
**JOINT INSTITUTE FOR NUCLEAR RESEARCH**
Veksler and Baldin laboratory of High Energy Physics

# Centrality estimation in the Fixed Target Mode at MPD experiment

**Supervisor**
Dr. Vadim Kolesnikov
Dr. Ivonne Alicia Maldonado Cervantes
Dr. Viktar Kireyeu
Dr. Natalia Kolomoyets
**Students**
Adrián Lara Tlaxcala
Francisco Reyes

August 2, 2024

# Contents

# 1 Introduction

The MPD experiment will start taking data in the fixed target configuration by mid−2025. It is therefore necessary to characterize the detector response and obtain some of the global variables such as centrality. In this report we show a preliminary analysis of the centrality classes obtained in collisions $Xe + W$ at $E_{lab} = 2.5GeV$ ($\sqrt{s_{AB}} = 2.9$ GeV) simulated with the event generator UrQMD. For the analysis we use the multiplicity distribution of tracks reconstructed by the TPC that meet certain selection criteria. We use the MC Glauber and GammaFit methods of centrality extraction supported by the CentralityFramework[1] software which has already implemented for the MPD experiment.

In the following sections we show the methodology used to accomplish this task. In the section 2 we describe the parameters used in UrQMD to generate $\approx 0.3M$ of events, also we show a preliminary set of cuts, both in the primary vertex and cuts used for track. In the section 3 we explain the parameters implemented to run the Centrality Framework within the NCX cluster[2]. And finally in section 4, the results obtained with both methods.

# 2 Event generation for Fixed Target Mode Configuration

For our analysis, we simulate 384,800 Xe (A = 124) + W collisions at $E_{lab} = 2.5$ GeV using UrQMD (Ultra Relativistic Quark model Dynamics)[3] as a Monte Carlo Event Generator with the following input:

```
pro 124 54          //Proyectile Atomic_mass Atomic_number
tar 184 74          //Target Atomic_mass Atomic_number

nev 200             //Number of Events
imp -14.71             //Impact Parameter
ene 2.5             //Kinetic Energy
tim 200 200         //Time

cto 27 1            //Target Mode Option

rsd 16537010       //Random Number
f13
#f14                //Output File
f15
f16
f19
f20

xxx
```

We configure the script runMC.C for Fixed Target (FXT) mode setting the variables as appears in the following box:

```
primGen->SetBeam(0.0, 0.0, 1e-6, 1e-6);
primGen->SetTarget(-85.0, 0.0);
primGen->SmearGausVertexZ(kFALSE);
primGen->SmearVertexXY(kFALSE);
```

Finally we process the events with the runReco.C script to obtain the mpddst.root files.

The mpddst.root files contain all the information about event and the reconstructed tracks.

### 2.0.1 Primary Vertex

As a first step we analyze the distribution of the z coordinate of the reconstructed primary vertex, shown in Figure 1.

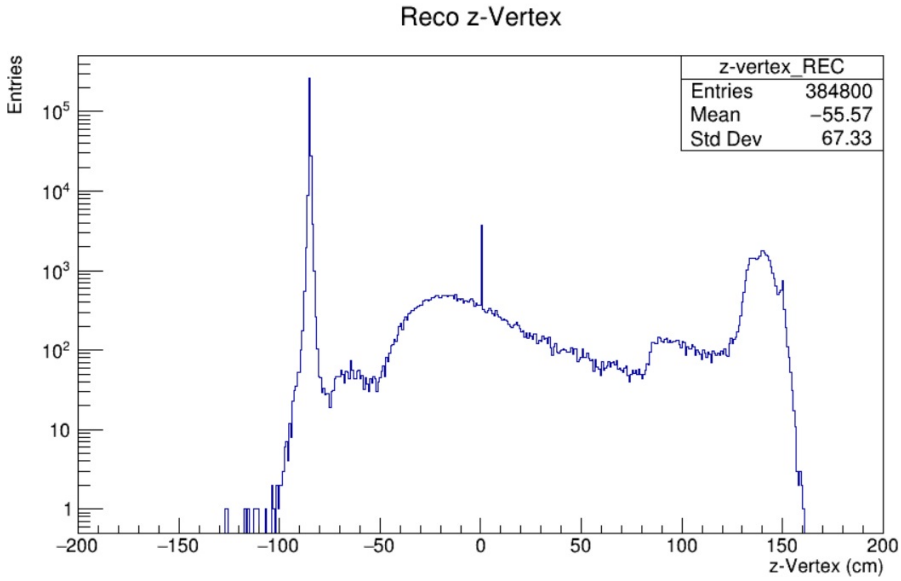



Figure 1: Distribution of z-coordinate of reconstructed primary vertex

The distribution has a peak at $z_{Vertex} = -85.0$ cm, however there are several events in the range -50 cm to 150 cm. To estimate the contribution of the events in this region, we calculate the percentage of events around the -85 cm peak. The percentage of events with $z_{vertex} \in (-100, -70)$ cm its 80.164% of the total events. To reject this events we apply a cut to the number of MCTracks from each event, for the analysis we select events with more than 308 tracks (124 from Xenon and 184 from Wolfram), so we ensure events with more particles than the initial ones. The distribution obtained is shown on Figure 2.

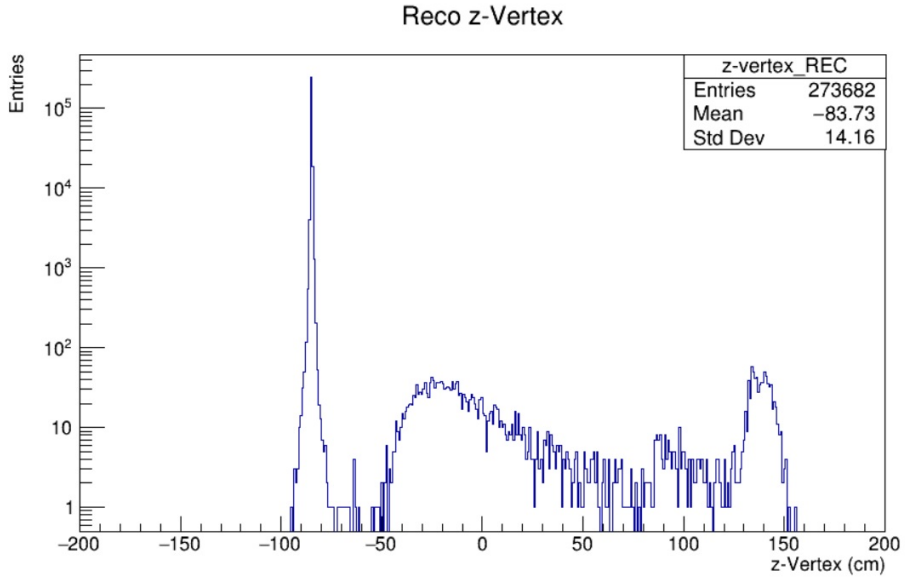



Figure 2: Primary z-Vertex distribution with number of MCTracks higher than 308.

Again we calculate the percentage of events around the -85 cm peak and now it is 98.95% of the total events. We can conclude that contribution of events on $z_{vertex} \in (-50, 150)$ cm its not relevant and can be ignored. Most of the rejected events correspond to peripheral events, as we can see in the distribution of the impact parameter in figure 3.

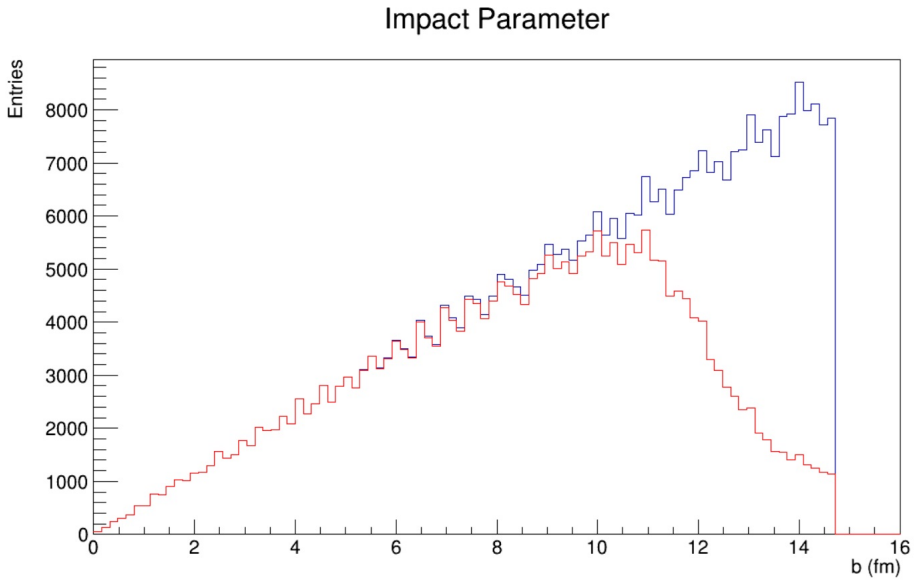



Figure 3: Distribution of the Impact parameter of the events without (blue) cut on the number of MCTracks and with cut (red) on MCTracks $> 308$.

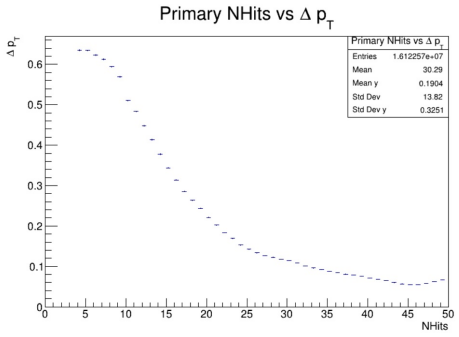
## 2.1 Track selection to get Multiplicity distribution

To run the centrality Framework we require the multiplicity distribution of reconstructed tracks. To select the tracks, we analyze the transverse momentum resolution

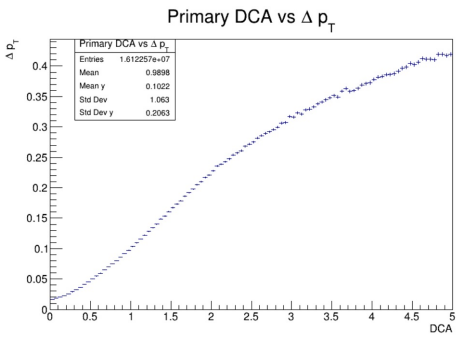
$$\Delta p_T = \frac{\left| p_T^{Reco} - p_T^{MC} \right|}{p_T^{MC}} \tag{1}$$

as a function of different variables used to select good reconstructed tracks, as the number of hits in the TPC (Time Projection Chamber), the pseudorapidity $\eta$ and the DCA (Distance of Closest Approach).

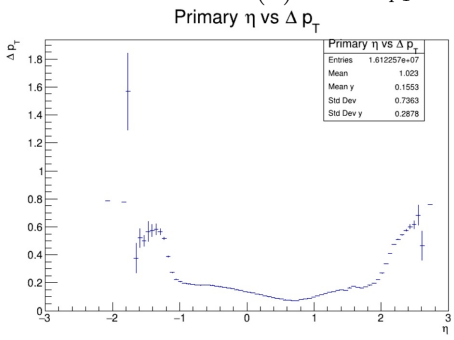
The distribution of $\Delta p_T$ is shown in the figures 4 and 5 for primary and secondary particles respectively. Besides the cut on pseudorapidity acceptance $-1 < eta < 2$ we can select events with the $NHits > 20$ and $DCA < 2.0$cm to have a transverse momentum resolution below 0.2 for both primary and secondary tracks. Further detailed analysis will be presented later.




(a) Number of Hits vs $p_T$ Resolution for Primary Particles.

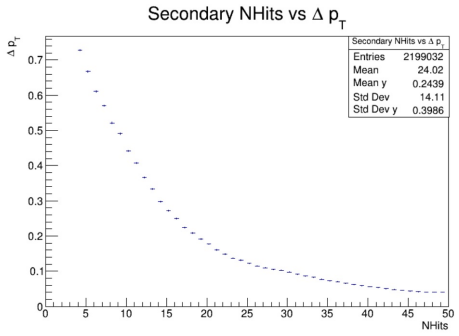



(b) DCA vs $p_T$ Resolution for Primary Particles.

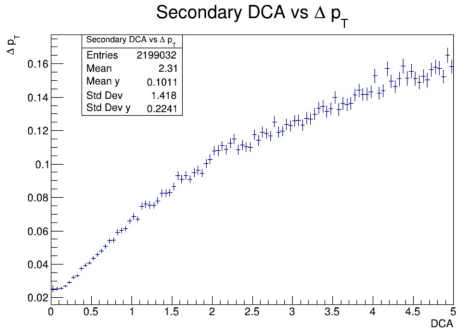



(c) $\eta$ vs $p_T$ Resolution for Primary Particles.

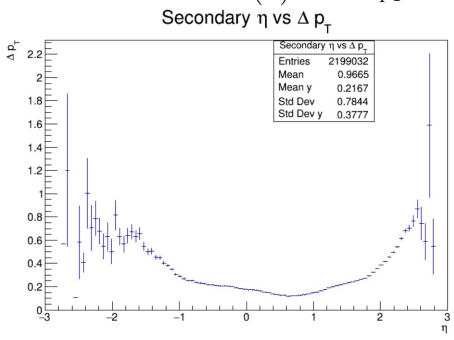
Figure 4: Primary Particles selected with MC association.




(a) Number of Hits vs $p_T$ Resolution for Secondary Particles.




(b) DCA vs $p_T$ Resolution for Secondary Particles.



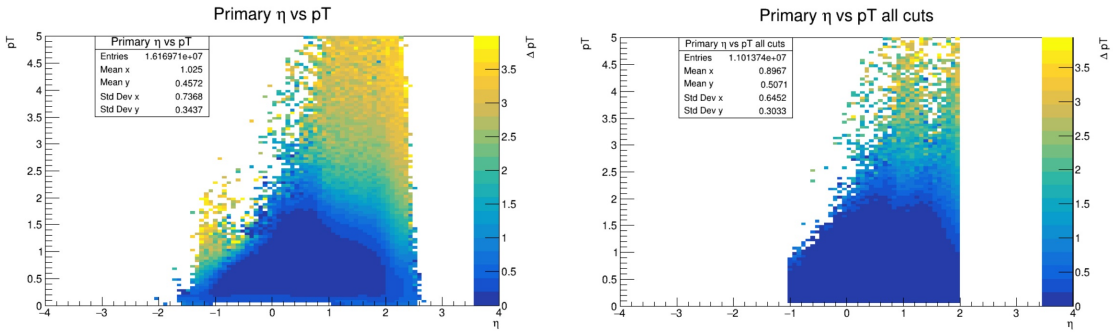

(c) $\eta$ vs $p_T$ Resolution for Secondary Particles.

Figure 5: Secondary Particles selected with MC association.

The selected cuts to have a transverse momentum resolution better than 20% are:

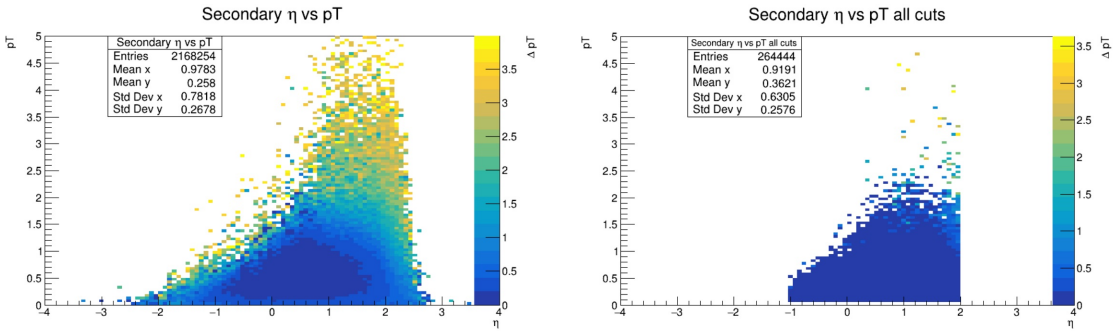
1. Number of Hits $\geq 20$
2. DCA $\leq 2$
3. $\eta \in$ [-1,2]

The comparison of the phase space distribution of reconstructed tracks with and without cuts on impact parameter, Number of Hits, DCA and $\eta$, is shown in the figures 6 and 7 for primary and secondary particles respectively. We observe that for primary particles the $p_T$ resolution is too big for particles with $p_T > 2$GeV/c.




(a) Without cuts (b) With cuts

Figure 6: Primary particles $\eta$ vs pT resolution.




(a) Without cuts (b) With cuts

Figure 7: Secondary particles $\eta$ vs pT resolution.

With the previous cuts we obtain the multiplicity distribution, shown in figure 8 that we are going to use to estimate the centrality classes of the events.

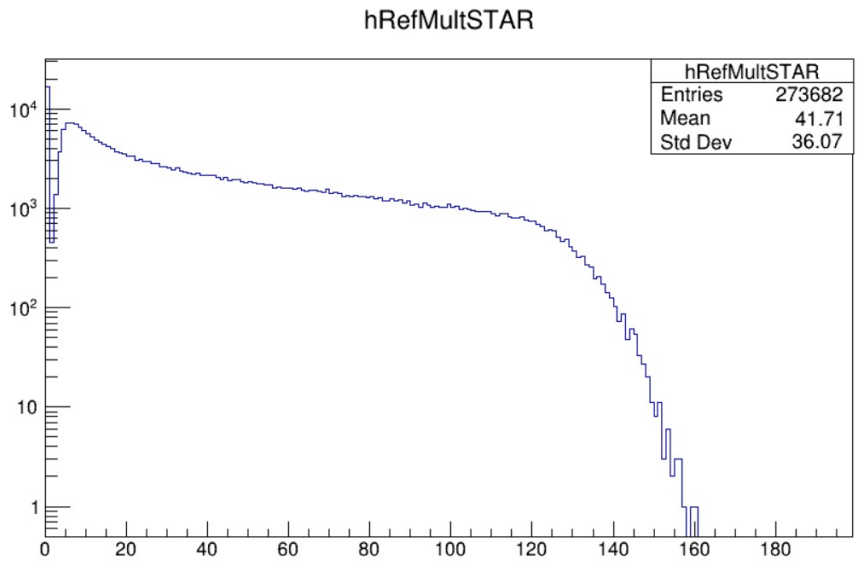



Figure 8: Multiplicity distribution cuts on b, Number of Hits, DCA and $\eta$.

# 3 Implementation of the methods with the Centrality Framework

## 3.1 Experimental information needed

For both of the methods it is needed to know:

- The nucleon nucleon Inelastic Cross Section $\sigma_{NN}$.
- Multiplicity distribution.

Formula and procedure to obtain inelastic cross section are in the appendix. From the calculation we obtain a result of $\sigma_{inel} = 26.94035$ mb.

## 3.2 Packages and Versions used

To run the Centrality Framework in the ncx cluster, we need to modify the macros from PBS to SLURM scripts and select a proper version of root, different from the one use at ref. [1], because it gave us problems when we run the scripts.

In the following subsections we will describe the steps followed.

## 3.3 Centrality Framework

### 3.3.1 Installation

The modules and root version used on the NCX Cluster to run the Framework, are the following:

```
source /cvmfs/nica.jinr.ru/sw/os/login.sh
module add ROOT/v6.32.02-1
module add OpenSSL/v3.2.1-1
module add CMake/v3.28.3-1
```

To download Centrality Framework we used the following lines of code:

```
git clone https://github.com/FlowNICA/CentralityFramework.git
cd CentralityFramework/Framework/McGaluber/centrality-master/
mkdir build/ && cd build/
cmake ..
make
```

## 3.4 GammaFit

For the use of GammaFit from the Framework software is necessary to use MpdDstReader.C macro which allows us to get in a .root file a Multiplicity Histogram. The process data for the analysis is stored in the following route:

```
/scratch1/maldonado/FXT/SIM_85_XeW_2.5GeV/
```

For running the MpdDstReader.C macro you need to follow the following steps, keeping in mind that you must have loaded the relevant root version before using MPDRoot:

```
root -l
root [0] .L MpdDstReader.C
root [1] TChain *sim=new TChain("mpdsim");
root [2] for(Int_t j=1;j<21;++j)sim->AddFile(Form("mpddst%d.root",j));
root [3] MpdDstReader(sim,"outfile.root");
```

The file returns a histogram with the name: hRefMultSTAR that can be read by the GammaFit software, but in our case we applied $\eta \in [-1, 2]$ cut on DCA$\leq$2 and Number of Hits$\geq$20, so we use the class that we are working on and thus get a file of multiplicity without much noise.

For the use of the method you have to configure the GammaFit2.C macro in line 17 if you know the input parameters teta, sigma, nknee, a1, a2, a3 you should put GetSigma = false, then in line 18 the cross section the collision should be added, On line 25 the rest of parameters are specified. If the parameters in line 17 are not known, it needs to be configured with GetSigma = true and all parameters will be searched automatically. For example, if the parameters are known, a configuration similar to the following could be used:

```
bool GetSigma = false;
Float_t sigma = 607.084;
Float_t teta = 1.1, n_knee = 220, a1 = -5., a2 = 2., a3 = -5.,
chi2, NDF, chi2_NDF;
```

But in our case we worked not to knowing the parameters. For the execution of the method it is necessary to have the file containing the information of the multiplicity in the same folder where the GammaFit2 is and use the following line:

```
root GammaFit2.C++'("./mult.root","hRefMultSTAR",
"./out.root",10,"hBvsRefMult")'
```

For the use of the macro it is necessary to have previously loaded the root version that is found on section "Packages and Versions used". First file is "mult.root" which is the file where the multiplicity histogram is located, and "hRefMultSTAR" is the name of the multiplicity histogram, "out.root" is the output name of the analysis, 10 refers to the measured multiplicity limit for a file and "hBvsRefMult" will be the name of the histogram where the adjustments are made.

For the case of a very low energy or a very light system, it is necessary to configure line 24, the default macro parameter is used as a starting point for adjustment but it doesn't work well on lightweight systems.

Finally, we use printFinal.C, when it is executed you get a file in C++ format with centrality values. To run the macro you have to execute following line:

```
root printFinal.C'("./out.root","out.C")'
```

To run the last file with multiplicities you can use root by executing the following lines:

```
./out.root
root [0] GetCentMult(Int_t mult)
```

## 3.5 MCGlauber method

In ref. [1] there is a manual document on the Documentation folder that explains how the code should be run. Here we are going to do a recapitulation.
The MCGlauber method is divided in two main procedures. The MCGlauber and Centrality Framework.

### 3.5.1 MCGlauber package

**Installation**

From the web page MC Glauber package is downloaded: https://tglaubermc.hepforge.org/downloads/
To download it on the NCX cluster we use the following command:

```
wget https://tglaubermc.hepforge.org/downloads/?f=TGlauberMC-3.2.tar.gz
```

To unpack it we use:

```
tar -xf *.tar.gz
rm *.tar.gz
```

From the MC Glauber package we only need the "runAndSaveNtuple" script, but we have to modify the macro with the specific colliding system that we are studying.
For that purpose on the main script "runglauber_v3.2.C" we need to manually modify it, with the following data (under the line number 1172):

- "Xempd","Wmpd" - Wanted name for our nucleons.
- fN - Mass number.

- fR - Radius of the nucleus.
- fA - Skin Thickness of the nucleus.
- fW - deformation parameter of the nucleus.
- fF - Type of the nuclear density function (Default: 1 - 3 Parameter Fermi Function).
- fZ - Atomic Number.

Specifically for our objective:

```
else if (TString(name) == "Xempd")
    {fN = 124; fR = 4.763; fA = 0.523; fW = 0; fF = 1; fZ = 54;}
else if (TString(name) == "Wmpd")
    {fN = 184; fR = 5.373; fA = 0.523; fW = 0; fF = 1; fZ = 74;}
```

**Usage**

First we need to set up the root enviroment. The root version used was: v6.32.02-1.
To initiate the root version on the NCX cluster we used:

```
source /cvmfs/nica.jinr.ru/sw/os/login.sh
module add ROOT/v6.32.02-1
```

Then we use the following commands to run Glauber Monte Carlo:

```
root -l
.L runglauber_v3.2.C
runAndSaveNtuple(Nev, sysA, sysB, signn)
.q
```

where:

- Nev - Number of events.
- sysA, sysB - colliding nuclei ("Xempd", "Wmpd" in our case ).
- signn - Inelastic nucleon - nucleon cross section (calculated earlier).

Finally the MCGlauber will write a .root file with the TNtuple nt_Xempd_Wmpd in it.

### 3.5.2 Centrality Framework

**Configuration**

After the installation of Centrality Framework there are some files that needs to be set up in order to run the Framework. They are located on:

```
cd CentralityFramework/scripts/templates/
```

### Configuring start.sh

We need to set up the start of the Framework, for that we need to configure the "start.sh" file. On the original manual [1] the initialization of the Framework is with qsub command from TORQUE/PBS scheduler, but due to problems with the cluster we ended up using sbatch command from SLURM.

This macro is used to create different jobs in order to run 200 different intervals on $f$ and $k$ variables at the same time. To modify it we replace every line that has #$ with the following code:

```
#!/bin/bash
#SBATCH --job-name=run_centrality_fit
#SBATCH --ntasks-per-node=1
#SBATCH --mem-per-cpu=8192
#SBATCH --time=14-00:00:0
#SBATCH -p nica
#SBATCH --array=1-200

#SBATCH -o CentralityFramework/Framework/McGlauber/scripts/template/TMP/output.%j
#SBATCH -e CentralityFramework/Framework/McGlauber/scripts/template/TMP/error.%j
```

where "CentralityFramework/Framework/McGlauber/scripts/template" has to be change to the full path of your Centrality Framework.

Also we need to change the JOB_ID and the TASK_ID as follows:

```
export JOB_ID=${SLURM_JOBID}
export TASK_ID=${SLURM_ARRAY_TASK_ID}
```

MAIN_DIR variable should be change to the full path where start.sh is located. Finally, it should be modified the root version on the file, it looks like this:

```
# Sourcing ROOT
source /opt/fairsoft/mpd/new/bin/thisroot.sh
```

and must be change to the actual root and CMake version that is being used, in our case:

```
source /cvmfs/nica.jinr.ru/sw/os/login.sh
module add ROOT/v6.32.02-1
module add OpenSSL/v3.2.1-1
module add CMake/v3.28.3-1
```

### Configuring config.txt.template

Once the input files from MC Glauber and data are ready. We need to change the full path to the MC Glauber output file in the first line, then the name of the TNtuple in second line (nt_Xempd_Wmpd) and full path to the .root file with multiplicity distribution obtain from the cuts in the third line. If the name of the histogram was modified it should be set in the line 4 of the file. Finally the name of the parameterization for $N_a$ (number of ancestors) should be modified on the line 14 of the file.

**Configuring parameter.list**

It has 200 lines with the parameters configuration for each job in the following pattern:

```
f_min:f_max:k_min:k_max:Mult_min:Mult_max
```

We only need to modify the Multiplicity intervals, for that we can use the following:

```
sed -i "s/20:360/Mult_min:Mult_max/" parameter.list
```

Here Mult_min and Mult_max are chosen based on the multiplicity distribution.
Once everything is set up we run the start.sh file:

```
sbatch start.sh
```

It took about 3 hours to finish. All the result are on the output directory OUT alongside start.sh script.

### 3.5.3 Final Steps

**Finding the best fit**

We need to merge all fit result from the 200 folder inside of OUT/. For that we use:

```
hadd -k -f -j 20 fit_merge.root OUT/*/file/root/fit/*.root
```

It will output a fit_merge.root file. Then we need to use CentralityFramework/Framework/McGlauber/Chi2.C macro:

```
root -l -b -q Chi2.C'(".........../fit_merge.root")'
```

Chi2.root will output a line with:

```
f = 1 +- 0.0004 mu = 0.451677 +- 0.261894 k = 38 +- 54.629 chi2 = 1.97936 +- 0.183907
```

With that information one has to find the corresponding glauber_qa directory inside of one folder on the OUT/ directory. To do so, find the line in parameter.list which contains the found $f$ and $k$ parameters. The number of the line it will be the directory with the best $\chi^2$.

**Dividing multiplicity into centrality classes using multiplicity cuts**

Now that we have the best fit, inside of the glauber_qa file is the glauber_qa_N.root. In the file HistoCut.C one should modify the lines 2 and 3 with the full path to the file glauber_qa_N.root. If the name of the multiplicity histogram was changed, it needs to be modified in line 5.
To run the macro:

```
root -l -b -q HistoCut.C'(10)'
```

The argument 10 will produce 10 centrality classes within 0-100% range. The output file will be HistoCutResult.root which contains multiplicity distribution for each centrality cut for both fitted multiplicity function and multiplicity distribution.

**Mapping parameters from MC Glauber with centrality classes**

Similarly to the previous steps, in the file CentralityClasses.C one has to modify lines 2 and 3 with the full path to the HistoCutResult.root file and glauber_qa_N.root file. And then run:

```
root -l -b -q CentralityClasses.C'(10)'
```

The argument is also the number of centrality classes used on the HistoCut.root script.
The resulting file will be FINAL.root containing impact parameter (b vs CentralityClasses), number of participants (Npart vs CentralityClasses), number of binary nucleon - nucleon collisions (Ncoll vs CentralityClasses) distributions for each class and centrality dependence of their mean values.

Finally, we can use the printFinal.C macro to display this information in a simple way on latex format:

```
root -l -b -q printFinal.C'(".....FINAL.root","./example.tex")'
```

## 3.6 Summary

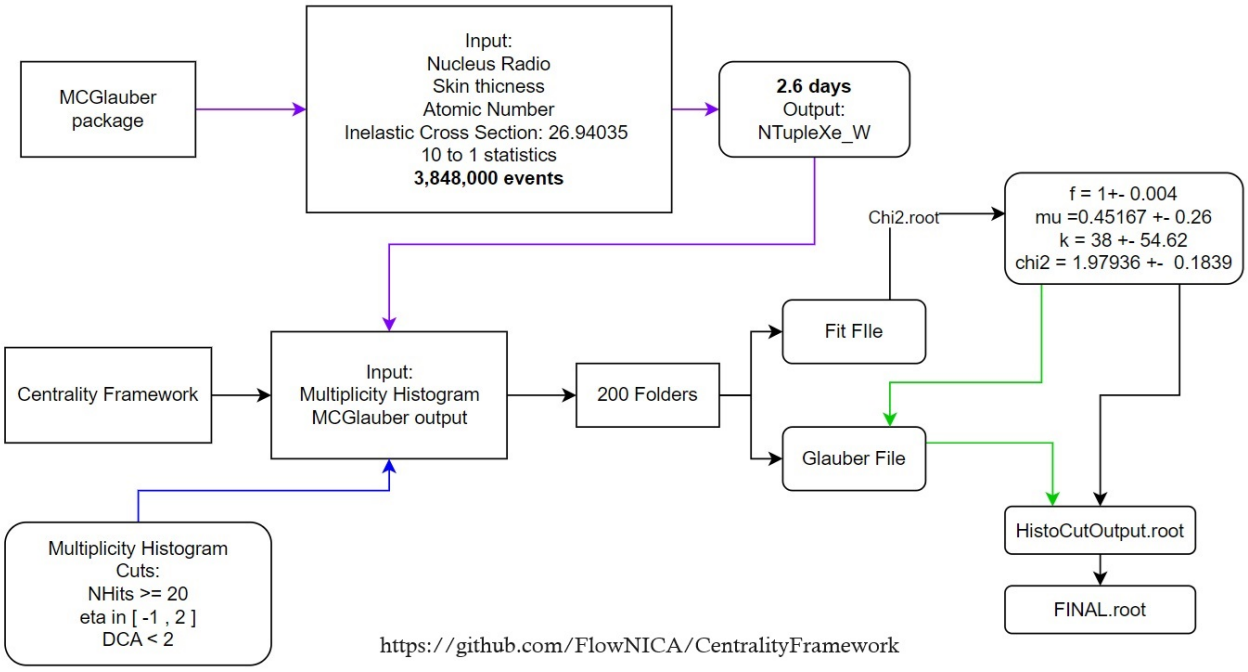



Figure 9: Graphic Summary of the MC Glauber procedure

# 4 Results

## 4.1 GammaFit results

For the use of the sotfware is used an input file multiplicity, to this one is made a configuration as you can see in Figure 10, the right side is the input file and on the left side is the adjustment made by the software

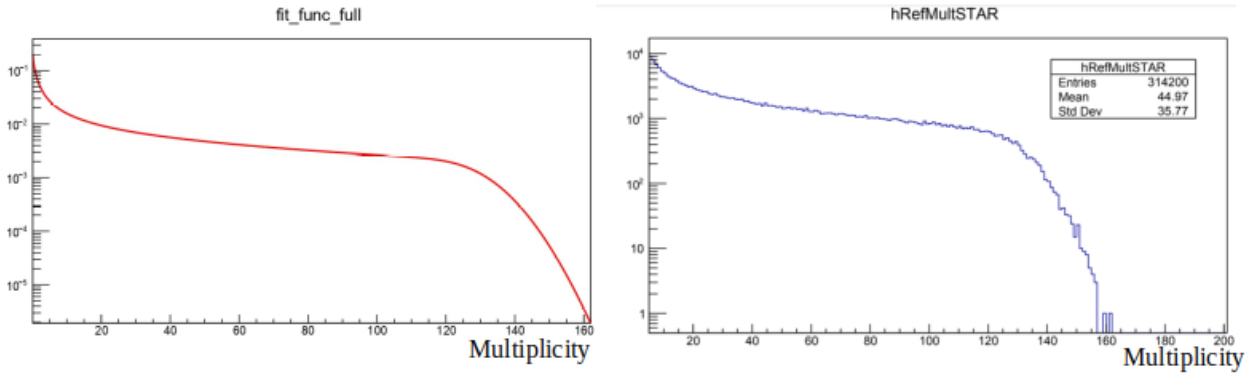



Figure 10: input file multiplicity in blue and red multiplicity adjustment

Below you will find the histograms of the centrality parameter setting impact by centrality compared to that used from the input file.

The Figure 11 shows as an example for centralities of 5-10%, 40-50% and 80-100% of the right side are reconstructed from the adjustment and left are calculated from the information provided by the UrQMD generator.

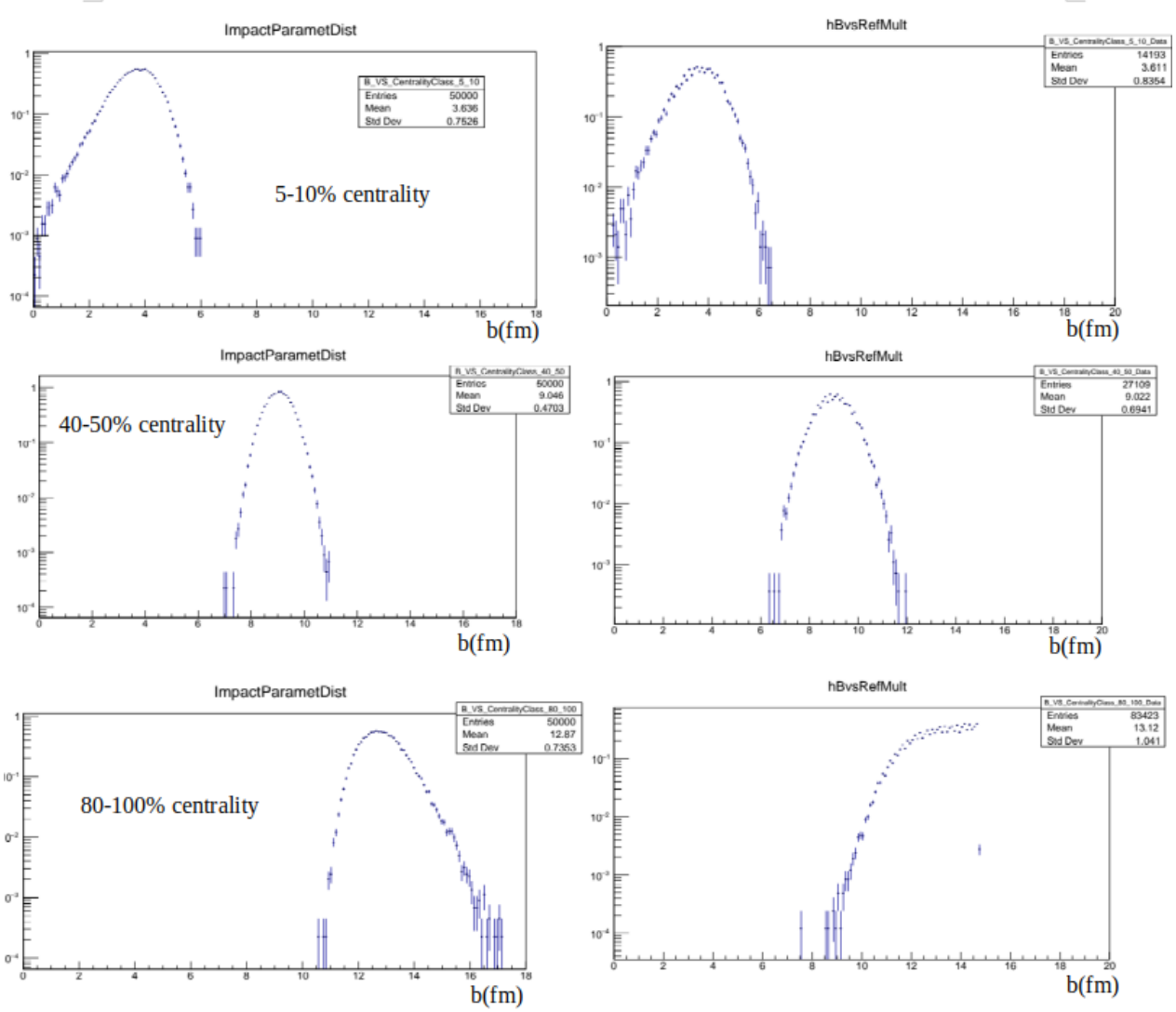



Figure 11: Impact parameter reconstructed by the adjustment and obtained from UrQMD information

Figure 12 shows the reason for the charge of particles in relation to the number of events as a function of the multiplicity, in the image are the data provided by our input file represented by blue dots and the red line represents the adjustment.

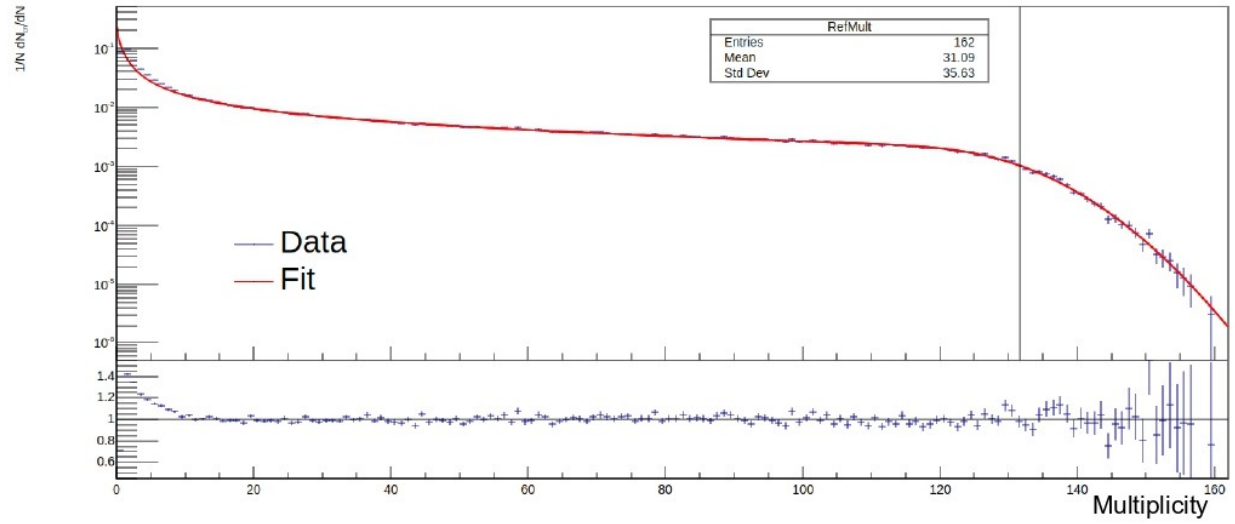



Figure 12: Fit and input file data

Finally, the impact parameter is extrapolated and compared with the direct one from our UrQMD information, as well as a histogram showing how good the adjustment was presented in figure 13.

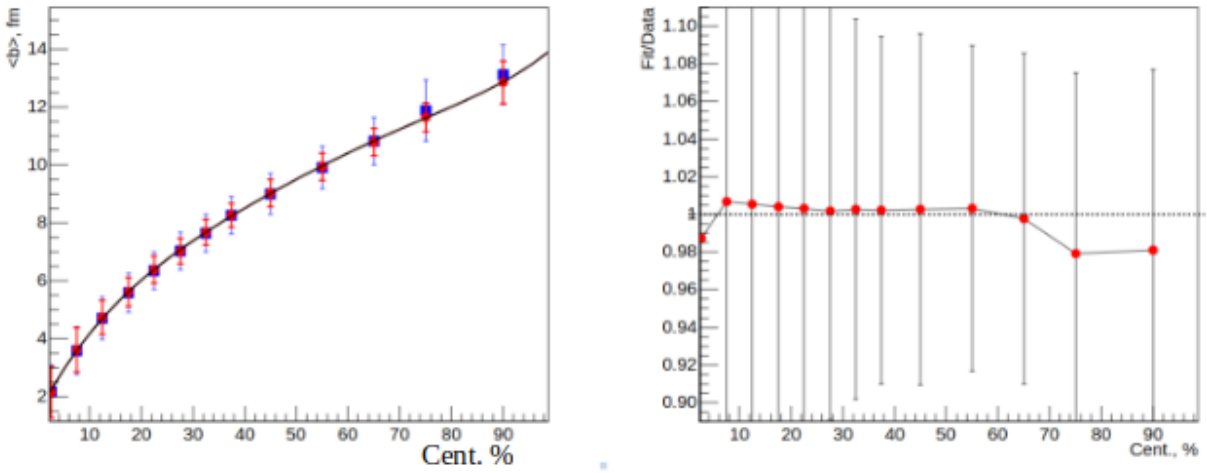



Figure 13: Impact parameter as function of the centrality of the adjustment vs the extracted from the generator of events.

In the file ". root" it is also possible to find the different parameters that we could give to configure them to know, because in our case we did not know that the values presented are taken directly from the adjustment, Figure 12 shows a histogram where one of these parameter values is found but it is better to extract it from the file out.C, the rest of parameters are shown in table 1.

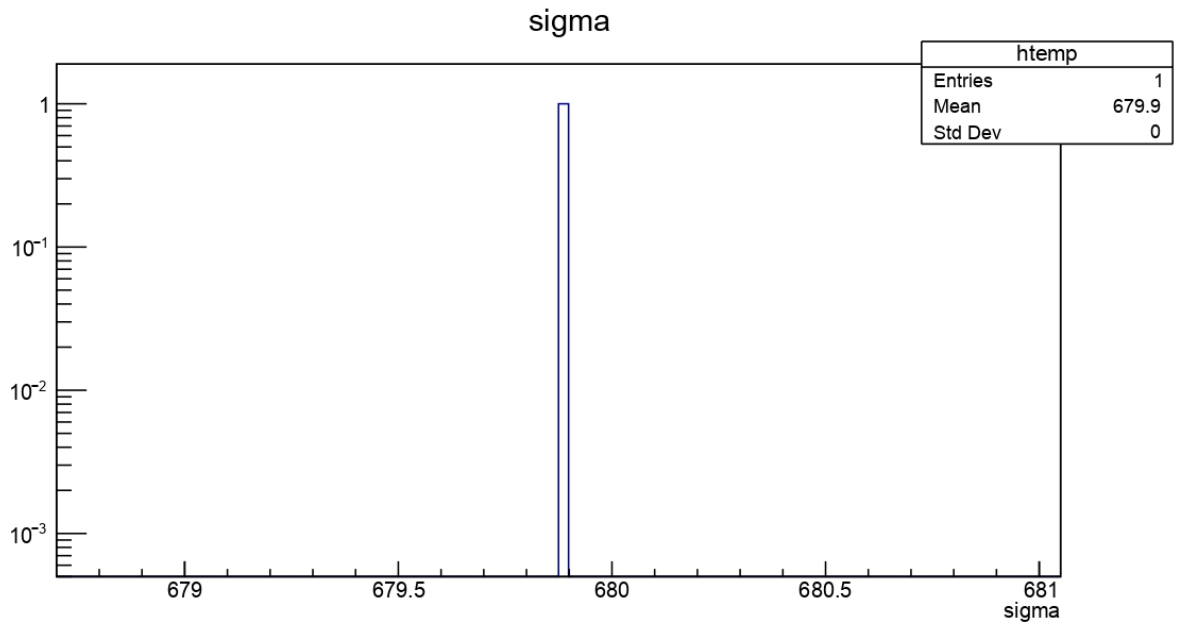



Figure 14: sigma parameter.

| Parameter | Value |
|---|---|
| NDF | 143 |
| a1 | -3.73 |
| a2 | 0.164 |
| a3 | -2.84 |
| $chi^2$ | 163.5 |
| nknee | 131.7 |
| sigma | 679.9 |
| teta | 0.647 |
| $chi^2/NDF$ | 1.143111 |

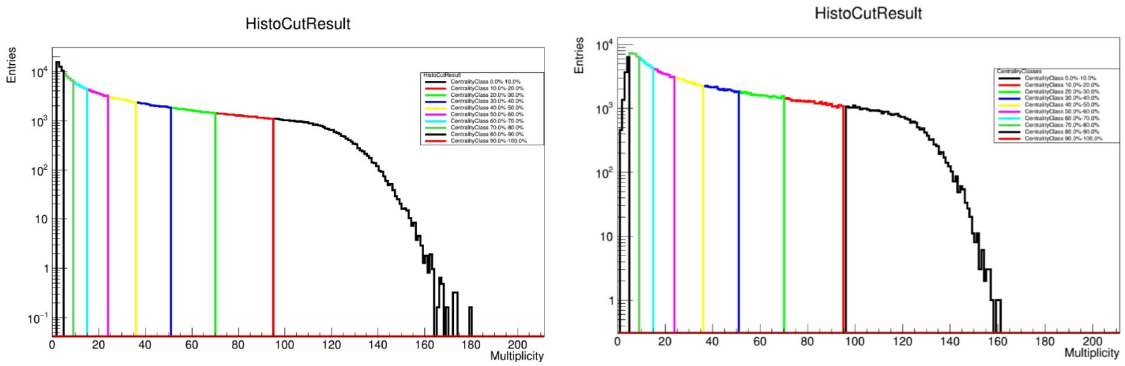
Table 1: Centrality, multiplicity and impact parameter data

With the file "out. C" were extracted the values of centrality and impact parameter presented below in table 1.

| Centrality % | Multiplicity | Impact parameter |
|---|---|---|
| 0-10 | 162-96 | 2.94-5.18 |
| 10-20 | 96-70 | 5.18-6.75 |
| 20-30 | 70-50 | 6.75-7.39 |
| 30-40 | 50-34 | 7.39-7.97 |
| 40-50 | 34-23 | 7.97-9.51 |
| 50-60 | 23-15 | 9.51-10.42 |
| 60-70 | 15-9 | 10.42-11.24 |
| 70-80 | 9-5 | 11.24-12.01 |
| 80-100 | 5-1 | 12.01-14.10 |

Table 2: Centrality, multiplicity and impact parameter data

## 4.2 MCGlauber results




(a) Multiplicity fitted function (b) Multiplicity input ditribution

Figure 15: Multiplicity ditribution divided by centrality Classes

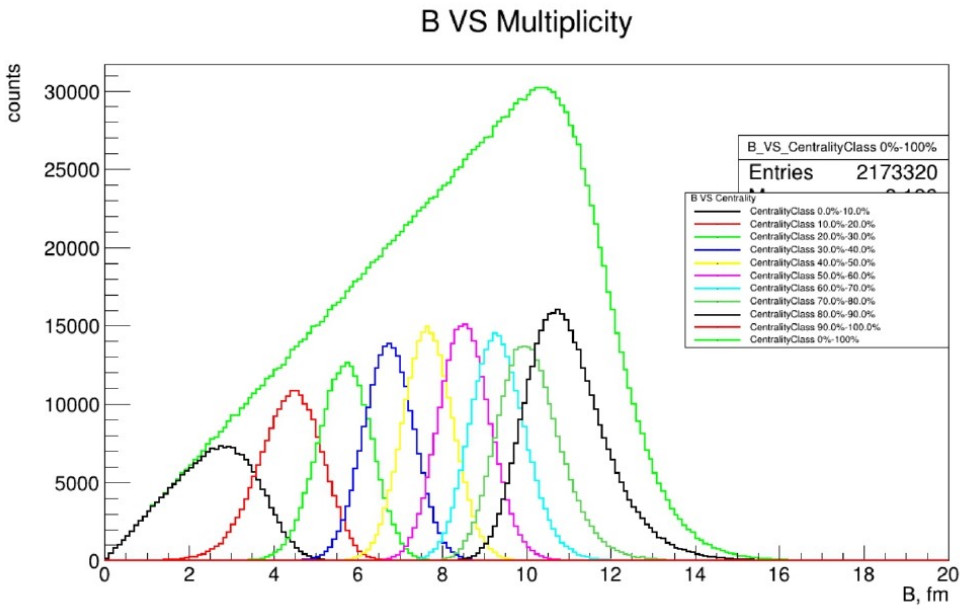
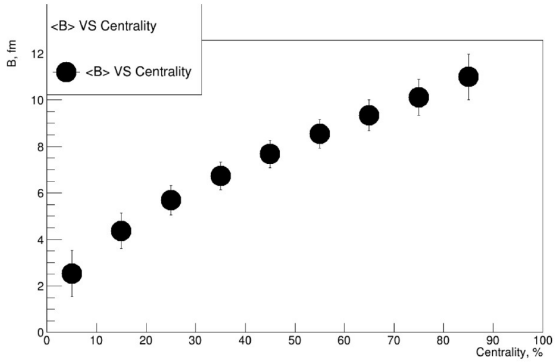
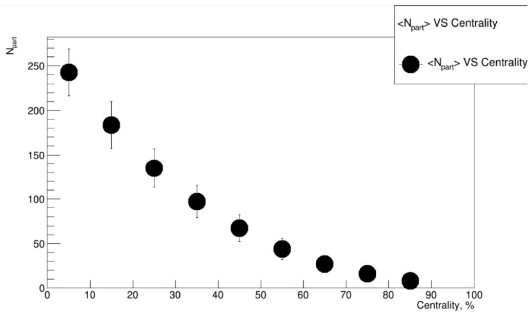


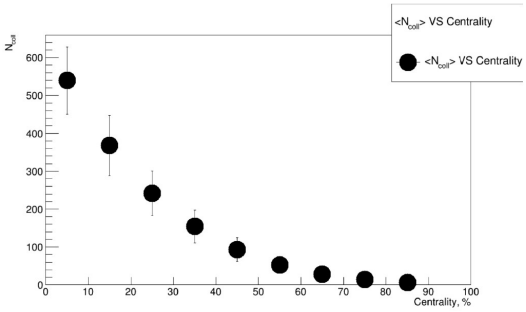

Figure 16: Impact parameter divided by centrality Classes




(a) Mean impact parameter vs Centrality




(b) Mean Number of participants vs Centrality




(c) Mean Number of collision vs Centrality

Figure 17: Mean values vs Centrality Classes

Table of $N_{part}$, $N_{coll}$, mean $N_{part}$, $N_{coll}$ and mean impact parameter for each centrality class.

| Centrality, % | $N_{ch}^{min}$ | $N_{ch}^{max}$ | $\langle b \rangle$, fm | RMS | $\langle N_{part} \rangle$ | RMS | $\langle N_{coll} \rangle$ | RMS |
|---|---|---|---|---|---|---|---|---|
| 0 - 10 | 95 | 164 | 2.54 | 1.00 | 242.75 | 26.44 | 539.73 | 88.76 |
| 10 - 20 | 70 | 95 | 4.37 | 0.76 | 183.50 | 26.19 | 367.84 | 79.38 |
| 20 - 30 | 51 | 70 | 5.69 | 0.63 | 134.96 | 21.54 | 241.37 | 59.12 |
| 30 - 40 | 36 | 51 | 6.74 | 0.59 | 97.33 | 17.96 | 154.42 | 43.75 |
| 40 - 50 | 24 | 36 | 7.68 | 0.59 | 67.38 | 14.84 | 93.66 | 31.25 |
| 50 - 60 | 15 | 24 | 8.55 | 0.61 | 43.96 | 11.82 | 52.81 | 20.86 |
| 60 - 70 | 9 | 15 | 9.35 | 0.66 | 27.27 | 9.01 | 28.33 | 13.12 |
| 70 - 80 | 5 | 9 | 10.11 | 0.78 | 16.04 | 6.73 | 14.59 | 8.12 |
| 80 - 90 | 2 | 5 | 10.99 | 0.99 | 8.23 | 4.74 | 6.60 | 4.77 |

Table 3: Results table of MC Glauber method

## 4.3 Comparison of results

The following tables show the data compared for the two methods of centrality, namely multiplicity and impact parameter.

| Centrality, % | $N_{chMCGlauber}^{min}$ | $N_{chMCGlauber}^{max}$ | $N_{chGammaFit}^{min}$ | $N_{chGammaFit}^{max}$ | $\Delta N_{ch}^{min}$ | $\Delta N_{ch}^{max}$ |
|---|---|---|---|---|---|---|
| [0 - 10] | 95 | 164 | 96 | 162 | 1 | 2 |
| [10 - 20] | 70 | 95 | 70 | 96 | 0 | 1 |
| [20 - 30] | 51 | 70 | 50 | 70 | 1 | 0 |
| [30 - 40] | 36 | 51 | 34 | 50 | 2 | 1 |
| [40 - 50] | 24 | 36 | 33 | 34 | 9 | 2 |
| [50 - 60] | 15 | 24 | 15 | 23 | 0 | 1 |
| [60 - 70] | 9 | 15 | 9 | 15 | 0 | 0 |
| [70 - 80] | 5 | 9 | 5 | 9 | 0 | 0 |
| [80 - 90] | 2 | 5 | 1 | 5 | 1 | 0 |

Table 4: Multiplicities extracted with the two methods and difference between them divided by centrality classes

| Centrality % | $\langle b_{MCGlauber} \rangle$ | $\langle b_{GammaFit} \rangle$ |
|---|---|---|
| 0-10 | 2.54 | 2.21 |
| 10-20 | 4.37 | 4.7 |
| 20-30 | 5.69 | 5.62 |
| 30-40 | 6.74 | 6.4 |
| 40-50 | 7.68 | 7.68 |
| 50-60 | 8.55 | 8.24 |
| 60-70 | 9.35 | 9.02 |
| 70-80 | 10.11 | 10.84 |
| 80-100 | 10.99 | 11.63 |

Table 5: Average impact parameter comparison between methods

# 5 Conclusions

# 6 Appendix

## 6.1 Inelastic Cross Section

To obtain the inelastic cross section we used the following formula:

$$\sigma_{inel} = \sigma_{total} - \sigma_{ela} \tag{2}$$

where:

- $\sigma_{total}$ - Total cross section
- $\sigma_{ela}$ - Elastic cross section
- $\sigma_{inel}$ - Inelastic cross section

To obtain total and elastic cross section it was used the following formulas:

$$\sigma_{total} = 48.0 + 0.522ln^2(p_{lab}) - 4.51ln(p_{lab}) \tag{3}$$
$$\sigma_{ela} = 11.9 + 26.9p_{lab}^{-1.21} + 0.169ln^2(p_{lab}) - 1.85ln(p_{lab}) \tag{4}$$

where: $p_{lab}$ - momentum in the lab [Gev/c]

The momentum in the lab needs to be calculated. Our experiment is a Fixed Target experiment at T = 2.5 GeV. So our $p_{lab}$ has to be the momentum of the projectile.

$$T = E_{proy} - E_0$$
$$E_{proy} = T + E_0 = 2.5 + 0.938 = 3.438$$
$$E_{proy}^2 = m^2 + p_{lab}^2$$
$$p_{lab} = \sqrt{E_{proy}^2 - m^2} = 3.3075$$

From the calculation we obtain a result of $\sigma_{inel} = 26.94035$ mb.

# Bibliography

[1] Peter Parfenov. *Centrality Framework*.

[2] The NICA LHEP offline computing cluster. Accesed: 2024-07-31.

[3] M. Bleicher et al. Relativistic hadron hadron collisions in the ultrarelativistic quantum molecular dynamics model. *J. Phys. G*, 25:1859–1896, 1999.