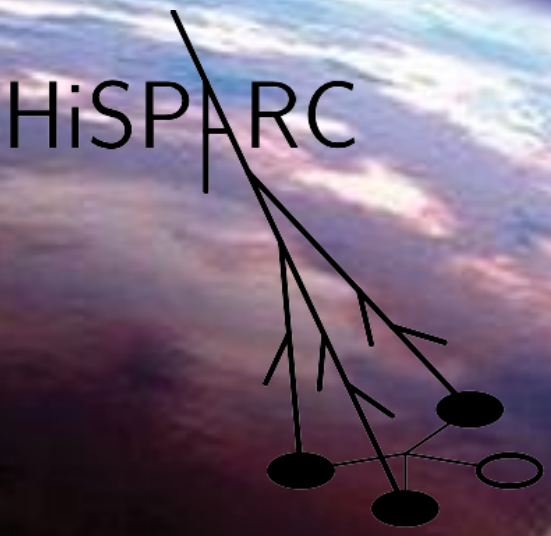
# Determining the Gerasimova-Zatsepin effect

**Dave Otte**

**Supervisor: Dr. Charles Timmermans**




**April 2014**

HOGESCHOOL UTRECHT

# Introduction

This paper is written within the context of my physics master of education at the university of applied sciences in Utrecht. I did my research for the department of experimental high energy physics at the Institute of Mathematics, Astrophysics and Particle Physics (IMAPP) at the Radboud University Nijmegen under supervision of dr. Charles Timmermans.

This research is a follow-up of the earlier research done by Erik Hermsen [17] and Margot Peters [12]. We try to find experimental evidence for the Gerasimova-Zatsepin effect when cosmic rays enter our Solar System. To detect this effect we use the HiSPARC detector network, which is a network of more than hundred muon detectors spread mainly across the Netherlands, with few detectors in England and Denmark.

The first chapter explains the theory behind cosmic rays and the Gerasimova-Zatsepin effect. In the second chapter the HiSPARC network will be explained and the way it collects data you will find in chapter 3. In chapter 3 also some theoretical assumptions for this research will be explained. Very important statistical methods used for our data analysis, especially the Feldman and Cousins method, are described in chapter 4 to better understand the results. Chapter 5 presents the results and chapter 6 gives a conclusion after analysing these results.

At the end I reserved some space to thank my supervisor and roommates.

I hope this paper gives rise to further research in this field of physics. Enjoy reading.

Dave Otte
University of applied sciences Utrecht
Physics Master of education

# Contents

# 1. Introduction

### 1.1 Cosmic Rays: What are they?

Every second the Earth is bombarded by more than 1000 particles per square meter. These particles are different ionized nuclei, and mostly originate from the sun and from the outskirts of our universe. They are called **cosmic rays**. The types of nuclei that enter the Earth's atmosphere are typically protons (about 90%), alpha particles (about 9%) and the remainder are heavier nuclei. The energies of these nuclei can be very big; where some of the particles even reach energies up to $10^{20}$ **eV**.

Apparently our universe is capable of accelerating particles to such high energies, but we are not able to confirm any theory of the processes behind those accelerations yet.
It is believed nowadays that supernova remnants are the main source of cosmic rays below $10^{15}$ eV. In 2013 direct evidence that cosmic-ray protons are accelerated in supernova remnants was given [1].

Scientists are also wondering about another question: 'where do these particles come from?'. This question is hard to answer for the charged high energetic particles. They travel through space where they are influenced by electromagnetic fields and change their direction accordingly. Only the very high-energy particles follow a nearly straight path, but these particles are very rare. In general the direction of the particle when colliding on the Earth's atmosphere does not point back to its origin.

### 1.1.2 Spallation

The cosmic rays constantly change direction when travelling through the **interstellar medium** because of the randomly oriented magnetic fields. Besides that, they are interacting with matter in different ways. One of those ways is through the ionisation of atoms they encounter, but the most important type of interaction is spallation. This is a process where a particle collides with another particle and falls apart. Other elements are created in those collisions. By measuring the abundances of nuclei we detect on Earth

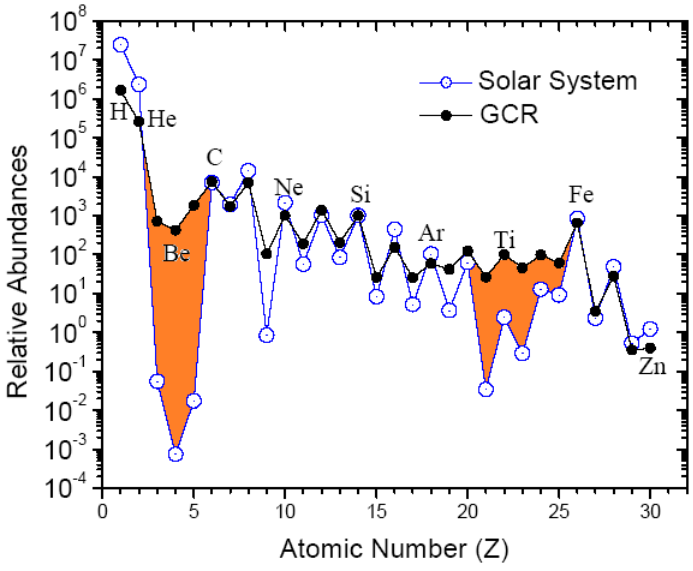



Figure 1.1:
Relative abundances of different elements in our Solar system compared to cosmic rays. The open circles represent the elements in the Solar system. The closed circles represent the relative abundances of the galactic cosmic rays measured by the CRIS instrument. [2]

and comparing those with the abundances of nuclei the sun is creating we see that some elements are more abundant than they should be, see the orange coloured spaces in figure 1.1. Scientists believe that this is caused by **spallation** in the interstellar medium, where these 'extra' elements are created by the falling apart of Carbon and Oxygen into Boron, Beryllium and Lithium. In the same way Iron falls apart into sub-iron elements, which explains the second orange area in the graph!

### 1.2 The Gerasimova-Zatsepin effect

When cosmic rays enter our solar system the same processes as in the interstellar medium, deflection and spallation, influence them. The deflection is caused by the magnetic field of the sun. The spallation is triggered by the elements in our solar system.
Besides that spallation can also be caused by **photodisintegration**, which is a process where a high-energy gamma ray of the sun is absorbed by an atomic nucleus and causes it to enter an excited state. The nucleus then immediately decays by emitting a subatomic particle, e.g. a proton or neutron or an alpha particle, see figure 1.2.
After the photodisintegration the separated particles follow in principle almost exactly the same way, because of the high energy of the primary particle, but they are deflected differently by the suns magnetic field. In which way the different particles are deflected is caused by the charge-mass ratio.
In 1960, Gerasimova and Zatsepin first talked about the effect of Photodisintegration in the solar system [3,4]. That's why this is called the Gerasimova-Zatsepin effect (GZ effect).

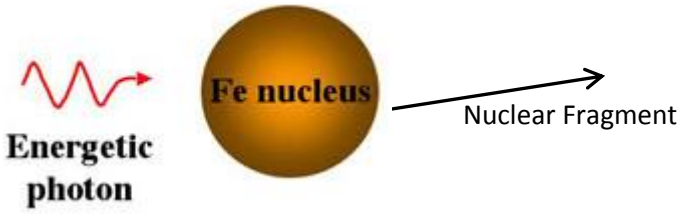


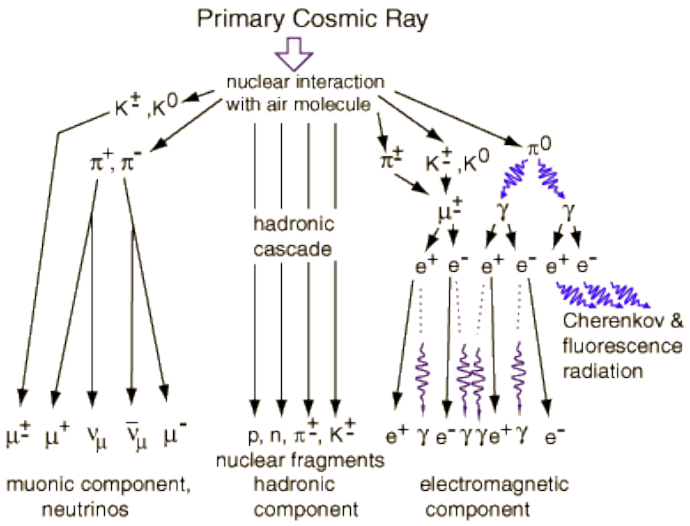

Figure 1.2: Photodisintegration of a particle. In this case an iron nucleus.

### 1.3 Primary and secondary particles

When a particle, after travelling through the interstellar medium and our solar system, is disintegrated by spallation, it's fragments can still enter the Earth's atmosphere with a lot of energy. These particles we call **'primary particles'**. When these particles collide with molecules, secondary particles are formed. These secondary particles collide again and disintegrate again etc. This is a process that produces a lot of secondary particles and is called a cascade or a **cosmic shower**. Eventually some of the primary particles have enough energy (above $10^{15}$ eV) to produce secondary particles that hit the Earth surface at sea-level, although most of them are absorbed by our atmosphere.

A shower 'ends' when the energy of the secondary particles becomes too low to produce new particles. The higher the energy of the primary particle the more particles are generated and the bigger the diameter of the cosmic shower.

The whole process starts when the primary particle enters the upper atmosphere and produces large numbers of particles, mostly charged and neutral pions. Only the uncharged pions decay into high-energy photons and eventually they form the starting point of cascades of electrons, gamma rays and positrons. The charged pions decay into muons and muon neutrinos.

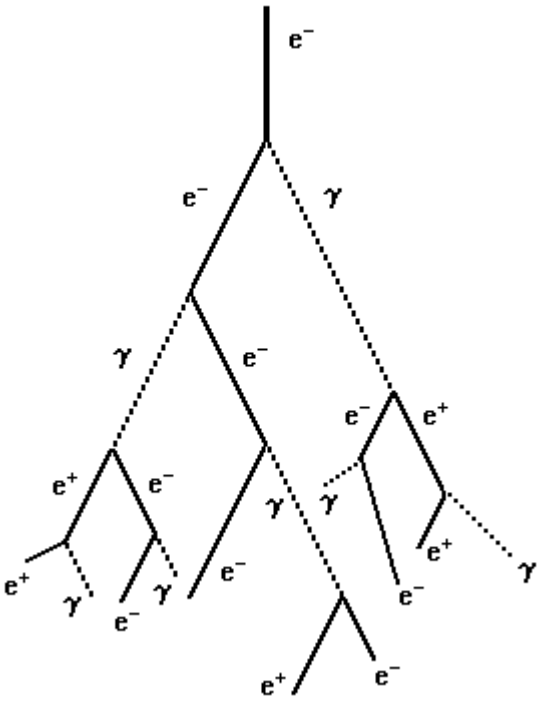



**Figuur 1.3: A typical cosmic shower. Different kinds of cascades produces by the same primary particle.**

It is important to divide a cosmic shower, produced by a single primary particle, into three different types of cascades: electromagnetic, muonic and hadronic (figure1.3). Each cascade produces his own particles and/or radiation. Eventually the setup for your experiment has to take in account the sensitivity for each kind of cascade.

### 1.3.1 The electromagnetic cascade

The electromagnetic cascade is called that way because it consists of electrons, positrons and photons (gamma rays). The neutral pion decays into high-energy gamma rays which in turn materialize into electron-positron pairs under influence of the electric field of the atoms in the atmosphere. These pairs are created with a threshold-energy of 1 MeV, while the gamma ray has in the order of $10^3$ times as much energy. The electrons and positrons move on with the energy of the former gamma ray and interact with the molecules they encounter. They again produce new gamma rays, though with less energy, but enough to make new electron-positron pairs. This continuing cycle generates a large cascade of particles. The process is drawn
in figure 1.4.



**Figuur 1.4: The electromagnetic cascade. Gamma rays produce electron-positron pairs and electron and/or positrons produce gamma rays [7].**

### 1.3.2 The hadronic cascade

The hadronic cascade is produced in a completely different way. Hadrons are subatomic particles made of quarks. They are divided into baryons (e.g. protons and neutrons) and mesons (pions and kaons). In general at each collision half of the incident hadron energy is transferred to the hadronic cascade [11]. The other half is transferred into different kind of particles (e.g. low energy pions) and other processes.

The neutral pions, roughly a third of all produced pions, are eventually dissipated into electromagnetic showers [10]. The charged pions, which have a longer decay length than the neutral pions, can either decay of re-interact. It depends on their energy. High energy charged pions interact and low energy charged pions decay into muons and muon neutrinos [11].

The most important characteristic in general of the hadronic shower compared to the electromagnetic one is that it takes more time to develop [10].

### 1.3.3 The muonic cascade

The muonic cascade is triggered only by the decay of charged pions, where muon neutrinos are a by-product [12]. This means that the muonic cascade is fed by the hadronic cascade which produces the charged pions. When a muon is formed it eventually decays into an electron or positron, a muon neutrino and an electron neutrino [12]. After this decay the cascade ends. With the HiSPARC network, explained in the next chapter, we mainly detect muons!

# 2. The HiSPARC network

Since 2003 there was a cooperation in Nijmegen, called the 'Nijmegen Area High School Array (NAHSA) who first made a network of detectors around the Catholic University of Nijmegen. Their aim was to detect cosmic showers. A secondary goal was the cooperation with schools in the neighbourhood, to deflect scientific knowledge to surrounding schools. Nowadays there is a network of 107 detectors that is used as our experimental setup. In several countries there are multiple detectors placed, mostly on roofs of universities and schools. These form so-called clusters, a couple of stations in one city. These can detect showers from a single particle, because they are so close to each other. The majority of the detectors are placed in the Netherlands, but also in Denmark, England and Germany are some detectors placed, see figure 2.1. This is perfect for measuring the GZ-effect because this means that the spacing between the stations varies between 10 m and 700 km.
In general the aim of the HiSPARC network is to detect air showers. While detecting them you can calculate their direction and/or the mass of the particle. How to calculate the direction of the air shower is explained in chapter 3.3.

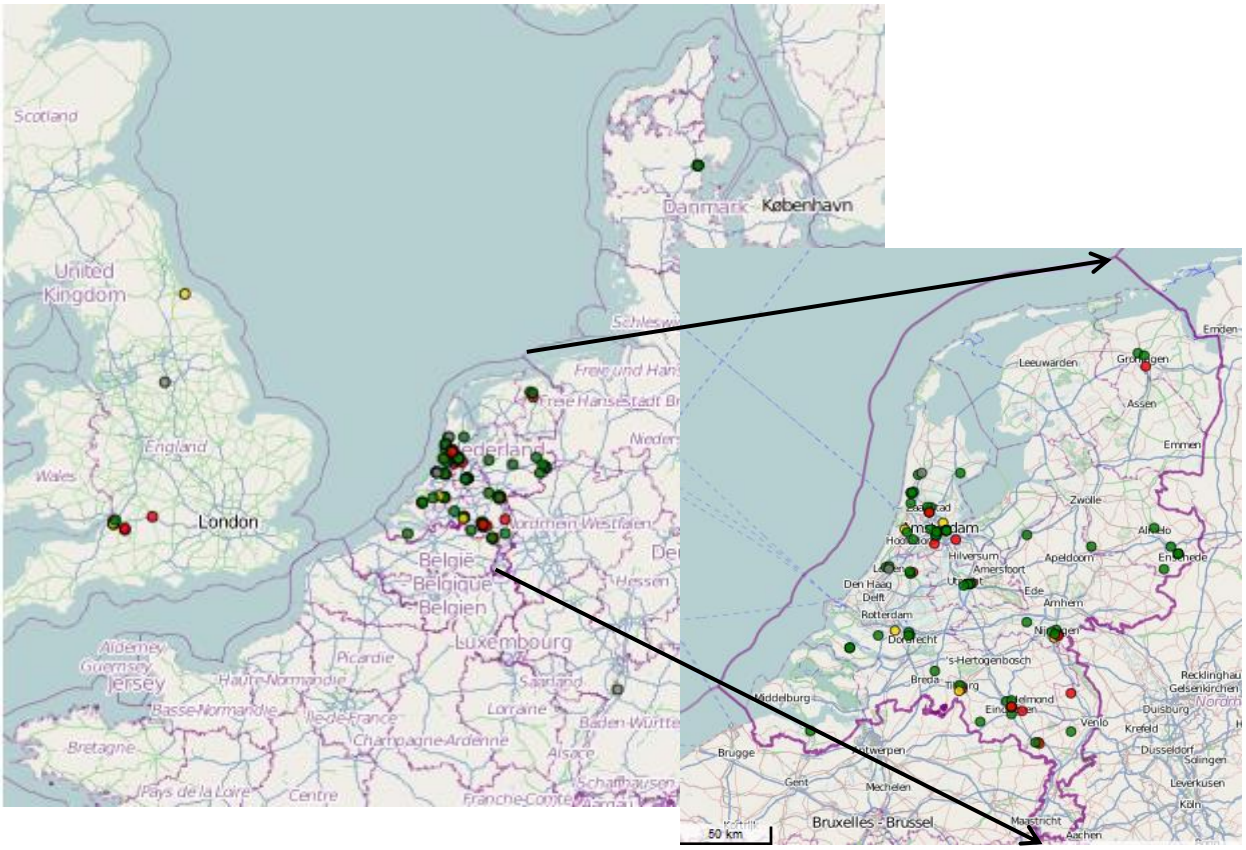


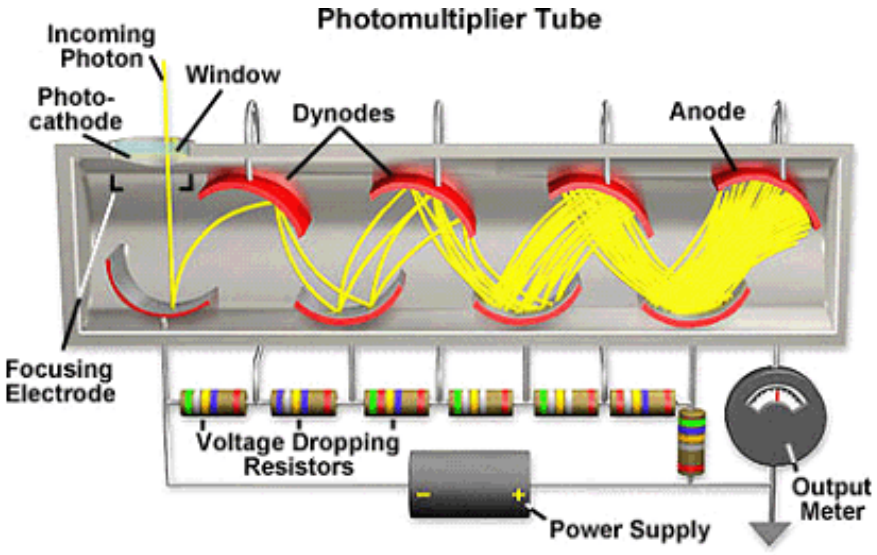

Figure 2.1: The HiSPARC network. Each dot represent a HiSPARC detector station

## 2.1 How the detector works

To detect cosmic rays we use two scintillators (see figure 2.2). These scintillators are made of sensitive material with the property to produce light when a charged particle flies through. The light is reflected internally and is eventually directed into a photomultiplier. In this photomultiplier tube (see figure 2.3) the incoming photon collides with a cathode, and an electron is created. This electron is multiplied in several stages, by which numerous electrons are produced. These electrons together form a large enough signal to read out by a oscilloscope. [9]


Figure 2.2: Two detectors on a rooftop

We always have to use two scintillators on the same roof. This is because there are a lot of particles passing through each detector each second, but only when two particles fly through both detectors at nearly the same time, then we detect an air shower. This is called an event and we say that the station is triggered.




2.3 A Photomultiplier tube. Electrons produced by an incoming photon are multiplied to create a stronger signal.

### 2.2 Finding the GZ-effect with the HiSPARC network

To find the GZ-effect we need an experimental setup where we can detect both air showers from the cosmic nucleus and the ejected nucleus. It is expected that the distance between the two showers of the GZ-effect is typically hundred till several hundred kilometres [13,14]. Furthermore calculations show that the fraction of cosmic particles that are participating in the GZ-effect is in the order of $10^{-5}$ [12,13].

The HiSPARC network is ideal to find the GZ-effect. Stations are grouped closely together in cities, thus we can detect single air showers. To detect multiple showers simultaneously we have different clusters all over different cities. When two separate showers hit two different clusters this could be evidence for the GZ-effect.

Furthermore it is important to realize that the uptime of the HiSPARC network is close to 100%. This means that we measure almost constantly. So when the GZ-effect really occurs we should detect it eventually when collected enough data, although the fraction of cosmic particles that participate in the GZ-effect is very small.

### 2.3 The LAAS project

In Japan there is an experimental setup comparable with the HiSPARC network in Europe, called the Large Area Air Showers (LAAS) experiment. With this experiment they tried to search for simultaneous and parallel Extended Air Showers (EAS) at multiple EAS stations due to the GZ-effect. Their main goal was to find excesses of these events in the solar direction and/or the lunar direction. In the solar direction they found no excesses, but in the lunar direction they found a small, but insignificant, deviation [14].

With our experiment we only focus on finding the GZ-effect. We are not interested in the solar direction and/or the lunar direction.

# 3. Analysis: The hunt for coincidences

This chapter explains how we searched for coincidences. The method is described, and the calculation for the direction of the shower is explained.

### 3.1 Algorithms

To find coincidences we use the raw data files. These raw data files contain timestamps of every detection made by a single detector. Eventually there are a lot of data, which have to be interpreted with algorithms, see Appendix B. These algorithms we had to make our self.

### 3.2 Finding coincidences in the raw data files

In order to find the Gerasimova-Zatsepin effect (GZ-effect), we search for a coincidence between a **cluster** (stations close to each other, typically hundreds of meters away) and a station not belonging to that cluster. To find these coincidences we select **timestamps** in the raw data files. These timestamps are the recorded times of each particle detection in a station, which is called an **'event'**. We time-order the timestamps, and compare these in order with the timestamps of other stations. If the timestamps of one or more additional stations are within a certain logical 'time window' we mark them as a **coincidence**. A coincidence, thus events at nearly the same time, means that certain events 'probably' belong to the same shower (in a **cluster**) or share the same primary particle. The Gerasimova–Zatsepin effect creates two incident primary particles at the same time, and a detection of each shower at nearly the same time.

#### 3.2.1 Background

Eventually we are looking for a signal on top of a normal background. The time window we use to find coincidences is set to 100 ms. The maximum distance between two stations is roughly 1000 km. If the GZ-effect is occurring than the time difference between detecting two showers of the same primary particle is 1000 km divided by the speed of light, thus 3,3 ms (when hitting the detectors under an angle of 90°), see figure 3.1.

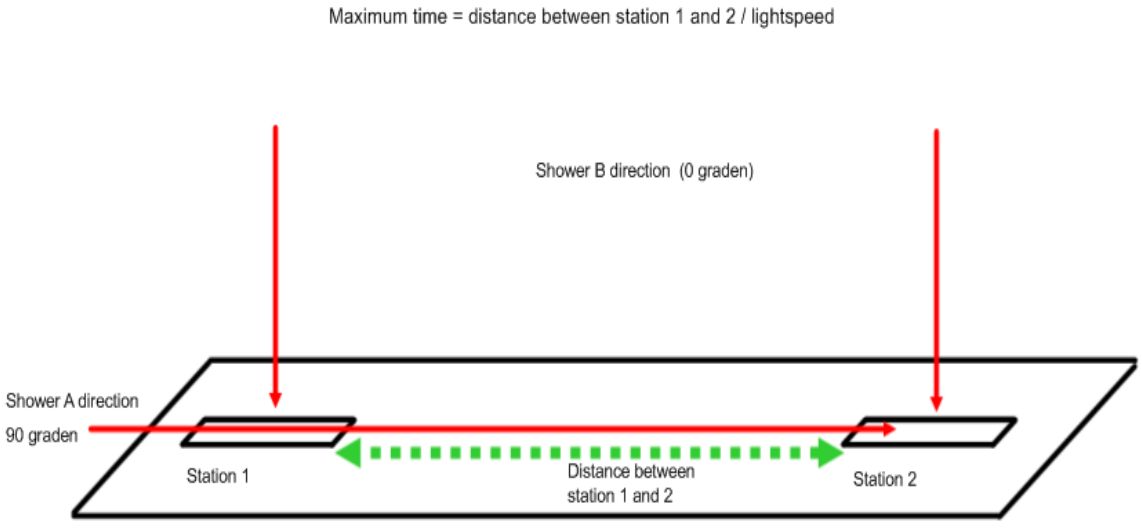



**Figure 3.1: The maximum time difference between two stations for finding coincidence**

To really detect a signal and be able to extract the background, we set a time window minimally 10 times larger than this, thus 33 ms for the maximum time difference, otherwise we can't determine the background properly, see table 3.1. Off course for all stations that are closer to each other we use less time difference, and thus are also captured by this maximum time window of 100 ms.

| Distance (log) | Absolute distance between two stations (m) | Time at 1x speed of light (s) | Time at 10x speed of light, for capturing background signal (s) | Additional information |
|---|---|---|---|---|
| -1 | 0,1 | 0,33 x$10^{-9}$ | 3,3 x$10^{-9}$ | Stations on a single roof |
| 0 | 1 | 3,3 x$10^{-9}$ | 33 x$10^{-9}$ | |
| 1 | 10 | 33 x$10^{-9}$ | 330 x$10^{-9}$ | |
| 2 | 100 | 330 x$10^{-9}$ | 3,3 x$10^{-6}$ | Stations in a clusters |
| 3 | 1k | 3,3 x$10^{-6}$ | 33 x$10^{-6}$ | |
| 4 | 10k | 33 x$10^{-6}$ | 330 x$10^{-6}$ | |
| 5 | 100k | 330 x$10^{-6}$ | 3,3 x$10^{-3}$ | Expected GZ-effect |
| 6 | 1000k | 3,3 x$10^{-3}$ | 33 x$10^{-3}$ | Expected GZ-effect |

**Table 3.1: Time difference at every distant between two stations**

### 3.2.2 Histogram: background and signal

To find and visualize the background and our signal we use histograms, see figure 3.2. Each histogram is three dimensional. On the x-axis you see the time difference between two stations. The numbers 1 till 10 are calculated in the following manner:

$$timedifference = 0.3x\frac{dt}{r_{stations}} \quad \textbf{(3.1)}$$

Time difference is a dimensionless number.

0.3 = the speed of light in km/ns

dt = The time difference between two stations of a coincidence (ns)

$r_{stations}$ = The distance between two stations (km)

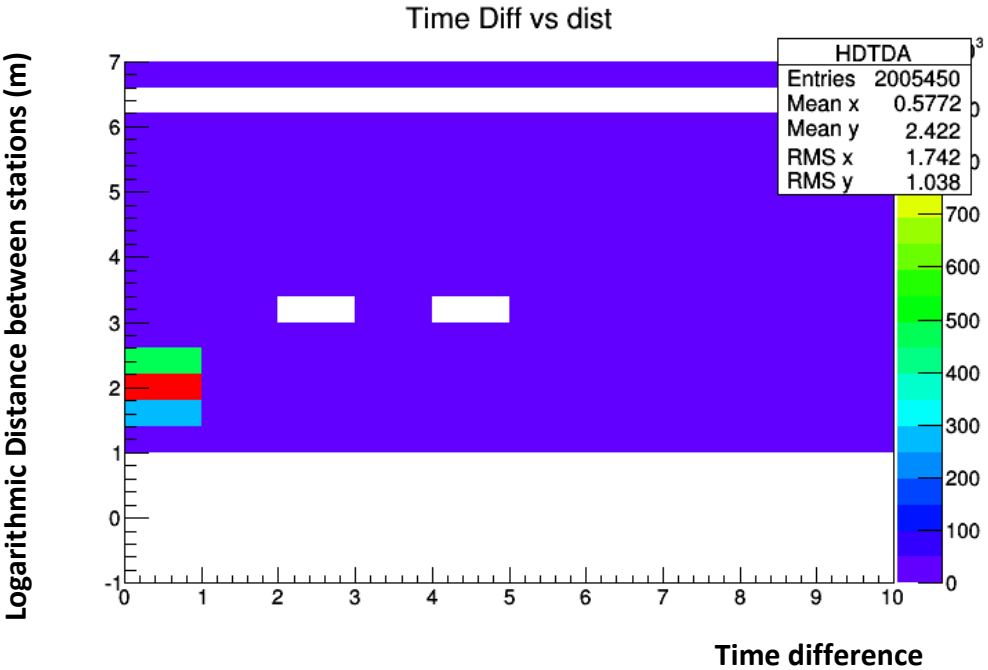



Figure 3.2: Time difference vs distance between station

visualizes the distance between two stations, pointed out on a logarithmic scale.
The third dimension is the colour code, which indicates how much coincidences there are found at each point in the histogram. To make a projection of this third dimension you can plot for example the black line in figure 3.2 in the histogram and you get figure 3.3.

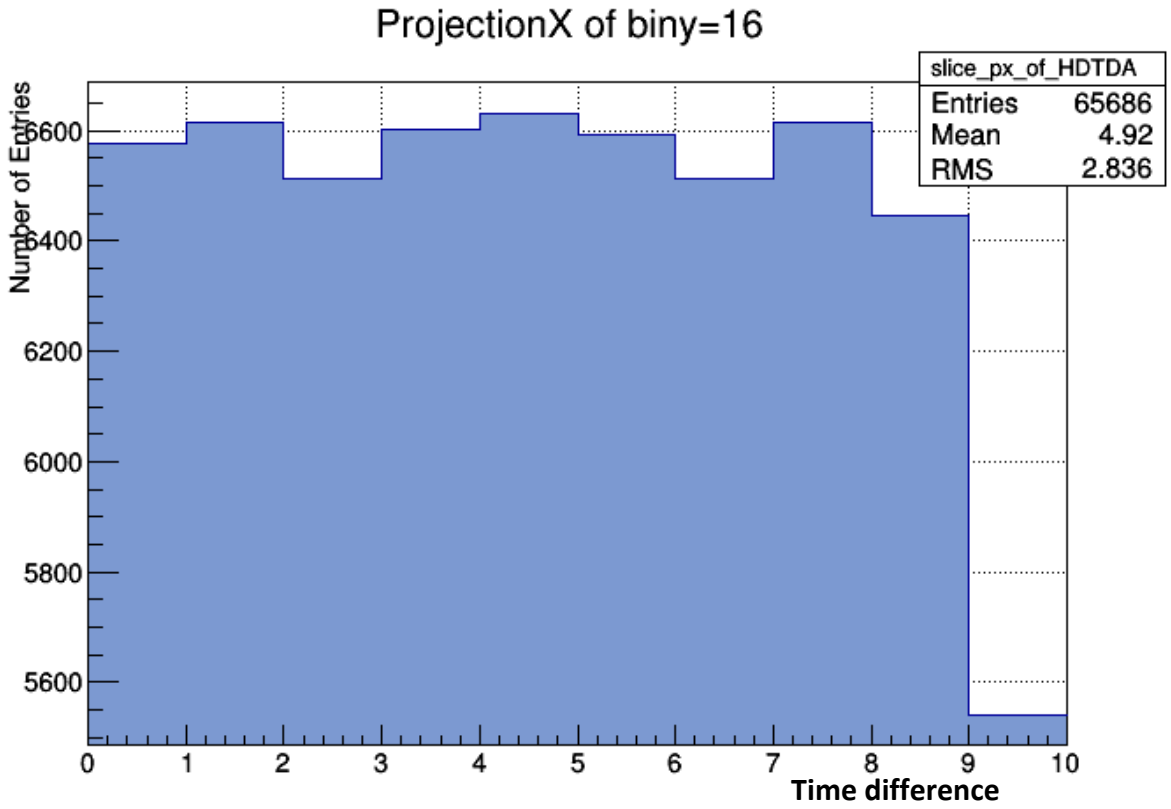



**Figure 3.3: Number of coincidences in a certain bin, compared to the time difference**

In figure 3.3 you can easily see the background. Everything after number 1 on the x-axis can't be the GZ-effect, so it forms our background signal. If there is a GZ-effect than we should find it in our first bin on the x-axis.

### 3.3 Calculating the direction of the shower

If we find clusters in the raw data we use the timestamps of the stations in that cluster to calculate the angle $\theta$ and $\varphi$ of the shower, see figure 3.4.

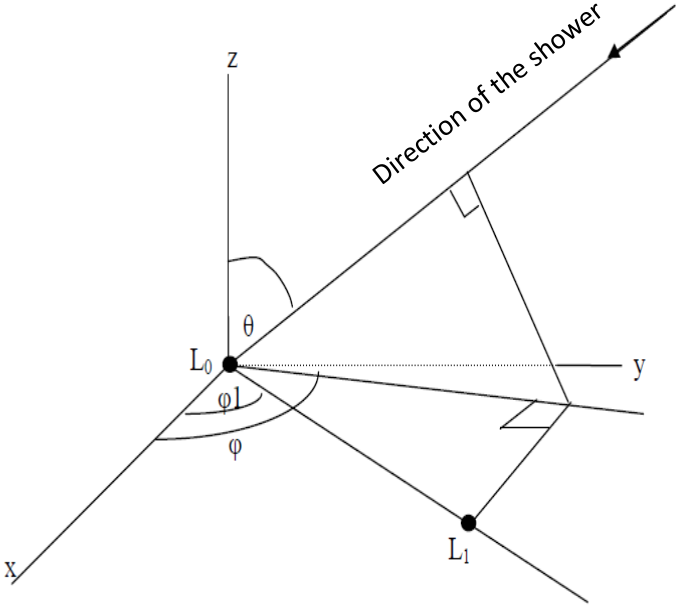



Figure 3.4:
Direction of shower compared to two stations L0 and L1 on earth. [8]

$\theta$ = Angle of the shower with the vertical z-axis
$\varphi$ = Angle on the x,y plane of the projection of the direction of the shower

We calculate these angles in the following manner:

Between *two* stations L0 and L1 a measured time difference **$dt_{01}$** exists and the following relation 3.2 holds:

$$c \cdot dt_{01} = L0L1 \cdot \cos(\varphi - \varphi_1) \cdot \sin(\theta) \qquad \textbf{(3.2)}$$

Between *three* stations, L0, L1 and L2, there are two such equations 3.3 and 3.4:

$$c \cdot dt_{01} = L0L1 \cdot \cos(\varphi - \varphi_1) \cdot \sin(\theta) \qquad \textbf{(3.3)}$$

$$c \cdot dt_{02} = L0L2 \cdot \cos(\varphi - \varphi_2) \cdot \sin(\theta) \qquad \textbf{(3.4)}$$

dt = Measured time difference between two events in two stations
$L_0L_1$ = Distance between two stations
c = Speed of light

With these equations you can calculate the angles $\theta$ and $\varphi$:

$$\tan(\varphi) = \frac{k\cos(\varphi_1) - \cos(\varphi_2)}{\sin(\varphi_2) - k\sin(\varphi_2)}$$

$$\text{, where } k = \frac{L0L2}{L0L1} \cdot \frac{dt_{01}}{dt_{02}}$$

$$\sin(\theta) = \frac{c \cdot dt_{01}}{L0L1 \cdot \cos(\varphi - \varphi_1)}$$

Now we know what the direction of the shower is. We can use this direction to calculate the time needed for a **shower front** to hit the detector of station 1 after it had hit the detector of station 2, see figure 2. We call this time the calculated time, dtc**,** and it is easily calculated using the path length difference s. We assume here also that the shower front is moving with the speed of light.

To calculate the direction of the shower we simplified reality within reasonable boundaries. I want to summarize these assumptions and clarify their effects on the measured and the calculated time:

### 3.3.1 Assumption 1: Shower front has a flat front

In figure 3.5 we simplify the real physical situation. The shower front is represented as a 'straight' front while in reality it's more 'circular' shaped. The effect is that the time measured differences slightly from the time calculated according to the direction of the shower. This effect is larger when the stations are more separated and when the shower has a large zenith angle. Offcourse when the zenith angle is zero than the time difference, assuming a flat front, is also zero. In reality the curvature of the shower causes deviations!

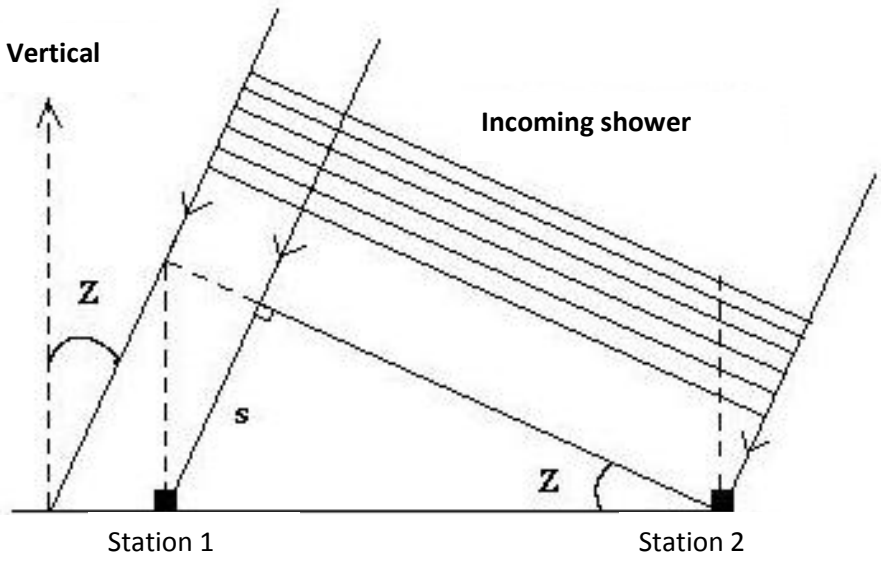



Figure 3.5: Incoming shower front and difference in detection time between two stations, 1 and 2!

### 3.3.2 Assumption 2: The Earth is flat

The other simplification is that we assume that the Earth is flat! This way of thinking results also in time differences between dt and dtc. This effect is also larger when the two stations are further apart. Eventually we are searching for evidence of the GZ-effect, thus for coincidences between two stations that are far apart, the time differences will be even greater than inside a cluster. Keep in mind that in the case of a coincidence between stations 100 or more kilometres apart it's not one shower anymore which causes that coincidence, but two!

### 3.3.3 Assumption 3: We measure the first particle of the shower

We assume that each event measures the first particle of the shower front. This is off course a misleading thought. You measure a particle, but it's impossible to know which particle of the shower it is. This effect creates an inaccuracy of the measurements of dt.

### 3.4 Plotting the measured time versus the calculated time graph

When we plot the measured time (dt on the x-axis) against the calculated time (dtc on the y-axis) we expect the following graph in general if the GZ-effect is really happening, see figure 3.6. In area A the slope has to be 45$^{o}$ especially near the origin. The measured time should be more or less the calculated time because the stations are close to each other in a cluster. The assumptions of the straight Earth and the straight shower don't effect the plot in this area that much.

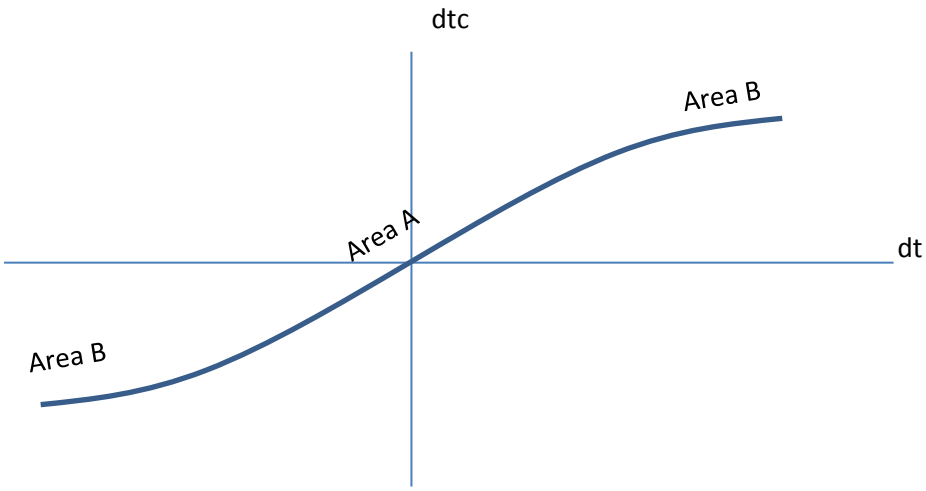



Figure 3.6:
Expected plot of the measured time against the calculated time

In area B, when dt becomes larger, the slope will go more straight. Thus dt is progressing more than dtc. The explanation for this flattening is that dt is influenced by the radius of curvature of the shower and the GZ-effect, see figure 3.7 on the next page. When a particle disintegrates and forms two new particles, there is a **radius of curvature** to take into account. The opening angle is very small compared to the opening angles of the showers created by both particles when entering the earth's atmosphere. If you compare the radius of curvatures of the shower and the photodisintergration process, the showers create smaller radii. Thus when dt becomes larger, the two stations are further apart and the effect of the radius of curvature becomes more apparent. The measured time between two events becomes larger than you would expect on the premise of the calculated time, presuming that the shower front is straight.
On the other hand it also shows that the GZ-effect is apparent in area B because the coincidence is caused by two showers separately .

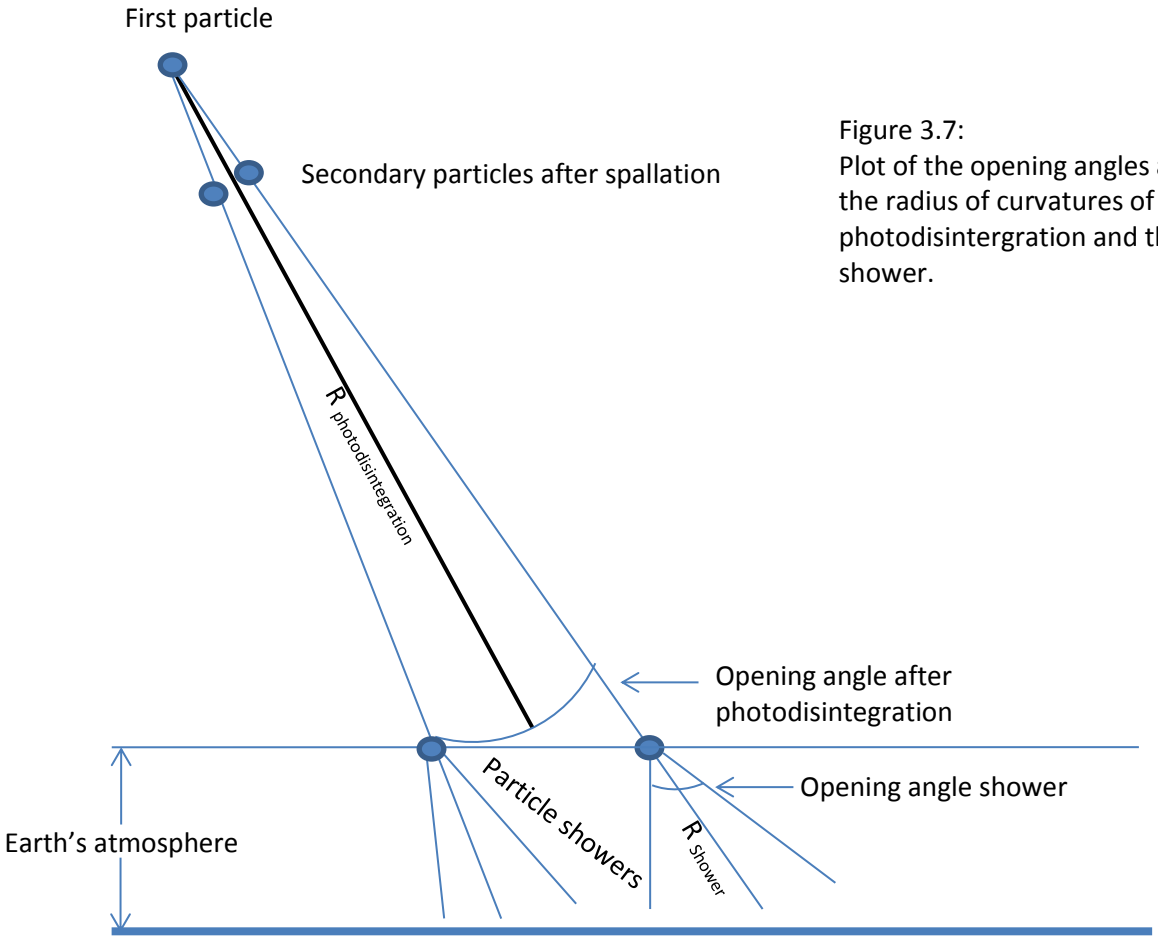



Figure 3.7:
Plot of the opening angles and the radius of curvatures of photodisintergration and the shower.

In conclusion we are looking for the flattening part of the plot. The greater the distance and/or dt the more logical it would be to find the GZ-effect caused by two different showers.

# 4. Statistical analysis: Feldman and Cousins

Conducting experimental science involves using statistics to interpret the relationship between measured values and the 'most likely' true values of some defined variables. In this experiment we are searching for coincidences between stations at different distances. This generates a lot data that will be analysed statistically to infer information about the true signal, within certain confidence levels.

### 4.1 Background versus signal

This whole research is about detecting cosmic ray particles in scintillators. These detections are called events. In our statistical analysis we want to know what the probability is of measuring value x, compared to the mean value μ. The Poisson distribution gives us a useful formula to find this probability [12].

$$p(x|\mu) = \frac{e^{-\mu}\mu^{x}}{x!} \qquad (4.1)$$

According to this formula, with a mean value of μ = 1,2 events per day, the probability that you measure 5 events is 0.00624.

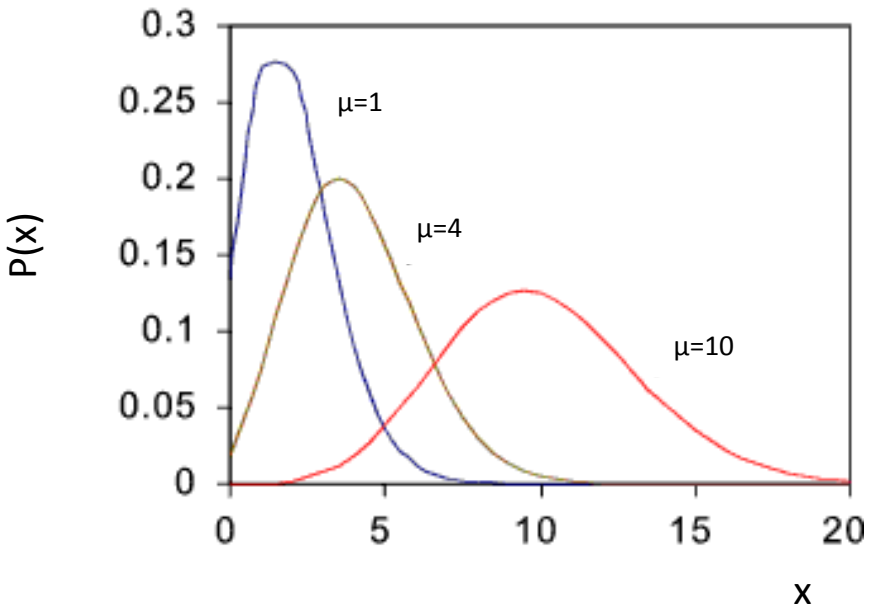



Figure 4.1: Poisson distribution. Different values of μ, compared to the measured value x.

In our research we don't know our mean value, we only measure events during a certain time and eventually try to find coincidences between stations. When we use this data we can derive a background signal b, as described above in section 3.2.2, and a measured value x. When taking this background signal into account, the formula for calculating the probability becomes the following [12].

$$p(x|\mu,b) = \frac{e^{-(\mu+b)}(\mu+b)^x}{x!} \qquad (4.2)$$

Although we can calculate probabilities with the Poisson distribution, we eventually want to calculate confidence levels of having a true signal compared to the background.

**4.2 From Neyman's classical intervals to Feldman and Cousins**

Neyman was the first to introduce confidence intervals. He says that the true value of a measurement is, with a certain probability, inside this interval or not. Thus the true value is fixed and can't be in the interval partly. The classical confidence interval with $\mu_{min}$ till $\mu_{max}$ has a certain confidence level β, which is an indication of the reliability of the estimation. So if we measure a certain x than the interval gives us a probability that the true value of μ is between $\mu_{min}$ and $\mu_{max}$ within the confidence level. The higher the confidence level the larger the interval.
Neyman's method was good at first but it produced intervals in unphysical regions. If for example the height above the ground is measured, the interval can't begin below zero, but with Neymans's method it unfortunately can.

In 1999 G.J. Feldman and R.D. Cousins [15] developed a different method to find upper and lower limits of the classical confidence interval from Neyman [16] dealing with Neyman's problem.
They introduced the likelihood-ratio:

$$R(x|) = \frac{P(x|\mu)}{P(x|\mu_{best})} \qquad (4.3)$$

It's a method to find the most likely mean value $\mu_{best}$ for each x that maximizes $P(x|\mu)$. The method is a way of submitting values of x with the largest R(x) to the acceptance region of μ. The second largest R(x) is then admitted etc. The whole procedure stops when the sum of all $P(x|\mu)$ is equal to the confidence level (for example 90%). Eventually you get a region $(x_{min}, x_{max}]$ where we find x in a fraction β of the cases, with a mean of μ.

Eventually we want to have a lower limit and an upper limit for every measurement within a confidence interval. When we use a real measurement where x = 1145 events and the background is 104,7 events with a 95% confidence level, we get an interval of [1271,6, 1421,88]. In this case there is a signal of at least 1271,6 with a certainty of 95%. This signal is very high, but it was taken at a distance of 15 meters between the detectors, so it must be single shower and thus a high signal is expected.

# 5. Results

The previous chapters explained the method of searching for coincidences. The final result can be provided as a table with data, and a figure to represent this table.

The table below (table 5.1) shows the data grouped in 20 different distance bins. Each distance bin represents all the stations that form a pair of stations within a certain distance. The numbers shown under signal events are a summation of all signals in one distance bin. The background is calculated as explained in paragraph 3.2.2. With this information we calculate the lower and upper limit shown in the table. Besides that also the mean is calculated, see figure 5.1.

| Bin # | Maximal Distance between stations (km) | # of signal events | # of background events | Lower Limit (LL) | Upper Limit (UL) | Factor of time needed for signal to appear |
|---|---|---|---|---|---|---|
| **0** | 0.0003 | 0 | 0.000542 | 0 | 3.092 | |
| **1** | 0.001 | 0 | 0.000542 | 0 | 3.092 | |
| **2** | 0.003 | 0 | 0.000542 | 0 | 3.092 | |
| **3** | 0.006 | 0 | 0.000542 | 0 | 3.092 | |
| **4** | 0.016 | 26633 | 1604.58 | 24715 | 25355 | |
| **5** | 0.04 | 258150 | 230996 | 26331 | 28397 | |
| **6** | 0.1 | 989754 | 28009 | 960061 | 964020 | |
| **7** | 0.25 | 460523 | 54519 | 404800 | 407563 | |
| **8** | 0.63 | 17320 | 3814 | 13250 | 13769 | |
| **9** | 1.58 | 333 | 2.4 | 296 | 368 | |
| **10** | 3.98 | 53 | 61.5 | 0 | 8.6 | |
| **11** | 10 | 167 | 162 | 0 | 31.7 | |
| **12** | 25 | 955 | 955.7 | 0 | 62 | |
| **13** | 63 | 2368 | 2582 | 0 | 16.5 | |
| **14** | 158 | 6577 | 6833 | 0 | 39.4 | |
| **15** | 398 | 9585 | 9730 | 0 | 86 | |
| **16** | 1000 | 853 | 836 | 0 | 75.9 | 27x |
| **17** | 2500 | 0 | 0.000542211 | 0 | 3.092 | |
| **18** | 6300 | 779 | 1399 | 0 | 3.116 | |
| **19** | 10000 | No data | - | - | - | - |

Table 5.1: Shows the data of all the different distances. The lower and upper limits are calculated with a confidence level of 95%. Per distance cluster also the total signal events and the background signal is shown.

Next, we plot the data of table 5.1 in figure 5.1. It shows the number of coincidences per day for every combination of cities. We included long and short distances to see also individual stations coincide. This is off course what you should expect when you detect a single shower. The figure shows a strong signal around 10 to 20 meters, so that gives confidence that our calculations are working well.

The results show a strong signal at very small distances (till 0.016 km). This signal is due to single showers in two detectors on one roof. After that there is a decrease (at 0.04 km) and then again an increase, which is due to the single air showers measured in clusters in a city. As expected we see with increasing distances a decrease in coincidences.

Our main focus of our research was to determine the Gerasimova-Zatsepin-effect. According to theory we expect that to occur between stations at distances of hundred km to several hundreds of kilometres away from each other. Looking closely at our table 5.1, we can't se any remarkable signal. The lower limit in that range is zero, which means that with an certainty of 95% there is no signal. On the other hand it looks promising. The data we used to create our table is only from 12 months. We calculated how much time we still have to collect data to see a signal appearing, provided that the progression is linear for the 'signal events' and the 'background signal'. We only did this for a distance of 1000 km because it is in the expected Gerasimova-Zatsepin effect region (see table 5.1) and furthermore it's the only candidate while it has a signal bigger than the background. You can also see in figure 5.1 that the mean signal is above the zero, which is promising.

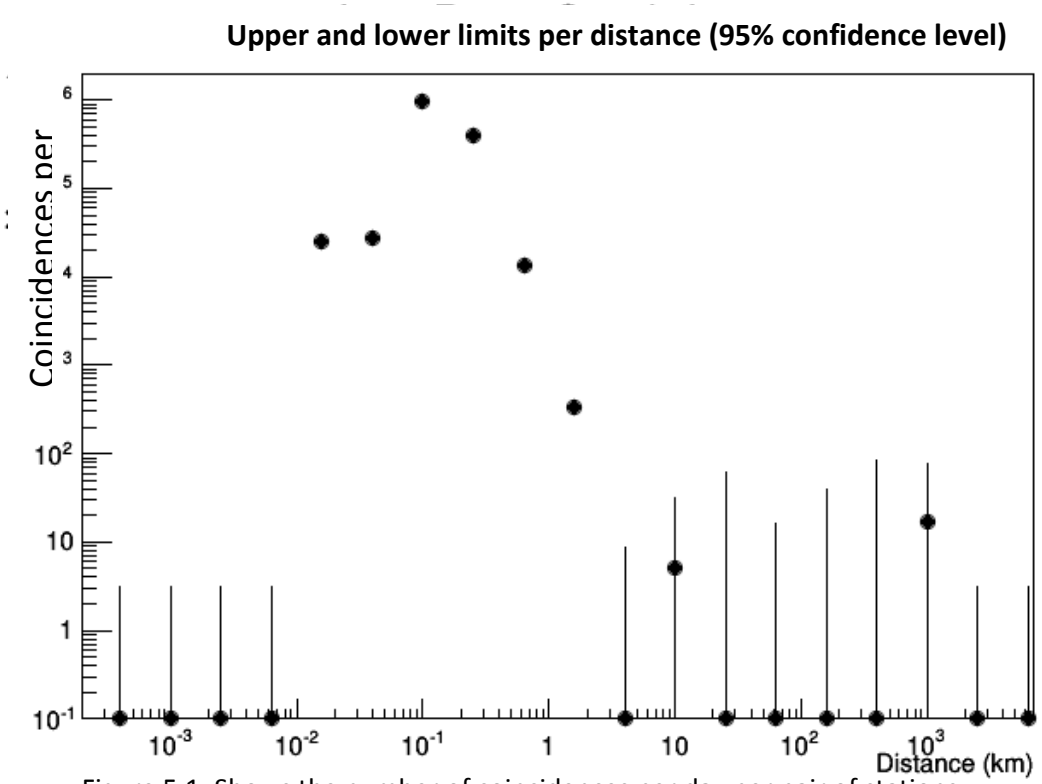



Figure 5.1: Shows the number of coincidences per day per pair of stations compared to the distance between the stations. Per clustered distance there is a upper limit and a lower limit with a confidence level of 95%. The dot represents the mean value.

# 6. Conclusion

The aim of this research was to find evidence for the Gerasimova-Zatsepin (GZ) effect. It's the effect that cosmic particles disintegrate via photodisintegration when entering our solar system. There is a chance that these fragments enter our Earth's atmosphere and form two separate air showers. Typically they should be hundred till several hundreds of kilometres away from each other, according to Lafebre et al. [13] and Medina and Watson [14].

Making use of the theory of cosmic radiation and the statistical analysis of Feldman and Cousins we tried to determine the magnitude of the GZ effect. For our analysis we used the data collected in twelve months via cosmic ray detectors of the HiSPARC network, mostly found on the rooftops of school buildings. Via the search for coincidences of events in different detectors, we hoped to see a strong signal at the distances described above.

The results of our analysis at this moment are not very distinct. We found signals at short distances, caused by single air showers, as expected. Unfortunately we didn't find a strong signal at the distances we were aiming at. There is a small peak present in bin 16 at a distance of 1000 km and that's promising. A simple calculation shows that we probably need as much as 27 times the amount of data we used now, to have a significant signal to say: 'we found something!'.

More research is needed to really say if the HiSPARC network is suitable for detecting the GZ effect. This analysis is a beginning but has to be looked at in more detail to create a better picture of what is happening and to filter all possible little mistakes.

# 7. Epilogue

This internship at HiSPARC was an experience I will always carry with me. Afterwards it's very clear to me that it's a necessity for educators to have worked in a scientific environment. The way scientist, doctoral students and master students work together and especially to see how they think was truly valuable to me. Besides that I was amazed by their readiness to help whenever I needed help. At first it felt awkward to ask them for explanations about all sort of subject, but very quick I learned that it's the normal way of working together.

Without their help I could never have finished my research the way I did now. I would like to thank my roommates for being patient with me and my questions. You always were fair and honest to me and critical to my writings. We also could had a good laugh sometimes and I always felt welcome to work with you. Thanks, Guus, Stefan and Stefan G., for making my internship really pleasant.

My special thanks goes to Charles Timmermans. He gave me the chance to have a look at real scientific research, although my educational background was not always profound enough. It started with the way you selected me for the internship at HiSPARC, on a rooftop with detectors. You wanted to see who I was in the right environment. I readily saw that you had experience with people like me and I liked your way of thinking. Through the whole period you supported me greatly, were always ready to talk to me and you helped me focussing when it was needed. Thanks a lot for your support and critical view. I enjoyed working with you.

Thanks everybody,

Dave

# Bibliography


[1] M. Ackermann. (2013). *Detection of the characteristic Pion-decay signature in Supernova Remnants* Science. Vol339 no 6121 pp 801-811

[2] George, J.S. (2009). *ELEMENTAL COMPOSITION AND ENERGY SPECTRA OF GALACTIC COSMIC RAYS DURING SOLAR CYCLE 23.* Astroph. J 698, 16666-1681

[3] Zatsepin, G. T. (1951). *On the photodisintegration of heavy cosmic-ray particles by solar radiation*. Dokl. Akad. Nauk. SSSR, 80, 577–578.

[4] Gerasimova, N. M. and Zatsepin, G. T. (1960). *Splitting of cosmic ray.* Sov. Phys. JETP vol. 11 , 899:

[5] Gerasimova, N. M. and Zatsepin, G. T. (1960). *nuclei by solar photons*. ZhETF, 38, 1245–1252, 1960.

[6] Ans nuclear cafe (2013). *Cosmic ray cascade chart.* Geraadpleegd op 28 november 2013 via http://ansnuclearcafe.org/2011/08/17/up-up-and-away-victor-hess-and-the-cosmic-ray/cosmic-ray-cascade-chart

[7] teachers.web.cern.ch (2014). *Schematic representation of an electron initiated electromagnetic cascade.* Geraadpleegd op 3 december 2013 via http://teachers.web.cern.ch/teachers/archiv/HST2000/teaching/expt/muons/cascades.htm

[8] Awater, H. (2005). *Bepaling van de energie van kosmische deeltjes uit deeltjeslawine in de atmosfeer.* Pagina 11.

[9] HiSPARC (2014). HiSPARC website geraadpleegd op 15 januari 2014 via http://www.HiSPARC.nl/over-HiSPARC/HiSPARC-detector

[10] Wikipedia (2014). Particle shower. Geraadpleegd op 10 januari 2014 via http://en.wikipedia.org/wiki/Particle_shower

[11] Stanev, T. (2004). *High energy cosmic rays second edition*. Page 185-186. Springer: praxis.

[12] Peters, M. (2011). *Using the HiSPARC network to measure the Gerasimova-Zatsepin.*

[13] Lafebre S. et al. (2008). *Prospects for direct cosmic ray mass measurements through the Gerasimova Zatsepin.* ect. Astronomy and Astrophysics, 485:1.

[14] Medina-Tanco G.A. and Watson A.A. (1999)*The photodisintegration of cosmic ray nuclei by solar photons: the gerasimova-zatsepin. ect revisited.* Astroparticles Physics, 10:157 {164}.

[15] Feldman G.J and Cousins R.D. (1998). *A unified approach to the classical statistical analysis of small signals.* Phys. Rev. D 57, 3873.

[16] Neyman J. (1935). *On the problem of confidence intervals. Ann. Math. Stat., 6:111-116.*

[17] Hermsen E. (2008). Stage HiSPARC. HEN, 463.

# Appendix A: Definitions of concepts

**cascade:**
In particle physics, a **shower** is a cascade of secondary particles produced as the result of a high-energy particle interacting with dense matter. The incoming particle interacts, producing multiple new particles with lesser energy; each of these then interacts in the same way, a process that continues until many thousands, millions, or even billions of low-energy particles are produced. These are then stopped in the matter and absorbed.

**cluster:**
A couple of detector stations clustered in one city

**coincidence:**
A coincidence is a collection of two or more events related in time

**cosmic rays:**
**Cosmic rays** are very high-energy particles, mainly originating outside the Solar System. They may produce **showers** of **secondary particles** that penetrate and impact the Earth's atmosphere and sometimes even reach the surface. They consist of protons and different kind of nuclei.

**cosmic shower or extended air shower (EAS)**
An cosmic shower is an cascade of electromagnetic and ionized particles in our earth's atmosphere. It is triggered when a primary particle hits an atom in our atmosphere.

**eV**
This is a unit for the energy of a primary particle. It's the abbreviation of electron Volt.

**Event**
A detection of a particle in a detector. Every detector on a single roof consists of two separate detectors. When both detectors detect particles within a certain time window then we call it an event, meaning we are detecting part of a shower!

**GZ effect**
The effect that photodisintegration of cosmic rays with solar photons produces two nuclei

**interstellar medium**
interstellar medium is the matter that exists in the space between the star systems in a galaxy.

**muons**
A muon is similar to an electron, but it's 207x heavier and it has a different spin.

**opening angle**
This is the angle of the shower after the primary particle hits an atom in our atmosphere

**photodisintegration**

Photodisintegration is a physical process where an atomic nucleus emits a subatomic particle after an extremely high energy gamma ray is absorbed by an atomic nucleus and caused it to enter an excited state.

**spin**

It's a fundamental characteristic of a nucleus or particle. Just like charge is also a characteristic of an electron.

**Appendix B: Used algorithms**

We used C and C++ as our programming languages.

### 1. Angles file

With this file we calculate the angles of incidence of a shower!

---

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>

typedef struct{
  int stat;
  float lat;
  float lng;
  float x;
  float y;
  float z;
}Station;
Station HiSPARC[200];

int nHiSPARC=0;

int evt,stat[100],sec[100],nsec[100];
float mip[100];

long int nsdt(ia,ib){
  long int idt;
#define GIGA 1000000000
  idt = GIGA*(sec[ia]-sec[ib]);
  idt+=(nsec[ia]-nsec[ib]);
  return(idt);
}


int calc_angles(int ev1, int ev2, int ev3, float *phi, float *theta)
{
#define SPLIGHT 0.3 // m/ns
#define PI 3.14159
  float k;
  float L12,L13;
  float phi1,phi2,phit,thetat;
  int ih,h1,h2,h3;

  h1 = -1;
  h2 = -1;
  h3 = -1;
  for(ih=0;ih<nHiSPARC;ih++){
    if(HiSPARC[ih].stat == stat[ev1]) h1 = ih;
    if(HiSPARC[ih].stat == stat[ev2]) h2 = ih;
    if(HiSPARC[ih].stat == stat[ev3]) h3 = ih;
  }
  if(h1 == -1 || h2 == -1 || h3 == -1) return(-1);
```

```
  L12 = (HiSPARC[h1].x-HiSPARC[h2].x)*(HiSPARC[h1].x-HiSPARC[h2].x)+
    (HiSPARC[h1].y-HiSPARC[h2].y)*(HiSPARC[h1].y-HiSPARC[h2].y);
  L12 = sqrt(L12);
  L13 = (HiSPARC[h1].x-HiSPARC[h3].x)*(HiSPARC[h1].x-HiSPARC[h3].x)+
    (HiSPARC[h1].y-HiSPARC[h3].y)*(HiSPARC[h1].y-HiSPARC[h3].y);
  L13 = sqrt(L13);
  if(L12>10000. || L13 > 10000.) return(-2); // only within 10 km
  if(fabs(SPLIGHT*nsdt(ev1,ev2)) > L12 ||
fabs(SPLIGHT*nsdt(ev1,ev3)) > L13) return(-3);
  phi1 = atan2(HiSPARC[h2].y-HiSPARC[h1].y,HiSPARC[h2].x-
HiSPARC[h1].x);
  phi2 = atan2(HiSPARC[h3].y-HiSPARC[h1].y,HiSPARC[h3].x-
HiSPARC[h1].x);
  phit = atan2(L13*nsdt(ev1,ev2)*cos(phi2)-
L12*nsdt(ev1,ev3)*cos(phi1),
            L12*nsdt(ev1,ev3)*sin(phi1)-
L13*nsdt(ev1,ev2)*sin(phi2));
  //printf("phi12=%g L12=%g
c*DT12=%g\n",57.3*phi1,L12,SPLIGHT*nsdt(ev1,ev2));
  //printf("phi13=%g L13=%g
c*DT13=%g\n",57.3*phi2,L13,SPLIGHT*nsdt(ev1,ev3));
  if((cos(phi1-phit)*nsdt(ev1,ev2))<0)phit+=PI;
  if(phit<0) phit+=2*PI;
  if(phit> (2*PI)) phit -=2*PI;
  k = SPLIGHT*nsdt(ev1,ev2)/(L12*cos(phit - phi1));
  if(k>1. || k < -1.) {
    // printf("Error: %g %g
(%g,%g),(%g,%g)\n",k,SPLIGHT*nsdt(ev1,ev3)/(L13*cos(phit - phi2)),
    //
fabs(SPLIGHT*nsdt(ev1,ev2)),L12,fabs(SPLIGHT*nsdt(ev1,ev3)),L13);
    return(-4);
  }
  thetat = asin(k);
  //printf("%d =? %g\n",(int)nsdt(ev1,ev2),L12*cos(phi1-
phit)*sin(thetat)/SPLIGHT);
  //printf("%d =? %g\n",(int)nsdt(ev1,ev3),L13*cos(phi2-
phit)*sin(thetat)/SPLIGHT);
  if(thetat < 0) {
    thetat = -thetat;
    phit = phit+PI;
  }
  *phi = phit;
  *theta = thetat;
  return(1);
}

int calctime(int ev1,int ev2,float phi,float theta,float chimin)
{
#define SPLIGHT 0.3 // m/ns
#define PI 3.14159
  float L12;
  float phi1,chi2;
  float dtc,dt;
  int ih,h1,h2;
  int ir=0;
```

```
  h1 = -1;
  h2 = -1;
  for(ih=0;ih<nHiSPARC;ih++){
    if(HiSPARC[ih].stat == stat[ev1]) h1 = ih;
    if(HiSPARC[ih].stat == stat[ev2]) h2 = ih;
  }
  L12 = (HiSPARC[h1].x-HiSPARC[h2].x)*(HiSPARC[h1].x-HiSPARC[h2].x)+
    (HiSPARC[h1].y-HiSPARC[h2].y)*(HiSPARC[h1].y-HiSPARC[h2].y);
  L12 = sqrt(L12);
  phi1 = atan2(HiSPARC[h2].y-HiSPARC[h1].y,HiSPARC[h2].x-
HiSPARC[h1].x);
  if(L12>10000. && fabs(SPLIGHT*nsdt(ev1,ev2)) < L12){
    dt = nsdt(ev1,ev2);
    dtc = L12*cos(phi1-phi)*sin(theta)/SPLIGHT;
    if((fabs(dt-dtc)/fabs(dt))<0.01){
      printf("%g-%g = %g (%g,%g,%g)\n",dt,dtc,dt-
dtc,L12/1000.,fabs(dt-dtc)/fabs(dt),chimin);
      ir = 1;
    }
  }
  return(ir);
}

float calc_chi2(int ev1, int nevt, float phi, float theta)
{
#define SPLIGHT 0.3 // m/ns
#define PI 3.14159
  int iev,ndf;
  float L12;
  float phi1,chi2;
  int ih,h1,h2;

  h1 = -1;
  h2 = -1;
  for(ih=0;ih<nHiSPARC;ih++){
    if(HiSPARC[ih].stat == stat[ev1]) h1 = ih;
  }
  chi2 = 0;
  ndf = 0;
  for(iev=0;iev<nevt;iev++){
    for(ih=0;ih<nHiSPARC;ih++){
      if(HiSPARC[ih].stat == stat[iev]) h2 = ih;
    }
    L12 = (HiSPARC[h1].x-HiSPARC[h2].x)*(HiSPARC[h1].x-
HiSPARC[h2].x)+
      (HiSPARC[h1].y-HiSPARC[h2].y)*(HiSPARC[h1].y-HiSPARC[h2].y);
    L12 = sqrt(L12);
    if(L12>10000.) continue; // only within 10 km
    if(fabs(SPLIGHT*nsdt(ev1,iev)) > L12) continue;
    phi1 = atan2(HiSPARC[h2].y-HiSPARC[h1].y,HiSPARC[h2].x-
HiSPARC[h1].x);
    chi2+=(nsdt(ev1,iev)-L12*cos(phi1-
phi)*sin(theta)/SPLIGHT)*(nsdt(ev1,iev)-L12*cos(phi1-
phi)*sin(theta)/SPLIGHT);
    //printf("Ev=(%d,%d) DT=%g L=%g\n",ev1,iev,(nsdt(ev1,iev)-
L12*cos(phi1-phi)*sin(theta)/SPLIGHT),L12);
```

```
    ndf++;
  }
  //printf("Chi=%g/%d\n",chi2,ndf);
  return(chi2/(25*ndf));
}

main()
{
  FILE *fp;
  int prevevt=-1;
  int nevt=0;
  int i,ip,month;
  int dt;
  int nlines=0;
  char line[200],fname[200];
  int ih1,ih2,ih3;
  float phi, theta;
  float rdist;
  float chi,chimin,phimin,thmin;
  int evmin;

  fp = fopen("location.tbl","r");
  while (fgets(line,199,fp) == line){
    if(line[0] == '#') continue;
    if(sscanf(line,"%d %g %g %g %g
%g",&(HiSPARC[nHiSPARC].stat),&(HiSPARC[nHiSPARC].lat),&(HiSPARC[nHi
SPARC].lng),

&(HiSPARC[nHiSPARC].z),&(HiSPARC[nHiSPARC].x),&(HiSPARC[nHiSPARC].y)
)> 0) nHiSPARC++;
  }
  fclose(fp);

  for(month=6;month<9;month++){
    sprintf(fname,"coin2013_%d",month);
    fp=fopen(fname,"r");
    while (fgets(line,199,fp) == line){
      nlines++;
      if(sscanf(line,"%d %d %d.%09d
%g",&evt,&stat[nevt],&sec[nevt],&nsec[nevt],&mip[nevt])<=0){
     printf("Error line %d !!%s!!\n",nlines,line);
     continue;
      }
      if(evt != prevevt && nevt !=0){ // done!
     if(nevt >= 3){
       //printf("--------------------------------\n");
       chimin = -1;
       for(ih1=0;ih1<(nevt-2);ih1++){
         for(ih2=ih1+1;ih2<(nevt-1);ih2++){
           for(ih3=ih2+1;ih3<nevt;ih3++){
          if(calc_angles(ih1,ih2,ih3,&phi,&theta)>0) {
            chi = calc_chi2(ih1,nevt,phi,theta);
            if(chi<chimin || chimin<0){
              chimin = chi;
              phimin = phi;
              thmin = theta;
```

```
                evmin = ih1;
              }
            }
             }
           }
         }
         ih2 = 0;
         for(ih1=0;(ih1<nevt&&chimin>-0.1);ih1++){
           ih2+=calctime(evmin,ih1,phimin,thmin,chimin);
         }
         //   printf("%d Nevt=%d (%d) %g %g
Chi=%g\n",prevevt,nevt,evmin,phimin*57.3,thmin*57.3,chimin);
         if(ih2>0) printf("============================\n");
       }
       stat[0]=stat[nevt];
       sec[0]=sec[nevt];
       nsec[0]=nsec[nevt];
       mip[0]=mip[nevt];
       nevt=0;
       //break;
        }
       nevt++;
       prevevt = evt;
     }
     fclose(fp);
     printf("Read %d lines\n",nlines);
   }
}
```

2. H5read file

```
#include "hdf5.h"
#include<sys/time.h>
#include<string.h>
#include<math.h>

#define MAXDATA 500000
#define MAXEVTS 7000000
int station_id;

/*
typedef struct{
  int16_t baseline[4];          //0
  int8_t datared;               //8
  uint32_t event_id;            //9
  float event_rate;             //13
  int64_t ext_timestamp;        //17
  int32_t integrals[4];         //25
  float n1;                     //41
  float n2;                     //45
  float n3;                     //49
  float n4;                     //53
  int16_t n_peaks[4];           //57
  int32_t nanoseconds;          //65
  int16_t pheight[4];           //69
  int16_t std_dev[4];           //77
  float t1;                     //85
  float t2;                     //89
  float t3;                     //93
  float t4;                     //97
  uint32_t timestamp;           //101
  int32_t traces[4];            //105
  uint32_t trigger;             //121
}HiSPARC;
*/
typedef struct{
  int station;
  int sec;
  int nsec;
  float nmip;
  int ip;
}event;

typedef struct{
  int stat;
  float lat;
  float lng;
  float x;
  float y;
  float z;
}Station;
Station HiSPARC[200];
int nHiSPARC = 0;


char buffer[125*MAXDATA];
event evdata[MAXEVTS];

int n_evdata = 0;

static herr_t iter_func(void *elem,hid_t type_id, unsigned ndim, const hsize_t
*point, void *operator_data)
{
  int i;
  char *bf;
  int sec;
  bf = (char *)elem;
```

```
  //printf("Npoint = %d Ndim = %d Type =
%d\n",(int)*point,ndim,(int)H5Tget_class(type_id));
  //printf("%d %d %d",dat->event_id,dat->timestamp,dat->nanoseconds);
  memcpy(&sec,&bf[9],4);
  //printf("Sec = %d\n",sec);
  return(0);
}
void read_data(hid_t dp)
{
  hid_t ds,dtype,stid,ttid,memspace,ds2;
  int i,j,rank,coord[2];
  int sec,nsec;
  float nmip[2];
  hsize_t dims[1];
  hssize_t np;
  size_t size;
  H5T_class_t t_class;                 /* data type class */
  H5T_order_t order;                  /* data order */
  char
*stype[]={"INTEGER","FLOAT","TIME","STRING","BITFIELD","OPAQUE","COMPOUND","REFEREN
CE","ENUM","VLEN","ARRAY"};
  //char buffer[125];

  dtype  = H5Dget_type(dp);
  size  = H5Tget_size(dtype);
  t_class     = H5Tget_class(dtype);
  //if ((int)t_class >=0) printf("Data set has %s type. Nmembers = %d
\n",stype[(int)t_class],H5Tget_nmembers(dtype));
  for(i=0;i<H5Tget_nmembers(dtype);i++) {
    ttid = H5Tget_member_type(dtype,i);
    t_class     = H5Tget_class(ttid);
    //printf("Name %s Type %s\n",H5Tget_member_name(dtype, i ),stype[t_class]);
    H5Tclose(ttid);
  }
  order     = H5Tget_order(dtype);
  //if (order == H5T_ORDER_LE) printf("Little endian order \n");
  ds = H5Dget_space(dp);
  rank      = H5Sget_simple_extent_ndims(ds);
  np = H5Sget_select_npoints(ds);
  if(np>MAXDATA) {
    fprintf(stderr,"N points = %d Size = %d Rank %d \n",(int)np,(int)size,rank);
  } else{
    //H5Diterate(buffer,dtype,ds,iter_func, NULL);
    H5Dread(dp,dtype,H5S_ALL,H5S_ALL,H5P_DEFAULT,buffer);
    for(i=0;i<np;i++){
      memcpy(&sec,&buffer[125*i+101],4);
      memcpy(&nsec,&buffer[125*i+65],4);
      memcpy(nmip,&buffer[125*i+41],8);
      evdata[n_evdata].sec = sec;
      evdata[n_evdata].nsec = nsec;
      evdata[n_evdata].nmip = nmip[0]+nmip[1];
      evdata[n_evdata].ip = 0;
      evdata[n_evdata].station = station_id;
      if(n_evdata<MAXEVTS) n_evdata++;
      else{
       fprintf(stderr,"Memory too small for all events\n");
      }
      //printf("Station=%d sec = %d.%09d Nmip =
%g\n",station_id,sec,nsec,nmip[0]+nmip[1]);
    }
  }
  H5Tclose(dtype);
  H5Sclose(ds);
}

void loop_group(hid_t fp)
{
  hid_t dp;
```

```
  H5G_info_t gi;
  H5O_info_t oi;
  char name[100];
  int iret;
  int i;


  iret = H5Gget_info(fp,&gi);
  if(iret<0) printf("An error occured %d\n",(int)iret);
  for(i=0;i<(int)gi.nlinks;i++){

H5Lget_name_by_idx(fp,".",H5_INDEX_NAME,H5_ITER_NATIVE,(hsize_t)i,name,99,H5P_DEFAU
LT);
    //printf("Object name: !!%s!!\n",name);
    if(strncmp(name,"station",7) == 0) sscanf(name,"station_%d",&station_id);
    if(H5Oget_info_by_name(fp,name,&oi,H5P_DEFAULT)<0) break;
    if(oi.type == 0){
      dp = H5Gopen(fp,name,H5P_DEFAULT);
      loop_group(dp);
      H5Gclose(dp);
    }
    else if(oi.type == 1){
      if (strcmp(name,"events") == 0 ){
       dp = H5Dopen(fp,name,H5P_DEFAULT);
       read_data(dp);
       H5Dclose(dp);
       }
    }
  }
}
int ev_compare(const void *a, const void *b)
{ /* sorting in order*/
  event *t1,*t2;
  t1 = (event *)a;
  t2 = (event *)b;
  if(t1->sec < t2->sec) return(-1);
  if(t1->sec == t2->sec){
    if(t1->nsec < t2->nsec) return(-1);
    else if(t2->nsec < t1->nsec) return(1);
    else return(0);
  }
  return(1);
}
printevt(int ie,int iev)
{
  //printf("%6d %6d %d.%09d
%g\n",ie,evdata[iev].station,evdata[iev].sec,evdata[iev].nsec,evdata[iev].nmip);
  evdata[iev].ip = ie;
}

long int mudt(int ia,int ib){
  long int idt;
#define MEGA 1000000
  idt = MEGA*(evdata[ia].sec-evdata[ib].sec);
  idt+=((evdata[ia].nsec-evdata[ib].nsec)/1000);
  if(idt<0) idt = -1*idt;
  return(idt);
}

float splight_time(int ia,int ib)
{
  float rdist;
  long int dt;

  int ih,ih1,ih2;
  ih1 = -1;
  ih2 = -1;
  for(ih=0;ih<nHiSPARC;ih++){
```

```
    if(HiSPARC[ih].stat == evdata[ia].station) ih1 = ih;
    if(HiSPARC[ih].stat == evdata[ib].station) ih2 = ih;
  }
  if(ih1 <0 || ih2< 0) {
    rdist = -1;
  } else {
    rdist = (HiSPARC[ih1].x-HiSPARC[ih2].x)*(HiSPARC[ih1].x-HiSPARC[ih2].x)+
      (HiSPARC[ih1].y-HiSPARC[ih2].y)*(HiSPARC[ih1].y-HiSPARC[ih2].y);
    rdist = sqrt(rdist)/1000.; //distance in km
    //printf("%d %d %d %d %g\n",HiSPARC[ih1].stat,HiSPARC[ih2].stat,stat[i],stat[i-
1],rdist);
  }
  dt = mudt(ia,ib);
  if(dt <0) dt = -1*dt;
  if(rdist > 0.01) rdist = 0.3*dt/rdist; //0.3 musec/km
  else rdist = 999;
  return(rdist);
}

read_setup()
{
  FILE *fp;
  char line[200];
  fp = fopen("location.tbl","r");
  while (fgets(line,199,fp) == line){
    if(line[0] == '#') continue;
    if(sscanf(line,"%d %g %g %g %g
%g",&(HiSPARC[nHiSPARC].stat),&(HiSPARC[nHiSPARC].lat),&(HiSPARC[nHiSPARC].lng),

&(HiSPARC[nHiSPARC].z),&(HiSPARC[nHiSPARC].x),&(HiSPARC[nHiSPARC].y))> 0)
nHiSPARC++;
  }
  fclose(fp);
}
main(int argc,char **argv)
{
  hid_t fp;
  int i,ind,ievt,iok; //wat betekenen al die woorden?!
  int istart,iend;
  int ncl,stdif,ind2;

  read_setup();
  fp = H5Fopen(argv[1],H5F_ACC_RDONLY,H5P_DEFAULT);
  //printf("Group / \n");
  loop_group(fp);
  H5Fclose(fp);
  //printf("Read %d events\n",n_evdata);
  qsort(evdata,n_evdata,sizeof(event),ev_compare);
  ievt = 0;
  iok = 0;
  istart = -1;
  iend = -1;
  for(i=1;i<n_evdata;i++){
    if((evdata[i].sec-evdata[i-1].sec)<1){ //waarom kleinder dan 2, randgevallen
uitsluiten!
      if(mudt(i,i-1)<100 ){ //verschil in tijd tussen twee stations is 100
mircoseconde of kleiner
       if(istart<0) istart = i-1;
       iend = i;
       iok = 1;
       if(evdata[i-1].ip == 0) printevt(ievt,i-1);
       if(evdata[i].ip == 0) printevt(ievt,i);
      }  else {
       if(iok == 1){
         if((iend-istart)>2){
           for(ind=istart;ind<=iend;ind++){ //index
             ncl = 0;
             for(ind2=ind;ind2<=iend;ind2++){
```

```
              stdif = evdata[ind].station - evdata[ind2].station; //stations
difference
              if(stdif<0) stdif = -1*stdif;
              if(stdif<50) ncl++;
             }
             if(ncl>=3) break; // wat betekent ncl? aantal stations in cluster?
           }
           if(ncl>=3){        //als cluster groter is dan 3 stations dan...
             for(ind2=istart-1;ind2>=0;ind2--){
              if(mudt(istart,ind2)>=50000) break; // als het verschil groter is dan
50000 microseconde?
             }
             for(iend=iend;iend<n_evdata;iend++){
              if(mudt(iend,istart)>=50000) break;
             }
             for(ind=ind2;ind<=iend;ind++){
              if(splight_time(ind,istart)<10 || mudt(ind,istart)<100|| ind ==
istart){ // of kleiner dan 10 microseconde of kleiner dan 100 microseconde
                // of ind is gelijk aan Istart?
                printf("%6d %6d %d.%09d
%g\n",ievt,evdata[ind].station,evdata[ind].sec,evdata[ind].nsec,evdata[ind].nmip);
                //betekenis nmip? minimum ionising particles.
              }
             }
           }
         }
         ievt++;
         iok = 0;
       }
       istart = -1;
       iend = -1;
      }
    }
  }
}
```

3. Mkhist file

```
{
  FILE *fp;
  int evt,stat,sec,nsec,cldt,cldta;
  float mip,rdist,opang;
  int prevevt=-1;
  int nevt=0;
  int i,ip,month;
  int dt;
  int nlines=0;
  char line[200],fname[200];
  typedef struct{
    int stat;
    float lat;
    float lng;
    float x;
    float y;
    float z;
  }Station;
  Station HiSPARC[200];
  int nHiSPARC=0;
  int ih,ih1,ih2;
  TH1F *hdt=new TH1F("HDT","Time Diff",1000,0.,100000);
  TH2F *hdtds=new TH2F("HDTDS","Time Diff vs dist",10,0,10,20,-
1.,7.);
  TH2F *hdtda=new TH2F("HDTDA","Time Diff vs dist",10,0.0,10.0,20,-
1.,7.);
  TH1F *hop = new TH1F("HOP","OP ang",180,0.,180.);

  fp = fopen("location.tbl","r");
  while (fgets(line,199,fp) == line){
    if(line[0] == "#") continue;
    if(sscanf(line,"%d %g %g %g %g
%g",&(HiSPARC[nHiSPARC].stat),&(HiSPARC[nHiSPARC].lat),&(HiSPARC[nHi
SPARC].lng),

&(HiSPARC[nHiSPARC].z),&(HiSPARC[nHiSPARC].x),&(HiSPARC[nHiSPARC].y)
)> 0) nHiSPARC++;
  }
  fclose(fp);

  sprintf(fname,"clustercoin");
  fp=fopen(fname,"r");
  while (fgets(line,199,fp) == line){
    nlines++;
    if(sscanf(line,"%d %d %d.%09d %g %g %d %d
%g",&evt,&stat,&sec,&nsec,&mip,&rdist,&cldt,&cldta,&opang)<=0){
      printf("Error line %d !!%s!!\n",nlines,line);
     continue;
    }
    //printf("%s %d %d %g\n",line,evt,cldt,rdist);
    if(rdist>0) {
     // if(rdist>2000 && rdist < 10000){
```

```
     //     printf("%d %d %g %g %g\n",evt,stat,rdist,cldt,cldta);
     // }
        if(rdist>2.5E5 && rdist < 1.5E6
&&(0.3*fabs(cldt)/rdist)<1.){
           printf("%d %d %g
(%g)\n",stat,evt,rdist,(0.3*cldt/rdist));
        }
      hdtds->Fill(0.3*fabs(cldt)/rdist,log(rdist)/log(10.));
      hdtda->Fill(0.3*fabs(cldta)/rdist,log(rdist)/log(10.));
      if(opang > 1.e-4 && rdist>500000.) hop->Fill(57.3*opang);
    }
    //break;
  }
  fclose(fp);
  printf("Read %d lines\n",nlines);
}
```