Fortgeschrittenen-Praktikum LA WS 13/14
FP 2 experiment

Z^0-resonance

Jonas Hiestand, Steffen Ludwig

24.03. - 04.04.2014

Inhaltsverzeichnis

1 Definition of task

The Z^0 boson is a gauge boson of the weak interaction. The experiment's goal is to measure the decay width and the particle's mass. Therefore data from the LEP-Collider at CERN is used. The collision of electrons and positrons produces Z^0-resonance. The particle's decay in several channels is detected by the OPAL-detector.
The single steps of the experiment are:

- Computing theoretically the following values:
  - the decay widths of the different charged leptons Γ_e, Γ_μ, Γ_τ, of the neutrinos $\Gamma_{\nu(e)}$, $\Gamma_{\nu(\mu)}$, $\Gamma_{\nu(\tau)}$ and of the quarks Γ_u, Γ_d, Γ_c, Γ_s, Γ_b.
  - the total decay width Γ_Z, the charged leptonic width $\Gamma_{lep,\ total}$, the neutral leptonic width $\Gamma_{\nu,\ total}$ and the hadron width $\Gamma_{q,\ total}$.
  - the partial cross section at the resonance maximum.
  - the percentage variation of the resonance width, if an additional fermion couple would be possible.
  - the angular distribution of the s- and t-channel in the e^+ e^- process.
- Analysing the different decays of the Z^0 with the programme GROPE (data of Monte Carlo method).
- Finding the cuts to separate the channels in the real data. Computing the efficiency matrix.
- Separating the s- and t-channel of the electron decay and correcting the number of electrons.
- Computing the real number of decays with the efficiency matrix.
- Computing the cross section with the luminosity and the number of events.
- Determining the mass of the Z^0 boson M_Z, the total width Γ_Z and the width of the four channels Γ_e, Γ_μ, Γ_τ and Γ_q with a fit of a Breit Wigner function.
- Computing the invisible width $\Gamma_{\nu,\ total}$ and the number of neutrino families.
- Determining the forwards backwards asymmetry and the weinberg angle $\sin^2\Theta_W$ for the Monte Carlo Data and the real Data at resonance maximum.

2 Theoretical background

2.1 The standard model of particle physics

The standard model is a theory, which describes the elementary particles and the electromagnetic, weak, and strong nuclear interactions. A distinction is made between the **fermions** with a spin of one half and the **bosons** with a spin of one. Fermions respect the Pauli exclusion principle. The elementary particles are including six flavours of quarks and six sorts of leptons (both are fermions) and also four different gauge bosons: the photon (electromagnetic force), eight sorts of gluons (strong force), the W^+ and W^- and the Z^0 boson (weak force). In addition there is the theoretical Higgs boson with a spin of zero.
Particles, which are composed of quarks by strong force are called hadrons. There is a distinction of the heavy **baryons** (like protons or neutrons), made up of three quarks, and the **mesons** (for example the pion), made up of a quark and an anti quark[1]. Because of this, baryons are fermions and mesons are bosons.

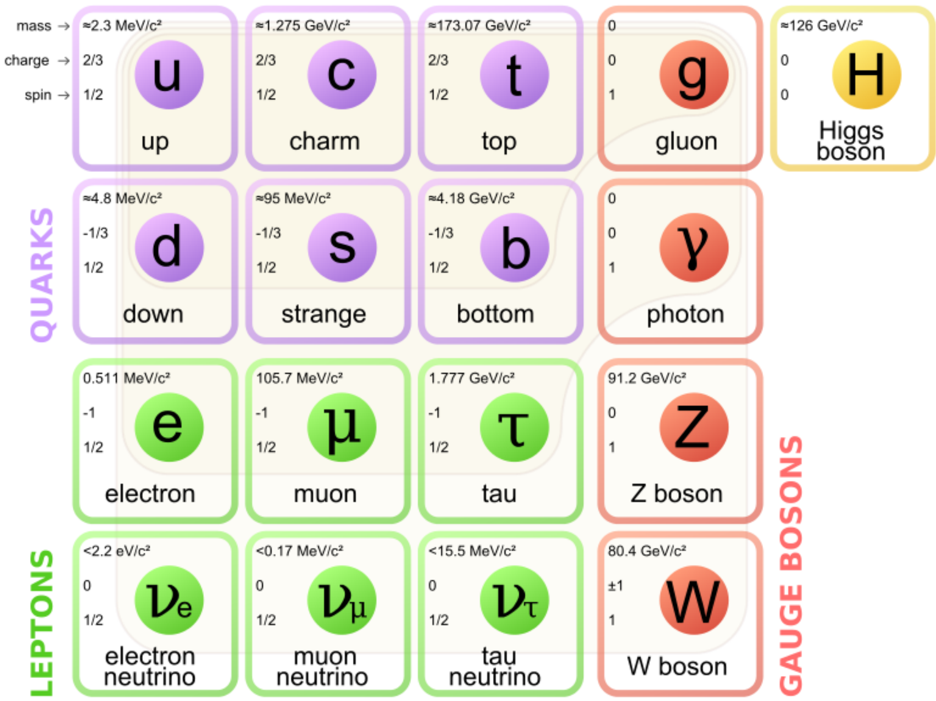

Abbildung 1: The standard model of elementary particles: six leptons, six quarks, four different guage bosons and the higgs boson.

Each of the particles has its own mass in eV, a specific charge and its spin, pictured in figure 1. **Virtual particles** are only existing for a limited time and are transient fluctuations. Interactions between ordinary particles are described as an exchange of virtual particles. They do not necessarily carry the same mass as the corresponding real particle because of the Heisenberg uncertainty principle.

[1] Antiparticles are particles with the same mass, but an opposite charge. The antiparticle of an electron for example is the positron.

2.1.1 Leptons

These ”light” particles do not undergo the strong interactions. There are six types of them, known as flavours: The electron, the muon and the tauon. Every particle has its specific neutrino, a neutral particle with nearly no mass. The heavy leptons μ and τ are rapidly changing into electrons through a process of particle decay. They are not object of the strong interaction, but of the weak interaction and the charged leptons of the electromagnetic interaction. The specific lepton numbers L_e, L_μ and L_τ are conserved quantities. It is +1 for a lepton and -1 for a anti lepton.

2.1.2 Quarks and Hadrons

Hadrons are made up of quarks and are also object on the strong interaction. The six flavours of the quark are the up- and the down-, the strange- and the charme- and the top- and the bottom-quark (also called truth and beauty). Every quark has its own colour charge (red, green or blue). The addition of the three colours is 0.
The most important baryons are the proton, made up of two up- and one down-quark and the neutron, which is made of one up- and two down-quarks. There is only a difference of one quark. Because of the electromagnetic repulsion, the strong force is important to bind the quarks.

2.1.3 Fundamental interactions and the gauge bosons

In fundamental physics there is a distinction of four fundamental forces between the particles. The gauge bosons are carrying the forces between the particles.

Electromagnetic interaction: Electromagnetism is the force that acts between electrically charged particles. The photon is its gauge boson. It is a particle with a mass of zero and a velocity of the light speed. Because of this, its helicity[2] is -1 or 1.

Weak nuclear interaction: The weak interaction is responsible for some nuclear phenomena like the beta decay. The carriers of the weak force are the massive gauge bosons called, the W and Z bosons (which is the object of our experiment). Because of their huge mass (80 and 91 GeV), the interaction has only a low range. Electromagnetism and weak force are understood as two aspects of an unified electroweak interaction. We will have a closer look later.

Strong nuclear interaction: The strong force holds only inside the atomic nucleus. It is about 100 times stronger than electromagnetism. The mediators are the eight different gluons, which also have colour loads.

Gravitional interaction: Gravitation is the weakest of the four interactions. There is no proof of a mediator yet. It is not necessary in particle physics.

[2]Helicity is the projection of the spin onto the direction of momentum.

2.2 Electroweak interaction

The Quantum electrodynamics describes the interaction between charged particles. The coupling strength for the electromagnetic force is the electron charge e. Because of the U(1) symmetry it has an gauge boson Y^0. For the weak force, the three gauge bosons W^+, W^- and W^0 (with mass M_W) are used because of U(2)-symmetry arguments. The coupling strength g is connected with the Fermi constant G_F:

$$G_F = \frac{\sqrt{2}g^2}{8M_W^2} = 1,663 \cdot 10^{-5} \frac{1}{\mathrm{GeV}^2} \tag{1}$$

For high energies the two coupling constants are nearly the same, so there is an unified description: The electroweak interaction. The spontaneous symmetry breaking causes the W^0 and the Y^0 bosons to coalesce together into the Z^0 boson and the photon. The two fields are orthogonal to each other:

$$\gamma = W^0 \sin\Theta_W + Y^0 \cos\Theta_W \tag{2}$$

$$Z^0 = W^0 \cos\Theta_W - Y^0 \sin\Theta_W \tag{3}$$

Θ_W is the weak mixing angle (Weinberg-angle). The masses of W and Z are related with it:

$$\cos\Theta_W = \frac{M_W}{M_Z} \tag{4}$$

2.3 The Z^0 boson

There are three bosons for the weak force: On the one hand there are the charged W^+ and W^-, which are important for the β-decay. A neutron decays into a proton, an electron and an electron anti neutrino. Because of the quark-model, there is a change of a d-quark into an u-quark, while radiating a W^--boson. This W-boson decays into a fermion and an anti fermion, in the case of β-decay it is the e^- and the $\overline{\nu_e}$. The neutral Z^0 boson is important for processes without changing charges. It represents the neutral current, for example the elastic scattering. Because of this, Z^0 boson interaction is even involving neutrinos.
Z bosons decay into all fermions and their antiparticles, which are possible because of their mass. The rest-mass of the neutrino is very big: 92 MeV. In an electron-positron-collider with a center of mass energy of m_Z it is possible to detect real Z^0 bosons.

2.4 Electron-positron-interaction - Z^0 resonance

In the experiment we use data of an e^+ e^- collision inside the OPAL-detector on the LEP-storage ring. The import interactions of this process are:

- The annihilation of the e^+ and e^- in two or three real photons. The whole energy of the two particles is changed into the photon energy.
- The annihilation into a virtual photon or Z^0 boson. This is a decay into a fermion anti fermion couple, like muons, tauons, quarks after a short time. The energy of the boson must be the sum of the rest energy of the two resulting particles. Free quarks can not exist, so one is only detecting a bundle of quarks, so called ”jets”.

- The elastic scattering, "Bhabha-scattering": It is the process: $e^+ + e^- \to e^+ + e^-$, so the particles do not change. Here several scattering processes are possible.
- Inelastic scattering: Two virtual photons interact and can create a hadron (Two-photon physics, also called gamma-gamma-physics)

2.4.1 Bhabha-scattering

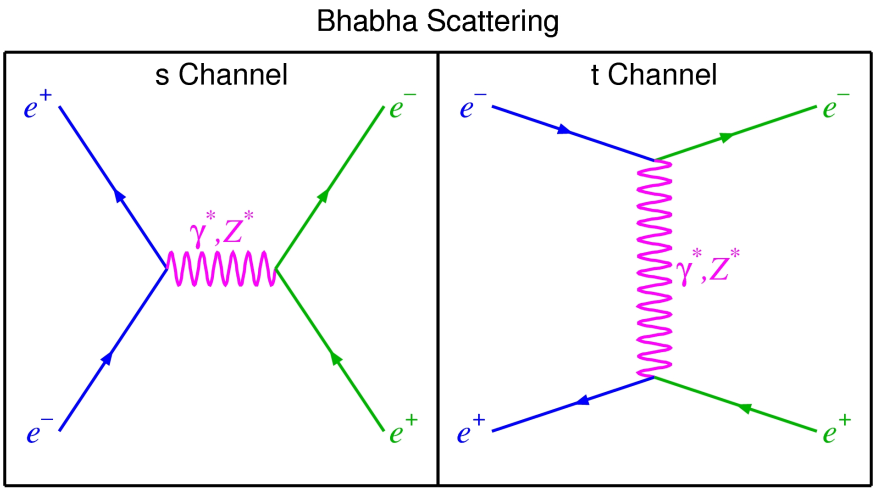

Abbildung 2: The two feynman graphs of the Bhabha-scattering

The Bhabha-scattering has two different Feynman diagrams contributing to this interaction: the annihilation process (s-channel) and the scattering process (t-channel). The ending results are identical, but the two processes have different scattering angels Θ (the angle between the incoming electron and the outcoming one). For big scattering angels the s-channel dominates, for small ones it is the t-channel. For the Z^0 resonance in our experiment, only the s-channel is relevant for decay width. It is:

$$\frac{\partial \sigma}{\partial \Omega} = s \cdot (1 + \cos(\Theta)^2) + t \cdot \frac{1}{(1 - \cos(\Theta))^2} \tag{5}$$

2.4.2 Annihilation in fermion couples

The decay into a fermion and antifermion couple is possible by a gamma-quantum or a Z^0 boson. Important for our experiment is the Z^0 boson. The cross section is dependent of the center of mass energy. In a diagram you will get a resonance graph with the resonance on the rest mass of the Z^0 boson. If we collide several electron-positron couples with different center of mass energies, we will get resonance graph by plotting the cross section σ over the center of mass energy $E = \sqrt{s}$. The plot is described by the Breit Wigner Distribution:

$$\sigma(s) = \frac{12\pi}{M_Z^2} \frac{s\Gamma_e \Gamma_x}{(s - M_Z^2)^2 + (s^2 \Gamma_Z^2 / M_Z^2)} \tag{6}$$

$M_Z = 91{,}182$ GeV is the rest mass of the Z^0. Γ_x is for example the leptonic or hadronic width, Γ_e the electronic width and Γ_Z the total width (see next chapter).
The electron and the positron are able to radiate a gamma-quantum. This reduces the energy and changes the cross section. If we calculate this radiation correction and set the energy as the rest energy of the Z^0 (resonance maximum), we will get:

$$\sigma^{res}(s) = \frac{12\pi}{M_Z^2}\frac{\Gamma_e\Gamma_x}{\Gamma_Z^2}(1+\delta) \tag{7}$$

2.4.3 Decay width

With the decay width Γ it is possible to calculate the lifetime of a particle. If the decay of a particle is to fast, so it is not possible to measure the lifetime. There is the possibility to measure the decay width of the cross section. This is the FWHM ("full width half maximum")at the Breit Wigner function. Because of the energy time uncertainty it is:

$$\Gamma = \hbar\lambda = \frac{\hbar}{\tau} \tag{8}$$

The Z^0-boson decays in several end products (decay channels), so there are several decay widths Γ_i. The more channels are possible, the bigger is the decay width and the shorter is the lifetime. The possible decay widths for the Z^0 are:

- $\Gamma_l = \Gamma_e = \Gamma_\mu = \Gamma_\tau$
- $\Gamma_\nu = \Gamma_{\nu(e)} = \Gamma_{\nu(\mu)} = \Gamma_{\nu(\tau)}$
- $\Gamma_u = \Gamma_c$[3]
- $\Gamma_d = \Gamma_s = \Gamma_b$

For the total decay width it is the sum of all of them:

$$\Gamma_Z = \sum_{i=1}^{n}\Gamma_i = 3\Gamma_l + 3\Gamma_\nu + \Gamma_{hadrones} \tag{9}$$

For energies near the mass of the Z^0 you get for the decay width for the several fermions Γ_f:

$$\Gamma_f = \frac{\sqrt{2}}{12\pi}\cdot N_c^f \cdot G_f \cdot M_Z^3 \cdot \left[(g_V^f)^2 + (g_A^f)^2\right] \tag{10}$$

N_c^f is the colourfactor (1 for leptons, 3 for quarks). g_V^f is caused of the vector-coupling and $g_A^f$of the axial-vector-coupling:

$$g_V^f = I_3^f - 2\cdot Q_f \cdot \sin(\Theta_w)^2 \tag{11}$$

$$g_A^f = I_3^f \tag{12}$$

I_3^f is the third component of the isospin, Q_f is the fermion charge. With this constants it is possible to compute the decay width of each fermion theoretically.

[3]The top-Quark is to heavy, so it is not an end product

Another important definition is the **branching ratio** BR_i.

$$BR_i = \frac{\Gamma_i}{\Gamma_Z} \tag{13}$$

The bigger the BR, the bigger is the possibility for this several decay. Because of the definition the sum of all branching ratios is 100 %.

2.4.4 Radiation corrections

The measured dates have to be corrected because of inefficiencies of the detector or because of criteria of the selection. Furthermore, the Born approximation is not adequate: Radiation corrections have to be considered. One distinguishes real and virtual corrections. The real radiation processes are combined of braking deceleration (bremsstrahlung) at the beginning and the end and also the interference of theses two effects. Virtual radiation processes are a result of the same end states like in the Born approximation.
It is the case, that the exact computation of the radiation corrections is very difficult, in the experiment we will use given values.

2.5 Forward-Backward-Asymmetry

Above and below the resonance maximum of the rest mass of the Z^0 there are asymmetries of the cross section. This is an result of the interference of the electromagnetic vector-interaction and the weak axial-vector-interaction. It is defined as:

$$A_{FB} = \frac{\int_0^1 \frac{d\Theta}{d\cos\Theta} d\cos\Theta - \int_{-1}^0 \frac{d\Theta}{d\cos\Theta} d\cos\Theta}{\int_0^1 \frac{d\Theta}{d\cos\Theta} d\cos\Theta + \int_{-1}^0 \frac{d\Theta}{d\cos\Theta} d\cos\Theta} \tag{14}$$

Because of the weak interaction there are asymmetries in the scattering angles in the forward and backward scattering. In the Born approximation the partial cross section is given with:

$$\frac{d\sigma_f}{d\Omega} = \frac{\alpha^2 \cdot N_c^f}{4s} \Big(F_1(s)(1 + \cos\Theta^2) + 2F_2(s)\cos\Theta \Big) \tag{15}$$

The asymmetry is defined as the fraction

$$A_{FB} = \frac{3}{4}\frac{F_2}{F_1} \tag{16}$$

The weinberg angle Θ_W is related with the asymmetry at the resonance maximum of the Z-particle:

$$A_{FB}^{peak} = 3 \cdot \Big(1 - 4\sin^2\Theta_W\Big)^2 \tag{17}$$

A measurement of the asymmetry at resonance maximum comes to a measurement of $\sin^2\Theta_W$.

2.6 Particle Detectors

2.6.1 Interaction of particles with matter

The interaction of a particle in matter is dependent to the sort of the particle, its energy and to the sensor material. Charged and neutral particles reacts different. Important is the electromagnetic force and the strong force. This is important for the verification of the particles. If a charged particle interacts wit the coulomb field of a nucleus of an atom, it decelerates and sends radiation: the **bremsstrahlung**. It is important for electrons.
Every charged particle looses energy in matter because of interaction with the electrons of atoms in the material. The interaction **excites or ionizes the atoms** and leads to an energy loss of the particle. The Bethe formula describes the energy loss per distance $\frac{dE}{dx}$. It is important for protons, alpha particles and atomic ions , but not for electrons. The relativistic version of the formula is:

$$-\frac{dE}{dx} = \frac{4\pi n z^2}{m_e c^2 \beta^2} \cdot \left(\frac{e^2}{4\pi\varepsilon_0}\right)^2 \cdot \left[\ln\left(\frac{2m_e c^2\beta^2}{I \cdot (1-\beta^2)}\right) - \beta^2\right] \tag{18}$$

There is no continuous energy loss of photons, they disappear (**photoelectric effect**, **pair production**) or are scattered (**compton-effect**). If the velocity of a particle is bigger than the lightspeed in the material, you can detect the **Cherenkov-radiation** as a coniform wave front.

2.6.2 Proportional counters

A charged particle is able to ionize surrounding gaseous atoms. The resulting ions and electrons are accelerated by the electric field around the wire to this anodes and ionize for their part new atoms. In proportional counters one is using lots of wires as anodes, it is possible to count particles and determine their energy. The time difference is proportional to the distance of the wire and the particle, in drift chambers it is possible to measure the location in this way.

2.6.3 Calorimeters

A calorimeter is an experimental apparatus that measures the energy of particles. Most particles enter and initiate a particle shower. The particles' energy is deposited in the calorimeter, collected, and measured. There are differences between an electromagnetic shower and a hadron shower. Electrons loose primarily energy by bremsstrahlung, this produces photons, which decay in an electron positron couple. These loose again their energy and produce new photons until the energy of the particles is too low. The lateral size of the shower is limited.
A hadron calorimeter measures particles that interact via the strong nuclear force. About half of the incident hadron energy is passed on to additional secondaries (decays in neutrinos or muons are detectable). A characteristic of the hadron shower is, that it takes longer to develop than the electromagnetic shower.

2.6.4 Luminosity

An important value in particle physics is the luminosity. It is the ratio of the number of events detected in a certain time (event rate) to the interaction cross-section:

$$\frac{dN}{dt} = \sigma \cdot L \qquad n = \sigma \int Ldt \tag{19}$$

$\int Ldt$ is called the integrated luminosity L_{int}. It is a useful value to characterize the performance of a particle accelerator.

2.7 The OPAL detector

The OPAL-detector (**O**mni-**P**urpose-**A**paratus for **L**EP) was one of the four big detectors sited on the LEP-storage-ring at CERN. It measured the interactions between electrons and positrons, which collided at the centre of the detector. The detector was dismantled in 2000 to make way for LHC equipment. We use data from this detector.

2.7.1 Technical construction of the detector

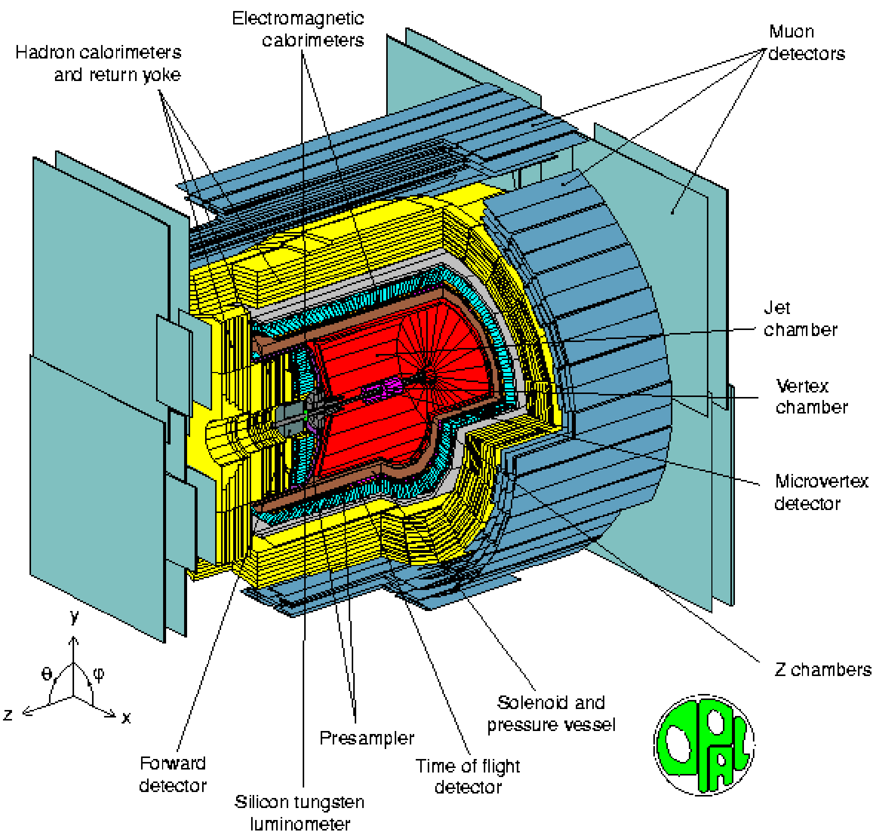

Abbildung 3: A cut-away view of the OPAL detector.

The incoming electrons and positrons approach the centre of the detector from opposite directions, along the beam pipe, an evacuated straight metal cylinder. The several components are described now from inside to the outside:

- The central tracking system consists of a silicon **microvertex detector**, a **vertex detector**, a **jet chamber**, and 24 **z-chambers**. They work by observing the ionization of atoms. The two vertex detectors locate decay vertices of short-lived particles, the bigger jet chamber with 24 sectors deduces the trajectory and the momentum because of the curvature of a magnetic field. The z-chambers enable precise measurements of the z-coordinates of the tracks.
- The **time of flight detector** measures the flying time and triggers the detector.
- The **electromagnetic calorimeters** are made of lead-glass blocks. They cover nearly all angles from the beam direction. They measure the energies and positions of electromagnetic showers (electrons, positrons and photons).
- The **hadron calorimeter** lies outside the electromagnetic calorimeter. It is of iron and catches particles which has penetrated through the electromagnetic detector.
- The gas-filled **muon detector** is constructed around the system. Muons penetrate and leave a single clean track.
- The **forward detector** measures the luminosity by detecting Bhabha-scattering.

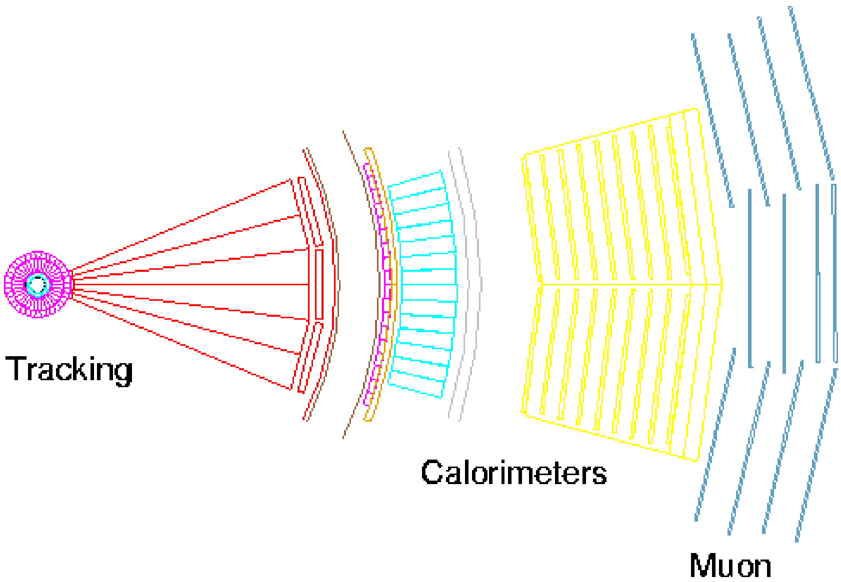

Abbildung 4: A schematic slice of the OPAL detector. You can see the tracking system with the vertex detector (magenta) and the jet chamber (red), the electromagnetic (cyan) and hadron (yellow) calorimeter and also the muon chamber (blue).

2.7.2 Identification of particles

To associate the detector dates with the several particles, it is important to have a short view about the particle tracks. With the program GROPE we are able to detect the tracks and get several dates of the experiments. The detector measures the following important components, which are given as the following variables:

PCHARGED The sum of the momentums of the charged particles in the tracking system

E_ECAL The total energy measured in the electromagnetic calorimeter

H_ECAL The total energy measured in the hadron calorimeter

NCHARGED The number of tracks in the tracking system

The particles, we detect, have individual signatures:

- The shower of **electrons** and **positrons** is complete in the em calorimeter. Because of the charge, there is a track in the tracking system.
- The shower of **hadrons** is wider and start later, so they are mainly in the hadron calorimeter.[4] Charged hadrons (like π^+ or K^+) leave a track in the tracking system, while uncharged (like π^0) are only detectable by the shower. Hadrons are always detected when the Z^0-bosons decays in quarks.
- **Muons** are minimal ionising, so there is no shower. They are detectable by a signal in the muon detector.
- **τ-leptons** can decay in lots of different end states, mostly in pions and neutrinos.
- **Photons** have a similar signature like an electron, but no track in the tracking system. The convert in an electron positron couple is also possible, this becomes apparant, that there is a v-shaped track.
- The **neutral pion** π^0 decays directly in two photons, which causes two showers in the electromagnetic calorimeter.[5]
- Neutrinos aren not detectable in the system, because they only interact with the weak interaction. This is called ”invisible decay modes”.

While limiting the several variables with **cuts** in energy areas for example, it is possible to seperate the decay processes of the Z^0-bosons decay. Here it is important to get a good **acceptance**: The number of desirable events must be big, but the number of undesirable events should be minimized. The acceptance is the quotient of the detected events and the total events. The number of total events is ascertainable with simulations, like the **Monte Carlo method**.

[4]This is only the case if the energy is big enough. Under 2 GeV the shower of hadrons and electrons is not distinguishable.

[5]Normally the distance between the two showers is small, so you can not distinguish them from a single photon.

3 Analysis

3.1 Theoretical values

First of all we will have a closer look on some theoretical values and at the process of the s-channel and t-channel.

3.1.1 The decay width

With formula (10) on page 8 we are able to compute the decay width of the different leptons. We use the following values:

$\sin(\Theta_W)^2 = 0,2312$
$N_c^f = 1$ (for leptons), $N_c^f = 3$ (for quarks)
$G_f = 1,663 \cdot 10^{-5} \frac{1}{\text{GeV}^2}$
$M_Z = 91,182$ GeV
The isospin I_3^f is for charged leptons and the down, strange and bottom quark -1/2, for neutrinos and the up and charm quarks it is 1/2.
The charge Q_f for e, μ and τ is -1, for neutrinos it is 0, for the down, strange and bottom quark it is -1/3 and for the up and charme quark it is 2/3.

We compute the values and compare them with the values given in the instructions on page 27:

	computed value [MeV]	value in instruction [MeV]	difference [%]
$\Gamma_e = \Gamma_\mu = \Gamma_\tau$	83,4	83,8	0,5
$\Gamma_{\nu(e)} = \Gamma_{\nu(\mu)} = \Gamma_{\nu(\tau)}$	165,8	167,6	1,0
$\Gamma_u = \Gamma_c$	285,0	299,0	4,5
$\Gamma_d = \Gamma_s = \Gamma_b$	368,0	378,0	2,7

There are differences between the computed values and the literature values of 0,5 % and 4,5 %. This may cause because of the negligence of the mass of the particle, radiation sections and approximations in the formula. There is no error given on the values, so it is difficult to compare them.
We also compute

- the charged lepton width $\Gamma_{lep,\ total} = \Gamma_e + \Gamma_\mu + \Gamma_\tau$,
- the neutral lepton width (invisible width) $\Gamma_{\nu,\ total} = \Gamma_\nu + \Gamma_{\nu(e)} + \Gamma_{\nu(\mu)} + \Gamma_{\nu(\tau)}$,
- the hadron width $\Gamma_{q,\ total} = \Gamma_u + \Gamma_c + \Gamma_d + \Gamma_s + \Gamma_b$,
- and the total decay width of Z^0 $\Gamma_Z = \Gamma_{lep,\ total} + \Gamma_{\nu,\ total} + \Gamma_{q,\ total}$

on the one hand with the computed values, on the other hand with the given values.

	computed value [MeV]	value in instruction [MeV]	difference [%]
$\Gamma_{lep,\ total}$	250,2	251,4	0,5
$\Gamma_{\nu,\ total}$	497,5	502,8	1,0
$\Gamma_{q,\ total}$	1674,0	1732,0	3,3
Γ_Z	2422,0	2486,0	2,6

The theoretical value for the total decay width from the Particle Data Book[6] is $\Gamma_Z = 2490 \pm 7$ MeV. The theoretical value of the instruction is inside one standard derivation, the computed value is not very good. Because of this, we use in the following the more exact theoretical values.

3.1.2 Branching ratio and number of fermion couples

With the decay widths we are able to compute the branching ratio $BR_i = \frac{\Gamma_i}{\Gamma_Z}$ (formula (13)). We get for the charged leptons, the quarks and the neutrinos:

$BR_{charged\ leptons} = 10,1\%$
$BR_{neutrinos} = 20,2\%$
$BR_{quarks} = 69,7\%$

The most probable decay is the decay in quarks with nearly 70%.
If there is the possibility of another, fourth fermion couple (charged lepton and neutrino), we would have a bigger decay width for the Z^0. It would increase by $\Gamma_l + \Gamma_\nu = 251,4$ MeV. The percentage variation of Γ_Z would be 10,1 %.
If there is only another possible neutrino and anti neutrino couple, the percentage variation would be 6,7 %.

3.1.3 Partial cross section

With equation 7 on page 8 we are able to compute the partial cross section at the resonance maximum. We ignore the radiation correction term $(1+\delta)$. We use the literature values of the instruction for Γ_x. We get for the cross sections in nano barn:

$$
\begin{aligned}
&\sigma_e = \sigma_\mu = \sigma_\tau = && 0,515 \cdot 10^{-5}\ 1/\mathrm{GeV}^2 = && 2,00\ \mathrm{nb}\\
&\sigma_{\nu(e)} = \sigma_{\nu(\mu)} = \sigma_{\nu(\tau)} = && 1,030 \cdot 10^{-5}\ 1/\mathrm{GeV}^2 = && 2,00\ \mathrm{nb}\\
&\sigma_u = \sigma_c = && 1,838 \cdot 10^{-5}\ 1/\mathrm{GeV}^2 = && 7,15\ \mathrm{nb}\\
&\sigma_d = \sigma_s = \sigma_b = && 2,324 \cdot 10^{-5}\ 1/\mathrm{GeV}^2 = && 9,04\ \mathrm{nb}
\end{aligned}
$$

Here we use that 1 nb $= 2,57 * 10^{-6}\ \frac{1}{\mathrm{GeV}^2}$ as the conversion factor. The addition of the cross section of all quarks is 41,43 nb.

[6] We use the extract in the instruction on page 67 ff.

3.1.4 Angular distribution

Because of the Bhabha-scattering we have to distinguish the s-channel-process and the t-channel-process. For this distinction it is important, that the s-channel is proportional to $(1 + \cos(\Theta)^2)$ and the t-channel is proportional to $\frac{1}{1-\cos(\Theta)^2}$. Θ is the scattering angle. We plot this two distributions and the sum of them in a diagram.

The two distributions are overlapping, for a positive $\cos\Theta$ the t-channel is dominating, for a

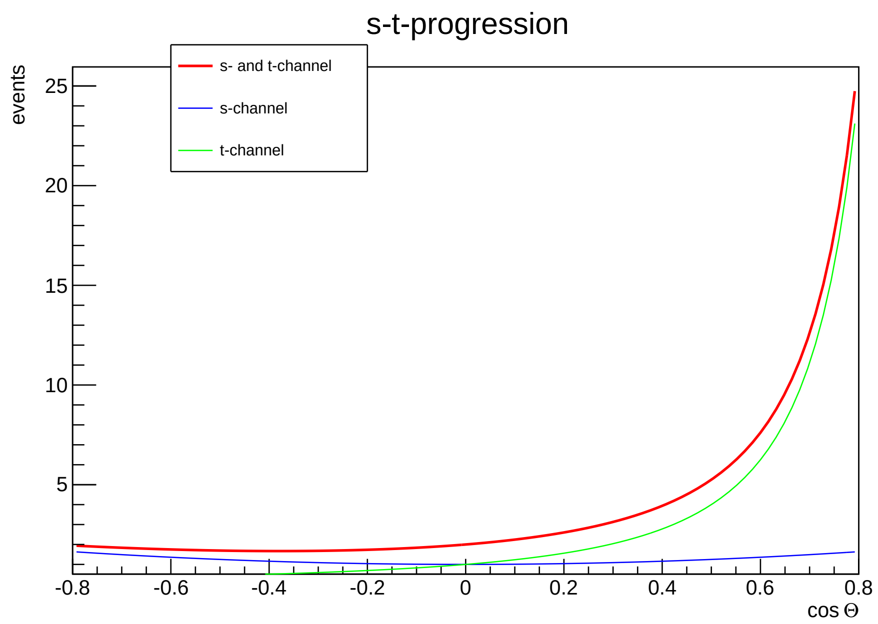

Abbildung 5: The theoretical plot of the s-channel (blue) and the t-channel (green) and the addition of them (red). For big positive angles, the t-channel is dominating, for negative ones it is the s-channel.

With the function (5) on page 7 we are able to get the values S and T to get the percentage share of the two channels.

There is a problem for the t-channel: The function is diverging for $\cos\Theta \rightarrow 1$. If we would use an area for these theoretic functions between -1 and 1, we would have an infinite number of t-channel electrons. There is also a boarder smaller than one, where the real detector is not able to measure the angle any more. Unfortunately we do not know this exactly. Later we have to estimate it.

3.2 The simulated data on GROPE

We had four different data sets on the PC, one for each decay channel. The data is simulated with the Monte Carlo method, because of this, the total number of the several decays in a channel is known. With the program GROPE[7] it is possible to have a visual 2D-projection on the tracks and the showers. We analysed the four channels and customized a table for ten exemplary decays. The parameters *Run* and *Event* are only for the identification. The last column *e_beam* is the energy of one beam in the accelerator, so twice this value is the center of mass energy $\sqrt{s}$ of the Z^0 boson. For the run 2566, $\sqrt{s}$ is 91,22 GeV. For the run 2568 $\sqrt{s}$ is 93,36 GeV.
We picture for each channel a typical depiction and have a short explanation for the decay and its energies. The tracks and showers are pictured in several colours:

blue - track in the tracking system

green - Time-Of-Flight-Detector

yellow - shower in electromagnetic calorimeter

purple - shower in hadron calorimeter

red - track in the muon chamber

Ctrk(N) and *Ctrk(Sump)* in the GROPE-figure are the values *ncharged* and *pcharged*.

3.2.1 e^+ e^- events

In figure 6 you can see an e^+ e^- decay. Two opposite tracks are identifiable. There is only an electromagnetic shower for each particle, no hadron one. Nearly the whole energy is detected in the electromagnetic calorimeter. The values in the tabular confirm this: The values for the electromagnetic energys are between 80 and 100 GeV, this is the most important criterion for the cut. The hadron energy is usually 0. Sometimes there are some other effects, like electronic noise or beam-gas events, so there are fluctuations in the values (for example the 3 at *ncharged*)

Run	Event	Ncharged	Pcharged [GeV]	E_ecal [GeV]	E_hcal [GeV]	E_beam [GeV]
2566	163733	2	50,9	82,6	0,0	45,61
2566	165523	2	91,9	90,0	0,0	45,61
2566	165548	3	82,5	92,3	0,0	45,61
2566	165576	2	80,9	86,8	0,0	45,61
2566	166436	2	38,1	89,5	0,0	45,61
2566	167987	2	83,8	87,5	0,0	45,61
2566	168389	2	87,4	93,2	0,0	45,61
2566	170045	2	69,3	90,7	0,0	45,61
2566	170379	2	86,1	89,4	0,5	45,61
2566	197594	2	90,3	90,6	0,0	45,61

[7] Graphical Reconstruction of OPAL Events

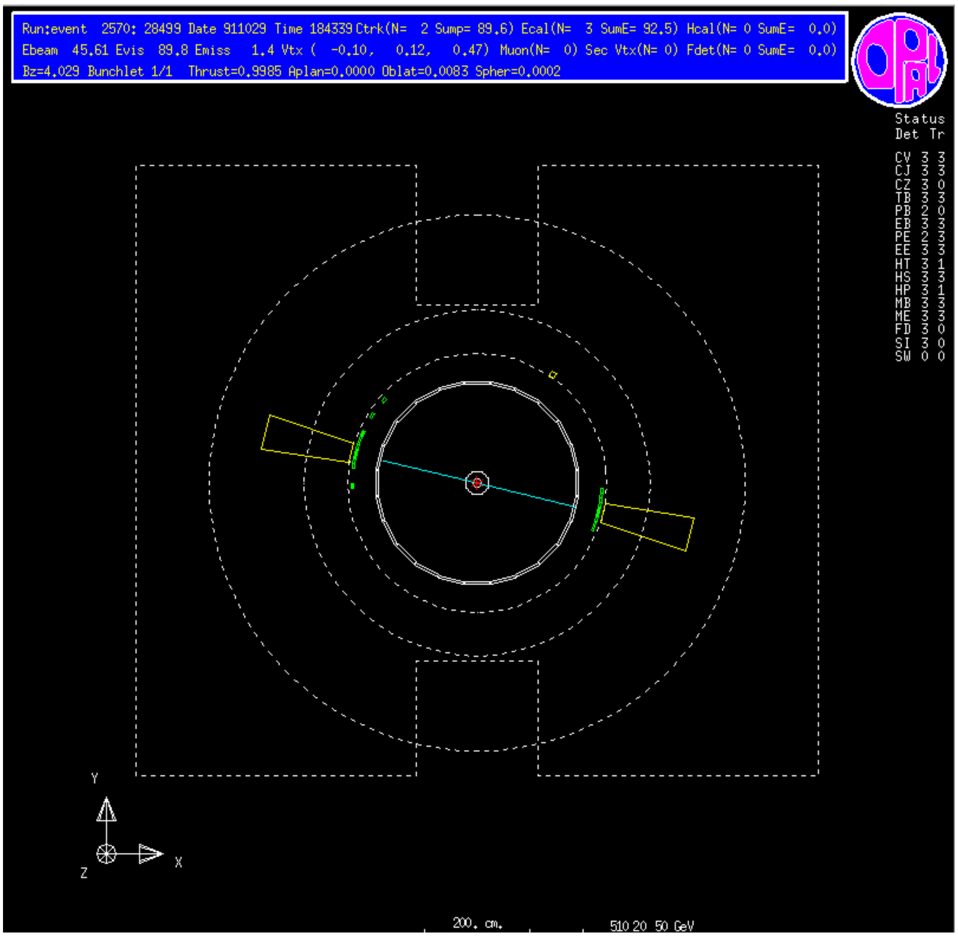

Abbildung 6: The decay of the Z^0 boson in an electron positron couple, detected with the Opal detector. The figure is made with GROPE.

3.2.2 μ^+ μ^- events

At the muon decay you can easily identify the two tracks in figure 7. The red arrows left and right detect the muon in the muon chamber. Like the electron there are usually only two tracks, because the muon is minimal ionising there is left energy in the electromagnetic and the hadron calorimeter. In the tabular you see, that the electromagnetic energies are smaller than 4 GeV (which makes it possible to distinguish them from the electron decays). The energy of the tracks *pcharged* is between 80 and 100.

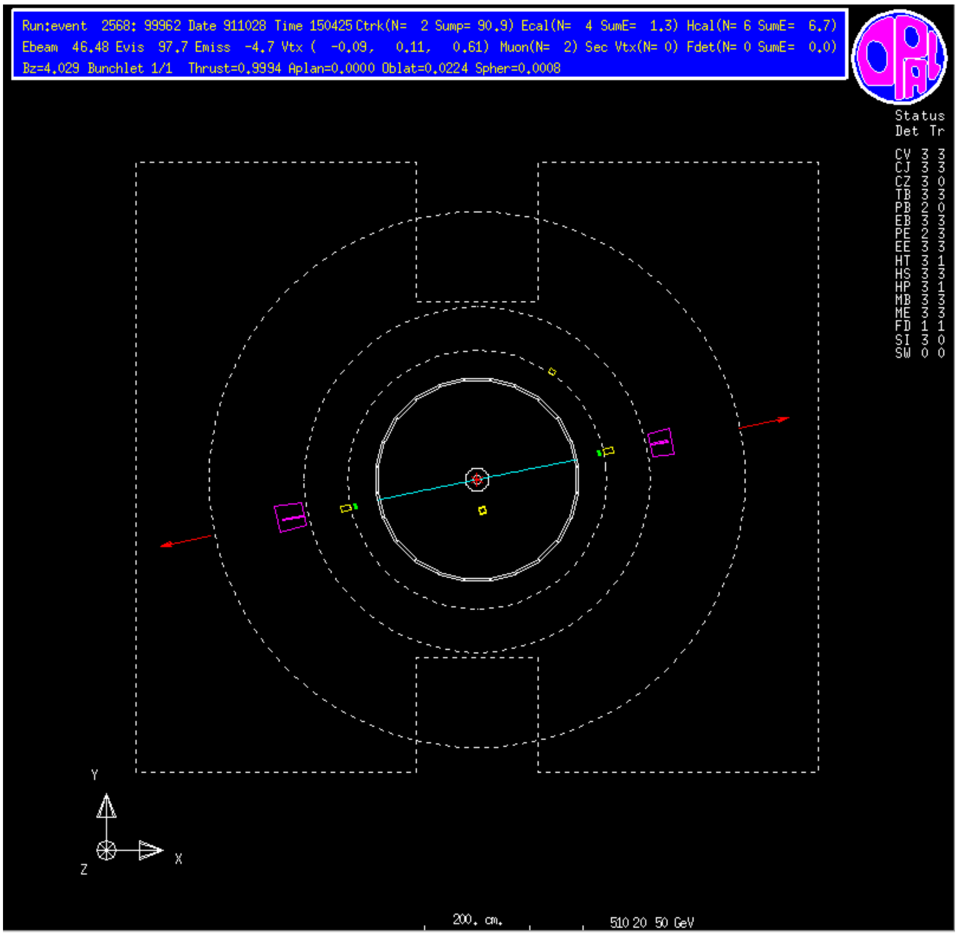

Abbildung 7: The decay of the Z^0 boson in a mion couple, detected with the Opal detector. The figure is made with GROPE.

Run	Event	Ncharged	Pcharged [GeV]	E_ecal [GeV]	E_hcal [GeV]	E_beam [GeV]
2568	80617	2	90,1	1,6	7,0	46,48
2568	84297	2	93,0	1,6	8,7	46,48
2568	85398	2	96,8	2,0	0,0	46,48
2568	87693	2	89,1	2,3	8,5	46,48
2568	89929	2	90,5	1,5	7,2	46,48
2568	91048	2	91,8	1,8	4,3	46,48
2568	92681	2	86,3	3,7	3,3	46,48
2568	93199	2	99,2	1,3	2,9	46,48
2568	95202	2	88,2	1,6	3,0	46,48
2568	99962	2	90,9	1,3	6,7	46,48

3.2.3 $\tau^+ \tau^-$ events

The decay in a tauon couple is more difficult then the first two decays. In figure 8 you can see the two tracks (blue) of the tauon-particles, but there are also a few with more tracks. The tauon-particle has a short life time and it decays in several final states, like pions or neutrinos. The neutrinos are not detectable, so the energy in the calorimeters is less than the center of mass energy. Sometimes there where one or two detected muons. In the tabular you can see the electromagnetic energy with a big range between circa 2 and 50 GeV and the hadron energy between circa 4 and 21 GeV. The energy of the tracks is usually smaller than 70, this distinguishes the events from the muon events.

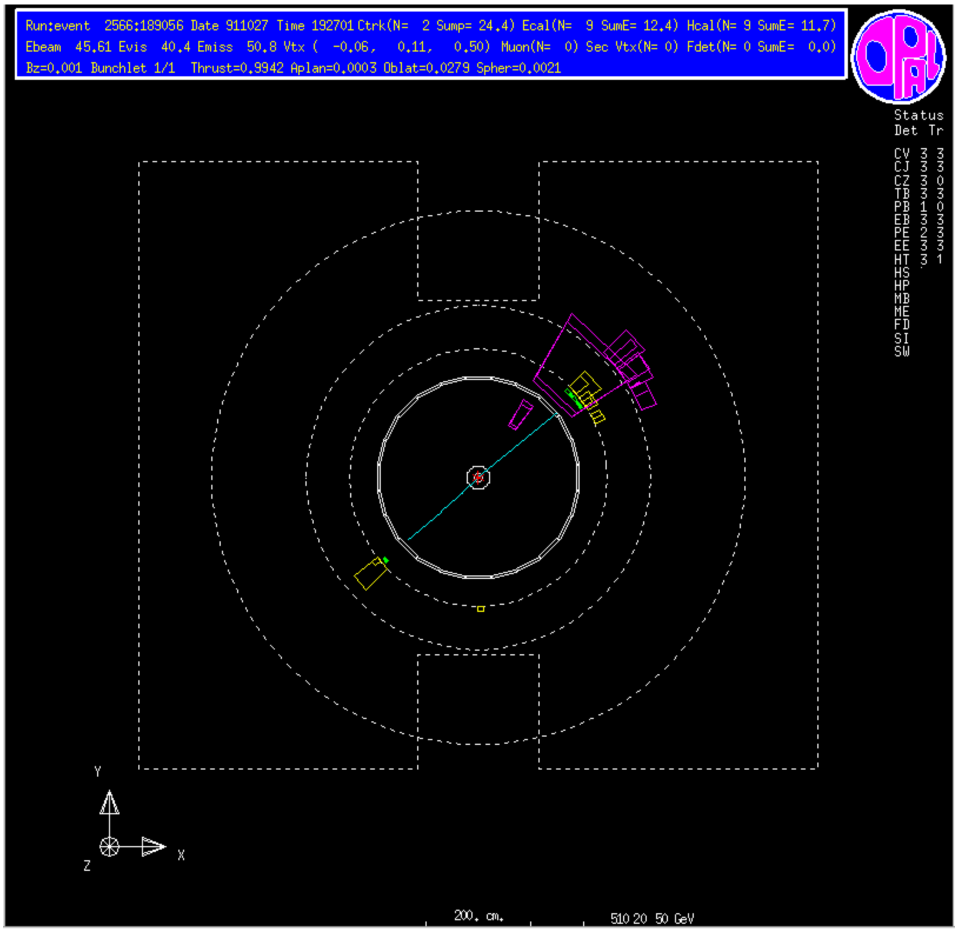

Abbildung 8: The decay of the Z^0 boson in a tau couple, detected with the Opal detector. The figure is made with GROPE.

Run	Event	Ncharged	Pcharged [GeV]	E_ecal [GeV]	E_hcal [GeV]	E_beam [GeV]
2566	170371	5	74,0	51,1	10,2	45,61
2566	170508	2	46,5	17,3	8,2	45,61
2566	179750	2	30,8	1,6	6,3	45,61
2566	184010	2	29,5	10,2	4,1	45,61
2566	184435	2	33,1	1,5	10,6	45,61
2566	189056	2	24,4	12,4	11,7	45,61
2566	208314	4	36,0	16,1	5,7	45,61
2566	212745	2	41,3	11,1	20,0	45,61
2570	29664	2	49,7	5,2	20,3	45,61
2570	30348	2	33,4	23,6	6,9	45,61

3.2.4 q $\bar{q}$ events

If the Z^0 decays in a quark anti quark couple, the energy of the strong field is big enough to produce other quark anti quark couples. The quarks get together to a variety of hadrons. This causes lots of tracks, which are shown in figure 9. In the tabular the nunber of tracks is between 8 and 46, this is the most important difference to the lepton events. The hadrons left showers in the electromagnetic calorimeter as well as in the hadron calorimeter.

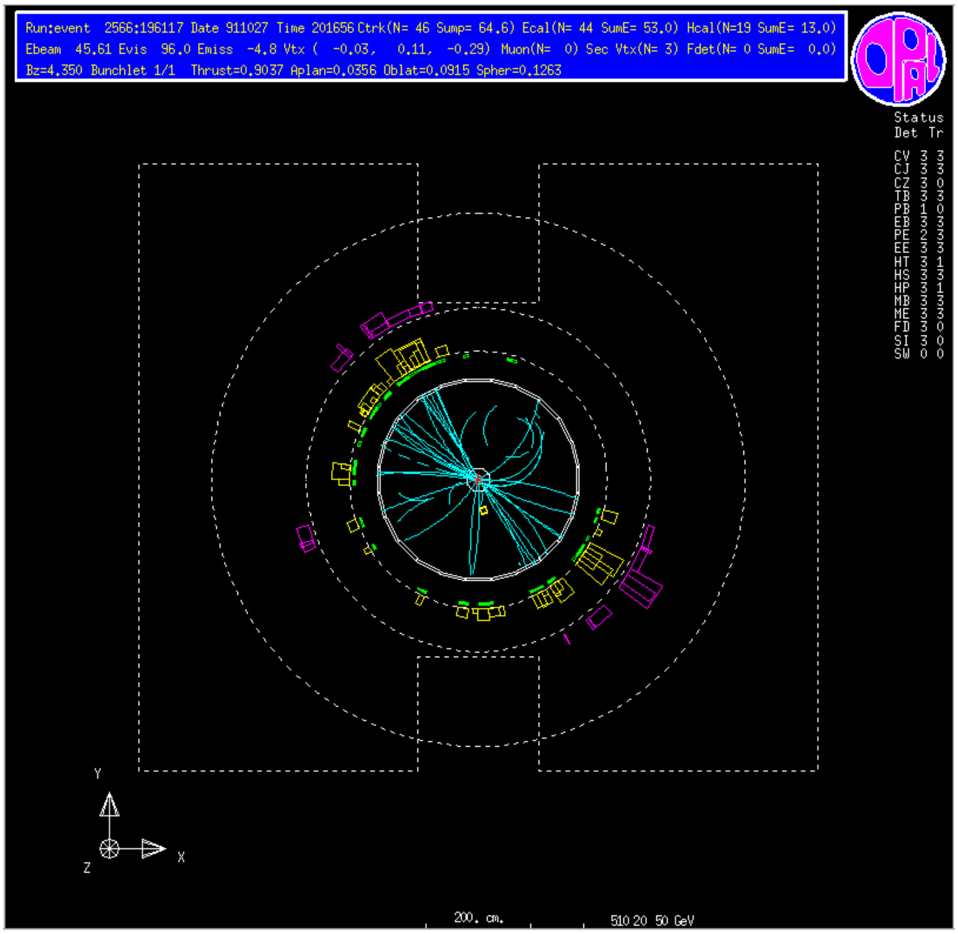

Abbildung 9: The decay of the Z^0 boson in a quark anti quark couple, detected with the Opal detector. The figure is made with GROPE.

Run	Event	Ncharged	Pcharged [GeV]	E_ecal [GeV]	E_hcal [GeV]	E_beam [GeV]
2566	164184	15	37,7	37,0	14,1	45,61
2566	195995	17	39,2	66,8	9,9	45,61
2566	196117	46	64,6	53,0	13,0	45,61
2566	196548	8	33,3	67,5	13,3	45,61
2568	78191	36	45,3	53,2	7,7	46,48
2568	78425	41	59,5	53,2	13,8	46,48
2568	78553	9	21,9	65,2	8,8	46,48
2568	78787	16	55,9	50,4	24,3	46,48
2568	79038	30	38,1	68,3	13,8	46,48
2568	79043	22	34,4	75,5	6,2	46,48

3.3 Analysing the simulated data - defining the cuts

3.3.1 The cuts in Ncharged, Pcharged, E_ecal and E_hcal

To define the cuts, we compare the values *ncharged*, *pcharged*, *e_ecal* and *e_hcal*. For this, we plot these values for every channel in a histogram. Figured are the number of events in percent for each value of the energy respectively the number. You can see the graph for the energy of the electromagnetic calorimeter in figure 10, for the hadron calorimeter in figure 11, for the track energy in figure 13 and for the number of tracks in figure 13.

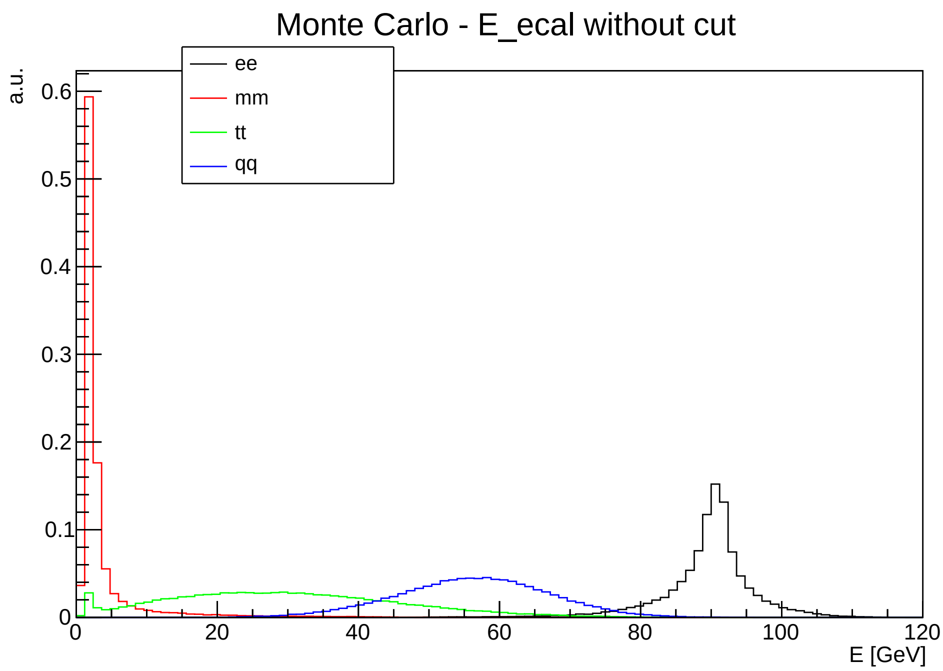

Abbildung 10: The plots of the **energy of the electromagnetic calorimeter** for the four different decay channels.

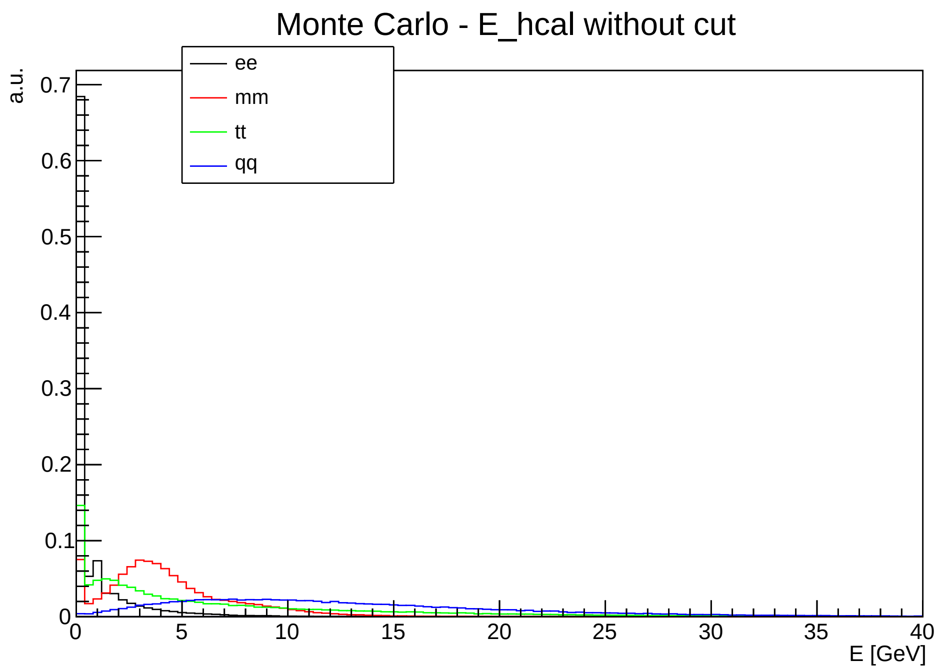

Abbildung 11: The plots of the **energy of the hadron calorimeter** for the four different decay channels.

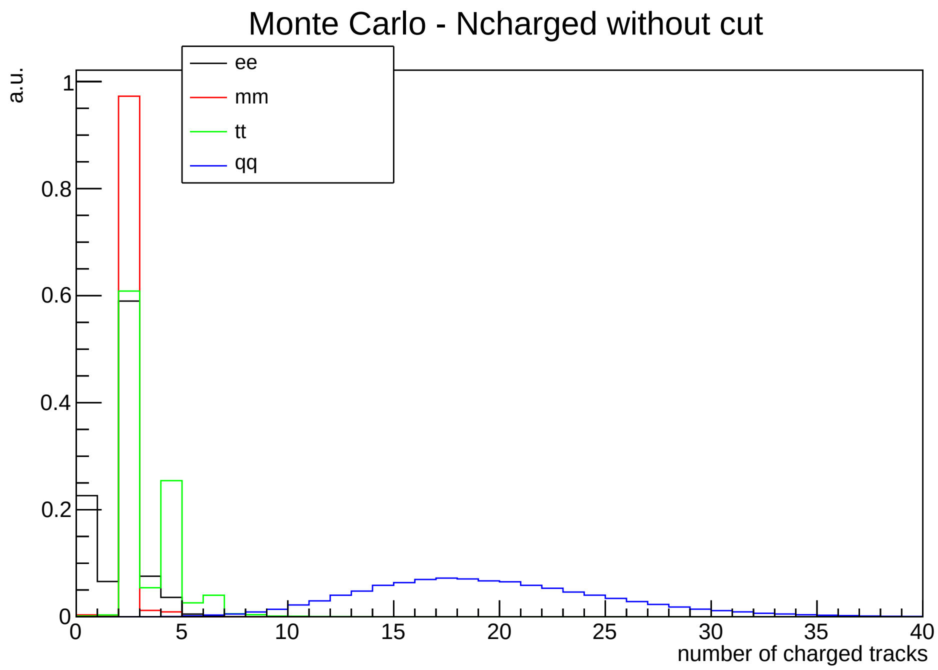

Abbildung 12: The plots of the **number of tracks** for the four different decay channels.

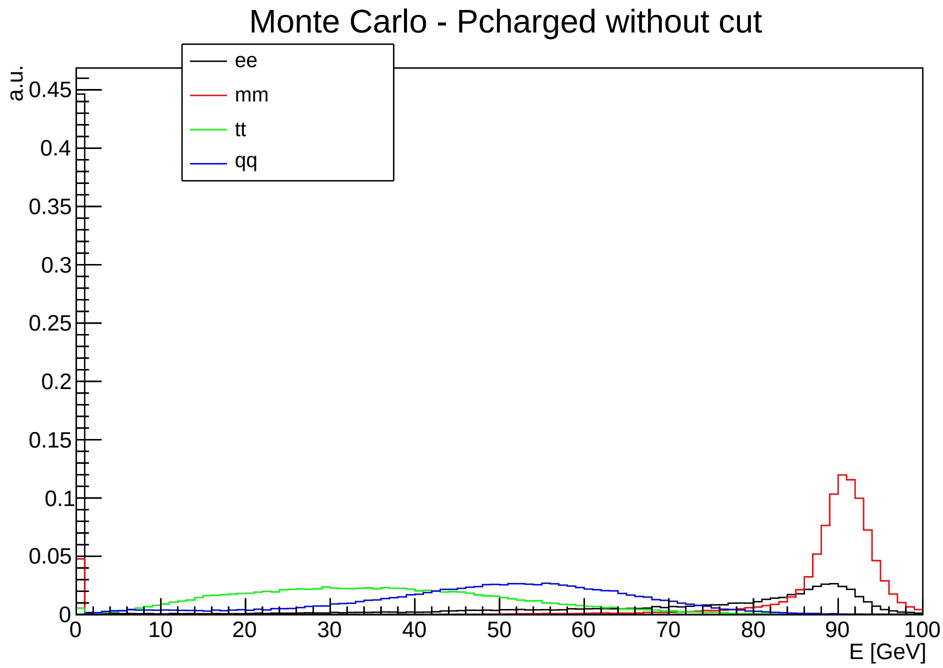

Abbildung 13: The plots of the **energy of all tracks** for the four different decay channels.

For the electron events, there is a huge amount of decays with $ncharged = 0$. This is, because the efficiency of the tracking system is not 100 %. Furthermore, there can be a huge bremsstrahlung, so the energy of the electrons is to low for a track. Because of this, sometimes there is only one or zero tracks detected.

With this plots we are able to define the cuts. These are important to separate the four channels in the real data. We try to find boarders for the for values for each channel with to aims: The efficiency should be big. This is the number of events after the cuts in comparison with the total number. Second we need a big purity. Less other events should be in the data package after the cut. We decided upon the following cuts:

channel	Ncharged	Pcharged [GeV]	E_ecal [GeV]	E_hcal [GeV]
$e^+ e^-$	0<n<4		80<e	
$\mu^+ \mu^-$	0<n<5	70<p	e<30	
$\tau^+ \tau^-$	1<n<6	p<70	e<75	
q $\bar{q}$	7<n			

The important cut for separating the quark-events is the number of tracks, which is bigger than seven. For the other events it is smaller than seven.

To separate the electrons we have a look on the electromagnetic energy, which is bigger than the energy of the three other events. We decided for a cut at 80 GeV. With this cut we reduce the efficiency, because we cut off lots of electron events, but also we reduce the number of foreign taus in the cut. The big purity in the electron cut is important for the s-t-channel separation. The overlapping with the quark plot does not matter, we separated them with the cut in the number of events at 4.

With the energy of the tracks, it is possible to separate the muons from the tauons and quarks

by cutting at an energy of 70. Now it is important to cut off them from the electrons. This is possible with the energy in the electromagnetic calorimeter, it should be smaller than 30 GeV. The cuts for the tauons are a little more difficult: We separated them from the quarks with the cut in the number of tracks. The cutting off the electrons is realized again with the energy in the electromagnetic calorimeter, it should be smaller than 75 GeV. With the cut of 70 GeV in the energy of the tracks we are able to separate them from the muons.
Also possible in the real data are two-photon-events. These we also want to cut off. Photons have no charge, therefore the number of tracks is zero. We cut them off while saying the number of events must be bigger than 0.

3.3.2 The cut in $\cos\Theta$ for the electrons

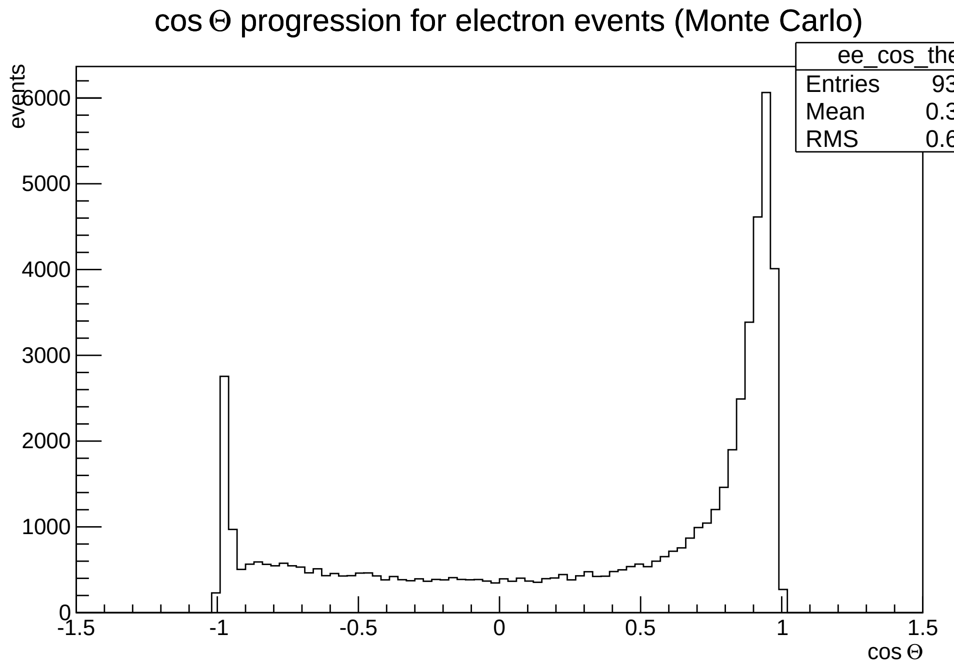

Abbildung 14: The plots of the number of electrons over cosinus Θ.

For the electrons we have to add a further cut on $\cos\Theta$. For the s-t-channel separation we only can use electrons where a definite scattering angle was measured. First, there is a huge number of electrons with an angle of ca. 1000. Here the detector wasn't able to detect an angle.[8] We can't use these electrons. Additionally a cut between -1 and 1 does not make sense, because at these boarders the detector is not able to measure the angle. Unfortunately we do not know this boarders of the detector resolution, so we try to estimate them with an exactness of 0,05. If you have a look in the graph 14, it seems to be the best to define the cut at -0,95 and 0,95, at this points the number of events is decreasing. Later we will discuss the error on this estimation. Second we want to have a short look on the big peaks on the left and the right side of graph

[8] We can't say for sure, why lots of events have this impossible angle or if there is a correlation to a certain event. We are able to cut it off, but because of the unknowingness of the particles inside, we will later have a problem at the s-t-channel separation. We will have a discussion later.

14. These are results of the events of electrons, where the number of tracks is zero. You can see the plot for the electron events with $ncharged = 0$ over $\cos\Theta$ in figure 15. Because of the missing tracks, an angle measurement is not possible. They will sort at $\cos\Theta \approx -1$ respectively $\cos\Theta \approx +1$. With our cut for $ncharged > 0$ we are able to sort out this two peaks.

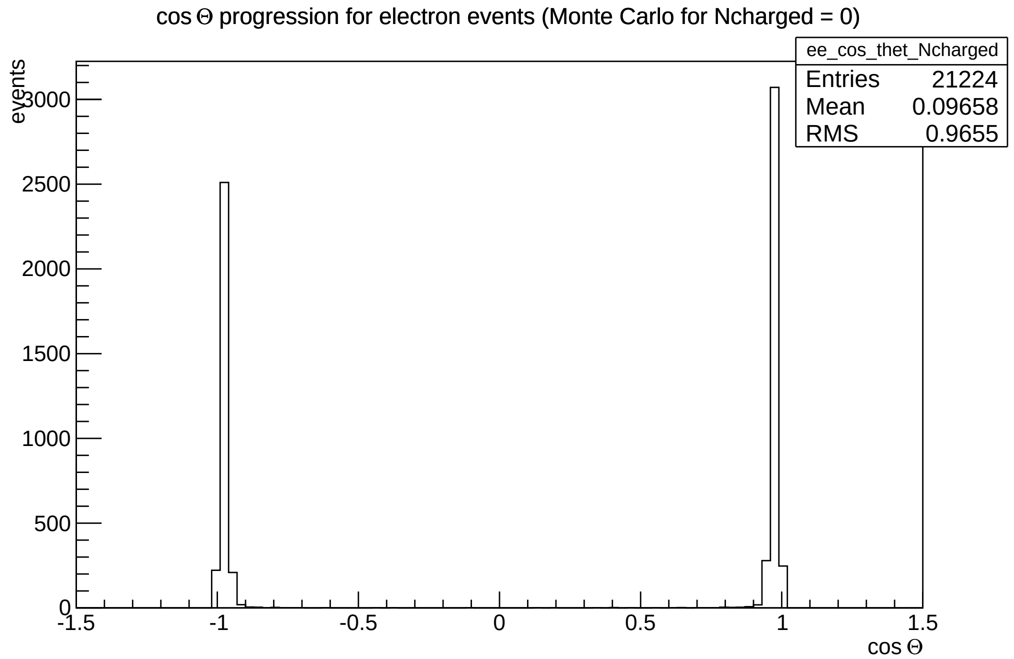

Abbildung 15: The plots of the number of electrons with $ncharged = 0$ over cosinus Θ. These particles are the reason for the two peaks at the left and right side.

3.4 The efficiency matrix

In the following we distinguish the numbers from the **simulated** data $N_{\mathbf{s}}$ (one package with electron-decay, one with muon-decay, one with tauon-decay and one with quark-decay) and the numbers from the **real** data $N_{\mathbf{r}}$ (the real experiment with all possible decays)
In the simulated data we know the numbers of total events for electrons, muons, tauons and quarks $N^e_{s,total}$, $N^\mu_{s,total}$, $N^\tau_{s,total}$ and $N^q_{s,total}$. After realizing the cuts, we have a number of electrons, muons, tauons and quarks in each cut. For example in the electron cut, we get: $N^e_{s,e-cut}$, $N^\mu_{s,e-cut}$, $N^\tau_{s,e-cut}$ and $N^q_{s,e-cut}$. For the other three cuts we also get these four variables. Now we are able to compute four different efficiencies for every cut. It is the quotient of the number of the particles in the cut through the total number of the particle in the simulated data:

$$\varepsilon^j_i = \frac{N^j_{s,i-cut}}{N^j_{s,total}} \tag{20}$$

j is the concerning particle species, i is the concerning cut. With this we are able to define the following four times four efficiency matrix:

	electrons	muons	tauons	quarks
e-cut	ε^{e}_{e-cut}	$\varepsilon^{\mu}_{e-cut}$	$\varepsilon^{\tau}_{e-cut}$	ε^{q}_{e-cut}
μ-cut	$\varepsilon^{e}_{\mu-cut}$	$\varepsilon^{\mu}_{\mu-cut}$	$\varepsilon^{\tau}_{\mu-cut}$	$\varepsilon^{q}_{\mu-cut}$
τ-cut	$\varepsilon^{e}_{\tau-cut}$	$\varepsilon^{\mu}_{\tau-cut}$	$\varepsilon^{\tau}_{\tau-cut}$	$\varepsilon^{q}_{\tau-cut}$
q-cut	ε^{e}_{q-cut}	$\varepsilon^{\mu}_{q-cut}$	$\varepsilon^{\tau}_{q-cut}$	ε^{q}_{q-cut}

We get the following values for the number N before and after the four different cuts:

	electrons	muons	tauons	quarks
$N^{j}_{s,total}$	93802	94381	79214	98563
$N^{j}_{s,e-cut}$	44420	1	59	1
$N^{j}_{s,\mu-cut}$	0	86880	874	0
$N^{j}_{s,\tau-cut}$	1166	6759	72580	200
$N^{j}_{s,q-cut}$	6	0	544	97550

With these values we get the following efficiency matrix M (all values in percent):

$$\mathbf{M}_{\varepsilon} = \begin{pmatrix} 47,35 & 0,0011 & 0,074 & 0,0010 \\ 0 & 92,06 & 1,10 & 0 \\ 1,24 & 7,16 & 91,62 & 0,203 \\ 0,006 & 0 & 0,69 & 98,97 \end{pmatrix}$$

The diagonal elements are the efficiency of the particle species in their own cut, the best value is 100 %. The other elements are the particle values in an other cut, they should be as less as possible. The best efficiency matrix would be a matrix with the value 100 in the diagonal elements , but this is indeed not possible. Our matrix is a very good compromise for the both criteria.

The error on the efficiency matrix is computed in the following way: The total numbers $N^{j}_{s,total}$ are errorless. The numbers $N^{j}_{s,i-cut}$ are binomial distributed, with the total number $n = N^{e}_{s,total}$ and the possibility $p = \varepsilon^{j}_{i}$. The variance of a binomial distributed event is $\sigma^2 = \sqrt{n \cdot p \cdot (1-p)}$. Because of this, the error on $\varepsilon^{j}_{i} = \dfrac{N^{j}_{s,i-cut}}{N^{j}_{s,total}}$ is:

$$s_{\varepsilon^{j}_{i}} = \frac{\sqrt{N^{j}_{s,total} \cdot \varepsilon^{j}_{i} \cdot (1-\varepsilon^{j}_{i})}}{N^{j}_{s,total}} = \sqrt{\frac{\varepsilon^{j}_{i} \cdot (1-\varepsilon^{j}_{i})}{N^{j}_{s,total}}} \tag{21}$$

With this equation we get for every matrix element an error. This results to the following error matrix:

$$\mathbf{M}_{\mathbf{s}_{\varepsilon}} = \begin{pmatrix} 0,16 & 0,0011 & 0,009 & 0,0010 \\ 0 & 0,09 & 0,03 & 0 \\ 0,04 & 0,08 & 0,09 & 0,015 \\ 0,003 & 0 & 0,03 & 0,03 \end{pmatrix}$$

The biggest efficiency is possible for the quarks, here we are able to have a good cut in *ncharged*. The efficiency for the electrons is very small. This relies on the cut in $\cos\Theta$, we have to cut lots of events with the saved angle of 1000. The tauons make the most problems in the separation. In the cut with the tauons there a lots of foreign particles, furthermore there are always tau-particles in the other cuts. Unfortunately a better separation is not possible. With this matrix we have a good compromise between efficiency and purity.

3.5 Cutting the real data - separating the channels

For the real data from the OPAL detector we use data package 1, with a total number of decay of

$$N_{r,total} = 175883$$

The data package contains seven different energies of the beams, we separate them with cuts to get seven data packages with the center of mass energy E1 to E7.

E1	E2	E3	E4	E5	E6	E7
88,47 GeV	89,46 GeV	90,22 GeV	91,22 GeV	91,97 GeV	92,96 GeV	93,71 GeV

For each energy we have to separate the four different decay channels. We use our defined cuts to get four different data packages with the numbers $N'_{r,e-cut}$[9], $N_{r,\mu-cut}$, $N_{r,\tau-cut}$ and $N_{r,q-cut}$. Because of the different incoming energies of the data packages, we will have different detected energies in the tracking system and in the electromagnetic and hadron calorimeter. We have to correct our cut boarders and scale them on the energies E1 till E7. We scale for this the cut condition on the beam energy of the simulated data $\frac{\text{cut}_n}{2 \cdot e_{lep,j}}$. (The center of mass energy is twice the energy of one beam e_{lep}.) After that we multiple it with the center of mass energy of the real data E1, E2, ..., E7. This corrects our cut condition respectively the incoming energy. Because of the binomial distribution, the error on each number is

$$s_{N_{r,i-cut}} = \sqrt{N_{r,i-cut}}$$

We get the following numbers of particles in each cut for the different energies:

	E1	E2	E3	E4
$N'_{r,e-cut}$	680 ± 30	660 ± 30	590 ± 20	5030 ± 70
$N_{r,\mu-cut}$	139 ± 12	257 ± 16	349 ± 19	4040 ± 60
$N_{r,\tau-cut}$	221 ± 15	262 ± 16	303 ± 17	4100 ± 60
$N_{r,q-cut}$	3530 ± 60	5320 ± 70	7540 ± 90	92700 ± 300

	E5	E6	E7
$N'_{r,e-cut}$	790 ± 30	380 ± 20	540 ± 20
$N_{r,\mu-cut}$	710 ± 30	281 ± 17	338 ± 18
$N_{r,\tau-cut}$	670 ± 30	333 ± 18	345 ± 19
$N_{r,q-cut}$	15290 ± 120	6640 ± 80	7420 ± 90

[9]We will correct this number later to N, so for now it is called N'

3.6 s-t-channel-separation

In the data of electron-events are both the annihilation process (s-channel) and the scattering process (t-channel). We have to correct the number of electron events $N'_{r,e-cut}$ to separate it from the t-channel.
For this we make a histogram with the number of the events (y) and the scattering-angle $\cos\Theta$ (x). We plot the number of events over the scattering angle $\cos\Theta = x$ between our defined cuts -0,95 and 0,95. In the histogram we see the addition of the events of the s-channel and the t-channel. (Compare with the theoretical plot in chapter 3.1.4 at page 16) We fit the function (5) on the histogram

$$y = s \cdot (1 + x^2) + t \cdot \frac{1}{(1-x)^2}$$

to get the parameters s and t. As mentioned, at $\cos\Theta = 1$ and $\cos\Theta = -1$ a defined angle measurement is not possible. We already tried to cut them off with the cuts at 0,95. But there are still some small effects of the decreasing, which have a huge effect on the fit. (For example in figure 16 is one entry on the right side which is definite to small. This is no problem for the integration or the cutting boarders, but for the fit function.) To make the fit more stable, we decided to reduce the fit boarders to 0,90 and -0,90. This is the best compromise for a big area for the fit and less disturbing effects. Our boarders also are the best for a χ^2 which is as small as possible with them. This method is not exact, so we later add a mistake because of these boarders.
With the parameters s and t we are able to compute the theoretical numbers of events in the two channels. For this we have to integrate over the two functions $s \cdot (1+\cos\Theta^2)$ and $t \cdot \frac{1}{(1-\cos\Theta)^2}$. The bounds of integration are of course the cutting boarders of -0,95 and 0,95. We only have a look on the particles between these boarders. Now we compute the amount of events of the s-channel in comparison to the total number of events (the addition of s- and t-channel) to get the correction factor *cor* (individual for the seven energies):

$$cor = \frac{\int\limits_{-0,95}^{0,95} s \cdot (1+\cos\Theta^2)\, d\cos\Theta}{\int\limits_{-0,95}^{0,95} s \cdot (1+\cos\Theta^2)\, d\cos\Theta + \int\limits_{-0,95}^{0,95} t \cdot \frac{1}{(1-\cos\Theta)^2}\, d\cos\Theta} = \frac{s \cdot int_s}{s \cdot int_s + t \cdot int_t} \tag{22}$$

The Gaussian error on the correction factor is

$$s_{cor} = \frac{int_t/int_s}{(s + t \cdot int_t/int_s)^2} \cdot \sqrt{t^2 \cdot s_s^2 + \cdot s^2 \cdot s_t^2} \tag{23}$$

with s_s as the error on the factor s and s_t as the error on t.
In figure 16 you can see the plot for the energy E4. Included is the fit function (red) and also the two parts of it for the s-channel (blue) and the t-channel (green). For negative angles the s-channel is dominating, for big positive angles it is the t-channel. The area under the two plots is proportional to the number of events.

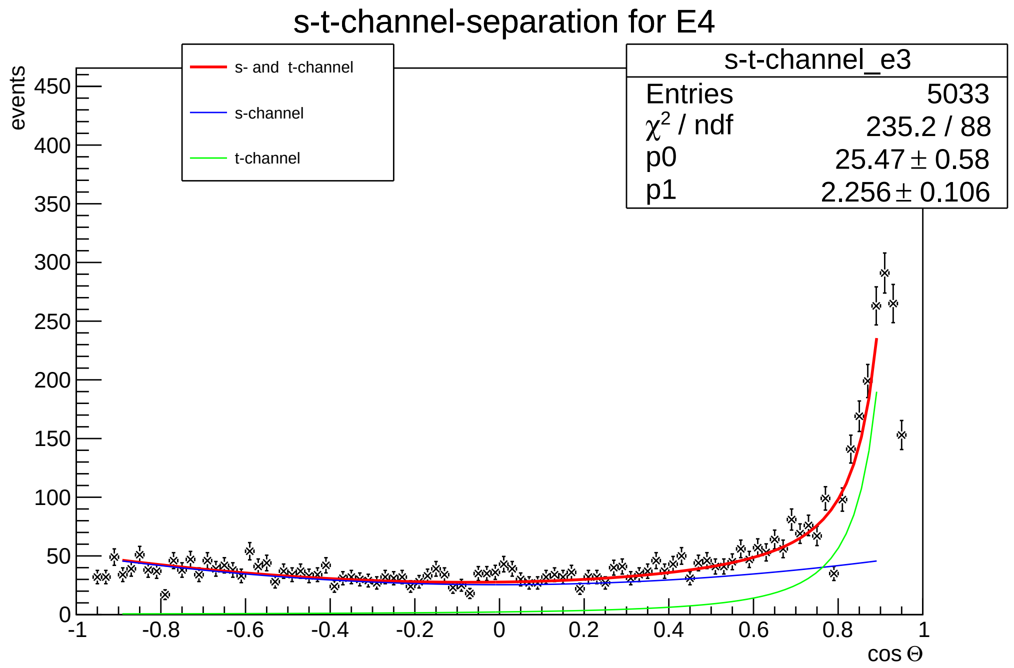

Abbildung 16: The fit for the separation of the s- and t-channel in the electron data for E4.

The plots for the other six energies an be found in appendix A "Plots and fits for s-t-channel separation" on page 45.
We get the following values of the fits:

	E1	E2	E3	E4
χ^2	0,944	0,922	0,922	2,673
Parameter s	0,71 ± 0,14	1,33 ± 0,16	1,75 ± 0,16	25,5 ± 0,6
Parameter t	0,77 ± 0,05	0,58 ± 0,04	0,42 ± 0,44	2,26 ± 0,11
cor	0,104 ± 0,019	0,23 ± 0,02	0,35 ± 0,03	0,589 ± 0,013

	E5	E6	E7
χ^2	1,316	0,765	0,972
Parameter s	2,9 ± 0,2	1,13 ± 0,14	1,50 ± 0,15
Parameter t	0,40 ± 0,04	0,25 ± 0,03	0,31 ± 0,04
cor	0,48 ± 0,03	0,36 ± 0,04	0,38 ± 0,04

The χ^2 is around 1, only for E4 it is around 2,7. All in all, it validate our fit-function. Because of the not clear defined boarders of the fit and the cuts, we try to estimate an additional error while changing these boarders a little. Because of this we add an additional error of 10% with the Gaussian propagation of uncertainty. Finally we get the following correction factors:

energy E_k	cor_{E_k}	s_{cor}
E1	0,10	0,02
E2	0,23	0,03
E3	0,35	0,05
E4	0,59	0,06
E5	0,48	0,06
E6	0,36	0,06
E7	0,38	0,05

We multiple this correction factors to the numbers of particles in the electron cut, to only get the amount of s scattering.

$$N_{r,e-cut,E_k} = cor_{E_k} \cdot N'_{r,e-cut,E_k} \tag{24}$$

The new error on the data with the e-cut is computed with Gauß:

$$s_{N_{r,e-cut}} = \sqrt{(N'_{r,e-cut} \cdot s_{cor})^2 + (\sqrt{N'_{r,e-cut}} \cdot cor)^2} \tag{25}$$

The error on the correction factor is dominating, because of this, there is a bigger error on the first row with the e-cuts.
We get the following corrected values for the number of electrons in the cuts in real data:

	E1	E2	E3	E4
$N_{r,e-cut}$	70 ± 15	150 ± 20	200 ± 30	3000 ± 300
$N_{r,\mu-cut}$	139 ± 12	257 ± 16	349 ± 19	4040 ± 60
$N_{r,\tau-cut}$	221 ± 15	262 ± 16	303 ± 17	4100 ± 60
$N_{r,q-cut}$	3530 ± 60	5320 ± 70	7540 ± 90	92700 ± 300

	E5	E6	E7
$N_{r,e-cut}$	380 ± 50	140 ± 20	200 ± 30
$N_{r,\mu-cut}$	710 ± 30	281 ± 17	338 ± 18
$N_{r,\tau-cut}$	670 ± 30	333 ± 18	345 ± 19
$N_{r,q-cut}$	15290 ± 120	6640 ± 80	7420 ± 90

Unfortunately there are some problems in this s-t-channel separation, we would like to discuss in the following: First of all, there is the inexact definition of the boarders. We tried to take this into consideration with the bigger error. Now we are able to determine the ratio of s and t-events inside this area. Later we will use this ratio for the whole data, also for the data in the cosinus = 1000 peak we already cut. Due to the fact, that we do not know why we have this peak, we can not say for sure, that the ratio of s and t-events for this data is the same. We have no other chance to suppose this, but if it is not the same, it will falsify the results. Furthermore there are still some foreign particles in the e-cut data, which do not have a t-channel event. But they should be less, we tried to make the purity for the electron cut as big as possible.

All in all we only know efficiency factor ε for the total data, but do not know if it would change for the s-data we separated. The cut in $\cos\Theta$ cut more t-events than s-events, furthermore we do not know enough about the cosinus = 1000 events, these effects can change the efficiency factor in an unpredictable way. We will later see at our results, that this way of s-t-channel separation is not exact. Unfortunately there is no better possibility because of the unknown factors.

3.7 The total number of the particles

Now we have the numbers of relevant particles in each cut: $N_{r,e-cut}$, $N_{r,\mu-cut}$, $N_{r,\tau-cut}$ and $N_{r,q-cut}$ for the seven different center of mass energies. To compute the real numbers of each particle $N^j_{r,total}$, we use the following system of equations:

$$\varepsilon^e_{e-cut}\cdot N^e_{r,total}+\varepsilon^\mu_{e-cut}\cdot N^\mu_{r,total}+\varepsilon^\tau_{e-cut}\cdot N^\tau_{r,total}+\varepsilon^q_{e-cut}\cdot N^q_{r,total}=N_{r,e-cut} \tag{26}$$
$$\varepsilon^e_{\mu-cut}\cdot N^e_{r,total}+\varepsilon^\mu_{\mu-cut}\cdot N^\mu_{r,total}+\varepsilon^\tau_{\mu-cut}\cdot N^\tau_{r,total}+\varepsilon^q_{\mu-cut}\cdot N^q_{r,total}=N_{r,\mu-cut} \tag{27}$$
$$\varepsilon^e_{\tau-cut}\cdot N^e_{r,total}+\varepsilon^\mu_{\tau-cut}\cdot N^\mu_{r,total}+\varepsilon^\tau_{\tau-cut}\cdot N^\tau_{r,total}+\varepsilon^q_{\tau-cut}\cdot N^q_{r,total}=N_{r,\tau-cut} \tag{28}$$
$$\varepsilon^e_{q-cut}\cdot N^e_{r,total}+\varepsilon^\mu_{q-cut}\cdot N^\mu_{r,total}+\varepsilon^\tau_{q-cut}\cdot N^\tau_{r,total}+\varepsilon^q_{q-cut}\cdot N^q_{r,total}=N_{r,q-cut} \tag{29}$$

We can easier write this, if we define a 4-vector with the total numbers for each particle $N^j_{r,total}$, a 4-vector for each numbers in the cut $N_{r,i-cut}$ and use the efficiency matrix:

$$\mathbf{M}_\varepsilon\cdot\begin{pmatrix}N^e_{r,total}\\N^\mu_{r,total}\\N^\tau_{r,total}\\N^q_{r,total}\end{pmatrix}=\begin{pmatrix}N_{r,e-cut}\\N_{r,\mu-cut}\\N_{r,\tau-cut}\\N_{r,q-cut}\end{pmatrix} \tag{30}$$

To get the total numbers of each particle sort, it is

$$\begin{pmatrix}N^e_{r,total}\\N^\mu_{r,total}\\N^\tau_{r,total}\\N^q_{r,total}\end{pmatrix}={\mathbf{M}_\varepsilon}^{-1}\cdot\begin{pmatrix}N_{r,e-cut}\\N_{r,\mu-cut}\\N_{r,\tau-cut}\\N_{r,q-cut}\end{pmatrix} \tag{31}$$

Because of this, we have to invert the efficiency matrix:

$${\mathbf{M}_{\varepsilon,\mathbf{ji}}}^{-1}=\begin{pmatrix}2,112 & 0,000109 & -0,00172 & -0,00002\\0,00034 & 1,087 & -0,0131 & 0,000026\\-0,0287 & -0,0850 & 1,093 & -0,00224\\0,00006 & 0,00059 & -0,0076 & 1,0100\end{pmatrix}$$

Contrary to the normal matrix, in the inverted matrix the columns are called i and the rows j. The error on the elements of the inverted matrix is really difficult to compute with the Gaussian propagation of uncertainty. We use a trick, which makes it easier. We add and subtract the error on the efficiency matrix, to get a matrix with the biggest and smallest efficiency inside one standard derivation. We invert this two matrices, to get one inverted matrix with bigger and one with smaller values. From each value we take the difference, the half of it is the error on the inverted matrix

$$\mathbf{M_{s_\varepsilon}}^{-1} = \begin{pmatrix} 0,007 & 0,000007 & 0,00020 & 0,00002 \\ 0,000018 & 0,0010 & 0,0004 & 0,000003 \\ 0,0007 & 0,0008 & 0,0010 & 0,00016 \\ 0,00004 & 0,00003 & 0,0003 & 0,0003 \end{pmatrix}$$

Now we are able to compute the total numbers of each particle sort with equation (31). The error on each value is computed with Gauß:

$$s_{N^j_{r,total}} = \sqrt{\sum_{i=1}^{N} \left((s_{M^{-1}_{\varepsilon,ji}} \cdot N_{r,i-cut})^2 + (s_{N_{r,i-cut}} \cdot M^{-1}_{\varepsilon,ji})^2 \right)} \tag{32}$$

We get the following numbers for the electrons, muons, taus and quarks for the seven energies:

	E1	E2	E3	E4
$N^e_{r,total}$	150 ± 30	310 ± 50	430 ± 60	6200 ± 600
$N^\mu_{r,total}$	148 ± 13	276 ± 17	380 ± 20	4340 ± 70
$N^\tau_{r,total}$	220 ± 16	248 ± 18	278 ± 19	3850 ± 70
$N^q_{r,total}$	3560 ± 60	5380 ± 70	7620 ± 90	93700 ± 300

	E5	E6	E7
$N^e_{r,total}$	800 ± 100	290 ± 50	430 ± 60
$N^\mu_{r,total}$	760 ± 30	301 ± 18	360 ± 20
$N^\tau_{r,total}$	630 ± 30	320 ± 20	330 ± 20
$N^q_{r,total}$	15450 ± 130	6710 ± 80	7500 ± 90

Because of the s-t-channel-separation the percentage error on the electrons is much bigger. Due to the big number of quark events, this percentage error is only around 1%.

3.8 Computing the cross section

With equation (19) on page 11 we are able to compute the cross section for each particle and energy. The different integrated luminosities are given with the data set:

Energy	Luminosity L	s_L
E1	675,859	5,712
E2	543,627	4,831
E3	419,776	3,975
E4	3122,204	22,318
E5	639,838	5,577
E6	479,240	4,482
E7	766,838	6,498

The cross section is computed with:

$$\sigma = \frac{N_{r,total}}{L} \tag{33}$$

with an error of:

$$s_\sigma = \frac{N_{r,total}}{L} \cdot \sqrt{\frac{s_{N_{r,total}}}{N_{r,total}}^2 + \frac{s_L}{L}^2} \tag{34}$$

We add or subtract the following beam corrections (the hadron correction for quarks, the lepton correction for electrons, muons, taus), to get the real cross section:

Energy	hadron correction	lepton correction
E1	+2,0	+0,09
E2	+4,3	+0,20
E3	+7,7	+0,36
E4	+10,8	+0,52
E5	+4,7	+0,22
E6	-0,2	-0,01
E7	-1,6	-0,08

There are no errors on the correction values, the error does not change.
Finally we get the following cross sections (all values in nanobarn (nb)):

	E1	E2	E3	E4
σ_e	0,31 ± 0,05	0,78 ± 0,09	1,38 ± 0,14	2,5 ± 0,2
σ_μ	0,31 ± 0,02	0,71 ± 0,03	1,26 ± 0,05	1,91 ± 0,02
σ_τ	0,41 ± 0,02	0,66 ± 0,03	1,02 ± 0,05	1,75 ± 0,02
σ_q	7,27 ± 0,10	14,19 ± 0,16	25,9 ± 0,3	40,8 ± 0,2

	E5	E6	E7
σ_e	1,47 ± 0,16	0,60 ± 0,10	0,48 ± 0,08
σ_μ	1,41 ± 0,05	0,62 ± 0,04	0,39 ± 0,03
σ_τ	1,21 ± 0,05	0,66 ± 0,04	0,34 ± 0,03
σ_q	28,8 ± 0,3	13,8 ± 0,2	8,16 ± 0,14

3.9 Breit Wigner Fit

With the final cross section it is possible to plot these values over the center of mass energy $E = s^2$. We do this separate for each of the four particle sorts and use s_σ for the y-errorbars. The error on the center of mass energy is very small. The theoretical course is the Breit Wigner distribution. We have three unknown parameters: M_Z, the total width Γ_Z and the specific width of the channel Γ_x. First we do it for the electrons. Here it is:

$$\sigma(s) = \frac{12\pi}{M_Z^2} \frac{s\Gamma_e^2}{(s - M_Z^2)^2 + (s^2\Gamma_Z^2/M_Z^2)} \cdot 2,57 \cdot 10^{-6} \tag{35}$$

With the determined electronic width, we can fit the other three channels:

$$\sigma(s) = \frac{12\pi}{M_Z^2} \frac{s\Gamma_e \cdot \Gamma_{\mu/\tau/q}}{(s - M_Z^2)^2 + (s^2\Gamma_Z^2/M_Z^2)} \cdot 2,57 \cdot 10^{-6} \tag{36}$$

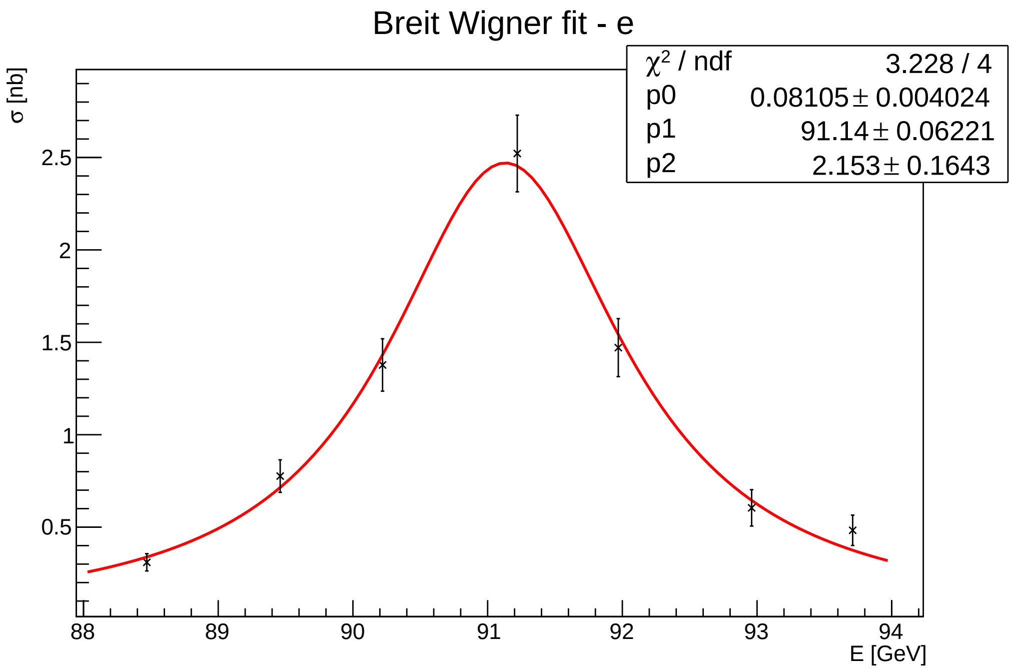

Abbildung 17: The Breit Wigner fit for the cross section of electrons.

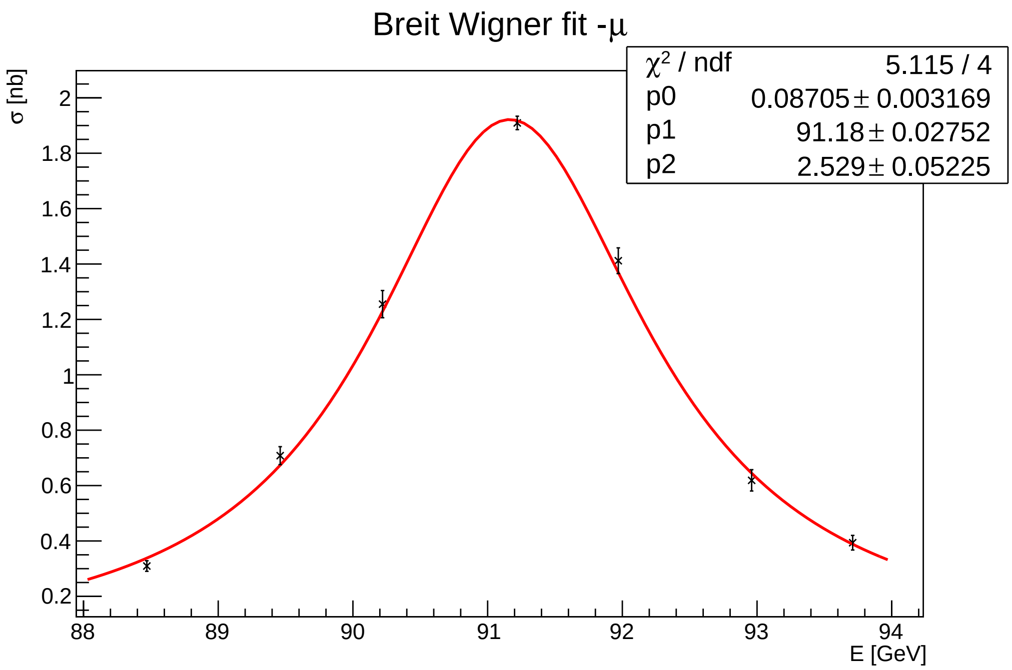

Abbildung 18: The Breit Wigner fit for the cross section of muons.

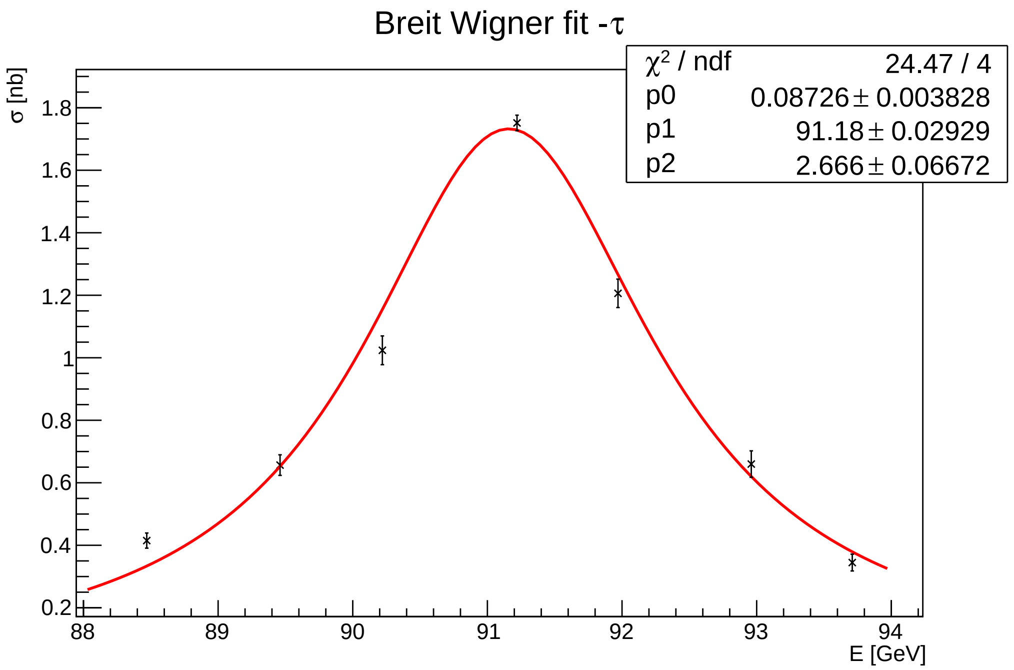

Abbildung 19: The Breit Wigner fit for the cross section of tauons.

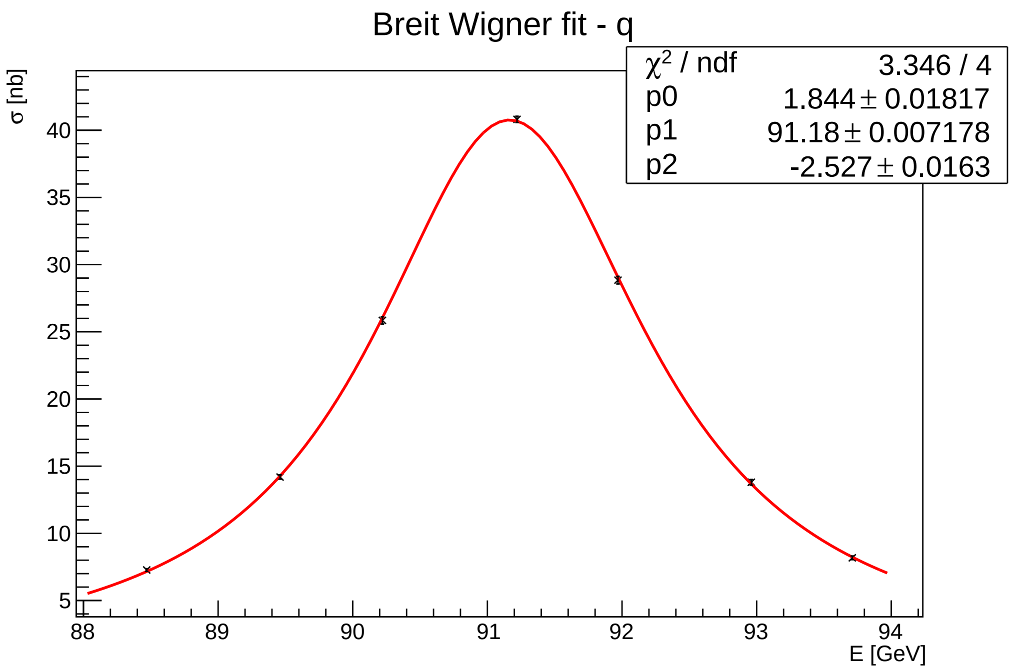

Abbildung 20: The Breit Wigner fit for the cross section of quarks.

We get the following values of the fits:

	electrons	muons	tauons	quarks
χ^2	0,807	1,279	6,117	0,836
M_Z [GeV]	91,14 ± 0,06	91,18 ± 0,03	91,18 ± 0,03	91,182 ± 0,007
Γ_Z [GeV]	2,15 ± 0,16	2,52 ± 0,05	2,66 ± 0,07	2,527 ± 0,016
Γ_x [GeV]	0,081 ± 0,004	0,087 ± 0,003	0,087 ± 0,004	1,844 ± 0,018

For the fit we use the electronic width $\Gamma_e = (0,081 \pm 0,004)$ GeV. This value has an error of 4,97%, which is not taken into account in the fit function. In the multiplication the factor $\Gamma_{\mu/\tau/q}$ is dependent of this value. The bigger Γ_e is, the smaller is it. Because of this we add the percentage value with Gaussian error propagation to the determined width to have a more realistic error on these values.

	electrons	muons	tauons	quarks
Γ_x [GeV]	0,081 ± 0,004	0,087 ± 0,005	0,087 ± 0,006	1,84 ± 0,09

The χ^2 for electrons, muons and quarks is around 1, this confirms our estimated values. However for the tauons it is to huge with around 6, the errors are to small and there are further errors. A reason can be the small purity after the cuts in the tau cut.

3.9.1 Partial cross section on resonance maximum

We read the cross section on the resonance maximum in the four plots. This should be at $\sqrt{s} = M_Z$. For the error we use the error on the cross section of E3, they should be equal.

$$\sigma_{resonance,e} = (2,5 \pm 0,2)\ nb$$

$$\sigma_{resonance,\mu} = (1,92 \pm 0,02)\ nb$$

$$\sigma_{resonance,\tau} = (1,73 \pm 0,02)\ nb$$

$$\sigma_{resonance,q} = (40,8 \pm 0,2)\ nb$$

3.9.2 The Z^0 boson's mass

With the four masses of the four fits we want to compute the final mass with the weighted mean:

$$\overline{M_Z} = \frac{\sum_{i=1}^{4} \frac{M_{Z,i}}{s^2_{M_{Z,i}}}}{\sum_{i=1}^{4} \frac{1}{s^2_{M_{Z,i}}}} \tag{37}$$

$$s_{\overline{M_Z}} = \frac{1}{\sqrt{\sum_{i=1}^{4} \frac{1}{s^2_{M_{Z,i}}}}} \tag{38}$$

We get

$$\overline{M_Z} = (91,182 \pm 0,007)\ \text{GeV}$$

3.9.3 The total width Γ_Z and leptonic width Γ_l

In the same way, we compute the weighted mean of the total width $\overline{\Gamma_Z}$.
We get

$$\overline{\Gamma_Z} = (2,531 \pm 0,015) \text{ GeV}$$

For the width of the charged leptons Γ_l we compute the weighted mean of Γ_e, Γ_μ and Γ_τ:

$$\Gamma_l = (0,084 \pm 0,003) \text{ GeV}$$

3.9.4 The branching ratio

We compute the branching ratio $BR_{charged\ leptons} = \frac{\Gamma_l}{\Gamma_Z}$ and $BR_{quarks} = \frac{\Gamma_q}{\Gamma_Z}$ (formula (13) on page 9). The error is computed with Gauß:

$$s_{BR_x} = \frac{\Gamma_x}{\Gamma_Z} \cdot \sqrt{\frac{s_{\Gamma_x}}{\Gamma_x}^2 + \frac{s_{\overline{\Gamma_Z}}}{\overline{\Gamma_Z}}^2} \tag{39}$$

We get the following values:

$$BR_{charged\ leptons} = (10,0 \pm 0,3)\ \%$$

$$BR_{quarks} = (73 \pm 4)\ \%$$

3.9.5 The invisible width and the number of neutrino families

We compute the decay width of the neutrinos $\Gamma_{\nu,total}$ with the following formula:

$$\Gamma_{\nu,total} = \Gamma_Z - \Gamma_e - \Gamma_\mu - \Gamma_\tau - \Gamma_q \tag{40}$$

The error is computed with:

$$s_{\Gamma_{\nu,total}} = \sqrt{(s_{\Gamma_Z})^2 + (s_{\Gamma_e})^2 + (s_{\Gamma_\mu})^2 + (s_{\Gamma_\tau})^2 + (s_{\Gamma_q})^2} \tag{41}$$

We get:

$$\Gamma_{\nu,total} = (0,43 \pm 0,09) \text{ GeV}$$

To determine the number of neutrino families N_ν, we use the theoretical value for the single neutrino width $\Gamma_\nu = 167{,}6$ MeV.

$$N_\nu = \frac{\Gamma_{\nu,total}}{\Gamma_\nu} \tag{42}$$

$$s_{N_\nu} = \frac{s_{\Gamma_{\nu,total}}}{\Gamma_\nu} \tag{43}$$

We get:

$$N_\nu = 2,6 \pm 0,6$$

3.10 Forward backward asymmetry

At last we want to have a look on the forward backward asymmetry. For this we want to analyse on the one hand the Monte Carlo data for muons, on the other the real data on the resonance energy of 91,22 GeV. We plot the number of events over $\cos\Theta = x$. The error on each bin is again $\sqrt{N}$. We fit the following function to get the parameters A and B:

$$N = A \cdot (1 + x^2) + B \cdot 2x \tag{44}$$

The boarders of the fit are chosen as -0,85 and 0,85 to avoid the disturbing effects for big or small angles. With equation (16) on page 9 it is:

$$A_{FB} = \frac{3}{4}\frac{F_2}{F_1} = \frac{3}{4}\frac{B}{A} \tag{45}$$

With this equation we are able to compute the asymmetry with an error of

$$s_{A_{FB}} = \frac{3}{4}\frac{B}{A}\sqrt{\frac{s_A}{A}^2 + \frac{s_B}{B}^2} \tag{46}$$

Now we can compute the Weinberg angle respectively $\sin\Theta_W^2$ (formula (17) on page 9):

$$\sin\Theta_W^2 = \frac{1}{4} - \frac{1}{4} \cdot \sqrt{\frac{A_{FB}}{3}} \tag{47}$$

with an error:

$$s_{\sin\Theta_W^2} = \frac{1}{4} \cdot \frac{s_{A_{FB}}}{2\sqrt{3A_{FB}}} \tag{48}$$

In figure 21 is the plot for the Monte Carlo Data, in figure 22 you can see the plot for the real data. The error bars are again the binomial error with $\sqrt{N}$.

After the fit we get the following values:

	Monte Carlo Data	Real Data (E4)
χ^2	1,113	1,031
A	1436 ± 5	62,0 ± 1,1
B	12 ± 7	-0,8 ± 1,5
A_{FB}	0,006 ± 0,004	0,010 ± 0,018
$\sin\Theta_W^2$	0,238 ± 0,003	0,235 ± 0,013

The χ^2 is around one, this confirms again our errors. The asymmetry is very small, because of this, the parameter B is very small in comparison to the errors of the values. The error on this parameter is at the real data over 100 %. Also the error on the asymmetry is this big. Because of this, we do not have a good result for the asymmetry for this experiment. We would need much more data to reduce the error bars. A fit on another energy except E4 would be even worse, because here we have even less data.

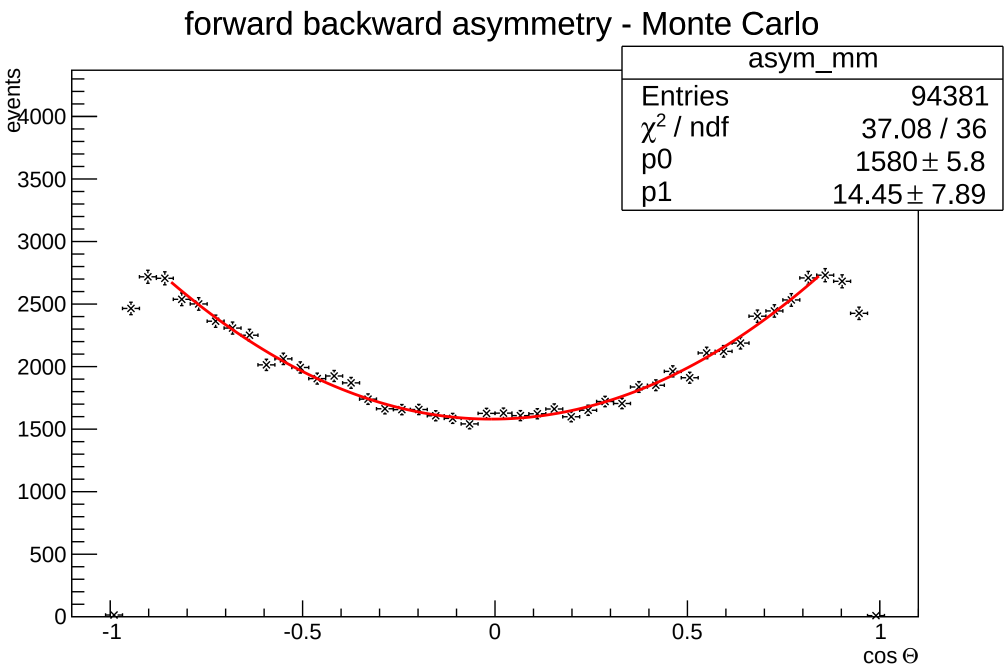

Abbildung 21: The fit for the forwards backwards asymmetry of muons with the Monte Carlo data.

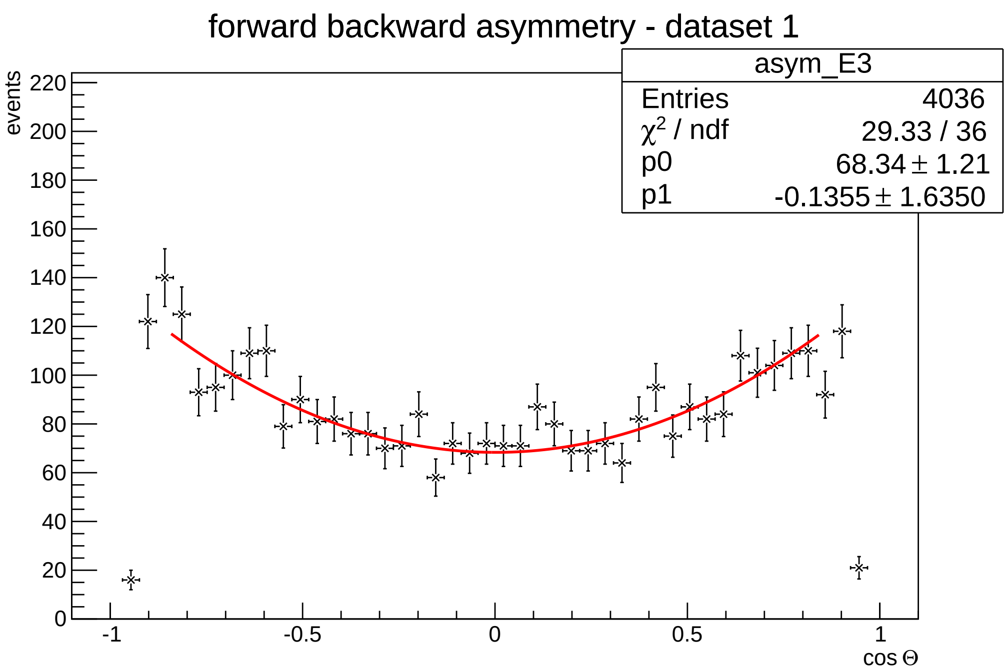

Abbildung 22: The fit for the forwards backwards asymmetry of muons with the real data of the resonance energy E3.

4 Result and discussion

"Physical Review D: Particle and Fields" (E.J. Weinberg and G.L. Nordstrom) gives the comparing data to give a more reliable view on the experiments values.

4.1 Partial cross section

We determined the partial cross section at the resonance maximum at all four channels. Caused by the lack of literature values, we compare our results with the computed theoretical values.

	result [nb]	theoretical value [nb]
$\sigma_{resonance,e}$	2,5 ± 0,2	2,00
$\sigma_{resonance,\mu}$	1,92 ± 0,02	2,00
$\sigma_{resonance,\tau}$	1,73 ± 0,02	2,00
$\sigma_{resonance,q}$	40,8 ± 0,2	41,43

These values are not very good, none is in the standard derivation of the computed value. One has to regard, that we do not compare them with literature values, but only with theoretical values, which are only an approximation. Above all the quarks are really good. Because of the lepton universality, the cross sections for electrons, muons and tauons should be almost the same. Also this we can't confirm. The value for the electrons is to high. This could be a result of the problems in the s-t-channel separation. The factor is to big in this case. At last it is possible, that there is a mistake in the computing, which we didn't notice. It is interesting, that the other values we determine are much better. We should be sceptical about the cross section at resonance maximum.

4.2 The mass of the Z^0 boson

With the Breit Wigner Fit we were able to find a value for the mass of the Z^0 boson M_Z for every channel. We computed the weighted mean to get a final result. Here we can compare it with the literature value.

	result [GeV]	literature value [GeV]
M_Z	$91,182 \pm 0,007$	$91,1876 \pm 0,0021$

The literature value of the boson mass is inside one standard derivation of our result.

4.3 The total decay width Γ_Z

Also with the Breit Wigner fit we where able to find four values for the total width Γ_Z. The error on the electron fit is very big in comparison to the error of the quark decay for example. We used the weighted mean to get a final result:

	result [GeV]	literature value [GeV]
Γ_Z	$2,531 \pm 0,015$	$2,4952 \pm 0,0023$

Here we are inside three standard derivations. Considering the problems we represented, it is a good result.

4.4 The leptonic and hadronic width

We get with the Breit Wigner fit the specific width for all four channels. With the weighted mean of the electron, muon and tauon width we were able to determine the leptonic width. The quark width is the hadron width. Because of the error Γ_e in the fit function we had to increase the error of the fit. We get the following results and literature values:

	result [GeV]	literature value [GeV]
Γ_e	81 ± 4	$83,984 \pm 0,086$
Γ_μ	87 ± 5	$83,984 \pm 0,086$
Γ_τ	87 ± 6	$83,984 \pm 0,086$
Γ_l	84 ± 3	$83,984 \pm 0,086$
Γ_q	1840 ± 90	$1744,4 \pm 2,0$

All the three results for the lepton and even the mean of the three lepton channels include the literature value inside one standard derivation. This confirms the lepton universality, in contrary to the cross section. Above all the median is rather good. The electron width is smaller than the literature value, due to this fact the width of muons and tauons is because of the product bigger. The median is compensating this.
Because of the same reason, the width of the quarks is to big. The literature value has an distance of circa one standard derivation.

4.5 The branching ratio

We computed the branching ratio for the charged leptons and quarks. We will compare it again with our theoretical computed value.

	result [%]	theoretical value [%]
$BR_{charged\ leptons}$	$10,0 \pm 0,3$	10,1
BR_{quarks}	73 ± 4	69,7

Both computed values are inside one standard derivation with our determined values. Again these results are rather good

4.6 The invisible width

With the total decay width and our four determined width we where able to compute the absent width. With the theoretical value of the width of the neutrinos we compute their number.

	result	literature value
Γ_ν	(0,43 ± 0,09) GeV	(0,4990 ± 0,0015) GeV
N_ν	2,6 ± 0,6	3

Both values are inside one standard derivation with the literature value. Still we can't determine the number of neutrino families exact, because also two has a distance of one standard derivation of our result.

4.7 Forward backward asymmetry

Finally we determined the asymmetry for the Monte Carlo data of muons and for the resonance energy E4 for muons. We get the following results: (We take the literature value for the Weinberg angle of the instruction)

	result	literature value
$A_{FB,MC}$	0,006 ± 0,004	
$A_{FB,E4}$	0,010 ± 0,018	
$\sin\Theta^2_{W\,MC}$	0,238 ± 0,003	0,2312
$\sin\Theta^2_{W\,E4}$	0,235 ± 0,013	0,2312

The distance between the Weinberg angle of the Monte Carlo data and the literature value is two standard derivations. Maybe here are additional errors because of the not perfect simulated data. The error on the asymmetry of the real data is really big. Because of this the Weinberg angle is inside one standard derivation. Due to the huge errors in the real data we decided against the investigation of other energies.

A Plots and fits for s-t-channel separation

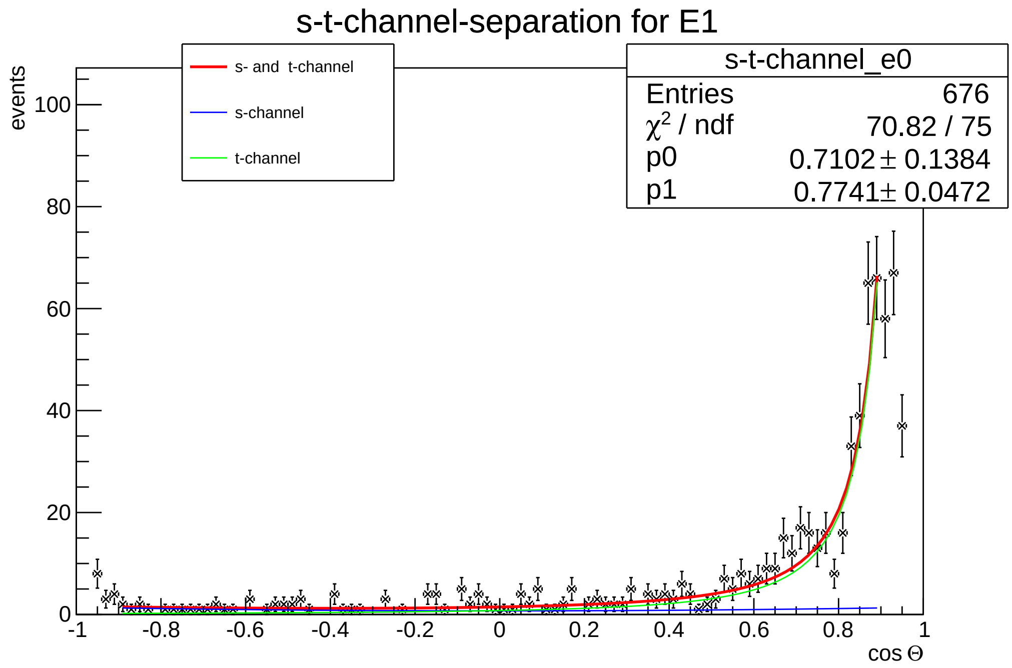

Abbildung 23: The fit for the separation of the s- and t-channel in the electron data for E1.

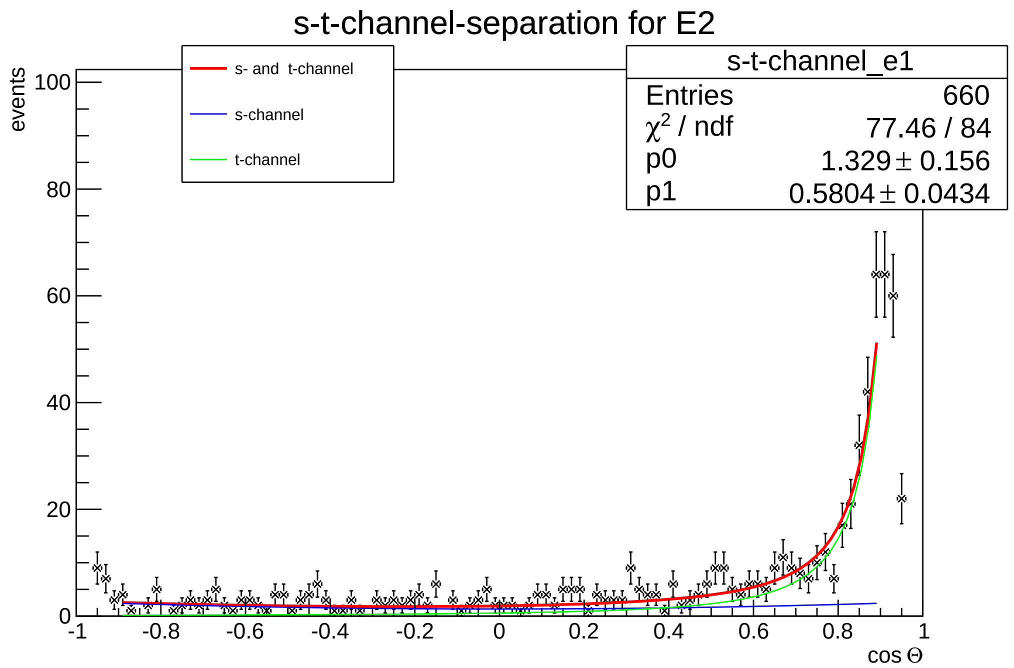

Abbildung 24: The fit for the separation of the s- and t-channel in the electron data for E2.

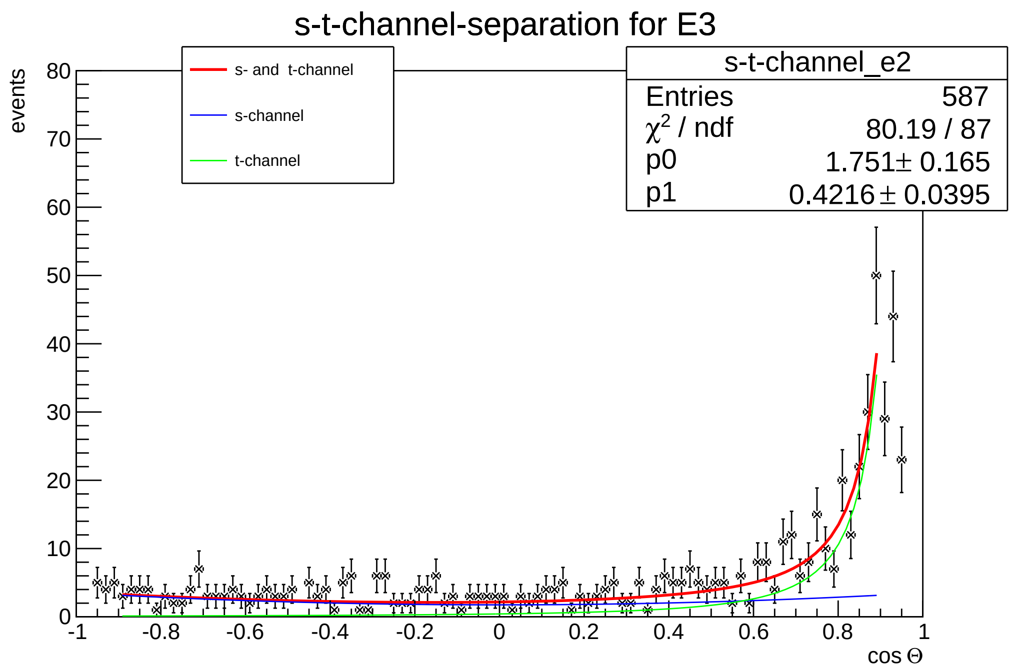

Abbildung 25: The fit for the separation of the s- and t-channel in the electron data for E3.

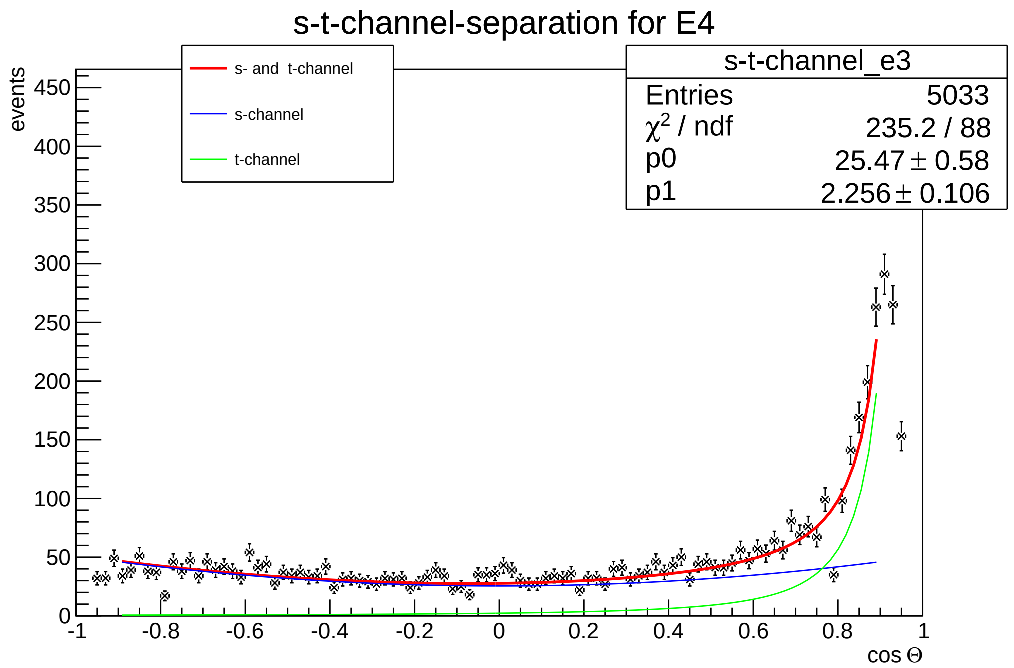

Abbildung 26: The fit for the separation of the s- and t-channel in the electron data for E4.

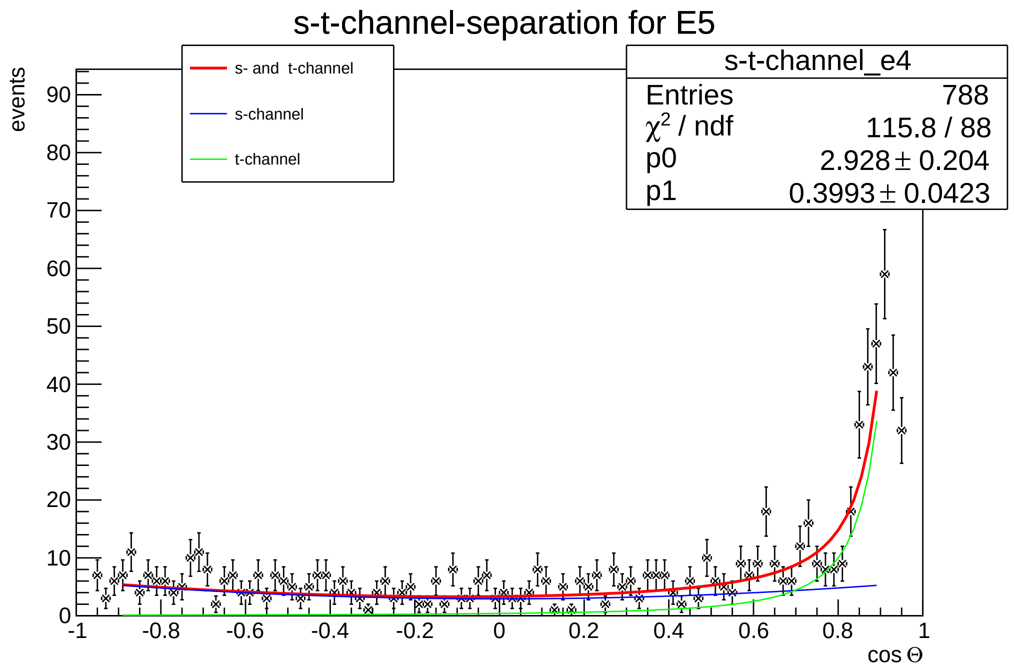

Abbildung 27: The fit for the separation of the s- and t-channel in the electron data for E5.

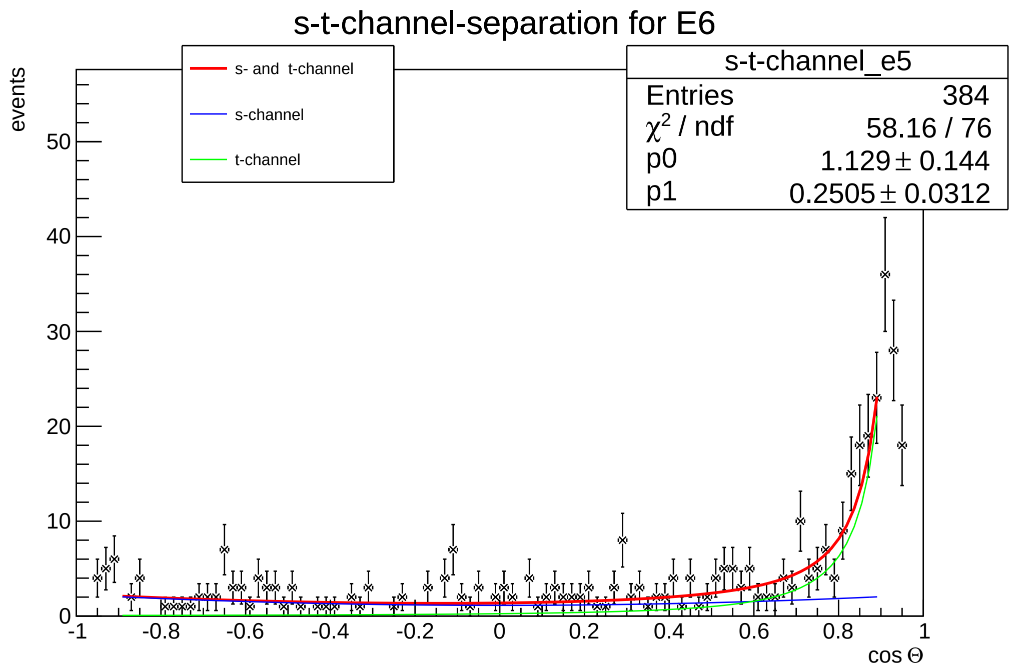

Abbildung 28: The fit for the separation of the s- and t-channel in the electron data for E6.

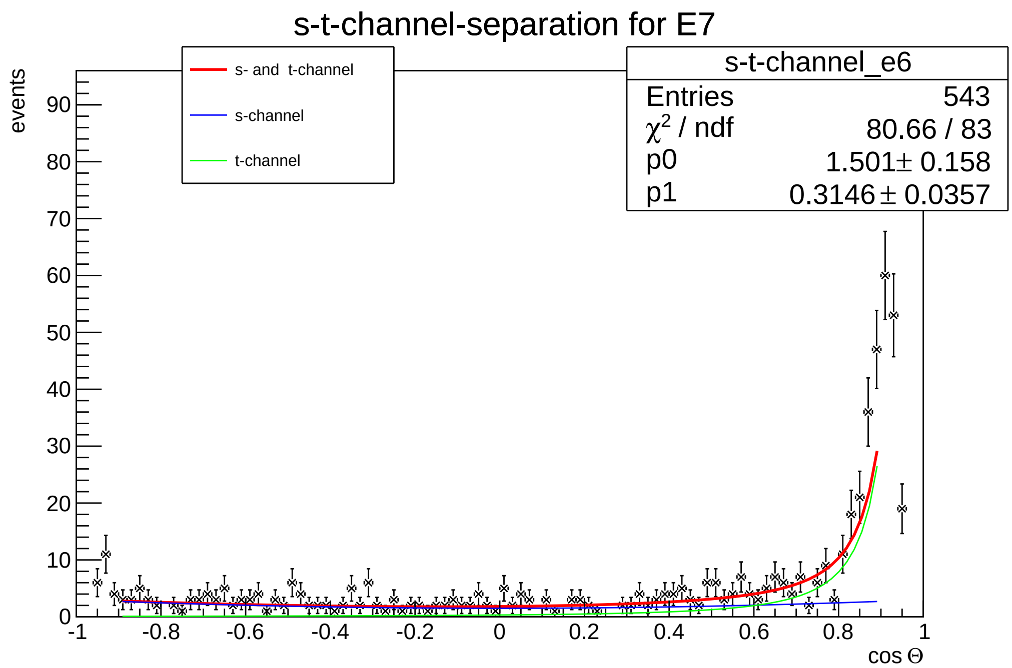

Abbildung 29: The fit for the separation of the s- and t-channel in the electron data for E7.

B The code in root

On the following pages we figure our written code in the root file. Nearly all of the computing is based on this file, only the theoretical values and the weighted means at the end we computed separate with excel

```
// run.C
// mainfile, type .x run.C++ to excecute the whole analysis in root


// Bibliotheken
   #include "TString.h"
   #include "TError.h"
   #include "TMatrixT.h"
   #include "TLatex.h"
   #include <iostream>
   #include <vector>


// Funktionen
   #include "print.C"
   #include "stack.C"
   #include "cut.C"
   #include "fit_s_t.C"
   #include "theo_s_t.C"
   #include "fit_bw.C"
   #include "fit_asym.C"


// namespace
   using namespace std;


void run(){

   // Parameter

      // s-t-Kanaltrennung

         // Fitgrenzen
            float l = -0.9;
            float r = 0.9;

         // Integralgrenzen
            float o = 0.95;
            float u = -0.95;


      // allgemeine cut-Bedingungen für Ncharged
         TString cut_cond[4][3];

         // e-cut
            cut_cond[0][0] = "0<Ncharged && Ncharged<4";
            cut_cond[0][0] += " && ";
            cut_cond[0][0] += u;
            cut_cond[0][0] += "<cos_thet && cos_thet<";
            cut_cond[0][0] += o;

         // m-cut
            cut_cond[1][0] = "0<Ncharged && Ncharged<5";

         // t-cut
            cut_cond[2][0] = "1<Ncharged && Ncharged<6";

         // q-cut
            cut_cond[3][0] = "7<Ncharged";

         // Setzen der allgemeinen cut-Bedingungen
            for(int i=0; i<4; i=i+1){
               cut_cond[i][1] = cut_cond[i][0];
               cut_cond[i][2] = cut_cond[i][0];
            }


      // cut-Bedingungen für Effizienzmatrix

         // e-cut
            cut_cond[0][1] += " && 80<E_ecal";

         // m-cut
```

```
            cut_cond[1][1] += " && 70<Pcharged && E_ecal<30";

         // t-cut
            cut_cond[2][1] += " && Pcharged<70 && E_ecal<75";

         // q-cut
            cut_cond[3][1] += "";


      // cut-Bedingungen für Ereignismatrix

         // e-cut
            cut_cond[0][2] += " && ";
            cut_cond[0][2] += 80/45.64;
            cut_cond[0][2] += "<E_ecal/E_lep";

         // m-cut
            cut_cond[1][2] += " && ";
            cut_cond[1][2] += 70/45.62;
            cut_cond[1][2] += "<Pcharged/E_lep && E_ecal/E_lep<";
            cut_cond[1][2] += 30/45.62;

         // t-cut
            cut_cond[2][2] += " && Pcharged/E_lep<";
            cut_cond[2][2] += 70/45.64;
            cut_cond[2][2] += " && E_ecal/E_lep<";
            cut_cond[2][2] += 75/45.64;

         // q-cut
            cut_cond[3][2] += "";


      // cut-Bedingungen für Energien
         TString cut_cond_E[7] = {
            " && 44.2<E_lep && E_lep<44.3",     // E1
            " && 44.64<E_lep && E_lep<44.76",   // E2
            " && 45.05<E_lep && E_lep<45.2",    // E3
            " && 45.5<E_lep && E_lep<45.7",     // E4
            " && 45.9<E_lep && E_lep<46.05",    // E5
            " && 46.45<E_lep && E_lep<46.55",   // E6
            " && 46.84<E_lep && E_lep<46.9"     // E7
         };


      // Vorwärts-Rückwärts-Asymmetrie

         // Fitgrenzen
            float l_asym[2] = {
               -0.85,   // Monte Carlo
               -0.85,   // E3
            };

            float r_asym[2] = {
               0.85,    // Monte Carlo
               0.85,    // E3
            };


   // Begrenzung der Ausgaben im Terminal
      gErrorIgnoreLevel = kError;


   // Styles
      gROOT->Reset();
      gROOT->SetStyle("Modern");


   // Monte Carlo Histogramme

      // Parameter
         TString input_filename_mc[4] = {"ee", "mm", "tt", "qq"};
         TString title_mc[4] = {"Ncharged", "Pcharged", "E_ecal", "E_hcal"};
         TString histogram_x_axis[4] = {"number of charged tracks", "E [GeV]", "E [GeV]", "E [GeV]"};
```

```
        float size_mc[4][3] = {
            {40, 0, 40},
            {100, 0, 100},
            {100, 0, 120},
            {100, 0, 40}
        };

    // einzelne Histogramme ausgeben
        for(int i=0; i<4; i=i+1){
            for(int j=0; j<4; j=j+1){

            // ohne cut-Bedingungen
                print(input_filename_mc[j], "h3", title_mc[i], size_mc[i][0], size_mc[i][1], size_mc[i][2], title_mc[i]+"_"+input_filename_mc[j], "", title_mc[i]+"_"+input_filename_mc[j], histogram_x_axis[i], "events");

            // mit cut-Bedingungen
                print(input_filename_mc[j], "h3", title_mc[i], size_mc[i][0], size_mc[i][1], size_mc[i][2], title_mc[i]+"_cut_"+input_filename_mc[j], cut_cond[j][1], title_mc[i]+"_cut_"+input_filename_mc[j], histogram_x_axis[i], "events");
            }

            // ee_cos_thet
                print("ee", "h3", "cos_thet", 100, -1.5, 1.5, "ee_cos_thet", "", "cos #Theta progression for electron events (Monte Carlo)", "cos #Theta", "events");

            // ee_cos_thet, Ncharged==0
                print("ee", "h3", "cos_thet", 100, -1.5, 1.5, "ee_cos_thet_Ncharged", "Ncharged==0", "cos #Theta progression for electron events (Monte Carlo for Ncharged = 0)", "cos #Theta", "events");


        // Histogramme zusammenfügen
            stack(title_mc[i], input_filename_mc, "stack_"+title_mc[i], "Monte Carlo - "+title_mc[i]+" without cut", histogram_x_axis[i], "a.u.");
            stack(title_mc[i]+"_cut", input_filename_mc, "stack_cut_"+title_mc[i], "Monte Carlo - "+title_mc[i]+" applied cut", histogram_x_axis[i], "a.u.");
        }


  // Effizienzmatrix
    vector<float> e_cut[4];
    vector<float> m_cut[4];
    vector<float> t_cut[4];
    vector<float> q_cut[4];

    TMatrixT<float>eff_n(4,4);
    TMatrixT<float>eff_cut(4,4);
    TMatrixT<float>eff(4,4);

    // cuts
        for(int i=0; i<4; i=i+1){
            e_cut[i] = cut("ee", "h3", cut_cond[i][1]);
            m_cut[i] = cut("mm", "h3", cut_cond[i][1]);
            t_cut[i] = cut("tt", "h3", cut_cond[i][1]);
            q_cut[i] = cut("qq", "h3", cut_cond[i][1]);

            eff_n(i,0) = e_cut[i].at(0);
            eff_n(i,1) = m_cut[i].at(0);
            eff_n(i,2) = t_cut[i].at(0);
            eff_n(i,3) = q_cut[i].at(0);
            eff_cut(i,0) = e_cut[i].at(1);
            eff_cut(i,1) = m_cut[i].at(1);
            eff_cut(i,2) = t_cut[i].at(1);
            eff_cut(i,3) = q_cut[i].at(1);
            eff(i,0) = e_cut[i].at(2);
            eff(i,1) = m_cut[i].at(2);
            eff(i,2) = t_cut[i].at(2);
            eff(i,3) = q_cut[i].at(2);
        }

    // --> Zeilenvektor: cut
    //
    //  |  Spaltenvektor: events im cut
```

```
      //  v

      // Ausgaben
         cout << endl;
         cout << "Anzahl simulierter Ereignisse ohne cut:" << endl;
         eff_n.Print();
         cout << endl;
         cout << endl;

         cout << "Anzahl simulierter Ereignisse mit cut:" << endl;
         eff_cut.Print();
         cout << endl;
         cout << endl;

         cout << "Effizienzmatrix:" << endl;
         eff.Print();
         cout << endl;
         cout << endl;

      // Fehler auf Effizienzmatrix
         TMatrixT<float> s_eff(4,4);

         for(int i=0; i<4; i=i+1){
            for(int j=0; j<4; j=j+1){
               s_eff(i,j) = sqrt(eff(i,j)*(1-eff(i,j))/eff_n(i,0));
            }
         }

         // Ausgabe
            cout << "Fehler auf Effizienzmatrix:" << endl;
            s_eff.Print();
            cout << endl;
            cout << endl;


   // invertierte Effizienzmatrix
      TMatrixT<float> eff_inv = eff;
      eff_inv = eff_inv.Invert();

      // Ausgabe
         cout << "invertierte Effizienzmatrix:" << endl;
         eff_inv.Print();
         cout << endl;
         cout << endl;

      // Fehler auf invertierte Effizienzmatrix
         TMatrixT<float> eff_plus(4,4);
         TMatrixT<float> eff_minus(4,4);

         for(int i=0; i<4; i=i+1){
            for(int j=0; j<4; j=j+1){
               eff_plus(i,j) = eff(i,j)+s_eff(i,j);
               eff_minus(i,j) = eff(i,j)-s_eff(i,j);
            }
         }

         TMatrixT<float> eff_plus_inv(4,4);
         eff_plus_inv = eff_plus;
         eff_plus_inv = eff_plus_inv.Invert();

         TMatrixT<float> eff_minus_inv(4,4);
         eff_minus_inv = eff_minus;
         eff_minus_inv = eff_minus_inv.Invert();

         TMatrixT<float> s_eff_inv(4,4);
         TMatrixT<float> s_eff_inv_proc(4,4);
         for(int i=0; i<4; i=i+1){
            for(int j=0; j<4; j=j+1){
               s_eff_inv(i,j) = fabs((eff_plus_inv(i,j)-eff_minus_inv(i,j))/2);
               s_eff_inv_proc(i,j) = fabs((eff_plus_inv(i,j)-eff_minus_inv(i,j))/2)/fabs(eff_inv
(i,j))*100;
            }
         }

```

```
         // Ausgabe
            cout << "Fehler auf invertierte Effizienzmatrix:" << endl;
            s_eff_inv.Print();
            cout << endl;
            cout << endl;

            cout << "prozentualer Fehler auf invertiere Effizienzmatrix:" << endl;
            s_eff_inv_proc.Print();
            cout << endl;
            cout << endl;


   // Histogramme daten_1

      // Parameter
         TString title_d[4] = {"Ncharged", "Pcharged", "E_ecal", "E_hcal"};
         TString histogram_x_axis_d[4] = {"number of charged tracks", "E [GeV]", "E [GeV]", "E [GeV]"};
         float size_d[4][3] = {
            {40, 0, 40},
            {100, 0, 100},
            {100, 0, 120},
            {100, 0, 40}
         };
         TString dataset_d[4] = {"e", "m", "t", "q"};

      // einzelne Histogramme ausgeben
         for(int i=0; i<4; i=i+1){

            // ohne cut-Bedingungen
               print("daten_1", "h33", title_d[i], size_d[i][0], size_d[i][1], size_d[i][2], "stack_daten_1_"+title_d[i], "", "Datensatz 1 - "+title_d[i], histogram_x_axis_d[i], "events");

            // mit cut-Bedingungen
               for(int j=0; j<4; j=j+1){
                  print("daten_1", "h33", title_d[i], size_d[i][0], size_d[i][1], size_d[i][2], "daten_1_cut_"+title_d[i]+"_"+dataset_d[j], cut_cond[j][2], "daten_1_cut_"+title_d[i]+"_"+dataset_d[j], histogram_x_axis_d[i], "events");
               }

         // Histogramme zusammenfügen
            stack("daten_1_cut_"+title_d[i], dataset_d, "stack_daten_1_cut_"+title_d[i], "dataset 1 - "+title_d[i]+" applied cut", histogram_x_axis_d[i], "a.u.");
         }


   // Anzahl realer Daten
      vector<float> E0_cut[4];
      vector<float> E1_cut[4];
      vector<float> E2_cut[4];
      vector<float> E3_cut[4];
      vector<float> E4_cut[4];
      vector<float> E5_cut[4];
      vector<float> E6_cut[4];

      TMatrixT<float>real_n(4,7);
      TMatrixT<float>real_cut(4,7);
      TMatrixT<float>real(4,7);

      for(int i=0; i<4; i=i+1){
         E0_cut[i] = cut("daten_1", "h33", cut_cond[i][2]+cut_cond_E[0]);
         E1_cut[i] = cut("daten_1", "h33", cut_cond[i][2]+cut_cond_E[1]);
         E2_cut[i] = cut("daten_1", "h33", cut_cond[i][2]+cut_cond_E[2]);
         E3_cut[i] = cut("daten_1", "h33", cut_cond[i][2]+cut_cond_E[3]);
         E4_cut[i] = cut("daten_1", "h33", cut_cond[i][2]+cut_cond_E[4]);
         E5_cut[i] = cut("daten_1", "h33", cut_cond[i][2]+cut_cond_E[5]);
         E6_cut[i] = cut("daten_1", "h33", cut_cond[i][2]+cut_cond_E[6]);

         real_n(i,0) = E0_cut[i].at(0);
         real_n(i,1) = E1_cut[i].at(0);
         real_n(i,2) = E2_cut[i].at(0);
         real_n(i,3) = E3_cut[i].at(0);
         real_n(i,4) = E4_cut[i].at(0);
         real_n(i,5) = E5_cut[i].at(0);
```

```
            real_n(i,6) = E6_cut[i].at(0);

            real_cut(i,0) = E0_cut[i].at(1);
            real_cut(i,1) = E1_cut[i].at(1);
            real_cut(i,2) = E2_cut[i].at(1);
            real_cut(i,3) = E3_cut[i].at(1);
            real_cut(i,4) = E4_cut[i].at(1);
            real_cut(i,5) = E5_cut[i].at(1);
            real_cut(i,6) = E6_cut[i].at(1);

            real(i,0) = E0_cut[i].at(2);
            real(i,1) = E1_cut[i].at(2);
            real(i,2) = E2_cut[i].at(2);
            real(i,3) = E3_cut[i].at(2);
            real(i,4) = E4_cut[i].at(2);
            real(i,5) = E5_cut[i].at(2);
            real(i,6) = E6_cut[i].at(2);
         }

         // Ausgaben
            cout << "Anzahl realer Daten ohne cut:" << endl;
            real_n.Print();
            cout << endl;
            cout << endl;

            cout << "Anzahl realer Daten mit cut:" << endl;
            real_cut.Print();
            cout << endl;
            cout << endl;

         // Fehler auf Anzahl realer Daten mit cut
            TMatrixT<float> s_real_cut(4,7);

            for(int i=0; i<4; i=i+1){
               for(int j=0; j<7; j=j+1){
                  s_real_cut(i,j) = sqrt(real_cut(i,j));
               }
            }

            // Ausgabe
               cout << "Fehler auf Anzahl realer Daten mit cut:" << endl;
               s_real_cut.Print();
               cout << endl;
               cout << endl;


      // s-t-Kanaltrennung

         // Prints
            print("daten_1", "h33", "cos_thet", 100, -1, 1, "s-t-channel_e0", cut_cond[0][2]
      +cut_cond_E[0], "s-t-channel-separation for E1", "cos #Theta", "events");
            print("daten_1", "h33", "cos_thet", 100, -1, 1, "s-t-channel_e1", cut_cond[0][2]
      +cut_cond_E[1], "s-t-channel-separation for E2", "cos #Theta", "events");
            print("daten_1", "h33", "cos_thet", 100, -1, 1, "s-t-channel_e2", cut_cond[0][2]
      +cut_cond_E[2], "s-t-channel-separation for E3", "cos #Theta", "events");
            print("daten_1", "h33", "cos_thet", 100, -1, 1, "s-t-channel_e3", cut_cond[0][2]
      +cut_cond_E[3], "s-t-channel-separation for E4", "cos #Theta", "events");
            print("daten_1", "h33", "cos_thet", 100, -1, 1, "s-t-channel_e4", cut_cond[0][2]
      +cut_cond_E[4], "s-t-channel-separation for E5", "cos #Theta", "events");
            print("daten_1", "h33", "cos_thet", 100, -1, 1, "s-t-channel_e5", cut_cond[0][2]
      +cut_cond_E[5], "s-t-channel-separation for E6", "cos #Theta", "events");
            print("daten_1", "h33", "cos_thet", 100, -1, 1, "s-t-channel_e6", cut_cond[0][2]
      +cut_cond_E[6], "s-t-channel-separation for E7", "cos #Theta", "events");

            // theoretischer Verlauf
            theo_s_t();

         // Korrekturfaktoren für Ereignismatrix
            vector<float> s_t[7];

            s_t[0] = fit_s_t("s-t-channel_e0", l, r, o, u);
            s_t[1] = fit_s_t("s-t-channel_e1", l, r, o, u);
            s_t[2] = fit_s_t("s-t-channel_e2", l, r, o, u);
            s_t[3] = fit_s_t("s-t-channel_e3", l, r, o, u);
```

```
        s_t[4] = fit_s_t("s-t-channel_e4", l, r, o, u);
        s_t[5] = fit_s_t("s-t-channel_e5", l, r, o, u);
        s_t[6] = fit_s_t("s-t-channel_e6", l, r, o, u);

        TMatrixT<float> s_t_factor(1,7);
        for(int i=0; i<7; i=i+1){
           s_t_factor(0,i) = s_t[i].at(3);
        }

     // Ausgabe
        cout << "s-t-Korrekturfaktor:" << endl;
        s_t_factor.Print();
        cout << endl;
        cout << endl;

     // Fehler auf s-t-Korrekturfaktor
        TMatrixT<float> s_s_t_factor(2,7);

        // Gaußfehler
           for(int i=0; i<7; i=i+1){
              s_s_t_factor(0,i) = s_t[i].at(6);
           }

        // Gesamtfehler
           float proc = 0.1;

           for(int i=0; i<7; i=i+1){
              s_s_t_factor(1,i) = sqrt(s_s_t_factor(0,i)*s_s_t_factor(0,i)+proc*s_t_factor
(0,i)*proc*s_t_factor(0,i));
           }

        // Ausgabe
           cout << "Fehler auf s-t-Korrekturfaktor:" << endl;
           s_s_t_factor.Print();
           cout << endl;
           cout << endl;


  // Anzahl realer Daten mit cut und s-t-Trennung
     TMatrixT<float> real_cut_s_t(4,7);

     //Matrix füllen
        for(int i=0; i<4; i=i+1){
           for(int j=0; j<7; j=j+1){
              if(i==0){
                 real_cut_s_t(0,j) = real_cut(0,j)*s_t_factor(0,j);
              }
              else{
                 real_cut_s_t(i,j) = real_cut(i,j);
              }
           }
        }

     // Ausgabe
        cout << "Anzahl realer Daten mit cut und s-t-Trennung:" << endl;
        real_cut_s_t.Print();
        cout << endl;
        cout << endl;

     // Fehler auf Ereignismatrix
        TMatrixT<float> s_real_cut_s_t(4,7);

        for(int i=0; i<4; i=i+1){
           for(int j=0; j<7; j=j+1){
              if(i==0){
                 s_real_cut_s_t(0,j) = sqrt(s_t_factor(0,j)*s_real_cut(0,j)*s_t_factor
(0,j)*s_real_cut(0,j)+real_cut(0,j)*s_s_t_factor(1,j)*real_cut(0,j)*s_s_t_factor(1,j));
              }
              else{
                 s_real_cut_s_t(i,j) = s_real_cut(i,j);
              }
           }
        }
```

```
          // Ausgabe
             cout << "Fehler auf Anzahl realer Daten mit cut und s-t-Trennung:" << endl;
             s_real_cut_s_t.Print();
             cout << endl;
             cout << endl;


    // Ereignismatrix
       TMatrixT<float> erg(4,7);

       TMatrixT<float> n(4,1);
       TMatrixT<float> real_E_cut_s_t(4,1);

       for(int i=0; i<7; i=i+1){
          for(int j=0; j<4; j=j+1){
             real_E_cut_s_t(j,0) = real_cut_s_t(j,i);
          }

          n = eff_inv*real_E_cut_s_t;

          for(int k=0; k<4; k=k+1){
             erg(k,i) = n(k,0);
          }
       }

       // Ausgabe
          cout << "Ereignismatrix:" << endl;
          erg.Print();
          cout << endl;
          cout << endl;

       // Fehler auf Ereignismatrix
          TMatrixT<float> s_erg(4,7);
          float temp[5];

          for(int i=0; i<7; i=i+1){
             for(int j=0; j<4; j=j+1){
                for(int k=0; k<4; k=k+1){
                   temp[k] = eff_inv(j,k)*s_real_cut_s_t(k,i)*eff_inv(j,k)*s_real_cut_s_t(k,i)+s_eff_inv(j,k)*real_cut_s_t(k,i)*s_eff_inv(j,k)*real_cut_s_t(k,i);
                }

                temp[4] = sqrt(temp[0]+temp[1]+temp[2]+temp[3]);
                s_erg(j,i) = temp[4];
             }
          }

          // Ausgaben
             cout << "Fehler auf Ereignismatrix:" << endl;
             s_erg.Print();
             cout << endl;
             cout << endl;

             TMatrixT<float> s_erg_proc(4,7);
             for(int i=0; i<4; i=i+1){
                for(int j=0; j<7; j=j+1){
                   s_erg_proc(i,j) = s_erg(i,j)/erg(i,j)*100;
                }
             }

             cout << "prozentualer Fehler auf Ereignismatrix:" << endl;
             s_erg_proc.Print();
             cout << endl;
             cout << endl;


    // Wirkungsquerschnitt

       // Luminositäten
          float lumi[7];
          lumi[0] = 675.8590;
          lumi[1] = 543.6270;
          lumi[2] = 419.7760;
          lumi[3] = 3122.204;
```

```
        lumi[4] = 639.8380;
        lumi[5] = 479.2400;
        lumi[6] = 766.8380;

    // Korrekturen
        float cross_section_korr[2][7] = {
            {2.0, 4.3, 7.7, 10.8, 4.7, -0.2, -1.6},
            {0.09, 0.20, 0.36, 0.52, 0.22, -0.01, -0.08}
        };

    TMatrixT<float>cross_section(4,7);

    for(int i=0; i<4; i=i+1){
         for(int j=0; j<7; j=j+1){
            if(i!=3){
                cross_section(i,j) = erg(i,j)/lumi[j]+cross_section_korr[1][j];
            }
            else{
                cross_section(i,j) = erg(i,j)/lumi[j]+cross_section_korr[0][j];
            }
        }
    }

    // Ausgabe
        cout << "Wirkungsquerschnitt:" << endl;
        cross_section.Print();
        cout << endl;
        cout << endl;

    // Fehler auf Wirkungsquerschnitt
        float s_lumi[7];
        s_lumi[0] = 5.721257;
        s_lumi[1] = 4.830643;
        s_lumi[2] = 3.974844;
        s_lumi[3] = 22.31760;
        s_lumi[4] = 5.577354;
        s_lumi[5] = 4.481870;
        s_lumi[6] = 6.497519;

        TMatrixT<float>s_cross_section(4,7);
        for(int i=0; i<4; i=i+1){
            for(int j=0; j<7; j=j+1){
                s_cross_section(i,j) = erg(i,j)/lumi[j]*sqrt(s_erg(i,j)/erg(i,j)*s_erg(i,j)/erg(i,j)+s_lumi[j]/lumi[j]*s_lumi[j]/lumi[j]);
            }
        }

        // Ausgaben
            cout << "Fehler auf Wirkungsquerschnitt:" << endl;
            s_cross_section.Print();
            cout << endl;
            cout << endl;

            TMatrixT<float> s_cross_section_proc(4,7);
            for(int i=0; i<4; i=i+1){
                for(int j=0; j<7; j=j+1){
                    s_cross_section_proc(i,j) = s_cross_section(i,j)/cross_section(i,j)*100;
                }
            }

            cout << "prozentualer Fehler auf Wirkungsquerschnitt:" << endl;
            s_cross_section_proc.Print();
            cout << endl;
            cout << endl;


  // Breit-Wigner-Fits
    vector<float> bw[4];
    TString fit_bw_function[2];

    fit_bw_function[0] = "12*pi*x*x*[0]*[0]*0.384*10^6/([1]*[1]*((x*x-[1]*[1])^2+(x*x*x*x*[2]*[2]/([1]*[1]))))";
    bw[0] = fit_bw("bw_e", 0, cross_section, s_cross_section, fit_bw_function[0], 88, 94, "Breit Wigner fit - e");
```

```
        cout << endl;

        fit_bw_function[1] = "12*pi*x*x*";
        fit_bw_function[1] += bw[0].at(1);
        fit_bw_function[1] += "*[0]*0.384*10^6/([1]*[1]*((x*x-[1]*[1])^2+(x*x*x*x*[2]*[2]/([1]*
[1]))))";

        bw[1] = fit_bw("bw_m", 1, cross_section, s_cross_section, fit_bw_function[1], 88, 94,
"Breit Wigner fit - #mu");
        cout << endl;

        bw[2] = fit_bw("bw_t", 2, cross_section, s_cross_section, fit_bw_function[1], 88, 94,
"Breit Wigner fit - #tau");
        cout << endl;

        bw[3] = fit_bw("bw_q", 3, cross_section, s_cross_section, fit_bw_function[1], 88, 94,
"Breit Wigner fit - q");
        cout << endl;

        TString ausgabe[4] = {"e", "m", "t", "q"};

        for(int i=0; i<4; i=i+1){
           cout << endl;
           cout << ausgabe[i] << endl;
           cout << endl;
           cout << "chi^2_reduziert: " << bw[i].at(0) << endl;
           cout << "M_z: " << bw[i].at(2) << " +- " << bw[i].at(5) << endl;
           cout << "G_z: " << bw[i].at(3) << " +- " << bw[i].at(6) << endl;
           cout << "G_e,m,t,q: " << bw[i].at(1) << " +- " << bw[i].at(4) << endl;
           cout << endl;
        }

        cout << endl;

        // Ausgaben
           for(int i=0; i<7; i=i+1){
              cout << "E" << i << endl;
              cout << "N = " << s_t[i].at(7) << endl;          // N
              cout << "N_s+N_t = " << s_t[i].at(10) << endl;  // N_s+N_t
              //cout << "N_s = " << s_t[i].at(8) << endl;      // N_s
              //cout << "N_t = " << s_t[i].at(9) << endl;      // N_t
              cout << endl;
           }
           cout << endl;

     // Vorwärts-Rückwärts-Asymmetrie

        // Monte Carlo
           print("mm", "h3", "cos_thet", 50, -1.1, 1.1, "asym_mm", "", "forward backward asymmetry
- Monte Carlo", "cos #Theta", "events");
           fit_asym("asym_mm", l_asym[0], r_asym[0]);

        // daten_1
           print("daten_1", "h33", "cos_thet", 50, -1.1, 1.1, "asym_E3", cut_cond[1][2]+cut_cond_E
[3],"forward backward asymmetry - dataset 1", "cos #Theta", "events");
           fit_asym("asym_E3", l_asym[1], r_asym[1]);
}
```

```
// print.C
// Ausgabe von Histogrammen aus trees


// Bibliotheken
   #include "TString.h"
   #include "TROOT.h"
   #include "TFile.h"
   #include "TTree.h"
   #include "TCanvas.h"
   #include "TH1F.h"
   #include "TStyle.h"
   #include <iostream>


// namespace
   using namespace std;


void print(
   TString input_filename,
   TString branch,
   TString title,
   float n_bins,
   float l,
   float r,
   TString output_filename,
   TString cut_cond,
   TString histogram_title,
   TString histogram_x_axis,
   TString histogram_y_axis
){

   // Styles
      gROOT->Reset();
      gROOT->SetStyle("Modern");


   // output
      TFile *output_file = new TFile("print/"+output_filename+".root","RECREATE");


   // Daten einlesen
      TString data = "daten/"+input_filename+".root";
      TFile *input_file = new TFile(data);
      TTree *tree = (TTree*)input_file->Get(branch);


   // Canvas erzeugen
      TCanvas *canvas = new TCanvas(output_filename, output_filename, 1200, 800);
      TH1F *histogram = new TH1F(output_filename, output_filename, n_bins, l, r);


   // Styles festlegen
      histogram->SetLineColor(1);
      histogram->SetTitle(histogram_title);                 // Titel
      histogram->GetXaxis()->SetTitle(histogram_x_axis);    // x-Achse
      histogram->GetYaxis()->SetTitle(histogram_y_axis);    // y-Achse
      histogram->Draw();


   // Draw
      TString draw = title+" >> "+output_filename;
      tree->Draw(draw, cut_cond);


   // Daten in output schreiben
      output_file->cd();
      histogram->Write();


   // Canvas speichern
      canvas->Print("print/"+output_filename+".pdf", "pdf Portrait");
      canvas->Print("print/"+output_filename+".png", "png");
```

```
    // close
       delete canvas;
       input_file->Close();
       output_file->Close();
}
```

```
// stack.C
// Funktion zum Stacken von trees


// Bibliotheken
   #include <iostream>
   #include "TString.h"
   #include "TROOT.h"
   #include "THStack.h"
   #include "TCanvas.h"
   #include "TFile.h"
   #include "TH1F.h"
   #include "TLegend.h"


// namespace
   using namespace std;


void stack(
   TString titlename,
   TString dataset[4],
   TString output_filename,
   TString histogram_title,
   TString histogram_x_axis,
   TString histogram_y_axis
){

   // Styles
      gROOT->Reset();
      gROOT->SetStyle("Modern");


   // input
      int color[4] = {1, 2, 3, 4};


   // Stack erzeugen
      THStack *stack = new THStack("stack","stack");


   // Histogramme einlesen und zu Stack hinzufügen
      TString title[4];
      TString data[4];
      TFile *input_file[4];
      TH1F *histogram[4];

      for(int i=0; i<4; i=i+1){
         title[i] = titlename+"_"+dataset[i];
         data[i] = "print/"+title[i]+".root";
         input_file[i] = new TFile(data[i]);
         histogram[i] = (TH1F*)input_file[i]->Get(title[i]);

         // Style und Scale
            histogram[i]->SetLineColor(color[i]);
            histogram[i]->Scale(1./histogram[i]->GetEntries());

         stack->Add(histogram[i]);
      }


   // Canvas erzeugen
      TCanvas *canvas = new TCanvas(output_filename, output_filename, 1200, 800);


   // Legende erstellen
      TLegend *legend = new TLegend(0.2, 0.735, 0.4, 0.935);
      legend->SetFillColor(0);

      for(int i=0; i<4; i=i+1 ){
         legend->AddEntry(histogram[i], dataset[i], "l");
      }


```

```
   // Draw
      stack->Draw("nostack");
      legend->Draw();


   // Style
      stack->SetTitle(histogram_title);                  // Titel
      stack->GetXaxis()->SetTitle(histogram_x_axis);     // x-Achse
      stack->GetYaxis()->SetTitle(histogram_y_axis);     // y-Achse
      gPad->Modified();


   // Canvas speichern
      canvas->Print("print/"+output_filename+".root", "root");
      canvas->Print("print/"+output_filename+".pdf", "pdf Portrait");
      canvas->Print("print/"+output_filename+".png", "png");


   // Ende
      delete canvas;
      for(int i=0; i<4; i=i+1){
         input_file[i]->Close();
      }
}
```

```
// cut.C
// Schnittfunktion


// Bibliotheken
   #include "TString.h"
   #include "TFile.h"
   #include "TTree.h"
   #include <iostream>
   #include <vector>


// namespace
   using namespace std;


vector<float> cut(TString filename, TString branch, TString cut_cond){

   // output
      vector<float> output;
      output.clear();


   // Daten einlesen
      TString data = "daten/"+filename+".root";
      TFile *file = new TFile(data);
      TTree *tree = (TTree*)file->Get(branch);


   // cut
      float output_cut[3];
      output_cut[0] = tree->GetEntries();
      output_cut[1] = tree->GetEntries(cut_cond);
      output_cut[2] = output_cut[1]/output_cut[0];

      for(int i=0; i<3; i=i+1){
         output.push_back(output_cut[i]);
      }


   // Ende
      file->Close();


   return(output);
}
```

```
// fit_s_t.C
// Fitfunktion für s-t-Kanaltrennung


// Bibliotheken
   #include "TFile.h"
   #include "TTree.h"
   #include "TString.h"
   #include "TH1F.h"
   #include "TROOT.h"
   #include "TGraph.h"
   #include "TF1.h"
   #include "TStyle.h"
   #include "TGraphErrors.h"
   #include "TCanvas.h"
   #include <iostream>
   #include <vector>


// namespace
   using namespace std;


vector<float> fit_s_t(
   TString filename, float l, float r, float o, float u
){

   // Styles
      gROOT->Reset();
      gROOT->SetStyle("Modern");


   // input
      TString function = "[0]*(1+x^2)+[1]*(1/(1-x)^2)";


   // output
      vector<float> output;
      output.clear();


   // Styles
      gStyle->SetOptStat(11);
      gStyle->SetOptFit(1);


   // Daten einlesen
      TString file = "print/"+filename+".root";
      TFile *input = new TFile(file);


   // Canvas erzeugen
      TCanvas *canvas = new TCanvas(filename, filename, 1200, 800);
      TH1F *histogram = (TH1F*)input->Get(filename);

      TF1 *fit = new TF1("fit", function, l, r);
      TF1 *fit_s = new TF1("fit_s", "[0]*(1+x^2)", l, r);
      TF1 *fit_t = new TF1("fit_t", "[0]*(1/(1-x)^2)", l, r);


   // Legende erstellen
      TLegend *legend = new TLegend(0.2, 0.735, 0.4, 0.935);
      legend->SetFillColor(0);

      legend->AddEntry(fit, "s- and  t-channel", "l");
      legend->AddEntry(fit_s, "s-channel", "l");
      legend->AddEntry(fit_t, "t-channel", "l");


   // Styles festlegen
      histogram->SetMarkerStyle(5);

      fit->SetLineColor(kRed);

```

```
      fit_s->SetLineColor(4);
      fit_s->SetLineWidth(1);

      fit_t->SetLineColor(3);
      fit_t->SetLineWidth(1);

   // Y-Achsenbereich vergrößern
      histogram->SetMaximum(histogram->GetBinContent(histogram->GetMaximumBin())*1.6);


   // Draw
      histogram->Fit("fit", "", "", l, r);
      histogram->Draw("E1");

      // s- und t-Kanal
         fit_s->SetParameter(0,fit->GetParameter(0));
         fit_t->SetParameter(0,fit->GetParameter(1));

         fit_s->Draw("same");
         fit_t->Draw("same");

      // Legende
         legend->Draw("same");


   // Berechnung Integralverhältnisse
      float p[2];
      p[0]=fit->GetParameter(0);
      p[1]=fit->GetParameter(1);

      float s_p[2];
      s_p[0]=fit->GetParError(0);
      s_p[1]=fit->GetParError(1);

      float integral[2];
      integral[0] = o+o*o*o/3-(u+u*u*u/3); // s-Integral
      integral[1] = -1/(o-1)-(-1/(u-1)); // t-Integral


   // output vorbereiten
      float output_fit[11];
      output_fit[0] = fit->GetChisquare()/fit->GetNDF(); // chi^2_reduziert
      output_fit[1] = p[0]; // s
      output_fit[2] = p[1]; // t
      output_fit[3] = p[0]/(p[0]+p[1]*integral[1]/integral[0]); // Korrekturfaktor
      output_fit[4] = s_p[0]; // s_s
      output_fit[5] = s_p[1]; // s_t
      output_fit[6] = integral[1]/integral[0]/((p[0]+p[1]*integral[1]/integral[0])*(p[0]+p[1]*integral[1]/integral[0]))*sqrt(p[1]*p[1]*s_p[0]*s_p[0]+p[0]*p[0]*s_p[1]*s_p[1]); // s_Korrekturfaktor
      output_fit[7] = histogram->GetEntries(); // N
      output_fit[8] = p[0]*integral[0]/histogram->GetXaxis()->GetBinWidth(1); // N_s
      output_fit[9] = p[1]*integral[1]/histogram->GetXaxis()->GetBinWidth(1); // N_t
      output_fit[10] = output_fit[8]+output_fit[9]; // N_s+N_t


   // Canvas speichern
      canvas->Print("print/"+filename+".pdf", "pdf Portrait");
      canvas->Print("print/"+filename+".png", "png");


   // close
      delete canvas;
      input->Close();


   // output schreiben
      for(int i=0; i<11; i=i+1){
         output.push_back(output_fit[i]);
      }


   // Ausgabe
      cout << endl;
```

```
    cout << filename << endl;
    cout << "chi^2_reduziert: " << output.at(0) << endl;
    cout << "s: " << output.at(1) << " +- " << output.at(4) << endl;
    cout << "t: " << output.at(2) << " +- " << output.at(5) << endl;
    cout << "Korrekturfaktor: " << output.at(3) << " +- " << output.at(6) << endl;
    cout << "N_s: " << output.at(7) << endl;
    cout << "N_t: " << output.at(8) << endl;
    cout << endl;
    cout << endl;

    return(output);
}
```

```
// theo_s_t.C
// theoretischer s-t-Kanal-Verlauf


// Bibliotheken
   #include "TString.h"
   #include "TROOT.h"
   #include "TH1F.h"
   #include "TF1.h"
   #include "TCanvas.h"
   #include "TGraph.h"
   #include <iostream>


// namespace
   using namespace std;


void theo_s_t(){

   // Styles
      gROOT->Reset();
      gROOT->SetStyle("Modern");


   // Literaturverlauf s-t-Kanaltrennung
      TString filename = "s-t-Kanaltrennung";
      float l = -0.8;
      float r = -l;

      TCanvas *canvas = new TCanvas(filename, filename, 1200, 800);

      TF1 *lit = new TF1("lit", "[0]*(1+x^2)+[1]*(1/(1-x)^2)", l, r);
      lit->SetParameter(0, 1);
      lit->SetParameter(1, 1);
      lit->SetLineColor(kRed);

      TF1 *lit_s = new TF1("lit_a", "[0]*(1+x^2)", l, r);
      lit_s->SetParameter(0, 1);
      lit_s->SetLineColor(4);
      lit_s->SetLineWidth(1);

      TF1 *lit_t = new TF1("lit_b", "[0]*(1/(1-x)^2)", l, r);
      lit_t->SetParameter(0, 1);
      lit_t->SetLineColor(3);
      lit_t->SetLineWidth(1);

      lit->Draw("");
      lit_s->Draw("same");
      lit_t->Draw("same");

      lit->SetTitle("s-t-progression");
      lit->GetHistogram()->GetXaxis()->SetTitle("cos #Theta");
      lit->GetHistogram()->GetYaxis()->SetTitle("events");

      // Legende erstellen
         TLegend *legend = new TLegend(0.2, 0.735, 0.4, 0.935);
         legend->SetFillColor(0);

         legend->AddEntry(lit, "s- and t-channel", "l");
         legend->AddEntry(lit_s, "s-channel", "l");
         legend->AddEntry(lit_t, "t-channel", "l");

         legend->Draw("same");

   // Canvas speichern
      canvas->Print("print/theo_s-t.pdf", "pdf Portrait");
      canvas->Print("print/theo_s-t.png", "png");

   // Ende
      delete canvas;
}
```

```
// fit_bw.C
// Funktion für Breit-Wigner-Fits


// Bibliotheken
   #include "TFile.h"
   #include "TTree.h"
   #include "TString.h"
   #include "TH1F.h"
   #include "TROOT.h"
   #include "TGraph.h"
   #include "TF1.h"
   #include "TStyle.h"
   #include "TGraphErrors.h"
   #include "TCanvas.h"
   #include "TGraphErrors.h"
   #include <iostream>
   #include <vector>


// namespace
   using namespace std;


vector<float> fit_bw(
   TString input_file,
   int dataset,
   TMatrixT<float>input,
   TMatrixT<float>input_error,
   TString input_function,
   float l,
   float r,
   TString histogram_title
){

   // Styles
      gROOT->Reset();
      gROOT->SetStyle("Modern");
      gStyle->SetOptStat(11);
      gStyle->SetOptFit(1);


   // output
      TString output_file = "print/"+input_file+".root";
      TFile* file = new TFile(output_file,"RECREATE");


   // Canvas erzeugen
      TCanvas *canvas = new TCanvas("Canvas","Canvas",1200,800);


   // Daten einlesen
      TGraphErrors* error = new TGraphErrors(7);

      float x[7]={88.47, 89.46, 90.22, 91.22, 91.97, 92.96, 93.71};
      float y[7]={input(dataset,0), input(dataset,1), input(dataset,2), input(dataset,3), input
(dataset,4), input(dataset,5), input(dataset,6)};
      float yerror[7]={input_error(dataset,0), input_error(dataset,1), input_error(dataset,2),
input_error(dataset,3), input_error(dataset,4), input_error(dataset,5), input_error(dataset,6)};


   // Fehlerbalken
      for( int i=0; i<7; i=i+1){
         error->SetPoint(i, x[i], y[i]);
         error->SetPointError(i,0,yerror[i]);
      }

      // Styles festlegen
         error->SetMarkerStyle(5);

         canvas->Modified();
         canvas->cd();
         canvas->SetSelected(canvas);

```

```
    // Fit
       TF1 *fit = new TF1("fit", input_function, l, r);
       fit->SetParameters(0.084,91.18,2.5);

       // Styles festlegen
       fit->SetLineColor(kRed);
       error->SetTitle(histogram_title);            // Titel
       error->GetXaxis()->SetTitle("E [GeV]");      // x-Achse
       error->GetYaxis()->SetTitle("#sigma [nb]");  // y-Achse


    // Draw
       error->Fit("fit","","",l,r);
       error->Draw("ap");


    // output vorbereiten
       float p[3];
       p[0]=fit->GetParameter(0);
       p[1]=fit->GetParameter(1);
       p[2]=fit->GetParameter(2);

       float s_p[3];
       s_p[0]=fit->GetParError(0);
       s_p[1]=fit->GetParError(1);
       s_p[2]=fit->GetParError(2);

       float output_fit[7];
       output_fit[0] = fit->GetChisquare()/fit->GetNDF(); // chi^2_reduziert
       output_fit[1] = p[0];
       output_fit[2] = p[1];
       output_fit[3] = p[2];
       output_fit[4] = s_p[0];
       output_fit[5] = s_p[1];
       output_fit[6] = s_p[2];


    // close
       file->cd();
       error->SetName(input_file);
       error->Write();
       canvas->Print("print/"+input_file+".png");
       canvas->Print("print/"+input_file+".pdf");
       canvas->Close();
       file->Close();

    // output schreiben
       vector<float> output;
       output.clear();

       for(int i=0; i<7; i=i+1){
          output.push_back(output_fit[i]);
       }

       return(output);
  }
```

```
// fit_asym.C
// Fitfunktion für Vorwärts-Rückwärts-Asymmetrie


// Bibliotheken
   #include "TFile.h"
   #include "TTree.h"
   #include "TString.h"
   #include "TH1F.h"
   #include "TROOT.h"
   #include "TGraph.h"
   #include "TF1.h"
   #include "TStyle.h"
   #include "TGraphErrors.h"
   #include "TCanvas.h"
   #include <iostream>
   #include <vector>


// namespace
   using namespace std;

vector<float> fit_asym(
   TString filename, float l, float r
){

   // Styles
      gROOT->Reset();
      gROOT->SetStyle("Modern");
      gStyle->SetOptStat(11);
      gStyle->SetOptFit(1);


   // input
      TString function = "[0]*(1+x^2)+2*[1]*x";


   // output
      vector<float> output;
      output.clear();


   // Daten einlesen
      TString file = "print/"+filename+".root";
      TFile *input = new TFile(file);


   // Canvas erzeugen
      TCanvas *canvas = new TCanvas(filename, filename, 1200, 800);
      TH1F *histogram = (TH1F*)input->Get(filename);
      TF1 *fit = new TF1("fit", function, l, r);


   // Styles festlegen
      histogram->SetLineColor(1);
      histogram->SetMarkerStyle(5);

      // Y-Achsenbereich vergrößern
         histogram->SetMaximum(histogram->GetBinContent(histogram->GetMaximumBin())*1.6);

      fit->SetLineColor(kRed);


   // Fit
      histogram->Fit("fit", "", "", l, r);
      histogram->Draw("E1");


   // output vorbereiten
      float p[2];
      p[0]=fit->GetParameter(0);
      p[1]=fit->GetParameter(1);

      float s_p[2];
```

```
      s_p[0]=fit->GetParError(0);
      s_p[1]=fit->GetParError(1);

      float output_fit[9];
      output_fit[0] = fit->GetChisquare()/fit->GetNDF();             // chi^2_reduziert
      output_fit[1] = p[0];                                          // p[0]
      output_fit[2] = p[1];                                          // p[1]
      output_fit[3] = s_p[0];                                        // s_p[0]
      output_fit[4] = s_p[1];                                        // s_p[1]
      float tmp = 1;                                                 // --> float
      output_fit[5] = tmp*3/4*fabs(p[1])/p[0];                       // A_FB
      output_fit[6] = tmp*3/4*fabs(p[1])/p[0]*sqrt(s_p[0]/p[0]*s_p[0]/p[0]+s_p[1]/fabs(p[1])*s_p[1]/fabs(p[1]));  // s_A_FB
      output_fit[7] = tmp*1/4-tmp*1/4*sqrt(output_fit[5]/3);         // Weinberg-Winkel
      output_fit[8] = output_fit[6]/(8*tmp*sqrt(3*output_fit[5]));   // s_Weinberg-Winkel


   // Canvas speichern
      canvas->Print("print/"+filename+".pdf", "pdf Portrait");
      canvas->Print("print/"+filename+".png", "png");


   // close
      delete canvas;
      input->Close();


   // output schreiben
      for(int i=0; i<9; i=i+1){
         output.push_back(output_fit[i]);
      }


   // Ausgabe
      cout << endl;
      cout << filename << endl;
      cout << "chi^2_reduziert: " << output.at(0) << endl;
      cout << "p[0]: " << output.at(1) << " +- " << output.at(3) << endl;
      cout << "p[1]: " << output.at(2) << " +- " << output.at(4) << endl;
      cout << "A_FB: " << output.at(5) << " +- " << output.at(6) << endl;
      cout << "Weinberg-Winkel: " << output.at(7) << " +- " << output.at(8) << endl;
      cout << endl;
      cout << endl;

      return(output);
}
```

C The readout of the root script

On the following pages we figure the the readout of the root script. It includes all the matrices we figured in the analysis.

```
  *******************************************
  *                                         *
  *        W E L C O M E  to  R O O T       *
  *                                         *
  *   Version   5.34/00       5 June 2012   *
  *                                         *
  *  You are welcome to visit our Web site  *
  *          http://root.cern.ch            *
  *                                         *
  *******************************************

ROOT 5.34/00 (branches/v5-34-00-patches@44555, Mar 14 2013, 11:26:00 on linuxx8664gcc)

CINT/ROOT C/C++ Interpreter version 5.18.00, July 2, 2010
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
root [0] .x run.C++
Info in <TUnixSystem::ACLiC>: creating shared library /home/steffen/ownCloud/FP/Z0/Auswertung/./run_C.so

Anzahl simulierter Ereignisse ohne cut:

4x4 matrix is as follows

     |      0    |      1    |      2    |      3    |
---------------------------------------------------------
   0 |   9.38e+04   9.438e+04   7.921e+04   9.856e+04
   1 |   9.38e+04   9.438e+04   7.921e+04   9.856e+04
   2 |   9.38e+04   9.438e+04   7.921e+04   9.856e+04
   3 |   9.38e+04   9.438e+04   7.921e+04   9.856e+04



Anzahl simulierter Ereignisse mit cut:

4x4 matrix is as follows

     |      0    |      1    |      2    |      3    |
---------------------------------------------------------
   0 |  4.442e+04           1          59           1
   1 |          0   8.688e+04         874           0
   2 |       1166        6759   7.258e+04         200
   3 |          6           0         544   9.755e+04



Effizienzmatrix:

4x4 matrix is as follows

     |      0    |      1    |      2    |      3    |
---------------------------------------------------------
   0 |     0.4735    1.06e-05   0.0007448   1.015e-05
   1 |          0      0.9206     0.01103           0
   2 |    0.01243     0.07161      0.9162    0.002029
   3 |  6.396e-05           0    0.006867      0.9897



Fehler auf Effizienzmatrix:

4x4 matrix is as follows

     |      0    |      1    |      2    |      3    |
---------------------------------------------------------
   0 |    0.00163   1.063e-05   8.908e-05    1.04e-05
   1 |          0   0.0008829   0.0003411           0
   2 |  0.0003618   0.0008419   0.0009046   0.0001469
   3 |  2.611e-05           0   0.0002696   0.0003296



invertierte Effizienzmatrix:

```

```
4x4 matrix is as follows

     |      0    |      1    |      2    |      3    |
------------------------------------------------------------
   0 |      2.112   0.0001093   -0.001718  -1.813e-05
   1 |  0.0003437       1.087    -0.01309   2.684e-05
   2 |   -0.02868    -0.08499       1.093    -0.00224
   3 |  6.251e-05   0.0005897   -0.007581        1.01



Fehler auf invertierte Effizienzmatrix:

4x4 matrix is as follows

     |      0    |      1    |      2    |      3    |
------------------------------------------------------------
   0 |   0.007264   7.468e-06   0.0001975   2.145e-05
   1 |  1.879e-05    0.001001   0.0003798    2.71e-06
   2 |  0.0007087    0.000837    0.001032    0.000159
   3 |  4.254e-05   2.876e-05   0.0002879   0.0003348



prozentualer Fehler auf invertiere Effizienzmatrix:

4x4 matrix is as follows

     |      0    |      1    |      2    |      3    |
------------------------------------------------------------
   0 |      0.344        6.83        11.5       118.3
   1 |      5.466     0.09209       2.901        10.1
   2 |      2.471      0.9849     0.09444         7.1
   3 |      68.05       4.878       3.798     0.03313



Anzahl realer Daten ohne cut:

4x7 matrix is as follows

     |      0    |      1    |      2    |      3    |      4    |
----------------------------------------------------------------------
   0 |  1.759e+05   1.759e+05   1.759e+05   1.759e+05   1.759e+05
   1 |  1.759e+05   1.759e+05   1.759e+05   1.759e+05   1.759e+05
   2 |  1.759e+05   1.759e+05   1.759e+05   1.759e+05   1.759e+05
   3 |  1.759e+05   1.759e+05   1.759e+05   1.759e+05   1.759e+05


     |      5    |      6    |
----------------------------------------------------------------------
   0 |  1.759e+05   1.759e+05
   1 |  1.759e+05   1.759e+05
   2 |  1.759e+05   1.759e+05
   3 |  1.759e+05   1.759e+05



Anzahl realer Daten mit cut:

4x7 matrix is as follows

     |      0    |      1    |      2    |      3    |      4    |
----------------------------------------------------------------------
   0 |        676         660         587        5033         788
   1 |        139         257         349        4036         709
   2 |        221         262         303        4101         674
   3 |       3528        5322        7542   9.277e+04   1.529e+04


     |      5    |      6    |
----------------------------------------------------------------------
   0 |        384         543
   1 |        281         338
```

```
   2 |        333        345
   3 |       6644       7421



Fehler auf Anzahl realer Daten mit cut:

4x7 matrix is as follows

     |      0    |      1    |      2    |      3    |      4    |
----------------------------------------------------------------------
   0 |         26      25.69      24.23      70.94      28.07
   1 |      11.79      16.03      18.68      63.53      26.63
   2 |      14.87      16.19      17.41      64.04      25.96
   3 |       59.4      72.95      86.84      304.6      123.7


     |      5    |      6    |
----------------------------------------------------------------------
   0 |       19.6       23.3
   1 |      16.76      18.38
   2 |      18.25      18.57
   3 |      81.51      86.15



 FCN=70.8205 FROM MIGRAD    STATUS=CONVERGED      30 CALLS          31 TOTAL
                     EDM=1.53966e-22    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           7.10226e-01   1.38399e-01   5.26451e-04  -1.34968e-10
   2  p1           7.74140e-01   4.71835e-02   1.79480e-04  -7.91778e-11

s-t-channel_e0
chi^2_reduziert: 0.944274
s: 0.710226 +- 0.138399
t: 0.77414 +- 0.0471835
Korrekturfaktor: 0.104232 +- 0.0190633
N_s: 676
N_t: 87.7692


 FCN=77.4605 FROM MIGRAD    STATUS=CONVERGED      30 CALLS          31 TOTAL
                     EDM=7.56989e-23    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           1.32942e+00   1.56063e-01   6.14180e-04   5.78448e-11
   2  p1           5.80443e-01   4.34242e-02   1.70894e-04  -1.24734e-10

s-t-channel_e1
chi^2_reduziert: 0.922148
s: 1.32942 +- 0.156063
t: 0.580443 +- 0.0434242
Korrekturfaktor: 0.225099 +- 0.0242813
N_s: 660
N_t: 164.288


 FCN=80.1874 FROM MIGRAD    STATUS=CONVERGED      30 CALLS          31 TOTAL
                     EDM=1.14086e-22    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           1.75107e+00   1.64674e-01   6.54638e-04  -5.42699e-11
   2  p1           4.21582e-01   3.95318e-02   1.57153e-04   2.26068e-10

s-t-channel_e2
chi^2_reduziert: 0.921695
s: 1.75107 +- 0.164674
t: 0.421582 +- 0.0395318
Korrekturfaktor: 0.345036 +- 0.0300118
N_s: 587
N_t: 216.396
```

```
 FCN=235.184 FROM MIGRAD    STATUS=CONVERGED      30 CALLS          31 TOTAL
                     EDM=5.28014e-22    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           2.54670e+01   5.76193e-01   3.89552e-03   3.64800e-12
   2  p1           2.25619e+00   1.05740e-01   7.14887e-04  -2.98177e-10

s-t-channel_e3
chi^2_reduziert: 2.67255
s: 25.467 +- 0.576193
t: 2.25619 +- 0.10574
Korrekturfaktor: 0.588752 +- 0.0126006
N_s: 5033
N_t: 3147.19


 FCN=115.782 FROM MIGRAD    STATUS=CONVERGED      30 CALLS          31 TOTAL
                     EDM=3.12425e-23    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           2.92845e+00   2.04236e-01   9.61868e-04   3.69356e-11
   2  p1           3.99342e-01   4.23011e-02   1.99221e-04   1.78330e-10

s-t-channel_e4
chi^2_reduziert: 1.3157
s: 2.92845 +- 0.204236
t: 0.399342 +- 0.0423011
Korrekturfaktor: 0.481886 +- 0.0316645
N_s: 788
N_t: 361.895


 FCN=58.1578 FROM MIGRAD    STATUS=CONVERGED      30 CALLS          31 TOTAL
                     EDM=3.01643e-22    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           1.12934e+00   1.44229e-01   4.83620e-04   3.67304e-11
   2  p1           2.50486e-01   3.11941e-02   1.04598e-04   8.49138e-10

s-t-channel_e5
chi^2_reduziert: 0.765234
s: 1.12934 +- 0.144229
t: 0.250486 +- 0.0311941
Korrekturfaktor: 0.363799 +- 0.0412857
N_s: 384
N_t: 139.562


 FCN=80.6558 FROM MIGRAD    STATUS=CONVERGED      30 CALLS          31 TOTAL
                     EDM=1.29978e-22    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           1.50128e+00   1.57644e-01   6.18945e-04  -1.14799e-10
   2  p1           3.14608e-01   3.56997e-02   1.40164e-04  -2.53468e-10

s-t-channel_e6
chi^2_reduziert: 0.971756
s: 1.50128 +- 0.157644
t: 0.314608 +- 0.0356997
Korrekturfaktor: 0.377036 +- 0.0363135
N_s: 543
N_t: 185.527


s-t-Korrekturfaktor:

1x7 matrix is as follows

     |      0    |      1    |      2    |      3    |      4    |
----------------------------------------------------------------------
   0 |     0.1042      0.2251       0.345      0.5888      0.4819


     |      5    |      6    |
```

0	0.3638	0.377

Fehler auf s-t-Korrekturfaktor:

2x7 matrix is as follows

	0	1	2	3	4
0	0.01906	0.02428	0.03001	0.0126	0.03166
1	0.02173	0.03311	0.04573	0.06021	0.05766

	5	6
0	0.04129	0.03631
1	0.05503	0.05235

Anzahl realer Daten mit cut und s-t-Trennung:

4x7 matrix is as follows

	0	1	2	3	4
0	70.46	148.6	202.5	2963	379.7
1	139	257	349	4036	709
2	221	262	303	4101	674
3	3528	5322	7542	9.277e+04	1.529e+04

	5	6
0	139.7	204.7
1	281	338
2	333	345
3	6644	7421

Fehler auf Anzahl realer Daten mit cut und s-t-Trennung:

4x7 matrix is as follows

	0	1	2	3	4
0	14.94	22.6	28.11	305.9	47.41
1	11.79	16.03	18.68	63.53	26.63
2	14.87	16.19	17.41	64.04	25.96
3	59.4	72.95	86.84	304.6	123.7

	5	6
0	22.3	29.75
1	16.76	18.38
2	18.25	18.57
3	81.51	86.15

Ereignismatrix:

4x7 matrix is as follows

	0	1	2	3	4
0	148.4	313.2	427.1	6249	800.5
1	148.4	276.2	375.8	4338	762.6
2	219.7	248.2	278.7	3845	631
3	3563	5376	7619	9.37e+04	1.545e+04

	5	6
0	294.4	431.7
1	301.4	363.3
2	321	325.7
3	6711	7496

Fehler auf Ereignismatrix:

4x7 matrix is as follows

	0	1	2	3	4
0	31.54	47.75	59.39	646.3	100.2
1	12.82	17.43	20.32	69.22	28.96
2	16.29	17.77	19.14	72.48	28.61
3	60.03	73.73	87.79	309.3	125.1

	5	6
0	47.11	62.85
1	18.23	19.99
2	20.03	20.41
3	82.39	87.08

prozentualer Fehler auf Ereignismatrix:

4x7 matrix is as follows

	0	1	2	3	4
0	21.26	15.24	13.91	10.34	12.51
1	8.642	6.312	5.407	1.596	3.798
2	7.414	7.16	6.87	1.885	4.534
3	1.685	1.372	1.152	0.3301	0.8095

	5	6
0	16	14.56
1	6.049	5.504
2	6.239	6.267
3	1.228	1.162

Wirkungsquerschnitt:

4x7 matrix is as follows

	0	1	2	3	4
0	0.3095	0.7762	1.377	2.522	1.471
1	0.3095	0.7081	1.255	1.909	1.412
2	0.4151	0.6566	1.024	1.751	1.206
3	7.272	14.19	25.85	40.81	28.84

	5	6
0	0.6042	0.4829
1	0.6189	0.3937
2	0.6599	0.3447
3	13.8	8.175

```
Fehler auf Wirkungsquerschnitt:

4x7 matrix is as follows

     |      0    |      1    |      2    |      3    |      4    |
----------------------------------------------------------------------
   0 |    0.04671     0.08798      0.1418      0.2075      0.1569
   1 |    0.01906     0.03239     0.04914     0.02429     0.04644
   2 |    0.02426     0.03294     0.04604     0.02483     0.04553
   3 |     0.0994      0.1616      0.2707      0.2363      0.2872


     |      5    |      6    |
----------------------------------------------------------------------
   0 |    0.09846     0.08209
   1 |    0.03849     0.02638
   2 |    0.04226     0.02686
   3 |     0.2161      0.1406



prozentualer Fehler auf Wirkungsquerschnitt:

4x7 matrix is as follows

     |      0    |      1    |      2    |      3    |      4    |
----------------------------------------------------------------------
   0 |      15.09       11.34        10.3       8.229       10.67
   1 |      6.159       4.574       3.915       1.272       3.289
   2 |      5.845       5.017       4.496       1.418       3.775
   3 |      1.367       1.139       1.047       0.579      0.9958


     |      5    |      6    |
----------------------------------------------------------------------
   0 |       16.3          17
   1 |      6.219       6.701
   2 |      6.405       7.791
   3 |      1.566       1.719



 FCN=3.22768 FROM MIGRAD    STATUS=CONVERGED      76 CALLS          77 TOTAL
                     EDM=3.4979e-07    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           8.10488e-02   4.02421e-03   1.78252e-06  -3.34256e-01
   2  p1           9.11441e+01   6.22052e-02   5.95137e-05  -3.70176e-03
   3  p2           2.15258e+00   1.64276e-01   7.31168e-05   4.59829e-03

 FCN=5.11477 FROM MIGRAD    STATUS=CONVERGED      68 CALLS          69 TOTAL
                     EDM=4.73044e-07    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           8.70468e-02   3.16877e-03   1.10912e-06  -3.88365e-01
   2  p1           9.11815e+01   2.75205e-02   4.34787e-05  -1.87912e-02
   3  p2           2.52852e+00   5.22458e-02   1.81864e-05   1.01831e-02

 FCN=24.4682 FROM MIGRAD    STATUS=CONVERGED      83 CALLS          84 TOTAL
                     EDM=2.13193e-08    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           8.72593e-02   3.82824e-03   2.52591e-06  -9.28664e-03
   2  p1           9.11774e+01   2.92923e-02   7.09383e-05  -4.04634e-03
   3  p2           2.66650e+00   6.67227e-02   4.38968e-05   3.38917e-03

 FCN=3.34598 FROM MIGRAD    STATUS=CONVERGED     215 CALLS         216 TOTAL
                     EDM=7.60351e-09    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           1.84423e+00   1.81651e-02   7.11799e-06  -2.23254e-03
   2  p1           9.11824e+01   7.17828e-03   4.34791e-05   1.06805e-02
   3  p2          -2.52690e+00   1.63014e-02   6.41293e-06  -7.07491e-03
```

```
e

chi^2_reduziert: 0.806919
M_z: 91.1441 +- 0.0622052
G_z: 2.15258 +- 0.164276
G_e,m,t,q: 0.0810488 +- 0.00402421


m

chi^2_reduziert: 1.27869
M_z: 91.1815 +- 0.0275205
G_z: 2.52852 +- 0.0522458
G_e,m,t,q: 0.0870468 +- 0.00316877


t

chi^2_reduziert: 6.11704
M_z: 91.1774 +- 0.0292923
G_z: 2.6665 +- 0.0667227
G_e,m,t,q: 0.0872593 +- 0.00382824


q

chi^2_reduziert: 0.836494
M_z: 91.1824 +- 0.00717828
G_z: -2.5269 +- 0.0163014
G_e,m,t,q: 1.84423 +- 0.0181651


E0
N = 676
N_s+N_t = 842.059

E1
N = 660
N_s+N_t = 729.848

E2
N = 587
N_s+N_t = 627.167

E3
N = 5033
N_s+N_t = 5345.53

E4
N = 788
N_s+N_t = 750.997

E5
N = 384
N_s+N_t = 383.625

E6
N = 543
N_s+N_t = 492.069


 FCN=37.0818 FROM MIGRAD    STATUS=CONVERGED      32 CALLS          33 TOTAL
                     EDM=5.53677e-08    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           1.58037e+03   5.80819e+00   1.75009e-02  -4.06004e-12
   2  p1           1.44530e+01   7.88535e+00   2.37596e-02  -4.22010e-05

asym_mm
chi^2_reduziert: 1.03005
p[0]: 1580.37 +- 5.80819
p[1]: 14.453 +- 7.88535
A_FB: 0.00685899 +- 0.00374225
```

```
Weinberg-Winkel: 0.238046 +- 0.00326101


 FCN=29.3271 FROM MIGRAD    STATUS=CONVERGED      28 CALLS          29 TOTAL
                     EDM=2.52541e-13    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           6.83438e+01   1.20784e+00   3.24785e-03   8.20400e-12
   2  p1          -1.35461e-01   1.63496e+00   4.39634e-03   4.34685e-07

asym_E3
chi^2_reduziert: 0.814642
p[0]: 68.3438 +- 1.20784
p[1]: -0.135461 +- 1.63496
A_FB: 0.00148654 +- 0.0179419
Weinberg-Winkel: 0.244435 +- 0.0335838
```

D Sources

- Physical Review D: Particles and Fields. Part I. Review of particle physics. Publishes by the American Physical Society. Editors: E. J. Weinberg, D. L. Nordstrom. New York. 2002
- Versuchsanleitung Fortgeschrittenen Praktikum Teil II: Z^0-Resonanz. Institut für Mathematik und Physik, Albert Ludwigs Universität Freiburg im Breisgau. 2. März 2012
- Analyse von Z^0-Zerfällen. Universität Freiburg, Fortgeschrittenenprakikum Teil 2. 9. Februar 1995
- Experimentalphysik 4: Kern-, Teilchen- und Astrophysik. W. Demtröder. 3. Auflage. Springer Verlag Berlin-Heidelberg. 2010
- Elementare Teilchen: Von den Atomen über das Standard-Modell bis zum Higgs-Boson. J. Bleck-Neuhaus. 2. Auflage. Springer Verlag Berlin-Heidelberg. 2013

E Picture sources

1 (page 4) http://bit.ly/1ixZdAl (20.03.2014)

2 (page 7) http://www-zeus.physik.uni-bonn.de/∼brock/feynman/vtp_ws0506/chapter01/bhabha.jpg (20.03.2014)

3 (page 11) http://opal.web.cern.ch/Opal/tour/detector.html (20.03.2014)

4 (page 12) http://opal.web.cern.ch/Opal/tour/layers.html (21.03.2014)

The pictures 5, 6, 7 and 8 are screenshots of the program GROPE.

All the other pictures are plots we made with root.

Abbildungsverzeichnis