

Alignment of a Muon Tomography Station with GEM detectors

by

Afrouz Ataei

Bachelor of Science
in Physics
Isfahan University of Technology
2013

A thesis
submitted to the College of Science at
Florida Institute of Technology
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Physics

Melbourne, Florida
July 2016

© Copyright 2016 Afrouz Ataei
All Rights Reserved

The author grants permission to make single copies _____

We the undersigned committee hereby recommend
that the attached document be accepted as fulfilling in
part the requirements for the degree of
Master of Science in Physics.

“Alignment of a Muon Tomography Station with GEM Detectors”,
a thesis by Afrouz Ataei

Marcus Hohlmann, Ph.D.
Professor, Physics and Space Sciences
Dissertation Advisor

Francisco Yumiceva, Ph.D
Associate Professor, Physics and Space Sciences

Debasis Mitra, Ph.D.
Professor, Computer Science

Daniel. Batcheldor, Ph.D
Professor and Head, Physics and Space Sciences

Abstract

Title: Alignment of a Muon Tomography Station with GEM
Detectors
Author: Afrouz Ataei
Major Advisor: Marcus Hohlmann, Ph.D.

Muon Tomography is a technique for detecting nuclear material in road transport vehicles and cargo containers. It is an imaging technique that uses muons to detect nuclear materials such as uranium. This thesis starts with a review of gas detectors and triple-GEM design and then covers the alignment of the detectors, the main part of this work. The alignment of GEM detectors by using selected tracks are presented which is based on calculating residual distributions.

Contents

Abstract	iii
List of Figures	viii
List of Tables	ix
Abbreviations	x
Acknowledgments	xi
1 Introduction	1
2 The Gas Electron Multiplier Detector	4
2.1 The Gas Electron Multiplier	4
2.1.1 Gas Multiplication	4
2.1.2 Detector Gas	5
2.1.3 The Gas Electron Multiplier	6
2.2 Design of GEM Detector	8
2.2.1 The Foils	8
2.2.2 The full Detector	9
3 Data Acquisition	12
3.1 Muon Tomography Station	12
3.2 Electronics	12
3.2.1 SRS Hardware	12
3.2.2 The APV25	13
3.2.3 Analog-Digital Converter Card	15
3.2.4 Front-End Concentrator Card	15

3.3	Online Analysis	16
3.3.1	Trigger	16
3.3.2	Data Acquisition	17
3.3.3	Pedestal	18
3.3.4	Troubleshooting	19
3.3.5	Online/Offline Processing	22
4	Alignment of GEM Detectors	27
4.1	The Reason for Aligning	27
4.2	The Coordinate System	29
4.3	Track Selection	32
4.4	The Alignment	33
4.4.1	Local Alignment	33
4.4.2	Cross check using track angles	53
4.4.3	Global Alignment	59
5	Summary and Conclusions	61
5.1	Summary	61
5.2	Future Work	61
A	Directory of Data Collection Used for Alignment	67
B	Location of APVs Based on the Local Coordinate	68
C	Script Used for Alignment	73

List of Figures

2.1	GEM foil under electron microscope [6].	7
2.2	Electric field in GEM foil [7].	7
2.3	The triple-GEM detector and the amplification concept [8].	8
2.4	Stretched GEM foil [26].	9

2.5	2-D readout structure [7].	9
2.6	Schematic view of a triple GEM [12].	10
2.7	30 cm x 30 cm triple-GEM detector used in Florida Tech muon tomography station.	11
3.1	The SRS architecture [18].	13
3.2	The SRS Components [17].	14
3.3	Two APV Hybrids [18].	14
3.4	Raw APV output.	15
3.5	SRS Analog-Digital Converter card [20].	16
3.6	SRS Analog-Digital Converter card [20].	16
3.7	Sample of trigger rates from Muonic.	17
3.8	Description of the AMORE module [15].	19
3.9	The publisher model in AMORE [15].	19
3.10	Raw APV output.	20
3.11	Sample of pedestal offset and width of pedestal distribution for one APV.	20
3.12	Problematic raw APV output signal.	21
3.13	Sample of effect of problematic APV on pedestal offset and noise and Hit maps.	21
3.14	Raw APV output with the GEM signal.	23
3.15	The appropriate time bin for storing the data by the FECs.	23
3.16	Distribution of time bins containing pulse maxima. The distribution for every detector over multiple time bins can be seen.	24
3.17	2D Hit map for all GEMs in local detector coordinates.	25
3.18	Hit map, Charge sharing, Cluster Size and Cluster Multiplicity for a detector plane.	26
4.1	The cubic-foot MTS. Yellow lines show GEM locations; green lines show the position of scintillators for triggering [26].	28
4.2	A schematic view of track fit method and calculating residual.	29
4.3	Local Coordinates and global Coordinates of 8 GEM detectors in MTS.	31
4.4	Track angle distribution for Top and Bottom detectors.	32

4.5	Track angle distribution for Left and Right detectors.	33
4.6	The residual distributions before (left) and after (right) alignment (150 iterations) for top and bottom detectors. The distributions are fitted with a double-Gaussian function.	35
4.7	The residual distributions before (left) and after (right) alignment (150 iterations) for top and bottom detectors. The distributions are fitted with a double-Gaussian function.	36
4.8	The residual distributions before (left) and after (right) alignment (150 iterations) for top and bottom detectors. The distributions are fitted with a double-Gaussian function.	37
4.9	The residual distributions before (left) and after (right) alignment (150 iterations) for top and bottom detectors. The distributions are fitted with a double-Gaussian function.	38
4.10	The residual distributions before (left) and after (right) alignment (250 iterations) for left and right detectors. The distributions are fitted with a double-Gaussian function.	39
4.11	The residual distributions before (left) and after (right) alignment (250 iterations) for left and right detectors. The distributions are fitted with a double-Gaussian function.	40
4.12	The residual distributions before (left) and after (right) alignment (250 iterations) for left and right detectors. The distributions are fitted with a double-Gaussian function.	41
4.13	The residual distributions before (left) and after (right) alignment (250 iterations) for left and right detectors. The distributions are fitted with a double-Gaussian function.	42
4.14	Residual mean vs. iteration number for top and bottom detectors.	43
4.15	Residual mean vs. iteration number for left and right detectors.	43
4.16	Chi Square vs. shift parameters in XY-plane for top and bottom detectors.	44
4.17	Chi Square vs. shift parameters in XY-plane for top and bottom detectors.	45
4.18	Chi Square vs. rotation angle in XY-plane for GEM 2.	46
4.19	Chi Square vs. rotation angle in XY-plane for GEM 3.	47

4.20	Chi Square vs. rotation angle in XY-plane for GEM 4.	48
4.21	Chi Square vs. shift parameters in XZ-plane for the left and right detectors.	49
4.22	Chi Square vs. shift parameters in XZ-plane for the left and right detectors.	50
4.23	Chi Square vs. rotation angle in XZ-plane for GEM 6.	51
4.24	Chi Square vs. rotation angle in XZ-plane for GEM 7.	52
4.25	Chi Square vs. rotation angle in XZ-plane for GEM 8.	53
4.26	Correlation of track angles measured with pairs of top and bottom detectors before alignment.	54
4.27	Correlation of track angles measured with pairs of top and bottom detectors after alignment.	55
4.28	Correlation of track angles measured with pairs of left and right detectors before alignment.	56
4.29	Correlation of track angles measured with pairs of left and right detectors after alignment.	57
4.30	A schematic view of calculating track angle difference.	60
4.31	The track angle difference before (left) and after (right) shifting GEMs in 3 directions.	60
B.1	The schematic view of the position of APVs on GEM detectors based on their numbers.	69

List of Tables

4.1	Final shift parameters for the TOP and BOTTOM GEMs in local coordinate	58
4.2	Final rotation parameters for the TOP and BOTTOM GEMs in local coordinate	58
4.3	Final shift parameters for the LEFT and RIGHT GEMs in local coordinate	59
4.4	Final rotation parameters for the LEFT and RIGHT GEMs in local coordinate	59
4.5	Final shift parameters for the global alignment	60
B.1	Location of APVs Based on the Local Coordinate	70
B.2	Location of APVs Based on the Local Coordinate	71
B.3	Location of APVs Based on the Local Coordinate	72
C.1	Lists of scripts and functions used in the scripts	73

List of Abbreviations

ADC	Analog/Digital Converter
AMORE	Automatic MONitoring Environment
DATE	ALICE Data Acquisition and Test Environment
FEC	Front End Concentrator
GEM	Gas Electron Multiplier
MTS	Muon Tomography Station
PCB	Printed Circuit Board
SRS	Scalable Readout System

Acknowledgements

Firstly, I would like to express my sincere gratitude to my advisor Prof. Hohlmann for his patience, motivation and immense knowledge during this research. He provided me an opportunity to join his group. His experience and guidance were invaluable to me during this work.

I would like to thank all the students in HEP lab. In particular, I am grateful to Dr. Aiwu Zhang for enlightening me during this research. He introduced me GEM detector, data acquisition and analysis.

I also place on record my sense of gratitude to all the teachers and instructors I have ever had, who directly or indirectly, helped me during this journey by teaching me the valuable knowledge. They provided me with a deep insight into science and its application which has buried deep into my mind forever.

Last but not least, my deepest thanks goes to my parents for their constant support and encouragement and I want to dedicate this thesis to them.

Chapter 1

Introduction

Over 120 million vehicles enter the United States each year. Some of them can transport hidden nuclear weapons or nuclear material. The X-ray radiography is limited because its energy is too low to penetrate many cargo containers. Muons are elementary charged particles that go through the same interactions as electrons, but have much greater masses. Because of this, muons could be an alternative to electrons in scattering tomography, which is the pattern formed by the scattering of charged particles as they pass through a material.

Muon Tomography is a technique which can detect shielded special nuclear material (SNM) by using the multiple scattering of cosmic ray muons. This technique is used for both SNM and high-Z shielding materials.

The earth is continuously bombarded by energetic particles, mostly protons. These interact in the upper atmosphere through the nuclear force, producing showers of particles that include many short-lived particles called pions. The pions decay, producing muons. Muons interact with matter primarily through the Coulomb force and have no nuclear interaction. The Coulomb force removes

energy from the muons more slowly than nuclear interactions. Consequently, all of the muons arrive at the earth's surface as penetrating, weakly interacting charged radiation. The flux at sea level is about 1 muon/cm²/min in an energy and angular range useful for tomography [1].

This technique was first developed at Los Alamos National Laboratory in 2003. This method uses multiple Coulomb scattering of muons to create a tomographic reconstruction of material through which the muons pass. So, we will be able to distinguish high-Z from low-Z materials, where Z is the atomic number.

Multiple Coulomb scattering is a non-deterministic process resulting from scattering of the electrically charged muon by the charged nuclei of atoms along its path. This produces a distribution of scattering angles where the central 98% can be well approximated by a Gaussian distribution with width [2]

$$\theta_0 = \frac{13.6 \text{ Mev}}{\beta c p} z \sqrt{\frac{x}{X_0}} [1 + 0.038 \ln \left(\frac{x}{X_0} \right)]$$

where p is the momentum of the muon, βc is its velocity, z is the charge number of the incident particle (in this case 1), and x/X_0 is the amount of material traversed by the muon in radiation lengths. The functional dependence allowing the ability to discriminate between different materials comes from this final term. The radiation length X_0 of a material is well approximated by the formula [2]

$$X_0 = \frac{716.4 \text{ g cm}^{-2} \cdot A}{Z (Z + 1) \ln \left(\frac{287}{\sqrt{Z}} \right)}$$

where Z is the atomic number of the material being traversed and A is the atomic mass of the material.

Chapter 2

The Gas Electron Multiplier Detector

2.1 The Gas Electron Multiplier

The Gas Electron Multiplier (GEM) was introduced by Fabio Sauli in 1997 [3]. The GEM belongs to a larger class of detectors called Micro-Pattern Gaseous Detectors (MPGD) [4]. The base material of the GEM is $50\mu\text{m}$ Kapton foil with $5\mu\text{m}$ of copper that has been coated on both sides. GEMs are the position sensitive detecting elements in the muon tomography station that is used in this work.

2.1.1 Gas Multiplication

Gas Multiplication is a process that, in a sufficiently high electric field, the electron-ion pairs produced in a gas by incident radiation generate additional ion pairs. If the energy of an electron exceeds the first ionization potential of the gas

(15.7 eV for Argon), the result of a collision can be an electron-ion pair, leaving the incident electron free to continue in the electric field. The probability for ionization rapidly increases above threshold and in typical gases has a maximum for electron energies around 100 eV [5]. Therefore, it is necessary to provide an electric field strength capable of accelerating electrons to an energy sufficient to produce additional ionization.

2.1.2 Detector Gas

An electron avalanche can be created by applying a large electric field to a gas to knock a few electrons out of the atoms so that they are then accelerated by the electric field. With the extra energy imparted to a few free electrons, they will soon impact other atoms to knock off more electrons. In the avalanche process excited and ionized atoms are produced, which have to return to their ground state. In noble gases, gas multiplication occurs at lower fields than in gases composed of complex molecules, making noble gases the main component of most detector gas fillings. Excited noble gas atoms can only return to their ground state through the emission of a photon. The minimum energy of this photon is 11.6 eV for argon, which is well above the ionizing potential of the copper electrodes in the detector, and which therefore can release electrons that cause new avalanches. The same effect can be caused by ionized argon atoms being neutralized on the cathode, radiating the energy balance as a photon. Additional avalanche can causes permanent discharges and the created photons have to be absorbed before reaching to the electrodes. Polyatomic molecules have a large amount of non-radiative excited states that allow the absorption of photons. Such gases are called quenchers. CO_2 is a simple non-flammable

quencher gas and because of that CO_2 is a well suited gas in the GEM detectors.

For the Gas Electron Multiplier(GEM) detectors, a mixture of Ar and CO_2 with the volume ratio 70/30 is used. The ratio 70/30 was chosen because it provides good discharge protection and less gain change with a changing electrical field than a mixture with a smaller CO_2 content. This relaxes the mechanical tolerances for the detectors. However, the 30% CO_2 content makes it necessary to use rather high operating voltages in order to achieve sufficiently high gains.

2.1.3 The Gas Electron Multiplier

The GEM belongs to a larger class of detectors called Micro-Pattern Gaseous Detectors (MPGD) [5]. The GEM consists of a thin insulating polymer foil which is on both sides coated with thin metal layers. The whole structure is perforated with a large number of circular holes in a regular pattern. The holes are biconical in shape with outer diameters of $70\mu\text{m}$ and inner diameters of $50\mu\text{m}$. They are equally spaced over the surface of the GEM foil with a pitch of $140\mu\text{m}$. An image of a GEM foil under an electron microscope can be found in Figure 2.1. Because of the potential difference between the two electrodes of the foil, a high electric field is generated inside the holes. So, avalanche multiplication occurs if electrons drift into the hole region [3].

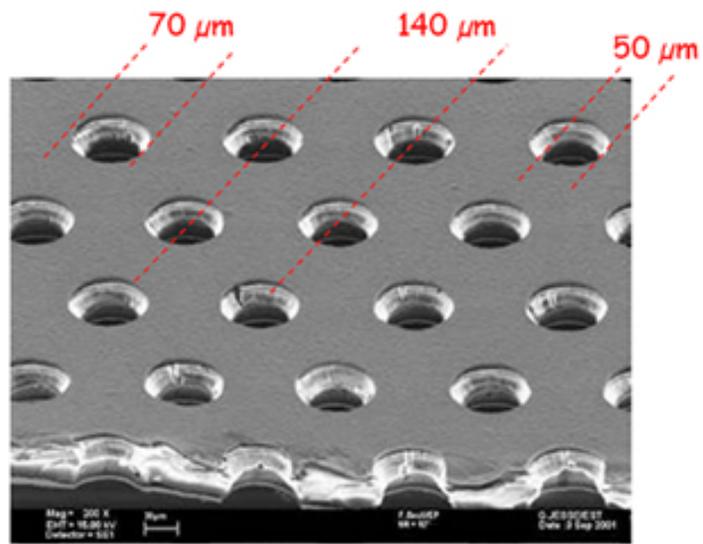


Figure 2.1: GEM foil under electron microscope [6].

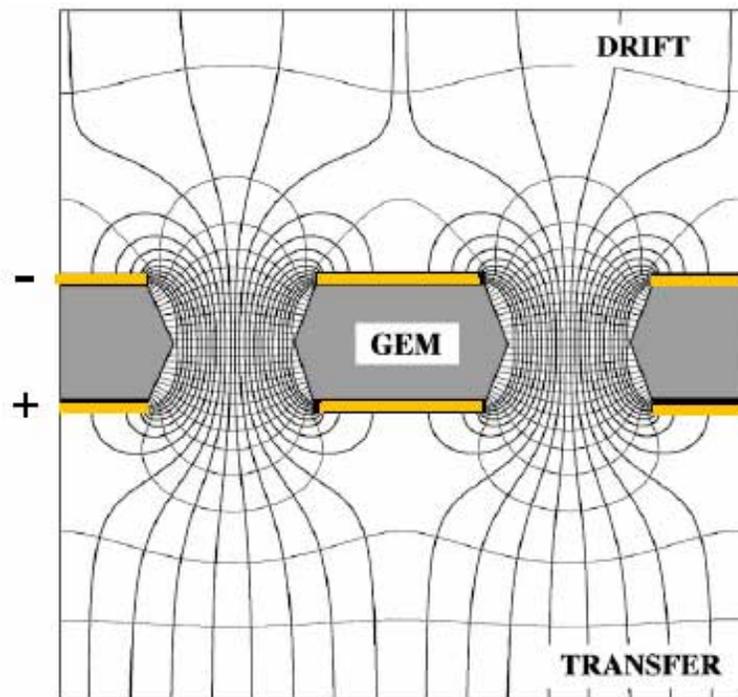


Figure 2.2: Electric field in GEM foil [7].

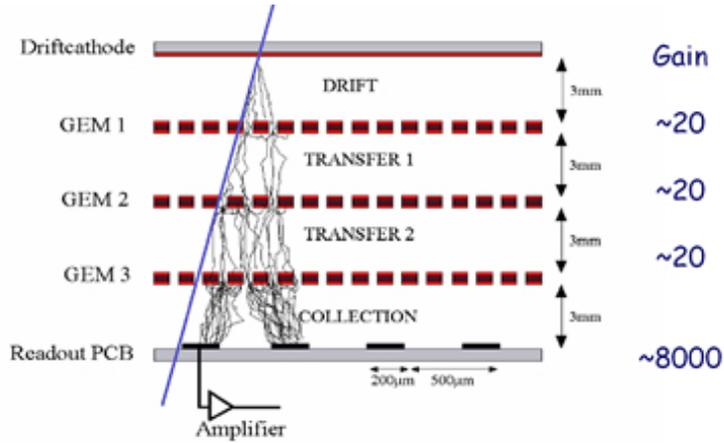


Figure 2.3: The triple-GEM detector and the amplification concept [8].

Also the electric field lines in a GEM holes are shown in Figure 2.2. In the case of the GEM detectors used for this study, three GEM layers are cascaded to reach even higher gains with very low discharge probability [9]. Three GEM layers are used to produce an effective gain with very low discharge probability. This concept is shown in Figure 2.3.

2.2 Design of GEM Detector

2.2.1 The Foils

The detector foils are produced from 50 μm thick kapton with 5 μm copper coating on both sides. We are using 30 cm x 30 cm GEM foils based on an upgraded version of the original foils used by the COMPASS experiment at CERN [10]. The foils have 12 parallel sectors on the top side in order to reduce the available energy in case of a discharge. Figure 2.4 shows a GEM foil.

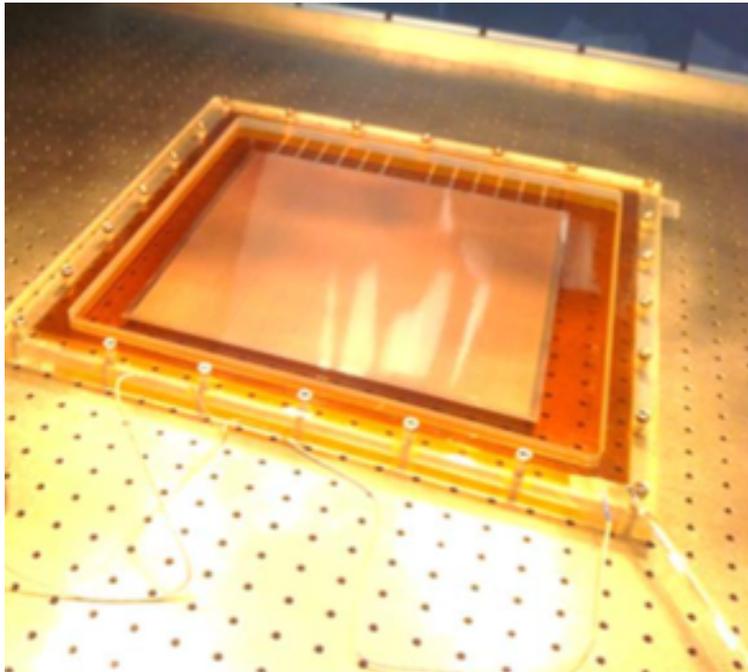


Figure 2.4: Stretched GEM foil [26].

2.2.2 The full Detector

On a two-dimensional readout, consisting of a PCB with two layers of perpendicular copper strips, the charge from the three multipliers are collected. Figure 2.5 shows a schematic drawing of the two-dimensional readout structure.

Figure 2.6 shows an exploded view of a COMPASS triple-GEM detector. The

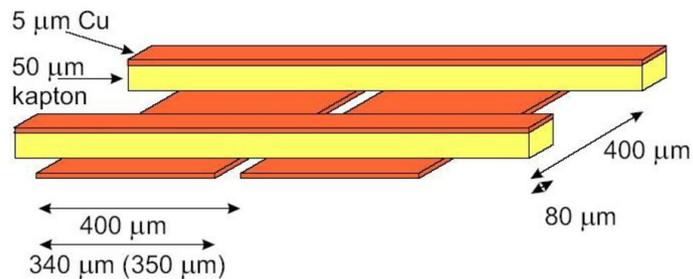


Figure 2.5: 2-D readout structure [7].

GEM layer is separated by a fiberglass spacer frame with a 3 mm drift gap. First, the drift foil is glued to the top honeycomb plate. Then, GEM foils and the spacer grids are positioned one after the other. Finally, the detector is closed with the bottom honeycomb which was glued on the readout board. In a next step, the front-end electronics are installed on the larger honeycomb. Eight triple-GEM were assembled for use in the muon tomography station [11].

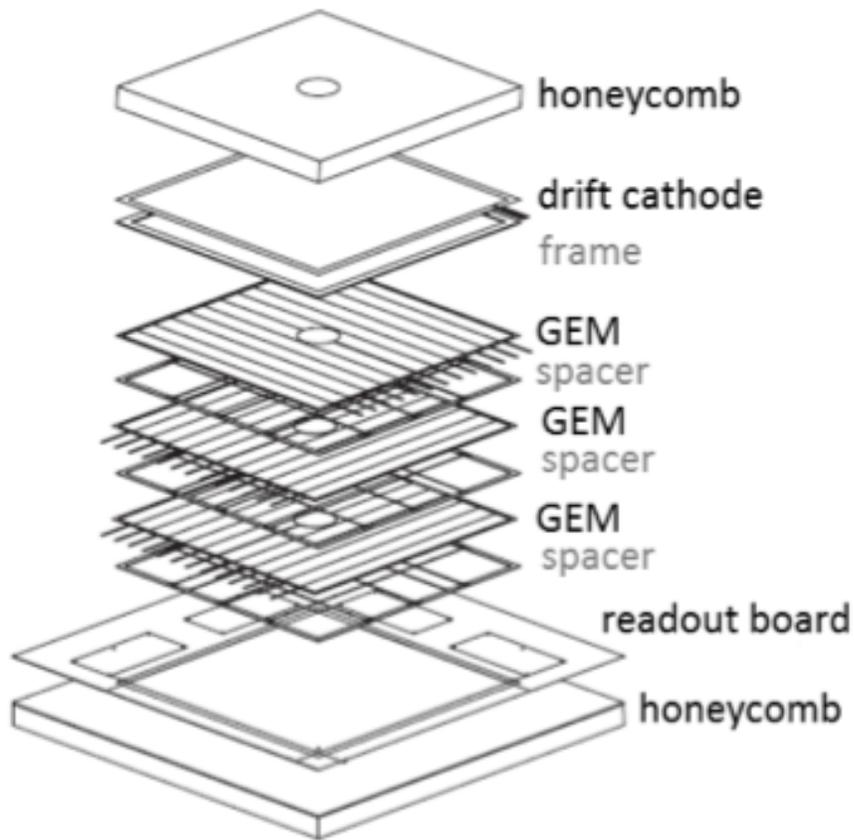


Figure 2.6: Schematic view of a triple GEM [12].

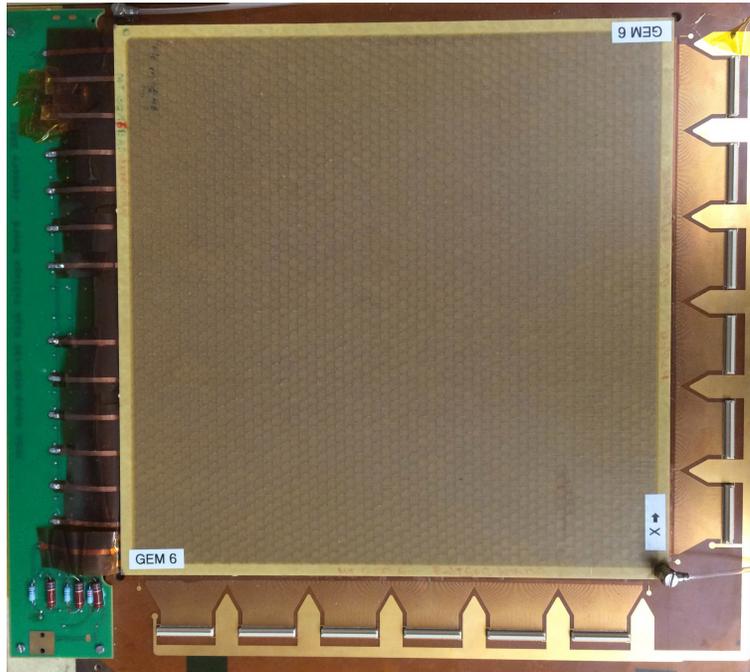


Figure 2.7: 30 cm x 30 cm triple-GEM detector used in Florida Tech muon tomography station.

Chapter 3

Data Acquisition

3.1 Muon Tomography Station

The detectors are divided into four groups on the four different sides. There is a aluminum frame which supports PVC plates that hold the GEM detectors in place. The readout system is divided into different components. At minimum, two GEM detectors are required for measuring a 2D track. There is enough room for other aspects of the infrastructure such as gas lines and cables for the DAQ electronics.

3.2 Electronics

3.2.1 SRS Hardware

The Scalable Readout System (SRS) was developed within the RD51 collaboration [13] as a complete readout system for gas detectors such as GEMs. When the FEC receives a trigger, the APVs front-end chips send analog data to the

ADS. So, the output of the APV is digitalized by transmitting to ADC. The data is sent to Front End Concentrators (FEC) and transferred to a DAQ computer. The final formatting of data on the DAQ computer is handled by DATE, which was originally developed for use by the ALICE collaboration [14]. A monitoring package was developed using ALICE and AMORE framework [15] which allows both online monitoring and offline analysis of raw data files saved by DATE. Figure 3.1 shows all components of the SRS .

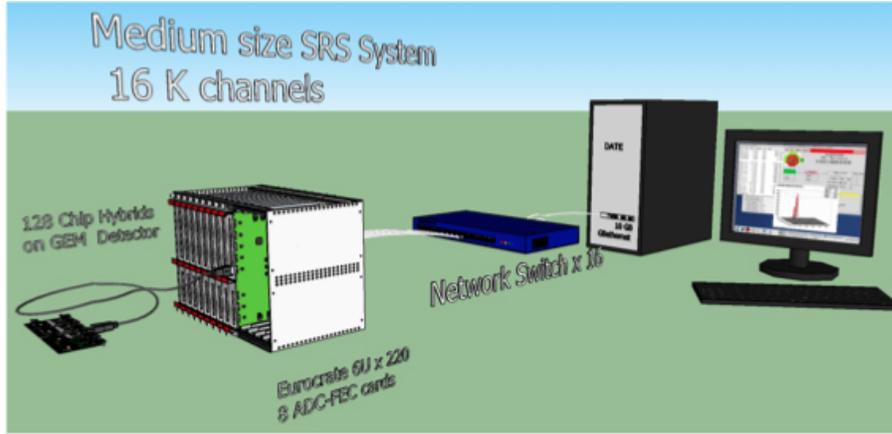


Figure 3.1: The SRS architecture [18].

3.2.2 The APV25

The APV25 is fabricated in a 0.25 μ m CMOS process, the thin gate oxide together with special layout techniques ensuring radiation tolerance [16]. Figure 3.3 shows the APV that is used for the data in this thesis. The exact position of APVs based on the local coordinate can be found in appendix B. The APV chip has 128 readout channels. Each channel consists of a pipeline with 192 memory elements. There is a maximum delay of 4 μ s between the event and the arrival of the trigger at the chip. There is a latency which is the time between the

event and the trigger that defines the amount of time that the chip goes back in the pipeline to find the correct signal that should be read out. When the APV receives a trigger, it sends the data to the ADC(Figure 3.2).

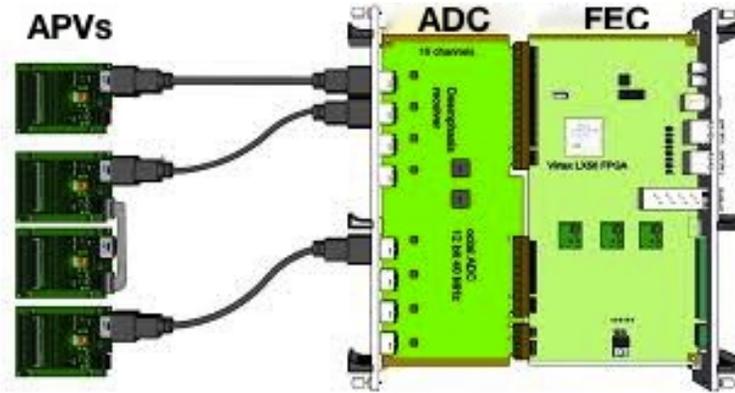


Figure 3.2: The SRS Components [17].



Figure 3.3: Two APV Hybrids [18].

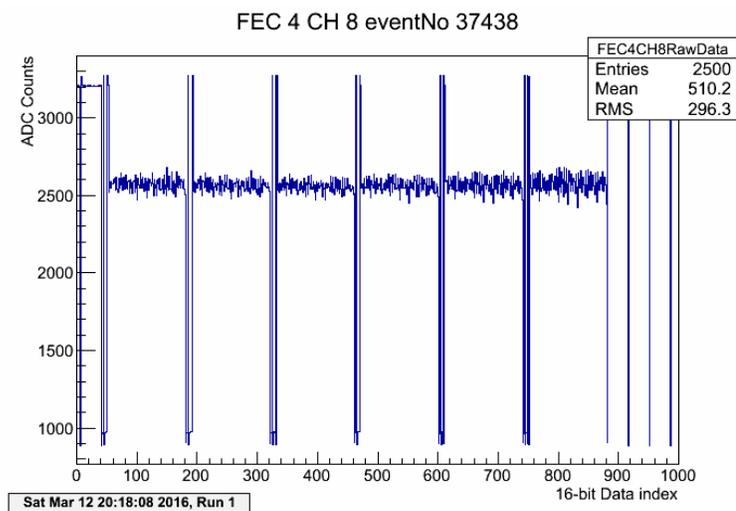


Figure 3.4: Raw APV output.

In order to decrease the number of connections to the ADC, two hybrid master/slave APVs are connected together. Raw APV signal is shown in figure 3.4.

3.2.3 Analog-Digital Converter Card

The APVs (master/slave) are connected to the ADC via an HDMI cable. The APV sends its data header information to the ADC and the ADC recognizes the information and generates a new header for the data that the APV sends out.

3.2.4 Front-End Concentrator Card

The most popular adapter card (Figure3.5) was originally designed to interface the RD51 APV25 ASIC hybrid [19]. The front-end cards are mounted to the detector and bounded to the readout. The FEC is built with I/O for trigger and



Figure 3.5: SRS Analog-Digital Converter card [20].

clocks. The FECs are connected to the SRS online system by gigabit Ethernet cables.



Figure 3.6: SRS Analog-Digital Converter card [20].

3.3 Online Analysis

3.3.1 Trigger

The trigger signal is provided by a Quarknet discriminator / coincidence card [21] which is fanned out by a NIM module to the NIM input on the 6 different FECs. The trigger system for MTS consists of four scintillators. Two out of

four scintillators must fire in coincidence. The selection of tracks in amoreMTS is based on one cluster in each plane of all detectors crossed by the track. This cut is selected as good events. Based on Monte Carlo simulation $\sim 17\%$ [12] of recorded events triggered by coincidence in the plastic scintillator paddles could be selected as good events. The current trigger rate of the MTS is ~ 30 Hz. Figure 3.7 shows an example of a plot from Muonic for the rates of the hits of muons on the scintillators .

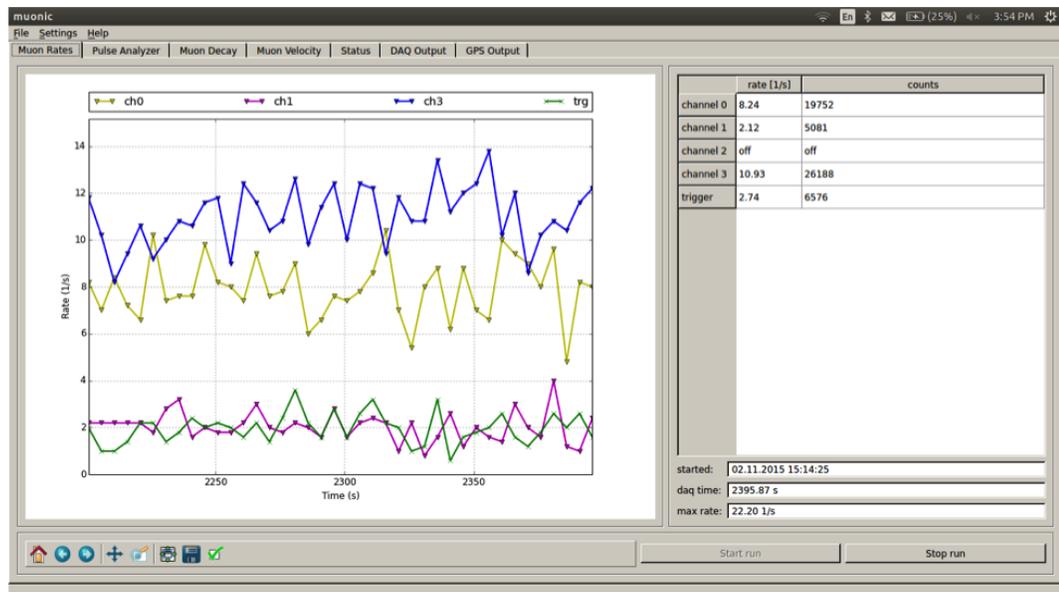


Figure 3.7: Sample of trigger rates from Muonic.

3.3.2 Data Acquisition

The original DAQ software for the implementation of the SRS consists of two software packages developed for the ALICE experiment at CERN [22]. The detector data are sent to DAQ for each event. As DAQ software the ALICE acquisition system DATE (Data Acquisition and Test Environment) [23] is used.

DATE is used for collecting raw data from all FECs and converting the data into a binary format which can be used for online monitoring. DATE runs on top of Scientific Linux. Command editDb grants access to a GUI which is used for configuring DATE. The user can add different equipment and configuration by using the GUI. There are packages which can be used for online and offline analysis. One of them is AMORE (Automatic MONitoring Environment). DATE and AMORE are using MySQL to store data before writing to a disk. Since all of these interactions are done automatically, there is no need to interact directly with MySQL. Users, usually detector teams, develop modules that are typically split into two main parts corresponding to the publishing and the subscribing sides of the framework [23]. The description is shown in Figure 3.8. For all AMORE version, which are created for MTS the common and publisher folders are used. The design is shown in Figure 3.9. AMORE is based on the DATE monitoring library and ROOT. There is a need to develop AMORE for a new application to create a module to publish and subscribe methods for a particular detector type. There are different agents which allow processing of multiple data streams at the same time.

3.3.3 Pedestal

The baseline differs from channel to channel and has to be determined. The baseline values of individual channels are called pedestal. Since the goal is to recover the true signal, the pedestals have to be subtracted from the APV output. In order to do this, the ADC should subtract the pedestals for every individual channel. The pedestal values are transferred to the ADC by software. A raw APV signal is shown in figure 3.10. The bias voltage is set to 2000 V

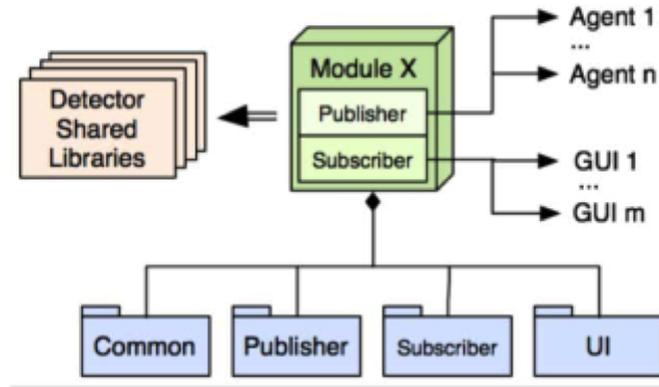


Figure 3.8: Description of the AMORE module [15].

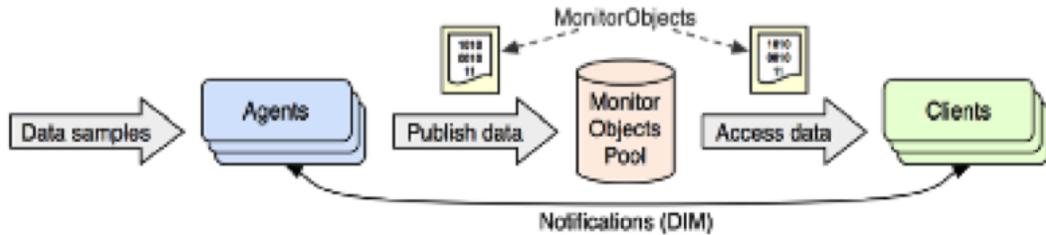


Figure 3.9: The publisher model in AMORE [15].

for taking pedestal, which is below the required voltage for amplification. Also in this voltage there is no gas gain in the detector. The "Pedestal" run type is used for taking pedestal data. There are pedestal offset and noise for all the 128 channels for an APV (Figure 3.11).

3.3.4 Troubleshooting

There is an online check of the performance of all detectors by online analysis. By monitoring their performance, it is possible to detect problems as early as possible.

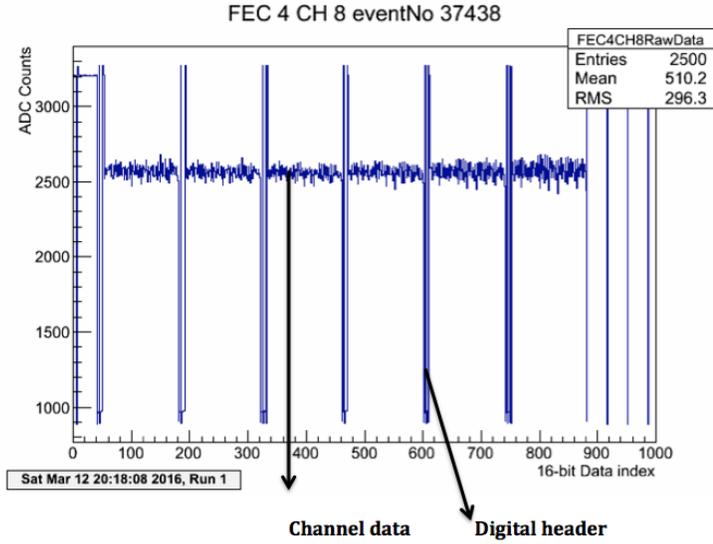


Figure 3.10: Raw APV output.

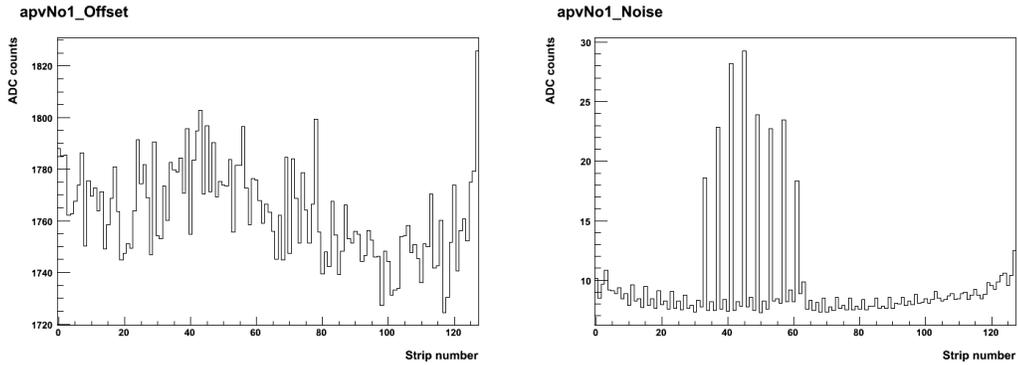


Figure 3.11: Sample of pedestal offset and width of pedestal distribution for one APV.

While data are taken, all the APVs can be monitored. It is possible to have bad APV raw data as it is shown in Figure 3.12. The effect of a problematic APV on the offline processed pedestal is shown in Figure 3.13. Figure 3.14 shows the the pedestal offset and noise for the APV with an issue.

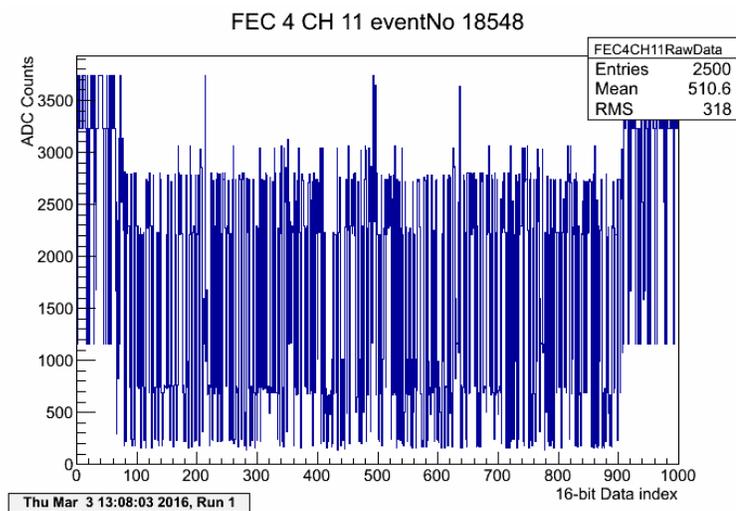


Figure 3.12: Problematic raw APV output signal.

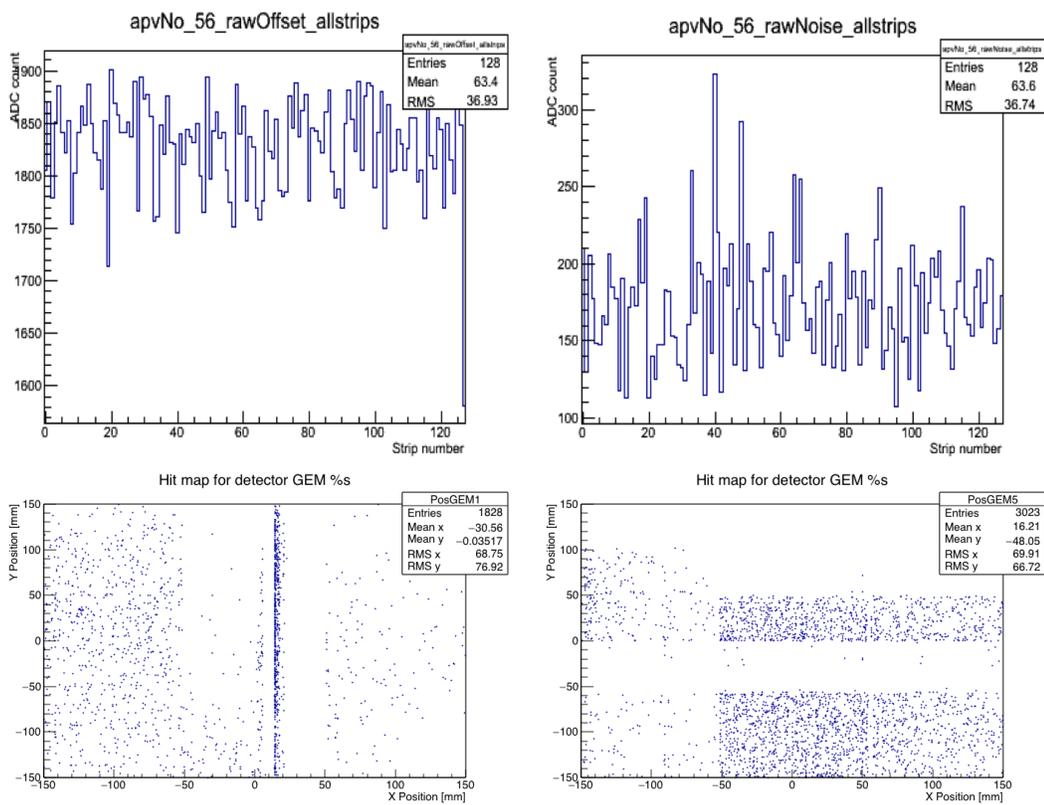


Figure 3.13: Sample of effect of problematic APV on pedestal offset and noise and Hit maps.

There are some ways to fix this issue:

- Swapping the problematic APV with another APV to see if the issue is relevant to the APV or to the detector.
- Replacing the cable which is connecting the APV to the ADC.
- Checking the connection part between the ADC and the FEC.
- Checking the two connected pins between the APV and detector plane.
- Checking the SRS configuration.
- Checking the strip occupancy plot to find the exact location of the problem

3.3.5 Online/Offline Processing

There are different run types and agents for taking data that produces monitoring plots. At first the pedestal is taken and processed. For taking real data the bias voltage is set to 4200 V. As a result of increasing the voltage, the signal can be seen in the output (Figure 3.15). For each trigger DATE writes the raw data that is taken from the FEC. Each event is separated into six 25 ns time bins. The timing of the signals in the detector is an important issue. There exists a time between the event and the trigger which is called latency. Latency defines the amount of time that the chip goes back in the pipeline to find the correct signal. The right latency should be set for the detectors. It should be understood which of the signals has the correct timing. The timing plots are shown in Figure 3.16 and 3.17.

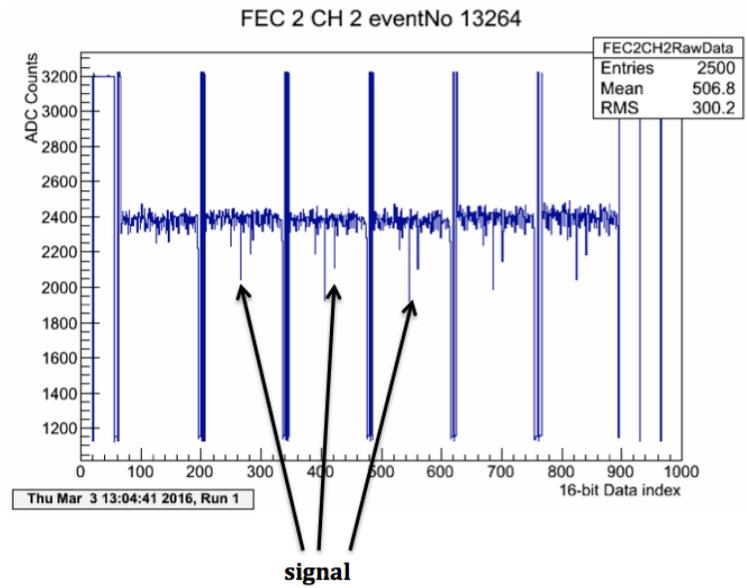


Figure 3.14: Raw APV output with the GEM signal.

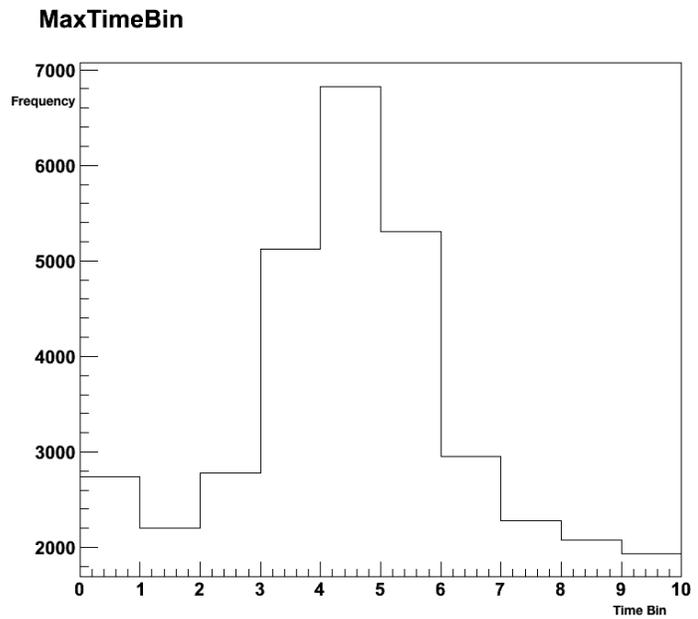


Figure 3.15: The appropriate time bin for storing the data by the FECs.

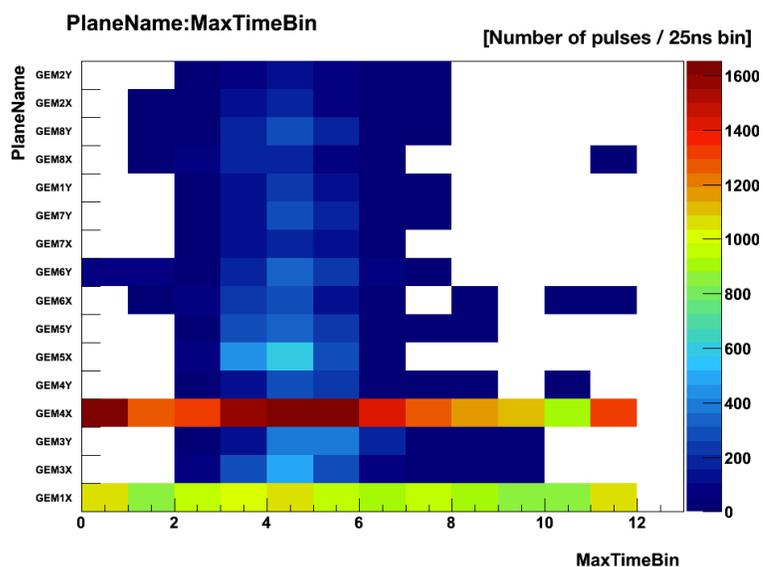


Figure 3.16: Distribution of time bins containing pulse maxima. The distribution for every detector over multiple time bins can be seen.

The 2D hit map shows the hits in x strip vs. y strip in the entire detector. There is no track selection for the 2D hit map. Figure 3.16 shows all the 2D hit maps for every GEM in offline processing. The geometrical properties of the detectors such as sector boundaries on a GEM foil can be seen. The horizontal and vertical lines are due to the spacer grid. The empty horizontal part in some GEMs is related to the sector part which may have an issue. The directory of the raw data that is used for getting these results can be found in Appendix A.

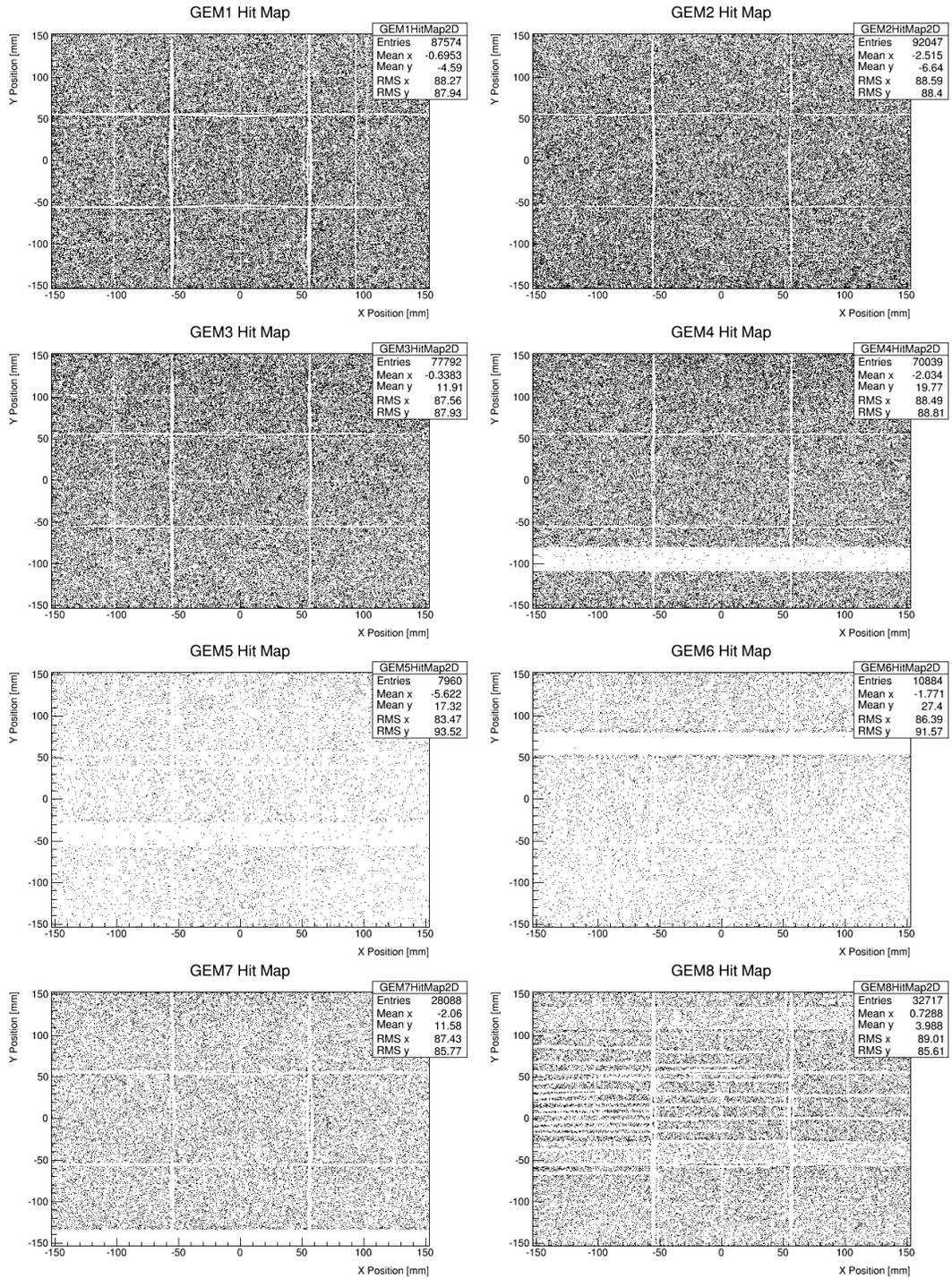


Figure 3.17: 2D Hit map for all GEMs in local detector coordinates.

The 2D charge sharing map shows the charge sum in the x strip clusters versus the charge sum in the y strip clusters. The sample of online processing plots are shown in Figure 3.18.

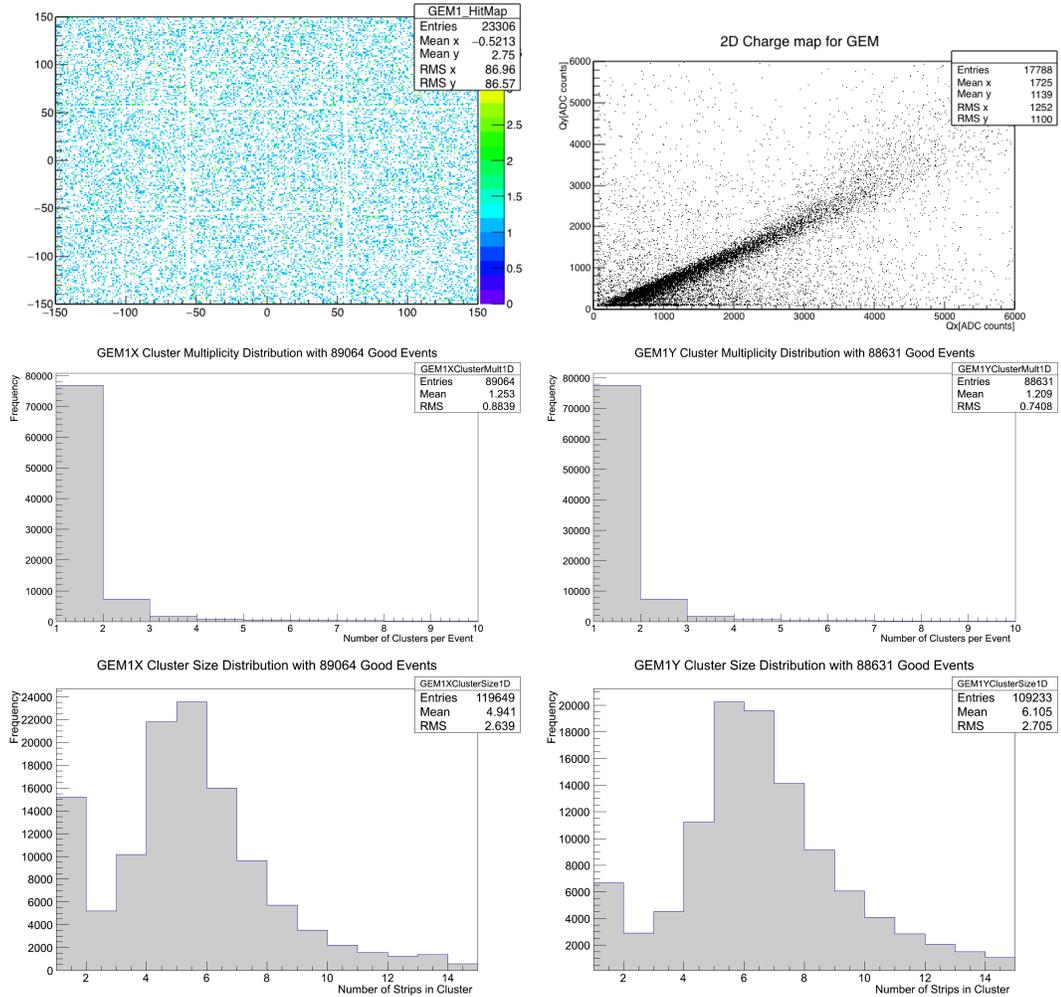


Figure 3.18: Hit map, Charge sharing, Cluster Size and Cluster Multiplicity for a detector plane.

Chapter 4

Alignment of GEM Detectors

4.1 The Reason for Aligning

There are some uncertainties in the exact position and orientation of the detectors. Getting close to the natural resolution of 50-200 μm of the GEM detectors, the uncertainty of alignment should be negligible in comparison with natural resolution. This goal is achievable by hardware and software alignment. There are also advanced hardware alignment systems such as the muon alignment system of CMS [24]. Hardware alignment methods for CMS limit the uncertainty in the nominal detector positions to the level of 100-500 μm [25]. During the design of the cubic-foot MTS it was envisioned that simple measurements of the detector positions would be able to achieve an uncertainty of $\sim 500 \mu\text{m}$ in the detector positions [26]. The cubic-foot MTS is shown in Figure 4.1. The method that is used in this thesis for the alignment is called the track-based method. This method is based on finding the residual distances between the hit positions and the points of a fitted track and tries to minimize this residual

distance. The residual distance is illustrated in Figure 4.2.

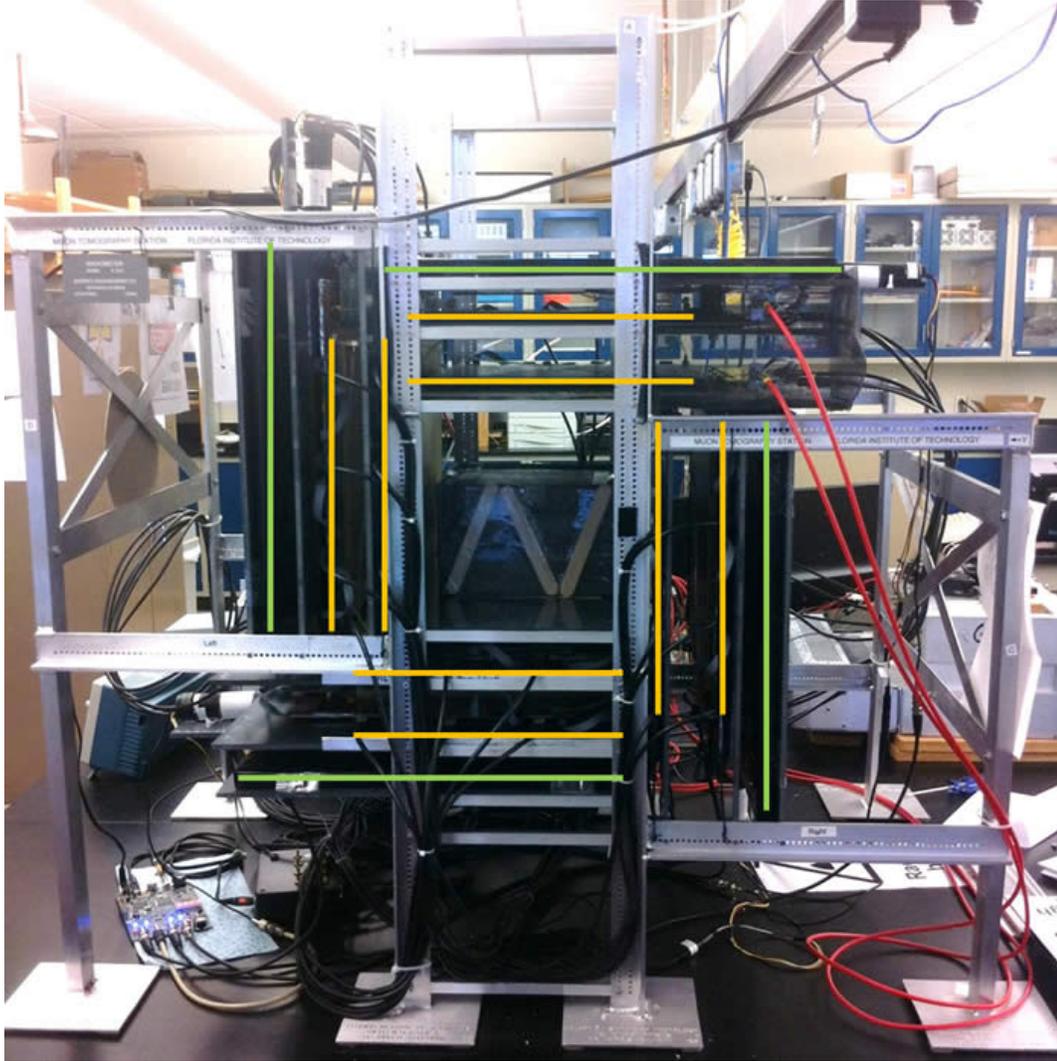


Figure 4.1: The cubic-foot MTS. Yellow lines show GEM locations; green lines show the position of scintillators for triggering [26].

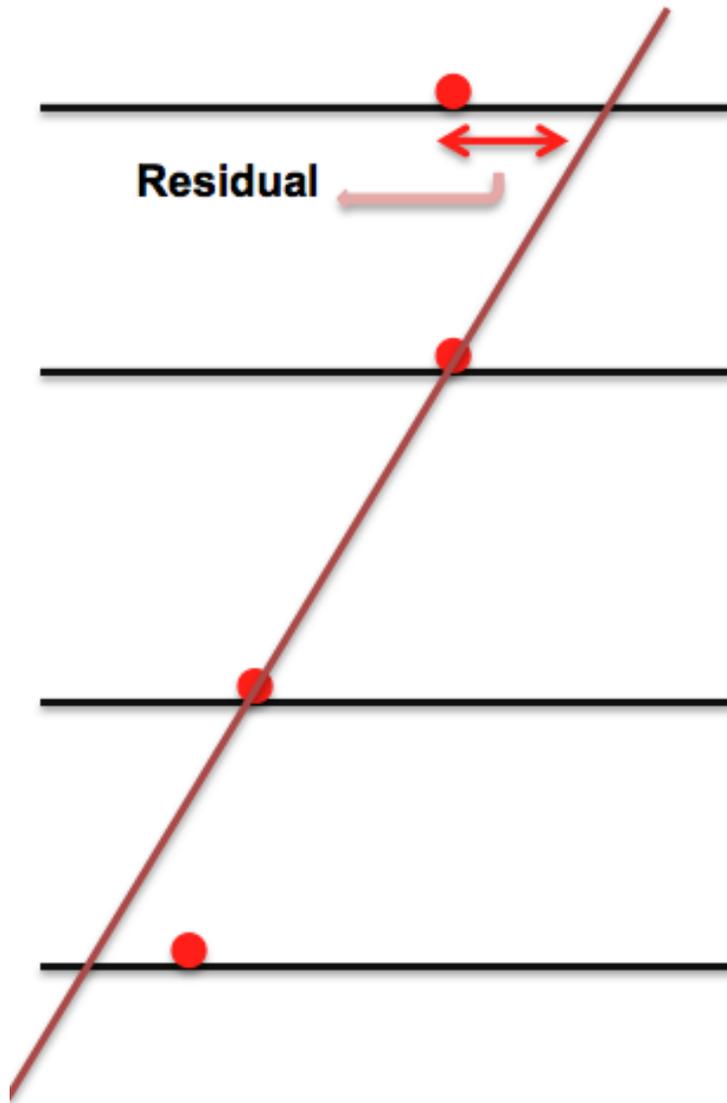


Figure 4.2: A schematic view of track fit method and calculating residual.

4.2 The Coordinate System

The position and orientation of all tracks should be determined with respect to a specific coordinate system. The GEMs are located in a local coordinate system. The orientation of local coordinate axes is illustrated in Figure 4.3. In the final alignment the global coordinate must be used. Since the global coordinate is considered for the alignment, some changes should be applied to the local coordinates in order to make the orientation of all local coordinate axes same as the global coordinate. For this reason, before adding any cut for selecting specific tracks, a minus sign should be applied to some directions. A minus sign should be applied to the X and Y directions in GEMs 1 and 2 and to the Z direction in GEMs 5 and 6.

As it is shown, Z and Y axis are fixed for top and bottom detectors respectively. The Z and Y coordinates are determined by distance measurements.

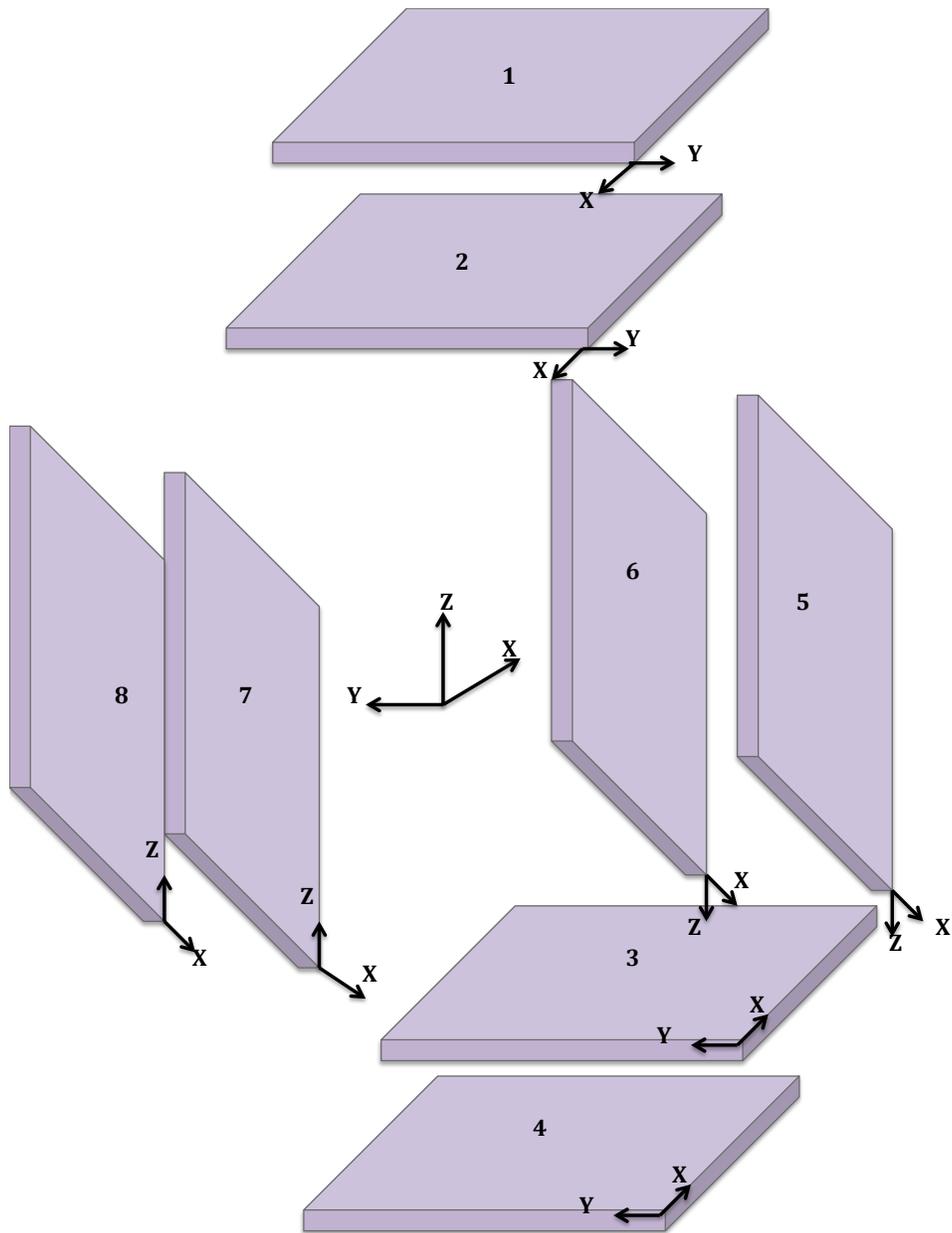


Figure 4.3: Local Coordinates and global Coordinates of 8 GEM detectors in MTS.

4.3 Track Selection

Before selecting specific tracks, GEM 1 and GEM 5 are defined as reference detectors for the alignment. For local alignment of top and bottom GEMs, GEM 1 is used as a reference. Similarly, GEM 5 is used as a reference for aligning left and right GEMs. Only close to perpendicular tracks are considered for the alignment. In this case we can assume that there is no scattering. Because there should be enough tracks to align the station, it is better to plot the angle distribution. In this plot the track angle relative to detector vs. the number of tracks per detector can be seen. By considering the angle distribution, a proper range can be selected. The angle distributions are shown in Figure 4.4 and 4.5.

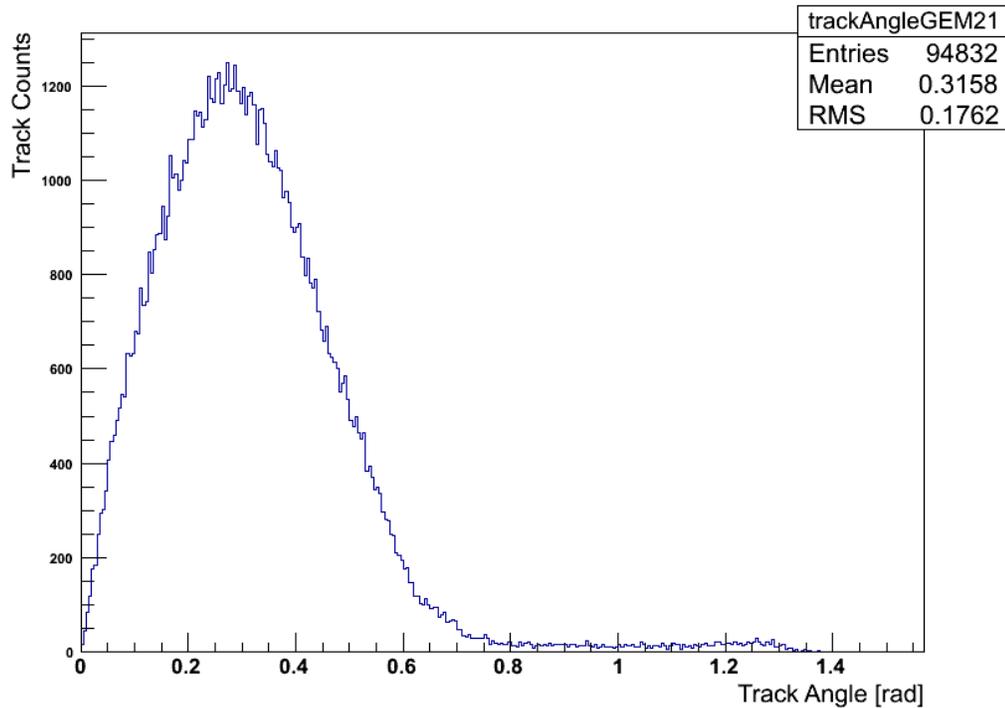


Figure 4.4: Track angle distribution for Top and Bottom detectors.

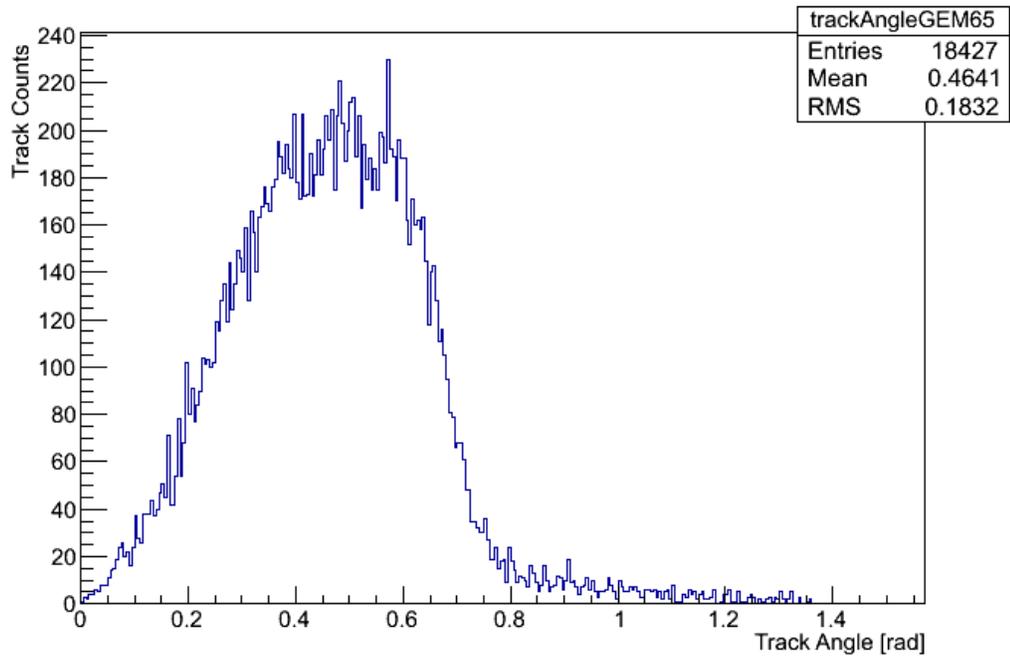


Figure 4.5: Track angle distribution for Left and Right detectors.

The angle range that is used for selecting tracks for alignment of the top and bottom detectors is from 0 to 0.08 radian ($0 \leq \theta_{TB} \leq 0.08$). There exists 4668 tracks in this range.

The angle range which is picked for left and right detectors is from 0 to 0.25 radian ($0 \leq \theta_{LR} \leq 0.25$). The number of tracks in this range are 1946.

4.4 The Alignment

4.4.1 Local Alignment

After selecting the tracks the residual distance between the hit positions and the points of fitted track is calculated. The goal is to minimize the residual distance

and plot the residual distributions for every detector. First of all, the detectors are shifted iteratively in their plane. Top and bottom detectors are shifted in the XY-plane and side detectors are shifted in the XZ-plane. In every step 20% of the residual mean value is taken as the shift parameter in the next iteration. In the next step the detectors are rotated. Detectors are rotated around Z axis for the top and bottom detectors. This axis is changed to Y for the left and right detectors. A double-Gaussian function is used to fit residual distributions. The residual distributions before and after iterative alignment can be found in Figure 4.6 to 4.11. The final residual distributions are plotted after shifting and rotating detectors.

The scripts that are used for the alignment can be found in appendix C.

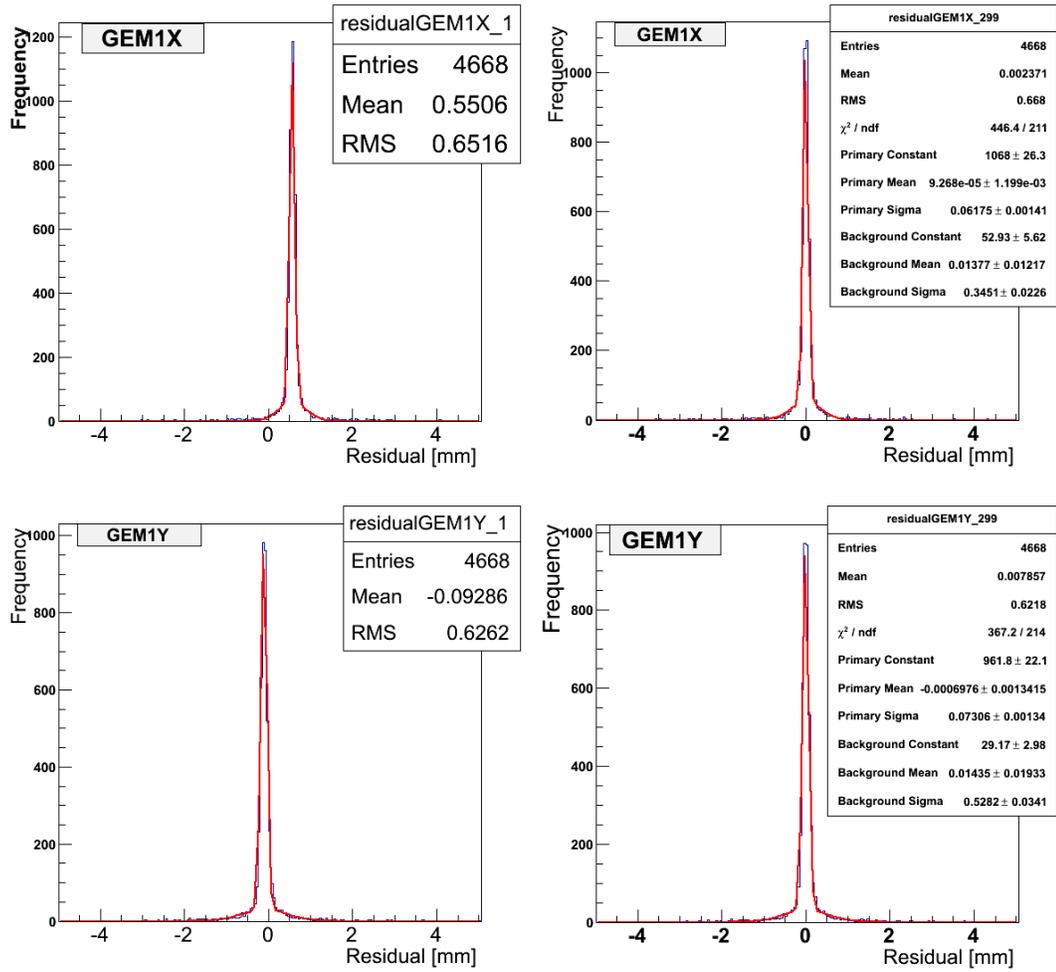


Figure 4.6: The residual distributions before (left) and after (right) alignment (150 iterations) for top and bottom detectors. The distributions are fitted with a double-Gaussian function.

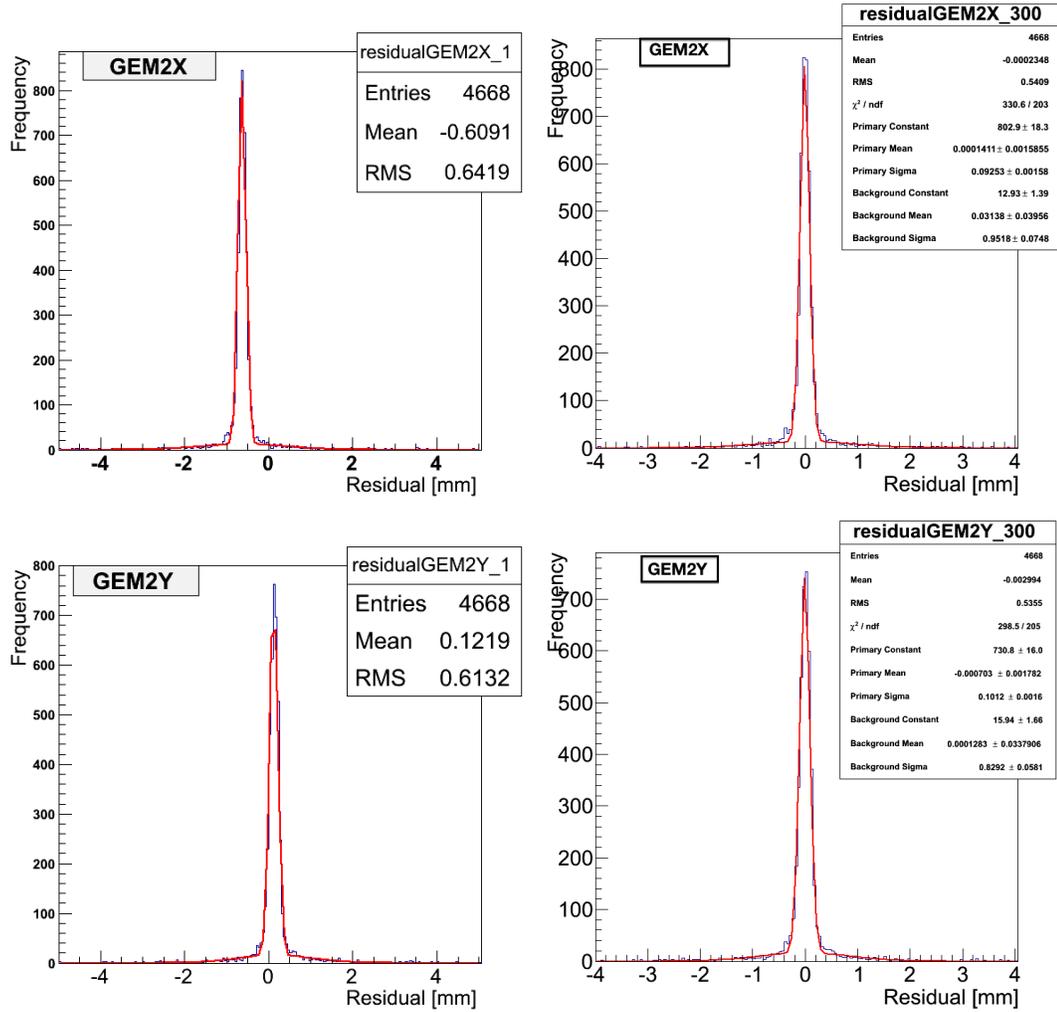


Figure 4.7: The residual distributions before (left) and after (right) alignment (150 iterations) for top and bottom detectors. The distributions are fitted with a double-Gaussian function.

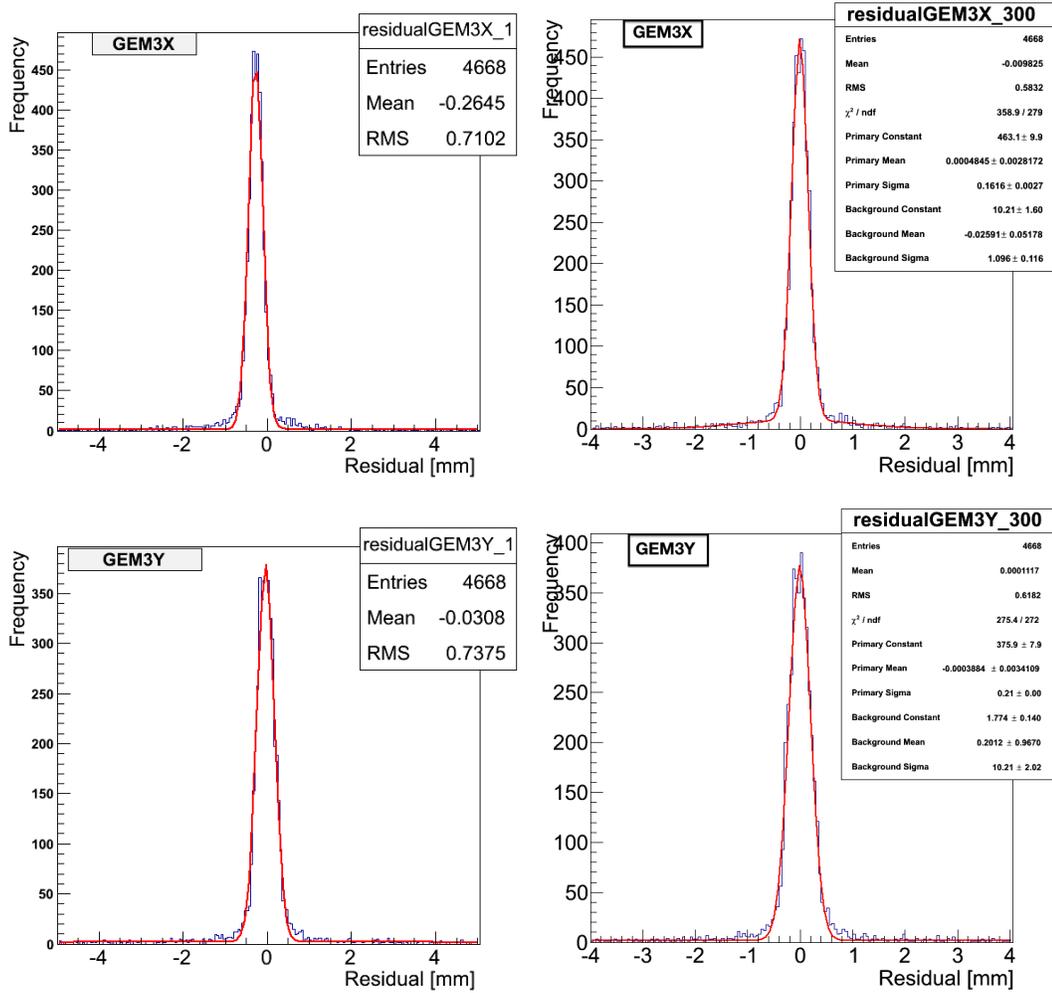


Figure 4.8: The residual distributions before (left) and after (right) alignment (150 iterations) for top and bottom detectors. The distributions are fitted with a double-Gaussian function.

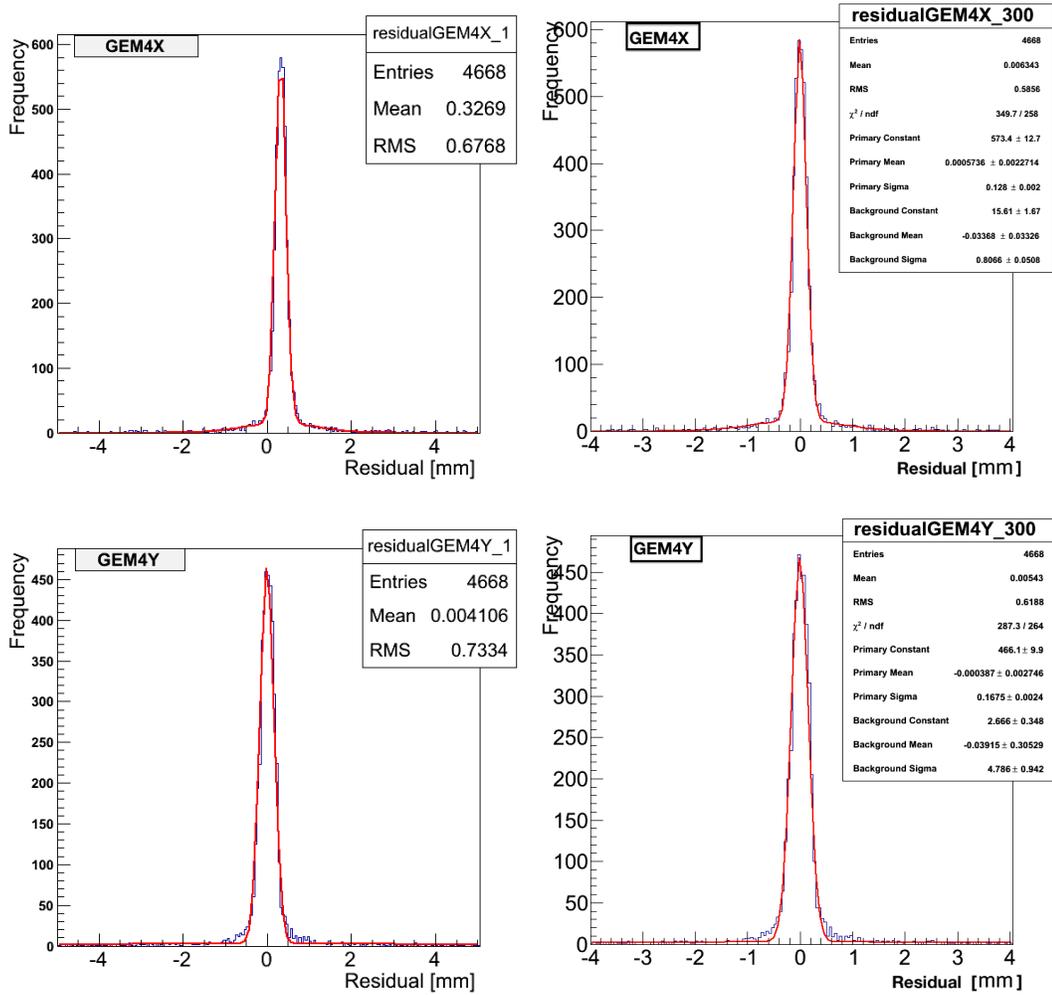


Figure 4.9: The residual distributions before (left) and after (right) alignment (150 iterations) for top and bottom detectors. The distributions are fitted with a double-Gaussian function.

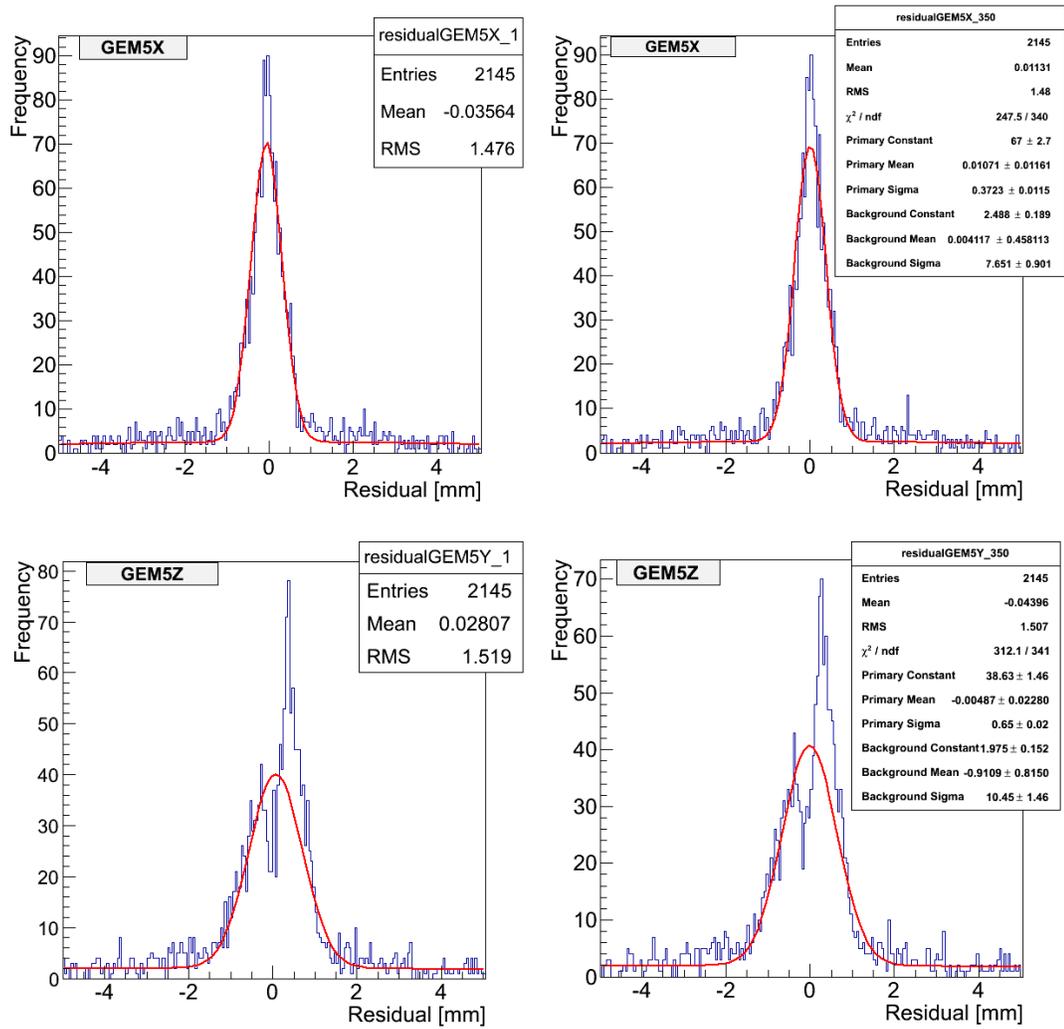


Figure 4.10: The residual distributions before (left) and after (right) alignment (250 iterations) for left and right detectors. The distributions are fitted with a double-Gaussian function.

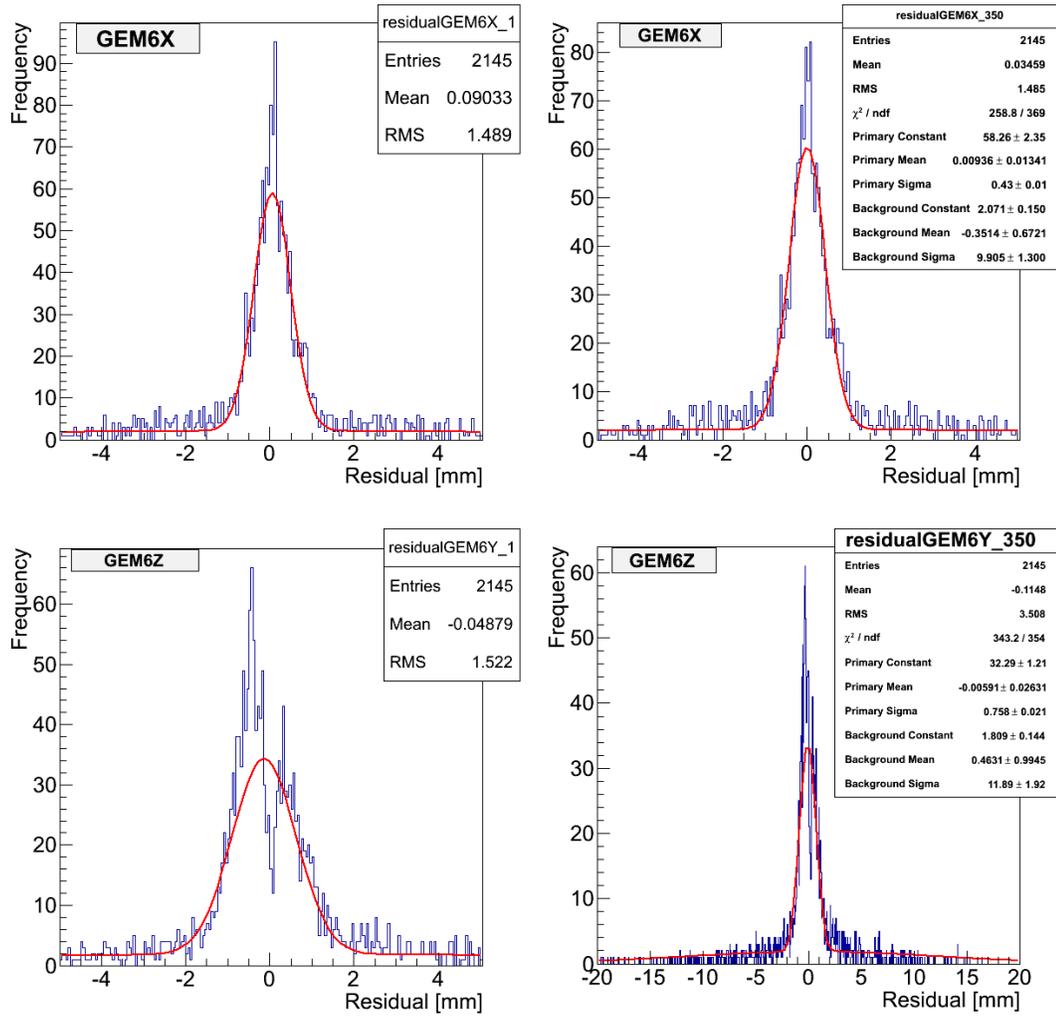


Figure 4.11: The residual distributions before (left) and after (right) alignment (250 iterations) for left and right detectors. The distributions are fitted with a double-Gaussian function.

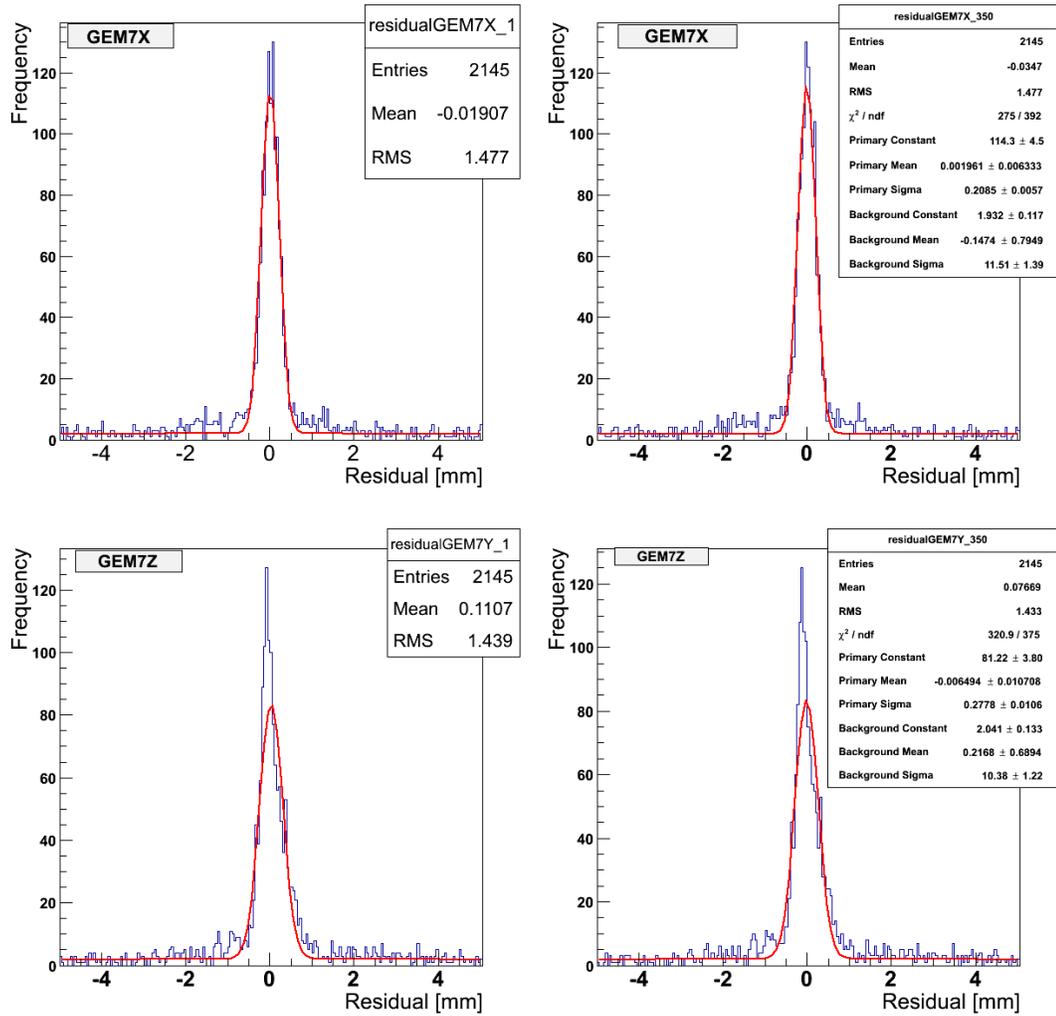


Figure 4.12: The residual distributions before (left) and after (right) alignment (250 iterations) for left and right detectors. The distributions are fitted with a double-Gaussian function.

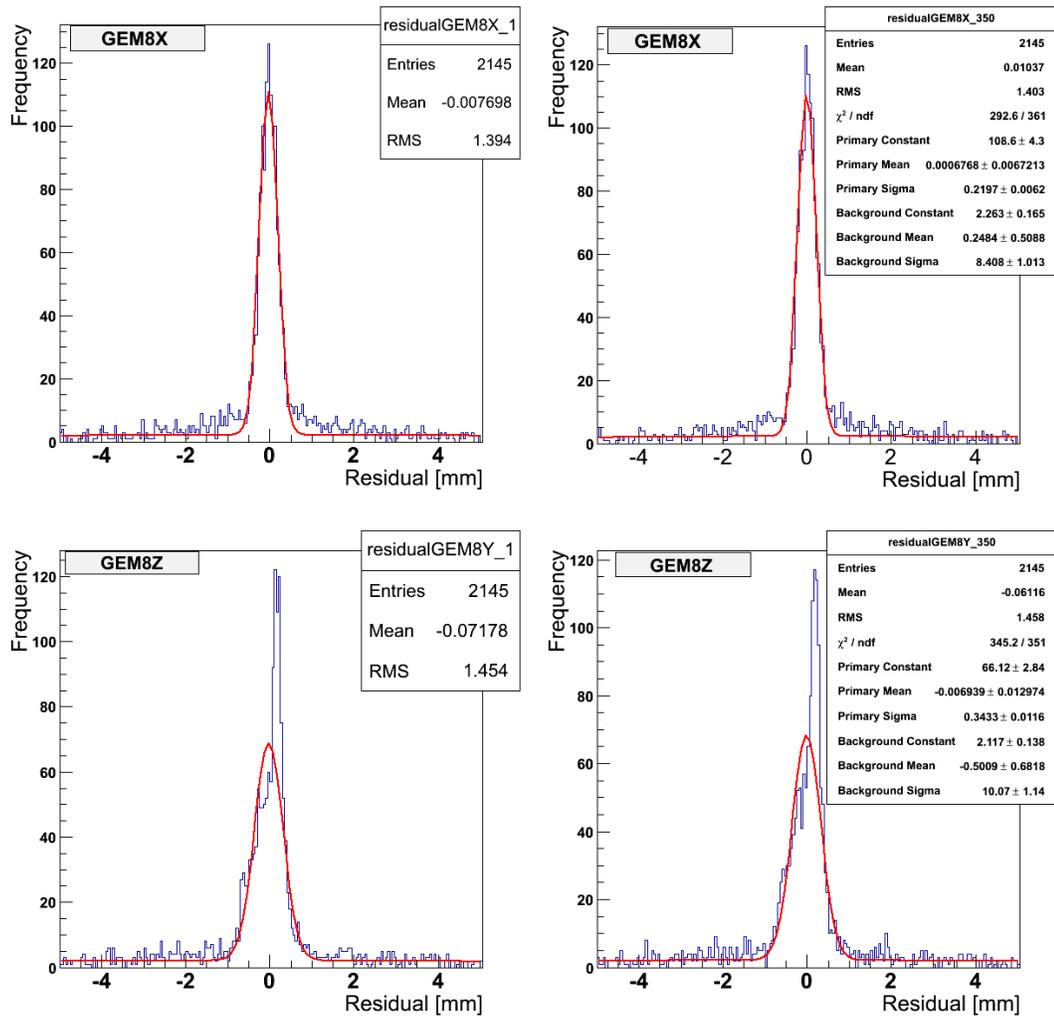


Figure 4.13: The residual distributions before (left) and after (right) alignment (250 iterations) for left and right detectors. The distributions are fitted with a double-Gaussian function.

In the residual mean versus iteration number plots, it can be seen that the mean values converge toward zero after 50 iterations (Figures 4.12 and 4.13).

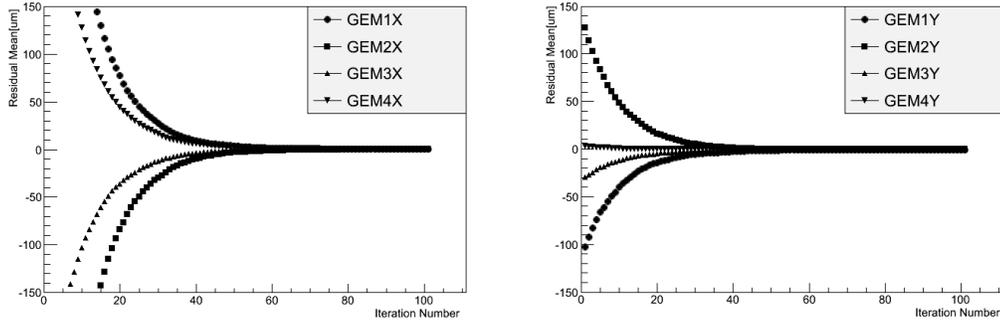


Figure 4.14: Residual mean vs. iteration number for top and bottom detectors.

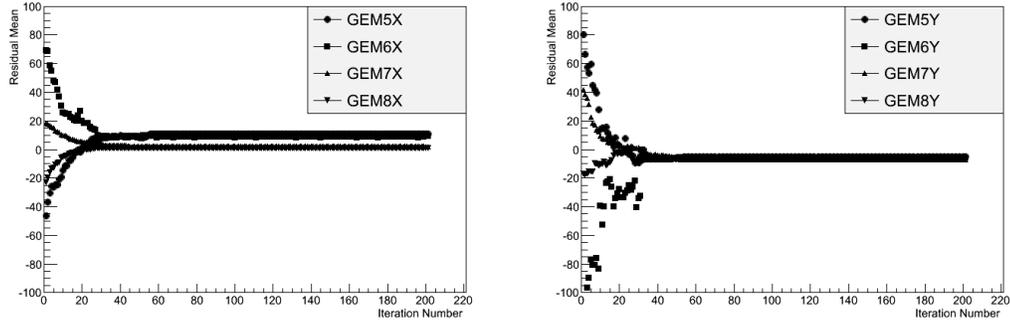


Figure 4.15: Residual mean vs. iteration number for left and right detectors.

By calculating the chi square of shift and rotation parameters and plotting them for X, Y and Z direction, parabolic curves are found. For optimizing the rotation angle for every detector, shift parameters and rotation angles are kept fixed for other detectors and every single rotation angle is changed in a small range. From the minimum of parabolic fits, the final rotation angles can be found in every direction (Figures 4.14-4.23).

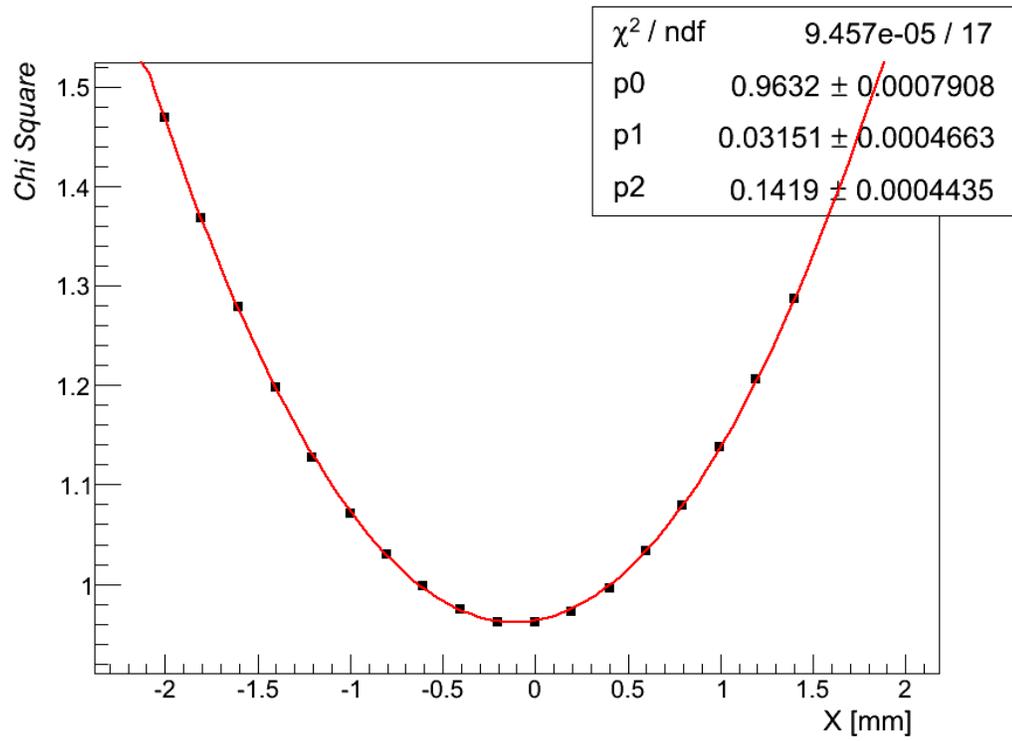


Figure 4.16: Chi Square vs. shift parameters in XY-plane for top and bottom detectors.

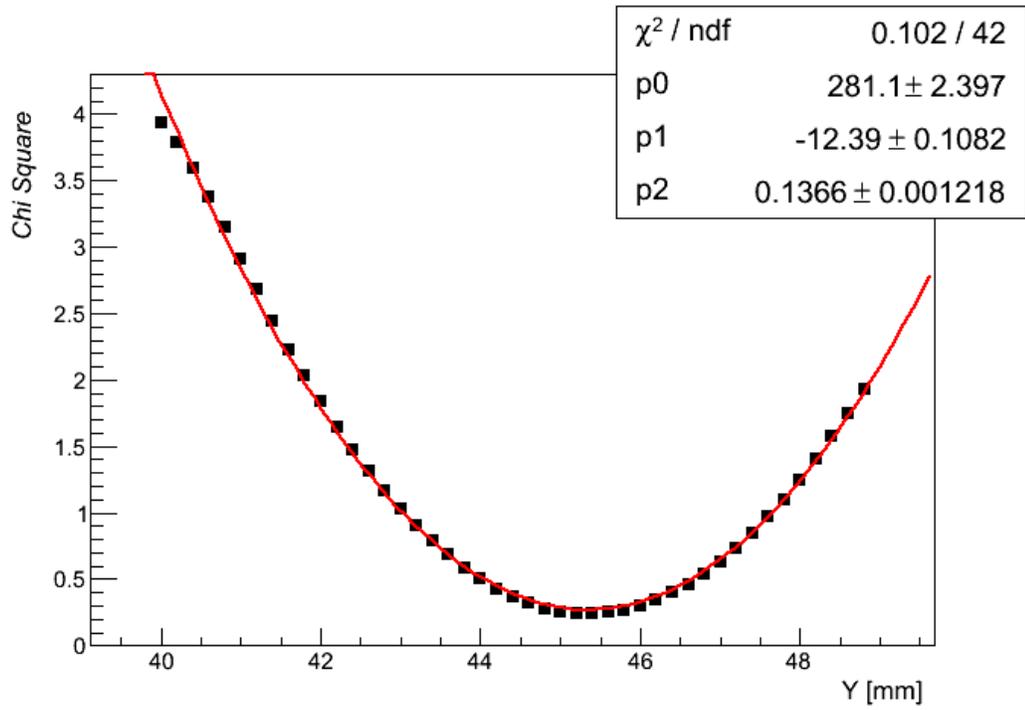


Figure 4.17: Chi Square vs. shift parameters in XY-plane for top and bottom detectors.

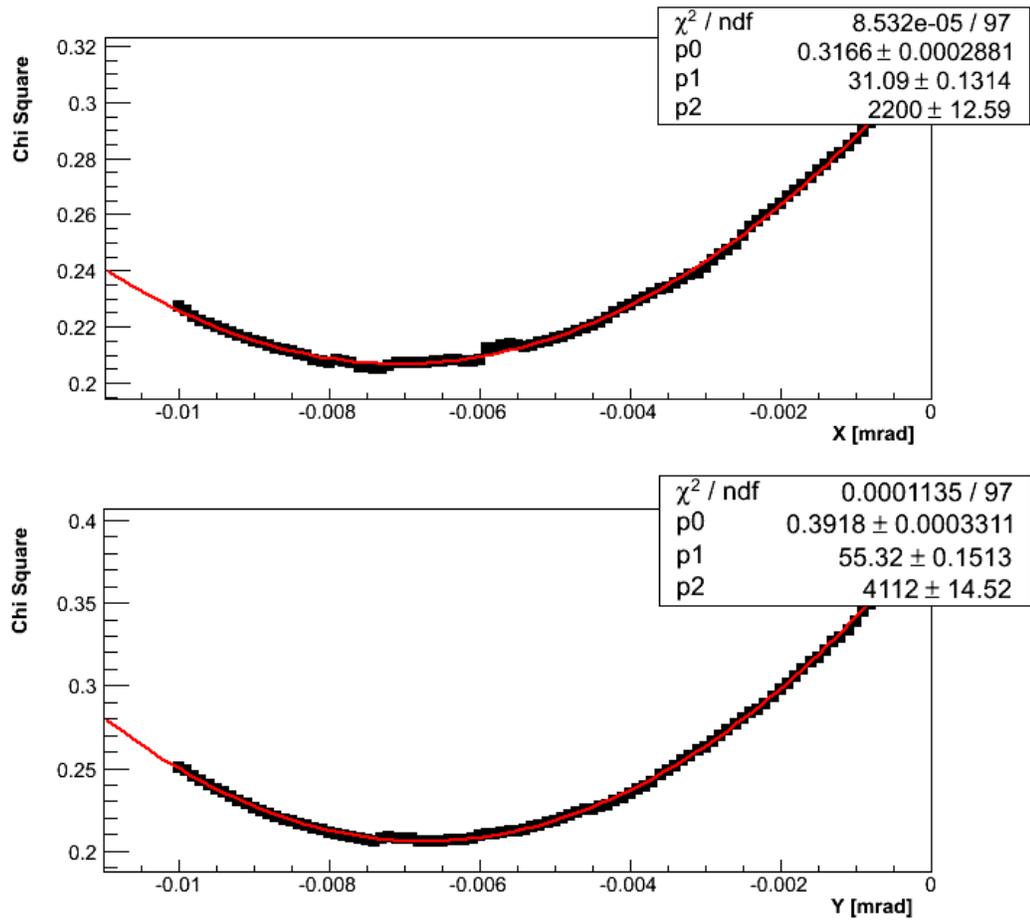


Figure 4.18: Chi Square vs. rotation angle in XY-plane for GEM 2.

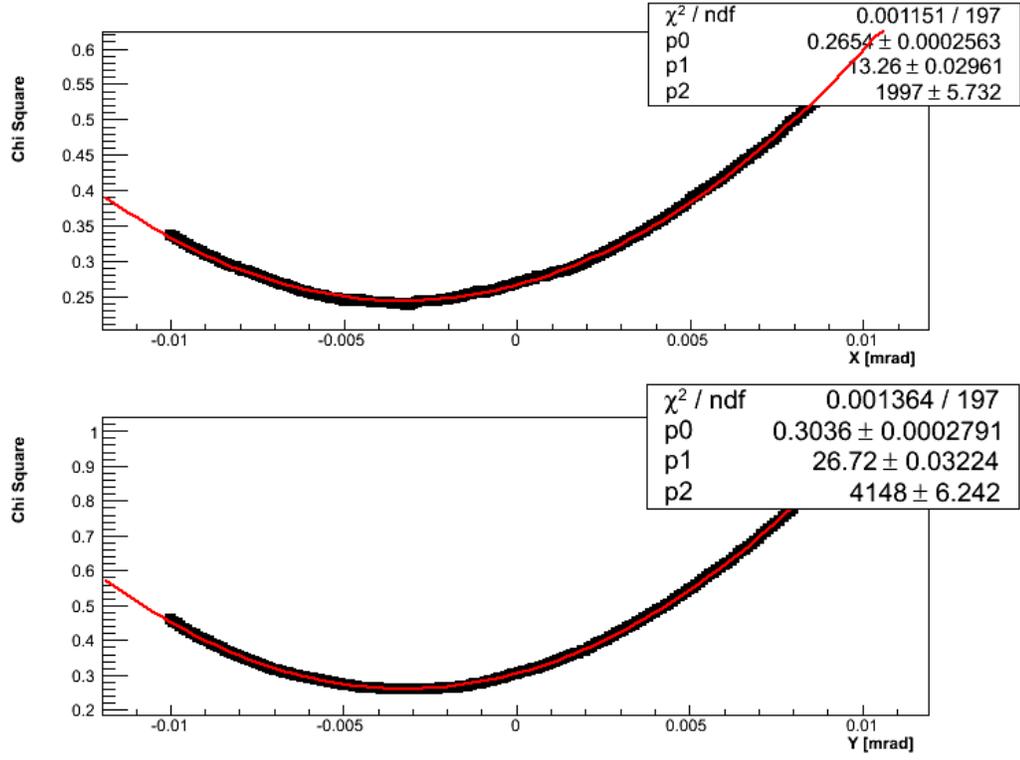


Figure 4.19: Chi Square vs. rotation angle in XY-plane for GEM 3.

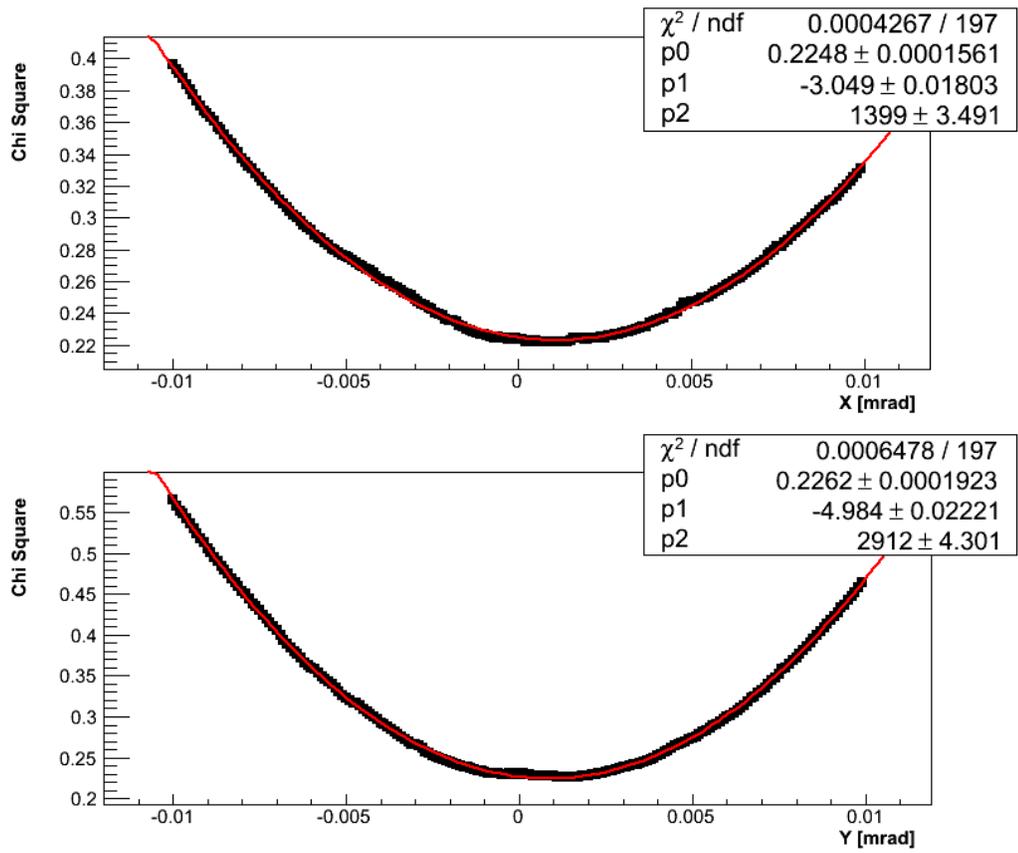


Figure 4.20: Chi Square vs. rotation angle in XY-plane for GEM 4.

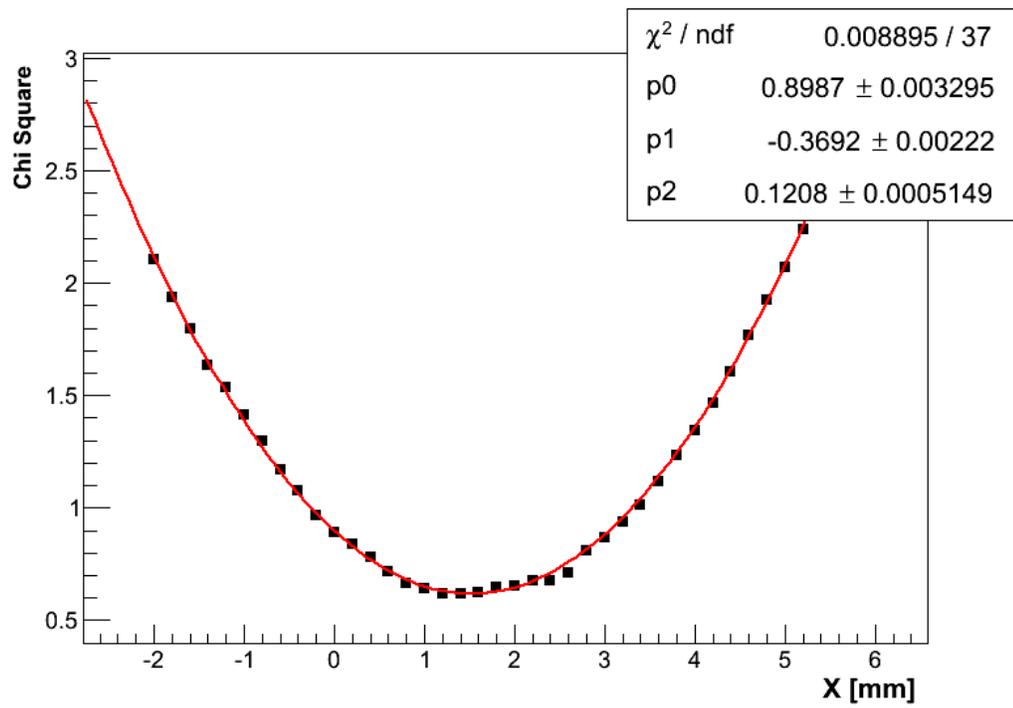


Figure 4.21: Chi Square vs. shift parameters in XZ-plane for the left and right detectors.

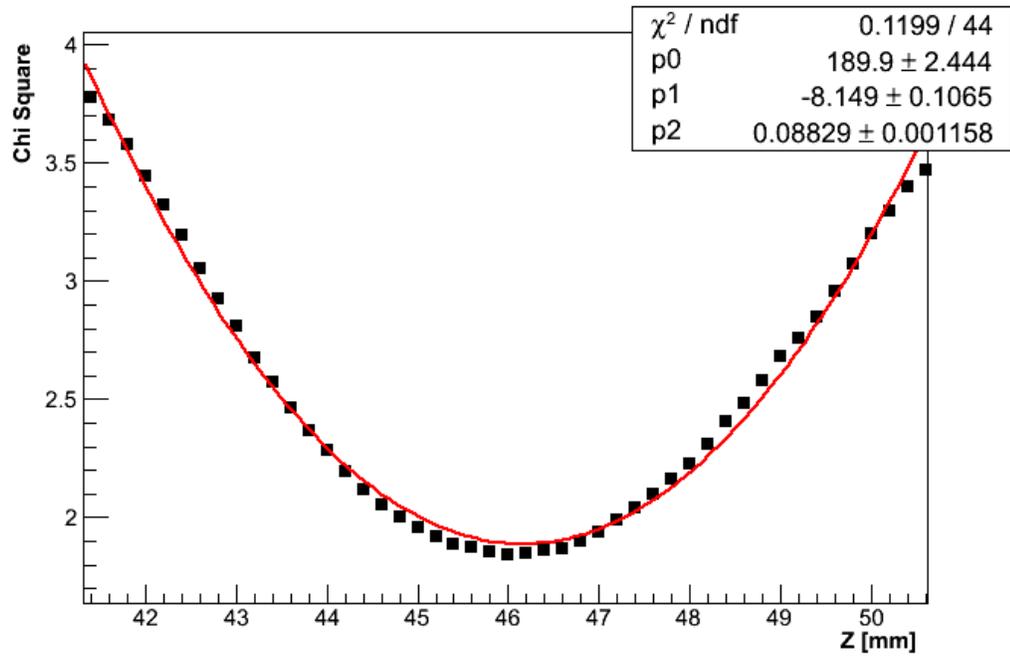


Figure 4.22: Chi Square vs. shift parameters in XZ-plane for the left and right detectors.

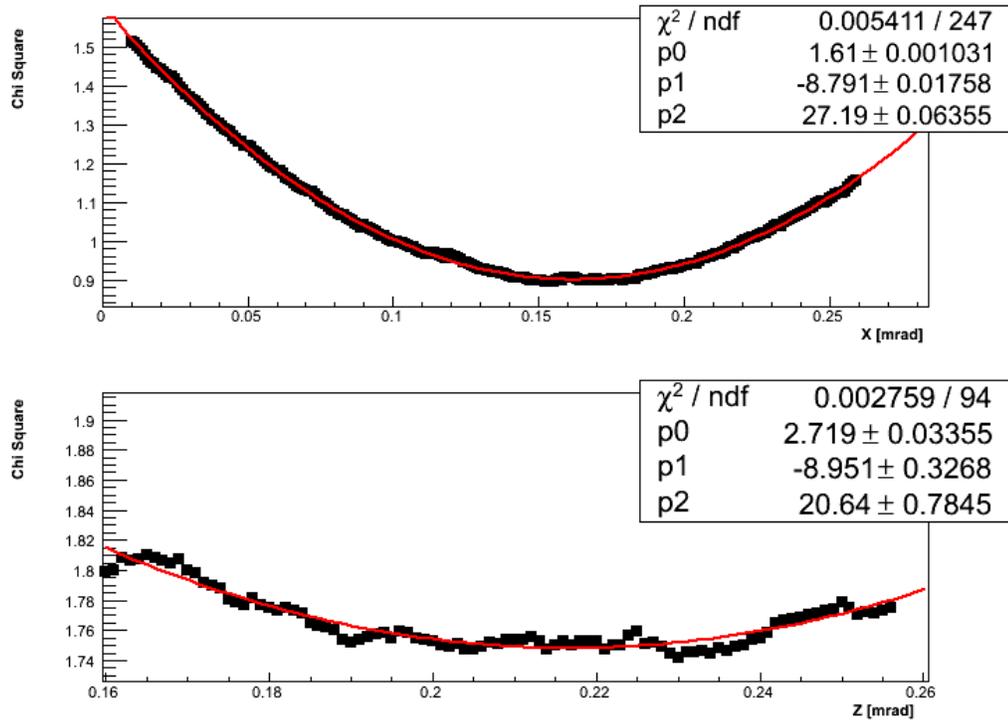


Figure 4.23: Chi Square vs. rotation angle in XZ-plane for GEM 6.

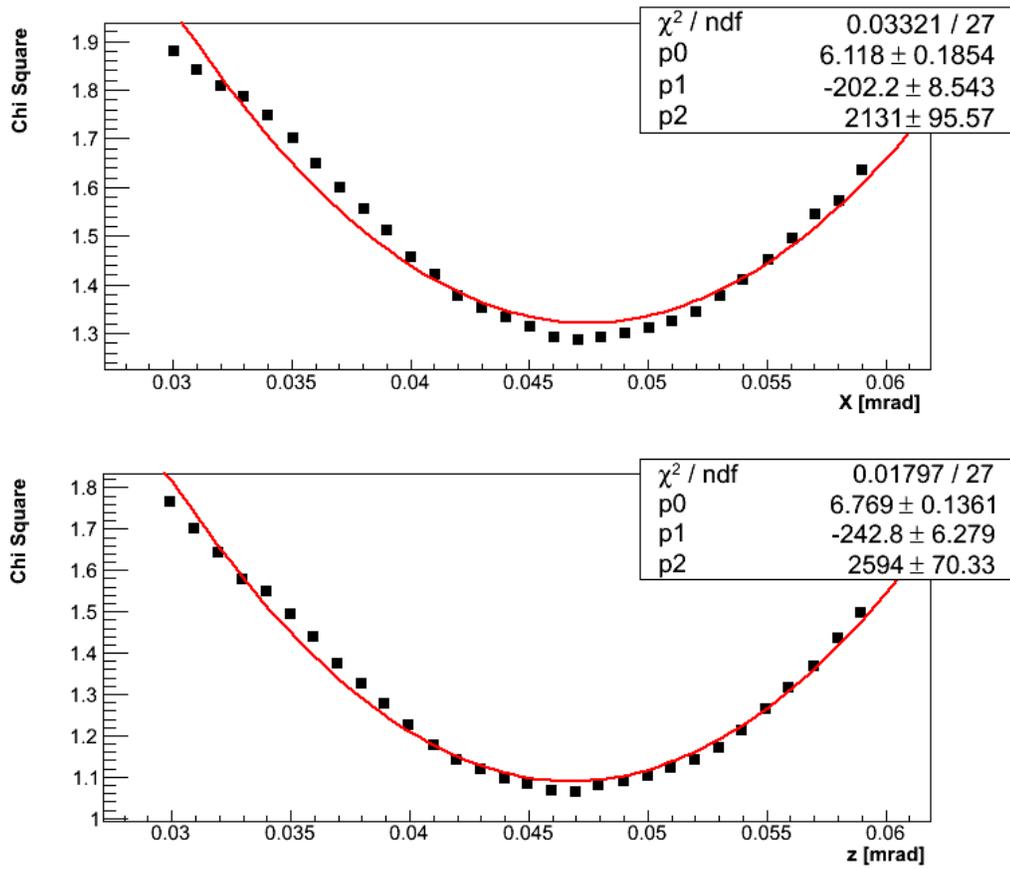


Figure 4.24: Chi Square vs. rotation angle in XZ-plane for GEM 7.

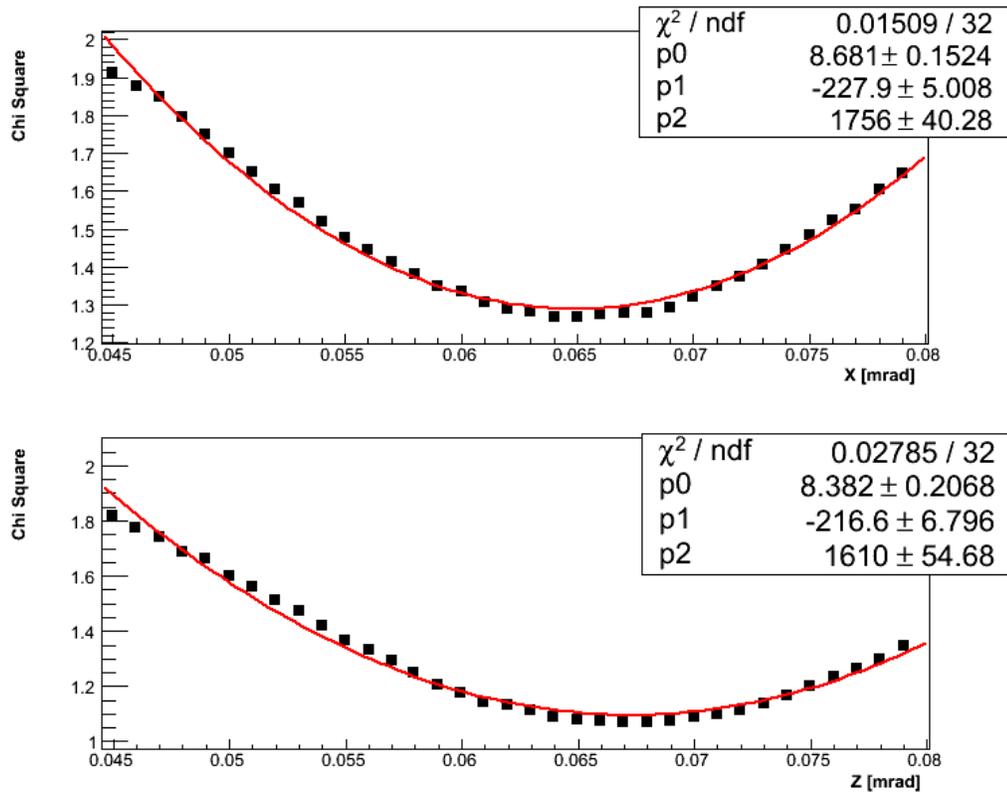


Figure 4.25: Chi Square vs. rotation angle in XZ-plane for GEM 8.

4.4.2 Cross check using track angles

For the local alignment the GEMs are divided into two groups:

- GEMs 1 to 4 (top and bottom)
- GEMs 5 to 8 (left and right)

The track angle between every group can be measured and plotted. After alignment the residual distances are minimized and because of this the track angle between detectors should be the same. Calculating the track angles between detectors and observing a diagonal trend after alignment, is a proof of precise alignment. In the following plots the track angles before and after alignment

are shown (Figures 4.18-4.21).

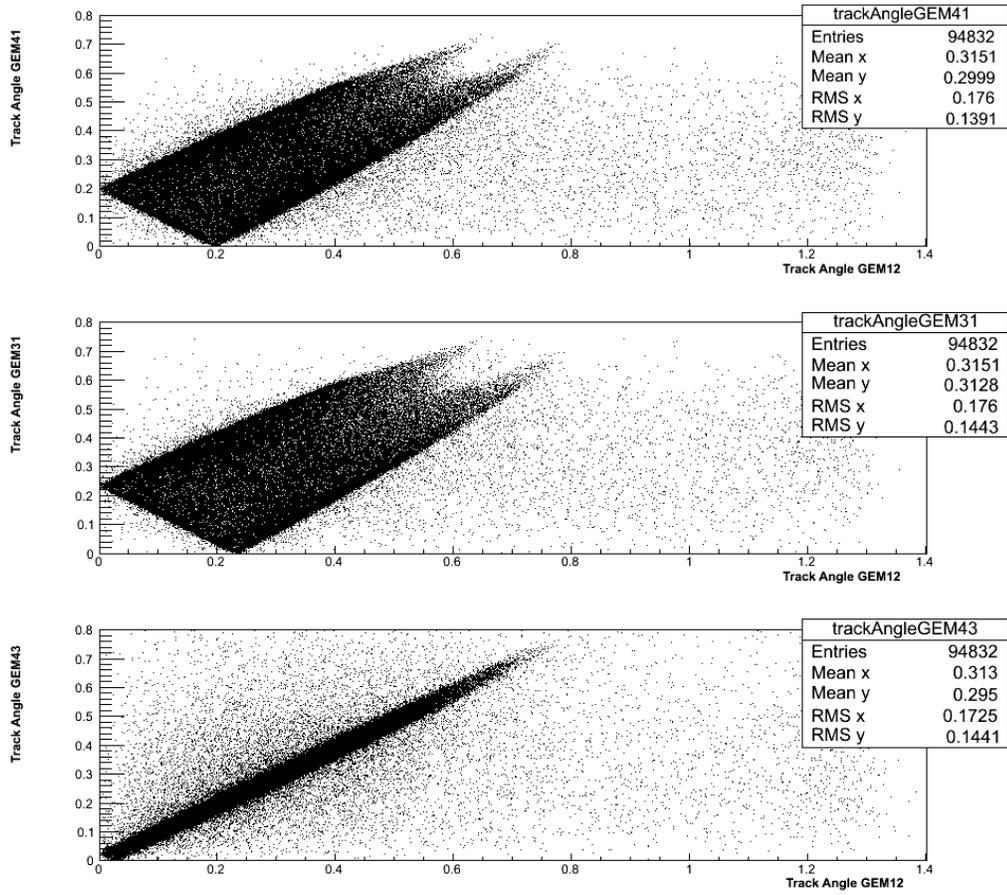


Figure 4.26: Correlation of track angles measured with pairs of top and bottom detectors before alignment.

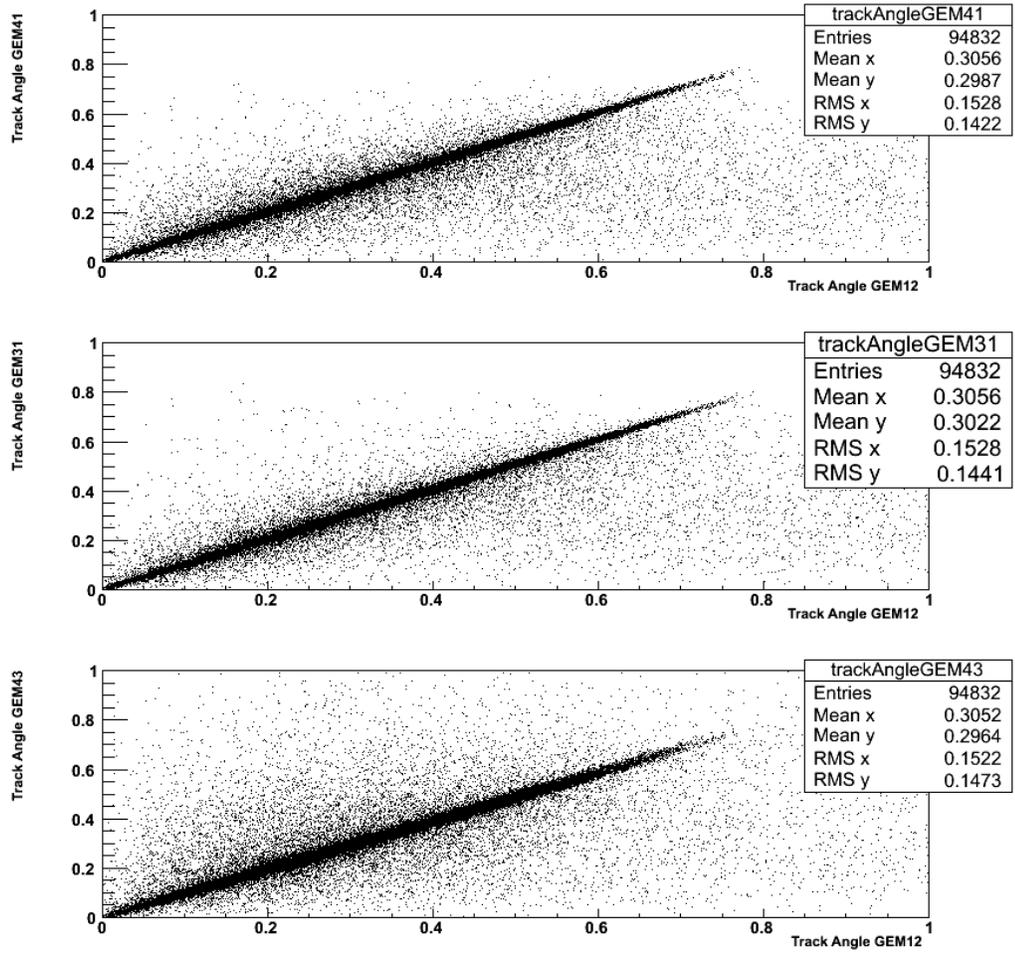


Figure 4.27: Correlation of track angles measured with pairs of top and bottom detectors after alignment.

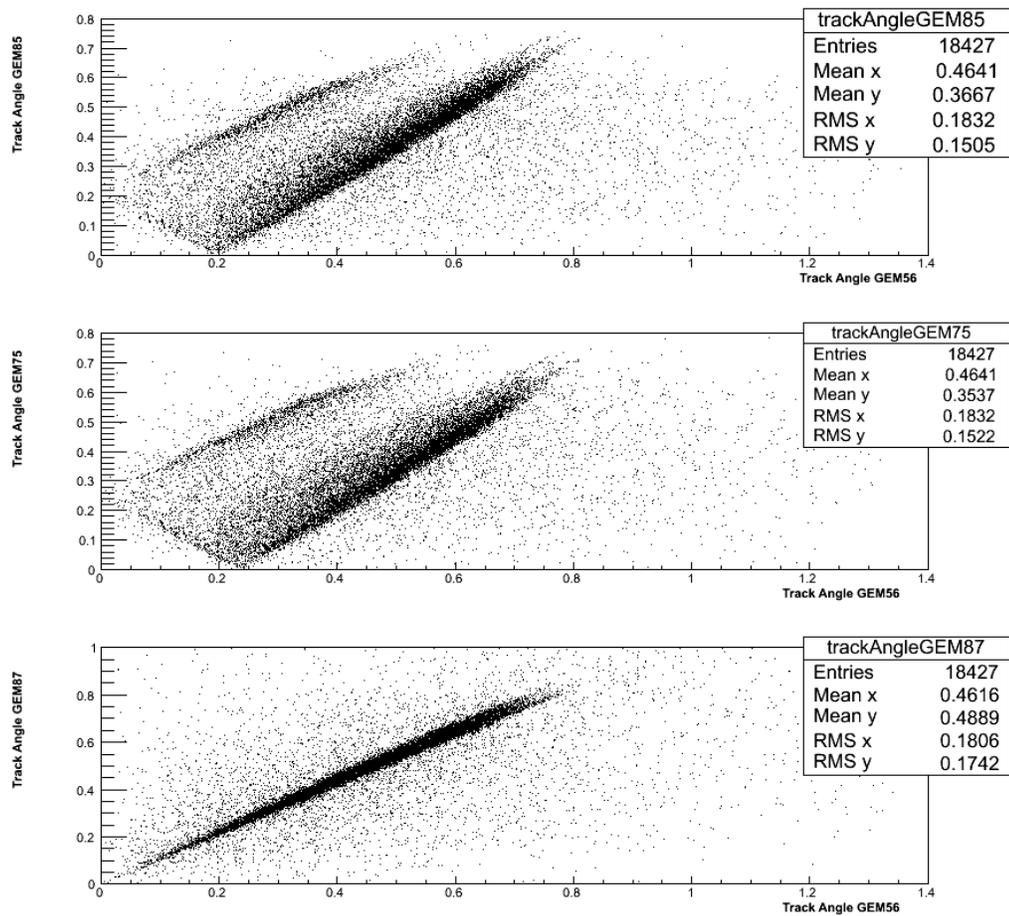


Figure 4.28: Correlation of track angles measured with pairs of left and right detectors before alignment.

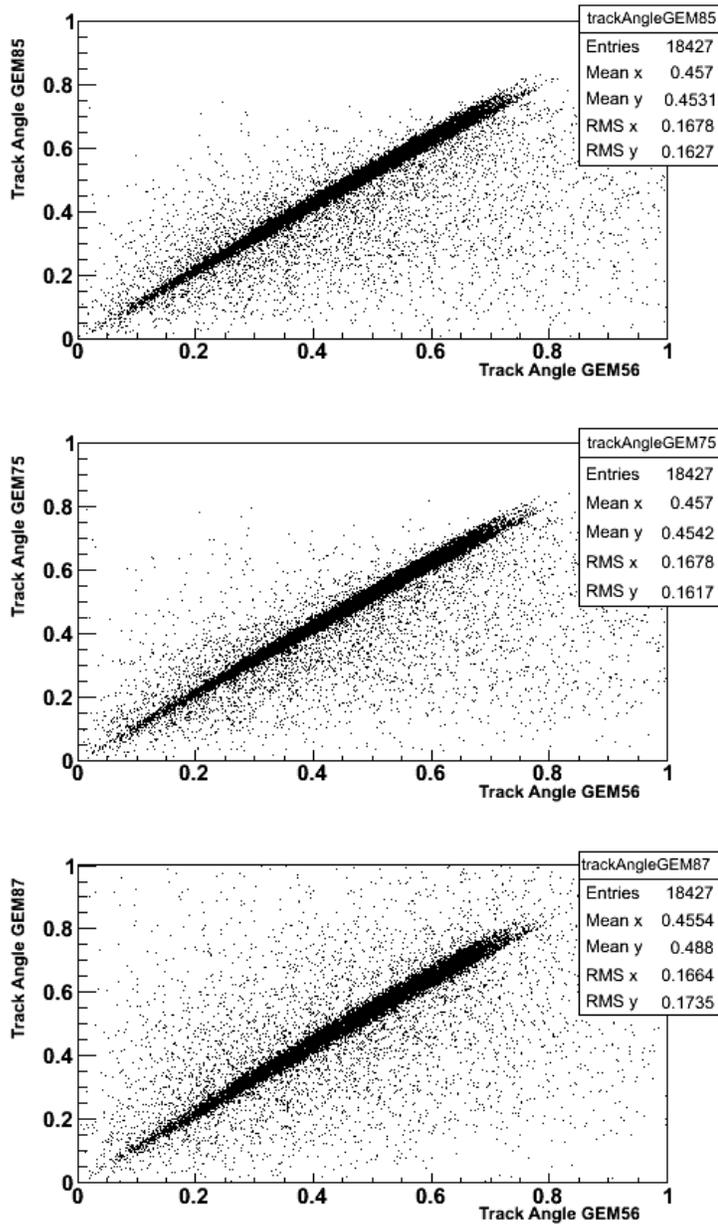


Figure 4.29: Correlation of track angles measured with pairs of left and right detectors after alignment.

Because of the uncertainties in the position and orientation of the detectors, in some of the residual distributions a big sigma or a double-peaked structure are

observed. This double-peaked structure arises from different contributions to the residuals from tracks through different combinations of the detector modules [26]. Also, GEM 5 and GEM 6 are not located completely vertical in the support structure holding the detectors. The PVC plates flex under the weight of some detectors and the flexibility of the material allows for an uncertainty in the position of detectors. The aluminum frame is quite thin and can cause misalignment due to the deformation of the frame.

As it was shown before, every detector has its own local coordinate. There is an argument that the axes in local coordinates are not vertical completely and cause misalignment. Final shift and rotation parameters can be found in the following tables (Tables 4.1- 4.4) .

Shift Parameters [mm]

GEM1X	GEM1Y	GEM2X	GEM2Y	GEM3X	GEM3Y	GEM4X	GEM4Y
-0.33	-45.29	0.54	-45.39	0.33	45.30	-0.56	45.40

Table 4.1: Final shift parameters for the TOP and BOTTOM GEMs in local coordinate.

Rotation Parameters [rad]

θ	GEM2	GEM3	GEM4
θ_x	0.0025	0.0031	0.00108
θ_y	0.0017	0.0032	0.00056
θ_{avg}	0.0021	0.0031	0.00082

Table 4.2: Final rotation parameters for the TOP and BOTTOM GEMs in local coordinate.

Shift Parameters [mm]

GEM5X	GEM5Z	GEM6X	GEM6Z	GEM7X	GEM7Z	GEM8X	GEM8Z
-1.68	-45.40	-1.77	-45.19	1.46	45.52	1.53	45.59

Table 4.3: Final shift parameters for the LEFT and RIGHT GEMs in local coordinate.

Rotation Parameters [rad]

θ	GEM6	GEM7	GEM8
θ_x	0.0051	0.038	0.043
θ_y	0.01	0.035	0.049
θ_{avg}	0.008	0.037	0.046

Table 4.4: Final rotation parameters for the LEFT and RIGHT GEMs in local coordinate.

4.4.3 Global Alignment

For the global alignment the two top GEMs are defined as a reference. As it is shown in Figure 4.22, two points are selected and the track angle is calculated. In a specific range, X, Y and Z of the left and right detectors are shifted by the same amount and the angle difference is calculated. The goal is to find the points where the angle difference is zero. Figure 4.23 illustrates the smallest track angle mean which could be found after shifting the detectors in three directions at the same time. The final numbers where the mean angle difference is close to zero, can be found in table 4.5.

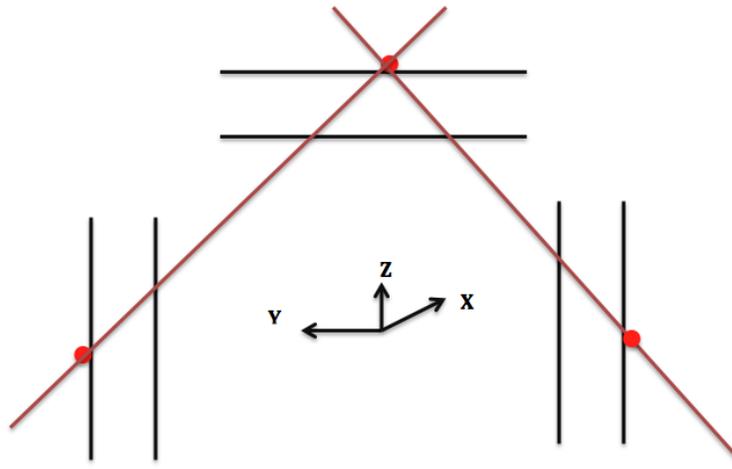


Figure 4.30: A schematic view of calculating track angle difference.

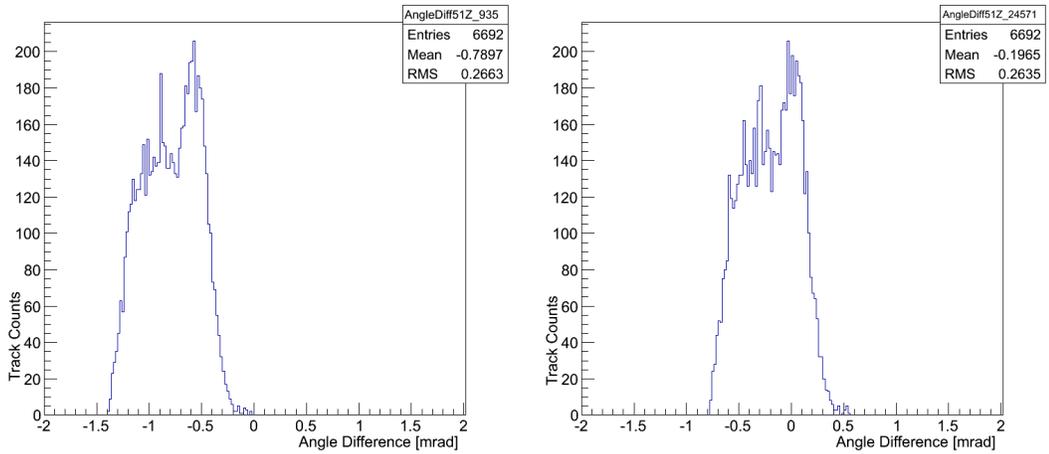


Figure 4.31: The track angle difference before (left) and after (right) shifting GEMs in 3 directions.

Shift Parameters [mm]		
X	Y	Z
5.8	-1.8	6.6

Table 4.5: Final shift parameters for the global alignment.

Chapter 5

Summary and Conclusions

5.1 Summary

The studies presented in this thesis have shown that the alignment is done precisely. The residual mean values are almost zero after iterative alignment. All the rotation angles are optimized and the chi square of the rotation angles have the parabolic trend.

The plots of track angles between GEM detectors are diagonal after alignment.

5.2 Future Work

One of the issues that has been mentioned in this thesis was the track selection. For getting a better results and fitted track, the number detectors can be increased.

The frame which is holding the detectors needs to be redesigned to keep the detectors completely vertical. For doing a global alignment it is recommended to use a 3D fit for finding the residuals.

To decrease time spent collecting and processing data, web base software Slow Control Run Initialization Byte-wise Environment (SCRIBE) is recommended. Using cluster for online and offline processing will make the analysis faster. By applying these changes, we hoped to achieve the full advantage of Gas Electron Multiplier detectors.

References

- [1] Los Alamos National Laboratory, “Tomographic Imaging with Cosmic Ray Muons ” ,2008.[Online].Available:
<https://www.princeton.edu/sgs/publications/sgs/archive/16-1-Morris.pdf>.
- [2] H. Bichsel, D. E. Groom and S. R. Klein, “Passage of Particles Through Matter,” *Journal of Physics G: Nuclear and Particle Physics*, vol. 37, no. 7A, pp. 285-299, 2010.
- [3] F. Sauli, “GEM: A new concept for electron amplification in gas detectors”, *Nucl. Inst. Meth. A386*, 531 (1997).
- [4] F. Sauli and A. Sharma, “Micropattern Gaseous Detectors,” *Annu. Rev. Nuck. Part. Sci.*, vol. 49, pp. 341-388, 1999.
- [5] F. Sauli, “Principles of Operation of Multiwire Proportional and Drift Chambers”, CERN 77-09 (1977).
- [6] F.Murtas, “ Applications of triple GEM detectors beyond particle and nuclear physics”, 2014 JINST 9 C01058
- [7] M. Altunbas, et al.,“Aging measurements with the Gas Electron Multiplier (GEM),” *Nuclear Instruments and Methods in Physics Research A*, no. 515, pp. 249-254, 2003.

- [8] B. Ketzer, Q. Weitzel, S. Paul, F. Sauli and L. Ropelewski, "Performance of triple GEM tracking detectors in the COMPASS experiment," *Nuclear Instruments and Methods in Physics Research A*, vol. 535, pp. 314-318, 2004.
- [9] F. Murtas, "Development of a gaseous detector based on GEM Technology," 28 November 2002. [Online]. Available:
http://www.lnf.infn.it/seminars/talks/murtas_28-11-02.pdf
- [10] C. Altunbas et al., "Construction, Test and Commissioning of the Triple GEM Tracking Detector for COMPASS," *Nucl. Instrum. Meth. A*, vol. 490, pp. 177-203, 2002.
- [11] A. Q. Segovia, "Construction and Performance of Gas Electron Multiplier Detectors for Nuclear Contraband Detection using Muon Tomography," Master's Thesis, Florida Institute of Technology, 2010.
- [12] B. Ketzer, Q. Weitzel, S. Paul, F. Sauli and L. Ropelewski, "Performance of triple GEM tracking detectors in the COMPASS experiment," *Nuclear Instruments and Methods in Physics Research A*, vol. 535, pp. 314-318, 2004.
- [13] RD51 Collaboration, "Development of micro-pattern gas detectors technologies," [Online]. Available: <http://rd51-public.web.cern.ch/rd51-public/>
- [14] The ALICE Collaboration, "The ALICE experiment at the CERN LHC," *Journal of Instrumentation*, vol. 3, no. S08002, 2008.
- [15] "ALICE DAQ and ECS Manual," December 2010. [Online]. Available: <https://edms.cern.ch/document/1056364/>.
- [16] Deep submicron technologies for HEP, A. Marchioro, Proceedings of 4th workshop on electronics for LHC experiments, CERN/LHCC/98-36, 40-46
- [17] S. Martoiu, CERN, "SRS Front_END Components". [Online]. Available: <http://indico.cern.ch/event/142994/contributions/166255/attachments/>

130530/185329/SRS_Front-end.pdf

[18] V. Gezer, “A LabVIEW based Test and Hardware Configuration System for SRS.” [Online]. Available:

https://indico.cern.ch/event/176664/contributions/1442116/attachments/229610/321235/9_Gezer_srtcs.pdf

[19] S. Martoiu, H. Muller and J. Toledo, “Front-End Electronics for the Scalable Readout System of RD51”, IEEE Nucl. Sci. Symp. Conf. Rec., Valencia, October 2011.

[20] J. Toledo, et al., “The Front-End Concentrator card for the RD51 Scalable Readout System,” in Topical Workshop on Electronics for Particle Physics, Vienna, 2011.

[21] J. Rylander, et al., “Quarknet Cosmic Ray Muon Detector User’s Manual Series ”6000” DAQ,” January 2010. [Online]. Available:

<http://www18.i2u2.org/cosmic/library/upload/b/ba/6000CRMDUserManual.pdf>.

[22] The ALICE Collaboration, “The ALICE experiment at the CERN LHC,” Journal of Instrumentation, vol. 3, no. S08002, 2008.

[23] CERN ALICE DAQ group, “ALICE DATE Users Guide, DATE 3.7”, ALICE Internal Note/DAQ ALICE-INT-2000-31 v.2 (2001).

[24] M. Hohlmann, et al., “Design and Performance of the Alignment System for the CMS Muon Endcaps,” in IEEE Nuclear Science Symposium Conference Record, San Diego, 2006.

[25] T. Lampen, “General alignment concept of the CMS,” Nuclear Instruments and Methods in Physics Research A, vol. 566, pp. 100-103, 2006.

[26] M. Staib, “Muon Tomography for Detection of Shielded Nuclear Contraband Using GEM Detectors,” Master’s Thesis, Florida Institute of Technology, 2012.

Appendix A

Directory of Data Collection Used for Alignment

The raw data used for alignment are found in the below directories:

Date PC:

home/MTSUser/MTSdata

amore PC:

home/SRSUser/afrouzdata/MTSdata

nas1:

mnt/nas1/DATE_Afrouz_bkp_MTSdata

Appendix B

Location of APVs Based on the Local Coordinate

The table presented below shows the exact location of every APV based on the local coordinate.

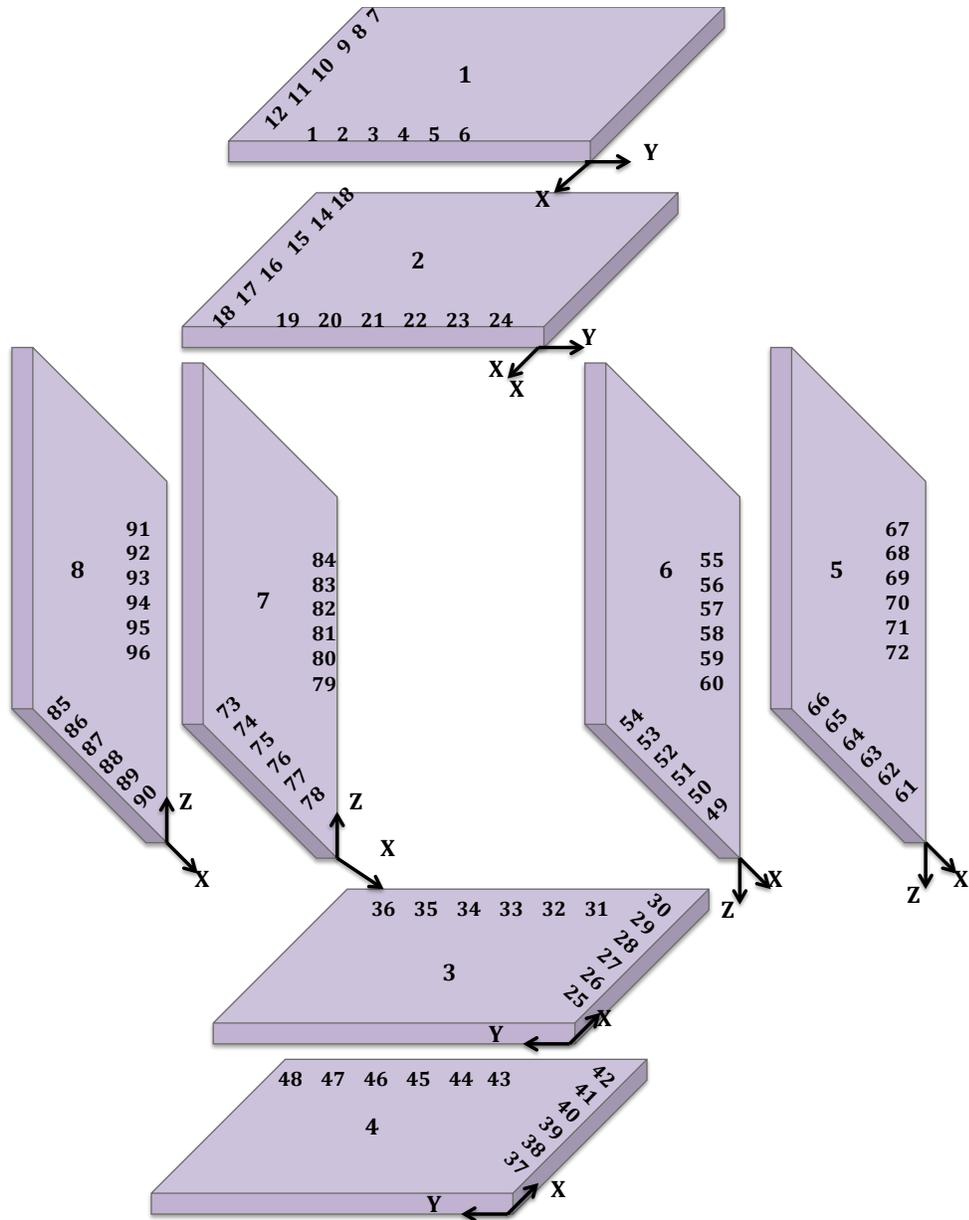


Figure B.1: The schematic view of the position of APVs on GEM detectors based on their numbers.

Location of APVs				
APV#	GEM#	Axis	FEC#	CHANNEL#
1	1	X	1	0
2	1	X	1	1
3	1	X	1	2
4	1	X	1	3
5	1	X	1	4
6	1	X	1	5
7	1	Y	1	6
8	1	Y	1	7
9	1	Y	1	8
10	1	Y	1	9
11	1	Y	1	10
12	1	Y	1	11
13	2	X	1	12
14	2	X	1	13
15	2	X	1	14
16	2	X	1	15
17	2	X	2	0
18	2	X	2	1
19	2	Y	2	2
20	2	Y	2	3
21	2	Y	2	4
22	2	Y	2	5
23	2	Y	2	6
24	2	Y	2	7
25	3	X	2	8
26	3	X	2	9
27	3	X	2	10
28	3	X	2	11
29	3	X	2	12
30	3	X	2	13
31	3	Y	2	14
32	3	Y	2	15
33	3	Y	3	0
34	3	Y	3	1
35	3	Y	3	2
36	3	Y	3	3
37	4	X	3	4
38	4	X	3	5

Table B.1: Location of APVs Based on the Local Coordinate.

Location of APVs				
APV#	GEM#	Axis	FEC#	CHANNEL#
39	4	X	3	6
40	4	X	3	7
41	4	X	3	8
42	4	X	3	9
43	4	Y	3	10
44	4	Y	3	11
45	4	Y	3	12
46	4	Y	3	13
47	4	Y	3	14
48	4	Y	3	15
49	5	X	4	0
50	5	X	4	1
51	5	X	4	2
52	5	X	4	3
53	5	X	4	4
54	5	X	4	5
55	5	Z	4	6
56	5	Z	4	7
57	5	Z	4	8
58	5	Z	4	9
59	5	Z	4	10
60	5	Z	4	11
61	6	X	4	12
62	6	X	4	13
63	6	X	4	14
64	6	X	4	15
65	6	X	5	0
66	6	X	5	1
67	6	Z	5	2
68	6	Z	5	3
69	6	Z	5	4
70	6	Z	5	5
71	6	Z	5	6
72	6	Z	5	7
73	7	X	5	8
74	7	X	5	9
75	7	X	5	10

Table B.2: Location of APVs Based on the Local Coordinate.

Location of APVs				
APV#	GEM#	Axis	FEC#	CHANNEL#
76	7	X	5	11
77	7	X	5	12
78	7	X	5	13
79	7	Z	5	14
80	7	Z	5	15
81	7	Z	6	0
82	7	Z	6	1
83	7	Z	6	2
84	7	Z	6	3
85	8	X	6	4
86	8	X	6	5
87	8	X	6	6
88	8	X	6	7
89	8	X	6	8
90	8	X	6	9
91	8	Z	6	10
92	8	Z	6	11
93	8	Z	6	12
94	8	Z	6	13
95	8	Z	6	14
96	8	Z	6	15

Table B.3: Location of APVs Based on the Local Coordinate.

Appendix C

Script Used for Alignment

This script listed below written to align the GEM detectors. In order to run this code, a root session should be opened with the command “root -l” and enter “.L Shift_TB_GEMs.C” and then ”main()”.The doubleGausFit.C has been run under this script.

The EventSelection script has used for selecting events.

Lists of functions and scripts used for alignment

Script Name	Function Name
Shift_TB_GEMs.C	CaculateCosTheta1
Shift_TB_GEMs.C	CaculateCosTheta2
doubleGausFit.C	-

Table C.1: Lists of scripts and functions used in the scripts.

```

#include <vector.h>
#include "doubleGausFit_withHistParameter.C"

double CalculateCosTheta1(double x, double y, double delta);

void tracking(string thestring){
    string txtfilename = thestring + ".txt";
    string shiftHead = "RotationBack_shiftParameters_";
    string rotateHead = "RotationBack_angles_";
    string residualHead = "RotationBack_residuals_";
    string ResidualRHead="RotationBack_Residual_";
    string foutname = shiftHead+thestring+"_exclusive.txt";
    string fout1name = residualHead+thestring+"_exclusive.txt";
    string foutRotationName = rotateHead + thestring +
"_exclusive.txt";
    fstream fin(txtfilename.c_str(),ios::in);
    if(!fin){cout<<"file not read"<<endl; return;}
    else cout<<"processing "<<txtfilename<<endl;
    fstream fout(foutname.c_str(),ios::out);
    fstream fout1(fout1name.c_str(),ios::out);
    fstream fout2(foutRotationName.c_str(),ios::out);
    double pGEM1X=0.8585, pGEM1Y=14.19;
    double pGEM2X=1.671, pGEM2Y=21.92;
    double pGEM3X=1.917, pGEM3Y=-30.98;
    double pGEM4X=1.606, pGEM4Y=-24.19;
    vector<double> vpGEM1X; vector<double> vpGEM1Y;
    vector<double> vpGEM2X; vector<double> vpGEM2Y;
    vector<double> vpGEM3X; vector<double> vpGEM3Y;
    vector<double> vpGEM4X; vector<double> vpGEM4Y;

    Int_t nbLines=0;

while(fin>>pGEM1X>>pGEM1Y>>pGEM2X>>pGEM2Y>>pGEM3X>>pGEM3Y>>pGEM4X
>>pGEM4Y){
    vpGEM1X.push_back(pGEM1X); vpGEM1Y.push_back(pGEM1Y);
vpGEM2X.push_back(pGEM2X); vpGEM2Y.push_back(pGEM2Y);
    vpGEM3X.push_back(pGEM3X); vpGEM3Y.push_back(pGEM3Y);
vpGEM4X.push_back(pGEM4X); vpGEM4Y.push_back(pGEM4Y);
nbLines++;
    }
    fin.close();

    double shiGEM1X=0, shiGEM1Y=0;
    double shiGEM2X=0, shiGEM2Y=0;
    double shiGEM3X=0, shiGEM3Y=0;
    double shiGEM4X=0, shiGEM4Y=0;

    double aGEM2GEM1=0;
    double aGEM3GEM1=0;
    double aGEM4GEM1=0;

```

```

double tempGEM1X, tempGEM1Y, tempGEM2X, tempGEM2Y, tempGEM3X,
tempGEM3Y, tempGEM4X, tempGEM4Y;//, tempEta5;

    double meanGEM1X=0.0, meanGEM1Y=0.0;
    double meanGEM2X=0.0, meanGEM2Y=0.0;
    double meanGEM3X=0.0, meanGEM3Y=0.0;
    double meanGEM4X=0.0, meanGEM4Y=0.0;
    double meanAngleGEM2=0.0;
    double meanAngleGEM3=0.0;
    double meanAngleGEM4=0.0;

Int_t iterNb=0;
while(1){
    char rootfile[50];
    sprintf(rootfile, "_iter%i_inclusive.root", iterNb);
    string outputrootname=ResidualRHead+thestring+rootfile;
    TFile* f = new TFile(outputrootname.c_str(), "recreate");
    iterNb++;

char
nameRes1X[20]; sprintf(nameRes1X, "residualGEM1X_%i", iterNb); char
nameRes1Y[20]; sprintf(nameRes1Y, "residualGEM1Y_%i", iterNb);
    char
nameRes2X[20]; sprintf(nameRes2X, "residualGEM2X_%i", iterNb); char
nameRes2Y[20]; sprintf(nameRes2Y, "residualGEM2Y_%i", iterNb);
    char
nameRes3X[20]; sprintf(nameRes3X, "residualGEM3X_%i", iterNb); char
nameRes3Y[20]; sprintf(nameRes3Y, "residualGEM3Y_%i", iterNb);
    char
nameRes4X[20]; sprintf(nameRes4X, "residualGEM4X_%i", iterNb); char
nameRes4Y[20]; sprintf(nameRes4Y, "residualGEM4Y_%i", iterNb);

TH1F* residualGEM1X = new TH1F(nameRes1X, "", 160, -4, 4);
residualGEM1X->SetXTitle("Residual [mm]"); residualGEM1X->
SetYTitle("Frequency"); residualGEM1X->
SetLabelSize(0.045, "XY"); residualGEM1X->
SetTitleSize(0.045, "XY");
    TH1F* residualGEM1Y = new TH1F(nameRes1Y, "", 160, -4, 4);
residualGEM1Y->SetXTitle("Residual [mm]"); residualGEM1Y->
SetYTitle("Frequency"); residualGEM1Y->
SetLabelSize(0.045, "XY"); residualGEM1Y->
SetTitleSize(0.045, "XY");
    TH1F* residualGEM2X = new TH1F(nameRes2X, "", 160, -4, 4);
residualGEM2X->SetXTitle("Residual [mm]"); residualGEM2X->
SetYTitle("Frequency"); residualGEM2X->
SetLabelSize(0.045, "XY"); residualGEM2X->
SetTitleSize(0.045, "XY");
    TH1F* residualGEM2Y = new TH1F(nameRes2Y, "", 160, -4, 4);
residualGEM2Y->SetXTitle("Residual [mm]"); residualGEM2Y->

```

```

>SetYTitle("Frequency");residualGEM2Y-
>SetLabelSize(0.045,"XY");residualGEM2Y-
>SetTitleSize(0.045,"XY");
    TH1F* residualGEM3X = new TH1F(nameRes3X,"",160,-4,4);
residualGEM3X->SetXTitle("Residual [mm]"); residualGEM3X-
>SetYTitle("Frequency");residualGEM3X-
>SetLabelSize(0.045,"XY");residualGEM3X-
>SetTitleSize(0.045,"XY");
    TH1F* residualGEM3Y = new TH1F(nameRes3Y,"",160,-4,4);
residualGEM3Y->SetXTitle("Residual [mm]"); residualGEM3Y-
>SetYTitle("Frequency");residualGEM3Y-
>SetLabelSize(0.045,"XY");residualGEM3Y-
>SetTitleSize(0.045,"XY");
    TH1F* residualGEM4X = new TH1F(nameRes4X,"",160,-4,4);
residualGEM4X->SetXTitle("mm"); residualGEM4X-
>SetYTitle("Frequency");residualGEM4X-
>SetLabelSize(0.045,"XY");residualGEM4X-
>SetTitleSize(0.045,"XY");
    TH1F* residualGEM4Y = new TH1F(nameRes4Y,"",160,-4,4);
residualGEM4Y->SetXTitle("mm"); residualGEM4Y-
>SetYTitle("Frequency");residualGEM4Y-
>SetLabelSize(0.045,"XY");residualGEM4Y-
>SetTitleSize(0.045,"XY");

TH1F* angleGEM2 = new TH1F("angleGEM2","Rotation angle
distribution of GEM2 and GEM1",1000,-0.5,0.5); angleGEM2-
>SetXTitle("Angle [radian]"); angleGEM2->SetYTitle("Frequency");
TH1F* angleGEM3 = new TH1F("angleGEM3","Rotation angle
distribution of GEM3 and GEM1",1000,-0.5,0.5); angleGEM3-
>SetXTitle("Angle [radian]"); angleGEM3->SetYTitle("Frequency");
TH1F* angleGEM4 = new TH1F("angleGEM4","Rotation angle
distribution of GEM4 and GEM1",1000,-0.5,0.5); angleGEM4-
>SetXTitle("Angle [radian]"); angleGEM4->SetYTitle("Frequency");

fout<<"shift:
"<<shiGEM1X<<"\t"<<shiGEM1Y<<"\t"<<shiGEM2X<<"\t"<<shiGEM2Y<<"\t"
<<shiGEM3X<<"\t"<<shiGEM3Y<<"\t"<<shiGEM4X<<"\t"<<shiGEM4Y<<endl;
    fout2<<"rotation:
"<<aGEM2GEM1<<"\t"<<aGEM3GEM1<<"\t"<<aGEM4GEM1<<endl;
    int nnnn=0;
    for(Int_t i=0;i<vpGEM1X.size();i++){
        //shift
        vpGEM1X[i] = vpGEM1X[i] - shiGEM1X; vpGEM1Y[i] = vpGEM1Y[i]
- shiGEM1Y;
        vpGEM2X[i] = vpGEM2X[i] - shiGEM2X; vpGEM2Y[i] = vpGEM2Y[i]
- shiGEM2Y;
        vpGEM3X[i] = vpGEM3X[i] - shiGEM3X; vpGEM3Y[i] = vpGEM3Y[i]
- shiGEM3Y;
        vpGEM4X[i] = vpGEM4X[i] - shiGEM4X; vpGEM4Y[i] = vpGEM4Y[i]
- shiGEM4Y;
    }

```

```

tempGEM1X=vpGEM1X[i]; tempGEM1Y=vpGEM1Y[i];
tempGEM2X=vpGEM2X[i]; tempGEM2Y=vpGEM2Y[i];
tempGEM3X=vpGEM3X[i]; tempGEM3Y=vpGEM3Y[i];
tempGEM4X=vpGEM4X[i]; tempGEM4Y=vpGEM4Y[i];

TGraph* g1 = new TGraph();
g1->SetPoint(0,0, vpGEM1X[i]);
g1->SetPoint(1,-75.5, vpGEM2X[i]);
g1->SetPoint(2,-382.0, vpGEM3X[i]);
g1->SetPoint(3,-457.5, vpGEM4X[i]);
TF1* f1 = new TF1("line1","[0]+[1]*x");
g1->Fit("line1","Q");
double intercept1 = f1->GetParameter(0);
double slope1 = f1->GetParameter(1);
double MeasuredGEM1X = intercept1 + slope1*0.0;
double MeasuredGEM2X = intercept1 + slope1*(-75.5);
double MeasuredGEM3X = intercept1 + slope1*(-382.0);
double MeasuredGEM4X = intercept1 + slope1*(-457.5);
residualGEM1X->Fill(MeasuredGEM1X-vpGEM1X[i]);
residualGEM2X->Fill(MeasuredGEM2X-vpGEM2X[i]);
residualGEM3X->Fill(MeasuredGEM3X-vpGEM3X[i]);
residualGEM4X->Fill(MeasuredGEM4X-vpGEM4X[i]);
delete f1; delete g1;

TGraph* g2 = new TGraph();
g2->SetPoint(0,0, vpGEM1Y[i]);
g2->SetPoint(1,-75.5, vpGEM2Y[i]);
g2->SetPoint(2,-382.0, vpGEM3Y[i]);
g2->SetPoint(3,-457.5, vpGEM4Y[i]);

TF1* f2 = new TF1("line2","[0]+[1]*x");
g2->Fit("line2","Q");
double intercept2 = f2->GetParameter(0);
double slope2 = f2->GetParameter(1);
double MeasuredGEM1Y = intercept2 + slope2*0.0;
double MeasuredGEM2Y = intercept2 + slope2*(-75.5);
double MeasuredGEM3Y = intercept2 + slope2*(-382.0);
double MeasuredGEM4Y = intercept2 + slope2*(-457.5);

residualGEM1Y->Fill(MeasuredGEM1Y-vpGEM1Y[i]);
residualGEM2Y->Fill(MeasuredGEM2Y-vpGEM2Y[i]);
residualGEM3Y->Fill(MeasuredGEM3Y-vpGEM3Y[i]);
residualGEM4Y->Fill(MeasuredGEM4Y-vpGEM4Y[i]);
delete f2; delete g2;

double cosineGEM2 =
CalculateCosTheta1(vpGEM1X[i],vpGEM1Y[i],vpGEM2X[i],vpGEM2Y[i]);
double cosineGEM3 =
CalculateCosTheta1(vpGEM1X[i],vpGEM1Y[i],vpGEM3X[i],vpGEM2Y[i]);
double cosineGEM4 =

```

```

CalculateCosTheta1(vpGEM1X[i],vpGEM1Y[i],vpGEM4X[i],vpGEM2Y[i]);
    angleGEM2->Fill(cosineGEM2);
    angleGEM3->Fill(cosineGEM3);
    angleGEM4->Fill(cosineGEM4);

nnnn++;
}
gStyle->SetOptFit(1111);

    I2GFvalues myValues;
    myValues = I2GFmainLoop(residualGEM1X, 1, 10 , 1);
    meanGEM1X = myValues.mean; sigmaGEM1X=myValues.sigma;
    myValues = I2GFmainLoop(residualGEM1Y, 1, 10 , 1);
    meanGEM1Y = myValues.mean; sigmaGEM1Y=myValues.sigma;
    myValues = I2GFmainLoop(residualGEM2X, 1, 10 , 1);
    meanGEM2X = myValues.mean; sigmaGEM2X=myValues.sigma;
    myValues = I2GFmainLoop(residualGEM2Y, 1, 10 , 1);
    meanGEM2Y = myValues.mean; sigmaGEM2Y=myValues.sigma;
    myValues = I2GFmainLoop(residualGEM3X, 1, 10 , 1);
    meanGEM3X = myValues.mean; sigmaGEM3X=myValues.sigma;
    myValues = I2GFmainLoop(residualGEM3Y, 1, 10 , 1);
    meanGEM3Y = myValues.mean; sigmaGEM3Y=myValues.sigma;
    myValues = I2GFmainLoop(residualGEM4X, 1, 10 , 1);
    meanGEM4X = myValues.mean; sigmaGEM4X=myValues.sigma;
    myValues = I2GFmainLoop(residualGEM4Y, 1, 10 , 1);
    meanGEM4Y = myValues.mean; sigmaGEM4Y=myValues.sigma;

cout<<"residual mean:
"<<meanGEM1X<<"\t"<<meanGEM1Y<<"\t"<<meanGEM2X<<"\t"<<meanGEM2Y<<
"\t"<<meanGEM3X<<"\t"<<meanGEM3Y<<"\t"<<meanGEM4X<<"\t"<<meanGEM4
Y<<endl;
//<<meanZZ1<<"\t"<<meanZZ2<<endl;
    fout1<<"residual mean:
"<<meanGEM1X<<"\t"<<meanGEM1Y<<"\t"<<meanGEM2X<<"\t"<<meanGEM2Y<<
"\t"<<meanGEM3X<<"\t"<<meanGEM3Y<<"\t"<<meanGEM4X<<"\t"<<meanGEM4
Y<<endl;

    myValues = I2GFmainLoop(angleGEM2, 1, 10 , 1);
    meanAngleGEM2=myValues.mean;
    myValues = I2GFmainLoop(angleGEM3, 1, 10 , 1);
    meanAngleGEM3=myValues.mean;
    myValues = I2GFmainLoop(angleGEM4, 1, 10 , 1);
    meanAngleGEM4=myValues.mean;
    f->Write();
    f->Close();

double factor = -0.2;
shiGEM1X = meanGEM1X*factor; shiGEM1Y = meanGEM1Y*factor;
shiGEM2X = meanGEM2X*factor; shiGEM2Y = meanGEM2Y*factor;
shiGEM3X = meanGEM3X*factor; shiGEM3Y = meanGEM3Y*factor;

```

```

        shiGEM4X = meanGEM4X*factor; shiGEM4Y = meanGEM4Y*factor;
        aGEM2GEM1 = 0; aGEM3GEM1 = 0;
        aGEM4GEM1 = 0;

        if(iterNb==50) break;
    }
    fout.close();
    fout1.close();
    fout2.close();
}

void main(){
    string name[1]={
        "File Name"
    };
    for(int i=0;i<1;i++) tracking(name[i]);
}

double CalculateCosTheta1(double x, double y, double x1,double
y1){
    double sineTheta = (x1*y - x*y1)/(x*x + y*y);
    return asin(sineTheta);
}

double CalculateCosTheta2(double x, double y, double delta){
    double cosineTheta = ( y*(y+delta) - x*sqrt(x*x-2.0*y*delta-
delta*delta) )/(x*x+y*y);
    return acos(cosineTheta);
}

```

```

#include <vector>
#include <iostream>
#include <string>
using namespace std;
#include "TR00T.h"
#include "TFitResult.h"
#include "TCanvas.h"
#include "TDateTime.h"
#include "TRandom.h"
#include "TStyle.h"
#include "TLegend.h"
#include "TLatex.h"
#include "TH1.h"
#include "TH2.h"
#include "TF1.h"
#include "TH1F.h"
#include "TGraph.h"
#include "TGraphErrors.h"
#include "TMarker.h"
#include "TString.h"
#include "TFile.h"
#include "TMath.h"
#include "TLine.h"
#include "TBranch.h"
#include "TTree.h"
#include "TPad.h"
#include "TChain.h"
#include "TSystem.h"
#include "TKey.h"
#include <cmath>
#include "TSpectrum.h"

struct I2GFvalues
{
    float rchi2;
    float mean;
    float mean_err;
    float sigma;
    float sigma_err;
    TF1 *fit_func;
};

I2GFvalues I2GFmainLoop(TH1F *htemp, int N_iter, float
N_sigma_range, bool ShowFit)
{
    I2GFvalues myI2GFvalues;

    myI2GFvalues.rchi2 = -100;
    myI2GFvalues.mean = -100;
    myI2GFvalues.mean_err = -100;

```

```

myI2GFvalues.sigma = -100;
myI2GFvalues.sigma_err = -100;

TSpectrum *s = new TSpectrum();
Int_t NPeaks;
Float_t *Peak; Float_t *PeakAmp;
float peak_pos = 0;
float peak_pos_amp = 0; int peak_pos_bin = 0;

int binMaxCnt = 0;
float binMaxCnt_value = 0;
int binMaxCnt_counts = 0;

int Nbins = 0;
Int_t zero_value_bin = 0;
float low_limit = 0;
float high_limit = 0;
float peak_pos_count = 0;
float zero_bin_value = 0;
float max_bin_value = 0;

TF1 *func;
TF1 *func1;
TF1 *func2;
TF1 *func3;
float Chi2;
int NDF = 1;

float f_RChi2 = 1;
float f_const = 1;
float f_mean = 1;
float f_mean_err = 1;
float f_sigma = 1;
float f_sigma_err = 1;

float peak = 1;

float f_const2 = 1;
float f_mean2 = 1;
float f_mean_err2 = 1;
float f_sigma2 = 1;
float f_sigma_err2 = 1;

binMaxCnt = htemp->GetMaximumBin();binMaxCnt_value = (Double_t)
htemp->GetXaxis()->GetBinCenter(binMaxCnt); binMaxCnt_counts =
(Int_t) htemp->GetBinContent(binMaxCnt);

```

```

if (ShowFit) NPeaks = s->Search(htemp, 2, "goff", 0.5);
Peak = s->GetPositionX();
PeakAmp = s->GetPositionY();

for (int i=0; i<NPeaks; i++)
{
    if (peak_pos_amp < PeakAmp[i])
    {
        peak_pos_amp = PeakAmp[i];        peak_pos = Peak[i];
    }
}
peak_pos_bin = htemp->GetXaxis()->FindBin(peak_pos);
peak_pos_count = htemp->GetBinContent(peak_pos_bin);
zero_value_bin = htemp->GetXaxis()->FindBin(0.0); Nbins =
htemp->GetSize() - 2;
zero_bin_value = htemp->GetXaxis()->GetBinCenter(0);
max_bin_value = htemp->GetXaxis()->GetBinCenter(Nbins);
int TS = 0;
if (peak_pos >= zero_bin_value && peak_pos <= max_bin_value)
{
    TS=1;
    low_limit = peak_pos - (0.1 * abs(max_bin_value-
zero_bin_value)); high_limit = peak_pos + (0.1 *
abs(max_bin_value-zero_bin_value));
    func = new TF1("func", "gaus");

    func->SetParameter(0, peak_pos_count);
    func->SetParameter(1, peak_pos);
}
else
{
    low_limit = binMaxCnt_value - (0.1 * abs(max_bin_value-
zero_bin_value));
    high_limit = binMaxCnt_value + (0.1 * abs(max_bin_value-
zero_bin_value));
    func = new TF1("func", "gaus");
    func->SetParameter(0, binMaxCnt_counts);
    func->SetParameter(1, binMaxCnt_value);
}

htemp->Fit("gaus", "Q0", "", low_limit, high_limit);
func = htemp->GetFunction("gaus");
Chi2 = func->GetChisquare();
NDF = func->GetNDF();
if (NDF != 0) f_RChi2 = Chi2/NDF;
f_const = func->GetParameter(0);
f_mean = func->GetParameter(1);
f_mean_err = func->GetParError(1);

```

```

    f_sigma = func->GetParameter(2);
    f_sigma_err = func->GetParError(2);
    for (int i=0; i< 2; i++) {
        htemp->Fit("gaus", "Q0", "", (f_mean -
(N_sigma_range*f_sigma)), (f_mean + (N_sigma_range*f_sigma) ) );
        func = htemp->GetFunction("gaus");
        Chi2 = func->GetChisquare();
        NDF = func->GetNDF();
        if (NDF != 0) f_RChi2 = Chi2/NDF;
        f_const = func->GetParameter(0);
        f_mean = func->GetParameter(1);
        f_mean_err = func->GetParError(1);
        f_sigma = func->GetParameter(2);
        cout << " sigma " << f_sigma << endl;
        f_sigma_err = func->GetParError(2);
    }

    peak = func->GetParameter(0);
    float bgd_h = 0.25;

    func1 = new TF1("func1", "gaus");
    func2 = new TF1("func2", "gaus");

    func3 = new TF1("func3", "func1 + func2", (f_mean - 3*f_sigma),
(f_mean + 3*f_sigma) );
    func3->SetParameters(f_const, f_mean, f_sigma, f_const/10,
f_mean, 4*f_sigma);
    func3->SetParLimits(3, 0, (0.05*peak_pos_amp) );

    htemp->Fit("func3", "Q0");
    func = htemp->GetFunction("func3");
    Chi2 = func3->GetChisquare();
    NDF = func->GetNDF();
    if (NDF != 0) f_RChi2 = Chi2/NDF;
    f_const = func3->GetParameter(0);
    f_mean = func3->GetParameter(1);
    f_mean_err = func3->GetParError(1);
    f_sigma = func3->GetParameter(2);
    f_sigma_err = func3->GetParError(2);

    f_const2 = func3->GetParameter(3);
    f_mean2 = func3->GetParameter(4);
    f_mean_err2 = func3->GetParError(4);
    f_sigma2 = func3->GetParameter(5);
    f_sigma_err2 = func3->GetParError(5);
    func3->SetParNames("Primary Constant", "Primary Mean", "Primary
Sigma", "Background Constant", "Background Mean", "Background
Sigma");
    for (int j=0; j<4; j++)

```

```

    {
        func3->SetParameters(f_const, f_mean, f_sigma, f_const2,
f_mean2, f_sigma2);

        if (ShowFit) htemp->Fit("func3", "Q");
        else htemp->Fit("func3", "Q0");

        func3 = htemp->GetFunction("func3");
        func3->SetLineColor(2);
Chi2 = func3->GetChisquare();
NDF = func3->GetNDF();
        if (NDF != 0) f_RChi2 = Chi2/NDF;
        f_const = func3->GetParameter(0);
        f_mean = func3->GetParameter(1);
        f_mean_err = func3->GetParError(1);
        f_sigma = func3->GetParameter(2);
        f_sigma_err = func3->GetParError(2);
        f_const2 = func3->GetParameter(3);
        f_mean2 = func3->GetParameter(4);
        f_mean_err2 = func3->GetParError(4);
        f_sigma2 = func3->GetParameter(5);
        f_sigma_err2 = func3->GetParError(5);
    }

    if (abs(f_const-peak) < abs(f_const2-peak))
    {
        myI2GFvalues.rchi2 = f_RChi2;
        myI2GFvalues.mean = f_mean;
        myI2GFvalues.mean_err = f_mean_err;
        myI2GFvalues.sigma = abs(f_sigma);
        myI2GFvalues.sigma_err = f_sigma_err;
    }
    else
    {
        myI2GFvalues.rchi2 = f_RChi2;
        myI2GFvalues.mean = f_mean2;
        myI2GFvalues.mean_err = f_mean_err2;
        myI2GFvalues.sigma = abs(f_sigma2);
        myI2GFvalues.sigma_err = f_sigma_err2;
    }

    return myI2GFvalues;

    delete s;
    delete func;
    delete func1;
    delete func2;
    delete func3;
}

```

```

#include "TString.h"

int read(){

    fstream fin("input file",ios::in);
    fout("output file",ios::out);
    TString theA;
    string detName;
    int clusterSize;
    float totalQ;
    float centroid;
    float stripCentroid;
    int stripNo;
    float stripQ;

    int evtNb=0;

    float x[4], y[4];
    float z[4] = {0.0, -75.5, -382.0, -457.0};
    int evtFlag = -100;

while(fin.good()){
    fin>>theA;
    if(theA.EqualTo("EventNb")){
        fin>>evtNb;
        evtFlag = 0;
        for(int i=0; i<4;i++){x[i]=-1000; y[i]=-1000;}
        if(evtNb%500==0) cout<<theA.Data()<<"\t"<<evtNb<<endl;
    }
    else{
fin>>clusterSize>>totalQ>>centroid>>stripCentroid;
        for(int i = 0; i<clusterSize; i++) fin>>stripNo>>stripQ;

        for(int i=0;i<4;i++){
            TString tempX = Form("GEM%dX",i+5);
            TString tempY = Form("GEM%dY",i+5);

            if(theA.EqualTo(tempX)) {
                if(i==0 || i==1) x[i] = -1.0*centroid;
                else x[i] = centroid;
                evtFlag++;
            }

            if(theA.EqualTo(tempY)) {

                if(i==2 || i==3) y[i] = -1.0*centroid;
                else y[i] = centroid;
                evtFlag++;
            }
        }
    }
}
}

```

```
    }  
  }  
  if(evtFlag == 8){  
    for(int i=0;i<4;i++){  
      fout<<x[i]<<"\t"<<y[i]<<"\t";  
    }  
    fout<<endl;  
  
    rootfile->Close();  
  }  
}
```