower energy after a quality cut that removes all events with a reconstructed vertex outside the intrumented volume [39]. This sample originally contained 4071 events, of which 856 remain after the cuts, which is quite a lot considering the strong hit constraints described in chapter 3.3. Of course, this is due to the simulated spectrum that creates too many high energetic events that pass these constraints and increase the number of successfully reconstructed events. However, in this work the reported resolutions could not be reproduced when using the new MC event files listed below. The analysis of the new MC files shows a slightly worse reconstruction quality. Possible explanations for this are given in chapter 4.8.

The next step was to prepare the existing algorithm for real *ANTARES* data and adapt it to be able to reconstruct and find real hadronic shower events. Therefore, three new MC samples with larger statistics have been generated for a neutrino spectrum that is more consistent with expectations from the atmospheric neutrino spectrum. As the major part of detected neutrinos will be atmospheric neutrinos, the expected spectrum for real events will mostly follow the $E^{-3.7}$ atmospheric neutrino spectrum.

To generate, however, a MC sample with the real atmospheric neutrino spectrum is probably no good idea, as one won't have any high energetic events to study, unless one has enough computing power to generate billions of events. Hence the simulated spectrum was set to E^{-2} and each event has been provided with a specific weight that produces, if applied to the weight in histograms, a sample that corresponds to the real spectrum. The summation over all bin contents in weighted histograms, reffered to as *Integral* in the following, then corresponds to the number of events within a certain time period that has been set to one year within this study. To read more about the practical use of the MC weights, see [42].

All MC samples have been generated for a 12 line *ANTARES* detector using the *GENHEN* [43] simulation package for the particle generation and the *GEASIM* [44] code for the Čerenkov photon tracking and the detector simulation. As the first intention, however, was to study the real data that have been recorded during the 5 line data taking period from February to December 2007, these files have been used for the studies of showers in a 5 line detector by simply removing all hits from the strings 6 - 12. Due to some problems with the event classification (see chapter 5) it was soon forseeable that it is very difficult to recognize any shower event, whereupon the analysis has been extended to 12 line events. In the following chapters the results of 12 and 5 lines studies will always be printed simultaneously to see the differences between the two detector sizes. Thereby plots of 12 line data are printed in blue, whereas 5 line data is plotted in red.

The following tables contain information about the three shower MC samples that have been used within this thesis. These samples have been generated assuming muon neutrinos as primary particles. But as NC reactions can be induced by all three neutrino flavors and as one expects almost the same abundance for all three flavors due to neutrino oscillations, the expected number of triggered events from the MC files has to be multiplied by a factor of 3. This factor is already considered in the following tables.

Sample 1

Generated neutrino spectrum	E^{-2}
Neutrino energy range	10^2 GeV to 10^7 GeV
OM angular acceptance	original measured acceptance [30]
Generation volume	*ANTARES* 12 line detector + 55 m can[1]
Neutrino direction of incidence	4π isotropically
# of generated events	50000
# of 12 line events	32271

[1] One absorbtion length in water around the detector size

# of 5 line events	18311
expected triggered 12 line events per year	1374
expected triggered 5 line events per year	503

Sample 2

Generated neutrino spectrum	E^{-2}
Neutrino energy range	10^2 GeV to 10^7 GeV
OM angular acceptance	original measured acceptance [30]
Generation volume	*ANTARES* 12 line detector + 55 m can
Neutrino direction of incidence	4π isotropically
# of generated events	100000
# of 12 line events	64046
# of 5 line events	36346
expected triggered 12 line events per year	1329
expected triggered 5 line events per year	515

Sample 3

Generated neutrino spectrum	E^{-2}
Neutrino energy range	10^2 GeV to 10^7 GeV
OM angular acceptance	Genova acceptance [45]
Generation volume	*ANTARES* 12 line detector + 55 m can
Neutrino direction of incidence	4π isotropically
# of generated events	100000
# of 12 line events	67783
# of 5 line events	39156
expected triggered 12 line events per year	1926
expected triggered 5 line events per year	724

Sample 1 and *sample 2* differ only in the number of generated events. The interaction vertex positions of all three samples have been generated isotropically within the *ANTARES* cylindrical 12 line detector plus one absorption length of light in water around the detector, referred to as *can*. The angular acceptance of the OM in the first two samples, i.e. the sensitivity of the PM with respect to the photon direction of incidence, has been derived from lab measurements. However, first measurements of real atmospheric muons showed a greater flux than predicted by MC studies, whereupon the angular acceptance has been remeasured recently in a lab in Genova and thus has been named *Genova* acceptance [46] [45]. These new measurements showed a higher acceptance also for larger arrival angles with respect to the axis of the OM. Therefore the event numbers are higher in *sample 3* than in *sample 2*, which are apart from the

angular acceptance generated in exactly the same way. The number of 12/5 line events includes all events that give at least one hit in the according detector. The expected events per year can be derived from the MC weights.

4.2 Interaction time in MC and real events

When trying to prepare the *ShAuerReco* for real data one important point is to find suitable start values and fit boundaries for the *Simulated Annealing* minimizer, of which the fit value *interaction time t* is the one that requires most of the attention. In shower MC events the absolute time, when the throughgoing neutrino interacts with a nucleus and creates the shower, is per definition *always* 0. So what has been done when reconstructing MC shower events is: Set the start value for the interaction time fit to 0, give the minimizer enough space for the fit and go on. Of course in the case of MC data, one knows that the interaction time *is* 0, so it seems quite logical to adapt the fit boundaries around this value and make them not too big, to avoid long computing times and the risk of incorrect convergence in too big search spaces. Unfortunately for real data it is not possible to proceed like this, as one does not know the interaction time and thus how to adjust the boundaries. Available is just a list of hits, of which one first does not know, which hits come from the real physics event and which of them are just background.

There are two possibilities now. The first one is to make a rough estimation about what the interaction time could be and adjust quite big fit boundaries according to the estimated value. But as this estimated time could also be completely off, the boundaries have to be big enough to make sure, that the real interaction time lies within them. However, big fit boundaries are never a good solution due to two reasons: The fit procedure is one in seven dimensions and thus is already complex enough. To enlarge the fit boundaries would extend the computing time up to intolerable dimensions, as with larger fit boundaries also the maximum number of iterations has to be increased to give the minimizer the chance to converge. The other reason is that extending the fit boundaries may create a bigger number of possibilities (i.e. more minima) and run the risk that the minimizer gets stuck in one of these false minima. Later will be shown that it is due to the complexity of the seven-dimensional likelihood function that the *Simulated Annealing* fails to find the real global minimum the more often the bigger the search space is set. Hence it is better to take the second possibility: keep the boundaries as small as possible and determine the estimated interaction time as exactly as possible.

4.3 Start values and fit boundaries for *Simulated Annealing*

The analysis of the *ShAuerReco* algorithm throughout the following chapters requires the generation of hundreds of plots of all kind of parameters and values. To create these plots the ROOT data analysis package [47] has been used. In order to avoid the time consuming process each time when generating a plot, first a plotting tool had to be implemented to accelerate and simplify the generation of many plots. This tool should be generic enough to provide its services to other possible users as well. The so-called *Root-plots* project, is described in detail in the appendix C.

4.3.1 Start value for the interaction time fit

The estimation of the interaction time to find a suitable start value for the time fit confronts one with two problems. The first one is to separate the physics hits within an event from background hits. The probably best way to come as close as possible to the real interaction time is to identify the first physics hit in an event. But as in an event with a lot of background this might be difficult, one could also imagine to take the hit with the biggest amplitude as start value, as due to the nature of the background (see chapter 2.4) it is, at least for high energetic events, quite likely that the biggest hit comes from the shower. The second problem is that, if one hit is once assigned as the start time defining hit, one does not know how "far" it is away from the true interaction time. Of course the vertex interaction has to take place *before* the first/biggest hit appears, thus to avoid unnecessary big fit boundaries the upper limit for the time fit can be set to this hit time. But when doing so, one has to ensure that the algorithm that determines the start value will **never** accidentally choose a hit that is earlier than the interaction time, as in this case the *ShAuerReco* will have no chance to reconstruct this event!

For the list of possible hit candidates to define the start value also the first or biggest of the *triggered hits* has to be taken into account, as the aim of the software trigger is to filter out background hits and to identify the physics event (see chapter 2.5). However, due to accidental coincidences too many background hits still pass the trigger and thus the whole reconstruction will fail if only one of them appears before the actual interaction time. As the *ShAuerReco* requires strong hit constraints (i.e. only hits with an amplitude above 3 pe) to suppress background hits for the reconstruction (see chapter 3.3), the first of *these* is probably a true hit. But as these constraints reduce the efficiency of the algorithm enormously, they probably have to be to replaced by a more efficient hit selection one day, and should not be considered as the first choice to

force the assured identification of the first true hit. In order to finally find the right hit, a background noise of 120 kHz has been added to the MC *sample 2* (see chapter 4.1) within a time space from 500 ns before the first hit to 500 ns after the last hit in each event to simulate real events taken with the detector. The sample then has been reconstructed as 5 line data.

The time distributions for the appearance of the first hit are shown in Figure 4.1 after different hit filters. Due to the background noise in the unfiltered sample of course the first hit is mostly a noise hit. In MC events one knows the interaction time that is 0, i.e. all first hits in the dark red colored interval are definitely noise hits that are not to be selected as start value. Even after the software trigger many noise hits that are the first remain in the sample. However, the trigger is necessary, as the *ShAuerReco* hit constraints also let pass some noise hits that could be the first one (in this case this is just in one event). Only the combination of trigger and hit constraints filters out all noise hits and ensures that no hit that is earlier than the actual interaction time will be regarded as start value.

The distributions for the biggest hit are shown in Figure 4.2. To look for the biggest hit is probably a more confident criterion to set the start value for the time fit, as noise barely contributes to big hits. However, it can happen that - especially for low energetic events - coincidences of noise hits will create the biggest hit that possibly might be the first. After the sample passed the trigger, the *ShAuerReco* hit constraints or both of them, the biggest hit indeed appears after the interaction time, which makes the biggest hit in a filtered sample of hits a good candidate for the start value. But as it mostly appears later than the first hit (compare the mean values to those of Figure 4.1) the thus selected startvalue would be more off the true interaction time than a first hit is. This would require a bigger search space for the time fit, and thus the *first triggered hit with an amplitude bigger than* 3 pe is probably a better choice, as it is now definitely a true hit as close as possible to the true interaction time.

4.3.2 Fit boundaries for the interaction time fit

After having chosen a suitable start value for the interaction time fit in chapter 4.3.1 that is the first triggered hit with an amplitude bigger than 3 pe, now the search space for this fit has to be set. As the neutrino interaction hardly will take place *after* the first hit that it causes, the upper limit for this search space is of course the start value. However, to avoid some edge effects that might occur when the minimization reaches its boundaries, the upper limit will be set to 10 ns after the selected start value.

The question now is how much space to give the minimizer in the other direction. The plot down right in Figure 4.1 shows that for some events the first hit appears

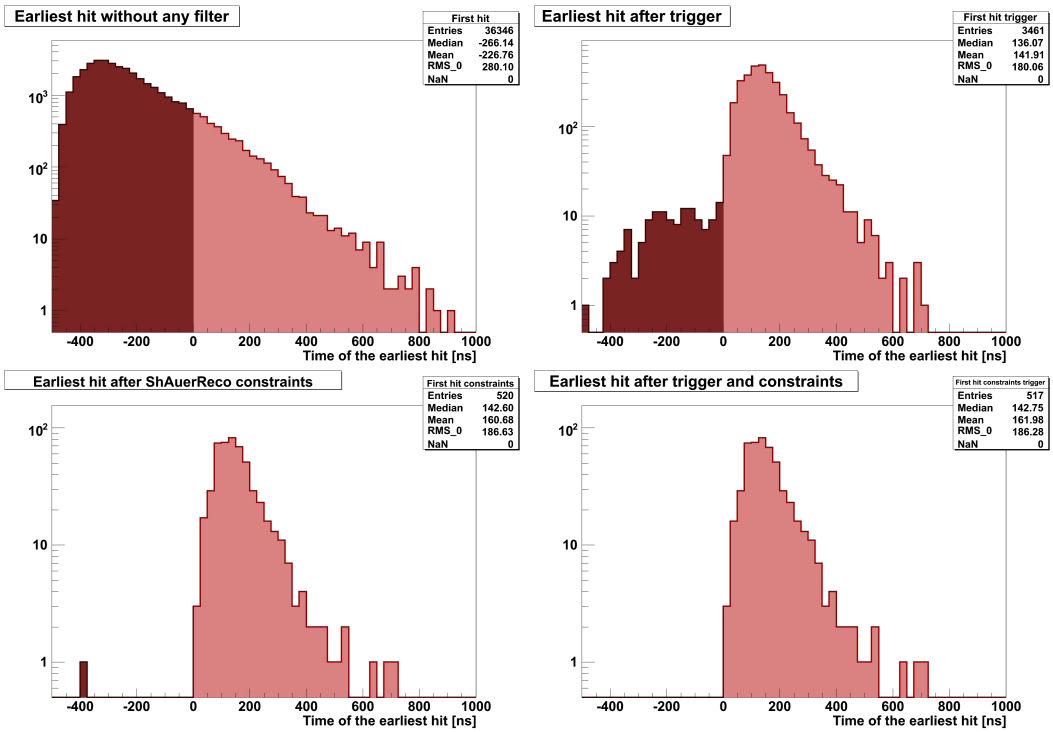

Figure 4.1: *Arrival time of the first hit in a sample of shower events.* The dark red colored interval marks the area, where the first hit appears *before* the actual interaction time.

even 700 ns after the interaction time, which corresponds to a distance of 154 m, i.e. almost half of the detector height. To set the lower limit e.g. to −300 ns with respect to the start value, as it was usually done, means that for these "ultimate" events the reconstruction has no chance to find the true interaction time that is at a time up to −700 ns. One could also imagine an interaction vertex at the top right of the detector and a first hit down left at the bottom of the detector, which means a distance of almost 400 m (the diameter of the 5 line detector) and an arrival time of the first hit of 1800 ns. On the other hand it is quite unlikely that such an event would have a chance for a good reconstruction or pass the trigger anyway.

To find a suitable lower limit, *sample 2*, including background noise like above, has been processed with the *ShAuerReco* with the first triggered hit as start value and various lower limits for the search space. The fit boundaries for the vertex location have been set to ±200 m for the x and y axes and ±300 m for the z axis around the center of hit amplitudes (see chapter 4.3.3). Figure 4.3 shows the resolutions for the different lower limits.

Plotted is the median of the error distributions (i.e. the absolute difference between the fitted and the MC value) for the direction and the vertex position. For the shower

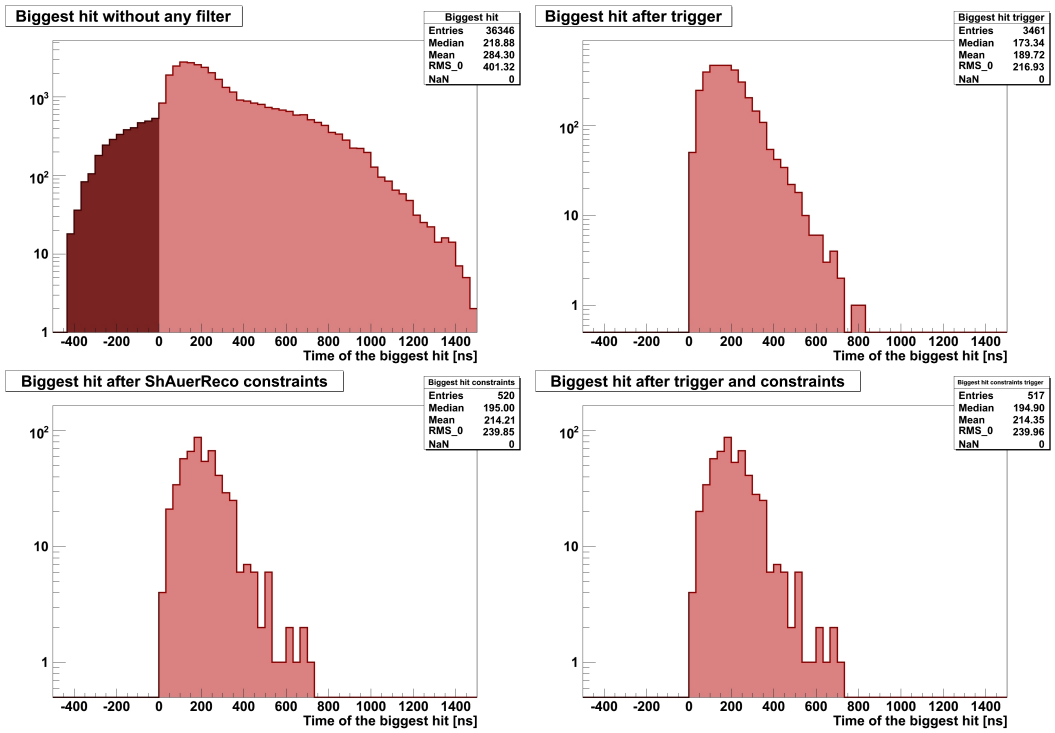

Figure 4.2: *Arrival time of the biggest hit in a sample of shower events.* The dark red colored interval marks the area, where the first hit appears *before* the actual interaction time.

energy error the RMS_0[2] of the distribution of log_{10}(fitted energy/MC shower energy) is plotted.

The plots show that it is not really neccessairy to take care about what might happen in events with extremely late arriving first hits, as the minimization anyway won't work, if the lower limit is set too low. As the first studies have been done with 5 line data, the optimum value for the lower limit has been set according to the best fit results for 5 line data, which is at -300 ns. If the search space for the time fit is set too small, i.e. a lower limit closer to 0, of course the reconstruction cannot work for most of the events, as their true interaction time lies outside the fit interval. But when offering more search space to the minimization the quality also becomes quite bad, which is a little surprising at first sight. Obviously for too big search spaces *Simulated Annealing* prefers a different possibility to the real global minimum.

In the following a possible explanation of what happens here is given. Figure 4.4 shows detailed fit results of the *ShAuerReco* 12 line runs, one for the optimum lower limit of -300 ns and one for the worst case -1000 ns. The absolute number of entries is smaller than in the plots above, as these plots have been created using the *ANTARES*

[2]The RMS_0 is the root-mean-square with respect to a mean value that is forced to 0. This offers a better statement on the reconstruction quality as in the best case the error is expected to be 0.

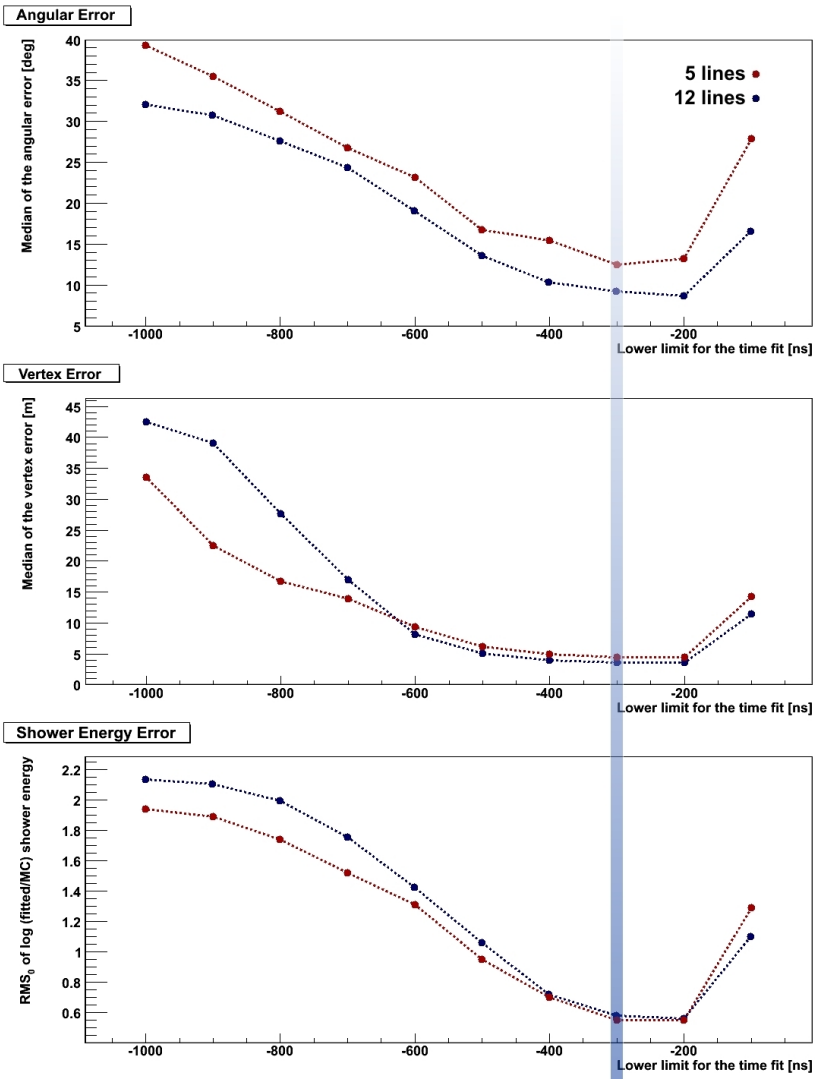

Figure 4.3: *Reconstruction quality depending on the lower limit for the time fit.* The lower limit is respective to the time of the first triggered hit that defines the start value for the fit. The blue lines represent a 12 line reconstruction, the red lines are for 5 lines. The blue bar signals the finally chosen value for all upcoming fits.

MC weights, i.e. the numbers represent a data taking period of one year.

Obviously the following happens for bad reconstructed events: The minimizing algorithm tries to fit a track and an interaction vertex that makes the hit distribution look spherical in shape, which is assumed to be the case for shower events. The quadrupole moment is a measure for the spatial hit distribution along its track. If it is 0, the hits are arranged spherically around one point on the track (i.e. exactly that shape that is expected for showers). A quadrupole moment of 1 corresponds to a hit distribution on a thin disk perpendicular to the track, where as a negative value stands for a tube

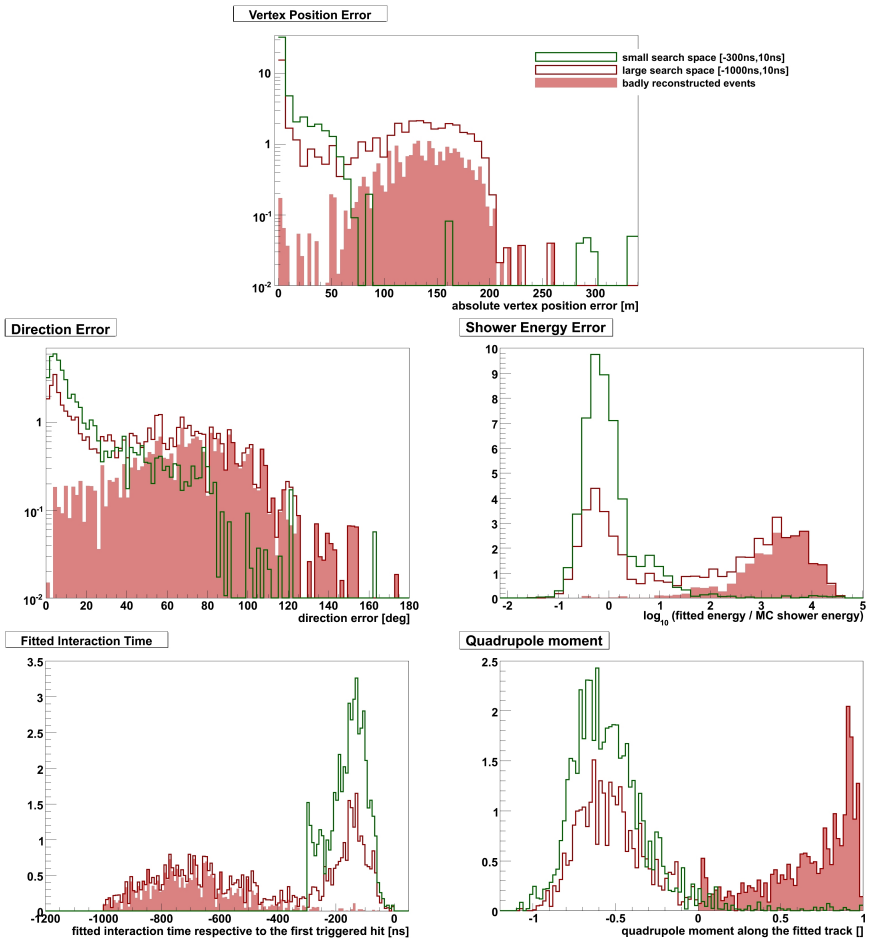

Figure 4.4: *Reconstruction results for two different lower limits for the time fit.* The green line represents the reconstruction with the optimum lower limit of −300 ns, the red line that for a lower limit of −1000 ns. The red colored area marks events that are obviously reconstructed very badly (see text for further explanations).

like distribution with its longest axis along the track. This tube is the more extended the smaller the value is (the minimum is at -2). The quadrupole moment for hadronic showers along the MC track is between −1 and 0, especially events with higher energies are no longer strictly spherical in their shape but a little bit elongated.

The quadrupole moments in these plots have been calculated with the same hits that have been used for the reconstruction, i.e. all hits with an amplitude bigger than 3 pe. One can see that obviously a fraction of the events that have been well reconstructed for the −300 ns limit (green line), separate from the rest concerning its quadrupole moment when being reconstructed with a lower limit of −1000 ns. All events that have a quadrupole moment bigger than 0 are marked as a red colored area in the plots. These events that have obviously been fitted quite badly have a reconstructed interaction time

that is in its maximum 700 ns too early. In 700 ns the photons cover a distance of about 150 m, which fits perfectly to the maximum in the vertex position error of these false fitted events. Characteristic for these events is furthermore a direction error of about 90° in their maximum and a fitted energy that is about $3-4$ magnitudes too high.

What might happen, if the algorithm is given too much space in the fit interval, is that the *ShAuerReco* finds another way to fit an interaction vertex, respective to that the hit distribution seems to be spherical again. Figure 4.5 shows an illustration of what is described in the following. By simply turning the track through 90° with respect to the true direction and fitting a vertex position that lies far outside the hit distribution the spherical shape is restored. The hits now represent a small fraction of the surface of a globe (the blue one in the figure) that is much bigger than the extension of the hit distribution. This new composition of hit distribution, vertex and track increases the quadrupole moment to almost 1 as after the turn through 90° the track now is perpendicular to the thin disc of hits, which fits perfectly to the plots in 4.4. Moreover the fitted shower energy has to be increased by far as the hits and their amplitudes are still the same but the vertex has a bigger distance to them now than before the turn. An event that causes the same hit amplitudes but has taken place further off must have a higher energy than one that occurs close to the hits. The increase of the energy is also verified by the plots. Of course such high energetic events having taken place in the center of the blue globe would cause hits on the whole globe's surface and spherically all around the vertex, but as the *ShAuerReco* algorithm only takes *detected hits* into account and does not take the information of all the OMs with *no* detections, the sample of hits on the small fraction of the globe's surface is also an allowed signature for the shower event. In this case it is assumed that the photons, which are emitted in the remaining directions, simply hit no OM. This hypothesis seems to fit perfectly to the plots in 4.4. What, however, remains unsolved is the question, what kind of hits are susceptible to this kind of rotation.

To avoid this problem one can either choose a small search space (this is what has been done) or give the timing information from the look-up table a higher weight than the amplitude information. The arrival times of the photons on the OMs must differ for the true and the false interaction vertex. Respective to the true vertex in Figure 4.5 a hit at a distant OM has a bigger arrival time than one at an OM close to the vertex. Regarding the false fitted vertex, however, one would expect almost the same arrival time for all hits, as they all are located on the sphere. So if the weight for the timing information W_T in equation (3.3) is set higher than W_A for the amplitude information, this problem possibly can be avoided. On the other hand decreasing the weight of the amplitude information will lay the stress of the reconstruction on the vertex reconstruction and *not* on the shower energy that is actually the more interesting parameter concerning showers. Thus simply the search space has been restricted as

described above, instead of changing the weights.

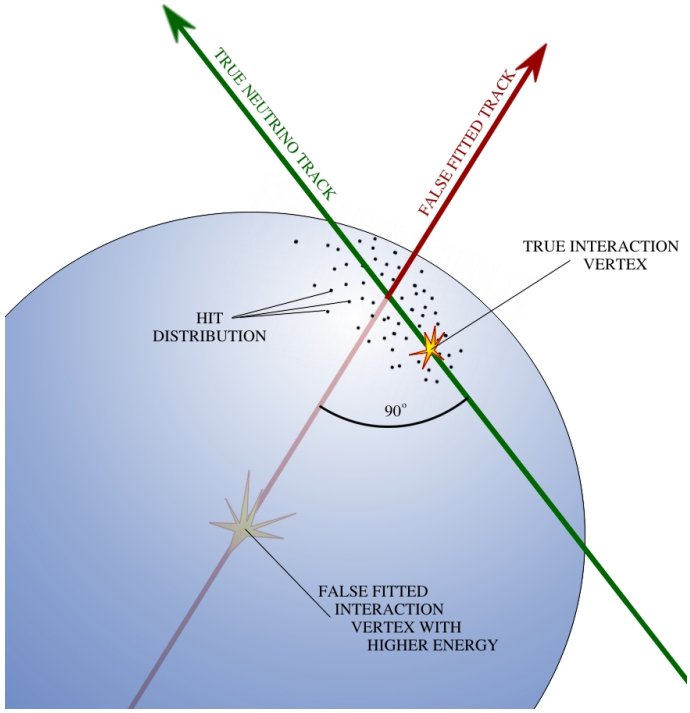

Figure 4.5: *Illustration of a false fitted track that occurs, when the search space for the interaction time fit is too large.*

To finally conclude the search for suitable boundaries for the time fit, the results are outlined: When choosing a too large search space for the time fit the algorithm will not work. Therefore the lower limit has been set to $-300\,\text{ns}$ with respect to the first triggered hit and risk therefore to lose some events that have a late arriving first triggered hit above 300 ns after the interaction. However, these events probably are either low energetic events with only a few hits or come from an interaction that took place outside the detector and will probably be badly reconstructed anyway.

4.3.3 The vertex position fit

To choose a start value and corresponding boundaries for the vertex position fit is quite simple, after the work is already done for the time fit. Of course, the search space for time and that for the interaction vertex are connected to each other by the well-known equation

$$c_w = s/t \tag{4.1}$$

where $c_w = 22\,\text{cm/ns}$ is the light velocity in water. A suitable start value for the location fit has already been implemented for MC events and can be adopted to real

events too. There are two possibilities to set this start value. The first one is simply the center of the detector. This, however, is no good idea, as the space for the interaction time fit has been restricted to −300 ns before the first triggered hit, which corresponds to a distance of 66 m. Due to equation (4.1) the fitted value for the vertex position can be thus at most a sphere with a radius of 66 m around the detector center. All vertices that are located at the outer regions of the detector can then not be reconstructed. So it is better to choose the second possibility, which is the center of gravity of the hits' amplitudes. As showers have an extension of less than 10 m and the center of amplitudes lies certainly inside the hit distribution, 66 m should be enough to find the interaction vertex. However, after Figure 4.6 there are some events with a fitted interaction vertex that is more than 66 m off the center of amplitude. In these cases *Simulated Annealing* obviously assumes a vertex location with a bigger distance and stops the minizing at this point, when it does not succeed in finding a better possible state with a lower log-likelihood. Of course these events are then reconstructed quite badly but the hope is to be able to filter out muon track events by this criterion, as for track events this phenomenon is expected to happen more often. (for more details on this see 5.2.4).

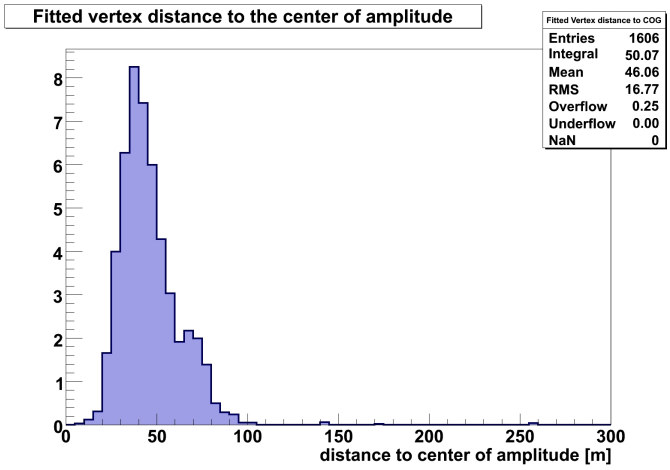

Figure 4.6: *Distance of the fitted vertex location to the center of amplitude.* The fit has been done on weighted 12 line MC with the optimum fit limits for the time fit of [−300 ns, 10 ns] regarding the first triggered hit and a search space for the vertex location fit that exceeds the restrictions of the time boundaries. According to equation 4.1 the fitted vertex is expected to be within a radius of 66 m around the center of amplitude for well reconstructed events.

The fit boundaries respective to the start location thus have to be at least 66 m in each direction to not restrict the physical search space further. If the boundaries are set to closely, too much real interaction vertices may lay outside the search space. Broader boundaries in the contrary will have no effect on the fit results, because the fit is already restricted by the −300 ns$\hat{=}$66 m criterion, as described above. Figure 4.8 proves the expected behavior. Plotted is again the median of the error distributions

(i.e. the absolute difference between the fitted and the MC values) for the direction and the vertex position. For the shower energy error the RMS_0 of the distribution of log_{10}(fitted energy/MC shower energy) is plotted. Small variations in the resolutions are due to low statistics, which will be explained in chapter 4.5. For fit boundaries bigger than $\pm 300\,\mathrm{m}$ the resolutions don't vary anymore as the whole search space already exceeds the size of the *can*. The *ShAuerReco* before fitting checks the size of the search space and adjusts it to the size of the can, if parts of it are beyond the can borders. This adjustment is illustrated in Figure 4.7 and is done for the same reasons for which the time fit boundaries have been restricted (see also chapter 4.3.2).

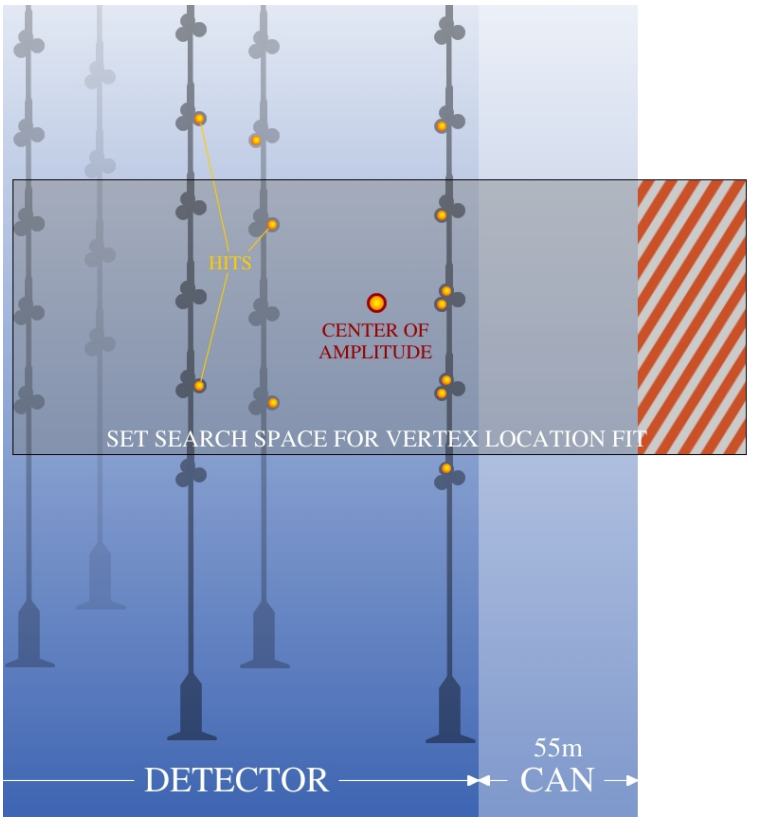

Figure 4.7: *Illustration of the adjustment of a vertex location search space that exceeds the size of the can.* From the hits (yellow points) the center of amplitude is calculated, to which the search space for the vertex location fit is adjusted to. In this case the search space has boundaries that exceed the size of the can, why the *ShAuerReco* removes the areas outside (the red marked area).

The outcome of this chapter is that due to the restricting boundaries for the time fit it is insignificant, which boundaries for the vertex location fit are chosen, as long as they are not too small. Above a limit of about 66 m regarding the center of amplitude, which is the start value for the fit, no observable changes can be seen in the fit results. So the fit boundaries have arbitrary been set to $[-130\,\mathrm{m}, 130\,\mathrm{m}]$ for x and y and $[-180\,\mathrm{m}, 180\,\mathrm{m}]$ for z. These limits are set not too big in order to avoid an unnecessary long computing

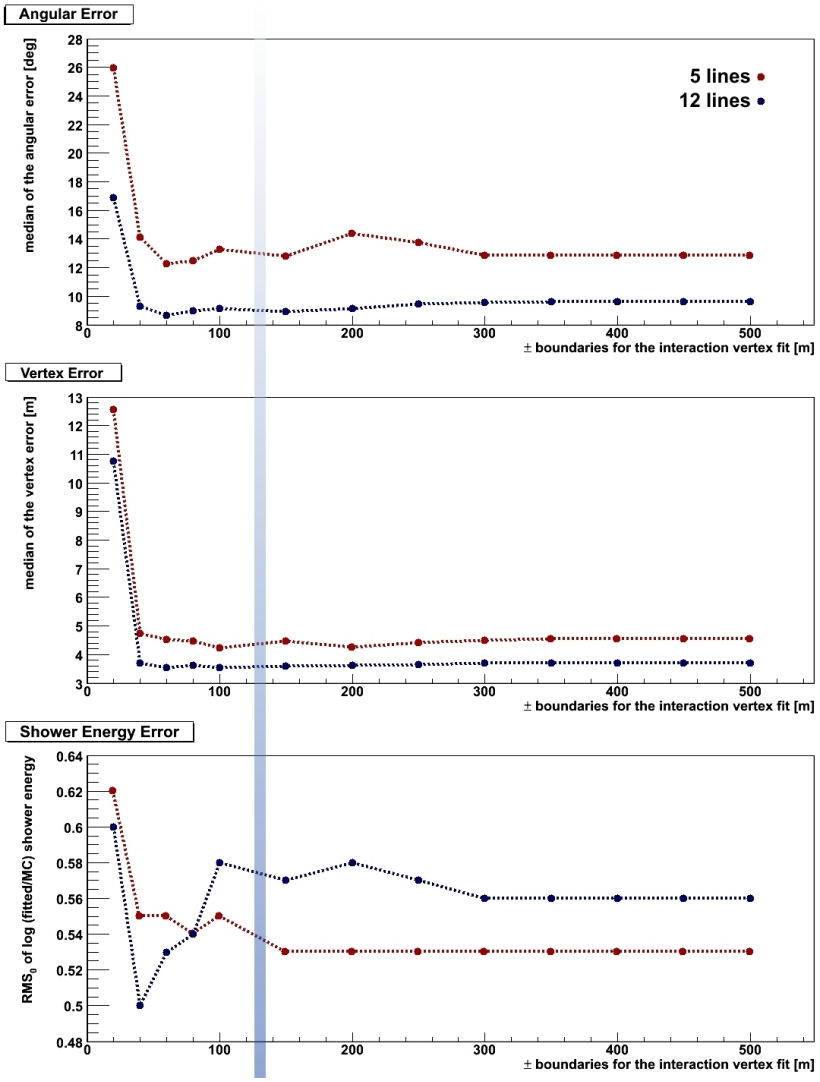

Figure 4.8: *Reconstruction quality depending on the boundaries for the vertex location fit.* The limits have been set symmetrically around its start value, the center of amplitude. The blue lines represent a 12 line reconstruction, the red lines are for 5 lines. The fits have been done with a search space of $[-300\,\text{ns}, 10\,\text{ns}]$ for the interaction time fit regarding the first triggered hit. The blue bar signals the finally chosen value for all upcoming fits.

time, but are big enough to offer the algorithm the possibility to create those unphysical vertex locations which will help us later to filter out events that don not come from showers but from muon tracks (see chapter 5).

4.3.4 The energy and direction fits

Concerning fit start values and boundaries for the energy and direction fits one has no choice without a dedicated prefit, but to offer the algorithm the whole possible range in energy and 4π solid angle. Later there will indeed be a cut on all events with a downgoing direction to suppress the atmospheric muon background, but if the direction is already restricted before the actual fit, all downgoing events will be fitted as upgoing and thus one loses the possibility to cut on the zenith direction.

Thus the boundaries are set to $[0, \pi]$ for Θ, $[0, 2\pi]$ for Φ and $[1, 10^7\,\mathrm{GeV}]$ for the shower energy. The start values have been set to the middle of the search space for the angular parameters and the median magnitude for the shower energy, i.e. $\pi/2$, π and $10^4\,\mathrm{GeV}$.

4.4 Differences between 5 line and 12 line MC events

This chapter is a short description of the differences between the 5 line and the 12 line reconstruction. The studies in the chapters before included first comparisons between the two detector sizes concerning their reconstruction quality. From the plots in the Figures 4.3 and 4.8 it can be derived that the improvement in resolution when replacing the 5 line detector by a 12 line detector is just 4° in angular resolution, 1 m for the vertex position and about 0.05 magnitudes in logarithmic energy resolution for the final chosen fit boundaries, which is just a little improvement[3].

There are two reasons for building huge detectors with thousands of OMs and a huge number of strings: One is to increase the number of events taken place per time in order to enlarge statistics. The other one is to improve the resolutions by being able to detect more information (more hits) from one event. For muon track events, which those detectors are originally designed for, this improvement can be seen, as muon tracks can be several kilometers long (see Figure 2.6) and a bigger detector of course includes more of this track. Concering showers a bigger detector increases the flux of detectable events, whereas the improvement in reconstruction quality is negligible and especially for the shower energy, which is the observable one is mostly interested in, there is practically no improvement.

The reasons why it makes no difference, whether a 5 line or a 12 line detector is used, is shown in Figure 4.9. No matter which detector size is chosen, the greater part of the events will rarely hit more than 5 strings, assuming the same distance between the strings for both detector sizes. The only way to improve the reconstruction quality

[3]The resolutions mentioned here are without any quality cut!

for showers is to increase the density of OMs by placing the strings more closely to each other. But as this cannot be simply done, the only improvement for shower reconstruction resolutions can be expected for some high energetic showers in the TeV region, which of course can hit more strings. The resolutions calculated in this studies, however, are derived from the whole sample's energy range, wherein these few high energetic events play no role.

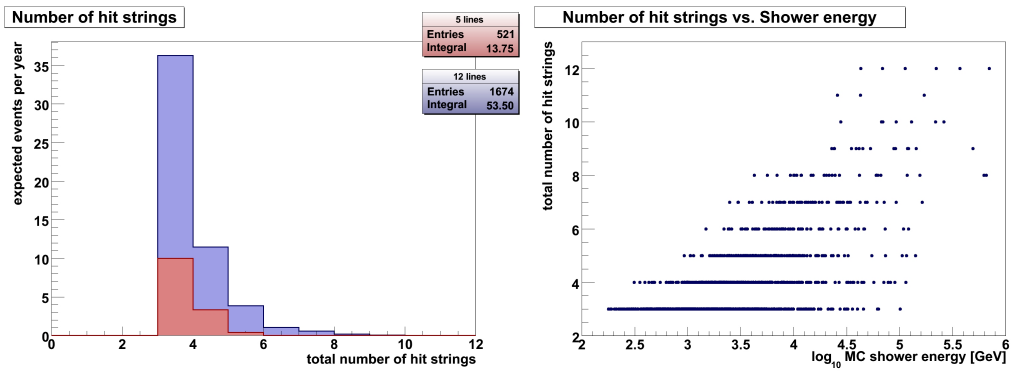

Figure 4.9: *Total number of hit strings in successfully reconstructed events after the ShAuerReco hit constraints.* The histogram to the left has been generated for weighted 5 line events (red) and weighed 12 line events (blue). The graph to the right shows the number of hits strings depending on the MC shower energy of the (unweighted) events.

4.5 The influence of the background hit rate

In order to find a suitable search space for the fit boundaries, the *ShAuerReco* has reconstructed *sample 2* already quite a lot of times with different parameters for each run. For all reconstructions until now a white background noise at a constant rate of 120 kHz per PM has been added to the shower sample. To assume such a high background rate is a little pessimistic, as according to [35] the measured average ^{40}K noise rate is about 55 kHz. On the other hand ^{40}K is not the only background source in *ANTARES*. So it is quite interesting to know, if the reconstruction quality improves for lower background rates or even worsens, if the rate gets even higher during some data taking runs. The *ShAuerReco* hit constraints include a filter that removes all hits with an amplitude that is smaller than 3 pe, which actually should filter out all those background hits. But at a certain rate level there will occur more and more accidental coincidences between noise hits and true hits within one integration time window of the PM, and thus a hit may be created whose amplitude information contains also noise and that thus passes the constraints. Figure 4.10 shows the influence of various background rates on the reconstruction resolution.

What can be seen in the plots is that with an increasing background rate, the reso-

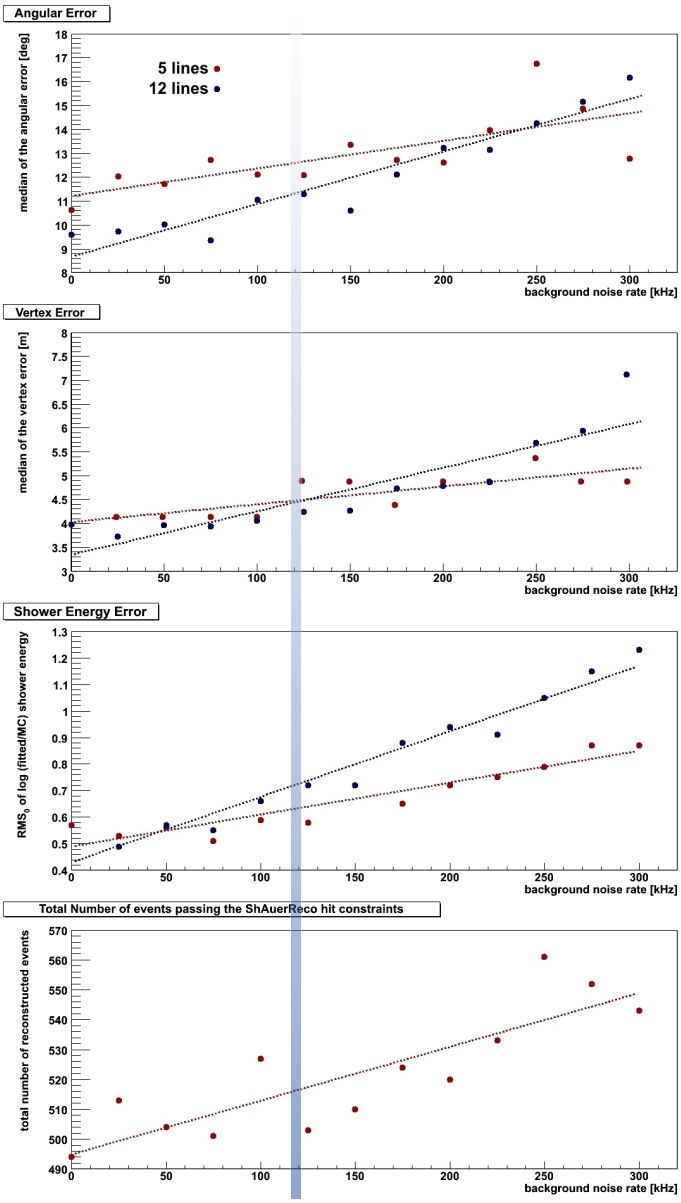

Figure 4.10: *Reconstruction quality for various background rates.* The blue values represent a 12 line reconstruction, the red ones are for 5 lines. The dotted lines are linear chi-square fits to the entries. The blue bar signals the finally chosen value for all upcoming fits.

lution for all three observables (direction, vertex and shower energy) decrease slightly. For even higher background rates beyond the plotted range (i.e. up to 1 MHz) the resolutions degrade even more rapidly. Despite the fact that the detector is turned off at rates above 500 kHz to avoid aging effects of the PMs, these studies have been carried out to even higher rates ut to 1 MHz. For low background rates the accidentally occurring background hits are mostly not yet big enough in number to create coincidence hits that exceed the threshold of 3 pe. All damage a noise hit can do at this

level is to join a real physics hit and thus exceed the threshold. By doing so, the noise hit can modify the true hit amplitude or destroy the timing information, if it hits the OM within the time integration window *before* the physics hit. In both cases a wrong information will be sent into the reconstruction, which of course decreases its quality more and more, when due to an increasing noise rate more noise hits are available to make trouble. At higher rates, however, coincidences of two or more noise hits become more frequent and the noise hits can not just join with a true hit, but also create a pure noise hit that passes the hit constraints, if at least three noise hits coincide. According to Figure 4.11 every fifth event has such a pure noise hit at a rate of 300 kHz. At this rate a noise hit will appear on average at every third OM, assuming an event duration of about 1000 ns.

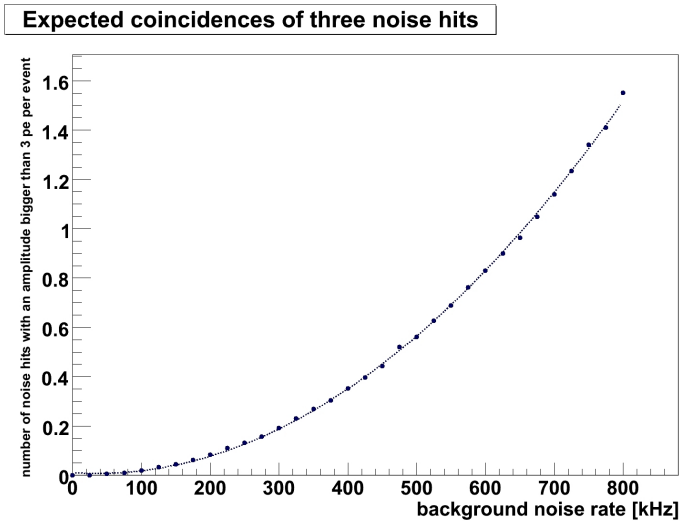

Figure 4.11: *Expected coincidences of three or more noise hits within a* 40 ns *integration time window of an OM.* Plotted is the expected number of such coincidences in the 12 line detector per event. To gain these numbers white noise has been added to 10000 *empty* events (containing no hits), assuming an event length of 1000 ns. Including the offset time for the noise before and after the actual event a total time period of 2000 ns has been filled with noise and processed with the *ANTARES* DAQ simulations.

The fluctuations in the number of successfully reconstructed events, i.e. those that passed the hit constraints, are of statistical nature. The hit constraints for each event can be regarded as a Bernoulli experiment with the two options *pass* or *do not pass* the filter. To simplify matters it is assumed for this estimation that the number would not increase with the background rate. Thus the mean value of the number of expected events is 522. As the probability of how many events pass the filter follows the Poissonian distribution, the variance of the values has to be identical to the mean value. The calculated variance from the values actually is 429, and is therefore within the Poissonian error interval.

Remarkable is furthermore that for the 12 line detector the decline of the quality

is steeper for higher background rates as for the 5 line detector, which can be seen by means of the linear fit lines that have been calculated by a chi-square fit. This discrepancy between the two detectors is due to the bigger number of OMs in the bigger detector, which increases the total number of noise hits and thus the probability of an accidental coincidence of noise hits at one or more OMs.

This chapter has shown that for realistic background rates, for that a practical data taking run can be done at all (let's say up to 150 kHz), the reconstruction quality of the *ShAuerReco* varies somewhat but due to the strong hit constraints, noise is no fundamental problem for shower reconstructions. There is one thing that is to be mentioned further: The resolutions for the specific noise rate of 120 kHz in the plots in Figure 4.10 within this chapter are slightly worse than they have been during the studies of the fit boundaries (Figures 4.3 and 4.8), which also have been reconstructed with a background rate of 120 kHz and thus are expected to be the same. This is no mistake, but is due to two different distributions for the time and amplitude smearing in the *ANTARES* PM simulator module. For the studies of the fit boundaries these PM simulations have been done with the more recent distribution from a tool called *TriggerEfficiency*. When using these distributions, the hit times are smeared by a Gaussian distribution whose width broadens with a higher hit amplitude. For very big hits this sometimes causes a time shift of a few hundred nanoseconds, which is quite bad, if the shifted hit was a triggered one and its shift is to earlier times. By this way sometimes a triggered hit slips before the interaction time and thus becomes the start value for the time fit as *first triggered hit*. To avoid this problem, the PM simulation is done from this chapter on with older distributions that come from the original *ANTARES* Fortran code. A different smearing distribution influences the resulting hitmaps and thus the results of the reconstruction, in this case unfortunately to a slightly lower quality.

4.6 Effects of hardware properties of the optical module

As the *ShAuerReco* algorithm does not calculate a configurable likelihood function from some model, but gets the likelihood values from look-up tables that have been generated from MC events including the whole detector simulation, a successful reconstruction actually requires look-up tables that fit to the currently used hardware in the detector. The properties of the optical modules are included in these tables, and one time they change, new look-up tables have to be generated. Unfortunately it is the case that these properties vary quite a lot. Apart from aging processes that will reduce the performance of the OM, there are two main parameters by which the property can be changed manually: The *integration time window* and the subsequent *dynamic range*

of the ADCs within an OM (see chapters 2.2 and 2.5 for particulars about the OMs). These values have been changed quite often for several periods of data taking with *ANTARES*. So actually, it should be necessary to have a whole set of look-up tables for each possible combination of integration time and dynamic range and choose the right one that fits to the DAQ settings. The studies in this chapter show, however, that the effects are quite small and can be neglected. One can thus take the same tables for all kind of runs.

4.6.1 Integration time of the photomultipliers

The *integration time* of the PM is the time gate in which the ADC collects the data. Two hits arriving at the OM within this time window will be merged to one. Thus the hitmap that will be used for the reconstruction depends on this gate. The hitmap is in the nomenclature of the *IceTray* framework an *I3RecoPulse* map that will be created in the module *I3AntPMSimulator* that simulates the properties of the OM for MC events and of course needs the integration time as one of its parameters. The look-up tables of the *ShAuerReco* have been generated with an integration time of 25 ns. Figure 4.12 shows the reconstruction quality for hitmaps that have been generated with different integration times in the PM simulator and reconstructed with same (25 ns) look-up tables.

What can be concluded from the plots is that the reconstruction quality indeed varies slightly when the PM integration time is changed, but concerning the reconstruction of the shower energy (and this is what one is mostly interested in) the effect is so small, that it is not essential to generate new look-up tables for different integration times.

As the existing look-up tables are made for an integration time of 25 ns, one could expect there a minimum in the resolution of the three observables. What is however seen in the plots is that the quality has no minimum but is improved for shorter integration times, which appears to be a comprehensible behavior, as for shorter integration times more single hits will remain single hits, whereas for longer integration times more and more single hits will be merged together. This is also the reason, why for longer integration times the total number of events passing the *ShAuerReco* hit constraints increases, as with a greater number of merged hits also the probability for more (merged) hits with an amplitude bigger 3 pe is increased. The following reconstructions will be done with an integration time of 40 ns as this was the set value in the 5 line data taking period in 2007 [48].

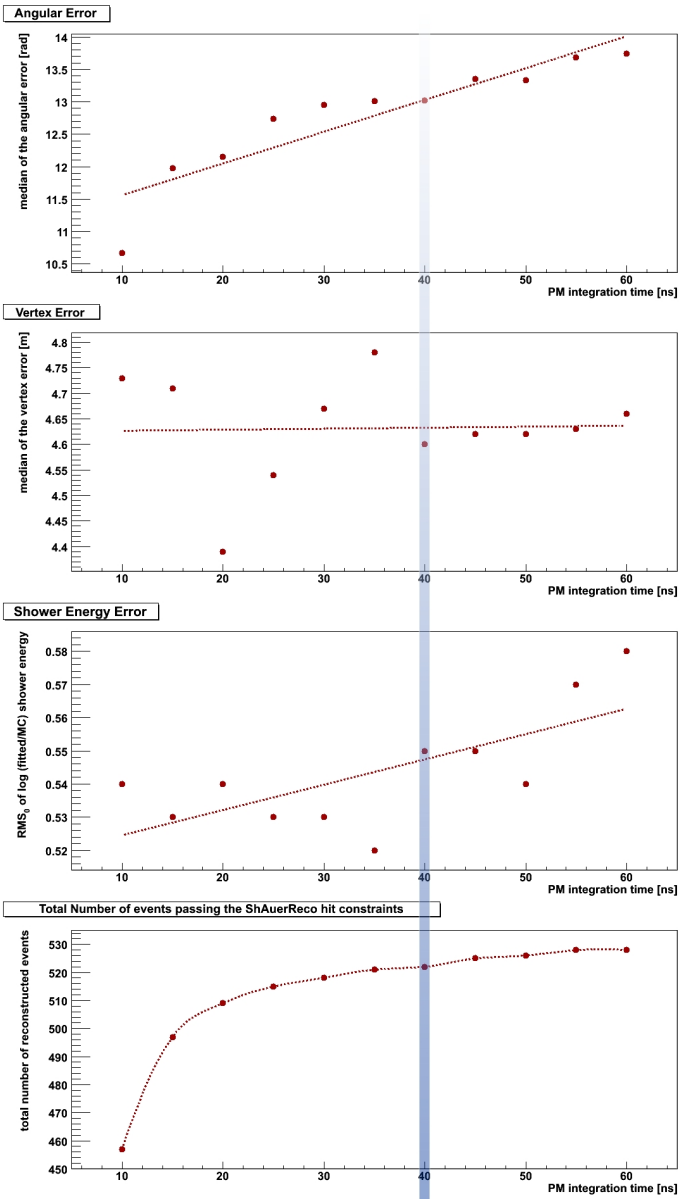

Figure 4.12: *Reconstruction quality for 5 line events using different values for the PM integration time.* The dotted lines in the first three plots are linear chi-square fits to the entries. The blue bar signals the finally chosen value for all upcoming fits.

4.6.2 Dynamic range of the analog to digital converters

Compared to muon track events, where the energy of the muon is deposited along the track, a lot of energy is deposited in a very small volume in shower events. Thus the Čerenkov photons can cause single hits with an amplitude of several hundred photo electrons already at low energies. For the highest energetic events in *sample 2* there are some hits that exceed 10000 pe. Of course such hits cannot be measured in real data,

as the PMs are limited by the dynamic range of the ADCs. So in order to compare MC to real data, also for MC events all hits with an amplitude bigger than the saturation value of the PM have to be modified. There are two possibilities to handle these hits: Either clip higher amplitudes to the maximum saturation limit and keep the hits in the hitmap, or remove those hits completely from the hitmap and don not consider them in the reconstruction.

The look-up table for the amplitude likelihood has been generated without any cut or clip. All MC events contribute with their full amplitude information to the probability distributions. Hence to use these tables for real data is in fact disputable as they base on hits that cannot be measured at all and will provide false information for events with high amplitude hits. Actually the dynamic range had to be taken into account when creating these tables. Unfortunately the saturation limit for the PM is not a constant, but can be varied. Furthermore the dynamic range can be different for each OM! Figure 4.13 shows the dependency of the reconstruction resolution for different saturation limits. Therefore *sample 2* has been processed in two different ways. First with a clip of all amplitudes to the various saturation limits, and afterwards with all hits removed that have an amplitude above the saturation limit.

What can be seen in the plots is that for a lower dynamic range, of course, the reconstruction quality decreases, as more and more hits with an high amplitude are either clipped and made to low amplitude hits (red line), or even are removed (green line). For the vertex error this effect, however, cannot be measured at all as the fluctuations are too big and thus the resolution is assumed to be constant within the plotted range. At the typical value for the PM saturation of 20 pe (that also will be used in further MC reconstructions) one can see that the resolution is always better when the exceeding hits are removed instead of clipped to a lower amplitude. This can be understood quite well, as the *ShAuerReco* only considers OMs with hits and does not take care about OMs with no hits. So if a high amplitude hit will be clipped it still remains in the hitmap and contributes a false amplitude information to the reconstruction. When completely removed it can cause no damage, as the corresponding OM will simply not be taken into account furthermore. But removing big hits means also to decrease the probability of an event passing the *ShAuerReco* hit constraints that require at least five hits with an amplitude bigger than 3 pe. Hence to avoid a further reduction of the efficiency for MC studies the clip method is chosen from now on, as in this case no hits will be removed. Besides in real data too big hits will also be clipped and not removed, as the OM simply reaches its saturation and does not know that the hit actually is bigger.

This chapter has shown that the loss of reconstruction quality when using look-up tables that do not fit to the actual OM properties is quite marginal for the shower

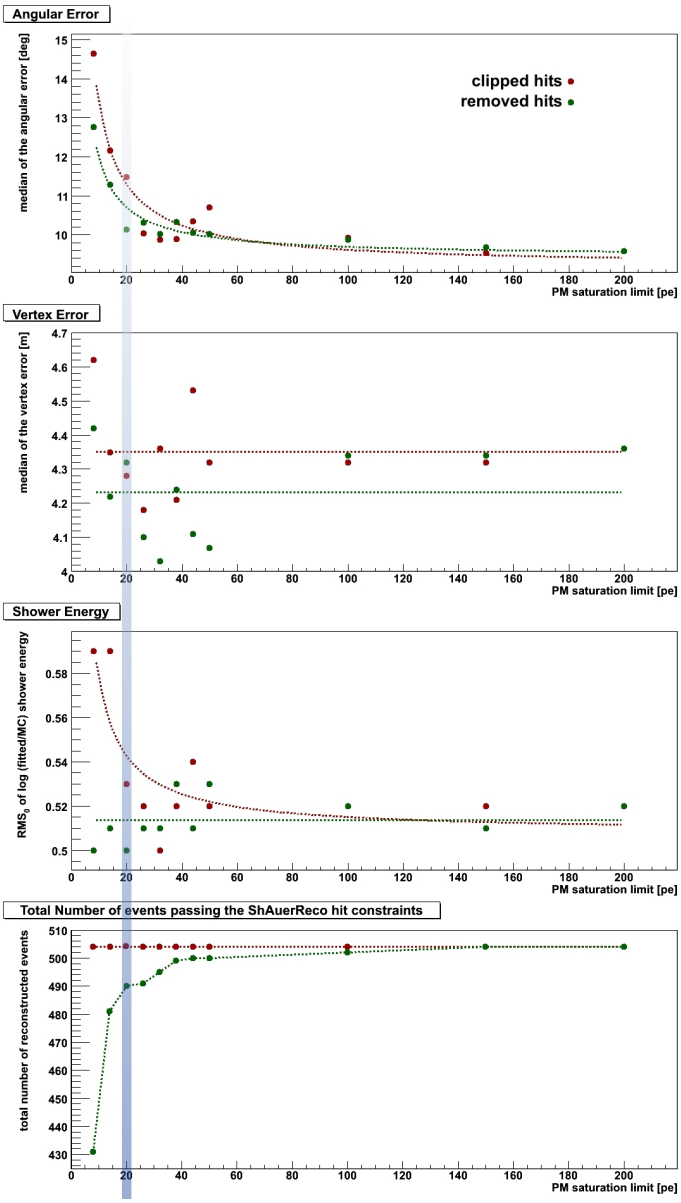

Figure 4.13: *Reconstruction quality for 5 line events using different values for the PM saturation limits (i.e. the dynamic range).* The red values represent the reconstructions with clipped hits, the green values those, for which the hits exceeding the saturation limit are completely removed from the hitmap. To guide the eye chi-square fits of linear or $1/x$ functions have been calculated, that are printed as the dotted lines. The blue bar signals the finally chosen value for all upcoming fits.

energy. Thus the existing tables can also be used for reconstructing live data as long as the integration time does not exceed 60 ns and the saturation limit does not become lower than 10 pe. In this case one will lose about 3° in angular resolution, whereas the shower energy reconstruction is quite insensitive to those variations.

4.7 Other hit constraints to improve the efficiency

As described in chapter 3.3 the strong hit constraints that are necessary to make the *ShAuerReco* work properly decrease the efficiency enormously and filter out a lot of events. From MC weights it can be derived an expected total number of triggered shower events of about 580 per year for the 5 line detector, of which only about 40 remain after the hit constraints filter. This corresponds to an efficiency of about 7%. For the 12 line detector some 170 of about 1540 triggered events will be reconstructed, which results in an efficiency of 11%. The studies in chapter 5 show that various quality and purity cuts to filter out atmospheric muons and neutrino tracks reduce both efficiencies to about 1%.

This chapter introduces some not too sophisticated attempts to increase the efficiency by modifying the hit constraints. Unfortunately it is shown that with simple variations of the constraints and the hit selections the efficiency cannot be improved without decreasing the reconstruction quality to an unacceptable level. All reconstructions in this chapter are done with 5 line data of *sample 2* with 120 kHz background and the optimum fit start values and search space boundaries derived in the chapters above.

4.7.1 The low amplitude cut

The *ShAuerReco* hit constraints consist of two parts. First a hit selection is done that consists in the removal of all hits with an amplitude below 3 pe, and afterwards it is checked on how many strings and OMs the remaining hits give a signal (see also chapter 3.3). At first only the first condition has been relaxed to increase the efficiency while the second kept unchanged. One could imagine to remove only hits with an amplitude below 2 pe and thus have a bigger list of hits that can fulfill the second condition. Figure 4.14 shows the reconstruction qualities for different low amplitude cuts. This study has been done just with 5 line data, as for 12 lines the effects are assumed to be the same.

What can be seen in the plots is that below a cut value of 2 pe the reconstruction will fail completely. The improvement of the reconstructio quality for a cut value of 3 pe instead of 2 pe is about a factor of 2 whereas the efficiency is decreased by a factor of 3. The efficiency is the fraction of weighted events after the hit constraints filter per year over all triggered events per year. From 3 pe up to higher cut values no further improvement can be achieved, as the greatest part of noise hits is already removed at a cut value of 3 pe. Towards lower cut values the resolution and efficiency suddenly increase and reach the maximum for about 1 pe, which is obvious, as the simulated noise

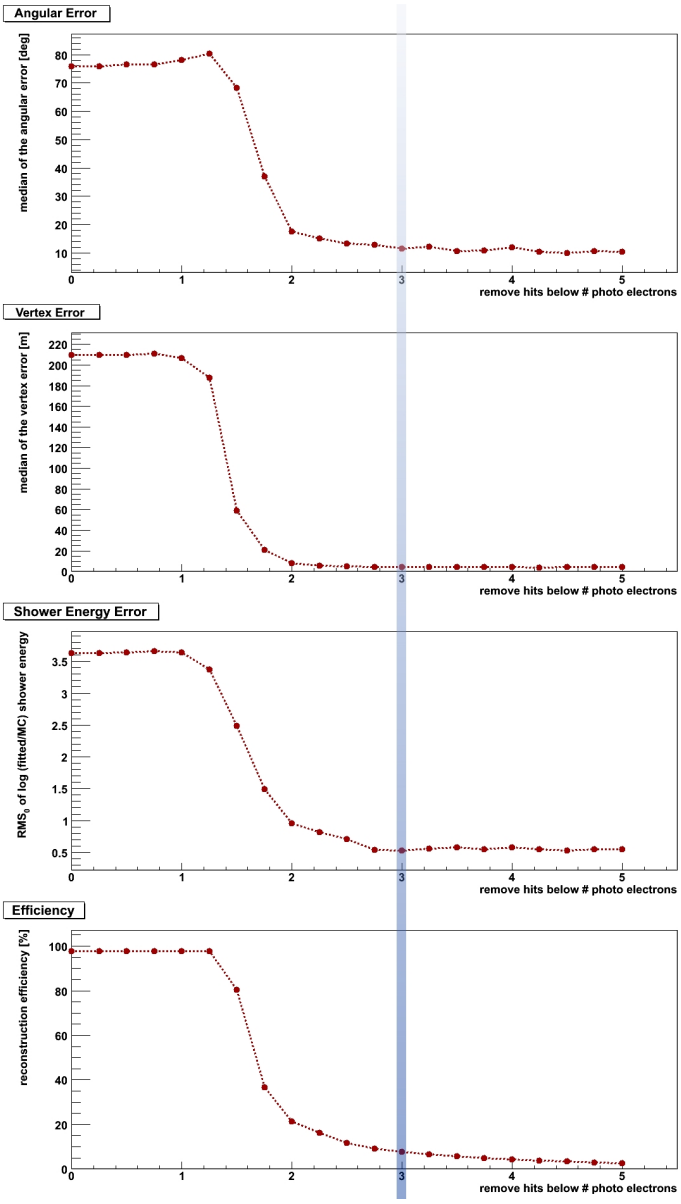

Figure 4.14: *Reconstruction quality for 5 line events using different low amplitude cuts as hit selections.* In the reconstructions all hits are taken into account that have an amplitude bigger than the cut value and pass the original hit constraints from chapter 3.3. The blue bar signals the finally chosen value for all upcoming fits.

hits have an amplitude of 1 pe and thus almost every hit will be taken into account now. The only events that will be filtered out here are those that cause less than 5 hits. Of course, due to the amount of noise hits, the reconstruction fails for almost every event which explains the extremely bad quality for lower cut values.

When reconstructing showers one is mostly interested in their energy and thus it is

essential to have a good reconstruction quality for the shower energy. Of course, a cut value of 2 pe increases the efficiency to 21%, but a corresponding shower energy error of one magnitude doesn't sound so promising. To achieve the best possible reconstruction quality, that is about 0.55 orders of magnitudes (corresponds to a factor of 3.5), the cut value has to be set to at least 2.75 that corresponds to an efficiency of 9%. However, this work has been pursued with a cut on 3 pe.

4.7.2 Hitmaps from the *ANTARES* trigger

Obviously the main problem for the *ShAuerReco* algorithm is the noise. The use of a low amplitude cut on the hitmap removes most of the noise but strongly decreases the efficiency, as mentioned above. Another idea to get rid of the noise hits is to use the hitmap of *triggered hits* instead of a low amplitude cut on the whole hit map. The hitmap of *L1 hits* contains all hits above a preset amplitude threshold that has been set to 3 pe until May 25th 2007 and from then was 10 pe [49], and hits that coincide on one storey within a time window of 20 ns. (NOTE: The trigger threshold of 3 pe is no amplitude cut as done in the *ShAuerReco* low amplitude cut! A hit with an amplitude exceeding this threshold is automatically triggered without looking for coincidences. For more details on the trigger, see chapter 2.5.) To use the *L1 hits* for the constraint filter (that is mainly at least 5 hits on 3 strings) offers a bigger choice of hits and thus increases the efficiency up to 35%. Unfortunately, too many accidental coinciding noise hits pass this first trigger level, which finally results in 5 line reconstruction errors of 30.59° median for the direction, 15.6 m median for the vertex and 1.3 magnitudes RMS_0 for the shower energy. The tables in chapter 4.8 will show that these resolutions are far below those that can be reached with the original *ShAuerReco* hit constraints.

Also the second trigger level that filters out *L1 hits* that cannot be causally connected gives no improvement, as equation (2.4) in principle also allows some noise hits that have passed the first stage to pass the second. The total number of hits in the map of *L2 hits*, also called *triggered hits*, is a little bit smaller, thus the efficiency only reaches 32%, but as probably some of the L1 noise hits have been filtered out, the quality is improved to 25.9° median for the direction, 10.5 m median for the vertex and 1.18 magnitudes RMS_0 for the shower energy. However both hitmaps from the trigger still contain too much noise to be really an option for the reconstruction hitmap.

4.7.3 The constraints filter

After the low amplitude cut the remaining hits in the event have to pass the hit constraints filter that finally decides if an event has the chance to be successfully recon-

structed by simply looking at the geometrical hit distribution in the detector. By default this filter selects only events that cause at least 5 hits on 5 different OMs that are located on at least three different strings. As there are only few hits with an amplitude bigger 3 pe that fulfill these requirements this filter is the main reason for the low efficiency (see also Figure 4.9). In the chapters above it was shown that it is not possible to relax the low amplitude cut or to use other hitmaps as input for this filter. So the next step is to try to relax the constraints of the filter and see, if the reconstruction again falls off in quality or not.

Figure 4.15 shows the reconstruction quality and efficiency for different minimum constraints. According to these plots the only way to increase the efficiency is to use just one or two strings that have to be at least hit instead of three. But in this case the vertex and direction reconstruction will lead to disastrous results, as for all events that hit just 2 strings the hit distribution is a plane, which offers much more (false) possibilities to fit a spherical shape than with three dimensional hit distributions. For a plane distribution there are always two equiprobable possibilities for the vertex position that are symmetrically mirrored respective to the plane. Statistically seen for half of the events, the reconstruction will choose the wrong side, which of course also results in failures for the direction reconstruction. For events that hit only one string this effect even worsens, as in this case there is a whole circle of possible vertices around the string.

The only observable that is less affected by the ambiguity of the vertex is the shower energy. In principle, if one is just interested in the energy and does not care about vertex and direction, the constraints can be relaxed to at least two hits on one string, whereas a loss of just 0.2 magnitudes in the shower energy error has to be taken. Therefore the efficiency will be increased to tremendous 91%! But there is also a disadvantage. The reconstruction of the direction will completely fail and one will have no possibility to separate upgoing from downgoing events, which is at this moment the most effective criterion to suppress atmospheric muon background that only cause downgoing events. So one has to know in advance that the event *is* a shower event. Unfortunately at present there is no possibility to do this, and hence to increase the efficiency for the energy reconstruction this way is currently no option. But finding other hit selections that are independent from the reconstructed direction in order to be able to reconstruct the energy of neutrinos from the diffuse cosmic neutrino flux with high efficiencies is of course worth being considered further.

But as within this thesis the upgoing cut is mandatory, the hit constraints filter will be used as given with a 3 pe cut and a hitmap that includes at least 5 hits on different OMs and on 3 different strings.

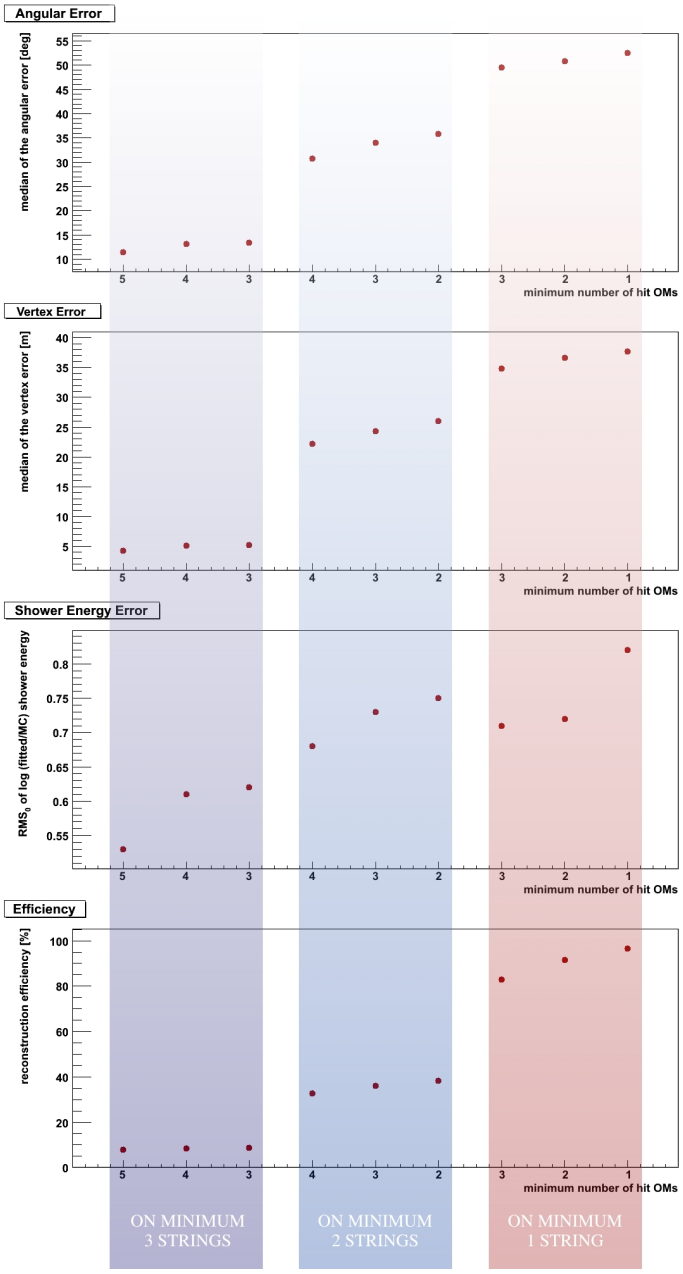

Figure 4.15: *Reconstruction quality for 5 line events using different settings in the hit constraints filter* Plotted are the quality and efficiency of the reconstruction after an event filter that lets pass only events with hits on at least a certain number of OMs (x-axis in the plots) on a minimum number of strings (different colored areas).

4.8 Resolutions and efficiencies without any quality cut

To conclude this section, in which the *ShAuerReco* algorithm has been prepared for real data reconstruction, the gained resolutions and efficiencies for 5 line and 12 line

reconstructions are summarized. In the following table the listed values are derived from weighted MC events, i.e. they represent the real expected energy spectrum. The efficiency is calculated by dividing the number of reconstructed shower events by the number of all triggered shower events. As can be seen all throughout the studies above, the resolutions are subject to statistical fluctuations. A small variation of the background rate, for example, can change the error completely, as can be clearly seen in Figure 4.10. From that figure thus an error has been estimated that is attached to the values, just to show that the derived values are not to be understood as 100 percent constant. The errors for the total numbers are the Poissonian errors, and those for the efficiency are calculated via Gaussian error propagation. The final resolutions and efficiencies have been derived from the reconstruction of *sample 2* and *sample 3* with the optimum reconstruction parameters obtained in the chapters above. The given values are those of the pure reconstruction without any kind of quality cut.

event type	**angular error median**	**vertex error median**	**log shower energy error RMS_0**	**reconst. events per year**	**efficiency**
5 line MC	$(13 \pm 1)^\circ$	(4.9 ± 0.5) m	(0.6 ± 0.1)	(41 ± 6)	$(8.0 \pm 1.5)\%$
12 line MC	$(10 \pm 1)^\circ$	(4.0 ± 0.5) m	(0.6 ± 0.1)	(161 ± 13)	$(12.0 \pm 1.3)\%$
5 line MC Genova	$(10 \pm 1)^\circ$	(3.7 ± 0.5) m	(0.6 ± 0.1)	(47 ± 7)	$(6.6 \pm 1.3)\%$
12 line MC Genova	$(10 \pm 1)^\circ$	(3.6 ± 0.5) m	(0.6 ± 0.1)	(185 ± 14)	$(9.6 \pm 1.1)\%$

Concerning the quality in shower energy reconstruction there is no great difference between 5 line and 12 line reconstruction due to the extension of a shower event that is just a few metres. For MC events that have been generated with the new *Genova* angular acceptance for the OMs of course more events will be detected as the *Genova* acceptance includes higher photon fluxes for larger incidence angles. This increases the number of triggered events as well as those passing the hit constraints. However, the efficiency for *Genova* MCs seems to be slightly lower than for the old acceptance, which looks strange but can either be a statistical phenomenon or just as well due to different implementations in the PM simulator code for the two acceptances. Besides they are comperable within the error ranges.

The comparison of these resolutions to those that are predicted in [39] shows a discrepancy, especially in the angular error. Whereas in that paper the median error without quality cuts for the vertex position is 3.8 m and the RMS for the energy error is 0.52, which is consistent with the values obtained here, the angular median error of 6.5° differs from the resolutions above. This can be due to various reasons. The reconstruction in that paper has been done without the clip at the PM saturation limit of 20 pe which probably explains the slightly better energy reconstruction. Furthermore

the events have been treated as unweighted MC events, i.e. the start value for the time fit has always been 0, which avoids associated problems when finding a suitable startvalue (see chapter 4.3). However, this is inevitable for real data analysis. When reconstructing this sample as weighted 12 line data with the same settings as for the samples used within this thesis, one obtains an angular error of 11°, which fits to the results above.

With all the obtained optimum parameters from this chapter a real data taking run has been simulated by trying to reconstruct also atmospheric muon events and muon neutrino tracks with the *ShAuerReco* algorithm. A lot of different cut values will be necessary to identify the shower events within the huge amount of atmospheric muons and to improve the resolutions by cutting away also bad reconstructed events. First attempts to do this are presented in the following chapter.

Chapter 5

Event classification

In the previous chapter the *ShAuerReco* algorithm has been prepared for the reconstruction of real data taken with the *ANTARES* detector. The optimum parameters to run the module have been determined and it was shown that slightly varying hardware specific properties of the OMs do not affect the reconstruction quality too much. Even background noise rates up to about 150 kHz decrease the quality just slightly, which of course is due to the strong hit constraints. These constraints, however, also severely reduce the efficiency. But in principle the algorithm works well for shower events.

Unfortunately when analyzing real data it is difficult to determine the class of an event (neutrino induced muon track, atmospheric muon or hadronic shower), unless an efficient event classification algorithm is available, which is currently developed by Friederike Schöck [50] and will hopefully soon be able to classify event signatures *before* any reconstruction is done. However another promising way to classify events, although perhaps less elegant, is to reverse the processing order: First send all data to the specific reconstruction and try to filter out unwanted events by various cuts on parameters that are determined by the reconstruction algorithm, i.e. for example likelihood values.

Although the *ShAuerReco* reconstruction takes quite some time to reconstruct shower events (up to a few seconds for high energetic events), the total computing time will not be escalating, as due to the hit constraints just every 20th event (in a MC event file) will be reconstructed anyway.

In the following several variables on which one can cut in order to identify the event class or to improve the reconstruction quality are introduced and explained. Towards the end of this chapter these values will be used to try to filter out muon tracks, which unfortunately does not work without cutting also shower events away and thus decrease the efficiency further. Even when combining all possible cut values, it is still quite difficult to cut away all unwanted events and simultaneously leave an acceptable amount of showers in the sample. Every cut that filters out all atmospheric muons will also filter out almost all shower events.

5.1 The MC files for a real data MC sample

At first it is required to have suitable MC files that represent the neutrino induced muon track events and atmospheric muon background. The first studies to find cut values have been done with 5 line data, wherefore the *samples 1 - 3* that are described more in detail in chapter 4.1 have been used for hadronic shower events. The muon events are represented by the new 5 line *Mupage* [51] production for atmospheric muons and an older 5 line *GENHEN* [43] and *KM3* [52] simulation, which will be referred to as *prod0401*, for neutrino induced muons.

The hadronic showers within this realistic real data simulation are represented by the *samples 1 & 2*, which of together contain 150k shower events and another 100k events from *sample 3* that has been generated using the *Genova* angular acceptance for the OMs. Combining these three event files and regarding them as one big shower sample creates a mixture of *Genova* and *"old"* shower events that ensures that both types of simulations are taken into account when looking for suitable cuts. To improve statistics the reconstruction qualities and efficiencies after the cuts will also be derived from this mixture.

The *Mupage* production is a MC simulation of atmospheric muons based on parametrized distributions that have been adapted to real measurements. The simulation produces atmospheric single muons and muon bundles, whose flux and multiplicity depends on the water depths and thus has been adjusted to the *ANTARES* site and detector. To read more details on the simulation, see [51]. As the rate of triggered atmospheric muon events is considerably higher than for hadronic showers, a huge number of events is required when comparing these with the shower events. Hence 100 files of the *Mupage* production have been processed with the *ShAuerReco* algorithm, namely the first 100 files called *km3.run_#### .evt* with *####* being an even number. Each file corresponds to a livetime of $8.06 * 10^3$ s, wherefrom the number of expected events per year can be derived. The following table gives a short overview about these files.

Mupage

Generated cosmic rays spectrum	after the *poly-gonato* model [53]
Primary energy range	1 TeV to 10^6 TeV (depending on the particle)
Primary particle types	p, He, CNO, Mg, Fe
OM angular acceptance	Genova acceptance [45]
Detector simulation	l05_c09_s02.det
Muon direction	0° to 85° zenith[1] angle
expected triggered 5 line events per year	$1.38 * 10^8$

[1] a zenith angle of 0° represents a vertical downgoing particle

expected 5 line events per year passing the *ShAuerReco* hit constraints	$9.44 * 10^6$

The MC simulation *prod0401* that contains neutrino induced muon tracks from atmospheric neutrinos has been generated in two different neutrino energy ranges for antineutrinos and neutrinos. As the high energetic simulation (HE) from 10^6 GeV to 10^8 GeV contributes only a very small part to the number of expected events per year (see table below) they will not be considered further, as their fluxes are already very low. Of course the lower energetic part (LE) from 10 GeV to 10^7 GeV has to be taken into account. These studies have been restricted to the event files containing neutrinos (not antineutrinos) as primary particles. There are two simulations, one for the 5 line and one for the 12 line detector. For both detectors all available files have been processed. The total number of expected events per year can be derived from the *ANTARES* MC event weights [42].

Note: As the neutrino induced muon MC events are simulated for upgoing events only, the actual flux is twice the flux in these files. However, due to the orientation of the OMs the detector is more sensitive to the lower hemisphere, so that the given events per year in the tables are not to be multiplied with a factor of 2 but somewhat between 1 and 2 (in comparison with the shower MC files that have been generated isotropically and from which the MC zenith angle distribution for detected events is shown in the plots in chapter 5.2.5).

LE Prod0401

Generated neutrino spectrum	$E^{-1,4}$
Neutrino energy range	10 GeV to 10^7 GeV
OM angular acceptance	old measured acceptance [30]
Detector simulation	l05_c09_s00.det (5 lines) / d10_c00_s00.det (12 lines)
Neutrino direction	lower hemisphere
# of generated events	$95 * 10^{10}$ (for each detector)
expected triggered 5 line events per year	1939
expected 5 line events per year passing the *ShAuerReco* hit constraints	89
expected triggered 12 line events per year	5868
expected 12 line events per year passing the *ShAuerReco* hit constraints	615

HE Prod0401

Generated neutrino spectrum	$E^{-1,4}$
Neutrino energy range	10^6 GeV to 10^8 GeV
OM angular acceptance	old measured acceptance [30]
Detector simulation	l05_c09_s00.det (5 lines) / d10_c00_s00.det (12 lines)
Neutrino direction	lower hemisphere
# of generated events	$9 * 10^9$ (for each detector)
expected triggered 5 line events per year	0.05
expected 5 line events per year passing the *ShAuerReco* hit constraints	0.03
expected triggered 12 line events per year	0.09
expected 12 line events per year passing the *ShAuerReco* hit constraints	0.06

As the *Mupage* production only was available for a 5 line detector during the work on this thesis, for 12 line reconstruction the older *Corsika* simulation for atmospheric muons has been used. *Corsika* is a full MC simulation of the interactions between cosmic rays and the earth's atmosphere. The MC files have been generated with almost the same simulation parameters as the *Mupage* production. In this case, however, the simulations have been done separately for each type of primary particle, whereof the studies within this thesis have been restricted to only protons as primaries. To check the consistency with the *Mupage* production, this originally 12 line simulation has also been processed with a 5 line detector by cutting away all hits appearing on strings 6 to 12. More information can be found in [54].

Corsika

Generated cosmic rays spectrum	$E^{-2,0}$
Primary energy range	1 TeV to 10^5 TeV
OM angular acceptance	old measured acceptance [30]
Detector simulation	d10_c00_s00.det
Muon direction	0° to 85° zenith angle
# of triggered 5 line events per year	$9.49 * 10^6$
# of 5 line events per year passing the *ShAuerReco* hit constraints	$5.69 * 10^5$
# of triggered 12 line events per year	$3.16 * 10^7$
# of 12 line events per year passing the *ShAuerReco* hit constraints	$4.45 * 10^6$

The comparison of the expected events per year between the 5 line *Mupage* and *corsika* simulations shows that after the *Mupage* production the number of triggered

hits is expected to be more than one order of magnitude bigger. There are two reasons that play an important role for that. The first is that from the *Corsika* simulation only the files with protons as primaries has been used, whereas the *Mupage* production includes the whole cosmic rays abundances. The second is the different acceptances for the OM that have been used. *Mupage* uses the extended angular acceptance, called *Genova*. It has already been shown for shower MC simulations (in chapter 4.1) that the use of the *Genova* acceptance increases the flux by a factor of about 1.4. 85% of the primary cosmic rays are protons, i.e. the difference in the flux between a pure proton and an all particle simulation is a factor 1.2. These two effects taken together could increase the flux by 1.7, which is far below the given difference of a whole magnitude! At this point, however, one has to have a more profund knowledge on these simulations in order to give further explanations. As this, however, exceeds the scope of this thesis, it only should be advised to the problem, which in this case is only a minor problem, as the aim is to primarily filter them all out. Fortunately the difference in the flux is the only great discrepancy. Concerning the now following cut parameters, both simulations behave almost equally.

5.2 Potential cut parameters

This section introduces various potential cut parameters to separate hadronic showers from muon track events after a reconstruction with the *ShAuerReco*. The following studies have been done on 5 line data, using *sample 1-3* for hadronic showers, the *Mupage* sample for atmospheric muons and the *LE prod0401* sample for 5 line neutrino induced muon tracks. A background noise of 120 kHz has been added to all samples before reconstructing them with the *ShAuerReco* algorithm.

5.2.1 *ShAuerReco* likelihood values

After a successful reconstruction the *ShAuerReco* module writes out a data storage class, called *I3ShAuerRecoParams* that contains by now six values. Four of them are finally obtained minimum log-likelihood values (further described in chapter 3.1 and later within this chapter) that are a measure for the quality of the reconstruction. A lower final log-likelihood value means that the algorithm succeeded in finding a suitable shower signature more effectively than for higher log-likelihood values. Originally, these values were meant to be used for quality cuts to remove bad reconstructed showers and thus increase the resolutions. But they can also be used as a criterion for the event signature. Therefore one can assume that the hit distribution of a muon track does not look like one produced by a shower event and hence the event will hopefully

be reconstructed quite badly and produce high log-likelihood values. (To simplify the denotation, the prefix "log" will henceforce be left out. Whenever a likelihood is mentioned now, the log-likelihood is meant.)

Figure 5.1 shows an illustration of at which steps in the reconstruction procedure which kind of likelihood value will be written out. A detailed description of the internal workflow of the *ShAuerReco* is described in chapter 3. When reconstructing with *FitType 2*, a prefit is done using only the timing information from the look-up tables. The outcoming minimum likelihood value will be referred to as *pre likelihood*. The so (pre-)fitted values for x, y, z and t will be used as start values for the final fit that now takes both, timing and amplitude information, into account. The obtained minimum likelihood value from the timing look-up table will be referred to as *time likelihood*[2]. The look-up table containing the *time likelihood* is of course the same as that for the *pre likelihood*. Even so the derived minima will differ, as in this case the fit is a combined fit, where the amplitude information affects the minimization of the *time likelihood*, whereas in the prefit the *time likelihood* is minimized alone. The minimum from the amplitude look-up table in the final fit is called *amplitude likelihood*[3]. According to equation (3.3) the global likelihood, here named *final likelihood*[4] is the weighted summation of $W_T *$ *time likelihood* and $W_A *$ *amplitude likelihood*.

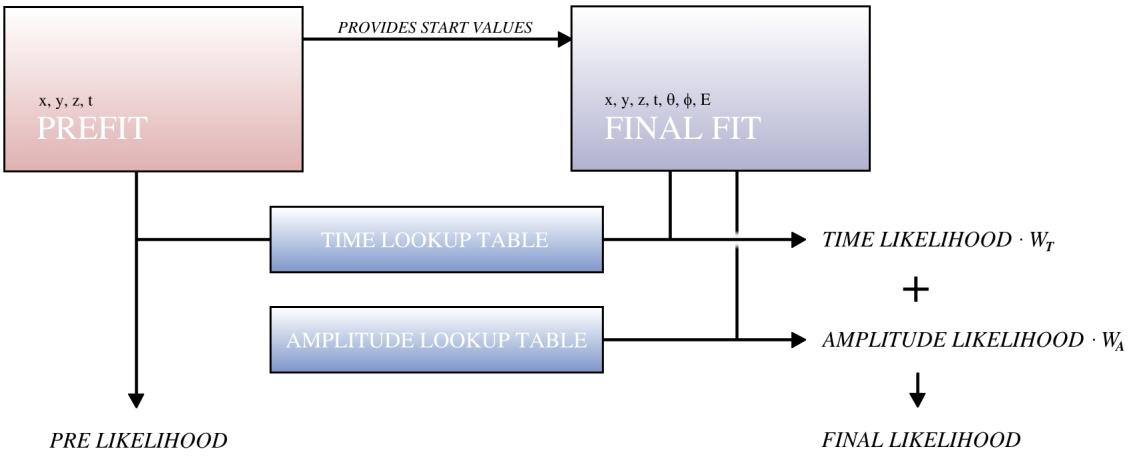

Figure 5.1: *Various likelihood values in the ShAuerReco reconstruction.*

The resulting likelihood values for the three major event signatures after reconstructing them as showers are shown in Figure 5.2. As expected, muon track signatures will

[2]Within the *I3ShAuerRecoParams* container this value is called *time residuum*. To avoid confusion the value has been renamed within this thesis, as in many other *ANTARES* reconstruction algorithms the denotation *time residuum* terms the difference between expected and calculated hit arrival times at the OM.

[3]Within the *I3ShAuerRecoParams* container this value is called *amplitude residuum*.

[4]Within the *I3ShAuerRecoParams* container this value is called *log likelihood*.

be reconstructed less successfully than shower events, which results in high likelihood values in all four plots. However there are nevertheless some muon events reaching the same quality in their likelihood value as the shower events, which makes it difficult to filter them all out. Note that in these plots each distribution has been adjusted in a way that its maximum is set to 1. Otherwise one would not recognize any shower or neutrino induced muon event, as the total number of atmospheric muons exceed their number by several magnitudes. The plots have been generated in order to identify an optimum cut value. The total number of events and the hence resulting efficiencies will be discussed later in chapter 5.3.2. So when trying to identify shower events, one can concentrate predominantly on removing the atmospheric muon events, as they represent the real background problem, not the neutrino induced muon tracks. The yellow line in the plots represents an optimum cut value that will remove as much muon events as possible and keep a many shower events as possible.

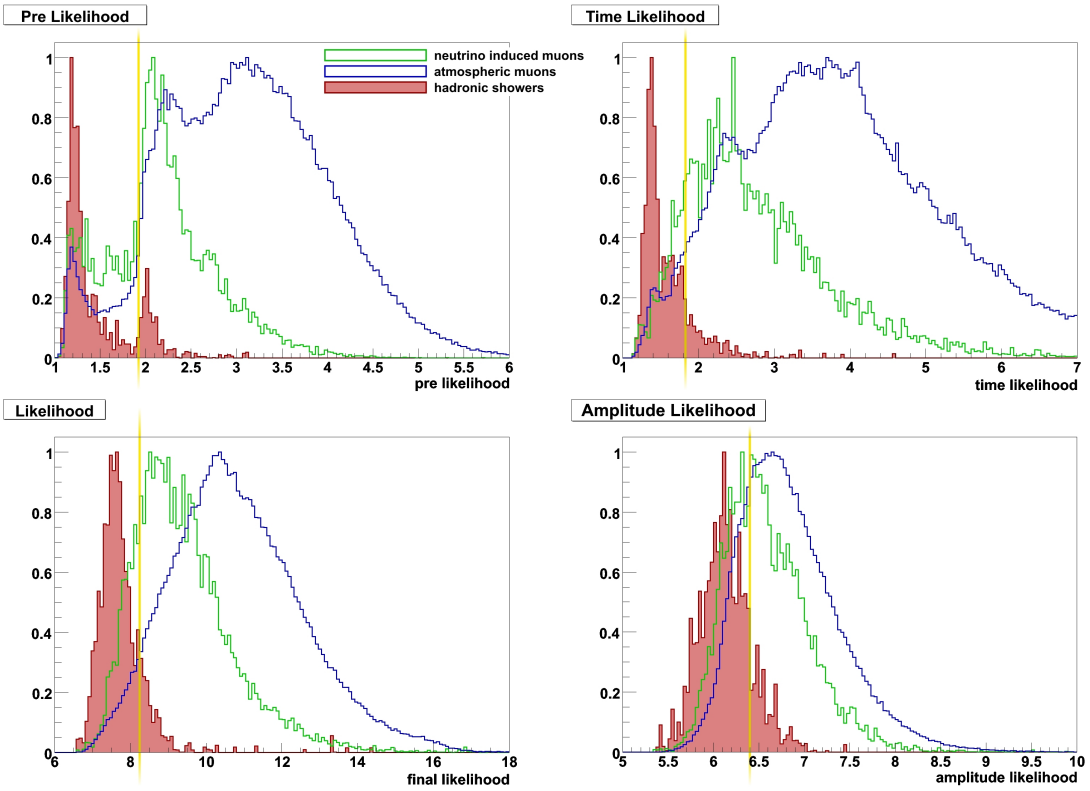

Figure 5.2: *Likelihood values for different weighted event signatures after reconstructing them as showers.* The aim is to remove all neutrino induced muon tracks (green) and atmospheric muon events (blue) and keep as many shower events (red) as possible. The yellow lines mark an optimum compromise for the cuts. For each event type the distribution has been adjusted in a way that its maximum is set to 1.

The problem with cutting on these likelihoods is that for the marked cut values all events with a fitted shower energy higher than 10^5 GeV will be removed from the

sample. Events with such high energies are more difficult to reconstruct as they contain many hits. The probability distribution for an event with many hits has a very complex (seven dimensional) surface with many local minima. To find a suitable signature that fits to many hits is thus more complicated as one for only a few hits. This effect will not decrease the statistics, as the reconstructions are dealing with the whole samples and there are just a few such high energetic events in the spectrum, but should nevertheless be considered. In studies that are focused on high energetic events, this cut cannot be made. On the other hand, as will be shown later, these cuts are strongly required to get rid of at least the greater part of the muon background.

What is remarkable about the *pre likelihood* distributions is a second peak at a value of 2. Most of the (red colored) shower events will already in the prefit be reconstructed quite well with a *pre likelihood* value between 1 to 1.5. But there are some events with a higher value accumulating around 2. What happens here is analogous to the phenomenon that has been described in detail in chapter 4.3.2. By increasing the fitted shower energy also more distant vertex positions are possible realizations for the minimization. This effect occurs on some of the shower events during the prefit, but vanishes in the final fit, as for the final fit also the amplitude information is considered, which makes it more difficult for the minimizer to simply increase the energy. In the prefit in principle every imaginable energy that fits to the timing information is possible.

As on the other hand muon track events do not have a spherical shower signature, this "alternative" solution for the reconstruction is mostly the only possible way to fit the parameters, which explains, why for the greater part of muon events the distribution lies beyond this peak. Due to the completely different time signatures of a muon track event, of course, the track shift is not just 90° but any angle that restores the spherical shape.

5.2.2 Total number of used hits

The remaining two values in the *I3ShAuerRecoParams* class contain information about the total number of hits that have been used for the reconstruction. Due to the used hit constraints there have to be at least five hits with an amplitude bigger than 3 pe. The internal algorithm deals with two different kind of hit types. The hitmap of *SEP hits* contains the full hit information of all reco pulses, i.e. time, amplitude, trigger information, etc., and will be used for the time likelihood minimization. By adding the amplitudes of all hits appearing at the same OM a second hitmap with so called *INT hits* is arranged that will be used for the amplitude likelihood minimization. Of course these *INT hits* have no information about the arrival time anymore. Figure 5.3 shows the distribution of used hits for the different signatures.

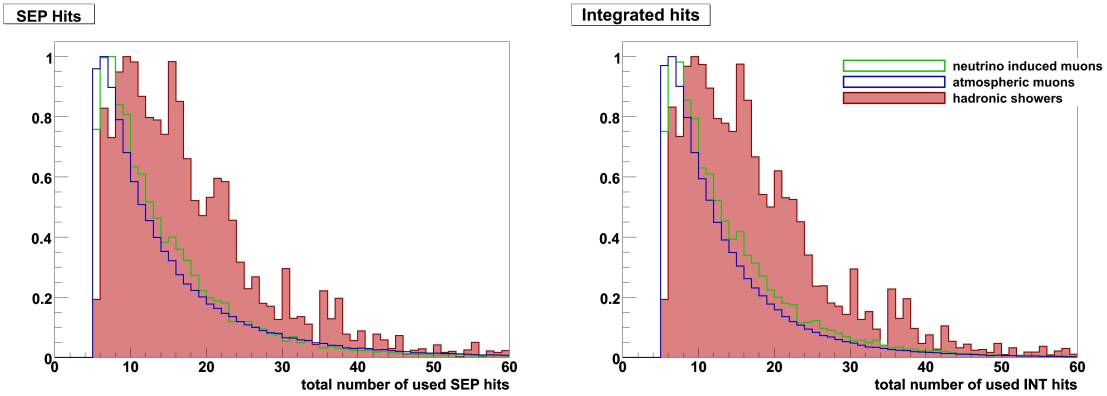

Figure 5.3: *Total number of used hits for different weighted event signatures for the ShAuerReco.* The aim is to remove all neutrino induced muon tracks (green) and atmospheric muon events (blue) and keep as many shower events (red) as possible. For each event type the distribution has been adjusted in a way that its maximum is set to 1.

These plots don not contain the yellow line which used to mark the optimum cut value, as a hard cut on the number of used hits is not recommended and should only be considered in exceptional cases. As the total number of hits in an event is connected with its energy, a cut on the hit number would limit the range in the energy spectrum of potentially reconstructable events. One can see in the plots that showers generally produce more light within the detector volume than a simple muon track. Thus a cut on a minimum of used hits (maybe # of SEP hits > 8) is in principal imaginable but will remove a great part of very low energetic events. Hence such a cut should be considered as a second option and not be done in the first place.

5.2.3 The fitted interaction time

Another possible way to get rid of muon track events is to cut on the fitted interaction time. In the reconstruction the search space for the interaction time has been limited to an interval of $[-300\,\text{ns}, 10\,\text{ns}]$ respective to the first triggered hit. For shower events this is enough to fit a vertex position nearby the center of amplitude. Muon track events however have a hit distribution along a track that can extend across the whole detector. To find a spherical signature that might fit to those hits (this is what the *ShAuerReco* is trying to do) is only possible for a vertex that lies far outside these hits in most of the cases. But as the light propagation from such a distant vertex to the OMs probably takes more than 300 ns, the reconstruction algorithm will mostly run towards its boundaries. Figure 5.4 shows the fitted interaction time for showers and muon tracks respective to the first triggered hit.

One can see the expected behavior for neutrino induced and atmospheric muon

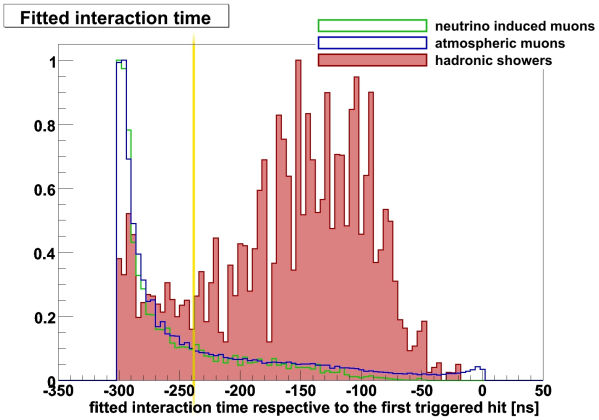

Figure 5.4: *Fitted interaction time for different weighted event signatures after reconstructing them as showers.* The aim is to remove all neutrino induced muon tracks (green) and atmospheric muon events (blue) and keep as many shower events (red) as possible. The yellow lines mark an optimum compromise for the cuts. For each event type the distribution has been adjusted in a way that its maximum is set to 1.

tracks. For the greater part of events the time fit stops at the lower fit boundary, whereas shower events converge properly at times widely spread over the whole interval (of course only up to 0). However, also here it is not possible to remove all atmospheric muons, as there are still some (still several 10.000 times as many as shower events), whose relative interaction time reaches up to 0. Total numbers of events will be discussed later in chapter 5.3.2.

5.2.4 The fitted vertex position

A cut on the fitted vertex position can be regarded as both, a quality and a purity cut. Events, whose interaction vertex is located in the outer regions of the detector or even beyond the detector volume, can obviously not be reconstructed as well as those that are totally contained, as for "edge"-events only parts of the whole shower can be detected. Hence to remove these events from the sample will probably increase the overall reconstruction quality for shower events. According to the resolutions given in chapter 4.8 without any cuts the fitted vertex position misses the true one only by a few meters, so that a restriction of the fitted vertex to the actual detector volume will indeed remove the edge events and increase the quality.

When trying to reconstruct muon track events on the other hand, the *ShAuerReco* algorithm also tries to fit a spherical shape, which results in a fitted vertex position, that is for muons more distant from the center of amplitude than for shower events due to reasons that have already been discussed now several times. Hence a cut on the vertex location respective to the center of amplitude will be able to separate showers

from muon background to some extend. In Figure 5.5 the distributions of these two position criteria are shown.

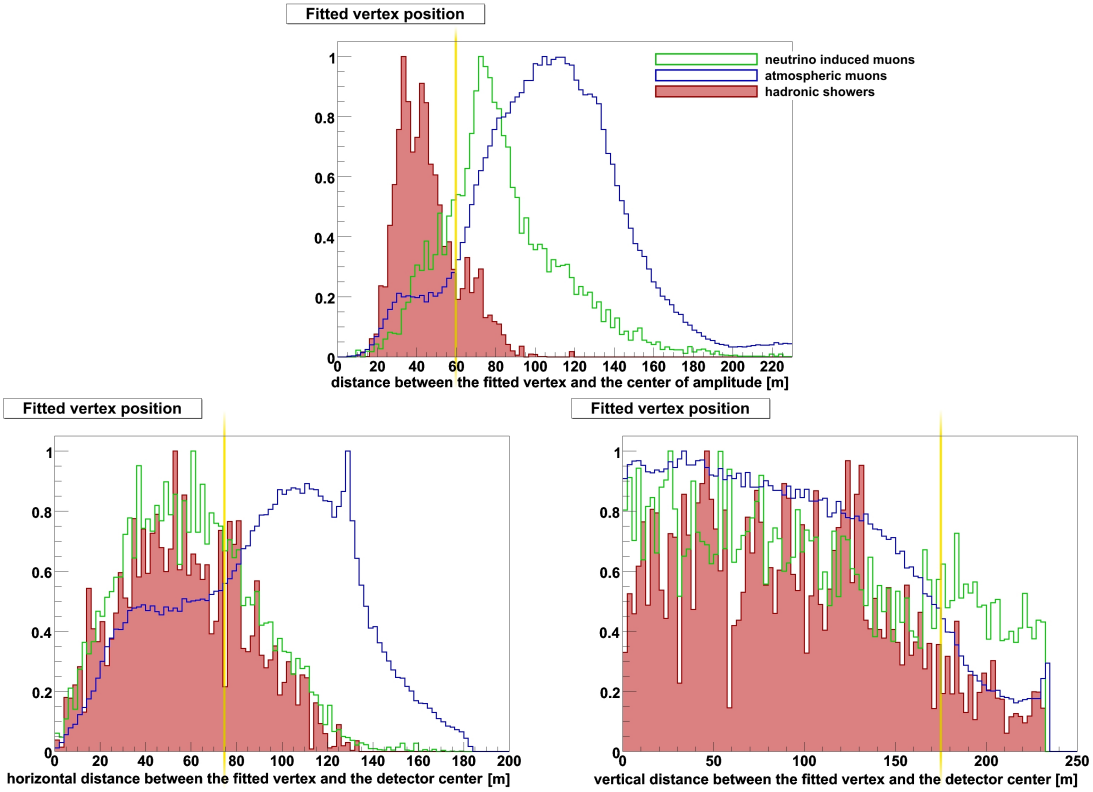

Figure 5.5: *Fitted vertex position relative to various reference points for different weighted event signatures after reconstructing them as showers.* The aim is to remove all neutrino induced muon tracks (green) and atmospheric muon events (blue) and keep as many shower events (red) as possible. The yellow lines mark an optimum compromise for the cuts. For each event type the distribution has been adjusted in a way that its maximum is set to 1.

The two lower plots show the distance of the vertex to the detector center. The marked cut values are quality cuts that remove all events with a fitted vertex position outside of the detector. For the 5 line detector the height is 352 m and the radius 75 m. According to the search space adjustments described in chapter 4.3.3 the fit boundaries are always restricted to at maximum the size of the can, which is 55 m in each direction around the detector size. No matter what fit boundaries the user has set, the minimization will never exceed these restrictions. The maximum distance of the vertex to the detector center thus is 75 m + 55 m = 130 m along the x- and the y-axis. For some atmospheric muon events the reconstruction algorithm stops at this boundary, either for x or for y, which can be seen as a little peak in the horizontal distance at 130 m. But as the search space is a square and not a circle, also more distant vertices are allowed (with its maximum horizontal distance at $\sqrt{130^2 + 130^2}$ m $\approx$ 184 m). The maximum allowed vertical distance is half the detector height plus the can length

(176 m + 55 m = 231 m). As the vertical direction contains just one dimension, the distribution shown down left ends at this maximum distance without the tail that is attached after the peak in the horizontal distance. When a cut on these values is to be applied on 12 line data, the different radius that is 101 m here has to be considered for a contained-cut.

5.2.5 The fitted direction

The final observables that are to be reconstructed are the energy and the direction of the inducing neutrino. Thus a cut on the neutrino direction seems to be a little bit misplaced at first sight. But after a closer look to the issue it becomes clear that the removal of all downgoing events is indispensable at least for muon track reconstruction, as there is no other possibility to get rid of atmospheric muons otherwise. Neutrino induced muons (including those from atmospheric neutrinos) arrive from all directions at the detector, whereas atmospheric muons only come from above.

But as shower events mostly have a completely different signature it is imaginable to abstain from this upgoing-cut and filter out atmospheric muons **and** neutrino induced muons by other means. Furthermore the upgoing-cut is no guarantee for a successful removal of all atmospheric muons as it is only possible to cut on the *fitted* direction and some downgoing muons will be misleadingly reconstructed as upgoing, as can be seen in Figure 5.6. On the other hand the *ANTARES* OMs are orientated 45° below the horizon, that is why the detector is more sensitive to upgoing events anyway. The MC distribution of the detected events on the left side in Figure 5.6 shows that below a zenith angle of 60° only a small fraction of all shower events will be detected anyway. So one can as well cut away events with a direction range from 0° to about 60°, which is a little bit less restrictive than a total upgoing-cut.

5.2.6 The amplitude quadrupole moment

The quadrupole moment is calculated for the same hit distribution that has been used for the reconstruction, in this case all those with an amplitude bigger 3 pe, along the reconstructed track, which is of course in shower events no real track but the direction of the inducing neutrino. The quadrupole moment is a measure for the spatial hit distribution along the track. If its 0, the hits are arranged spherically around one point on the track. A quadrupole moment of 1 corresponds to a hit distribution on a thin disk perpendicular to the track, whereas a negative value stands for a tube like distribution with its longest axis along the track. This tube is the more extended the smaller the value is (the minimum is at -2).

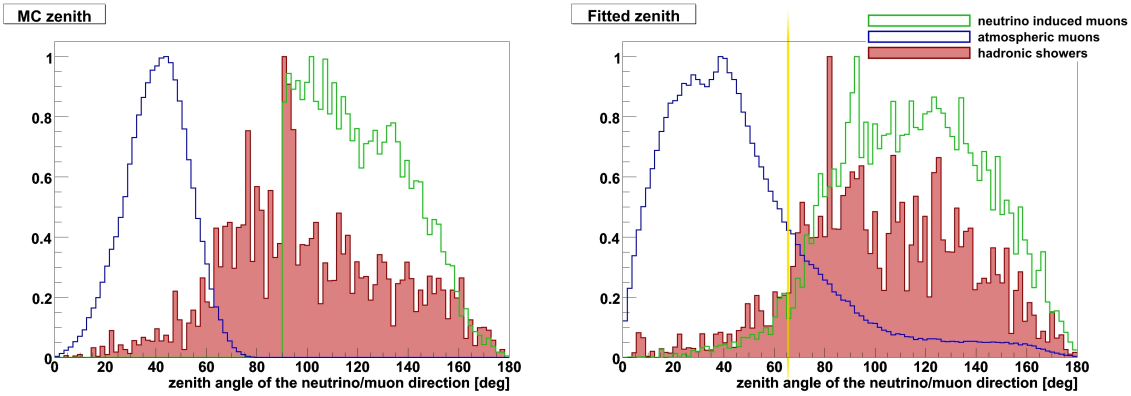

Figure 5.6: *MC and fitted zenith angle of the direction for different weighted event signatures after reconstructing them as showers.* A zenith angle of 0° is downgoing, whereas 180° is upgoing. The aim is to remove all neutrino induced muon tracks (green) and atmospheric muon events (blue) and keep as many shower events (red) as possible. The yellow lines mark an optimum compromise for the cuts. For each event type the distribution has been adjusted in a way that its maximum is set to 1. Note: The MC distribution of neutrino induced muons is abrupt at 90° as the simulation was restricted on upgoing events. One can figure the distribution as continued in a similar way as for hadronic showers.

For showers, whose hit distributions reach from spherical to slightly elongated, one therefore expects a quadrupole moment between 0 and −1, assuming a correctly reconstructed track of course. The quadrupole moment of muon track events is expected to be even more negative along their **real** muon track, as the track can be quite long. However, when reconstructing muon track events as showers, the quadrupole moment is expected to be more positive and reach up to 1 for some events. The reason for this is again that the only way for the algorithm to fit a spherical shape to the hit distribution is to choose a distant vertex and a track towards the hits, which let them appear more and more concentrated on a thin disk perpendicular to the track. (see also chapter 4.3.2 and 5.2.1)

Figure 5.7 shows the calculated values for the quadrupole moment for different event signatures. Thereby it is important to calculate this value with the hitmap *after* the hit constraints, as otherwise a lot of noise hits will deteriorate the calculation results. One can see that also the quadrupole moment is suited as a cut parameter for purity cuts.

5.2.7 The spherical shower parameter

Originally this search for potential cut values ended with the amplitude quadrupole moment. With all cut parameters listed above it can be tried to remove all atmospheric muon events to finally have a pure shower sample. Unfortunately there was no possible way to do so. Every cut also removes parts of the shower events, so that in the end

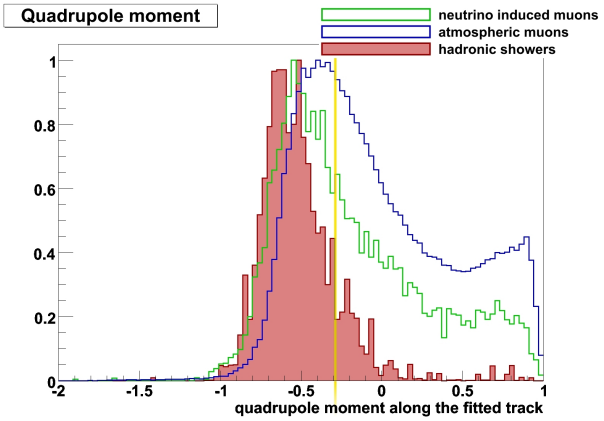

Figure 5.7: *Amplitude quadrupole moment along the fitted track for different weighted event signatures after reconstructing them as showers.* The aim is to remove all neutrino induced muon tracks (green) and atmospheric muon events (blue) and keep as many shower events (red) as possible. The yellow lines mark an optimum compromise for the cuts. For each event type the distribution has been adjusted in a way that its maximum is set to 1.

almost no shower will be left in the sample. In chapter 5.3.2 an attempt to find a suitable cut will be described in more detail.

Therefore another cut value has been added, which is called the *spherical shower parameter* and that is based on an idea from [55]. The hope is to finally solve the problem by cutting on a criterion that has nothing to do with any likelihood or specific fit boundaries, but deals with the event signatures themselves. The idea was to separate shower events from others on the basis of a distinctive characteristic of a shower event which is of course its spherical shape. Although it is not quite correct, for the calculation of this parameter it has been assumed, that the hit distribution in a shower event has a totally spherically symmetric shape.

In a symmetrical shower the light will spread isotropically in all directions from the interaction vertex. The expected arrival time of a hit at an OM that is located at the position $\vec{r}_{\mathrm{OM}}$ from the vertex is respective to the interaction time

$$t_{\mathrm{calc}} = \frac{|\vec{r}_{\mathrm{OM}} - \vec{r}_{\mathrm{vertex}}|}{v_{\mathrm{light}}} \tag{5.1}$$

where $v_{\mathrm{light}} \approx 22\,\mathrm{cm/ns}$ is the light velocity in water. The time residual Δt is the absolute difference between the actual measured arrival time and the calculated time based on these assumptions.

$$\Delta t = |t_{\mathrm{measured}} - t_{\mathrm{interaction}} - t_{\mathrm{calc}}| \tag{5.2}$$

For a shower event the time residuals for each OM, no matter how distant it is from the vertex, are expected to be small, if only the light from the shower really spreads

isotropically and spherically. If the shower though is more elongated the time residuals will slightly increase with the distance to the vertex.

For muon track events on the contrary one expect a different behaviour. Muon tracks generally have no interaction vertex inside the detector volume, from which the spherical shower parameter could be calculated. So at first a virtual vertex on the track has to be assumed. In this case the time residuals considering a spherical light propagation may be still quite small close to the vertex but will increase with the distance from the virtual vertex. For distant OMs the residuals will be much greater than for a shower event, as in showers all photons can spread with v_{light} from the vertex, whereas the muon propagates with almost vacuum light velocity $c = 30\,\text{cm/ns}$, which is much faster than the photons. Hence in track direction the Čerenkov photons from the track will arrive at the distant OM much earlier than an expected photon emitted at the vertex. Figure 5.8 gives a graphic illustration of what is meant here. The effect even intensifies if the virtual vertex lies outside the track, as in this case almost all hits contribute with a high time residual.

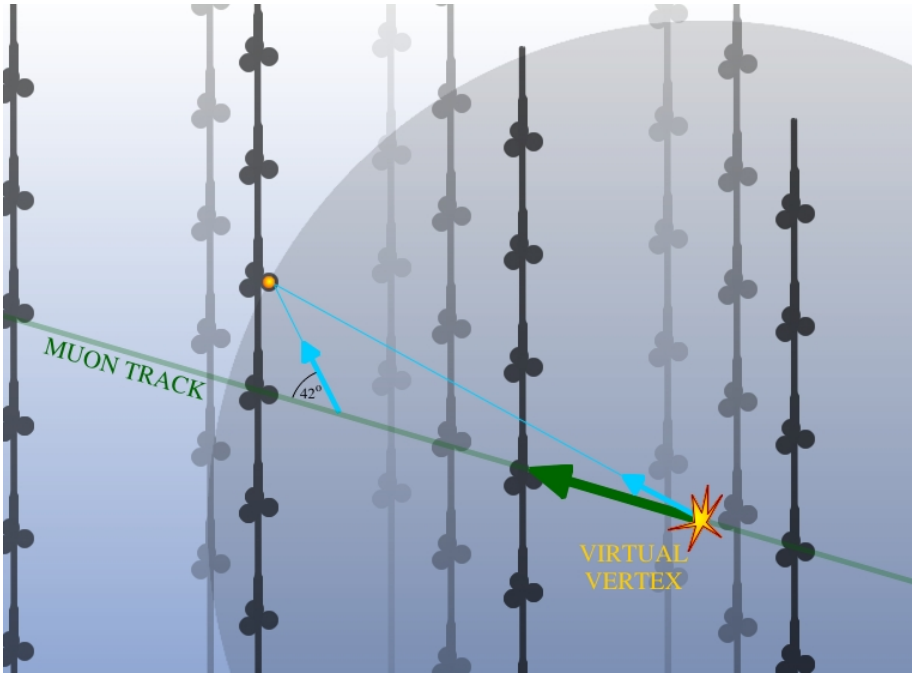

Figure 5.8: *Difference between the expected photon arrival time from the spherical shower assumption and the measured arrival time of the Čerenkov photon emitted from the muon track.* The muon track photon will arrive earlier at the distant OM than the assumed photon from the vertex shower, as the propagation velocity of the muon (green arrow) is greater than the light velocity in water (blue arrows).

The time residuals on distant OMs seem thus to be a perfect parameter to separate showers from tracks. The calculation of these residuals however requires, that noise hits have been removed previously, as statistical noise hits always produce a very big time

residual that would ruin the whole separation. Hence it is again neccessairy to use the hitmap with hits bigger 3 pe. By doing so one unfortunately can not just concentrate at distant OMs, as in most of the shower events there will be no hit left, that has a greater amplitude than 3 pe **and** appears far away from the vertex. So all hits have to be taken into account.

The spherical shower parameter is now the summation over all time residuals in one event. The already reconstructed event provides a (virtual) vertex, from which the residuals can be calculated. Those that are close to this vertex are quite small and will not contribute much to the sum. But if a residual of a distant OM, which is expected to be large for muon track events, is included into the sum, it will dominate the small residuals and thus sign that this event obviously has to be a track event. Finally the sum is normalized to the number of hits.

$$\frac{1}{N_{Hits}} \sum_{i=1}^{N_{Hits}} \Delta t_i \tag{5.3}$$

Evidence that the spherical parameter works is given in Figure 5.9, where the parameter is plotted for the three different signatures. However, this criterion offers, like all the others described above, no way to get rid of the muon background with just one single cut. To filter them out finally one has to try to combine more than one of these cut parameters.

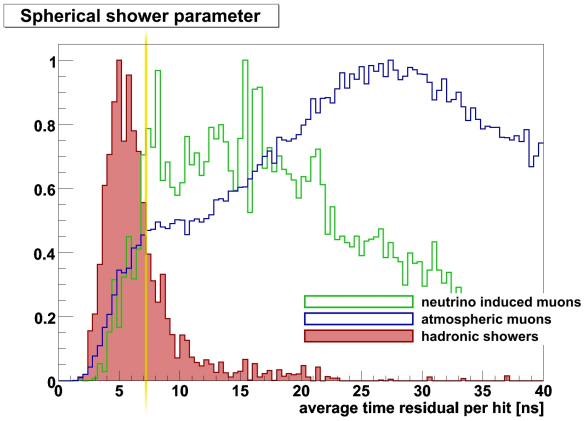

Figure 5.9: *Spherical shower parameter respective to the fitted vertex position for different weighted event signatures after reconstructing them as showers.* The aim is to remove all neutrino induced muon tracks (green) and atmospheric muon events (blue) and keep as many shower events (red) as possible. The yellow lines mark an optimum compromise for the cuts. For each event type the distribution has been adjusted in a way that its maximum is set to 1.

5.3 Combined quality and purity cuts

The introduced potential cut parameters of the last chapters can now be used for some quality or purity cuts. Although these parameters have all been calculated from 5 line reconstruction data, they can be tried for 12 line data as well, as following the investigations in chapter 4.4 the difference between the 5 and the 12 line detector is negligible concerning shower reconstruction. Nevertheless all the studies shown above also have been done for 12 line reconstruction. What came out was that indeed the calculations are quite identical to the 5 line analysis, except of course that for the horizontal distance between the detector center and the reconstructed vertex. If a cut on this parameter is wanted, the final cut value has to be adjusted to the detector radius which is bigger for the 12 line detector.

5.3.1 Quality cuts to improve the resolutions

Before trying to filter out the muon background, first it has been tried to improve the resolutions given in chapter 4.8 by applying some quality cuts on the pure shower samples after the reconstruction. To do so, those parameters have been chosen that contain the information about the quality of the fit. These are *the horizontal and vertical distance between the vertex and the detector center*, as events with an interaction vertex outside the detector obviously are not reconstructed with the same quality as events that are totally contained due to the loss of hits. The cut on these two parameters is done for the yellow marked cut values in Figure 5.5, which corresponds to a selection of all contained events. For the 12 line detector of course the cut value for the horizontal distance has been adjusted to 101 m, which is the detector radius in this case.

The other four parameters are the likelihoods, as an event with a lower final likelihood value is obviously better reconstructed than one with a high value. Again the yellow lines in Figure 5.2 have been used as cut values. Originally these cut values are adjusted to remove as many muon events as possible but can be used here as well. Of course the cut can be strengthened to still lower likelihoods, but thereby one loses more events and decrease the efficiency. When applying the here mentioned cuts the following resolutions and efficiencies can be derived:

Applied quality cuts:

horizontal distance between vertex and detector center:	$[0\,\text{m}, 75\,\text{m}]$ (for 5 lines)
horizontal distance between vertex and detector center:	$[0\,\text{m}, 101\,\text{m}]$ (for 12 lines)
vertical distance between vertex and detector center:	$[0\,\text{m}, 175\,\text{m}]$
pre likelihood:	$[0, 1.9]$
final likelihood:	$[0, 8.2]$

time likelihood: $[0, 1.8]$
amplitude likelihood: $[0, 6.4]$

event type	**angular error median**	**vertex error median**	**log shower energy error RMS_0**	**reconst. events per year**	**efficiency**
5 line MC	$(10 \pm 1)^\circ$	(3.6 ± 0.5) m	(0.4 ± 0.1)	(17 ± 4)	$(3.3 \pm 0.9)\%$
12 line MC	$(9 \pm 1)^\circ$	(3.2 ± 0.5) m	(0.4 ± 0.1)	(61 ± 8)	$(4.5 \pm 0.7)\%$
5 line MC Genova	$(8 \pm 1)^\circ$	(2.6 ± 0.5) m	(0.4 ± 0.1)	(17 ± 4)	$(2.3 \pm 0.6)\%$
12 line MC Genova	$(8 \pm 1)^\circ$	(2.6 ± 0.5) m	(0.4 ± 0.1)	(66 ± 8)	$(3.4 \pm 0.5)\%$

The efficiency has been calculated by dividing the number of reconstructed shower events by the number of all triggered shower events. The errors are calculated as described in chapter 4.8. For more comments on the resolutions see the notes on the reconstruction quality without cuts in that chapter.

Whereas the comparison of *Genova* and *old* PM parametrization on uncut samples showed a discrepancy in the number of reconstructed events per year, the numbers are equal after the quality cuts. When using the *Genova* parametrization one obtains more triggered events and more reconstructed events per year before the cuts. The fact, that the flux is now almost the same, means that the *Genova* parametrization indeed increases the triggered rate but not necessarily the reconstruction quality for those events that have been added. The quality cuts seem to remove the additional *Genova* events, that is why the efficiency is lower in this case than with the *old* generated events, as the calculation of the efficiency relates to the original number of triggered events.

5.3.2 Purity cuts to suppress the muon background

As seen above, the use of cut parameters to improve the resolutions works in principal and might be quite useful. However, when analyzing real data getting good resolutions is not the only problem to solve. The huge amount of muon background events that in this case are atmospheric and neutrino induced muons has to be filtered out. By applying the six quality cuts mentioned above the filtering process is far from successfully removing the whole background. To recall the numbers which one has to deal with, the following table gives an overview of the expected fluxes of reconstructed events. The calculation of the purity is done by dividing the number of reconstructed shower events

by all expected reconstructed events. The efficiency applies only to shower events of course, where from now on the calculated values are derived from all three shower samples in common. It is calculated by dividing the number of reconstructed shower events by the number of all triggered shower events.

	showers per year	**neutrino induced muons per year**	**atmospheric muons per year**	**purity**	**efficiency**
5 lines	(42 ± 6)	(134 ± 45)	$(5.0 \pm 4.4) * 10^6$	$(8.4 \pm 8.6) * 10^{-4}\%$	$(7.0 \pm 1.3)\%$
12 lines	(171 ± 13)	(922 ± 308)	$4.4 * 10^6 \pm 2 * 10^3$	$(3.9 \pm 0.3) * 10^{-3}\%$	$(10.8 \pm 1.1)\%$

The errors for the number of showers are the standard error ($\sqrt{N}$) from Poissonian statistics. The neutrino induced muon MC sample has been generated just for upgoing events. As the detector is more sensitive to the lower hemisphere one does not know the exact detected flux of this event signature that comes from above. Hence the number of expected events for upgoing events, derived from the MC weights, has been multiplied by a factor 1.5, and for the error by 0.5. This allows for an expected number in a range from once to twice the upgoing events. The large error in the atmospheric muon flux is due to the discrepancies between the *Corsika* and *Mupage* generation (see chapter 5.1). The given value is the mean between the two, the error is half the difference, so that in principle it could be the *Mupage* or the *Genova* flux. For 12 line data only the *Genova* generation is available, why here the Poissonian error has been applied, which is of course very small. All further errors are calculated via Gaussian error propagation. (When comparing *Corsika* 12 line and *Corsika* 5 line data, of course the muon flux for a 12 line detector is greater than for the 5 line detector! The reason why in these tables the 5 line flux is higher than the 12 lines flux is that *Mupage* predicts a flux of about one magnitude above *Corsika*, which also effects the mean value.)

The next step is to filter out the muon background by cutting on as many parameters as possible. The major problem here is that the cut parameters are not independent from each other. After the first look on the distributions one might be quite hopeful that one cut removes the greater part of all muon events, and others filter out the rest. Unfortunately the greatest part of muon events that will be removed by one cut, will as well be removed by another. An event that takes place at the edge of the detector probably will cause a convergence of the minimization right at the fit boundaries for the time or the vertex fit which consequently results in big likelihood values. Thus it is the same event that will be filtered out by every cut. However, due to the big number of cut parameters it looked quite promising that it might work.

In the following all introduced parameters from above are used for the cut with its cut value at the yellow marked lines in the corresponding previous plots. In addition

to the six quality cuts from the last chapter these are:

Applied purity cuts:

interaction time respective to the first triggered hit:	$[-240\,\text{ns}, 50\,\text{ns}]$
distance between vertex and center of amplitude:	$[0\,\text{m}, 60\,\text{m}]$
zenith angle of direction:	$[65°, 180°]$
quadrupole moment:	$[-2, -0.3]$
spherical shower parameter:	$[0, 7.5]$

+ quality cuts

The following tables list the results for the *ShAuerReco* reconstruction after all these cuts. The errors have been calculated in the same way as for the tables above.

event type	**angular error median**	**vertex error median**	**log shower energy error RMS_0**	**reconst. events per year**	**efficiency**
5 line MC	$(9 \pm 1)°$	$(3.1 \pm 0.5)\,\text{m}$	(0.4 ± 0.1)	(12.7 ± 3.6)	$(2.5 \pm 0.8)\%$
12 line MC	$(8 \pm 1)°$	$(2.9 \pm 0.5)\,\text{m}$	(0.3 ± 0.1)	(47.5 ± 6.9)	$(3.5 \pm 0.6)\%$
5 line MC Genova	$(8 \pm 1)°$	$(2.6 \pm 0.5)\,\text{m}$	(0.4 ± 0.1)	(12.8 ± 3.6)	$(1.7 \pm 0.5)\%$
12 line MC Genova	$(7 \pm 1)°$	$(2.5 \pm 0.5)\,\text{m}$	(0.4 ± 0.1)	(48.6 ± 7.0)	$(2.5 \pm 0.4)\%$

	showers per year	**neutrino induced muons per year**	**atmospheric muons per year**	**purity**	**efficiency**
5 lines	(12.8 ± 3.6)	(1.5 ± 0.5)	$(5.2 \pm 3.8) * 10^3$	$(0.2 \pm 0.2)\%$	$(2.1 \pm 0.7)\%$
12 lines	(48.0 ± 6.9)	(31.2 ± 10.4)	$4.5 * 10^3 \pm 67$	$(1.1 \pm 0.2)\%$	$(3.0 \pm 0.5)\%$

The following can be seen in these tables: In comparison to the reconstruction quality after the quality cuts in chapter 5.3.1 the resolutions improve only slightly, whereas one gains a factor of over 200 in the purity of the shower sample respective to the original sample of all reconstructed events! Therefore one has to put up with a further decrease of the efficiency by a factor of 3. A factor 3 versus 200 - what actually sounds quite good is unfortunately not good enough. Regarding the 5 line data, there are still several thousand atmospheric muon events left per year, whereas the number of shower events has been diminished from originally about 42 to 13. Figure 5.10 shows the distributions for the cut parameters after these cuts. Obviously there is no further single cut that can eliminate all muons and spare the shower events.

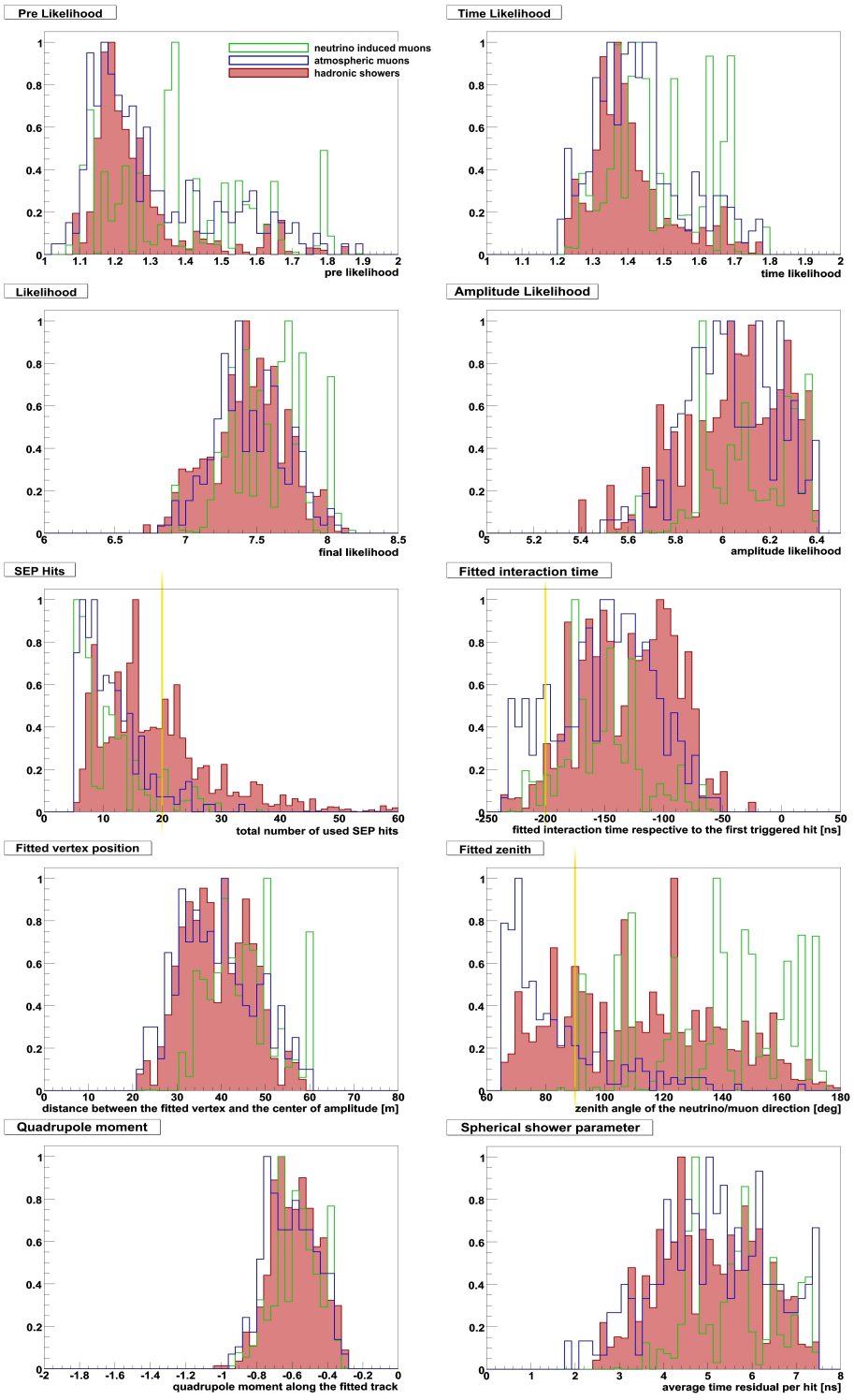

Figure 5.10: *Distributions of the cut parameters after the quality and purity cuts.* The three yellow bars mark the cut values for the following final cut to remove all atmospheric muons.

A huge number of possible combinations of these parameters have been tried out to finally solve the problem and in the end a solution came out, that filters out all atmospheric muons. The additional cuts values that are necessary to achieve this are printed as yellow lines in the table of plots in Figure 5.10.

Additional cut values for the "final" cut:

interaction time respective to the first triggered hit: $[-200\,\text{ns}, 50\,\text{ns}]$
zenith angle of direction: $[90°, 180°]$
sep hits: $[20, \infty]$
+ purity cuts from above
+ quality cuts

But unfortunately the price is high for these cuts. Even if all atmospheric muons have been filtered out now, there are only about 4 shower events left from originally 42 reconstructed and about 600 triggered in one year of 5 line data. For 12 lines 15 events from about 170 reconstructed and about 1500 triggered remain. The whole scale of the impact of the final cut is listed in the following tables.

event type	**angular error median**	**vertex error median**	**log shower energy error RMS_0**	**reconst. events per year**	**efficiency**
5 line MC	$(7 \pm 1)°$	$(3.7 \pm 0.5)\,\text{m}$	(0.5 ± 0.1)	(3.5 ± 1.9)	$(0.7 \pm 0.4)\%$
12 line MC	$(6 \pm 1)°$	$(2.3 \pm 0.5)\,\text{m}$	(0.4 ± 0.1)	(13.5 ± 3.7)	$(1.0 \pm 0.3)\%$
5 line MC Genova	$(6 \pm 1)°$	$(2.3 \pm 0.5)\,\text{m}$	(0.4 ± 0.1)	(5.0 ± 2.2)	$(0.7 \pm 0.3)\%$
12 line MC Genova	$(5 \pm 1)°$	$(2.0 \pm 0.5)\,\text{m}$	(0.4 ± 0.1)	(16.4 ± 4.0)	$(0.9 \pm 0.2)\%$

	showers per year	**neutrino induced muons per year**	**atmospheric muons per year**	**purity**	**efficiency**
5 lines	(4.1 ± 2.0)	(0.2 ± 0.1)	0	$(95.3 \pm 7.8)\%$	$(0.7 \pm 0.4)\%$
12 lines	(14.6 ± 3.8)	(7.8 ± 2.6)	0	$(65.2 \pm 24.5)\%$	$(0.9 \pm 0.3)\%$

Even if these final cuts are applied to data taken with a large volume detector, where then maybe a few hundred showers events will remain, one has to consider that these cuts also restrict the range of the potentially detectable shower energy. A cut on at least 20 *SEP hits* means that only events with 20 hits ore more with an amplitude bigger than 3 pe will be detectable, which of course removes all low energetic events from the

sample. On the other hand the energy spectrum will also be restricted from above, as high energetic events will always be reconstructed slightly worse (see chapter 5.2.1). Hence a cut on the likelihood parameters will restrict the maximum reconstructable energy. From the MC shower energy of all remaining shower events one can derive that out of the original samples which are generated with a neutrino energy range from 10^2 GeV to 10^7 GeV, only those within an energy range of 10^3 GeV to 10^5 GeV survive the cuts.

But there are also some good news. The final cut to filter out atmospheric muons works with 5 line data as well as with 12 lines, and it filters out all atmospheric muons from the MC samples, no matter if they are generated with *Corsika* or *Mupage*, i.e. one does not have to decide whether the one or the other generation should be trusted, one does not have to argue about OM parametrization, it will filter them all out. How statistically significant this "0" indeed is, is admittedly to be checked up further. The total number of events in the atmospheric muon sample (*Mupage* for 5 lines) is $7.4*10^7$, whereof $3.3*10^6$ are passing the *ANTARES* trigger. All these events have been filtered out. But although this number is quite huge, the whole sample corresponds to a data taking time period of only about nine days. So this actually means that one filtered out all atmospheric muons within a time period of 9 days. The question to which extend the remaining 40 times "9 days" will, statistically seen, differ from the first ones, may be answered by someone else.

5.4 Some closing words on muon events in the *ShAuerReco*

As shown above it is difficult to analyze real data with the *ShAuerReco* algorithm by now without some further work on either the algorithm itself or the pre-hitselections and the cuts afterwards. One could wonder about why there are still so many muon events, which obviously differ quite a lot from shower events concerning signature and hit distribution, left after all the quality cuts. Apparently the *ShAuerReco* reconstruction recognizes them as shower events, though they are not, and succeeds in fitting a spherical shape to the hits, which results in good likelihoods that reach the same quality as for shower events.

The first idea to identify these surviving events was that this could have to do with muon bundle events with a multiplicity bigger than 1. In those events not just single muons go through the detector but many almost parallel muon tracks at the same time (a muon bundle) cause a signal. The number of muons in a bundle is called multiplicity. However, this theory was shown to be wrong. Figure 5.11 shows

the multiplicity distribution of all atmospheric muon events before and after the cuts. The distribution of all triggered events follows basically the generated distribution, described further in [51]. After the cuts single muons are still present in the sample, which proves that it is not the muon bundles that are misinterpreted as showers. Events with high multiplicities rather will be removed by these cuts!

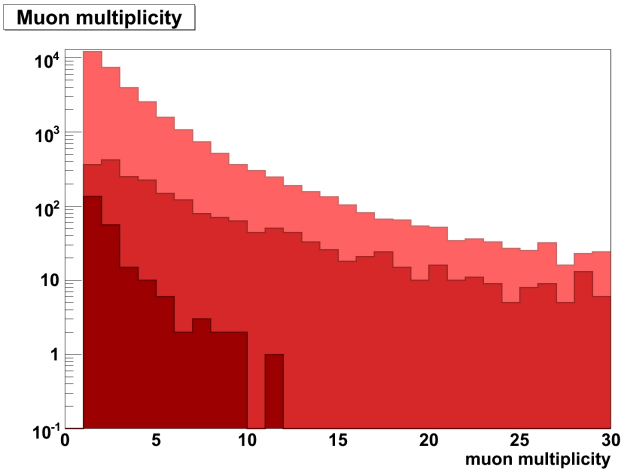

Figure 5.11: *Muon multiplicities of Mupage atmospheric muons at different cut stages.* The upper histogram (light red) shows the distribution for triggered events, the second (red) events that pass the *ShAuerReco* hit constraints and will be reconstructed whereas the lower histogram (dark red) represents all events before the final cut.

Remarkable is furthermore that there are still neutrino induced muon events in the sample after the final cuts, while all atmospheric muons are filtered out. There obviously has to be a difference between neutrino induced and atmospheric muons concerning their signature in the detector. Atmospheric muons are produced in the earth's atmosphere and what reaches the detector is just the muon track. Neutrino induced muons can be generated under water (and also within the detector volume) where always an additional hadronic shower is produced at the interaction vertex. If this vertex lies close or in the detector the event will be a mixed type event containing a muon track and a shower that both are part of the MC event generation for neutrino induced muons. Hence such an event, or at least parts of it, is more shower-like than an atmospheric muon. A closer look on the remaining neutrino induced muon events after the final cut has shown that it is indeed the events with its vertex within the detector that pass the final cuts.

Furthermore the basic cut to filter out atmospheric muons is the upgoing cut, which is of course less effective for neutrino induced muons, as they arrive from all directions. In the *Mupage* production atmospheric muons originate from a zenith angle of maximum 85°. As the blue histogram in Figure 5.6 has shown, the greater part of all (downgoing) atmospheric muons will be reconstructed as downgoing, however with a

great error in the direction, as the *ShAuerReco* is not intended for the reconstruction of muon directions, but anyhow the classification in upgoing/downgoing seems to work. Most of the remaining events that are wrongly reconstructed as upgoing events are thus fitted so badly that the cut on the likelihoods will remove them. Neutrino induced muons that originally can also be upgoing therefore cannot be removed as easily.

However to fully understand the details that cause these problems, and if they are just effects that occur when using MC events or will as well be noticeable in real data, one has to know the exact assumptions that have been made when generating the MC files. To check the conformity of the MC studies within this thesis to real data and to find out if real events behave like it is expected from MC events, a small sample of real 5 line data from *ANTARES* has been processed. The results are described in the last chapter.

Chapter 6

First tests with real data

6.1 The first real data samples

This last chapter is to give a first insight in reconstructing real data with the *ShAuer-Reco* reconstruction. In chapter 4 the algorithm has been prepared to be able to reconstruct real *ANTARES* data by figuring out the optimum parameters to run the module. Chapter 5 dealt with the neutrino induced and atmospheric muon background and how to get rid of it. Unfortunately the result of these attempts to classify the different event signatures showed that it is not possible to filter out the background without decreasing the reconstruction efficiency down to below 1%, which corresponds to about 4 expected shower events per year in the 5 line detector.

However, some real data has been reconstructed with the algorithm and afterwards the cuts have been applied on it, in order to see, whether the obtained distributions of the cut parameters somehow correspond to those from MC events. As this is not a physics analysis but just a first look, the analysis has been restricted to one month of the 5 line data taking period in 2007. 85 golden runs from June 2007 has been processed in the same way, as it was done with MC events. (Actually there have been 89 golden runs in June, but four of them caused problems with the calibration). The corresponding python script is printed in the appendix B.4.

The runs last about 5 hours each, thus the whole reconstructed sample represents a total data taking time of about 17 days. A golden run is a description for a run that has been recorded under optimum conditions. The requirements on a run to be named as golden is for example a background baseline (from ^{40}K-decays and bioluminescence) that does not exceed 120 kHz and a burstfraction (bioluminescence) of maximum 20% (see chapter 2.4 for further details on background sources). More conditions of a golden run are listed in [56].

The results are shown in Figure 6.1, where the distributions for the real data (green)

are printed in comparison to the expected ones from MC (red). For the latter ones the three shower MC samples, the neutrino induced muon sample *LE prod0401* and the *Mupage* sample for atmospheric muons (the MC files are further described in chapter 5.1) have been combined accounting for the correct ratio in the fluxes. Hence the plots basically show the distributions from atmospheric muons. The printed fluxes represent the data taking time of 17 days and thus can be directly compared to those from real data.

What is shown in the plots is that basically the distributions for real data follow the MC studies, what is quite pleasing. However there are some differences, of which the most remarkable is the discrepancy in the flux. The *Mupage* MC sample that has been generated with the new *Genova* OM parametrization predicts a total number of $4.4 * 10^5$ atmospheric muons events within the time period of 17 days, whereas only $2.5 * 10^5$ are detected. On the other hand when trusting in the *Corsika* atmospheric muon production one would expect just $2.7 * 10^4$ events. From this one can conclude that the *Mupage* sample is closer to the truth, as its fluxes are just twice the real measured fluxes, whereas *Corsika* is a factor 9 too low.

This discrepance in the flux can be explained by aging processes to some extend. Of course, the MC generation assumes perfect working hardware. Aging processes within the optical modules that might decrease the sensitivity can effectuate that not all events that would actually pass the trigger are indeed detected. Furthermore some of the OMs are damaged, which also decreases the number of detected photons. To answer the question, whether the discrepance can be completely imputed to those effects or is also due to some features during the simulation, one has to have a closer look on the generation of the atmospheric muon MC files.

However, the major problem that has to be mentioned concerning the event classification is that, regarding the likelihoods, the real events are reconstructed slightly worse than MC events, as they have bigger likelihood values. As far as this phenomenon affects only muons, this would be no problem, as it will even make it easier to filter them out. But in fact one does not know, if this is not a global effect that affects also showers. If so, this could mean, that the cuts will not work anymore, as they possibly will remove everything. To find this out it is necessary to reconstruct the data of the whole year and see what remains after the cuts.

Furthermore there are greater differences regarding the quadrupole moment and the fitted zenith angle. While among MC events there are much more events with a quadrupole moment around 0.5 (spherical hit distribution, somewhat elongated along the fitted track) than with one of 1 (flat hit distribution almost perpendicular to the track), the distribution for real data is almost uniformly distributed. This means that in a real sample even more events are affected by the rotated track phenomenon described

in chapter 4.3.2. The algorithm fits a track perpendicular to a plane distribution of only a few hits, as this is the best way to get a spherical light distribution from a distant vertex.

Another problem is the fact that in real data obviously more atmospheric muons are reconstructed as upgoing, which of course makes the upgoing cut less effective. On the other hand the *ShAuerReco* is not designed to reconstruct directions of muon track events, why the final fitted direction not necessarily has something to do with the real direction for muon events. However the algorithm can obviously also give a rough estimation on the muon track, as otherwise one would not see an accumulation of downgoing events. The hope is again that all atmospheric muon events misinterpreted as upgoing, have likelihood values that are high (bad) enough to be filtered out by other cuts.

6.2 Quality and purity cuts on real data

To finally find out how these discrepancies influence the purity cuts, the same cut procedure as shown in chapter 5.3.2 has been applied to the reconstructed real sample. What has been referred to as first *quality and purity cut* during the MC studies that is a cut on most obvious values represented by the yellow lines in the plots 5.2 to 5.9 has now been used to filter out real atmospheric muons. The results are given in Figure 6.2.

After the first cuts the discrepancy in the flux between MC and real data increases, which is an indication for the worse reconstructability of real events. Whereas before the cuts about 57% of the expected *Mupage* events are actually detected, it is now only 7%. At this cut level 0.6 hadronic shower events are expected from MC events but also about 420 atmospheric muons (according to *Mupage*) within the data taking period of 17 days. According to Poissonian statistics for a mean value of 0.6 there could be either 0 or 1 or maybe 2 shower events within this sample of totally 420 remaining MC events. Concerning the real sample there are two possibilities now: either there are really 0, 1 or 2 real shower events in the sample and the real atmospheric muons can be indeed filtered out more easily, or the worse reconstructability affects all types of events and the remaining events are just chosen incidentally. Unfortunately with these low statistics it is not possible to argue for the one or the other option.

Finally it was applied what was called the *final cut* in chapter 5.3.2. Afterwards of course no event will remain, as just the *SEP hits* cut that demands the number of *SEP hits* to be bigger than 20, will remove all events alone. From MC one expects 0.2 shower events and 0.01 neutrino induced muons after the final cuts. Poissonian statistics give

a probability of 82% to measure no event at all, why the fact that everything is filtered out is not so bad at all, but actually this could mean as well that the cuts are too strong and there will never be an event left, no matter how long *ANTARES* is taking data.

The next step to figure this out would be to process all golden runs from 2007, which, however, will not be done at this point. All golden runs taken together represent a data taking period of 127 days, where one can expect 1.4 shower events after the final cut. Thus with a probability of 94% something between 0 and 3 events will be measured. On the other hand the probability to detect 0 again is still 25%. So if there was still no event within 127 days, this again cannot prove or disprove anything. The detection of one or two events on the other hand could be regarded as a small success. But by the way: to invest so much computing power in proving that a cut procedure with an efficiency of less than 1% works, might be designated as "overkill".

And thus the final word is: there is simply too few statistics in order to make any concrete statements, but as has been shown in Figure 6.2 it might look quite promising...

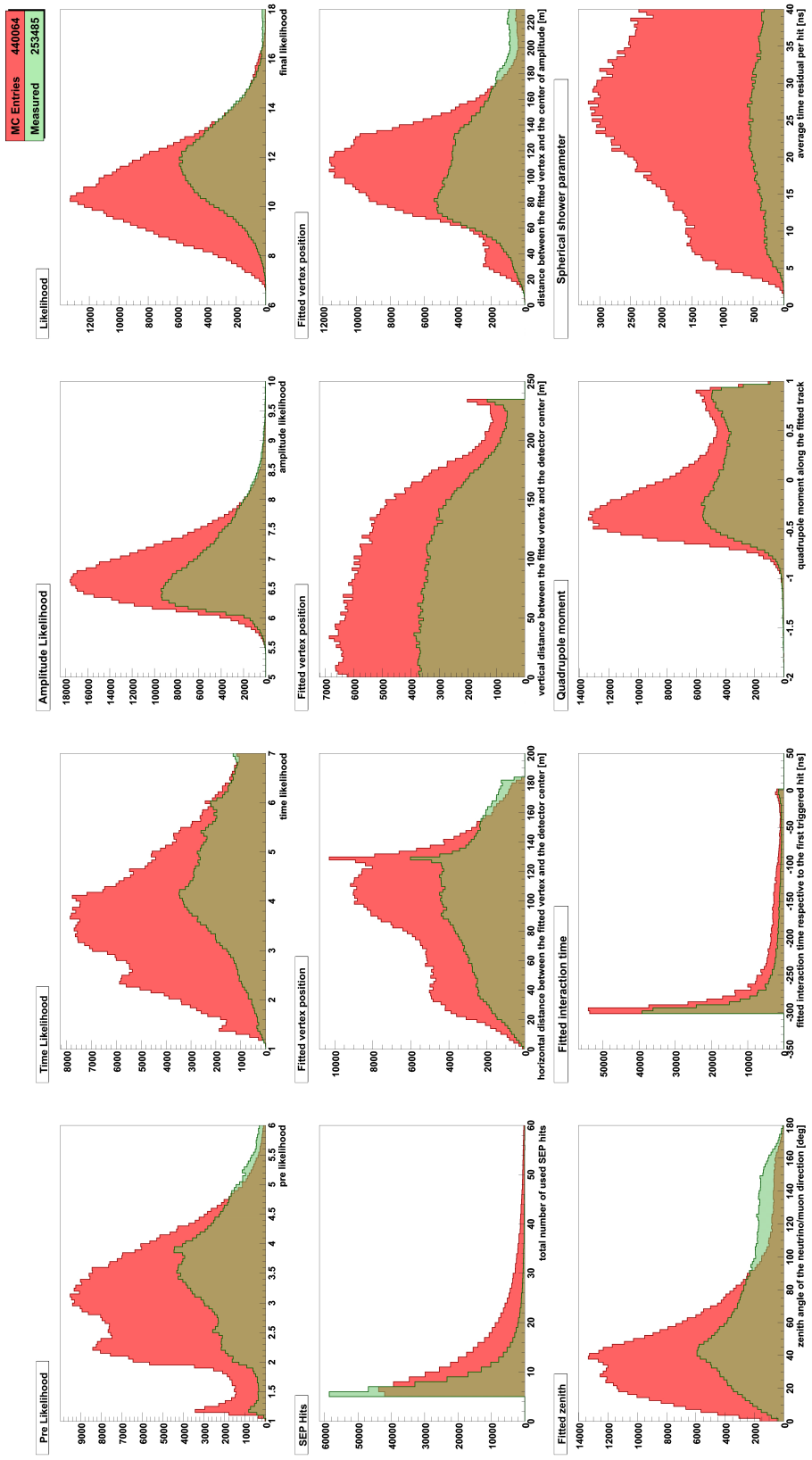

Figure 6.1: *Distribution of the cut parameters for real (green) and MC (red) events.*

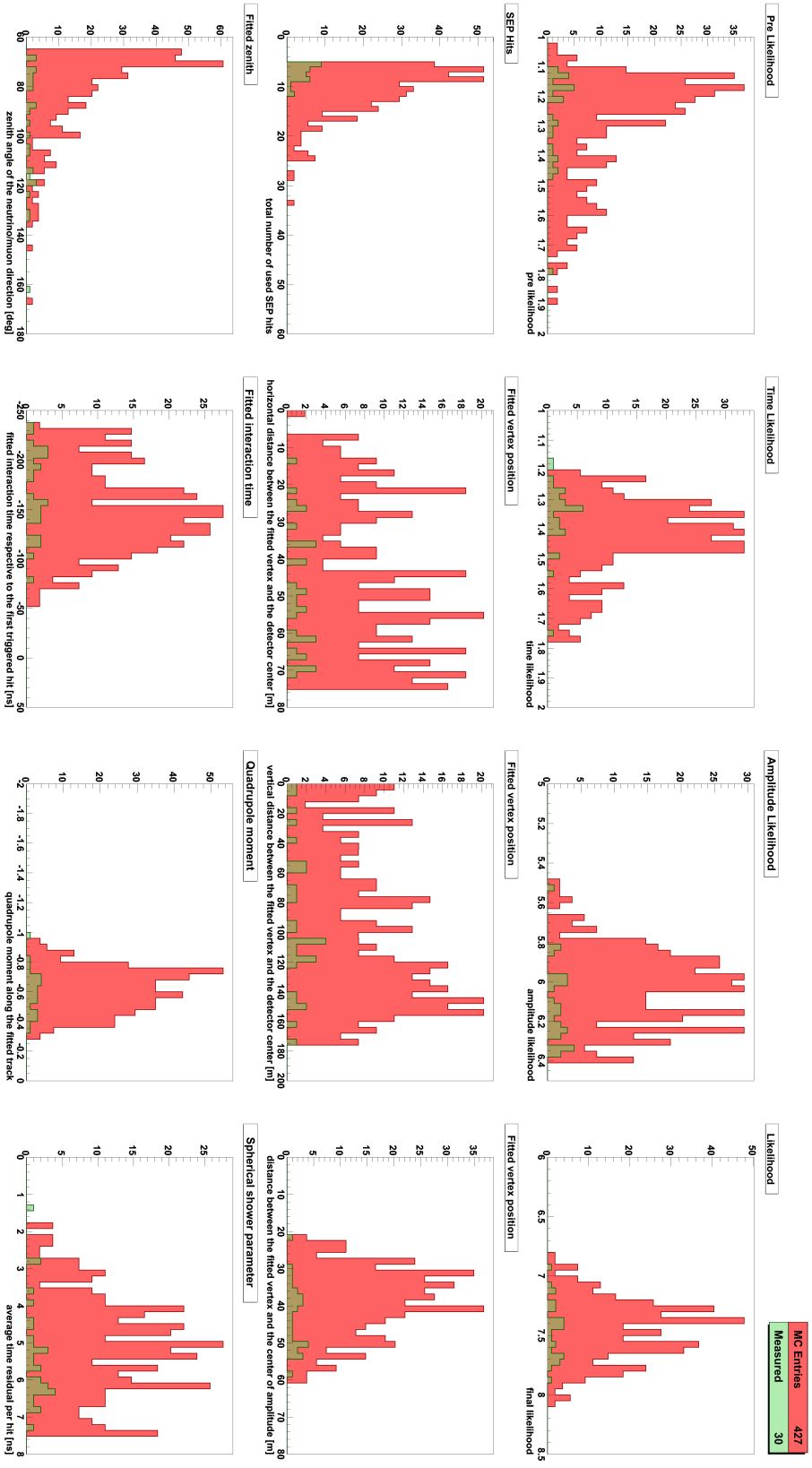

Figure 6.2: *Distribution of the cut parameters for real (green) and MC (red) events after first purity cuts.*

Summary and outlook

The outcome of this thesis is that the *ShAuerReco* reconstruction algorithm works well within the given resolutions from chapter 4.8 without any quality cuts or those from chapter 5.3.1 after cuts and is ready to be applied to real shower events. The optimum start values and fit boundaries for the *Simulated Annealing* minimizer have been determined to guarantee a proper reconstruction of MC and real events. Thereby the fit boundaries for the interaction time fit have to be adjusted to the first triggered hit in a real event. The search space for all observables has to be kept preferably small to avoid ambiguities in the reconstruction but on the other hand has to be big enough to include the true values for the interaction time, the vertex position, etc.

Furthermore it has been showed that a bigger detector gives no great improvement to the reconstruction quality for most of the events as long as the OM density is not increased. Due to the strong hit constraints that are required for a successful reconstruction the algorithm is able to process shower events including a background noise rate up to 150 kHz per OM. In principle the reconstruction even works for noise rates up to 300 kHz where admittedly the quality in the shower energy reconstruction worsens by a factor of 2. Changing hardware properties of the OMs, e.g. the integration gate and saturation level, however, have only a minor effect on the reconstruction quality and can be neglected.

Finally various post reco cut parameters have been introduced to seperate real showers from the huge number of muon track events that are basically atmospheric muons. A combined cut on multiple parameters, like likelihood values, the vertex position, number of used hits, the interaction time, the neutrino direction, the amplitude quadrupole moment and a parameter that is called the spherical shower parameter and that basically is a summation of time residuals, is able to filter out all atmospheric muons but will as well strongly deplete the sample of shower events.

Attempts to increase the efficiency by replacing the given hit constraints by another hit selection or simply the *triggered hits* unfortunately led to a quite bad reconstruction quality. What nevertheless looks promising is that the first reconstructions of real data with the *ShAuerReco* show a quite good agreement of the results with those from MC studies. However, to finally be able to do physics analysis with the algorithm, still a

long way has to be gone. There are three main points, where some work is required.

The first thing is of course to come up with either some better cut parameters or a tool that systematically looks for correlations between the single parameters in order to possibly find better cut values. The *TMVA* package [57] for multivariate data analysis (based on the *ROOT* framework) is such a tool, which is at the moment used by Friedrike Schöck for event classification even before any reconstruction is done. Either this pre classification will be good enough in the foreseeable future and can provide a pure shower sample or the *TMVA* will help with "post classification" when applied to the cut parameters. This could be worth a try.

The other thing, which is probably much more important, is to increase the efficiency of the algorithm from scratch. Even without any cuts the maximum possible efficiency from MC events is just $7-11\%$ regarding a realistic flux in the energy range of 10^2 GeV to 10^7 GeV. Of course this problem is caused by the strong hit constraints filter. It would be fine if somebody would find a more sophisticated background rejection than the 3 pe-criterion. But due to geometrical reasons the requirement of at least 5 hits on 3 strings is mandatory. What comes supposedly along with such a new hit selection is the need for a solution of the noise problem. As the algorithm is extremely susceptible to noise hits, the strong hit constraints have become necessary in the first place. If they can be relaxed, probably the algorithm has to be modified in a way that noise hits no longer will affect the whole reconstruction.

The third, but more elaborate, solution is to modify the implementation of the *ShAuerReco* algorithm in a way, that the reconstruction will not only consider OMs, where a hit has been detected, but also take all OMs with **no** hit into account. In chapter 4.3.2 it was shown how a rotated track can restore a spherical shape even if there originally was none, or worse even if it is a muon track. In most of the cases the minimizer will run to its fit limits, which made it possible to filter out these events. But still a large number of muons remain in the sample. A reconstruction algorithm that also considers where no hit has occurred, will not allow the minimizer to fool the actual algorithm by simply rotating the track. One could imagine, that with such a feature even more muon events will get a high likelihood value and thus can be better removed from the sample.

Appendix A

The *KM3Tray* framework

The *KM3Tray* framework is a modification of the *IceTray* software framework used by the *IceCube* Collaboration for physics analysis [58]. The *IceTray* framework has been developed for the data analysis of the *IceCube* experiment. In 2008 the *IceCube* Collaboration granted the *KM3NeT* experiment access to the *IceTray* code, whereupon the software group in Erlangen adapted the *IceTray* framework to the requirements of the analysis of underwater neutrino telescopes and included the *ANTARES* software into the framework, so that now the whole data analysis for *ANTARES* can also be done within the *KM3Tray*.

The essential elements of a software framework are a modular code and a corresponding data flow. As the physics analysis of experiments like *ANTARES* is very complex, a user cannot and is not expected to understand the whole code of the analysis programs. The modularization of the code provides the flexibility to change small parts of the analysis software without modifying the rest of the code. It is furthermore very easy to change the reconstruction chain by simply adding, removing or exchanging single modules [59].

The *KM3Tray* framework basically consists of three components: *modules*, *services* and *frames*. A *frame* is a container for all possible kind of data that will be passed through the analysis chain. Hence one frame contains all information that describes a single event, including the physics, the calibration, etc., in the detector. Thus a data sample of a certain time period will contain a list of frames, of which each will be successively processed through the analysis chain. A *module* is a piece of algorithm that can get data out of the current frame, modify parts of the data and write new information into the frame. The sequence of modules defines the reconstruction chain. For example, the first module usually is a reader module that reads MC or detector data into the frame. The following modules can be for data calibration, triggers, event filters, specific reconstruction algorithms, etc... So if one intends to add a data filter at a certain step, this can easily be done by simply adding a filter module into the analysis

chain. It is also possible to write out parts of the data at any point, by including a module that creates output files. A *service* is an algorithm that is not a real part of the reconstruction chain, but can be called by a chain module and provides information. For example a random number generator could be used by multiple modules and thus will be implemented as a service the modules can access. An illustration of such an analysis chain is shown in Figure A.1.

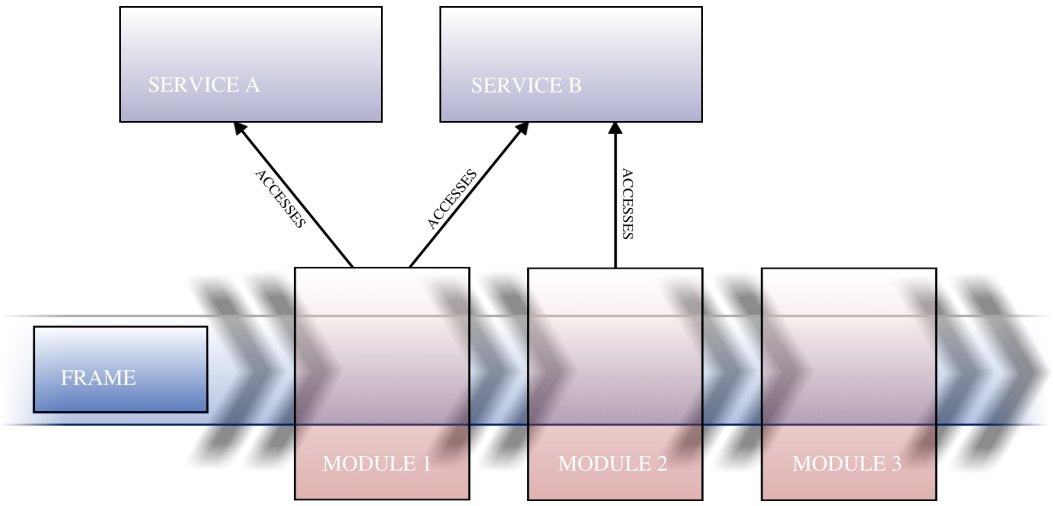

Figure A.1: *Illustration of the KM3Tray framework.* A physics frame is consecutively passed through the single modules that can modify the frame data or add new data to the frame. In this example *module 1* needs information from *service A* and *service B*, whereas *module 2* only needs *service B*. The last module works without any external service.

The configuration of the *KM3Tray* framework is done by a python [60] steering script, where the needed services and modules are added to a so called "tray". Each service or module can have specific steering parameters, by which it can be configured. Algorithm A.1 shows a part of such a python script, where some services and modules are included. To see a complete script, please see the final reconstruction chain scripts for hadronic showers, which are printed in appendix B.2.

To include a new module, one has to write a C++ class that is derived from the virtual base class *I3Module*. It has to consist of a method called *Configure()*, in which the module obtains the steering parameters from the python script. This method configures the module *before* processing the single frames. The method *Physics(I3FramePtr frame)* contains the code that modifies the data in each frame and needs a shared pointer (*I3FramePtr*) to the frame as argument. Finally a method called *Finish()* concludes the module *after* the last frame has been processed.

Services are derived from the base class *I3ServiceFactory* and contain an additional method called *InstallService(I3Context& services)* that installs the service into the

```
# CREATE A NEW TRAY
# ------------------------------------------------------------------------
        tray = I3Tray()
# ------------------------------------------------------------------------

# INCLUDE SERVICES
# ------------------------------------------------------------------------
        # A service to read in data
        tray.AddService("I3ReaderServiceFactory","imported_i3_file")(
                ("Filename", "RawData.i3")
                )

        # A random generator
        tray.AddService("I3SPRNGRandomServiceFactory","random")(
                ("Seed",       3425),
                ("NStreams",   3426),
                ("StreamNum", 1)
                )
# ------------------------------------------------------------------------

# INCLUDE MODULES
# ------------------------------------------------------------------------
        # Module to put the input data from the reader service in frames
        tray.AddModule("I3Muxer","muxer")(
                )

        # A module to add white noise to the event
        tray.AddModule("I3NoiseHitsAdder","addnoise")(
                ("WhiteNoiseRatePerOM",120.*I3Units.kilohertz),
                ("NoiseOffsetTime",500.*I3Units.ns),
                ("InputMCHits","RawHits"),
                ("OutputMCHits","RawHitsWithNoise")
                )

        # A module to write out the modified frame
        tray.AddModule("I3Writer","writer")(
                ("filename", "RawDataWithNoise.i3")
                )
# ------------------------------------------------------------------------
```

Algorithm A.1: *Example of a python steering script for the KM3Tray framework.* This script reads in raw data from the detector, adds hits from a simulation of background noise to each event (frame) and writes it out. The service *I3ReaderServiceFactory* reads in the file containing the raw data, here named *RawData.i3*. The module *I3Muxer* creates the physics frame by accessing the reader service. Afterwards white noise is added to the event by the module *I3NoiseHitsAdder* that needs random numbers provided by the service *I3SPRNGRandomServiceFactory*. *Seed, NStreams* and *StreamNum* are specific parameters for the random generator. The frame in this case has to contain a map of hits, called *RawHits*. The noise adding module gets these hits, adds white noise and writes a new hitmap including both, the original and the noise hits, as *RawHitsWithNoise* back into the frame. Now both hitmaps are contained in the frame. Finally the frame will be written out into a new data file *RawHitsWithNoise.i3*

framework by adding it to a parameter called *I3Context*. Each module can access this variable and thus get all provided services. Services do not have the methods *Physics* and *Finish*, as they do not act on frames.

Appendix B

I3ShAuerReco user's guide

This part gives an overview of how to use the *ShAuerReco* algorithm as an *KM3Tray* module. The detailed functionality of the algorithm is explained in chapter 3, so to understand the following explanations of the steering parameters and the example scripts, it is recommended to study this chapter before continuing here.

To reconstruct hadronic showers one simply has to add the module *I3ShAuerReco*, which is part of the *shauer-reco* project, to the python script. As input the module requires an *I3RecoPulseSeriesMap* that is a map of hits sorted by the corresponding OM. One also has to set the correct path to the look-up tables, that are stored in the "resources/files" directory of the project. The module is highly configurable by various steering parameters.

B.1 Steering parameters and their default values

The following parameters can be set for the *I3ShAuerReco* module. The expressions in brackets represent the default values, if existent.

Strategy *("ShAuerReco")*
Name of the data structure in the frame holding the reconstruction results. The prefit particle, the fitted particle and the *ShAuerRecoParams* will have this string as prefix occurring in their name.

FitType *(1)*
Defines the operation mode of the algorithm. Possible are four different types. *1:* No prefit is done. All seven observables will be fitted in the main fit. *2:* A prefit provides preliminary values for x, y, z and t that will be used as start values for the seven dimensional final fit. *3:* Identical to FitType 2, but the search space for x, y, z and t is reduced according to what is set in the parameter "Tolerance". *4:* The prefit provides final results for x, y, z and t. Only E, Θ and Φ will be fitted in the main fit.

Tolerance *([50, 50, 50, 50])*
Values by which the search space for x, y, z and t will be reduced when using FitType 3. The new search space will be adjusted around the fit results from the prefit by adding and subtracting the corresponding tolerance value from the fitted value.

StartTime *("Biggest")*
Sets the start value for the interaction time fit. Possible are: "First" (the first hit in the hitmap), "Biggest" (the hit with the biggest amplitude), "FirstTriggered" (the first hit with the flag *isTriggered*), "BiggestTriggered" (the biggest hit with the flag *isTriggered*) and "MC" (sets the startvalue to 0).

Center *("coa")*
Sets the reference position according to that the fit boundaries for the position fit are adjusted. Possible are: "cog" (= center of gravity. Takes the center of the detector as reference), "coa" (= center of amplitude. Calculates the center of the hits' amplitudes and takes this position as reference)

LowerBounds *([-300, -200, -200, -300, 1, 0, 0])*
Lower limit for the fit search space for $[t, x, y, z, E, \Theta, \Phi]$. The t bound is adjusted respective to the "StartTime". The position boundaries are adjusted respective to "Center".

UpperBounds *([300, 200, 200, 300, 1E7, 3.14, 6.28])*
Upper limit for the fit search space for $[t, x, y, z, E, \Theta, \Phi]$. The t bound is adjusted respective to the "StartTime". The position boundaries are adjusted respective to "Center".

Start *(0.5)*
Sets the start value for the (Pre-)fit for all observables, expect t, that is set by "StartTime". The parameter has to be a number in the interval $[0, 1]$. 0 means a start value at the "LowerBound", 1 is at the "UpperBound". Between these two limits, the start values are linearly interpolated.

Can *([55, 55, 55])*
Defines the three spatial components of the can around the detector. The search space for the fit will always be adjusted in a way that it does not exceed the can borders.

MAPName *()*
The name of the input hits for the reconstruction in the frame. This map has to be of the kind *I3RecoPulseSeriesMap*.

TIMEName *()*
The path and filename of the root file containing the look-up table for the timing

distributions.

AMPName *()*
The path and filename of the root file containing the look-up table for the amplitude distributions.

THist *()*
The name of the root histogram containing the timing distributions.

AHist *()*
The name of the root histogram containing the amplitude distributions.

AmpWeight *(1)* / **TimeWeight** *(1)*
Value by which the likelihoods are weighted when calculating the combined likelihood $= AmpWeight * amplikelihood + TimeWeight * timelikelihood$.

SpeedUp *(True)*
Speeds up the module by utilizing some features of the *Simulated Annealing* minimizer.

PreConvergence *([250, 1E-6, 0.85, 1E5, 20, 5, 0, -32, 6])*
Specific convergence parameters for the *Simulated Annealing* minimizer that will be used in the prefit. The nine parameters have to be set in a vector in the following order: Start temperature, convergence tolerance ϵ, temperature reduction factor r_T, maximum number of iterations, number of iteration in the s-cycle N_S, number of iteration in the t-cycle N_T, print level, seed for the random generator, number of convergence comparisons N_ϵ. See also chapter 3.2 for more information.

FitConvergence *([250, 1E-6, 0.85, 1E5, 20, 5, 0, -32, 6])*
Specific convergence parameters for the *Simulated Annealing* minimizer that will be used in the main fit. See "PreConvergence".

B.2 Steering python script for 5 line MC reconstruction

The following script is the steering python script that has been used for 5 line MC event reconstruction in chapter 5. The optimum parameters for the *I3ShAuerReco* module and the order of modules have been obtained throughout the studies in chapter 4. The script is to process 5 line events that come from 12 lines simulations. Therefore 7 strings have to be removed from the hitlist. Of course for the *Mupage* simulations and other MC files that are already 5 line files this cut can be left out.

```
1 #!/usr/bin/env python

3 from I3Tray import *

5 from os.path import expandvars
```

```
import os
import sys

import tasks.antares #includes preset scripts to execute the trigger

# LOAD THE PROJECTS' LIBRARIES
#--------------------------------------------------------

load("libicetray")
load("libdataclasses")
load("libinterfaces")
load("libphys-services")
load("libdataio")
load("libantares-reader")
load("libhit-selector")
load("libshauer-reco")
load("libicepick")
load("libroot-plots")
load("libantares-tools")

#--------------------------------------------------------

tray = I3Tray()

# DEFINE IN AND OUT FILES
infile = expandvars("folger_100k_E-2_1e2-1e7GeV_nc_120kHz.evt");
outfile  = expandvars("folger_100k.i3");
geofile  = expandvars("d10_c00_s00.det"); # The new mupage atmospheric muon simulation
     requires a different detectorfile! (l05_c09_s02.det)

# GET THE DETECTOR GEOMETRY FROM THE GEOFILE
tray.AddService("I3AntTextFileGeometryServiceFactory","geometryreader")(
     ("AntaresGeoFile", geofile),
     ("OMAngularParametrization","old") # For the mupage or other simulations, that have
          been done with the Genova angular acceptance, this has to be set to "Genova".
     )

# REDUCE THE DETECTOR TO A 5 LINE DETECTOR
tray.AddService("I3GeometrySelectorServiceFactory","line05detector")(
     ("StringsToUse","1:5"),
     ("StationsToUse","1:5"),
     ("GeoSelectorName","Line05Detector")
     )

# READ IN THE EVENT FILE
tray.AddService("I3AntTxtReaderServiceFactory","anttxtreader")(
     ("Filename", infile),
     ("NEvents", 999999999),
     ("MCSeriesName", "MCHits")
     )

# INSTALL EMPTY STREAMS FOR STATUS AND CALIBRATION
tray.AddService("I3EmptyStreamsFactory","empty_streams")(
     ("InstallGeometry",False),
     ("InstallEvent",False),
     ("InstallStatus",True),
     ("InstallCalibration",True),
     ("NFrames",999999999)
     )

# INSTALL A RANDOM NUMBER GENERATOR
seed = 2291;
tray.AddService("I3SPRNGRandomServiceFactory","random")(
     ("Seed",       seed),
     ("NStreams",   seed+1),
     ("StreamNum", 1)
     )

print("================================0")

# LINK ALL TOGETHER
tray.AddModule("I3Muxer","muxer")(
     ("GeometryService","Line05Detector")
     )

print("================================1")

# REMOVE ALL HITS THAT ARE OUTSIDE THE 5 LINE DETECTOR
tray.AddModule("I3HitSelectorModule<I3ExistingGeometryOnly>","indetector")(
     ("INmap", "MCHits"),
     ("OUTmap", "MCHitsLine5"),
     ("WriteDeselectedHits", False),
     ("NoFrameNamingPostfix", True)
     )

# ADD A BACKGROUND NOISE WITH A 120kHz RATE
```

```
tray.AddModule("I3NoiseHitsAdder","addnoise")(
    ("WhiteNoiseRatePerOM",120.*I3Units.kilohertz),
    ("NoiseOffsetTime",500.*I3Units.ns),
    ("RemoveOldNoise",True),
    ("InputMCHits","MCHitsLine5"),
    ("OutputMCHits","MCHitsWithNoiseLine5")
    )

# BUILD RECO PULSES BY SIMULATING THE PM PROPERTIES
ARSThresholdAmp = 0.3
ARSIntegrationTime = 40.*I3Units.ns
ARSDeadTime = 250.*I3Units.ns
tray.AddModule("I3AntPMSimulator","simpm")(
    ("InputMCHits","MCHitsWithNoiseLine5"),
    ("OutputRecoPulses","RawHitsAfterL0"),
    ("OutputHitRelationMap","RecoPulseToMCHitRelation"),
    ("NumARS",2),
    ("IntegrationTimeARS",ARSIntegrationTime),
    ("DeadTimeARS",ARSDeadTime),
    ("AmplitudeThresholdARS",ARSThresholdAmp),
    ("DoTrigEffTTS", False), # WARN: Do NOT use the distribution from "TriggerEfficiency"
        at this point!
    ("DoTrigEffGainRandomization", False)
    )

# CLIP HITS WITH AN AMPLITUDE ABOVE THE PM SATURATION
tray.AddModule("I3PulseHighAmpCutoff", "clip")(
    ("InputRecoPulses", "RawHitsAfterL0"),
    ("OutputRecoPulses", "RawHitsAfterClip"),
    ("HighAmplitudeCutoff", 20.)
    )

# THE ANTARES TRIGGER
tasks.antares.mc.addMakeL1Hits(tray, inputbox = 'RawHitsAfterClip', outputbox = '
    RecoPulseSeriesAfterARS_L1')
tasks.antares.mc.addTriggerSim(tray, L0_inputbox = 'RawHitsAfterClip', L1_inputbox = '
    RecoPulseSeriesAfterARS_L1', L0_outputbox='RecoPulseSeriesTriggeredHits', trigger=['3N
    '])

# SET THE HITS FLAG "ISTRIGGERED" FOR THE TRIGGERED HITS
tray.AddModule("I3SetTriggeredFlag", "triggeredflag")(
    ("RawHitsMap", "RawHitsAfterClip"),
    ("TriggeredHitsMap", "RecoPulseSeriesTriggeredHits")
    )

# CUT ALL HITS WITH A TOO LOW AMPLITUDE BELOW 3pe
tray.AddModule("I3HitSelectorModule<I3LowHighCutOff>","LowHigh_3_200")(
    ("SelectionName", "LowHigh_3_200"),
    ("INmap", "RawHitsAfterClip"),
    ("WriteDeselectedHits", False),
    ("NoFrameNamingPostfix", True),
    ("OUTmap", "Bigger_3"),
    ("MinNPE", 3),
    ("MaxNPE", 200)
    )

# THE HIT CONSTRAINTS FILTER
tray.AddModule("I3IcePickModule<I3MinimumConstraintsFilter>","MinHits")(
    ("DecisionName", "MinHits"),
    ("DiscardEvents", True),
    ("Map","Bigger_3"),
    ("MinHits",5),
    ("MinOMs",5),
    ("MinStrings",3),
    ("MinLCMs",3),
    ("MinMLCMs", 0)
    )

################################
# AND FINALLY THE SHAUERRECO #
################################

# SET THE PATHS TO THE LOOKUP TABLES
ampfile  = expandvars("$I3_SRC/shauer-reco/resources/files/cerenkov_3d_complete.root");
timefile = expandvars("$I3_SRC/shauer-reco/resources/files/time_normalized_t.root");

# THE SHAUER RECO...
tray.AddModule("I3ShAuerReco","showerreco")(
    ("Strategy","ShAuerReco"),
    ("MAPname","Bigger_3"),
    ("FitType",2),
    ("SpeedUp",True),
    ("LowerBounds",[-300,-130,-130,-180,1,0,0]),
    ("UpperBounds",[10,130,130,180,1E7,3.14,6.28]),
    ("Can", [55,55,55]),
    ("Start",0.5),
    ("StartTime", "FirstTriggered"),
    ("Tolerance",[50,50,50,50]),
```

```
        ("PreConvergence",[250, 1E-6, 0.90, 1E6, 20, 5, 0, -32, 6]),
177     ("FitConvergence",[250, 1E-6, 0.90, 1E6, 20, 5, 0, -32, 6]),
        ("AmpWeight",1),
179     ("TimeWeight",1),
        ("AMPname",ampfile),
181     ("AHist","normhist"),
        ("TIMEname",timefile),
183     ("THist","normhist_time"),
        ("Center", "coa"),
185     )

187 # WRITE EVERYTHING OUT TO THE OUTFILE
    tray.AddModule("I3Writer","writer")(
189     ("filename",outfile)
        )
191
    # FINISHING
193 tray.AddModule("TrashCan", "the can")

195 print("================================2")

197 tray.Execute()
    tray.Finish()
```

B.3 Steering python script for 12 line MC reconstruction

The following script is the steering python script that has been used for 12 line MC event reconstruction in chapter 5. The optimum parameters for the *I3ShAuerReco* module and the order of modules have been obtained throughout the studies in chapter 4.

```
    #!/usr/bin/env python
2
    from I3Tray import *
4
    from os.path import expandvars
6
    import os
8   import sys

10
    import tasks.antares #includes preset scripts to execute the trigger
12
    # LOAD THE PROJECTS' LIBRARIES
14  #--------------------------------------------------------

16  load("libicetray")
    load("libdataclasses")
18  load("libinterfaces")
    load("libphys-services")
20  load("libdataio")
    load("libantares-reader")
22  load("libhit-selector")
    load("libshauer-reco")
24  load("libicepick")
    load("libroot-plots")
26  load("libantares-tools")

28  #--------------------------------------------------------

30  tray = I3Tray()

32  # DEFINE IN AND OUT FILES
    infile = expandvars("folger_100k_E-2_1e2-1e7GeV_nc_120kHz.evt");
34  outfile  = expandvars("folger_100k.i3");
    geofile  = expandvars("d10_c00_s00.det");
36
    # GET THE DETECTOR GEOMETRY FROM THE GEOFILE
38  tray.AddService("I3AntTextFileGeometryServiceFactory","geometryreader")(
        ("AntaresGeoFile", geofile),
40      ("OMAngularParametrization","old") # For simulations, that have been done with the
            Genova angular acceptance, this has to be set to "Genova".
        )
42
    # READ IN THE EVENT FILE
44  tray.AddService("I3AntTxtReaderServiceFactory","anttxtreader")(
        ("Filename", infile),
```

```
46      ("NEvents", 999999999),
        ("MCSeriesName", "MCHits")
48      )

50  # INSTALL EMPTY STREAMS FOR STATUS AND CALIBRATION
    tray.AddService("I3EmptyStreamsFactory","empty_streams")(
52      ("InstallGeometry",False),
        ("InstallEvent",False),
54      ("InstallStatus",True),
        ("InstallCalibration",True),
56      ("NFrames",999999999)
        )
58
    # INSTALL A RANDOM NUMBER GENERATOR
60  seed = 2291;
    tray.AddService("I3SPRNGRandomServiceFactory","random")(
62      ("Seed",       seed),
        ("NStreams",   seed+1),
64      ("StreamNum", 1)
        )
66
    print("==============================0")
68
    # LINK ALL TOGETHER
70  tray.AddModule("I3Muxer","muxer")(
        )
72
    print("==============================1")
74
    # ADD A BACKGROUND NOISE WITH A 120kHz RATE
76  tray.AddModule("I3NoiseHitsAdder","addnoise")(
        ("WhiteNoiseRatePerOM",120.*I3Units.kilohertz),
78      ("NoiseOffsetTime",500.*I3Units.ns),
        ("RemoveOldNoise",True),
80      ("InputMCHits","MCHits"),
        ("OutputMCHits","MCHitsWithNoise")
82      )

84  # BUILD RECO PULSES BY SIMULATING THE PM PROPERTIES
    ARSThresholdAmp = 0.3
86  ARSIntegrationTime = 40.*I3Units.ns
    ARSDeadTime = 250.*I3Units.ns
88  tray.AddModule("I3AntPMSimulator","simpm")(
        ("InputMCHits","MCHitsWithNoise"),
90      ("OutputRecoPulses","RawHitsAfterL0"),
        ("OutputHitRelationMap","RecoPulseToMCHitRelation"),
92      ("NumARS",2),
        ("IntegrationTimeARS",ARSIntegrationTime),
94      ("DeadTimeARS",ARSDeadTime),
        ("AmplitudeThresholdARS",ARSThresholdAmp),
96      ("DoTrigEffTTS", False), # WARN: Do NOT use the distribution from "TriggerEfficiency"
            at this point!
        ("DoTrigEffGainRandomization", False)
98      )

100 # CLIP HITS WITH AN AMPLITUDE ABOVE THE PM SATURATION
    tray.AddModule("I3PulseHighAmpCutoff", "clip")(
102     ("InputRecoPulses", "RawHitsAfterL0"),
        ("OutputRecoPulses", "RawHitsAfterClip"),
104     ("HighAmplitudeCutoff", 20.)
        )
106
    # THE ANTARES TRIGGER
108 tasks.antares.mc.addMakeL1Hits(tray, inputbox = 'RawHitsAfterClip', outputbox = '
        RecoPulseSeriesAfterARS_L1')
    tasks.antares.mc.addTriggerSim(tray, L0_inputbox = 'RawHitsAfterClip', L1_inputbox = '
        RecoPulseSeriesAfterARS_L1', L0_outputbox='RecoPulseSeriesTriggeredHits', trigger=['3N
        '])
110
    # SET THE HITS FLAG "ISTRIGGERED" FOR THE TRIGGERED HITS
112 tray.AddModule("I3SetTriggeredFlag", "triggeredflag")(
        ("RawHitsMap", "RawHitsAfterClip"),
114     ("TriggeredHitsMap", "RecoPulseSeriesTriggeredHits")
        )
116
    # CUT ALL HITS WITH A TOO LOW AMPLITUDE BELOW 3pe
118 tray.AddModule("I3HitSelectorModule<I3LowHighCutOff>","LowHigh_3_200")(
        ("SelectionName", "LowHigh_3_200"),
120     ("INmap", "RawHitsAfterClip"),
        ("WriteDeselectedHits", False),
122     ("NoFrameNamingPostfix", True),
        ("OUTmap", "Bigger_3"),
124     ("MinNPE", 3),
        ("MaxNPE", 200)
126     )

128 # THE HIT CONSTRAINTS FILTER
    tray.AddModule("I3IcePickModule<I3MinimumConstraintsFilter>","MinHits")(
```

```
    ("DecisionName", "MinHits"),
    ("DiscardEvents", True),
    ("Map","Bigger_3"),
    ("MinHits",5),
    ("MinOMs",5),
    ("MinStrings",3),
    ("MinLCMs",3),
    ("MinMLCMs", 0)
    )

################################
# AND FINALLY THE SHAUERRECO #
################################

# SET THE PATHS TO THE LOOKUP TABLES
ampfile  = expandvars("$I3_SRC/shauer-reco/resources/files/cerenkov_3d_complete.root");
timefile = expandvars("$I3_SRC/shauer-reco/resources/files/time_normalized_t.root");

# THE SHAUER RECO...
tray.AddModule("I3ShAuerReco","showerreco")(
    ("Strategy","ShAuerReco"),
    ("MAPname","Bigger_3"),
    ("FitType",2),
    ("SpeedUp",True),
    ("LowerBounds",[-300,-130,-130,-180,1,0,0]),
    ("UpperBounds",[10,130,130,180,1E7,3.14,6.28]),
    ("Can", [55,55,55]),
    ("Start",0.5),
    ("StartTime", "FirstTriggered"),
    ("Tolerance",[50,50,50,50]),
    ("PreConvergence",[250, 1E-6, 0.90, 1E6, 20, 5, 0, -32, 6]),
    ("FitConvergence",[250, 1E-6, 0.90, 1E6, 20, 5, 0, -32, 6]),
    ("AmpWeight",1),
    ("TimeWeight",1),
    ("AMPname",ampfile),
    ("AHist","normhist"),
    ("TIMEname",timefile),
    ("THist","normhist_time"),
    ("Center", "coa"),
    )

# WRITE EVERYTHING OUT TO THE OUTFILE
tray.AddModule("I3Writer","writer")(
    ("filename",outfile)
    )

# FINISHING
tray.AddModule("TrashCan", "the can")

print("================================2")

tray.Execute()
tray.Finish()
```

B.4 Steering python script for 5 line real data reconstruction

The following script is the steering python script that has been used for 5 line real data reconstruction in chapter 6. The optimum parameters for the *I3ShAuerReco* module have been obtained throughout the studies in chapter 4.

```
#!/usr/bin/env python

from I3Tray import *

from os.path import expandvars

import os
import sys

import tasks.antares #includes preset scripts to execute the trigger

# LOAD THE PROJECTS' LIBRARIES
```

```
14 #--------------------------------------------------------------

16 load("libicetray")
   load("libdataclasses")
18 load("libinterfaces")
   load("libphys-services")
20 load("libdataio")
   load("libantares-reader")
22 load("libhit-selector")
   load("libshauer-reco")
24 load("libicepick")
   load("libroot-plots")
26 load("libantares-tools")

28 #--------------------------------------------------------------

30 tray = I3Tray()

32 # DEFINE IN AND OUT FILES
   infile = expandvars(".....hpss/data/raw/Line5/sea/2007/06/Antares_[RunNumber].root");
34 outfile  = expandvars("Antares_[RunNumber].i3");

36 # CONFIGURE THE CONNECTION TO THE ANTARES DATABASE
   db_user = .....
38 db_pass = .....
   oracle_connect_string = db_user + "/" + db_pass + "@(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(
       PROTOCOL=tcp)(HOST=ccdbcl01.in2p3.fr)(PORT=1521))(ADDRESS=(PROTOCOL␣=␣TCP)(HOST␣=␣
       cdbcl02.in2p3.fr)(PORT␣=␣1521))␣(LOAD_BALANCE=yes))(CONNECT_DATA␣=(SERVICE_NAME␣=␣
       ccora10g.in2p3.fr)))"
40
   # ACCESS THE ANTARES DATABASE
42 tray.AddService("I3OTLServiceFactory","otlservice")(
       ("ConnectString",oracle_connect_string))
44
   # SERVICE TO GET RUN INFORMATION FROM THE DATABASE
46 tray.AddService("I3AntDbRunInformationServiceFactory","dbruninfo")()

48 # SERVICE TO ASSIGN THE OM KEYS
   tray.AddService("I3AntaresDbOMKey2LCMFactory","omkey2lcm")
50
   # SERVICE TO GET THE CORRESPONDING GEOMETRY FROM THE DATABASE
52 tray.AddService("I3AntaresDbGeometryServiceFactory","geometry_from_db")(
       ("GeoServiceName",     "geometry_from_db"),
54     ("UseNewAngleFormat", True) # use the new heading/roll/pitch information
       )
56
   # SERVICE TO GET THE CALIBRATION
58 tray.AddService("I3AntaresDbCalibrationServiceFactory","calibration")(
       ("UseASingleCalibrationForAllTimes", False)
60     )

62 # READ IN THE ANTARES ROOT FILE
   tray.AddService("I3AntReaderServiceFactory","antreader")(
64     ("Filename",infile)
       )
66
   # INSTALL EMPTY STREAMS FOR STATUS
68 tray.AddService("I3EmptyStreamsFactory","empty_streams")(
       ("InstallGeometry",False),
70     ("InstallEvent",False),
       ("InstallStatus",True),
72     ("InstallCalibration",False)
       )
74
   print("================================0")
76
   # LINK ALL TOGETHER
78 tray.AddModule("I3Muxer","muxer")(
       ("GeometryService","geometry_from_db")
80     )

82 print("================================1")

84 # CALIBRATE THE RAW DATA
   tray.AddModule("I3AntSPEHitCalibrator", "spe-calib")(
86     ("InputSPEHits",       "AntSPESeriesMap"),
       ("OutputRecoPulses",   "CalibratedPulses")
88     )

90 # FIND MINIMUM BIAS EVENTS
   tray.AddModule("I3IcePickModule<I3TriggerSatisfiedFilter>", "find-minimum-bias")(
92     ("DiscardEvents",              False),
       ("DecisionName",               "IsMinimumBias"),
94     ("TriggerHierarchyName",       "AntTriggerHierarchy"),
       ("Type",                       ["ANT_TRIGGER_TRIGGER_MINIMUM_BIAS"]),
96     ("Source",                     ["ANTARES_DAQ"])
       )
98
```

```
# REMOVE MINIMUM BIAS EVENTS
tray.AddModule("I3IcePickModule<I3InverseFilter>", "no-minimum-bias")(
    ("DiscardEvents",           True),
    ("InputDecisionName",       "IsMinimumBias")
    )

# CUT ALL HITS WITH A TOO LOW AMPLITUDE BELOW 3pe
tray.AddModule("I3HitSelectorModule<I3LowHighCutOff>","LowHigh_3_200")(
    ("SelectionName", "LowHigh_3_200"),
    ("INmap", "CalibratedPulses"),
    ("WriteDeselectedHits", False),
    ("NoFrameNamingPostfix", True),
    ("OUTmap", "Bigger_3"),
    ("MinNPE", 3),
    ("MaxNPE", 200)
    )

# THE HIT CONSTRAINTS FILTER
tray.AddModule("I3IcePickModule<I3MinimumConstraintsFilter>","MinHits")(
    ("DecisionName", "MinHits"),
    ("DiscardEvents", True),
    ("Map","Bigger_3"),
    ("MinHits",5),
    ("MinOMs",5),
    ("MinStrings",3),
    ("MinLCMs",3),
    ("MinMLCMs", 0)
    )

################################
# AND FINALLY THE SHAUERRECO #
################################

# SET THE PATHS TO THE LOOKUP TABLES
ampfile  = expandvars("$I3_SRC/shauer-reco/resources/files/cerenkov_3d_complete.root");
timefile = expandvars("$I3_SRC/shauer-reco/resources/files/time_normalized_t.root");

# THE SHAUER RECO...
tray.AddModule("I3ShAuerReco","showerreco")(
    ("Strategy","ShAuerReco"),
    ("MAPname","Bigger_3"),
    ("FitType",2),
    ("SpeedUp",True),
    ("LowerBounds",[-300,-130,-130,-180,1,0,0]),
    ("UpperBounds",[10,130,130,180,1E7,3.14,6.28]),
    ("Can", [55,55,55]),
    ("Start",0.5),
    ("StartTime", "FirstTriggered"),
    ("Tolerance",[50,50,50,50]),
    ("PreConvergence",[250, 1E-6, 0.90, 1E6, 20, 5, 0, -32, 6]),
    ("FitConvergence",[250, 1E-6, 0.90, 1E6, 20, 5, 0, -32, 6]),
    ("AmpWeight",1),
    ("TimeWeight",1),
    ("AMPname",ampfile),
    ("AHist","normhist"),
    ("TIMEname",timefile),
    ("THist","normhist_time"),
    ("Center", "coa"),
    )

# WRITE EVERYTHING OUT TO THE OUTFILE
tray.AddModule("I3Writer","writer")(
    ("filename",outfile)
    )

# FINISHING
tray.AddModule("TrashCan", "the can")

print("================================2")

tray.Execute()
tray.Finish()
```

Appendix C

The *Root-plots* project user's guide

This abstract contains a description of the *Root-plots* project and a listing of all parameters and its default values that can be set for this project. First the internal implementation of the project is introduced in chapter C.1 followed by a manual how to use the project in C.2. At the end of this section a short manual how to add new subclasses is given in C.3.

C.1 Basic description of the internal implementation

The *Root-plots* project is embedded into the *KM3Tray* framework to create plots of data from various dataclasses stored in the frame at any possible step within the analysis chain. In the framework, the plot module can be linked dynamically into the python script at any position. The configuration via the steering parameters makes it easy to exchange the values to be plotted, or create a large number of plots with one script by adding the same module several times to the tray. Theses advantages were the motivation to implement such a plot module.

C.1.1 The *Root-plots* base classes

The kernel of the *Root-plots* project is a class template called *I3RootPlotsModule* that is the actual module to be included into the python script by adding the command

```
tray.AddModule("I3RootPlotsModule<[plotter],[data_collector]>",...
```

The included template module needs two classes as template arguments. The first one defines the kind of plot that should be created (e.g. a histogram or a graph), whereas the latter calls the suitable subclass for the data sourcing. The project currently contains two different **plotters**, which both are subclasses derived from a virtual base class

I3RootPlots. The subclass *I3RootHistoPlots* plots one-, two- and three dimensional histograms with different plot options and a configurable box for statistical values (like for example the median, meanvalue, etc...). *I3RootGraphPlots* creates one- and two dimensional graphs also with different plot options and optional error bars. After the processing of all frames the *Finish()* method of the main module calls the method *TObject* GetTObject()* that prompts the chosen *I3RootPlots* subclass to return the final plot and writes the plot into a root file.

Data collectors are subclasses that are derived from a virtual base class, called *I3DataSource*. These subclasses contain a method called *Source(I3FramePtr frame)*, which gets the data out of the frame and calculates values that will be sent back to the main module. The parameters that pass these values back are

$$x, y, z, w, x\_err, y\_err, z\_err \text{ and } w\_err \tag{C.1}$$

Of course these parameters have, regarded separately from the rest of the project, no meaning. They are just tools to carry the information from the data collecting part via the main module to the plot classes. It is the implementation of the *I3RootPlots* subclasses that define the meaning of each of these parameters. x, y and z are naturally used for the values along the three axes in the plot. The parameter w defines the function value in a graph, whereas in histograms it provides the weight of the entry. The parameters $*\_err$ contain the corresponding error values. Figure C.1 shows a graphical scheme of the dependencies between the single classes.

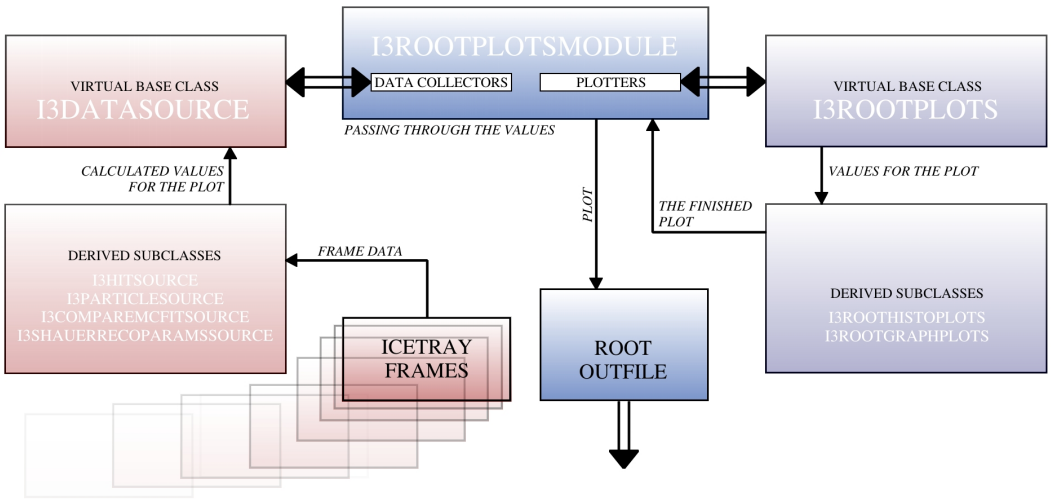

Figure C.1: *Illustration of the Root-plots project.* For further explanations see the text.

In addition to the parameters listed in (C.1) four string parameters can also be passed through to the plotters. These strings are used to provide default axis titles for the plots. A virtual method in the *I3DataSource* subclasses called *GetDefaultAxisTitles()*

sets the four string parameters

$$x\_title, y\_title, z\_title \text{ and } w\_title \tag{C.2}$$

The default axis titles usually contain the value type (for example *Fit_Energy*) and its dimension (for example GeV). Of course they can also be changed by the steering parameters of the plotters. The idea of default axis titles is to create meaningful plots with a minimum of configuring work.

By now four different *data collectors* are implemented. *I3ParticleSource* just collects data out of a single *I3Particle*, which is the *KM3Tray* class for a reconstructed track. This class was first implemented in the project, but as all possible data has to come from *one* particle it is not possible to make plots that compare fitted data with MC data. Thus the subclass *I3CompareMCFitSource* is much more important, as it is able to create combined plots from MC and fitted particles and furthermore calculates some additional values, like the quadrupole moment. *I3ShAuerRecoParamsSource* is able to plot specific values of the *ShAuerReco* algorithm (see chapter 3.1), like likelihood values and the number of used hits. This feature will only be installed to the *KM3Tray* framework if the project *shauer-reco* is already installed, as otherwise one will not have any use for it. *I3HitSeriesSource* is a data collector that handles hit maps and plots times or amplitudes of the first/biggest hit, noise hit fractions etc... As the latter two subclasses inherit from *I3CompareMCFitSource*, their value types can be combined with those of the mother class. For a detailed description of all steering parameters for the data collectors and plotters and how to use them, see chapter C.2.

The big advantage of this two-template-plot-module is that the actual plot creation is completely separated from the part that deals with getting the data. This makes it very easy to exchange one part without taking care of the other. One can simply plot a histogram and a graph with the same data, by just exchanging the plotter. Just as well can the data collector be replaced by another one. If it becomes necessary to implement a new data collector for some new kind of frame dataclasses (for how to do this, see chapter C.3), one can write the code without having to take care about how to get the data into the suitable histogram, as the already existing histogram plotter of course can be used for the new tool as well.

C.1.2 Additional services

In order to make the use of the project more comfortable, two additional services have been implemented that can be optionally included into the python script and to which the main module *I3RootPlotsModule* has access. The *Root-plots* project taken as a module will produce root files including the created plots. If one additionally wants the plots

to be printed out in the pdf file format[1], the service *I3RootPDFPlotterServiceFactory* will collect the single plots in the script and save them in the pdf file format. Thereby each plot can be chosen individually to appear in the pdf file or not by setting the value *Make_PDF_Printouts* to true or false. The steering parameters of the service finally configure the page layout of the file.

When producing plots in a complex python script with long computing times it might be useful to have a look at the plots before they are completely finished. The service *I3LivePrintoutsServiceFactory* provides the opportunity to print snapshots of the current status of the plots on screen and update them in regular time intervals. This feature can be used in any analysis script also as a monitoring of the script to avoid errors in the reconstruction. By setting the parameter *Make_Live_Printouts* to true or false, each plot can be individually arranged in the live plot monitor. The update rate is controlled by the parameter *Events_Per_Printout* that creates, if set to a value n, an update of the live plot after each n-th frame. Figure C.2 shows an example of how a live printout window may look like.

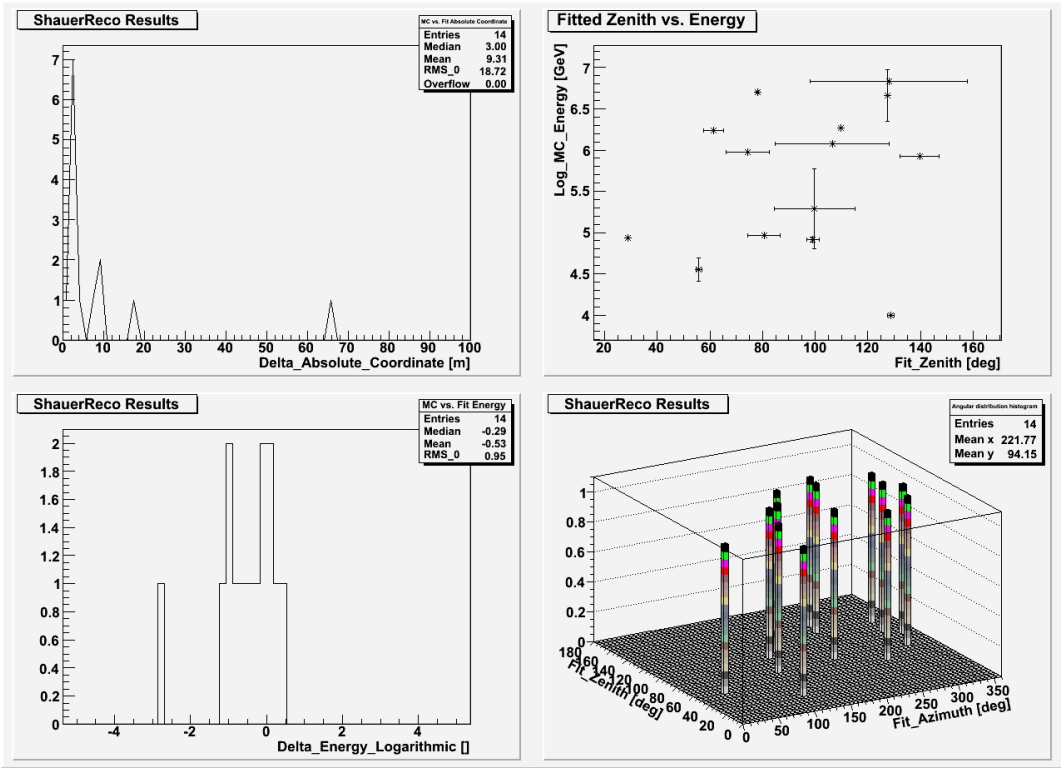

Figure C.2: *Example of a live printout.* NOTE: This figure is an example for the plot module and does not show real physics results!

The corresponding python script that creates these live plots is the following. The script

[1]Portable Document Format, provided by Adobe Systems Incorporated

creates two 1D histograms, a 2D histogram and two graphs, one with error bars and one without. The root file that is written out contains all five plots in the end, whereas only four of them will be seen as live snapshots, as for the last one *Make_Live_Printouts* was set to false.

```
#!/usr/bin/env python

from I3Tray import *

from os.path import expandvars

import os
import sys

# LOAD THE PROJECTS' LIBRARIES
#--------------------------------------------------------

load("libicetray")
load("libdataclasses")
load("libinterfaces")
load("libphys-services")
load("libdataio")
load("libroot-plots")

#--------------------------------------------------------

tray = I3Tray()

# DEFINE IN AND OUT FILES
infilename = expandvars("$I3_PORTS/test-data-antares/sim/example_root-plots_2.i3");
outfilename = expandvars("LivePrintoutsExample.root");


# READ IN THE FILE
tray.AddService("I3ReaderServiceFactory","imported_i3_file")(
    ("Filename",infilename)
    )


#####################################################################
# TO MAKE LIVE PRINTOUTS WE NEED THE I3LIVEPRINTOUTSSERVICE #
#####################################################################
tray.AddService("I3LivePrintoutsServiceFactory","liveprintout")(
    ("Number_Of_Plots_X",2),
    ("Number_Of_Plots_Y",2),
    ("Live_Window_Size_X",1280), #Pixels
    ("Live_Window_Size_Y",950),  #Pixels
    ("Events_Per_Printout", 1), #this means, the live window will be actualised on every
        frame.
    ("Speed_Up", False),
    ("Live_Window_Name", "Test-live-printout"),
    )


print("================================0")

# LINK ALL TOGETHER
tray.AddModule("I3Muxer","muxer")(
    )

print("================================1")


##############################
# NOW WE CREATE SOME PLOTS #
##############################

# A HISTOGRAM WITH LINES
tray.AddModule("I3RootPlotsModule<I3RootHistoPlots,I3CompareMCFitSource>","Plot1")(
    # Main parameters of the I3RootPlotsModule and the I3RootPlots and I3DataSource
        motherclasses
    ("Output_File", outfilename),
    ("Write_Option", "RECREATE"), # Creates a new root file
    ("Title", "MC vs. Fit Absolute Coordinate"),
    ("Name", "ShauerReco Results"),
    ("Make_Live_Printouts", True),
    # Specific parameters of I3RootHistoPlots
    ("Histo_Dimension", 1),
    ("Auto_Size",False),
    ("X_Min", 0),
    ("X_Max", 100),
    ("Draw_Option", "L"),
    ("Print_Statistics_Type", ["Entries","Mean","RMS_0","Median","Overflow"]),
```

```
    # Specific parameters of I3CompareMCFitSource
    ("X_Axis_Values", "Delta_Absolute_Coordinate"),
    ("Fit_Header", "ShAuerRecoShower"),
    ("MC_Header", "AntMCTree")
    )

# A GRAPH WITH ERROR BARS
tray.AddModule("I3RootPlotsModule<I3RootGraphPlots,I3CompareMCFitSource>","Plot2")(
    # Main parameters of the I3RootPlotsModule and the I3RootPlots and I3DataSource
        motherclasses
    ("Output_File", outfilename),
    ("Write_Option", "UPDATE"), # NOTE!! This has to be "Update". Otherwise the former
        plot in the rootfile will be overwritten
    ("Title", "Fitted Zenith vs. Energy"),
    ("Name", "ShauerReco Results"),
    ("Make_Live_Printouts", True),
    # Specific parameters of I3RootGraphPlots
    ("Graph_Dimension", 1),
    ("Auto_Size",True),
    ("Use_Error_Bars", True),
    # Specific parameters of I3CompareMCFitSource
    ("X_Axis_Values", "Fit_Zenith"),
    ("X_Error_Values", "Delta_Zenith"),
    ("Function_Values", "Log_MC_Energy"),
    ("Function_Error_Values", "Delta_Energy_Logarithmic"),
    ("Fit_Header", "ShAuerRecoShower"),
    ("MC_Header", "AntMCTree")
    )

# A STANDARD HISTOGRAM
tray.AddModule("I3RootPlotsModule<I3RootHistoPlots,I3CompareMCFitSource>","Plot3")(
    # Main parameters of the I3RootPlotsModule and the I3RootPlots and I3DataSource
        motherclasses
    ("Output_File", outfilename),
    ("Write_Option", "UPDATE"), # NOTE!! This has to be "Update". Otherwise the former
        plot in the root file will be overwritten
    ("Title", "MC vs. Fit Energy"),
    ("Name", "ShauerReco Results"),
    ("Make_Live_Printouts", True),
    # Specific parameters of I3RootHistoPlots
    ("Histo_Dimension", 1),
    ("Auto_Size",True),
    ("Print_Statistics_Type", ["Entries","Mean","RMS_0","Median"]),
    # Specific parameters of I3CompareMCFitSource
    ("X_Axis_Values", "Delta_Energy_Logarithmic"),
    ("Fit_Header", "ShAuerRecoShower"),
    ("MC_Header", "AntMCTree")
    )

# A COLORED 2D HISTOGRAM
tray.AddModule("I3RootPlotsModule<I3RootHistoPlots,I3CompareMCFitSource>","Plot4")(
    # Main parameters of the I3RootPlotsModule and the I3RootPlots and I3DataSource
        motherclasses
    ("Output_File", outfilename),
    ("Write_Option", "UPDATE"), # NOTE!! This has to be "Update". Otherwise the former
        plot in the root file will be overwritten
    ("Title", "Angular distribution histogram"),
    ("Name", "ShauerReco Results"),
    ("Make_Live_Printouts", True),
    # Specific parameters of I3RootHistoPlots
    ("Histo_Dimension", 2),
    ("Auto_Size",False),
    ("X_Min", 0),
    ("X_Max", 360),
    ("Y_Min", 0),
    ("Y_Max", 180),
    ("Draw_Option","LEGO2"),
    ("Print_Statistics_Type", ["Entries","Mean"]),
    # Specific parameters of I3CompareMCFitSource
    ("X_Axis_Values", "Fit_Azimuth"),
    ("Y_Axis_Values", "Fit_Zenith"),
    ("Fit_Header", "ShAuerRecoShower"),
    ("MC_Header", "AntMCTree")
    )

# ANOTHER GRAPH
tray.AddModule("I3RootPlotsModule<I3RootGraphPlots,I3CompareMCFitSource>","
    Angulardistgraph")(
    # Main parameters of the I3RootPlotsModule and the I3RootPlots and I3DataSource
        motherclasses
    ("Output_File", outfilename),
    ("Write_Option", "UPDATE"),
    ("Title", "Angular distribution graph"),
    ("Name", "ShauerReco Results"),
    ("Make_Live_Printouts", False), #we want this graph in the file, but NOT on the
        monitor as live printout
    # Specific parameters of I3RootGraphPlots
    ("Graph_Dimension", 1),
```

```
157     ("Auto_Size", True), # this option will ignore the Maxs and Mins now and find its own
        # Specific parameters of I3CompareMCFitSource
159     ("X_Axis_Values", "Fit_Zenith"),
        ("Function_Values", "Fit_Azimuth"),
161     ("Fit_Header", "ShAuerRecoShower"),
        ("MC_Header", "AntMCTree")
163     )

165 # tray.AddModule("Dump","dump")
    tray.AddModule("TrashCan", "the can")
167
    print("==============================2")
169
    tray.Execute()
171 tray.Finish()
```

C.2 Parameters and default values

The parameters signed with an ($\star$) are mandatory to set, as they have no default values.

C.2.1 *I3RootPlotsModule* (available for each plot)

C.2.1.1 Parameters of the main module

The parameters of the main module can always be set, no matter with which template arguments the module is called.

parameter	**description**	**default**
Output_File	Name of the root file	"standard_out.root"
Write_Option	Root write option for this file (see C.2.1.2)	"RECREATE"
Make_Live_Printouts	Prints a snapshot on screen during runtime[2] (see C.2.4.3)	False
Live_Printouts_Service_Name	Name of *I3LivePrintoutsService* in *I3Context*	"I3LivePrintoutsService"
Make_PDF_Printouts	Prints the plot in a pdf file[3] (see C.2.4.1)	False
PDF_Printouts_Service_Name	Name of *I3RootPDFPlotterService* in *I3Context*	"I3RootPDFPlotterService"
Name	TNamed::Name of the plot	" "
Title	TNamed::Title of the plot	" "
Set_Log_X	Creates a logarithmic x-axis[4]	False

[2] To use live printouts the *I3LivePrintoutsServiceFactory* has to be installed

[3] To use pdf printouts the *I3RootPDFPlotterServiceFactory* has to be installed

[4] As the logarithmic axis scale is an attribute of a TCanvas, this parameter will only have an effect in live printouts and pdf files, but **not** in usual root files!

Set_Log_Y	Creates a logarithmic y-axis[3]	False
Set_Log_Z	Creates a logarithmic z-axis[3]	False

C.2.1.2 Write options for the root file

The possible write options for a root file that can be set by the parameter *Write_Option* are:

write_option	**description**
"RECREATE"	Creates a new file. If the file already exists, it will be overwritten.
"CREATE"	Creates a new file and opens it for writing. If the file already exists, it will not be opened.
"UPDATE"	Opens an existing file for writing. If no file exists, it will be created.

NOTE: If it is intended to write out more than one plot in the same file, the *Write_Option* in every call, except for the first one, has to be *UPDATE*. Otherwise, the previous plots will be deleted.

C.2.2 The data collector subclasses

C.2.2.1 Parameters

The data collecting subclasses get the data out of the frame. *I3ParticleSource* can get all possible values, stored in a single *I3Particle* or the most energetic primary particle of an *I3Tree<I3Particle>* (for value types see C.2.2.2), whereas *I3CompareMCFitSource* gets combined data from an *I3Particle* that usually contains the fit results and the most energetic primary of the MC Tree (see C.2.2.3). *I3ShAuerRecoParamsSource* and *I3HitSeriesSource* are derived classes from *I3CompareMCFitSource*, thus both can get the same values as *I3CompareMCFitSource* and in addition the former can also read some specific values from the *ShAuerReco* algorithm (see C.2.2.4) and the latter calculates various values from *hit series maps* (see C.2.2.5). The following parameters can be set for all four subclasses:

parameter	**description**	**default**
X_Axis_Values	Values for the x-axis (see C.2.2.2 - C.2.2.5)	$\star$

Y_Axis_Values	Values for the y-axis (see C.2.2.2 - C.2.2.5)	" " (e.g. 0)
Z_Axis_Values	Values for the z-axis (see C.2.2.2 - C.2.2.5)	" " (e.g. 0)
Function_Values	Function values in graphs and weight in histograms (see C.2.2.2 - C.2.2.5)	" " (e.g. 1)
X_Axis_Units	Units for the x-axis values (see C.2.2.2 - C.2.2.5)	"m", "GeV", "deg", "m/s", "ns", "pe", "%", "kHz" or " " [5]
Y_Axis_Units	Units for the y-axis values (see C.2.2.2 - C.2.2.5)	"m", "GeV", "deg", "m/s", "ns", "pe", "%", "kHz" or " " [5]
Z_Axis_Units	Units for the z-axis values (see C.2.2.2 - C.2.2.5)	"m", "GeV", "deg", "m/s", "ns", "pe", "%", "kHz" or " " [5]
Function_Axis_Units	Units for the function values (see C.2.2.2 - C.2.2.5)	"m", "GeV", "deg", "m/s", "ns", "pe", "%", "kHz" or " " [5]
X_Error_Values	Error for the x-values (see C.2.2.2 - C.2.2.5)	" " (e.g. 0)
Y_Error_Values	Error for the y-values (see C.2.2.2 - C.2.2.5)	" " (e.g. 0)
Z_Error_Values	Error for the z-values (see C.2.2.2 - C.2.2.5)	" " (e.g. 0)
Function_Error_Values	Error for the function values (see C.2.2.2 - C.2.2.5)	" " (e.g. 0)

NOTE: In histograms the values for the two axes (assuming a 2D histogram) are represented by the values *X_Axis_Values* and *Y_Axis_Values*, whereas in graphs the last axis is always defined by the function values! A 1D graph thus needs the parameters *X_Axis_Values* and *Function_Values*.

Additional parameters only for *I3ParticleSource*:

I3Particle_Header	Name of the *I3Particle* or *I3Tree<I3Particle>* in the frame	$\star$

Additional parameters only for *I3CompareMCFitSource*, *I3ShAuerRecoParamsSource* and *I3HitSeriesSource*:

[5] depending on the chosen values for this axis

Fit_Header	Name of the fitted *I3Particle* in the frame	⋆
MC_Header	Name of the *I3MCTree* in the frame	⋆
MC_Particle_Type	Defines the MC particle. Possible are *Primary*, which is the neutrino, or *Muon*	"Primary"
Plot_Failed_Fits	if true plots all events, otherwise only those with *FitStatus::OK*	False
AntMCAdditional-Params_Header	Title of the *I3AntMCAdditionalParams* in the frame, that contains MC event weights [6]	"AntMCAdditional-Parameters"
Generated_Events	Total number of generated events for all MC files, that are included in the script.[6]	⋆ [6]
Irradiation_Time	Time to that the MC sample corresponds [6]	⋆ [6]
Reco_Pulse_Header	Title of the *I3RecoPulseSeriesMap* in the frame that contain the hits [7]	⋆ [7]
I3Double_Header	Title of the *I3Double* value in the frame, that is to be plotted [8]	⋆ [8]
I3Int_Header	Title of the *I3Int* value in the frame, that is to be plotted [9]	⋆ [9]

Additional parameters only for *I3ShAuerRecoParamsSource*:

ShAuerRecoParams_-Header	Name of the *I3ShAuerRecoParams* class in the frame	⋆

Additional parameters only for *I3HitSeriesSource*:

HitSeriesMap_Header	Name of the hit series map in the frame	⋆

[6] This parameter is needed only, if one value type is set to *Antares_MC_Weights*. This feature will not be installed to the KM3Tray framework, if the project *antares-reader* is not installed. To read more about the use of MC weights in *ANTARES* event files, please see [42].

[7] This parameter is needed only, if the quadrupole moment is to be plotted.

[8] This parameter is needed only, if one value type is set to *Double_Value*.

[9] This parameter is needed only, if one value type is set to *Int_Value*.

C.2.2.2 Value types for *I3ParticleSource*

I3ParticleSource gets values from an *I3Particle* that is either a single particle or the most energetic primary of an *I3Tree<I3Particle>*. The following values can be set to the axes or its errors:

value	description	possible units
"X-Coordinate"	Vertex x coordinate	"mm", "cm", "m", "km", "ft"
"Y-Coordinate"	Vertex y coordinate	"mm", "cm", "m", "km", "ft"
"Z-Coordinate"	Vertex z coordinate	"mm", "cm", "m", "km", "ft"
"Zenith"	Direction zenith angle	"deg", "rad"
"Theta"	Direction (Pi - zenith) angle	"deg", "rad"
"Azimuth"	Direction azimuth angle	"deg", "rad"
"Phi"	Direction (Pi + azimuth) angle	"deg", "rad"
"Energy"	Energy of the particle	"eV", "keV", "MeV", "GeV", "TeV", "PeV", "J"
"Speed"	Speed of the particle	"m/s", "km/h"
"Time"	Interaction time	"ps", "ns", "ms", "s", "min", "h", "d"
"Length"	Track length	"mm", "cm", "m", "km", "ft"
"Starttime"	Starttime of event	"ps", "ns", "ms", "s", "min", "h", "d"
"Stoptime"	Endtime of event	"ps", "ns", "ms", "s", "min", "h", "d"
"Bjorken-x"	Bjorken x value	" " (no units)
"Bjorken-y"	Bjorken y value	" " (no units)
"Type"	Particle type	" " (no units)
"Shape"	Particle shape	" " (no units)
"Fit-Status"	Status of the fit	" " (no units)

C.2.2.3 Value types for *I3CompareMCFitSource*

I3CompareMCFitSource gets combined data from an *I3Particle* that contains the fit results and the most energetic primary of an *I3Tree<I3Particle>* that is the MC particle. The following values can be set to the axes or its errors:

value	description	possible units
"MC_X-Coordinate"	Monte carlo vertex x coordinate	"mm", "cm", "m", "km", "ft"
"MC_Y-Coordinate"	Monte carlo vertex y coordinate	"mm", "cm", "m", "km", "ft"
"MC_Z-Coordinate"	Monte carlo vertex z coordinate	"mm", "cm", "m", "km", "ft"
"MC_Vertical_Distance_-To_Detector_Center"	Vertical distance of the MC vertex to the detector center	"mm", "cm", "m", "km", "ft"
"MC_Horizontal_Distance_-To_Detector_Center"	Horizontal distance of the MC vertex to the detector center	"mm", "cm", "m", "km", "ft"
"MC_Absolute_Distance_-To_Detector_Center"	Absolute distance of the MC vertex to the detector center	"mm", "cm", "m", "km", "ft"
"MC_Zenith"	Monte carlo direction zenith angle	"deg", "rad"
"MC_Theta"	Monte carlo direction (Pi - zenith) angle	"deg", "rad"
"MC_Azimuth"	Monte carlo direction azimuth angle	"deg", "rad"
"MC_Phi"	Monte carlo direction (Pi + azimuth) angle	"deg", "rad"
"MC_Time"	Monte carlo interaction time	"ps", "ns", "ms", "s", "min", "h", "d"
"MC_Energy"	Monte carlo energy of the particle	"eV", "keV", "MeV", "GeV", "TeV", "PeV", "J"
"Log_MC_Energy"	Log Monte carlo energy of the particle [10]	"eV", "keV", "MeV", "GeV", "TeV", "PeV", "J"
"MC_Shower_Energy"	Monte carlo shower energy (= Energy · Bjorken_y)	"eV", "keV", "MeV", "GeV", "TeV", "PeV", "J"
"Log_MC_Shower_-Energy"	Log Monte carlo shower energy [10]	"eV", "keV", "MeV", "GeV", "TeV", "PeV", "J"
"MC_Quadrupole_-Moment"	Amplitude quadrupole moment along the MC track [11]	" " (no units)
"Fit_X-Coordinate"	Fitted vertex x coordinate	"mm", "cm", "m", "km", "ft"

[10] Useful when using histograms as the use of *Set_Log_X* results in a weird binning

[11] The calculation of the quadrupole moment requires the single hits, that are to be set by the parameter *Reco_Pulse_Header*

"Fit_Y-Coordinate"	Fitted vertex y coordinate	"mm", "cm", "m", "km", "ft"
"Fit_Z-Coordinate"	Fitted vertex z coordinate	"mm", "cm", "m", "km", "ft"
"Fit_Vertical_Distance_-To_Detector_Center"	Vertical distance of the fitted vertex to the detector center	"mm", "cm", "m", "km", "ft"
"Fit_Horizontal_Distance_-To_Detector_Center"	Horizontal distance of the fitted vertex to the detector center	"mm", "cm", "m", "km", "ft"
"Fit_Absolute_Distance_-To_Detector_Center"	Absolute distance of the fitted vertex to the detector center	"mm", "cm", "m", "km", "ft"
"Fit_Zenith"	Fitted direction zenith angle	"deg", "rad"
"Fit_Theta"	Fitted direction (Pi - zenith) angle	"deg", "rad"
"Fit_Azimuth"	Fitted direction azimuth angle	"deg", "rad"
"Fit_Phi"	Fitted direction (Pi + azimuth) angle	"deg", "rad"
"Fit_Time"	Fitted interaction time	"ps", "ns", "ms", "s", "min", "h", "d"
"Fit_Energy"	Fitted energy of the particle	"eV", "keV", "MeV", "GeV", "TeV", "PeV", "J"
"Log_Fit_Energy"	Log Fitted energy of the particle [10]	"eV", "keV", "MeV", "GeV", "TeV", "PeV", "J"
"Fit_Quadrupole_-Moment"	Amplitude quadrupole moment along the fitted track [11]	" " (no units)
"Delta_X-Coordinate"	Difference (Fit - MC) of vertex x coordinate	"mm", "cm", "m", "km", "ft"
"Delta_Y-Coordinate"	Difference (Fit - MC) of vertex y coordinate	"mm", "cm", "m", "km", "ft"
"Delta_Z-Coordinate"	Difference (Fit - MC) of vertex z coordinate	"mm", "cm", "m", "km", "ft"
"Delta_Absolute_-Coordinate"	Absolute difference (Fit - MC) of vertex	"mm", "cm", "m", "km", "ft"
"Delta_Zenith"	Difference (Fit - MC) of direction zenith angle	"deg", "rad"
"Delta_Theta"	Difference (Fit - MC) of direction zenith angle	"deg", "rad"
"Delta_Azimuth"	Difference (Fit - MC) of direction azimuth angle	"deg", "rad"
"Delta_Phi"	Difference (Fit - MC) of direction azimuth angle	"deg", "rad"

"Delta_Absolute_-Angular"	Absolute difference (Fit - MC) of direction angle	"deg", "rad"
"Delta_Time"	Difference (Fit - MC) of interaction time	"ps", "ns", "ms", "s", "min", "h", "d"
"Delta_Energy"	Difference (Fit - MC) of energy of the particle	"eV", "keV", "MeV", "GeV", "TeV", "PeV", "J"
"Delta_Energy_-Logarithmic"	Logarithmic difference of energy log(Fit/MC)	"eV", "keV", "MeV", "GeV", "TeV", "PeV", "J"
"Delta_Shower_Energy"	Difference (Fit - MC) of shower energy [12]	"eV", "keV", "MeV", "GeV", "TeV", "PeV", "J"
"Delta_Shower_Energy_-Logarithmic"	Logarithmic difference of shower energy log(Fit/MC) [12]	"eV", "keV", "MeV", "GeV", "TeV", "PeV", "J"
"Delta_Quadrupole_-Moment"	Difference (Fit - MC) of amplitude quadrupolemoment [11]	" " (no units)
"Delta_Coordinate_-Distance_Track"	Closest approach distance between MC and fitted track	"mm", "cm", "m", "km", "ft"
"Delta_Length"	Difference (Fit - MC) of track length	"mm", "cm", "m", "km", "ft"
"Antares_MC_Weights"	Weight value for *ANTARES* MC events [13]	" " (no units)
"Double_Value"	Optional *I3Double_Value* from the frame (for example the *Aart lambda parameter* [61]) [14]	any possible unit [15]
"Int_Value"	Optional *I3Int_Value* from the frame (for example the muon multiplicity) [16]	any possible unit [15]

[12] compares the fitted energy to the MC shower energy

[13] The use of this value type requires the steering parameters *Generated_Events* and *Irradiation_Time* to be set. As in histograms the *Function_Values* are used for the entries' weights, usually this value type is only used for *Function_Values*. This feature will not be installed to the KM3Tray framework, if the project *antares-reader* is not installed. To read more about the use of MC weights in *ANTARES* event files, please see [42].

[14] The name of the *I3Double* in the frame has to be set by the parameter *I3Double_Header*.

[15] As the module doesn't know the correct unit of this value, any possible unit can be chosen. In this case, the module assumes, that the user knows what he is doing.

[16] The name of the *I3Int* in the frame has to be set by the parameter *I3Int_Header*.

C.2.2.4 Value types for *I3ShAuerRecoParamsSource*

I3ShAuerRecoParamsSource[17] gets values from the *I3ShAuerRecoParams* container that contains specific values of the *ShAuerReco* algorithm. As this class inherits from *I3CompareMCFitSource*, in addition all values from *I3CompareMCFitSource* can be read out. The following values can be set in additional to the axes or its errors.

value	description	possible units
"Time_Residuum"	The final time likelihood	" " (no units)
"Amplitude_Residuum"	The final amp likelihood	" " (no units)
"Pre_Likelihood"	Overall log likelihood value of the prefit	" " (no units)
"Likelihood"	Overall log likelihood value of the fit	" " (no units)
"Sep_Hits"	The number of single hits used for the fit	" " (no units)
"Integrated_Hits"	The number of integrated hits used for the fit	" " (no units)

C.2.2.5 Value types for *I3HitSeriesSource*

I3HitSeriesSource can calculate various values from a set of hits. These hits have to be stored in a *hit series map* that is a map of pairs, whereof the first (*the key value*) is the number of the OM and the second is the list of hits at this OM. In the *KM3Tray* framework there are three different kinds of hit series maps: The *I3RecoPulseSeriesMap*, the *I3RecoHitSeriesMap* and the *I3MCHitSeriesMap*. Depending on what kind of map is read in by the parameter *HitSeriesMap_Header*, different value types are allowed to be set. The calculation of this values is done by functions of the namespace *I3SeriesMapUtils* that are part of the project *phys-services*. So consider that this feature will not work, if the namespace is not installed! As this class inherits from *I3CompareMCFitSource*, in addition all values from *I3CompareMCFitSource* can be read out.

The following value types can be set for an *I3RecoPulseSeriesMap*:

value	description	possible units
"Time_Of_Earliest_Hit"	Absolute time, when the first hit appears	"ps", "ns", "ms", "s", "min", "h", "d"

[17] This feature will only be installed to the KM3Tray framework, if also the project *shauer-reco* is installed.

"Time_Of_Latest_Hit"	Absolute time, when the last hit appears	"ps", "ns", "ms", "s", "min", "h", "d"
"Time_Of_Biggest_Hit"	Absolute time, when the biggest hit appears	"ps", "ns", "ms", "s", "min", "h", "d"
"Total_Number_Of_Hits"	Total number of hits in the event	" " (no units)
"Amplitude_Of_Earliest_Hit"	Charge of the first hit	"pe"[18], "C"
"Amplitude_Of_Latest_Hit"	Charge of the last hit	"pe"[18], "C"
"Amplitude_Of_Biggest_Hit"	Charge of the biggest hit	"pe"[18], "C"
"Integrated_Amplitude"	Summation over the charge of all hits	"pe"[18], "C"
"Average_Amplitude"	Average charge of all hits	"pe"[18], "C"
"Time_Of_Earliest_Triggered_Hit"	Absolute time, when the first triggered hit appears [19]	"ps", "ns", "ms", "s", "min", "h", "d"
"Time_Of_Latest_Triggered_Hit"	Absolute time, when the last triggered hit appears [19]	"ps", "ns", "ms", "s", "min", "h", "d"
"Time_Of_Biggest_Triggered_Hit"	Absolute time, when the biggest triggered hit appears [19]	"ps", "ns", "ms", "s", "min", "h", "d"
"Total_Number_Of_Triggered_Hits"	Total number of triggered hits in the event [19]	" " (no units)
"Amplitude_Of_Earliest_Triggered_Hit"	Charge of the first triggered hit [19]	"pe"[18], "C"
"Amplitude_Of_Lastest_Triggered_Hit"	Charge of the last triggered hit [19]	"pe"[18], "C"
"Amplitude_Of_Biggest_Triggered_Hit"	Charge of the biggest triggered hit [19]	"pe"[18], "C"
"Triggered_Integrated_Amplitude"	Summation over the charge of all triggered hits [19]	"pe"[18], "C"
"Triggered_Average_Amplitude"	Average charge of all triggered hits [19]	"pe"[18], "C"
"Fraction_Of_Triggered_Hits"	Number of triggered hits / Number of all hits [19]	"%". " " (i.e. 0,...)
"Total_Number_Of_Hit_Strings"	Number of all hit strings by the hit series	" " (no units)
"Fit_Time_Difference_To_First_Triggered_Hit"	Time difference between the first triggered hit and the fitted time [20]	"ps", "ns", "ms", "s", "min", "h", "d"

[18] photo electrons

[19] This value type requires the previously correct assignment of the hit's attribute *isTriggered_*.

[20] This value type expects a fitted particle set by the parameter *Fit_Header* from *I3CompareMCFitSource*.

"MC_Time_Difference_-To_First_Triggered_Hit"	Time difference between the first triggered hit and the MC time [21]	"ps", "ns", "ms", "s", "min", "h", "d"
"Fit_Vertex_Distance_To_-Center_Of_Gravity"	Distance between the fitted vertex and the hits' center of amplitude [20]	"mm", "cm", "m", "km", "ft"
"MC_Vertex_Distance_To_-Center_Of_Gravity"	Distance between the MC vertex and the hits' center of amplitude [21]	"mm", "cm", "m", "km", "ft"

Value types that can be set for an *I3MCHitSeriesMap*:

value	**description**	**possible units**
"Time_Of_Earliest_Hit"	Absolute time, when the first hit appears	"ps", "ns", "ms", "s", "min", "h", "d"
"Time_Of_Latest_Hit"	Absolute time, when the last hit appears	"ps", "ns", "ms", "s", "min", "h", "d"
"Time_Of_Earliest_Noise_-Hit"	Absolute time, when the first noise hit appears [22]	"ps", "ns", "ms", "s", "min", "h", "d"
"Time_Of_Latest_Noise_-Hit"	Absolute time, when the last noise hit appears [22]	"ps", "ns", "ms", "s", "min", "h", "d"
"Total_Number_Of_Hits"	Total number of hits in the event	" " (no units)
"Total_Number_Of_Noise_-Hits"	Total number of noise hits in the event	" " (no units)
"Fraction_Of_Noise_Hits"	Number of noise hits / Number of all hits	"%". " " (i.e. 0,...)
"Estimated_Noise_Rate"	Calculates an estimated noise rate per OM by considering all noise hits in the event	"Hz", "kHz", "MHz"
"Total_Number_Of_Hit_-Strings"	Number of all hit strings by the hit series	" " (no units)

Value types that can be set for an *I3RecoHitSeriesMap*:

value	**description**	**possible units**
"Time_Of_Earliest_Hit"	Absolute time, when the first hit appears	"ps", "ns", "ms", "s", "min", "h", "d"
"Time_Of_Latest_Hit"	Absolute time, when the last hit appears	"ps", "ns", "ms", "s", "min", "h", "d"

[21] This value type expects a MC particle tree set by the parameter *MC_Header* from *I3CompareMCFitSource*.

[22] This value type requires the previously correct assignment of the hit's attribute *dueToEnvironmentalNoise_*.

"Total_Number_Of_Hits"	Total number of hits in the event	" " (no units)
"Time_Of_Earliest_-Triggered_Hit"	Absolute time, when the first triggered hit appears [23]	"ps", "ns", "ms", "s", "min", "h", "d"
"Time_Of_Latest_-Triggered_Hit"	Absolute time, when the last triggered hit appears [23]	"ps", "ns", "ms", "s", "min", "h", "d"
"Total_Number_Of_-Triggered_Hits"	Total number of triggered hits in the event [23]	" " (no units)
"Fraction_Of_Triggered_-Hits"	Number of triggered hits / Number of all hits [23]	"%". " " (i.e. 0,...)
"Fit_Time_Difference_-To_First_Triggered_Hit"	Time difference between the first triggered hit and the fitted time [24]	"ps", "ns", "ms", "s", "min", "h", "d"
"MC_Time_Difference_-To_First_Triggered_Hit"	Time difference between the first triggered hit and the MC time [25]	"ps", "ns", "ms", "s", "min", "h", "d"
"Fit_Vertex_Distance_To_-Center_Of_Gravity"	Distance between the fitted vertex and the hits' center of amplitude [24]	"mm", "cm", "m", "km", "ft"
"MC_Vertex_Distance_To_-Center_Of_Gravity"	Distance between the MC vertex and the hits' center of amplitude [25]	"mm", "cm", "m", "km", "ft"
"Total_Number_Of_Hit_-Strings"	Number of all hit strings by the hit series	" " (no units)

C.2.3 The plotter subclasses

C.2.3.1 Parameters for *I3RootHistoPlots*

The *I3RootHistoPlots* class creates one-, two- or three dimensional histograms that can be configured by the following parameters:

parameter	**description**	**default**
Histo_Dimension	Dimension of the histogram	1
X_Number_Of_Bins	Number of bins along the x-axis	60
Y_Number_Of_Bins	Number of bins along the y-axis	60
Z_Number_Of_Bins	Number of bins along the z-axis	60
X_Min	Minimum along the x-axis	0
X_Max	Maximum along the x-axis	100

[23] This value type requires the previously correct assignment of the hit's attribute *isTriggered_*.

[24] This value type expects a fitted particle set by the parameter *Fit_Header* from *I3CompareMCFitSource*.

[25] This value type expects a MC particle tree set by the parameter *MC_Header* from *I3CompareMCFitSource*.

Y_Min	Minimum along the y-axis	0
Y_Max	Maximum along the y-axis	100
Z_Min	Minimum along the z-axis	0
Z_Max	Maximum along the z-axis	100
X_Axis_Title	Title of the x-axis in the plot	(default string provided by the data collectors)
Y_Axis_Title	Title of the y-axis in the plot	(default string provided by the data collectors)
Z_Axis_Title	Title of the z-axis in the plot	(default string provided by the data collectors)
Draw_Option	Root draw option for the histogram	" " (for 1D), "LEGO" (for 2D & 3D)
Auto_Size	Finds suitable maxs and mins, that all values are in the plot	False
Print_Stats_Box	Prints the statistics box in the plot	True
Print_Statistics_Type	Sets what kind of entries appear in the stats box (see C.2.3.2)	["Default"] (e.g. the default root stats box)

C.2.3.2 Entries for the statistics box in histograms

The parameter *Print_Statistics_Type* defines, which statistical values appear in the stats box of the histogram. The entries that are to appear must be set in a vector of strings, for example ["Mean", "Overflow", "Underflow"]. If this parameter is set to ["Default"], the standard stats box of root will be created. If no stats box is wished, just set the parameter to [" "].

The following entries can be added to the stats box.

statistical value	**description**
"Entries"	Total number of entries in the histogram (incl. under- & overflow, but without NaN-values)
"Integral"	(Weighted) number of entries in the histogram (i.e. the sum over all bin contents)
"Mean"	Mean value of the (weighted) entries within the plot limits
"Median"	Value that divides the (weighted) number of all entries within the plot limits in to two equal parts

"RMS"	Root-Mean-Square of the (weighted) entries within the plot limits ($= \sqrt{\bar{x^2} - \bar{x}^2}$)
"RMS_0"	($= \sqrt{\bar{x^2}}$) i.e. this calculates a Root-Mean-Square where the mean value is forced to 0. This could be useful if one wants to plot errors, where the mean is expected to be 0 but it is not really. So *RMS_0* would give a more propriate resolution value than *RMS* does.
"Overflow"	(Weighted) number of entries outside the plot above the upper limit
"Underflow"	(Weighted) number of entries outside the plot below the lower limit
"NaN"	Total number of NaN-values. If a value is NaN, it won't be added to the histogram.

C.2.3.3 Parameters for *I3RootGraphPlots*

The *I3RootGraphPlots* class creates one- or two dimensional graphs that can be configured by the following parameters

parameter	**description**	**default**
Graph_Dimension	Dimension of the graph	1
X_Min	Minimum along the x-axis	0
X_Max	Maximum along the x-axis	100
Y_Min	Minimum along the y-axis	0
Y_Max	Maximum along the y-axis	100
Value_Min	Minimum along the value-axis	0
Value_Max	Maximum along the value-axis	100
X_Axis_Title	Title of the x-axis in the plot	(default string provided by the data collectors)
Y_Axis_Title	Title of the y-axis in the plot	(default string provided by the data collectors)
Value_Axis_Title	Title of the value-axis in the plot	(default string provided by the data collectors)
Draw_Option	Root draw option for the graph	"A*" (for 1D), "P0" (for 2D)
Auto_Size	Finds suitable maxs and mins, that all values are in the plot	False

Use_Error_Bars	Creates a plot with error bars	False

C.2.4 Additional services

C.2.4.1 Parameters for *I3RootPDFPlotterServiceFactory*

To print out a pdf file with the created root plots, add the service called *I3RootPDFPlotterServiceFactory* to the script. By setting the parameter *Make_PDF_Printouts* to true, it can be decided for each plot individually, whether it appears in the pdf file or not. The PDF plotter service can be configured by the following parameters:

parameter	**description**	**default**
Install_Service_As	Name of this service in the *I3Context*	"I3RootPDFPlotter-Service"
Output_File	Name of the pdf file	"standard_out.pdf"
Number_Of_Plots_X	Number of plots along the x-axis of the paper	1
Number_Of_Plots_Y	Number of plots along the y-axis of the paper	2
Paper_Orientation	*Landscape* or *Portrait* paper	"Portrait"

C.2.4.2 The standalone program *root-plots-PDFPlotter*

If a root file is already filled with plots, it can easily be converted into a pdf file without calling the whole *KM3Tray* framework again. Therefore call the program *root-plots-PDFPlotter* out of the shell. This program can be called with the following parameters:

```
root-plots-PDFPlotter [RootFile] [PdfFile] [Orientation] [PlotsInX] [PlotsInY]
```

parameter	**description**	**default**
RootFile	Path & Filename of the root-file	$\star$
PdfFile	Path & Filename of the pdf-file	"standard_out.pdf"
Orientation	*Landscape* or *Portrait* paper	"Portrait"
PlotsInX	Number of plots along the x-axis of the paper	1
PlotsInY	Number of plots along the y-axis of the paper	2

C.2.4.3 Parameters for *I3LivePrintoutsServiceFactory*

To get live snapshots of the plots during the runtime of the script, add the service *I3LivePrintoutsServiceFactory*. By setting the parameter *Make_Live_Printouts* to true, it can be decided for each plot individually, whether a live printout should be printed out or not. The live printouts service can be configured by the following parameters:

parameter	**description**	**default**
Install_Service_As	Name of this service in the *I3Context*	"I3LivePrintouts-Service"
Number_Of_Plots_X	Number of plots along the x-axis of the monitor	2
Number_Of_Plots_Y	Number of plots along the y-axis of the monitor	2
Live_Window_Size_X	Size in Pixels along the x-axis	1280
Live_Window_Size_Y	Size in Pixels along the y-axis	950
Events_Per_Printout	Number of frames, after which the live windows will be updated	1
Speed_Up	Speeds up scripts with a low live plot update rate [26]	false
Live_Window_Name	Name of the live canvas	"Root-plots Live Printout Monitor"

NOTE: For 2D graphs the live printouts unfortunately don't work correctly. So if live plots of 2D graphs are printed out, one will not see the axis titles and the graph will be printed with auto sized axis limits. However, in the final pdf or root files, the graphs are printed correctly with all attributes.

C.3 How to write a new subclass

C.3.1 New data collectors

As the different storable dataclasses in *KM3Tray* frames grow constantly in number, it may become necessary to write new data collecting subclasses to create plots from new value types. Furthermore there are already a lot of classes that can not be read in by now. For example, a certain reconstruction project may write out specific values into the frame that are stored in a class specifically created for this reconstruction

[26] *Speed_Up* may cause an empty white canvas, when the live window was hidden behind other windows on the desktop. In this case one has to wait until the next update to see the plots again.

algorithm. In order to plot these values with the *Root-plots* project it is essential to write a new data collector that suits to them. (According to the *I3ShAuerRecoParams* class that is a specific container for the *ShAuerReco* and for that such a subclass is already implemented).

The *Root-plots* project is designed to collect and plot **one** value and **one** optional error value per frame and plot dimension. What the project cannot do is to create a whole plot for **each** frame. For example, plots that contain the hit amplitude distribution of the hits in one event cannot be done here, as they contain more than one value per frame, viz. one for each hit. Only plots of the whole script (as a summary of all frames) can be created.

Adding a new data collector is very simple. One has to write a class that inherits from *I3DataSource* and sets the values that are to be plotted. The values and its errors that can be taken out of the frame or calculated elsewise are

$$x, y, z, w, x\_err, y\_err, z\_err \text{ and } w\_err \tag{C.3}$$

In histograms, x, y and z and their error values are used for the three axes and w is used for the weight of one entry, whereas w_err as an "error value" for the weight has of course no meaning in histogram plots. In graphs, on the contrary, w and its error are always used for the function value. x and y represent the remaining axes, where in graph plots z and z_err have no meaning. In principle all that has to be done is to write a class that sets these eight values. In addition, there is the option to set also four string values

$$x\_title, y\_title, z\_title \text{ and } w\_title \tag{C.4}$$

that contain default axis titles for the plotters. Algorithm C.1 shows a blueprint for the implementation of a data collector.

C.3.2 New plotters

The procedure to include a new plotter is almost the same as that for data collectors. The values and error values (C.3) and the default strings for the axis titles (C.4) that have been set by one of the data collectors arrive at the plotting part of the module and all a plotter has to do is to add these values to its plot and in the end of the script return the finished plot to the main module. All plotter subclasses have to be derived from the base class *I3RootPlots*. How to implement a new plotter is shown in the algorithm C.2.

```
// Include the motherclass
#include <root-plots/I3DataSource.h>

// Include specific classes concerning the data
...

// Implement the new class
class I3MyNewDataCollector : public I3DataSource {

 public:

   // CONSTRUCTOR
   I3MyNewDataCollector(const I3Context& context) :
     I3DataSource(context) // call the constructor of the I3DataSource motherclass
   {
     // Create steering parameters
     AddParameter("MyFirstParameter", "Description", myFirstParameter_);
     ...
   }


   // CONFIGURE
   virtual void Configure()
   {
     // Get the steering parameters from the python script
     GetParameter("MyFirstParameter", myFirstParameter_);
     ...

     // ... [additional configuring code] ...
   }


   // SOURCE
   virtual bool Source(I3FramePtr frame)
   {
     // ... [code that takes the KM3Tray frame and sets the values
     //      x, y, z, w, x_err, y_err, z_err and w_err] ...
   }


   // OPTIONAL METHOD TO RETURN DEFAULT AXIS TITLES
   virtual void GetDefaultAxisTitles()
   {
     // ... [code that provides default axis titles for the plots
     //      that are to be set to the string values
     //      x_title, y_title, z_title and w_title] ...
   }

 private:

   // Declaration of internal values
   WhatClassEver myFirstParameter_;
                ...
};
```

Algorithm C.1: *Blueprint for the implementation of a new data collector.* The new data collector class has to be derived from *I3DataSource* and contains basically three public methods. The **Constructor** that adds the python steering parameters, a method called **Configure** that gets the steering parameters from the script and does some additional configuring of the module *before* the processing of the first frame, and the method **Source** in which the actual values are calculated and set to the eight plot values (C.3). This method has to return *True*, if the values have been successfully set, otherwise *False*. An optional method **GetDefaultAxisTitles** can be used to provide default titles for the plot axes by setting the string values (C.4).

C.3.3 Register the new subclass

In order to make the new module work one has to register it in the *KM3Tray* framework. Due to compiler problems when using a template class with two arguments, the common registration macro does not work in this case. So to register a new data collector open the private file *IncludeI3Modules.cxx* and add the line

```
// Include the motherclass
#include<root-plots/I3RootPlots.h>

// Include specific root classes for the plot
...

// Implement the new class
class I3MyNewPlotter : public I3RootPlots {

 public:

   // CONSTRUCTOR
   I3MyNewCollector(const I3Context& context) :
     I3RootPlots(context) // call the constructor of the I3RootPlots motherclass
   {
     // Create steering parameters
     AddParameter("MyFirstParameter", "Description", myFirstParameter_);
     ...
   }


   // CONFIGURE
   virtual void Configure()
   {
     // Get the steering parameters from the python script
     GetParameter("MyFirstParameter", myFirstParameter_);
     ...

     // Initialize the plot
     myPlot_ = new TWhatPlotClassEver();

     // ... [additional configuring code] ...
   }


   // PLOT
   virtual void Plot()
   {
     // Add the values x, y, z, w, x_err, y_err, z_err and w_err to the plot
     // May be this works like this...
     myPlot_->Add(x,y,z,....);
   }


   // GET TOBJECT (TO RETURN THE FINISHED PLOT TO THE MAIN MODULE)
   virtual TObject* GetTObject()
   {
     // ... [code to finish the plot and do some formatting
     //      like for example set the axis titles] ...

     // Send the finished plot back to the main module
     return myPlot_;
   }

 private:

   // Declaration of internal values and the plot
   WhatClassEver myFirstParameter_;
   TWhatPlotClassEver* myPlot_;

};
```

Algorithm C.2: *Blueprint for the implementation of a new plotter.* The new plotter class has to be derived from *I3RootPlots* and contains basically four public methods. The **Constructor** that adds the python steering parameters as well as a method called **Configure** that gets the steering parameters from the script and does some additional configuring of the module *before* the processing of the first frame. This method also has to initialize the actual plot. The method **Plot** adds the values from the data collectors (C.3) to the plot and finally **GetTObject** returns the finished plot to the main module.

```
I3_MODULE_PLOTS(I3RootHistoPlots,I3MyNewDataCollector,"I3RootPlotsModule<I3RootHistoPlots,
    I3MyNewDataCollector>");
```

with all possible combinations of the new data collector and the existing plotter subclasses (that is here shown just for *I3RootHistoPlots*). Thereby the macro

I3_MODULE_PLOTS has three arguments. The first two of them are the two subclasses for a plotter and a data collector, whereas the third is the whole template name for this combination *as a string*! Afterwards also include the header file of the new class above. Including a new plotter works analogously.

To make the new module compile it has to appear in the project's cmake file *CMakeLists.txt*. As the final compilation of the *Root-plots* project depends on what is already installed in the *KM3Tray* framework (e.g. the data collector *I3ShAuerRecoParamsSource* will only be installed if there is the project *shauer-reco* installed) the list in the cmake file is quite long containing a lot of if-queries. So add the cxx file of the module in each desired query.

Bibliography

[1] Simon Mitton. *Cambridge Enzyklopädie der Astronomie, Deutsche Ausgabe*. Orbis Verlag für Publizistik GmbH, München, 1989.

[2] Albrecht Unsöld and Bodo Baschek. *Der neue Kosmos*. Springer-Verlag Berlin Heidelberg New York, 7. edition, 2005.

[3] Norbert Przybilla. Physik der Sternatmosphären lecture. *Friedrich-Alexander-Universität Erlangen-Nuremberg*, 2005.

[4] Encyclopaedia Britannica. *http://media-2.web.britannica.com/eb-media/52/752-004-6FE60E05.jpg*, website.

[5] High Performance Wireless Research and Education Network. *http://hpwren.ucsd.edu/images/20070804.jpg*, website.

[6] National Radio Astronomy Observatory / Associated Universities, Inc. / National Science Foundation. *http://images.nrao.edu/images/VLA730B_lo.jpg*, website.

[7] Lawrence Berkeley National Laboratory. *http://www.lbl.gov/Publications/Nobel/assets/img/hires_COBE_Sat3.jpg*, website.

[8] ESA Multimedia gallery. *http://esamultimedia.esa.int/images/Science/XMM1111.jpg*, website.

[9] DESY. *http://adweb.desy.de/ itphysik/gif/hess-telescope.gif*, website.

[10] F. P. Israel. Centaurus A - NGC 5128. *The Astronomy and Astrophysics Review*, 8:237–278, 1998.

[11] Chandra X-Ray Observatory. *http://chandra.harvard.edu/photo/2002/0157/centaurus-a.tiff*, website.

[12] Claus Grupen. *Astroparticlephysics*. Springer-Verlag Berlin Heidelberg New York, 2005.

[13] S.P. Swordy. The energy spectra and anisotropies of cosmic rays. *Space Science Reviews*, 99:85–94, 2001.

[14] G.T. Zatsepin and V. A. Kuz'min. Upper limit of the spectrum of cosmic rays. *Soviet Journal of Experimental and Theoretical Physics Letters*, 4:78–80, 1966.

[15] Kenneth Greisen. End to the cosmic-ray spectrum? *Phys. Rev. Lett.*, 16:748–750, 1966.

[16] Norbert Schmitz. *Neutrinophysik*. Teubner, 1997.

[17] C.Amsler. *Particle physics booklet*. Particle Data Group, 2008.

[18] Elisa Bernadini. Multi-Messenger Studies with AMANDA/IceCube: Observations and strategies. *arXiv:astro-ph/0509396v1*, 2005.

[19] Wolfgang Hampel. Solar neutrinos at GALLEX. *Phil. Trans. R. Soc. Lond. A*, 346:1–13, 1994.

[20] Pavel Alekseyevich Cerenkov. Visible radiation produced by electrons moving in a medium with velocities exceeding that of light. *Phys. Rev.*, 52:378–379, 1937.

[21] David Halliday and Robert Resnick. *Physik 1*. Walter de Gruyter & Co., Berlin, 1993.

[22] Jürgen Brunner. *The refraction index at the Antares site*. Antares Internal Note, 2000.

[23] The ANTARES collaboration. A deep sea telescope for high energy neutrinos. *astro-ph*, 9907432, 1999.

[24] The ANTARES collaboration. *ANTARES internal photo archive*.

[25] The ANTARES collaboration. *ANTARES online gallery, http://antares.in2p3.fr/Gallery/3D/index.html*, website.

[26] The K2K collaboration. *http://www-sk.icrr.u-tokyo.ac.jp/sk/ykphd/chap3-3.html*, website.

[27] Spencer R. Klein and The IceCube Collaboration. IceCube: A Cubic Kilometer Radiation Detector. *arXiv:0807.0034v2*, 2008.

[28] The KM3NeT collaboration. *http://km3net.org*, website.

[29] The ANTARES collaboration. *Technical design report of the ANTARES* $0,1km^2$ *project*. Antares Internal Note, 1. edition, 2001.

[30] P. Amram et al. The ANTARES optical module. *Nucl. Instr. and Meth.*, A484 (2002) 369, 2001.

[31] The ANTARES collaboration. *The ANTARES optical module.* Antares Internal Note, 2001.

[32] Claudio Kopper. *Energierekonstruktion von hochrelativistischen Myon-Neutrinos mit dem Wasser-Čerenkov-Detektor ANTARES, Diploma thesis.* University of Erlangen-Nuremberg, Germany, 2005.

[33] Bettina Hartmann. *Reconstruction of Neutrino-Induced Hardonic and Electromagnetic showers with the ANTARES Experiment.* PhD thesis, University of Erlangen-Nuremberg, Germany, 2006.

[34] Garmt de Vries-Uiterweerd. *Signal and Background in the Underwater Telescope ANTARES.* PhD thesis, Utrecht University, The Netherlands, 2007.

[35] Jürgen Brunner. *Upgrade of ^{40}K simulations.* Antares Internal Note, 2006.

[36] B.A.P. van Reens. *The software trigger in ANTARES.* Antares Internal Note, 2004.

[37] John Carr and Stéphanie Escoffier and Dmitry Zaborov. *Proposition for an alternative trigger based on the T3 cluster trigger.* Antares Internal Note, 2007.

[38] Ralf Auer. *Unknown title, in preparation.* PhD thesis, Universtiy of Erlangen-Nuremberg, Germany.

[39] Ralf Auer. Reconstruction of hadronic cascades in large-scale neutrino telescopes. *Nucl. Instr. and Meth.*, doi:10.1016/j.nima.2008.12.007, 2009.

[40] William L. Goffe. Simulated annealing: An initial application in econometrics. *Computer Science in Economics and Management*, 5:133–146, 1992.

[41] Ralf Auer. *Reconstruction of hadronic cascades. New features & results.* Talk on ANTARES-Meeting, 14.09.2006, CERN, 2006.

[42] Alexander Kappes and Uli Katz. *On the Use of Event Weights for the Simulation of Atmospheric Muons in ANTARES.* Antares Internal Note, 2005.

[43] *Genhen v5r1: Software Documentation.* Antares Internal Note, 2002.

[44] *A comparison between GEASIM and KM3 generators.* Antares Internal Note, 1999.

[45] M. Anghinolfi, H. Constantini, K. Fratini, D. Piombo and M. Taiuti. *New measurements of the angular acceptance of the Antares Optical Module.* Antares Internal Note, 2008.

[46] P. Kooijman. *On the angular acceptance of the optical module.* Antares Internal Note, 2007.

[47] European Organization for Nuclear Research (CERN). *http://root.cern.ch/*, website.

[48] ANTARES Data Analysis Electronic Logbook. *Entry from Tue Jun 26 16:57:00 2007.*

[49] ANTARES Data Analysis Electronic Logbook. *Entry from Mon Apr 28 13:59:53 2008.*

[50] Friedericke Schöck. *Unknown title, in preparation.* PhD thesis, Universtiy of Erlangen-Nuremberg, Germany.

[51] Y. Becherini, A. Margiotta, M. Sioli and M. Spurio. A parametrisation of single and multiple muons in the deep water or ice. *Astropart. Phys.*, 25:1–13, 2006.

[52] *KM3 v2r1: User Guide.* Antares Internal Note, 2002.

[53] Jörg R. Hörandel. On the knee in the energy spectrum of cosmic rays. *Astropart. Phys.*, 19:193–220, 2003.

[54] Elena Korolkova and Lee Thompson. *MC simulation of cosmic ray muons at sea level with CORSIKA.* Antares Internal Note, 2003.

[55] *personal conversation with Prof. Dr. Gisela Anton.*

[56] Jürgen Brunner. *Data quality.* Talk on the ANTARES Collaboration Meeting, CERN, 2006.

[57] Andreas Hölzer et al. *http://tmva.sourceforge.net/*, website.

[58] The IceCube Collaboration and T. DeYoung. *IceTray: A software framework for IceCube.* International Conference on Computing in High-Energy Physics and Nuclear Physics (CHEP 2004), 2004. available from *http://www.chep2004.org/*.

[59] Claudio Kopper. A software framework for KM3NeT. *Nuclear Instruments and Methods in Physics Research A*, 2008. doi:10.1016/j.nima.2008.12.047.

[60] Python Software Foundation. *http://www.python.org/*, website.

[61] Aart Heijboer. *Track reconstruction and point source searches with Antares.* PhD thesis, Universiteit van Amsterdam, Amsterdam, The Netherlands, 2004.

Acknowledgement

Abschließend möchte ich mich bei allen ganz herzlich bedanken, die mir während meiner Arbeit mit Rat und Tat zu Seite standen und zu deren Gelingen beigetragen haben.

Mein besonderer Dank gilt Herrn Prof. Dr. Uli Katz für die Vergabe dieses interessanten Themas und der Betreuung während meiner Bearbeitungszeit.

Des Weiteren bedanke ich mich bei Frau Prof. Dr. Gisela Anton für ihre Hilfsbereitschaft, die zahlreichen Denkanstösse und die Zweitkorrektur meiner Arbeit.

Bei Ralf Auer möchte ich mich für all die zahlreichen anregenden Diskussionen, seine intensive Unterstützung und fürs Korrekturlesen bedanken.

Ebenso bedanke ich mich bei Thomas Eberl fürs Korrekturlesen.

Claudio Kopper danke ich für seine Hilfe und Geduld bei zahlreichen Software-Fragen und -Problemen.

Ein herzliches Dankeschön geht an die Erlanger *ANTARES*-Software-Gruppe für hilfreiche Tipps und Anregungen, sowie an die Medphys-Gruppe für eine Menge amüsanter Kaffee-Pausen und Schafkopf-Partien.

Mein ganz besonderer Dank gilt schließlich meiner Familie, ohne deren Unterstützung und Rückhalt mein Studium überhaupt nicht möglich gewesen wäre.

Til slutt vil jeg gjerne takke Anja for understøttelsen og tålmodigheten hennes i løpet av det siste året.

Takk.

Obligatory declarations

I versichere diese Diplomarbeit selbstständig verfasst zu haben und keine anderen als die angegeben Hilfsmittel und Quellen benutzt zu haben. Die Verfassung in englischer Sprache erfolgte freiwillig und ohne fremden Zwang.

Erlangen, März 2009

Florian Folger