Cherenkov-Ring Energy Estimation

A New Energy Estimator for Extensive Air Showers

Henk Brans

Master Student in Particle and Astrophysics

Master Thesis

in the department of

High Energy Physics

written under the supervision of

prof. dr. Charles Timmermans

2020 - 2022

Institute for Mathematics,
Astrophysics and Particle Physics
Radboud University

Contents

1 Introduction

The goal of my Master Research was to develop an estimator for the amount of energy in the electromagnetic part of Extensive Air Showers using the radio signal on the Cherenkov-Ring. In this section, I introduce this idea and present the outline of my thesis.

1.1 The Cherenkov-Ring in Extensive Air Showers

The idea to use the Cherenkov-Ring as an estimator for the amount of energy in the electromagnetic part followed from my Bachelor Project. On the Cherenkov-Ring, radio signals emitted at different locations in the air shower, arrive at the same time. I probed this characteristic in more detail by developing a computer code that calculated the radio footprint at the Cherenkov-Ring for different zenith angles and values of X_{max}. In that research, we included a varying Cherenkov-angle that continuously corresponded with the atmospheric conditions at that altitude. As a result, we could simulate the signal in realistic atmospheric conditions and probe the effect of adding a variable Cherenkov-angle instead of using a fixed one. This allowed us to study the arrival times and the dependence of the Cherenkov-Ring shape on zenith angle and X_{max} with high precision. My Bachelor Research resulted in the following three conclusions. Firstly, we concluded that the signal arrives typically within two nanoseconds. Secondly, the Cherenkov-Ring on the ground is stretched out, primarily parallel to the shower axis, to an ellipse for more horizontal events. Thirdly, we concluded that the dependence on X_{max} is a secondary effect.

In my Master Research, I noticed a direct proportionality between the height of the signal peak on the Cherenkov-Ring and the number of electrons and positrons in the air shower. The three results of my Bachelor Research imply that the signal peak on the Cherenkov-Ring correspond to an integral of the radio signals emitted by the shower. After all, a sharp signal peak is formed at the Cherenkov-Ring, since the radio signal arrives within two nanoseconds. More precisely, it is an integral of the signal emitted by the main emission region around X_{max} at the (varying) Cherenkov-angle. The height of the signal peak should therefore be an energy estimator, because the number of electrons and positrons (emitters) scale with the primary energy. Since the X_{max} dependence is a secondary effect, the energy estimator is also approximately independent of X_{max}.

So, in brief, the research is based on the following three points:

- The sharp signal peak on the Cherenkov-Ring corresponds to an integral of the signals emitted by the shower.
- The height of the peak is proportional to the number of electrons and positrons.
- The electric field peak on the Cherenkov-Ring is an energy estimator.

1.2 Prospects of Cherenkov-Ring Energy Estimation

The experimental prospects of Cherenkov-Ring Energy Estimation (CREE) are promising thanks to the solid theoretical background and motivation. First of all, only two nanoseconds of data from one antenna on the Cherenkov-Ring is sufficient to obtain the estimate for shower energy. Meanwhile, the peak signal is 50 - 100 times stronger than the rest of the signal, thus resulting in a high signal to noise ratio. In addition, the Cherenkov-Ring is well distinguishable from the rest of the antennas. This is the case for horizontal events especially, since the shower core and Cherenkov-Ring are well separated on the larger radio footprint.

The Ring is circular symmetric in the shower plane, and this allows a consistent comparison between an entire library of events. I was able to finish the analysis of 726 events with primary energies of 0.02 EeV and 3.98 EeV, the energy limits of the library. Within this large energy range, the relation between electric field height and number of electrons and positrons is constant. This indicates that my method remains valid in a large energy range, especially in the GRAND EeV energy range.

- Goal: Construct an energy estimator from the electric field peak magnitude on the Cherenkov-Ring.
- Prospect: Less than five nanoseconds of data from one antenna on the Cherenkov-Ring is sufficient to obtain the energy estimator, provided the shower direction is known.

1.3 Research and Thesis Outline

My research was based on the simulated electric field of the GRAND Star Shape Library.[1] My Research commenced by studying the data and file structure. Subsequently, I determined which data was relevant and developed a method that allowed me to select the antennas on the Cherenkov-Ring. Thereafter, I automated the selection algorithm by developing a Python code. After this phase, I started with the analysis of the data from antennas on the Cherenkov-Ring. All of this is presented in this thesis.

The thesis started with a description of the motivation and background story on the idea. The theory is further explained in the next chapter. In chapters 3 and 4, a complete overview of the method is given. The angular distribution of the Cherenkov-Ring for one specific event is presented, and the conversion between ground plane and shower plane is provided. This example shows how the information of all events was compared. The results of the complete analysis of 726 events are presented in chapter 5. Following that, the conclusion is summarised in chapter 6. Finally, the thesis ends with a discussion on thoughts for further development of the method in the GRAND frequency range of 50 - 200 MHz. There is much more to be discovered on the topic! Ideas for further steps are included in chapter 7.

[1] Thanks to Matias Tueros for generating and providing this large library of events.

2 Theory

2.1 GRAND Convention

The Giant Radio Array for Neutrino Detection (GRAND) is a newly proposed neutrino and cosmic ray detector. The prototyping phase, GRANDProto300, is currently ongoing. GRAND focuses on the Ultra-High-Energy Cosmic Rays (UHECR) and neutrino research themes. The goal is to detect cosmic rays, gamma rays, and extragalactic neutrinos in the EeV range. GRAND will detect neutrinos by using tau-neutrino interactions in a nearby mountain. In charged-current interactions, a tau-lepton is formed in the mountain that decays after, on average 50 km, thus outside the mountain. As a result, the air shower starts from the decaying tau-lepton.[2] Therefore, the primary focus of the research is on the near horizontal, or highly inclined, air showers.

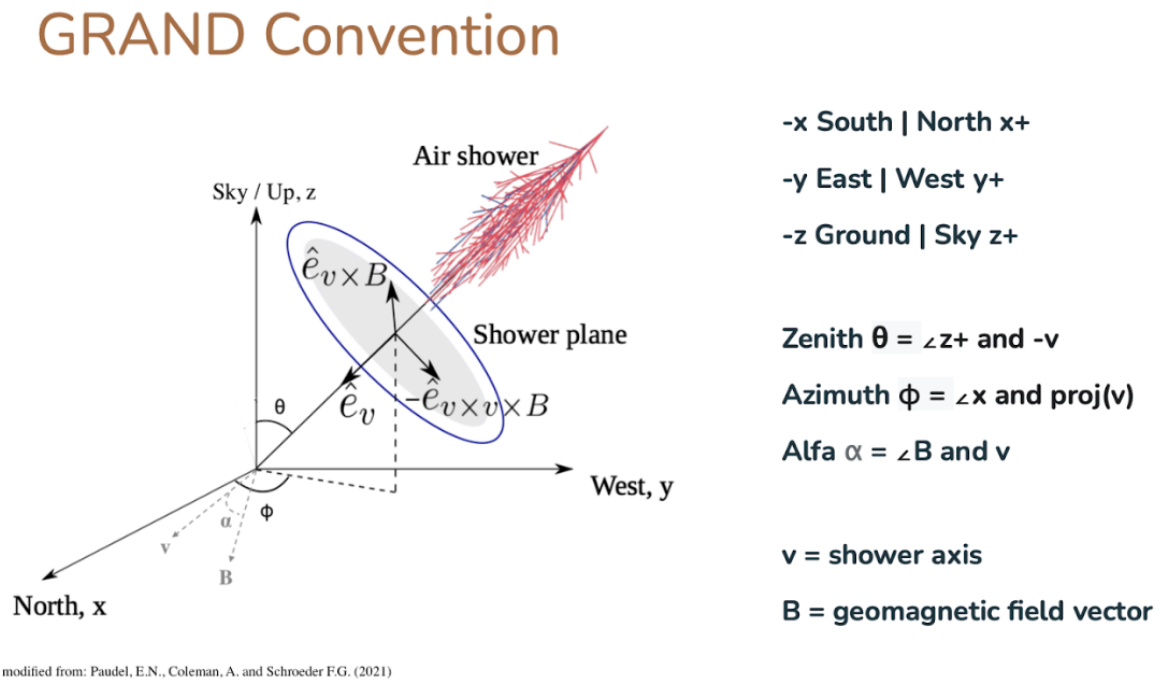

Figure modified from: Paudel, E.N., Coleman, A. and Schroeder F.G. (2021)

Figure 1: The coordinate system of GRAND. Note that the convention of axes and angles is different from common coordinate systems in cosmic ray physics. Figure modified from [2].

[2] The process, from tau-neutrino to radio emission, is discussed in more detail in my Bachelor Thesis. [1]

2.2 Characteristics of an Antenna on the Cherenkov-Ring

There are two main characteristics of the radio signal on the Cherenkov-Ring which I used in my Master Research. The first characteristic is the short duration of the pulse of less than two nanoseconds. The two nanoseconds corresponds to the sampling time of the GRAND electronics. Thus, the complete signal will be contained in one or two samples of the signal trace. The second characteristic is that the electric field peak of the signal on the Cherenkov-Ring is significantly higher compared to the rest of the radio footprint. As the signal is highly compressed in time compared to measurements at other locations on the surface, this was to be expected. The typical electric field trace of an antenna on the Cherenkov-Ring is shown in figure 2. These two characteristics imply that identification of an antenna on (or near) the Cherenkov-Ring is straightforward.

For my analysis, I use the total length of the electric field vector (the square root of the sum of the squares of the electric field in all directions). This corresponds to the radiation energy, a scalar. In this way, the value extracted is consistent between all events and straightforward to calculate.

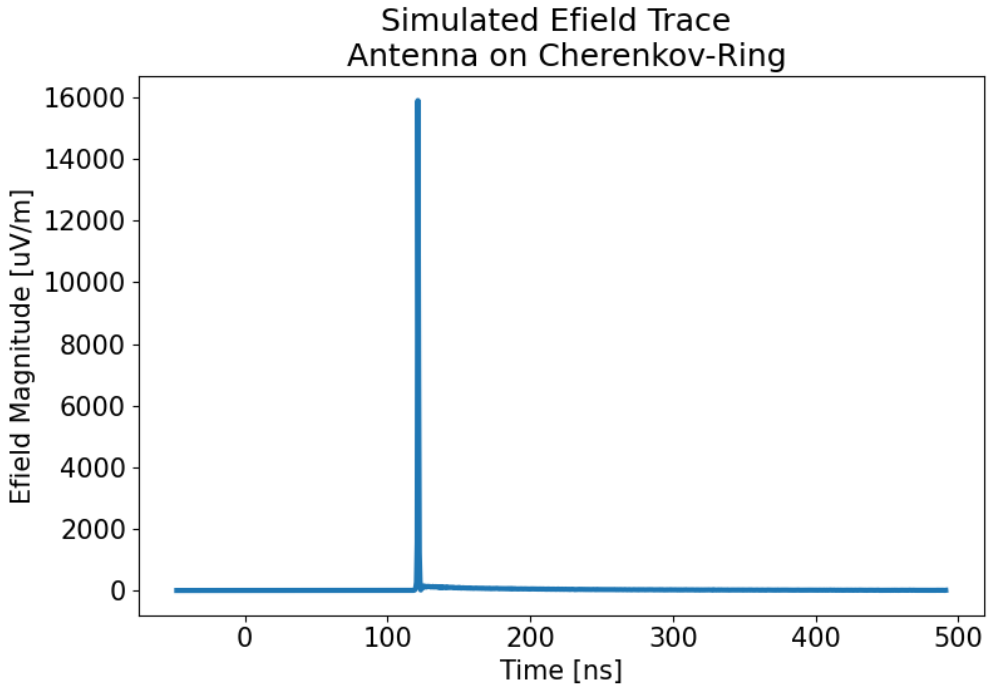

Figure 2: Magnitude (absolute value) of the electric field trace, corresponding to the radiation energy. The electric field shown contains the full bandwidth.

2.3 Cherenkov-Ring

Signals emitted at the Cherenkov angle form a coherent wave front, because the air shower, which emits the radiation, travels faster than the radio signals in air. The speed of the air shower is 0.999c and the average speed of light in air is 0.997c, where c is the light speed in vacuum. As a result, a significant stronger radio signal is observed on the Cherenkov-Ring. The Cherenkov-Ring is the ring on the ground plane (detector grid) on which these signals arrive shown schematically in figure 3. An interesting property of the Cherenkov-Ring is that the radio signals on the Cherenkov-Ring arrive simultaneously in terms of nanoseconds. So, signals, emitted at different retarded times, at the Cherenkov angle can arrive at the same point in space-time. The theoretical description is further discussed hereafter.

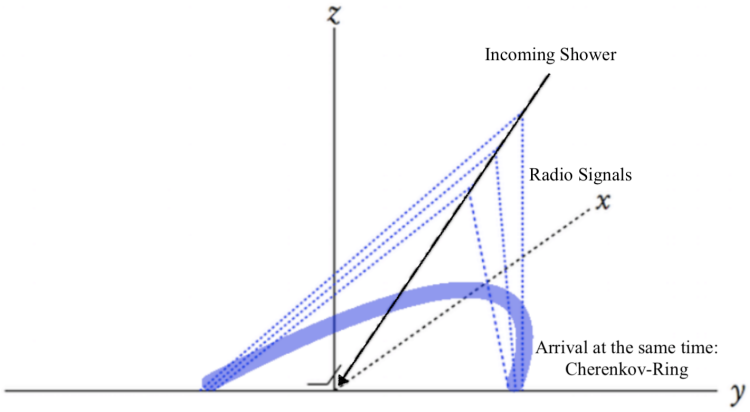

Figure 3: Schematic View of the Cherenkov-Ring: Radio signals (in blue) emitted at the Cherenkov angle, arrive simultaneously on the Cherenkov-Ring. As a result, a sharp peak is observed.

It is also illustrative to look at figure 5. One can see that on the Cherenkov-Ring, signals add up constructively, despite that the Cherenkov-angle varies due to the varying atmospheric conditions. Therefore, at this point a sharp peak is observed. The fact that the arrival times of the radio signal are (almost) equal follows from the description of the Cherenkov-Effect with a fixed Cherenkov-angle. In my Bachelor Research, briefly discussed in the next subsection, I proved experimentally that this fact is even more valid when performing the calculation with a varying Cherenkov-angle.

One can also think in a different way about the Cherenkov-Ring. The speed of the air shower is higher than the speed of light in air. The air shower thus travels faster than the radio signals. Consequently, close to the shower axis, one sees signals emitted closer to the ground earlier than the signals emitted highly above. As a result, an observer close to the shower axis, sees the shower development backwards: From the ground to the sky. An observer far away from the shower axis sees the development in the normal way: From the sky to the ground. When an observer is somewhere in between, he can be on a limit. At that point, he sees all radio signals simultaneously.

2.3.1 Numerical Calculation with varying Cherenkov Angle

In my Bachelor Research, I derived a formula that calculated the arrival times and positions of the Cherenkov-Ring. [1] For this analysis, we included a varying Cherenkov angle based on the atmospheric conditions at that altitude. The idea is shown in equation 1: I calculate the arrival time for a signal, which was emitted at a certain point X_{gram} in the slant depth of the shower, and arrives on a point Y on the ground. The point X_{gram} is given as function of the amount of air traversed along the path of the incident particle from the top of the atmosphere in g/cm^2. Figure 4 visualises how the two terms contribute to the calculation of the arrival time.

$$T_{arrival}(X_{gram}, Y) = T_{Core}(X_{gram}) + T_{radio}(Y) \tag{1}$$

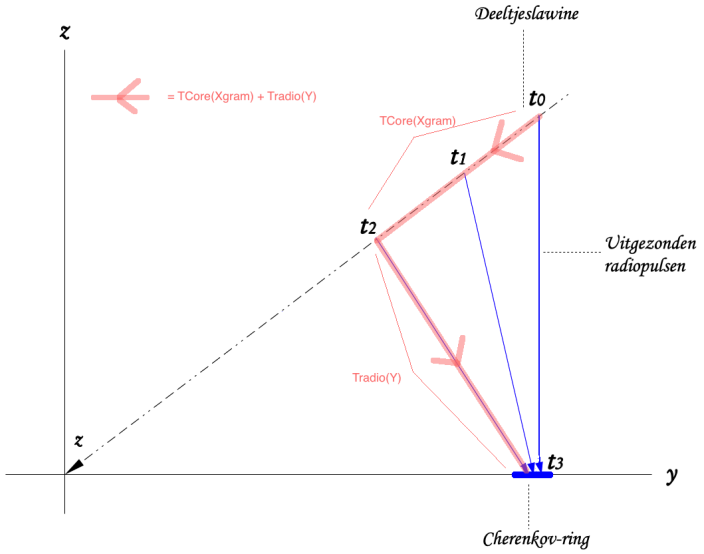

Figure 4: The arrival time on the Cherenkov-Ring is calculated by adding the travel time of the air shower up to the point of emission T_{core} and the travel time of the radio signal emitted at that point T_{radio}.

The calculation of the arrival time is performed for several points X_{gram} in the slant depth of the shower and arrival positions Y on the ground. Thereafter, the position was determined by setting the arrival times equal (equation 2). Solving the equation gives the position of the Cherenkov-Ring and provides an estimate of the duration of the peak.

$$T_{arrival}(600, Y) = T_{arrival}(700, Y) = T_{arrival}(800, Y) \tag{2}$$

This analysis showed that indeed on the Cherenkov-Ring, we observe signals simultaneously (within two nanoseconds) when emitted around the main emission region of X_{max}.

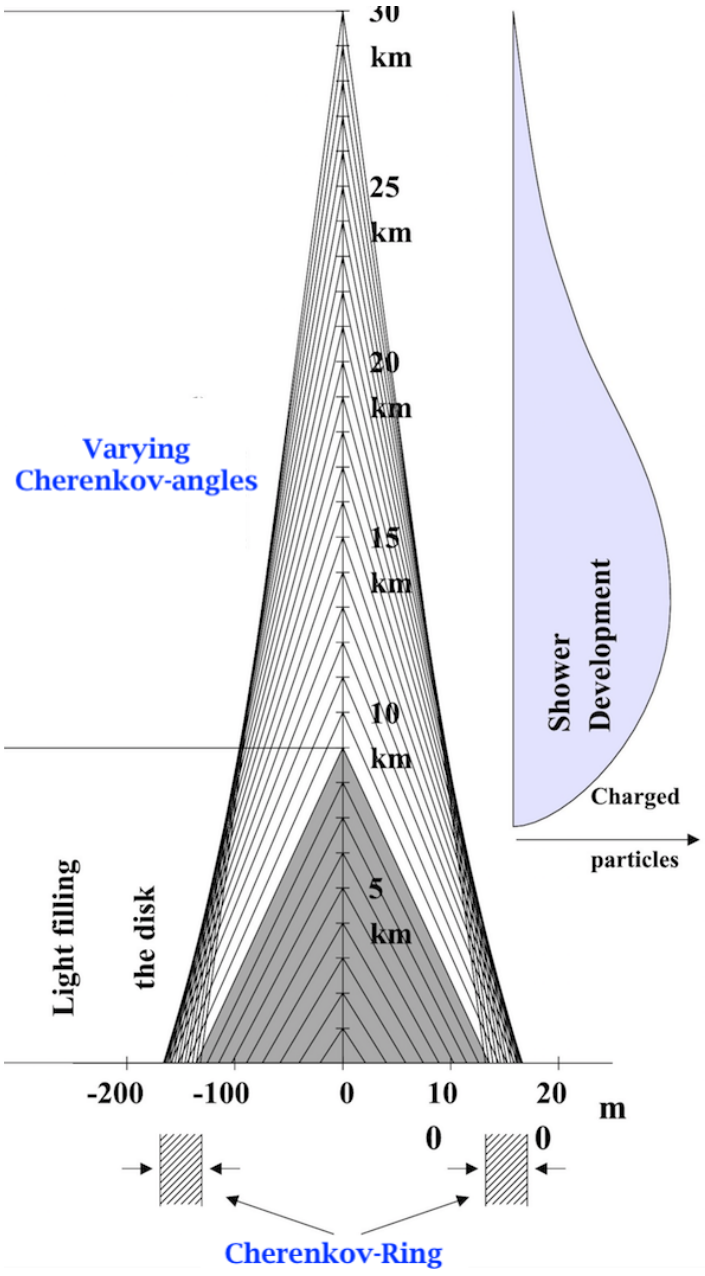

Figure 5: The Cherenkov-Ring is, more precisely, formed by signals, emitted in the main emission region around X_{max}. The Cherenkov angle varies along the slant depth, but still the signals from the main emission region arrive at the same position and time on the ground plane. This is the Cherenkov Effect. Only signals at the very end of the shower development do not contribute anymore. Figure from [3]

2.4 Theoretical Background of the Cherenkov Effect

In this subsection, I demonstrate the mathematical background of the Cherenkov Effect and explain the reason why signals arrive simultaneously on the Cherenkov-Ring. This is mostly based on the explanation of Cherenkov Radiation from a moving electron by Jackson and the Electrodynamics Course syllabus. [4] [5] Indeed, an air shower also emits Cherenkov Radiation, but this accounts for only about 1% of the total radiation energy. In the geomagnetic and Askaryan emission mechanisms is much more energy, but this is distributed over a larger area in a much longer time period. Thanks to the Cherenkov Effect, their radiation also adds up constructively on the Cherenkov-Ring and contribute to the wave front at the Cherenkov angle. The explanation is meant to provide insight into the physical mechanism when the emitting source travels faster than the light signals emitted. In this explanation, I simplify the air shower to a moving point charge Q at constant velocity $\vec{v}$.

2.4.1 Cherenkov Condition

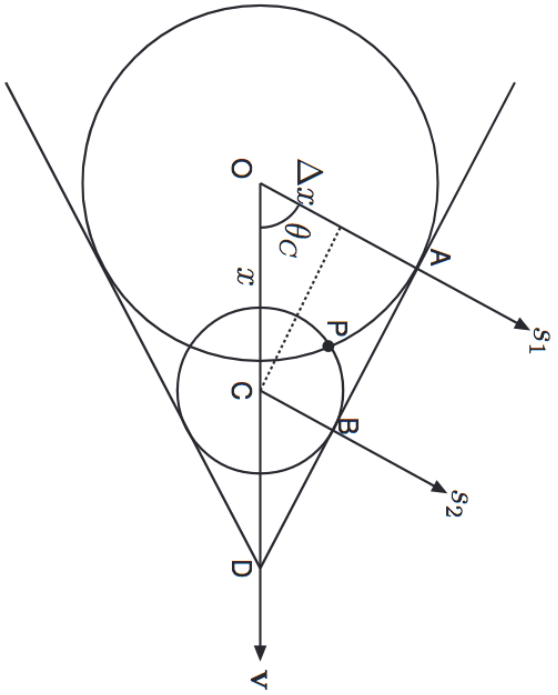

Figure 6: Imagine the air shower as a point charge moving with velocity $\vec{v}$, Signals s_1 and s_2 emitted at the Cherenkov angle θ_C, form the wave front $|AB|$. At a point P inside the Cherenkov Cone, the signals emitted at the Cherenkov angle arrive at the same time. These are the two special features of the Cherenkov Cone. Picture from [5]

First, let us look at figure 6 and suppose we have two signals s_1 and s_2 with angular frequency ω emitted at an arbitrary angle θ. We want to know for which angle θ the radiation emitted is

maximal. In this case, the signals interfere constructively and have zero phase difference. So, let us calculate their phase difference. The time difference between the emissiom times Δt is of course the distance x divided by the velocity v of the particle. The phase difference in time is the angular frequency ω multiplied by Δt. The phase difference in time is thus given by equation 3.

$$\begin{aligned} \Delta t &= \frac{x}{v} \\ \Delta\phi_t &= \omega\Delta t \\ \Delta\phi_t &= \frac{\omega x}{v} \end{aligned} \tag{3}$$

The difference Δx is the line segment $x\cos\theta$ divided by the light velocity in air $c_a = c/n$ where n is the average refractive index of air on this line segment. By multiplying with the angular frequency, we obtain the phase difference. (equation 4).

$$\begin{aligned} \Delta x &= \frac{x\cos\theta}{c/n} \\ \Delta\phi_x &= \omega\Delta x \\ \Delta\phi_x &= \frac{\omega x n\cos\theta}{c} \end{aligned} \tag{4}$$

When assuming a similar refractive index on both trajectories, the total phase difference $\Delta\phi$ between the signals s_1 and s_2 is then equal to equation 5.

$$\begin{aligned} \Delta\phi &= \Delta\phi_t - \Delta\phi_x \\ &= \left(\frac{\omega x}{v} - \frac{\omega x n\cos\theta}{c}\right) \end{aligned} \tag{5}$$

We obtain maximum constructive interference if the phase difference is exactly zero. The signals are then completely in phase and add up constructively. Therefore, we set the condition that the phase difference is zero. Solving this equation for θ, we obtain the corresponding emission angle (equation 6). This is exactly the Cherenkov condition!

$$\begin{aligned} \frac{\omega x}{v} - \frac{\omega x n\cos\theta}{c} &= 0 \\ \frac{1}{v} - \frac{n\cos\theta}{c} &= 0 \\ \frac{c}{vn} &= \cos\theta \end{aligned} \tag{6}$$

When writing $\beta = v/c$, we arrive at the Cherenkov condition. Cherenkov Radiation is indeed emitted at this angle, but the other signals emitted at θ_C by the particle (air shower) also add up constructively. This is the Cherenkov Effect and occurs when the emitting source moves faster than the velocity of light (in a dielectric medium). We arrived at the Cherenkov condition in equation 7. At the Cherenkov angle, we observe constructive interference of the signals emitted.

$$\boxed{\cos\theta_C = \frac{1}{\beta n}} \tag{7}$$

2.4.2 Inside the Cherenkov Cone: On the Cherenkov-Ring

We have seen that inside the Cherenkov Cone, signals interfere constructively. Consequently, two signals, emitted at two different retarded times and positions, arrive simultaneously inside the Cherenkov Cone. This is the result of the speed of light in air being lower than the speed of the emitting particle. The following derivation shows that two signals, emitted at two different retarded times, may contribute to the total potential in one single point P in space-time (inside the Cherenkov Cone).

Let us assume that the two potentials caused by the signals s_1 and s_2 from the previous subsection are equal in size. In this derivation, I show that these two potentials both contribute to the potential in point P and may therefore be added. In other words, at the point P, the two signal emitted at two retarded positions x_1' and x_2' and retarded times t_1' and t_2' contribute to the field observed in P. This is the Cherenkov Effect resulting from the Cherenkov 'shock wave'.

For simplicity, let us simplify the propagating air shower as a moving charge Q. R is the distance to the source, $\vec{n}$ is the unit vector pointing towards the source. t_1' and t_2' are the retarded times. The potential at point P is the result of the signal emitted by two points along the trajectory. Thus, we sum two potentials Φ of signals emitted at retarded times t_1' and t_2'. We define $\kappa = (1 - \vec{\beta}\cdot\vec{n})$ and then obtain equation 8

$$\begin{aligned} \Phi(\vec{x},t) &= \Phi(t_1') + \Phi(t_2') \\ &= \left[\frac{Q}{|\kappa R|}\right]_{t_1'} + \left[\frac{Q}{|\kappa R|}\right]_{t_2'} \end{aligned} \tag{8}$$

The particle travels at constant velocity $\vec{v}$, so its position is defined as $\vec{r}(t') = \vec{v}t'$. We aim to solve for the retarded time condition (light cone condition) in equation 9. For a signal emitted at $\vec{r}(t')$, at the retarded time t', the light cone condition yields the following:

$$\begin{aligned} c_a(t-t') &= |\vec{x} - \vec{r}(t')| \\ t' &= t - \frac{|\vec{x} - \vec{r}(t')|}{c_a} \end{aligned} \tag{9}$$

Thus for point P, all signals that fulfil the following equation 10:

$$t' + \frac{|\vec{x} - \vec{r}(t')|}{c_a} - t = 0 \tag{10}$$

Let us rewrite some terms below. The point where the signal is emitted at the retarded time $\vec{r}(t') = \vec{s}_1(t')$ is given by equation 11. The distance |P-P'| = $\vec{X}$ between point P and the point

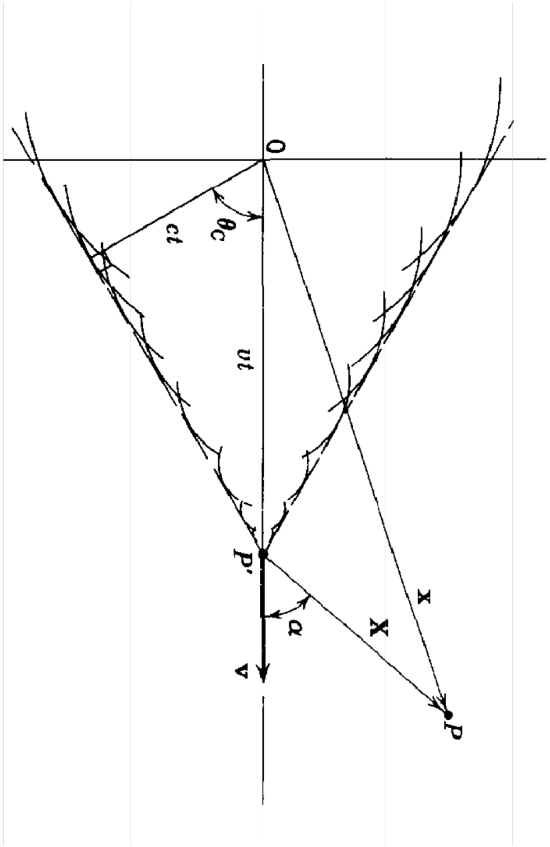

Figure 7: Point P is inside the Cherenkov wave front (Cherenkov Cone). Figure from [4]

P' of the particle, at time t when the signal is received in P is written in equation 11.

$$\begin{aligned} \vec{r}(t') &= \vec{v}t' \\ \vec{X} &= \vec{x} - \vec{v}t \end{aligned} \tag{11}$$

We define $c_a = c/n_{air}$ as the light velocity in air, where n_{air} is the refractive index of air. We square the expression and solve the quadratic equation 12.

$$\begin{aligned} (t-t') &= \frac{1}{c_a}|\vec{X} + \vec{v}(t-t')| \\ c_a^2(t-t')^2 &= \vec{X}^2 + \vec{v}^2(t-t')^2 + 2\vec{X}\cdot\vec{v}(t-t') \\ 0 &= (v^2 - c_a^2)(t-t')^2 - 2\vec{X}\cdot\vec{v}(t-t') - \vec{X}^2 \end{aligned} \tag{12}$$

Now, we obtain the solutions to the light cone (retarded time) condition (equation 13).

$$(t-t') = \frac{-\vec{X}\cdot\vec{v} \pm \sqrt{(\vec{X}\cdot\vec{v})^2 - (v^2 - c_a^2)X^2}}{(v^2 - c_a^2)} \tag{13}$$

We should carefully look in which case we obtain physical valid solutions. That is the case for angles inside the Cherenkov Cone. We should satisfy equation 14.

$$\begin{aligned}(\vec{X}\cdot\vec{v})^2-(\vec{v}^2-c_a^2)\vec{X}^2 &\geq 0\\ X^2v^2\cos^2\alpha-(v^2-c_a^2)X^2 &\geq 0\\ \cos^2\alpha &\geq (1-\frac{c_a^2}{v^2})\\ \cos\alpha &\geq \sqrt{1-\frac{c_a^2}{v^2}}\end{aligned} \tag{14}$$

So, the potentials are valid for angles α inside the Cherenkov Cone only for the interval 15:

$$\alpha \in \left\langle \arccos\left(-\sqrt{1-\frac{c_a^2}{v^2}}\right);\pi \right\rangle \tag{15}$$

The solutions of the quadratic equation allows to calculate κR. According to Jackson, the values of κR correspond to the two roots of the retarded time condition in equation 13. Note again that we must force κR to be positive; β_a can exceed one in this special case, because $v > c_a^2$. We obtain the solutions 16 after rewriting in terms of α and θ_C.

$$\begin{aligned}(\kappa R)_{1,2} &= \mp\frac{1}{c_a}\sqrt{(\vec{X}\cdot\vec{v})^2-(v^2-c_a^2)X^2}\\ |\kappa R|_{1,2} &= \frac{1}{c_a}\sqrt{X^2v^2\cos^2\alpha-(v^2-c_a^2)X^2}\\ &= X\sqrt{1-\frac{\sin^2\alpha}{\cos^2\theta_C}}\end{aligned} \tag{16}$$

The potentials 1 and 2 can be added, and the outcome represents a wave front travelling in the direction of the Cherenkov angle. The potential is valid only inside the Cherenkov Cone.

$$\boxed{\Phi(\vec{x},t)=\frac{2Q}{X\sqrt{1-\frac{\sin^2\alpha}{\cos^2\theta_C}}}} \tag{17}$$

The equation 17 represents a wave front travelling in the direction of θ_C. We prove that in the special case of the Cherenkov Effect, two signals s_1 and s_2, emitted at two different retarded times and positions, contribute to the potential observed in one single point in space-time. Hence, we observe the peak in radiation inside the Cherenkov Cone.

2.5 Distance to Xmax Effect

The distance to X_{max} along the Cherenkov-Ring varies for inclined showers, because the Cherenkov-Ring is projected on a plane inclined with respect to the direction in which the air shower particles move: the ground plane. As a result, the Cherenkov-Ring on the ground plane has an elliptic shape. For completeness, the geometry of a cone and an ellipse are shown in figure 8. The reader can see that the Cherenkov-Ring on the ground plane is, in essence, a conic section. The distance to the emission region along the ellipse varies and is described by a sine. The signal emitted is projected onto a ring whose area is proportional to the distance to this emission region. For the horizontal showers, analysed in this thesis, this effect dominates the angular distribution of the electric field height on the Cherenkov-Ring.

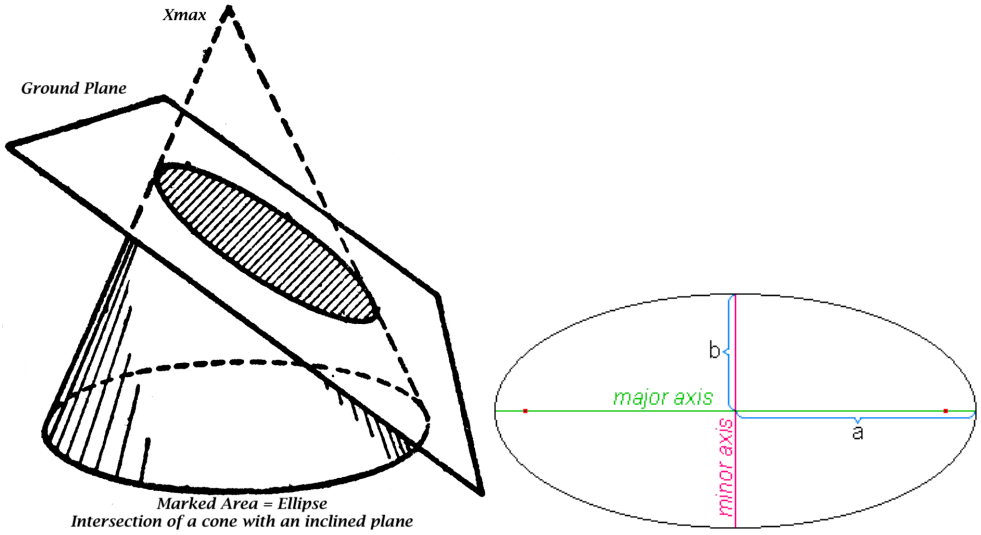

Figure 8: The left figure illustrates that the Cherenkov-Ring on the ground plane is in fact a conic section. The marked area is an ellipse. The right figure recalls the definition of the major axis (a) and minor axis (b) of an ellipse. Figures from [6] [7]

The maximal and minimal distance to X_{max} are on the contact points with the major axes of the ellipse. Whereas, the average distance corresponding to the offset of the sine distribution, is on the contact points with both minor axes. In figure 9, a schematic side view of an inclined shower is shown illustrating the maximum and minimum distance to X_{max}. The reader can see that the minimal distance is under the shower axis, on the contact point with the major axis. At the same time. The maximal distance is on the opposite side, on the opposite major axis. Along the ellipse, the distance to X_{max} is described by a sine function. The electric field is maximal where the distance to X_{max} is minimal, and the electric field is minimal where the distance to X_{max} is maximal. As a result, the electric field height along the ellipse follows a sine distribution for which a < 0 as in equation 18.

$$|\vec{E}_{antenna}|(\beta) \approx a\sin(\beta) + C \tag{18}$$

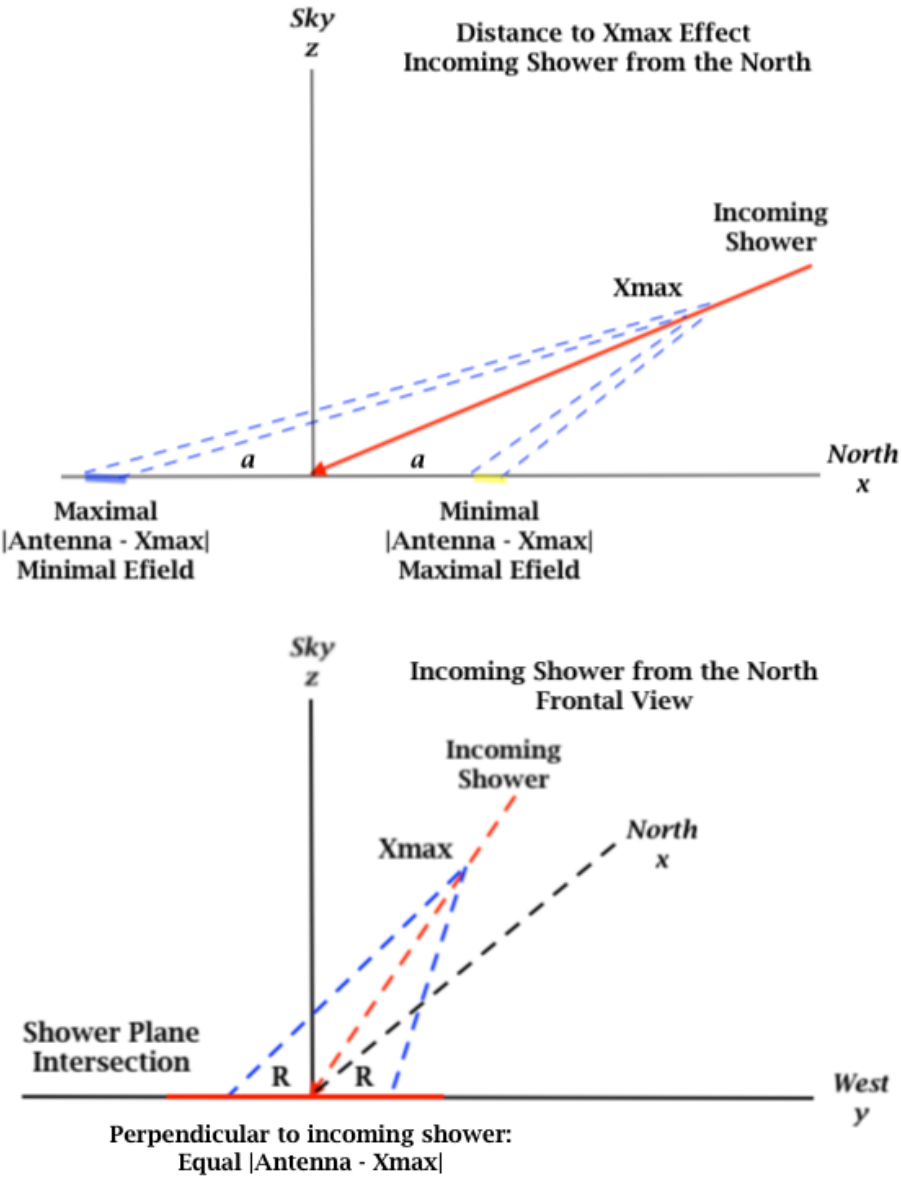

Figure 9: Above: Schematic side view of an incoming air shower illustrating the Distance to X_{max} Effect. Signals emitted from X_{max} at a fixed angle form a cone. An ellipse is formed when the cone is projected under a (highly) inclined angle onto the ground plane. The distance to X_{max} along the ellipse follows a sine distribution. The maximum and minimum are on the contact point with the major axes (a). The distance to X_{max} is minimal under the shower axis. It is maximal on the contact point with the opposite major axis. Below: Schematic front view illustrating the average distance to Xmax is on the minor axes (b) of the ellipse, perpendicular to the incoming shower. This is the offset of the angular distribution.

2.6 Ground Plane and Shower Plane

In cosmic ray physics, the ground plane coordinates are often transformed to shower plane coordinates for further analysis of the radio footprint. The shower plane is the plane perpendicular to the propagating air shower and the geomagnetic field. The specific coordinate transformation is demonstrated in the Method section. In the shower plane, the Cherenkov-Ring is a circle allowing convenient analysis and consistent comparison between events.

The shower plane is perpendicular to the incoming shower (v-axis). As a result, the projection into an ellipse is reversed and the perpendicular projection of a cone, a circle, is obtained. Hence, the Cherenkov-Ring is a circle in the shower plane. In addition, the shower plane is perpendicular to the geomagnetic field (B-vector). This combination makes the radio footprint universal for all shower directions allowing consistent comparison between events.

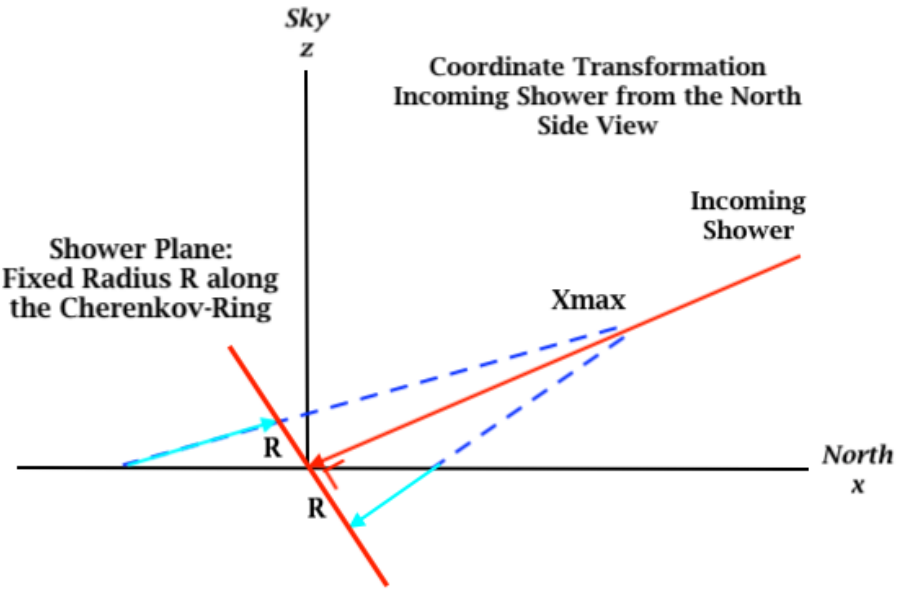

Figure 10: A schematic idea of the coordinate transformation to the shower plane is shown in the above figure. The most outer points of the ellipse, on the major axis, are transformed along the light blue arrow. Consequently, the radius to the origin is equal along the full Cherenkov-Ring thanks to the fixed emission angle.

It should be mentioned that this works well for the antenna positions in the simulations, because the star shape pattern was defined in the shower plane for each shower direction separately. When having a random or squared antenna grid on the ground in reality, the antennas would not align perfectly for each direction.

2.7 Interference Effect

The main emission mechanism in air showers is the result of charge separation induced by the geomagnetic field. Electrons and Positrons drift in opposite directions, resulting from their interactions with the geomagnetic field, due to their opposite electric charge. This causes in the most simplistic view a moving dipole which emits radiation. The polarisation of the radiation emitted by the geomagnetic effect is always pointed in the same direction: the negative v-cross-B-direction. The second emission mechanism, Askaryan emission (or charge excess) is the result of an unequal number of electrons and positrons in the front of the air shower. Electrons are liberated from air molecules and travel with the shower front, resulting in a higher number. Therefore, there is an excessive negative electric charge in the shower front with a series of positively charged ions lagging behind. The electrons move faster than the phase velocity of light and therefore emit a cone of coherent radio emission. The polarisation is pointed towards the centre of the air shower. Thus, we can add the term $b\cos(\beta)$ to our angular distribution and obtain equation 19.

$$|\vec{E}_{antenna}|(\beta) = a\sin(\beta) + b\cos(\beta) + C \tag{19}$$

When the polarisation pattern is projected onto the shower plane on a plane as in figure 11, it is clearly visible at which points of the radio footprint the polarisation of the two effects is destructively or constructively. The ratio of geomagnetic radiation and Askaryan emission is about 90% and 10% respectively, so the geomagnetic radiation is mostly dominant (in realistic scenarios; the exception is when the direction of a cosmic ray particle is close to the local direction of the Earth' magnetic field). The reader can see that on the positive v-cross-B-axis both effects add up constructively, resulting in a higher radiation energy. The minimum is on the negative v-cross-B-axis where both effects add up destructively. Again, we have an offset on both positive and negative v-cross-v-cross-B-axis. In conclusion, the angular distribution on the Cherenkov-Ring can be described as a cosine function, with maximum on the v-cross-B-axis.

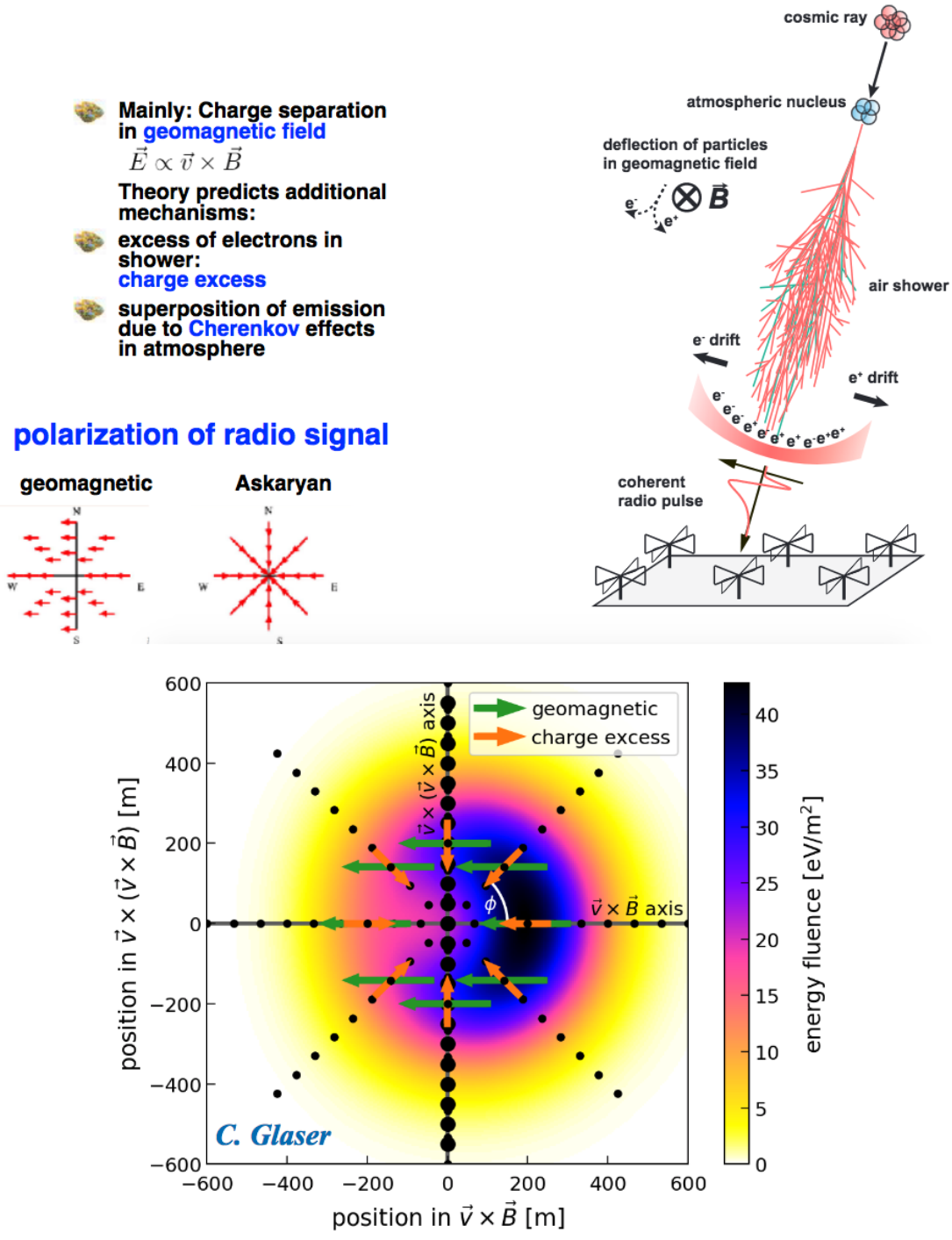

Figure 11: Above: Slide by Jörg Hörandel explaining the emission mechanisms in Extensive Air Showers. [8] Below: General radio footprint of a vertical air shower plotted in the shower plane. The interference effect is clearly visible, and the energy fluence is maximum on the positive v-cross-B-axis. The angle ϕ in this plot is referred to as β in this thesis. [9]

2.8 Scaling of the Electric Field with Zenith and Alpha angles

The total radiation energy emitted by an air shower depends on the geometry of the shower. The cause is the scaling of the emission mechanisms with the angle to the geomagnetic field (α) and air density. The geomagnetic component scales in $\sin(\alpha)$, because the mechanism is most efficient when the geomagnetic field is perpendicular to the shower axis. The Askaryan component scales with air density. For the general correction formula, the density at X_{max}: $\rho_{X_{max}}$ is taken. The general correction for these two effects is given in equation 20. The parameter a is the ratio between geomagnetic and Askaryan emission. Since, the geomagnetic component is dominant, this is also the dominant factor in the correction formula. Higher order terms are not discussed here and can be found in the literature. [10] [11] [12]

$$S_{RD} = \frac{\text{Energy}_{rad}}{\left(1 - a(\rho_{X_{max}})^2\right)\sin^2(\alpha) + a(\rho_{X_{max}})^2} \tag{20}$$

The formula above corrects the total radiation energy. However, on the ground plane, the radiation energy observed depends on the distance to the shower maximum. We have seen that the Distance to X_{max} effect also contributes to the angular distribution for inclined air showers. I corrected for the effect on the angular distribution by using a periodic deviation from the average. The antenna distance to X_{max} in general depends on the zenith angle. Therefore, this effect should be corrected for as well. In theory, a term with the distance of the antenna grid to X_{max}: $|\vec{A} - X_{max}|$ could be added as written in equation 21.

$$S_{RD} = \frac{\text{Energy}_{rad}}{a(\rho_{X_{max}})^2 + \left(1 - a(\rho_{X_{max}})^2\right)\sin^2(\alpha)} \frac{1}{|\vec{A} - X_{max}|^2} \tag{21}$$

The term $|\vec{A} - X_{max}|$ should be rewritten, because I aimed to construct an energy estimator which is independent of X_{max}. Therefore, at first, I tried to use $X_{start}(\theta)$ to 'translate' the dependence on the zenith angle. The definition of X_{start} is given in equation 22 and demonstrated in figure 12: It is the distance from the antenna grid to a point of fixed altitude. R is the radius of the Earth at the latitude of the antenna stations, a_{top} (35 km) the fixed altitude and $b_{station}$ (1086 m) is the height of the antenna stations above sea level. In other words, the distance travelled by the air shower, starting from a fixed altitude. This distance depends on the zenith angle.

$$X_{start}(\theta) = \sqrt{(R + a_{top})^2 - (\sin(\theta) \cdot (R + b_{station}))^2} - \cos(\theta) \cdot (R + b_{station}) \tag{22}$$

Since I neglect the secondary dependence on X_{max}, I remove the dependence of the Askaryan emission with air density at X_{max} from the correction formula. I use the (average) electric field on the Cherenkov-Ring ($\vec{E}_{ring}$) in my analysis and therefore remove the squares from the expression. Now, I obtain the simplified correction formula that only depends on the shower direction as written in equation 23.

$$\boxed{\vec{E}_{corrected}(\theta, \alpha) = \frac{\vec{E}_{ring}}{X_{start}(\theta)\sin(\alpha)}} \tag{23}$$

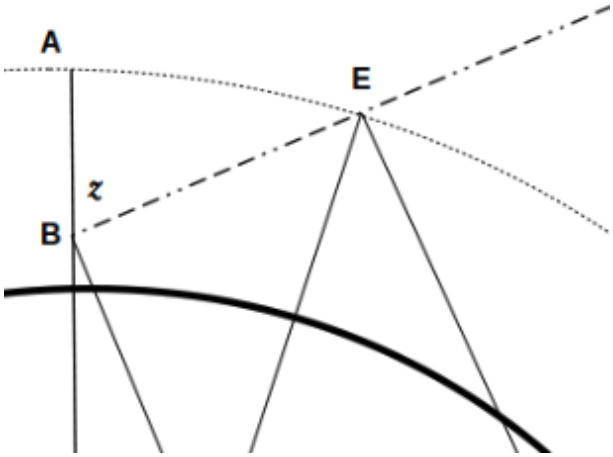

Figure 12: The above figure shows that X_{start} is the distance traversed by the air shower, starting from a fixed altitude E. The line segment |BE| is X_{start} and depends on the zenith angle $z = \theta$. The formula is derived in my Bachelor Thesis [1].

2.9 Geomagnetic Field Vector

The geomagnetic field plays a dominant role in the development of air showers. It is expressed by the geomagnetic field vector $\vec{B}_{geo}$. The geomagnetic field vector is commonly described by its length (absolute value) $|\vec{B}_{geo}|$ and the inclination angle ι and declination angle δ. The definitions of these angles are shown in figure 13. The geomagnetic field vector is used in the coordinate transformation in the Methods section.

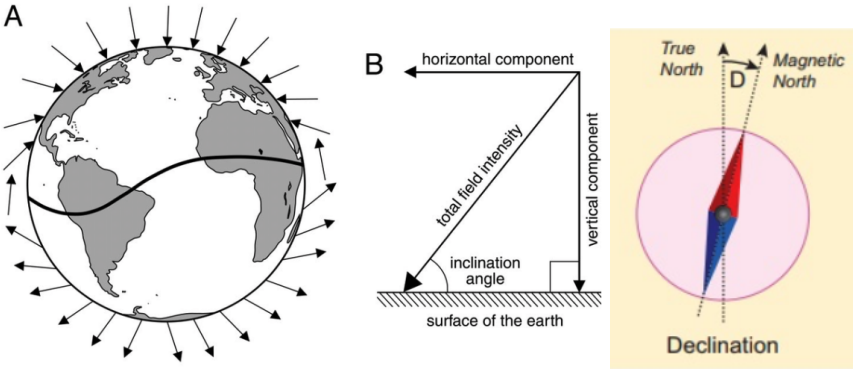

Figure 13: Left A: The geomagnetic field vector points upwards on the lower hemisphere and points downwards on the upper hemisphere. Left B: The elevation angle between the magnetic field vector and the earths surface is called the inclination angle (ι). Right: The angle between the geomagnetic north (magnetic field vector) and the geographic north is called the declination angle (δ). Figures from [13] [14]

3 Method Part I: Selection and Analysis of the Cherenkov-Ring

3.1 Star Shape Library

The Star Shape Library currently forms the basis for the study and analysis of the radio footprint of air showers within GRAND. The Star Shape Library is made by Matias Tueros and based on AIRES (AIR Shower Extend Simulations). ZHAireS, an extension to AIRES, simulates the radio signal emitted by the air shower. [15]. Matias Tueros produced a large library containing a detailed sampling of the radio footprint with 175 antennas. The simulations provide a detailed trace of the electric field with a sampling time of 0.5 ns for each antenna on the field. The simulation data of each event is stored in a HDF5 file containing detailed information of the event which is divided in data groups. Data groups included are, among others, simulation and event information, the complete electric field traces, and the longitudinal shower profile. After studying the file structure and contents, I identified the relevant information for my research. The next step of the file analysis was to write an algorithm that selects the antennas on the Cherenkov-Ring. This began by manually extracting the antennas of a few events, and ended with creating a selection algorithm described in the next subsection.

Set	Zenith [°]	Azimuth [°]	X_{max} [g/cm^2]	Number of Events
Proton 3.98 EeV 79deg	79.5	0, 90, 180	600 - 1000	66
Proton 3.98 EeV 38deg	38.5	0, 90, 180	600 - 1000	66
Proton 3.98 EeV	70 - 88	0, 90, 180	600 - 1000	726
Proton 0.02 EeV	70 - 88	0, 90, 180	600 - 1000	726

Table 1: Sets of events analysed in this thesis.

Some characteristics of the events analysed are shown in table 1. At the beginning stage of my project, two sets containing 66 events of horizontal and vertical showers were available. These events were used to study the event structure and construct the selecting algorithm. Later on, I requested more files and gradually gained access to a full library of 25 000 events. As an analysis of all events would be too extensive for a Master Project, I decided to study the energy limits of the library: 0.02 and 3.98 EeV and take a subset of horizontal showers with Zenith 70 - 88 degrees.

3.2 Selection Algorithm

The trace of the electric field magnitude (absolute value) of an antenna on the Cherenkov-Ring has two main characteristics: the short peak width and the remarkable height compared to antennas off the Ring. These two criteria were used to set thresholds that select antennas. These thresholds are shown in table 2 and explained in this subsection. The values of the thresholds were determined experimentally. I select antennas based on the height of the electric field and the narrow peak width. Therefore, I only use the Peak-to-Peak value

of the electric field and the trace of electric field magnitude, which are explained below. The polarisation or direction is thus not taken into account in order to make the analysis straightforward.

Selection Step	Parameter	Typical Value
Peak-to-Peak	T_{mean}	2 - 3
Relative voltage level defining the peak width	T_{peak}	0.3 - 0.7
Samples in the peak	T_{width}	5-8

Table 2: Thresholds used in the selection algorithm. The parameters are explained in the next subsections.

3.2.1 Pre-Selection with Peak-to-Peak Electric Field

The first step of the selection algorithm is the pre-selection and is based on the remarkable height of the signal on the Cherenkov-Ring. The pre-selection uses the Peak to Peak Electric Field of each antenna, which is one value for each antenna. The Peak-to-Peak Electric Field (P2P Efield) is the electric field difference between the top and bottom of the waveform. In essence, it is also equal to the full height of the waveform. The pre-selection works as follows: First, I extract the Peak to Peak Voltages of each antenna $E_{P2P}(A[i])$ and calculate the mean $\overline{E_{P2P}}$. Then, I define a threshold factor T_{mean}, which value was found by trial and error. The command for the pre-selection is as shown in equation 24. The function is 'find_HVM' in my code in the attachment.

$$\text{From all antennas A[i], select:}$$
$$E_{P2P}(A[i]) > \overline{E_{P2P}} * T_{mean} \tag{24}$$

3.2.2 Selection with Peak Width

The second selection step is based on the sharp electric field peak on the Cherenkov-Ring and therefore uses the peak width of the electric field. For this selection, I use the full trace of the antenna. The trace of an antenna is a list of values containing the full waveform of electric field measured by an antenna. Each subsequent bin contains the integrated electric field in a bin of width determined by the sampling time set by the simulation. The sampling time in our simulations is 0.5 ns allowing more detailed analysis, since the sampling time of the real detector is 2 ns. First, I extract the full electric field traces $E_{trace-x}$, $E_{trace-y}$, $E_{trace-z}$ of the antennas A[i] that passed the pre-selection. Next, I calculate absolute value of each bin j in the electric field to obtain the electric field magnitude trace E_{trace} (equation 25). Keep in mind that each component of the electric field has its own trace.

$$E_{trace}[j](A[i]) = \sqrt{E_{x-trace}[j]^2 + E_{y-trace}[j]^2 + E_{z-trace}[j]^2} \tag{25}$$

The peak width selection works with the electric field trace at each individual antenna location. The sampling time of the simulation is 0.5 ns, so the data files contain 0.5 ns bins of the

electric field in x, y and z direction. The selection algorithm uses the absolute value of the electric field. The bin with the highest value is saved (Max). The peak width is then determined with two thresholds: T_{peak} and T_{width}. T_{peak} is a percentage. A threshold T_{peak} * Max is set. The code checks how many samples are above threshold. The T_{width} is the number of samplings, and thus represents the duration of the peak. If the length of the peak is below T_{width}, it is selected to be on the Cherenkov-Ring.

$$\begin{aligned} &\text{From the bins around } E_{trace}[m](A) = E_{max}(A), \text{keep the bins:} \\ &E_{trace}[m-1](A) > E_{max}(A) * T_{peak} \\ &E_{trace}[m-2](A) > E_{max}(A) * T_{peak} \\ &\text{etc.} \\ &E_{trace}[m+1](A) > E_{max}(A) * T_{peak} \\ &E_{trace}[m+2](A) > E_{max}(A) * T_{peak} \\ &\text{etc.} \end{aligned} \tag{26}$$

The electric field peak $E_{peak}(A)$ of an antenna is in actually a part of the electric field magnitude trace (equation 27. It consists of the bins around the bin with the maximum electric field that are above the threshold set above.

$$E_{peak}(A) = (..., E_{trace}[m-1](A), E_{trace}[m](A), E_{trace}[m+1](A), ...) \tag{27}$$

Now, I can perform the actual selection. I set the value T_{width} and select with the length of the electric field peak. The length is in this case the number of bins $n\left(E_{peak}(A[i])\right)$. The peak width shall be below the threshold T_{width} (equation 28). Antennas that pass this selection are saved in the so-called 'cone_list' in my code.

$$\begin{aligned} &\text{From the remaining antennas A[i], select:} \\ &n\left(E_{peak}(A[i])\right) < T_{width} \end{aligned} \tag{28}$$

To summarise, the selection algorithm is set by the parameters in table 2. The values of the thresholds are found experimentally.

3.2.3 Inner and Outer Ring of Antennas

In most events, either two rings of eight antennas or one full ring and one partial ring survive the selection algorithm described above. Indeed, the properties of the Cherenkov-Ring stretch out to a certain width for inclined showers especially, which also resulted from the calculations of my Bachelor Thesis. Two full rings of eight antennas in both ground and shower plane are shown in figure 14. (The normalisation and coordinate transformation from ground plane to shower plane are explained in the next subsections.) In this figure, we see that there is a significant difference in electric field height between the inner and outer ring of antennas. Therefore, we could argue that one ring is 'closer to' the Cherenkov-Ring

than the other.

Therefore, I define the position of the Cherenkov-Ring as the point with the maximum electric field. In general, this is the full ring of antennas that survived the selection and is the one closest to the shower core. In theory, it should be the ring of antennas with the smallest peak width, but at the scale of (tenths of) nanoseconds, this property becomes indistinguishable. So, in order to select the antennas that are closest on the Cherenkov-Ring, that is on its maximum, we select on the on electric field strentgh, or the distance to the origin (shower core). For both, we should first calculate the azimuth antenna angle A_{angle}, which is the angle of the antenna starting from the x-axis (north-axis) and can be calculated with equation 29. Here, A_x, A_y are the x and y position or the (NS = NorthSouth) and (EW = EastWest) position respectively. The antennas on one 'arm' have the same azimuth antenna angle, and in this way we can group the antennas on the same arm. The atan2 function is used to have a range corresponding to 360 degrees.

$$A_{angle}[i] = \text{atan2}(A_y[i], A_x[i])[] \tag{29}$$

So, if there are antennas with the same antenna angle, I keep the antenna to closest to the shower core, or has the highest Peak-to-Peak electric field value. Both final selections worked properly.

$$\begin{gathered} \text{If} A_{angle}[i] = A_{angle}[j], \text{then select:} \\ \text{Min}(A_{radius}[i], A_{radius}[j]) \text{ or } \text{Max}(A_{P2P}[i], A_{P2P}[j]) \end{gathered} \tag{30}$$

3.2.4 Further Step: Selection with Fourier Spectrum

One could question of course whether the selection algorithm presented above will still work on real antenna data when the nice star shape pattern is absence. Inge van Rens discovered that it is convenient to select antennas based on the Fourier Spectrum of the electric field. We expect implementing this extra criterion would make the selection work sufficiently on real antenna data. The Fourier spectrum is flat for an antenna on the Cherenkov-Ring. Inge van Rens will write more on this in her Master Thesis later in 2022.

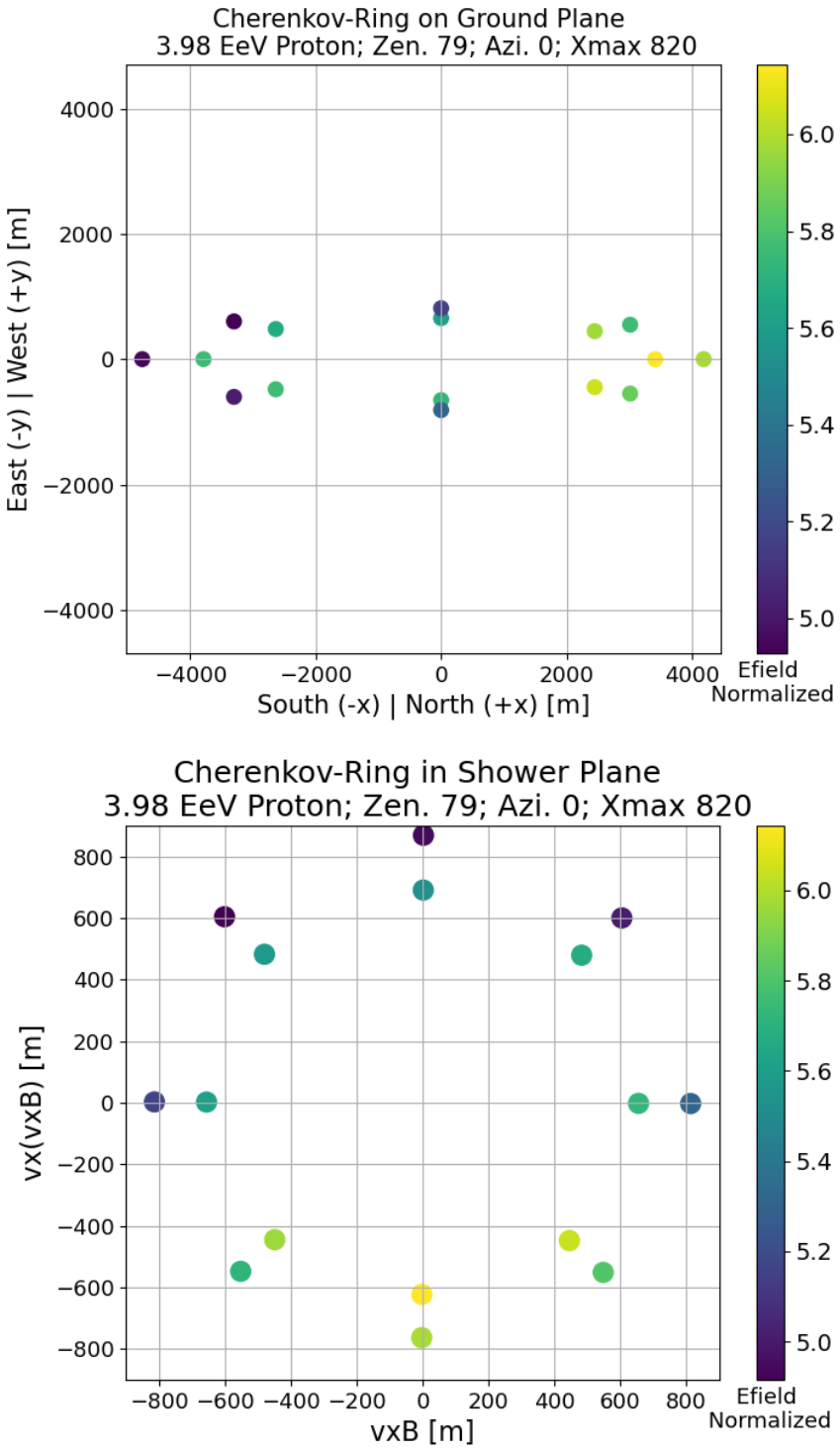

Figure 14: Above: Antennas that are selected by the algorithm. The inner ring of antennas, which have a higher electric field, are 'closer on' the Cherenkov-Ring. Below: The antenna positions in the shower plane, where they form a circular pattern.

3.3 Normalisation of the Electric Field

The Electric Field of each antenna location is normalised with the total number of electrons and positrons in the shower. This is done by dividing the electric field in each event by the number of electrons and positrons. This number is extracted by a summation along the longitudinal profile of the air shower (equation 31). Initially, I wanted to normalise and probe the relation with primary energy. However, a bug in the hadronic part of the AIRES simulation caused unrealistically large fluctuations (up to factor 3) in the electric field height of the entire radio footprint! The shape of the radio footprint was not affected at all. More on this issue is discussed in the results section below.

$$N_{e^+e^-} = \sum_{i=0}^{X_{ground}} n_{e^+e^-}(X_i) \tag{31}$$

3.4 Coordinate Transformation

3.4.1 Inconsistency in definition of Zenith and Azimuth Angle

The ZHAires convention on angles is different from GRAND. Therefore, the angles extracted from the data group 'EventInfo' must be converted to the GRAND convention. The conversion formulae for the zenith and azimuth angle are shown in equation 32 and 33. $\text{Zen}_{ZHAires}$ and $\text{Azi}_{ZHAires}$ are the zenith and azimuth angle in the HDF5 files, and θ and ϕ are the zenith and azimuth angle according to the GRAND convention presented in the Theory Section.

$$\theta = 180° - \text{Zen}_{ZHAires} \tag{32}$$

$$\phi = -180° + \text{Azi}_{ZHAires} \tag{33}$$

In the first stages of my Master Project, I used the conversion formula in equation 34 for the azimuth angle. This had no consequences to my results, because only azimuth angles of 0, 90, and 180 degrees were present (180, 270, and 0 degrees in ZHAires Convention). The conversion result was identical for 0 and 90 degrees. There was a plus and minus sign difference in the outcome for 180°. However, the meaning of an azimuth angle of +180°and -180°is identical. The imperfect conversion formula might still be present at several places in my code.

$$\phi = |\text{Azi}_{ZHAires} - 180°| \tag{34}$$

3.4.2 From Ground Plane to Shower Plane

The procedure to transform the antenna positions from the ground plane to the shower plane is as follows. In essence, we want to perform the coordinate transformation as shown in equation 35.

$$\begin{array}{ccc} \text{ground plane coordinates} & \rightarrow & \text{shower plane coordinates} \\ \begin{pmatrix} x \\ y \\ z \end{pmatrix} & \rightarrow & \begin{pmatrix} \vec{v} \times \vec{B} \\ \vec{v} \times (\vec{v} \times \vec{B}) \\ \vec{v} \end{pmatrix} \end{array} \tag{35}$$

We start by defining an antenna position $\vec{A}_{ground}$ in the ground plane coordinates as shown in equation 36. The v-vector and B-vector are calculated in equation 37, where (θ, ϕ) are the zenith and azimuth angle. The geomagnetic field vector (equation 38) is calculated using the geomagnetic field strength, inclination and declination angle. Note that the v-vector is taken along the particles travelling direction, thus pointing inside the ground. Hence, the minus signs. Take into account that values of all angles are in the same units. Due to the inconsistency in the GRAND convention coordinates and the common used definition of inclination and declination angle, the formula 38 for the B-vector below is only valid for the GRAND convention.

$$\vec{A}_{ground} = \begin{pmatrix} A_x \\ A_y \\ A_z \end{pmatrix} \tag{36}$$

$$\vec{v}(\theta, \phi) = \begin{pmatrix} -\sin(\theta)\cos(\phi) \\ -\sin(\theta)\sin(\phi) \\ -\cos(\theta) \end{pmatrix} \tag{37}$$

$$\vec{B}(|B|, \iota, \delta) = \begin{pmatrix} |B|\ \sin(\iota)\ \cos(\delta) \\ -|B|\ \sin(\iota)\ \sin(\delta) \\ |B|\ \cos(\iota) \end{pmatrix} \tag{38}$$

The 'v cross B' and 'v cross v cross B' vectors are of course calculated with the cross product. Before we can perform the coordinate transformation correctly, all vectors should be normalised to obtain unit vectors (equation 39). The v-vector is already an unit vector by definition above.

$$\begin{aligned} \hat{e}_{\vec{v}\times\vec{B}} &= \frac{\vec{v} \times \vec{B}}{||\vec{v} \times \vec{B}||} \\ \hat{e}_{\vec{v}\times(\vec{v}\times\vec{B})} &= \frac{\vec{v} \times (\vec{v} \times \vec{B})}{||\vec{v} \times (\vec{v} \times \vec{B})||} \end{aligned} \tag{39}$$

Finally, the coordinate transformation is performed by taking the dot product of each unit

vector with the original antenna coordinates.

$$\begin{pmatrix} A_{\vec{v}\times\vec{B}} \\ A_{\vec{v}\times(\vec{v}\times\vec{B})} \\ A_{\vec{v}} \end{pmatrix} = \begin{pmatrix} \vec{A}_{ground} & \cdot & \hat{e}_{\vec{v}\times\vec{B}} \\ \vec{A}_{ground} & \cdot & \hat{e}_{\vec{v}\times(\vec{v}\times\vec{B})} \\ \vec{A}_{ground} & \cdot & \hat{e}_{\vec{v}} \end{pmatrix} \tag{40}$$

Hence, we obtain the antenna position in shower plane coordinates (equation 41).

$$\vec{A}_{shower} = \begin{pmatrix} A_{\vec{v}\times\vec{B}} \\ A_{\vec{v}\times(\vec{v}\times\vec{B})} \\ A_{\vec{v}} \end{pmatrix} \tag{41}$$

Antenna locations are circular symmetric in the Shower Plane, allowing a consistent comparison between events.

3.5 Angular Distribution of the Cherenkov-Ring

There are two effects visible in the angular distribution of the Cherenkov-Ring, which were discussed in the theory. First, we observe a maximum on the negative v-cross-v-cross-B axis for horizontal events. This is the Distance to X_{max} effect, which was explained in section 2 and is described by a negative sine term. Second, we observe a higher electric field on the right side of the shower plane caused by the interference between the emission mechanisms. This is a secondary effect for horizontal events, but dominant for vertical events. The exact turnover point was not determined in this thesis, as for the events studied in this thesis the distance to X_{max} effect was the dominant effect. The the formula of the angular distribution is shown again in equation 42. The factor 10^8 is used to avoid very small decimal numbers.

The angular distribution of the antenna plot in the previous subsection is shown in figure 15. It is visible that the Distance to Xmax effect is higher for the outer ring of antennas. In addition, we note that the inner ring of antennas has a higher electric field. The Interference effect is about equal.

$$\boxed{\frac{|\vec{E}_{antenna}|(\beta)}{N_{e^+e^-}} \times 10^8 = a\sin(\beta) + b\cos(\beta) + C} \tag{42}$$

The angular distribution is described in the shower plane. I therefore calculate the antenna angle in the shower plane β with equation 43 using the antenna coordinates in the shower plane. It should be mentioned that the antenna positions are placed with respect to the ground plane. Therefore, they do not form a perfect circle in the shower plane. The deviation is about 10% of the circular shape. This was not problematic for the analysis.

$$\beta(A) = \text{atan2}(A_{\vec{v}\times(\vec{v}\times\vec{B})}, A_{\vec{v}\times\vec{B}}) \tag{43}$$

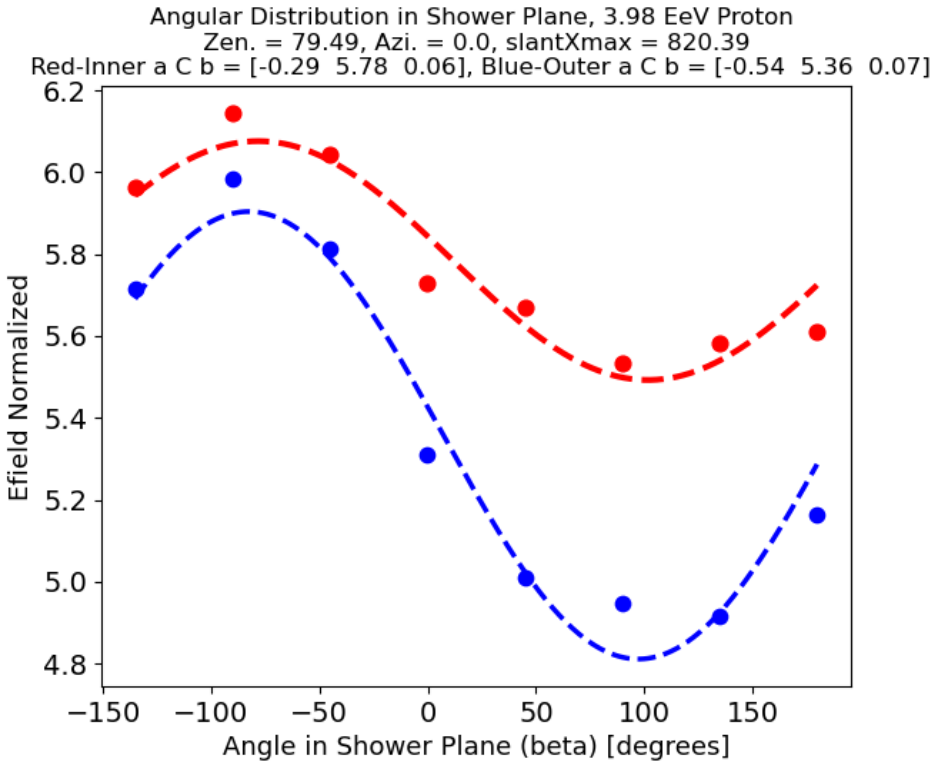

Figure 15: Angular Distribution of the antennas on the Cherenkov-Ring. I define the angle β to describe the angular distribution function. The parameters of the plot are shown in the title.

4 Method Part II: Cherenkov-Ring Energy Estimation

4.1 Signal Peak

I mainly analysed the electric field of antennas on the Cherenkov-Ring. The electric field on the ring is extracted from the trace of an antenna. The duration of the signal peak is about 1.5 ns, and this peak electric field strength is 50 - 100 times higher than the remainder of the signal. This means that extracting this short and high signal is possible, and it provides the information desired with minimal noise and interference. Typically, 2.5 ns is extracted from the trace. On figure 16, the eight antennas on the Cherenkov-Ring for one event are plotted on the shower plane. After selecting the antennas on the ring, I studied the dependence of data on the Cherenkov-Ring with respect to the event information. The main information I used are the zenith and azimuth angles, X_{max}, and number of electrons and positrons.

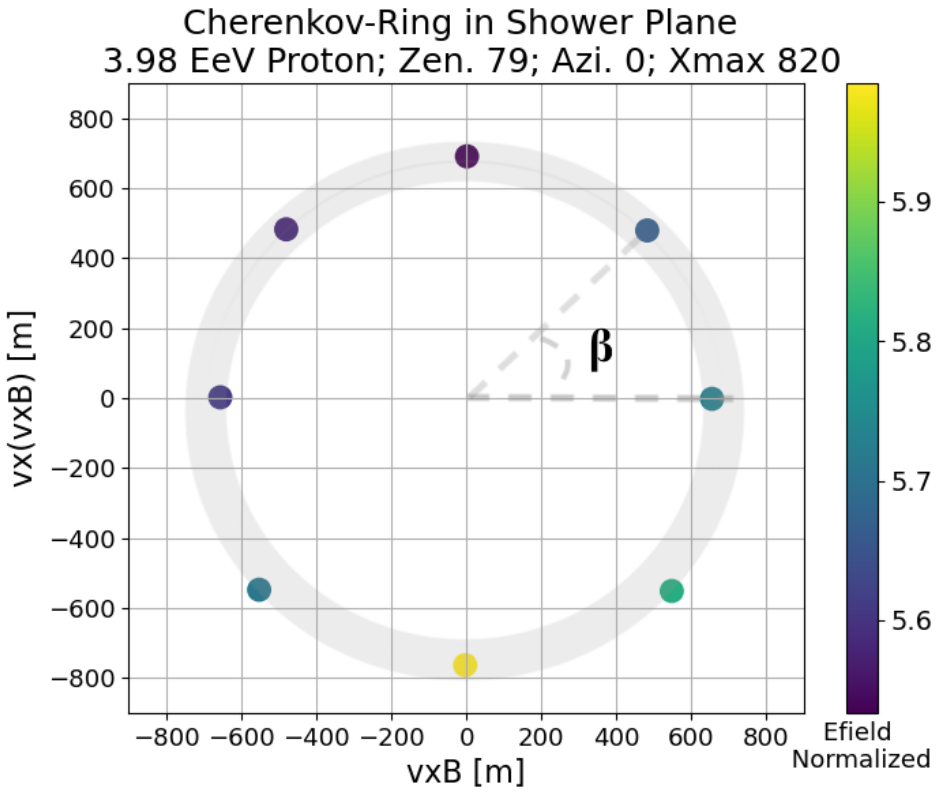

Figure 16: Antennas on the Cherenkov-Ring plotted on the shower plane with the normalised electric field height of the signal peak. These eight antennas are extracted for the analysis of all events. The angle β is used to describe the angular distribution of each event.

4.2 Energy Estimate Parameter C

The idea of CREE is that one antenna on the Cherenkov-Ring is sufficient to obtain the energy estimator if the shower geometry is known. However, we learned that the height of the electric field on the Cherenkov-Ring is not constant but follows an angular distribution. Therefore, I introduced the Energy Estimate Parameter C, which is a fixed (single) value for each event. Having C as a fixed value for each event, allows consistent comparison between all events in the library. The Energy Estimate Parameter is calculated using the angular distribution again given in equation 44. By taking the offset of the combined sine-cosine function, I obtain the average value of the angular distribution E_{ring}. The definition of C is the normalised value of E_{ring}, that is E_{ring} divided by $N_{e^+e^-}$ (equation 45). I chose to multiply the normalised electric field with a factor of 10^8 throughout my code for convenience.

$$\frac{|\vec{E}_{antenna}|(\beta)}{N_{e^+e^-}} \times 10^8 = a\sin(\beta) + b\cos(\beta) + C \tag{44}$$

$$\frac{|\vec{E}_{ring}|}{N_{e^+e^-}} \times 10^8 = C \tag{45}$$

The method thus consists of extracting the electric field peak magnitude of an antenna on the Cherenkov-Ring. Thereafter, the angular distribution allows to determine the average magnitude on the Cherenkov-Ring, a fixed value for each event: E_{ring}. So in reality, we must know the shower direction and core position in order to correct the electric field peak accordingly with the corresponding angular distribution. For the study of simulations, we always have a full ring of eight antennas, and therefore I was able to calculate the value of C for all events by using the fit function in Python. An example plot and fit is shown in figure 17. So, the Energy Estimate Parameter C is used to compare events and study the proportionality of the average field with respect to the size of the electromagnetic part of the air shower. Recall that $N_{e^+e^-}$ is the total number of electrons and positrons in the shower (equation 31).

In this thesis, I show that the value of C is constant for a fixed shower direction regardless the shower energy. So, this means that we prove a direct proportionality between the number of electrons and positrons. As a result, I aim to test and prove equation 46.

$$\boxed{|\vec{E}_{ring}| = C(\theta,\alpha) * \frac{N_{e^+e^-}}{10^8}} \tag{46}$$

The interpretation of C depends on your point of view:

- Offset of Angular Distribution of Cherenkov-Ring
- Average Normalised Electric field Peak Magnitude on the Cherenkov-Ring
- Gradient of Electric Field scaling with number of electrons and positrons

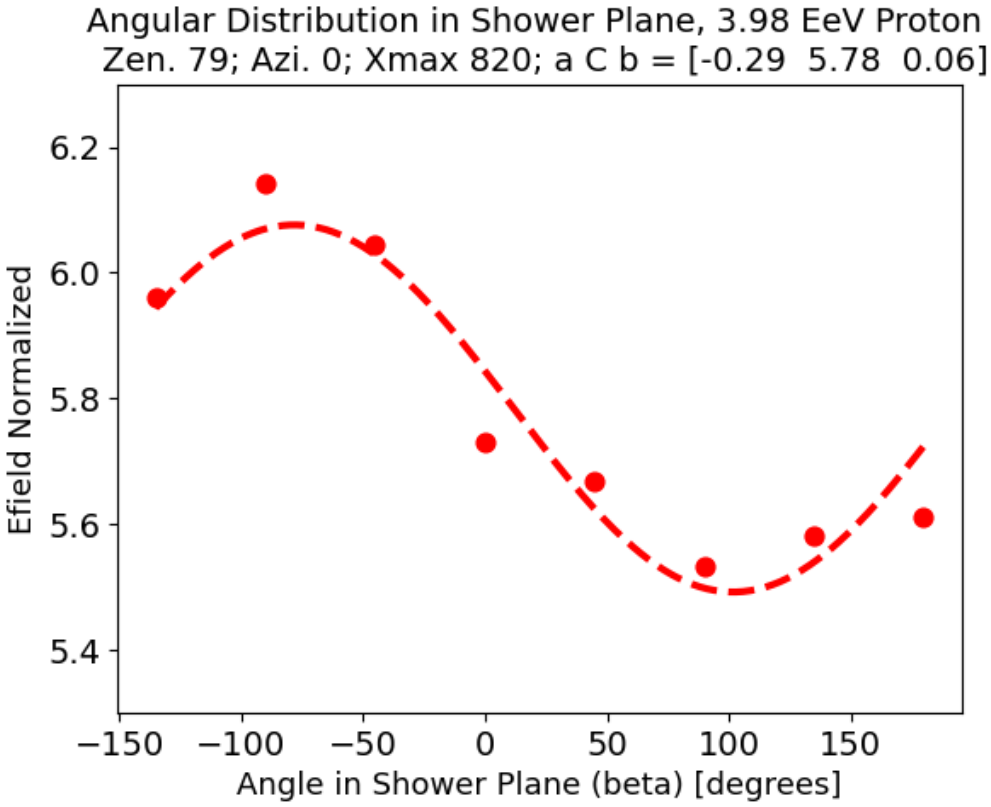

Figure 17: Angular Distribution of the antennas on the Cherenkov-Ring. I define the angle β to describe a function.

So, when performing CREE in real, first we correct the placement of the antenna with the angular distribution. Second, C is used to correct for the shower direction. The study of simulations allows us to calculate (estimate) the parameters. When we have $\vec{E}_{ring}$, the energy estimation is performed by using equation 47.

$$\boxed{\text{Energy}_{\text{EM}} \sim \frac{N_{e^+e^-}}{10^8} = \frac{|\vec{E}_{ring}|}{C(\theta,\alpha)}} \tag{47}$$

4.3 Zenith and Alpha Dependence of C

In the previous subsection, I aimed to prove that C is a constant for a fixed shower direction. Thus, we should study the dependence of C on shower direction in order to apply the method. At first, I aimed to correct the value of C for the direction using equation 48, which has an analogue for the electric field as was derived in the Theory Section. The shower direction is usually given by the zenith and azimuth angle. When studying radiation energy, it is common to study the dependence on zenith and alpha angle. The alpha angle is the angle between the shower axis and geomagnetic field vector (v-vector and B-vector). We use the

alpha angle, because the radiation emitted by the geomagnetic component scales as $\sin\alpha$. In the results section, I will note that this scaling stops for highly inclined showers. This is discussed later.

$$C_{corrected} = \frac{C(\theta,\alpha)}{X_{start}(\theta)\sin(\alpha)} \tag{48}$$

4.3.1 Zenith Angle

The fact that I uncovered the nontrivial scaling of radiation energy in highly inclined air showers, made that I could not construct the geometry correction. In this thesis, I therefore focus on describing the behaviour and dependence of C. In addition, I discuss the nontrivial scaling discovered in detail in my thesis. The zenith dependence is actually caused by the larger distance to X_{max} for more inclined air showers. Since I aim to construct an X_{max} independent energy estimator, I used the value of X_{start} to study the behaviour, which was explained in figure 12.

4.3.2 Alpha Angle

Let us first calculate α. First, we normalise the geomagnetic field vector in equation 49. The B-vector itself was already calculated in the coordinate transformation section. The v-vector was also calculated in that section and is a unit vector by definition.

$$\hat{e}_{\vec{B}} = \frac{\vec{B}}{||\vec{B}||} \tag{49}$$

Then, the cosine of an angle between two unit vectors is their dot product. The angle α is calculated by taking the arc cosine of the dot product. The dependence of the electric field strength scales with sine α and is therefore in general referred to as $\sin(\alpha)$. We may thus calculate $\sin(\alpha)$ with equation 50.

$$\begin{aligned} \cos\alpha &= \hat{e}_{\vec{v}} \cdot \hat{e}_{\vec{B}} \\ \alpha(\vec{v},\vec{B}) &= \arccos\left(\hat{e}_{\vec{v}} \cdot \hat{e}_{\vec{B}}\right) \\ \sin\alpha &= \sin\left(\arccos\left(\hat{e}_{\vec{v}} \cdot \hat{e}_{\vec{B}}\right)\right) \end{aligned} \tag{50}$$

5 Results

5.1 Relative Deviation of the Energy Estimate Parameter C

The value of the Energy Estimate Parameter C remains consistent in the energy range between 0.02 EeV and 3.98 EeV for fixed zenith angles in the range of 70 - 88 degrees. This is shown in figure 18. The relative deviation is 5% - 13% in the set of 0.02 EeV and 3.98 EeV events combined for fixed directions. In conclusion, the value of C depends mainly on the shower direction and its dependence on energy is tiny. The fact that the dependence of C on energy is tiny, confirms the direct proportionality between the electric field height on the Cherenkov-Ring and the number of electrons and positrons. Within a set of fixed energies and directions, the relative deviation reduces to 1.5% - 2.5% only. In that case, the margin is only caused the by secondary dependence on X_{max}, which was neglected in this analysis.

The main dependence of C is thus on zenith angle and thereafter on azimuth angle. The dependence on zenith angle is explained by the larger distance to X_{max} for more inclined events. The dependency on azimuth angle should be explained by the sine alpha scaling of the geomagnetic emission. However, I discovered that the dependence on sine alpha becomes smaller (almost indistinguishable) and nontrivial for highly inclined events. Therefore, I could not finalise the correction for shower direction within my Master Project as the behaviour of radiation energy apparently becomes nontrivial for highly inclined events. I present my results on this in the next subsection.

Within the zenith angle range analysed, we see that the relative deviation of C remains small between 5% and 13%. The uncertainty is plotted in figure 19. This is the relative deviation between the values of all events of both primary energies analysed. The relative deviation increases with the zenith angle. Even though the standard deviation does not increase, the signal strength weakens, and as a result the relative deviation increases.

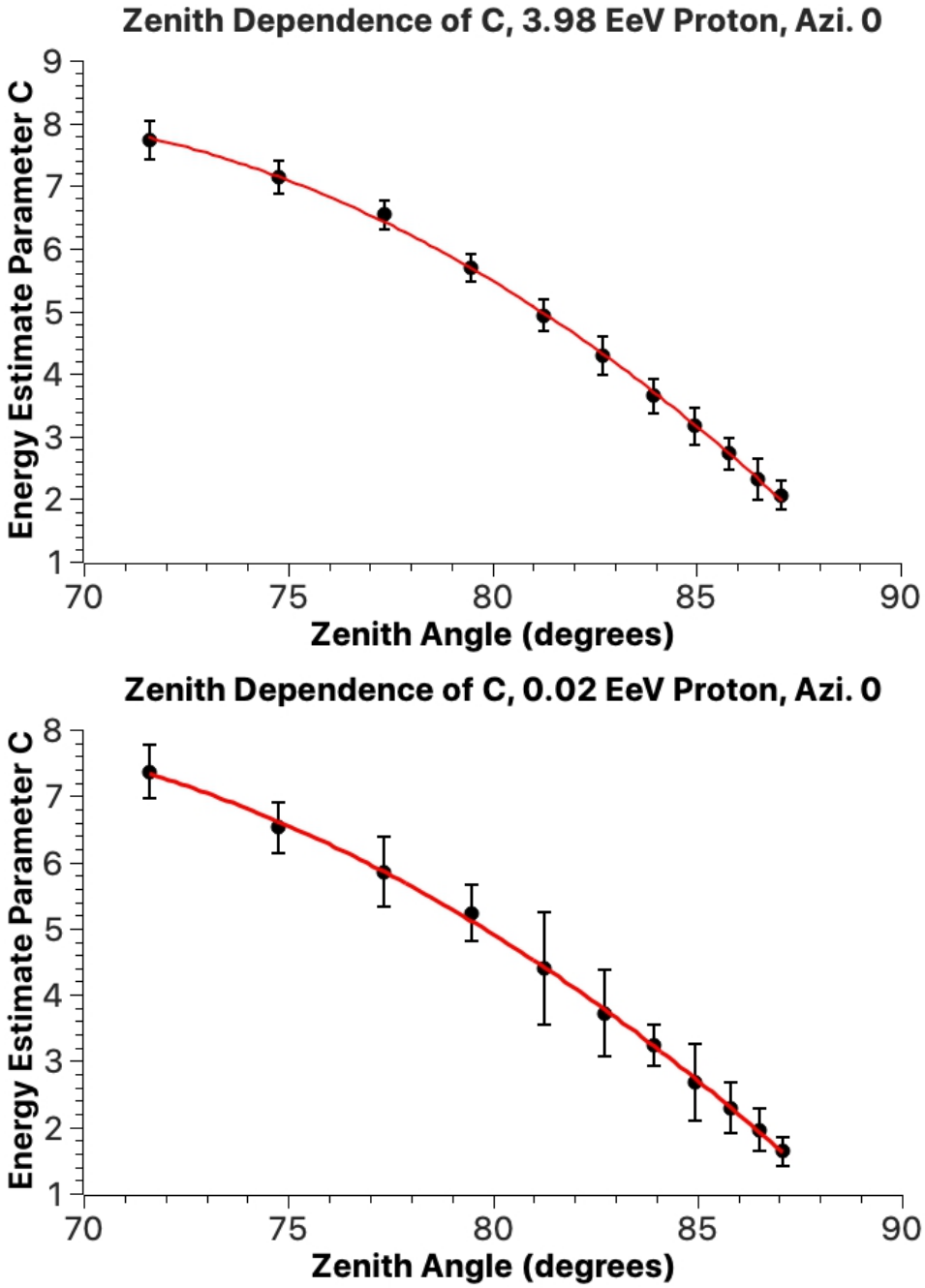

Figure 18: Comparison between the value of the Energy Estimate Parameter for a 3.98 EeV and 0.02 EeV events. Both sets contain 222 events. One can see that the value remains consistent in this large energy range, even as the scaling behaviour. The dependency of the Energy Estimate Parameter on Zenith Angle aligns with the theory: For a higher zenith angle, the distance to the emission region is longer resulting in a weaker signal.
Fit above: $C_{3.98EeV}(\theta) = -0.01442 \pm 0.0008\,\theta^2 + 1.915 \pm 0.135\,\theta - 55.44 \pm 5.37$ $[R^2 = 0.999]$
Fit below: $C_{0.02EeV}(\theta) = -0.01146 \pm 0.0009\,\theta^2 + 1.442 \pm 0.147\,\theta - 37.38 \pm 5.86$ $[R^2 = 0.999]$

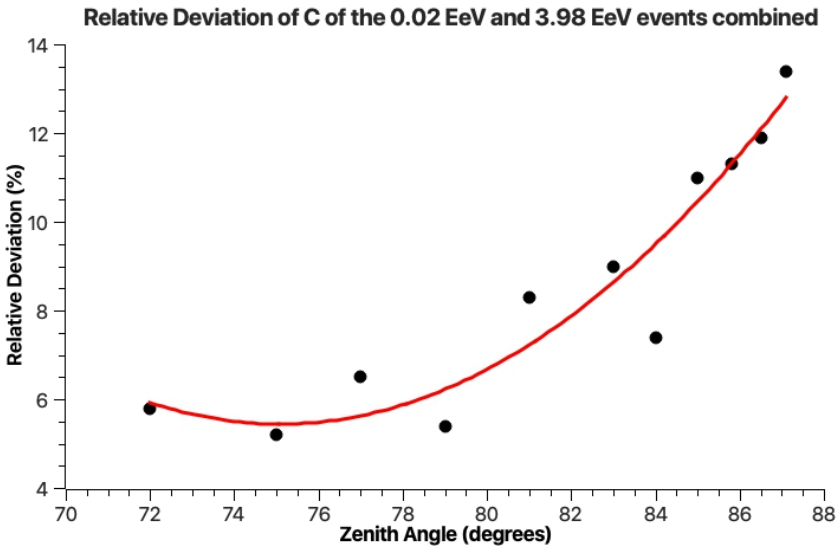

Figure 19: Relative deviation of C within the combined set of 0.02 and 3.98 EeV events (726 events). Fit: $RelDev_C(\theta) = -0.051 \pm 0.015\,\theta^2 - 7.66 \pm 2.42\,\theta - 0.051 \pm 0.015$ $[R^2 = 0.991]$

5.2 Nontrivial scaling of the Radiation Energy

The scaling of the radiation energy becomes nontrivial for highly inclined air showers. This result followed after analysing the Energy Estimate Parameter for events with different azimuth angles. For zenith angles beyond 80 degrees, there is no clear sine alpha scaling anymore for the value of the Energy Estimate Parameter. Figure 20 is illustrative. On the plot, it is visible that the value of C merges together for events with different alpha (azimuth) angles and is not clearly separated anymore. This indicates that the behaviour of the underlying emission mechanisms becomes nontrivial in highly inclined air showers. In particular, I observe an extra steep drop in radiation energy for the combined limit of Zenith → 90° and Sine Alpha → 1. For this reason, I was not able the correct radiation energy for shower direction with the known formulae.

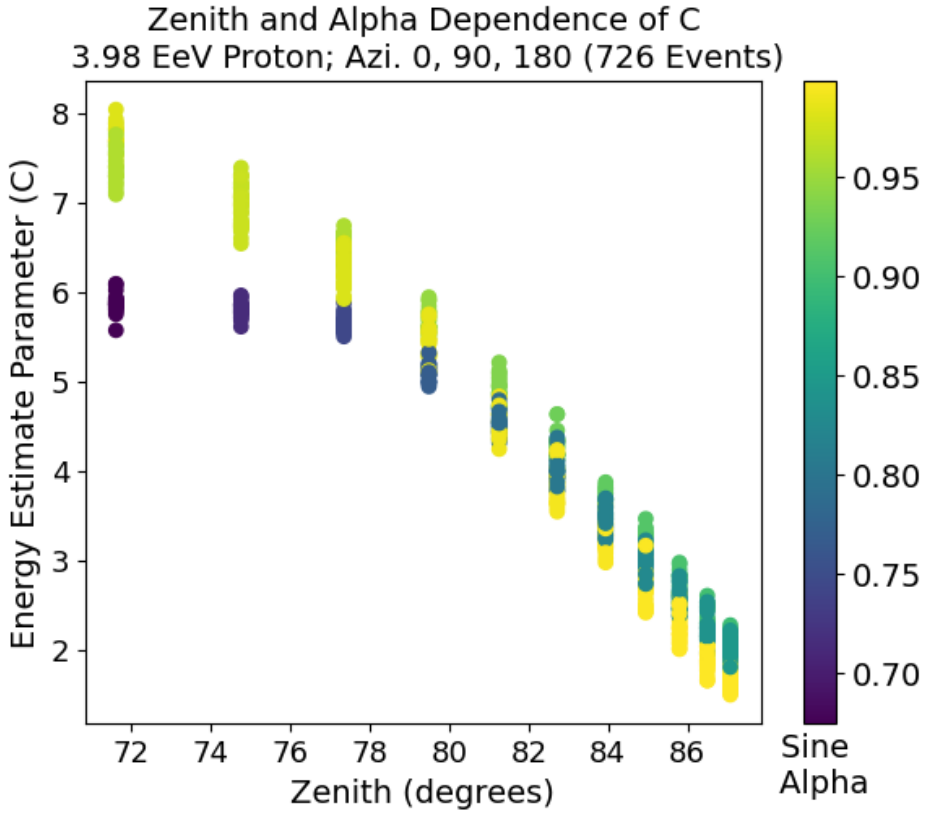

Figure 20: Value of the Energy Estimate Parameter for all Proton 3.98 EeV events in the library. After zenith 80 degrees, the value of the different azimuth angles merge together.

In figure 21, I noted the tipping points in the highly inclined events. They indicated that the linear scaling in sine alpha stops. In addition, the value of C for all events of azimuth 90° was lower than those of 0° and 180°, while the alpha angle corresponding to azimuth 90° is the highest. So, an extra steep drop in radiation energy for the combined limit of zenith → 90° and Sine Alpha → 1 is observed.

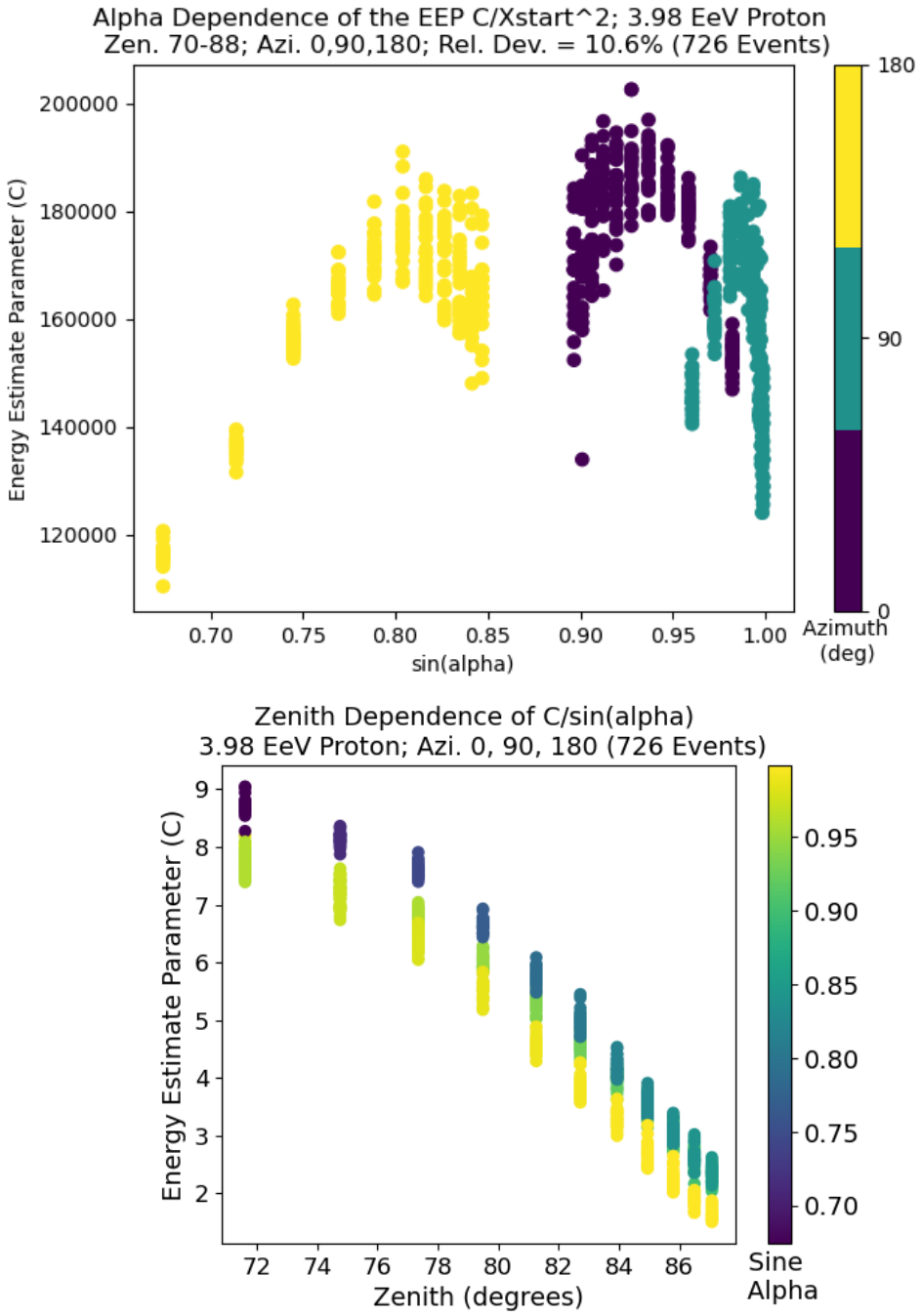

Figure 21: Above: C/X_{start} (C normalised with X_{start}) plotted against $\sin(\alpha)$. *We see the fall off of the radiation energy and typical tipping points in the limit* $\sin(\alpha) \rightarrow 1$.
Below: C/$\sin(\alpha)$ *(C normalised with* $\sin(\alpha)$*). The deviation, especially for the events above 80°, increases when normalising with* $\sin(\alpha)$. *This indicates that there is no more scaling in sine alpha anymore.*

In order to check whether the effects observed were not a fault in the simulated files, I used the set of zenith 38° to perform a short cross-check. In those vertical events, the scaling in sine alpha was observed as expected. So, the simulation seemed to work correctly indicating there was no systematic error in the simulated files.

5.2.1 Lower Efficiency of Geomagnetic Component

My results on the nontrivial scaling of the radiation energy align with presentations by Simon Chiche and Chao Zhang on the GRAND collaboration meeting in December 2021. [16] [17] They had similar results in their studies of the simulated air showers. Therefore, we are confident in presenting the result of the nontrivial scaling. The results indicate that it is caused by a drop in the geomagnetic emission.

The geomagnetic emission could become becomes less efficient in highly inclined showers, which develop in lower density atmosphere. This was concluded by combining the plots of the Energy Estimate Parameter and the parameter of the interference effect parameter of the angular distribution. In figure 22, the value of the interference effect parameter is plotted versus the zenith angle. It shows that the effect on the angular distribution vanishes in the limit of Zenith → 90°. The combination of this fact with the drop in radiation energy indicates that the geomagnetic component becomes less efficient in highly inclined air showers, which develop in lower density atmosphere.

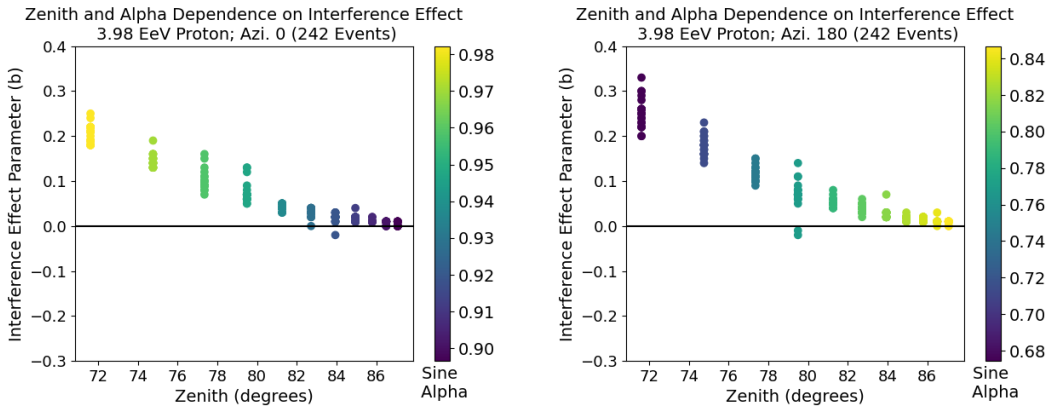

Figure 22: Value of the parameter describing the interference effect in the Angular Distribution, plotted for the shower directions analysed.

The conclusion is strengthened by the fact that I observe an extra steep drop in the radiation energy for the Combined limit of Zenith → 90° and Sine Alpha → 1. It indicates that there is indeed a difference in behaviour in the geomagnetic component, because it would usually be the strongest in the limit of Sine Alpha → 1. A possible explanation, which was shared by Harm Schoorlemmer, is the following: The thought is that the electron and positron drift is at its highest efficiency in this limit. At the same time, the geomagnetic emission mechanism is at its lowest efficiency. As a result, this causes particle (energy) loss from the shower, because

particles are lost (drifted far away) from the shower axis along the long trajectory.

Another deviation in behaviour is visible in figure 23, on which the interference effect parameter of the azimuth 90° events is plotted. These are again the events in the combined limit of Zenith → 90 and Sine Alpha → 1. In addition to the drop in radiation energy at the turnover point between Zenith 77° and 82°, a strong difference in the zenith distribution of the events with Azimuth 90° is observed. This implies that the physical emission mechanism is indeed behaving very differently in this special limit. In addition, there is another effect occurring around the bump of zenith 80°. After around zenith 80°, the inside and outside Cherenkov-Ring are flipped. This effect was found by a simulation of Harm Schoorlemmer.

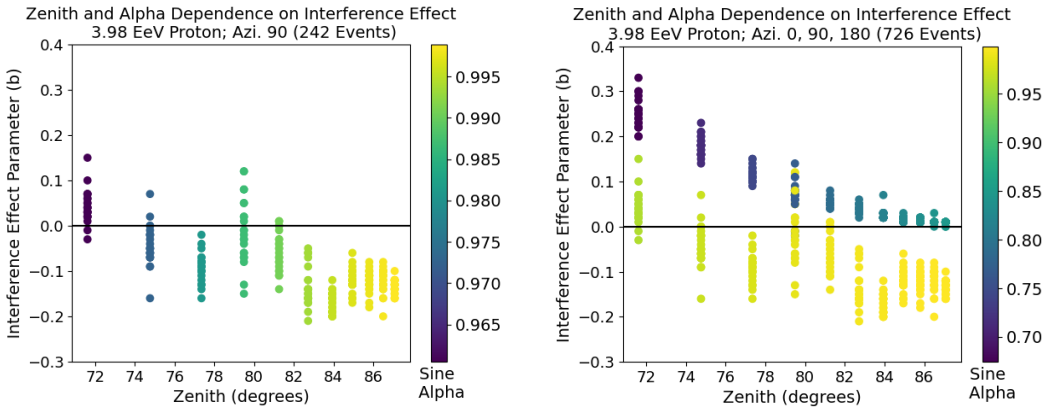

Figure 23: Left: The same plot as in figure 22, but now for azimuth 90°. Right: Combined plot of all three angles for comparison.

All in all, there are several nontrivial effects occurring in highly inclined air showers that have an impact on their radio footprint.

5.3 Problem Solved in the hadronic part of AIRES

There was a huge difference present, up to factor three, in the height of the electric field for similar events with equal direction, primary energy and X_{max}. This was not only the case on the Cherenkov-Ring, but on the entire radio foot print. Interestingly, the shape of the radio foot print was not affected at all. This means that there was an unrealistically large fluctuation in the radiation emitted. Therefore, I compared the number of electrons and positrons between similar events. These are after all the emitters of radiation. It followed that the number of electrons and positrons differed up to factor three between events of similar energy and direction. The electromagnetic component was thus significantly lower in about 35% of the events in the library. This could have been a problem for my research, but using the total number of electrons and positrons as a normalisation factor helped to work around it. Matias Tueros further investigated the case in the meantime.

It was concluded that the difference in electric field height was caused by a deficiency in the hadronic part of the simulation. It was caused by high energetic rare hadrons with a lifetime on the threshold of the duration of the air shower development. These high energetic hadrons, formed in the beginning of the air shower, had an energy up to half of the primary particle's energy. These hadrons were not handled by the simulation and travelled through the entire shower unaffected, taking half of the shower energy into the ground.

The hadronic component in air showers is simulated with complex Monte Carlo codes. All types of hadrons are produced in energetic hadronic interactions, and the hadrons created should be handled by the simulation appropriately. Unstable hadrons have a large range of lifetimes that can vary between 10^{-24} and 10^{-6} seconds. Hence, there are various approaches for the handling of rare hadrons and different hadronic simulation models exist. For example, a hadron could be forced to decay or to interact. The hadronic model used in our analysis is part of the AIRES code. An example of another hadronic model is CORSIKA. I discovered that the hadronic model used in simulations influences the height of the Electric field in the radio footprint.

The AIRES developers worked on resolving the bug. It turned out that it was introduced in AIRES version 19.04.00 by accident. Matias Tueros introduced AIRES version update 19.04.08 on the GRAND collaboration meeting in Paris in December 2021:

Matias Tueros: *„AIRES 19.04.00 had disabled the forced decay of short lived hadrons in post-LHC hadronic models. Short lived hadrons were propagated, but some were not given the possibility to have a new hadronic interaction. Most of these short lived hadrons decay immediately anyway (with lifetimes of 10-12 ns), except for example Sigmas, that have lifetimes of 0.8 ns. When by chance a very energetic sigma was generated, the decay mean free path could become very large and the particle could reach ground. Of all hadronic models, Sybill 2.3c is the one producing more of these events. AIRES 19.04.08 forces the decay of all short lived hadrons: problem solved."* [18]

6 Conclusion

My estimator for the of electromagnetic energy in air showers proved to be a successful tool. At first, my analysis led to the discovery of the problem in AIRES discussed above and consequently to version update 19.04.08. This supported the hypothesis that the radiation pulse on the Cherenkov-Ring is a good estimator for the total radiation energy. At a later stage of my research, I discovered, independently from the other researchers of the GRAND collaboration, that the radiation energy becomes nontrivial in highly inclined air showers. This was discovered thanks to the small resolution of the Cherenkov-Ring Energy Estimation. All in all, the radiation pulse on the Cherenkov-Ring proves to be a good estimator for the total radiation energy in inclined air showers.

6.1 Cherenkov-Ring Energy Estimation

One antenna on the Cherenkov-Ring estimates the number of electrons and positrons in the shower, provided the shower direction and shower core are known. The resolution is 5 - 13% for fixed directions. The conclusion is valid within the energy range of 0.02 EeV and 3.98 EeV and the zenith angle range of 70 - 88 degrees. Within a fixed direction and energy in this range, the resolution reduces to 1.5 - 2.5%. In conclusion, the Cherenkov-Ring is an estimator for the size of the electromagnetic part of the air shower.

6.2 Nontrivial Scaling of the Radiation Energy

The scaling of the radiation energy becomes nontrivial for highly inclined air showers, starting from a zenith angle of 77 degrees. The known correction formulae for radiation energy are therefore invalid for this class of events. In addition, the angular distribution of the radio footprint differs indicating that the ratio between the emission mechanisms changes. The results indicate that the geomagnetic component becomes less efficient in lower density atmosphere causing a drop in radiation energy in the (combined) limit of Zenith $\rightarrow 90°$ and Sine Alpha $\rightarrow 1$.

7 Discussion

7.1 Cherenkov-Ring Energy Estimation

Overall, Cherenkov-Ring Energy Estimation is promising thanks to the following three reasons. First of all, one antenna in the radio footprint allows us to obtain a reliable estimator for the electromagnetic energy. At the same time, this energy estimator provides a simple and clear comparison factor between different events. Secondly, two or three sampling times (bins) are sufficient to extract the electric field peak and calculate the energy estimator. A high signal to noise ratio is thus obtained, and only a tiny amount of data is necessary. Thirdly, the method has solid theoretical support from a physical point of view. All radio signals arrive instantaneously on the basis of the Cherenkov-effect and form an integral. So, the low relative deviation achieved from the analysis of simulated events and these three arguments form a basis for further research.

The next step for further research is the analysis of the method in the 50 - 200 MHz frequency range using reconstructed events. Reconstructed events are based on the simulated events and include a digital filter, similar to the filter in the real detector. I performed a preliminary analysis of reconstructed events with a primary energy of 3.98 EeV and a zenith angle of 79.5°. In these specific events, the electric field peak was wider, and a periodic fluctuation was visible caused by the periodic character of the Fourier transform. Further analysis should include a correction for these two effects. Fellow Master Student Inge van Rengs analysed the Fourier spectrum in more detail and concluded that the Fourier spectrum is flat on the Cherenkov-Ring. This is a specific character of an antenna on the Cherenkov-Ring in the frequency domain and could also be utilised. All in all, we expect the method is applicable in the GRAND frequency range and further research and development of the method is meaningful.

In a later stadium, challenges to implement the method on real data will come across. One point for further research is the relation between the electromagnetic energy of the air shower and the primary energy of the incident particle. So far, I relied on the proportionality between the electric field and the electromagnetic part of the shower. It was discussed in my thesis that I chose to probe this relation due to an deficiency in the hadronic part of the simulations used. In fact, it is unknown which is the most realistic simulation of the hadronic component in an air shower simulation. Consequently, probing the conversion factor between electromagnetic energy and primary energy is important to make the method applicable on real data.

The results of my research were presented on the international GRAND collaboration Meeting in Paris in December 2021. It was an honour to present my events there. To conclude, I am satisfied with the results of my analysis and would be honoured if the method invites further research and may even be implemented on the real detector in the future.

7.2 Nontrivial Scaling of the Radiation Energy

Further Research on both radiation energy scaling and energy estimation is of vital importance for GRAND, as it will focus on horizontal (highly inclined) air showers and aims to identify air showers in the specific EeV energy range. In addition, the strong signal on the Cherenkov-Ring may become extra important, since the general radio footprint is weaker due to the geomagnetic drop. It was concluded that the (low) air density influences the amount of radiation energy emitted by highly inclined air showers. The difference in radiation energy and radio footprint may be used to distinguish tau-neutrino induced showers exiting the mountain from horizontal cosmic rays. The shower development of both takes place in different atmospheric conditions, which would thus led to different radio footprints.

8 Attachment: Code Cherenkov-ring

```
## Masterstage Henk Brans
import h5py as hdf
import numpy as np
import os
from pathlib import Path
import math as m
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
from numpy import sin, cos, tan, sqrt, array, radians, degrees, cross
from numpy.linalg import norm
from itertools import groupby
from operator import itemgetter

#enter path to directory here:
directory = r'/Users/h.brans/Documents/Simulation Data'
#directory = r'/vol/auger6/GRAND/simulations/Matias_new'
#choose variables for selecting functions here:
RHVthreshold = 0.8
MEANthreshold = 2.
TRACEthreshold = 0.5
PEAKWIDTHthreshold = 6 #one step is equal to 0.5 ns

#list of simulated files is created:
files_list = array([entry.path for entry in os.scandir(directory) if entry.
    path.endswith('.hdf5')  ])

#plot electric field of one antenna
def plot_efield(filename, antenna):
    efieldtrace = extract_efieldtrace(filename, antenna)
    efield = [list[1] for list in efieldtrace]
    x = [list[0]+13800 for list in efieldtrace]
    plt.plot(x, efield, linewidth=3)
    plt.title('Simulated Efield Trace \n Antenna on Cherenkov-Ring',
    fontsize = 18)
    plt.yticks(fontsize = 15)
    plt.xticks(fontsize = 15)
    plt.xlabel('Time [ns]', fontsize = 15)
    plt.ylabel('Efield Magnitude [uV/m]',fontsize=15 )
    plt.show()

#extract P2Pefield from the HDF5 file
def extract_P2Pefield(filename):
    f = hdf.File(filename, 'r')
    groupname = Path(filename).stem
    group1 = f.get(groupname)
    dataset1 = array( group1.get('AntennaP2PInfo') )
    P2Pefield_list = (tuple[1] for tuple in dataset1)
    P2Pantennas_list = list( range(0, 160) )#no crosscheck antennas
```

```
47      P2Pefield_list = tuple(map (list, zip(P2Pantennas_list, P2Pefield_list)
        ) )
48      return P2Pefield_list #list of [antenna, P2Pvalue]
49
50  #sorts a 2-dim list on the 1st column X: [0, X]
51  def sort_list(list):
52      list_sorted = sorted(list, reverse=True, key=lambda x:x[1])
53      return list_sorted
54
55  #find High Values from Mean (HVM)
56  def find_HVM(P2Pefield_list_sorted):
57      #calculate the mean P2Pefield of all antennas
58      mean = np.mean(P2Pefield_list_sorted, axis = 0)[1]
59      #threshold is set
60      threshold2 = MEANthreshold * mean
61      k = 0
62      #the P2Pefield list is sorted from high to low, so we wait until the
        antenna number that does not pass the threshold
63      while P2Pefield_list_sorted[k][1] > threshold2:
64          k += 1
65      #keep first k elements, which passed the threshold
66      HVM_list = P2Pefield_list_sorted[:k]
67      return HVM_list
68
69  #extract trace of efield for an antenna from the HDF5 file
70  def extract_efieldtrace(filename, antenna):
71      f = hdf.File(filename, 'r')
72      if antenna < 160:
73          antennaname = 'A{}'.format(antenna)
74      else:
75          antennaname = 'CrossCheckA{}'.format(antenna)
76      groupname = Path(filename).stem
77      group1 = f.get( (groupname) )
78      group2 = group1.get('AntennaTraces')
79      group3 = group2.get( '/{}/AntennaTraces/{}'.format(groupname,
        antennaname) )
80      dataset3 = np.array( group3.get('efield') )
81      times = array([tuple[0] for tuple in dataset3])
82      magnitude_list = array([norm((item[1], item[2], item[3])) for item in
        dataset3])
83      efieldtrace_list = tuple(map (list, zip(times, magnitude_list) ) )
84      return efieldtrace_list #list of [time, magnitude of efield]
85
86  #extract trace of efield for an antenna in components and sums the absolute
        value of the entire efield trace
87  def extractsum_efieldtrace3(filename, antenna):
88      f = hdf.File(filename, 'r')
89      if antenna < 160:
90          antennaname = 'A{}'.format(antenna)
91      else:
92          antennaname = 'CrossCheckA{}'.format(antenna)
```

```
93      groupname = Path(filename).stem
94      group1 = f.get( (groupname) )
95      group2 = group1.get('AntennaTraces')
96      group3 = group2.get( '/{}/AntennaTraces/{}'.format(groupname,
        antennaname) )
97      dataset3 = list( group3.get('efield') )
98      times = [tuple[0] for tuple in dataset3]
99      Ex = [tuple[1] for tuple in dataset3]
100     Ey = [tuple[2] for tuple in dataset3]
101     Ez = [tuple[3] for tuple in dataset3]
102     E_NS = sum(np.abs(Ex))
103     E_EW = sum(np.abs(Ey))
104     E_V = sum(np.abs(Ez))
105     return E_NS, E_EW, E_V
106
107 #extract trace of efield for an antenna in components and sums the absolute
        value of the electric field peak only
108 def extractsum_efieldtrace3peak(filename, antenna):
109     f = hdf.File(filename, 'r')
110     if antenna < 160:
111         antennaname = 'A{}'.format(antenna)
112     else:
113         antennaname = 'CrossCheckA{}'.format(antenna)
114     groupname = Path(filename).stem
115     group1 = f.get( (groupname) )
116     group2 = group1.get('AntennaTraces')
117     group3 = group2.get( '/{}/AntennaTraces/{}'.format(groupname,
        antennaname) )
118     dataset3 = array( group3.get('efield') )
119     efieldtrace = extract_efieldtrace(filename, antenna)
120     efieldonly = [list[1] for list in efieldtrace]
121     highest = max(efieldonly)
122     highestindex = efieldonly.index(highest)
123     times = [tuple[0] for tuple in dataset3]
124     Ex = [tuple[1] for tuple in dataset3]
125     Ey = [tuple[2] for tuple in dataset3]
126     Ez = [tuple[3] for tuple in dataset3]
127     E_NS = (Ex[highestindex-2] + Ex[highestindex-1] + Ex[highestindex] + Ex
        [highestindex+1] + Ex[highestindex+2])*0.5
128     E_EW = (Ey[highestindex-2]+ Ey[highestindex-1] + Ey[highestindex] + Ey[
        highestindex+1] + Ey[highestindex+2] )*0.5
129     E_V = (Ez[highestindex-2] + Ez[highestindex-1] + Ez[highestindex] + Ez[
        highestindex+1] + Ez[highestindex+2])*0.5
130     return E_NS, E_EW, E_V
131
132 #squared Efield
133 def extractsum_efieldtrace4peak(filename, antenna):
134     f = hdf.File(filename, 'r')
135     if antenna < 160:
136         antennaname = 'A{}'.format(antenna)
137     else:
```

```
138         antennaname = 'CrossCheckA{}'.format(antenna)
139     groupname = Path(filename).stem
140     group1 = f.get( (groupname) )
141     group2 = group1.get('AntennaTraces')
142     group3 = group2.get( '/{}/AntennaTraces/{}'.format(groupname,
        antennaname) )
143     dataset3 = array( group3.get('efield') )
144     efieldtrace = extract_efieldtrace(filename, antenna)
145     efieldonly = [list[1] for list in efieldtrace]
146     highest = max(efieldonly)
147     highestindex = efieldonly.index(highest)
148     times = [tuple[0] for tuple in dataset3]
149     Ex = [tuple[1] for tuple in dataset3]
150     Ey = [tuple[2] for tuple in dataset3]
151     Ez = [tuple[3] for tuple in dataset3]
152     E_NS = (Ex[highestindex-2]**2 + Ex[highestindex-1]**2 + Ex[highestindex
        ]**2 + Ex[highestindex+1]**2 + Ex[highestindex+2]**2)*0.5
153     E_EW = (Ey[highestindex-2]**2+ Ey[highestindex-1]**2 + Ey[highestindex
        ]**2 + Ey[highestindex+1]**2 + Ey[highestindex+2]**2 )*0.5
154     E_V = (Ez[highestindex-2]**2 + Ez[highestindex-1]**2 + Ez[highestindex
        ]**2 + Ez[highestindex+1]**2 + Ez[highestindex+2]**2)*0.5
155     return E_NS, E_EW, E_V
156
157 #sorts a 2-dim list on column [X, 0]
158 def sort_list_zeroeth(list):
159     list_sorted = sorted(list, key=lambda x:x[0])
160     return list_sorted
161
162 #find peak of the electric field trace
163 def find_peak(efieldtrace_list):
164     highest = max(efieldtrace_list, key = lambda x:x[1])
165     highest_efield = highest[1]
166     highest_time = highest[0]
167     highest_index = efieldtrace_list.index(highest)
168     peakleft = []
169     peakright = []
170     peak = []
171     n = highest_index
172     p = highest_index
173     while efieldtrace_list[n][1] > TRACEthreshold * highest_efield:
174         peakright.append(efieldtrace_list[n])
175         n += 1
176     while efieldtrace_list[p][1] > TRACEthreshold * highest_efield:
177         peakleft.append(efieldtrace_list[p])
178         p -= 1
179     peak.extend(peakleft)
180     peak.append(highest)
181     peak.extend(peakright)
182     return peak #list of [time (ns), magnitude of efield (uV/m)]
183
184 #check peakwidth of selected antenna
```

```
185 def check_antenna(filename, antenna):
186     efieldtrace = extract_efieldtrace(filename, antenna) #
        extract_efieldtrace2 with own files
187     peak = find_peak(efieldtrace)
188     if len(peak) < PEAKWIDTHthreshold:
189         return True
190     else:
191         return False
192 
193 #find antennas on Cherenkov-cone
194 def find_cone(filename):
195     P2Pefield_list = extract_P2Pefield(filename)
196     P2Pefield_list_sorted = sort_list(P2Pefield_list)
197     HVM_list = find_HVM(P2Pefield_list_sorted)
198     cone_list = []
199     for item in HVM_list:
200         if check_antenna(filename, item[0]) == True:
201             cone_list.append(item)
202         else:
203             pass
204     return cone_list #list of antenna numbers, P2P
205 
206 #extract XYpositions of antennas
207 def extract_positions(filename):
208     f = hdf.File(filename, 'r')
209     groupname = Path(filename).stem
210     group1 = f.get(groupname)
211     dataset4 = array( group1.get('AntennaInfo') )
212     Xposition = [tuple[1] for tuple in dataset4]
213     Yposition = [tuple[2] for tuple in dataset4]
214     XYposition = tuple(map (list, zip(Xposition, Yposition) ) )
215     return XYposition #list of [X, Y] for all antennas A0 - A175
216 
217 def extract_azimuth(filename):
218     f = hdf.File(filename, 'r')
219     runinfo = array(f.get('RunInfo'))
220     azimuth = runinfo[0][5]
221     return azimuth #azimuth angle in degrees of the event
222 
223 def extract_zenith(filename):
224     f = hdf.File(filename, 'r')
225     runinfo = array(f.get('RunInfo'))
226     zenith = runinfo[0][4]
227     return zenith #zenith angle in degrees
228 
229 def extract_Xmaxdistance(filename):
230     f = hdf.File(filename, 'r')
231     runinfo = array(f.get('RunInfo'))
232     Xmaxdistance = runinfo[0][6]
233     return Xmaxdistance #Slant Xmax in g / cm2
234 
```

```
235 def extract_Xmax(filename):
236     f = hdf.File(filename, 'r')
237     runinfo = array(f.get('RunInfo'))
238     Xmax = runinfo[0][7]
239     return Xmax #Slant Xmax in g / cm2
240 
241 def extract_Bearth(filename):
242     f = hdf.File(filename, 'r')
243     groupname = Path(filename).stem
244     group1 = f.get( (groupname) )
245     eventinfo = array(group1.get('EventInfo'))
246     B = eventinfo[0][16]
247     Bincl = radians(90 + eventinfo[0][17])
248     Bdecl = radians(eventinfo[0][18])
249     return B, Bincl, Bdecl #Bincl, Bdecl in degrees
250 
251 def extract_Eprimary(filename):
252     f = hdf.File(filename, 'r')
253     groupname = Path(filename).stem
254     group1 = f.get( (groupname) )
255     eventinfo = array(group1.get('EventInfo'))
256     Eprimary = eventinfo[0][3]
257     return Eprimary #Eprimary in EeV
258 
259 def extract_primary(filename):
260     f = hdf.File(filename, 'r')
261     groupname = Path(filename).stem
262     group1 = f.get( (groupname) )
263     eventinfo = array(group1.get('EventInfo'))
264     primary = eventinfo[0][2]
265     return primary #primary particle string
266 
267 def extract_eplusminus(filename): #longitidunal distribution
268     f = hdf.File(filename, 'r')
269     groupname = Path(filename).stem
270     group1 = f.get( (groupname) )
271     group7 = group1.get('ShowerTables')
272     dataset5 = array( group7.get('NLongitudinalProfile') )
273     eplusminus = array([tuple[3] for tuple in dataset5])
274     #columndepth = array([tuple[0] for tuple in dataset5])#SlantDepth g/cm2
275     #delta_columndepth = [columndepth[i + 1] - columndepth[i] for i in
    range(len(columndepth) - 1)]
276     return eplusminus # list (delta columndepth, number of particles)
277 
278 def sum_eplusminus(filename): #sum of eplus_eminus
279     f = hdf.File(filename, 'r')
280     groupname = Path(filename).stem
281     group1 = f.get( (groupname) )
282     group7 = group1.get('ShowerTables')
283     dataset5 = array( group7.get('NLongitudinalProfile') )
284     eplusminus = array([tuple[3] for tuple in dataset5])
```

```
285     return sum(eplusminus)
286 
287 def calculate_azimuthantenna(filename, antenna):
288     azimuth = abs(extract_azimuth(filename) - 180)
289     XYposition = extract_positions(filename)[antenna]
290     angle = m.atan2(XYposition[1], XYposition[0])
291     azimuthantenna = m.degrees(angle) - azimuth
292     if azimuthantenna < -180:
293         azimuthantenna += 360
294     return azimuthantenna #antenna azimuth angle in degrees
295 
296 def calculate_azimuthshowerplane(vxB, vxvxB):
297     angle = m.atan2(vxB, vxvxB)
298     azimuthantenna = m.degrees(angle) - azimuth
299     if azimuthantenna < -180:
300         azimuthantenna += 360
301     return azimuthantenna #antenna azimuth angle in degrees
302 
303 #sum the whole efieldtrace
304 def sum_efieldtrace(filename, antenna):
305     efieldtrace = extract_efieldtrace(filename, antenna)
306     efieldonly = [list[1] for list in efieldtrace]
307     efieldsum = sum(efieldonly) * 0.5
308     return efieldsum #sum of efield in uV*ns/m
309 
310 #sum the peak efieldtrace with known antenna on cone, herziene versie
311 def sum_efieldpeak2(filename, antenna):
312     efieldtrace = extract_efieldtrace(filename, antenna)
313     efieldonly = [list[1] for list in efieldtrace]
314     highest = max(efieldonly)
315     highestindex = efieldonly.index(highest)
316     peaksum =  ( efieldonly[highestindex - 2] + efieldonly[highestindex -
    1] + highest + efieldonly[highestindex + 1] + efieldonly[highestindex
    +2]) * 0.5
317     #peaksum = highest
318     return peaksum #sum of efield in uV*ns/m
319 
320 #Improved version works for all events, asigns 0 to the outer ring of
    antennas and 1 to the inner ring of antennas, based on distance to
    origin
321 def define_innerouter(cone_list, filename):
322     for item in cone_list:
323         azimuthantenna = round(calculate_azimuthantenna(filename, item[0]))
324         radius = np.linalg.norm(extract_positions(filename)[item[0]])
325         item.append(azimuthantenna)
326         item.append(radius)
327     list_sorted = sorted(cone_list, key=lambda x:x[2])
328     i = 0
329     while i < (len(list_sorted)):
330         if i == len(list_sorted) - 1:
331             list_sorted[i].append(0)
```

```
            i += 1
            #if azimuth same
        elif list_sorted[i][2] == list_sorted[i+1][2]:
           #and if radius of element i is smaller
            if list_sorted[i][3] < list_sorted[i+1][3]:
               #element i is inner ring
                list_sorted[i].append(0)
                list_sorted[i+1].append(1)
                i += 2
            elif list_sorted[i][3] > list_sorted[i+1][3]:
                list_sorted[i].append(1)
                list_sorted[i+1].append(0)
                i += 2
        else:
            list_sorted[i].append(0)
            i += 1
    return list_sorted #list of [antenna, efield, azimuthantenna, radius,
    inner_ring = 0 outer_ring = 1]

#this functions does the same as the function above, but based on electric
    field height
def define2(cone_list, filename):
    for item in cone_list:
        azimuthantenna = round(calculate_azimuthantenna(filename, item[0]))
        radius = np.linalg.norm(extract_positions(filename)[item[0]])
        item.append(azimuthantenna)
        item.append(radius)
    inputData = sorted(cone_list, key=lambda x:x[2])
    #print(inputData)
    i = 0
    return list(map(lambda x: max(x[1], key=itemgetter(1)),        groupby(
    sorted(inputData, key=itemgetter(2)), key=itemgetter(2))))

#this function only keeps the inner ring of antennas. It may not be 100%
    accurate, but worked sufficiently for my analysis. It could be improved
    with the brief function above.
def define_keepinner(cone_list, filename):
    for item in cone_list:
        azimuthantenna = round(calculate_azimuthantenna(filename, item[0]))
        radius = np.linalg.norm(extract_positions(filename)[item[0]])
        item.append(azimuthantenna)
        item.append(radius)
    list_sorted = sorted(cone_list, key=lambda x:x[2])
    i=0
    while i < (len(list_sorted)):
        if i == len(list_sorted) - 1:
            list_sorted[i].append(0)
            i += 1
            #if azimuth same
        elif i == len(list_sorted) - 2:
            if list_sorted[i][2] == list_sorted[i+1][2]:
```

```
            #and if radius of element i is smaller
                 if list_sorted[i][1] > list_sorted[i+1][1]:
                #element i is inner ring
                     list_sorted[i].append(0)
                     list_sorted[i+1].append(1)
                     i += 2
                 elif list_sorted[i][1] < list_sorted[i+1][1]:
                     list_sorted[i].append(1)
                     list_sorted[i+1].append(0)
                     i += 2
             else:
                 list_sorted[i].append(0)
                 i += 1
        elif list_sorted[i][2] == list_sorted[i+1][2] and list_sorted[i][2]
     == list_sorted[i+2][2] :
            #and if radius of element i is smaller
             if list_sorted[i][1] > list_sorted[i+1][1] and list_sorted[i
    ][1] > list_sorted[i+2][1] :
                #element i is inner ring
                 list_sorted[i].append(0)
                 list_sorted[i+1].append(1)
                 list_sorted[i+2].append(1)
                 i += 3
             elif list_sorted[i+1][1] > list_sorted[i][1] and list_sorted[i
    +1][1] > list_sorted[i+2][1]:
                 list_sorted[i].append(1)
                 list_sorted[i+1].append(0)
                 list_sorted[i+2].append(1)
                 i += 3
             elif list_sorted[i+2][1] > list_sorted[i][1] and list_sorted[i
    +2][1] > list_sorted[i+1][1]:
                 list_sorted[i].append(1)
                 list_sorted[i+1].append(1)
                 list_sorted[i+2].append(0)
                 i += 3
        elif list_sorted[i][2] == list_sorted[i+1][2]:
            #and if radius of element i is smaller
             if list_sorted[i][1] > list_sorted[i+1][1]:
                #element i is inner ring
                 list_sorted[i].append(0)
                 list_sorted[i+1].append(1)
                 i += 2
             elif list_sorted[i][1] < list_sorted[i+1][1]:
                 list_sorted[i].append(1)
                 list_sorted[i+1].append(0)
                 i += 2
        else:
             list_sorted[i].append(0)
             i += 1
    list_inner = [list for list in list_sorted if list[4] == 0]
```

```
424     return list_inner #list of [antenna, efield, azimuthantenna, radius,
        inner_ring = 0 outer_ring = 1]
425
426 # This only works for the events with complete inner ring
427 def define_removeouter(cone_list, filename):
428     for item in cone_list:
429         azimuthantenna = round(calculate_azimuthantenna(filename, item[0]))
430         radius = np.linalg.norm(extract_positions(filename)[item[0]])
431         item.append(azimuthantenna)
432         item.append(radius)
433     list_sorted = sorted(cone_list, key=lambda x:x[2])
434     i = 0
435     while i < (len(list_sorted)-1):
436         if list_sorted[i][2] == list_sorted[i+1][2]:
437             if list_sorted[i][3] > list_sorted[i+1][3]:
438                 list_sorted.remove(list_sorted[i])
439             else:
440                 list_sorted.remove(list_sorted[i+1])
441             i += 1
442         else:
443             i += 1
444     return list_sorted #list of [antenna, efield, azimuthantenna, radius,
        inner_ring = 0 outer_ring = 1]
445
446
447 def plot_folder(files_list):
448     for item in files_list:
449         plot_angular3(item)
450
451 #Sine and Cosine Functions for the Angular Distribution
452 def objective0(x, a, b):
453     return a * np.sin(np.radians(x) - b)
454
455 def objective1(x, e, f):
456     return e * np.sin(np.radians(x)- f)
457
458 def objective2(x, a, c, d):
459     return a * sin(np.radians(x)) + c +  d*cos(np.radians(x))
460
461 def objective3(x, e, g, h):
462     return e * np.sin(np.radians(x)) + g + h*cos(np.radians(x))
463
464 def objective4(x, a):
465     return a / x**2
466
467
468 #GRANDConvention: (E_x, E_y, E_z) = (E_NS, E_EW, E_V)
469 #GRANDConvention: zenith = theta: angle between z-axis and v-axis = angle
    between vertical axis and shower axis
470 #GRANDConvention: azimuth = phi: angle between x-axis and v-axis  = angle
    between North-axis and projected shower axis
```

```
471
472 def calculate_alfa(vvector, bvector):
473     bvector = bvector/np.linalg.norm(bvector)
474     alfa = np.arccos( np.dot(vvector, bvector) )
475     return alfa
476
477
478 #plot position of Cherenkov Cone in the Shower Plane
479 def plot_positions4peak(filename):
480     cone_list = find_cone(filename)
481     cone_list = define2(cone_list, filename)
482     #cone_list = define_keepinner(cone_list, filename)
483     #cone_list = define_innerouter(cone_list, filename)
484     filewithoutpath = Path(filename).stem
485     #general event information:
486     azimuth = -180 + extract_azimuth(filename)
487     zenith =  180- extract_zenith(filename)
488     Xmax = extract_Xmax(filename)
489
490     #START: vxB and vxvxB vectors are calculated
491     #Input Angles in Matias coordinates
492     theta =   radians(zenith)
493     phi = radians(azimuth)
494     Xmax = extract_Xmax(filename)
495     bfield = extract_Bearth(filename)
496
497
498     bvector = np.zeros(3)
499     bvector[0] = bfield[0] * sin(bfield[1]) * cos(bfield[2])
500     bvector[1] = -bfield[0] * sin(bfield[1]) * sin(bfield[2])
501     bvector[2] = bfield[0] * cos(bfield[1])
502     bvector = bvector/np.linalg.norm(bvector)
503
504     vvector = -np.sin(theta)*np.cos(phi), -np.sin(theta)*np.sin(phi), -np.
    cos(theta)
505     vxB = np.cross(vvector, bvector)
506     vxvxB = np.cross(vvector, vxB)
507     vvector = vvector/np.sqrt(vvector[0]**2+vvector[1]**2+vvector[2]**2)
508     vxB = vxB/np.sqrt(vxB[0]**2+vxB[1]**2+vxB[2]**2)
509     vxvxB = vxvxB/np.sqrt(vxvxB[0]**2+vxvxB[1]**2+vxvxB[2]**2)
510     #END: vxB and vxvxB vectors are calculated:
511
512     vxBlist = []
513     vxvxBlist = []
514     Z = []
515
516     #Angle (alfa) between shower-axis and Bfield
517     alfa = calculate_alfa(vvector, bvector)
518     print('alfa = ',  np.degrees(alfa))
519     print('sin(alfa)^2 =', sin(alfa)**2)
520     print(azimuth)
```

```
    #print('azimuth=', azimuth)

    for item in cone_list:
        posNS = extract_positions(filename)[item[0]][0]
        posEW = extract_positions(filename)[item[0]][1]

        #START: Coordinate Transformation to shower plane
        coords = [posNS, posEW, 0]
        showercoords = np.zeros(3)

        showercoords[0] = vxB[0]*coords[0] + vxB[1]*coords[1] + vxB[2]*coords[2]
        showercoords[1] = vxvxB[0]*coords[0] + vxvxB[1]*coords[1] + vxvxB[2]*coords[2]
        showercoords[2] = vvector[0]*coords[0] + vvector[1]*coords[1] + vvector[2]*coords[2]
        #END: Coordinate Transformation to shower plane

        vxBlist.append(showercoords[0])
        vxvxBlist.append(showercoords[1])

        #START: Coordinate Transformation to shower plane
        E = extractsum_efieldtrace3peak(filename, item[0])

        #P2Pefield = extract_P2Pefield(filename)[item[0]]
        #efieldonly = P2Pefield[1]

        #Eshower = np.zeros(3)
        #Eshower[0] = vxB[0]*E[0] + vxB[1]*E[1] + vxB[2]*E[2]
        #Eshower[1] = vxvxB[0]*E[0] + vxvxB[1]*E[1] + vxvxB[2]*E[2]
        #Eshower[2] = vvector[0]*E[0] + vvector[1]*E[1] + vvector[2]*E[2]

        #END: Coordinate Transformation to shower plane

        E = np.linalg.norm(E)
        norm = sum_eplusminus(filename)

        E_norm = E / norm *10e7

        Z.append(E_norm)
    plt.figure(figsize=(8, 6))
    plt.scatter(vxBlist, vxvxBlist, c = Z, s=140)
    plt.xlabel('vxB [m]', fontsize = 16)
    plt.ylabel('vx(vxB) [m]', fontsize = 16)
    #plt.axis('equal')
    plt.grid()
    plt.title('Cherenkov-Ring in Shower Plane \n 3.98 EeV Proton; Zen. ' + str(round(zenith)) + '; Azi. ' +str(round(azimuth)) + '; Xmax ' +str(round(Xmax)), fontsize = 18)
    clb = plt.colorbar()
    clb.ax.set_xlabel('Efield \n Normalized ', fontsize = 13)
```

```
    clb.ax.tick_params(labelsize = 14)

    plt.xticks(fontsize= 13)
    plt.yticks(fontsize= 13)

    #plt.xlim(-900, 900)
    #plt.ylim(-900, 900)
    plt.show()
    #plt.savefig('/Users/h.brans/Pictures/21sepcone/'+str(filewithoutpath)+'.png')
    #plt.clf()

 #plot angular distribution with known efield in shower plane
def plot_angular4(filename):
    cone_list = find_cone(filename)
    filewithoutpath = Path(filename).stem
    #efieldfile = hdf.File(('/Users/h.brans/Documents/Efield/Efield_'+str(filewithoutpath)+'.hdf5'), 'r')
    X = []
    Y = []
    Z = []
    azimuth = -180 + extract_azimuth(filename)
    zenith = 180 - extract_zenith(filename)
    Xmax = extract_Xmax(filename)

    cone_list_sorted = define_innerouter(cone_list, filename)

    #START: vxB and vxvxB vectors are calculated
    theta =   radians(zenith)
    phi = radians(azimuth)
    Xmax = extract_Xmax(filename)
    bfield = extract_Bearth(filename)


    bvector = np.zeros(3)
    bvector[0] = bfield[0] * sin(bfield[1]) * cos(bfield[2])
    bvector[1] = -bfield[0] * sin(bfield[1]) * sin(bfield[2])
    bvector[2] = bfield[0] * cos(bfield[1])
    bvector = bvector/np.linalg.norm(bvector)
    vvector = -np.sin(theta)*np.cos(phi), -np.sin(theta)*np.sin(phi), -np.cos(theta)

    vvector = vvector/np.linalg.norm(vvector)
    vxB = cross(vvector, bvector)
    vxvxB = cross(vvector, vxB)

    vxB = vxB/np.linalg.norm(vxB)
    vxvxB = vxvxB/np.linalg.norm(vxvxB)
    #END: vxB and vxvxB vectors are calculated:

    for item in cone_list_sorted:
```

```
615         posNS = extract_positions(filename)[item[0]][0]
616         posEW = extract_positions(filename)[item[0]][1]
617 
618         #START: Coordinate Transformation to shower plane
619         coords = [posNS, posEW, 0]
620         showercoords = np.zeros(3)
621 
622         showercoords[0] = vxB[0]*coords[0] + vxB[1]*coords[1] + vxB[2]*
    coords[2]
623         showercoords[1] = vxvxB[0]*coords[0] + vxvxB[1]*coords[1] + vxvxB
    [2]*coords[2]
624         showercoords[2] = vvector[0]*coords[0] + vvector[1]*coords[1] +
    vvector[2]*coords[2]
625         #END: Coordinate Transformation to shower plane
626         angle = m.atan2(showercoords[1], showercoords[0] )
627 
628         X.append(np.degrees(angle))
629         #group5 = efieldfile.get('Sum of Efield')
630         #efieldsum = sum_efieldpeak2(filename, item[0])
631 
632 
633         #START: Coordinate Transformation to shower plane
634         E = extractsum_efieldtrace3peak(filename, item[0])
635 
636 
637         #END: Coordinate Transformation to shower plane
638 
639         Enorm = np.linalg.norm(E)
640 
641         Y.append(Enorm)
642         Z.append(item[4])
643 
644     norm = sum_eplusminus(filename)
645     Y = Y/norm*10e7
646 
647     data = list(map (list, zip(X, Y, Z)))
648     #clb = plt.colorbar()
649     #clb.ax.set_xlabel('Inner=0 / Outer=1')
650     #X1 = list( range(-175, 175) )
651     #plt.plot(X1, Y1)
652     #plt.plot(X1, Y2)
653 
654     X_inner = []
655     Y_inner = []
656     X_outer = []
657     Y_outer = []
658     #outerring = []
659     #print(data)
660 
661     for item in data:
662         if item[2] == 0:
```

```
            X_inner.append(item[0])
            Y_inner.append(item[1])
        elif item[2] == 1:
            X_outer.append(item[0])
            Y_outer.append(item[1])
    #print(X_inner)
   #START: Fitting Procedure
    popt_inner, _ = curve_fit(objective2, X_inner, Y_inner)
    popt_outer, _ = curve_fit(objective3, X_outer, Y_outer)
# summarize the parameter values
    a, c, b = popt_inner
    d, f, e = popt_outer
    print(popt_inner)
    print(popt_outer)
# plot input vs output
    #plt.figure(figsize=(, 10))
    plt.xlabel('Angle in Shower Plane (beta) [degrees]', fontsize = 13)
    plt.ylabel('Efield Normalized', fontsize = 13)
    plt.xticks(fontsize= 14)
    plt.yticks(fontsize= 14)
    plt.scatter(X_inner, Y_inner, color ='red', s=60 )
    plt.scatter(X_outer, Y_outer, color = 'blue', s = 55)
# define a sequence of inputs between the smallest and largest known inputs
    x0_line = np.arange(min(X_inner), max(X_inner), 1)
    x1_line = np.arange(min(X_outer), max(X_outer), 1)
# calculate the output for the range
    y0_line = objective2(x0_line, a, c, b)
    y1_line = objective3(x1_line, d, f, e)
# create a line plot for the mapping function
    plt.plot(x0_line, y0_line, '--', color='red', linewidth = 3)
    plt.plot(x1_line, y1_line, '--', color='blue', linewidth = 2.5)
    plt.title('Angular Distribution in Shower Plane, 3.98 EeV Proton \n Zen
    . = ' +str(zenith) + ', Azi. = ' + str(azimuth) + ', slantXmax = ' +str(
    Xmax) +' \n Red-Inner a C b = ' +str(np.around(popt_inner, 2)) +', Blue-
    Outer a C b = ' +str(np.around(popt_outer, 2)))
    #plt.title('Angular Distribution in Shower Plane, 3.98 EeV Proton \n
    Zen. ' + str(round(zenith)) + '; Azi. ' +str(round(azimuth)) + '; Xmax '
     +str(round(Xmax)) +'; a C b = ' +str(np.around(popt_inner, 2)) ,
    fontsize = 14 )

    #plt.ylim(5.3,6.3)
    plt.show()
    #plt.savefig('/Users/h.brans/Pictures/21sepangular/'+str(
    filewithoutpath)+'.png')
    #plt.clf()

#make a table of the parameters, perform the analysis on a complete folder
def table_results(filename):
    cone_list = find_cone(filename)
    filewithoutpath = Path(filename).stem
```

```
706     #efieldfile = hdf.File(('/Users/h.brans/Documents/Efield/Efield_'+str(
        filewithoutpath)+'.hdf5'), 'r')
707     X = []
708     Y = []
709     Z = []
710     azimuth = 180 + extract_azimuth(filename)
711     zenith = 180 - extract_zenith(filename)
712     Xmax = extract_Xmax(filename)
713 
714     cone_list_sorted = define_innerouter(cone_list, filename)
715 
716     #START: vxB and vxvxB vectors are calculated
717     theta =   radians(zenith)
718     phi = radians(azimuth)
719     Xmax = extract_Xmax(filename)
720     bfield = extract_Bearth(filename)
721 
722 
723     bvector = np.zeros(3)
724     bvector[0] = bfield[0] * sin(bfield[1]) * cos(bfield[2])
725     bvector[1] = -bfield[0] * sin(bfield[1]) * sin(bfield[2])
726     bvector[2] = bfield[0] * cos(bfield[1])
727     bvector = bvector/np.linalg.norm(bvector)
728     vvector = -np.sin(theta)*np.cos(phi), -np.sin(theta)*np.sin(phi), -np.
        cos(theta)
729 
730     vvector = vvector/np.linalg.norm(vvector)
731     vxB = cross(vvector, bvector)
732     vxvxB = cross(vvector, vxB)
733 
734     vxB = vxB/np.linalg.norm(vxB)
735     vxvxB = vxvxB/np.linalg.norm(vxvxB)
736     #END: vxB and vxvxB vectors are calculated:
737 
738     for item in cone_list_sorted:
739         posNS = extract_positions(filename)[item[0]][0]
740         posEW = extract_positions(filename)[item[0]][1]
741 
742         #START: Coordinate Transformation to shower plane
743         coords = [posNS, posEW, 0]
744         showercoords = np.zeros(3)
745 
746         showercoords[0] = np.dot(vxB, coords)
747         showercoords[1] = np.dot(vxvxB, coords)
748         showercoords[2] = np.dot(vvector, coords)
749         #END: Coordinate Transformation to shower plane
750         angle = m.atan2(showercoords[1], showercoords[0] )
751 
752         X.append(np.degrees(angle))
753         #group5 = efieldfile.get('Sum of Efield')
754         #efieldsum = sum_efieldpeak2(filename, item[0])
```

```
755
756
757         #START: Coordinate Transformation to shower plane
758         E = extractsum_efieldtrace3peak(filename, item[0])
759
760         #END: Coordinate Transformation to shower plane
761
762         Enorm = np.linalg.norm(E)
763
764         Y.append(Enorm)
765         Z.append(item[4])
766
767     norm = sum_eplusminus(filename)
768     Y = Y/norm*10e7
769
770     data = list(map (list, zip(X, Y, Z)))
771     #clb = plt.colorbar()
772     #clb.ax.set_xlabel('Inner=0 / Outer=1')
773     #X1 = list( range(-175, 175) )
774     #plt.plot(X1, Y1)
775     #plt.plot(X1, Y2)
776     plt.xlabel('Antenna Azimuth Angle [degrees]')
777     plt.ylabel('Efield Normalized')
778     X_inner = []
779     Y_inner = []
780     X_outer = []
781     Y_outer = []
782     #outerring = []
783     #print(data)
784
785     for item in data:
786         if item[2] == 0:
787             X_inner.append(item[0])
788             Y_inner.append(item[1])
789         elif item[2] == 1:
790             X_outer.append(item[0])
791             Y_outer.append(item[1])
792     #print(X_inner)
793     popt_inner, _ = curve_fit(objective2, X_inner, Y_inner)
794     popt_outer, _ = curve_fit(objective3, X_outer, Y_outer)
795 # summarize the parameter values
796     a, c, b = popt_inner
797     e, g, f = popt_outer
798     #print(popt_inner)
799     #print(popt_outer)
800 # plot input vs output
801     #plt.scatter(X_inner, Y_inner, color ='red')
802     #plt.scatter(X_outer, Y_outer, color = 'blue')
803 # define a sequence of inputs between the smallest and largest known inputs
804     #x0_line = np.arange(min(X_inner), max(X_inner), 1)
805     #x1_line = np.arange(min(X_outer), max(X_outer), 1)
```

```
# calculate the output for the range
    #y0_line = objective2(x0_line, a, c)
    #y1_line = objective3(x1_line, e, g)
# create a line plot for the mapping function
    #plt.plot(x0_line, y0_line, '--', color='red')
    #plt.plot(x1_line, y1_line, '--', color='blue')
    #plt.title('Angular Distribution in Shower Plane: a*sin(phi) + c \n azimuth = ' + str(azimuth) + ', zenith = ' +str(zenith) + ', slantXmax = ' +str(Xmax) +' \n Red-Inner a c = ' +str(np.around(popt_inner, 2)) +', Blue-Outer a c = ' +str(np.around(popt_outer, 2)))
    #plt.show()
    #plt.savefig('/Users/h.brans/Pictures/10sepangular/'+str(filewithoutpath)+'.png')
    return((zenith, azimuth, Xmax, np.around(a, 2),  np.around(c, 2), np.around(b, 2), np.around(e, 2), np.around(g, 2), np.around(f, 2)))

#This function only analyses the inner ring of antennas.
def table_results2(filename):
    cone_list = find_cone(filename)
    filewithoutpath = Path(filename).stem
    #efieldfile = hdf.File(('/Users/h.brans/Documents/Efield/Efield_'+str(filewithoutpath)+'.hdf5'), 'r')
    X = []
    Y = []
    Z = []
    azimuth = -180 + extract_azimuth(filename)
    zenith = 180 - extract_zenith(filename)
    Xmax = extract_Xmax(filename)
    Eprimary = extract_Eprimary(filename)
    cone_list_sorted = define_keepinner(cone_list, filename)

    #START: vxB and vxvxB vectors are calculated
    theta =   radians(zenith)
    phi = radians(azimuth)
    Xmax = extract_Xmax(filename)
    bfield = extract_Bearth(filename)
    bvector = np.zeros(3)
    bvector[0] = bfield[0] * sin(bfield[1]) * cos(bfield[2])
    bvector[1] = -bfield[0] * sin(bfield[1]) * sin(bfield[2])
    bvector[2] = bfield[0] * cos(bfield[1])
    bvector = bvector/np.linalg.norm(bvector)
    vvector = -np.sin(theta)*np.cos(phi), -np.sin(theta)*np.sin(phi), -np.cos(theta)

    vvector = vvector/np.linalg.norm(vvector)
    vxB = cross(vvector, bvector)
    vxvxB = cross(vvector, vxB)

    vxB = vxB/np.linalg.norm(vxB)
    vxvxB = vxvxB/np.linalg.norm(vxvxB)
    #END: vxB and vxvxB vectors are calculated:
```

```
850
851     for item in cone_list_sorted:
852         posNS = extract_positions(filename)[item[0]][0]
853         posEW = extract_positions(filename)[item[0]][1]
854
855         #START: Coordinate Transformation to shower plane
856         coords = [posNS, posEW, 0]
857         showercoords = np.zeros(3)
858
859         showercoords[0] = vxB[0]*coords[0] + vxB[1]*coords[1] + vxB[2]*
    coords[2]
860         showercoords[1] = vxvxB[0]*coords[0] + vxvxB[1]*coords[1] + vxvxB
    [2]*coords[2]
861         showercoords[2] = vvector[0]*coords[0] + vvector[1]*coords[1] +
    vvector[2]*coords[2]
862         #END: Coordinate Transformation to shower plane
863         angle = m.atan2(showercoords[1], showercoords[0] )
864
865         X.append(np.degrees(angle))
866         #group5 = efieldfile.get('Sum of Efield')
867         #efieldsum = sum_efieldpeak2(filename, item[0])
868
869
870         #START: Coordinate Transformation to shower plane
871         E = extractsum_efieldtrace3peak(filename, item[0])
872
873         #END: Coordinate Transformation to shower plane
874
875         Enorm = np.linalg.norm(E)
876
877         Y.append(Enorm)
878         Z.append(item[4])
879
880     norm = sum_eplusminus(filename)
881     Y = Y/norm*10e7
882
883     data = list(map (list, zip(X, Y, Z)))
884     #clb = plt.colorbar()
885     #clb.ax.set_xlabel('Inner=0 / Outer=1')
886     #X1 = list( range(-175, 175) )
887     #plt.plot(X1, Y1)
888     #plt.plot(X1, Y2)
889     plt.xlabel('Antenna Azimuth Angle [degrees]')
890     plt.ylabel('Efield Normalized')
891     X_inner = []
892     Y_inner = []
893     X_outer = []
894     Y_outer = []
895     #outerring = []
896     #print(data)
897
```

```
    for item in data:
        X_inner.append(item[0])
        Y_inner.append(item[1])
    #print(X_inner)
    popt_inner, _ = curve_fit(objective2, X_inner, Y_inner)

# summarize the parameter values
    a, c, b = popt_inner
    plt.scatter(X_inner, Y_inner)
    plt.show()
    return (zenith, azimuth, Xmax, np.around(a, 2),  np.around(c, 2), np.
   around(b, 2), Eprimary)

#create the txtfile with the inner and outer ring of antennas
def create_txtresults(files_list):
    f = open("/Users/h.brans/Downloads/7dec.txt","w+")
    #f=open("guru99.txt","a+")
    f.write('[Zenith, Azimuth, Xmax, Inner amplitude (a), Inner Offset (c),
    Inner vxB (b), Outer amplitude (a), Outer offset (c), Outer vxB (b),
   EPrimary]  \n \n' )
    f.write('[ \n')
    for item in files_list:
        text = table_results(item)
        f.write(str(text) + ',' + '\n')
        print(text)
    f.write(']')
    f.close()

#create the txtfile with the inner ring of antennas only
def create_txtresults2(files_list):
    f = open("/Users/h.brans/Downloads/22oktzenith.txt","w+")
    #f=open("guru99.txt","a+")
    f.write('[Zenith, Azimuth, Xmax, Inner amplitude (a), Inner Offset (c),
    Inner vxB (b), Primary Energy]  \n \n' )
    f.write('[ \n')
    for item in files_list:
        desiredE = extract_Eprimary(item)
        desiredparticle = extract_primary(item)
        if desiredE == 3.9811 and desiredparticle ==  b'Proton':
            text = table_results2(item)
            print(text)
            f.write(str(text) + ',' + '\n')
        else:
            pass
    f.write(']')
    f.close()
```

References

[1] H. Brans, "Inkijkje in GRANDProto," 2019. [Online]. Available: https://www.ru.nl/highenergyphysics/theses/bachelor-theses/.

[2] E. N. Paudel, A. Coleman, and F. Schroeder, "Parametrization of the relative amplitude of geomagnetic and askaryan radio emission from cosmic-ray air showers using corsika/coreas simulations," *Proceedings of 37th International Cosmic Ray Conference — PoS(ICRC2021)*, Jul. 2021. DOI: 10.22323/1.395.0429. [Online]. Available: http://dx.doi.org/10.22323/1.395.0429.

[3] M. de Naurois and D. Mazin, "Ground-based detectors in very-high-energy gamma-ray astronomy," *Comptes Rendus Physique*, vol. 16, no. 6, pp. 610–627, 2015, Gamma-ray astronomy / Astronomie des rayons gamma, ISSN: 1631-0705. DOI: https://doi.org/10.1016/j.crhy.2015.08.011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1631070515001462.

[4] J. D. Jackson, *Classical electrodynamics*, 3rd ed. New York, NY: Wiley, 1999. [Online]. Available: http://cdsweb.cern.ch/record/490457.

[5] G. de Wijs, *Electrodynamics Course Syllabus*. Radboud University, 2020.

[6] U. o. S. F. Florida Center for Instructional Technology College of Education, "Conic section using ellipse," Jul. 2021. [Online]. Available: https://etc.usf.edu/clipart/76100/76194/76194_conic_ellips.htm.

[7] H. Padleckas, "Conic sections/ellipse," Jul. 2021.

[8] J. Hörandel, "Astroparticle physics lecture 8: Radio detection of extensive air showers," Jul. 2021. [Online]. Available: http://particle.astro.ru.nl/goto.html?ehome.

[9] C. Glaser, Jul. 2021.

[10] E. Holt, F. Schröder, and A. Haungs, "Enhancing the cosmic-ray mass sensitivity of air-shower arrays by combining radio and muon detectors.," *Eur. Phys. J.*, vol. 79, no. 371, 2019. [Online]. Available: https://doi.org/10.1140/epjc/s10052-019-6859-4.

[11] C. Glaser, M. Erdmann, J. R. Hörandel, T. Huege, and J. Schulz, "Simulation of the radiation energy release in air showers," *EPJ Web of Conferences*, vol. 135, S. Buitink, J. Hörandel, S. de Jong, R. Lahmann, R. Nahnhauer, and O. Scholten, Eds., p. 01 016, 2017, ISSN: 2100-014X. DOI: 10.1051/epjconf/201713501016. [Online]. Available: http://dx.doi.org/10.1051/epjconf/201713501016.

[12] C. Welling, C. Glaser, and A. Nelles, "Reconstructing the cosmic-ray energy from the radio signal measured in one single station," *Journal of Cosmology and Astroparticle Physics*, vol. 2019, no. 10, pp. 075–075, Oct. 2019, ISSN: 1475-7516. DOI: 10.1088/1475-7516/2019/10/075. [Online]. Available: http://dx.doi.org/10.1088/1475-7516/2019/10/075.

[13] K. Lohmann, N. Putman, and C. Lohmann, "Geomagnetic imprinting: A unifying hypothesis of long-distance natal homing in salmon and sea turtles," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, pp. 19 096–101, Jan. 2009. DOI: 10.1073/pnas.0801859105.

[14] Brainkart, "Earth's magnetic field and magnetic elements," [Online]. Available: https://www.brainkart.com/article/Earth---s-magnetic-field-and-magnetic-elements_38442/.

[15] J. Alvarez-Muñiz, W. R. C. Jr., and E. Zas, *Monte carlo simulations of radio pulses in atmospheric showers using zhaires*, 2011. arXiv: 1107.1189 [astro-ph.HE].

[16] S. Chiche, "Grand collaboration meeting: Emission processes for inclined showers," Dec. 2021.

[17] C. Zhang, "Grand collaboration meeting: Polarization pattern for inclined showers," Dec. 2021.

[18] M. Tueros, "Grand collaboration meeting: ZHAires status update," Dec. 2021.

Supervisor:
prof. dr. Charles Timmermans

Second Corrector:
dr. Harm Schoorlemmer