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

FINAL REPORT ON THE SUMMER STUDENT PROGRAM

Improvement of methods for event reconstruction in the BM@N experiment.

Supervisors:
Segei P. Merts
Oleg V. Rogachevskiy

Student:
Andrew V. Zelenov
Peter the Great St.Petersburg Polytechnic University

Participation period:
July 10 – August 26

Dubna, 2017

Contents

1 Abstract

This report discusses several tasks that were solved during JINR Summer Students Program. They are all connected with the optimization of existing algorithms such as Primary Vertex Finding, Tracks Finding and Magnetic Field Map initialization.

The efficiency of Primary Vertex finding depends entirely on the quality of founded Tracks. So, this two tasks are the two inseparable parts.

The current algorithm of Primary Vertex Finding has two problems: Speed performance of algorithm and the accuracy of determining of Primary Vertex location. In this report will set out solutions to these problems.

The algorithm of Tracks Finding needed to be improved, because set the search parameters of the track make it impossible to find all the tracks, because of what the efficiency is not as high as could be.

Magnetic Field Map initialization is not a problem, but time of this initialization – is the biggest part of full initialization. So, it is reasonable to improve this and as one of the possible solutions to translate the map into a ROOT tree format.

2 Baryonic Matter at Nuclotron

1 Introduction

Relativistic heavy ion collisions provide the unique opportunity to study nuclear matter under extreme density and temperature. In the collision, nuclear matter is heated up and compressed for a very short period of time. At moderate temperatures, nucleons are excited to baryonic resonances which decay by the emission of mesons. At higher temperatures, also baryon-antibaryon pairs are created. This mixture of baryons, antibaryons and mesons, all strongly interacting particles, is denoted as hadronic matter or baryonic matter if baryons dominate. If the energy density in the formed fireball is sufficiently large the quark-gluon substructure of nucleons becomes visible. At even higher temperatures or densities hadrons melt, and the constituents, quarks and gluons, form a new phase, the Quark-Gluon Plasma (QGP). At these extreme conditions the following features of strongly interacting matter can be studied: the equation-of-state (EoS) of strongly interacting matter at high temperatures and high net-baryon densities; the microscopic structure of strongly interacting matter as a function of temperature and baryon density; the in-medium modifications of hadrons which might provide information on the onset of chiral symmetry restoration. Theoretical models, however, suggest different possible scenarios to describe these features of strongly interacting matter. So that new experimental data with high resolution and statistics are needed in order to disentangle different theoretical predictions.

2 Physical motivation: Nuclotron heavy-ion physics program

The ratio of produced mesons to baryons in the fireball increases with the collision energy. A nucleus-nucleus collision at the Nuclotron beam kinetic energy in the range from 1 to 4.5 GeV per nucleon produces a baryon dominated fireball contrary to higher energies at RHIC or SPS. According to the QGSM transport model calculations at the Nuclotron energies the nucleon densities in the collision zone of two gold nuclei exceed the saturation density by a factor of 3 - 4. At these densities nucleons start to overlap, and it is expected that under such extreme conditions the onset of chiral symmetry restoration might occur although quarks are still confined. It will reveal in in-medium modification of hadrons, in particular, in collisional broadening, dropping mass of vector mesons decaying into di-leptons which are not much affected by final-state interactions. The relevant degrees of freedom at the Nuclotron energies are first of all nucleons and their excited states followed by light and strange mesons. Also the partonic degrees of freedom should show up in small space-time volumes and leave their traces in final hadronic observables. The focus of experimental studies will be on hadrons with strangeness, which are early produced in the collision and not present in the initial state of two colliding nuclei, unlike nucleons made up from light (u, d)-quarks. The measured production yields of light and strange mesons, as well as of hyperons and anti-hyperons are show in figure 1 as a function of the nucleon-nucleon collision energy in c.m.s. The Nuclotron heavy ion beam energy range corresponds to $\sqrt{s_{NN}} = 2.3 - 3.5$ GeV. It is well suited for studies of strange mesons and multi-strange hyperons which are produced in nucleus-nucleus collisions close to the kinematic threshold. Heavy-ion collisions are a rich source of strangeness, and the coalescence of lambda-hyperons with nucleons can produce a variety of light hyper-nuclei. The study of the hyper-nuclei production is expected to provide new insights into the properties of the hyperon-nucleon and hyperon-hyperon interactions. Figure 2 presents the yields of hyper-nuclei as a function of the nucleon-nucleon collision energy in c.m.s. in Au+Au collisions, predicted by a thermal model. The maximum in the hyper-nuclei production rate is predicted at $\sqrt{s_{NN}} \sim 4 - 5$ GeV, which is close to the Nuclotron energy range. In sum, the research program on heavy-ion collisions at the Nuclotron includes the following topics: investigation of the reaction dynamics and nuclear EoS, study of the in-medium properties of hadrons, production of (multi)-strange hyperons at the threshold and search for hyper-nuclei. In order to interpret experimental data from heavy-ion collisions and to provide normalization for the measured A^+A spectra, a study of elementary reactions (p^+p, $p^+n(d)$) is planned.

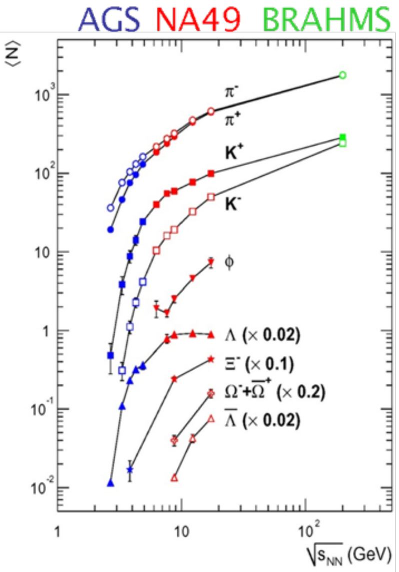

Figure 1: Yields of mesons and (anti-) hyperons as a function of the nucleon- nucleon collision energy in c.m.s. in Au^+Au/Pb^+Pb collisions, taken from. The Nuclotron BM@N heavy ion beam energy range corresponds to $\sqrt{s_{NN}} = 2.3 - 3.5$ GeV.

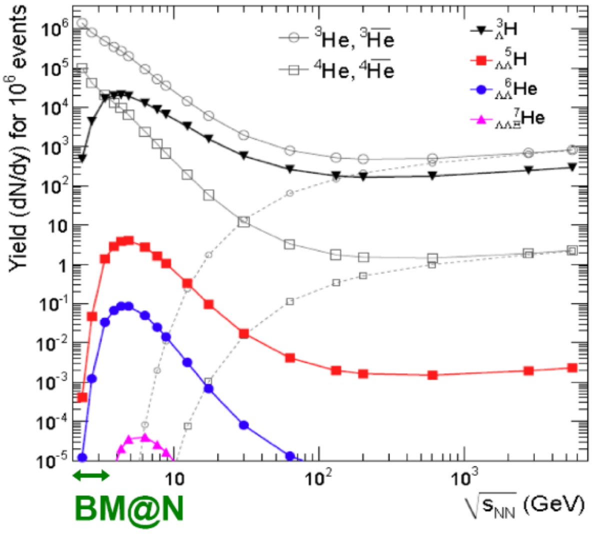

Figure 2: Yields of hyper-nuclei as a function of the nucleon-nucleon collision energy in c.m.s. in Au^+Au collisions, calculated with a thermal model. The predicted yields of 3He and 4He nuclei are included for comparison. The Nuclotron BM@N energy range $\sqrt{s_{NN}} = 2.3 - 3.5$ GeV is specified.

3 The BM@N detector

BM@N (Baryonic Matter at Nuclotron) is the first experiment at the accelerator complex of NICA Nuclotron. The schematic view of the NICA-Nuclotron complex and the position of the BM@N setup are presented in figure 3. The sources of light and heavy ions, the beam Booster, Nuclotron accelerator and NICA collider are shown. The heavy-ion physics program of the NICA accelerator complex and the MPD experiment planned at the NICA collider are described in. The aim of the BM@N experiment is to study interactions of relativistic heavy ion beams with fixed targets. The Nuclotron will provide verity of beams from protons to gold ions with the kinetic energy of ions ranging from 1 to 6 GeV per nucleon. The maximum kinetic energy for ions with the ratio of the charge to the atomic weight (Z/A) of 1/2 is 6 GeV per nucleon. The maximum kinetic energy for heavy ions with the ratio of Z/A $\sim$ 1/3 is 4.5 GeV per nucleon. The maximum kinetic energy of protons is 13 GeV. The beam line between the Nuclotron and the BM@N experiment is around 160 meter in length. It comprises 26 elements of magnetic optics: 8 dipole magnets and 18 quadruple lenses. An upgrade program of the beam line is foreseen to minimize the amount of scattering material on the way of heavy ions to the BM@N setup.

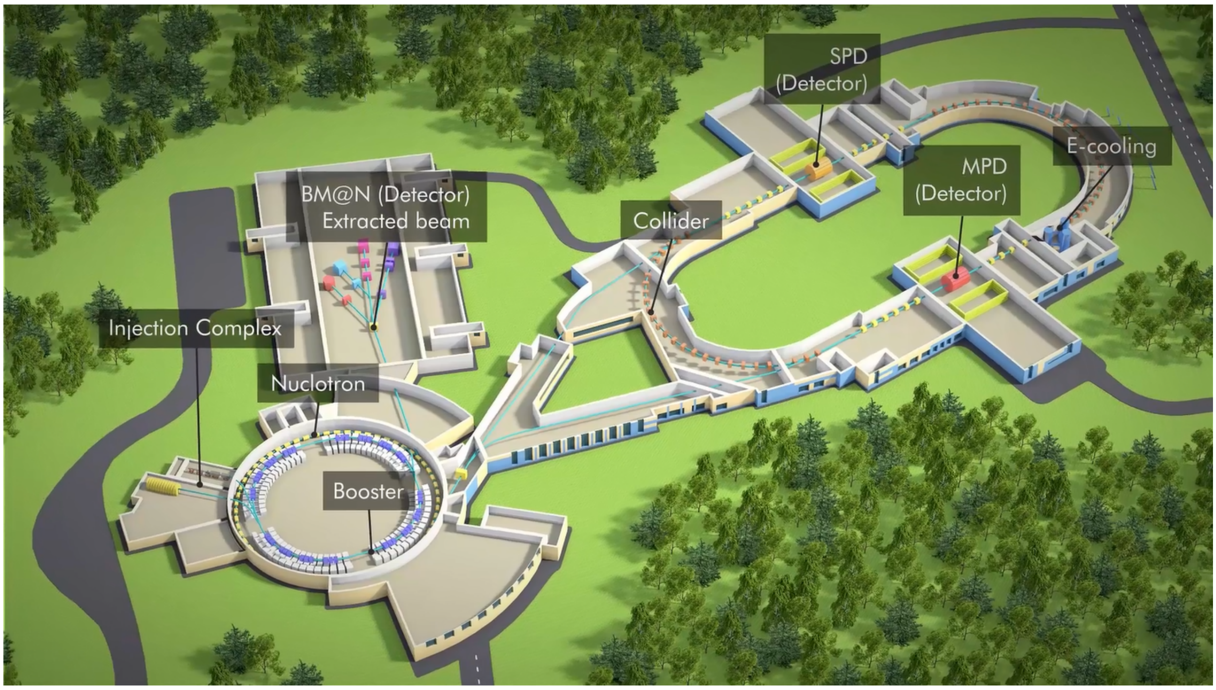

Figure 3: Schematic view of the NICA-Nuclotron complex and the position of the BM@N setup.

The planned intensity of the gold ion beam accelerated and accumulated in the Nuclotron and the Booster and transported to the BM@N experimental zone is up to 10^7 ions per second.

The gold ion beam is expected in the end of 2018. In the period before 2018 the following ions are foreseen to accelerate: the polarized deuteron beam in 2016, the carbon, argon and krypton beams in 2017. In this period of operation the planned intensity of the beam interacting with the target inside the BM@N setup is 106 ions per second. The proton-proton interactions will be studied after the Nuclotron upgrade in 2018 using the proton beam and the liquid hydrogen target. Figure 4 shows the diagram of the interaction rates accepted by data acquisition systems of heavy ion experiments running at different energies of colliding nuclei.

The beam energy range in the BM@N experiment overlaps partially with that in the HADES experiment. The interaction rate of triggered non peripheral central and intermediate events at the second stage of the BM@N experiment is expected to be around 50 kHz. It is limited by the capacity of the readout electronics and data acquisition system.

3.1 Main characteristics of the experimental setup

A sketch of the proposed experimental set-up is shown in Figure 5. The experiment combines high precision track measurements with time-of-flight information for particle identification and uses total energy measurements for the analysis of the collision centrality. The charged track momentum and multiplicity will be measured with the set of two

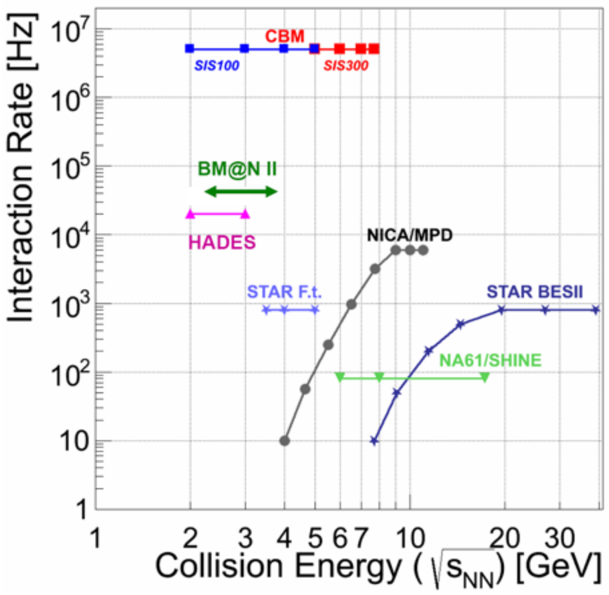

Figure 4: Heavy ion experiments: interaction rate and nucleon-nucleon collision energy in c.m.s.

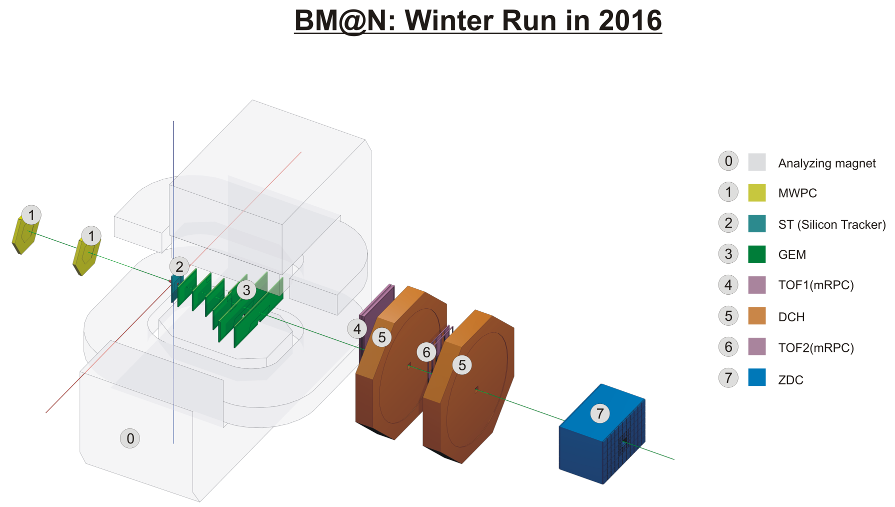

Figure 5: Schematic view of the BM@N setup.

coordinate planes of GEM (Gaseous Electron Multipliers) detectors located downstream of the target in the analyzing magnet and the drift/straw chambers (DCH, Straw) situated outside the magnetic field. The GEM detectors sustain high rates of particles and are operational in the strong magnetic field. The gap between the poles of the analyzing magnet is around 1 m. The magnetic field can be varied up to 1.2 T to get the optimal BM@N detector acceptance and momentum resolution for different processes and beam energies. The design parameters of the time-of-flight detectors based on multi-gap resistive plate chambers (mRPC-1,2) with a strip read-out allow us to discriminate between hadrons (π,K,p) as well as light nuclei with the momentum up to few GeV/c produced in multi-particle events. The zero degree calorimeter (ZDC) is designed for the analysis of the collision centrality by measuring the energy of forward going particles. The T0 detector, partially covering the backward hemisphere around the target, is planned to trigger central heavy ion collisions and provide a start time (T0) signal for the mRPC-1,2 detectors. An electro-magnetic calorimeter will be installed behind the outer drift/straw chambers and mRPC-2 wall to study processes with electro-magnetic probes (γ, e$\pm$) in the final state.[1]

3 Optimization of primary vertex reconstruction

The primary vertex is a point, where a high energy beam collides with a matter of target. In particle physics, an interaction point (IP) is the place where particles collide. One differentiates between the nominal IP, which is the design position of the IP, and the real or physics IP, which is the position where the particles actually collide. The real IP is the primary vertex of the particle collision.

1 Algorithm description

Previous version of algorithm based on method of virtual planes. The range from $-100cm$ to $100cm$ splitted by 400 virtual planes for Kalman filter. After that Kalman filter searched points of intersection between tracks and virtual planes. Then was calculated distances between points in every virtual plane to search mean distance. The Z coordinate of plane with minimal mean distance characterized as a location of Primary Vertex.

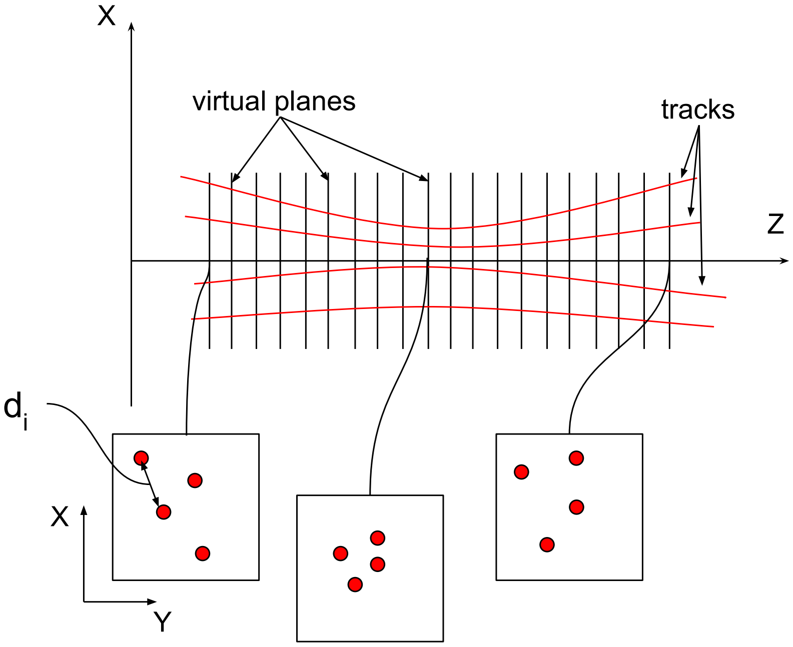

Figure 6: Virtual planes method

On the Figure 6 visualized virtual planes method. Blocks with red points – XY-projection of virtual planes and red points – points of intersection between tracks and virtual planes. Mean distance for every plane calculated by formula:

$$\mu_{dist} = \sum_i \frac{d_i}{N}$$

First of all was analyzed the code and revealed two problems:

1. Speed performance of algorithm. The resolution of primary vertex reconstruction was achieved by using a huge number of "virtual planes" for Kalman filter.

2. Gauss fitting of the result. Primary vertex reconstruction based on tracks reconstruction in GEM detector. It turns out that tracks of secondary particles give a contribution to the final distribution.

1.1 Kalman filter

The Kalman filter uses a system's dynamics model (e.g., physical laws of motion), known control inputs to that system, and multiple sequential measurements (such as from sensors) to form an estimate of the system's varying quantities (its state) that is better than the estimate obtained by using only one measurement alone. As such, it is a common sensor fusion and data fusion algorithm.

Noisy sensor data, approximations in the equations that describe the system evolution, and external factors that are not accounted for all place limits on how well it is possible to determine the system's state. The Kalman filter deals effectively with the uncertainty due to noisy sensor data and to some extent also with random external factors. The Kalman filter produces an estimate of the state of the system as an average of the system's predicted state and of the new measurement using a weighted average. The purpose of the weights is that values with better (i.e., smaller) estimated uncertainty are "trusted" more. The weights are calculated from the covariance, a measure of the estimated uncertainty of the prediction of the system's state. The result of the weighted average is a new state estimate that lies between the predicted and measured state, and has a better estimated uncertainty than either alone. This process is repeated at every time step, with the new estimate and its covariance informing the prediction used in the following iteration. This means that the Kalman filter works recursively and requires only the last "best guess", rather than the entire history, of a system's state to calculate a new state.

The relative certainty of the measurements and current state estimate is an important consideration, and it is common to discuss the response of the filter in terms of the Kalman filter's gain. The Kalman gain is the relative weight given to the measurements and current state estimate, and can be "tuned" to achieve particular performance. With a high gain, the filter places more weight on the most recent measurements, and thus follows them more responsively. With a low gain, the filter follows the model predictions more closely. At the extremes, a high gain close to one will result in a more jumpy estimated trajectory, while low gain close to zero will smooth out noise but decrease the responsiveness.

When performing the actual calculations for the filter (as discussed below), the state estimate and covariances are coded into matrices to handle the multiple dimensions involved in a single set of calculations. This allows for a representation of linear relationships between different state variables (such as position, velocity, and acceleration) in any of the transition models or covariances. [2]

2 Algorithm optimization

2.1 Speed performance

To solve the first problem, was added an iterative algorithm with 5 "virtual planes". In the first step it gets overrated range of tentatively primary vertex location. As a result, we have distribution function of mean dist between points in "virtual planes" from Z coordinate of this planes. This function is well fitted by a second degree polynomial. After fitting we get vertex of a parabola, which become a new center of overrated range of tentatively primary vertex location. But from iteration to iteration range is reduced in two times while it doesn't less than $0.1cm$. In the end we have primary vertex location with algorithm precision to $0.02cm$ in every event.

On the Figure 7 visualized a realization of this iteration method. Red lines are second degree polynomial fit functions.

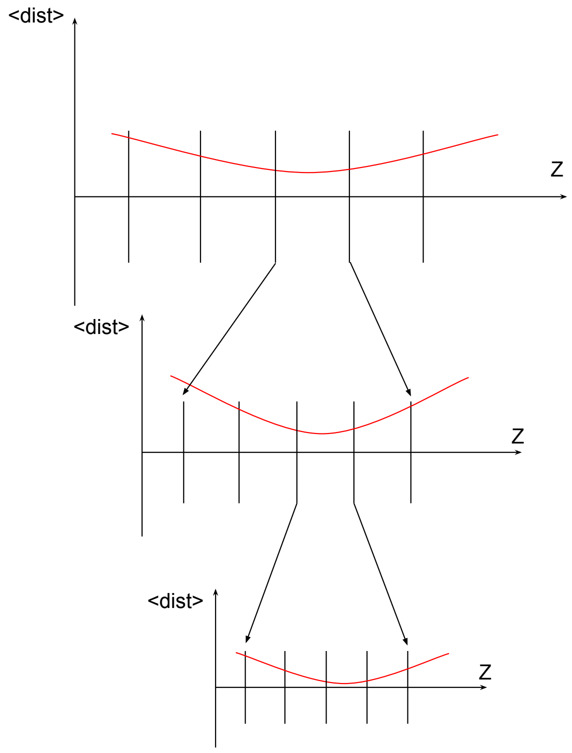

Figure 7: New Algorithm realization

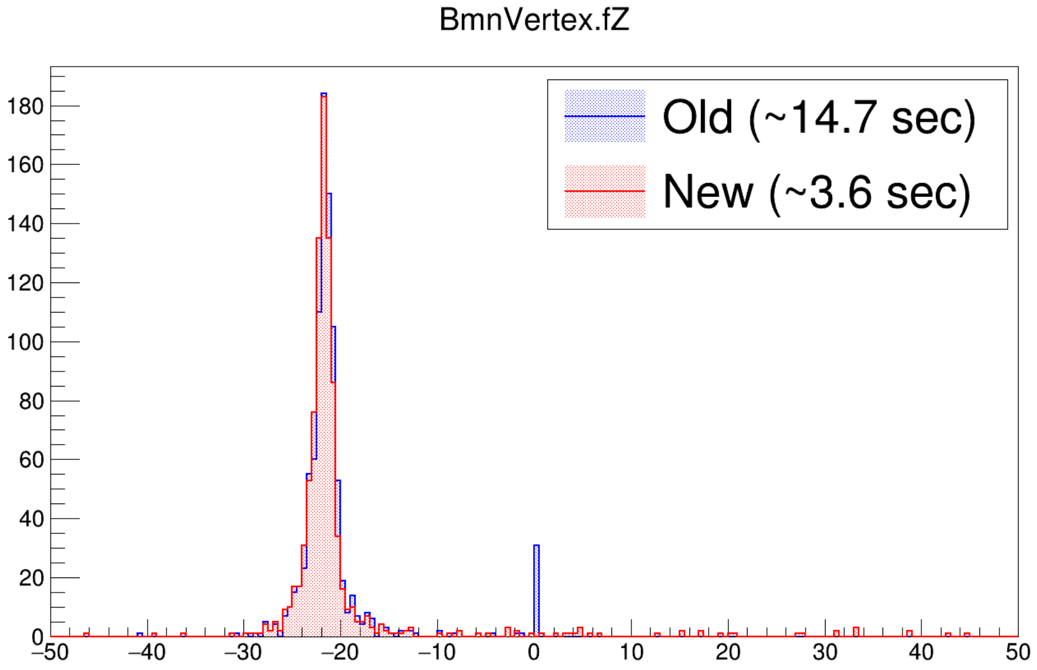

Figure 8: Speed performance of algorithm

We can see from Figure 8 that the algorithm became faster by 4 times. The accuracy of the determination of primary vertex is still remained at the same level.

To analyze performance improvements was used Google Performance Tools(GPT)[3]. GPT – is a set of tools, that allow to make performance analysis, analyse memory allocation and find memory leaks. In the Figure 9 and Figure 10 we can see graphical output of this analysis. But this output gives information, that for CPU using new algorithm become faster by 2 times.

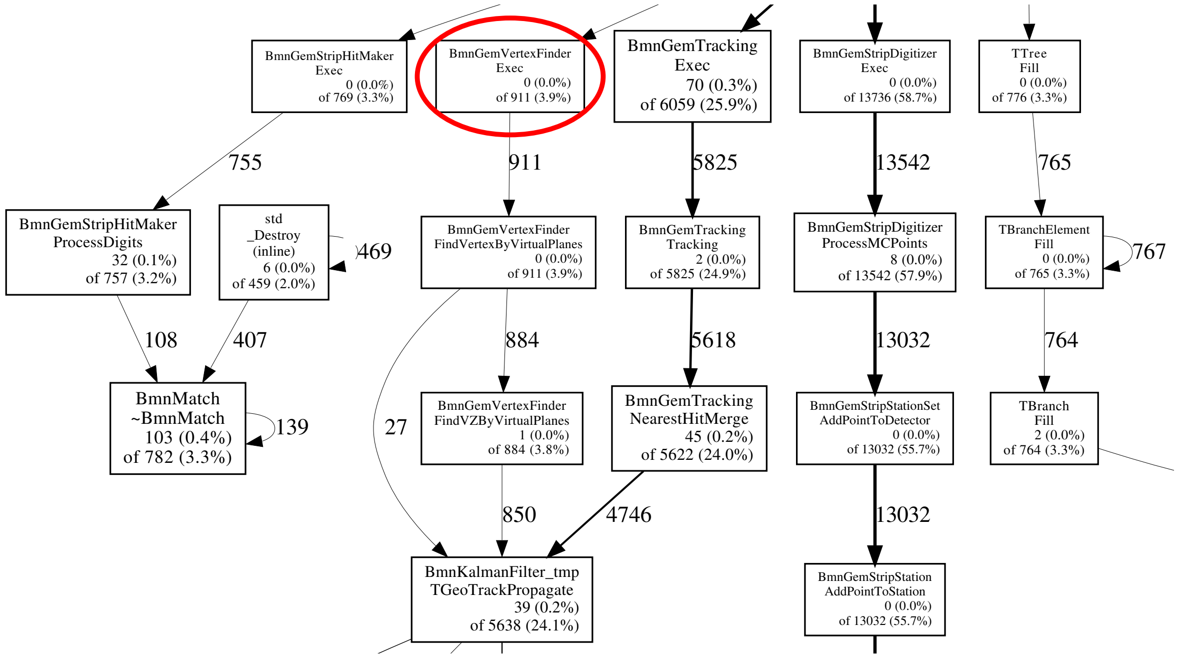
BmnGemStripHitMaker Exec 0 (0.0%) of 769 (3.3%)
BmnGemVertexFinder Exec 0 (0.0%) of 911 (3.9%)
BmnGemTracking Exec 70 (0.3%) of 6059 (25.9%)
BmnGemStripDigitizer Exec 0 (0.0%) of 13736 (58.7%)
TTree Fill 0 (0.0%) of 776 (3.3%)
755
911
5825
13542
765
BmnGemStripHitMaker ProcessDigits 32 (0.1%) of 757 (3.2%)
std _Destroy (inline) 6 (0.0%) of 459 (2.0%)
469
BmnGemVertexFinder FindVertexByVirtualPlanes 0 (0.0%) of 911 (3.9%)
BmnGemTracking Tracking 2 (0.0%) of 5825 (24.9%)
BmnGemStripDigitizer ProcessMCPoints 8 (0.0%) of 13542 (57.9%)
TBranchElement Fill 0 (0.0%) of 765 (3.3%)
767
108
407
884
5618
13032
764
BmnMatch ~BmnMatch 103 (0.4%) of 782 (3.3%)
139
27
BmnGemVertexFinder FindVZByVirtualPlanes 1 (0.0%) of 884 (3.8%)
BmnGemTracking NearestHitMerge 45 (0.2%) of 5622 (24.0%)
BmnGemStripStationSet AddPointToDetector 0 (0.0%) of 13032 (55.7%)
TBranch Fill 2 (0.0%) of 764 (3.3%)
850
4746
13032
BmnKalmanFilter_tmp TGeoTrackPropagate 39 (0.2%) of 5638 (24.1%)
BmnGemStripStation AddPointToStation 0 (0.0%) of 13032 (55.7%)

Figure 9: GPT output for old algorithm

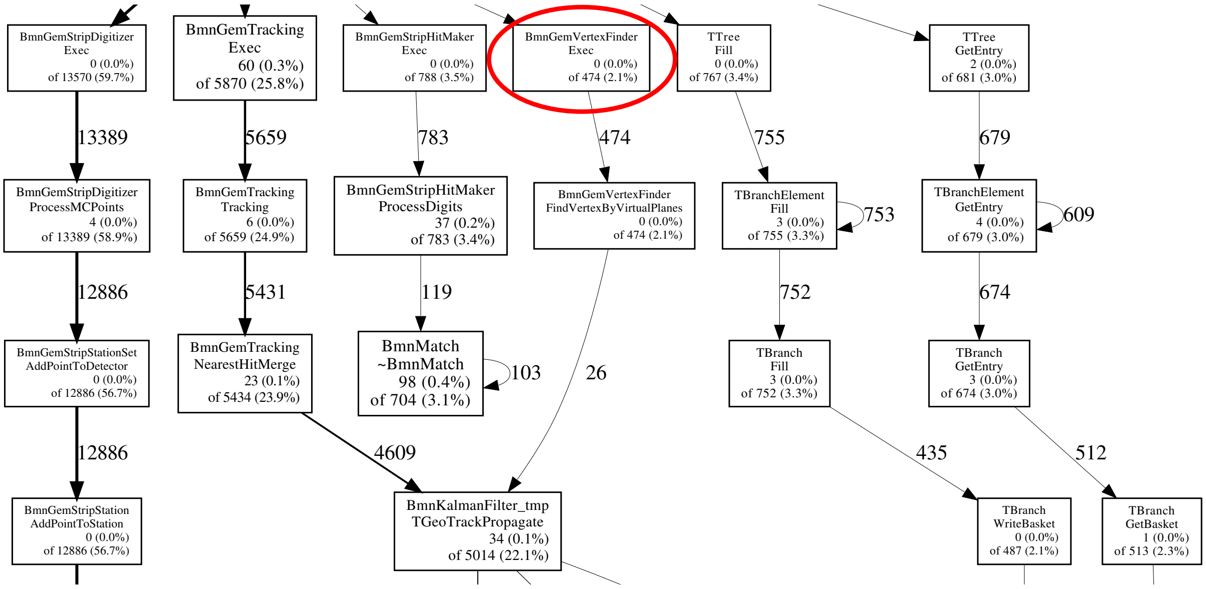
BmnGemStripDigitizer Exec 0 (0.0%) of 13570 (59.7%)
BmnGemTracking Exec 60 (0.3%) of 5870 (25.8%)
BmnGemStripHitMaker Exec 0 (0.0%) of 788 (3.5%)
BmnGemVertexFinder Exec 0 (0.0%) of 474 (2.1%)
TTree Fill 0 (0.0%) of 767 (3.4%)
TTree GetEntry 2 (0.0%) of 681 (3.0%)
13389
5659
783
474
755
679
BmnGemStripDigitizer ProcessMCPoints 4 (0.0%) of 13389 (58.9%)
BmnGemTracking Tracking 6 (0.0%) of 5659 (24.9%)
BmnGemStripHitMaker ProcessDigits 37 (0.2%) of 783 (3.4%)
BmnGemVertexFinder FindVertexByVirtualPlanes 0 (0.0%) of 474 (2.1%)
TBranchElement Fill 3 (0.0%) of 755 (3.3%)
753
TBranchElement GetEntry 4 (0.0%) of 679 (3.0%)
609
12886
5431
119
752
674
BmnGemStripStationSet AddPointToDetector 0 (0.0%) of 12886 (56.7%)
BmnGemTracking NearestHitMerge 23 (0.1%) of 5434 (23.9%)
BmnMatch ~BmnMatch 98 (0.4%) of 704 (3.1%)
103
26
TBranch Fill 3 (0.0%) of 752 (3.3%)
TBranch GetEntry 3 (0.0%) of 674 (3.0%)
12886
4609
435
512
BmnGemStripStation AddPointToStation 0 (0.0%) of 12886 (56.7%)
BmnKalmanFilter_tmp TGeoTrackPropagate 34 (0.1%) of 5014 (22.1%)
TBranch WriteBasket 0 (0.0%) of 487 (2.1%)
TBranch GetBasket 1 (0.0%) of 513 (2.3%)

Figure 10: GPT output for new algorithm

2.2 Primary Vertex location accuracy

To solve the second problem, was made an analysis. In itself, the final distribution is not Gaussian. Thus, it is necessary to define the boundaries of the approximation to the most accurate definition of primary vertex from the final distribution. The width of this distribution of about $20cm$. Firstly we analyzed which part of primary and secondary particles tracks is in range around primary vertex. To do this we need to match points from reconstructed track with points from MC track. After that we can determine does it primary particle track or not.

In the left part of Figure 11 we can see distribution in percents of tracks from primary particles, which outside the range. In the right part there is the same distribution but for tracks from secondary particles. Obviously for further analysis point $1cm$ is the most promising.

From Figure 12 we can get function of fit parameters and range around primary vertex. Also can more accurately assess the limits under which Gauss fit gives us the optimal result of primary vertex location.

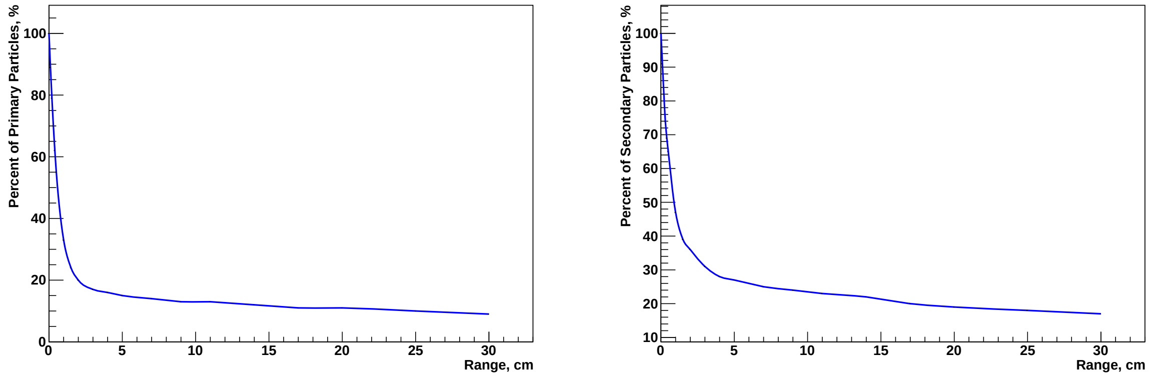

Figure 11: left: Distribution in percents of tracks from primary particles. right: Distribution in percents of tracks from secondary particles.

The most appropriate seems to be the point $0.2cm$, but from Figure 11 obviously, that out of range $0.2cm$ there are more than 80 percents of primary particles tracks.

As a result, we have optimal range $0.8cm$ where primary vertex located in $-21.87cm$ with an accuracy $0.4cm$

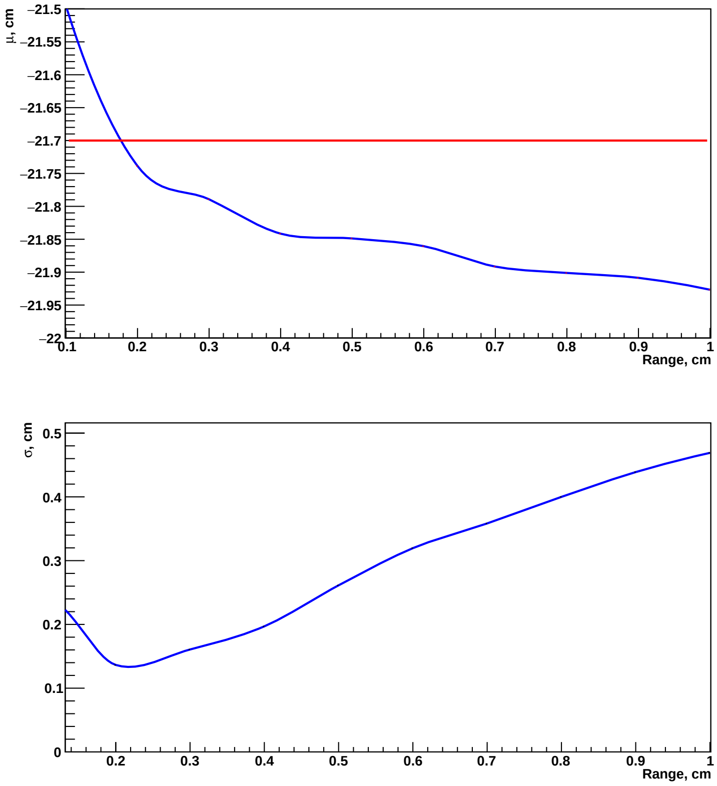

Figure 12: Gauss fit parameters of Primary Vertex distribution

In the Figure 13 we can see Primary Vertex location distribution per event for 10000 events of QGSM simulation with IP coordinate $-21.7cm$. Gauss fitting of this distribution gives us a mean in $-21.81cm$ with an accuracy $0.35cm$

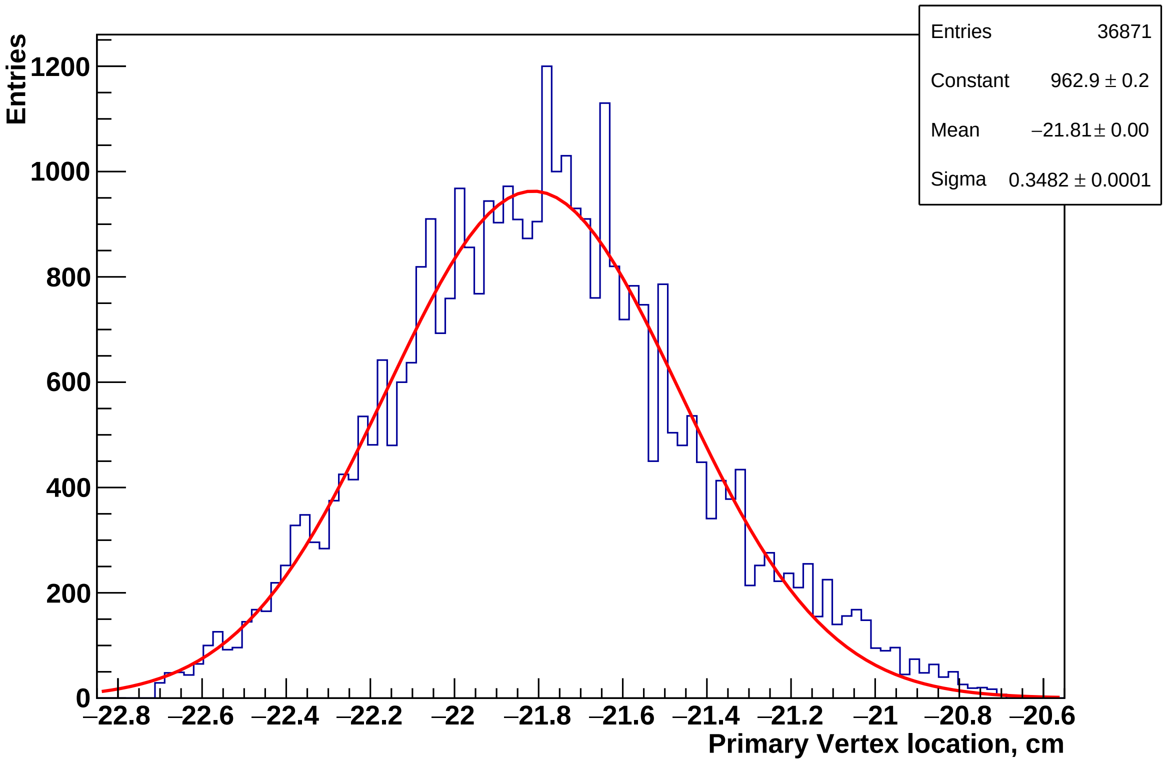

Figure 13: Distribution of Primary Vertex location

4 Magnetic field map uploading optimization

My mission was to improve the uploading of MFM(Magnetic Field Map) for tasks associated with the reconstruction of events and other similar tasks. Originally MFM loaded from ASCII file, which constructed by some rules:

1. First line is "name" of the map. This name also translated to the field type.
2. Next three lines are X, Y and Z limits and number of points.
3. The rest of the lines are B_x, B_y and B_z projection of the magnetic field on the X, Y and Z axis.

Was realized a macro to transform this MFM from ASCII file to ROOT tree. Why is it optimal?

In the “Input/Output” chapter, we saw how objects can be saved in ROOT files. In case you want to store large quantities of same-class objects, ROOT has designed the TTree and TNtuple classes specifically for that purpose. The TTree class is optimized to reduce disk space and enhance access speed. A TNtuple is a TTree that is limited to only hold floating-point numbers; a TTree on the other hand can hold all kind of data, such as objects or arrays in addition to all the simple types.

When using a TTree, we fill its branch buffers with leaf data and the buffers are written to disk when it is full. Branches, buffers, and leafs, are explained a little later in this chapter, but for now, it is important to realize that each object is not written individually, but rather collected and written a bunch at a time.

This is where the TTree takes advantage of compression and will produce a much smaller file than if the objects were written individually. Since the unit to be compressed is a buffer, and the TTree contains many same-class objects, the header of the objects can be compressed.

The TTree reduces the header of each object, but it still contains the class name. Using compression, the class name of each same-class object has a good chance of being compressed, since the compression algorithm recognizes the bit pattern representing the class name. Using a TTree and compression the header is reduced to about 4 bytes compared to the original 60 bytes. However, if compression is turned off, you will not see these large savings.

The TTree is also used to optimize the data access. A tree uses a hierarchy of branches, and each branch can be read independently from any other branch.

In this macro, "name" was written as a separate object. For X, Y and Z limits and number of points was made a structures, which include extra point – step. Step calculated by formula

$$Step_x = \frac{X_{max} - X_{min}}{Num_{ofpoints} - 1}$$

and similar for Y and Z axis.

B_x, B_y and B_z projection of the magnetic field on the X, Y and Z axis was written as a vectors of size 3.

As a result we have improved of MFM uploading time by 3 times.

5 Enhancement of tracks finding in GEM detector

1 About GEM detector

The BM@N central tracking system is comprised of GEM chambers with two-coordinate micro-trip readout. Each GEM chamber registers the ionization trail left by charged particle in the sensitive gas volume. The overall structure of the detector is a set of separated GEM stations that are located along the beam-axis at a certain distance from each other. Full GEM tracker configuration (Figure 11) includes 12 such stations, each of these has a certain type of the GEM chamber. GEM plane is divided into areas. The areas called "hot zones" are generally placed at the center of the station. Thus we have parts with strip readout electrically independent from each other in one chamber. It allows us to process high multiplicity events.

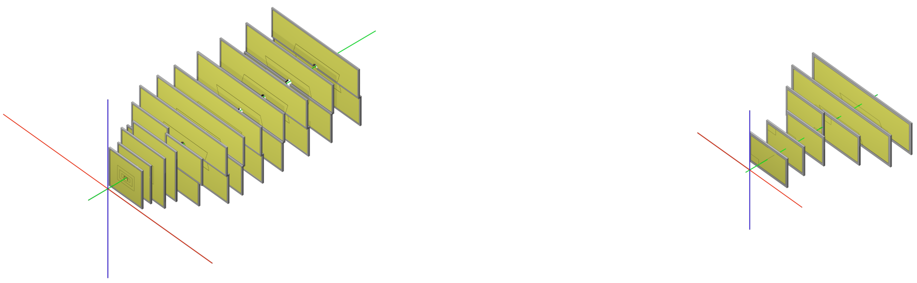

Figure 14: left: GEM tracker full configuration. right: GEM tracker test run configuration(2016-2017)

2 Algorithm description

The previous version of tracking had two parts: Seed finder and Track finder. In the first part realized an algorithm for selection of 3 hits in GEM detector. Those hits was written in a ROOT tree for further analysis. In Track finder part was used the Kalman filter. It took 3 seeds and propagate for next points to find track. If track was not found, this massive of seeds thrown away.

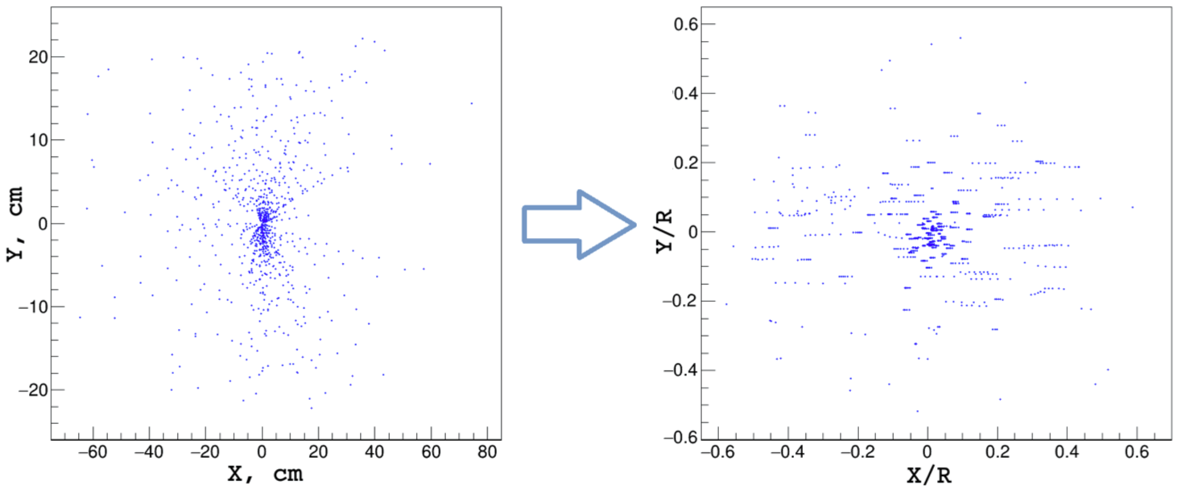

Figure 15: (x, y) to ($\frac{x}{R}$, $\frac{y}{R}$) transformation

From Figure 15 obviously, that there is no way to search track candidates in XY projection. The solution is to go to a coordinate space, where the hits are grouped in a more compact way.

$$\{x, y\} \to \left\{\frac{x}{R}, \frac{y}{R}\right\}, \quad R = \sqrt{x^2 + y^2 + z^2}.$$

For finding tracks candidates, space of normalized coordinates splitted by 1000 corridors in Y-axis. In every corridor searched all combinations between points to find 3 seed track candidate by some cuts.

3 New algorithm

Main idea was to make single macro with dynamic array of seed to realize an iteration track finder method. If triple of seeds gives us a track, we mark it as used. In the next iteration we change cut parameters and try to find tracks again from that massive of seeds. Iterations stop when there is no track has been found.

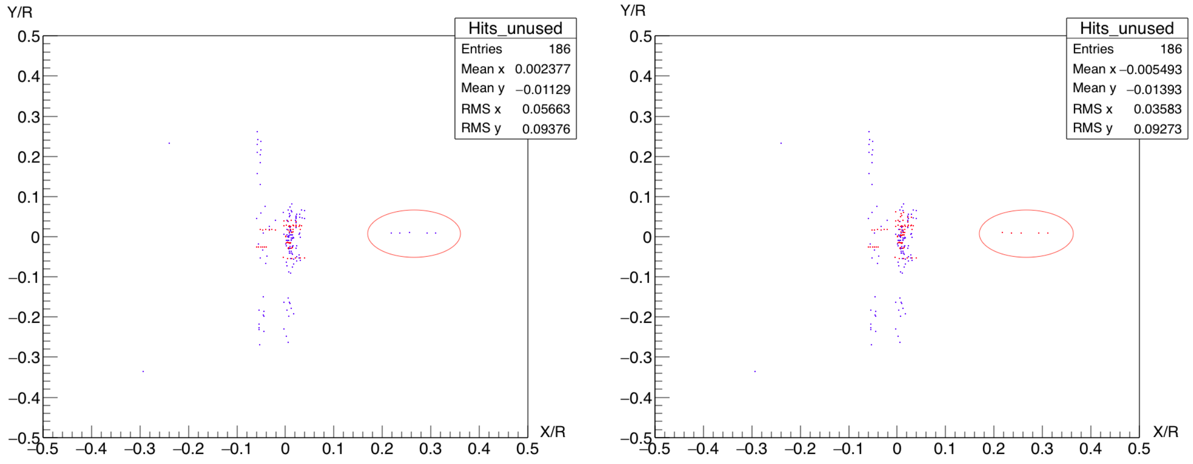

Figure 16: Visualisation of iteration method

In Figure 16 we can see how this method works. Dynamic changing of parameters gives an opportunity to find tracks, which can't be found with zero step cuts. Blue points are all seeds, prepared for tracking and red points – seeds, which used in tracks reconstruction.

Realization of this method gives an improved for efficiency of track finding.

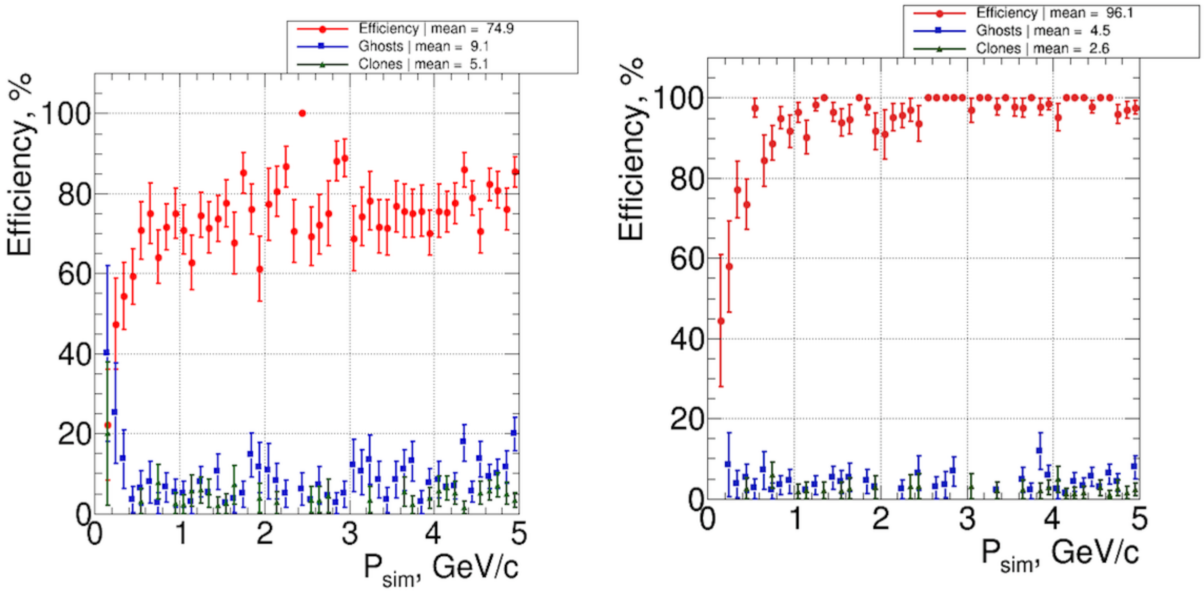

Figure 17: Efficiency improvement

How the efficiency of the track finding was calculated

$$Eff = \frac{N_{good} - N_{clones}}{N_{gen}} \cdot 100\%$$

$$Ghost = \frac{N_{bad}}{N_{rec}} \cdot 100\%$$

$$Clones = \frac{N_{clones}}{N_{rec}} \cdot 100\%$$

N_{gen} is a number of reconstructable tracks.
N_{rec} is a number of reconstructed tracks.
N_{good} is a number of good tracks.
$N_{bad} = N_{rec} - N_{good}$
N_{clones} is a number of clone tracks.

Some definitions

1. "Reconstructable" tracks are Monte Carlo tracks having at least 4 points in GEM.

2. "Good" tracks are tracks having at least 4 hits and 60% of them correspond the same Monte Carlo track.

3. "Clones" are two or more "good" tracks which correspond the same Monte Carlo track.

4. The number of "Clones" is substracted from the number of "good" tracks before calculation of efficiency is performed.

6 Conclusion

To sum up all of discussed issues, a few inferences might be done. The efficiency of Primary Vertex finding depends entirely on the quality of founded Tracks. So, for better Vertex finding need to to improve not only the efficiency but also the momentum resolution of reconstructed tracks.

It is also important to notice, that all of the results were implemented in BmnRoot (framework for simulation, reconstruction and analysing of data) software system[5]. Moreover, three of algorithms will be included to the systems as a tasks. Thus, they will provide a new opportunities for research at BM@N.

7 Acknowledgements

The author is very much obliged grateful to S.P.Merts for the support and invaluable assistance throughout the period of stay in JINR, for his patience, motivation, and immense knowledge. Also I would like to thank P.N.Batyuk for the help and for giving the opportunity to work on cluster. I am very grateful to O.V.Rogachevskiy for the strategical guidance of work, too. The author is also obliged to JINR for financial support.

8 References

1. BM@N Conceptual Design Report (BM@N collaboration)

2. R. Fruhwirth, "Application of Kalman filtering to track and vertex fitting", Nucl. Instrum. Meth. **A 262**, 444 (1987)

3. GPT https://github.com/gperftools/gperftools

4. BM@N colloboration ”BmnRoot start guide” , Dubna, 2016

5. BnmRoot framework http://mpd.jinr.ru/